



UNIÓN INTERNACIONAL DE TELECOMUNICACIONES

UIT-T

SECTOR DE NORMALIZACIÓN
DE LAS TELECOMUNICACIONES
DE LA UIT

T.172

(02/98)

SERIE T: TERMINALES PARA SERVICIOS DE
TELEMÁTICA

**MHEG-5 – Soporte para aplicaciones
interactivas de nivel básico**

Recomendación UIT-T T.172

(Anteriormente Recomendación del CCITT)

RECOMENDACIONES DE LA SERIE T DEL UIT-T
TERMINALES PARA SERVICIOS DE TELEMÁTICA



Para más información, véase la Lista de Recomendaciones del UIT-T.

RECOMENDACIÓN UIT-T.172

MHEG-5 – SOPORTE PARA APLICACIONES INTERACTIVAS DE NIVEL BÁSICO

Resumen

Esta Recomendación especifica la semántica y la sintaxis de intercambio en forma final para objetos MHEG-5, de acuerdo con los conceptos definidos en la Recomendación T.171. Estos objetos están destinados a ser utilizados en el dominio de simples aplicaciones multimedios interactivas cliente/servidor, por ejemplo, aplicaciones vídeo (cercanas) a petición, navegación y aplicaciones de examen rápido.

Orígenes

La Recomendación UIT-T T.172 ha sido preparada por la Comisión de Estudio 16 (1997-2000) del UIT-T y fue aprobada por el procedimiento de la Resolución N.º 1 de la CMNT el 6 de febrero de 1998.

La presente Recomendación está armonizada técnicamente con la Norma Internacional ISO/CEI 13522-5.

PREFACIO

La UIT (Unión Internacional de Telecomunicaciones) es el organismo especializado de las Naciones Unidas en el campo de las telecomunicaciones. El UIT-T (Sector de Normalización de las Telecomunicaciones de la UIT) es un órgano permanente de la UIT. Este órgano estudia los aspectos técnicos, de explotación y tarifarios y publica Recomendaciones sobre los mismos, con miras a la normalización de las telecomunicaciones en el plano mundial.

La Conferencia Mundial de Normalización de las Telecomunicaciones (CMNT), que se celebra cada cuatro años, establece los temas que han de estudiar las Comisiones de Estudio del UIT-T, que a su vez producen Recomendaciones sobre dichos temas.

La aprobación de Recomendaciones por los Miembros del UIT-T es el objeto del procedimiento establecido en la Resolución N.º 1 de la CMNT.

En ciertos sectores de la tecnología de la información que corresponden a la esfera de competencia del UIT-T, se preparan las normas necesarias en colaboración con la ISO y la CEI.

NOTA

En esta Recomendación, la expresión "Administración" se utiliza para designar, en forma abreviada, tanto una administración de telecomunicaciones como una empresa de explotación reconocida de telecomunicaciones.

PROPIEDAD INTELECTUAL

La UIT señala a la atención la posibilidad de que la utilización o aplicación de la presente Recomendación suponga el empleo de un derecho de propiedad intelectual reivindicado. La UIT no adopta ninguna posición en cuanto a la demostración, validez o aplicabilidad de los derechos de propiedad intelectual reivindicados, ya sea por los miembros de la UIT o por terceros ajenos al proceso de elaboración de Recomendaciones.

En la fecha de aprobación de la presente Recomendación, la UIT no ha recibido notificación de propiedad intelectual, protegida por patente, que puede ser necesaria para aplicar esta Recomendación. Sin embargo, debe señalarse a los usuarios que puede que esta información no se encuentre totalmente actualizada al respecto, por lo que se les insta encarecidamente a consultar la base de datos sobre patentes de la TSB.

© UIT 1998

Es propiedad. Ninguna parte de esta publicación puede reproducirse o utilizarse, de ninguna forma o por ningún medio, sea éste electrónico o mecánico, de fotocopia o de microfilm, sin previa autorización escrita por parte de la UIT.

ÍNDICE

Página

1	Alcance	1
1.1	Especificidad del alcance	1
1.2	Aspectos fuera del alcance de la presente Recomendación	1
2	Referencias normativas	1
3	Términos y definiciones	2
4	Conformidad	3
4.1	Conformidad de objetos MHEG-5	4
4.2	Conformidad de motores MHEG-5	4
4.2.1	Conformidad con la aceptación de un conjunto de Clases y Acciones Elementales	4
4.2.2	Conformidad con un conjunto de funcionalidades de motor	5
4.2.3	Requisitos adicionales para la especificación de conformidad	5
5	Visión general de las clases MHEG-5	8
5.1	Raíz	9
5.2	Grupo	9
5.3	Aplicación	9
5.4	Escena	10
5.5	Ingrediente	10
5.6	Enlace	10
5.7	Acción	11
5.8	Programa	11
5.9	Paleta, Tipo de carácter y Forma de cursor	11
5.10	Variable	12
5.11	Presentable	12
5.12	Grupo de testigos	12
5.13	Grupo de listas	13
5.14	Tren	13
5.15	Audio	13
5.16	Interactuable	13
5.17	Visible	13
6	Estructura de la presente Recomendación	15
7	Notaciones	15

7.1	Atributos	15
7.1.1	Atributos heredados	15
7.1.2	Atributos intercambiados propios.....	16
7.1.3	Atributos internos propios	16
7.2	Eventos.....	16
7.3	Comportamientos internos	16
7.4	Efecto de las acciones MHEG-5	16
7.5	Descripción formal.....	17
8	Clase Raíz	17
8.1	Atributos	17
8.1.1	Atributos heredados	17
8.1.2	Atributos intercambiados propios.....	17
8.1.3	Atributos MHEG-5 internos propios	17
8.2	Eventos.....	18
8.3	Comportamientos internos	19
8.4	Efecto de las acciones MHEG-5	20
8.5	Descripción formal.....	21
9	Clase de grupo.....	21
9.1	Atributos	21
9.1.1	Atributos heredados	21
9.1.2	Atributos intercambiados propios.....	22
9.1.3	Atributos internos propios	23
9.2	Eventos.....	23
9.3	Comportamientos internos	23
9.4	Efecto de las acciones MHEG-5	24
9.5	Descripción formal.....	24
10	Clase Aplicación	25
10.1	Atributos	25
10.1.1	Atributos heredados	25
10.1.2	Atributos heredados propios	25
10.1.3	Atributos internos propios	28
10.2	Eventos.....	29
10.3	Comportamientos internos	29
10.4	Efecto de las acciones MHEG-5	29
11	Clase Escena	36

	Página
11.1 Atributos	36
11.1.1 Atributos heredados	37
11.1.2 Atributos intercambiados propios.....	37
11.1.3 Atributos internos propios	38
11.2 Eventos.....	38
11.3 Comportamientos internos	39
11.4 Efecto de las acciones MHEG-5	39
11.5 Descripción formal.....	43
12 Clase Ingrediente.....	44
12.1 Atributos	44
12.1.1 Atributos heredados	44
12.1.2 Atributos intercambiados propios.....	44
12.1.3 Atributos internos propios	46
12.2 Eventos.....	46
12.3 Comportamiento internos	46
12.4 Efecto de las acciones MHEG-5	47
12.5 Descripción formal.....	49
13 Clase de enlace.....	50
13.1 Atributos	50
13.1.1 Atributos heredados	50
13.1.2 Atributos intercambiados propios.....	50
13.1.3 Atributos internos propios	50
13.2 Eventos.....	50
13.3 Comportamientos internos	51
13.4 Efectos de las acciones MHEG-5	51
13.5 Descripción formal.....	52
14 Clase Programa	52
14.1 Atributos	52
14.1.1 Atributos heredados	52
14.1.2 Atributos intercambiados propios.....	53
14.1.3 Atributos internos propios	53
14.2 Eventos.....	53
14.3 Comportamientos internos	53
14.4 Efectos de las acciones MHEG-5	54
14.5 Descripción formal.....	56

15	Clase Programa Residente	56
15.1	Atributos	56
15.1.1	Atributos heredados	56
15.1.2	Atributos intercambiados propios.....	57
15.1.3	Atributos internos propios	57
15.2	Eventos.....	57
15.3	Comportamientos internos	57
15.4	Efectos de las acciones MHEG-5	57
15.5	Descripción formal.....	57
16	Clase Programa Distante.....	57
16.1	Atributos	57
16.1.1	Atributos heredados	57
16.1.2	Atributos intercambiados propios.....	58
16.1.3	Atributos internos propios	58
16.2	Eventos.....	58
16.3	Comportamientos internos	58
16.4	Efectos de las acciones MHEG-5	58
16.5	Descripción formal.....	58
17	Clase Programa Intercambiado	59
17.1	Atributos	59
17.1.1	Atributos heredados	59
17.1.2	Atributos intercambiados propios.....	59
17.1.3	Atributos internos propios	59
17.2	Eventos.....	59
17.3	Comportamientos internos	59
17.4	Efectos de las acciones MHEG-5	59
17.5	Descripción formal.....	59
18	Clase Paleta.....	60
18.1	Atributos	60
18.1.1	Atributos heredados	60
18.1.2	Atributos intercambiados propios.....	60
18.1.3	Atributos internos propios	60
18.2	Eventos.....	60
18.3	Comportamientos internos	60

	Página
18.4 Efectos de las acciones MHEG-5	60
18.5 Descripción formal.....	60
19 Clase Tipo de Carácter.....	60
19.1 Atributos	61
19.1.1 Atributos heredados	61
19.1.2 Atributos intercambiados propios.....	61
19.1.3 Atributos internos propios	61
19.2 Eventos.....	61
19.3 Comportamientos internos.....	61
19.4 Efectos de las acciones MHEG-5	61
19.5 Descripción formal.....	61
20 Clase Forma de cursor	61
20.1 Atributos	62
20.1.1 Atributos heredados	62
20.1.2 Atributos intercambiados propios.....	62
20.1.3 Atributos internos propios	62
20.2 Eventos.....	62
20.3 Comportamientos internos.....	62
20.4 Efectos de las acciones MHEG-5	62
20.5 Descripción formal.....	62
21 Clase Variable	62
21.1 Atributos	62
21.1.1 Atributos heredados	63
21.1.2 Atributos intercambiados propios.....	63
21.1.3 Atributos internos propios	63
21.2 Eventos.....	63
21.3 Comportamientos internos.....	63
21.4 Efectos de las acciones MHEG-5	64
21.5 Descripción formal.....	65
22 Clase Variable booleana	65
22.1 Atributos	65
22.1.1 Atributos heredados	65
22.1.2 Atributos intercambiados propios.....	66
22.1.3 Atributos internos propios	66
22.2 Eventos.....	66

22.3	Comportamientos internos	66
22.4	Efectos de las acciones MHEG-5	66
22.5	Descripción formal.....	66
23	Clase Variable entera	66
23.1	Atributos	66
23.1.1	Atributos heredados	66
23.1.2	Atributos intercambiados propios.....	66
23.1.3	Atributos internos propios	66
23.2	Eventos.....	67
23.3	Comportamientos internos	67
23.4	Efectos de las acciones MHEG-5	67
23.5	Descripción formal.....	69
24	Clase Variable cadena de octetos.....	69
24.1	Atributos	69
24.1.1	Atributos heredados	69
24.1.2	Atributos intercambiados propios.....	69
24.1.3	Atributos internos propios	69
24.2	Eventos.....	69
24.3	Comportamientos internos	70
24.4	Efectos de las acciones MHEG-5	70
24.5	Descripción formal.....	70
25	Clase Variable referencia de objeto	70
25.1	Atributos	71
25.1.1	Atributos heredados	71
25.1.2	Atributos intercambiados propios.....	71
25.1.3	Atributos internos propios	71
25.2	Eventos.....	71
25.3	Comportamientos internos	71
25.4	Efectos de las acciones MHEG-5	71
25.5	Descripción formal.....	71
26	Clase Variable referencia de contenido	71
26.1	Atributos	71
26.1.1	Atributos heredados	71
26.1.2	Atributos intercambiados propios.....	72
26.1.3	Atributos internos propios	72

	Página
26.2 Eventos.....	72
26.3 Comportamientos internos.....	72
26.4 Efectos de las acciones MHEG-5	72
26.5 Descripción formal.....	72
27 Clase Presentable	72
27.1 Atributos	72
27.1.1 Atributos heredados	72
27.1.2 Atributos intercambiados propios.....	72
27.1.3 Atributos internos propios	72
27.2 Eventos.....	73
27.3 Comportamientos internos.....	73
27.4 Efectos de las acciones MHEG-5	73
27.5 Descripción formal.....	73
28 Clase Gestor de testigos	74
28.1 Atributos	74
28.1.1 Atributos heredados	74
28.1.2 Atributos intercambiados propios.....	74
28.1.3 Atributos internos propios	75
28.2 Eventos.....	75
28.3 Comportamientos internos.....	75
28.4 Efectos de las acciones MHEG-5	76
28.5 Descripción formal.....	77
29 Clase Grupo de testigos	77
29.1 Atributos	77
29.1.1 Atributos heredados	78
29.1.2 Atributos intercambiados propios.....	78
29.1.3 Atributos internos propios	78
29.2 Eventos.....	78
29.3 Comportamientos internos.....	78
29.4 Efectos de las acciones MHEG-5	79
29.5 Descripción formal.....	79
30 Clase Grupo de listas	80
30.1 Atributos	80
30.1.1 Atributos heredados	80
30.1.2 Atributos intercambiados propios.....	80

	Página
30.1.3 Atributos internos propios	81
30.2 Eventos.....	83
30.3 Comportamientos internos.....	84
30.4 Efectos de las acciones MHEG-5	86
30.5 Descripción formal.....	91
31 Clase Visible	91
31.1 Atributos	91
31.1.1 Atributos heredados	91
31.1.2 Atributos intercambiados propios.....	91
31.1.3 Atributos internos propios	92
31.2 Eventos.....	93
31.3 Comportamientos internos.....	93
31.4 Efectos de las acciones MHEG-5	94
31.5 Descripción formal.....	99
32 Clase Diagrama de bits	99
32.1 Atributos	99
32.1.1 Atributos heredados	99
32.1.2 Atributos intercambiados propios.....	99
32.1.3 Atributos internos propios	100
32.2 Eventos.....	100
32.3 Comportamientos internos.....	100
32.4 Efectos de las acciones MHEG-5	100
32.5 Descripción formal.....	101
33 Clase Arte lineal.....	101
33.1 Atributos	101
33.1.1 Atributos heredados	101
33.1.2 Atributos intercambiados propios.....	101
33.1.3 Atributos internos propios	103
33.2 Eventos.....	103
33.3 Comportamientos internos.....	104
33.4 Efectos de las acciones MHEG-5	104
33.5 Descripción formal.....	106
34 Clase Rectángulo	106
34.1 Atributos	106
34.1.1 Atributos heredados	107

	Página
34.1.2 Atributos intercambiados propios.....	107
34.1.3 Atributos internos propios	107
34.2 Eventos.....	107
34.3 Comportamientos internos.....	107
34.4 Efectos de las acciones MHEG-5	107
34.5 Descripción formal.....	107
35 Clase Arte lineal dinámica	107
35.1 Atributos	108
35.1.1 Atributos heredados	108
35.1.2 Atributos intercambiados propios.....	108
35.1.3 Atributos internos propios	108
35.2 Eventos.....	108
35.3 Comportamientos internos.....	108
35.4 Efectos de las acciones MHEG-5	108
35.5 Descripción formal.....	115
36 Clase Texto	115
36.1 Atributos	115
36.1.1 Atributos heredados	116
36.1.2 Atributos intercambiados propios.....	116
36.1.3 Atributos internos propios	119
36.2 Eventos.....	119
36.3 Comportamientos internos.....	119
36.4 Efectos de las acciones MHEG-5	119
36.5 Descripción formal.....	121
37 Clase Tren	121
37.1 Atributos	122
37.1.1 Atributos heredados	122
37.1.2 Atributos intercambiados propios.....	122
37.1.3 Atributos internos propios	123
37.2 Eventos.....	124
37.3 Comportamientos internos.....	125
37.4 Efectos de las acciones MHEG-5	126
37.5 Descripción formal.....	128
38 Clase Audio.....	129

	Página
38.1 Atributos	129
38.1.1 Atributos heredados	129
38.1.2 Atributos intercambiados propios.....	129
38.1.3 Atributos internos propios	129
38.2 Eventos.....	130
38.3 Comportamientos internos	130
38.4 Efectos de las acciones MHEG-5	130
38.5 Descripción formal.....	131
39 Clase Vídeo.....	131
39.1 Atributos	131
39.1.1 Atributos heredados	131
39.1.2 Atributos intercambiados propios.....	131
39.1.3 Atributos internos propios	132
39.2 Eventos.....	132
39.3 Comportamientos internos	132
39.4 Efectos de las acciones MHEG-5	132
39.5 Descripción formal.....	132
40 Clase Gráficos RT.....	133
40.1 Atributos	133
40.1.1 Atributos heredados	133
40.1.2 Atributos intercambiados propios.....	133
40.1.3 Atributos internos propios	133
40.2 Eventos.....	133
40.3 Comportamientos internos	133
40.4 Efectos de las acciones MHEG-5	134
40.5 Descripción formal.....	134
41 Clase Interactuable.....	134
41.1 Atributos	134
41.1.1 Atributos heredados	134
41.1.2 Atributos intercambiados propios.....	134
41.1.3 Atributos internos propios	135
41.2 Eventos.....	136
41.3 Comportamientos internos	136
41.4 Efectos de las acciones MHEG-5	137
41.5 Descripción formal.....	138

	Página
42 Clase Deslizador	139
42.1 Atributos	139
42.1.1 Atributos heredados	139
42.1.2 Atributos intercambiados propios.....	139
42.1.3 Atributos internos propios	141
42.2 Eventos.....	142
42.3 Comportamientos internos	142
42.4 Efectos de las acciones MHEG-5	142
42.5 Descripción formal.....	145
43 Clase Campo de entrada.....	145
43.1 Atributos	145
43.1.1 Atributos heredados	145
43.1.2 Atributos intercambiados propios.....	145
43.1.3 Atributos internos propios	146
43.2 Eventos.....	147
43.3 Comportamientos internos	147
43.4 Efectos de las acciones MHEG-5	148
43.5 Descripción formal.....	149
44 Clase Hipertexto	150
44.1 Atributos	150
44.1.1 Atributos heredados	150
44.1.2 Atributos intercambiados propios.....	150
44.1.3 Atributos internos propios	150
44.2 Eventos.....	150
44.3 Comportamientos internos	150
44.4 Efecto de las acciones MHEG-5	151
44.5 Descripción formal.....	151
45 Clase Botón.....	151
45.1 Atributos	151
45.1.1 Atributos heredados	152
45.1.2 Atributos intercambiados propios.....	152
45.1.3 Atributos internos propios	152
45.2 Eventos.....	152
45.3 Comportamientos internos	153
45.4 Efecto de las acciones MHEG-5	153

	Página
45.5 Descripción formal.....	154
46 Clase Punto focal	154
46.1 Atributos	154
46.1.1 Atributos heredados	154
46.1.2 Atributos intercambiados propios.....	154
46.1.3 Atributos internos propios	154
46.2 Eventos.....	155
46.3 Comportamientos internos	155
46.4 Efecto de las acciones MHEG-5	155
46.5 Descripción formal.....	155
47 Clase Botón de pulsación.....	155
47.1 Atributos	155
47.1.1 Atributos heredados	155
47.1.2 Atributos intercambiados propios.....	155
47.1.3 Atributos internos propios	156
47.2 Eventos.....	156
47.3 Comportamientos internos	156
47.4 Efecto de las acciones MHEG-5	156
47.5 Descripción formal.....	157
48 Clase Botón de conmutación	157
48.1 Atributos	157
48.1.1 Atributos heredados	158
48.1.2 Atributos intercambiados propios.....	158
48.1.3 Atributos internos propios	158
48.2 Eventos.....	158
48.3 Comportamientos internos	158
48.4 Efectos de las acciones MHEG-5	158
48.5 Descripción formal.....	159
49 Clase Acción	159
49.1 Atributos	160
49.1.1 Atributos heredados	160
49.1.2 Atributos intercambiados propios.....	160
49.2 Atributos internos propios	160
49.3 Descripción formal.....	160

	Página
50 Objetos de referencia, Contenido, Valores, Color y Posición XY	161
50.1 Referencia de objeto	161
50.2 Referencia de contenido.....	161
50.3 Referencia de objeto genérica	161
50.4 Referencia de contenido genérica	161
50.5 Entero genérico	162
50.6 Booleano genérico	162
50.7 Cadena de octetos genérica	162
50.8 Color	162
50.9 Posición XY	162
50.10 Resolución de valores genéricos	163
51 Objetos MHEG-5 de referencia	163
52 Espacios de nombres, Llamadas a Programas Distantes y Conexiones	164
53 Tratamiento de eventos	165
53.1 Tipos de eventos	165
53.2 Eventos síncronos y eventos asíncronos	166
53.3 Tratamiento de eventos y enlaces	166
53.4 Entrada de usuario.....	167
53.5 Interacción de usuario	167
53.6 Eventos de cursor.....	167
53.7 Tratamiento de errores	168
54 Presentación de objetos visibles	168
54.1 Sistema de coordenadas	168
54.2 Casilla delimitante	168
54.3 Pila de visualización	169
54.4 Objetos transparentes	169
54.5 Formato de pixel	170
Anexo A – Notación ASN.1	171
Anexo B – Notación textual para las aplicaciones MHEG-5.....	195
B.1 Definiciones generales	195
B.1.1 Código	195
B.1.2 Delimitador (Delimiter)	196
B.1.3 Comentario (Comment)	196
B.1.4 Rótulo (Tag)	196

B.2	Definición de símbolos	196
B.3	Símbolos terminales.....	196
B.3.1	INTEGER (ENTERO).....	196
B.3.2	BOOLEAN (BOOLEANO).....	197
B.3.3	STRING (CADENA).....	197
B.3.4	QPRINTABLE.....	197
B.3.5	BASE64	197
B.3.6	Null (Nulo)	198
B.3.7	Enumeration Values (Valores de enumeración).....	198
B.4	Definiciones de objetos MHEG-5.....	198
B.4.1	Root Class (Clase Raíz).....	198
B.4.2	Group Class (Clase Grupo).....	199
B.4.3	Application Class (Clase Aplicación).....	199
B.4.4	Scene Class (Clase Escena)	200
B.4.5	Ingredient Class (Clase Ingrediente).....	200
B.4.6	Link Class (Clase Enlace).....	200
B.4.7	Program Class (Clase Programa).....	201
B.4.8	ResidentProgram Class (Clase Programa Residente).....	201
B.4.9	RemoteProgram Class (Clase Programa Distante).....	201
B.4.10	InterchangedProgram Class (Clase Programa Intercambiado)	201
B.4.11	Palette Class (Clase Paleta)	201
B.4.12	Font Class (Clase Tipo de Carácter).....	201
B.4.13	CursorShape Class (Clase Forma de Cursor)	201
B.4.14	Variable Class (Clase Variable).....	201
B.4.15	BooleanVariable Class (Clase Variable Booleana)	202
B.4.16	IntegerVariable Class (Clase Variable Entera).....	202
B.4.17	OctetStringVariable Class (Clase Variable Cadena de Octetos)	202
B.4.18	ObjectRefVariable Class (Clase Variable Referencia de Objeto)	202
B.4.19	ContentRefVariable Class (Clase Variable Referencia de Contenido)	202
B.4.20	Presentable Class (Clase Presentable)	202
B.4.21	TokenManager Class (Clase Gestor de Testigo)	202
B.4.22	TokenGroup Class (Clase Grupo de Testigo).....	202
B.4.23	ListGroup Class (Clase Grupo de Listas)	202
B.4.24	Visible Class (Clase Visible).....	202
B.4.25	Bitmap Class (Clase Diagrama de Bits)	203
B.4.26	LineArt Class (Clase Arte Lineal)	203
B.4.27	Rectangle Class (Clase Rectángulo).....	203
B.4.28	DynamicLineArt Class (Clase Arte Lineal Dinámico).....	203

	Página
B.4.29 Text Class (Clase Texto)	203
B.4.30 Stream Class (Clase Tren)	203
B.4.31 Audio Class (Clase Audio)	204
B.4.32 Video Class (Clase Vídeo)	204
B.4.33 RTGraphics Class (Clase Gráficos RT).....	204
B.4.34 Interactable Class (Clase Interactuable)	204
B.4.35 Slider Class (Clase Deslizador)	204
B.4.36 EntryField Class (Clase Campo de Entrada)	204
B.4.37 HyperText Class (Clase Hipertexto).....	204
B.4.38 Button Class (Clase Botón)	204
B.4.39 Hotspot Class (Clase Punto Focal)	205
B.4.40 PushButton Class (Clase Botón de Pulsación)	205
B.4.41 SwitchButton Class (Clase Botón de Conmutación).....	205
B.4.42 Action Class (Clase Acción).....	205
B.4.43 Referencing Objects, Contents, Values, Colour and Position (Objetos de Referencia, Contenido, Valores, Color y Posición).....	212
Apéndice I – Propiedades de un motor MHEG-5	213
Apéndice II – Definición de dominios de aplicación	213
II.1 Formato de intercambio de objetos	213
II.2 Conjunto de clases	213
II.3 Conjunto de características	214
II.4 Codificación de datos de contenido	214
II.5 Registros de entrada de usuario	215
II.6 Restricciones semánticas impuestas a las aplicaciones MHEG-5	216
II.7 Evento motor.....	216
II.8 Obtención de soporte de motor	217
II.9 Correspondencia de protocolos e interacción externa.....	217

Recomendación T.172

MHEG-5 – SOPORTE PARA APLICACIONES INTERACTIVAS DE NIVEL BÁSICO

(Ginebra, 1998)

1 Alcance

La presente Recomendación especifica la semántica y la sintaxis de intercambio en forma final para objetos MHEG-5, de acuerdo con los conceptos definidos en la Recomendación T.171. Estos objetos están destinados a ser utilizados en el dominio de simples aplicaciones multimedios interactivas cliente/servidor, por ejemplo, aplicaciones vídeo (cercanas) a petición, navegación y aplicaciones de examen rápido.

1.1 Especificidad del alcance

Como se prevé utilizar esta Recomendación para el interfuncionamiento de aplicaciones a través de plataformas, el alcance se centra en una definición específica y precisa de clases MHEG-5. Esta Recomendación reconoce la semántica implicada por la especificación de los objetos MHEG-5 y por la interpretación de comportamientos MHEG-5 dentro del sistema usuario.

1.2 Aspectos fuera del alcance de la presente Recomendación

Se excluye toda normalización de modelos, servicios, sistemas, protocolos o aplicaciones que es probable utilicen los objetos MHEG-5.

La representación codificada de los datos de contenido no está comprendida en el alcance de la presente Recomendación.

2 Referencias normativas

Las siguientes Recomendaciones del UIT-T y otras referencias contienen disposiciones que, mediante su referencia en este texto, constituyen disposiciones de la presente Recomendación. Al efectuar esta publicación, estaban en vigor las ediciones indicadas. Todas las Recomendaciones y otras referencias son objeto de revisiones por lo que se preconiza que los usuarios de esta Recomendación investiguen la posibilidad de aplicar las ediciones más recientes de las Recomendaciones y otras referencias citadas a continuación. Se publica periódicamente una lista de las Recomendaciones UIT-T actualmente vigentes.

- Recomendación UIT-T X.680 (1994) | ISO/CEI 8824-1:1995, *Tecnología de la información – Notación de sintaxis abstracta uno: Especificación de la notación básica.*
- Recomendación UIT-T X.690 (1994) | ISO/CEI 8825-1:1995, *Tecnología de la información – Reglas de codificación de notación de sintaxis abstracta uno: Especificación de las reglas de codificación básica de las reglas de codificación canónica y de las reglas de codificación distinguida.*
- ISO/CEI 646:1991, *Information technology – ISO 7-bit coded character set for information interchange.*
- RFC 1521 (1993), *MIME (Multipurpose Internet Mail Extensions) Part One: Mechanisms for Specifying and Describing the Format of Internet Message Bodies.*

3 Términos y definiciones

En esta Recomendación se definen los términos siguientes.

3.1 clase abstracta: Clase que nunca es ejemplificada en un objeto MHEG-5 intercambiable.

NOTA – Una clase abstracta define atributos, comportamientos y semántica de acciones que pueden ser intercambiados como partes de cualquier objeto MHEG-5 de las subclases concretas de esta clase abstracta.

3.2 acción: Conjunto de acciones elementales.

3.3 activo: Estado de cualquier objeto MHEG-5 cuando el comportamiento *Activación* ha sido completado satisfactoriamente para este objeto.

Un objeto activo tiene su *Estado de ejecución (RunningStatus)* puesto a *verdadero*.

3.4 dominio de aplicación: Dominio específico de aplicaciones que utilizan esta Recomendación y proporcionan herramientas y valores adicionales para crear un caso práctico de un entorno MHEG-5.

NOTA – En el apéndice II figura más información sobre dominios de aplicación.

3.5 alcance de aplicación: Alcance común de todos los objetos MHEG-5 (escenas e ingredientes) accedidos desde una aplicación MHEG-5.

3.6 atributo: Valor denominado y tipificado asignado a una clase.

3.7 disponible: Estado de cualquier objeto MHEG-5 cuando el comportamiento *Preparación* ha sido completado satisfactoriamente para este objeto.

Un objeto disponible tiene su *Estado de disponibilidad (AvailabilityStatus)* puesto a *Verdadero*.

3.8 clase básica: Clase MHEG-5 que define algunos atributos, comportamientos y semántica de acciones que son compartidos por una clase MHEG-5 dada.

3.9 clase concreta: Clase de cualquier objeto MHEG-5 que puede ser codificada e intercambiada de acuerdo con las especificaciones proporcionadas en los anexos A o B.

3.10 acción elemental: Representación abstracta de un mensaje que puede ser enviado a un objeto.

NOTA – La presente Recomendación define la semántica de acciones elementales disponibles para cada clase MHEG-5. Obsérvese que la clase MHEG-5 Acción (con A mayúscula) tiene un significado diferente descrito en la cláusula 49.

3.11 evento: Representación abstracta de una ocurrencia de significado especial para cualquier objeto MHEG-5.

NOTA – Los eventos se utilizan para activar condiciones de Enlace y lanzar la ejecución de secuencias de acciones elementales.

3.12 atributo intercambiado: Atributo que forma parte de la representación codificada en octetos intercambiable de un objeto MHEG-5, y es transmitido con este objeto.

3.13 inactivo: Estado de cualquier objeto MHEG-5 cuando el comportamiento *Desactivación* ha sido completado satisfactoriamente o cuando no se ha aplicado satisfactoriamente ningún comportamiento *Activación* a este objeto.

Un objeto MHEG-5 inactivo tiene su *Estado de ejecución* puesto a *Falso*.

3.14 atributo heredado: Atributo que se define en una clase básica de la clase del objeto MHEG-5.

3.15 representación intercambiable: Cadena de octetos, que contiene los atributos intercambiados codificados de este objeto MHEG-5 de conformidad con la sintaxis ASN.1 y la codificación proporcionadas en el anexo A o, cuando se prefiere la representación textual, con el anexo B.

3.16 atributo interno: Estructura de datos abstractos, nunca codificados en códigos de octetos ni intercambiados, que se utiliza para definir la semántica de comportamientos o acciones internos para cualquier objeto MHEG-5.

NOTA – Cualquier motor MHEG-5 pudiera considerar que un atributo interno forma parte de la representación interna del objeto MHEG-5, aunque esto no es obligatorio. Lo que sí es obligatorio es implementar la funcionalidad descrita por estos atributos internos.

3.17 comportamiento interno: Función abstracta que define la semántica de acciones elementales MHEG-5 para cualquier clase MHEG-5.

NOTA – Un comportamiento interno de una clase es abrogado la mayor parte del tiempo por comportamiento internos de subclases de esta clase. Un motor MHEG-5 pudiera considerar que un comportamiento interno de una clase es un método privado de esta clase, aunque esto no es obligatorio.

3.18 aplicación MHEG-5: Conjunto de escenas e información de control que permite al usuario navegar entre escenas.

NOTA – La clase MHEG-5 Aplicación (con A mayúscula) tiene un significado diferente, más específico que se indica en la cláusula 10.

3.19 clase MHEG-5: Definición abstracta de atributos intercambiados e internos de partes de objetos multimedia/hipermedia intercambiables, así como definición de la semántica de comportamientos internos y el efecto de acciones MHEG-5 para estos objetos.

3.20 motor MHEG-5: Proceso o conjunto de procesos que interpretan objetos MHEG-5 codificados de acuerdo con las especificaciones de codificación definidas en los anexos A o B.

3.21 objeto MHEG-5: Un caso de cualquier clase MHEG-5.

NOTA – Un objeto MHEG-5 no es un objeto físico, sino una abstracción que puede tener muchas representaciones de tipos diferentes. Varios servicios informáticos tratan estas representaciones.

3.22 escena MHEG-5: Estructura que coordina la presentación (visual y audible) de objetos MHEG-5.

3.23 clase mezcla: Clase abstracta que no hereda de la clase Raíz.

Ejemplos: clase *Interactable*, clase *Gestor de testigo*.

3.24 no disponible: Estado de cualquier objeto MHEG-5 cuando se ha completado satisfactoriamente el comportamiento *Dstrucción* o cuando no se ha aplicado satisfactoriamente a este objeto ningún comportamiento *Preparación*.

Un objeto no disponible tiene su *Estado de disponibilidad* puesto a *FALSO*. Incluso si un objeto MHEG-5 no existe en el motor MHEG-5, su *Estado de disponibilidad* existe y está puesto a *FALSO*.

3.25 subclase: Cualquier clase MHEG-5 que comparte los mismos atributos, comportamientos y semántica de acciones que otra clase MHEG-5.

4 Conformidad

Esta cláusula especifica los requisitos de conformidad para motores MHEG-5 y para aplicaciones MHEG-5.

4.1 Conformidad de objetos MHEG-5

Cualquier objeto MHEG-5 tendrá una representación en octetos. Para fines de intercambio, la representación en octetos será conforme a la sintaxis ASN.1 y codificación definidas en el anexo A, o a la gramática de notación textual definida en el anexo B. El dominio de aplicación elegirá si ha de utilizar la representación del anexo A o la del anexo B, y esa representación será empleada exclusivamente en todo el dominio de aplicación.

Los atributos de cualquier objeto MHEG-5 cumplirán todos los requisitos definidos en las subcláusulas pertinentes de la presente Recomendación.

4.2 Conformidad de motores MHEG-5

La conformidad de los motores MHEG-5 sólo puede medirse con respecto a una definición completa del dominio de aplicación. Para especificar plenamente la conformidad, un dominio de aplicación definirá, además de la representación de intercambio, lo siguiente:

- 1) un conjunto de clases de la lista de todas las clases de la presente Recomendación, prescritas en 4.2.1;
- 2) un conjunto de características de la lista presentada en 4.2.2;
- 3) opciones concretas adicionales enumeradas en 4.2.3.

NOTA – Para un ejemplo de definición completa de un dominio de aplicación, véase el apéndice II.

4.2.1 Conformidad con la aceptación de un conjunto de Clases y Acciones Elementales

La conformidad con la aceptación de un conjunto de Clases y Acciones Elementales se define como sigue.

Todo motor MHEG-5 tiene que implementar por lo menos el siguiente conjunto mínimo de clases:

- Clase Aplicación
Se implementarán todos los atributos, eventos y comportamientos internos.
- Clase Escena
Se implementarán todos los atributos, eventos y comportamientos internos.
- Clase Enlace
Se implementarán todos los atributos, eventos y comportamientos internos.
- Clase Acción
Se implementarán todos los atributos, eventos y comportamientos internos.

Todos los dominios de aplicación definirán la conformidad con un conjunto de clases que contenga por lo menos el conjunto mínimo indicado anteriormente. Un dominio de aplicación puede especificar la conformidad con un conjunto mayor de clases y acciones elementales; en cualquier caso, el dominio de aplicación enumerará claramente las clases y acciones elementales admitidas.

Cuando se aplican clases adicionales en cualquier motor MHEG-5, el motor admitirá todos sus atributos, eventos, comportamientos internos y acciones elementales definidos en la presente Recomendación, con la posible excepción de las características facultativas enumeradas en 4.2.2. En relación con la clase Acción, el motor implementará todos los efectos de las acciones elementales MHEG-5 correspondientes al conjunto de clases especificado. Es cometido de cada dominio de aplicación elegir y definir bien el conjunto de clases que se requiere para ese dominio de aplicación específico.

Si una clase no es sustentada por un motor MHEG-5, y un objeto de esta clase es enviado al motor MHEG-5, esto origina un error que es manejado por el tratamiento de errores por defecto definido en 53.7.

4.2.2 Conformidad con un conjunto de funcionalidades de motor

La conformidad con un conjunto de funcionalidades de motor se define como sigue.

Todo motor MHEG-5 proporcionará todos los mecanismos normativos definidos en las cláusulas 51 a 54.

Todo motor MHEG-5 implementará todos los efectos de acciones MHEG-5 y los comportamientos internos de clases MHEG-5 incluidos en la definición de su dominio de aplicación, excepto las características facultativas siguientes:

- Conexiones auxiliares (correspondientes a las acciones Abrir conexión y Cerrar conexión).
- Cache (caching) (correspondiente al cache de objetos MHEG-5 y a los datos de contenido de objetos Ingredientes).
- Clonación (correspondiente a la acción Clonar definida en la clase Ingrediente).
- Cursor de movimiento libre.
- Escala de diagrama de bits y de vídeo (correspondiente a las acciones Aplicar escala de diagrama de bits y Aplicar escala de vídeo de las clases Diagrama de bits y Vídeo).
- Pila de aplicaciones (correspondiente a la acción Ramificación permanente de la clase Aplicación).
- Modo truco (trick mode) (correspondiente a la acción Fijar velocidad de la clase Tren).

Un dominio de aplicación definirá claramente una lista de las anteriores características que son obligatorias o facultativas para la conformidad con el dominio de aplicación.

4.2.3 Requisitos adicionales para la especificación de conformidad

Además de los dos puntos anteriores, cada dominio de aplicación dado especificará las siguientes tablas para definir plenamente la conformidad.

NOTA – Para cada una de estas tablas, se da un ejemplo concreto en el apéndice II, definiendo así un ejemplo de dominio de aplicación.

- Codificación de datos de contenido

El dominio de aplicación especificará qué tipo de datos de contenidos y qué tipo de codificación son admitidos. Se rellenarán las dos tablas siguientes para cada dominio de aplicación.

Tipo dato de contenido soportado

Atributo	Valores admisibles
Atributos de tipos de caracteres	
Nombre de tipos de caracteres	
Color absoluto	
Conjunto de caracteres	
Efecto de transición	

Tabla de codificación

Tipo de contenido	Codificación de contenido	Valores de gancho (entero)
Formato de codificación de tipo de caracteres		
Formato de codificación de paleta		
Formato de codificación de diagrama de bits		
Formato de codificación de texto		
Formato de codificación de campo de entrada		
Formato de codificación de hipertexto		
Formato de codificación de tren		
Formato de codificación de arte lineal		
Formato de codificación de forma de cursor		
Formato de codificación de programa intercambiado		
Formato de codificación de color absoluto		

- Registros de entrada de usuario

Para tener una norma práctica de trabajo, el dominio de aplicación especificará uno o más Registros de eventos de entrada (InputEventRegisters). Cada registro tiene un número, que es intercambiado como uno de los parámetros de un objeto escena y un contenido (que no es intercambiado) que consiste en un conjunto de números [que representan Rótulos de eventos de entrada de usuario (UserInputEventTags)] y un nombre. Los pares nombre/número vinculan un Rótulo de evento de entrada de usuario específico con un evento de entrada lógico. Corresponde al implementador del motor vincular el evento de entrada lógico con uno o más eventos de entrada físicos.

Se rellenará la siguiente tabla para cada dominio de aplicación.

Registro N.º	Rótulo de evento de entrada de usuario	Semántica	Comentario
(Entero)

- Evento de motor

Un dominio de aplicación MHEG-5 puede especificar un conjunto de números asociados con un Evento de motor (EngineEvent) para distinguir entre los distintos eventos externos que conducen a la generación de un Evento de motor. En este caso, cada uno de estos eventos de motor específicos corresponderá con un Entero de la tabla como se indica a continuación. Los valores no reservados por el dominio de aplicación están libres para ser utilizados por el programador de aplicaciones.

Evento de motor	Rótulo de evento
...	(Entero)

- Obtención de soporte de motor

Los dominios de aplicación pueden definir, además de las cadenas mencionadas en la presente Recomendación, otras cadenas admisibles para la acción Obtener soporte de motor. En este caso, el dominio de aplicación enumerará claramente estas cadenas adicionales.

- Restricciones semánticas de las aplicaciones MHEG-5

Para cada característica definida por una cadena para la acción Obtener soporte de motor, un dominio de aplicación puede elegir restringir sus aplicaciones de alguna manera. En estos casos, se proporcionará una tabla de restricciones.

Característica	Restricción
...	...

- Correspondencia de protocolos e interacción interna

Por último, hay tres acciones diferentes que tienen un efecto que es externo al tiempo de ejecución. Estas son funciones que extraen objetos, manipulan trenes y llaman funciones externas. Con miras a la interoperabilidad, estas acciones tendrán un efecto externo subyacente común. Esto supone generalmente que hay una correspondencia coherente de estas acciones en las funciones de comunicaciones externas subyacentes para los usuarios de dominios de aplicaciones MHEG-5. Las siguientes correspondencias serán proporcionadas por el dominio de aplicación.

Entidad MHEG-5	Correspondencia necesaria	Semántica de estructuras MHEG-5 que necesitan especificación
Abrir conexión, Cerrar conexión	Correspondencia con protocolos de gestión de conexión (y posiblemente de gestión de sesión) en el dominio de aplicación	<ul style="list-style-type: none"> En Abrir conexión: <ul style="list-style-type: none"> Protocolo Dirección
Objetos programa distante	Correspondencia con protocolo de llamada a programa distante en el dominio de aplicación	<ul style="list-style-type: none"> En Llamada y Ramificación temporal <ul style="list-style-type: none"> Nombre Parámetros Rótulo de conexión de programa
Espacio de nombre de aplicación	Correspondencia con espacio de nombre del dominio de aplicación	<ul style="list-style-type: none"> Referencia de objeto Referencia de contenido
Espacio de nombre de aplicación cuando una acción Transición a utiliza el parámetro Rótulo de conexión	Correspondencia con el espacio de nombre del dominio de aplicación	<ul style="list-style-type: none"> Referencia de objeto Referencia de contenido
Espacio de nombre de almacenamiento persistente	Correspondencia con el espacio de nombre del almacenamiento persistente	<ul style="list-style-type: none"> En Almacenamiento persistente y Lectura persistente: <ul style="list-style-type: none"> Nombre fichero de entrada, Nombre fichero de salida
Acciones de tren	Correspondencia con la interfaz de tren del dominio de aplicación	<ul style="list-style-type: none"> En tren: <ul style="list-style-type: none"> Velocidad Posición de contador
Eventos de tren	Correspondencia con estados de tren y eventos de tren en el dominio de aplicación	<ul style="list-style-type: none"> En tren: <ul style="list-style-type: none"> Presentación tren, Parada tren (correspondencia con la máquina de estados de trenes del dominio de aplicación) Posición de contador Rótulo de evento de tren


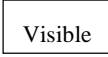
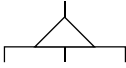
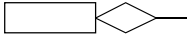
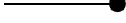
5 Visión general de las clases MHEG-5

La presente Recomendación se ha elaborado para sustentar la distribución de aplicaciones multimedios interactivas en una arquitectura de cliente/servidor¹ a través de plataformas de diferentes tipos y marcas. La presente Recomendación define una representación en forma final para el intercambio de aplicaciones. Las aplicaciones consisten principalmente en un código enunciativo, pero se han establecido disposiciones para un código de procedimiento. Las aplicaciones MHEG-5 sólo tienen que ser creadas una vez y después funcionan en cualquier plataforma que sea conforme a la presente Recomendación. Las aplicaciones desarrolladas residirían en el servidor, y como se necesitan porciones de la aplicación, éstas serán telecargadas al cliente. En un entorno de difusión, este mecanismo de telecarga podrá depender, por ejemplo, de la redifusión cíclica de todas las porciones de la aplicación. Es responsabilidad del cliente tener un tiempo de ejecución que interprete las partes de aplicación, presente la aplicación al usuario y trate la interacción local con el usuario.

Toda aplicación MHEG-5 está formada por escenas y objetos que son comunes a todas las escenas. Una escena contiene un grupo de objetos utilizados para presentar información (gráficos, sonido, vídeo, etc.) junto con comportamiento localizado basado en eventos de activación (por ejemplo, cuando se pulsa el Botón izquierdo se activa un sonido). A lo sumo, sólo habrá una escena activa a la vez. La navegación en una aplicación se hace efectuando transiciones entre escenas.

El sistema interactivo tiene la capacidad de presentar objetos visuales en un sistema de coordenadas rectangulares con un tamaño fijo y reproducir objetos audibles. Se pueden utilizar dispositivos de entrada de usuario (por ejemplo, telecontrol, controlador de juegos, etc.) con el tiempo de ejecución para permitir la interacción con las aplicaciones.

Las figuras en esta cláusula informativa presentan el diagrama de las clases de objeto definidas por la presente Recomendación. Su significado se explica en la figura 1.

	Las casillas muestran una clase MHEG-5 Los nombres de clases en negrita indican una clase concreta
	Los nombres de clases en estilo normal indican una clase abstracta
	Los triángulos muestran relaciones de herencia
	Los diamantes muestran relaciones de composición
	Los puntos negros muestran una relación cero-a-más

T1601930-97

Figura 1/T.172 – Leyenda de los diagramas de clases

¹ El servidor puede ser un servidor *virtual*, como la colección de varios canales de difusión en una red de difusión.

La figura 2 presenta una visión general de la capa superior en la jerarquía de clases MHEG-5. La siguiente parte de esta cláusula presenta los conceptos definidos por la presente Recomendación explicando las clases mostradas en esta figura y sus subclases.

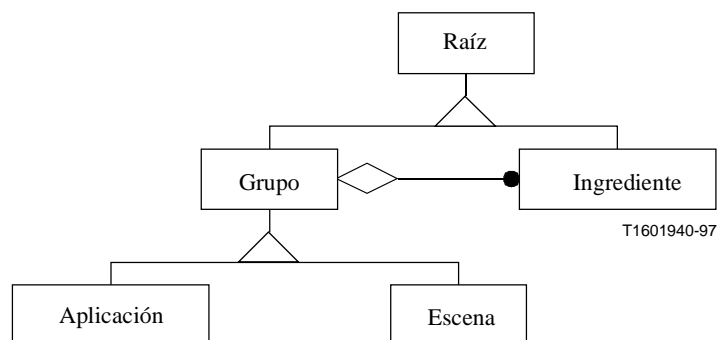


Figura 2/T.172 – Diagrama de la parte superior de la jerarquía de clases

5.1 Raíz

La clase Raíz es la clase básica abstracta para todas las otras clases MHEG-5². Su funcionalidad principal es proporcionar la semántica para el comportamiento MHEG-5 genérico (activación, desactivación, preparación, destrucción) y proporcionar un mecanismo para la identificación de objetos.

5.2 Grupo

La clase Grupo es una clase básica abstracta para las clases Aplicación y Escena. Su principal funcionalidad es permitir la agrupación de objetos de otras clases para intercambio entre el motor MHEG-5 y otras entidades (similar a un «conjunto» en la terminología orientada a objetos normalizada). Los objetos agrupados por esta clase son objetos de la clase Ingrediente. Cada Ingrediente siempre está contenido exactamente en un Grupo. Se puede hacer referencia a objetos dentro de un Grupo a partir de objetos en otros Grupos en determinadas condiciones (véase más adelante).

5.3 Aplicación

Los objetos de la clase Aplicación agrupan objetos de la clase Ingrediente. La clase Aplicación tiene también la restricción semántica de que sólo un objeto Aplicación puede estar activo³ a la vez, y ningún otro objeto puede estar activo a menos que esté activo un objeto Aplicación.

Un motor MHEG-5 en reposo arranca una aplicación preparando y activando el objeto Aplicación correspondiente. Cuando el objeto Aplicación está activo, ejecuta automáticamente una acción Arranque (OnStartup), que puede ser utilizada para ejecutar el primer objeto Escena de la aplicación. Como (exactamente) un objeto Aplicación está activo cuando otro objeto está activo, los ingredientes contenidos en el objeto aplicación son visibles y están disponibles para otros objetos que están

² Con la excepción de las clases mezcladas abstractas y la clase Acción.

³ Para la definición del término "activo", véase 3.3.

activos simultáneamente. Más concretamente, cualesquiera Ingredientes contenidos en un objeto Aplicación están disponibles para la Escena activa. Esto se puede utilizar para describir el comportamiento de toda la aplicación.

5.4 Escena

Los objetos de la clase Escena agrupan a objetos de la clase Ingrediente. La finalidad de la clase Escena es permitir la presentación coordinada espacial y/o temporalmente. Sólo un objeto Escena puede estar activo a la vez dentro de un motor MHEG-5. Un objeto Escena puede estar activo para cualquier ingrediente (esté contenido en una Escena o en un objeto Aplicación) que ha de ser visualizado.

La clase Escena proporciona la acción especial Transición a, que hace posible efectuar una transición gráfica entre dos escenas. Los objetos contenidos en una escena sólo pueden ser visualizados cuando la escena está activa. El objeto que tiene que ser visualizado a través de varias escenas (por ejemplo, para tener la presentación ininterrumpida en una escena, la transición debe estar contenida en un objeto aplicación). Por último, la clase Escena proporciona información sobre el sistema de coordenadas que se ha de utilizar para la presentación visual.

5.5 Ingrediente

La clase Ingrediente es una clase básica abstracta para las clases Enlace, Programa, Paleta, Tipo de carácter, Forma de cursor, Variable y Presentable. Las subclases de la clase Ingrediente se muestran en la figura 3. La funcionalidad principal de la clase Ingrediente es especificar el comportamiento genérico de objetos que pueden formar parte de una Escena o de un objeto Aplicación.

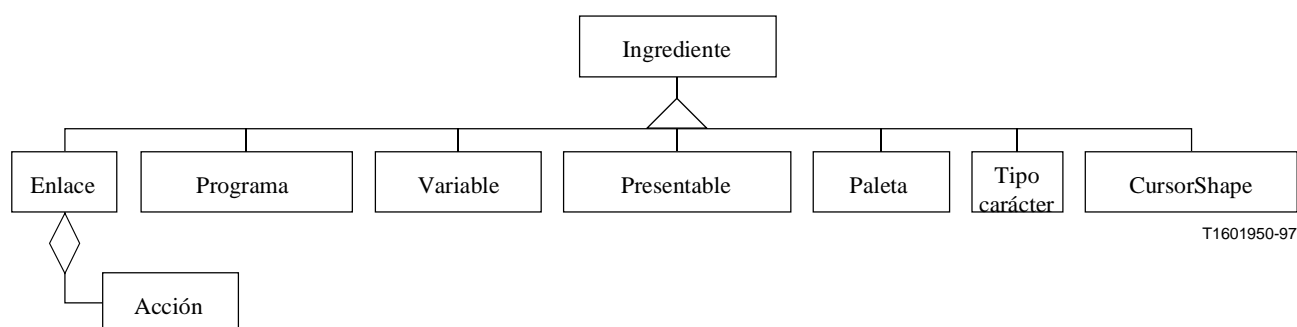


Figura 3/T.172 – Diagrama de subclases de la clase Ingrediente

5.6 Enlace

Los objetos Enlace se utilizan para expresar el comportamiento de aplicaciones MHEG-5. Un objeto Enlace consiste en una condición y en un objeto Acción. Cuando el valor de la parte condición es Verdadero, se dice que el enlace «dispara» («fire»); esto conduce a la ejecución del objeto Acción asociado. La condición contiene tres partes: un código de evento (que identifica el evento al que se ha de responder), una referencia al objeto del cual debe emanar el evento, y un valor que especifica el valor requerido del parámetro del evento. En otras palabras, un Enlace sólo dispara si está activo y si el evento correcto es generado por el objeto correcto y contiene el parámetro de evento correcto. Los Enlaces activos serán parte de una Escena activa o de una Aplicación activa.

5.7 Acción

Un objeto Acción tiene la funcionalidad de ejecutar, en secuencia síncrona, una serie de «acciones elementales» como resultado del disparo de un Enlace. Una acción elemental consiste en el objeto al cual la acción ha de ser «dirigida» y una lista de valores que representan los parámetros de la acción. De hecho, dirigir una acción elemental a un objeto corresponde a llamar a un método de un objeto en cualquier lenguaje de programación ordinario orientado a objetos. Todas las acciones elementales disponibles se enumeran en la cláusula 49.

La clase Acción no hereda de ninguna otra clase MHEG-5. Concretamente, no hereda de Raíz, lo que significa que los objetos Acción no pueden ser tratados como entidades individuales.

5.8 Programa

La clase Programa proporciona la funcionalidad de llamar a una pieza de código de procedimiento desde dentro del contexto MHEG-5 e intercambiar parámetros con éste.

La clase Programa tiene las siguientes tres subclases correspondientes a los tres tipos de llamadas de procedimiento que se pueden hacer:

- Programa residente (ResidentProgram)
Llamada de procedimiento a una pieza de código que es específica del dispositivo en el cual está funcionando el motor MHEG-5. Se puede utilizar, por ejemplo, para llamar bibliotecas de tiempo de ejecución específicas del dispositivo.
- Programa distante (RemoteProgram)
Llamada de procedimiento a una pieza de código que está situada en un dispositivo diferente a aquél donde está funcionando el motor MHEG-5. Se puede utilizar, por ejemplo, para efectuar una llamada a un programa distante, cuando el cuerpo real del programa está situado en el servidor en un sistema cliente-servidor.
- Programa intercambiado (InterchangedProgram)
Llamada de procedimiento a una pieza de código que es intercambiada como una parte de un objeto MHEG-5. La finalidad de esta clase es proporcionar la funcionalidad necesaria para intercambiar piezas de código de procedimientos, y para llamar a estas piezas de código.

5.9 Paleta, Tipo de carácter y Forma de cursor

La clase Paleta proporciona la posibilidad de encapsular la representación codificada de una tabla de mejora de colores (CLUT, *colour Look-up table*). La función de una CLUT es traducir un índice de colores a un valor de color verdadero. La Paleta se puede utilizar, por ejemplo, con diagramas de bits para especificar los colores en los cuales se ha de reproducir el diagrama de bits.

De manera similar, la clase Tipo de carácter permite que las aplicaciones encapsulen la representación codificada de un tipo de carácter. Un objeto Tipo de carácter, cuando está asociado con un objeto Texto, se utiliza para reproducir el texto de ese objeto.

Por último, la clase Forma de cursor permite que las aplicaciones encapsulen la representación codificada de los diagramas de bits y otros datos necesarios para lograr un cursor de movimiento libre. La forma del cursor de movimiento libre se puede fijar y reiniciar utilizando un método de la clase Escena.

Para los Tipos de caracteres, las Paletas y las Formas de cursor, la representación real de los objetos no se especifica en la presente Recomendación. Sin embargo, un dominio de aplicación en el cual los tipos de caracteres y/o CLUT y/o cursores de movimiento libre sean características necesarias de las aplicaciones, puede especificar su codificación y semántica.

5.10 Variable

La clase Variable proporciona la posibilidad de almacenar y extraer valores. La clase Variable tiene cinco subclases correspondientes a cinco tipos diferentes de variables:

- Variable booleana (BooleanVariable).
- Variable entera (IntegerVariable).
- Variable cadena de octetos (OctetStringVariable).
- Variable referencia de objeto (ObjectRefVariable).
- Variable referencia de contenido (ContentRefVariable).

Los posibles usos de las Variables comprenden los parámetros que pasan a y desde llamadas a Programas, almacenamiento del estado de otros objetos MHEG-5, paso de valores de parámetros indirecto a acciones y falta de dirección, es decir, como punteros a objetos MHEG-5.

5.11 Presentable

La clase Presentable es una clase básica abstracta para las clases MHEG-5 Audio, Visible, Grupo de testigos y Tren. Las subclases de la clase Presentable se muestran en la figura 4.

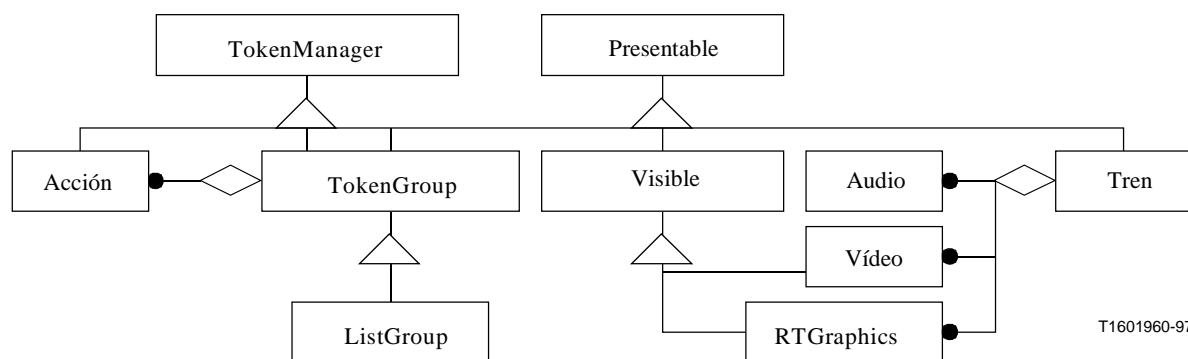


Figura 4/T.172 – Diagrama de subclases de la clase Presentable

Los objetos de la clase Presentable representan la información que puede ser vista u oída directamente por el usuario. Una funcionalidad importante de la clase Presentable es tratar la representación codificada real de los datos de contenido. Esto se puede hacer por «inclusión» o por «referencia». En el primer caso, los datos de contenido se transmiten realmente como parte del propio objeto presentable y en el segundo caso, el objeto Presentable sólo proporciona una referencia externa a los datos.

