Digital Terrestrial Television MHEG-5 Specification

Revision: 1.0
Revised: 28 February 1998 20:16

1 Status of this document

This document is the substantially complete specification for the MHEG-5 system to be implemented in the first generation of Digital Terrestrial TV receivers to be deployed the UK.

The technical details represent a widely supported *working assumption*. However, changes may be introduced as issues are uncovered and where clarification is required.

The following issues of detail are under review at the time of publication:

- The details of the definition of the colour palette, see "The Colour Palette" on page 30.
- The details of the method for coding text mark-up, "Text Objects" on page 54.
- The details of the character table to be used, "Character repertoire" on page 58.
- The details for how a receiver identifies which application to launch (See "Application Identification and Boot" on page 94) (topic under discussion within the DVB).

In addition certain topics are recognised as projects for future work:

- The Reference Decoder Model see page 70.
- The broadcaster MHEG-5 Authoring Guidelines see page 107.

2 The User Experience

2.1 Introduction

This section looks at the behaviour seen by the user. It is provided to give context to the receiver specifications. However, much of the behaviour shown here results from functionality coded into the broadcast application, rather than the receiver.

This broadcaster controlled behaviour should be seen as a concept model for how services may appear. As broadcasters develop their concepts for services, and gain experience from user feed-back, the detail of the behaviour they implement is likely to evolve.

2.1.1 Visual Appearance

Balance of AV and MHEG

Table 1 illustrates the range of different visual appearances the viewer might experience. Each "screen" shows a different balance between "conventional TV" AV content and information delivered via MHEG.

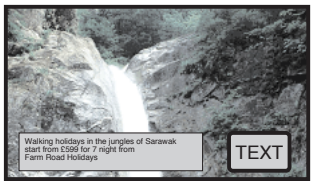
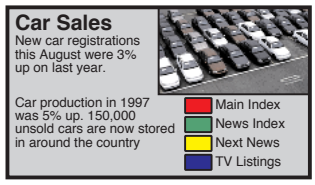
Visual Appearance	Description
	1. Conventional TV
	2. TV with visual prompt of available information
	3. TV with information overlaid
	4. Information with video or picture inset
	5. Just information

Table 1. Typical range of programme types perceived by viewers

Time and Space

The user may see a change in appearance either when they change channel or as a service changes through time.

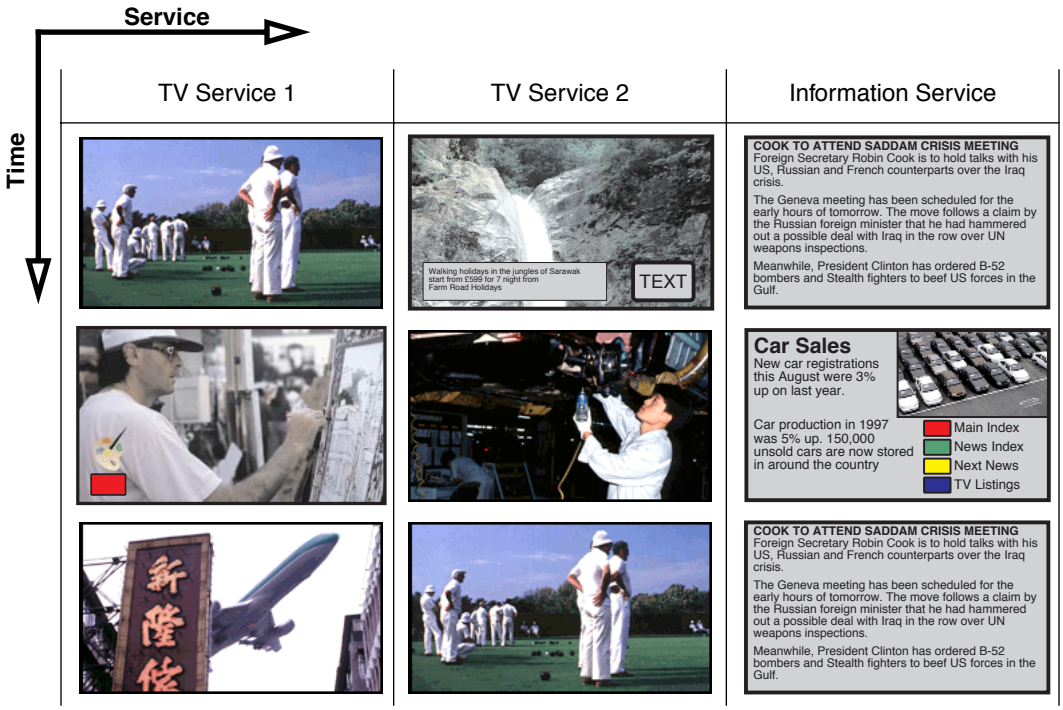


Figure 1. What might be seen across channels and through time

2.2 User Navigation

Channel Change

The user can use any of the navigation methods to change channel:

- "Surfing" using "Programme up" & "Programme down"
- Direct selection by pressing a favourite channel button
- Selection from options within the "Info" or "Guide" screens

These apply equally to TV and Information services. There may also be cases where the user changes channel from within an information service.

When a service has been selected, regardless of the service type, the user can again change channel in the usual way.

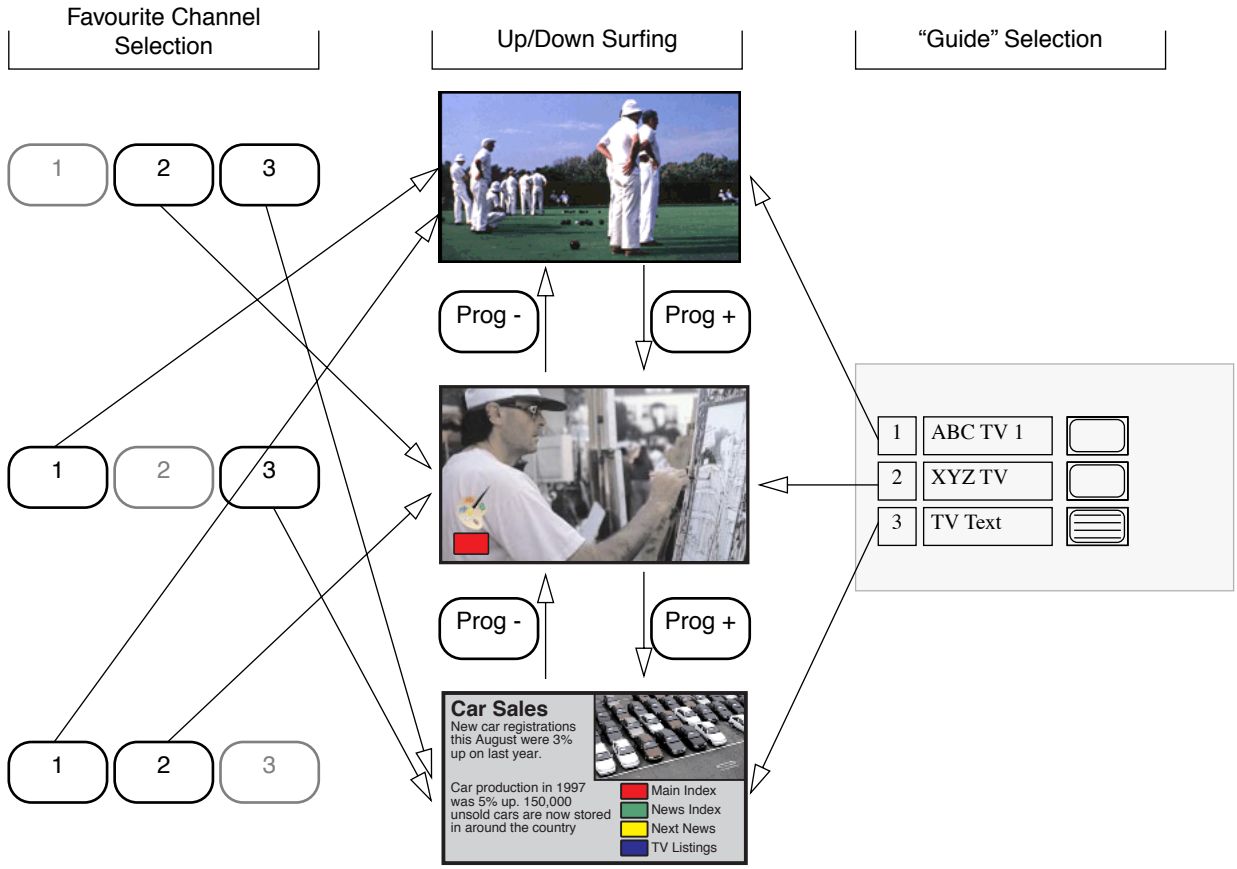


Figure 2. Changing channels

The “Text” Button

The effect of the “Text” button may vary slightly from programme to programme.

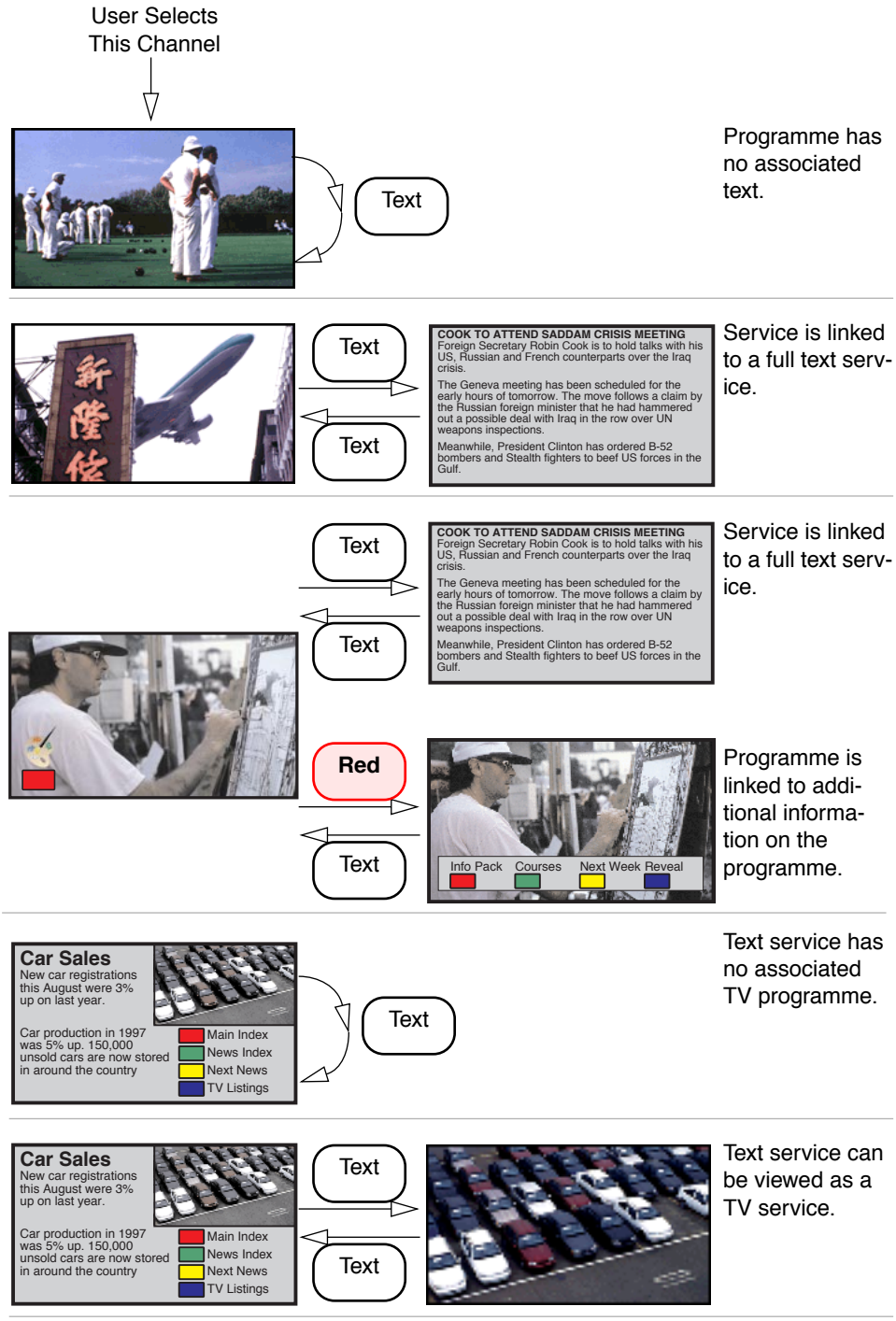


Figure 3. Using the text button

2.3 Remote Control Functions

Figure 4 illustrates the functions of the receiver remote control. This is provided to help explain the "function groups". The physical appearance of the remote control and the methods of interaction delivering the required functions may be quite different from the one illustrated.