5.12 Grupo de testigos

La clase Grupo de testigos proporciona la facilidad de que un testigo lógico navegue entre un conjunto de objetos Visibles. Esta estructura se puede utilizar, por ejemplo, para gestionar la navegación de un «enfoque» entre un conjunto de botones u otros elementos de una Escena. Es posible vincular también algunos conjuntos de acciones a los objetos y ejecutarlas a petición en el objeto Visible que tiene el testigo. Esta última característica proporciona una manera compacta de expresar el comportamiento cuando un elemento del grupo obtiene o pierde el enfoque.

5.13 Grupo de listas

La clase Grupo de listas completa la clase Grupo de testigos, que proporciona funcionalidad para seleccionar objetos en una larga lista. Es la más adecuada para implementar un menú para selección, un grupo de casillas, un carrusel de diagrama de bits, un formulario para rellenar, una lista de elementos que pueden desfilarse verticalmente, etc. Además, la clase Grupo de listas proporciona facilidades para añadir y suprimir dinámicamente elementos en el grupo.

5.14 Tren

La clase Tren define un multiplex de medios continuos para sincronización en el tiempo. Los objetos Audio, Vídeo y Gráficos RT pueden ser trenes elementales de un Multiplex de trenes: están destinados a ser presentados al mismo tiempo al usuario. Esta estructura se puede utilizar, por ejemplo, para presentar vídeo y audio sincronizados y conmutar de un canal de audio a otro. Los datos de contenido del objeto Tren es una referencia a un multiplex real que contiene los trenes elementales y algunos datos adicionales para sincronización. Durante el proceso de presentación, el ejecutor de Trenes genera eventos basados en el tiempo y eventos basados en el marcador que pudieran ser utilizados por la aplicación MHEG-5 para activar algunos Enlaces.

5.15 Audio

La clase Audio implementa una secuencia de datos de audio que se puede utilizar como un tren elemental de un Multiplex de trenes.

5.16 Interactuable

La clase Interactuable es una clase mezclada abstracta heredada por las clases MHEG-5 Hipertexto, Campo de entrada, Deslizador y Botón. Su principal funcionalidad es permitir que el usuario interactúe con objetos de sus subclases. Estas interacciones permiten al usuario cambiar el estado y/o la apariencia de los objetos, por ejemplo, introduciendo texto en un objeto Campo de entrada. Cuando se produce la interacción, una determinada clase de eventos, denominados «eventos de entrada de usuario», no son visibles a los objetos Enlace, porque se supone que estos eventos sean utilizados para la interacción de usuario. La interacción puede ser abortada dirigiendo una determinada acción al objeto Interactuable.

Otra funcionalidad de la clase Interactuable es la capacidad de generar eventos asociados con cursores de movimiento libre (Entrada de cursor, Salida cursor).

5.17 Visible

La clase Visible implementa la funcionalidad asociada con la presentación de piezas de material visual en algún lugar de la pantalla de visualización. En la figura 5 se presentan las subclases de la clase Visible.

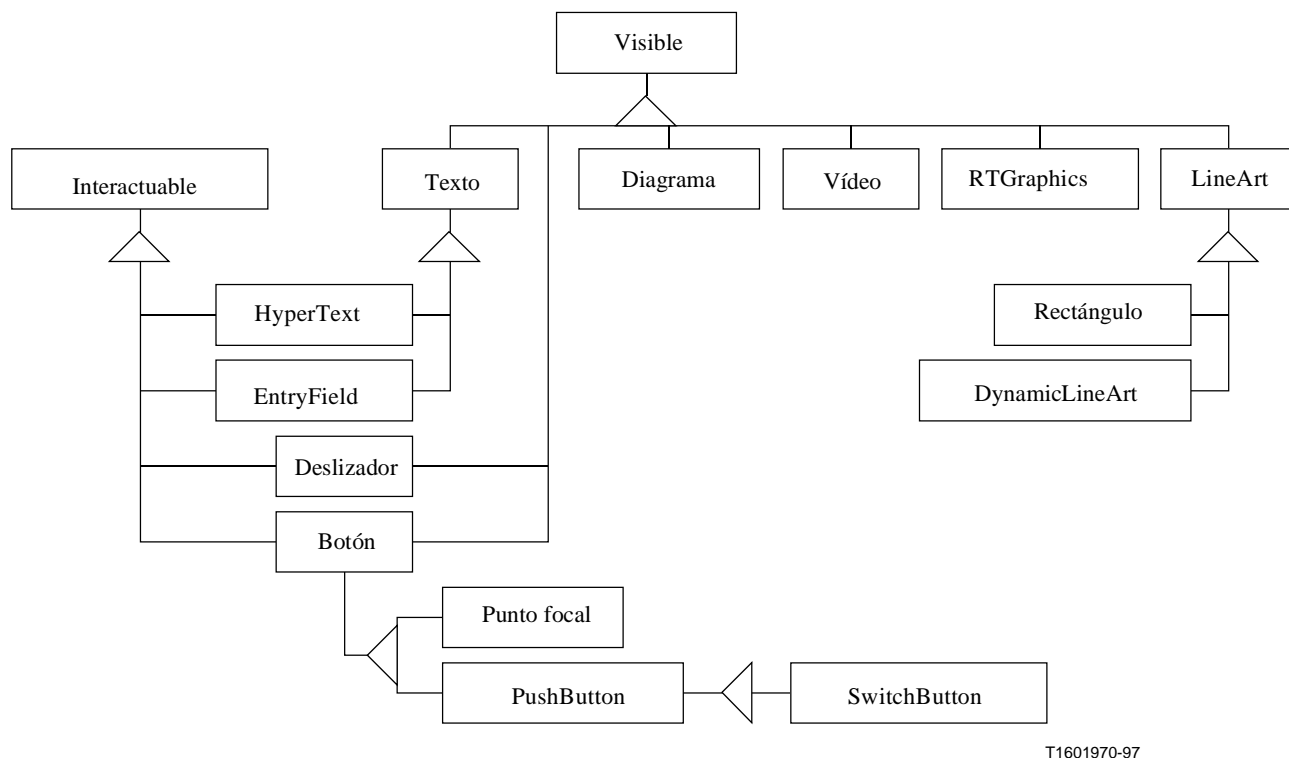


Figura 5/T.172 – Diagrama de subclases de la clase Visible

A continuación se describen las subclases de la clase Visible:

- Arte lineal (LineArt)
Un objeto de la clase Arte lineal representa un objeto gráfico con representación vectorial. Se puede utilizar, por ejemplo, para presentar objetos polilineales, elipses, curvas biseladas, etc.
- Arte lineal dinámico (DynamicLineArt)
Un objeto de la clase Arte lineal dinámico representa un objeto gráfico que puede ser cambiado dinámicamente. Se puede utilizar para dibujar líneas o curvas que tienen que ser presentadas al vuelo.
- Rectángulo (Rectangle)
- Diagrama de bits (Bitmap)
- Vídeo (Video)
- Gráficos RT (RealTimeGraphics)
Un objeto de la clase Gráficos RT [Gráficos en tiempo real, (*RTGraphics (real time graphics)*)] representa un tren de objetos gráficos que son visualizados utilizando colocación y sincronización autónomas. El tren de Gráficos RT se puede utilizar junto con vídeo y audio, por ejemplo, para crear una aplicación de subtítulo.
- Texto (Text)
Los objetos texto representan cadenas de texto. Un objeto Texto puede estar asociado con un objeto Tipo de carácter, que describe el tipo de carácter en el cual se debe presentar el texto (cuando no se indica el objeto tipo de carácter, se supone el tipo de carácter por defecto). El texto tiene dos subclases, Hipertexto y Campo de entrada, que implementan diferentes tipos de texto interactivo.

- **Deslizador (Slider)**
Un Deslizador es una clase Interactuable utilizada para dejar que el usuario fije linealmente una posición dentro de una determinada gama (comprendida entre un valor mínimo y un valor máximo).
- **Botón (Button)**
La clase Botón tiene dos subclases: Botón de pulsación y punto focal. La clase Botón de pulsación tiene una subclase: Botón de conmutación. Los botones son zonas rectangulares en la pantalla con las cuales el usuario puede interactuar. Cada uno de los tres tipos de botones tiene un comportamiento específico que genera eventos. Los Botones de pulsación y los Botones de conmutación están asociados con un elemento de texto, que representa el texto que se ha de visualizar en el medio del botón.

6 Estructura de la presente Recomendación

Las siguientes cláusulas de la presente Recomendación definen la semántica de las clases MHEG-5. Esto se hace sobre la base de la notación de sintaxis abstracta utilizada para describir los atributos de los objetos intercambiados. La semántica de las clases de objetos se describen en el texto normativo. Las cláusulas 51 a 54 definen algunos mecanismos normativos que el motor MHEG-5 tiene que ejecutar. El anexo A define la sintaxis en forma final que se ha de utilizar para el intercambio de objetos. El anexo B define un formato de intercambio textual que tiene una correspondencia de uno a uno con el formato de intercambio binario en forma final.

7 Notaciones

En las siguientes cláusulas se utilizan las siguientes notaciones para describir las clases MHEG-5 definidas por la presente Recomendación:

<Nombre de la clase>

Descripción	<Descripción breve de la semántica de la clase>
Clase básica	<Nombre de la clase básica>
Subclases	<Lista de subclases, si las hubiere>
Estado	<Clase abstracta Clase concreta>

7.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para la clase.

7.1.1 Atributos heredados

Los requisitos y restricciones de los atributos heredados de las clases básicas.

Nombre de atributo	Definido en	Restricciones y requisitos
<Nombre de atributo>	<Nombre de clase>	<Restricciones específicas para la subclase vigente>

7.1.2 Atributos intercambiados propios

Lista de atributos intercambiados para esta clase.

<Nombre de atributo> <Descripción del atributo>

<Tipo del atributo>

<Valor por defecto>

Cuando el atributo es facultativo, el valor por defecto es el valor que se utilizará cuando el atributo no está codificado. Cuando el atributo es obligatorio, el valor por defecto es una indicación del valor más usual que se ha de emplear.

7.1.3 Atributos internos propios

Lista de atributos internos para esta clase.

<Nombre de atributo> <Descripción del atributo>

<Valor inicial>

7.2 Eventos

Éstos son los eventos que pueden ser generados de objetos de esta clase. Se utilizan para expresar condiciones del enlace que serán comprobadas cuando se genera uno de estos eventos.

<Nombre de evento> <Descripción del evento>

<Contexto de ocurrencia del evento>

<Descripción de los datos de evento asociados con el evento, si los hubiere>

7.3 Comportamientos internos

Se definen los siguientes comportamientos internos para la mayoría de las clases MHEG-5:

- *Preparación (Preparation).*
- *Destrucción (Destruction).*
- *Activación (Activation).*
- *Desactivación (Deactivation).*

Además, se define el comportamiento interno *Interacción* para algunas clases MHEG-5. Estos comportamientos corresponden a requisitos semánticos para operaciones internas del motor MHEG-5, similares a la noción de métodos «privados» de una clase en la terminología orientada a objetos. No se codifican ni se transmiten dentro de una descripción de Aplicación.

Cuando no se describe un comportamiento para una clase MHEG-5, se aplica la semántica del comportamiento para su clase básica.

<Nombre de comportamiento> <Semántica del comportamiento dentro del contexto de la clase vigente>

7.4 Efecto de las acciones MHEG-5

Se define la semántica y la sintaxis de acciones MHEG-5 que pueden ser dirigidas a la clase MHEG-5 vigente.

<Nombre de acción> <Semántica de la acción dentro del contexto de la clase vigente>

7.5 Descripción formal

Descripción de un objeto MHEG-5 codificado de la clase vigente en el formato Backus-Naur ampliada (EBNF, Extended Backus-Naur Form):

Class Name	--> First part, Second part
First part	--> Subpart Alternative
Second part	--> Terminal (constant or type)

8 Clase Raíz

Descripción: Clase Raíz de todas las clases MHEG-5

Clase básica: Ninguna

Subclases: Grupo, Ingrediente

Estado: Clase abstracta

8.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

8.1.1 Atributos heredados

Esta clase no tiene atributos heredados.

8.1.2 Atributos intercambiados propios

Esta clase define el siguiente atributo intercambiado adicional:

- Identificador de objeto (ObjectIdentifier) Ésta es una estructura de datos obligatoria, compuesta de las siguientes partes:
- Identificador de grupo facultativo.
Identificador único de un grupo de objetos MHEG-5.
 - Cadena de octetos facultativa.
 - Valor por defecto: El Identificador de grupo del Grupo dentro del cual está codificado el objeto.
- La estructura real de este parámetro no se define en la presente Recomendación. Sin embargo, el dominio de aplicación definirá esta estructura. Véase la cláusula 51.
- Número de objeto.
Identificador único de cualquier objeto MHEG-5 dentro de un grupo.
 - Entero.

8.1.3 Atributos MHEG-5 internos propios

Esta clase define los siguientes atributos internos:

- Estado de disponibilidad* Estado de disponibilidad del objeto.
- Cuando el *Estado de disponibilidad* del objeto es Verdadero, el objeto está disponible, lo que significa que el comportamiento *Preparación* del objeto ha terminado satisfactoriamente.

Cuando el *Estado de disponibilidad* del objeto es Falso, el objeto no está disponible, lo que significa que el comportamiento *Preparación* no ha terminado satisfactoriamente o no ha sido invocado, o que se ha aplicado satisfactoriamente el comportamiento *Dstrucción*.

- Valor booleano.
- Valor por defecto: Falso.

Estado de ejecución

Estado de actividad del objeto.

Cuando el *Estado de ejecución* del objeto es Verdadero, el objeto está activo, lo que significa que se ha completado satisfactoriamente el comportamiento *Activación* del objeto.

Cuando el *Estado de ejecución* del objeto es Falso, el objeto está inactivo, lo que significa que no se ha completado satisfactoriamente o no ha sido invocado el comportamiento *Activación*, o que se ha aplicado satisfactoriamente el comportamiento *Desactivación*.

- Valor booleano.
- Valor por defecto: Falso.

8.2 Eventos

Esta clase define los siguientes eventos:

Está disponible (IsAvailable)

Este evento es generado cuando el atributo *Estado de disponibilidad* cambia de Falso a Verdadero como resultado del comportamiento *Preparación* completado satisfactoriamente para el objeto.

NOTA – La finalidad de este evento es indicar al motor MHEG-5 que el objeto está disponible y puede ser activado.

- Ningún dato asociado.

Contenido disponible (ContentAvailable)

Este evento es generado cuando el objeto y su contenido están disponibles en un estado óptimo para el motor MHEG-5. Su finalidad es indicar al motor MHEG-5 que el comportamiento *Activación* puede ocurrir oportunamente. Este evento es generado asíncronamente con el comportamiento *Preparación* para el objeto.

NOTA – Cada motor MHEG-5 puede elegir dar un significado diferente a ese grado de disponibilidad de contenido; la presente Recomendación no especifica ningún significado o requisito temporal en este asunto.

- Ningún dato asociado.

Está suprimido (IsDeleted)

Este evento es generado cuando el atributo *Estado de disponibilidad* cambia de Verdadero a Falso como resultado de la terminación satisfactoria del comportamiento *Dstrucción* para el objeto.

- Ningún dato asociado.

<i>Está funcionando</i> (<i>IsRunning</i>)	<p>Este evento es generado cuando el atributo <i>Estado de ejecución</i> cambia de Falso a Verdadero como resultado de la terminación satisfactoria del comportamiento <i>Activación</i> para el objeto.</p> <ul style="list-style-type: none"> Ningún dato asociado.
<i>Detenido</i> (<i>IsStopped</i>)	<p>Este evento es generado cuando el atributo <i>Estado de ejecución</i> cambia de Verdadero a Falso como un resultado de la terminación satisfactoria del comportamiento <i>Desactivación</i> para el objeto.</p> <ul style="list-style-type: none"> Ningún dato asociado.

8.3 Comportamientos internos

Esta clase define los siguientes comportamientos internos:

<i>Preparación</i>	<p>Este comportamiento tiene la semántica básica de asignar todos los recursos solicitados para tratar o presentar este objeto.</p> <p>Aplicar la siguiente secuencia de acciones:</p> <ol style="list-style-type: none"> 1) Si el atributo <i>Estado de disponibilidad</i> del objeto es Verdadero, abortar el comportamiento. En los demás casos: 2) Extraer el objeto de una entidad fuera del motor. 3) Fijar cada atributo interno del objeto a su valor inicial. 4) Fijar el atributo <i>Estado de disponibilidad</i> a Verdadero. 5) Generar un evento <i>Está disponible</i>. <p>Los pasos anteriores se ejecutan <i>síncronamente</i>. El siguiente paso es asíncrono:</p> <ol style="list-style-type: none"> 6) Generar un evento <i>Contenido disponible</i>.
<i>Destrucción</i>	<p>Este comportamiento tiene la semántica básica de pedir al motor MHEG-5 que suprima el objeto.</p> <p>Aplicar la siguiente secuencia de acciones:</p> <ol style="list-style-type: none"> 1) Si el atributo <i>Estado de disponibilidad</i> del objeto es Falso, abortar el comportamiento. En los demás casos: 2) Si el atributo <i>Estado de ejecución</i> del objeto es Verdadero: <ol style="list-style-type: none"> a) aplicar el comportamiento <i>Desactivación</i>; b) esperar un evento <i>Detenido</i> del objeto. <p>Esto se hará <i>síncronamente</i>.</p> 3) Si el atributo <i>Estado de ejecución</i> del objeto es Falso, ejecutar las siguientes acciones <i>síncronamente</i>. 4) Si el atributo <i>Prioridad de cache de grupo</i> del propio objeto o del grupo al que pertenece este objeto está puesto a 0, la máquina MHEG-5 liberará todos los recursos asignados al objeto. <p>Obsérvese que <i>Prioridad de cache de grupo</i> se define en la clase Grupo.</p> <ol style="list-style-type: none"> 5) Si el atributo <i>Prioridad de cache de grupo</i> del propio objeto o del grupo al que pertenece este objeto es diferente de cero, el motor

MHEG-5 puede decidir liberar todos los recursos asignados al objeto u ocultarlo.

- 6) Generar un evento *Está suprimido*.

Obsérvese que el evento *Está suprimido* será generado estén los recursos mencionados anteriormente realmente liberados o no; el objeto es suprimido en el sentido definido por esta Recomendación, incluso si algunos recursos asociados no lo son.¹¹

Activación

Este comportamiento tiene la semántica básica de hacer inmediatamente activo este objeto.

Aplicar la siguiente secuencia de acciones:

- 1) Si el atributo *Estado de ejecución* del objeto es Verdadero, abortar el comportamiento. En los demás casos:
- 2) Si el atributo *Estado de disponibilidad* del objeto es Falso:
 - a) aplicar el comportamiento *Preparación* al objeto;
 - b) esperar un evento *Está disponible* del objeto.

Estos pasos se ejecutan síncronamente, lo que significa que el motor no realizará otras acciones hasta que haya terminado el comportamiento *Preparación*.

NOTA 1 – El efecto del comportamiento *Activación* (por ejemplo, la visualización de un diagrama de bits) continuará incluso después que el propio comportamiento haya terminado.

NOTA 2 – La generación de un evento *Está funcionando* y la modificación del atributo interno *Estado de ejecución* son partes del comportamiento *Activación* de subclases de la clase Raíz.

Desactivación

Este comportamiento tiene la semántica básica de notificar al motor MHEG-5 que desactive este objeto inmediatamente.

Aplicar la siguiente secuencia de acciones:

- 1) Si el atributo *Estado de ejecución* del objeto es Falso, abortar el comportamiento. En los demás casos:
- 2) Fijar a Falso el atributo *Estado de ejecución* del objeto.
- 3) Generar un evento *Está detenido*.

8.4 Efecto de las acciones MHEG-5

Esta clase define las siguientes acciones MHEG-5 aplicables:

Obtener estado de disponibilidad
(*Variable estado de disponibilidad*)

Fijar la variable referenciada por la variable *estado de disponibilidad* al valor del atributo *Estado de disponibilidad*.

NOTA – Una acción Obtener estado de disponibilidad dirigida a un objeto inexistente en el motor MHEG-5 no es un error; el resultado es Falso.

[Get AvailabilityStatus
(*AvailabilityStatusVar*)]

Disposición de uso:

- La *Variable estado de disponibilidad* hará referencia a un objeto Variable booleana.

Descripción de sintaxis:

GetAvailabilityStatus	-->	Target ,
		AvailabilityStatusVar
Target	-->	GenericObjectReference
AvailabilityStatusVar	-->	ObjectReference

Obtener estado de ejecución
(*Variable estado de ejecución*)

[GetRunningStatus
(RunningStatus Var)]

Fijar la Variable a que hace referencia la *Variable estado de ejecución* al valor del atributo *Estado de ejecución*.

Disposiciones de uso:

- El objeto Objetivo estará disponible.
- La *Variable estado de ejecución* hará referencia a un objeto Variable booleana.

Descripción de sintaxis:

GetRunningStatus	-->	Target ,
		RunningStatusVar
Target	-->	GenericObjectReference
RunningStatusVar	-->	ObjectReference

8.5 Descripción formal

Root Class	-->	ObjectIdentifier
ObjectIdentifier	-->	ObjectReference

9 Clase de grupo

Descripción: Define la estructura y comportamiento de objetos utilizados como composición de Ingredientes

Clase básica: Raíz

Subclases: Escena, Aplicación

Estado: Clase abstracta

9.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

9.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las siguientes restricciones:

Nombre de atributo	Definido en	Restricciones y requisitos
Identificador de objeto	Raíz	<p>Este atributo es obligatorio para esta clase.</p> <p>La parte Identificador de grupo de este atributo es obligatoria y será única dentro del espacio de nombre del dominio de aplicación.</p> <p>La parte Número de objeto de este atributo se pondrá a cero.</p>

9.1.2 Atributos intercambiados propios

Esta clase define los siguientes atributos intercambiados adicionales:

Identificador de norma (StandardIdentifier)	<p>Ésta es una secuencia facultativa de dos Enteros. Cuando se codifica:</p> <ul style="list-style-type: none">• el primer Entero se pondrá siempre a 2, lo que significa «Conjunta ISO/UIT»;• el segundo Entero se pondrá siempre a 19, lo que significa MHEG.
Versión normalizada (StandardVersion)	<p>Éste es un Entero facultativo. Define la versión de la presente Recomendación con la cual se conforman los objetos Grupo y todos sus ítems.</p> <ul style="list-style-type: none">• Entero facultativo. Si se codifica, su valor será 1.• Valor por defecto: 1.
Información de objeto (ObjectInformation)	<p>Ésta es una Cadena de octetos facultativa. Cuando se codifica, mantendrá información sobre los objetos codificados en este Grupo. Esta información puede comprender información sobre Nombre de objeto, Propietario, Versión, Fecha, Palabras clave, Derechos de autor, Licencia y Comentarios.</p>
Arranque (OnStartUp)	<p>Conjunto de acciones elementales que se han de ejecutar al final del comportamiento <i>Activación</i> para el Grupo.</p> <ul style="list-style-type: none">• Inclusión facultativa de un objeto Acción.• Valor por defecto: Ninguno.
Cierre (OnCloseDown)	<p>Conjunto de acciones elementales que se han de ejecutar al comienzo del comportamiento <i>Desactivación</i> para el Grupo.</p> <ul style="list-style-type: none">• Inclusión facultativa de un objeto Acción.• Valor por defecto: Ninguno.
Prioridad de cache de grupo original (OriginalGroup-CachePriority)	<p>Indicación al motor MHEG-5 sobre la pertinencia de ocultar este Grupo de Ingredientes cuando es destruido.</p> <p>Valor de <i>Prioridad de cache de grupo</i> cuando el Grupo es preparado.</p> <ul style="list-style-type: none">• Entero facultativo dentro de la gama [0, 255].• Valor por defecto: 127.• Valor específico: 0 significa que la ocultación no está permitida para este Grupo y sus ingredientes. <p>NOTA – Como se especifica en 4.2.2, la ocultación de cualquier clase es una característica facultativa de un motor MHEG-5.</p>
Ítems	<p>Conjunto de objetos Ingredientes que pertenecen al Grupo. Cuando el Grupo no contiene Ingredientes, no se codificará este atributo. Cuando este atributo se codifica, contendrá por lo menos un ingrediente.</p> <ul style="list-style-type: none">• Atributo facultativo.• Secuencia de inclusiones de objetos Ingredientes.• Valor por defecto: Ninguno.

9.1.3 Atributos internos propios

Esta clase define el siguiente atributo interno:

<i>Prioridad de cache de grupo</i> (<i>GroupCachePriority</i>)	<p>Indicación al motor MHEG-5 sobre la pertinencia de ocultar este Grupo de Ingredientes cuando es destruido.</p> <p>La <i>Prioridad de cache de grupo</i> se puede comparar con <i>Prioridad de cache de grupo</i> de otros objetos Grupo para determinar cuál de varios grupos tiene la probabilidad más alta de ser requerido de nuevo por la aplicación, una vez que ha sido destruido. Un valor más alto indica un nivel de prioridad más alto. Es responsabilidad del diseñador de la aplicación mantener estos números en una gama coherente. Se recomienda al motor MHEG-5 que oculte objetos con prioridad más alta de preferencia a objetos con prioridad más baja.</p> <ul style="list-style-type: none">• Entero facultativo dentro de la gama [0, 255].• Valor inicial: Valor del atributo Prioridad de cache de grupo original.• Valor específico: 0 significa que la ocultación no está permitida para este Grupo y sus Ingredientes.
---	--

9.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

9.3 Comportamientos internos

La semántica de los siguientes comportamientos internos ha cambiado con respecto a la clase básica de este objeto:

<i>Preparación</i>	<ol style="list-style-type: none">1) Aplicar el comportamiento <i>Preparación</i> a todos los Ingredientes del Grupo que tienen el atributo Activo inicialmente puesto a Verdadero y a todos los Programas del Grupo que tienen el atributo Disponible inicialmente puesto a Verdadero en el orden en que están enumerados en el atributo Ítems.2) Aplicar el comportamiento <i>Preparación</i> heredado de la clase básica.
<i>Destrucción</i>	<ol style="list-style-type: none">1) Aplicar el comportamiento <i>Destrucción</i> a todos los Ingredientes del Grupo en el orden inverso en que están enumerados en el atributo Ítems.2) Aplicar el comportamiento <i>Destrucción</i> heredado de la clase básica.
<i>Activación</i>	<ol style="list-style-type: none">1) Aplicar el comportamiento <i>Activación</i> heredado de la clase básica.2) Ejecutar la acción contenida en el atributo Arranque.3) Aplicar el comportamiento <i>Activación</i> a todos los Ingredientes del Grupo que tienen el atributo Activo inicialmente puesto a Verdadero, en el orden en que están enumerados en el atributo Ítems.4) Fijar a Verdadero el atributo <i>estado de ejecución</i> del objeto Grupo.5) Generar un evento <i>Está funcionando</i>.

- Desactivación* Si el grupo no está activo, omitir el comportamiento. Si el grupo está activo, ejecutar los tres pasos siguientes:
- 1) Ejecutar la acción contenida en el atributo Cierre.
 - 2) Aplicar el comportamiento *Desactivación* a todos los Ingredientes activos del Grupo, en el orden inverso en que están enumerados en el atributo *Ítems*.
 - 3) Aplicar el comportamiento *Desactivación* heredado de la clase básica.

9.4 Efecto de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se define las siguiente acción MHEG-5 aplicable:

Fijar prioridad de cache
(*Nueva prioridad de cache*)
[SetCachePriority
(*NewCachePriority*)]

Fijar el atributo *Prioridad de cache de grupo* a *Nueva prioridad de cache*.

Disposiciones de uso:

- El objeto *Objetivo* estará disponible.
- *Nueva prioridad de cache* se fijará dentro de la gama [0, 255].

Descripción de sintaxis:

SetCachePriority	-->	Target , NewCachePriority
Target	-->	GenericObjectReference
NewCachePriority	-->	GenericInteger

9.5 Descripción formal

Group Class	-->	Root Class , StandardIdentifier? , StandardVersion? , ObjectInformation? , OnStartUp? , OnCloseDown? , OriginalGroupCachePriority? , Items?
StandardIdentifier	-->	Joint ISO ITU (2) , MHEG (19)
StandardVersion	-->	INTEGER
ObjectInformation	-->	OctetString
OnStartUp	-->	Action Class
OnCloseDown	-->	Action Class
OriginalGroupCachePriority	-->	INTEGER
Items	-->	Item+
Item	-->	ResidentProgram Class RemoteProgram Class InterchangedProgram Class

```

Palette Class |
Font Class |
CursorShape Class |
BooleanVariable Class |
IntegerVariable Class |
OctetStringVariable Class |
ObjectRefVariable Class |
ContentRefVariable Class |
Link Class |
Stream Class |
Bitmap Class |
LineArt Class |
DynamicLineArt Class |
Rectangle Class |
Hotspot Class |
SwitchButton Class |
PushButton Class |
Text Class |
EntryField Class |
HyperText Class |
Slider Class |
TokenGroup Class |
ListGroup Class

```

10 Clase Aplicación

Descripción: Define un conjunto de objetos Ingredientes, que son compartidos dentro un ámbito de aplicación

Clase básica: Grupo

Subclases: Ninguna

Estado: Clase concreta

10.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

10.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con semántica idéntica.

10.1.2 Atributos heredados propios

Esta clase define los siguientes atributos intercambiados adicionales:

Cierre de ramificación permanente (OnSpawnCloseDown)	Objeto acción que se ha de ejecutar cuando la Aplicación es cerrada por otra Aplicación por medio de la acción Ramificación permanente. Puede ser útil, por ejemplo, para almacenar información que será utilizada cuando se reanuda esta Aplicación.
---	---

	<p>Cierre de ramificación permanente puede ser invocada por la acción Ramificación permanente.</p> <ul style="list-style-type: none"> • Inclusión facultativa de un objeto Acción. • Valor por defecto: Ninguno.
Rearranque (OnRestart)	<p>Objeto acción que ha de ser ejecutado cuando se rearranca la Aplicación. Puede ser útil, por ejemplo, para extraer información que fue almacenada por Cierre de ramificación permanente.</p> <p>Rearranque sólo puede ser invocado por la acción Cesar (véase Cesar).</p> <ul style="list-style-type: none"> • Inclusión facultativa de un objeto Acción. • Valor por defecto: Ninguno
Atributos por defecto (DefaultAttributes)	<p>Define atributos por defecto de toda la Aplicación que serán utilizados como valores por defecto cuando no se especifica un correspondiente atributo de Ingrediente. Se pueden fijar los siguientes atributos por defecto:</p>
Conjunto de caracteres (CharacterSet)	<p>El conjunto de caracteres por defecto, o conjunto de conjuntos de caracteres, para la reproducción de texto en toda la aplicación, salvo cuando se especifica otra cosa. Este Entero se codificará con un valor que representa el conjunto. El dominio de aplicación definirá la gama de Conjunto de caracteres y su semántica.</p> <p>NOTA – El atributo <i>Conjunto de caracteres</i> de la Aplicación proporciona el conjunto de caracteres inicial para todos los objetos que contienen texto en la Aplicación que no lo especifica.</p> <ul style="list-style-type: none"> • Entero facultativo. • Valor por defecto: Ninguno
Color de fondo (BackgroundColour)	<p>Color por defecto que se ha de utilizar para reproducir el fondo de un objeto texto. Este atributo se interpreta como un índice basado en cero en la tabla de colores definida por el atributo <i>Referencia de paleta</i>, o como un valor de color directo, dependiendo del tipo de atributo.</p> <ul style="list-style-type: none"> • Entero facultativo o Cadena de octetos. Un Entero será interpretado como un índice en una Paleta; una Cadena de octetos será interpretada como un valor de color directo. • Valor por defecto: «transparente».
Color de texto (TextColour)	<p>Color por defecto que se ha de utilizar para reproducir el primer plano de un objeto texto. Este atributo se interpreta como un índice basado en cero en la tabla de colores definida por el atributo <i>Referencia de paleta</i> o como un valor de color directo, dependiendo del tipo de atributo.</p> <ul style="list-style-type: none"> • Entero facultativo o Cadena de octetos. Un Entero será interpretado como un índice en una Paleta; una Cadena de octetos será interpretada como un valor de color directo. • Valor por defecto: Cualquier color.

Tipo de carácter (Font)	<p>Tipo de carácter por defecto que se ha de utilizar cuando se presenta un objeto Texto.</p> <p>El atributo <i>Tipo de carácter</i> representa un nombre para un tipo de carácter (que reside en el motor MHEG-5) o una referencia a un objeto Tipo de carácter.</p> <p>Cuando no se ha codificado ninguna referencia de tipo de carácter en la Aplicación, el objeto Texto se presenta utilizando un tipo de carácter por defecto del motor MHEG-5.</p> <ul style="list-style-type: none"> • Atributo facultativo. • Cadena de octetos que representa un nombre de Tipo de carácter, o una referencia a un objeto Tipo de carácter. • Valor por defecto: Tipo de carácter por defecto.
Atributos de tipo de carácter (FontAttributes)	<p>Atributos de Tipo de carácter por defecto, tales como estilo, tamaño de carácter, color de texto y color de fondo.</p> <p>El formato de codificación exacto del atributo <i>Atributos de tipo de carácter</i> se relaciona con el valor del Tipo de objeto tipo de carácter mencionado por el atributo <i>Tipo de carácter</i>.</p> <ul style="list-style-type: none"> • Cadena de octetos facultativa. • Valor por defecto: Ningún conjunto de atributos específico.
Gancho de contenido de diagrama de bits (BitmapContent Hook)	<p>Valor de Gancho por defecto para todos los objetos Diagrama de bits.</p> <ul style="list-style-type: none"> • Entero facultativo. • Valor por defecto: Ninguno.
Gancho de contenido de tren (StreamContentHook)	<p>Valor de Gancho por defecto para todos los objetos Tren.</p> <ul style="list-style-type: none"> • Entero facultativo. • Valor por defecto: Ninguno.
Gancho de contenido de texto (TextContentHook)	<p>Valor de Gancho por defecto para todos los objetos Texto.</p> <ul style="list-style-type: none"> • Entero facultativo. • Valor por defecto: Ninguno.
Gancho de contenido de arte lineal (LineArtContentHook)	<p>Valor de Gancho por defecto para todos los objetos Arte lineal.</p> <ul style="list-style-type: none"> • Entero facultativo. • Valor por defecto: Ninguno.
Gancho de contenido de programas intercambiados (Interchanged Program ContentHook)	<p>Valor de Gancho por defecto para todos los objetos Programas intercambiados.</p> <ul style="list-style-type: none"> • Entero facultativo. • Valor por defecto: Ninguno.

Color de referencia de botón (ButtonRefColour)	<p>Color de botón por defecto para presentar el Botón.</p> <ul style="list-style-type: none"> Entero facultativo o Cadena de octetos. Valor por defecto: Ninguno.
Color de referencia de realce (HighlightRefColour)	<p>Color de referencia de realce por defecto para objetos Interactuables.</p> <ul style="list-style-type: none"> Entero facultativo o Cadena de octetos. Valor por defecto: Ninguno.
Color de referencia de deslizador (SliderRefColour)	<p>Color de deslizador por defecto para presentar el Deslizador.</p> <ul style="list-style-type: none"> Entero facultativo o Cadena de octetos. Valor por defecto: Ninguno.

10.1.3 Atributos internos propios

Esta clase define los siguientes atributos internos adicionales:

<i>Cómputo de bloqueo</i> (LockCount)	<p>Especificar si la pantalla de visualización está o no en un estado congelado.</p> <p>Cuando este atributo es positivo y diferente de cero, la pantalla no reflejará ningún cambio hecho a objetos visibles que normalmente resultaría en un cambio de su presentación. Sin embargo, todos los cambios se reflejarán enseguida tan pronto como la pantalla sea descongelada. Esto último es señalizado poniendo el atributo <i>Cómputo de bloqueo</i> a 0.</p> <p>Los Objetos audio (de un Múltiplex de trenes) continuarán siendo difundidos a través de una acción Bloquear pantalla, es decir, continuarán siendo oídos.</p> <p>Los Objetos visibles que forman parte de un Múltiplex de trenes continuarán siendo presentados, pero cualesquiera otros cambios de estos objetos (por ejemplo, posición, volumen, etc.) no se reflejarán hasta que la pantalla sea descongelada; continuar su difusión significa que continuarán moviendo su atributo <i>Posición de contador</i> cuando la pantalla está bloqueada, pero que la actualización de la presentación gráfica de estos objetos mientras la pantalla está bloqueada es facultativa. En algunos motores su presentación física continúa funcionando, en otros, la imagen se detiene hasta que la pantalla es desbloqueada.</p> <p>Cuando la pantalla es desbloqueada, la presentación de los Objetos visibles se conformará al valor del atributo <i>Posición de contador</i>.</p> <ul style="list-style-type: none"> Entero mayor o igual a cero. Valor inicial cero.
<i>Pila de visualización</i> (DisplayStack)	<p>Lista ordenada de referencias a Objetos visibles que indican cómo se organizan los Objetos visibles de la aplicación en capas gráficas.</p> <p>Los Objetos visibles en la parte inferior de la <i>Pila de visualización</i> se presentan en el fondo de la pantalla y los que están en la parte superior de la <i>Pila de visualización</i> se presentan en el primer plano de la pantalla.</p>

Obsérvese que la *Pila de visualización* puede contener referencias a Objetos visibles inactivos. En este caso, los Objetos visibles inactivos simplemente no aparecen en la pantalla, pero permanecen como elementos válidos en la *Pila de visualización*.

- Lista ordenada de referencias de objetos a Objetos visibles.
- Valor inicial: Lista vacía.

10.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica. Además, se define el siguiente evento:

Evento de motor (EngineEvent) Este evento es generado cuando se produce un evento particular en el entorno del motor MHEG-5. Esta Recomendación no especifica ninguno de estos eventos; el dominio de aplicación puede especificar la semántica de cada uno de estos eventos y el valor de sus datos asociados.

- Datos asociados: Rótulo de evento – Entero (EventTag – Integer).

10.3 Comportamientos internos

La semántica del siguiente comportamiento interno ha cambiado con respecto a la clase básica de este objeto:

Desactivación Ejecutar la siguiente secuencia de acciones:

- 1) Aplicar la acción Cerrar conexión a todas las conexiones auxiliares abiertas.
- 2) Aplicar el comportamiento *Desactivación* heredado de la clase básica.

10.4 Efecto de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se definen las siguientes acciones MHEG-5 aplicables:

Almacenamiento persistente (Almacenamiento conseguido, Variables de entrada, Nombre de fichero de salida) Pedir al motor MHEG-5 que conserve los datos de manera que puedan ser extraídos ulteriormente por la acción Lectura persistente.

Los datos que se han de conservar se pasan a través de un conjunto de variables referenciadas por el parámetro *Variables de entrada*. Estas variables pueden contener Booleanos, Enteros, Cadenas de octetos, Referencias de objeto y Referencias de contenido.

Los datos se conservan en una estructura de datos de fichero. El *nombre de fichero de salida* es otro parámetro de la acción Almacenamiento persistente. Esta Recomendación no define la naturaleza, estructura, propiedad, protección ni restricción del espacio de ficheros. Sin embargo, las acciones Almacenamiento persistente y Lectura persistente utilizarán el mismo espacio de nombre de fichero.

El efecto de la acción Almacenamiento persistente es síncrono. Al completarse satisfactoriamente la acción Almacenamiento persistente, la Variable referenciada por *Almacenamiento conseguido* se pondrá a Verdadero; en los demás casos, se pondrá a Falso.

Ejemplo: Considérese la siguiente acción elemental incluida como parte del atributo Cierre de ramificación permanente de una Aplicación:

```
:StorePersistent (("myApp" 0) ("myApp" 1) ("scene1" 1) ("scene1" 2))
"myfile.txt") ("myApp" 0)
```

es el Identificador de objeto de la Aplicación vigente. ("scene1" 1) y ("scene1" 2) son Identificadores de objeto de variables que mantienen información relativa a la Aplicación vigente, por ejemplo, información de usuario. Estos datos serán almacenados en el fichero denominado "myfile.txt". La variable ("myApp" 1) hace referencia a una variable Booleana que indica si se consiguió o no la acción elemental.

Los datos se pueden recuperar al comienzo o re arranque de esta Aplicación (o de otra) para evitar pedir al usuario la misma información dos veces.

Disposiciones de uso:

- El objeto *Objetivo* será el objeto de Aplicación activo.
- *Almacenamiento conseguido* hará referencia a un objeto Variable Booleana activo.
- *Variables de entrada* se pondrá a una lista no vacía de referencias a Objetos variables activos de cualquier tipo.

Descripción de sintaxis:

StorePersistent	-->	Target ,
		StoreSucceeded ,
		InVariables ,
		OutFileName
Target	-->	GenericObjectReference
StoreSucceeded	-->	ObjectReference
InVariables	-->	ObjectReference+
OutFileName	-->	GenericOctetString

Lectura persistente
(*Lectura conseguida, Variables de salida, Nombre de fichero de entrada*)

[ReadPersistent
(ReadSucceeded,
OutVariables,
InFileName)]

Pedir al motor MHEG-5 que lea datos que han sido conservados por la acción Almacenamiento persistente.

Los datos que se han de leer se recuperan a través de un conjunto de objetos Variables, referenciados por el parámetros *Variables de salida*.

Los datos han sido almacenados en una estructura de datos de fichero. El *Nombre de fichero de entrada* es otro parámetro de la acción Lectura persistente. Esta Recomendación no define la naturaleza, estructura, propiedad, protección ni restricción del espacio de ficheros. Sin embargo, las acciones Almacenamiento persistente y Lectura persistente utilizarán el mismo espacio de nombre de fichero.

El efecto de la acción Lectura persistente es síncrono. Al completarse satisfactoriamente la acción Lectura persistente, la variable referenciada por *Lectura conseguida* se pondrá a Verdadero; en los demás casos, se pondrá a Falso.

Ejemplo: Considérese la siguiente acción elemental siguiente como parte del atributo *Rearranque* de una Aplicación:

```
:ReadPersistent (("myApp" 0) ("myApp" 1) (("scene1" 1) ("scene1" 2))
"myfile.txt")
```

El contenido de las variables indicadas por ("scene1" 1) y ("scene1" 2) se pondrá a los valores lectura del fichero "myfile.txt" cuando se lanza la Aplicación.

Disposiciones de uso:

- El objeto *Objetivo* será el objeto Aplicación activo.
- *Lectura conseguida* hará referencia a un objeto Variable booleana.
- *Variables de salida* se pondrán a una lista no vacía de referencias a objetos variables activos.

Descripción de sintaxis:

ReadPersistent	-->	Target , ReadSucceeded , OutVariables , InFileName
Target	-->	GenericObjectReference
ReadSucceeded ,	-->	ObjectReference
OutVariables	-->	ObjectReference+
InFileName	-->	GenericOctetString

Lanzar (Launch)

Activar una nueva aplicación suprimiendo la que está activa en ese momento, si hubiere alguna.

Ejecutar síncronamente la siguiente secuencia de acciones:

- 1) Aplicar el comportamiento *Destrucción* del objeto Escena activo actualmente, si lo hubiere.
- 2) Aplicar el comportamiento *Destrucción* del objeto Aplicación activo actualmente, si lo hubiere.
- 3) Aplicar el comportamiento *Activación* del objeto aplicación al cual se ha dirigido la acción Lanzar.

NOTA – Cualesquiera eventos generados durante la ejecución de estos pasos se ponen en cola y son tratados solamente después que toda la secuencia ha terminado.

Disposición de uso:

- El objeto *Objetivo* será un objeto Aplicación no disponible.

Descripción de sintaxis:

Launch	-->	Target
Target	-->	GenericObjectReference

Ramificación
permanente
(Spawn)

Activar una nueva aplicación de manera que la aplicación vigente sea rearrancada cuando la nueva aplicación cesa.

Ejecutar sincronamente la siguiente secuencia de acciones:

- 1) Ejecutar la Acción *Cierre de ramificación permanente* del objeto Aplicación actualmente activo.
- 2) Almacenar el *Identificador de grupo* de la Aplicación actualmente activa en la pila de identificadores de aplicaciones, si la hubiere.
- 3) Ejecutar el efecto de la acción Lanzar descrita anteriormente.

La pila de identificadores de aplicaciones es una característica facultativa del motor MHEG-5. Si no existe, o si la pila de identificadores de aplicaciones está llena, esta acción se ejecutará como la acción Lanzar.

Disposición de uso:

- El objeto *Objetivo* será un objeto Aplicación que no que no está actualmente activo mientras un objeto Aplicación está activo en ese momento.

Descripción de sintaxis:

Spawn	-->	Target
Target	-->	GenericObjectReference

Cesar (Quit)

Cerrar una aplicación y rearrancar la aplicación anterior.

Ejecutar sincronamente la siguiente secuencia de acciones:

- 1) Aplicar el comportamiento *Destrucción* del objeto Escena activo en ese momento, si hubiere alguno.
- 2) Aplicar el comportamiento *Destrucción* del objeto Aplicación objetivo.
- 3) Si el motor MHEG-5 no ha implementado una pila de identificadores de aplicaciones, o si esta pila está vacía, el motor MHEG-5 volverá al estado reposo. En todos los demás casos, se ejecutarán los siguientes pasos:
- 4) Aplicar el comportamiento *Activación* al objeto Aplicación cuyo Identificador de grupo está en la cima de la pila de identificadores de aplicaciones. Obsérvese que esto incluye la ejecución de la Acción *Arrancar* de ese objeto.
- 5) Suprimir la entrada que está en la parte superior de la pila de identificadores de la aplicación.
- 6) Ejecutar la Acción *Rearrancar* del objeto Aplicación nuevamente activado.

Disposición:

- El objeto *Objetivo* será el objeto aplicación actualmente activo.

Descripción de sintaxis:

Quit	-->	Target
Target	-->	GenericObjectReference

Bloquear pantalla (LockScreen) Congelar la pantalla de visualización y evitar que refleje cambios de objetos visibles.

Ejecutar síncronamente la siguiente secuencia de acciones:

- 1) Incrementar el atributo interno *Cómputo de bloqueo* en 1.
- 2) Si el atributo *Cómputo de bloqueo* tiene ahora un valor estrictamente positivo, bloquear la pantalla de visualización.

Disposiciones de uso:

- El objeto *Objetivo* será el objeto Aplicación activo.

Descripción de sintaxis:

LockScreen	-->	Target
Target	-->	GenericObjectReference

Desbloquear pantalla (UnlockScreen) Esta acción puede refrescar la pantalla de visualización y reflejar enseguida todos los cambios de objetos visibles.

Ejecutar síncronamente la siguiente secuencia de acciones:

- 1) Disminuir el atributo interno *Cómputo de bloqueo* en 1. Si el resultado es inferior a cero, fijar el atributo *Cómputo de bloqueo* a 0.
- 2) Si el atributo *Cómputo de bloqueo* es igual a cero, refrescar la pantalla de visualización.

Disposición de uso:

- El objeto *Objetivo* será el objeto Aplicación activo.

Descripción de sintaxis:

UnlockScreen	-->	Target
Target	-->	GenericObjectReference

Abrir conexión (Apertura conseguida, Protocolo, Dirección, Rótulo de conexión) Intentar abrir una conexión con una entidad fuera del motor MHEG-5.

La acción Abrir conexión tiene los siguientes parámetros:

<i>Apertura conseguida</i>	Si la acción Abrir conexión termina satisfactoriamente, la Variable referenciada por <i>Apertura conseguida</i> se pondrá a Verdadero; en los demás casos se pondrá a Falso.
<i>Protocolo</i>	Identificador del protocolo que se ha de utilizar al establecer la conexión.
<i>Dirección</i>	Dirección de la contraparte con la cual se debe establecer la conexión. La codificación de este parámetro depende del valor de <i>Protocolo</i> .
<i>Rótulo de conexión</i>	Entero utilizado para hacer referencia a la conexión dentro de la aplicación.

Disposiciones de uso:

- El objeto *Objetivo* será el objeto Aplicación activo.
- *Apertura conseguida* hará referencia a un objeto Variable Booleana.

Descripción de sintaxis:

OpenConnection	-->	Target , OpenSucceeded , Protocol , Address , ConnectionTag
Target	-->	GenericObjectReference
OpenSucceeded	-->	ObjectReference
Protocol	-->	GenericOctetString
Address	-->	GenericOctetString
ConnectionTag	-->	GenericInteger

Cerrar conexión
(*Rótulo de
conexión*)

[CloseConnection
(ConnectionTag)]

Intentar cerrar una conexión con una entidad fuera del motor MHEG-5.

La acción Cerrar conexión tiene los siguientes parámetros:

Rótulo de conexión Entero que hace referencia a una conexión creada por la acción Abrir conexión.

Disposiciones de uso:

- El objeto *Objetivo* será el objeto Aplicación activo.
- Si la conexión referenciada por *Rótulo de conexión* no se ha establecido adecuadamente, se omite la acción Cerrar conexión.

Descripción de sintaxis:

CloseConnection	-->	Target , ConnectionTag
Target	-->	GenericObjectReference
ConnectionTag	-->	GenericInteger

Obtener soporte
de motor
(*Característica,
Respuesta*)

[GetEngineSupport
(Feature, Answer)]

Devolver una Variable Booleana que indica si el motor MHEG-5 ejecuta la opción específica o conjunto de opciones de la presente Recomendación. El resultado de esta acción se devuelve en una Variable Booleana referenciada por el parámetro *Respuesta* y puede ser utilizado para adaptar el comportamiento de la aplicación a las capacidades del motor.

Característica es una cadena codificada según ISO/CEI 646 y describe la opción o conjunto de opciones. A continuación se definen las cadenas autorizadas; el dominio de aplicación puede definir cadenas adicionales. Estas cadenas son sensibles a mayúsculas y minúsculas y los símbolos, N, W, H, X o Y entre paréntesis han de ser sustituidos por enteros.

La respuesta a cada una de estas cadenas será verdadero o falso.

– Conexiones auxiliares

Pregunta si el motor admite conexiones punto a punto auxiliares. Estas conexiones tratan las acciones Abrir conexión y Cerrar conexión y el atributo *Rótulo de conexión* en diversas acciones elementales.

- Pila de aplicaciones
Pregunta si el motor admite la acción Ramificación permanente de la clase Aplicación.
- Clonación
Pregunta si el motor admite la acción Clonar.
- Cursor de movimiento libre
Pregunta si el motor admite la clase Forma de cursor, para los eventos *Entrada de cursor* y *Salida de cursor*, y para las acciones Obtener posición de cursor, Fijar posición de cursor y Fijar forma de cursor.
- Múltiples trenes de audio (N)
Pregunta si el motor admite por lo menos N trenes de Audio simultáneos.
- Múltiples trenes de gráficos RT (N)
Pregunta si el motor admite por lo menos N trenes de gráficos RT simultáneos.
- Múltiples trenes de vídeo (N)
Pregunta si el motor admite por lo menos N trenes de Vídeo simultáneos.
- Visibles superpuestos (N)
pregunta si el motor admite por lo menos N objetos Visibles superpuestos.
- Escala
Pregunta si el motor admite las acciones Escala de diagrama de bits y Escala de vídeo.
- Formato de escena (W,H)
Pregunta si el motor admite un sistema de coordenadas dado. W y H son dos enteros, W/H es la relación de formato anchura/altura.
- Sistema de coordenadas de escena (X,Y)
Pregunta si el motor admite un sistema de coordenadas dado. X e Y son dos enteros que definen el sistema de coordenadas.
- Modos truco
Pregunta si el motor admite modos truco para Trenes.¹

Disposiciones de uso:

- La acción Obtener soporte de motor será dirigida solamente al objeto aplicación activo.
- *Respuesta* hará referencia a un objeto Variable Booleana activo.

Descripción de sintaxis:

GetEngineSupport	-->	Target ,
	-->	Feature ,
		Answer
Target	-->	GenericObjectReference
Feature	-->	GenericOctetString
Answer	-->	ObjectReference

10.5 Descripción formal

Application Class	-->	Group Class ,
		OnSpawnCloseDown? ,
		OnRestart?
		DefaultAttributes?
OnSpawnCloseDown	-->	Action Class
OnRestart	-->	Action Class
DefaultAttributes	-->	DefaultAttribute+
DefaultAttribute	-->	CharacterSet BackgroundColour
		TextColour Font FontAttributes
		BitmapContentHook
		InterchangedProgramContentHook
		StreamContentHook
		TextContentHook LineArtContentHook
		ButtonRefColour HighlightRefColour
		SliderRefColour
CharacterSet	-->	INTEGER
BackgroundColour	-->	Colour
TextColour	-->	Colour
Font	-->	OctetString ObjectReference
FontAttributes	-->	OctetString
BitmapContentHook	-->	INTEGER
StreamContentHook	-->	INTEGER
TextContentHook	-->	INTEGER
LineArtContentHook	-->	INTEGER
InterchangedProgram-	-->	INTEGER
ContentHook		
ButtonRefColour	-->	Colour
HighlightRefColour	-->	Colour
SliderRefColour	-->	Colour

11 Clase Escena

Descripción: Define un conjunto de objetos Ingredientes que han de ser activados juntos

Clase básica: Grupo

Subclases: Ninguna

Estados: Clase concreta

11.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

11.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con semántica idéntica.

11.1.2 Atributos intercambiados propios

Esta clase define los siguientes atributos intercambiados adicionales:

Registro de eventos de entrada (InputEventRegister)	<p>Registro de eventos de entrada de usuario admisibles para esta Escena.</p> <p>Mientras esta escena está activa, el motor MHEG-5 generará solamente eventos de entrada de usuario que tengan datos asociados coherentes con el contenido de este registro de eventos de entrada.</p> <p>En la presente Recomendación no se define el contenido de los Registros de eventos de entrada.</p> <ul style="list-style-type: none">• Entero que identifica a un Registro de eventos de entrada. <p>Ejemplo: Sería posible definir dos Registros de eventos de entrada: uno dedicado a eventos de entrada de punteros en movimiento (Clickeo del ratón, etc.) y otro dedicado a eventos de entrada de telecontrol (Hacia arriba, Hacia abajo, Hacia la izquierda, Hacia la derecha, Entrada, Salida, etc.). El conocimiento del tipo de eventos de entrada que son previstos por la escena permitirá al motor MHEG-5 utilizar los dispositivos de usuario físicos para generar tales eventos.</p>
Sistema de coordenadas de escena (SceneCoordinate-System)	<p>Tamaño del sistema de coordenadas de esta Escena.</p> <p>Este atributo se expresa en número de filas y columnas.</p> <ul style="list-style-type: none">• Dos enteros, que expresan los tamaños de escena x y de escena y.
Formato de imagen (AspectRatio)	<p>Formato original de la Escena. Este atributo se expresa mediante una relación anchura/altura.</p> <ul style="list-style-type: none">• Número racional facultativo.• Valor por defecto: 4/3.
Cursor en movimiento (MovingCursor)	<p>Indica si la Escena prevé un cursor de movimiento libre.</p> <p>La admisión por el motor MHEG-5 de cursores de movimiento libre es facultativa. Sin embargo, un dominio de aplicación de MHEG-5 puede declarar que sea obligatorio. Un motor que no admite cursores de movimiento libre omitirá este atributo. Un motor que sí los admite, actuará de la siguiente manera:</p> <ul style="list-style-type: none">• Cuando este atributo es Falso, el motor no visualizará un cursor en la pantalla.• Cuando este atributo es Verdadero, el motor visualizará un cursor en la pantalla. El usuario podrá mover este cursor a todas las posiciones dentro del espacio de coordenadas de la Escena. Cuando el cursor entra en (o sale de) la casilla delimitante de un objeto interactuable, se generará un evento Entrada de cursor (Salida de cursor) para ese objeto Interactable. El motor admitirá las acciones Fijar posición de cursor y Obtener posición de cursor.

En general, se aplica lo siguiente a este atributo:

- Booleano facultativo.
- Valor por defecto: Falso.

Escenas siguientes (NextScenes) Una lista facultativa de Cadenas de octetos, que será interpretada como Identificadores de grupo de objetos Escenas que pudieran ser presentadas después de ésta, junto con el factor de ponderación que mide la probabilidad de que estas escenas sean presentadas realmente. El factor de ponderación será un Entero en la gama [0, 255], en la que 255 indica la probabilidad más alta. Puede ser utilizado por el motor MHEG-5 para resolver conflictos de ocultación o carga previa.

11.1.3 Atributos internos propios

Esta clase define el siguiente atributo interno adicional:

Temporizadores (Timers) Lista de temporizadores que representan las posiciones temporales en las que la Escena recibirá eventos *Temporizador activado*.

Cada temporizador tiene un número de identidad único dentro de la lista *Temporizadores* y una posición temporal expresada en milisegundos. La posición temporal se mide a partir de la base de tiempo del temporizador. El temporizador será creado ejecutando la acción Fijar temporizador en la Escena activa. La base de tiempo del temporizador es por defecto la posición temporal donde se ejecuta la acción Fijar temporizador que ha creado este temporizador; sin embargo, si se codifica la variable booleana Tiempo absoluto y se pone a verdadero, la base de tiempo del temporizador es la posición temporal en la que se genera el evento en *está funcionando* de la escena.

- Secuencia de las siguientes estructuras de datos:
 - Identificador de temporizador: Entero.
 - Posición de temporizador: Entero.
 - Tiempo absoluto: Booleano facultativo, por defecto: Falso.
- Valor inicial: Secuencia vacía.

11.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica. Además, se definen los siguientes eventos:

Entrada de usuario (UserInput) Este evento será generado por el motor MHEG-5 para indicar que se ha producido una introducción de usuario.

- Datos asociados: Rótulo de evento de entrada de usuario – Entero. El valor de los datos asociados concordará con el contenido del atributo *Registro de eventos de entrada*.

Temporizador activado (TimerFired) Este evento es generado cuando se ha activado un temporizador.

- Datos asociados: Identificador de temporizador – Entero.

11.3 Comportamientos internos

Esta clase tiene los mismos comportamientos internos que su clase básica, con semántica idéntica.

11.4 Efecto de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se definen las siguientes acciones MHEG-5 aplicables:

Transición a
(*Rótulo de conexión, Efecto de transición*)

Verificar que la escena del objetivo es diferente de la escena activa; si no lo es, omitir la acción; si lo es, suprimir la Escena activa de la pantalla y sustituirla con la escena objetivo, como sigue.