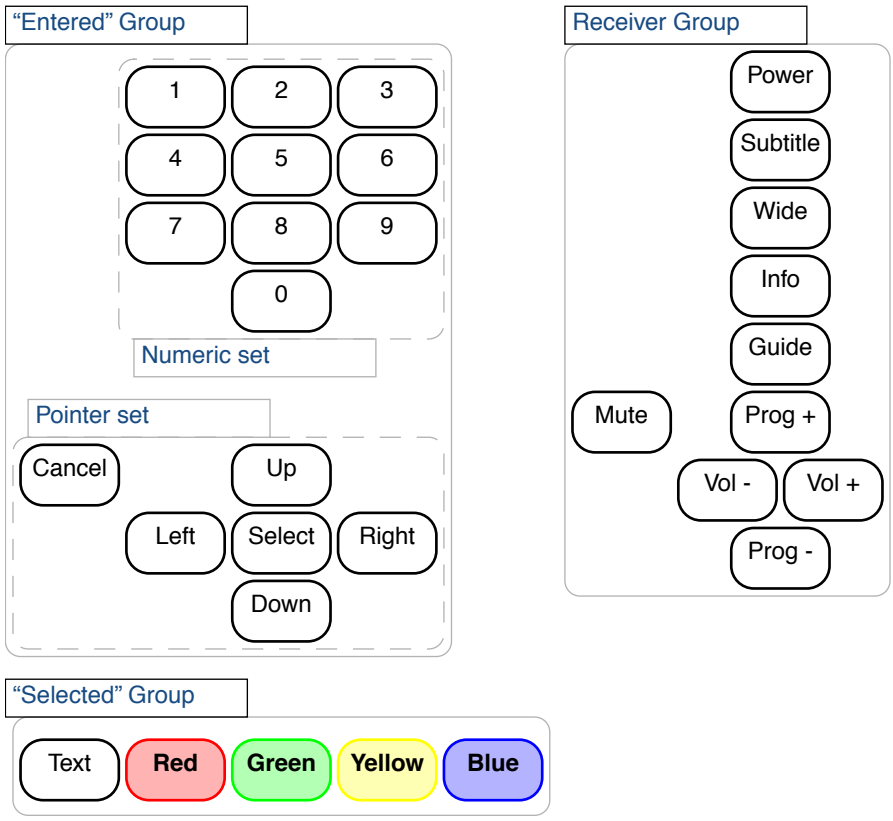


Figure 4. Remote Control Function Groups

Receiver Group

These provide the most frequently used receiver control functions.

They have the same effect whether the viewer is watching a TV service, an MHEG service or using other receiver screens. For example, the "Program up" & "Program down" always have the effect of changing channel.

All other receiver specific functions (e.g. to invoke the receiver set-up screens) are also within this group.

"Entered" Group

This group is available for use by receiver specific functions while the viewer is watching TV, surfing between channels or using receiver interaction screens. After the viewer *enters* a broadcast application these functions become exclusively used by the application.

Numeric set

The *numeric set* can be used by the application for navigation (e.g. selecting from list of options labelled 1-9) or for entering numbers (e.g. to enter a number in a maths quiz).

Pointer set

The *pointer set* provide "mouse" like ability to "touch" and "manipulate" things such as buttons and sliders.

"Selected" Group

When the viewer is watching TV these keys have no effect.

After *selecting* an MHEG service these keys become active allowing the user to select between a menu of options. These keys may also be used for navigating around receiver originated screens such as the "Info" & "Guide" screens.

2.4 Channel Selections within an Information Service

MHEG services can provide facilities to change channel. For example, they may provide listings of current programmes and the ability to select a programme.

The effect of changing channel from an MHEG application is exactly the same as if the user selected a channel from the receiver's "Guide".

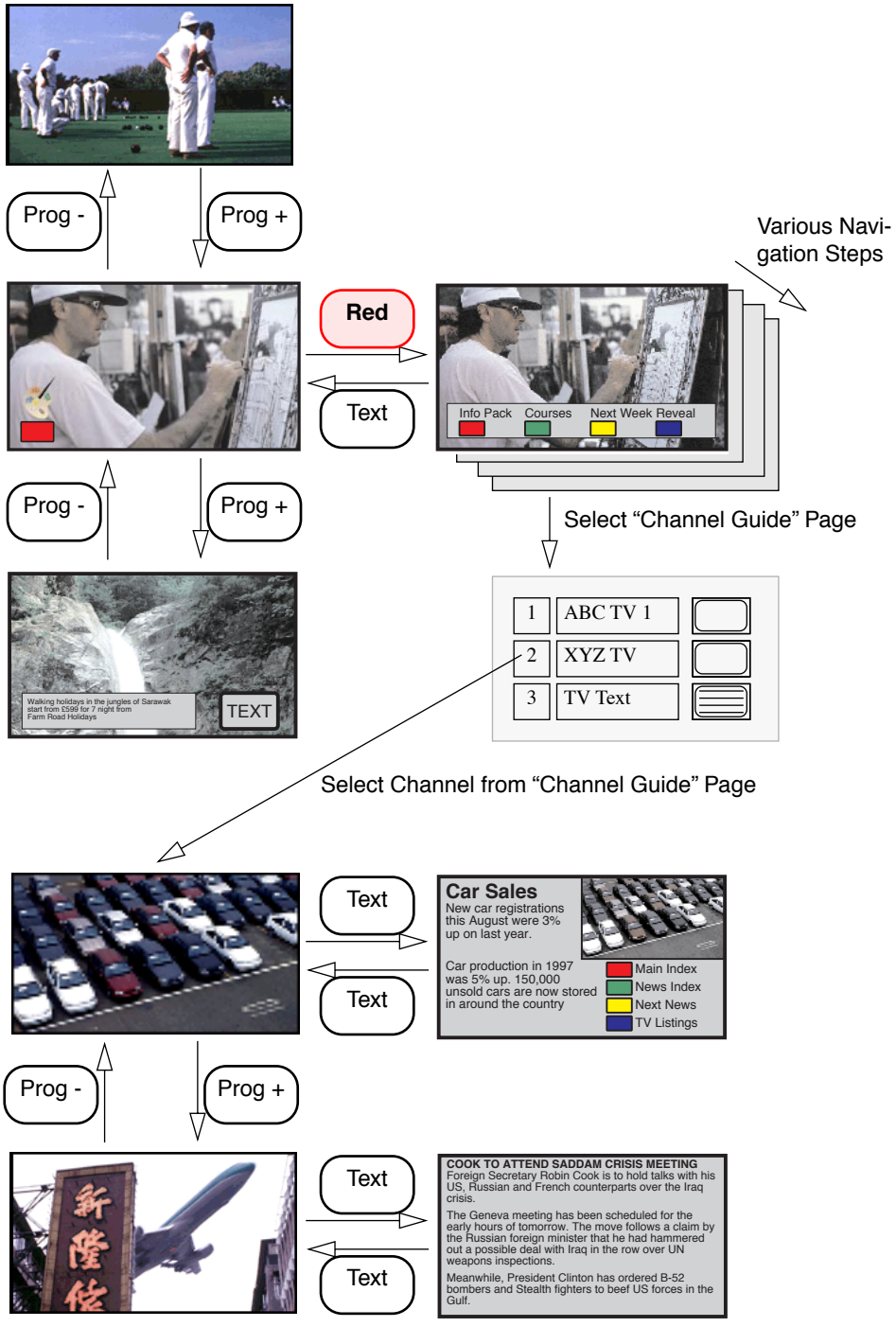


Figure 5. Using a TV listings page to change channel

2.5 Use of Video within an Information Service

MHEG services can use video inset into pages. For example, this could provide a video "preview" of the services on a multiplex. This might result in a channel change, but if the viewer "backs out" they will return to the service they started from.

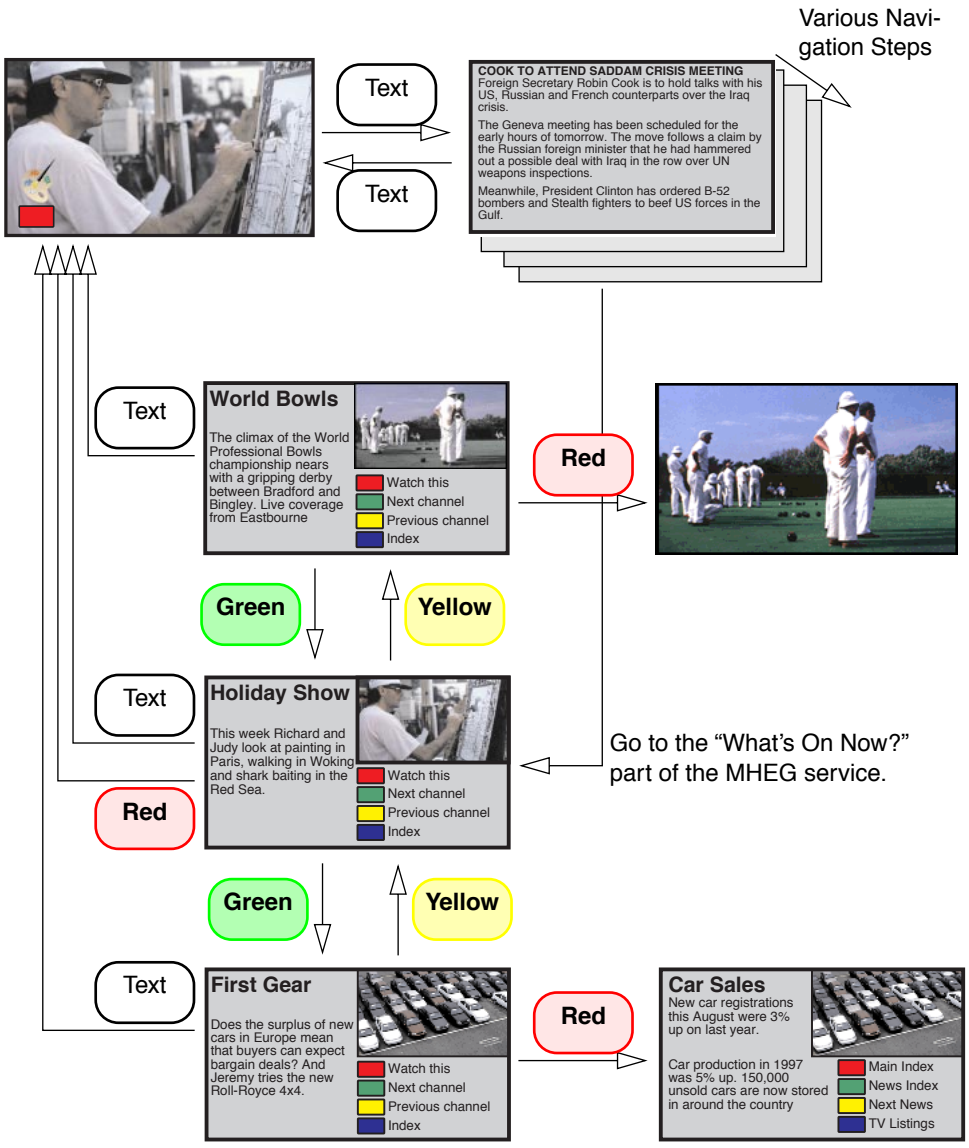


Figure 6. A possible "What's on now?" TV browser

3 MHEG-5 Engine Profile

This section provides the detail specification of the MHEG engine required in first generation digital TV receivers. This specification is an "application domain" in the terms set out in Annex D of ISO/IEC 13522-5.

This is "application domain" is referred to as "*UKEngineProfile1*"

3.1 Basic Specification

The engine shall be compliant with the ISO/IEC 13522-5 (MHEG-5) specification for a Hypermedia presentation engine using the "application domain" definition provided by this document.

The ISO/IEC 13522-5 specification should be considered in conjunction with the work of the MHEG-5 Maintenance Task Force (MTF) published at:

http://www.fokus.gmd.de/ovma/mug/mtf/entry.htm

3.2 Object interchange format

The ASN.1 notation defined in Annex A of ISO/IEC 13522-5 shall be used as the application interchange format. The encoding of the MHEG-5 objects from this ASN.1 syntax shall make use of the *Distinguished Encoding Rules* (DER) defined in ITU-T X.690.

3.3 Set of Classes

Table 2 identifies the classes that a receiver implementing the *UKEngineProfile1* shall implement.

Class	Abstract/ Concrete[a]	Required (Y/N[b])	Notes
Root	A	Y	
Group	A	Y	
Application	C	Y	
Scene	C	Y	
Ingredient	A	Y	
Link	C	Y	
Program	A	Y	
ResidentProgram	C	Y	See "Resident Programs" on page 26
RemoteProgram	C	*N*	
InterchangedProgram	C	*N*	
Palette	C	*N*	
Font	C	*N*	See "Text & Hypertext" on page 45
CursorShape	C	*N*	

Table 2. Classes supported by this engine profile (Sheet 1 of 2)

Class	Abstract/ Concrete[a]	Required (Y/N[b])	Notes
Variable	A	Y	See Section 3.11 on page 28
BooleanVariable	C	Y	
IntegerVariable	C	Y	
OctetStringVariable	C	Y	
ObjectRefVariable	C	Y	
ContentRefVariable	C	Y	
Presentable	A	Y	
TokenManager	A	Y	
TokenGroup	C	Y	
ListGroup	C	Y	
Visible	A	Y	
Bitmap	C	Y	See “BitmapObjects” on page 24 & “MPEG stills” on page 43.
LineArt	A	Y	See “LineArt & DynamicLineArt” on page 42.
Rectangle	C	Y	
DynamicLineArt	C	Y	See “LineArt & DynamicLineArt” on page 42.
Text	C	Y	See “Text Objects” on page 54.
Stream	C	Y	
Audio	C	Y	See “Audio Stream Decoders” on page 22 & “Stream “memory” formats” on page 24.
Video	C	Y	
RTGraphics	C	Y	
Interactible	A	Y	
Slider	C	Y	
EntryField	C	Y	See “Entry Fields” on page 57.
HyperText	C	Y	See “HyperText Objects” on page 55.
Button	A	Y	
HotSpot	C	Y	
PushButton	C	Y	
SwitchButton	C	Y	
Action	C	Y	

Table 2. Classes supported by this engine profile (Sheet 2 of 2)

a] Abstract classes are not “interchanged” (i.e. not directly used) in *UKEngineProfile1* MHEG applications

b] ‘Y’ = yes, i.e. receivers implementing *UKEngineProfile1* shall support these classes. ‘N’ = no, i.e. receivers implementing *UKEngineProfile1* are not required to support these classes. Also, classes marked ‘N’ may not be completely defined in the UK context at this time.

3.4 Set of Features

All receivers shall implement all of the effects of MHEG-5 actions and the internal behaviours of MHEG-5 classes required under "Set of Classes" on page 19. Table 3 identifies the *UKEngineProfile1* requirement with regard to features defined as "optional" within ISO/IEC 13522-5.

Feature	Required (Y/N[a])	Notes
Caching	Y	See "Reference Decoder Model" on page 70.
Ancillary connections	N	See "Not Required Features" on page 28.
Cloning	Y	
Free-moving cursor	N	See "Not Required Features" on page 28.
Bitmap scaling	N	Limited scaling is required for I frame bitmaps, but not for PNGs. See "Scaling" on page 24.
Video scaling	Y	See "Video Scaling Reference Model" on page 68.
Stacking of Applications	Y	See "Application Stacking" on page 71.
Trick Mode	N	Not applicable for any currently defined stream-items' content type. See "Not Required Features" on page 28.

Table 3. Requirements for MHEG-5 "Optional" Features

a] 'Y' = yes, i.e. receivers implementing *UKEngineProfile1* shall support these features.

3.4.1 GetEngineSupport "feature" strings

Table 4 identifies the GetEngineSupport feature strings that receivers implementing *UKEngineProfile1* shall support.

String	Constraint
AncillaryConnections	Shall return "false". See 3.13 on page 28.
ApplicationStacking	Shall return "true". See "Application Stacking" on page 71.
Cloning	Shall return "true".
Freemoving Cursor	Shall return "false". See 3.13 on page 28.
MultipleAudioStreams(N)	Shall return "true" for N≤1. Support for higher values of N is optional. See "Audio Stream Decoders" on page 22.
MultipleRTGraphicsStreams(N)	Shall return "true" for N≤1. This refers to the number of concurrent DVB Subtitle [2] streams that the engine can decode.
MultipleVideoStreams(N)	Shall return "true" for N≤1.
OverlappingVisibles(N)	Shall return "true" for all values of N. See "DVB Subtitle decoder performance" on page 68 & "Overlapping Visibles" on page 41.
Scaling	Shall return "false". See "Scaling" on page 22 & "Video Scaling" on page 68.
SceneAspectRation(W, H)	Shall return "true" for (W, H) is (4,3) or (16,9). See "Scene Aspect Ratio" on page 111.
SceneCoordinateSystem(X,Y)	Shall return "true" for (X,Y) is (720,576). See "The Graphics Plane" on page 29.
TrickModes	Shall return "false". See 3.13 on page 28.

Table 4. UKEngineProfile1 GetEngineSupport behaviour (Sheet 1 of 2)

String	Constraint
VideoScaling(X,Y)	Shall return "true" for (X,Y) is (720,576) and (360, 288). See "VideoScaling(X,Y)" on page 22 & "Video Scaling" on page 68.
UKEngineProfile(1)	Shall return "true" when N=1 to indicate that the engine at least supports the minimum requirements of *UKEngineProfile1*.

Table 4. UKEngineProfile1 GetEngineSupport behaviour (Sheet 2 of 2)

Unrecognised requests will always result in GetEngineSupport returning false. This implies that future MHEG engine implementations supporting more than these minimum features will recognise additional strings. These strings will have to be agreed by the DMux group so that consistency is maintained.

3.4.1.1 Audio Stream Decoders

Two types of audio decoder are considered in the context of the *UKEngineProfile1* MHEG engine:

1. Decode MPEG Audio [16] data "live" from a stream as it is broadcast (e.g. the sound track for a TV programme).
2. Decode an MPEG Audio [16] object from memory (See "Stream "memory" formats" on page 24).

UKEngineProfile1 receivers shall provide one MPEG Audio [16] decoder which can decode data either from a stream OR from an object in memory.

Facility for decoding AIFF-C [24] audio data is not part of *UKEngineProfile1*. The meaning of MultipleAudioStreams(N) for N>1 has not yet been defined in the UK context.

3.4.1.2 Scaling

GetEngineSupport with the feature string "Scaling" shall return "false" in *UKEngineProfile1* to avoid confusion with the DAVIC 1.3 Part 09 semantic where this engine string applies to both bitmaps and video.

VideoScaling(X,Y)

The GetEngineSupport with the feature string "VideoScaling(X,Y)" can be used to determine the video scaling modes supported by an engine. *UKEngineProfile1* receivers shall support the following 2 options:

- Full screen video
- Quarter screen video

GetEngineSupport with the feature string "VideoScaling(X,Y)" indicates this by returning "true" when (X,Y) is (720,576) or (360,288).

See "Video Scaling" on page 68.

3.5 Content data encoding

Table 5 identifies the minimum set of coding of attributes that shall be supported by the engine.

Attribute	Permissible values
FontAttributes	See "FontAttributes" on page 47
FontName	See "Invoking the font" on page 47
AbsoluteColour	See "Direct/Absolute colours" on page 40

Table 5. Content table (Sheet 1 of 2)

Attribute	Permissible values
CharacterSet	See “Character Set” on page 48
TransitionEffect	(None specified)

Table 5. Content table (Sheet 2 of 2)

Table 6 identifies the minimum set of data types that shall be supported by the engine for each type of content. It also identifies the content hook values for each data type.

Type of content	Specification (Data Types)	Hook values
Font	(None specified) See “Downloading” on page 46	
Palette	(None specified)	
Bitmap	reserved	1
	MPEG-2 Intra frame [15] See “MPEG stills” on page 43	2
	reserved	3
	PNG bitmap	4
Text	See “Text mark-up” on page 54.	1
EntryField	See “Character encoding” on page 45 See “Entry Fields” on page 57.	1
HyperText	See “HyperText Objects” on page 55	1
Stream	See Table 7.	
LineArt	(None specified)	
CursorShape	(None specified)	
InterchangedProgram	(None specified)	

Table 6. Encoding Table

Stream Component	Content-hook	
	1[a]	2[b]
MPEG-1 video (ISO/IEC 11172-2) Or MPEG-2 video (ISO/IEC 13818-2)	x	
MPEG-1 audio (ISO/IEC 11172-2)	x	x
DVB subtitling (ETS 300 743)	x	

Table 7. Content hooks for Streams

a] The stream “Storage” attribute shall be “stream”

b] The stream “Storage” attribute may be “stream” or “memory”

3.5.1 Use of negative hook values

Negative hook values are reserved for use by receiver manufacturers to signal manufacturer specific encoding formats. These shall never be signalled by a broadcaster and may only be used by the manufacturer for local applications.

3.5.2 BitmapObjects

Scaling

Scaling (the ScaleBitmap action) shall be supported for bitmap objects with MPEG I-frame content. See "MPEG stills" on page 43. Scaling shall not be supported for bitmap objects using PNG bitmaps

Tiling

Support for tiling is only required for PNG bitmaps.

Transparency

Transparency can be encoded within the PNG bitmaps. Support for object level transparency is not required for any bitmap type. See "Transparency & Overlapping Visibles" on page 41.

3.5.3 Stream "memory" formats

In this profile the only StreamComponent that accepts a Storage specification of memory is Audio. Video and RTGraphics can only be played from Stream.

Audio

Each "file" of audio content is an OctetString carrying Audio elementary stream data. Each "file" delivers an integer number of audio access units and the first byte of each file is the first byte of an audio access unit.

3.6 UserInput Registers

Table 8 defines the two input event registers (3 & 4) that shall be supported by receivers implementing *UKEngineProfile1*. The DAVIC defined registers 1 & 2 are shown for information only. See "Remote Control Functions" on page 15.

UserInput EventTag	Function Name	Register Number			
		1	2	3[a]	4[b]
1	Up	✓	✓		✓
2	Down	✓	✓		✓
3	Left	✓	✓		✓
4	Right	✓	✓		✓
5-14	0, 1, 2, 3, 4, 5, 6, 7, 8, 9 respectively	✓			✓
15	Select	✓	✓		✓
16	Cancel / Exit	✓	✓		✓
17	Help	✓	✓		
18-99	Reserved for future DAVIC specification				
100	Red			✓	✓
101	Green			✓	✓
102	Yellow			✓	✓
103	Blue			✓	✓
104	Text			✓	✓
105-128	Reserved for future UK specification				
≥129	Vendor specific				

Table 8. InputEvent Registers

a] The "Selected" Group

b] The "Entered" Group

3.7 Semantic constraints on MHEG-5 applications

As implied by the capabilities of the engine in "GetEngineSupport "feature" strings" on page 21.

See also See "Application impact on stream decoder specification" on page 68.

3.8 EngineEvent

Table 9 lists the minimum set of engine events that the engine is required to support.

EventTag		Notes
Name	**Value**	
	< 0	Manufacturer specific
Reserved	0	Reserved
QuitApp	1	See "Killing Applications" on page 28
ObjectRefError	2	See "Checking References" on page 27
ObjectRefInvalid	3	
ObjectRefOK	4	
ContentRefError	5	
ContentRefInvalid	6	
ContentRefOK	7	
TextFunction	8	Generated when the user input function "Text" happens. See "Remote Control Functions" on page 15.
Reserved	> 8	Reserved

Table 9. Required Engine Events

3.9 Protocol mapping and external interaction

MHEG-5 entity	Mapping needed	Semantics
OpenConnection, CloseConnection	Mapping to connection management	Not required in *UKEngineProfile1*. See "Not Required Features" on page 28.
RemoteProgram objects	Mapping to RemoteProgram call protocol in the application domain	
Application name space in case a TransitionTo action uses the ConnectionTag parameter	Mapping to the name space of the application domain	
Persistent storage name space	Mapping to the name space of the persistent storage	See "Persistent storage" on page 69.
Stream actions	Mapping to the stream interface of the application domain	See "Stream Events and Normal Play Time Mapping" on page 101
Stream events	Mapping to stream states and stream events in the application domain	

Table 10. Protocol Mapping

3.10 Resident Programs

Table 11 lists the ResidentPrograms that a *UKEngineProfile1* receiver shall implement.

Program	Short Name	Notes
GetCurrentDate	GCD	As defined by DAVIC 1.3 Part 09 [27]
FormatDate	FDa	
GetDayOfWeek	GDW	
Random	Rnd	
GetStringLength	GSL	
GetSubString	GSS	
CompareString	CSt	
SearchSubString	SSS	
SearchAndExtractSubString	SES	
CastToContentRef[a]	CTC	
CastToObjectRef[a]	CTO	
SI_GetServiceIndex	GSI	As defined by DAVIC 14B94R10 [28] and augmented by "Additional semantics" on page 27 (and reproduced below).
SI_TuneIndex	TIn	
CheckContentRef	CCR	See "CheckContentRef" on page 27
CheckObjectRef	COR	See "CheckObjectRef" on page 27

Table 11. MUKEngineProfile1 Resident Programs

a] See "Type conversion" on page 46

DAVIC definition

Several of the functions in Table 11 are defined by DAVIC [27], others are under consideration. Where DAVIC [27] develops suitable definitions these may be adopted in place of the definitions provided here.

Short names

The long form of ResidentProgram names is provided for consistency with DAVIC [27]. The short form is provided for greater efficiency. Receivers shall recognise both forms.

3.10.1 Service Selection

3.10.1.1 Extracts from DAVIC documents

The text here is an extract from DAVIC 14B94R10 [28] provided for information. The formal definition of these resident programs remains the DAVIC document *except where noted*.