Ejecutar sincronamente la siguiente secuencia de acciones:

[TransitionTo
(*ConnectionTag, TransitionEffect*)]

- 1) Aplicar el comportamiento *Desactivación* a todos los objetos Ingredientes activos del objeto Aplicación actualmente activo que tiene el parámetro Compartido puesto a Falso en el orden inverso al que están enumerados en el atributo Ítems del objeto aplicación.
- 2) Aplicar los comportamientos *Desactivación* y *Dstrucción* al objeto Escena actualmente activo, si lo hubiere. (Esto inicia el efecto de transición.)
- 3) Aplicar el comportamiento *Preparación* al objeto Escena al cual se ha dirigido la acción Transición a.
- 4) Aplicar el comportamiento *Activación* al objeto Escena al cual se ha dirigido la acción Transición a. (Esto detiene el efecto de transición.) El límite de tiempo para la nueva escena comienza después de la generación del evento *Está funcionando*.

Esta acción tiene un parámetro facultativo denominado *Rótulo de conexión*. Si este parámetro no está codificado, la referencia a la Escena se resolverá dentro del espacio de nombre del objeto Aplicación activo.

Si el parámetro *Rótulo de conexión* está codificado, la referencia a la Escena objetivo y todas las Referencias de contenido hechas a partir de la Escena objetivo serán resueltas dentro de un espacio de nombre que es utilizado para la comunicación por un enlace de comunicación con el rótulo *Rótulo de conexión* (véase la acción Abrir conexión en la clase Aplicación).

Además, esta acción tiene un parámetro *Efecto de transición* que determina el tipo de efecto de transición visual que se ha de aplicar cuando se ejecuta la acción Transición a. La ejecución de cualquier efecto de transición es facultativa para el motor MHEG-5. La codificación del atributo efecto de transición tiene que ser especificada por el dominio de aplicación.

NOTA – Para algunos efectos de transición, el motor puede tener que tratar el comportamiento *Dstrucción* de objetos visibles de manera diferente con respecto a otra desactivación (no introducida por Transición a). Por ejemplo, la representación visual de la escena actual puede ser conservada en el subsistema básico para un efecto de empuje, cuando la primera escena es sustituida por la segunda.

Disposición de uso:

- El objeto Objetivo será un objeto Escena no disponible.

Descripción de sintaxis:

TransitionTo	-->	Target , ConnectionTag? , TransitionEffect?
Target	-->	GenericObjectReference
ConnectionTag	-->	GenericInteger
TransitionEffect	-->	GenericInteger

Fijar temporizador
(*Id de temporizador*,
Valor de temporizador,
Tiempo absoluto)
[SetTimer
(*TimerId*,
TimerValue,
AbsoluteTime)]

Actualizar la lista de temporizadores de la Escena.

Ejecutar la siguiente secuencia de acciones:

- 1) Actualizar el atributo interno *Temporizadores* de la Escena, de acuerdo con la siguiente regla:
 - a) Si el *Id de temporizador* es el identificador de un Temporizador existente en la Escena, el nuevo *Valor de temporizador* sustituye al anterior. El parámetro *Tiempo absoluto* se omite; en otras palabras, un Temporizador absoluto no puede ser sustituido por un Temporizador relativo a la Escena.
 - b) Si no hay un Temporizador con identificador *Id de temporizador* en la Escena, insertar un nuevo Temporizador con identificador *Id de temporizador* y valores *Valor de temporizador* y *Tiempo absoluto* en la Escena. Si *Tiempo absoluto* no está codificado, se pone a Falso por defecto.
 - c) Si el *Valor de temporizador* no está codificado y hay un Temporizador con identificador *Id de temporizador* en la Escena, suprimir este Temporizador de la lista *Temporizadores*.
 - d) Si *Valor de temporizador* no está codificado y no hay Temporizador con identificador *Id de temporizador* en la lista *Temporizadores*, descartar esta acción.
- 2) La Escena activa recibirá eventos *Temporizador activado* de acuerdo con el nuevo valor de la lista *Temporizadores*.

Si *Tiempo absoluto* se pone a Verdadero, el parámetro *Valor de temporizador* de esta acción será interpretado como un desplazamiento temporal con respecto al tiempo cuando la Escena está activa. En los demás casos, el parámetro *Valor de temporizador* de esta acción será interpretado como un desplazamiento temporal con respecto al tiempo cuando la acción es invocada. En ambos casos, se mide en milisegundos.

Si el parámetro *Valor de temporizador* es cero y *Tiempo absoluto* es Falso, el Temporizador será activado inmediatamente.

La supresión o el cambio de un Temporizador no suprime los eventos pendientes del Temporizador anterior.

Disposición de uso:

- El objeto *Objetivo* será el objeto Escena activo.

Descripción de sintaxis:

SetTimer	-->	Target , TimerId , TimerValue? , AbsoluteTime?
Target	-->	GenericObjectReference
TimerId	-->	GenericInteger
TimerValue	-->	GenericInteger
AbsoluteTime	-->	GenericBoolean

Enviar evento
(Origen de evento
emulado, Tipo de
evento emulado,
Datos de evento
emulados)

[SendEvent
(EmulatedEvent-
Source, Emulated
Event-Type,
Emulated
Event-Data)]

Forzar la ocurrencia de un evento.

Ejecutar la siguiente secuencia de acciones:

- 1) Generar un evento que corresponde a *Tipo de evento emulado*, *Fuente de evento emulado*, y *Datos de evento emulado* como si fuese generado en la forma normal.
- 2) Almacenar este evento en la cola de eventos síncronos o asíncronos, de acuerdo con *Tipo de evento emulado*.

Disposiciones de uso:

- El objeto *Objetivo* será el objeto Escena activo.
- *Fuente de evento emulado* hará referencia a un objeto MHEG-5 compatible con *tipo de evento emulado*.
- *Datos de evento emulado* será un valor directo, o una referencia a un objeto Variable activo de un tipo compatible con los datos asociados de *tipo de evento emulado*.

Descripción de sintaxis:

SendEvent	-->	Target , EmulatedEventSource , EmulatedEventType , EmulatedEventData?
Target	-->	GenericObjectReference
EmulatedEventSource	-->	GenericObjectReference
EmulatedEventType	-->	IsAvailable ContentAvailable IsDeleted IsRunning IsStopped UserInput AnchorFired TimerFired AsynchStopped InteractionCompleted TestEvent TokenMovedFrom TokenMovedTo FirstItemPresented LastItemPresented HeadItems TailItems ItemSelected ItemDeselected StreamEvent StreamPlaying StreamStopped CounterTrigger HighlightOn HighlightOff CursorEnter CursorLeave IsSelected IsDeselected EntryFieldFull
EmulatedEventData	-->	GenericBoolean GenericInteger GenericOctetString

Fijar forma de cursor
(*Nueva forma de cursor*)

[SetCursorShape
(*NewCursorShape*)]

Fijar la forma del cursor de movimiento libre.

Esta acción tendrá un efecto solamente si la opción de cursor de movimiento libre es aplicada por el motor MHEG-5.

Si el parámetro *Nueva forma de cursor* no está codificado, se suprime el cursor de la Escena.

Disposiciones de uso:

- El objeto *Objetivo* será el objeto Escena activo.
- *Nueva forma de cursor* hará referencia a un objeto forma de cursor activo.

Descripción de sintaxis:

SetCursorShape	-->	Target , NewCursorShape?
Target	-->	GenericObjectReference
NewCursorShape	-->	GenericObjectReference

Fijar posición de cursor
(*Cursor X, Cursor Y*)

[SetCursorPosition
(*XCursor, YCursor*)]

Fijar la posición del cursor de movimiento libre.

Esta acción tendrá un efecto solamente si la opción de cursor de movimiento libre es aplicada por el motor MHEG-5.

Ejecutar la siguiente secuencia de acciones:

- 1) Fijar la posición del puntero del cursor dentro del espacio de coordenadas de la Escena.
- 2) Generar los eventos *Salida de cursor* y *Entrada de cursor* si los objetos Interactuables son afectados por esta acción.

Si un objeto Interactable B se superpone con otro A, una acción Fijar posición de cursor desde un punto en A (no en B) a un punto dentro de la zona de superposición generará un evento *Salida de cursor (A)* y una acción *Entrada de cursor (B)*.

Disposiciones de uso:

- El objeto *Objetivo* será el objeto Escena activo.
- *Cursor X* y *Cursor Y* corresponderán a una posición dentro del rectángulo definido por el atributo sistema de coordenadas de escena de la Escena activa.

Descripción de sintaxis:

SetCursorPosition	-->	Target , XCursor , YCursor
Target	-->	GenericObjectReference
Xcursor	-->	GenericInteger
Ycursor	-->	GenericInteger

Obtener posición de cursor
(*Salida X, salida Y*)
[GetCursorPosition
(*XOut,YOut*)]

Fijar las Variables referenciadas por *Salida X* y *Salida Y* a la posición del cursor de movimiento libre dentro del espacio de coordenadas de la Escena.

Esta acción tendrá un efecto solamente si la opción cursor de movimiento libre es aplicada por el motor MHEG-5.

Disposiciones de uso:

- El objeto *Objetivo* será el objeto Escena activo.
- *Salida X* y *Salida Y* harán referencia a objetos Variables enteras activos.

Descripción de sintaxis:

GetCursorPosition	-->	Target , XOut , YOut
Target	-->	GenericObjectReference
XOut	-->	ObjectReference
YOut	-->	ObjectReference

11.5 Descripción formal

Scene Class	-->	Group Class , InputEventRegister , SceneCoordinateSystem , AspectRatio? , MovingCursor? , NextScenes?
InputEventRegister	-->	INTEGER
SceneCoordinateSystem	-->	XScene , YScene
XScene ,	-->	INTEGER
Yscene	-->	INTEGER
AspectRatio	-->	Width , Height
Width	-->	INTEGER
Height	-->	INTEGER
MovingCursor	-->	BOOLEAN
NextScenes	-->	NextScene+
NextScene	-->	SceneRef , SceneWeight
SceneRef	-->	OctetString
SceneWeight	-->	INTEGER

12 Clase Ingrediente

Descripción: Define la funcionalidad asociada con clases que componen Escenas y Aplicaciones.

Clase básica: Raíz

Subclases: Enlace, Programa, Paleta, Tipo de caracteres, Forma de cursor, Variable, Presentable

Estado: Clase abstracta

12.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

12.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las siguientes restricciones:

Nombre de atributo	Definido en	Restricciones y requisitos
Identificador de objeto	Raíz	La parte de Número de objeto de este atributo será única dentro del grupo al que pertenece este objeto y no será cero. Si el Identificador de grupo está codificado, se pondrá al Identificador del grupo al que pertenece este objeto.

12.1.2 Atributos intercambiados propios

Esta clase define los siguientes atributos intercambiados adicionales:

Activo inicialmente (InitiallyActive) Este parámetro se utiliza para determinar los objetos en una Escena o en una Aplicación activos inicialmente.

- Booleano facultativo.
- Valor por defecto: Verdadero.

Gancho de contenido (ContentHook) Determinar el formato de codificación de los datos incluidos o referenciados por el atributo *Contenido*.

- Entero facultativo.
- Valor por defecto: Según las subclases, el valor codificado por la aplicación en uno de los siguientes atributos: Gancho de contenido de diagrama de bits, Gancho de contenido de arte lineal, Gancho de contenido de programas intercambiados, Gancho de contenido de tren y Gancho de contenido de texto.

Contenido original (OriginalContent) Valor del atributo interno *Contenido* en preparación.

Este atributo contiene datos incluidos o una referencia a una fuente de datos externa.

- Contenido original es un atributo facultativo.
- Tipo de datos: Contenido referenciado o contenido incluido.
- Valor por defecto: Ninguno.

Los datos incluidos se codifican directamente dentro de una Cadena de octetos.

La referencia a una fuente de datos externa se compone de:

- 1) Una Referencia de contenido, que es una Cadena de octetos que hace referencia a una pieza externa de datos.
 - Tipo de datos: Cadena de octetos.
- 2) Un Tamaño de contenido facultativo, que representa el tamaño en octetos de la fuente externa. Este atributo pudiera ser utilizado por el motor MHEG-5 para determinar cuántos recursos se necesitan para presentar el ingrediente. Es responsabilidad de la aplicación asegurar la compatibilidad del atributo Tamaño de contenido con el tamaño real de la fuente de datos externa.
 - Tamaño de contenido es un atributo facultativo.
 - Tipo de datos: Entero.
 - Valor por defecto: Ninguno.

Ejemplo: Considérese un objeto Diagrama de bits cuyo atributo Contenido original se pone a una referencia a un fichero de datos externo. Su atributo Tamaño de contenido debe contener una estimación del tamaño de los datos externos.

- 3) Prioridad de cache de contenido adicional, que representa la pertinencia de ocultar esta fuente de datos externa. Este atributo pudiera ser comparado por el motor MHEG-5 a otros atributos Prioridad de cache de contenido para determinar qué fuente de datos externa tiene la más alta prioridad para ser requerida de nuevo por la aplicación. Un valor más alto indica un nivel de prioridad más alto. Es responsabilidad de la aplicación mantener estas prioridades en una gama coherente. Se recomienda al motor MHEG-5 que oculte fuentes de datos externas con prioridad más alta de preferencia a fuentes de datos con prioridad más baja.
 - Prioridad de cache de contenido es un atributo facultativo.
 - Tipo de datos: Entero dentro de la gama [0, 255].
 - Valor por defecto: 127.
 - Valor específico: 0 significa que no se autoriza la ocultación de datos de contenido externos referenciados por este ingrediente.

Compartido (Shared)

Indicar si el objeto Ingrediente está previsto para presentación continua a través de una transición de Escenas. Esto se utiliza para evitar la destrucción de objetos que se utilizan en escenas consecutivas. Específicamente, cuando una acción Transición a está dirigida a una Escena *B* mientras la Escena *A* está activa, todos los ingredientes activos del objeto Aplicación activo son desactivados automáticamente, salvo aquéllos que tienen el atributo Compartido puesto a Verdadero.

Disposición de uso:

- Si el objeto Ingrediente es un elemento de una Escena o de una Plantilla, el atributo Compartido no será codificado.

Sinopsis:

- Booleano facultativo.
- Valor por defecto: Falso.

12.1.3 Atributos internos propios

Esta clase define los siguientes atributos internos adicionales:

<i>Contenido</i>	<p>Este atributo contiene datos incluidos o una referencia a una fuente de datos externa.</p> <p>Los datos incluidos se codifican directamente dentro de una Cadena de octetos.</p> <p>Una referencia a una fuente de datos externa se compone de:</p> <ol style="list-style-type: none">1) Una Referencia de contenido, que es una Cadena de octetos que hace referencia a una pieza de datos externa.<ul style="list-style-type: none">• Tipo de datos: Cadena de octetos.• Valor inicial: Tamaño de contenido del atributo contenido original.2) Un Tamaño de contenido facultativo, que representa el tamaño en octetos de la fuente externa.<ul style="list-style-type: none">• Tamaño de contenido es un atributo facultativo.• Tipo de datos: Entero.• Valor inicial: Tamaño de contenido del atributo contenido original.3) Una Prioridad de cache de contenido facultativa, que representa la pertinencia de ocultar esta fuente de datos externa.<ul style="list-style-type: none">• Prioridad de cache de contenido es un atributo facultativo.• Tipo de datos: Entero dentro de la gama [0, 255].• Valor inicial: Prioridad de cache de contenido del atributo Contenido original.• Valor específico: 0 significa que la ocultación no está autorizada para datos de contenido externos referenciados por este Ingrediente. <p>El atributo <i>Contenido</i> no se definirá para clases que especifican que el Contenido original no será codificado.</p> <ul style="list-style-type: none">• Atributo facultativo.• Tipo de datos: Contenido referenciado o Contenido incluido.• Valor inicial: Valor del contenido original.
------------------	---

12.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

12.3 Comportamiento internos

La semántica de los siguientes comportamientos internos ha cambiado con respecto a la clase básica de este objeto:

<i>Destrucción</i>	<p>Ejecuta la siguiente secuencia de acciones:</p> <ol style="list-style-type: none">1) Si el atributo <i>Contenido</i> está puesto a una referencia a fuente de datos externa y si Prioridad de cache de contenido está puesto a 0, el motor
--------------------	---

MHEG-5 liberará todos los recursos asignados a los datos de contenido externos del Ingrediente.

- 2) Si el atributo *Contenido* está puesto a una referencia a fuente de datos externa y si Prioridad de cache de contenido es diferente de 0, el motor MHEG-5 puede liberar todos los recursos asignados a los datos de contenido externos del Ingrediente, u ocultarlos.
- 3) Aplicar el comportamiento *Destrucción* heredado de la clase Raíz.

12.4 Efecto de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se definen las siguientes acciones MHEG-5 aplicables:

Fijar datos (Nuevo contenido)	Fijar el atributo <i>Contenido</i> del Ingrediente objetivo a <i>Nuevo contenido</i> .
[SetData (NewContent)]	<p>Disposiciones de uso:</p> <ul style="list-style-type: none"> • El objeto <i>Objetivo</i> será un objeto Ingrediente disponible. • El Gancho de contenido del objeto Ingrediente objetivo estará codificado. • Los datos incluidos o referenciados por <i>Nuevo contenido</i> tendrán el formato de codificación determinado por el Gancho de contenido del Ingrediente objetivo. • Si <i>Contenido</i> está actualmente puesto a datos incluidos, <i>Nuevo contenido</i> se pondrá o hará referencia a datos incluidos. • Si <i>Contenido</i> está actualmente puesto a una referencia a una fuente de datos externas, <i>Nuevo contenido</i> se pondrá o hará referencia a una referencia a una fuente de datos externa.

Descripción de sintaxis:

SetData	-->	Target , NewContent
Target	-->	GenericObjectReference
NewContent	-->	NewIncludedContent NewReferencedContent
NewIncludedContent	-->	GenericOctetString
NewReferencedContent	-->	NewContentReference , NewContentSize? , NewContentCachePriority?
NewContentReference	-->	GenericContentReference
NewContentSize	-->	GenericInteger
NewContentCachePriority	-->	GenericInteger

Clonar
(*Variable
referencia de clon*)

Si el motor admite la opción de Clonar, el efecto de esta acción se describe a continuación. Los motores que no admiten la opción de Clonar omitirán esta acción.

[Clone
(*CloneRefVar*)]

Esta acción copia el *Objetivo* añadiendo la copia al mismo grupo que contiene el *Objetivo* que utiliza una Referencia de objeto única obtenida del motor. La Referencia de objeto que hace referencia a la copia es devuelta en la variable referencia de objeto referenciada por la *Variable referencia de clon*.

Ejecutar la siguiente secuencia de acciones:

- 1) Determinar una Referencia de objeto única dentro del mismo grupo que el *Objetivo*.
- 2) Crear una copia del *Objetivo*, teniendo en cuenta solamente los atributos intercambiados y no los atributos internos. El atributo *Identificador de objeto* heredado de la clase Raíz no se copia, pero se pone a la referencia de objeto determinada en el paso 1.
- 3) Añadir la copia al grupo que contiene el objeto *Objetivo*.
- 4) Fijar la *Variable* de referencia de objeto referenciada por *Variable referencia de clon* a la referencia de objeto determinada en el paso 1.
- 5) Aplicar el comportamiento *Preparación* a la copia.

Un objeto creado utilizando esta acción es desactivado/destruido cuando el grupo que contiene ese objeto es desactivado/destruido. Los objetos son desactivados/destruidos junto con otros ingredientes estáticos en el orden inverso de creación. Un Ingrediente creado dinámicamente puede también ser destruido utilizando la acción elemental Descargar si su atributo *Contenido* no es Nulo.

Disposiciones de uso:

- El objeto *Objetivo* será un Ingrediente disponible.
- *Variable referencia de clon* será una *Variable* de referencia de objeto activa.

Descripción de sintaxis:

Clone	-->	Target , CloneRefVar
Target	-->	GenericObjectReference
CloneRefVar	-->	ObjectReference

Precargar
(Preload)

Prepara un Ingrediente y proporciona una indicación a la máquina MHEG-5 de que prepare los datos de contenido de un Ingrediente para uso futuro.

Ejecutar la siguiente secuencia de acciones:

- 1) Aplicar el comportamiento *Preparación*.
- 2) El motor MHEG-5 puede facultativamente extraer y/o decodificar los datos de contenido asociados con el objeto Ingrediente objetivo.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Ingrediente no disponible.
- El atributo *Contenido* del Ingrediente objetivo será diferente de Nulo.

Descripción de sintaxis:

Preload	-->	Target
Target	-->	GenericObjectReference

Descargar
(Unload)

Destruye un Ingrediente y proporciona una indicación al motor MHEG-5 de que libere recursos asignados a un Ingrediente.

Ejecutar la siguiente secuencia de acciones:

- 1) Aplicar el comportamiento *Destrucción*.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Ingrediente disponible e inactivo.
- El atributo *Contenido* del Ingrediente objetivo será diferente de nulo.

Descripción de sintaxis:

Unload	-->	Target
Target	-->	GenericObjectReference

12.5 Descripción formal

Ingredient Class	-->	Root Class, InitiallyActive?, ContentHook?, OriginalContent?, Shared?
InitiallyActive	-->	BOOLEAN
ContentHook	-->	INTEGER
OriginalContent	-->	IncludedContent ReferencedContent
IncludedContent	-->	OctetString
ReferencedContent	-->	ContentReference, ContentSize?, ContentCachePriority?
ContentSize	-->	INTEGER
ContentCachePriority	-->	INTEGER
Shared	-->	BOOLEAN

13 Clase de enlace

Descripción:	Define la funcionalidad asociada con la reacción a eventos ejecutando una secuencia de acciones elementales
Clase básica:	Ingrediente
Subclases:	Ninguna
Estado:	Clase concreta

13.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

13.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las siguientes restricciones:

Nombre de atributo	Definido en	Restricciones y requisitos
Gancho de contenido	Ingrediente	Este atributo no se codificará para esta clase.
Contenido original	Ingrediente	Este atributo no se codificará para esta clase.

13.1.2 Atributos intercambiados propios

Esta clase define los siguientes atributos intercambiados adicionales:

Condición de enlace (LinkCondition)	<p><i>Condición de enlace</i> consiste en una <i>Fuente de evento</i>, un <i>Tipo de evento</i> y <i>Datos de evento</i>.</p> <p>Cuando un evento emana de un objeto, el motor MHEG-5 activará un enlace específico solamente si:</p> <ul style="list-style-type: none">• el Enlace está activo;• la Fuente del evento es igual a la referencia del objeto del cual emanó el evento (el Identificador de grupo por defecto del Identificador del grupo en el cual el propio Enlace está incluido);• el Tipo de evento es igual al tipo de evento que se ha producido; y• los Datos de eventos son iguales al valor de datos proporcionados con el evento, o los Datos de evento no están codificados. <p>NOTA – El tipo de datos pasados con cada evento se describe en la cláusula 53. La activación de un objeto Enlace conduce a la ejecución de su Efecto de enlace.</p>
Efecto de enlace (LinkEffect)	<p>Inclusión de un objeto Acción.</p> <p>Cuando el Enlace es activado, las acciones elementales dentro de este objeto Acción son ejecutadas en orden síncrono.</p>

13.1.3 Atributos internos propios

Esta clase no define atributos internos adicionales.

13.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

13.3 Comportamientos internos

La semántica de los siguientes comportamientos internos ha cambiado con respecto a la clase básica de este objeto:

- Activación* Ejecutar la siguiente secuencia de acciones:
- 1) Aplicar el comportamiento *Activación* heredado de la clase básica.
 - 2) Hacer que el objeto Enlace sea receptivo a eventos que cumplen su Condición de enlace.
 - 3) Fijar el *estado de ejecución* del Enlace a Verdadero.
 - 4) Generar un evento *Está funcionando*.
- Desactivación* Ejecutar la siguiente secuencia de acciones:
- 1) Poner al objeto Enlace en un estado inactivo de modo que no sea receptivo a eventos.
 - 2) Aplicar el comportamiento *Desactivación* heredado de la clase básica.

13.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se definen las siguientes acciones MHEG-5 aplicables:

Activar (Activate) Hacer que un Enlace sea receptivo a eventos que cumplen su Condición de enlace.

Ejecutar las siguientes acciones:

- 1) Si el objeto Enlace del objetivo está activo, omitir esta acción.
- 2) Si el Enlace del objetivo está inactivo, aplicar el comportamiento *Activación* del objeto Enlace.

Descripción de sintaxis:

Activate	-->	Target
Target	-->	GenericObjectReference

Desactivar Hacer que un Enlace no sea receptivo a eventos.

(Deactivate)

Ejecutar las siguientes acciones:

- 1) Si el objeto Enlace del objetivo está inactivo, omitir esta acción.
- 2) Si el Enlace del objetivo está activo, aplicar el comportamiento *Desactivación* del objeto Enlace.

Disposición de uso:

- El objeto *Objetivo* será un objeto Enlace disponible.

Descripción de sintaxis:

Deactivate	-->	Target
Target	-->	GenericObjectReference

13.5 Descripción formal

Link Class	-->	Ingredient Class,
		LinkCondition,
		LinkEffect
LinkCondition	-->	EventSource,
		EventType,
		EventData?
LinkEffect	-->	Action Class
EventSource	-->	ObjectReference
EventType	-->	IsAvailable ContentAvailable
		IsDeleted IsRunning IsStopped
		UserInput AnchorFired
		TimerFired AsynchStopped
		InteractionCompleted TestEvent
		TokenMovedFrom TokenMovedTo
		FirstItemPresented
		LastItemPresented HeadItems
		TailItems ItemSelected
		ItemDeselected StreamEvent
		StreamPlaying StreamStopped
		CounterTrigger HighlightOn
		HighlightOff CursorEnter
		CursorLeave IsSelected
		IsDeselected EntryFieldFull
		EngineEvent
EventData	-->	OctetString BOOLEAN INTEGER

14 Clase Programa

Descripción: Define los medios para tratar la ejecución de piezas externas de código de procedimiento

Clase básica: Ingrediente

Subclases: Programa distante, Programa residente, Programa intercambiado

Estado: Clase abstracta

14.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

14.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las restricciones siguientes:

Nombre de atributo	Definido en	Restricciones y requisitos
Activo inicialmente	Ingrediente	Este atributo es obligatorio para esta clase y se pondrá a Falso.

14.1.2 Atributos intercambiados propios

Esta clase define los siguientes atributos intercambiados adicionales:

Nombre (Name)	Nombre del código de procedimiento externo que se ha de llamar cuando el objeto Programa es activado. La correspondencia de Nombre con el nombre real del código de procedimiento externo no es definida por la presente Recomendación. <ul style="list-style-type: none">• Cadena de octetos.
Disponible inicialmente (InitiallyAvailable)	Este parámetro se utiliza para determinar los Programas en una Escena o en una Aplicación que están preparados inicialmente. <ul style="list-style-type: none">• Booleano facultativo.• Valor por defecto: Verdadero.

14.1.3 Atributos internos propios

Esta clase no define atributos internos adicionales.

14.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica. Además se define el siguiente evento:

<i>Detenido asíncrono (AsynchStopped)</i>	Este evento será generado cuando un objeto Programa, que es ejecutado por la acción Ramificación temporal ha terminado su ejecución. <ul style="list-style-type: none">• Ningún dato asociado.
---	--

14.3 Comportamientos internos

La semántica de los siguientes comportamientos internos ha cambiado con respecto a la clase básica de este objeto:

<i>Activación</i>	<ol style="list-style-type: none">1) Si no se ha hecho durante la preparación, localizar el código de procedimiento externo utilizando el atributo nombre.2) Si no se ha encontrado el código de procedimiento externo, omitir esta acción. En los demás casos:3) Fijar los parámetros del código de procedimiento externo del Programa según lo indicado por <i>Parámetros</i>.4) Aplicar el comportamiento <i>Activación</i> heredado de la clase básica.5) Comenzar la ejecución del código de procedimiento externo síncrona o asíncronamente de acuerdo con la acción que invoca la ejecución.6) Fijar el atributo <i>Estado de ejecución</i> a Verdadero.7) Generar un evento <i>Está funcionando</i>.
<i>Desactivación</i>	Si el atributo <i>Está funcionando</i> es Falso, omitir esta acción. En los demás casos, ejecutar la siguiente secuencia de acciones: <ol style="list-style-type: none">1) Forzar el fin de la ejecución del Programa.2) Aplicar el comportamiento <i>Desactivación</i> heredado de la clase básica. NOTA – El comportamiento desactivación genera un evento <i>Detenido</i> , definido en Raíz.

14.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se definen las siguientes acciones MHEG-5 aplicables:

Fijar datos
(SetData) Esta acción no será aplicada a un objeto de cualquier subclase de la clase Programa.

Llamar
(Llamada
conseguida,
Parámetros)

Solicitar la ejecución de una pieza externa de código de procedimiento y esperar el fin de la ejecución.

[Call
(CallSucceeded,
Parameters)] Ejecutar la siguiente secuencia de acciones:

- 1) Si el Programa no está disponible, aplicar el comportamiento *Preparación*.
- 2) Si el Programa está activo, omitir esta acción. En los demás casos:
- 3) Aplicar el comportamiento *Activación*.
- 4) Esperar la ejecución de un código de procedimiento externo para terminar. Si el Programa termina anormalmente, fijar la Variable referenciada por *Llamada conseguida* a Falso. En los demás casos, fijarla a Verdadero.
- 5) Fijar el valor de las Variables referenciadas por *Parámetros* a los valores devueltos por el Programa (éstos pueden ser no válidos si *Llamada conseguida* es Falso).
- 6) Aplicar el comportamiento *Desactivación*.

Disposiciones de uso:

- *Llamada conseguida* se pondrá a un objeto Variable booleana activo.
- *Parámetros* se pondrá a una lista de valores correspondiente a los parámetros previstos del código de procedimiento externo. El orden de la lista *Parámetros* corresponderá con el orden de los parámetros del código de procedimiento. Los parámetros pasados por valor se pondrán directamente al valor correspondiente. Los parámetros pasados por referencia se pasarán por medio de objetos Variables activos del tipo de datos correspondiente.

Descripción de sintaxis:

Call	-->	Target , CallSucceeded , Parameters?
Target	-->	GenericObjectReference
CallSucceeded	-->	ObjectReference
Parameters	-->	Parameter+
Parameter	-->	GenericBoolean GenericInteger GenericOctetString GenericObjectReference GenericContentReference

Ramificación temporal
(*Ramificación conseguida*,
Parámetros)

[Fork
(*ForkSucceeded*,
Parameters)]

Solicitar la ejecución de una pieza externa de código de procedimiento sin esperar el fin de la ejecución.

Ejecutar la siguiente secuencia de acciones:

- 1) Si el Programa no está disponible, aplicar el comportamiento *Preparación*.
- 2) Si el Programa está activo, omitir esta acción. En los demás casos:
- 3) Aplicar el comportamiento *Activación*.
- 4) Devolver el control al motor MHEG-5 sin esperar que termine la ejecución del código de procedimiento externo.

Cuando termina la ejecución del código de procedimiento externo, ejecutar la siguiente secuencia de acciones:

- 1) Si el Programa termina anormalmente, fijar a Falso la Variable referenciada por *Ramificación temporal conseguida*; en los demás casos, fijarla a Verdadero.
- 2) Fijar el valor de las Variables referenciadas por *Parámetros* a los valores devueltos por el Programa (éstos pueden ser no válidos si *Ramificación temporal conseguida* es Falso).
- 3) Aplicar el comportamiento *Desactivación*.
- 4) Generar un evento *Detenido asíncrono*.

NOTA – Los parámetros pueden ser modificados por el Programa, en cuyo caso, estos parámetros no son definidos hasta que el Programa termina normalmente, es decir, hasta que se genera un evento *Detenido asíncrono*.

Disposiciones de uso:

- *Ramificación temporal conseguida* se pondrá a un objeto Variable Booleana activo.
- *Parámetros* se pondrá a una lista de valores correspondientes a los parámetros previstos del código de procedimiento externo. El orden de la lista de *Parámetros* corresponderá con el orden de los parámetros del código de procedimiento. Los parámetros pasados por valor se pondrán directamente al valor correspondiente. Los parámetros pasados por referencia serán pasados a través de objetos variables activos del tipo de datos correspondiente.

Descripción de sintaxis:

Fork	-->	Target , ForkSucceeded , Parameters?
Target	-->	GenericObjectReference
ForkSucceeded	-->	ObjectReference
Parameters	-->	Parameter+
Parameter	-->	GenericBoolean GenericInteger GenericOctetString GenericObjectReference GenericContentReference

Detener (Stop) Interrumpir la ejecución de una pieza externa de código de procedimiento.
Ejecutar la siguiente secuencia de acciones:

- 1) Si el Programa está inactivo, omitir esta acción. En los demás casos:
- 2) Aplicar el comportamiento *Desactivación*.

Disposición de uso:

- El objeto *Objetivo* será un objeto Programa disponible.

Descripción de sintaxis:

Stop	-->	Target
Target	-->	GenericObjectReference

14.5 Descripción formal

Program Class	-->	Ingredient Class,
		Name,
		InitiallyAvailable?
Name	-->	
InitiallyAvailable	-->	BOOLEAN

15 Clase Programa Residente

Descripción: Define los medios para tratar llamadas al código de procedimiento externo ejecutado localmente.

Un objeto Programa residente proporciona una interfaz a una pieza de código de procedimiento que es local al dispositivo en el cual está funcionando el motor MHEG-5. Esta Recomendación no especifica un paradigma de denominación para esta interfaz de procedimiento ni la semántica interna para llamar a este Programa.

Clase básica: Programa

Subclases: Ninguna

Estado: Clase concreta

15.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

15.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las siguientes restricciones:

Nombre de atributo	Definido en	Restricciones y requisitos
Gancho de contenido	Ingrediente	Este atributo no se codificará para esta clase.
Contenido original	Ingrediente	Este atributo no se codificará para esta clase.

15.1.2 Atributos intercambiados propios

Esta clase no define atributos intercambiados adicionales.

15.1.3 Atributos internos propios

Esta clase no define atributos internos adicionales.

15.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

15.3 Comportamientos internos

Esta clase tiene los mismos comportamientos internos que su clase básica, con semántica idéntica.

15.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica.

15.5 Descripción formal

ResidentProgram Class	-->	Program Class
-----------------------	-----	---------------

16 Clase Programa Distante

Descripción: Define los medios para tratar llamadas a un código de procedimiento ejecutado a distancia.

Un objeto Programa distante proporciona una interfaz a una pieza de código de procedimiento que se ha de ejecutar en un lugar fuera del dispositivo en el cual está funcionando el motor MHEG-5. La situación del objeto Programa distante es proporcionado por medio del atributo *Rótulo de conexión de programa*.

Clase básica: Programa

Subclases: Ninguna

Estado: Clase concreta

16.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

16.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las restricciones siguientes:

Nombre de atributo	Definido en	Restricciones y requisitos
Gancho de contenido	Ingrediente	Este atributo no se codificará para esta clase.
Contenido original	Ingrediente	Este atributo no se codificará para esta clase.

16.1.2 Atributos intercambiados propios

Esta clase define el siguiente atributo intercambiado adicional:

Rótulo de conexión de programa (ProgramConnection-Tag)	Rótulo de la conexión utilizada para localizar el código de procedimiento distante que se ha de llamar cuando se activa el objeto Programa. El Rótulo de conexión de programa es un identificador de una conexión abierta por la acción Abrir conexión de la clase Aplicación.
---	--

El Rótulo de conexión de programa es facultativo. Cuando no está codificado, el código de procedimiento externo se coloca en relación con el espacio de nombre por defecto de la aplicación.

- Entero facultativo.
- Valor por defecto: Ninguno

16.1.3 Atributos internos propios

Esta clase no define atributos internos adicionales.

16.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

16.3 Comportamientos internos

La semántica del siguiente comportamiento interno ha cambiado con respecto a la clase básica de este objeto:

- | | | |
|-------------------|----|---|
| <i>Activación</i> | 1) | Si no se ha hecho durante la preparación, localizar el código de procedimiento externo distante utilizando los atributos Nombre y Rótulo de conexión de programa. |
| | 2) | Si no se ha encontrado el código de procedimiento externo distante, omitir esta acción. En los demás casos: |
| | 3) | Aplicar el comportamiento <i>Activación</i> heredado de la clase básica. |

16.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica.

16.5 Descripción formal

RemoteProgram Class	-->	Program Class
		ProgramConnectionTag?
ProgramConnectionTag	-->	INTEGER

17 Clase Programa Intercambiado

Descripción: Define los medios para tratar el código de programa intercambiado como el Contenido original de un objeto Programa intercambiado y ejecutado o interpretado en el mismo dispositivo que el motor MHEG-5.

Esta Recomendación no especifica cómo se ejecuta o interpreta el código de procedimiento en el dispositivo en el cual está funcionando el motor MHEG-5.

Clase básica: Programa

Subclases: Ninguna

Estado: Clase concreta

17.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

17.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las restricciones siguientes:

Nombre de atributo	Definido en	Restricciones y requisitos
Gancho de contenido	Ingrediente	Este atributo o su atributo por defecto correspondiente en la clase Aplicación (Gancho de contenido de programa intercambiado) es obligatorio para esta clase.
Contenido original	Ingrediente	Este atributo es obligatorio para esta clase.

17.1.2 Atributos intercambiados propios

Esta clase no define atributos intercambiados adicionales.

17.1.3 Atributos internos propios

Esta clase no define atributos internos adicionales.

17.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

17.3 Comportamientos internos

Esta clase tiene los mismos comportamientos internos que su clase básica, con semántica idéntica.

17.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica.

17.5 Descripción formal

InterchangedProgram Class	-->	Program Class
---------------------------	-----	---------------

18 Clase Paleta

Descripción: Define una clase para representar una tabla de mejora de colores

Clase básica: Ingrediente

Subclases: Ninguna

Estado: Clase concreta

18.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

18.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las restricciones siguientes:

Nombre de atributo	Definido en	Restricciones y requisitos
Gancho de contenido	Ingrediente	Este atributo no se codificará para esta clase.
Contenido original	Ingrediente	Este atributo no se codificará para esta clase.
Activo inicialmente	Ingrediente	Si está codificado, este atributo se pondrá a Verdadero para esta clase.

18.1.2 Atributos intercambiados propios

Esta clase no define atributos intercambiados adicionales.

18.1.3 Atributos internos propios

Esta clase no define atributos internos adicionales.

18.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

18.3 Comportamientos internos

Esta clase tiene los mismos comportamientos internos que su clase básica, con semántica idéntica.

18.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica.

18.5 Descripción formal

Palette Class	-->	Ingredient Class
---------------	-----	------------------

19 Clase Tipo de Carácter

Descripción: Define una clase para representar un tipo de carácter utilizado para representar objetos texto

Clase básica: Ingrediente

Subclases: Ninguna
Estado: Clase concreta

19.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

19.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las siguientes restricciones:

Nombre de atributo	Definido en	Restricciones y requisitos
Gancho de contenido	Ingrediente	Este atributo es obligatorio para esta clase.
Contenido original	Ingrediente	Este atributo es obligatorio para esta clase.
Activo inicialmente	Ingrediente	Si está codificado, este atributo se pondrá a Verdadero para esta clase.

19.1.2 Atributos intercambiados propios

Esta clase no define atributos intercambiados adicionales.

19.1.3 Atributos internos propios

Esta clase no define atributos internos adicionales.

19.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

19.3 Comportamientos internos

Esta clase tiene los mismos comportamientos internos que su clase básica, con semántica idéntica.

19.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica.

19.5 Descripción formal

Font Class	-->	Ingredient Class
------------	-----	------------------

20 Clase Forma de cursor

Descripción: Define la encapsulación para las estructuras de datos utilizadas para representar un cursor de movimiento libre

Clase básica: Ingrediente

Subclases: Ninguna

Estado: Clase concreta

20.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

20.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las restricciones siguientes:

Nombre de atributo	Definido en	Restricciones y requisitos
Gancho de contenido	Ingrediente	Este atributo no se codificará para esta clase.
Contenido original	Ingrediente	Este atributo no se codificará para esta clase.
Activo inicialmente	Ingrediente	Si está codificado, este atributo se pondrá a Verdadero para esta clase.

20.1.2 Atributos intercambiados propios

Esta clase no define atributos intercambiados adicionales.

20.1.3 Atributos internos propios

Esta clase no define atributos internos adicionales.

20.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

20.3 Comportamientos internos

Esta clase tiene los mismos comportamientos internos que su clase básica, con semántica idéntica.

20.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica.

20.5 Descripción formal

<code>CursorShape Class</code> <code>--> Ingredient Class</code>

21 Clase Variable

Descripción: Define una variable dentro del contexto de un objeto Grupo

Clase básica: Ingrediente

Subclases: Variable booleana, Variable entero, Variable cadena de octetos, Variable referencia de objeto, Variable referencia de contenido

Estado: Clase abstracta

21.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

21.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las restricciones siguientes:

Nombre de atributo	Definido en	Restricciones y requisitos
Gancho de contenido	Ingrediente	Este atributo no se codificará para esta clase.
Contenido original	Ingrediente	Este atributo no se codificará para esta clase.
Activo inicialmente	Ingrediente	Si está codificado, este atributo se pondrá a Verdadero para esta clase.

21.1.2 Atributos intercambiados propios

Esta clase define los siguientes atributos intercambiados adicionales:

Valor original
(OriginalValue) Valor de la variable cuando se prepara por primera vez.
El atributo *Valor original* será uno de estos tipos: Booleano, Entero, Cadena de octetos, Referencia de objeto o Referencia de contenido.

21.1.3 Atributos internos propios

Esta clase define el siguiente atributo interno adicional:

Valor
(Value) Valor actual de la variable.
El atributo *Valor* puede ser uno de estos tipos: Booleano, Entero, Cadena de octetos, Referencia de objeto o Referencia de contenido. La única acción elemental que puede modificar este atributo es la acción Fijar variable.

- Valor inicial: Valor del atributo valor original.

21.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica. Además, se define el siguiente evento:

Evento de prueba
(TestEvent) Este evento será generado por el motor MHEG-5 para indicar que una subclase de Variable ha sido probada.

- Datos asociados: Booleanos. El resultado de la comparación entre la variable y el parámetro de la acción Probar variable.

21.3 Comportamientos internos

La semántica del siguiente comportamiento interno ha cambiado con respecto a la clase básica de este objeto:

Activación Ejecutar la siguiente secuencia de acciones:

- 1) Aplicar el comportamiento *Activación* definido en la clase básica.
- 2) Fijar el atributo *Estado de ejecución* a Verdadero y generar un evento *Está funcionando*.

21.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se definen las siguientes acciones MHEG-5 aplicables:

Fijar variable
(*Nuevo valor de variable*)

[SetVariable
(*NewVariable Value*)]

Fijar el atributo *Valor* del objeto *Objetivo* a *Nuevo valor de variable*.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto activo de una de las siguientes clases: Variable booleana, Variable entera, Variable cadena de octetos, Variable referencia de objeto o Variable referencia de contenido.
- *Nuevo valor de variable* se pondrá o hará referencia a un valor que no es necesariamente del mismo tipo que el atributo *Valor* actual de la Variable objetivo. Cuando *Nuevo valor de variable* y el objeto *Objetivo* son de dos tipos diferentes, *Nuevo valor de variable* se convierte automáticamente al tipo de clase de Objetivo.

NOTA – En las subcláusulas pertinentes a cada subclase de Variables se dan más detalles sobre las conversiones.

Descripción de sintaxis:

SetVariable	-->	Target , NewVariableValue
Target	-->	GenericObjectReference
NewVariableValue	-->	GenericInteger GenericBoolean GenericOctetString GenericObjectReference GenericContentReference

Probar variable
(*Objetivo, Operador, Valor de comparación*)

[TestVariable
(*Target, Operator, ComparisonValue*)]

Ejecutar la siguiente secuencia de acciones:

- 1) Comparar el *Valor* de la variable con el parámetro *Valor* de comparación. El valor de la Variable es el primer operando, el parámetro *Valor* de comparación es el segundo operando de la comparación.
- 2) Generar el Evento de prueba correspondiente.
 - Fuente: Objetivo de Variable de prueba.
 - Datos asociados: Booleanos.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto activo de una de las clases siguientes: Variable booleana, Variable entera, Variable cadena de octetos, Variable referencia de objeto o Variable referencia de contenido.
- El *Valor* de comparación será del tipo correspondiente (Booleano genérico, Entero genérico, Cadena de octetos genérica, Referencia de objeto genérica y Referencia de contenido genérica respectivamente). No se permite ninguna conversión de tipo implícita.

- Cuando los valores son Enteros, el Operador será un entero comprendido en la gama 1 a 6 con el siguiente significado:
1 significa igual, 2 no igual,
3 estrictamente menos que, 4 menos o igual que,
5 estrictamente mayor que, 6 mayor o igual que.
- Cuando los valores son Booleanos, Cadena de octetos, Referencia de objeto o Referencia de contenido, el Operador será un entero en la gama 1 a 2 con el significado siguiente:
1 significa igual, 2 no igual.

Descripción de sintaxis:

TestVariable	-->	Target , Operator , ComparisonValue
Target	-->	GenericObjectReference
ComparisonValue	-->	GenericInteger GenericBoolean GenericOctetString GenericObjectReference GenericContentReference
Operator	-->	GenericInteger

21.5 Descripción formal

Variable Class	-->	Ingredient Class , OriginalValue
OriginalValue	-->	BOOLEAN INTEGER OctetString ObjectReference ContentReference

22 Clase Variable booleana

Descripción: Define una variable de tipo Booleano dentro del contexto de un objeto Grupo

Clase básica: Variable

Subclases: Ninguna

Estado: Clase concreta

22.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

22.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, sin restricciones adicionales.

22.1.2 Atributos intercambiados propios

Valor original El atributo valor original será un Booleano.
(OriginalValue)

22.1.3 Atributos internos propios

Esta clase define el siguiente atributo interno adicional.

Valor • El atributo *Valor* será un Booleano.

22.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

22.3 Comportamientos internos

Esta clase tiene los mismos comportamientos internos que su clase básica, con semántica idéntica.

22.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se define la siguiente acción MHEG-5 aplicable:

Fijar variable Disposición de uso:
(*Nuevo valor de* • *Nuevo valor de variable* será un Booleano genérico.
variable)

22.5 Descripción formal

BooleanVariable Class --> Variable Class
--

23 Clase Variable entera

Descripción: Define una variable de tipo Entero dentro del contexto de un objeto Grupo

Clase básica: Variable

Subclases: Ninguna

Estado: Clase concreta

23.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

23.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, sin restricciones adicionales.

23.1.2 Atributos intercambiados propios

Valor original El atributo *Valor* original será un Entero.

23.1.3 Atributos internos propios

Esta clase define los siguientes atributos internos adicionales.

Valor (Value) • El atributo *Valor* será un entero.

23.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

23.3 Comportamientos internos

Esta clase tiene los mismos comportamientos internos que su clase básica, con semántica idéntica.

23.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se definen las siguientes acciones MHEG-5 aplicables:

- | | |
|---|---|
| Fijar variable
(<i>Nuevo valor de variable</i>)

[SetVariable
(<i>NewVariable Value</i>)] | Disposición de uso: <ul style="list-style-type: none">• <i>Nuevo valor de variable</i> será uno de los siguientes tipos: Entero o Cadena de octetos.<ul style="list-style-type: none">– Cuando <i>Nuevo valor de variable</i> es del tipo Entero, el valor de la variable <i>Objetivo</i> se pone al valor de <i>Nuevo valor de variable</i>.– Cuando <i>Nuevo valor de variable</i> es del tipo Cadena de octetos, se convierte primero a un Entero, después el valor de la variable <i>objetivo</i> se pone a un valor Entero. |
|---|---|

Reglas para la conversión:

- 1) La conversión de Cadena de octetos utilizará el atributo Conjunto de caracteres de la Aplicación. Si están enumerados varios conjuntos de caracteres, la conversión se efectúa en caracteres mientras sean "numéricos" en su conjunto de código.
- 2) La conversión considerará que la Cadena de octetos representa el Entero base 10.
- 3) La conversión de Cadena de octetos se efectuará en los primeros caracteres numéricos (en ISO/CEI 646: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9) y se detendrá tan pronto como se encuentre un carácter no numérico [con excepción del signo menos en primera posición, como se indica en la regla 4) siguiente]. Por ejemplo, una Cadena de octetos que representa "123abc" se convertirá en el entero 123.
- 4) El signo menos está permitido como primer carácter de una Cadena de octetos. Por ejemplo, la Cadena de octetos que representa "-123" se convertirá en Entero -123, pero "12-345" se convertirá en el entero 12 de acuerdo con la regla 3).

- | | |
|--|--|
| Añadir
(<i>Valor</i>)

[Add (<i>Value</i>)] | Añadir <i>Valor</i> a la Variable <i>Objetivo</i> . La variable <i>objetivo</i> es el primer operando de la operación infijo. El resultado se almacena en Variable <i>objetivo</i> . |
|--|--|

Disposición de uso:

- El objeto *Objetivo* será un objeto Variable entera activo.

Descripción de sintaxis:

Add	-->	Target ,
		Value
Target	-->	GenericObjectReference
Value	-->	GenericInteger

Sustraer
(*Valor*)

[Subtract (*Value*)]

Sustraer *Valor* de la variable *Objetivo*. La variable *Objetivo* es el primer operando de la operación infijo. El resultado se almacena en Variable de objetivo.

Disposición de uso:

- El objeto *Objetivo* será un objeto Variable entera activo.

Descripción de sintaxis:

Subtract	-->	Target ,
		Value
Target	-->	GenericObjectReference
Value	-->	GenericInteger

Multiplicar
(*Valor*)

[Multiply (*Value*)]

Multiplica la Variable *Objetivo* por *Valor*. La variable *Objetivo* es el primer operando de la operación infijo. El resultado se almacena en Variable objetivo.

Disposición de uso:

- El objeto *Objetivo* será un objeto Variable entera activo.

Descripción de sintaxis:

Multiply	-->	Target ,
		Value
Target	-->	GenericObjectReference
Value	-->	GenericInteger

Dividir
(*Valor*)

[Divide (*Value*)]

Divide la variable *Objetivo* por *Valor*. La variable *Objetivo* es el primer operando de la operación infijo. El resultado se almacena en la Variable objetivo. Cuando el resultado no es un valor entero, se redondea a 0.

Disposición de uso:

- El objeto *Objetivo* será un objeto Variable entera activo.

Descripción de sintaxis:

Divide	-->	Target ,
		Value
Target	-->	GenericObjectReference
Value	-->	GenericInteger

Módulo
(*Valor*)
[Módulo
(*Value*)]

Devuelve el módulo residuo *Valor* de *objetivo*, definido por las reglas aritméticas de enteros usuales, es decir, que para cualesquiera enteros *a* y *b* se satisface la siguiente ecuación:

$$(a \text{ DIV } b) \times b + (a \text{ MOD } b) = a.$$

La variable *Objetivo* es el primer operando de la operación infijo. El resultado se almacena en *Variable objetivo*.

Disposición de uso:

- El objeto *Objetivo* será un objeto *Variable* entera activo.

Descripción de sintaxis:

Modulo	-->	Target , Value
Target	-->	GenericObjectReference
Value	-->	GenericInteger

23.5 Descripción formal

IntegerVariable Class	-->	Variable Class
-----------------------	-----	----------------

24 Clase Variable cadena de octetos

Descripción: Define una variable de tipo Cadena de octetos dentro del contexto de un objeto Grupo

Clase básica: Variable

Subclases: Ninguna

Estado: Clase concreta

24.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

24.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, sin restricciones adicionales.

24.1.2 Atributos intercambiados propios

Valor original El atributo Valor original será una Cadena de octetos.

24.1.3 Atributos internos propios

Esta clase define el siguiente atributo interno adicional.

Valor

- El atributo *Valor* será una Cadena de octetos.

24.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

24.3 Comportamientos internos

Esta clase tiene los mismos comportamientos internos que su clase básica, con semántica idéntica.

24.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se definen las siguientes acciones MHEG-5 aplicables:

- | | |
|---|---|
| Fijar variable
(<i>Nuevo valor de variable</i>)
[SetVariable
(<i>NewVariable Value</i>)] | Disposición de uso: <ul style="list-style-type: none">• <i>Nuevo valor de variable</i> será uno de los siguientes tipos: Entero o Cadena de octetos.<ul style="list-style-type: none">– Cuando <i>Nuevo valor de variable</i> es del tipo Cadena de octetos, el valor de la Variable objetivo se pone al valor de <i>Nuevo valor de variable</i>.– Cuando <i>Nuevo valor de variable</i> es del tipo Entero, se convierte primero a una Cadena de octetos, después el valor de la Variable objetivo se pone al valor de esa Cadena de octetos. |
|---|---|

Reglas para la conversión:

- 1) La conversión de Entero a Cadena de octetos utilizará el atributo Conjunto de caracteres de la Aplicación.
- 2) La conversión se hará a la representación del entero base 10.

- | | |
|--|--|
| Anexar
(<i>Valor anexo</i>)
[Append
(<i>AppendValue</i>)] | Añade <i>Valor anexo</i> a la Variable <i>Objetivo</i> . La Variable <i>Objetivo</i> es el primer operando de la operación infijo. El resultado de almacena en Variable objetivo.

Disposición de uso: <ul style="list-style-type: none">• El objeto <i>Objetivo</i> será un objeto Variable de cadena de octetos. |
|--|--|

Descripción de sintaxis:

Append	-->	Target , AppendValue
Target	-->	GenericObjectReference
AppendValue	-->	GenericOctetString

24.5 Descripción formal

OctetStringVariable Class	-->	Variable Class
---------------------------	-----	----------------

25 Clase Variable referencia de objeto

- | | |
|---------------|---|
| Descripción: | Define una variable de tipo Referencia de objeto dentro del contexto de un objeto Grupo |
| Clase básica: | Variable |
| Subclases: | Ninguna |
| Estado: | Clase concreta |

25.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

25.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, sin restricciones adicionales.

25.1.2 Atributos intercambiados propios

Valor original El atributo Valor original será una Referencia de objeto.

25.1.3 Atributos internos propios

Esta clase define el siguiente atributo interno adicional.

Valor • El atributo *Valor* será una Referencia de objeto.

25.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

25.3 Comportamientos internos

Esta clase tiene los mismos comportamientos internos que su clase básica, con semántica idéntica.

25.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se define la siguiente acción MHEG-5 aplicable:

Fijar variable Disposición de uso:
(*Nuevo valor de variable*) • *Nuevo valor de variable* será una Referencia de objeto genérica.

25.5 Descripción formal

ObjectRefVariable Class --> Variable Class
--

26 Clase Variable referencia de contenido

Descripción: Define una variable de tipo Referencia de contenido dentro del contexto de un objeto Grupo

Clase básica: Variable

Subclases: Ninguna

Estado: Clase concreta

26.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

26.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, sin restricciones adicionales.

26.1.2 Atributos intercambiados propios

Valor original El atributo Valor original será una Referencia de contenido.

26.1.3 Atributos internos propios

Esta clase define el siguiente atributo interno adicional.

Valor • El atributo *Valor* será una Referencia de contenido.

26.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

26.3 Comportamientos internos

Los comportamientos internos de esta clase tienen la misma semántica que su clase básica.

26.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se define la siguiente acción MHEG-5 aplicable:

Fijar variable	Disposición de uso:
(<i>Nuevo valor de variable</i>)	• <i>Nuevo valor de variable</i> será una Referencia de contenido genérica.

26.5 Descripción formal

ContentRefVariable Class --> Variable Class

27 Clase Presentable

Descripción: Define el comportamiento de objetos que pueden ser presentados dentro de una Escena

Clase básica: Ingrediente

Subclases: Tren, Visible, Audio, Grupo de testigos

Estado: Clase abstracta

27.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

27.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con semántica idéntica.

27.1.2 Atributos intercambiados propios

Esta clase no define atribuidos intercambiados adicionales.

27.1.3 Atributos internos propios

Esta clase no define atributos internos adicionales.

27.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

27.3 Comportamientos internos

Los comportamientos internos de esta clase tienen la misma semántica que su clase básica.

27.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica, salvo lo que se indica a continuación (Fijar datos). Además, se definen las siguientes acciones MHEG-5 aplicables:

Fijar datos (SetData) Ejecutar sincronamente las siguientes acciones:

- 1) Ejecutar la acción Fijar datos definida en la clase Ingrediente.
- 2) Si el objeto Presentable está activo, presentarlo de nuevo inmediatamente utilizando el nuevo valor de *Contenido*.

Las disposiciones de uso y la descripción de sintaxis se proporcionan en la clase Ingrediente.

Ejecutar (Run) Ejecutar las siguientes acciones:

- 1) Si el objeto Presentable objetivo está activo, omitir esta acción.
- 2) Si el objeto Presentable objetivo no está disponible o está inactivo, aplicar el comportamiento *Activación* de la clase Presentable.

Descripción de sintaxis:

Run	-->	Target
Target	-->	GenericObjectReference

Detener (Stop) Ejecutar las siguientes acciones:

- 1) Si el objeto Presentable objetivo está inactivo, omitir esta acción.
- 2) Si el objeto Presentable objetivo está activo, aplicar el comportamiento *Desactivación* de la clase Presentable.

Disposición de uso:

- El objeto *Objetivo* estará disponible.

Descripción de sintaxis:

Stop	-->	Target
Target	-->	GenericObjectReference

27.5 Descripción formal

Presentable Class	-->	Ingredient Class
-------------------	-----	------------------

28 Clase Gestor de testigos

Descripción:	Clase mezclada que define funciones para gestionar la navegación de un testigo lógico entre un grupo de elementos. Se puede utilizar el testigo para dar un comportamiento especial a un elemento del grupo, tal como el realce en un esquema de navegación de salto-realce
Clase básica:	Ninguna (mezcla)
Subclases:	Grupo de testigos
Estado:	Clase abstracta

28.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

28.1.1 Atributos heredados

Esta clase no tiene atributos heredados.

28.1.2 Atributos intercambiados propios

Esta clase define los siguientes atributos intercambiados:

Tabla de movimiento (MovementTable)	Tabla que indica a cuál elemento se ha de mover el testigo de acuerdo con el elemento anterior y un número arbitrario de movimientos.
--	---

El atributo Tabla de movimiento describe una función discreta $c2 = f(c1, m)$, donde $c1$ y $c2$ son índices de elementos basados en uno y m es el parámetro *Movimiento* utilizado en la definición de la acción Mover. El índice 0 indica que ningún elemento retiene al testigo.