SI_GetServiceIndex()

Returns the corresponding index value of a service.

Synopsis:

```
SI_GetServiceIndex( serviceReference, serviceIndex)
```

Arguments:

```
input GenericOctetString serviceReference
output IntegerVariable serviceIndex
```

Description:

The Resident program returns the index of the Service in the Service list described by serviceReference. ServiceReference is the string used to define a Stream in the URL format defined in DAVIC 1.3 part 9 paragraph 9.5. **See also "Name Mapping" on page 101.**

SI_TuneIndex() Tunes to the given service.

Synopsis:

```
SI_TuneIndex( serviceIndex)
```

Arguments:

```
    input GenericIntegerserviceIndex,
```

Description:

Tunes/switches to the specified service. This function terminates the MHEG-5 application.

3.10.1.2 Additional semantics

The SI_GetServiceIndex() resident program shall return the serviceIndex -1 for services that are not available to the receiver. In this context a service is "available" if it is running in the current transport stream or is running in a transport stream to which the receiver can normally tune.

3.10.2 Checking References

This set of additional resident programs can be used by applications to determine if objects are available before embarking on a course of action that requires the objects.

3.10.2.1 CheckContentRef

Allows an application to check if an item of content is available.

Synopsis — CheckContentRef(ContentIdentifier)

Arguments — input GenericContentReference ContentIdentifier

output engine event ContentRefError or ContentRefOK (See Table 9 on page 25)

Description — CheckContentRef is "non blocking". The scene continues to operate after CheckContentRef is invoked. An application can act on the engine events ContentRefInvalid and ContentRefOK to "decide" whether to access the content specified by ContentIdentifier.

3.10.2.2 CheckObjectRef

Allows an application to check if an object is available.

Synopsis — CheckObjectRef(ObjectIdentifier)

Arguments — input GenericObjectReference ObjectIdentifier

output engine event ObjectRefError or ObjectRefOK (See Table 9 on page 25)

Description — CheckObjectRef is "non blocking". The scene continues to operate after CheckObjectRef is invoked. An application can act on the engine events ObjectRefInvalid and ObjectRefOK to "decide" whether to access the object specified by ObjectIdentifier.

3.10.2.3 Uncalled for events

The engine events ObjectRefError and ContentRefError shall be raised if an application accesses an object or some content that cannot be provided.

3.11 Limitations on Standard Data-Types

BooleanVariable

The BooleanVariable size is undefined as the implementation of its representation is not significant provided that all possible Boolean values can be represented. However, when modelling storage (e.g. when written to persistent storage) Booleans occupy 1 byte.

type of value	default value	size	min. value	max. value
true or false	False	Unspecified	n/a	n/a

IntegerVariable

type of value	default value	Size	min. value	max. value
signed integer	0	32 bits	-214783648	214783647

OctetString

The OctetString data type is restricted to be not longer than 2147483647 octets long, thus the only practical restriction is that of available memory within the RDM (See “Reference Decoder Model” on page 70).

A null OctetString value shall be encoded as “”.

ObjectNumber

The ObjectNumber data type is restricted in the following way:

type of value	default value	Size	min. value	max. value
signed integer	n/a	32 bits	0[a]	214783647

a] 0 for Group objects and 1 for Ingredient objects

GroupIndentifier

Group indentifiers shall not be more than 64 bytes long. See “Mapping Rules for GroupIdentifier” on page 102.

3.12 Special Engine Behaviour

3.12.1 Killing Applications

The “QuitApp” EngineEvent shall be sent to any executing application in the following cases:

- The service selected by the receiver changes for whatever reason

  For example, the service change may be invoked by the user communicating directly with the receiver’s navigator or might be originated by the application (using “SI_GetServiceIndex()”).

- The engine detects a problem with the application environment

The “QuitApp” EngineEvent gives the application a “grace period” in which to “tidy-up”. If the application does not quit within 500 ms of receiving a “QuitApp” EngineEvent the engine may terminate the application.

3.13 Not Required Features

This constraint is to ensure that, when implemented, these features are correctly specified for the UK.

Features identified as a not required in this profile SHALL NOT be implemented by receivers conforming to this profile.

4 MHEG Engine Graphics Model

4.1 The Graphics Plane

The "graphics plane" is used to represent all visibles except video streams and MPEG I-frame bitmap objects[1].

Drawing Area

The drawing area available for applications has pixel dimensions of 720x576.

Visible Area

See "Visible Area" on page 111.

Scene Aspect Ratio

See "Scene Aspect Ratio" on page 111.

Graphics/Video Pixel Alignment

The 720x576 coordinate system of the graphics plane shall align exactly with the 720x576 coordinate system of the decoded MPEG video provided that:

- the video object is displayed at full size
- the video object position is (0, 0)

and one of the following conditions applies:

- 4:3 content is displayed on a 4:3 aspect ratio display
- 4:3 content is displayed in a pillar box on a 16:9 aspect ratio display
- 16:9 content is displayed on a 16:9 aspect ratio display

Colour Range

The graphics plane shall be able to support colours at least subjectively equivalent to the 256 colours in "The Colour Palette" .

1. MPEG I-frames and video are assumed to reside in a separate truecolour display buffer.

4.2 The Colour Palette

The graphics plane shall support at least 256 colours[1].

Colours provided for MHEG applications

Two fixed palettes (A1 and A2) are built into all receivers. When no DVB Subtitle stream is being decoded at least the colours in palette A1 shall be available for use by the application. When a DVB Subtitle stream is being decoded at least the reduced set of colours in A2 shall be available.

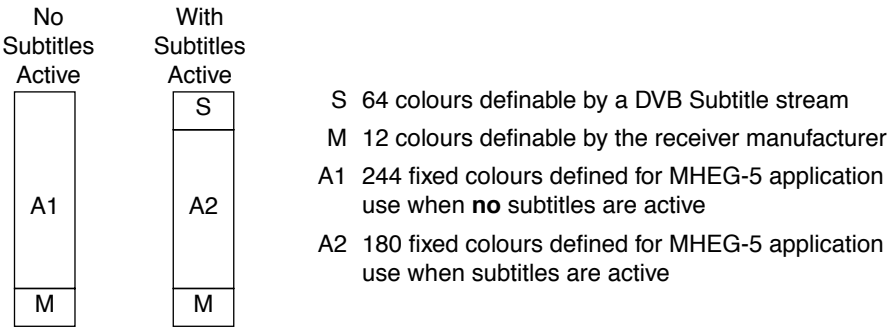


Figure 7. Colour Palette modes of operation

When an application invokes a colour in the currently available palette (A1 or A2) it shall be reproduced exactly.

If applications invoke colours that are not in the currently active palette they shall be reproduced in an implementation dependent way.

Colours provided for DVB Subtitles when sharing the display with an MHEG application.

When a decoded DVB Subtitle stream shares the display with an MHEG application it may use up to 64 different colours at any time (the S palette). These shall be reproduced correctly.

If a subtitle stream uses more than 64 colours, the colours shall be reproduced in an implementation dependent way.

Switching between palette modes

When **no** RTGraphics object with content type is DVB Subtitles[2] is in scope[3] at least the colours in palette A1 shall be available for use by the application.

Palette definition

The tables 13 and 14 list equivalent RGB and YCrCb definitions for each colour in palettes A1 and A2. The component values in each case are gamma pre-corrected. The rules for the construction of the palettes are given in Table 12. Figure 8 on page 39 illustrates the opaque colours in the two palettes.

Palette	T	Red	Green	Blue
A1	100%	0, 51, 102, 152, 205, 255		
A2[a]		0, 51, 102, 152, 205, 255	0, 51, 128, 205, 255	
A1, A2	50%	0, 128, 255		

Table 12. (DRAFT) Palette construction rules

a] Plus in also (186, 186, 186) & (224, 224, 224)

1. This specification allows the graphics plane to be implemented as either an 8 bit indexed store or a truecolour store.
2. The only content encoding permitted in *UKEngineProfile1*
3. I.e. referenced in the application object or the current scene

NOTE: The values in these palettes are the subject of ongoing research.

Between the two palettes 96 of the opaque colours are common and all of the 50% transparent colours are common. Indexes are listed purely for information, colour indexes are not used in *UKEngineProfile1* applications.

Index		T	Red	Green	Blue	Y	Cr	Cb
in A1	in A2	0…100	0…255			16…235	16…240	
0	0	100	0	0	0	16	128	128
1	1	0	0	0	0	16	128	128
2	2	0	0	0	51	20	124	150
3		0	0	0	102	25	120	172
4		0	0	0	154	31	117	195
5	4	0	0	0	205	36	113	218
6	5	0	0	0	255	40	109	239
7	6	0	0	51	0	41	109	113
8	7	0	0	51	51	46	105	135
9		0	0	51	102	51	101	157
10		0	0	51	154	56	98	180
11	9	0	0	51	205	61	94	203
12	10	0	0	51	255	66	91	225
13		0	0	102	0	67	90	98
14		0	0	102	51	72	86	120
15		0	0	102	102	77	83	143
16		0	0	102	154	82	79	165
17		0	0	102	205	87	75	188
18		0	0	102	255	92	72	210
19		0	0	154	0	93	71	83
20		0	0	154	51	98	67	105
21		0	0	154	102	103	64	127
22		0	0	154	154	108	60	150
23		0	0	154	205	113	56	173
24		0	0	154	255	118	53	195
25	16	0	0	205	0	119	52	68
26	17	0	0	205	51	124	48	90
27		0	0	205	102	129	45	113
28		0	0	205	154	134	41	135
29	19	0	0	205	205	139	37	158
30	20	0	0	205	255	144	34	180
31	21	0	0	255	0	144	34	53
32	22	0	0	255	51	149	30	76
33		0	0	255	102	154	26	98
34		0	0	255	154	159	23	121
35	24	0	0	255	205	164	19	143
36	25	0	0	255	255	169	16	165
37	26	0	51	0	0	29	150	120
38	27	0	51	0	51	34	146	142
39		0	51	0	102	39	143	165
40		0	51	0	154	44	139	188
41	29	0	51	0	205	49	135	210
42	30	0	51	0	255	54	132	232
43	31	0	51	51	0	54	131	105
44	32	0	51	51	51	59	128	128
45		0	51	51	102	64	124	150
46		0	51	51	154	69	120	173
47	34	0	51	51	205	74	117	195
48	35	0	51	51	255	79	113	217

Table 13. (DRAFT) Application palette A1 (Sheet 1 of 5)

Index		T	Red	Green	Blue	Y	Cr	Cb
in A1	in A2	0…100	0…255			16…235	16…240	
49		0	51	102	0	80	112	90
50		0	51	102	51	85	109	113
51		0	51	102	102	90	105	135
52		0	51	102	154	95	101	158
53		0	51	102	205	100	98	180
54		0	51	102	255	105	94	202
55		0	51	154	0	106	93	75
56		0	51	154	51	111	90	98
57		0	51	154	102	116	86	120
58		0	51	154	154	121	82	143
59		0	51	154	205	126	79	165
60		0	51	154	255	131	75	187
61	41	0	51	205	0	132	75	60
62	42	0	51	205	51	137	71	83
63		0	51	205	102	142	67	105
64		0	51	205	154	147	64	128
65	44	0	51	205	205	152	60	150
66	45	0	51	205	255	157	56	172
67	46	0	51	255	0	157	56	46
68	47	0	51	255	51	162	52	68
69		0	51	255	102	167	49	91
70		0	51	255	154	172	45	113
71	49	0	51	255	205	177	41	136
72	50	0	51	255	255	182	38	158
73	51	0	102	0	0	42	172	112
74	52	0	102	0	51	47	169	135
75		0	102	0	102	52	165	157
76		0	102	0	154	57	161	180
77	54	0	102	0	205	62	158	202
78	55	0	102	0	255	67	154	224
79	56	0	102	51	0	67	154	98
80	57	0	102	51	51	72	150	120
81		0	102	51	102	77	146	142
82		0	102	51	154	82	143	165
83	59	0	102	51	205	87	139	188
84	60	0	102	51	255	92	135	210
85		0	102	102	0	93	135	83
86		0	102	102	51	98	131	105
87		0	102	102	102	103	128	128
88		0	102	102	154	108	124	150
89		0	102	102	205	113	120	173
90		0	102	102	255	118	117	195
91		0	102	154	0	119	116	68
92		0	102	154	51	124	112	90
93		0	102	154	102	129	108	112
94		0	102	154	154	134	105	135
95		0	102	154	205	139	101	158
96		0	102	154	255	144	97	180
97	66	0	102	205	0	145	97	53
98	67	0	102	205	51	150	93	75
99		0	102	205	102	155	90	98
100		0	102	205	154	160	86	120
101	69	0	102	205	205	165	82	143

Table 13. (DRAFT) Application palette A1 (Sheet 2 of 5)

Index		T	Red	Green	Blue	Y	Cr	Cb
in A1	in A2	0…100	0…255			16…235	16…240	
102	70	0	102	205	255	170	79	165
103	71	0	102	255	0	170	79	38
104	72	0	102	255	51	175	75	61
105		0	102	255	102	180	71	83
106		0	102	255	154	185	68	106
107	74	0	102	255	205	190	64	128
108	75	0	102	255	255	195	60	150
109	76	0	154	0	0	55	195	105
110	77	0	154	0	51	60	191	127
111		0	154	0	102	65	188	149
112		0	154	0	154	70	184	172
113	79	0	154	0	205	75	180	195
114	80	0	154	0	255	80	177	217
115	81	0	154	51	0	81	176	90
116	82	0	154	51	51	86	173	112
117		0	154	51	102	91	169	135
118		0	154	51	154	96	165	157
119	84	0	154	51	205	101	162	180
120	85	0	154	51	255	106	158	202
121		0	154	102	0	106	158	75
122		0	154	102	51	111	154	97
123		0	154	102	102	116	150	120
124		0	154	102	154	122	147	143
125		0	154	102	205	127	143	165
126		0	154	102	255	131	139	187
127		0	154	154	0	133	138	60
128		0	154	154	51	138	135	82
129		0	154	154	102	143	131	105
130		0	154	154	154	148	128	128
131		0	154	154	205	153	124	150
132		0	154	154	255	158	120	172
133	91	0	154	205	0	158	120	45
134	92	0	154	205	51	163	116	67
135		0	154	205	102	168	112	90
136		0	154	205	154	173	109	113
137	94	0	154	205	205	178	105	135
138	95	0	154	205	255	183	102	157
139	96	0	154	255	0	184	101	30
140	97	0	154	255	51	189	98	53
141		0	154	255	102	194	94	75
142		0	154	255	154	199	90	98
143	99	0	154	255	205	204	87	121
144	100	0	154	255	255	209	83	142
145	101	0	205	0	0	68	218	97
146	102	0	205	0	51	73	214	120
147		0	205	0	102	78	210	142
148		0	205	0	154	83	207	165
149	104	0	205	0	205	88	203	187
150	105	0	205	0	255	93	199	209
151	106	0	205	51	0	94	199	82
152	107	0	205	51	51	99	195	105
153		0	205	51	102	104	191	127
154		0	205	51	154	109	188	150

Table 13. (DRAFT) Application palette A1 (Sheet 3 of 5)

Index		T	Red	Green	Blue	Y	Cr	Cb
in A1	in A2	0…100	0…255			16…235	16…240	
155	109	0	205	51	205	114	184	172
156	110	0	205	51	255	119	181	194
157		0	205	102	0	120	180	67
158		0	205	102	51	125	176	90
159		0	205	102	102	130	173	112
160		0	205	102	154	135	169	135
161		0	205	102	205	140	165	157
162		0	205	102	255	145	162	179
163		0	205	154	0	146	161	52
164		0	205	154	51	151	157	75
165		0	205	154	102	156	154	97
166		0	205	154	154	161	150	120
167		0	205	154	205	166	146	142
168		0	205	154	255	171	143	164
169	116	0	205	205	0	171	142	37
170	117	0	205	205	51	176	138	60
171		0	205	205	102	181	135	82
172		0	205	205	154	187	131	105
173	119	0	205	205	205	192	128	128
174	120	0	205	205	255	196	124	149
175	121	0	205	255	0	197	124	23
176	122	0	205	255	51	202	120	45
177		0	205	255	102	207	116	68
178		0	205	255	154	212	113	91
179	124	0	205	255	205	217	109	113
180	125	0	205	255	255	222	106	135
181	126	0	255	0	0	81	239	90
182	127	0	255	0	51	86	236	112
183		0	255	0	102	91	232	135
184		0	255	0	154	96	229	157
185	129	0	255	0	205	101	225	180
186	130	0	255	0	255	106	221	202
187	131	0	255	51	0	107	221	75
188	132	0	255	51	51	112	217	97
189		0	255	51	102	117	213	120
190		0	255	51	154	122	210	143
191	134	0	255	51	205	127	206	165
192	135	0	255	51	255	132	203	187
193		0	255	102	0	132	202	60
194		0	255	102	51	137	198	82
195		0	255	102	102	142	195	105
196		0	255	102	154	147	191	128
197		0	255	102	205	152	187	150
198		0	255	102	255	157	184	172
199		0	255	154	0	159	183	45
200		0	255	154	51	164	179	67
201		0	255	154	102	169	176	90
202		0	255	154	154	174	172	113
203		0	255	154	205	179	168	135
204		0	255	154	255	184	165	157
205	141	0	255	205	0	184	164	30
206	142	0	255	205	51	189	160	52
207		0	255	205	102	194	157	75

Table 13. (DRAFT) Application palette A1 (Sheet 4 of 5)

Index		T	Red	Green	Blue	Y	Cr	Cb
in A1	in A2	0…100	0…255			16…235	16…240	
208		0	255	205	154	199	153	98
209	144	0	255	205	205	204	149	120
210	145	0	255	205	255	209	146	142
211	146	0	255	255	0	210	146	16
212	147	0	255	255	51	215	142	38
213		0	255	255	102	220	138	60
214		0	255	255	154	225	135	83
215	149	0	255	255	205	230	131	106
216	150	0	255	255	255	235	128	128
217	217	50	0	0	0	16	128	128
218	218	50	0	0	128	28	118	184
219	219	50	0	0	255	40	109	239
220	220	50	0	128	0	80	80	90
221	221	50	0	128	128	93	71	146
222	222	50	0	128	255	105	62	202
223	223	50	0	255	0	144	34	53
224	224	50	0	255	128	157	25	110
225	225	50	0	255	255	169	16	165
226	226	50	128	0	0	48	184	109
227	227	50	128	0	128	61	175	165
228	228	50	128	0	255	73	166	221
229	229	50	128	128	0	113	137	71
230	230	50	128	128	128	125	128	128
231	231	50	128	128	255	138	118	183
232	232	50	128	255	0	177	90	34
233	233	50	128	255	128	189	81	91
234	234	50	128	255	255	202	72	146
235	235	50	255	0	0	81	239	90
236	236	50	255	0	128	94	230	146
237	237	50	255	0	255	106	221	202
238	238	50	255	128	0	146	192	52
239	239	50	255	128	128	158	183	109
240	240	50	255	128	255	170	174	164
241	241	50	255	255	0	210	146	16
242	242	50	255	255	128	222	137	72
243	243	50	255	255	255	235	128	128
244 : 255		Reserved for manufacturer use						

Table 13. (DRAFT) Application palette A1 (Sheet 5 of 5)

Index		T	Red	Green	Blue	Y	Cr	Cb
in A2	in A1	0…100	0…255			16…235	16…240	
0	0	100	0	0	0	16	128	128
1	1	0	0	0	0	16	128	128
2	2	0	0	0	51	20	124	150
3		0	0	0	128	28	118	184
4	5	0	0	0	205	36	113	218
5	6	0	0	0	255	40	109	239
6	7	0	0	51	0	41	109	113

Table 14. (DRAFT) Application palette A2 (Sheet 1 of 5)

Index		T	Red	Green	Blue	Y	Cr	Cb
in A2	in A1	0…100	0…255			16…235	16…240	
7	8	0	0	51	51	46	105	135
8		0	0	51	128	54	100	169
9	11	0	0	51	205	61	94	203
10	12	0	0	51	255	66	91	225
11		0	0	128	0	80	80	90
12		0	0	128	51	85	77	113
13		0	0	128	128	93	71	146
14		0	0	128	205	100	66	180
15		0	0	128	255	105	62	202
16	25	0	0	205	0	119	52	68
17	26	0	0	205	51	124	48	90
18		0	0	205	128	131	43	124
19	29	0	0	205	205	139	37	158
20	30	0	0	205	255	144	34	180
21	31	0	0	255	0	144	34	53
22	32	0	0	255	51	149	30	76
23		0	0	255	128	157	25	110
24	35	0	0	255	205	164	19	143
25	36	0	0	255	255	169	16	165
26	37	0	51	0	0	29	150	120
27	38	0	51	0	51	34	146	142
28		0	51	0	128	41	141	176
29	41	0	51	0	205	49	135	210
30	42	0	51	0	255	54	132	232
31	43	0	51	51	0	54	131	105
32	44	0	51	51	51	59	128	128
33		0	51	51	128	67	122	161
34	47	0	51	51	205	74	117	195
35	48	0	51	51	255	79	113	217
36		0	51	128	0	93	103	83
37		0	51	128	51	98	99	105
38		0	51	128	128	106	94	139
39		0	51	128	205	113	88	173
40		0	51	128	255	118	85	195
41	61	0	51	205	0	132	75	60
42	62	0	51	205	51	137	71	83
43		0	51	205	128	144	65	117
44	65	0	51	205	205	152	60	150
45	66	0	51	205	255	157	56	172
46	67	0	51	255	0	157	56	46
47	68	0	51	255	51	162	52	68
48		0	51	255	128	170	47	102
49	71	0	51	255	205	177	41	136
50	72	0	51	255	255	182	38	158
51	73	0	102	0	0	42	172	112
52	74	0	102	0	51	47	169	135
53		0	102	0	128	54	163	169
54	77	0	102	0	205	62	158	202
55	78	0	102	0	255	67	154	224
56	79	0	102	51	0	67	154	98
57	80	0	102	51	51	72	150	120
58		0	102	51	128	80	144	154
59	83	0	102	51	205	87	139	188

Table 14. (DRAFT) Application palette A2 (Sheet 2 of 5)

Index		T	Red	Green	Blue	Y	Cr	Cb
in A2	in A1	0…100	0…255			16…235	16…240	
60	84	0	102	51	255	92	135	210
61		0	102	128	0	106	125	75
62		0	102	128	51	111	122	98
63		0	102	128	128	119	116	131
64		0	102	128	205	126	111	165
65		0	102	128	255	131	107	187
66	97	0	102	205	0	145	97	53
67	98	0	102	205	51	150	93	75
68		0	102	205	128	158	88	109
69	101	0	102	205	205	165	82	143
70	102	0	102	205	255	170	79	165
71	103	0	102	255	0	170	79	38
72	104	0	102	255	51	175	75	61
73		0	102	255	128	183	69	94
74	107	0	102	255	205	190	64	128
75	108	0	102	255	255	195	60	150
76	109	0	154	0	0	55	195	105
77	110	0	154	0	51	60	191	127
78		0	154	0	128	68	186	161
79	113	0	154	0	205	75	180	195
80	114	0	154	0	255	80	177	217
81	115	0	154	51	0	81	176	90
82	116	0	154	51	51	86	173	112
83		0	154	51	128	93	167	146
84	119	0	154	51	205	101	162	180
85	120	0	154	51	255	106	158	202
86		0	154	128	0	120	148	67
87		0	154	128	51	125	144	90
88		0	154	128	128	132	139	124
89		0	154	128	205	140	133	157
90		0	154	128	255	145	130	179
91	133	0	154	205	0	158	120	45
92	134	0	154	205	51	163	116	67
93		0	154	205	128	171	111	101
94	137	0	154	205	205	178	105	135
95	138	0	154	205	255	183	102	157
96	139	0	154	255	0	184	101	30
97	140	0	154	255	51	189	98	53
98		0	154	255	128	196	92	87
99	143	0	154	255	205	204	87	121
100	144	0	154	255	255	209	83	142
101	145	0	205	0	0	68	218	97
102	146	0	205	0	51	73	214	120
103		0	205	0	128	81	208	153
104	149	0	205	0	205	88	203	187
105	150	0	205	0	255	93	199	209
106	151	0	205	51	0	94	199	82
107	152	0	205	51	51	99	195	105
108		0	205	51	128	106	190	138
109	155	0	205	51	205	114	184	172
110	156	0	205	51	255	119	181	194
111		0	205	128	0	133	170	60
112		0	205	128	51	138	167	82