Por ejemplo, si el testigo está en elemento 2, y se ejecuta la acción Mover(4), la expresión $f(2, 4)$ evalúa al número del elemento para obtener el testigo.

La función f se representa como un conjunto $N \times M$, donde N es el número de elementos en el grupo y M es el número de posibles movimientos. Se hace referencia a los elementos mediante un índice numérico basado en uno.

- Atributo facultativo.
- Secuencia de estructuras de datos de Movimiento. Cada Movimiento tiene un Identificador de movimiento implícito, que es un índice de enteros basado en uno, calculado de acuerdo con la posición del Movimiento en la secuencia. Cada Movimiento consiste además de:
 - Elementos de objetivo: Secuencia de enteros. Estos Enteros definen qué elemento obtendrá el testigo cuando se realiza un Movimiento.
- Valor por defecto: Ninguno.

Ejemplo de Tabla de movimientos:

Aparición de elementos en la pantalla:

Elemento 1	Elemento 2
Elemento 3	Elemento 4

Contenido de la tabla de movimientos:

Testigo en:	Elemento 1	Elemento 2	Elemento 3	Elemento 4
Mover(1)	1	2	1	2
Mover(2)	3	4	3	4
Mover(3)	1	1	3	3
Mover(4)	2	2	4	4
Mover(5)	4	2	3	4
Mover(6)	1	2	2	4

De acuerdo con esta tabla, la acción Mover(3) cuando el testigo está en el elemento 2 resultaría en la obtención del testigo por el elemento 1. La acción Mover(4) cuando el testigo está en elemento 4 no tendría ningún efecto.

Los movimientos mostrados en la tabla podrían corresponder con claves de telecontrol como sigue:

Mover(1) → Flecha hacia arriba Mover(4) → Flecha a la derecha

Mover(2) → Flecha hacia abajo Mover(5) → Derecha diagonal hacia abajo

Mover(3) → Flecha a la izquierda Mover(6) → Derecha diagonal hacia arriba

NOTA – Si una célula en la Tabla de movimiento contiene el valor 0, el Movimiento correspondiente resultará en que ningún elemento tendrá el testigo. Véase a continuación *Posición de testigo*.

28.1.3 Atributos internos propios

Esta clase define el siguiente atributo interno adicional:

Posición de testigo Índice del elemento que tiene el testigo en ese momento.
(*TokenPosition*)

- Entero dentro de la gama [0, número de elementos].
- Valor inicial: 1.
- El valor 0 tiene el significado especial de que ningún elemento tiene el testigo.

28.2 Eventos

Esta clase define los siguientes eventos:

Testigo movido desde (*TokenMoved From*) Este evento es generado cuando un elemento pierde el testigo.

- Datos asociados: Índice. Éste es un Entero que representa el índice del elemento que pierde el testigo.

Testigo movido a (*TokenMovedTo*) Este evento es generado cuando un elemento obtiene el testigo.

- Datos asociados: Índice. Éste es un Entero que representa el índice del elemento que obtiene el testigo.

28.3 Comportamientos internos

Esta clase define el siguiente comportamiento interno:

Transferencia de testigo (Elemento objetivo)
[TransferToken (TargetElement)]

Ejecutar la siguiente secuencia de acciones:

- 1) Generar un evento *Testigo movido desde* con el valor del atributo *Posición de testigo* como datos asociados.
- 2) Fijar el atributo *Posición de testigo* al valor del parámetro *Elemento objetivo*.
- 3) Generar un evento *Testigo movido a* con el valor del atributo *Posición de testigo* como datos asociados.

Los eventos *Testigo movido a* y *Testigo movido desde* serán generados incluso si el valor de *Posición de testigo* después (antes) de que se moviese el testigo era 0.

28.4 Efectos de las acciones MHEG-5

Esta clase define las siguientes acciones MHEG-5 aplicables:

Mover (Move) Mover el testigo entre elementos del grupo. El movimiento que se ha de aplicar desde cualquier posición de elemento particular se describe en el atributo *Tabla de movimientos*.

Ejecutar la siguiente secuencia de acciones:

- 1) Determinar el *Elemento objetivo* utilizando la Tabla de movimientos.
- 2) Si el *Elemento objetivo* no tiene aún el testigo, aplicar el comportamiento *Transferencia de testigo (Elemento objetivo)* del Gestor de testigos.

Disposición de uso:

- El objeto *Objetivo* será un Gestor de testigos disponible.

Descripción de sintaxis:

Move	-->	Target ,
		Movement Identifier
Target	-->	GenericObjectReference
Movement Identifier	-->	GenericInteger

Mover a (Índice) Mover el testigo a un elemento específico del grupo.

[MoveTo (Index)] Ejecutar las siguientes secuencias de acciones:

- 1) Determinar el *Elemento objetivo* utilizando el parámetro *Índice* de la acción Mover a.
- 2) Si el elemento objetivo no tiene aún el testigo, aplicar el comportamiento *Transferencia de testigo (Elemento objetivo)* del Gestor de testigos.

Disposición de uso:

- El objeto *Objetivo* será un Gestor de testigos disponible.
- El *Índice* se fijará dentro de la gama $[0, N]$, donde N es el número de elementos en el grupo.

Descripción de sintaxis:

MoveTo	-->	Target , Index
Target	-->	GenericObjectReference
Index	-->	GenericInteger

Obtener posición
de testigo
(*Variable posición
de testigo*)

[GetTokenPosition
(*TokenPositionVar*)]

Fijar la Variable referenciada por *Variable posición de testigo* al valor del atributo *Posición de testigo*.

Disposiciones de uso:

- El objeto *Objetivo* será un Gestor de testigos disponible.
- *Variable posición de testigo* hará referencia a un objeto Variable entera activo.

Descripción de sintaxis

GetTokenPosition	-->	Target , TokenPositionVar
Target	-->	GenericObjectReference
TokenPositionVar	-->	ObjectReference

28.5 Descripción formal

TokenManager Class	-->	MovementTable?
MovementTable	-->	Movement+
Movement	-->	TargetElement+
TargetElement	-->	INTEGER

29 Clase Grupo de testigos

Descripción: Define un grupo de objetos Visibles para navegación

Cada objeto Visible del grupo puede retener un testigo, definido en la clase Gestor de testigo. Sobre la base de los eventos que son generados cuando ese testigo se mueve, se puede aplicar un comportamiento especial para el objeto Visible que mantiene el testigo, por ejemplo, un realce

Clase básica: Presentable, Gestor de testigos

Subclases: Grupo de lista

Estado: Clase concreta

29.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

29.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las siguientes restricciones:

Nombre de atributo	Definido en	Restricciones y requisitos
Gancho de contenido	Ingrediente	Este atributo no se codificará para esta clase.
Contenido original	Ingrediente	Este atributo no se codificará para esta clase.
<i>Posición de testigo</i>	Gestor de testigo	En la clase Grupo de testigos, este atributo tomará valores solamente en la gama $[0, N]$, donde N es el número de objetos Visibles en el Grupo de testigos. El valor 0 significa que ningún objeto Visible tiene el testigo.

29.1.2 Atributos intercambiados propios

Esta clase define los siguientes atributos intercambiados adicionales:

Ítems de grupo de testigos (TokenGroupItems)	<p>Conjunto de Ítems de grupo de testigo que pertenecen al grupo. Cada Ítem de grupo de testigo contiene una referencia a un objeto Visible y a un número de Acciones, denominadas Intervalos de acciones. Estas acciones pueden ser ejecutadas a petición por la acción Llamar intervalo de acciones.</p> <ul style="list-style-type: none">• Secuencia de las siguientes estructuras de datos:<ul style="list-style-type: none">– Visible: Referencia a un objeto Visible.– Intervalos de acciones: Secuencia facultativa de objetos Acción.
Intervalos de acciones sin testigo (NoTokenAction-Slots)	<p>Conjunto de Intervalo de acciones que pueden ser ejecutadas a petición por la acción Llamar intervalo de acciones cuando ningún ítem tiene el testigo.</p> <ul style="list-style-type: none">• Atributo facultativo.• Secuencia de Intervalo de acciones.

29.1.3 Atributos internos propios

Esta clase no define atributos internos adicionales.

29.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

29.3 Comportamientos internos

Los siguientes comportamientos internos de esta clase tienen cambios de semántica:

<i>Activación</i>	<p>Ejecutar la siguiente secuencia de acciones:</p> <ol style="list-style-type: none">1) Aplicar el comportamiento <i>Activación</i> heredado de la clase Presentable.2) Aplicar el comportamiento <i>Activación</i> a cada ítem en el orden dado por ítems de grupos de testigos.3) Generar un evento <i>Testigo movido</i> a con el valor del atributo <i>Posición de testigo</i> como datos asociados.4) Fijar <i>Estado de ejecución</i> a Verdadero y generar un evento <i>Está funcionando</i>.
<i>Desactivación</i>	<ol style="list-style-type: none">1) Generar un evento <i>Testigo movido</i> desde con el valor del atributo <i>Posición de testigo</i> como datos asociados.

- 2) Aplicar el comportamiento *Desactivación* heredado de la clase Presentable.

29.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica, excepto lo que se define a continuación. Además, se define la siguiente acción MHEG-5 aplicable:

Llamar intervalo de acciones (*Índice*) Ejecutar un objeto Acción asociado con el ítem que actualmente tiene el testigo.

[CallActionSlot (*Index*)] Ejecutar la siguiente secuencia de acciones:

- 1) Si *Posición de testigo* se pone a 0, considerar la secuencia de Intervalos de acciones del atributo Intervalos de tiempo sin testigos.
- 2) Si *Posición de testigo* es diferente de 0, considerar la secuencia de Intervalos de acciones que está asociada con el ítem que actualmente tiene el testigo.
- 3) Considerar el Intervalo de acciones que está en posición *Índice* en la secuencia de Intervalos de acciones. Si ese intervalo de tiempo es *NULO*, omitir el efecto de la acción Llamar intervalo de acciones. En los demás casos, ejecutar ese Intervalo de acciones.

La ejecución de la acción Llamar intervalo de acciones no se completa hasta que la acción Intervalo de acciones ha sido ejecutada.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Grupo de testigos activo.
- *Índice* se fijará en la gama [0, número de ítems en el Ítem de grupo de testigos].

Descripción de sintaxis:

CallActionSlot	-->	Target , Index
Target	-->	GenericObjectReference
Index	-->	GenericInteger

29.5 Descripción formal

TokenGroup Class	-->	Presentable Class , TokenManager Class , TokenGroupItems , NoTokenActionSlots?
NoTokenActionSlots	-->	ActionSlots
TokenGroupItems	-->	TokenGroupItem+
TokenGroupItem	-->	AVisible , ActionSlots?
Avisible	-->	ObjectReference
ActionSlots	-->	ActionSlot+
ActionSlot	-->	Action Class NULL

30 Clase Grupo de listas

Descripción: Esta clase define posiciones en la pantalla para cada posición definida en la *Tabla de movimientos* heredada de la clase básica. Además, define la funcionalidad para presentar objetos Visibles contenidos en una lista interna en esas posiciones y añadir y suprimir objetos Visibles en esta lista interna

Clase básica: Grupo de testigos

Subclases: Ninguna

Estado: Clase concreta

NOTA 1 – Fijar posición, Obtener posición, Ejecutar, Detener y otras acciones elementales pueden ser dirigidas a objetos visibles en Grupo de listas, sin embargo, el autor debe advertir los posibles efectos laterales de estas acciones (por ejemplo, en relación con *Posición* de objetos Visibles).

NOTA 2 – Aunque no se recomienda, se permite tener un objeto Visible referenciado por varios grupos de listas, en cuyo caso los comportamientos de ese objeto Visible dependen de qué Grupo de listas o acciones elementales son invocadas. El autor debe estar consciente de los posibles efectos laterales.

30.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

30.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica.

30.1.2 Atributos intercambiados propios

Esta clase define los siguientes atributos intercambiados adicionales:

Posiciones (Positions) Conjunto de coordenadas de pantalla asociadas con las posiciones especificadas por el atributo *Tabla de movimiento de testigos* heredado de Grupo de testigos. Juntos estos atributos especifican las *células* del Grupo de listas. Cada célula tiene un índice, idéntico a la posición lógica definida en la *Tabla de movimiento de testigos*.

- Secuencia de la siguiente estructura de datos:
 - Posición: Par de enteros (Posición X, Posición Y).

Retorno a comienzo de línea (WrapAround) Un Booleano facultativo que indica el comportamiento del Grupo de listas con respecto a la presentación de los ítems en la *Lista de ítems*.

- Booleano facultativo.
- Valor por defecto: Falso.

Selección múltiple (MultipleSelection) Un Booleano facultativo que determina si el Grupo de listas permite que múltiples ítems tengan su *Estado de selección de ítem* puesto a Verdadero simultáneamente.

- Booleano facultativo.
- Valor por defecto: Falso.

30.1.3 Atributos internos propios

Esta clase define los siguientes atributos internos adicionales:

Lista de ítems (ItemList) Conjunto de referencias a objetos Visibles que pertenecen al grupo. Cada referencia tiene implícitamente un número de índice, comenzando en 1. Este número de índice se utiliza para hacer referencia a las referencias en el atributo *Lista de ítems*.

Cada ítem en la *Lista de ítems* tiene un atributo *Estado de selección de ítem*. Este *Estado de selección de ítem* es un Booleano, con un valor por defecto de Falso. Este atributo se puede fijar utilizando las acciones elementales *Seleccionar ítem* y *Deseleccionar ítem*; su valor puede ser devuelto utilizando la acción elemental *Obtener estado de ítem*. Obsérvese que este atributo no forma parte de los objetos Visibles referenciados en la *Lista de ítems*.

Cuando el Grupo de listas está activo, los ítems de la *Lista de ítems*, comenzando con el ítem que es indicado por el atributo *primer ítem*, son presentados en las células. Los otros ítems están inactivos. La presentación depende también del valor del atributo Retorno a comienzo de línea.

En la preparación, este conjunto es inicializado al conjunto de referencias enumeradas en el atributo *Ítems de grupos de testigo*.

- Conjunto de referencias a objetos Visibles.
- Valor inicial: Conjunto vacío.

Primer ítem (FirstItem) El índice del ítem de la *Lista de ítems* que es presentada en la primera célula. Define una "ventana" en la lista ordenada de ítems en la *Lista de ítems*. Esta ventana tiene un tamaño igual al número de células y la posición de esta ventana con respecto a los ítems puede ser cambiada utilizando la acción elemental *Desfile vertical de ítems*.

La presentación de los ítems en la lista depende de la posición de esta ventana y del valor del atributo *Retorno a comienzo de línea*.

Si hay tantos ítems en la *Lista de ítems* como células, la ventana puede ser visualizada fácilmente. Esta situación se muestra dos veces a continuación. En el primer ejemplo, *Retorno a comienzo de línea* es Falso, es decir, la lista no retorna:

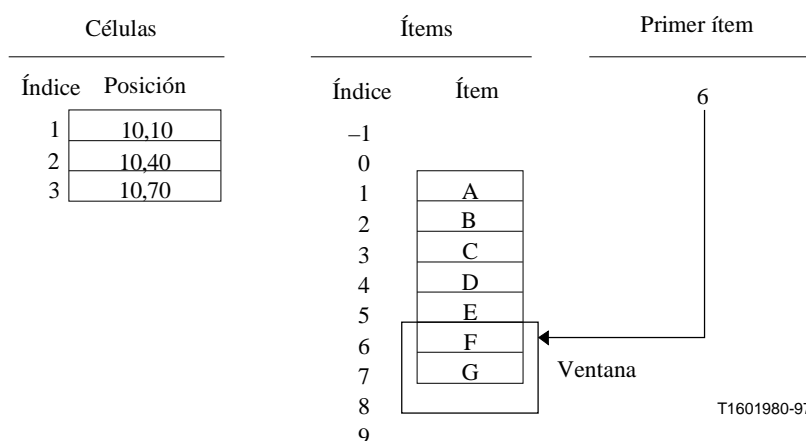


Figura 6/T.172 – Presentación cuando Retorno a comienzode línea es Falso

En este caso, los ítems F y G son presentados en las células 1 y 2 respectivamente, mientras que los otros ítems no son presentados. La célula 3 permanece inutilizada.

En el siguiente ejemplo, retorno a comienzo de línea es verdadero, es decir, la lista retorna:

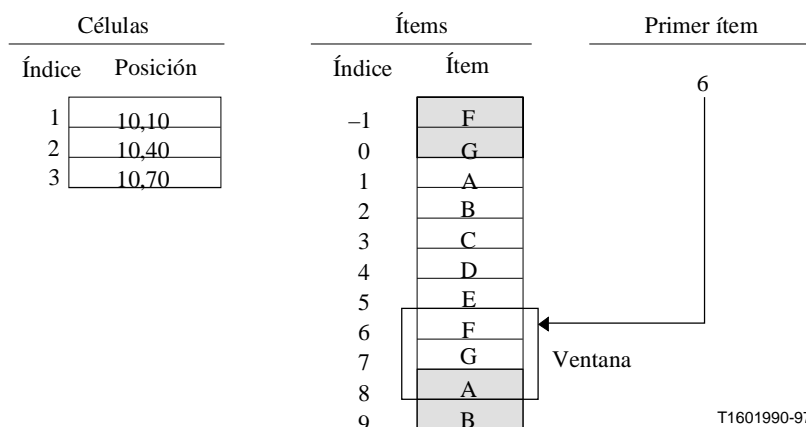


Figura 7/T.172 – Presentación cuando Retorno a comienzo de línea es Verdadero

En estos casos, los ítems F, G y A son presentados en las células 1, 2 y 3 respectivamente, mientras que los otros ítems no son presentados. Obsérvese que en esta lista los índices de los ítems son ampliados (módulo del número de ítems en la lista). Por ejemplo, ..., -4, 3, 10, ... son todos índices válidos para el ítem C en el ejemplo anterior.

La otra situación se produce si el número de ítems en la lista de ítems es menor que el número de células. Esta situación se muestra dos veces a continuación. En el siguiente ejemplo, *Retorno a comienzo de línea* es Falso, es decir la lista no retorna:

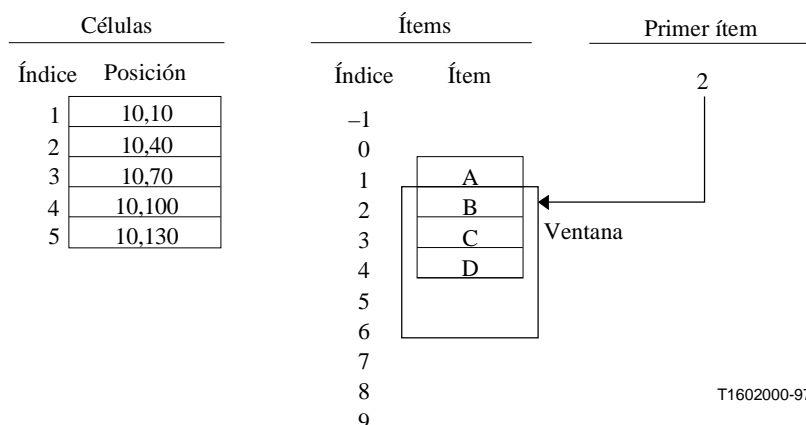


Figura 8/T.172 – Presentación de algunos ítems cuando Retorno a comienzo de línea es Falso

Este caso es muy parecido al primer ejemplo, solamente que la ventana es mayor. Los ítems B, C y D son presentados en las células 1, 2 y 3, respectivamente, mientras que el ítem A no es presentado. Las células 4 y 5 permanecen inutilizadas.

En el siguiente ejemplo, *Retorno a comienzo de línea* es verdadero, es decir, la lista retorna:

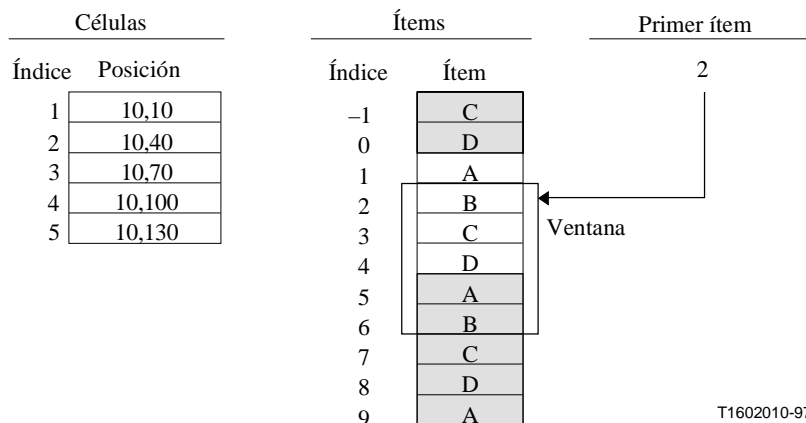


Figura 9/T.172 – Presentación de algunos ítems cuando Retorno a comienzo de línea es Verdadero

Lo importante en este caso es que hay dos células candidatas para presentar el ítem B. Como los objetos Visibles sólo pueden tener una posición, los ítems sólo pueden ser presentados en una célula. Si de acuerdo con la ventana un ítem puede ser presentado en más de una célula, será presentado solamente en la célula con el índice más bajo. En este caso, el ítem B será presentado en la célula 1. Los ítems C, D y A son presentados en las células 2, 3 y 4 respectivamente. La célula 5 permanece inutilizada. Obsérvese que cuando el número de ítems es inferior al número de células, las células permanecen inutilizadas, incluso si la lista retorna.

Por último, cualquier valor Entero está autorizado para el Primer ítem. En algunos casos, cuando Retorno a comienzo de línea es Falso, puede suceder que cada célula esté vacía; en este caso, cuando no se visualiza ningún objeto Visible en la ventana, está permitido.

- Entero.
- Valor inicial: 1.

30.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica. Además, se definen los siguientes eventos:

Primer ítem presentado (FirstItemPresented) Este evento es generado cada vez que cambia el estado de presentación del primer ítem en la *Lista de ítems*. El valor asociado reflejará el nuevo estado de presentación. El estado de presentación del ítem puede cambiar si cambia el atributo *Primer ítem*, o si cambia el número de ítems en la lista.

- Datos asociados, Booleanos: Verdadero si se presenta el ítem, Falso si no se presenta.

Último ítem presentado (LastItemPresented)	<p>Este evento es generado cada vez que cambia el estado de presentación del último ítem en la <i>Lista de ítems</i>. El valor asociado reflejará el nuevo estado de presentación. El estado de presentación del ítem puede cambiar si cambia el atributo <i>Primer ítem</i>, o si cambia el número de ítems en la lista.</p> <ul style="list-style-type: none"> Datos asociados, Booleanos: Verdadero si se presenta el ítem, Falso si no se presenta.
Ítems de cabecera (HeadItems)	<p>Este evento es generado cada vez que cambia el número de ítems en la <i>Lista de ítems</i> con índice en la gama [1, <i>Primer ítem</i> - 1]. Este número puede cambiar si cambia el atributo <i>Primer ítem</i> o si cambia el número de ítems en la lista.</p> <ul style="list-style-type: none"> Datos asociados, Entero: El número de ítems en la <i>Lista de ítems</i> con índice menor que el <i>Primer ítem</i>.
Ítems de cola (TailItems)	<p>Este evento es generado cada vez que cambia el número de ítems en la <i>Lista de ítems</i> con un índice en la gama [<i>Primer ítem</i>, número de ítems]. Este número puede cambiar si cambia el atributo <i>Primer ítem</i>, o si cambia el número de ítems en la lista.</p> <ul style="list-style-type: none"> Datos asociados, Entero: El número de ítems en la <i>Lista de ítems</i> con índice mayor que el <i>Primer ítem</i>.
Ítem seleccionado (ItemSelected)	<p>Este evento es generado si el <i>Estado de selección de un ítem</i> cambia a Verdadero.</p> <ul style="list-style-type: none"> Datos asociados, Entero: El índice del ítem.
Ítem deseleccionado (ItemDeselected)	<p>Este evento es generado si el <i>Estado de selección de un ítem</i> cambia a Falso.</p> <ul style="list-style-type: none"> Datos asociados, Entero: El índice del ítem.

30.3 Comportamientos internos

Esta clase define los siguientes comportamientos internos:

<i>Preparación</i>	<p>Ejecutar la siguiente secuencia de acciones:</p> <ol style="list-style-type: none"> Aplicar el comportamiento <i>Preparación</i> heredado de la clase básica. Añadir cada referencia enumerada en el atributo <i>Ítems de grupo de testigos</i> a la <i>Lista de ítems</i>. Si un objeto Visible es referenciado más de una vez en <i>Ítems de grupo de testigos</i>, se añade solamente una vez en la <i>Lista de ítems</i>.
<i>Destrucción</i>	<p>Ejecutar la siguiente secuencia de acciones:</p> <ol style="list-style-type: none"> Reiniciar todos los objetos Visibles del Grupo de listas a su Posición original. Aplicar el comportamiento <i>Destrucción</i> heredado de la clase básica.
<i>Activación</i>	<p>Ejecutar la siguiente secuencia de acciones:</p> <ol style="list-style-type: none"> Aplicar el comportamiento <i>Activación</i> heredado de la clase básica. Aplicar el comportamiento <i>Actualización</i>.

<i>Desactivación</i>	<p>Ejecutar la siguiente secuencia de acciones:</p> <ol style="list-style-type: none"> 1) Aplicar el comportamiento <i>Desactivación</i> a todos los objetos Visibles referenciados en la Lista de ítems. 2) Aplicar el comportamiento <i>Desactivación</i> definido en la clase básica.
<i>Actualización</i> (<i>Update</i>)	<p>Ejecutar la siguiente secuencia de acciones:</p> <ol style="list-style-type: none"> 1) Para cada ítem que se ha de presentar en una determinada célula, fijar su <i>Posición</i> (atributo interno) a la posición definida para esa célula. Después, si <i>Estado de ejecución</i> del Grupo de listas es verdadero, y el ítem está inactivo, aplicarle el comportamiento <i>Activación</i>. 2) Para cada ítem que no se ha de presentar, fijar su posición a su atributo posición original. Después, si <i>Estado de ejecución</i> del Grupo de listas es verdadero, y el ítem está activo, aplicarle el comportamiento <i>Desactivación</i>.
<i>Adición de ítem</i> (<i>Índice, Ítem</i>) [<i>Additem</i> (<i>Index, Item</i>)]	<p>Ejecutar la siguiente secuencia de acciones:</p> <ol style="list-style-type: none"> 1) Si el objeto Visible referenciado por el parámetro <i>Ítem</i> está ya en la <i>Lista de ítems</i>, omitir este comportamiento. 2) Si <i>Índice</i> es menor que 1 o mayor que el número actual de la Lista de ítems + 1, omitir este comportamiento. 3) Añadir la referencia del Ítem en la posición <i>Índice</i> en la <i>Lista de ítems</i>. (El ítem previo en la posición <i>Índice</i> en la Lista de ítems – si hubiere alguno – tendrá ahora el índice <i>Índice</i> + 1 y, de manera similar, cada ítem con un índice mayor que índice tendrá un <i>Índice</i> aumentado en 1.) 4) Aplicar el comportamiento <i>Actualización</i>.
<i>Supresión de ítem</i> (<i>Ítem</i>) [<i>Delitem</i> (<i>Item</i>)]	<p>Ejecutar las siguientes secuencias de acciones:</p> <ol style="list-style-type: none"> 1) Si el objeto visible referenciado por el parámetro <i>Ítem</i> no está en la <i>Lista de ítems</i>, omitir esta acción. 2) Suprimir la referencia indicada por el parámetro <i>Ítem</i> de la <i>Lista de ítems</i>. (El índice de ítems de los siguientes ítems en la lista de ítems disminuirá en 1.) 3) Fijar la posición del objeto Visible referenciado a su <i>Posición original</i>. 4) Aplicar el comportamiento <i>Actualización</i>.
<i>Seleccionar</i> (<i>Índice de ítems</i>) [<i>Select</i> (<i>ItemIndex</i>)]	<p>Ejecutar la siguiente secuencia de acciones:</p> <ol style="list-style-type: none"> 1) Si <i>Estado de selección de ítems</i> del ítem con índice <i>Índice de ítem</i> es Verdadero, omitir este comportamiento. 2) Si <i>Múltiple selección</i> es Falso, aplicar el comportamiento interno <i>Deselección(Índice)</i> para todos los ítems con índice <i>Índice</i>, para los cuales <i>Estado de selección de ítem</i> es Verdadero. 3) Fijar el <i>Estado de selección de ítem</i> del ítem con el índice <i>Índice de ítem</i> a Verdadero.

- 4) Generar un evento *Ítem seleccionado* con *Índice de ítem* como datos asociados.

Deseleccionar
(*Índice de ítem*)
[*Deselect*
(*ItemIndex*)]

Ejecutar la siguiente secuencia de acciones:

- 1) Si *Estado de selección de ítem* del ítem con índice *Índice de ítem* es Falso, omitir este comportamiento.
- 2) Fijar el *Estado de selección del ítem* con índice *Índice* a Falso.
- 3) Generar un evento *Ítem deseleccionado* con *Índice* como datos asociados.

30.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se definen las siguientes acciones MHEG-5 aplicables:

Añadir ítem
(*Índice de ítem*,
Referencia visible)

Aplicar el comportamiento interno *Añadir ítem*(*Índice de ítem*, *referencia visible*), para añadir la referencia al objeto Visible, indicado por el parámetro *Referencia visible* en la posición indicada por el parámetro *Índice de ítems*.

[*AddItem*
(*ItemIndex*,
VisibleReference)]

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Grupo de listas disponible.
- *Referencia visible* hará referencia a un objeto Visible disponible, que no está referenciado por ninguna referencia en la *Lista de ítems*.

Descripción de sintaxis:

AddItem	-->	Target ,
		ItemIndex ,
		VisibleReference
Target	-->	GenericObjectReference
ItemIndex	-->	GenericInteger
VisibleReference	-->	GenericObjectReference

Suprimir ítem
(*Referencia visible*)

Aplicar el comportamiento interno *Suprimir ítem*(*Referencia visible*) para suprimir la referencia al objeto visible, indicado por el parámetro *Referencia visible* del atributo Lista de ítems, si se produce.

[*DelItem*
(*VisibleReference*)]

Disposición de uso:

- El objeto *Objetivo* será un objeto Grupo de listas disponible.

Descripción de sintaxis:

DelItem	-->	Target ,
		VisibleReference
Target	-->	GenericObjectReference
VisibleReference	-->	GenericObjectReference

Obtener ítem de lista
(*Índice de ítem*,
Variable de referencia de ítem)
[GetListItem
(*ItemIndex*,
ItemRefVar)]

Devolver la referencia incluida en el atributo *Lista de ítems* con el índice especificado por el parámetro *Índice de ítem* en la Variable referencia de objeto referenciada por *Variable referencia de ítem*.

Si *Retorno a comienzo de línea* es Falso, y si *Índice de ítem* especifica un índice menor que 1 o mayor que el número de ítems en el atributo *Lista de ítems*, omitir la acción.

Si *Retorno a comienzo de línea* es verdadero y si *Índice de ítem* especifica un índice menor que 1 o mayor que el número de ítems en el atributo *Lista de ítems*, *Índice de ítems* será interpretado módulo del número de ítems en el atributo *Lista de ítems*.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Grupo de listas disponible.
- *Variable referencia de ítem* hará referencia a un objeto Variable de referencia de objeto activo.
- Si *Retorno a comienzo de línea* es Falso, *Índice de ítem* estará en la gama [1, Número de ítems].

Descripción de sintaxis:

GetListItem	-->	Target , ItemIndex , ItemRefVar
Target	-->	GenericObjectReference
ItemIndex	-->	GenericInteger
ItemRefVar	-->	ObjectReference

Obtener ítem de célula
(*Índice de célula*,
Variable referencia de ítem)
[GetCellItem
(*CellIndex*,
ItemRefVar)]

Devolver la referencia del objeto visible presentado en la célula indicada por el parámetro *Índice de células* en la Variable referencia de objeto referenciada por *Variable referencia de ítem*.

Si *Índice de célula* especifica un índice menor o igual a 1, devolver la referencia del objeto visible presentado en la primera célula. Si *Índice de célula* especifica un índice mayor o igual al número de células, devolver la referencia del objeto visible presentado en la última célula. Si no se presenta ningún objeto visible en la célula indicada, devolver *NULO*.

NOTA – La acción Obtener posición de testigo puede ser utilizada para obtener el índice de la célula que actualmente tiene el testigo, definida en la clase básica.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Grupo de listas disponible.
- *Variable de referencia de ítem* hará referencia a un objeto Variable de referencia de objeto activo.

Descripción de sintaxis:

GetCellItem	-->	Target , CellIndex , ItemRefVar
Target	-->	GenericObjectReference
CellIndex	-->	GenericInteger
ItemRefVar	-->	ObjectReference

Obtener estado de ítem
(Índice de ítem, Variable estado de ítem)

[GetItemStatus
(ItemIndex, ItemStatusVar)]

Devolver el valor del atributo *Estado de selección de ítem* del ítem en la *Lista de ítems* con índice *Índice de ítems* en la Variable booleana referenciada por *Variable de estado de ítem*.

Si *Retorno a comienzo de línea* es Falso, y si *Índice de ítems* especifica un índice menor que 1 o mayor que el número de ítems en el atributo *Lista de ítems*, omitir la acción.

Si *Retorno a comienzo de línea* es Verdadero, y si *Índice de ítems* especifica un índice menor o mayor que el número de ítems en el atributo *Lista de ítems*, el *Índice de ítems* será interpretado módulo del número de ítems en el atributo *Lista de ítems*.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Grupo de listas disponible.
- *Variable referencia de ítem* hará referencia a un objeto Variable booleana activo.
- Si *Retorno a comienzo de línea* es Falso, *Índice de ítem* estará en la gama [1, Número de ítem].

Descripción de sintaxis:

GetItemStatus	-->	Target , ItemIndex , ItemStatusVar
Target	-->	GenericObjectReference
ItemIndex	-->	GenericInteger
ItemStatusVar	-->	ObjectReference

Seleccionar ítem
(Índice de ítem)

[SelectItem
(ItemIndex)]

Si *Retorno a comienzo de línea* es Falso, y si *Índice de ítem* especifica un índice menor que 1 o mayor que el número de ítems en el atributo *Lista de ítems*, omitir esta acción; en los demás casos:

Si *Retorno a comienzo de línea* es Verdadero y si *Índice de ítems* especifica un índice menor que 1 o mayor que el número de ítems en el atributo *Lista de ítems*, *Índice de ítem* será interpretado módulo del número de ítems en el atributo *Lista de ítems*.

Aplicar el comportamiento interno *Seleccionar(Índice de ítem)*.

Disposición de uso:

- El objeto *Objetivo* será un objeto Grupo de listas disponible.

Descripción de sintaxis:

SelectItem	-->	Target ,
		ItemIndex
Target	-->	GenericObjectReference
ItemIndex	-->	GenericInteger

Deseleccionar ítem
(Índice de ítem)

[DeselectItem
(ItemIndex)]

Si *Retorno a comienzo de línea* es Falso, y si *Índice de ítem* especifica un índice menor que 1 o mayor que el número de ítems en el atributo *Lista de ítems*, omitir esta acción; en los demás casos:

Si *Retorno a comienzo de línea* es Verdadero y si *Índice de ítems* especifica un índice menor que 1 o mayor que el número de ítems en el atributo *Lista de ítems*, *Índice de ítem* será interpretado módulo del número de ítems en el atributo *Lista de ítems*.

Aplicar el comportamiento interno *Deseleccionar*(índice de ítem).

Disposición de uso:

- El objeto *Objetivo* será un objeto Grupo de listas disponible.

Descripción de sintaxis:

DeselectItem	-->	Target ,
		ItemIndex
Target	-->	GenericObjectReference
ItemIndex	-->	GenericInteger

Bascular ítem
(Índice de ítem)

[ToggleItem
(ItemIndex)]

Si *Retorno a comienzo de línea* es Falso, y si *Índice de ítem* especifica un índice menor que 1 o mayor que el número de ítems en el atributo *Lista de ítems*, omitir esta acción; en los demás casos:

Si *Retorno a comienzo de línea* es Verdadero y si *Índice de ítems* especifica un índice menor que 1 o mayor el número de ítems en el atributo *Lista de ítems*, *Índice de ítems* será interpretado módulo del número de ítems en el atributo *Lista de ítems*.

Si *Estado de selección de ítem* del ítem indicado por *Índice de ítem* es Verdadero, aplicar el comportamiento interno *Deseleccionar*(Índice de ítem); en los demás casos, aplicar el comportamiento interno *Seleccionar*(Índice de ítem).

Disposición de uso:

- El objeto *Objetivo* será un objeto Grupo de listas disponible.

Descripción de sintaxis:

ToggleItem	-->	Target ,
		ItemIndex,
Target	-->	GenericObjectReference
ItemIndex	-->	GenericInteger

Desfilar ítems
(*Ítems que han de desfilar*)

[ScrollItems
(ItemsToScroll)]

Añadir *Ítems que han de desfilar* al atributo *Primer ítem*, y aplicar el comportamiento *Actualizar*.

Disposición de uso:

- El objeto *Objetivo* será un objeto Grupo de listas disponible.

Descripción de sintaxis:

ScrollItems	-->	Target ,
		ItemsToScroll
Target	-->	GenericObjectReference
ItemsToScroll	-->	GenericInteger

Fijar primer ítem
(*Nuevo primer ítem*)

[SetFirstItem
(NewFirstItem)]

Fijar el valor del atributo *Primer ítem* a *Nuevo primer ítem*. Aplicar el comportamiento *Actualizar*.

Disposición de uso:

- El objeto *Objetivo* será un objeto *Grupo* de listas disponible.

Descripción de sintaxis:

SetFirstItem	-->	Target ,
		NewFirstItem
Target	-->	GenericObjectReference
NewFirstItem	-->	GenericInteger

Obtener primer ítem
(*Variable primer ítem*)

[GetFirstItem
(FirstItemVar)]

Devolver el valor actual del atributo *Primer ítem* en la *Variable* entera referenciada por *Variable primer ítem*.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Grupo de listas disponible.
- *Variable primer ítem* hará referencia a un objeto *Variable* entero activo.

Descripción de sintaxis:

GetFirstItem	-->	Target ,
		FirstItemVar
Target	-->	GenericObjectReference
FirstItemVar	-->	ObjectReference

Obtener tamaño de lista
(*Variable tamaño*)

Devolver el número de ítems en la *Lista de ítems* en la *Variable* entera referenciada por *Variable tamaño*.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Grupo de listas disponible.
- *Variable tamaño* hará referencia a un objeto *Variable* entera activo.

[GetListSize
(SizeVar)]

Descripción de sintaxis:

GetListSize	-->	Target , SizeVar
Target	-->	GenericObjectReference
SizeVar	-->	ObjectReference

30.5 Descripción formal

ListGroup Class	-->	TokenGroup Class , Positions , WrapAround? , MultipleSelection?
Positions	-->	Position*
Position	-->	XYposition ,
WrapAround	-->	BOOLEAN
MultipleSelection	-->	BOOLEAN

31 Clase Visible

Descripción: Define el comportamiento de objetos Presentables que tienen una representación visual en la pantalla

Clase básica: Presentable

Subclases: Vídeo, Gráficos RT, Diagrama de bits, Arte lineal, Texto, Cursor, Botón

Estado: Clase abstracta

31.1 Atributos

Esta subcláusula define los atributos heredados, intercambiados e internos para esta clase.

31.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con semántica idéntica.

31.1.2 Atributos intercambiados propios

Esta clase define los siguientes atributos intercambiados adicionales:

Tamaño de casilla original
(OriginalBoxSize)

Tamaño original de la casilla del objeto Visible con respecto al sistema de coordenadas de la Escena. El tamaño viene dado por un par de Enteros que representan respectivamente el ancho (longitud a lo largo del eje X) y altura (longitud a lo largo del eje Y) de la casilla.

No se utilizarán valores negativos. Se autorizan valores que excedan del sistema de coordenadas de la escena, en cuyo caso sólo se reproducirá la parte del objeto Visible dentro del sistema de coordenadas de la escena.

- Par de Enteros (Tamaño de casilla X, Tamaño de casilla Y).

Posición original
(OriginalPosition) Posición original de la esquina izquierda superior del objeto Visible con respecto al sistema de coordenadas de la Escena.

Éste es un conjunto de coordenadas (X, Y) que expresan la posición original de la esquina izquierda superior del objeto visible en el sistema de coordenadas de la Escena actualmente activa.

Se permiten valores negativos y valores que excedan del sistema de coordenadas de la escena, en cuyo caso sólo se reproducirá la parte del objeto Visible dentro del sistema de coordenadas de la escena.

- Par facultativo de Enteros (Posición X, Posición Y).
- Valor por defecto: (0,0).

Referencia de paleta original
(OriginalPaletteRef) Indicar el objeto Paleta inicial que será utilizado para reproducir colores del objeto Visible.

- Referencia facultativa a un objeto Paleta.
- Valor por defecto: Ninguna paleta.

Este atributo es obligatorio cuando se fijan otros atributos de color y necesitan referencias de Paleta.

31.1.3 Atributos internos propios

Esta clase define los siguientes atributos internos adicionales:

Tamaño de casilla
(BoxSize) Tamaño de la casilla delimitante del objeto Visible con respecto al sistema de coordenadas de la Escena. No se reproducirán las partes del objeto visible que caen fuera de los límites de la casilla. Si el objeto Visible no abarca completamente el contenido de la casilla, las partes restantes de la casilla serán transparentes.

- Par de Enteros (Tamaño de casilla X, Tamaño de casilla Y).
- Valor inicial: Valor del atributo Tamaño de casilla original.

Posición
(Position) Posición de la esquina izquierda superior del objeto Visible con respecto al sistema de coordenadas en la Escena. Cuando se reproduce, el objeto Visible tendrá su esquina superior izquierda en esta posición.

- Par de Enteros (Posición X, Posición Y).
- Valor inicial: Valor del atributo Posición original.

<i>Referencia de paleta</i> (<i>PaletteRef</i>)	<p>Si se utiliza una paleta, la referencia al objeto Paleta que será utilizado para reproducir colores del objeto Visible.</p> <p>El atributo <i>Paleta</i> no se definirá para clases que especifican que referencia de paleta original no será codificada.</p> <ul style="list-style-type: none"> • Referencia facultativa a un objeto Paleta. • Valor inicial: Valor del atributo Referencia de paleta original.
--	---

31.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

31.3 Comportamientos internos

Esta clase tiene los mismos comportamientos internos que su clase básica, con los siguientes cambios de semántica:

<i>Preparación</i>	<p>Ejecutar la siguiente secuencia de acciones:</p> <ol style="list-style-type: none"> 1) Ejecutar los pasos 1), 2) y 3) del comportamiento <i>Preparación</i> definido en la clase Raíz. 2) Si el objeto Visible no es referenciado en <i>Pila de visualización</i> del objeto Aplicación activo, añadir una referencia a este objeto Visible en la parte superior de la <i>Pila de visualización</i>. 3) Ejecutar los pasos 4), 5) y 6) del comportamiento <i>Preparación</i> definido en la clase Raíz.
<i>Dstrucción</i>	<p>Ejecutar la siguiente secuencia de acciones:</p> <ol style="list-style-type: none"> 1) Si el objeto Visible es referenciado en <i>Pila de visualización</i> del objeto Aplicación activo, suprimir la referencia a este objeto Visible de la <i>Pila de visualización</i>. 2) Ejecutar el comportamiento <i>Dstrucción</i> definido en la clase básica.
<i>Activación</i>	<p>Ejecutar la siguiente secuencia de acciones:</p> <ol style="list-style-type: none"> 1) Ejecutar el comportamiento <i>Activación</i> definido en la clase básica. 2) Visualizar el objeto Visible de acuerdo con su posición en la <i>Pila de visualización</i> y con la <i>Posición</i> y la casilla delimitante definidas por los atributos posición y <i>Tamaño de casilla</i>. 3) Fijar el <i>Estado de ejecución</i> a Verdadero y generar un evento <i>Está funcionando</i>.
<i>Desactivación</i>	<p>Ejecutar la siguiente secuencia de acciones:</p> <ol style="list-style-type: none"> 1) Si el atributo Estado de ejecución del objeto es Falso, omitir el comportamiento; en los demás casos: 2) Detener la visualización del objeto Visible. 3) Ejecutar el comportamiento <i>Desactivación</i> definido en la clase básica.

31.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se definen las siguientes acciones MHEG-5 aplicables:

Fijar posición
(*Nueva posición X*,
Nueva posición Y)
[SetPosition
(*NewXPosition*,
NewYPosition)]

Cambiar la posición del objeto Visible objetivo.
Ejecutar la siguiente secuencia de acciones:

- 1) Fijar el atributo *Posición* de acuerdo con *Nueva posición X* y *Nueva posición Y*.
- 2) Si el objeto Visible está activo, dibujar de nuevo el accesorio gráfico que representa el objeto en la pantalla en la casilla delimitante definida por los atributos con *Tamaño de casilla* y *Posición* y de acuerdo con su posición en la *Pila de visualización* del objeto Aplicación activo.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Visible disponible.
- *Nueva posición X* y *Nueva posición Y* corresponderán a una posición interpretada en el sistema de coordenadas de la Escena definida por el atributo Sistema de coordenadas de la escena actualmente activa.

Descripción de sintaxis:

SetPosition	-->	Target , NewXPosition , NewYPosition
Target	-->	GenericObjectReference
NewXPosition	-->	GenericInteger
NewYPosition	-->	GenericInteger

Obtener posición
(*Variable posición X*,
variable posición Y)
[GetPosition
(*XPositionVar*,
YPositionVar)]

Devuelve la posición del objeto Visible objetivo.
Fijar las Variables referenciadas por *Variable posición X* y *Variable posición Y* al valor de las posiciones X e Y del objeto Visible objetivo, respectivamente.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Visible disponible.
- *Variable posición X* y *Variable posición Y* harán referencia a objetos Variables enteras activos.

Descripción de sintaxis:

GetPosition	-->	Target , XPositionVar , YPositionVar
Target	-->	GenericObjectReference
XpositionVar	-->	ObjectReference
YpositionVar	-->	ObjectReference

Fijar tamaño de casilla
(*Nuevo tamaño de casilla X*,
Nuevo tamaño de casilla Y)
[SetBoxSize
(*XNewBoxSize*,
YnewBoxSize)]

Cambiar el tamaño de la casilla delimitante del objeto Visible objetivo.

Ejecutar la siguiente secuencia de acciones:

- 1) Fijar el atributo *Tamaño de casilla*.
- 2) Dibujar de nuevo el accesorio gráfico que representa el objeto en la pantalla en la casilla delimitante definida por los atributos *Tamaño de casilla* y *Posición* y de acuerdo con su posición en la *Pila de visualización* del objeto Aplicación activo.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Visible disponible.
- *Nuevo tamaño de casilla X* y *Nuevo tamaño de casilla Y* serán valores positivos y diferentes de 0.

Descripción de sintaxis:

SetBoxSize	-->	Target , XNewBoxSize , YNewBoxSize
Target	-->	GenericObjectReference
XNewBoxSize	-->	GenericInteger
YNewBoxSize	-->	GenericInteger

Obtener tamaño de casilla
(*Variable tamaño de casilla X*,
Variable tamaño de casilla Y)
[GetBoxSize
(*XBox-SizeVar*,
YBox-SizeVar)]

Devuelve el tamaño de la casilla delimitante del objeto Visible objetivo.

Fijar las variables referenciadas por *Variable tamaño de casilla X* y *Variable tamaño de casilla Y* al valor de las posiciones X e Y del objeto Visible objetivo, respectivamente.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Visible disponible.
- *Variable tamaño de casilla X* y *Variable tamaño de casilla Y* harán referencia a objetos Variables enteras activos.

Descripción de sintaxis:

GetBoxSize	-->	Target , XBoxSizeVar , YBoxSizeVar
Target	-->	GenericObjectReference
XBoxSizeVar	-->	ObjectReference
YBoxSizeVar	-->	ObjectReference

Traer hacia
adelante
(BringToFront)

Poner un objeto Visible en el primer plano de la pantalla.

Ejecutar la siguiente secuencia de acciones:

- 1) Si el objeto Visible objetivo no está referenciado en la *Pila de visualización* del objeto Aplicación activo, omitir esta acción.
- 2) Si el objeto Visible objetivo está referenciado en la *Pila de visualización* del objeto Aplicación activo, mover la referencia a este objeto Visible en la parte superior de la *Pila de visualización*.
- 3) Si el objeto Visible objetivo está activo, dibujar de nuevo el accesorio gráfico que representa el objeto en la pantalla en la casilla delimitante definida por los atributos *Tamaño de casilla* y *posición* y de acuerdo con su posición en la *Pila de visualización* del objeto Aplicación activo.

Disposición de uso:

- El objeto *Objetivo* será un objeto Visible disponible.

Descripción de sintaxis:

BringToFront	-->	Target
Target	-->	GenericObjectReference

Enviar hacia atrás
(SendToBack)

Poner un objeto Visible en el fondo de la pantalla.

Ejecutar la siguiente secuencia de acciones:

- 1) Si el objeto Visible objetivo no está referenciado en la *Pila de visualización* del objeto Aplicación activo, omitir esta acción.
- 2) Si el objeto Visible objetivo está referenciado en la *Pila de visualización* del objeto Aplicación activo, mover la referencia a este objeto Visible en la parte inferior de la *Pila de visualización*.
- 3) Si el objeto Visible objetivo está activo, dibujar de nuevo el accesorio gráfico que representa el objeto en la pantalla en la casilla delimitante definida por los atributos *Tamaño de casilla* y *Posición* y de acuerdo con su posición en la *Pila de visualización* del objeto Aplicación activo.

Disposición de uso:

- El objeto *Objetivo* será un objeto Visible disponible.

Descripción de sintaxis:

SendToBack	-->	Target
Target	-->	GenericObjectReference

Poner delante
(Visible de
referencia)

Poner un objeto Visible exactamente enfrente de otro en la pila de visualización.

Ejecutar la siguiente secuencia de acciones:

[PutBefore
(ReferenceVisible)]

- 1) Si el objeto Visible objetivo no está referenciado en la *Pila de visualización* del objeto Aplicación activo, omitir esta acción.

- 2) Si el objeto Visible objetivo está referencia en la *Pila de visualización* del objeto Aplicación activo, mover la referencia a este objeto Visible en la posición exactamente delante de la referencia a *Visible de referencia* en la *Pila de visualización*.
- 3) Si el objeto Visible objetivo está activo, dibujar de nuevo el accesorio gráfico que representa el objeto en la pantalla en la casilla delimitante definida por los atributos *Tamaño de casilla* y *Posición* y de acuerdo con su posición en la *Pila de visualización* del objeto Aplicación activo.

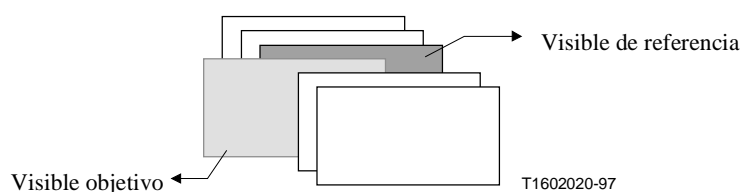


Figura 10/T.172 – Efecto de la acción Poner delante

Disposiciones de uso:

- El objeto *Objetivo* será una referencia a un objeto Visible disponible.
- *Visible de referencia* será una referencia a un objeto Visible disponible.
- La *Pila de visualización* del objeto Aplicación activo contendrá una referencia al objeto *Visible de referencia*.

Descripción de sintaxis:

PutBefore	-->	Target , ReferenceVisible
Target	-->	GenericObjectReference
ReferenceVisible	-->	GenericObjectReference

Poner detrás
(*Visible de referencia*)

[PutBehind
(*ReferenceVisible*)]

Poner un objeto Visible exactamente debajo de otro en la pila de visualización.

Ejecutar la siguiente secuencia de acciones:

- 1) Si el objeto Visible objetivo no está referenciado en la *Pila de visualización* del objeto Aplicación activo, omitir esta acción.
- 2) Si el objeto Visible objetivo está referencia en la *Pila de visualización* del objeto Aplicación activo, mover la referencia a este objeto Visible en la posición exactamente detrás de la referencia a *Visible de referencia* en la *Pila de visualización*.
- 3) Si el objeto Visible objetivo está activo, dibujar de nuevo el accesorio gráfico que representa el objeto en la pantalla en la casilla delimitante definida por los atributos *Tamaño de casilla* y *Posición* y de acuerdo con su posición en la *Pila de visualización* del objeto Aplicación activo.

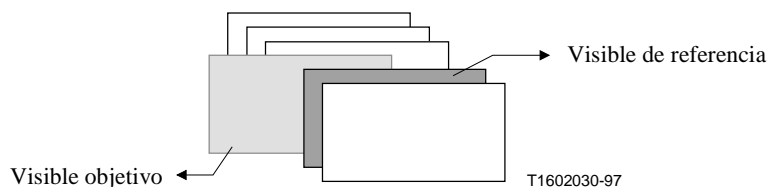


Figura 11/T.172 – Efecto de la acción Poner detrás

Disposiciones de uso:

- El objeto *Objetivo* será una referencia a un objeto Visible disponible.
- *Visible de referencia* se fijará o hará referencia a una referencia a un objeto Visible disponible.
- La *Pila de visualización* del objeto aplicación activo contendrá una referencia al objeto *Visible de referencia*.

Descripción de sintaxis:

PutBehind	-->	Target , ReferenceVisible
Target	-->	GenericObjectReference
ReferenceVisible	-->	GenericObjectReference

Fijar referencia de paleta
(*Nueva referencia de paleta*)

[SetPaletteRef
(*NewPaletteRef*)]

Cambiar la tabla de mejora de colores utilizada para reproducir colores del objeto Visible.

Ejecutar la siguiente secuencia de acciones:

- 1) Fijar *Referencia de paleta* del objeto Visible objetivo a *Nueva referencia de paleta*.
- 2) Si el objeto Visible objetivo está activo, dibujar de nuevo el objeto Visible teniendo en cuenta el nuevo valor de *Referencia de paleta*.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Visible disponible.
- *Nueva referencia de paleta* contendrá una referencia a un objeto Paleta activo.

Descripción de sintaxis:

SetPaletteRef	-->	Target , NewPaletteRef
Target	-->	GenericObjectReference
NewPaletteRef	-->	GenericObjectReference

31.5 Descripción formal

Visible Class	-->	Presentable class, OriginalBoxSize, OriginalPosition?, OriginalPaletteRef?
OriginalBoxSize	-->	XLength, Ylength
Xlength	-->	INTEGER
Ylength	-->	INTEGER
OriginalPosition	-->	XYPosition
OriginalPaletteRef	-->	ObjectReference

32 Clase Diagrama de bits

Descripción: Define el comportamiento de un conjunto bidimensional de pixels

Clase básica: Visible

Subclases: Ninguna

Estado: Clase concreta

32.1 Atributos

Esta subcláusula define los atributos heredados, intercambiados e internos para esta clase.

32.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las siguientes restricciones:

Nombre de atributo	Definido en	Restricciones y requisitos
Gancho de contenido	Ingrediente	Este atributo o su atributo por defecto correspondiente en la clase Aplicación (Gancho de contenido de diagrama de bits) es obligatorio para esta clase.
Contenido original	Ingrediente	Este atributo es obligatorio para esta clase.

32.1.2 Atributos intercambiados propios

Esta clase define los siguientes atributos intercambiados adicionales:

Enlosamiento (Tiling) Cuando es verdadero, este atributo indica que el diagrama de bits se debe repetir (enlosado) en el *Tamaño de casilla* disponible del diagrama de bits.

- Booleano facultativo.
- Valor por defecto: Falso.

Transparencia original (Original Transparency) Este atributo define la transparencia inicial de los pixels del diagrama de bits que están marcados como transparentes.

En los casos en que la representación codificada del propio diagrama de bits define el valor de Transparencia, la transferencia definida por la codificación del contenido y la transparencia definida por este atributo se deben combinar.

El algoritmo exacto no se define en la presente Recomendación.

- Entero facultativo en la gama [0, 100] (porcentaje).
- Valor por defecto: 0%.

32.1.3 Atributos internos propios

Esta clase define el siguiente atributo interno adicional:

Transparencia (<i>Transparency</i>)	Define la transparencia de los pixels del diagrama de bits que están marcados como transparentes.
	<ul style="list-style-type: none"> • Entero facultativo en la gama [0, 100] (porcentaje). • Valor por inicial: Valor del atributo Transparencia original.

32.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

32.3 Comportamientos internos

Esta clase tiene los mismos comportamientos internos que su clase básica, con semántica idéntica.

32.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además se definen las siguientes acciones MHEG-5 aplicables.

Escala de diagrama de bits (<i>Escala X, escala Y</i>)	Si el motor MHEG-5 implementa la opción Escala, el efecto de esta acción es que se aplica una escala al contenido del Diagrama de bits de acuerdo con el tamaño (<i>Escala X, Escala Y</i>). Los motores que no implementan la acción de escala omitirán esta acción.
---	---

[ScaleBitmap (<i>Xscale, YScale</i>)]	Obsérvese que esta acción no afecta al atributo interno <i>Tamaño de casilla</i> del objeto diagrama de bits; en otras palabras, se aplica la escala al Diagrama de bits, pero su casilla permanece igual.
--	--

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Diagrama de bits disponible.
- *Escala X* y *Escala Y* serán Enteros positivos.

Descripción de sintaxis:

ScaleBitmap	-->	Target ,
		XScale, Yscale
Target	-->	GenericObjectReference
XScale, Yscale	-->	GenericInteger

Fijar transparencia (Nueva transparencia)	Ejecutar la siguiente secuencia de acciones:
	1) Cambiar el valor del atributo <i>Transparencia</i> .
[SetTransparency (<i>NewTransparency</i>)]	2) Si el diagrama de bits está activo, dibujarlo de nuevo utilizando el nuevo valor del atributo <i>Transparencia</i> de acuerdo con su posición en la <i>Pila de visualización</i> del objeto Aplicación activo.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Diagrama de bits disponible.
- *Nueva transparencia* se fija dentro de la gama [0, 100].