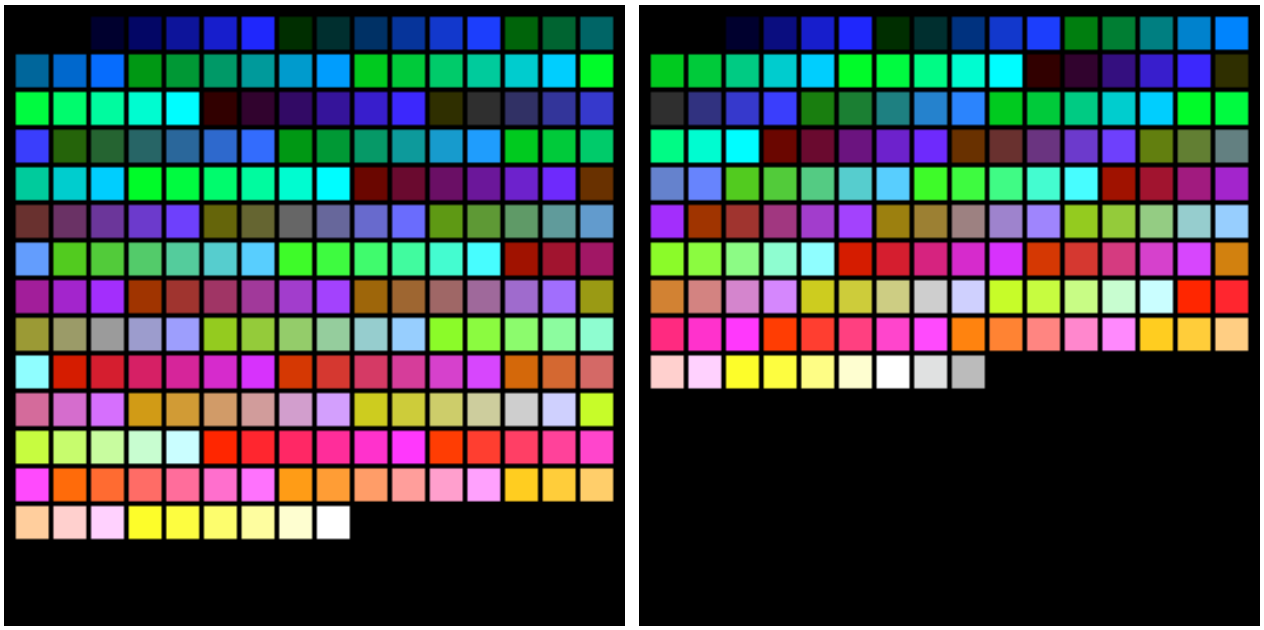
Table 14. (DRAFT) Application palette A2 (Sheet 3 of 5)

Index		T	Red	Green	Blue	Y	Cr	Cb
in A2	in A1	0…100	0…255			16…235	16…240	
113		0	205	128	128	145	161	116
114		0	205	128	205	153	156	150
115		0	205	128	255	158	152	172
116	169	0	205	205	0	171	142	37
117	170	0	205	205	51	176	138	60
118		0	205	205	128	184	133	94
119	173	0	205	205	205	192	128	128
120	174	0	205	205	255	196	124	149
121	175	0	205	255	0	197	124	23
122	176	0	205	255	51	202	120	45
123		0	205	255	128	209	115	79
124	179	0	205	255	205	217	109	113
125	180	0	205	255	255	222	106	135
126	181	0	255	0	0	81	239	90
127	182	0	255	0	51	86	236	112
128		0	255	0	128	94	230	146
129	185	0	255	0	205	101	225	180
130	186	0	255	0	255	106	221	202
131	187	0	255	51	0	107	221	75
132	188	0	255	51	51	112	217	97
133		0	255	51	128	119	212	131
134	191	0	255	51	205	127	206	165
135	192	0	255	51	255	132	203	187
136		0	255	128	0	146	192	52
137		0	255	128	51	151	189	75
138		0	255	128	128	158	183	109
139		0	255	128	205	166	178	142
140		0	255	128	255	170	174	164
141	205	0	255	205	0	184	164	30
142	206	0	255	205	51	189	160	52
143		0	255	205	128	197	155	86
144	209	0	255	205	205	204	149	120
145	210	0	255	205	255	209	146	142
146	211	0	255	255	0	210	146	16
147	212	0	255	255	51	215	142	38
148		0	255	255	128	222	137	72
149	215	0	255	255	205	230	131	106
150	216	0	255	255	255	235	128	128
151		0	224	224	224	208	128	128
152		0	186	186	186	175	128	128
153 : 216		Reserved for DVB Subtitles						
217	217	50	0	0	0	16	128	128
218	218	50	0	0	128	28	118	184
219	219	50	0	0	255	40	109	239
220	220	50	0	128	0	80	80	90
221	221	50	0	128	128	93	71	146
222	222	50	0	128	255	105	62	202
223	223	50	0	255	0	144	34	53
224	224	50	0	255	128	157	25	110
225	225	50	0	255	255	169	16	165
226	226	50	128	0	0	48	184	109

Table 14. (DRAFT) Application palette A2 (Sheet 4 of 5)

Index		T	Red	Green	Blue	Y	Cr	Cb
in A2	in A1	0…100	0…255			16…235	16…240	
227	227	50	128	0	128	61	175	165
228	228	50	128	0	255	73	166	221
229	229	50	128	128	0	113	137	71
230	230	50	128	128	128	125	128	128
231	231	50	128	128	255	138	118	183
232	232	50	128	255	0	177	90	34
233	233	50	128	255	128	189	81	91
234	234	50	128	255	255	202	72	146
235	235	50	255	0	0	81	239	90
236	236	50	255	0	128	94	230	146
237	237	50	255	0	255	106	221	202
238	238	50	255	128	0	146	192	52
239	239	50	255	128	128	158	183	109
240	240	50	255	128	255	170	174	164
241	241	50	255	255	0	210	146	16
242	242	50	255	255	128	222	137	72
243	243	50	255	255	255	235	128	128
244 : 255		Reserved for manufacturer use						

Table 14. (DRAFT) Application palette A2 (Sheet 5 of 5)



A1 Palette A2 Palette

Figure 8. A1 & A2 Palettes (showing opaque colours only)

4.3 Colour Representation

4.3.1 Colour Space

The engine is responsible for converting between colour spaces as is required.

Depending on the content type the MHEG engine handles colours in both RGB (colour for buttons, text etc. and PNG graphics) and YC_rC_b (MPEG stills and DVB subtitles) colour spaces.

This engine specification doesn't comment on the colour representation used in the engine's framestore(s).

4.3.2 Gamma

MPEG video and DVB Subtitles deliver YC_rC_b data in accordance with ITU-R 601. These signals are gamma pre-corrected. **Receivers shall assume that ALL RGB values invoked by MHEG applications are comparably gamma pre-corrected.**

Application authors are advised to pre-correct RGB values (such as those in PNG graphics) to be consistent with the pre-correction applied to the MPEG video.

4.3.3 Direct/Absolute colours

Direct/Absolute colour values in MHEG applications shall be expressed as a 4 byte Octet-String constructed as shown in Figure 9.

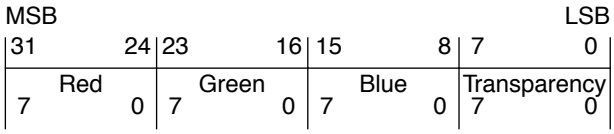


Figure 9. Absolute colour format

The Red, Green and Blue code values in the range 0 to 255. The Transparency byte codes values in the range 0 to 255 representing transparency. 0 is opaque.

4.3.4 PNG Modes

Engines are required to support ALL of the PNG colour types defines in PNG Specification Version 1.0 (see Table 15). Engines are responsible for mapping these colours to those used by the engine's OSD.

Colour Type	Allowed Bit Depths	Interpretation
0	1,2,4,8,16	Each pixel is a grayscale sample.
2	8,16	Each pixel is an R,G,B triple.
3	1,2,4,8	Each pixel is a palette index; PLTE chunk must appear.
4	8,16	Each pixel is a grayscale sample, followed by an alpha sample.
6	8,16	Each pixel is an R,G,B triple, followed by an alpha sample.

Table 15. PNG Formats

Any combination of PNGs with different colour types may be active at any one time. Similarly, engines are responsible for mapping RGB16 direct colour specifications to colours that the OSD can support.

Where PNG graphics use colours defined in the currently active application palette (A1 or A2) these colours shall be reproduced correctly. Other colours shall be reproduced in an implementation dependent way.

Receivers should ignore gAMA (gamma) and cHRM (chromaticity) chunks in PNG files.

4.4 Overlapping Visibles

4.4.1 Transparency & Overlapping Visibles

Levels of transparency

Receivers are required to implement 3 levels of transparency 0% (opaque), 50% and 100% (completely transparent). Implementation of additional intermediate levels of transparency is optional.

Overlaying visibles

When visibles overlap the MHEG-5 rules for Rendering Visibles shall be observed where transparency is 0% or 100% or where semi-transparent pixels are the only visible[1] pixels above MPEG video or an MPEG I-frame.

Where intermediate levels of transparency overlay other forms of Visible certain approximations are permitted. If semi-transparent pixels overlay one or more layers of semi-transparent pixels the top most semi-transparent pixel may “punch through” to the video or be treated as opaque. However, semi-transparent pixels may not “punch through” opaque pixels.

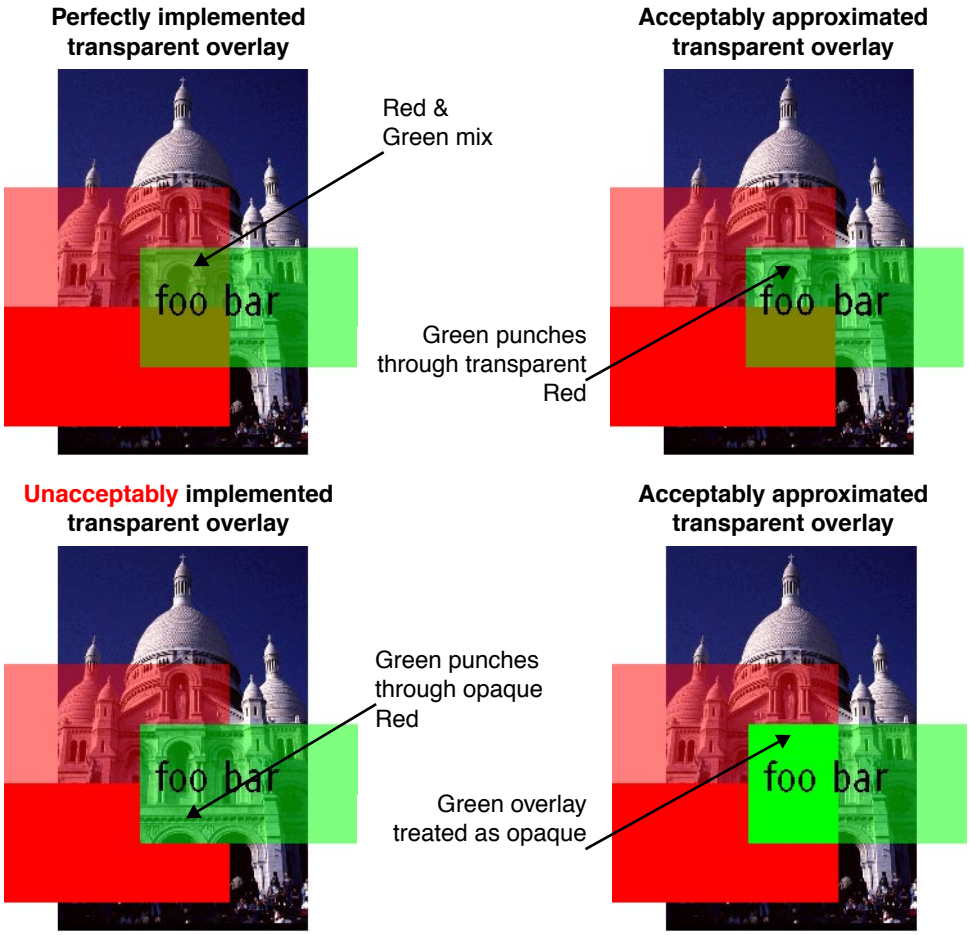


Figure 10. Approximation of transparency

1. I.e. with transparency < 100%.

Rendering performance

Authors should be aware that the graphics drawing speed may decrease where visibles with an intermediate level of transparency are placed directly over objects other than MPEG video or MPEG I-frames .

Also, visible objects may overlay RTGraphics or the TV service's DVB Subtitles. However, the RTGraphic/Subtitle rendering speed may degrade and possibly even stop.

4.4.2 RTGraphics & Overlapping Visibles

In this profile receivers always treat an RTGraphics object as being directly above a Video object. This overrides certain aspects of the normal MHEG-5 behaviour for controlling the position of objects within the display stack:

- Actions that affect the position of the Video object in the display stack implicitly also affect the position of the RTGraphics object.
- Where an RTGraphics object and a Video object are concurrently active actions that should normally affect the stack position of the RTGraphics object shall be ignored.

Where no Video object is concurrently active with an RTGraphics object actions that affect the stack position of the RTGraphics object shall treated as normal. See also "DVB Subtitle decoder performance" on page 68.

4.5 LineArt & DynamicLineArt

The following allowances are made to assist receiver implementation. Authors should take account of the implied authoring constraints.

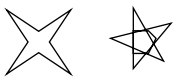
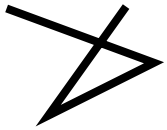
Topic	Receiver allowance	Authoring guideline
LineStyle attribute	Implement ALL line styles as solid	Avoid using dashed or dotted line styles (other line attributes such as width or colour should be used to differentiate line styles)
Filled closed shapes	The receiver behaviour when filling certain shapes is not defined. These shapes are: • concave polygons • self-crossing polygons All other shapes shall be completely filled with the colour defined by the RefFillColour attribute.	Avoid using filled concave polygons and filled self crossing polygons. I.e. avoid filling shapes like:  If these shapes such as these are required filled they may be constructed from primitive elements such as triangles which are guaranteed to fill in a predictable way.
Self crossing polygons and polylines.	The appearance of pixels at the junction of self crossing lines is not defined.	Avoid self crossing lines such as: 

Table 16. Limitations on LineArt and DynamicLineArt

4.6 PNG Aspect ratios

PNG bitmaps shall carry a pHYs chunk indicating the pixel aspect ratio of the bitmap. This aspect ratio should be the same as that of the scene containing the bitmap.