Descripción de sintaxis:

ScaleBitmap	-->	Target , NewTransparency
Target	-->	GenericObjectReference
NewTransparency	-->	GenericInteger

32.5 Descripción formal

Bitmap Class	-->	Visible Class , Tiling? , OriginalTransparency?
Tiling	-->	BOOLEAN
OriginalTransparency	-->	INTEGER

33 Clase Arte lineal

Descripción: Define la funcionalidad asociada con la representación vectorial de objetos gráficos

Clase básica: Visible

Subclases: Rectángulo, Arte lineal dinámico

Estado: Clase concreta

33.1 Atributos

Esta subcláusula define los atributos heredados, intercambiados e internos para esta clase.

33.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las siguientes restricciones:

Nombre de atributo	Definido en	Restricciones y requisitos
Gancho de contenido	Ingrediente	Este atributo o sus atributos por defecto correspondiente en la clase Aplicación (Contenido de gancho de arte lineal) es obligatorio para esta clase.
Contenido original	Ingrediente	Este atributo es obligatorio para esta clase.

33.1.2 Atributos intercambiados propios

Esta clase define los siguientes atributos intercambiados adicionales:

Casilla delimitante con bordes (Bordered-BoundingBox)	El atributo Casilla delimitante con bordes determina si la casilla delimitante – definida por los atributos Tamaño de casilla y Posición – está delimitada o no por líneas. Cuando este atributo se pone a Verdadero, todos los otros dibujos estarán dentro de los bordes y no
--	---

encima de ellos. La Anchura de línea original, el Estilo de línea original y el Color de línea de referencia original se utilizará para presentar la casilla delimitante.

- Atributo booleano facultativo.
- Valor por defecto: Verdadero.

Anchura de línea original
(OriginalLineWidth)

El atributo Anchura de línea original determina la anchura de línea original del objeto gráfico que incluye la casilla delimitante.

El atributo Anchura de banda original se expresa en pixels en el espacio de coordenadas de la escena. Se especifica en altura de pixels para líneas horizontales y en anchura de pixels para líneas verticales.

La resolución y la precisión de presentación reales del motor MHEG-5 está fuera del alcance de la presente Recomendación.

- Atributo Entero facultativo.
- Valor por defecto: 1.

Estilo de línea original
(OriginalLineStyle)

El atributo Estilo de línea original determina el esquema utilizado para reproducir las Líneas del objeto arte lineal.

- Atributo Entero facultativo – Uno de 1, 2 ó 3.
1 significa trazo continuo, 2 significa trazo interrumpido, 3 significa trazo con puntos.
- Valor por defecto: 1 (trazo continuo).

Color de línea de referencia original
(OriginalRefLineColour)

Color de referencia inicial para las líneas utilizadas para reproducir el objeto Arte lineal.

El valor del atributo Color de línea de referencia original se expresa como una cadena que representa un valor de color absoluto o como un entero que representa un índice basado en cero en una tabla de mejora de colores. En el último caso, el atributo Referencia de paleta original debe contener una referencia a un objeto Paleta que es utilizado para traducir el índice a un valor de color real.

La codificación de valores de colores absolutos así como la resolución de color real en el proceso de presentación están fuera del alcance de la presente Recomendación.

- Atributo facultativo.
- Valor por defecto "negro".
La aparición exacta en la pantalla del valor por defecto "negro" no se especifica plenamente en esta subcláusula.
- Cadena de octetos o entero.

Color de relleno de referencia original (OriginalRefFillColour)	<p>Color de referencia inicial para el interior de un objeto Arte lineal cerrado.</p> <p>El valor del atributo Color de relleno de referencia original se expresa como una cadena que representa un valor de color absoluto o como un entero que representa un índice basado en cero en una tabla de mejora de colores. En el último caso, el atributo Referencia de paleta original debe contener una referencia a un objeto Paleta que es utilizado para producir el índice a un valor de color real.</p> <p>Si Color de relleno de referencia original no está codificado, un objeto Arte lineal que representa una forma cerrada se reproducirá con un color de relleno transparente.</p> <p>La codificación de valores de colores absolutos así como la resolución de color real en el proceso de reproducción están fuera del alcance de la presente Recomendación.</p> <ul style="list-style-type: none"> • Atributo facultativo. • Valor por defecto: "transparente". • Cadena de octetos o Entero.
--	--

33.1.3 Atributos internos propios

Esta clase define los siguientes atributos internos adicionales:

<i>Anchura de línea</i> (LineWidth)	<p>El atributo <i>Anchura de línea</i> determina la anchura de línea de los objetos gráficos que son dibujados por las acciones gráficas.</p> <ul style="list-style-type: none"> • Entero. • Valor inicial: Anchura de línea original.
<i>Estilo de línea</i> (LineStyle)	<p>Anchura de la línea que el objeto de Arte lineal ha de reproducir.</p> <ul style="list-style-type: none"> • Uno de 1, 2 ó 3. 1 significa trazo continuo, 2 significa trazo interrumpido; 3 significa trazo con puntos. • Valor inicial: Estilo de línea original.
<i>Color de línea de referencia</i> (RefLineColour)	<p>Color de referencia para las líneas utilizadas para reproducir el objeto Arte lineal.</p> <ul style="list-style-type: none"> • Cadena de octetos o Entero. • Valor inicial: Color de línea de referencia original.
<i>Color de relleno de referencia</i> (RefFillColour)	<p>Color de referencia para el color de fondo utilizado para reproducir el objeto Arte lineal.</p> <ul style="list-style-type: none"> • Cadena de octetos o Entero. • Valor inicial: Color de relleno de referencia original.

33.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

33.3 Comportamientos internos

Esta clase no define atributos internos adicionales.

33.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se definen las siguientes acciones MHEG-5 aplicables.

Fijar anchura de línea
(*Nueva anchura de línea*)
[SetLineWidth
(*NewLineWidth*)]

Cambiar la anchura de las líneas de un objeto Arte lineal.
Ejecutar la siguiente secuencia de acciones:

- 1) Fijar el valor del atributo Anchura de línea a *Nueva anchura de línea*.
- 2) Si el objeto Arte lineal objetivo está activo, dibujar de nuevo inmediatamente el objeto objetivo teniendo en cuenta el nuevo valor del atributo Anchura de línea y de acuerdo en su posición en la *Pila de visualización* del objeto Aplicación activo.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Arte lineal disponible.
- *Nueva anchura de línea* se fijará o hará referencia a un valor entero positivo.

Descripción de sintaxis:

SetLineWidth	-->	Target , NewLineWidth
Target	-->	GenericObjectReference
NewLineWidth	-->	GenericInteger

Fijar estilo de línea
(*Nuevo estilo de línea*)
[SetLineStyle
(*NewLineStyle*)]

Cambiar el estilo de línea de un objeto Arte lineal.
Ejecutar la siguiente secuencia de acciones:

- 1) Fijar el valor del atributo Estilo de línea a *Nuevo estilo de línea*.
- 2) Si el objeto Arte lineal objetivo está activo, dibujar de nuevo inmediatamente el objeto objetivo teniendo en cuenta el nuevo valor del atributo Estilo de línea y de acuerdo en su posición en la *Pila de visualización* del objeto Aplicación activo.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Arte lineal disponible.
- *Nuevo estilo de línea* se fijará o hará referencia a un valor entero positivo: 1, 2 ó 3.

Descripción de sintaxis:

SetLineStyle	-->	Target , NewLineStyle
Target	-->	GenericObjectReference
NewLineStyle	-->	GenericInteger

Fijar color de línea
(*Nuevo color*)

[SetLineColour
(*NewColour*)]

Cambiar el color de las líneas de un objeto a Arte lineal.

Ejecutar la siguiente secuencia de acciones:

- 1) Fijar el valor del atributo Color de línea de referencia a *Nuevo color*.
- 2) Si el objeto objetivo está activo, dibujar de nuevo inmediatamente el objeto objetivo teniendo en cuenta el nuevo valor del atributo Anchura de línea y de acuerdo con su posición en la *Pila de visualización* del objeto Aplicación activo.

Nuevo color puede ser un valor de color absoluto o el índice basado en cero de un color de la tabla referenciada por el atributo *Referencia de paleta*.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Arte lineal disponible.
- Si Color de línea de referencia está fijado actualmente a un valor de color absoluto, *Nuevo color* se fijará o hará referencia a un valor de color absoluto.
- Si Color de línea de referencia está puesto actualmente a un índice de una tabla de mejora de colores, *Nuevo color* se fijará o hará referencia a un índice de una tabla de mejora de colores.

Descripción de sintaxis:

SetLineColour	-->	Target , NewColour
Target	-->	GenericObjectReference
NewColour	-->	NewColourIndex NewAbsoluteColour
NewColourIndex	-->	GenericInteger
NewAbsoluteColour	-->	GenericOctetString

Fijar color de
relleno
(*Nuevo color*)

[SetFillColour
(*NewColour*)]

Cambiar el color de relleno de un objeto Arte lineal.

Ejecutar la siguiente secuencia de acciones:

- 1) Fijar el valor del atributo Color de relleno de referencia a *Nuevo color*.
- 2) Si el objeto objetivo está activo, dibujar de nuevo inmediatamente el objeto objetivo teniendo en cuenta el nuevo valor del atributo Color de relleno de referencia y de acuerdo con su posición en la *Pila de visualización* del objeto Aplicación activo.

Nuevo color puede ser un valor de color absoluto o el índice basado en cero de un color en la tabla referenciada por el atributo *Referencia de paleta*.

Si *Nuevo color* no está codificado, el atributo Color de relleno de referencia se pondrá a "transparente".

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Arte lineal disponible.
- Si *Color de relleno de referencia* no está fijado, *Nuevo color* se fijará o hará referencia a un valor de color absoluto.

- Si Color de relleno de referencia está fijado a un valor de color absoluto, *Nuevo color* se fijará o hará referencia a un valor de color absoluto.
- Si Color de relleno de referencia está fijado a un índice de una tabla de mejora de colores, *Nuevo color* se fijará o hará referencia a un índice de una tabla de mejora de colores.

Descripción de sintaxis:

SetFillColour	-->	Target , NewColour?
Target	-->	GenericObjectReference
NewColour	-->	NewColourIndex NewAbsoluteColour
NewColourIndex	-->	GenericInteger
NewAbsoluteColour	-->	GenericOctetString

33.5 Descripción formal

LineArt Class	-->	Visible Class , BorderedBoundingBox? , OriginalLineWidth? , OriginalLineStyle? , OriginalRefLineColour? , OriginalRefFillColour?
BorderedBoundingBox	-->	BOOLEAN
OriginalLineWidth	-->	INTEGER
OriginalLineStyle	-->	
OriginalRefLineColour	-->	Colour
OriginalRefFillColour	-->	Colour

34 Clase Rectángulo

Descripción: Define una estructura de datos que trata rectángulos

Clase básica: Arte lineal

Subclases: Ninguna

Estado: Clase concreta

34.1 Atributos

Esta subcláusula define los atributos heredados, intercambiados e internos para esta clase.

34.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las siguientes restricciones:

Nombre de atributo	Definido en	Restricciones y requisitos
Gancho de contenido	Ingrediente	Este atributo no se codificará para esta clase.
Contenido original	Ingrediente	Este atributo no se codificará para esta clase.
<i>Tamaño de casilla, Posición</i>	Visible	Además de definir la casilla delimitante para el Rectángulo, estos atributos definen también el tamaño del propio Rectángulo. El tamaño del Rectángulo será tal que encaje precisamente dentro de su casilla delimitante. Esto significa que los bordes del Rectángulo ocupan pixels de <i>Anchura de línea</i> en el interior de la casilla delimitante del Rectángulo.
Casilla delimitante con bordes	Arte lineal	Este atributo no se codificará para esta clase. Se interpretará siempre como Verdadero. A diferencia de su clase progenitora, la Casilla delimitante con bordes de un rectángulo (es decir, la forma del Rectángulo) se dibujará utilizando Anchura de línea, Estilo de línea y Color de línea de referencia, en vez de Anchura de línea original, Estilo de línea original y Color de línea de referencia original.

34.1.2 Atributos intercambiados propios

Esta clase no define atributos intercambiados adicionales.

34.1.3 Atributos internos propios

Esta clase no define atributos internos adicionales.

34.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

34.3 Comportamientos internos

Los comportamientos internos de esta clase son iguales que los de su clase básica, con semántica idéntica.

34.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica.

34.5 Descripción formal

Rectangle Class	-->	LineArt Class
-----------------	-----	---------------

35 Clase Arte lineal dinámica

Descripción: Define los medios para dibujar dinámicamente objetos gráficos vectoriales

Clase básica: Arte lineal

Subclases: Ninguna

Estado: Clase concreta

35.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

35.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las siguientes restricciones:

Nombre de atributo	Definido en	Restricciones y requisitos
Gancho de contenido	Ingrediente	Este atributo no se codificará para esta clase.
Contenido original	Ingrediente	Este atributo no se codificará para esta clase.

35.1.2 Atributos intercambiados propios

Esta clase no define atributos intercambiados adicionales.

35.1.3 Atributos internos propios

Esta clase no define atributos internos adicionales.

35.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

35.3 Comportamientos internos

Esta clase no define comportamientos internos.

35.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se definen las siguientes acciones MHEG-5 aplicables:

Fijar tamaño de casilla (SetBoxSize)	Esta acción dibuja de nuevo el objeto Arte lineal dinámico con su nuevo tamaño, liberado de todos sus dibujos anteriores, y se rellena con <i>Color de relleno de referencia original</i> (en el caso particular cuando el nuevo tamaño es exactamente el antiguo, esta acción es similar a una acción Liberar). Cuando Casilla delimitante con bordes se pone a Verdadero, los bordes se dibujan de nuevo.
Fijar posición, Traer hacia adelante, Enviar hacia atrás, Poner delante, Poner detrás (SetPosition, BringToFront, SendToBack, PutBefore, PutBehind)	Después de cada una de estas acciones, el accesorio Arte lineal dinámico se dibuja de nuevo, es liberado de todos sus dibujos anteriores, y se rellena con <i>Color de relleno de referencia original</i> . Cuando Casilla delimitante con bordes se pone a Verdadero, los bordes se dibujan de nuevo.

Fijar anchura de línea
(SetLineWidth) Fijar un nuevo valor de *Anchura de línea* no modificará la anchura de línea de todos los gráficos existentes del objeto Arte lineal dinámico, se utilizará para presentar los siguientes gráficos que se han de dibujar.

Fijar estilo de línea
(SetLineStyle) Fijar un nuevo valor de *Estilo de línea* no modificará el estilo de línea de todos los gráficos existentes del objeto Arte lineal dinámico, se utilizará para presentar los siguientes gráficos que se han de dibujar.

Fijar color de línea
(SetLineColour) Fijar un nuevo valor de *Color de línea de referencia* no modificará el estilo de línea de todos los gráficos existentes del objeto Arte lineal dinámico, se utilizará para presentar los siguientes gráficos que se han de dibujar.

Fijar color de relleno
(SetFillColour) Fijar un nuevo valor de *Color de relleno de referencia* no modificará el estilo de línea de todos los gráficos existentes del objeto Arte lineal dinámico, se utilizará para presentar los siguientes gráficos que se han de dibujar

Obtener anchura de línea
(*Variable anchura de línea*)
[GetLineWidth
(*LineWidthVar*)] Devuelve la *Anchura de línea* actual en la *Variable anchura de línea*.
Disposiciones de uso:

- *Variable anchura de línea* hará referencia a un objeto Variable entera activo.
- El objeto *Objetivo* será un objeto Arte lineal dinámico disponible.

Descripción de sintaxis:

GetLineWidth	-->	Target ,
		LineWidthVar
Target	-->	GenericObjectReference
LineWidthVar	-->	ObjectReference

Obtener estilo de línea
(*Variable estilo de línea*)
[GetLineStyle
(*LineStyleVar*)] Devuelve el *Estilo de línea* actual en la *Variable estilo de línea*.
Disposiciones de uso:

- *Variable estilo de línea* hará referencia a un objeto Variable entera activo.
- El objeto *Objetivo* será un objeto Arte lineal dinámico disponible.

Descripción de sintaxis:

GetLineStyle	-->	Target ,
		LineStyleVar
Target	-->	GenericObjectReference
LineStyleVar	-->	ObjectReference

Obtener color
de línea
(*Variable color
de línea*)
[GetLineColour
(LineColourVar)]

Devuelve el *Color de línea* actual en la *Variable color de línea*.

Disposiciones de uso:

- *Variable color de línea* hará referencia a un objeto Variable cadena de octetos si el Color de línea de referencia se especifica como una Cadena de octetos. *Variable color de línea* hará referencia a un objeto Variable entera activo si Color de línea de referencia se especifica como un Entero.
- El objeto *Objetivo* será un objeto Arte lineal dinámico disponible.

Descripción de sintaxis:

GetLineColour	-->	Target , LineColourVar
Target	-->	GenericObjectReference
LineColourVar	-->	ObjectReference

Obtener color
de relleno
(*Variable color
de relleno*)
[GetFillColour
(FillColourVar)]

Devuelve el *Color de relleno de referencia* actual en la *Variable color de relleno*.

Disposiciones de uso:

- *Variable color de relleno* hará referencia a un objeto Variable cadena de octetos si el Color de relleno de referencia se especifica como una Cadena de octetos. *Variable color de relleno* hará referencia a un objeto Variable entera activo si Color de relleno de referencia se especifica como un Entero.
- El objeto *Objetivo* será un objeto Arte lineal dinámico disponible.

Descripción de sintaxis:

GetFillColour	-->	Target , FillColourVar
Target	-->	GenericObjectReference
FillColourVar	-->	ObjectReference

Dibujar arco
(*Anchura de
elipse X, Y,
Altura de elipse,
Ángulo de
comienzo,
Ángulo de arco*)
[DrawArc
(X, Y, EllipseWidth,
EllipseHeight,
StartAngle,
ArcAngle)]

Dibuja un arco entre *Ángulo de comienzo* y *Ángulo de comienzo + Ángulo de arco* (arco BC en la figura 12 a continuación).

Arco se dibuja en *Color de línea*.

Los puntos X, Y son relativos al atributo *Posición* del objeto. Es decir, que X=0, Y=0 corresponden a la esquina superior izquierda de la casilla delimitante. Se permiten valores de X e Y fuera de la casilla delimitante, pero solamente se reproducirá la parte del dibujo dentro de la casilla delimitante.

Los ángulos se expresan en sesenta y cuatroavos de grados, y están en el intervalo [0, 23039]. *Arco de ángulo* no será 0.

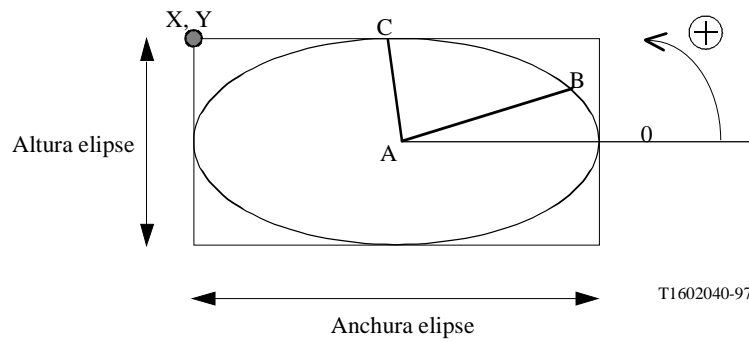


Figura 12/T.172 – Ilustración de parámetros de dibujo

Disposición de uso:

- El objeto *Objetivo* será un objeto arte lineal dinámico disponible.

Descripción de sintaxis:

DrawArc	-->	Target,
		X,
		Y,
		EllipseWidth,
		EllipseHeight,
		StartAngle,
		ArcAngle
Target	-->	GenericObjectReference
X	-->	GenericInteger
Y	-->	GenericInteger
EllipseWidth	-->	GenericInteger
EllipseHeight	-->	GenericInteger
StartAngle	-->	GenericInteger
ArcAngle	-->	GenericInteger

Dibujar sector
(Anchura de
elipse X, Y,
Altura de elipse,
Ángulo de
comienzo,
Ángulo de arco)

[DrawSector
(X, Y, EllipseWidth,
EllipseHeight,
StartAngle,
ArcAngle)]

Dibuja un sector entre *Ángulo de comienzo* y *Ángulo de comienzo + Ángulo de arco* (la superficie ABC en la figura 12 a continuación).

Las líneas se dibujan con color de *Línea de referencia* y la superficie se rellena *Color de relleno de referencia*.

Los puntos X, Y son relativos al atributo *Posición* del objeto. Es decir que X=0, Y=0 corresponden a la esquina superior izquierda de la casilla delimitante. Se permiten valores de X e Y fuera de la casilla delimitante, pero solamente se reproducirá la parte del dibujo dentro de la casilla delimitante.

Los ángulos se expresan en sesenta y cuatroavos de grados, y están en el intervalo [0, 23039]. *Arco de ángulo* no será 0.

Disposición de uso:

- El objeto *Objetivo* será un objeto Arte lineal dinámico disponible.

Descripción de sintaxis:

DrawSector	-->	Target ,
		X,
		Y,
		EllipseWidth,
		EllipseHeight,
		StartAngle,
		ArcAngle
Target	-->	GenericObjectReference
X	-->	GenericInteger
Y	-->	GenericInteger
EllipseWidth	-->	GenericInteger
EllipseHeight	-->	GenericInteger
StartAngle	-->	GenericInteger
ArcAngle	-->	GenericInteger

Dibujar línea
(X1, Y1, X2, Y2)

[DrawLine
(X1, Y1, X2, Y2)]

Dibuja una línea entre (X1, Y1) y (X2, Y2).

Los puntos X1, Y1 y X2, Y2 son relativos al atributo *Posición* del objeto. Se permiten valores fuera de la casilla delimitante, pero sólo se reproducirá la parte del dibujo dentro de la casilla delimitante.

Disposición de uso:

- El objeto *Objetivo* será un objeto Arte lineal dinámico disponible.

Descripción de sintaxis:

DrawLine	-->	Target ,
		X1 ,
		Y1 ,
		X2 ,
		Y2
Target	-->	GenericObjectReference
X1	-->	GenericInteger
Y1	-->	GenericInteger
X2	-->	GenericInteger
Y2	-->	GenericInteger

Dibujar óvalo
(*Anchura de
elipse, X, Y,
Altura de elipse*)

[DrawOval
(*X, Y, EllipseWidth,
EllipseHeight*)]

Dibuja una elipse limitada por el rectángulo definido por los parámetros (véase la figura 12 a continuación).

Esta elipse se rellena con *Color de relleno de referencia*.

Los puntos *X, Y* son relativos al atributo *Posición* del objeto. Es decir, que *X=0, Y=0* corresponden a la esquina superior izquierda de la casilla delimitante. Se permiten valores de *X* e *Y* fuera de la casilla delimitante, pero sólo se reproducirá la parte del dibujo dentro de la casilla delimitante.

Disposición de uso:

- El objeto *Objetivo* será un objeto Arte lineal dinámico disponible.

Descripción de sintaxis:

DrawOval	-->	Target ,
		X ,
		Y ,
		EllipseWidth ,
		EllipseHeight
Target	-->	GenericObjectReference
X	-->	GenericInteger
Y	-->	GenericInteger
EllipseWidth	-->	GenericInteger
EllipseHeight	-->	GenericInteger

Dibujar polígono
(*Lista de puntos*)
[DrawPolygon
(*PointList*)]

Dibuja un polígono cerrado.

Este polígono se rellena con *Color de relleno de referencia*.

Lista de puntos es una lista de Puntos.

Un Punto es definido por la coordenada X e Y, con respecto al atributo *Posición* del objeto. Se permiten valores de X e Y fuera de la casilla delimitante, pero sólo se reproducirá la parte del dibujo dentro de la casilla delimitante.

Disposición de uso:

- El objeto *Objetivo* será un objeto arte lineal dinámico disponible.

Descripción de sintaxis:

DrawPolygon	-->	Target , PointList
Target	-->	GenericObjectReference
PointList	-->	Point+
Point	-->	X, Y
X	-->	GenericInteger
Y	-->	GenericInteger

Dibujar polilíneas
(*Línea de puntos*)
[DrawPolyline
(*PointList*)]

Dibuja series de líneas juntas.

Lista de puntos es una lista de Puntos.

Un punto es definido por la coordenada X e Y, con respecto al atributo *Posición* del objeto. Se permiten valores de X e Y fuera de la casilla delimitante, pero sólo se reproducirá la parte del dibujo dentro de la casilla delimitante.

Disposición de uso:

- El objeto *Objetivo* será un objeto arte lineal dinámico disponible.

Descripción de sintaxis:

DrawPolyline	-->	Target , PointList
Target	-->	GenericObjectReference
PointList	-->	Point+
Point	-->	X, Y
X	-->	GenericInteger
Y	-->	GenericInteger

Dibujar rectángulo (X1, Y1, X2, Y2)

Dibuja un rectángulo.

Este rectángulo se rellena con *Color de relleno de referencia*.

[DrawRectangle (X1, Y1, X2, Y2)]

El punto Superior izquierdo es (X1, Y1) y el punto Inferior derecho es (X2, Y2). (X1, Y1) y (X2, Y2) son relativos al atributo *Posición* del objeto. Se permiten valores fuera de la casilla delimitante, pero sólo se reproducirá la parte del dibujo dentro de la casilla delimitante.

Disposición de uso:

- El objeto *Objetivo* será un objeto Arte lineal dinámico disponible.

Descripción de sintaxis:

DrawRectangle	-->	Target ,
		X1 ,
		Y1 ,
		X2 ,
		Y2
X1	-->	GenericInteger
Y1	-->	GenericInteger
X2	-->	GenericInteger
Y2	-->	GenericInteger

Liberar (Clear)

Rellena la casilla delimitante con el *Color de relleno de referencia original*.

Cuando la Casilla delimitante con bordes está puesta a Verdadero, el borde no se rellena con *Color de relleno de referencia original*.

Disposición de uso:

- El objeto *Objetivo* será un objeto Arte lineal dinámico disponible.

Descripción de sintaxis:

Clear	-->	Target
Target	-->	GenericObjectReference

35.5 Descripción formal

DynamicLineArt Class	-->	LineArt Class
----------------------	-----	---------------

36 Clase Texto

Descripción: Define atributos y comportamientos de piezas de información textual

Clase básica: Visible

Subclases: Campo de entrada, Hipertexto

Estado: Clase concreta

36.1 Atributos

Esta subcláusula define los atributos heredados, intercambiados e internos para esta clase.

36.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las siguientes restricciones:

Nombre de atributo	Definido en	Restricciones y requisitos
Gancho de contenido	Ingrediente	Este atributo o su atributo por defecto correspondiente en la clase Aplicación (Gancho de contenido de texto) es obligatorio para esta clase.
Contenido original	Ingrediente	Este atributo es obligatorio para esta clase.

36.1.2 Atributos intercambiados propios

Esta clase define los siguientes atributos intercambiados adicionales:

Tipo de carácter original (OriginalFont) Indicación del tipo de carácter que se ha de utilizar cuando se presenta inicialmente el objeto Texto.

El atributo Tipo de carácter original representa un nombre para un tipo de carácter (residente en el motor MHEG-5) o una referencia a un objeto Tipo de carácter. En ambos casos, el tipo de carácter indicado se utilizará para reproducir el objeto Texto.

Cuando no hay ninguna referencia a tipo de carácter codificada, el objeto Texto se presentará utilizando el tipo de carácter por defecto referenciado en el objeto Aplicación activo, y si en el mismo no se hace referencia a ningún tipo de carácter, se utilizará un tipo de carácter por defecto del motor MHEG-5.

NOTA – El atributo Tipo de carácter original proporciona el tipo de carácter inicial para un objeto Texto. Además, el formato de codificación de texto puede contener secuencias de escape para conmutar entre tipos de caracteres.

- Atributo facultativo.
- Cadena de octetos que representa un nombre de Tipo de carácter, o una referencia a un objeto Tipo de carácter.
- Valor por defecto: Valor codificado por la Aplicación.

Atributos de tipo de carácter (FontAttributes)

Este atributo se utiliza para fijar atributos de Tipo de carácter, tales como estilo, tamaño de carácter, color de texto y color de fondo.

El formato de codificación exacto de *Atributos de tipo de carácter* se relaciona con el valor del objeto Tipo de carácter mencionado por el atributo *Tipo de carácter*.

Cuando no hay atributos Tipo de carácter codificados, el objeto Texto será presentado utilizando *Atributos de tipo de carácter* por defecto codificados en el objeto Aplicación activo, y si en el mismo no se hace referencia a ningún *Atributos de tipo de carácter*, no se fijan atributos específicos.

- Cadena de octetos facultativa.
- Valor por defecto: Valor codificado por la Aplicación.

Color de texto (TextColour)	<p>Indica el color que se ha de utilizar para reproducir el primer plano del objeto texto. Este atributo es interpretado como un índice basado en cero en la tabla de mejora de colores definido por el atributo <i>Referencia de paleta</i>, o un valor de color directo, dependiendo del tipo de atributo.</p> <ul style="list-style-type: none"> • Entero facultativo o Cadena de octetos. Un Entero se interpretará como un índice en una Paleta; una Cadena de octetos se interpretará como un valor de color directo. • Valor por defecto: Valor codificado por la Aplicación.
Color de fondo (BackgroundColour)	<p>Indica el color que se ha de utilizar para reproducir el fondo del objeto texto. Este atributo es interpretado como un índice basado en cero en la tabla de mejora de colores, definido por el atributo <i>Referencia de paleta</i>, o un valor de color directo, dependiendo del tipo de atributo.</p> <ul style="list-style-type: none"> • Entero facultativo o Cadena de octetos. Un entero se interpretará como un índice en una Paleta; una Cadena de octetos se interpretará como un valor de color directo. • Valor por defecto: Valor codificado por la Aplicación.
Conjunto de caracteres (CharacterSet)	<p>Identificación del conjunto de caracteres, o conjunto de conjuntos de caracteres, que serán utilizados por defecto para reproducir Texto. Este Entero se codificará con un valor que representa el conjunto de caracteres. El dominio de aplicación definirá una gama de Conjunto de caracteres en su semántica.</p> <p>NOTA – El atributo <i>Conjunto de caracteres</i> proporciona el conjunto de caracteres inicial para un objeto Texto. Además, el formato de codificación de texto puede contener secuencias de escape para conmutar entre conjuntos de caracteres.</p> <ul style="list-style-type: none"> • Entero facultativo. • Valor por defecto: El valor del atributo <i>Conjunto de caracteres</i> del objeto Aplicación, si se especifica ese atributo.
Justificación horizontal (HorizontalJustification)	<p>El atributo <i>Justificación horizontal</i> indica cómo las líneas de texto se justifican con respecto a los bordes verticales de la casilla delimitante definida por los atributos <i>Tamaño de casilla</i> y <i>Posición</i> del objeto Texto.</p> <p>Este atributo se puede omitir si una representación codificada del propio texto tiene la misma funcionalidad para especificar este tipo de presentación. El dominio de aplicación basado en la presente Recomendación definirá cada Gancho de contenido para el cual se omite el atributo.</p> <ul style="list-style-type: none"> • Atributo facultativo – Uno de <i>comienzo</i> <i>fin</i> <i>centro</i> <i>justificado</i>. • Valor por defecto: <i>comienzo</i>.
Justificación vertical (VerticalJustification)	<p>El atributo <i>Justificación vertical</i> indica cómo las líneas de texto se justifican con respecto a los bordes horizontales de la casilla delimitante definida por los atributos <i>Tamaño de casilla</i> y <i>Posición</i> del objeto Texto.</p>

Este atributo se puede omitir si una representación codificada del propio texto tiene la misma funcionalidad para especificar este tipo de presentación. El dominio de aplicación basado en la presente Recomendación definirá cada Gancho de contenido para el cual se omite el atributo.

- Atributo facultativo – Uno de *comienzo* | *fin* | *centro* | *justificado*.
- Valor por defecto: *comienzo*.

Orientación de línea (LineOrientation)

El atributo *Orientación de línea* se combina con el atributo *Esquina de comienzo* para determinar la manera en que los caracteres se organizan en líneas, y las líneas en secuencias de líneas.

Este atributo se puede omitir si una representación codificada del propio texto tiene la misma funcionalidad para especificar este tipo de presentación. El dominio de aplicación basado en esta Recomendación definirá cada Gancho de contenido para el cual se omite el atributo.

- Atributo facultativo – Uno de *vertical* | *horizontal*.
- Valor por defecto: *horizontal*.

NOTA – Para la orientación de línea *vertical*, se supone que la orientación del carácter sea normal (es decir, toda la línea rota 90° con respecto a *horizontal*), a menos que el contenido del *Texto* especifique otra cosa.

Esquina de comienzo (StartCorner)

El atributo *Esquina de comienzo* contiene una identificación de la esquina de la zona de presentación donde debe comenzar la presentación del texto.

Este atributo se puede omitir si una representación codificada del propio texto tiene la misma funcionalidad para especificar este tipo de reproducción. El dominio de aplicación basado en esta Recomendación definirá cada Gancho de contenido para el cual se omite el atributo.

- Atributo facultativo – Uno de *superior izquierda* | *superior derecha* | *inferior izquierda* | *inferior derecha*.
- Valor por defecto: *superior izquierda*.

Retorno a comienzo de línea de texto (TextWrapping)

Indica si el texto continúa en una nueva línea al final de la línea, o si es recortado.

Si el atributo Retorno a comienzo de línea de texto se pone a Verdadero, el texto continúa. Si se pone a Falso, se recorta.

Este atributo se puede omitir si una representación codificada del propio texto tiene la misma funcionalidad para especificar este tipo de presentación. El dominio de aplicación basado en esta Recomendación definirá cada Gancho de contenido para el cual se omite el atributo.

- Booleano facultativo.
- Valor por defecto: Falso.

36.1.3 Atributos internos propios

Esta clase define los siguientes atributos internos adicionales:

<i>Datos de texto</i> (<i>TextData</i>)	<p>Valor de los datos de contenido textual del objeto Texto.</p> <p>Si el atributo <i>Contenido</i> del objeto Texto es un Contenido incluido, <i>Datos de texto</i> se pone inicialmente a Contenido incluido.</p> <p>Si el atributo <i>Contenido</i> del objeto Texto es una referencia a una fuente de datos externa, <i>Datos de texto</i> se pone inicialmente a una Cadena de octetos que representa el contenido de esta fuente de datos externa. En este caso, el valor del atributo Gancho de contenido se pudiera utilizar para el formato del valor de <i>Datos de texto</i>.</p> <ul style="list-style-type: none">• Cadena de octetos.• Valor inicial: Contenido incluido o un Contenido referenciado.
<i>Tipo de carácter</i> (<i>Font</i>)	<p>Tipo de carácter que se ha de utilizar cuando se presenta el objeto Texto.</p> <p>El atributo <i>Tipo de carácter</i> representa un nombre para un tipo de carácter (que reside en el motor MHEG-5) o una referencia a un objeto Tipo de carácter. En ambos casos, el tipo de carácter indicado se utilizará para presentar el objeto Texto.</p> <p>NOTA – El atributo <i>Tipo de carácter</i> proporciona el tipo de carácter inicial para un objeto Texto. Además, el formato de codificación del texto puede contener secuencias de escape para conmutar entre tipos de carácter.</p> <ul style="list-style-type: none">• Atributo facultativo.• Cadena de octetos que representa un nombre de Tipo de carácter, o una referencia a un objeto Tipo de carácter.• Valor inicial: Valor del atributo Tipo de carácter original.

36.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

36.3 Comportamientos internos

Los comportamientos internos de esta clase tienen la misma semántica que su clase básica.

36.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica, salvo para Fijar datos. Además, se definen las siguientes acciones MHEG-5 aplicables:

Fijar datos (<i>SetData</i>)	<p>Ejecutar sincronamente las siguientes acciones:</p> <ol style="list-style-type: none">1) Actualizar el valor del atributo interno <i>Datos de texto</i> del Texto objetivo.2) Ejecutar la acción Fijar datos definida en la clase Presentable. <p>Las disposiciones de uso y la descripción de sintaxis se proporcionan en la clase Ingrediente.</p>
-----------------------------------	--

Obtener contenido de texto
(*Variable contenido de texto*)
[GetTextContent
(TextContentVar)]

Fijar la variable referenciada por *Variable contenido de texto* al valor del atributo *Contenido*.

NOTA – Si se incluye el atributo *Contenido* del objeto Texto, *Variable contenido de texto* devuelve Cadena de octetos de texto; si se hace referencia al atributo *Contenido* del objeto texto, *Variable contenido de texto* devuelve esa referencia de objeto.

Disposiciones de uso:

- *Variable contenido de texto* hará referencia a un objeto variable Cadena de octetos activo o a un objeto Variable de referencia de contenido activo.
- El objeto *Objetivo* será un objeto Texto disponible.

Descripción de sintaxis:

GetTextContent	-->	Target , TextContentVar
Target	-->	GenericObjectReference
TextContentVar	-->	ObjectReference

Obtener datos de texto
(Variable datos de texto)
[GetTextData
(TextDataVar)]

Fijar la variable referenciada por *Variable de datos de texto* al valor del atributo *Datos de texto*.

Disposiciones de uso:

- Variable de *Datos de texto* hará referencia a un objeto Cadena de octetos activo.
- El objeto *Objetivo* será un objeto Texto disponible.

Descripción de sintaxis:

GetTextData	-->	Target , TextDataVar
Target	-->	GenericObjectReference
TextDataVar	-->	ObjectReference

Fijar referencia de tipo de carácter
(*Nuevo tipo de carácter*)
[SetFontRef
(NewFont)]

Cambiar el tipo de carácter utilizado para presentar un texto.

Ejecutar las siguientes secuencias de acciones:

- 1) Fijar el valor del atributo *Tipo de carácter* a *Nuevo tipo de carácter*.
- 2) Si el objeto Texto objetivo está activo, dibujar de nuevo el objeto objetivo inmediatamente teniendo en cuenta el nuevo valor de *Tipo de carácter* y de acuerdo con su posición en la *Pila de visualización* del objeto Aplicación activo.

Nuevo tipo de carácter pudiera ser una referencia a un objeto Tipo de carácter o a un nombre de tipo de carácter. Esta Recomendación no define cómo los nombres de tipo de carácter son gestionados por el motor MHEG-5.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Texto disponible.

- Si *Nuevo tipo de carácter* hace referencia a un objeto Tipo de carácter, este objeto estará disponible.

Descripción de sintaxis:

SetFontRef		Target , NewFont
Target	-->	GenericObjectReference
NewFont	-->	NewFontName NewFontReference
NewFontName	-->	GenericOctetString
NewFontReference	-->	GenericObjectReference

36.5 Descripción formal

Text Class	-->	Visible Class, OriginalFont?, FontAttributes?, TextColour?, BackgroundColour?, CharacterSet?, HorizontalJustification?, VerticalJustification?, LineOrientation?, StartCorner?, TextWrapping?
OriginalFont	-->	OctetString ObjectReference
FontAttributes	-->	OctetString
TextColour	-->	Colour
BackgroundColour	-->	Colour
CharacterSet	-->	INTEGER
HorizontalJustification	-->	start end centre justified
VerticalJustification	-->	start end centre justified
LineOrientation	-->	vertical horizontal
StartCorner	-->	upper-left upper-right lower-left lower-right
TextWrapping	-->	BOOLEAN

37 Clase Tren

Descripción: Define el comportamiento de una composición de medios continuos (Vídeo, Audio y Gráficos RT) que se presentan en sincronización

Clase básica: Presentable

Subclases: Ninguna

Estado: Clase concreta

37.1 Atributos

Esta subcláusula define los atributos heredados, intercambiados e internos para esta clase.

37.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las siguientes restricciones:

Nombre de atributo	Definido en	Restricciones y requisitos
Gancho de contenido	Ingrediente	Este atributo o su atributo por defecto correspondiente en la clase Aplicación (Gancho de contenido de tren) es obligatorio para esta clase.
Contenido original	Ingrediente	Este atributo es obligatorio para esta clase. Contiene una referencia a un multiplex completo de medios sincronizados.

37.1.2 Atributos intercambiados propios

Esta clase define los siguientes atributos internos adicionales:

Multiplex	<p>Lista de inclusión de objetos Vídeo, Audio y Gráficos RT que han de ser presentados simultáneamente. Éstos se denominan a continuación Componentes de tren. Cada componente de tren tiene un rótulo, que se utiliza para identificarlo de manera única dentro del tren.</p> <p>El atributo <i>Activo inicialmente</i> de cada Componente de tren determina si el tren elemental correspondiente es ejecutado automáticamente como resultado de que el multiplex de trenes es activado por primera vez.</p> <p>Obsérvese que todos los comportamientos asociados con sincronización son accedidos siempre a través de la clase Tren; por ejemplo, no es posible dirigir la acción Fijar posición de contador a un objeto audio.</p> <ul style="list-style-type: none">• Secuencia de inclusiones de objetos Vídeo, Audio y Gráficos RT.
Almacenamiento (Storage)	<p>Indicar si la composición de medios continuos está cargada en memoria antes de la presentación o si se presentan directamente fuera del tren, por ejemplo, desde un servidor. Para el motor MHEG-5, la diferencia del tratamiento es que en el caso <i>memoria</i>, el motor MHEG-5 sincronizará el tren, mientras que en el caso <i>tren</i>, el tren es sincronizado por el servidor.</p> <ul style="list-style-type: none">• Atributo facultativo: Uno de <i>memoria</i> <i>tren</i>.• Valor por defecto: <i>tren</i>.
Puesta en bucle (Looping)	<p>Número de presentaciones del objeto Tren.</p> <p>En el cómputo de bucles, el cómputo real se produce a medida que el Tren llega a su fin. Como resultado, la acción Fijar posición de contador será interpretada en un bucle dado.</p> <p>Cuando un Tren es presentado y detenido antes de que se efectúen todos los bucles, la siguiente acción de difusión continuará a partir de ese bucle (y desde esa Posición de contador, a menos que sea fijada por una acción específica); en otras palabras, el atributo Puesta en bucle representará el número total de bucles.</p>

Cuando un Tren ha llegado al fin de todos los bucles y es detenido, una nueva activación ejecutará el Tren como si fuese activado por la primera vez, es decir, la puesta en bucle de acuerdo con el atributo Puesta en bucle.

- Entero facultativo.
- Valor por defecto.
- Valor especial: 0 significa *infinito*.

37.1.3 Atributos internos propios

Esta clase define los siguientes atributos internos propios:

Velocidad
(*Speed*)

Velocidad a la cual se presenta la composición de medios continuos.

El atributo *Velocidad* es un número racional, presentado como dos Enteros (a, b). Velocidad se define como a/b . La semántica de los siguientes valores difieren según si el motor MHEG-5 tiene acceso o no a una capa subyacente para admitir modos truco.

NOTA – Los modos Truco pudieran ser proporcionado por el protocolo DSM-CC de ISO/CEI 13818-6.

	Modo sin truco	Modo truco
-1/1	Tratado como 0	Inverso Hacia atrás a velocidad normal
0/1	Parada Congela en la posición vigente	Parada Congela en la posición vigente
1/1	Reproducción normal Reproduce desde un punto que no está bajo el control del motor MHEG-5	Reproducción normal Reproduce desde el punto donde estaba el tren cuando abandonó su modo previo

Se permiten otros valores de Velocidad (por ejemplo, 1/2 ó 2/1) en el modo truco. Cuando no se aplica el modo truco, estos otros valores de velocidad serán tratados como ejecución normal (1/1) cuando son positivos y como parada (0/1) cuando son negativos o nulos.

- Par de enteros: Un numerador y un Denominador facultativo con valor por defecto 1.
- Valor por defecto: 1/1.

Posición de contador
(*CounterPosition*)

Posición temporal vigente del Tren dentro de la duración de un tren a velocidad normal.

Este atributo se expresa en Unidades de contador de trenes.

La definición real de la Unidad de contador de trenes está fuera del alcance de la presente Recomendación.

- Entero.
- Valor por defecto: 0.

<i>Posición de fin de contador</i> (<i>CounterEndPosition</i>)	<p>Posición de la última trama de un tren ejecutado a velocidad normal. El tren se detendrá automáticamente cuando alcanza esta posición. Se genera el evento <i>Tren detenido</i>.</p> <p>Este atributo se expresa en Unidades de contador de trenes.</p> <p>La definición real de la Unidad de contador de trenes está fuera del alcance de la presente Recomendación.</p> <ul style="list-style-type: none"> • Entero. • Valor por defecto: -1, que significa Fin de tren.
<i>Activadores de contador</i> (<i>CounterTriggers</i>)	<p>Lista de valores que representan las posiciones de contador del Tren cuando el ejecutor de trenes genera eventos <i>Activador de contador</i> para este Tren.</p> <p>Cada activador tiene un identificador único dentro de la lista de <i>Activadores de contador</i> y una posición de contador expresada en Unidades de contador de trenes.</p> <ul style="list-style-type: none"> • Secuencia de las siguientes estructuras de datos: <ul style="list-style-type: none"> – identificador de activador: Entero; – posición de contador: Entero. • Valor inicial: Secuencia vacía.

37.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica. Además, se definen los siguientes eventos:

<i>Evento de tren</i> (<i>StreamEvent</i>)	<p>Este evento es generado automáticamente por el ejecutor de Trenes cuando el múltiplex de Trenes atraviesa un marcador específico. El marcador se reconoce por una etiqueta que puede estar codificada dentro de la estructura de datos de contenido del Tren. Pudiera haber varios marcadores con la misma identidad en un múltiplex de Trenes.</p> <ul style="list-style-type: none"> • Datos asociados: Rótulo de evento de tren – Cadena de octetos. <p>NOTA 1 – De acuerdo con la semántica de la clase Enlace, un enlace fijado para activar Eventos de tren de un objeto tren específico, que no tiene sus Datos de evento fijados, activará todos los Eventos de tren de ese objeto.</p> <p>NOTA 2 – La codificación de Eventos de trenes se puede hacer utilizando el protocolo DSM-CC de ISO/CEI 13818-6.</p>
<i>Ejecución de tren</i> (<i>StreamPlaying</i>)	<p>Este evento es generado cuando un Múltiplex de trenes ha comenzado la ejecución. Más específicamente, es generado simultáneamente con la primera pieza de contenido de datos (trama vídeo, muestra de audio) que se presenta al usuario.</p> <ul style="list-style-type: none"> • Ningún dato asociado.
<i>Tren detenido</i> (<i>StreamStopped</i>)	<p>Este evento es generado cuando la ejecución de un Múltiplex de trenes ha detenido la ejecución. Más específicamente, es generado tan pronto como la última pieza del contenido de datos (trama vídeo, muestra de audio) ha sido</p>

presentada al usuario. Obsérvese que el *Estado de ejecución* del objeto Tren no es afectado por la ocurrencia de un evento *Tren detenido*.

- Ningún dato asociado.

Activador de contador
(CounterTrigger)

Este evento es generado automáticamente por el ejecutor de Trenes cuando la *Posición de contador* del objeto Tren atraviesa un valor fijado con la acción Fijar activador de contador. Pudiera haber varios activadores de Contador activados en la misma posición de contador de un tren.

- Datos asociados: Identificador de activador – Entero.

NOTA – La codificación de la posición de contador dentro del tren se puede hacer utilizando el protocolo DSM-CC de ISO/CEI 13818-6.

37.3 Comportamientos internos

La semántica de los siguientes comportamientos internos ha cambiado con respecto a la clase básica de objeto.

Preparación

Ejecutar la siguiente secuencia de acciones:

- 1) Aplicar los tres primeros pasos del comportamiento *Preparación* definido en la clase Raíz.
- 2) Aplicar el comportamiento *Activación* a todos los Componentes de los trenes que tienen puesto a Verdadero el atributo *Activo inicialmente*, en el orden en que aparecen en el Múltiplex de trenes.
- 3) Aplicar los pasos 4 a 6 del comportamiento *Preparación* definido en la clase Raíz.

Destrucción

Ejecutar la siguiente secuencia de acciones:

- 1) Aplicar el comportamiento *Destrucción* a todos los Componentes de trenes del Tren, en el orden inverso en que aparecen en el Múltiplex de trenes.
- 2) Aplicar el comportamiento *Destrucción* definido en la clase básica.

Activación

Ejecutar la siguiente secuencia de acciones:

- 1) Aplicar el comportamiento *Activación* definido en la clase básica.
- 2) Comenzar la ejecución de todos los Componentes de tren.
- 3) Fijar el atributo *Estado de ejecución* a Verdadero y generar un evento *Está funcionando*.

La Activación y Desactivación de Componentes de tren tales como Audio, Vídeo o Gráficos RT corresponde a habilitarlos o inhabilitarlos en un tren. Por ejemplo cuando se ejecuta un Tren, la activación de Audio hará que ese componente sea audible (además de los otros).

Desactivación

Ejecutar la siguiente secuencia de acciones:

- 1) Si el atributo *Estado de ejecución* del objeto es Falso, omitir el comportamiento. En los demás casos:
- 2) Detener la ejecución de todos los Componentes de tren activos.
- 3) Ejecutar el comportamiento *Desactivación* definido en la clase básica.

37.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se definen las siguientes acciones MHEG-5 aplicables.

Fijar datos La acción Fijar datos definida en la clase Ingrediente sólo se ejecutará cuando el tren está inactivo.

Clonar Esta acción no se dirigirá a un Tren.

Fijar activador de contador Actualizar la lista de activadores de Contador de un objeto tren.

(*Identificador de contados,*
Nuevo valor de contador)

Ejecutar la siguiente secuencia de acciones:

[SetCounterTrigger
(*TriggerIdentifier,*
NewCounterValue)]

- 1) Actualizar el conjunto de activadores de la lista *Activadores de contador* del Tren objetivo, de acuerdo con las reglas siguientes:
 - a) Si *Identificador de activador* es el identificador de un activador existente en *Activadores de contador*, el nuevo activador sustituye al anterior.
 - b) Si no hay activador con *Identificador de activador* en *Activadores de contador*, insertar un nuevo activador con *Identificador de activador* y el valor *Nuevo valor de contador* en *Activadores de contador*.
 - c) Si *Nuevo valor de contador* no está codificado y hay un activador con *Identificador de activador* en *Activadores de contador*, suprimir este activador de la lista de *Activadores de contador*.
 - d) Si *Nuevo valor de contador* no está codificado y no hay activadores con *Identificador de activador* en *Activadores de contador*, descartar esta acción.
- 2) Si el objeto Tren objetivo está activo, el motor MHEG-5 generará eventos de *Activador de contador* de acuerdo con el nuevo valor de la lista de *Activadores de contador*.

Disposición de uso:

- El objeto *Objetivo* será un objeto Tren disponible.

Descripción de sintaxis:

SetCounterTrigger	-->	Target ,
		TriggerIdentifier ,
		NewCounterValue?
Target	-->	GenericObjectReference
TriggerIdentifier	-->	GenericInteger
NewCounterValue	-->	GenericInteger

Fijar velocidad
(*Nueva velocidad*)

[SetSpeed
(*NewSpeed*)]

Cambiar la velocidad de presentación de un tren.

Ejecutar la siguiente secuencia de acciones:

- 1) Fijar el valor del atributo *Velocidad* del objeto Tren objetivo a *Nueva velocidad*.
- 2) Si el Tren objetivo está activo, actualizar inmediatamente la presentación del tren teniendo en cuenta el nuevo valor del atributo *Velocidad*.

El atributo *Nueva velocidad* se define como una relación Numerador/Denominador.

NOTA 1 – Como se ha indicado anteriormente, en un entorno de ejecución, las acciones ejecutar y detener comenzarán cuando sea posible en el tren difundido. En otros casos, la fijación de nueva velocidad (incluida la ejecución y la detención normales) se efectuará en la posición de contador vigente, o lo más próxima posible a ésta (por ejemplo, siguiente trama I). La posición del contador se puede fijar mediante la acción apropiada.

NOTA 2 – Si los *Modos truco* no son admitidos por el motor, el valor de Velocidad se puede fijar a cualquier valor y el motor la interpretará como se explica en el atributo interno *Velocidad*.

Disposición de uso:

- El objeto *Objetivo* será un objeto Tren disponible.

Descripción de sintaxis:

SetSpeed	-->	Target , NewSpeed
Target	-->	GenericObjectReference
NewSpeed	-->	Rational
Rational	-->	Numerator , Denominator?
Numerator	-->	GenericInteger
Denominator	-->	GenericInteger

Fijar posición de contador
(*Nueva posición de contador*)

[SetCounterPosition
(*NewCounterPosition*)]

Cambiar la posición vigente dentro de un tren.

Ejecutar la siguiente secuencia de acciones:

- 1) Si el motor MHEG-5 no tiene una capa de presentación subyacente que admita modos truco, descartar esta acción.
- 2) Fijar el valor del atributo *Posición de contador* del tren objetivo a *Nueva posición de contador*.
- 3) Si el tren objetivo está activo, saltar inmediatamente a la nueva posición sin cambiar el *Estado de ejecución* del Tren objetivo.

NOTA – No se generan eventos de tren debido a una acción Fijar posición de contador.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Tren disponible.
- *Nueva posición de contador* indicará una posición válida dentro del Tren objetivo.

Descripción de sintaxis:

SetCounterPosition	-->	Target ,
		NewCounterPosition
Target	-->	GenericObjectReference
NewCounterPosition	-->	GenericInteger

Fijar posición de fin de contador
(*Nueva posición de fin de contador*)

[SetCounterEndPosition
(*NewCounterEndPosition*)]

Cambiar la posición final de un tren.

Ejecutar la siguiente secuencia de acciones:

- 1) Si el motor MHEG-5 no tiene una capa de presentación subyacente que admita modos truco, descartar esta acción.
- 2) Fijar el valor del atributo *Posición de fin de contador* del tren objetivo a *Nueva posición de fin de contador*.
- 3) Si el Tren objetivo está activo, y *Nueva posición de fin de contador* ya ha pasado, detener el Tren objetivo.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Tren disponible.
- *Nueva posición de contador* indicará una posición válida dentro del Tren objetivo.

Descripción de sintaxis:

SetCounterPosition	-->	Target ,
		NewCounterPosition
Target	-->	GenericObjectReference
NewCounterPosition	-->	GenericInteger

37.5 Descripción formal

Stream Class	-->	Presentable Class ,
		Multiplex ,
		Storage? ,
		Looping?
Multiplex	-->	StreamComponent+
StreamComponent	-->	Audio class Video class RTGraphics class
Storage	-->	memory stream
Looping	-->	INTEGER

38 Clase Audio

Descripción:	Define los atributos y el comportamiento de un tren de audio elemental de un múltiplex de Trenes. El objeto Audio será un Componente de tren de un objeto tren
Clase básica:	Presentable
Subclases:	Ninguna
Estado:	Clase concreta

38.1 Atributos

Esta subcláusula define los atributos heredados, intercambiados e internos para esta clase.

38.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las siguientes restricciones:

Nombre de atributo	Definido en	Restricciones y requisitos
Estado de ejecución	Raíz	Este atributo expresa que este objeto está habilitado o inhabilitado para ejecución cuando el objeto Tren que lo contiene está activo. El objeto sólo es ejecutado cuando este atributo es Verdadero y el estado de ejecución del Tren es verdadero.
Gancho de contenido, Contenido original, Compartido	Ingrediente	Estos atributos no se codificarán para esta clase.

38.1.2 Atributos intercambiados propios

Esta clase define los siguientes atributos intercambiados adicionales:

Volumen original (OriginalVolume)	<p>Volumen del objeto Audio cuando está disponible por primera vez.</p> <p>El atributo Volumen original se expresa en dB, donde 0 dB se define como el volumen normalizado para la ejecución de audio. Esta exactitud precisa de la reproducción del volumen está fuera del alcance de la presente Recomendación.</p> <ul style="list-style-type: none">• Entero facultativo.• Valor por defecto: 0.
Rótulo de componente (ComponentTag)	<p>Un identificador único para el tren de audio elemental dentro de un tren de medios multiplexados.</p> <ul style="list-style-type: none">• Entero.

38.1.3 Atributos internos propios

Esta clase define el siguiente atributo interno adicional:

Volumen (Volume)	<p>Volumen vigente del objeto Audio, definido de la misma manera que Volumen original.</p> <ul style="list-style-type: none">• Entero.• Valor inicial: <i>Volumen original</i>.
---------------------	--

38.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

38.3 Comportamientos internos

Esta clase no define comportamientos internos.

38.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se definen las siguientes acciones MHEG-5 aplicables:

Fijar datos La acción Fijar datos, definida en la clase Ingrediente, sólo se ejecutará cuando el Audio está inactivo.

Clonar Clonar no se dirigirá a Audio.

Fijar volumen Cambiar el volumen de un Audio.

(*Nuevo volumen*) Ejecutar la siguiente secuencia de acciones:

- [SetVolume
(*NewVolume*)]
- 1) Fijar el atributo *Volumen* del objeto Audio objetivo a *Nuevo volumen*.
 - 2) Si se está ejecutando el tren de audio elemental identificado por el atributo Rótulo de componente, actualizar la reproducción del Audio teniendo en cuenta el nuevo *Volumen*.

Disposición de uso:

- El objeto *Objetivo* será un objeto Audio disponible.

Descripción de sintaxis:

SetVolume	-->	Target , NewVolume
Target	-->	GenericObjectReference
NewVolume	-->	GenericInteger

Obtener volumen Devuelve el volumen de un Audio.

(*Variable volumen*) Disposiciones de uso:

- [GetVolume
(*VolumeVar*)]
- El objeto *Objetivo* será un objeto Audio disponible.
 - *Variable volumen* será un objeto Variable entera activo.

Descripción de sintaxis:

GetVolume	-->	Target , VolumeVar
Target	-->	GenericObjectReference
VolumeVar	-->	ObjectReference

38.5 Descripción formal

Audio Class	-->	Presentable Class, ComponentTag OriginalVolume?
ComponentTag	-->	INTEGER
OriginalVolume	-->	INTEGER

39 Clase Vídeo

Descripción: Define los atributos y comportamiento de un tren de vídeo elemental de un Múltiplex de trenes. El objeto Vídeo será un Componente tren de un objeto Tren

Clase básica: Visible

Subclases: Ninguna

Estado: Clase concreta

39.1 Atributos

Esta subcláusula define los atributos heredados, intercambiados e internos para esta clase.

39.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las restricciones siguientes:

Nombre de atributo	Definido en	Restricciones y requisitos
Estado de ejecución	Raíz	Este atributo expresa que este objeto está habilitado o inhabilitado para ejecución cuando el objeto Tren lo contiene activo. El objeto se ejecuta solamente cuando este atributo es Verdadero y el estado de ejecución del Tren es Verdadero.
Gancho de contenido, Contenido original, Compartido	Ingrediente	Estos atributos no se codificarán para esta clase.
Referencia de paleta original	Visible	Este atributo no se codificará para esta clase.

39.1.2 Atributos intercambiados propios

Esta clase define los siguientes atributos intercambiados adicionales:

Terminación (Termination) Este atributo indica si la última trama del vídeo desaparecerá cuando termina la presentación del vídeo, o si será congelada.

- Atributo facultativo: Uno de *congelar* | *desaparecer*.
- Valor por defecto: *desaparecer*.

Rótulo de componente (ComponentTag) Un identificador único para el tren de vídeo elemental dentro de un tren de medios multiplexados.

- Entero.

39.1.3 Atributos internos propios

Esta clase no define atributos internos adicionales.

39.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

39.3 Comportamientos internos

Esta clase no define comportamientos internos.

39.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se definen las siguientes acciones MHEG-5 aplicables:

Fijar datos La acción Fijar datos definida en la clase Ingrediente sólo se ejecutará cuando el Vídeo está inactivo.

Clonar Clonar no se dirigirá a Vídeo.

Escala de vídeo
(*Escala X*,
Escala Y) Si el motor MHEG-5 implementa la opción Escala, el efecto de esta acción es adaptar la presentación del vídeo a las dimensiones *Escala X* y *Escala Y*.

[ScaleVideo
(*XScale*, *YScale*)] Los parámetros Escala X y Escala Y representan las dimensiones finales del Vídeo en números de pixels. De este modo, la representación gráfica del Vídeo puede no mantener su formato original.

Obsérvese que esta acción no afecta al atributo interno *Tamaño de casilla* del objeto Vídeo.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Vídeo disponible.
- *Escala X* y *Escala Y* serán Enteros positivos.

Descripción de sintaxis:

ScaleVideo	-->	Target , XScale, YScale
Target	-->	GenericObjectReference
XScale, YScale	-->	GenericInteger

39.5 Descripción formal

Video Class	-->	Visible Class, ComponentTag Termination?
Termination	-->	freeze disappear
ComponentTag	-->	INTEGER

40 Clase Gráficos RT

Descripción	Define los atributos y comportamientos de objetos gráficos no persistentes, definidos como de un tren elemental de un Múltiplex de trenes. El objeto Gráficos RT será un Componente de tren de un objeto Tren
Clase básica	Visible
Subclases	Ninguna
Estado	Clase concreta

40.1 Atributos

Esta subcláusula define los atributos heredados, intercambiados e internos para esta clase.

40.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las siguientes restricciones:

Nombre de atributo	Definido en	Restricciones y requisitos
Estado de ejecución	Raíz	Este atributo expresa que este objeto está habilitado o inhabilitado para ejecución cuando el objeto Tren lo contiene activo. El objeto se ejecuta solamente cuando este atributo es Verdadero y el Estado de ejecución del Tren es Verdadero.
Gancho de contenido, Contenido original, Compartido	Ingrediente	Estos atributos no se codificarán para esta clase.

40.1.2 Atributos intercambiados propios

Esta clase define los siguientes atributos intercambiados adicionales:

Rótulo de componente (<i>ComponentTag</i>)	Un identificador único para el tren de Gráficos RT dentro de un tren de medios multiplexados. <ul style="list-style-type: none">Entero.
Terminación (<i>Termination</i>)	Este atributo indica si la última imagen de los gráficos desaparecerá cuando termine la presentación, o si será congelada. <ul style="list-style-type: none">Atributo facultativo – Uno de <i>congelar</i> <i>desaparecer</i>.Valor por defecto: <i>desaparecer</i>.

40.1.3 Atributos internos propios

Esta clase no define atributos internos adicionales.

40.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

40.3 Comportamientos internos

Esta clase no define comportamientos internos adicionales.

40.4 Efectos de las acciones MHEG-5

Esta clase tiene los mismos conjuntos de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se definen las siguientes acciones MHEG-5 aplicables:

Fijar datos La acción Fijar datos definida en la clase Ingrediente sólo se ejecutará cuando Gráficos RT está inactivo.

Clonar Clonar no se dirigirá a Gráficos RT.

40.5 Descripción formal

RTGraphics Class	-->	Visible Class,
		ComponentTag
		Termination?
ComponentTag	-->	INTEGER
Termination	-->	freeze disappear

41 Clase Interactuable

Descripción: Define la funcionalidad asociada con un comportamiento de interacción de objetos Visibles

Clase básica: Ninguna (clase mezclada)

Subclases: Deslizador, Hipertexto, Campo de entrada, Botón

Estado: Clase abstracta

41.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

41.1.1 Atributos heredados

Esta clase no tiene atributos heredados.

41.1.2 Atributos intercambiados propios

Esta clase define los siguientes atributos intercambiados:

Responsabilidad de motor Determina dónde está la responsabilidad de generar retroinformación visual al usuario como resultado de cambios del atributo interno *Estado de realce*.

(EngineResp) Si *Responsabilidad de motor* está puesta a verdadero, el motor MHEG-5 generará retroinformación visual al usuario cuando hay un cambio de estado en el atributo interno *Estado de realce*. Cuando *responsabilidad de motor* está puesta a Falso, el motor no generará retroinformación visual.

La naturaleza exacta de la retroinformación visual está fuera del alcance de la presente Recomendación.

- Booleano facultativo.
- Valor por defecto: Verdadero.

Color de referencia de realce (HighlighRefColour) Color de referencia para la retroinformación visual generada cuando el atributo *Estado de realce* es verdadero. El color real para reproducir el realce está fuera del alcance de la presente Recomendación; sin embargo, se recomienda que el motor MHEG-5 trate de reproducir el realce utilizando *Color de referencia de realce*.

Si el atributo Referencia de paleta original está codificado, el atributo Color de referencia de realce será un índice basado en cero de un color en la tabla de mejora de colores definida por esa Paleta. En los demás casos, será una Cadena de octetos que codifica un color real.

- Entero facultativo o Cadena de octetos.
- Valor por defecto: Valor codificado por aplicación.

41.1.3 Atributos internos propios

Esta clase define los siguientes atributos internos.

Estado de realce (HighlighStatus) Este atributo está asociado con un determinado tipo de retroinformación visual al usuario.

Cuando *Estado de realce* y *Responsabilidad de motor* son Verdaderos, el motor MHEG-5 generará una retroinformación visual al usuario, por ejemplo, en forma de una línea dibujada alrededor del objeto Interactuable. En los demás casos, no se generará retroinformación. Para la generación de esta retroinformación visual, se puede usar Color de referencia de realce.

NOTA 1 – Aunque esta retroinformación visual no cambia el comportamiento del objeto, este atributo se puede utilizar para señalar al usuario que el objeto Interactuable está preparado para interactuar, por ejemplo, al implementar una navegación salto-realce.

NOTA 2 – La única manera de cambiar el atributo *Estado de realce* es mediante la acción Fijar estado de realce.

- Valor booleano.
- Valor inicial: Falso.

Estado de interacción (InteractionStatus) Este atributo describe si el usuario está interactuando o no con el objeto Interactuable.

Si *Estado de interacción* es falso, el usuario no está interactuando en ese momento con el objeto Interactuable. El tratamiento de eventos de entrada de usuario se produce normalmente. Véase la cláusula 53.

Si *Estado de interacción* es verdadero, el usuario está interactuando con el objeto Interactuable. En consecuencia, ningún evento del tipo Entrada de usuario será generado por el objeto Escena activo. Estos eventos serán tratados directamente por el objeto Interactuable.

En cualquier momento, como máximo un objeto Interactuable tendrá su *Estado de interacción* puesto a Verdadero.

NOTA 1 – La única manera de cambiar el atributo interno *Estado de interacción* es mediante la acción Fijar estado de interacción.

NOTA 2 – Aunque los Enlaces que activan eventos de Entrada de usuario no pueden disparar cuando el atributo interno *Estado de interacción* es Verdadero, otros Enlaces

pueden disparar. Esto hace posible, por ejemplo, aplicar temporizaciones para el proceso de interacción.

- Valor booleano.
- Valor inicial: Falso.

41.2 Eventos

Esta clase define los siguientes eventos:

Interacción completada (Interaction-Completed) Este evento es generado como resultado de un cambio de un objeto Interactuable. El evento es generado solamente una vez por interacción, es decir, es generado solamente cuando el atributo interno *Estado de interacción* vuelve a Falso después de una interacción.

- Ningún dato asociado.

Realce activado (HighlightOn) Este evento es generado cuando el atributo *Estado de realce* del objeto Interactuable cambia de Falso a Verdadero.

- Ningún dato asociado.

Realce desactivado (HighlightOff) Este evento es generado cuando el atributo *Estado de realce* del objeto Interactuable cambia de Verdadero a Falso.

- Ningún dato asociado.

Entrada de cursor (CursorEnter) Este evento es generado automáticamente por el motor MHEG-5 solamente si:

- 1) El motor aplica la opción Cursor de movimiento libre.
- 2) El cursor en movimiento ha entrado en la zona definida por el objeto Interactuable.
- 3) El objeto Interactuable está activo.

NOTA – Para más detalles, véase 53.6 sobre los eventos *Salida de cursor* y *Entrada de cursor*.

Salida de cursor (CursorLeave) Este evento es generado automáticamente por el motor MHEG-5 solamente si:

- 1) El motor aplica la opción Cursor de movimiento libre.
- 2) El cursor en movimiento ha dejado la zona definida por el objeto Interactuable.
- 3) El objeto Interactuable está activo.

NOTA – Para más detalles sobre la generación de los eventos *Salida de cursor* y *Entrada de cursor*, véase 53.6 sobre los mecanismos principales.

41.3 Comportamientos internos

Esta clase define el siguiente comportamiento interno:

Interacción Ejecutar los siguientes pasos sincronamente:

- 1) Fijar el atributo interno *Estado de interacción* a Verdadero.
- 2) Generar retroinformación visual al usuario que señala el hecho de que puede interactuar con un objeto interactuable.

NOTA – La mayoría de las subclases Interactuables amplían este comportamiento.

41.4 Efectos de las acciones MHEG-5

Esta clase define las siguientes acciones MHEG-5 aplicables:

Fijar estado de interacción

(*Nuevo estado de interacción*)

[SetInteraction-Status

(*NewInteraction-Status*)]

Esta acción se utiliza para influir el atributo interno *Estado de interacción*.

Ejecutar la siguiente secuencia de acciones:

- 1) Si *Nuevo estado de interacción* está puesto a verdadero:
 - a) el Interactable objetivo o cualquier otro Interactable en la Escena vigente tiene su atributo *Estado de interacción* puesto a Verdadero, descartar la opción. En los demás casos:
 - b) aplicar el comportamiento *Interacción* del Interactable objetivo.
- 2) Si *Nuevo estado de interacción* está puesto a Falso:
 - a) si el Interactable objetivo tiene su *Estado de interacción* puesto a Falso, descartar la acción. En los demás casos:
 - b) interrumpir inmediatamente el comportamiento *Interacción* que se está produciendo para el objeto Interactable objetivo. El estado del objeto interactuable objetivo después que el comportamiento *Interacción* es interrumpido reflejará cualquier interacción que se ha producido hasta el punto donde se interrumpió la interacción,
 - c) generar el evento *Interacción completada*.

Disposición de uso:

- El objeto *Objetivo* será un objeto Interactable activo.

Descripción de sintaxis:

SetInteractionStatus	-->	Target ,
		NewInteractionStatus
Target	-->	GenericObjectReference
NewInteractionStatus	-->	GenericBoolean

Obtener estado de interacción

(*Variable estado de interacción*)

[GetInteractionStatus

(*InteractionStatus-Var*)]

Fijar la Variable referenciada por *Variable estado de interacción* al valor del atributo *Estado de interacción*.

Disposiciones de uso:

- El objeto *Objetivo* estará disponible.
- *Variable Estado de interacción* hará referencia a un objeto Variable booleana.

Descripción de sintaxis:

GetInteractionStatus	-->	Target ,
		InteractionStatusVar
Target	-->	GenericObjectReference
InteractionStatusVar	-->	ObjectReference

Fijar estado de realce
(*Nuevo estado de realce*)

[SetHighlightStatus
(*NewHighlight-Status*)]

Cambiar el estado realzado de un objeto Interactuable.

Ejecutar síncronamente la siguiente secuencia de acciones:

- 1) Si el *Estado de realce* vigente es igual a *Nuevo estado de realce*, descartar la acción. En los demás casos:
- 2) Fijar el atributo *Estado de realce* del objeto Interactuable objetivo a *Nuevo estado de realce*.
- 3) Si el objeto Interactuable objetivo está activo, si Responsabilidad de motor está puesta a Verdadero y si *Estado de realce* está puesto a Verdadero, dibujar de nuevo el objeto Interactuable objetivo para presentar la retroinformación visual asociada con el estado realzado.
- 4) Si el objeto Interactuable objetivo está activo, si Responsabilidad de motor está puesto a Verdadero y si *Estado de realce* está puesto a Falso, dibujar de nuevo el objeto Interactuable objetivo para suprimir la retroinformación visual asociada con el estado realzado.

Disposición de uso:

- El objeto *Objetivo* será un objeto Interactuable disponible.

Descripción de sintaxis:

SetHighlightStatus	-->	Target , NewHighlightStatus
Target	-->	GenericObjectReference
NewHighlightStatus	-->	GenericBoolean

Obtener estado de realce
(*Variable estado de realce*)

[GetHighlightStatus
(*HighlightStatus-Var*)]

Fijar la Variable referenciada por *Variable estado de realce* al valor del atributo *Estado de realce*.

Disposiciones de uso:

- El objeto *Objetivo* estará disponible.
- *Variable Estado de realce* hará referencia a un objeto Variable booleana.

Descripción de sintaxis:

GetHighlightStatus	-->	Target , HighlightStatusVar
Target	-->	GenericObjectReference
HighlightStatusVar	-->	ObjectReference

41.5 Descripción formal

Interactible Class	-->	EngineResp? , HighlightRefColour? ,
EngineResp	-->	BOOLEAN
HighlightRefColour	-->	Colour

42 Clase Deslizador

Descripción:	Define un accesorio de interacción utilizado para fijar una posición dentro de una gama lineal
Clase básica:	Visible, Interactuable
Subclases:	Ninguna
Estado:	Clase concreta

42.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

42.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las siguientes restricciones:

Nombre de atributo	Definido en	Restricciones y requisitos
Gancho de contenido	Ingrediente	Este atributo no se codificará para esta clase.
Contenido original	Ingrediente	Este atributo no se codificará para esta clase.
<i>Posición de casilla,</i> <i>Posición</i>	Visible	Además de definir la casilla delimitante del Deslizador, estos atributos definirán también el tamaño real del Deslizador. Este tamaño será tal que el Deslizador llene completamente su casilla delimitante.

42.1.2 Atributos intercambiados propios

Esta clase define los siguientes atributos intercambiados adicionales:

Orientación (Orientation)	<p>Orientación del eje principal del Deslizador.</p> <p>La orientación especifica el sentido en el cual el Deslizador se mueve del valor mínimo al máximo.</p> <p>Aunque la reproducción exacta del objeto Deslizador no se especifica en detalle en la presente Recomendación, se presentará de modo que su orientación se conforme a este atributo.</p> <ul style="list-style-type: none">Valores posibles: <i>izquierda</i> <i>derecho</i> <i>arriba</i> <i>abajo</i>.
Valor inicial (InitialValue)	<p>El valor inicial del Deslizador, es decir, el valor que tiene cuando es activado por primera vez después que ha sido preparado.</p> <p>El valor será un Entero y concordará con Valor mínimo, Valor máximo y Tamaño de paso como se describe a continuación.</p> <ul style="list-style-type: none">Valor facultativo.Valor por defecto: Valor mínimo.
Valor mínimo (MinValue)	<p>El valor más bajo que puede tener el atributo <i>Valor de deslizador</i>.</p> <p>El valor concordará con Valor inicial, Valor máximo y Tamaño de paso como se describe a continuación.</p> <ul style="list-style-type: none">Entero facultativo.Valor por defecto: 1.

Valor máximo (MaxValue)	<p>Valor máximo que puede tener el atributo <i>Valor de deslizador</i>.</p> <p>El valor concordará con Valor inicial, Valor mínimo y Tamaño de paso como se describe a continuación:</p> <ul style="list-style-type: none"> • Valor entero. • Valor por defecto: Ninguno.
Porción inicial (InitialPortion)	<p>Representa una porción de la gama [Valor mínimo, Valor máximo]. Este valor será menor o igual al número (Valor máximo – Valor mínimo).</p> <p>Este valor se codificará solamente si el valor del atributo Estilo de deslizador es <i>proporcional</i>, en cuyo caso la doble desigualdad siguiente es siempre Verdadera:</p> $\text{Valor mínimo} \leq \text{Valor inicial} \leq \text{Valor máximo} - \text{Porción inicial}$ <ul style="list-style-type: none"> • Entero facultativo. • Valor por defecto: Ninguno.
Tamaño de paso (StepSize)	<p>El valor más pequeño en que se puede aumentar o disminuir el valor del atributo interno <i>Valor de deslizador</i>.</p> <p>El valor puede ser un Entero positivo y concordará con Valor inicial, Valor mínimo y Valor máximo de la siguiente manera:</p> <p>Todos los valores que puede tomar el deslizador se expresan como:</p> $v_i = \text{valor mínimo} + i \times \text{tamaño de paso},$ <p>donde $i \in \{0 .. (\text{valor máximo} - \text{valor mínimo}) / \text{tamaño de paso}\}$.</p> <p>Los atributos Valor inicial, Valor mínimo, Valor máximo y Tamaño de paso se conformarán con los criterios siguientes:</p> <p>Valor inicial = Valor mínimo + M × tamaño de paso, para algunos $M \in \{0 .. (\text{Valor máximo} - \text{Valor mínimo}) / \text{Tamaño de paso}\}$.</p> <p>Valor mínimo < Valor máximo.</p> <p>$N \times \text{Tamaño de paso} = (\text{Valor máximo} - \text{Valor mínimo})$, donde N es algún Entero positivo.</p> <ul style="list-style-type: none"> • Entero facultativo. • Valor por defecto: 1.
Estilo de deslizador (SliderStyle)	<p>Este atributo puede adoptar el valor <i>valor</i>, <i>termómetro</i> y <i>proporcional</i>. El atributo <i>Estilo de deslizador</i> influye en la presentación del Deslizador de la siguiente manera:</p> <ul style="list-style-type: none"> • Si el <i>Estilo de deslizador</i> está puesto a <i>normal</i>, el Deslizador se reproduce como un «marcador», que se coloca en un «eje principal» en la posición correspondiente al atributo Valor de deslizador. • Si el <i>Estilo de deslizador</i> está puesto a <i>termómetro</i>, el Deslizador se reproduce como un «eje principal» que se rellena desde su comienzo en la posición correspondiente al atributo <i>Valor de deslizador</i>.

- Si el *Estilo de deslizador* está puesto a *proporcional*, el Deslizador se reproduce como un «eje principal» que se rellena desde la posición correspondiente al atributo *Valor de deslizador* hasta la posición correspondiente a la suma de los atributos internos *Valor de deslizador* y *Porción*.

Esta Recomendación no especifica exactamente cómo debe aparecer esta presentación. El siguiente cuadro se proporciona solamente a modo de ejemplo:

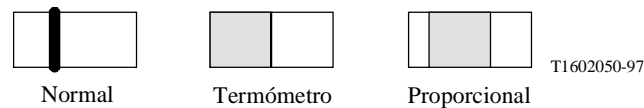


Figura 13 – Ejemplos de Deslizadores en diferentes *Estilo de deslizador*

- Atributo facultativo.
- Posibles valores: *normal* / *termómetro* / *proporcional*.
- Valor por defecto: *normal* – Especificar un color que puede ser utilizado por el motor para presentar el objeto deslizador.

Color de deslizador (SliderColour) El valor de *Color de deslizador* se expresa como un valor de color absoluto o como un índice basado en cero en una tabla de mejora de colores. En el último caso, el objeto Deslizador debe tener codificado el atributo *Referencia de paleta*, que se utiliza para traducir el índice a un valor de color real.

La presente Recomendación no especifica exactamente cómo este color se utiliza para reproducir el Deslizador. Se proporciona como una indicación al motor MHEG-5 sobre el esquema de colores que se ha de utilizar para presentar el Deslizador.

La resolución de color real en el proceso de presentación está fuera del alcance de la presente Recomendación.

- Atributo facultativo.
- Valor por defecto: Valor codificado por la Aplicación.

42.1.3 Atributos internos propios

Esta clase define los siguientes atributos internos adicionales:

Valor de deslizador (SliderValue) El valor vigente del deslizador.

Porción (Portion) El estado vigente del atributo que rige la presentación del Deslizador cuando el *Estilo del deslizador* es *proporcional*.

Si el *Estilo de deslizador* está puesto a *proporcional*, el Deslizador se reproduce como un «eje principal» que se rellena desde la posición correspondiente al atributo *Valor de deslizador* hasta la posición correspondiente a la suma de los atributos internos *Valor de deslizador* y *Porción*.

42.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

42.3 Comportamientos internos

La semántica de los siguientes comportamientos internos ha cambiado con respecto a la clase básica de este objeto:

<i>Interacción</i>	Ejecutar la siguiente secuencia de acciones:
	1) Aplicar el comportamiento <i>Interacción</i> definido en la clase <i>Interactuable</i> .
	2) Permitir al usuario que interactúe con el objeto <i>Deslizador</i> moviendo el marcador a lo largo del eje principal. Esta Recomendación no especifica exactamente cómo se produce esta interacción de usuario. Sin embargo, el desplazamiento más pequeño del marcador será proporcional al valor del atributo <i>Tamaño de paso</i> .
	3) Cuando el marcador se ha detenido en una nueva posición:
	a) fijar el atributo <i>Valor de deslizador</i> a un valor que corresponda con la nueva posición de marcador;
	b) fijar el atributo <i>Estado de interacción</i> a Falso, y
	c) generar un evento <i>Interacción completada</i> .

42.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se definen las siguientes acciones MHEG-5 aplicables:

Paso	Fijar un nuevo valor a un deslizador con respecto a su valor vigente.
(Número de pasos)	Ejecutar la siguiente secuencia de acciones:
[Step (NbOfSteps)]	1) Si <i>Número de pasos</i> es positivo, aumentar el valor de <i>Valor de deslizador</i> por $\text{Número de pasos} \times \text{Tamaño de paso}$.
	2) Si <i>Número de pasos</i> es negativo, disminuir el valor de <i>Valor de deslizador</i> por $\text{Número de pasos} \times \text{Tamaño de paso}$.
	3) Si el <i>Deslizador</i> objetivo está activo, dibujar de nuevo el <i>Deslizador</i> utilizando el nuevo valor de <i>Valor de deslizador</i> y de acuerdo con su posición en la <i>Pila de visualización</i> del objeto <i>Aplicación</i> activo.
	4) Generar un evento <i>Interacción completada</i> .
	Disposiciones de uso:
	• El objeto <i>Objetivo</i> será un objeto <i>Deslizador</i> disponible.
	• <i>Número de pasos</i> se fijará de modo que:
	$\text{Valor mínimo} \leq (\text{Número de pasos} \times \text{Valor de paso}) + \text{Valor de deslizador},$
	y
	$(\text{Número de pasos} \times \text{Valor de pasos}) + \text{Valor de deslizador} + \text{Porción} \leq \text{valor máximo}$
	En las expresiones anteriores se considerará que el valor de <i>Porción</i> es 0 si <i>Estilo de deslizador</i> no es proporcional.

Descripción de sintaxis:

Step	-->	Target , NbOfSteps
Target	-->	GenericObjectReference
NbOfSteps	-->	GenericInteger

Fijar valor de
deslizador
(*Nuevo valor de
deslizador*)

[SetSliderValue
(*NewSliderValue*)]

Fijar un valor absoluto a un deslizador.

Ejecutar la siguiente secuencia de acciones:

- 1) Fijar el atributo *Valor de deslizador* del Deslizador objetivo a *Nuevo valor de deslizador*.
- 2) Si el Deslizador objetivo está activo, dibujarlo de nuevo teniendo en cuenta el nuevo valor de *Valor de deslizador* y de acuerdo con su posición en la *Pila de visualización* del objeto Aplicación activo.
- 3) Generar un evento *Interacción completada*.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Deslizador disponible.
Nuevo valor de deslizador estará en la gama [Valor mínimo, Valor máximo – *Porción*].
El valor de *Porción* en la expresión anterior se considerará que es 0 si Estilo de deslizador no es proporcional.
- (*Nuevo valor de deslizador* – Valor mínimo) MOD Tamaño de paso será igual a 0.

Descripción de sintaxis:

SetSliderValue	-->	Target , NewSliderValue
Target	-->	GenericObjectReference
NewSliderValue	-->	GenericInteger

Obtener valor de
deslizador
(*Variable valor
de deslizador*)

[GetSliderValue
(*SliderValueVar*)]

Fijar la Variable referenciada por *Variable valor de deslizador* al valor del atributo *Valor de deslizador*.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Deslizador disponible.
- *Variable valor de deslizador* hará referencia a un objeto Variable entera.

Descripción de sintaxis:

GetSliderValue	-->	Target , SliderValueVar
Target	-->	GenericObjectReference
SliderValueVar	-->	ObjectReference

Fijar porción
(*Nueva porción*)

[SetPortion
(*NewPortion*)]

Fijar el tamaño de la porción representada por un deslizador de estilo *Proporcional*.

Ejecutar la siguiente secuencia de acciones:

- 1) Fijar el atributo *Porción* del Deslizador objetivo a *Nueva porción*.
- 2) Si el Deslizador objetivo está activo, dibujar de nuevo el Deslizador de acuerdo con el nuevo valor de *Porción* y de acuerdo con su posición en la *Pila de visualización* del objeto Aplicación activo.
- 3) Generar un evento *Interacción completada*.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Deslizador disponible de estilo *proporcional*.
- *Nueva porción* será menor o igual que (Valor máximo – Valor de deslizador).

Descripción de sintaxis:

SetPortion	-->	Target , NewPortion
Target	-->	GenericObjectReference
NewPortion	-->	GenericInteger

Obtener porción
(*Variable porción*)

[GetPortion
(*PortionVar*)]

Fijar la Variable referenciada por *Variable porción* al valor del atributo *Porción*.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Deslizador disponible de estilo *proporcional*.
- *Variable porción* hará referencia a un objeto Variable entera activo.

Descripción de sintaxis:

GetPortion	-->	Target , PortionVar
Target	-->	GenericObjectReference
PortionVar	-->	ObjectReference

42.5 Descripción formal

Slider Class	-->	Visible Class, Interactable Class, Orientation, Max Value, Min Value?, InitialValue?, InitialPortion?, StepSize?, SliderStyle?, SliderColour?
Orientation	-->	left right up down
InitialValue	-->	INTEGER
InitialPortion	-->	INTEGER
Min Value	-->	INTEGER
Max Value	-->	INTEGER
StepSize	-->	INTEGER
SliderStyle	-->	normal thermometer proportional
SliderColour	-->	Colour

43 Clase Campo de entrada

Descripción: Define un accesorio de interacción utilizado por el usuario final para editar y/o modificar un texto

Clase básica: Texto, Interactable

Subclases: Ninguna

Estado: Clase concreta

43.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

43.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con la misma semántica.

43.1.2 Atributos intercambiados propios

Esta clase define los siguientes atributos intercambiados adicionales:

Tipo de entrada (InputType) Tipo de caracteres autorizados.

Si el atributo está puesto a *alfa*, sólo se aceptarán como entradas caracteres que no son cifras (es decir, 0123456789). Si este atributo está puesto a *numérico*, sólo se aceptarán como entradas caracteres que son cifras. Si este atributo está puesto a *cualquiera*, se aceptarán todos los caracteres. Si este atributo está puesto a *enumerado*, sólo se aceptarán los caracteres que se proporcionan en el atributo Lista de caracteres del Campo de entrada. Esto proporciona la posibilidad de adaptar el filtro de entrada.

NOTA – El valor *cualquiera* permite también la entrada de símbolos (por ejemplo, &*\$#@!).

- Valores posibles: *alfa* | *numérico* | *cualquiera* | *enumerado*.
- Valor por defecto: *cualquiera*.

Lista de caracteres (CharList)	<p>Los caracteres que pudieran ser introducidos en este Campo de entrada.</p> <p>Este atributo estará codificado siempre cuando Tipo de entrada está puesto a <i>enumerado</i>. En los demás casos no se codificará.</p> <ul style="list-style-type: none">• Cadena de octetos facultativa.• Valor por defecto: Ninguno.
Entrada ocultada (ObscuredInput)	<p>Indica cómo devolver en eco caracteres de entrada. Esto se puede utilizar para la entrada de contraseñas.</p> <p>Si este atributo se pone a Verdadero, los caracteres introducidos serán devueltos en eco en la pantalla en una forma ilegible. En los demás casos, los caracteres introducidos serán devueltos en eco a la pantalla en una forma legible.</p> <ul style="list-style-type: none">• Valor booleano facultativo.• Valor por defecto: Falso.
Longitud máxima (MaxLength)	<p>Proporciona el número máximo de caracteres introducidos previstos. Cuando se alcanza este número máximo de caracteres en el Campo de entrada, se genera un evento Campo de entrada lleno.</p> <p>Si <i>Longitud máxima</i> se pone a 0, el número de caracteres previsto es indefinido. En este caso, el motor MHEG-5 proporcionará al usuario algunos medios para terminar la interacción.</p> <ul style="list-style-type: none">• Entero facultativo.• Valor por defecto: 0.

43.1.3 Atributos internos propios

Esta clase define los siguientes atributos internos adicionales:

<i>Punto de entrada</i> (EntryPoint)	<p>Define (como un índice basado en cero) dónde se ha de colocar el siguiente carácter que se ha de introducir en el Campo de entrada.</p> <ul style="list-style-type: none">– Si este atributo es igual a 0, el siguiente carácter se colocará antes del primer carácter del Campo de entrada.– Si este atributo es igual o mayor que la longitud del texto vigente en el Campo de entrada, el siguiente carácter se añadirá al final del Campo de entrada.– Para todos los otros valores <i>n</i> de este atributo, el siguiente carácter se insertará después del <i>n</i> carácter del Campo de entrada. <ul style="list-style-type: none">• Valor entero.• Valor inicial: 0.
---	--

<i>Modo sobreescritura (OverwriteMode)</i>	<p>Determina si los nuevos caracteres introducidos son escritos sobre los caracteres del texto existente o si son insertados entre ellos.</p> <p>Si este atributo es Verdadero, cada carácter introducido en el Campo de entrada sustituye (sobreescribe) al carácter que estaba colocado previamente en <i>Punto de entrada</i> y todos los demás caracteres no son alterados.</p> <p>Si este atributo es Falso, la entrada se inserta exactamente antes del carácter colocado en el <i>Punto de entrada</i>.</p> <p>Si <i>Punto de entrada</i> está puesto a un valor igual o mayor que la longitud del texto del Campo de entrada, los caracteres que son introducidos se añaden al final del Campo de entrada, con independencia del valor de este atributo.</p> <ul style="list-style-type: none"> • Booleano. • Valor por defecto: Falso.
--	---

43.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica. Además, se definen los siguientes eventos:

<i>Interacción completada (Interaction- Completed)</i>	<p>Este evento se genera como resultado del fin del proceso de modificación de un Campo de entrada.</p>
<i>Campo de entrada lleno (EntryFieldFull)</i>	<p>Este evento se genera cuando se ha alcanzado la capacidad del Campo de entrada. Sólo será generado cuando el atributo Longitud máxima está codificado, en cuyo caso se genera cuando el número de caracteres del Campo de entrada alcanza la Longitud máxima.</p> <p>El evento es asíncrono.</p> <p>Una acción Fijar datos no generará eventos Campo de entrada lleno.</p> <ul style="list-style-type: none"> • Ningún dato asociado.

43.3 Comportamientos internos

La semántica del siguiente comportamiento interno ha cambiado con respecto a la clase básica de este objeto:

<i>Interacción</i>	<p>Ejecutar la siguiente secuencia de acciones:</p> <ol style="list-style-type: none"> 1) Aplicar el comportamiento <i>Interacción</i> definido en la clase Interactuable. 2) Permitir que el usuario modifique el atributo <i>Datos de texto</i> del objeto Campo de entrada introduciendo caracteres. 3) Después que se introduce cada carácter, actualizar el atributo <i>Datos de texto</i> en consecuencia (teniendo en cuenta el valor de los atributos <i>Punto de entrada</i> y <i>Sobreescritura</i>), actualizar el atributo <i>Punto de entrada</i>.
--------------------	--

- 4) Cuando la entrada está completa (porque el usuario termina la entrada o porque la aplicación la termina utilizando la acción Fijar estado de interacción):
 - a) fijar el atributo *Estado de interacción* a Falso;
 - b) generar un evento *Interacción completada*.

43.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se definen las siguientes acciones MHEG-5 aplicables:

Fijar modo sobreescritura (<i>Nuevo modo sobreescritura</i>) [SetOverwriteMode (NewOverwrite-Mode)]	<p>Cambiar un campo de entrada al modo sobreescritura y viceversa.</p> <p>Ejecutar la siguiente secuencia de acciones:</p> <ol style="list-style-type: none"> 1) Si <i>Modo sobreescritura</i> vigente es igual a <i>Nuevo modo sobreescritura</i>, descartar la acción. En los demás casos: 2) Fijar <i>Modo sobreescritura</i> del campo de entrada objetivo a <i>Nuevo modo sobreescritura</i>. 3) Si el Campo de entrada objetivo está activo, actualizar la representación visible del Campo de entrada de acuerdo con el nuevo valor de <i>Modo sobreescritura</i>.
--	--

Disposición de uso:

- El objeto *Objetivo* será un objeto Campo de entrada disponible.

Descripción de sintaxis:

SetOverwriteMode	-->	Target , NewOverwriteMode
Target	-->	GenericObjectReference
NewOverwriteMode	-->	GenericBoolean

Obtener modo sobreescritura (<i>Variable modo sobreescritura</i>) [GetOverwriteMode (OverwriteMode-Var)]	<p>Fijar la Variable referenciada por <i>Variable modo sobreescritura</i> al valor del atributo <i>Modo sobreescritura</i>.</p> <p>Disposiciones de uso:</p> <ul style="list-style-type: none"> • El objeto <i>Objetivo</i> será un objeto Campo de entrada disponible. • <i>Variable modo sobreescritura</i> hará referencia a un objeto Variable booleana activo.
---	---

Descripción de sintaxis:

GetOverwriteMode	-->	Target , OverwriteModeVar
Target	-->	GenericObjectReference
OverwriteModeVar	-->	ObjectReference

Fijar punto de entrada
(*Nuevo punto de entrada*)
[SetEntryPoint
(*NewEntryPoint*)]

Cambiar la posición del punto de entrada de un campo de entrada.
Ejecutar la siguiente secuencia de acciones:
1) Fijar *Punto de entrada* del Campo de entrada objetivo a *Nuevo punto de entrada*.
2) Si el Campo de entrada objetivo está activo, actualizar la representación visible del Campo de entrada de acuerdo con el nuevo valor de *Punto de entrada*.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Campo de entrada disponible.
- *Nuevo punto de entrada* será mayor o igual que 0.

Descripción de sintaxis:

SetEntryPoint	-->	Target , NewEntryPoint
Target	-->	GenericObjectReference
NewEntryPoint	-->	GenericInteger

Obtener punto de entrada
(*Variable punto de entrada*)
[GetEntryPoint
(*EntryPointVar*)]

Fijar la Variable referenciada por *Variable punto de entrada* al valor del atributo Punto de entrada.

Disposiciones de uso:

- El objeto *Objetivo* será un objeto Campo de entrada disponible.
- *Variable punto de entrada* hará referencia aun objeto Variable entera activo.

Descripción de sintaxis:

GetEntryPoint	-->	Target , EntryPointVar
Target	-->	GenericObjectReference
EntryPointVar	-->	ObjectReference

43.5 Descripción formal

EntryField Class	-->	Text Class , Interactable Class , InputType? , CharList? , ObscuredInput? , MaxLength?
InputType	-->	alpha numeric any listed
CharList	-->	OctetString
ObscuredInput	-->	BOOLEAN
MaxLength	-->	INTEGER

44 Clase Hipertexto

Descripción: La clase Hipertexto es una subclase de la clase Texto, que tiene la propiedad especial de tener en cuenta porciones del texto que han de estar asociadas con una pieza de información. Estas porciones de texto se denominan *anclas* (*anchors*) en el resto de esta cláusula

Clase básica: Texto, Interactuable

Subclases: Ninguna

Estado: Clase concreta

44.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

44.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con la misma semántica.

44.1.2 Atributos intercambiados propios

Esta clase no define atributos intercambiados adicionales.

44.1.3 Atributos internos propios

Esta clase define el siguiente atributo interno adicional:

*Última ancla
activada
(LastAnchorFired)* Rótulo de la última ancla activada.

- Cadena de octetos.
- Valor inicial: Cadena vacía.

44.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica. Además, se define el siguiente evento:

*Ancla activada
(AnchorFired)* Señaliza que el usuario ha seleccionado una de las anclas en el objeto Hipertexto.

- Datos asociados: El rótulo del ancla activada – Cadena de octetos.

44.3 Comportamientos internos

La semántica de los siguientes comportamientos internos ha cambiado con respecto a la clase básica de este objeto:

Interacción Ejecutar la siguiente secuencia de acciones:

- 1) Aplicar el comportamiento *Interacción* heredado de la clase Interactuable.
- 2) Permitir que el usuario mueva el enfoque hacia el conjunto de anclas en el objeto Hipertexto y seleccione el ancla enfocada. Cada vez que se selecciona un ancla, se genera un evento *Ancla activada*.

- 3) Cuando la acción se ha completado (porque el usuario termina la interacción o porque la aplicación la termina utilizando la acción Fijar estado de interacción):
 - a) poner el atributo *Estado de interacción* a falso;
 - b) generar un evento *Interacción completada*.

44.4 Efecto de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se define la siguiente acción MHEG-5 aplicable:

Obtener última ancla activada (<i>Variable última ancla activada</i>) [GetLastAnchorFired (<i>LastAnchorFiredVar</i>)]	Fijar la Variable referenciada por <i>Variable última ancla activada</i> al valor del atributo <i>Última ancla activada</i> . Disposiciones de uso: <ul style="list-style-type: none"> • El <i>objeto Objetivo</i> será un objeto Hipertexto disponible. • <i>Última ancla activada</i> hará referencia a un objeto variable Cadena de octetos.
---	--

Descripción de sintaxis:

GetLastAnchorFired	-->	Target ,
		LastAnchorFiredVar
Target	-->	GenericObjectReference
LastAnchorFiredVar	-->	ObjectReference

44.5 Descripción formal

HyperText Class	-->	Text Class ,
		Interactive Class

45 Clase Botón

Descripción: Define la funcionalidad asociada con la presentación e interacción con botones de un estado y dos estados

Clase básica: Visible, Interactuable

Subclases: Punto focal, Botón de pulsación

Estado: Clase abstracta

45.1 Atributos

Esta subclase define atributos heredados, intercambiados e internos para esta clase.

45.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con las siguientes restricciones:

Nombre de atributo	Definido en	Restricciones y requisitos
Gancho de contenido	Ingrediente	Este atributo no se codificará para esta clase.
Contenido original	Ingrediente	Este atributo no se codificará para esta clase.
<i>Estado de interacción</i>	Interactuable	Este atributo interno no se define para la clase Botón.
<i>Tamaño de casilla,</i> <i>Posición</i>	Visible	Además de definir la casilla delimitante del Botón, estos atributos definen también el tamaño real del Botón. Este tamaño será tal que el Botón llene completamente su casilla delimitante.

45.1.2 Atributos intercambiados propios

Esta clase define el siguiente atributo intercambiado adicional:

Color de botón
(ButtonColour)

Especificar un color que puede ser utilizado por el motor para reproducir el objeto Botón.

El valor del atributo *Color de botón* se expresa como un valor de color absoluto o como un índice basado en cero en una tabla de mejora de colores. En el último caso, el objeto Botón debe tener codificado el atributo *Referencia de paleta*, que se utiliza para traducir el índice a un valor de color real.

La presente Recomendación no especifica exactamente cómo se utiliza este color para reproducir el Botón. Se proporciona como una indicación al motor MHEG-5 sobre el esquema de colores que se ha de utilizar para reproducir el Botón.

La resolución de color real en el proceso de presentación está fuera del alcance de la presente Recomendación.

- Atributo facultativo.
- Valor por defecto: Valor codificado por la aplicación.

45.1.3 Atributos internos propios

Esta clase define el siguiente atributo interno adicional:

Estado de selección
(*SelectionStatus*)

Cada botón puede almacenar un bit de información. Este atributo es Verdadero cuando el objeto Botón está en el estado seleccionado. Se pasa al estado seleccionado como resultado de la invocación de la acción Seleccionar.

- Valor booleano.
- Valor inicial: Falso.

45.2 Eventos

Seleccionado
(*IsSelected*)

Este evento es generado cuando el *Estado de selección* del Botón cambia de Falso a Verdadero.

- Ningún asociado.

Deseleccionado
(*IsDeselected*)

Este evento es generado cuando el *Estado de selección* del Botón cambia de Verdadero a Falso.

- Ningún dato asociado.

45.3 Comportamientos internos

La semántica de los siguientes comportamientos internos ha cambiado con respecto a la clase básica de este objeto:

<i>Interacción</i> (<i>Interaction</i>)	Este comportamiento no se define para la clase Botón. NOTA – Como resultado, la clase Botón no generará ningún evento interacción completada.
<i>Selección</i> (<i>Selection</i>)	Ejecutar la siguiente secuencia de acciones: <ol style="list-style-type: none">1) Fijar <i>Estado de selección</i> a Verdadero.2) Si el atributo <i>Responsabilidad de motor</i> es Verdadero y el Botón está activo, dibujar de nuevo el objeto botón teniendo en cuenta el nuevo valor de Estado de selección y de acuerdo con su posición en la pila de visualización del objeto Aplicación activo.3) Generar un evento <i>Seleccionado</i>.
<i>Deselección</i> (<i>Deselection</i>)	Ejecutar la siguiente secuencia de acciones: <ol style="list-style-type: none">1) Fijar <i>Estado de selección</i> a Falso.2) Si el atributo <i>Responsabilidad de motor</i> es Verdadero y el Botón está activo, dibujar de nuevo el objeto botón teniendo en cuenta el nuevo valor de Estado de selección y de acuerdo con su posición en la <i>Pila de visualización</i> del objeto Aplicación activo.3) Generar un evento <i>Deseleccionado</i>.

45.4 Efecto de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se definen las siguientes acciones MHEG-5 aplicables:

Fijar estado de interacción	Esta acción no será dirigida a un objeto Botón.
Obtener estado de interacción	Esta acción no será dirigida a un objeto Botón.
Seleccionar	<p>Ejecutar la siguiente secuencia de acciones:</p> <ol style="list-style-type: none">1) Si <i>Estado de selección</i> está actualmente puesto a Verdadero, omitir esta acción.2) Si <i>Estado de selección</i> está puesto actualmente a Falso, aplicar el comportamiento <i>selección</i> del Botón objetivo. <p>Disposición de uso:</p> <ul style="list-style-type: none">• El objeto <i>Objetivo</i> será un Botón disponible.

Descripción de sintaxis:

Select	-->	Target
Target	-->	GenericObjectReference

- Deseleccionar Ejecutar la siguiente secuencia de acciones:
- 1) Si *Estado de selección* está puesto a Falso, omitir esta acción.
 - 2) Si *Estado de selección* está puesto a Verdadero, aplicar el comportamiento *Deselección* del Botón objetivo.

Disposición de uso:

- El objeto *Objetivo* será un Botón disponible.

Descripción de sintaxis:

Deselect	-->	Target
Target	-->	GenericObjectReference

45.5 Descripción formal

Button Class	-->	Visible Class, Interactable Class, ButtonColour?
ButtonColour	-->	Colour

46 Clase Punto focal

Descripción: Define zonas rectangulares no etiquetadas invisibles en la pantalla que pueden interactuar con el usuario para producir eventos *Seleccionado*

Clase básica: Botón

Subclases: Ninguna

Estado: Clase concreta

46.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

46.1.1 Atributos heredados

Esta clase tiene todos los atributos de sus clases básicas, con semántica idéntica, salvo para el atributo siguiente:

Estado de selección
(*SelectionStatus*) La presentación de un objeto Punto focal dependerá del atributo *Estado de selección*. Cuando este atributo y el atributo Responsabilidad de motor son verdaderos, el Punto focal se reproducirá de manera que señalice al usuario que se ha efectuado la selección. Para este proceso de presentación, se puede utilizar el atributo Color de botón. En los demás casos, el punto focal no tendrá representación visual, salvo que esta reproducción sea prescrita por el Estado de realce de su clase básica.

46.1.2 Atributos intercambiados propios

Esta clase no define atributos intercambiados adicionales.

46.1.3 Atributos internos propios

Esta clase no define atributos internos adicionales.

46.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

46.3 Comportamientos internos

Esta clase tiene los mismos comportamientos internos que su clase básica, con semántica idéntica.

46.4 Efecto de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. La siguiente semántica de acciones MHEG-5 ha cambiado con respecto a la clase básica de este objeto:

Seleccionar Ejecutar la siguiente secuencia de acciones:

- 1) Aplicar el comportamiento *Selección* definido en la clase básica.
- 2) Aplicar el comportamiento *Deselección* definido en la clase básica.

Las disposiciones de uso y descripción de sintaxis de la acción no se han modificado.

46.5 Descripción formal

Hotspot Class	-->	Button Class
---------------	-----	--------------

47 Clase Botón de pulsación

Descripción: Define zonas ampliamente rectangulares etiquetadas en la pantalla que pueden interactuar con el usuario para producir eventos *Seleccionado*

Clase básica: Botón

Subclases: Botón de conmutación

Estado: Clase concreta

47.1 Atributos

Esta subcláusula define atributos heredados, intercambiados e internos para esta clase.

47.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con semántica idéntica, salvo para el siguiente atributo:

<i>Estado de selección</i> (<i>SelectionStatus</i>)	La reproducción de un objeto Botón de pulsación dependerá del atributo <i>Estado de selección</i> . Cuando el atributo <i>Estado de selección</i> y el atributo Responsabilidad de motor son Verdaderos, Botón de pulsación se presentará de manera que señalice al usuario que se ha efectuado la selección. Esta presentación mostrará un botón que ha sido pulsado. En los demás casos, el Botón se reproducirá de manera que señalice al usuario que no se ha efectuado la selección. Esta presentación mostrará un botón que no ha sido pulsado.
--	---

47.1.2 Atributos intercambiados propios

Esta clase define los siguientes atributos intercambiados adicionales:

Etiqueta original (OriginalLabel)	<p>Pieza de texto en línea que representa la etiqueta inicial de un Botón de pulsación.</p> <ul style="list-style-type: none"> • Cadena de octetos facultativa. • Valor por defecto: Cadena vacía.
Juego de caracteres (CharacterSet)	<p>Identificación del juego de caracteres o del conjunto de juegos de caracteres que serán utilizados por defecto para reproducir la etiqueta. Este Entero se codificará con un valor que representa el juego de caracteres. El dominio de aplicación definirá una gama de Juegos de caracteres y su semántica.</p> <p>NOTA – El atributo <i>Juego de caracteres</i> proporciona el juego de caracteres inicial para una etiqueta. Además, el formato de codificación de texto puede contener secuencias de escape para conmutar entre juegos de caracteres.</p> <ul style="list-style-type: none"> • Entero facultativo. • Valor por defecto: El valor del atributo <i>Juego de caracteres</i> del objeto Aplicación, si se especifica ese atributo.

47.1.3 Atributos internos propios

Esta clase define el siguiente atributo interno adicional:

<i>Etiqueta</i> (Label)	<p>Etiqueta del Botón de pulsación.</p> <ul style="list-style-type: none"> • Cadena de octetos facultativa. • Valor inicial: Valor del atributo etiqueta original.
----------------------------	--

47.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

47.3 Comportamientos internos

Esta clase tiene todos los comportamientos internos de su clase básica, con semántica idéntica.

47.4 Efecto de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con los siguientes cambios y extensiones:

Seleccionar (Select)	<p>Ejecutar la siguiente secuencia de acciones:</p> <ol style="list-style-type: none"> 1) Aplicar el comportamiento <i>Selección</i> definido en la clase básica. 2) Aplicar el comportamiento <i>Deselección</i> definido en la clase básica. <p>Las disposiciones de uso y la descripción de sintaxis de la acción no se modifican.</p>
Fijar etiqueta (Nueva etiqueta) [SetLabel (NewLabel)]	<p>Cambiar la etiqueta de un Botón de pulsación.</p> <p>Ejecutar la siguiente secuencia de acciones:</p> <ol style="list-style-type: none"> 1) Fijar el atributo <i>Etiqueta</i> del Botón de pulsación objetivo a <i>Nueva etiqueta</i>. 2) Si el Botón de pulsación está activo, dibujarlo de nuevo teniendo en cuenta el nuevo valor de <i>Etiqueta</i> y de acuerdo con su posición en la <i>Pila de visualización</i> del objeto Aplicación activo.

Disposición de uso:

- El objeto *Objetivo* será un Botón de pulsación disponible.

Descripción de sintaxis:

SetLabel	-->	Target , NewLabel
Target	-->	GenericObjectReference
NewLabel	-->	GenericOctetString

Obtener etiqueta (Variable etiqueta) Fijar la Variable referenciada por *Variable etiqueta* al valor del atributo *Etiqueta*.

[GetLabel (LabelVar)] Disposición de uso:

- *Variable etiqueta* hará referencia a un objeto variable Cadena de octetos activo.

Descripción de sintaxis:

GetLabel	-->	Target , LabelVar
Target	-->	GenericObjectReference
LabelVar	-->	ObjectReference

47.5 Descripción formal

PushButton Class	-->	Button Class , OriginalLabel? , CharacterSet?
OriginalLabel	-->	OctetString

48 Clase Botón de conmutación

Descripción: Define una estructura de datos para tratar zonas ampliamente rectangulares etiquetadas en la pantalla que pueden interactuar con el usuario para producir eventos *Seleccionado* y *Deseleccionado*

Clase básica: Botón de pulsación

Subclases: Ninguna

Estado: Clase concreta

48.1 Atributos

Esta clase tiene todos los atributos de su clase básica, con semántica idéntica, salvo el atributo siguiente:

Estado de selección La presentación de un objeto Botón de conmutación dependerá del atributo *Estado de selección*.

Cuando los atributos *Estado de selección* y *Responsabilidad de motor* son Verdaderos, el Botón de conmutación se presentará de manera que señalice al usuario que se ha efectuado la selección. Esta presentación mostrará un botón radio o una casilla de marca que ha sido seleccionada, o un botón de pulsación que ha sido pulsado, según el atributo *Estilo de botón*. En los demás casos, el Botón se presentará de manera que señalice al usuario que no se ha efectuado la selección. Esta presentación mostrará un botón radio o una casilla que no ha sido seleccionada o un botón que no ha sido pulsado.

48.1.1 Atributos heredados

Esta clase tiene todos los atributos de su clase básica, con semántica idéntica.

48.1.2 Atributos intercambiados propios

Esta clase define el siguiente atributo intercambiado adicional:

Estilo de botón (ButtonStyle)	Estilo de presentación del Botón de conmutación. • Posibles valores: <i>botón de pulsación</i> <i>botón radio</i> <i>casilla de selección</i> .
----------------------------------	--

48.1.3 Atributos internos propios

Esta clase no define atributos internos adicionales.

48.2 Eventos

Esta clase tiene los mismos eventos que su clase básica, con semántica idéntica.

48.3 Comportamientos internos

Esta clase tiene los mismos comportamientos internos que su clase básica, con semántica idéntica.

48.4 Efectos de las acciones MHEG-5

Esta clase tiene el mismo conjunto de acciones MHEG-5 que su clase básica, con semántica idéntica. Además, se definen las siguientes acciones MHEG-5 aplicables:

Obtener estado de selección (<i>Variable estado de selección</i>)	Devolver el valor del atributo <i>Estado de selección</i> en forma de un booleano en la Variable referenciada por el parámetro <i>Variable estado de selección</i> .
[GetSelectionStatus (<i>SelectionStatus-Var</i>)]	Disposiciones de uso: <ul style="list-style-type: none"> • El objeto <i>Objetivo</i> será un objeto Botón de conmutación disponible. • <i>Variable estado de selección</i> hará referencia a un objeto Variable booleana activo.

Descripción de sintaxis:

GetSelectionStatus	-->	Target ,
		SelectionStatusVar
Target	-->	GenericObjectReference
SelectionStatusVar	-->	ObjectReference

Seleccionar	Si el atributo <i>Estado de selección</i> es verdadero, omitir esta acción. En los demás casos, invocar el comportamiento <i>Selección</i> .
-------------	--

Deseleccionar Si el atributo *Estado de selección* es Falso, omitir esta acción. En los demás casos, invocar el comportamiento *Deselección*.

Disposición de uso:

- El objeto *Objetivo* será un objeto Botón de conmutación disponible.

Descripción de sintaxis:

Deselect	-->	Target
Target	-->	GenericObjectReference

Bascular (Toggle) Si el atributo *Estado de selección* es Falso, invocar el comportamiento *Selección*. En los demás casos, invocar el comportamiento *Deselección*.

Disposición de uso:

- El objeto *Objetivo* será un objeto Botón de conmutación disponible.

Descripción de sintaxis:

Toggle	-->	Target
Target	-->	GenericObjectReference

Fijar etiqueta (SetLabel)

Ejecutar la siguiente secuencia de acciones:

- 1) Fijar el atributo *Etiqueta* al nuevo valor.
- 2) Si el Botón de conmutación *Objetivo* está activo, dibujar de nuevo el Botón de conmutación teniendo en cuenta el nuevo valor del atributo *Etiqueta* y de acuerdo con su posición en la Pila de visualización del objeto Aplicación activo.

Disposición de uso:

- El objeto *Objetivo* será un objeto Botón de conmutación disponible.

La sintaxis de esta acción se define en la clase Botón de pulsación.

48.5 Descripción formal

SwitchButton Class	-->	PushButton Class,
		ButtonStyle
ButtonStyle	-->	pushbutton radiobutton checkbox

49 Clase Acción

Descripción: La clase Acción es una lista de acciones elementales destinadas a ser ejecutadas síncronamente

La clase Acción no hereda de ninguna clase MHEG-5; como consecuencia, no se puede hacer referencia a objetos Acción individualmente

Clase básica: Ninguna

Subclases: Ninguna

Estado: Clase concreta

49.1 Atributos

Esta subcláusula define los atributos heredados, intercambiados e internos de esta clase.

49.1.1 Atributos heredados

Esta clase no tiene atributos heredados.

49.1.2 Atributos intercambiados propios

Esta clase define el siguiente atributo intercambiado:

Acciones elementales Lista de acciones elementales incluidas.
(ElementaryActions)

49.2 Atributos internos propios

Esta clase no define atributos internos.

49.3 Descripción formal

```
Action Class      --> ElementaryAction+
ElementaryAction  --> Activate | Add | AddItem | Append | BringToFront |
                    Call | CallActionSlot | Clear | Clone |
                    CloseConnection | Deactivate | DelItem | Deselect |
                    DeselectItem | Divide | DrawArc | DrawLine | DrawOval
                    | DrawPolygon | DrawPolyline | DrawRectangle
                    | DrawSector | Fork | GetAvailabilityStatus |
                    GetBoxSize | GetCellItem | GetCursorPosition |
                    GetEngineSupport | GetEntryPoint | GetFillColour |
                    GetFirstItem | GetHighlightStatus |
                    GetInteractionStatus | GetItemStatus | GetLabel |
                    GetLastAnchorFired | GetLineColour | GetLineStyle |
                    GetLineWidth | GetListItem | GetListSize |
                    GetOverwriteMode | GetPortion | GetPosition |
                    GetRunningStatus | GetSelectionStatus | GetSliderValue
                    | GetTextContent | GetTextData | GetTokenPosition |
                    GetVolume | Launch | LockScreen | Modulo | Move |
                    MoveTo | Multiply | OpenConnection | Preload |
                    PutBefore | PutBehind | Quit | ReadPersistent | Run |
                    ScaleBitmap | ScaleVideo | ScrollItems | Select |
                    SelectItem | SendEvent | SendToBack | SetBoxSize |
                    SetCachePriority | SetCounterEndPosition |
                    SetCounterPosition | SetCounterTrigger |
                    SetCursorPosition | SetCursorShape | SetData |
                    SetEntryPoint | SetFillColour | SetFirstItem |
                    SetFontRef | SetHighlightStatus | SetInteractionStatus
                    | SetLabel | SetLineColour | SetLineStyle |
                    SetLineWidth | SetOverwriteMode | SetPaletteRef |
                    SetPortion | SetPosition | SetSliderValue | SetSpeed |
                    SetTimer | SetTransparency | SetVariable | SetVolume |
                    Spawn | Step | Stop | StorePersistent | Subtract |
                    TestVariable | Toggle | ToggleItem | TransitionTo |
                    Unload | UnlockScreen
```

NOTA – La semántica y la sintaxis de las acciones elementales se proporcionan anteriormente en esta Recomendación.

50 Objetos de referencia, Contenido, Valores, Color y Posición XY

50.1 Referencia de objeto

Descripción Este tipo de datos se utiliza para hacer referencia a objetos. El objeto referenciado será visible al objeto desde el cual se hace la referencia. Esto significa que el objeto será un objeto Aplicación o Escena, o será parte de la Aplicación activa o de la Escena activa.

La referencia consiste en un Identificador de grupo y un Número de objeto, ambos facultativos. El valor por defecto del Identificador de grupo es el Identificador de grupo de la Escena o el objeto Aplicación desde el cual se hizo la referencia.

ObjectReference	-->	GroupIdentifier?, ObjectNumber
GroupIdentifier		OctetString
ObjectNumber	-->	INTEGER

50.2 Referencia de contenido

Descripción Este tipo de datos se utiliza para hacer referencia a fuentes de datos externas. La Referencia de contenido consiste en una Cadena de octetos.

ContentReference	-->	OctetString
------------------	-----	-------------

50.3 Referencia de objeto genérica

Descripción Tipo de datos que permite la referencia directa a un objeto o la referencia indirecta mediante un objeto Variable.