The PNG specification indicates that if the aspect ratio is absent square pixels should be assumed. To avoid overriding this specification the aspect ratio should be signalled explicitly.

4.7 Numbers of Objects

The minimum number of concurrently active MHEG-5 objects that this profile of engine is required to support are:

- 1 Video object (an MPEG video stream) **OR** 1 Bitmap object using MPEG I frame encoding
- 1 Audio object (an MPEG audio stream) with source ‘stream’ or ‘memory’
- 1 DVB Subtitle stream

See “Use of stream decoders” on page 111.

The numbers of other presentable object types (e.g. PNG bitmaps, buttons etc.) are only limited by the decoder memory model.

4.8 MPEG stills

4.8.1 File format

The payload of a file delivering an MPEG I frame shall:

- be a valid video_sequence() including a sequence_extension()
- contain one I frame, i.e. one picture_header(), one picture_coding_extension(), and one picture_data() encoded as an intra coded frame, with picture structure = “frame”

That is the structure is:

```
sequence_header()
sequence_extension()
extension_and_user_data(0)
optional group_of_pictures_header() and extension_and_user_data(2)
picture_header(picture_coding_type = “I frame”)
picture_coding_extension(picture_structure = “frame picture”)
picture_data()
sequence_end_code()
```

4.8.2 Semantics

An MPEG-2 video decoder conforming to the same behaviour as the main video decoder (ETR 154 etc.) is used to decode the fragment of data containing the I-frame. The I-frame is scaled in the same way that MPEG video would be scaled. ScaleBitmap(X,Y) may be used with the same constraints as ScaleVideo(X, Y). See “Video Scaling” on page 68.

So, a bitmap object could have content which is an MPEG I-frame. This frame could be coded as 352x288 and subject to a ScaleBitmap(360, 288) action to provide a 1/4 screen truecolour still picture.

4.9 Graphics priority

The priority to access display resources is (in order of decreasing priority):

- Other display using processes (e.g. receiver displays such the navigator, displays produced by the CA system etc.)
- Broadcast MHEG-5 applications
- Broadcast services (video and subtitling)

When a process with higher priority requests access to the display resources it shall be granted it. This specification doesn't specify in detail the manner in which a receiver provides this resource. However:

- When a higher priority process takes the display from a broadcast MHEG-5 application the applications shall be "paused" rather than killed.
- While the MHEG-5 application does not have access to the display the engine is not required to register real-time events. So, for example, stream or timer events may be missed. See "Missed events" on page 111.
- When the display is restored to the application, the engine is responsible for any redrawing that may be required.

5 Text & Hypertext

5.1 Character encoding

5.1.1 Presentable text

This section describes the character encoding used for **all** presentable text. That is the text used as included or referenced content by the following classes:

- Text
- EntryField
- HyperText
- Button, PushButton, SwitchButton

Additionally the Value of an OctetStringVariable shall be encoded in this way.

"Character repertoire" on page 58 lists the minimum set of characters supported by the engine. Characters shall be encoded according to ISO 10646-1 [18] and the Universal Character Set Transformation Format, 8-bit format (UTF-8) which is standardised as amendment 2 to ISO 10646-1 [18].

UTF-8

Table 17 reproduces the UTF-8 coding scheme. The character repertoire in Table 30 will only require 1, 2 or 3 byte codes. Where text is in English, Welsh or Gaelic, the majority of characters will be coded on 1 byte.

ISO 10646-1 value	1st byte	2nd byte	3rd byte	4th byte
0000 0000 0xxx xxxx	0xxx xxxx			
0000 0yyy yyxx xxxx	110y yyyy	10xx xxxx		
zzzz yyyy yyxx xxxx	1110 zzzz	10yy yyyy	10xx xxxx	
1101 10ww wwzz zzyy + 1101 11yy yyxx xxxx	1111 0uuu[a]	10uu zzzz	10yy yyyy	10xx xxxx

Table 17. UTF-8 Bit Distribution

a] Where uuuuu = wwww + 1

CharacterSet attribute

Table 18 identifies the minimum set of values of CharacterSet attribute that the engine shall support.

Attribute	Character Set
1	The subset of ISO 10646-1 tabulated in Table 30 encoded with UTF-8.
>1	Reserved for future use

Table 18. Engine CharacterSet Attributes

5.1.2 Non-presented text

This section describes the character encoding used for text which is **not** for presentation. That is the text used in Object and Content references.

OctetStrings shall uses octet values in the range 0x01 to 0x7f.

5.1.3 Type conversion

Where OctetStrings are converted between from “presentable” to “non-presentable” each **character** in the “presentable” text shall be converted to a **single byte** by masking with 0x7f.

This type conversion will be applied by the “CastToContentRef” and “CastToObjectRef[a]” resident programs (see page 26).

5.2 Fonts

5.2.1 Downloading

Receivers implementing *UKEngineProfile1* shall not support downloadable fonts or the Font class. Application references to fonts shall be direct (i.e. an OctetString representing the name of the font).

5.2.1.1 Future compatibility

The measures outlined below are designed to improve interoperability with possible future services that make use of downloaded fonts. The preferred approach is the “Authoring solution”. However, receivers shall implement the “Receiver defensive response”.

Authoring solution

Future applications shall be able to determine (via the GetEngineSupport mechanism) the capabilities of the receiver. Using this information applications can adapt their behaviour to avoid problems.

Receiver defensive response

Receivers shall implement the following measures to ensure robust behaviour with any future applications that do use downloaded fonts, or font characteristics that the receiver doesn’t recognise:

- The receiver shall use its in-built font.
- If the font style (e.g. bold or plain) is recognised that shall be used, if not plain shall be used in lieu.
- If the font size does not match one of those of the in-built font the receiver shall substitute the next smaller size provided by the in-built font. If the required font is smaller than the smallest available, then the smallest available in-built size shall be used.
- Where there are unrecognised character codes, the receiver shall display character ‘!’ (0x0021) in that position.

5.2.2 Embedded font

5.2.2.1 The DTG/RNIB font design project

The Project

A font design and testing program is being conducted by the DTG and the RNIB with technical support from Bitstream. The objective of this program is to deliver a font with good legibility for use in TV subtitling. The acceptability tests were conducted at the end of 1997 with a view to delivering the finished font to the DTG at the end of January 1998. The same font design will be suitable for use in MHEG-5 applications.

Font Characteristics

The font is a kerned sans-serif (like Helvetica or Arial) designed to look good on both 4:3 and 16:9 displays. See “Text on a 4:3 display” on page 51.

IPR

This font, “UK1”, is provided by the RNIB on a royalty free basis for use in digital TV products. Manufacturers should contact the RNIB regarding license arrangements if they wish to use the font for another purpose.

5.2.2.2 Required sizes and styles

Receivers shall implement at least the DTG/RNIB font "UK1" in at least the sizes identified in Table 19. This shall be the default font implemented by the engine.

Size (points)	Informative Name	Styles	
		Plain	Bold
32	Heading / Large subtitle	✓	TBD
28	Subtitle	✓	TBD
24	Body	✓[a]	TBD
20	Footnote	✓	TBD

Table 19. "UK1" sizes and styles

a] The default size and style

5.2.3 Invoking the font

The font name "UK1" can be used as a FontName in the application Font attribute or as a text object OriginalFont.

5.2.4 FontAttributes

Receivers shall support two font attribute formats, one is DAVIC-like (but verbose), the second is terser. The short and long forms shall not be mixed within an attribute string.

5.2.4.1 DAVIC-like long form

This string format is <style>.<size>.<linespace>.<letterspace> is carried in an OctetString.[1]

Field	Set of allowed values	Meaning
style	'bold'	bold text
	'plain'	plain text
size	'20' '24' '28' '32'	font size in points as decimal integer strings
linespace	'0' ... '255'	space between the baselines of adjacent lines of text in points as decimal integer strings
letterspace	'-32767' ... '32767'	increase in spacing in 1/256 points between consecutive characters expressed as a signed decimal integer string

Table 20. Long form text format parameters

1. For example, "bold.28.32.0" means bold 28 point text on 32 point line spacing with default letterspace.

5.2.4.2 Short form

The font attributes are a 5 byte OctetString.

syntax	bits	type	allowed values
`style`	8	bslbf	'b' or 'p'
`size`	8	uimsbf	0x14, 0x18, 0x1C or 0x20
`linespace`	8	uimsbf	0...255
`letterspace`	16	tcimsbf	-32767...32767

Table 21. Short form text format parameters

style: 'b' signals bold, 'p' signals plain.

size: An 8 bit unsigned integer giving the height of the font face in points.

linespace: An 8 bit unsigned integer giving the spacing between the baselines of adjacent lines of text in points.

letterspace: A 16 bit signed integer specifying in units of 1/256th point the required increase in the spacing between consecutive characters.

5.2.5 Character Set

Character Set	Meaning
1	The character set listed in "Character repertoire" on page 58
all other values	Reserved

Table 22. Supported values of character set

5.2.6 TextColour

The default text colour is opaque 'white' RGB24 value 0xCDCDCD.

5.3 Text placement

5.3.1 Font Definition

The characteristics of the fonts used by the engine shall be defined in terms of the Portable Font Resource (PFR) defined in DAVIC 1.3 Part 09 [27]. The PFR is a "public" data structure that can describe both scalable outline fonts and bitmap fonts in a number of sizes. In *UKEngineProfile1* the PFR is used as a registration format for a small set of character sizes.

For *UKEngineProfile1* receivers a PFR shall be provided that gives BOTH a scalable outline representation of the font "UK1" AND a set of bitmap representations of the font in the sizes and styles identified in Table 19 on page 47.

Kerning

Pair kerning information shall be defined using pairKernData items ("Pair kerning Data") embedded in the physFontRecord (See "Physical font record") of the PFR.

5.3.2 “logical” text width rules

To ensure that text will flow identically[1] on different receivers and authoring stations, regardless of the quality of the character rendering a set of “logical” text width rules are defined here. These rules are a simplification of the rules that might be applied in a typographic rendering system. The objective of these simplifications is to reduce the receiver complexity required to ensure exact correlation of text flow behaviour.

The “logical” text width shall be used in the following cases:

- to determine when to wrap lines of text within a text object
- to determine which tab stops text has passed when implementing tab characters

These rules also provide a baseline method for computing character placement when rendering characters. However, receivers may also use the higher precision methods.

5.3.2.1 Computing “logical” text width

The key parameters when calculating the width of a string of N characters are:

- Text font size (one of the values in Table 19, ““UK1” sizes and styles,” on page 47)
- charSetWidth (carried by the “Character Record”)
- The metricsResolution (carried by the “Physical font record”)
- Any kerning adjustment (See “Pair kerning Data”)

Font sizes

Font sizes are expressed as the size of an “Em”[2] in units of “points”[3].

Character widths

The Physical font record carries a Character Record for each character code. This gives the width of each character relative to the size of an Em in metricsResolution units[4].

Kerning

For certain character combinations (a “kerning pair”) a kerning adjustment may also be provided. Typically kerning reduces character spacing for pairs such as AV instead of AV

The Pair kerning Data within the pairKernData extra data items in the Physical font record defines this adjustment. This defines a baseAdjustment (shared by several character pairs) and an adjustment (for a particular character pair). These provide a signed adjustment to the nominal charSetWidth of the first character.

Like charSetWidth kerning adjustments are in terms of metricsResolution units.

Kerning adjustments only apply between characters, not between the start of a line of text and the edge of the text object.

Tracking

Tracking (defined in the font attribute of the text object See “FontAttributes” on page 47) allows for an expansion/condensation of the character spacing for all of the characters in a text object.

1. I.e. lines and words will break at the same character position.
2. Broadly speaking this is the minimum distance between the baselines of consecutive lines of text in the given font. If text is 48 point then the Em at that size is 48 points.
3. An archaic typographical unit. Traditionally there were 72.27 points to an inch. Computerised systems now use 72 points per inch for simplicity.
4. So, if metrics are specified in 1/1000ths of an Em a character with a width of 0.6 Em will have a set width of 600.

5.3.2.2 Logical text width

The equation below shows how the width of a string of N characters is computed.

$$\text{logical width of N characters}_{\text{points}} =$$

$$\text{div}((N-1) \times \text{track},256) + \text{div}(\text{fontsize} \times \left(\sum_{1}^{N} \text{charSetWidth[i]} + \sum_{1}^{N-1} \text{kern[i,i+1]} \right), \text{metricsResolution})$$

$$\text{logical width of N characters}_{\text{pixels}} = \text{div}(\text{logical width of N characters}_{\text{points}} \times 45,56)$$

Where in div(A, B):

- B is unsigned and A is signed

  and

- div(A, B) = ceil(A / B)

Where '/' is a rational divide and ceil(A) is the first integral number greater than or equal to A. So, the calculations round up when reducing precision and tend to over estimate the width of text.