En el caso de referencia directa, esta referencia determina una Referencia de objeto directamente al objeto objetivo.

En el caso de referencia indirecta esta referencia determina una Referencia de objeto a un objeto Variable referencia de objeto. El Objeto variable referencia de objeto contendrá entonces una Referencia de objeto al objeto objetivo o NULO.

GenericObjectReference	-->	DirectReference IndirectReference
DirectReference	-->	ObjectReference
IndirectReference	-->	ObjectReference

50.4 Referencia de contenido genérica

Descripción Tipo de datos que permite la referencia directa a un objeto o la referencia indirecta mediante un objeto Variable.

En el caso de referencia directa, esta referencia determina una Referencia de contenido directamente.

En el caso de referencia indirecta, esta referencia de contenido determina una Referencia de objeto a un objeto Variable referencia de contenido. El objeto

Variable referencia de contenido contendrá entonces una referencia al contenido o NULO.

GenericContentReference	-->	DirectContentReference IndirectReference
DirectContentReference	-->	ContentReference
IndirectReference	-->	ObjectReference

50.5 Entero genérico

Descripción Tipo de datos que permite la inclusión directa de un Entero o la referencia a un objeto Variable entera.

GenericInteger	-->	Value IndirectReference
Value	-->	INTEGER
IndirectReference	-->	ObjectReference

50.6 Booleano genérico

Descripción Tipo de datos que permite la inclusión directa de un Booleano o la referencia a un objeto Variable booleana.

GenericBoolean	-->	Value IndirectReference
Value	-->	BOOLEAN
IndirectReference	-->	ObjectReference

50.7 Cadena de octetos genérica

Descripción Tipo de datos que permite la inclusión directa de una Cadena de octetos o la referencia a un objeto Variable cadena de octetos.

GenericOctetString	-->	Value IndirectReference
Value	-->	OctetString
IndirectReference	-->	ObjectReference

50.8 Color

Descripción Tipo de datos utilizado para especificar un color por un nombre (una Cadena de octetos) o por un índice (un Entero que hace referencia a un objeto paleta).

Colour	-->	ColourIndex AbsoluteColour
ColourIndex	-->	INTEGER
AbsoluteColour	-->	OctetString

50.9 Posición XY

Descripción Tipo de datos para especificar la posición (X,Y) en un sistema de coordenadas de Escena.

XYPosition	-->	XPosition, YPosition
XPosition	-->	INTEGER
YPosition	-->	INTEGER

50.10 Resolución de valores genéricos

Los valores genéricos (Referencia de contenido genérica, Referencia de objeto genérica, Entero genérico, Booleano genérico y Cadena de octetos genérica) se utilizan solamente como parámetros de acciones elementales. La resolución se efectúa cuando la acción es invocada. Por ejemplo, considérese una Variable entera, V, que inicialmente está puesta a 10. Si se invocan las siguientes acciones:

- 1) fijar V a 15,
- 2) fijar un *Valor* de Deslizador a V,
- 3) fijar V a 20,

el *Valor* de Deslizador se pondrá a 15. En otras palabras, el valor del Deslizador es fijado por valor, no por referencia.

51 Objetos MHEG-5 de referencia

Las referencias a objetos MHEG-5 son representadas por Referencias de objeto. En cualquier momento, una Referencia de objeto será determinada teniendo en cuenta el objeto Grupo MHEG-5 que contiene esa referencia (una Escena o una Aplicación) y el contenido de los atributos Identificador de grupo y Número de objeto de la Referencia de objeto.

A continuación se presenta cómo la Referencia de objeto será codificada de acuerdo con el Grupo de origen y la naturaleza del objeto MHEG-5 referenciado:

- i) Dentro de un objeto Escena
 - a) Referencia a un Ingrediente de la Escena activa:
 - 1) El Identificador de grupo puede estar codificado o no.
 - 2) El Número de objeto contendrá el número de objeto del Ingrediente dentro de la Escena.
 - b) Referencia a un Ingrediente compartido de la Aplicación activa:
 - 1) El Identificador de grupo contendrá el identificador de grupo del objeto Aplicación activo.
 - 2) El Número de objeto contendrá el número de objeto del Ingrediente dentro del objeto Aplicación.
 - c) Referencia a la propia Escena activa:
 - 1) El Identificador de grupo no tiene que estar codificado.
 - 2) El Número de objeto se pondrá a 0.
 - d) Referencia a otra Escena:
 - 1) El Identificador de grupo contendrá el identificador de grupo del objeto Escena referenciado.
 - 2) El Número de objeto se pondrá a 0.
 - e) Referencia a la Aplicación activa:
 - 1) El Identificador de grupo contendrá el identificador de grupo del objeto Aplicación activo.
 - 2) El Número de objeto se pondrá a 0.
 - f) Referencia a otro objeto Aplicación:
 - 1) El Identificador de grupo contendrá el identificador de grupo del objeto Aplicación referenciado.

- 2) El Número de objeto se pondrá a 0.
- ii) Dentro del objeto Aplicación
 - a) Referencia a un Ingrediente del grupo Aplicación:
 - 1) El Identificador de grupo puede estar codificado o no.
 - 2) El Número de objeto contendrá el número de objeto único del Ingrediente dentro de la Aplicación.
 - b) Referencia a una Escena:
 - 1) El Identificador de grupo contendrá el identificador de grupo del objeto Escena referenciado.
 - 2) El Número de objeto se pondrá a 0.
 - c) Referencia a la propia Aplicación:
 - 1) El Identificador de grupo no tiene que estar codificado.
 - 2) El Número de objeto se pondrá a 0.
 - d) Referencia a otro objeto Aplicación:
 - 1) El Identificador de grupo contendrá el identificador de grupo del objeto Aplicación referenciado.
 - 2) El Número de objeto se pondrá a 0.

La presente Recomendación no define la codificación real de la Cadena de octetos de Identificador de grupo. Cada dominio de aplicación MHEG-5 definirá formas específicas del Identificador de grupo.

52 Espacios de nombres, Llamadas a Programas Distantes y Conexiones

El motor MHEG-5 tiene un espacio de nombre por defecto, que es el espacio de nombre del objeto Aplicación activo. Todos los demás objetos Aplicación que pueden ser alcanzados (por Lanzamiento o Ramificación permanente) estarán también en ese espacio de nombre. Este espacio de nombre, en un sentido amplio, debe incluir también información suficiente para efectuar llamadas distantes a métodos denominados a través de objetos programa MHEG-5.

NOTA – En apéndice I figura una explicación sobre cómo el motor MHEG-5 se vincula a este espacio de nombre.

Sin embargo, es también posible que una aplicación MHEG-5 se conecte con otro espacio de nombre temporalmente, utilizando la acción Abrir conexión. Ese nuevo espacio de nombre puede ser utilizado para llamadas a métodos denominados (a través de objetos Programa MHEG-5) y para acceder a un objeto Escena MHEG-5 que está situado en otro espacio de nombre que su objeto Aplicación. La siguiente regla se aplica a las referencias a y de este objeto Escena:

- 1) La referencia a un objeto Escena en otro espacio de nombre que el objeto Aplicación se codificará dentro de una acción Transición a, cuyo parámetro Rótulo de conexión indica la conexión con la entidad que administra ese otro espacio de nombre.
- 2) Todas las Referencias de contenido de ese objeto Escena serán interpretadas en el «otro» espacio de nombre.
- 3) Todas las referencias de Identificador de grupo de ese objeto Escena serán interpretadas en el espacio de nombre por defecto, salvo en el caso indicado en la regla 1) anterior.

Recordatorio: Corresponde al dominio de aplicación decidir si el motor MHEG-5 efectúa o no las acciones Abrir conexión y Cerrar conexión.

53 Tratamiento de eventos

53.1 Tipos de eventos

Esta Recomendación define los siguientes eventos: *Disponible, Contenido disponible, Suprimido, en funcionamiento, Detenido, Evento de prueba, Entrada de usuario, Temporizador activado, Detenido asíncrono, Interacción completada, Testigo movido desde, Testigo movido a, Primer ítem presentado, Último ítem presentado, Ítems de cabecera, Ítems de cola, Ítem seleccionado, Ítem deseleccionado, Evento de tren, Tren en ejecución, Tren detenido, Activador de contador, Realce activado, Realce desactivado, Entrada de cursor, Salida de cursor, Ancla activada, Seleccionado, Deseleccionado, Campo de entrada de lleno, Evento de motor (IsAvailable, ContentAvailable, IsDeleted, IsRunning, IsStopped, TestEvent, UserInput, TimerFired, AsynchStopped, InteractionCompleted, TokenMovedFrom, TokenMovedTo, FirstItemPresented, LastItemPresented, HeadItems, TailItems, ItemSelected, ItemDeselected, StreamEvent, StreamPlaying, StreamStopped, CounterTrigger, HighlightOn, HighlightOff, CursorEnter, CursorLeave, AnchorFired, IsSelected, IsDeselected, EntryFieldFull, EngineEvent).*

Un evento siempre emana de un objeto específico, denominado la fuente de evento. La semántica de cada clase MHEG-5 define las circunstancias en las cuales un objeto de esa clase genera un evento específico.

Algunos de los tipos de eventos mencionados anteriormente tienen asociado un valor de datos. Ese valor se utiliza para determinar si el Enlace asociado debe disparar, como se describe a continuación. La siguiente tabla enumera los valores de datos asociados con cada tipo de evento:

Tipo de evento	Datos asociados	Tipo de valor asociado
AnchorFired	AnchorTag	OctetString
CounterTrigger	Identifier	INTEGER
EngineEvent	EventTag	INTEGER
FirstItemPresented	Index	BOOLEAN
HeadItems	Number of items	INTEGER
ItemDeselected	Index	INTEGER
ItemSelected	Index	INTEGER
LastItemPresented	Index	BOOLEAN
StreamEvent	StreamEventTag	OctetString
TailItems	Number of items	INTEGER
TestEvent	TestResult	BOOLEAN
TimerFired	TimerIdentifier	INTEGER
TokenMovedFrom	Index	INTEGER
TokenMovedTo	Index	INTEGER
UserInput	UserInputEventTag	INTEGER
Todos los otros	Ninguno	N/A

Los eventos cuyos tipos tienen datos asociados deben ser generados siempre con esos datos. Por ejemplo, cuando se genera un evento *Temporizador activado (TimerFired)*, debe estar acompañado por el *Identificador de temporizador (TimerIdentifier)*.

53.2 Eventos síncronos y eventos asíncronos

Los eventos pueden suceder por dos razones.

- 1) Un proceso que es asíncrono al motor MHEG-5 produce un evento.

El evento resultante se denomina asíncrono. Son eventos asíncronos todos los eventos de los tipos *Ancla activada*, *Detenido asíncrono*, *Contenido disponible*, *Activador de contador*, *Entrada de cursor*, *Salida de cursor*, *Evento de motor*, *Campo de entrada lleno*, *Interacción completada*, *Evento de tren*, *Tren en ejecución*, *Tren detenido*, *Temporizador activado* y *Entrada de usuario* (*AnchorFired*, *AsynchStopped*, *ContentAvailable*, *CounterTrigger*, *CursorEnter*, *CursorLeave.*, *EngineEvent*, *EntryFieldFull*, *InteractionCompleted*, *StreamEvent*, *StreamPlaying*, *StreamStopped*, *TimerFired* and *UserInput*).

- 2) El evento es el resultado directo de la ejecución de una acción elemental.

El evento resultante se denomina síncrono. Son eventos síncronos todos los eventos de los tipos *Primer ítem presentado*, *Ítems de cabecera*, *Realce desactivado*, *Realce activado*, *Disponible*, *Suprimido*, *Deseleccionado*, *En funcionamiento*, *Seleccionado*, *Detenido*, *Ítem deseleccionado*, *Ítem seleccionado*, *Último ítem presentado*, *Ítems de cola*, *Evento de prueba*, *Testigo movido desde* y *Testigo movido a* (*FirstItemPresented*, *HeadItems*, *HighlightOff*, *HighlightOn*, *IsAvailable*, *IsDeleted*, *IsDeselected*, *IsRunning*, *IsSelected*, *IsStopped*, *ItemDeselected*, *ItemSelected*, *LastItemPresented*, *TailItems*, *TestEvent*, *TokenMovedFrom* and *TokenMovedTo*). La semántica de las clases de la presente Recomendación indica explícitamente cuándo se genera cada evento.

53.3 Tratamiento de eventos y enlaces

Cada Enlace MHEG-5 tiene una Condición de enlace y un Efecto de enlace. Cuando el motor MHEG-5 examina un evento que se ha producido, comprobará todos los Enlaces activos (de los objetos Aplicación y Escena activos) para ver si sus atributos Tipo de evento y Fuente de evento concuerdan con el tipo y fuente del evento en cuestión. Para cada uno de los Enlaces que cumplen esta condición, los datos asociados del evento se comprueban contra el atributo facultativo Datos de evento del Enlace. Se dice también que los Enlaces que cumplen esta condición (o que la cumplen por defecto porque no tienen Datos de evento) están *activados* (*fired*).

El motor MHEG-5 es activado por la ocurrencia de eventos asíncronos. Cuando se produce un evento asíncrono, el motor MHEG-5 examinará todos los Enlaces activos de los objetos Aplicación y Escena activos para determinar si han disparado. Para cada uno de los Enlaces activados, las acciones elementales de su Efecto de enlace serán almacenadas en una cola para ejecución secuencial. Esta Recomendación *no* especifica el orden en el que se han de tratar dos Enlaces que disparan en relación con el mismo evento.

El tratamiento de errores por defecto es el siguiente: si una de las acciones elementales mencionadas produce un error, se pasará por alto dicha acción. Sin embargo, una aplicación puede utilizar Eventos de motor para indicar la situación de error a la aplicación MHEG-5 (por ejemplo, para enviar un mensaje de error).

Como resultado directo de un Efecto de enlace en ejecución, se pueden producir eventos síncronos. Estos eventos serán tratados directamente por el motor MHEG-5. En otras palabras, después de la ejecución de cada acción elemental, el motor MHEG-5 verificará si otros Enlaces adicionales han sido activados como resultado de un evento síncrono. Si éste es el caso, ese Enlace y todos sus efectos serán procesados completamente antes de que el motor MHEG-5 continúe procesando la siguiente acción elemental del Enlace original.

Cualesquiera eventos asíncronos (como el evento Entrada de usuario) que se produzcan mientras se están procesando los eventos asíncronos originales no son tratados hasta que se completa todo el proceso anterior. Los eventos asíncronos que no son tratados, son puestos en cola.

Las acciones que cambian el contexto del procesamiento de la acción vigente influirán en la cola de los eventos asíncronos y en la cola de las acciones que esperan procesamiento. El contexto es cambiado por las acciones "Transición a", "Lanzar", "Ramificación permanente" y "Cesar". Si se produce una de estas acciones, los eventos asíncronos pendientes originados después de la conmutación del contexto serán suprimidos de la cola de eventos asíncronos. Las acciones elementales que esperan ejecución serán suprimidas también de la cola de acciones.

En este contexto, se debe señalar que algunas acciones MHEG-5, tales como Ejecutar, tiene un efecto que continúa después de la compleción de la propia acción. Por ejemplo, cuando se ejecuta un objeto Diagrama de bits, la acción Ejecutar retorna tan pronto como se ha visualizado el Diagrama de bits, permitiendo así al procesador de Enlaces continuar su trabajo. El efecto de esa acción (es decir, el diagrama de bits que está en la pantalla) continúa aún después de la compleción de la acción.

Otro aspecto importante es que es posible que un Enlace se desactive en su Efecto de enlace. Esta acción será pospuesta hasta que se haya ejecutado completamente el Efecto de enlace.

53.4 Entrada de usuario

La presente Recomendación no especifica los dispositivos de entrada de usuario ni los tipos de eventos generados. La entrada de usuario «bruta» recibida por el motor MHEG-5 (por ejemplo, en forma de teleinstrucciones de control) será traducida por el motor, cuando proceda, en ocurrencias de eventos *Entrada de usuario*. Todos estos eventos tienen un rótulo, que especifica el evento *Entrada de usuario* que se ha producido. El rótulo es un entero y la semántica del rótulo es definida por un Registro de eventos de entrada vinculado al objeto Escena.

Cuando el motor MHEG-5 ha traducido la entrada de usuario bruta en uno o más eventos *Entrada de usuario*, esos eventos son tratados por el motor como eventos asíncronos, según se describe anteriormente.

53.5 Interacción de usuario

Los objetos MHEG-5 que pertenecen a la clase Interactuable pueden estar en un determinado estado, denominado «interactuación» («interacting»), que es señalado por el atributo *Estado de interacción* del objeto puesto a Verdadero. Cuando un objeto está en este estado, ningún evento *Entrada de usuario* será generado por el motor MHEG-5. El motivo de esta regla es que el motor MHEG-5 pudiera tener que utilizar la entrada de usuario «bruta» para interactuar con la clase Interactuable y, por tanto, puede no ser capaz de generar los eventos Entrada de usuario normales.

En todo momento, como máximo, sólo un objeto estará interactuando.

No obstante, todos los otros eventos serán generados. Esto permite, por ejemplo, aplicar temporizaciones.

53.6 Eventos de cursor

Es necesario precisar lo siguiente con respecto a los eventos *Salida de cursor* y *Entrada de cursor* definidos anteriormente. Puede suceder que un cursor se mueva inmediatamente de una zona definida por un objeto Interactuable activo a otra zona definida por otro interactuable activo; puede ser así, por ejemplo, cuando una zona está en la parte superior de la otra, es decir cuando una zona está por encima de la otra en la Pila de visualización y se superponen. En este caso, el evento *Salida de cursor* será generado por el primer objeto interactuable (que está más alto en la Pila de

visualización), seguido por un evento *Entrada de cursor* generado por el segundo Interactuable (que está más bajo en la Pila de visualización).

Es preciso generar también eventos de cursor si un objeto Interactuable activo se mueve por debajo de la posición del cursor, o si un Interactuable por debajo de la posición del cursor pasa a estar activo, o si cambia el orden de la pila de interactivables por debajo de la posición del cursor, de manera que también cambia el Interactuable en la parte más alta por debajo de la posición del cursor.

En general sólo un Interactuable activo puede tener el cursor en su casilla delimitante a la vez. Esto tiene que ser reflejado por las secuencias apropiadas de los eventos Salida de cursor y Entrada de cursor.

Por último, se debe recordar que un objeto Interactuable inactivo no generará eventos de cursor.

53.7 Tratamiento de errores

En general, el mecanismo de errores por defecto del motor MHEG-5 consiste en pasar por alto la causa del error y continuar al paso siguiente. Por ejemplo, si un objeto intercambiado es un caso de una clase que el motor MHEG-5 no reconoce, el objeto será pasado por alto. Cualquier acción subsiguiente que pueda producirse en relación con este objeto y que produce un error, será pasada por alto. De manera más general, el resultado de cualquier causa de error (por ejemplo en una acción elemental) es que se pasará por alto esa causa (por ejemplo, la acción elemental).

Sin embargo, como se menciona anteriormente, un dominio de aplicación puede utilizar Eventos de motor para indicar la situación de error a la aplicación MHEG-5 (por ejemplo, para enviar un mensaje de error).

54 Presentación de objetos visibles

54.1 Sistema de coordenadas

Esta subcláusula describe la semántica asociada con la visualización de objetos de clases que heredan de la clase Visible. El motor MHEG-5 tendrá acceso exactamente a una visualización, que se denomina la «pantalla» en el resto de esta subcláusula. La pantalla es un sistema de coordenadas ortogonales, con un eje x de izquierda a derecha y un eje y de arriba a abajo. El origen está en la esquina superior izquierda. De acuerdo con los atributos Sistemas de coordenadas de escena y Formato de imagen del objeto Escena activo, el sistema de coordenadas puede tener diferentes tamaños y formatos de imagen.

54.2 Casilla delimitante

Cada objeto visible tiene una casilla delimitante, especificada por los atributos internos Posición y Tamaño de casilla. Estos atributos se expresan en el espacio de coordenadas de la Escena. El contenido del objeto Visible será anclado en la esquina superior izquierda de la casilla delimitante. Además, el contenido del objeto Visible será recortado de modo, que no se reproduzcan las partes que quedan fuera de la casilla delimitante. El recorte se efectuará como sigue:

- a) Para Texto y sus subclases, el recorte se efectúa de manera que se respete la colocación del texto dentro de la casilla delimitante. Sólo se explica el caso donde Orientación de línea es *horizontal*; el caso *vertical* se obtiene análogamente.
 - Justificación horizontal:
Si el objeto Texto tiene el atributo Retorno a comienzo de línea de texto puesto a Verdadero, su contenido se divide en tantas líneas como sea necesario para que entre en la casilla delimitante. Además, si el atributo Justificación horizontal del objeto Texto

está puesto a *justificado*, las líneas se visualizan de modo que su comienzo esté alineado con el borde izquierdo de la casilla delimitante y su fin esté alineado con el borde derecho de la casilla delimitante.

Si el atributo Retorno a comienzo de línea de texto se pone a Falso, se aplican las siguientes reglas para la justificación horizontal:

- Si la Justificación horizontal se pone a *comienzo*, las líneas son recortadas de modo que el comienzo de cada línea esté alineado con el borde izquierdo de la casilla delimitante.
- Si la Justificación horizontal se pone a *fin*, las líneas son recortadas de modo que el fin de cada línea esté alineado con el borde derecho de la casilla delimitante.
- Si la Justificación horizontal se pone a *centro*, las líneas son recortadas en ambos lados, de modo que el centro de cada línea esté alineado con una línea vertical virtual en el centro de la casilla delimitante.

- Justificación vertical:

Para un Texto que tiene el atributo Justificación vertical puesto a *fin*, su contenido se recorta de modo que el fin del texto esté alineado con la parte inferior de la casilla delimitante; de este modo, el comienzo del texto es ocultado. A la inversa, si Justificación vertical se pone a *comienzo* el texto es recortado de modo que su fin es ocultado. Por último, si Justificación vertical se pone a *centro*, el texto es recortado de modo que el centro del texto esté alineado con el centro de la casilla delimitante, lo que significa que el comienzo y el fin son ocultados.

- Color de fondo:

Si el color de fondo del objeto texto no es transparente, toda su casilla delimitante será rellenada con ese color, aunque el texto no llene la casilla delimitante.

- b) Para Deslizador, Rectángulo y las subclases de Botón, el objeto no es recortado, sino que se le aplica una escala para que encaje exactamente dentro de su casilla delimitante.
- c) Otros objetos Visibles son recortados de modo que su esquina superior izquierda esté colocada en la esquina superior izquierda de la casilla delimitante. Para estos objetos Visibles, alguna parte de la casilla delimitante puede estar vacía, y esa parte será transparente.

54.3 Pila de visualización

El motor MHEG-5 implementará una pila de visualización. Cada objeto Visible es reproducido inicialmente de acuerdo con su posición en el atributo *Pila de visualización* del objeto Aplicación vigente. Los objetos visibles pueden ser movidos hacia arriba y hacia abajo en la pila de visualización utilizando las acciones Traer hacia adelante, Enviar hacia atrás, Poner delante y Poner detrás. Cuando un objeto Visible es movido de esta manera, (partes de) otros objetos Visibles pueden ser ocultados o descubiertos. En este caso, la visualización será actualizada en consecuencia.

54.4 Objetos transparentes

La transparencia de los objetos se especifica con un entero en la gama [0, 100] (porcentaje). La interpretación del entero entre 1 y 99 depende de la implementación, y esta Recomendación no la define. Asimismo, esta Recomendación tampoco especifica la exactitud de la reproducción y visualización reales, ni los algoritmos utilizados para realizar estas tareas. Sin embargo, se especifican valores de transparencias de 0% y 100%, que se definen como sigue.

Los objetos de transparencia 0% serán reproducidos como objetos no transparentes, es decir, los colores originales serán reproducidos sin ninguna modificación. Los objetos de transparencia 100% serán reproducidos como objetos completamente transparentes, es decir, los colores originales no serán reproducidos, y se percibirá el color de fondo.

- i) Para cada posición en el espacio de coordenadas de la Escena, como máximo se consideran los N objetos cuyas casillas delimitante contienen esa posición. Estos objetos están ordenados en la denominada pila de visualización. Consideremos los objetos O_1, O_2, \dots, O_N , donde O_N denota el objeto en la posición más alta.
- ii) El cálculo del valor de pixel final se efectúa a partir de la parte inferior de la pila. El valor de pixel después de tener en cuenta los j objetos se denomina V_j . Por consiguiente, el valor de pixel final se denomina V_N .
- iii) El cálculo del valor de pixel comienza a partir de un fondo completamente negro, a saber, $V_0 = \text{<negro>}$.
- iv) Para un j objeto que toma sucesivamente los valores 1 a N , se hace lo siguiente:
 - a) Si el valor de pixel O_j que se ha de considerar se denomina P_j y es translúcido p porcentaje:

$$V_j = V_{j-1} \frac{P}{100} + P_j \left(1 - \frac{P}{100} \right).$$

Esta fórmula se puede aplicar a todos los componentes de color de un pixel si el espacio de color no se basa en las señales de luminancia y de diferencia de color (por ejemplo RGB). Sin embargo, esta fórmula se debe aplicar solamente a la luminancia de un pixel si el espacio de color se basa en las señales de luminancia y de diferencia de color, como YUV. Además, se supone que los valores de esta fórmula se expresan en el mismo espacio de color lineal. Si se utiliza un espacio de color no lineal, se puede tener en cuenta esta fórmula para la reproducción precisa.

- b) Específicamente, si el pixel es completamente transparente [invisible]:

$$V_j = V_{j-1}.$$

- c) Específicamente si el pixel es completamente opaco:

$$V_j = P_j.$$

No se interpretará que este modelo impone ningún proceso específico de reproducción y/o visualización en una implementación concreta de un motor MHEG-5. Su única finalidad es definir inequívocamente la semántica de los objetos gráficos transparentes diferentes combinados. La presente Recomendación no define la exactitud de la reproducción y visualización reales, ni los algoritmos utilizados para realizar estas tareas.

54.5 Formato de pixel

Se supone en general que el formato de pixel de una Escena y su contenido son iguales; un pixel del contenido de la escena corresponde exactamente con un pixel de la Escena, y no es necesario aplicar ninguna escala. Sin embargo, si los dos formatos de imagen son diferentes, el motor MHEG-5 puede aplicar la escala al contenido, teniendo en cuenta los dos formatos de imagen, con el fin de compensar la presentación del contenido. Obsérvese que la aplicación de esta escala depende de la implementación del motor MHEG-5 y es una característica facultativa, por lo que la presente Recomendación no define ninguna manera específica de hacerlo.

ANEXO A

Notación ASN.1

Este anexo describe la notación ASN.1 para la sintaxis de objetos MHEG-5 conformes a ISO/CEI 13522-5.

Para la codificación de los objetos MHEG-5 de acuerdo con la sintaxis ASN.1 se utilizarán las *reglas de codificación distinguida (DER, distinguished encoding rules)* definidas en la Rec. UIT-T X.690 | ISO/CEI 8825-1. La codificación alternativa es la notación textual definida en el anexo B.

La sintaxis que se utilizará en DER se detalla a continuación.

```
-- $PREFIX=ISOMHEG-mheg-5:mheg-5
-- Module: mheg-5
```

ISO13522-MHEG-5 {joint-iso-itu-t(2) mheg(19) version(1) mheg-5(17)}

DEFINITIONS IMPLICIT TAGS ::=
BEGIN

```
-- This module defines the MHEG-5 abstract syntax which consists of data values of type:
-- ISO13522-MHEG-5.InterchangedObject.
-- This abstract syntax is identified by the name: {joint-iso-itu-t(2) mheg(19) version(1) mheg-5(17)}.
```

InterchangedObject ::= CHOICE

```
{
  application [0] ApplicationClass,
  scene [1] SceneClass
}
```

```
-- A.1 Root Class _____
```

RootClass ::= ObjectReference

```
-- A.2 Group Class _____
```

GroupClass ::= SET

```
{
  RootClass (WITH COMPONENTS
    {external-reference (WITH COMPONENTS {..., object-number (0)}) PRESENT,
     internal-reference ABSENT}),
  standard-identifier [2] StandardIdentifier OPTIONAL,
  standard-version [3] INTEGER (1) OPTIONAL,
  object-information [4] OCTET STRING OPTIONAL,
  on-start-up [5] ActionClass OPTIONAL,
  on-close-down [6] ActionClass OPTIONAL,
  original-group-cache-priority [7] INTEGER (0..255) DEFAULT 127,
  items [8] SEQUENCE SIZE (1..MAX) OF GroupItem OPTIONAL
}
```

StandardIdentifier ::= SEQUENCE

```
{
  joint-iso-itu-t INTEGER (2),
  mheg INTEGER (19)
}
```

GroupItem ::= CHOICE

```
{
  resident-program [9] ResidentProgramClass,
  remote-program [10] RemoteProgramClass,
  interchanged-program [11] InterchangedProgramClass,
  palette [12] PaletteClass,
  font [13] FontClass,
  cursor-shape [14] CursorShapeClass,
  boolean-variable [15] BooleanVariableClass,
  integer-variable [16] IntegerVariableClass,
  octet-string-variable [17] OctetStringVariableClass,
  object-ref-variable [18] ObjectRefVariableClass,
  content-ref-variable [19] ContentRefVariableClass,
  link [20] LinkClass,
  stream [21] StreamClass,
  bitmap [22] BitmapClass,
  line-art [23] LineArtClass,
  dynamic-line-art [24] DynamicLineArtClass,
  rectangle [25] RectangleClass,
  hotspot [26] HotspotClass,
  switch-button [27] SwitchButtonClass,
  push-button [28] PushButtonClass,
  text [29] TextClass,
  entry-field [30] EntryFieldClass,
  hyper-text [31] HyperTextClass,
  slider [32] SliderClass,
  token-group [33] TokenGroupClass,
  list-group [34] ListGroupClass
}
```

-- A.3 Application Class _____

ApplicationClass ::= SET

```
{
  COMPONENTS OF GroupClass,
  on-spawn-close-down [35] ActionClass OPTIONAL,
  on-restart [36] ActionClass OPTIONAL,
  default-attributes [37] SEQUENCE SIZE (1..MAX) OF DefaultAttribute OPTIONAL
}
```

DefaultAttribute ::= CHOICE

```
{
  character-set [38] INTEGER,
  background-colour [39] Colour,
  text-content-hook [40] INTEGER,
  text-colour [41] Colour,
  font [42] FontBody,
  font-attributes [43] OCTET STRING,
  interchanged-program-content-hook [44] INTEGER,
  stream-content-hook [45] INTEGER,
  bitmap-content-hook [46] INTEGER,
  line-art-content-hook [47] INTEGER,
  button-ref-colour [48] Colour,
  highlight-ref-colour [49] Colour,
  slider-ref-colour [50] Colour
}
```



```

FontBody ::= CHOICE
{
    direct-font OCTET STRING,
    indirect-font ObjectReference
}

```

-- A.4 Scene Class _____

```

SceneClass ::= SET
{
    COMPONENTS OF GroupClass,
    input-event-register [51] INTEGER,
    scene-coordinate-system [52] SceneCoordinateSystem,
    aspect-ratio [53] AspectRatio DEFAULT {width 4, height 3},
    moving-cursor [54] BOOLEAN DEFAULT FALSE,
    next-scenes [55] SEQUENCE SIZE (1..MAX) OF NextScene OPTIONAL
}

```

```

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

```

```

AspectRatio ::= SEQUENCE
{
    width INTEGER,
    height INTEGER
}

```

```

NextScene ::= SEQUENCE
{
    scene-ref OCTET STRING,
    scene-weight INTEGER (0..255)
}

```

-- A.5 Ingredient Class _____

```

IngredientClass ::= SET
{
    RootClass (WITH COMPONENTS
        {..., external-reference (WITH COMPONENTS {..., object-number (1..MAX)}})),
    initially-active [56] BOOLEAN DEFAULT TRUE,
    content-hook [57] INTEGER OPTIONAL,
    original-content [58] ContentBody OPTIONAL,
    shared [59] BOOLEAN DEFAULT FALSE
}

```

```

ContentBody ::= CHOICE
{
    included-content OCTET STRING,
    referenced-content ReferencedContent
}

```

```

ReferencedContent ::= SEQUENCE
{
    content-reference ContentReference,
    content-size [60] INTEGER OPTIONAL,
    content-cache-priority [61] INTEGER (0..255) DEFAULT 127
}

```

```
LinkClass ::= SET
{
  COMPONENTS OF IngredientClass
  (WITH COMPONENTS {..., content-hook ABSENT, original-content ABSENT}),
  link-condition [62] LinkCondition,
  link-effect [63] ActionClass
}
```

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

```
EventType ::= ENUMERATED
{
  is-available(1),
  content-available(2),
  is-deleted(3),
  is-running(4),
  is-stopped(5),
  user-input(6),
  anchor-fired(7),
  timer-fired(8),
  asynch-stopped(9),
  interaction-completed(10),
  token-moved-from(11),
  token-moved-to(12),
  stream-event(13),
  stream-playing(14),
  stream-stopped(15),
  counter-trigger(16),
  highlight-on(17),
  highlight-off(18),
  cursor-enter(19),
  cursor-leave(20),
  is-selected(21),
  is-deselected(22),
  test-event(23),
  first-item-presented(24),
  last-item-presented(25),
  head-items(26),
  tail-items(27),
  item-selected(28),
  item-deselected(29),
  entry-field-full(30),
  engine-event(31)
}
```

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

-- A.7 Program Class _____

```
ProgramClass ::= SET
{
  COMPONENTS OF IngredientClass
  (WITH COMPONENTS {..., initially-active (FALSE) PRESENT}),
  name [64] OCTET STRING,
  initially-available [65] BOOLEAN DEFAULT TRUE
}
```

-- A.8 Resident Program Class _____

```
ResidentProgramClass ::= ProgramClass
(WITH COMPONENTS {..., content-hook ABSENT, original-content ABSENT})
```

-- A.9 Remote Program Class _____

```
RemoteProgramClass ::= SET
{
  COMPONENTS OF ProgramClass
  (WITH COMPONENTS {..., content-hook ABSENT, original-content ABSENT}),
  program-connection-tag [66] INTEGER OPTIONAL
}
```

-- A.10 Interchanged Program Class _____

```
InterchangedProgramClass ::= ProgramClass
(WITH COMPONENTS {..., original-content PRESENT})
```

-- A.11 Palette Class _____

```
PaletteClass ::= IngredientClass
(WITH COMPONENTS
  {..., content-hook PRESENT, original-content PRESENT, initially-active (TRUE)})
```

-- A.12 Font Class _____

```
FontClass ::= IngredientClass
(WITH COMPONENTS
  {..., content-hook PRESENT, original-content PRESENT, initially-active (TRUE)})
```

-- A.13 Cursor Shape _____

```
CursorShapeClass ::= IngredientClass
(WITH COMPONENTS
  {..., content-hook PRESENT, original-content PRESENT, initially-active (TRUE)})
```

-- A.14 Variable Class _____

```
VariableClass ::= SET
{
  COMPONENTS OF IngredientClass
  (WITH COMPONENTS {..., content-hook ABSENT, original-content ABSENT, initially-active (TRUE)}),
  original-value [67] OriginalValue
}
```

```
OriginalValue ::= CHOICE
{
  boolean BOOLEAN,
  integer INTEGER,
  octetstring OCTET STRING,
```

```
object-reference [68] ObjectReference,  
content-reference [69] ContentReference  
}
```

-- A.15 Boolean Variable Class _____

```
BooleanVariableClass ::= VariableClass  
(WITH COMPONENTS {..., original-value (WITH COMPONENTS {..., boolean PRESENT})))
```

-- A.16 Integer Variable Class _____

```
IntegerVariableClass ::= VariableClass  
(WITH COMPONENTS {..., original-value (WITH COMPONENTS {..., integer PRESENT})))
```

-- A.17 Octet String Variable Class _____

```
OctetStringVariableClass ::= VariableClass  
(WITH COMPONENTS {..., original-value (WITH COMPONENTS {..., octetstring PRESENT})))
```

-- A.18 Object Reference Variable Class _____

```
ObjectRefVariableClass ::= VariableClass  
(WITH COMPONENTS {..., original-value (WITH COMPONENTS {..., object-reference PRESENT})))
```

-- A.19 Content Reference Variable Class _____

```
ContentRefVariableClass ::= VariableClass  
(WITH COMPONENTS {..., original-value (WITH COMPONENTS {..., content-reference PRESENT})))
```

-- A.20 Presentable Class _____

```
PresentableClass ::= IngredientClass
```

-- A.21 Token Manager Class _____

```
TokenManagerClass ::= SET  
{  
  movement-table [70] SEQUENCE SIZE (1..MAX) OF Movement OPTIONAL  
}
```

```
Movement ::= SEQUENCE SIZE (1..MAX) OF INTEGER
```

-- A.22 Token Group Class _____

```
TokenGroupClass ::= SET  
{  
  COMPONENTS OF PresentableClass  
    (WITH COMPONENTS {..., content-hook ABSENT, original-content ABSENT}),  
  COMPONENTS OF TokenManagerClass,  
  token-group-items [71] SEQUENCE SIZE (1..MAX) OF TokenGroupItem,  
  no-token-action-slots [72] SEQUENCE SIZE (1..MAX) OF ActionSlot OPTIONAL  
}
```

```
TokenGroupItem ::= SEQUENCE  
{  
  a-visible ObjectReference,  
  action-slots SEQUENCE SIZE (1..MAX) OF ActionSlot OPTIONAL  
}
```

```

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

```

-- A.23 List Group Class _____

```

ListGroupClass ::= SET
{
  COMPONENTS OF TokenGroupClass,
  positions [73] SEQUENCE SIZE (1..MAX) OF XYPosition,
  wrap-around [74] BOOLEAN DEFAULT FALSE,
  multiple-selection [75] BOOLEAN DEFAULT FALSE
}

```

-- A.24 Visible Class _____

```

VisibleClass ::= SET
{
  COMPONENTS OF PresentableClass,
  original-box-size [76] OriginalBoxSize,
  original-position [77] XYPosition DEFAULT {x-position 0, y-position 0},
  original-palette-ref [78] ObjectReference OPTIONAL
}

```

```

OriginalBoxSize ::= SEQUENCE
{
  x-length INTEGER (0..MAX),
  y-length INTEGER (0..MAX)
}

```

-- A.25 Bitmap Class _____

```

BitmapClass ::= SET
{
  COMPONENTS OF VisibleClass
  (WITH COMPONENTS {..., original-content PRESENT}),
  tiling [79] BOOLEAN DEFAULT FALSE,
  original-transparency [80] INTEGER (0..100) DEFAULT 0
}

```

-- A.26 Line Art Class _____

```

LineArtClass ::= SET
{
  COMPONENTS OF VisibleClass
  (WITH COMPONENTS {..., original-content PRESENT}),
  bordered-bounding-box [81] BOOLEAN DEFAULT TRUE,
  original-line-width [82] INTEGER DEFAULT 1,
  original-line-style [83] INTEGER {solid(1), dashed(2), dotted(3)} DEFAULT solid,
  original-ref-line-colour [84] Colour OPTIONAL,
  original-ref-fill-colour [85] Colour OPTIONAL
}

```

-- A.27 Rectangle Class _____

```

RectangleClass ::= LineArtClass
(WITH COMPONENTS
  {..., content-hook ABSENT, original-content ABSENT, bordered-bounding-box ABSENT})

```

-- A.28 Dynamic Line Art Class _____

```
DynamicLineArtClass ::= LineArtClass
(WITH COMPONENTS
 {..., content-hook ABSENT, original-content ABSENT})
```

-- A.29 Text Class _____

```
TextClass ::= SET
{
  COMPONENTS OF VisibleClass
  (WITH COMPONENTS {..., original-content PRESENT}),
  original-font [86] FontBody OPTIONAL,
  font-attributes [43] OCTET STRING OPTIONAL,
  text-colour [41] Colour OPTIONAL,
  background-colour [39] Colour OPTIONAL,
  character-set [38] INTEGER OPTIONAL,
  horizontal-justification [87] Justification DEFAULT start,
  vertical-justification [88] Justification DEFAULT start,
  line-orientation [89] LineOrientation DEFAULT horizontal,
  start-corner [90] StartCorner DEFAULT upper-left,
  text-wrapping [91] BOOLEAN DEFAULT FALSE
}
```

```
Justification ::= 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)
}
```

-- A.30 Stream Class _____

```
StreamClass ::= SET
{
  COMPONENTS OF PresentableClass
  (WITH COMPONENTS {..., original-content PRESENT}),
  multiplex [92] SEQUENCE SIZE (1..MAX) OF StreamComponent,
  storage [93] Storage DEFAULT stream,
  looping [94] INTEGER {infinity(0)} DEFAULT 1
}
```

```
StreamComponent ::= CHOICE
{
  audio [95] AudioClass,
  video [96] VideoClass,
  rtgraphics [97] RTGraphicsClass
}
```

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

-- A.31 Audio Class _____

```
AudioClass ::= SET
{
  COMPONENTS OF PresentableClass
  (WITH COMPONENTS {..., content-hook ABSENT, original-content ABSENT, shared ABSENT}),
  component-tag [98] INTEGER,
  original-volume [99] INTEGER DEFAULT 0
}
```

-- A.32 Video Class _____

```
VideoClass ::= SET
{
  COMPONENTS OF VisibleClass
  (WITH COMPONENTS {..., content-hook ABSENT, original-content ABSENT, shared ABSENT, original-
palette-ref ABSENT}),
  component-tag [98] INTEGER,
  termination [100] Termination DEFAULT disappear
}
```

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

-- A.33 RTGraphics Class _____

```
RTGraphicsClass ::= SET
{
  COMPONENTS OF VisibleClass
  (WITH COMPONENTS {..., content-hook ABSENT, original-content ABSENT, shared ABSENT}),
  component-tag [98] INTEGER,
  termination [100] Termination DEFAULT disappear
}
```

-- A.34 Interactable Class _____

```
InteractableClass ::= SET
{
  engine-resp [101] BOOLEAN DEFAULT TRUE,
  highlight-ref-colour [49] Colour OPTIONAL
}
```

-- A.35 Slider Class _____

```
SliderClass ::= SET
{
  COMPONENTS OF VisibleClass
  (WITH COMPONENTS {..., content-hook ABSENT, original-content ABSENT}),
  COMPONENTS OF InteractableClass,
  orientation [102] Orientation,
  max-value [103] INTEGER,
  min-value [104] INTEGER DEFAULT 1,
  initial-value [105] INTEGER OPTIONAL,
  initial-portion [106] INTEGER OPTIONAL,
  step-size [107] INTEGER DEFAULT 1,
  slider-style [108] SliderStyle DEFAULT normal,
  slider-ref-colour [50] Colour OPTIONAL
}
```

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

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

-- A.36 Entry Field Class _____

EntryFieldClass ::= SET
{
 COMPONENTS OF TextClass,
 COMPONENTS OF InteractableClass,
 input-type [109] InputType DEFAULT any,
 char-list [110] OCTET STRING OPTIONAL,
 obscured-input [111] BOOLEAN DEFAULT FALSE,
 max-length [112] INTEGER DEFAULT 0
}

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

-- A.37 Hyper Text Class _____

HyperTextClass ::= SET
{
 COMPONENTS OF TextClass,
 COMPONENTS OF InteractableClass
}

-- A.38 Button Class _____

ButtonClass ::= SET
{
 COMPONENTS OF VisibleClass
 (WITH COMPONENTS {..., content-hook ABSENT, original-content ABSENT}),
 COMPONENTS OF InteractableClass,
 button-ref-colour [48] Colour OPTIONAL
}

-- A.39 Hotspot Class _____

HotspotClass ::= ButtonClass

-- A.40 Push Button Class _____

PushButtonClass ::= SET
{
 COMPONENTS OF ButtonClass,
 original-label [113] OCTET STRING OPTIONAL,
 character-set [38] INTEGER OPTIONAL
}

-- A.41 Switch Button Class _____

SwitchButtonClass ::= SET
{
 COMPONENTS OF PushButtonClass,
 button-style [114] ButtonStyle
}

ButtonStyle ::= ENUMERATED
{
 pushbutton(1),
 radiobutton(2),
 checkbox(3)
}

ActionClass ::= SEQUENCE SIZE (1..MAX) OF ElementaryAction

ElementaryAction ::= CHOICE

```
{
  activate [115] GenericObjectReference,
  add [116] Add,
  add-item [117] AddItem,
  append [118] Append,
  bring-to-front [119] GenericObjectReference,
  call [120] Call,
  call-action-slot [121] CallActionSlot,
  clear [122] GenericObjectReference,
  clone [123] Clone,
  close-connection [124] CloseConnection,
  deactivate [125] GenericObjectReference,
  del-item [126] DelItem,
  deselect [127] GenericObjectReference,
  deselect-item [128] DeselectItem,
  divide [129] Divide,
  draw-arc [130] DrawArc,
  draw-line [131] DrawLine,
  draw-oval [132] DrawOval,
  draw-polygon [133] DrawPolygon,
  draw-polyline [134] DrawPolyline,
  draw-rectangle [135] DrawRectangle,
  draw-sector [136] DrawSector,
  fork [137] Fork,
  get-availability-status [138] GetAvailabilityStatus,
  get-box-size [139] GetBoxSize,
  get-cell-item [140] GetCellItem,
  get-cursor-position [141] GetCursorPosition,
  get-engine-support [142] GetEngineSupport,
  get-entry-point [143] GetEntryPoint,
  get-fill-colour [144] GetFillColour,
  get-first-item [145] GetFirstItem,
  get-highlight-status [146] GetHighlightStatus,
  get-interaction-status [147] GetInteractionStatus,
  get-item-status [148] GetItemStatus,
  get-label [149] GetLabel,
  get-last-anchor-fired [150] GetLastAnchorFired,
  get-line-colour [151] GetLineColour,
  get-line-style [152] GetLineStyle,
  get-line-width [153] GetLineWidth,
  get-list-item [154] GetListItem,
  get-list-size [155] GetListSize,
  get-overwrite-mode [156] GetOverwriteMode,
  get-portion [157] GetPortion,
  get-position [158] GetPosition,
  get-running-status [159] GetRunningStatus,
  get-selection-status [160] GetSelectionStatus,
  get-slider-value [161] GetSliderValue,
  get-text-content [162] GetTextContent,
  get-text-data [163] GetTextData,
  get-token-position [164] GetTokenPosition,
  get-volume [165] GetVolume,
  launch [166] GenericObjectReference,
  lock-screen [167] GenericObjectReference,
  modulo [168] Modulo,
```

```

move [169] Move,
move-to [170] MoveTo,
multiply [171] Multiply,
open-connection [172] OpenConnection,
preload [173] GenericObjectReference,
put-before [174] PutBefore,
put-behind [175] PutBehind,
quit [176] GenericObjectReference,
read-persistent [177] ReadPersistent,
run [178] GenericObjectReference,
scale-bitmap [179] ScaleBitmap,
scale-video [180] ScaleVideo,
scroll-items [181] ScrollItems,
select [182] GenericObjectReference,
select-item [183] SelectItem,
send-event [184] SendEvent,
send-to-back [185] GenericObjectReference,
set-box-size [186] SetBoxSize,
set-cache-priority [187] SetCachePriority,
set-counter-end-position [188] SetCounterEndPosition,
set-counter-position [189] SetCounterPosition,
set-counter-trigger [190] SetCounterTrigger,
set-cursor-position [191] SetCursorPosition,
set-cursor-shape [192] SetCursorShape,
set-data [193] SetData,
set-entry-point [194] SetEntryPoint,
set-fill-colour [195] SetFillColour,
set-first-item [196] SetFirstItem,
set-font-ref [197] SetFontRef,
set-highlight-status [198] SetHighlightStatus,
set-interaction-status [199] SetInteractionStatus,
set-label [200] SetLabel,
set-line-colour [201] SetLineColour,
set-line-style [202] SetLineStyle,
set-line-width [203] SetLineWidth,
set-overwrite-mode [204] SetOverwriteMode,
set-palette-ref [205] SetPaletteRef,
set-portion [206] SetPortion,
set-position [207] SetPosition,
set-slider-value [208] SetSliderValue,
set-speed [209] SetSpeed,
set-timer [210] SetTimer,
set-transparency [211] SetTransparency,
set-variable [212] SetVariable,
set-volume [213] SetVolume,
spawn [214] GenericObjectReference,
step [215] Step,
stop [216] GenericObjectReference,
store-persistent [217] StorePersistent,
subtract [218] Subtract,
test-variable [219] TestVariable,
toggle [220] GenericObjectReference,
toggle-item [221] ToggleItem,
transition-to [222] TransitionTo,
unload [223] GenericObjectReference,
unlock-screen [224] GenericObjectReference
}

```

```

Add ::= SEQUENCE
{
    target GenericObjectReference,
    value GenericInteger
}

AddItem ::= SEQUENCE
{
    target GenericObjectReference,
    item-index GenericInteger,
    visible-reference GenericObjectReference
}

Append ::= SEQUENCE
{
    target GenericObjectReference,
    append-value GenericOctetString
}

Call ::= SEQUENCE
{
    target GenericObjectReference,
    call-succeeded ObjectReference,
    parameters SEQUENCE SIZE (1..MAX) OF Parameter OPTIONAL
}

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

Clone ::= SEQUENCE
{
    target GenericObjectReference,
    clone-ref-var ObjectReference
}

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

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

DeselectItem ::= SEQUENCE
{
    target GenericObjectReference,
    item-index GenericInteger
}

Divide ::= SEQUENCE
{
    target GenericObjectReference,
    value GenericInteger
}

```

```

DrawArc ::= SEQUENCE
{
    target GenericObjectReference,
    x GenericInteger,
    y GenericInteger,
    ellipse-width GenericInteger,
    ellipse-height GenericInteger,
    start-angle GenericInteger,
    arc-angle GenericInteger
}

DrawLine ::= SEQUENCE
{
    target GenericObjectReference,
    x1 GenericInteger,
    y1 GenericInteger,
    x2 GenericInteger,
    y2 GenericInteger
}

DrawOval ::= SEQUENCE
{
    target GenericObjectReference,
    x GenericInteger,
    y GenericInteger,
    ellipse-width GenericInteger,
    ellipse-height GenericInteger
}

DrawPolygon ::= SEQUENCE
{
    target GenericObjectReference,
    pointlist SEQUENCE SIZE (1..MAX) OF Point
}

DrawPolyline ::= SEQUENCE
{
    target GenericObjectReference,
    pointlist SEQUENCE SIZE (1..MAX) OF Point
}

DrawRectangle ::= SEQUENCE
{
    target GenericObjectReference,
    x1 GenericInteger,
    y1 GenericInteger,
    x2 GenericInteger,
    y2 GenericInteger
}

DrawSector ::= SEQUENCE
{
    target GenericObjectReference,
    x GenericInteger,
    y GenericInteger,
    ellipse-width GenericInteger,
    ellipse-height GenericInteger,
    start-angle GenericInteger,
    arc-angle GenericInteger
}

```

```

Fork ::= SEQUENCE
{
    target GenericObjectReference,
    fork-succeeded ObjectReference,
    parameters SEQUENCE SIZE (1..MAX) OF Parameter OPTIONAL
}

```

```

GetAvailabilityStatus ::= SEQUENCE
{
    target GenericObjectReference,
    availability-status-var ObjectReference
}

```

```

GetBoxSize ::= SEQUENCE
{
    target GenericObjectReference,
    x-box-size-var ObjectReference,
    y-box-size-var ObjectReference
}

```

```

GetCellItem ::= SEQUENCE
{
    target GenericObjectReference,
    cell-index GenericInteger,
    item-ref-var ObjectReference
}

```

```

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

```

```

GetEngineSupport ::= SEQUENCE
{
    target GenericObjectReference,
    feature GenericOctetString,
    answer ObjectReference
}

```

```

GetEntryPoint ::= SEQUENCE
{
    target GenericObjectReference,
    entry-point-var ObjectReference
}

```

```

GetFillColorour ::= SEQUENCE
{
    target GenericObjectReference,
    fill-colour-var ObjectReference
}

```

```

GetFirstItem ::= SEQUENCE
{
    target GenericObjectReference,
    first-item-var ObjectReference
}

```

```
GetHighlightStatus ::= SEQUENCE
{
    target GenericObjectReference,
    highlight-status-var ObjectReference
}
```

```
GetInteractionStatus ::= SEQUENCE
{
    target GenericObjectReference,
    interaction-status-var ObjectReference
}
```

```
GetItemStatus ::= SEQUENCE
{
    target GenericObjectReference,
    item-index GenericInteger,
    item-status-var ObjectReference
}
```

```
GetLabel ::= SEQUENCE
{
    target GenericObjectReference,
    label-var ObjectReference
}
```

```
GetLastAnchorFired ::= SEQUENCE
{
    target GenericObjectReference,
    last-anchor-fired-var ObjectReference
}
```

```
GetLineColour ::= SEQUENCE
{
    target GenericObjectReference,
    line-colour-var ObjectReference
}
```

```
GetLineStyle ::= SEQUENCE
{
    target GenericObjectReference,
    line-style-var ObjectReference
}
```

```
GetLineWidth ::= SEQUENCE
{
    target GenericObjectReference,
    line-width-var ObjectReference
}
```

```
GetListItem ::= SEQUENCE
{
    target GenericObjectReference,
    item-index GenericInteger,
    item-ref-var ObjectReference
}
```

```
GetListSize ::= SEQUENCE
{
    target GenericObjectReference,
    size-var ObjectReference
}
```

```
GetOverwriteMode ::= SEQUENCE
{
    target GenericObjectReference,
    overwrite-mode-var ObjectReference
}
```

```
GetPortion ::= SEQUENCE
{
    target GenericObjectReference,
    portion-var ObjectReference
}
```

```
GetPosition ::= SEQUENCE
{
    target GenericObjectReference,
    x-position-var ObjectReference,
    y-position-var ObjectReference
}
```

```
GetRunningStatus ::= SEQUENCE
{
    target GenericObjectReference,
    running-status-var ObjectReference
}
```

```
GetSelectionStatus ::= SEQUENCE
{
    target GenericObjectReference,
    selection-status-var ObjectReference
}
```

```
GetSliderValue ::= SEQUENCE
{
    target GenericObjectReference,
    slider-value-var ObjectReference
}
```

```
GetTextContent ::= SEQUENCE
{
    target GenericObjectReference,
    text-content-var ObjectReference
}
```

```
GetTextData ::= SEQUENCE
{
    target GenericObjectReference,
    text-data-var ObjectReference
}
```

```
GetTokenPosition ::= SEQUENCE
{
    target GenericObjectReference,
    token-position-var ObjectReference
}
```

```
GetVolume ::= SEQUENCE
{
    target GenericObjectReference,
    volume-var ObjectReference
}
```

```

Modulo ::= SEQUENCE
{
    target GenericObjectReference,
    value GenericInteger
}

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

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

Multiply ::= SEQUENCE
{
    target GenericObjectReference,
    value GenericInteger
}

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

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

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

ReadPersistent ::= SEQUENCE
{
    target GenericObjectReference,
    read-succeeded ObjectReference,
    out-variables SEQUENCE SIZE (1..MAX) OF ObjectReference,
    in-file-name GenericOctetString
}

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

```



```

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

ScrollItems ::= SEQUENCE
{
    target GenericObjectReference,
    items-to-scroll GenericInteger
}

SelectItem ::= SEQUENCE
{
    target GenericObjectReference,
    item-index GenericInteger
}

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

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

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

SetCounterEndPosition ::= SEQUENCE
{
    target GenericObjectReference,
    new-counter-end-position 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
}

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

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

SetFirstItem ::= SEQUENCE
{
    target GenericObjectReference,
    new-first-item GenericInteger
}

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

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

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

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

```

```

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

SetLineStyle ::= SEQUENCE
{
    target GenericObjectReference,
    new-line-style GenericInteger
}

SetLineWidth ::= SEQUENCE
{
    target GenericObjectReference,
    new-line-width GenericInteger
}

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

SetPaletteRef ::= SEQUENCE
{
    target GenericObjectReference,
    new-palette-ref GenericObjectReference
}

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
}

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

```

```

NewTimer ::= SEQUENCE
{
    timer-value GenericInteger,
    absolute-time GenericBoolean OPTIONAL
}

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

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

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

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

StorePersistent ::= SEQUENCE
{
    target GenericObjectReference,
    store-succeeded ObjectReference,
    in-variables SEQUENCE SIZE (1..MAX) OF ObjectReference,
    out-file-name GenericOctetString
}

Subtract ::= SEQUENCE
{
    target GenericObjectReference,
    value GenericInteger
}

TestVariable ::= SEQUENCE
{
    target GenericObjectReference,
    operator GenericInteger,
    comparison-value ComparisonValue
}

ToggleItem ::= SEQUENCE
{
    target GenericObjectReference,
    item-index GenericInteger
}

```

```

TransitionTo ::= SEQUENCE
{
    target GenericObjectReference,
    connection-tag-or-null ConnectionTagOrNull,
    transition-effect GenericInteger OPTIONAL
}

ConnectionTagOrNull ::= CHOICE
{
    connection-tag GenericInteger,
    null NULL
}

ComparisonValue ::= CHOICE
{
    new-generic-boolean [225] GenericBoolean,
    new-generic-integer [226] GenericInteger,
    new-generic-octetstring [227] GenericOctetString,
    new-generic-object-reference [228] GenericObjectReference,
    new-generic-content-reference [229] GenericContentReference
}

EmulatedEventData ::= CHOICE
{
    new-generic-boolean [225] GenericBoolean,
    new-generic-integer [226] GenericInteger,
    new-generic-octet-string [227] GenericOctetString
}

NewColour ::= CHOICE
{
    new-colour-index [230] GenericInteger,
    new-absolute-colour [231] GenericOctetString
}

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

NewFont ::= CHOICE
{
    new-font-name [232] GenericOctetString,
    new-font-reference [233] GenericObjectReference
}

NewReferencedContent ::= SEQUENCE
{
    generic-content-reference GenericContentReference,
    new-content-size [234] NewContentSize,
    new-content-cache-priority [235] GenericInteger OPTIONAL
}

NewContentSize ::= CHOICE
{
    content-size GenericInteger,
    null NULL
}

```

```

NewVariableValue ::= CHOICE
{
  new-generic-integer [226] GenericInteger,
  new-generic-boolean [225] GenericBoolean,
  new-generic-octet-string [227] GenericOctetString,
  new-generic-object-reference [228] GenericObjectReference,
  new-generic-content-reference [229] GenericContentReference
}

```

```

Parameter ::= CHOICE
{
  new-generic-boolean [225] GenericBoolean,
  new-generic-integer [226] GenericInteger,
  new-generic-octetstring [227] GenericOctetString,
  new-generic-object-reference [228] GenericObjectReference,
  new-generic-content-reference [229] GenericContentReference
}

```

```

Point ::= SEQUENCE
{
  x GenericInteger,
  y GenericInteger
}

```

```

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

```

-- A.43 Referencing Objects, Contents, Values, Colour and Position _____

```

ObjectReference ::= CHOICE
{
  external-reference ExternalReference,
  internal-reference INTEGER (1..MAX)
}

```

```

ExternalReference ::= SEQUENCE
{
  group-identifier OCTET STRING,
  object-number INTEGER (0..MAX)
}

```

```

IndirectReference ::= [236] ObjectReference

```

```

ContentReference ::= OCTET STRING

```

```

GenericObjectReference ::= CHOICE
{
  direct-reference ObjectReference,
  indirect-reference IndirectReference
}

```

```

GenericContentReference ::= CHOICE
{
  content-reference [69] ContentReference,
  indirect-reference IndirectReference
}

```

```

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

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

GenericOctetString ::= CHOICE
{
    octetstring OCTET STRING,
    indirect-reference IndirectReference
}

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

XYPosition ::= SEQUENCE
{
    x-position INTEGER,
    y-position INTEGER
}

END

```

ANEXO B

Notación textual para las aplicaciones MHEG-5

Este anexo describe la notación textual de objetos MHEG-5 conformes a ISO/CEI 13522-5. La representación alternativa es la notación ASN.1 definida en el anexo A.

B.1 Definiciones generales

B.1.1 Código

El conjunto de códigos que ha de ser utilizado por la notación textual será ISO/CEI 646 "Information technology – ISO 7-bit coded character set for information interchange" (Conjunto de caracteres codificados en 7 bits de la ISO para el intercambio de información). La notación textual utilizará un subconjunto de ISO/CEI 646, que estará comprendido en la gama de caracteres de SP (0x20) a ~ (0x7e), más HT (0x09), LF (0x0a), FF (0x0c) y CR (0x0d).

No se utilizarán otros caracteres.

NOTA 1 – A pesar de que la notación textual limita los códigos de caracteres que se han de utilizar, el contenido de datos de 8 bits puede ser codificado por medio de QPRINTABLE (véase B.3.4), BASE64 (véase B.3.5) y contenido externo referenciado por ContentReference.

NOTA 2 – Un dominio de aplicación podrá ampliar los códigos de caracteres que se han de utilizar en la notación textual en la medida en que no viole la gramática. Por ejemplo, se pudiera permitir también la utilización de caracteres de 0x80 a 0xfe para STRING y comentarios.

B.1.2 Delimitador (Delimiter)

HT (0x09), LF (0x0a), FF (0x0c), CR (0x0d) y SP (0x20) se denominan delimitadores.

La gramática descrita por la notación textual se basa en palabras. Una palabra es un paréntesis ("(" or ")"), un corchete ("{" or "}"), un rótulo (véase B.1.4) o un símbolo terminal (véase B.3). Se puede insertar cualquier número de delimitadores entre dos palabras adyacentes, sin cambiar la interpretación de las palabras. Sin embargo, por lo menos habrá un delimitador entre dos símbolos terminales y entre cualquier rótulo y cualquier símbolo terminal, puesto que si no serían interpretados como un terminal o un rótulo.

B.1.3 Comentario (Comment)

"/" (0x2f 0x2f) que no está dentro de una STRING, QPRINTABLE y BASE64 (véase B.3.3, B.3.4 y B.3.5) se utiliza para indicar el comienzo de un comentario. Todos los caracteres entre "/" (incluido //) y la siguiente aparición de LF (0x0a), FF (0x0c) o CR (0x0d) serán pasados por alto.

NOTA – HT (0x09) y SP (0x20) no indican el fin de comentarios.

B.1.4 Rótulo (Tag)

Un testigo que empieza con ":" (0x3a) se denomina "rótulo". Un rótulo está precedido por "{" (0x7b) cuando se utiliza al comienzo de objetos MHEG-5. El rótulo se utiliza para distinguir los objetos MHEG-5 y sus valores de atributo asociados, en general. Los rótulos son insensibles a letras mayúsculas o minúsculas, por ejemplo, ":Root", ":root", ":ROOT", ":rOOt", etc., son iguales. Sin embargo, en esta notación textual se utilizan algunas combinaciones de letras mayúsculas y minúsculas para facilitar la comprensión y aumentar la legibilidad.

B.2 Definición de símbolos

El cuadro B.1 muestra los símbolos utilizados en la notación textual y sus significados.

Cuadro B.1/T.172 – Definición de símbolos en notación textual

Símbolo	Definición
::=	Se define como
	Alternativo
<">	Dobles comillas (0x22)
"text"	Literales incluidos entre comillas
<text>	Descripción en texto claro que explica los códigos presentes
*	La unidad sintáctica precedente puede ser repetida ninguna vez o más veces
+	La unidad sintáctica precedente puede ser repetida una o más veces
[]	La unidad sintáctica dentro de los corchetes es facultativa. Puede producirse ninguna vez o una vez
.	Fin de cláusula

B.3 Símbolos terminales

Todos los símbolos terminales utilizados en la notación textual se definen como sigue.

B.3.1 INTEGER (ENTERO)

Un valor decimal o entero hexadecimal positivo.

Definición:

INTEGER ::= DECINT | HEXINT | "0".
DECINT ::= ["-"] DIGIT [DIGIT0]*.
DIGIT ::= "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9".
DIGIT0 ::= DIGIT | "0".
HEXINT ::= HEXPREFIX HEXDIGIT0 [HEXDIGIT0]*.
HEXPREFIX ::= "0x" | "0X".
HEXDIGIT0 ::= DIGIT | "0" | "a" | "b" | "c" | "d" | "e" | "f" | "A" | "B" | "C" | "D" | "E" | "F".

B.3.2 BOOLEAN (BOOLEANO)

Un valor booleano puede ser verdadero o falso. El terminal BOOLEAN es insensible a las letras mayúsculas o minúsculas, es decir, "True", "TRUE" y "true" son iguales; "False", "FALSE" y "false" son iguales.

Definición:

BOOLEAN ::= "true" | "false".

B.3.3 STRING (CADENA)

Un valor de cadena colocado entre comillas dobles puede contener un número arbitrario de caracteres imprimibles (de 0x20 a 0x7e). Una doble comilla (0x22) dentro de una STRING se codificará como "\" (0x5c 0x22) y una barra inclinada (0x5c) se codificará como "\\" (0x5c 0x5c).

Obsérvese que no se incluyen cortes de línea en STRING, es decir, el contenido de texto multilíneas se codificará utilizando QPRINTABLE o BASE64.

Definición:

STRING ::= <"> STRINGCHAR* <">.
STRINGCHAR ::= <any single printable character except <"> and "\"> | "\" | "\\".

B.3.4 QPRINTABLE

Un valor de cadena colocado entre comillas contendrá un contenido codificado imprimible definido en RFC 1521. Sin embargo, " " (0x27) se recodificará como =27. Los caracteres en minúscula, tales como "a", "b", "c", "d", "e" y "f", se pueden utilizar como una representación general de 8 bits definida por la regla #1 de la sección 5.1 en RFC 1521. El número de caracteres en una línea no está limitado. Los cortes de línea no tienen que ser convertidos a CR (0x0d)/LF (0x0a), sin embargo, se utilizará por lo menos uno de LF (0x0a), FF (0x0c) y CR (0x0d) que representan un corte de línea.

Definición:

QPRINTABLE ::= "" QPRINTABLECHAR* "".
QPRINTABLECHAR ::= <a character as defined above> | <encoding sequence of a character as defined above>.

B.3.5 BASE64

Un valor de cadena colocado entre comillas contendrá un contenido codificado en BASE64 definido en RFC 1521. Sin embargo, el número de caracteres en una línea no está limitado en esta norma. El contenido BASE64 codificado puede ser dividido en varios segmentos por al menos uno de LF (0x0a), FF (0x0c) o CR (0x0d). Sin embargo, estos caracteres serán pasados por alto, y los segmentos codificados BASE64 introducidos serán tratados como si estuviesen en una línea.

Definición:

BASE64 ::= "" BASE64CHAR* "".
BASE64CHAR ::= <an encoding sequence of a character as defined above>.

B.3.6 Null (Nulo)

Null representa un terminal especial cuya semántica depende de la definición de objetos MHEG-5. El terminal Null es insensible a mayúsculas o minúsculas, es decir "NULL" y "null" son iguales.

Definición:

Null ::= "NULL".

B.3.7 Enumeration Values (Valores de enumeración)

Una palabra terminal que comienza con un carácter del alfabeto se denomina "valor de enumeración" (todos los valores de enumeración están colocados entre comillas dobles en la gramática siguiente). Un valor de enumeración se utiliza como uno de los símbolos terminales que representan una semántica específica dependiendo de su uso. Todos los valores de enumeración son insensibles a mayúsculas o minúsculas, es decir "IsAvailable" o "isavailable" son iguales.

B.4 Definiciones de objetos MHEG-5

La notación textual de objetos MHEG-5 se define a continuación.

El cuadro B.2 resume las abreviaturas utilizadas en los rótulos adoptando longitudes razonables a la vez que mantiene la legibilidad.

Cuadro B.2/T.172 – Abreviaturas de rótulo

Bordered Bounding Box	BBBox
Content Cache Priority	CCPriority
Content Hook	CHook
Coordinate System	CS
Generic	G
Group Cache Priority	GCPriority
Horizontal Justification	HJustification
Identifier	ID
Information	Info
Interchanged	Interchg
OctetString	OString
Original	Orig
Program	Prg
Reference	Ref
Register	Reg
Standard	Std
Variable	Var (excepto para rótulos de acciones elementales)
Vertical Justification	VJustification

B.4.1 Root Class (Clase Raíz)

Root ::= **ObjectIdentifier**.

ObjectIdentifier ::= **ObjectReference**.

B.4.2 Group Class (Clase Grupo)

```
Group      ::= Root [StandardIdentifier]
              [StandardVersion] [ObjectInformation]
              [OnStartUp] [OnCloseDown]
              [OriginalGroupCachePriority] [Items].
StandardIdentifier  ::= ":StdID" JointIsoItuIdentifier
              MHEGStandardIdentifier.
JointIsoItuIdentifier  ::= INTEGER.
MHEGStandardIdentifier  ::= INTEGER.
StandardVersion      ::= ":StdVersion" INTEGER.
ObjectInformation     ::= ":ObjectInfo" OctetString.
OnStartUp            ::= ":OnStartUp" ActionClass.
OnCloseDown          ::= ":OnCloseDown" ActionClass.
OriginalGroupCachePriority  ::= ":OrigGCPriority" INTEGER.
Items                ::= ":Items" "(" GroupItem+ ")".
GroupItem            ::= ResidentProgramClass |
              RemoteProgramClass |
              InterchangedProgramClass |
              PaletteClass |
              FontClass |
              CursorShapeClass |
              BooleanVariableClass |
              IntegerVariableClass |
              OctetStringVariableClass |
              ObjectRefVariableClass |
              ContentRefVariableClass |
              LinkClass |
              StreamClass |
              BitmapClass |
              LineArtClass |
              DynamicLineArtClass |
              RectangleClass |
              HotspotClass |
              SwitchButtonClass |
              PushButtonClass |
              TextClass |
              EntryFieldClass |
              HyperTextClass |
              SliderClass |
              TokenGroupClass |
              ListGroupClass.
```

B.4.3 Application Class (Clase Aplicación)

```
ApplicationClass  ::= "{:Application" Group
              [OnSpawnCloseDown] [OnRestart]
              [DefaultAttributes] }".
OnSpawnCloseDown  ::= ":OnSpawnCloseDown" ActionClass.
OnRestart         ::= ":OnRestart" ActionClass.
DefaultAttributes ::= DefaultAttribute+.
DefaultAttribute  ::= CharacterSet | BackgroundColour
              | TextContentHook
              | TextColour | Font
              | FontAttributes
              | InterchangedProgramContentHook
              | StreamContentHook
              | BitmapContentHook
              | LineArtContentHook | ButtonRefColour
              | HighlightRefColour | SliderRefColour.
```

CharacterSet ::= ":CharacterSet" INTEGER.
BackgroundColour ::= ":BackgroundColour" Colour.
TextContentHook ::= ":TextCHook" INTEGER.
TextColour ::= ":TextColour" Colour.
Font ::= ":Font" FontBody.
FontBody ::= DirectFont | IndirectFont.
DirectFont ::= OctetString.
IndirectFont ::= ObjectReference.
FontAttributes ::= ":FontAttributes" OctetString.
InterchangedProgramContentHook ::= ":InterchgPrgCHook" INTEGER.
StreamContentHook ::= ":StreamCHook" INTEGER.
BitmapContentHook ::= ":BitmapCHook" INTEGER.
LineArtContentHook ::= ":LineArtCHook" INTEGER.
ButtonRefColour ::= ":ButtonRefColour" Colour.
HighlightRefColour ::= ":HighlightRefColour" Colour.
SliderRefColour ::= ":SliderRefColour" Colour.

B.4.4 Scene Class (Clase Escena)

SceneClass ::= "{:Scene" Group InputEventRegister
 SceneCoordinateSystem [AspectRatio]
 [MovingCursor] [NextScenes] }".
InputEventRegister ::= ":InputEventReg" INTEGER.
SceneCoordinateSystem ::= ":SceneCS" XScene YScene.
XScene ::= INTEGER.
YScene ::= INTEGER.
AspectRatio ::= ":AspectRatio" Width Height.
Width ::= INTEGER.
Height ::= INTEGER.
MovingCursor ::= ":MovingCursor" BOOLEAN.
NextScenes ::= ":NextScenes" "(" NextScene+ ")".
NextScene ::= "(" SceneRef SceneWeight ")".
SceneRef ::= OctetString.
SceneWeight ::= INTEGER.

B.4.5 Ingredient Class (Clase Ingrediente)

Ingredient ::= Root [InitiallyActive] [ContentHook]
 [OriginalContent] [Shared].
InitiallyActive ::= ":InitiallyActive" BOOLEAN.
ContentHook ::= ":CHook" INTEGER.
OriginalContent ::= ":OrigContent" ContentBody.
ContentBody ::= IncludedContent | ReferencedContent.
IncludedContent ::= OctetString.
ReferencedContent ::= ":ContentRef" "(" ContentReference
 [ContentSize] [ContentCachePriority]
 ")".
ContentSize ::= ":ContentSize" INTEGER.
ContentCachePriority ::= ":CCPriority" INTEGER.
Shared ::= ":Shared" BOOLEAN.

B.4.6 Link Class (Clase Enlace)

LinkClass ::= "{:Link" Ingredient LinkCondition
 LinkEffect }".
LinkCondition ::= EventSource EventType [EventData].
EventSource ::= ":EventSource" ObjectReference.
EventType ::= ":EventType" EventTypeEnum.

```

EventTypeEnum ::= "IsAvailable" | "ContentAvailable"
               | "IsDeleted" | "IsRunning"
               | "IsStopped" | "UserInput"
               | "AnchorFired" | "TimerFired"
               | "AsynchStopped" | "InteractionCompleted"
               | "TokenMovedFrom" | "TokenMovedTo"
               | "StreamEvent" | "StreamPlaying"
               | "StreamStopped" | "CounterTrigger"
               | "HighlightOn" | "HighlightOff"
               | "CursorEnter" | "CursorLeave"
               | "IsSelected" | "IsDeselected"
               | "TestEvent" | "FirstItemPresented"
               | "LastItemPresented" | "HeadItems"
               | "TailItems" | "ItemSelected"
               | "ItemDeselected" | "EntryFieldFull"
               | "EngineEvent".
EventData ::= ":EventData" EventDataBody.
EventDataBody ::= OctetString | BOOLEAN | INTEGER.
LinkEffect ::= ":LinkEffect" ActionClass.

```

B.4.7 Program Class (Clase Programa)

```

Program ::= Ingredient Name [InitiallyAvailable].
Name ::= ":Name" OctetString.
InitiallyAvailable ::= ":InitiallyAvailable" BOOLEAN.

```

B.4.8 ResidentProgram Class (Clase Programa Residente)

```

ResidentProgramClass ::= "{:ResidentPrg" Program "}".

```

B.4.9 RemoteProgram Class (Clase Programa Distante)

```

RemoteProgramClass ::= "{:RemotePrg" Program
[ProgramConnectionTag] "}".
ProgramConnectionTag ::= ":ConnectionTag" INTEGER.

```

B.4.10 InterchangedProgram Class (Clase Programa Intercambiado)

```

InterchangedProgramClass ::= "{:InterchgPrg" Program "}".

```

B.4.11 Palette Class (Clase Paleta)

```

PaletteClass ::= "{:Palette" Ingredient "}".

```

B.4.12 Font Class (Clase Tipo de Carácter)

```

FontClass ::= "{:Font" Ingredient "}".

```

B.4.13 CursorShape Class (Clase Forma de Cursor)

```

CursorShapeClass ::= "{:CursorShape" Ingredient "}".

```

B.4.14 Variable Class (Clase Variable)

```

Variable ::= Ingredient OriginalValue.
OriginalValue ::= ":OrigValue" OriginalValueBody.
OriginalValueBody ::= BOOLEAN | INTEGER | OctetString
| ObjectReferenceValue
| ContentReferenceValue.
ObjectReferenceValue ::= ":ObjectRef" ObjectReference.
ContentReferenceValue ::= ":ContentRef" ContentReference.

```

B.4.15 BooleanVariable Class (Clase Variable Booleana)

BooleanVariableClass ::= "{:BooleanVar" Variable "}".

B.4.16 IntegerVariable Class (Clase Variable Entera)

IntegerVariableClass ::= "{:IntegerVar" Variable "}".

B.4.17 OctetStringVariable Class (Clase Variable Cadena de Octetos)

OctetStringVariableClass ::= "{:OStringVar" Variable "}".

B.4.18 ObjectRefVariable Class (Clase Variable Referencia de Objeto)

ObjectRefVariableClass ::= "{:ObjectRefVar" Variable "}".

B.4.19 ContentRefVariable Class (Clase Variable Referencia de Contenido)

ContentRefVariableClass ::= "{:ContentRefVar" Variable "}".

B.4.20 Presentable Class (Clase Presentable)

Presentable ::= Ingredient.

B.4.21 TokenManager Class (Clase Gestor de Testigo)

TokenManager ::= [MovementTable].
MovementTable ::= ":MovementTable" "(" Movement+ ")".
Movement ::= "(" TargetElement+ ")".
TargetElement ::= INTEGER.

B.4.22 TokenGroup Class (Clase Grupo de Testigo)

TokenGroupClass ::= "{:TokenGroup" TokenGroupBody "}".
TokenGroupBody ::= Presentable TokenManager TokenGroupItems
[NoTokenActionSlots].
TokenGroupItems ::= ":TokenGroupItems" "(" TokenGroupItem+ ")".
TokenGroupItem ::= "(" AVisible [ActionSlots] ")".
AVisible ::= ObjectReference.
ActionSlots ::= ":ActionSlots" "(" ActionSlot+ ")".
ActionSlot ::= ActionClass | Null.
NoTokenActionSlots ::= ":NoTokenActionSlots" "(" ActionSlot+ ")".

B.4.23 ListGroup Class (Clase Grupo de Listas)

ListGroupClass ::= "{:ListGroup" TokenGroupBody
Positions [WrapAround]
[MultipleSelection] "}".
Positions ::= ":Positions" "(" Position+ ")".
Position ::= "(" XYPosition ")".
WrapAround ::= ":WrapAround" BOOLEAN.
MultipleSelection ::= ":MultipleSelection" BOOLEAN.

B.4.24 Visible Class (Clase Visible)

Visible ::= Presentable OriginalBoxSize
[OriginalPosition] [OriginalPaletteRef].
OriginalBoxSize ::= ":OrigBoxSize" BoxSize.
BoxSize ::= XLength YLength.
XLength ::= INTEGER.
YLength ::= INTEGER.

OriginalPosition ::= ":OrigPosition" XYPosition.
OriginalPaletteRef ::= ":OrigPaletteRef" ObjectReference.

B.4.25 Bitmap Class (Clase Diagrama de Bits)

BitmapClass ::= "{:Bitmap" Visible [Tiling]
[OriginalTransparency] }".
Tiling ::= ":Tiling" BOOLEAN.
OriginalTransparency ::= ":OrigTransparency" INTEGER.

B.4.26 LineArt Class (Clase Arte Lineal)

LineArtClass ::= "{:LineArt" LineArtBody }".
LineArtBody ::= Visible [BorderedBoundingBox]
[OriginalLineWidth]
[OriginalLineStyle]
[OriginalRefLineColour]
[OriginalRefFillColour] }".
BorderedBoundingBox ::= ":BBBox" BOOLEAN.
OriginalLineWidth ::= ":OrigLineWidth" INTEGER.
OriginalLineStyle ::= ":OrigLineStyle" INTEGER.
OriginalRefLineColour ::= ":OrigRefLineColour" Colour.
OriginalRefFillColour ::= ":OrigRefFillColour" Colour.

B.4.27 Rectangle Class (Clase Rectángulo)

RectangleClass ::= "{:Rectangle" LineArtBody }".

B.4.28 DynamicLineArt Class (Clase Arte Lineal Dinámico)

DynamicLineArtClass ::= "{:DynamicLineArt" LineArtBody }".

B.4.29 Text Class (Clase Texto)

TextClass ::= "{:Text" TextBody }".
TextBody ::= Visible [OriginalFont] [FontAttributes]
[TextColour] [BackgroundColour]
[CharacterSet]
[HorizontalJustification]
[VerticalJustification]
[LineOrientation] [StartCorner]
[TextWrapping].
OriginalFont ::= ":OrigFont" FontBody.
HorizontalJustification ::= ":HJustification" JustificationEnum.
JustificationEnum ::= "start" | "end" | "centre" | "justified".
VerticalJustification ::= ":VJustification" JustificationEnum.
LineOrientation ::= ":LineOrientation" LineOrientationEnum.
LineOrientationEnum ::= "vertical" | "horizontal".
StartCorner ::= ":StartCorner" StartCornerEnum.
StartCornerEnum ::= "upper-left" | "upper-right"
| "lower-left" | "lower-right".
TextWrapping ::= ":TextWrapping" BOOLEAN.

B.4.30 Stream Class (Clase Tren)

StreamClass ::= "{:Stream" Presentable Multiplex
[Storage] [Looping] }".
Multiplex ::= ":Multiplex" "(" StreamComponent+ ")".
StreamComponent ::= AudioClass | VideoClass | RTGraphicsClass.
Storage ::= ":Storage" StorageEnum.
StorageEnum ::= "memory" | "stream".
Looping ::= ":Looping" INTEGER.

B.4.31 Audio Class (Clase Audio)

```
AudioClass      ::= "{:Audio" Presentable ComponentTag
                  [OriginalVolume] "}".
ComponentTag    ::= ":ComponentTag" INTEGER.
OriginalVolume  ::= ":OrigVolume" INTEGER.
```

B.4.32 Video Class (Clase Vídeo)

```
VideoClass      ::= "{:Video" Visible ComponentTag
                  [Termination].
Termination     ::= ":Termination" TerminationEnum.
TerminationEnum ::= "freeze" | "disappear".
```

B.4.33 RTGraphics Class (Clase Gráficos RT)

```
RTGraphicsClass ::= "{:RTGraphics" Visible ComponentTag
                  [Termination] "}".
```

B.4.34 Interactable Class (Clase Interactuable)

```
Interactable    ::= [EngineResp] [HighlightRefColour].
EngineResp      ::= ":EngineResp" BOOLEAN.
```

B.4.35 Slider Class (Clase Deslizador)

```
SliderClass      ::= "{:Slider" Visible Interactable
                  Orientation MaxValue [MinValue]
                  [InitialValue] [InitialPortion]
                  [StepSize] [SliderStyle]
                  [SliderRefColour] "}".
Orientation      ::= ":Orientation" OrientationEnum.
OrientationEnum  ::= "left" | "right" | "up" | "down".
MaxValue         ::= ":MaxValue" INTEGER.
MinValue         ::= ":MinValue" INTEGER.
InitialValue     ::= ":InitialValue" INTEGER.
InitialPortion   ::= ":InitialPortion" INTEGER.
StepSize         ::= ":StepSize" INTEGER.
SliderStyle      ::= ":SliderStyle" SliderStyleEnum.
SliderStyleEnum  ::= "normal" | "thermometer" | "proportional".
```

B.4.36 EntryField Class (Clase Campo de Entrada)

```
EntryFieldClass  ::= "{:EntryField" TextBody Interactable
                  [InputType] [CharList]
                  [ObscuredInput] [MaxLength] "}".
InputType        ::= ":InputType" InputTypeEnum.
InputTypeEnum     ::= "alpha" | "numeric" | "any" | "listed".
CharList         ::= ":CharList" OctetString.
ObscuredInput    ::= ":ObscuredInput" BOOLEAN.
MaxLength        ::= ":MaxLength" INTEGER.
```

B.4.37 HyperText Class (Clase Hipertexto)

```
HyperTextClass   ::= "{:HyperText" TextBody Interactable
                  "}".
```

B.4.38 Button Class (Clase Botón)

```
Button           ::= Visible Interactable [ButtonRefColour].
```


B.4.39 Hotspot Class (Clase Punto Focal)

HotspotClass ::= "{:Hotspot" Button "}".

B.4.40 PushButton Class (Clase Botón de Pulsación)

PushButtonClass ::= "{:PushButton" PushButtonBody "}".
PushButtonBody ::= Button [OriginalLabel] [CharacterSet].
OriginalLabel ::= ":OrigLabel" OctetString.

B.4.41 SwitchButton Class (Clase Botón de Conmutación)

SwitchButtonClass ::= "{:SwitchButton" PushButtonBody
ButtonStyle "}".
ButtonStyle ::= ":ButtonStyle" ButtonStyleEnum.
ButtonStyleEnum ::= "pushbutton" | "radiobutton" | "checkbox".

B.4.42 Action Class (Clase Acción)

ActionClass ::= "(" ElementaryAction+ ")".
ElementaryAction ::= Activate

| Add
| AddItem
| Append
| BringToFront
| Call
| CallActionSlot
| Clear
| Clone
| CloseConnection
| Deactivate
| DelItem
| Deselect
| DeselectItem
| Divide
| DrawArc
| DrawLine
| DrawOval
| DrawPolygon
| DrawPolyline
| DrawRectangle
| DrawSector
| Fork
| GetAvailabilityStatus
| GetBoxSize
| GetCellItem
| GetCursorPosition
| GetEngineSupport
| GetEntryPoint
| GetFillColour
| GetFirstItem
| GetHighlightStatus
| GetInteractionStatus
| GetItemStatus
| GetLabel
| GetLastAnchorFired
| GetLineColour
| GetLineStyle
| GetLineWidth
| GetListItem
| GetListSize

- | **GetOverwriteMode**
- | **GetPortion**
- | **GetPosition**
- | **GetRunningStatus**
- | **GetSelectionStatus**
- | **GetSliderValue**
- | **GetTextContent**
- | **GetTextData**
- | **GetTokenPosition**
- | **GetVolume**
- | **Launch**
- | **LockScreen**
- | **Modulo**
- | **Move**
- | **MoveTo**
- | **Multiply**
- | **OpenConnection**
- | **Preload**
- | **PutBefore**
- | **PutBehind**
- | **Quit**
- | **ReadPersistent**
- | **Run**
- | **ScaleBitmap**
- | **ScaleVideo**
- | **ScrollItems**
- | **Select**
- | **SelectItem**
- | **SendEvent**
- | **SendToBack**
- | **SetBoxSize**
- | **SetCachePriority**
- | **SetCounterEndPosition**
- | **SetCounterPosition**
- | **SetCounterTrigger**
- | **SetCursorPosition**
- | **SetCursorShape**
- | **SetData**
- | **SetEntryPoint**
- | **SetFillColour**
- | **SetFirstItem**
- | **SetFontRef**
- | **SetHighlightStatus**
- | **SetInteractionStatus**
- | **SetLabel**
- | **SetLineColour**
- | **SetLineStyle**
- | **SetLineWidth**
- | **SetOverwriteMode**
- | **SetPaletteRef**
- | **SetPortion**
- | **SetPosition**
- | **SetSliderValue**
- | **SetSpeed**
- | **SetTimer**
- | **SetTransparency**
- | **SetVariable**
- | **SetVolume**
- | **Spawn**
- | **Step**
- | **Stop**

| StorePersistent
| Subtract
| TestVariable
| Toggle
| ToggleItem
| TransitionTo
| Unload
| UnlockScreen.

Activate ::= ":Activate" "(" Target ")".
Add ::= ":Add" "(" Target Value ")".
AddItem ::= ":AddItem" "(" Target ItemIndex
VisibleReference ")".
Append ::= ":Append" "(" Target AppendValue ")".
BringToFront ::= ":BringToFront" "(" Target ")".
Call ::= ":Call" "(" Target CallSucceeded
[Parameters] ")".
CallActionSlot ::= ":CallActionSlot" "(" Target Index ")".
Clear ::= ":Clear" "(" Target ")".
Clone ::= ":Clone" "(" Target CloneRefVar ")".
CloseConnection ::= ":CloseConnection" "(" Target
ConnectionTag ")".
Deactivate ::= ":Deactivate" "(" Target ")".
DelItem ::= ":DelItem" "(" Target VisibleReference
")".
Deselect ::= ":Deselect" "(" Target ")".
DeselectItem ::= ":DeselectItem" "(" Target ItemIndex ")".
Divide ::= ":Divide" "(" Target Value ")".
DrawArc ::= ":DrawArc" "(" Target X Y EllipseWidth
EllipseHeight StartAngle ArcAngle ")".
DrawLine ::= ":DrawLine" "(" Target X1 Y1 X2 Y2 ")".
DrawOval ::= ":DrawOval" "(" Target X Y EllipseWidth
EllipseHeight ")".
DrawPolygon ::= ":DrawPolygon" "(" Target PointList ")".
DrawPolyline ::= ":DrawPolyline" "(" Target PointList ")".
DrawRectangle ::= ":DrawRectangle" "(" Target X1 Y1 X2 Y2
")".
DrawSector ::= ":DrawSector" "(" Target X Y EllipseWidth
EllipseHeight StartAngle ArcAngle ")".
Fork ::= ":Fork" "(" Target ForkSucceeded
[Parameters] ")".
GetAvailabilityStatus ::= ":GetAvailabilityStatus" "(" Target
AvailabilityStatusVar ")".
GetBoxSize ::= ":GetBoxSize" "(" Target XBoxSizeVar
YBoxSizeVar ")".
GetCellItem ::= ":GetCellItem" "(" Target CellIndex
ItemRefVar ")".
GetCursorPosition ::= ":GetCursorPosition" "(" Target XOut YOut
")".
GetEngineSupport ::= ":GetEngineSupport" "(" Target Feature
Answer ")".
GetEntryPoint ::= ":GetEntryPoint" "(" Target EntryPointVar
")".
GetFillColour ::= ":GetFillColour" "(" Target FillColourVar
")".
GetFirstItem ::= ":GetFirstItem" "(" Target FirstItemVar
")".
GetHighlightStatus ::= ":GetHighlightStatus" "(" Target
HighlightStatusVar ")".
GetInteractionStatus ::= ":GetInteractionStatus" "(" Target
InteractionStatusVar ")".

```

GetItemStatus      ::= ":GetItemStatus" "(" Target
                        ItemIndex ItemStatusVar ")".
GetLabel           ::= ":GetLabel" "(" Target LabelVar ")".
GetLastAnchorFired ::= ":GetLastAnchorFired" "(" Target
                        LastAnchorFiredVar ")".
GetLineColour      ::= ":GetLineColour" "(" Target LineColourVar
                        ")".
GetLineStyle       ::= ":GetLineStyle" "(" Target LineStyleVar
                        ")".
GetLineWidth       ::= ":GetLineWidth" "(" Target LineWidthVar
                        ")".
GetListItem        ::= ":GetListItem" "(" Target ItemIndex
                        ItemRefVar ")".
GetListSize        ::= ":GetListSize" "(" Target SizeVar ")".
GetOverwriteMode   ::= ":GetOverwriteMode" "(" Target
                        OverwriteModeVar ")".
GetPortion         ::= ":GetPortion" "(" Target PortionVar ")".
GetPosition        ::= ":GetPosition" "(" Target XPositionVar
                        YPositionVar ")".
GetRunningStatus   ::= ":GetRunningStatus" "(" Target
                        RunningStatusVar ")".
GetSelectionStatus ::= ":GetSelectionStatus" "(" Target
                        SelectionStatusVar ")".
GetSliderValue     ::= ":GetSliderValue" "(" Target
                        SliderValueVar ")".
GetTextContent     ::= ":GetTextContent" "(" Target
                        TextContentVar ")".
GetTextData        ::= ":GetTextData" "(" Target TextDataVar ")".
GetTokenPosition   ::= ":GetTokenPosition" "(" Target
                        TokenPositionVar ")".
GetVolume          ::= ":GetVolume" "(" Target VolumeVar ")".
Launch             ::= ":Launch" "(" Target ")".
LockScreen         ::= ":LockScreen" "(" Target ")".
Modulo             ::= ":Modulo" "(" Target Value ")".
Move               ::= ":Move" "(" Target MovementIdentifier ")".
MoveTo             ::= ":MoveTo" "(" Target Index ")".
Multiply           ::= ":Multiply" "(" Target Value ")".
OpenConnection     ::= ":OpenConnection" "(" Target OpenSucceeded
                        Protocol Address ConnectionTag ")".
Preload           ::= ":Preload" "(" Target ")".
PutBefore          ::= ":PutBefore" "(" Target ReferenceVisible
                        ")".
PutBehind          ::= ":PutBehind" "(" Target ReferenceVisible
                        ")".
Quit              ::= ":Quit" "(" Target ")".
ReadPersistent     ::= ":ReadPersistent" "(" Target ReadSucceeded
                        OutVariables InFileName ")".
Run               ::= ":Run" "(" Target ")".
ScaleBitmap        ::= ":ScaleBitmap" "(" Target XScale YScale
                        ")".
ScaleVideo         ::= ":ScaleVideo" "(" Target XScale YScale ")".
ScrollItems        ::= ":ScrollItems" "(" Target ItemsToScroll ")".
Select             ::= ":Select" "(" Target ")".
SelectItem         ::= ":SelectItem" "(" Target ItemIndex ")".
SendEvent          ::= ":SendEvent" "(" Target EmulatedEventSource
                        EmulatedEventType [EmulatedEventData]
                        ")".
SendToBack         ::= ":SendToBack" "(" Target ")".
SetBoxSize         ::= ":SetBoxSize" "(" Target XNewBoxSize
                        YNewBoxSize ")".

```

```

SetCachePriority ::= ":SetCachePriority" "(" Target
    NewCachePriority ")".
SetCounterEndPosition ::= ":SetCounterEndPosition" "(" Target
    NewCounterEndPosition ")".
SetCounterPosition ::= ":SetCounterPosition" "(" Target
    NewCounterPosition ")".
SetCounterTrigger ::= ":SetCounterTrigger" "(" Target
    TriggerIdentifier [NewCounterValue]
    ")".
SetCursorPosition ::= ":SetCursorPosition" "(" Target XCursor
    YCursor ")".
SetCursorShape ::= ":SetCursorShape" "(" Target
    [NewCursorShape] ")".
SetData ::= ":SetData" "(" Target NewContent ")".
SetEntryPoint ::= ":SetEntryPoint" "(" Target NewEntryPoint
    ")".
SetFillColour ::= ":SetFillColour" "(" Target [NewColour]
    ")".
SetFirstItem ::= ":SetFirstItem" "(" Target NewFirstItem
    ")".
SetFontRef ::= ":SetFontRef" "(" Target NewFont ")".
SetHighlightStatus ::= ":SetHighlightStatus" "(" Target
    NewHighlightStatus ")".
SetInteractionStatus ::= ":SetInteractionStatus" "(" Target
    NewInteractionStatus ")".
SetLabel ::= ":SetLabel" "(" Target NewLabel ")".
SetLineColour ::= ":SetLineColour" "(" Target NewColour ")".
SetLineStyle ::= ":SetLineStyle" "(" Target NewLineStyle
    ")".
SetLineWidth ::= ":SetLineWidth" "(" Target NewLineWidth
    ")".
SetOverwriteMode ::= ":SetOverwriteMode" "(" Target
    NewOverwriteMode ")".
SetPaletteRef ::= ":SetPaletteRef" "(" Target NewPaletteRef
    ")".
SetPortion ::= ":SetPortion" "(" Target NewPortion ")".
SetPosition ::= ":SetPosition" "(" Target NewXPosition
    NewYPosition ")".
SetSliderValue ::= ":SetSliderValue" "(" Target
    NewSliderValue ")".
SetSpeed ::= ":SetSpeed" "(" Target NewSpeed ")".
SetTimer ::= ":SetTimer" "(" Target TimerID
    [TimerValue] [AbsoluteTime] ")".
SetTransparency ::= ":SetTransparency" "(" Target
    NewTransparency ")".
SetVariable ::= ":SetVariable" "(" Target
    NewVariableValue ")".
SetVolume ::= ":SetVolume" "(" Target NewVolume ")".
Spawn ::= ":Spawn" "(" Target ")".
Stop ::= ":Stop" "(" Target ")".
Step ::= ":Step" "(" Target NbOfSteps ")".
StorePersistent ::= ":StorePersistent" "(" Target
    StoreSucceeded InVariables OutFileName
    ")".
Subtract ::= ":Subtract" "(" Target Value ")".
TestVariable ::= ":TestVariable" "(" Target Operator
    ComparisonValue ")".
Toggle ::= ":Toggle" "(" Target ")".
ToggleItem ::= ":ToggleItem" "(" Target ItemIndex ")".
TransitionTo ::= ":TransitionTo" "(" Target [ConnectionTag]
    [TransitionEffect] ")".

```

Unload ::= ":Unload" "(" Target ")".
UnlockScreen ::= ":UnlockScreen" "(" Target ")".

AbsoluteTime ::= ":AbsoluteTime" GenericBoolean.
Address ::= GenericOctetString.
Answer ::= ObjectReference.
AppendValue ::= GenericOctetString.
ArcAngle ::= GenericInteger.
AvailabilityStatusVar ::= ObjectReference.
CallSucceeded ::= ObjectReference.
CellIndex ::= GenericInteger.
CloneRefVar ::= ObjectReference.
ComparisonValue ::= NewGenericBoolean | NewGenericInteger
 | NewGenericOctetString
 | NewGenericObjectReference
 | NewGenericContentReference.

ConnectionTag ::= ":ConnectionTag" GenericInteger.
Denominator ::= GenericInteger.
EllipseHeight ::= GenericInteger.
EllipseWidth ::= GenericInteger.
EmulatedEventData ::= NewGenericBoolean | NewGenericInteger
 | NewGenericOctetString.
EmulatedEventSource ::= GenericObjectReference .
EmulatedEventType ::= EventTypeEnum.
EntryPointVar ::= ObjectReference.
ForkSucceeded ::= ObjectReference.
Feature ::= GenericOctetString.
FillColourVar ::= ObjectReference.
FirstItemVar ::= ObjectReference.
HighlightStatusVar ::= ObjectReference.
Index ::= GenericInteger.
InFileName ::= GenericOctetString.
InteractionStatusVar ::= ObjectReference.
InVariables ::= "(" ObjectReference+ ")".
ItemIndex ::= GenericInteger.
ItemRefVar ::= ObjectReference.
ItemStatusVar ::= ObjectReference.
ItemsToScroll ::= GenericInteger.
LabelVar ::= ObjectReference.
LastAnchorFiredVar ::= ObjectReference.
LineColourVar ::= ObjectReference.
LineStyleVar ::= ObjectReference.
LineWidthVar ::= ObjectReference.
MovementIdentifier ::= GenericInteger.
NbOfSteps ::= GenericInteger.
NewAbsoluteColour ::= ":NewAbsoluteColour" GenericOctetString.
NewCachePriority ::= GenericInteger.
NewColour ::= NewColourIndex | NewAbsoluteColour.
NewColourIndex ::= ":NewColourIndex" GenericInteger.
NewContent ::= NewIncludedContent | NewReferencedContent.
NewContentCachePriority ::= ":NewCCPriority" GenericInteger.
NewCounterEndPosition ::= GenericInteger.
NewCounterPosition ::= GenericInteger.
NewContentSize ::= ":NewContentSize" GenericInteger.
NewCounterValue ::= GenericInteger.
NewCursorShape ::= GenericObjectReference.
NewEntryPoint ::= GenericInteger.
NewFirstItem ::= GenericInteger.
NewFont ::= NewFontName | NewFontReference.
NewFontName ::= NewGenericOctetString.
NewFontReference ::= NewGenericObjectReference.

NewGenericBoolean ::= ":GBoolean" GenericBoolean.
NewGenericInteger ::= ":GInteger" GenericInteger.
NewGenericOctetString ::= ":GOctetString" GenericOctetString.
NewGenericObjectReference ::= ":GObjectRef" GenericObjectReference.
NewGenericContentReference ::= ":GContentRef" GenericContentReference.
NewHighlightStatus ::= GenericBoolean.
NewIncludedContent ::= GenericOctetString.
NewInteractionStatus ::= GenericBoolean.
NewLabel ::= GenericOctetString.
NewLineStyle ::= GenericInteger.
NewLineWidth ::= GenericInteger.
NewOverwriteMode ::= GenericBoolean.
NewPaletteRef ::= GenericObjectReference.
NewPortion ::= GenericInteger.
NewReferencedContent ::= ":NewRefContent" "(" GenericContentReference
[NewContentSize]
[NewContentCachePriority] ")".
NewSliderValue ::= GenericInteger.
NewSpeed ::= Rational.
NewTransparency ::= GenericInteger.
NewVariableValue ::= NewGenericInteger | NewGenericBoolean
| NewGenericOctetString
| NewGenericObjectReference
| NewGenericContentReference.
NewVolume ::= GenericInteger.
NewXPosition ::= GenericInteger.
NewYPosition ::= GenericInteger.
Numerator ::= GenericInteger.
OpenSucceeded ::= ObjectReference.
Operator ::= GenericInteger.
OutFileName ::= GenericOctetString.
OutVariables ::= "(" ObjectReference+ ")".
OverwriteModeVar ::= ObjectReference.
Parameter ::= NewGenericBoolean | NewGenericInteger
| NewGenericOctetString
| NewGenericObjectReference
| NewGenericContentReference.
Parameters ::= Parameter+.
Point ::= "(" X Y ")".
PointList ::= "(" Point+ ")".
PortionVar ::= ObjectReference.
Protocol ::= GenericOctetString.
Rational ::= Numerator [Denominator].
ReadSucceeded ::= ObjectReference.
ReferenceVisible ::= GenericObjectReference.
RunningStatusVar ::= ObjectReference.
SelectionStatusVar ::= ObjectReference.
SizeVar ::= ObjectReference.
SliderValueVar ::= ObjectReference.
StartAngle ::= GenericInteger.
StoreSucceeded ::= ObjectReference.
Target ::= GenericObjectReference.
TextContentVar ::= ObjectReference.
TextDataVar ::= ObjectReference.
TimerID ::= GenericInteger.
TimerValue ::= GenericInteger.
TokenPositionVar ::= ObjectReference.
TransitionEffect ::= GenericInteger.
TriggerIdentifier ::= GenericInteger.
Value ::= GenericInteger.
VisibleReference ::= GenericObjectReference.

VolumeVar ::= **ObjectReference**.
X ::= **GenericInteger**.
X1 ::= **GenericInteger**.
X2 ::= **GenericInteger**.
XBoxSizeVar ::= **ObjectReference**.
XCursor ::= **GenericInteger**.
XNewBoxSize ::= **GenericInteger**.
XOut ::= **ObjectReference**.
XPositionVar ::= **ObjectReference**.
XScale ::= **GenericInteger**.
Y ::= **GenericInteger**.
Y1 ::= **GenericInteger**.
Y2 ::= **GenericInteger**.
YBoxSizeVar ::= **ObjectReference**.
YCursor ::= **GenericInteger**.
YNewBoxSize ::= **GenericInteger**.
YOut ::= **ObjectReference**.
YPositionVar ::= **ObjectReference**.
YScale ::= **GenericInteger**.

B.4.43 Referencing Objects, Contents, Values, Colour and Position (Objetos de Referencia, Contenido, Valores, Color y Posición)

ObjectReference ::= **ExternalReference** | **InternalReference**.
ExternalReference ::= "(" **GroupIdentifier** **ObjectNumber** ")."
InternalReference ::= **ObjectNumber**.
GroupIdentifier ::= **OctetString**.
ObjectNumber ::= **INTEGER**.

ContentReference ::= **OctetString**.

GenericObjectReference ::= **DirectReference** | **IndirectReference**.
DirectReference ::= **ObjectReference**.
IndirectReference ::= ":"**IndirectRef**" **ObjectReference**.

GenericContentReference ::= **ContentReference** | **IndirectReference**.

GenericInteger ::= **INTEGER** | **IndirectReference**.

GenericBoolean ::= **BOOLEAN** | **IndirectReference**.

GenericOctetString ::= **OctetString** | **IndirectReference**.

OctetString ::= **STRING** | **QPRINTABLE** | **BASE64**.

Colour ::= **ColourIndex** | **AbsoluteColour**.
ColourIndex ::= **INTEGER**.
AbsoluteColour ::= **OctetString**.

XYPosition ::= **XPosition** **YPosition**.
XPosition ::= **INTEGER**.
YPosition ::= **INTEGER**.

APÉNDICE I

Propiedades de un motor MHEG-5

El motor MHEG-5 puede estar en dos estados principales: reposo o activo. Un motor MHEG-5 en reposo es aquél en el cual ningún objeto Aplicación está activo. En cambio, en un motor MHEG-5 activo, exactamente un objeto Aplicación está activo. El motor pasa del estado activo al estado reposo solamente mediante la ejecución de la acción Cesar; la semántica de esta acción se explica en la cláusula 10. El motor pasa del estado reposo al estado activo:

- 1) como un paso en la ejecución de una acción Lanzamiento o Ramificación permanente; la semántica se explica en la cláusula 10; o
- 2) como resultado de que el motor MHEG-5 en reposo es forzado desde el exterior a activar un objeto Aplicación.

El segundo caso es el tema de esta subcláusula.

Desde el exterior, un motor MHEG-5 es típicamente una aplicación que funciona en algún dispositivo en un entorno de soporte lógico. Cuando esta aplicación es activada por la primera vez, obtendrá información del exterior donde se ha de encontrar el primer objeto Aplicación. Esta información puede venir, por ejemplo, del usuario, de la memoria o de la red, y debe permitir que el motor MHEG-5 se vincule al espacio de nombre del objeto Aplicación y halle el propio objeto Aplicación.

Una vez que el motor MHEG-5 tiene la información, trata de extraer ese objeto Aplicación y activarlo. Esto último se hace invocando implícitamente el comportamiento *Activación* del objeto. Para seleccionar también una Escena (sin la cual ninguna entrada de usuario o visualización es posible), el objeto Aplicación debe tener codificada en su Objeto arranque una acción Transición a; esta acción activa la primera Escena de la Aplicación.

APÉNDICE II

Definición de dominios de aplicación

Este apéndice especifica las características de la presente Recomendación que tienen que ser definidas por un dominio de aplicación MHEG-5 específico. La finalidad de este apéndice es proporcionar un conjunto de herramientas para crear un caso de un motor MHEG-5 y ejemplos de valores de tablas.

II.1 Formato de intercambio de objetos

Como un ejemplo, se especifica aquí que el dominio de aplicación elige la notación ASN.1 definida en el anexo A para el formato de intercambio.

Formato de intercambio: ASN.1.

II.2 Conjunto de clases

El dominio de aplicación define un conjunto de clases obligatorias, como son las siguientes:

Acción, Aplicación, Audio, Diagrama de bits, Variable booleana, Variable referencia de contenido, Campo de entrada, Punto focal, Hipertexto, Variable entera, Programa intercambiado, Enlace, Grupo de listas, Variable referencia objeto, Variable cadena de octetos, Botón de pulsación, Rectángulo, Programa distante, Programa residente, Escena, Deslizador, Tren, Botón de conmutación, Texto, Grupo de testigos, Vídeo.

II.3 Conjunto de características

El dominio de aplicación define un conjunto de características obligatorias y facultativas, como se indica en el ejemplo siguiente:

Característica	Requisito
Acciones auxiliares	Obligatorio
Ocultación	Opcional
Clonación	Obligatorio
Cursor de movimiento libre	Opcional
Escala (Vídeo y Diagrama de bits)	Opcional
Pila de aplicaciones	Opcional
Modos truco	Opcional

II.4 Codificación de datos de contenido

Esta Recomendación es genérica en el sentido de que no especifica exactamente cómo se codifican los datos de contenido (por ejemplo, datos de diagrama de bits). Sin embargo, la presente Recomendación es también específica en el sentido que:

- 1) especifica el tipo de datos de contenidos admitido, y
- 2) proporciona un método para definir, para un dominio de aplicación específico, una lista de las codificaciones reales utilizadas en ese dominio.

El siguiente cuadro ejemplifica cómo un dominio de aplicación podrá especificar los formatos necesarios; los valores enumerados a continuación se proporcionan simplemente como un ejemplo de valores que un dominio de aplicación podría elegir.

Tabla de contenido

Atributo	Valores admisibles
Atributos de tipos de caracteres	Negritas, cursivas, subrayado, doble subrayado
Nombre de tipo de carácter	(No se especifica ninguno)
Color absoluto	RGB16, que es la codificación de gráficos en 16 bits por pixel, asignando 5 bits a los componentes rojo y azul y 6 bits al componente verde
Juego de caracteres	1 – hace referencia al subconjunto ISO/CEI 8859-1 especificado en HTML 2.0
Efecto de transición	(No se especifica ninguno)

Cuadro de codificación

Tipo de contenido	Especificación (Tipos de datos)	Valores de gancho
Formato de codificación de tipos de carácter	(No se especifica ninguno)	
Formato de codificación de paleta	(No se especifica ninguno)	
Formato de codificación de diagrama de bits	Gráficos 2D definidos en DAVIC 1.0	1
	Tramas I vídeo MPEG-2 (ISO/CEI 13818-2)	2
Formato de codificación de texto	El subconjunto de ISO/CEI 8859-1 especificado en HTML 2.0	1
Formato de codificación de campo de entrada	El subconjunto de ISO/CEI 8859-1 especificado en HTML 2.0	1
Formato de codificación de hipertexto	El subconjunto de ISO/CEI 8859-1 especificado en HTML 2.0	1
Formato de codificación de tren	Sistemas MPEG-2 especificados en ISO/CEI 13818-1	1
	Audio MPEG-1 especificado en ISO/CEI 11172-3	2
	AIFF-C	3
	Trenes en tiempo real definidos en DAVIC 1.0	4
Formato de codificación de arte lineal	(No se especifica ninguno)	
Formato de codificación de forma de cursor	(No se especifica ninguno)	
Formato de codificación de programas intercambiados	MHEG-6	1

II.5 Registros de entrada de usuario

Para tener una norma de trabajo, el dominio de aplicación debe especificar uno o más Registros de eventos de entrada. Cada registro tiene un número, que es intercambiado como uno de los parámetros de un objeto escena. El contenido de un Registro de eventos de entrada (que no es intercambiado) es un conjunto de números (que representan Rótulos de eventos de entrada de usuario) y un nombre. Se recomienda que los nombres tengan el mayor contenido semántico posible («Up» es un nombre adecuado; «13» es un nombre inadecuado). Los pares de nombre/número vinculan un Rótulo de evento de entrada de usuario específica con un evento de entrada lógico. Corresponde al implementador del motor vincular el evento de entrada lógico con uno o más eventos de entrada físicos.

El siguiente cuadro es un ejemplo de un conjunto de Rótulos de eventos de entrada:

Registro N.º	Rótulo de evento de entrada de usuario	Nombre	Comentario
1	1	Up (Arriba)	
	2	Down (Abajo)	
	3	Left (Izquierda)	
	4	Right (Derecha)	
	5-14	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, respectivamente	
	15	Select (Seleccionar)	
	16	Exit (Salida)	
	17	Help (Ayuda)	
	18-99	Reservado para especificación futura	
	100-	Específico del fabricante	

II.6 Restricciones semánticas impuestas a las aplicaciones MHEG-5

Un dominio de aplicación MHEG-5 puede restringir sus aplicaciones en determinadas dimensiones. El siguiente cuadro contiene una lista de características indicadas en la presente Recomendación que son facultativas o pueden ser admitidas en mayor o menor grado. En relación con cada uno de estos puntos, un dominio de aplicación MHEG-5 debe tomar una decisión, esbozada en la columna de la derecha.

El siguiente cuadro es un ejemplo de estas decisiones:

Característica	Restricción
Cursor de movimiento libre	Opcional
Pila de aplicaciones	Opcional
Escala	Opcional
Sistema de coordenadas de escena(X,Y)	Se admiten los siguientes sistemas de coordenadas: 720 × 576, 704 × 576, 640 × 576, 544 × 576, 540 × 576, 480 × 576, 352 × 576, 352 × 288, 720 × 480, 704 × 480, 640 × 480, 544 × 480, 480 × 480, 352 × 480 y 352 × 240
Formato de imagen de escena(W,H)	Se admiten los siguientes formatos: 1/1, 4/3 y 16/9
Conexiones auxiliares	Opcional
Modos truco	Opcional
Múltiples trenes de gráficos RT(N)	Un gráfico RT activo a la vez
Múltiples trenes audio(N)	Un tren de audio de tipo <i>stream</i> y uno de tipo <i>memory</i> a la vez
Múltiples trenes vídeo(N)	Se presenta un tren vídeo a la vez
Visibles superpuestos(N)	Ninguna restricción
Clonación	Obligatoria

II.7 Evento motor

Este dominio de aplicación no reserva un valor particular al Evento motor, que se deja al diseñador de la aplicación.

II.8 Obtención de soporte de motor

Las cadenas admisibles para la acción Obtener soporte de motor son las definidas en la presente Recomendación. Además de estas cadenas, un dominio de aplicación dado puede definir otras, en cuyo caso, el dominio de aplicación debe definirlas con precisión en una tabla similar a la anterior. Este ejemplo no define ninguna cadena adicional.

II.9 Correspondencia de protocolos e interacción externa

El siguiente cuadro proporciona ejemplos para definir la correspondencia con el entorno externo.

Entidad MHEG-5	Correspondencia necesaria	Semántica de las estructuras MHEG-5 que necesita especificación
Abrir conexión, Cerrar conexión	Correspondencia con gestión de conexión	<ul style="list-style-type: none"> En Abrir conexión: <ul style="list-style-type: none"> Protocolo: una de las dos cadenas siguientes: "PSTN" o "ISDN" Dirección: E.164 NSAP
Objetos programa distante	Correspondencia con protocolo de llamada a programa distante en el dominio de aplicación	<ul style="list-style-type: none"> En Llamada y Ramificación temporal: <ul style="list-style-type: none"> Nombre Parámetro Rótulo de conexión de programa
Espacio de nombre de aplicación	Correspondencia con espacio de nombre del dominio de aplicación	<p>Concordará con DSM-CC</p> <ul style="list-style-type: none"> Referencia de objeto Referencia de contenido <p>Concordará con DSM-CC</p>
Espacio de nombre de aplicación cuando la acción Transición a utiliza el parámetro rótulo de conexión	Correspondencia con el nombre de espacio del dominio de aplicación	<ul style="list-style-type: none"> Referencia de objeto Referencia de contenido <p>Concordará con DSM-CC</p>
Espacio de nombre de almacenamiento persistente	Correspondencia con el nombre de espacio del almacenamiento persistente	<ul style="list-style-type: none"> En Almacenamiento persistente y Lectura persistente: <ul style="list-style-type: none"> Nombre de fichero de entrada, nombre de fichero de salida <p>Concordará con DSM-CC</p>
Acciones de tren	Correspondencia con la interfaz de tren del dominio de aplicación	<ul style="list-style-type: none"> En Tren <ul style="list-style-type: none"> Velocidad Posición de contador
Eventos de tren	Correspondencia con estados de tren y eventos de tren en el dominio de aplicación	<ul style="list-style-type: none"> En Tren <ul style="list-style-type: none"> Tren en ejecución, tren detenido (correspondencia con la máquina de estado de trenes del dominio de aplicación) Posición de contador Rótulo de evento de tren

SERIES DE RECOMENDACIONES DEL UIT-T

Serie A	Organización del trabajo del UIT-T
Serie B	Medios de expresión: definiciones, símbolos, clasificación
Serie C	Estadísticas generales de telecomunicaciones
Serie D	Principios generales de tarificación
Serie E	Explotación general de la red, servicio telefónico, explotación del servicio y factores humanos
Serie F	Servicios de telecomunicación no telefónicos
Serie G	Sistemas y medios de transmisión, sistemas y redes digitales
Serie H	Sistemas audiovisuales y multimedios
Serie I	Red digital de servicios integrados
Serie J	Transmisiones de señales radiofónicas, de televisión y de otras señales multimedios
Serie K	Protección contra las interferencias
Serie L	Construcción, instalación y protección de los cables y otros elementos de planta exterior
Serie M	RGT y mantenimiento de redes: sistemas de transmisión, circuitos telefónicos, telegrafía, facsímil y circuitos arrendados internacionales
Serie N	Mantenimiento: circuitos internacionales para transmisiones radiofónicas y de televisión
Serie O	Especificaciones de los aparatos de medida
Serie P	Calidad de transmisión telefónica, instalaciones telefónicas y redes locales
Serie Q	Conmutación y señalización
Serie R	Transmisión telegráfica
Serie S	Equipos terminales para servicios de telegrafía
Serie T	Terminales para servicios de telemática
Serie U	Conmutación telegráfica
Serie V	Comunicación de datos por la red telefónica
Serie X	Redes de datos y comunicación entre sistemas abiertos
Serie Y	Infraestructura mundial de la información
Serie Z	Lenguajes de programación