The width of pixels is 56/45 points, see "Converting measurements to display pixels".

5.3.2.3 Converting measurements to display pixels

The calculations above are in a high resolution physical coordinate system. These measurements need to be converted into the pixel resolution of the display/printing device to allow characters to be rendered.

Computer display systems typically assume a 72 pixel per inch display. So, as each point is 1/72 inch, the horizontal and vertical size of each pixel is 1 point.

Vertical resolution

The computer convention is preserved for the vertical dimension of the MHEG display (i.e. each of 576 OSD lines is considered to be 1 point high).

Horizontal resolution

Due to the non-square pixel nature of broadcast displays the 1 pixel = 1 point convention cannot be preserved.

From a typographic view point the OSD is 8 inches high. Correspondingly the width of a 4:3 display is 10.66˙ inches (768 points) and a 16:9 display 14.22˙ inches (1024 points). So, the width of each of the 720 pixels on a 4:3 display is 48/45 points. Similarly, on a 16:9 display the pixels are 64/45 points wide.

A simplifying constraint

To simplify the *UKEngineProfile1* ALL receivers shall assume that each of the 720 pixels has a uniform width of 56/45 points regardless of the scene aspect ratio and display aspect ratio. This corresponds to a 14:9 aspect ratio. So, all text will be subject to moderate aspect ratio distortion. However, broadcasters will be able to predict text flow without having to author specifically for each display type. Also, this allows receivers to be implemented with font bitmaps in a single aspect ratio.

Text on a 4:3 display

The quick brown fox jumped over the lazy dog.
Cozy lummox gives smart squid who asks for job pen.

Text on a 16:9 display

The quick brown fox jumped over the lazy dog.
Cozy lummox gives smart squid who asks for job pen.

5.3.3 Control of text flow

5.3.3.1 Line breaking

If the TextWrapping attribute of an object is set to “true” the engine shall wrap text within the text box. The behaviour is equivalent to identifying the first word that won’t fit completely within the text box and replacing the space character that precedes the word with a Carriage Return character. I.e. line breaks are only inserted where there are space characters.

If a single word is bigger than the text box, or TextWrapping is set to “false” the engine shall truncate text just before the last character that won’t fit completely within the text box.

The Nth character, after which to break a line, is determine by testing to see if the “logical” width in pixels of the N characters is greater than the distance to the edge of the text box in pixels from the origin of the N characters[1].

1. Which will either be the starting edge of the text box or a subsequent tab stop

5.3.3.2 Tabulation

In left aligned text (HorizontalJustification = start) tab stops are defined horizontally every 56 points (45 pixels) from the left edge of the text box. 'Horizontal Tabulation' advances the origin of the next character to be rendered to the next tab stop in the direction that the text is currently flowing (the character repertoire in Table 30 only requires left to right text).

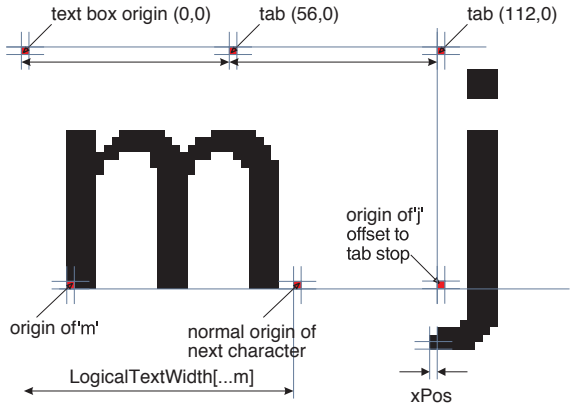


Figure 11. Effect of horizontal tabulation

Note: a tab ALWAYs advances the rendering of the text. I.e. if the normal origin of the next character to be rendered is on a tab stop, a tab character will advance the rendering to the subsequent tab stop.

5.3.3.3 Line spacing

The line space FontAttribute defines the space between the baselines of consecutive lines of text in points.

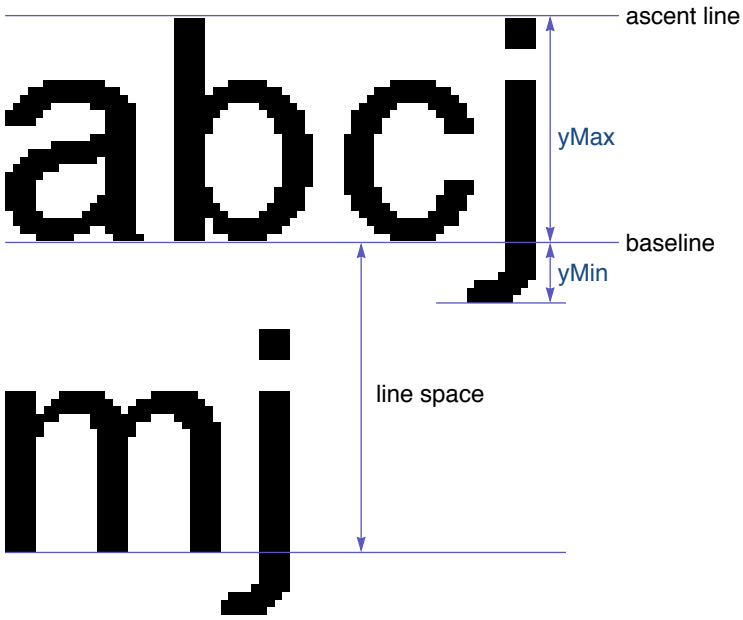


Figure 12. Vertical measures

5.3.3.4 Reproducing Justification

Vertical - start

When VerticalJustification = start, the baseline of the top line of text shall be yMax below the origin of the text box. See Figure 13.

Horizontal - start

When HorizontalJustification = start (left aligned text) the origin of the first (left most) character shall be at the edge of the enclosing text object. Certain characters (e.g. j) may extend left of the origin (i.e. xPos is negative). Text hanging outside of a text object shall be drawn provided that it lies within the OSD area[1]. See Figure 13.

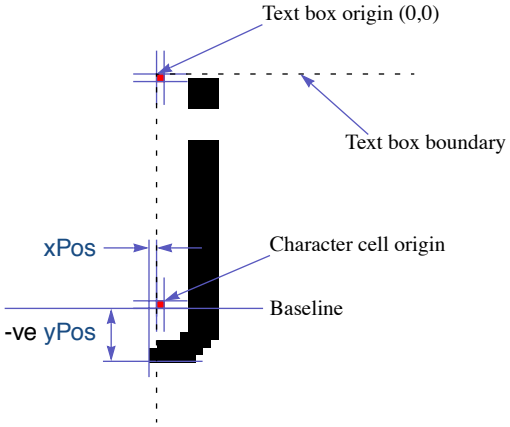


Figure 13. Top-left detail

Vertical - end

When VerticalJustification = end, the baseline of the bottom line of text shall be yMin above the bottom of the text box.

Horizontal - end

When HorizontalJustification = end (right aligned text), the origin of the last (right most) character shall be charSetWidth from the right side of the text box. As with Horizontal - start, certain characters may extend beyond the right hand edge of the text object.

1. Where text objects are enclosed by other graphics (e.g. a rectangle) it is the author's responsibility to size the objects if they wish to avoid text affecting an enclosing object.

5.4 Text Objects

All objects using the Text class shall use the character coding described in “Character encoding” on page 45 and the fonts described in “Fonts” on page 46.

5.4.1 Text mark-up

See “Text and HyperText encoding” on page 112.

5.4.1.1 White Space Characters

Certain non-printing characters have special meaning. These are identified in Table 23.

UTF8 Value(s)	Name	Meaning
0x09	Tab	See “Tabulation” on page 52
0x0D	Carriage Return	Causes the text flow to break. The origin for the next character to be rendered moves to a new baseline “linespace” below that just rendered. The horizontal position of the next line will depend on the setting of HorizontalJustification.
0x20	Space	Spaces text by the width defined for the space character. When a text object has the TextWrapping attribute set to “true” lines may be broken at a space. See “Line breaking” on page 51.
0xC2 0xA0	Non-breaking space	Identical spacing characteristics to 0x20 but is not seen as word boundary for deciding a position to break a line of text. (0xC2A0 is the UTF-8 representation of 0x00A0)

Table 23. Special characters

5.4.1.2 Format Control Mark-up

Within text object mark-up codes can be used to control the presentation of text. The sequence in Table 24 marks the start of some marked-up text. For each “start of mark-up” a corresponding “end of mark-up” is defined. The byte sequence for the “end of mark-up” is illustrated in Table 25.

	bits	value	note
`start_of_markup`	8	0x1B	Escape
`markup_start_identifier`	8	0x40-0x5E	‘@’ to ‘^’
`parameters_length`	8	N	
`for( i=0; i<N; i++ ) {`			
`  parameter_byte`	8	0x00..0xFF	
`}`			

Table 24. General format for start of text mark-up

	bits	value	note
`end_of_markup`	8	0x1B	Escape
`markup_end_identifier`	8	0x60-0x7E	“‘’ to ‘~’

Table 25. General format for end of text mark-up

start mark-up	end mark-up	description
0x1B 0x42 0x00	0x1B 0x42	Applies 'bold' style to the text enclosed
0x1B 0x43 0x04 0xrr 0xgg 0xbb 0xtt	0x1B 0x43	Applies colour to the text enclosed. 0xrr specifies the red intensity, 0xgg the green, 0xbb the blue and 0xtt the transparency.

Table 26. Text object mark-up codes

5.4.1.3 Future compatibility

Compatible extensions to the set of mark-up codes may be defined in future profiles. For each the markup_end_identifier will be 32 (0x20) greater than the markup_start_identifier. Engines should ignore unrecognised mark-up and should display any text enclosed within an unrecognised mark-up.

5.4.2 Control of text flow

Required flow modes

Receivers shall implement at least the required set of text flow modes identified in Table 30.

Attribute	Required Values	Optional Values	Notes
HorizontalJustification	start, end, centre	justified	See "Reproducing Justification" on page 53
VerticalJustification	start, end, centre	justified	
LineOrientation	horizontal	vertical	
StartCorner	upper-left	upper-right, lower-left, lower-right	
TextWrapping	true, false		See "Line breaking" on page 51

Table 27. Required set of text flow modes

5.5 HyperText Objects

In addition to the mark-up codes identified for Text objects in "Text mark-up" on page 54 HyperText objects can also include the mark-up in Table 28. See "Text and HyperText encoding" on page 112.

start mark-up	end mark-up	description
0x1B 0x41 0xnn tag_bytes	0x1B 0x41	Associates the OctetString tag_bytes with the Anchor text enclosed between the start mark-up and the end mark-up. 0xnn is the length of the tag_bytes OctetString.

Table 28. Additional HyperText mark-up codes

5.5.1 HyperText anchors

After the application transfers the focus of user interaction to a HyperText object (by using the SetInteractionStatus action) the engine is responsible for managing interaction with the HyperText object.

The OctetString tag_bytes is the "tag" of the anchor as defined by MHEG-5. When the anchor "fires" the tag_bytes are returned as the associated data of the AnchorFired event and are placed in the LastAnchorFired internal attribute.

The engine is responsible for:

- the movement of the user focus amongst the anchors in the HyperText in response to user input
- providing feedback to the user as their focus moves
- allowing the user to select an anchor

The exact behaviour during this interaction is receiver dependant. However, the following recommendations are made:

- links shall be fired by the user activating the “Select” user interface function
- interaction with the HyperText object shall be terminated if the user activates the “Cancel” user interface function

5.6 Rendering Text

This specification does not define the exact method for rendering characters. The following text is provided “for information”. It describes a baseline approach to determining the positions at which to render character bitmaps.

5.6.1 Placing character cells

The position of each character cell is determined using a resolution measure from the start of text (see Figure 14).

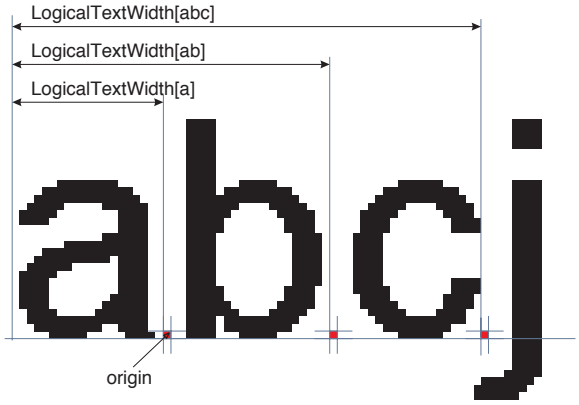


Figure 14. Calculation of character placement

5.6.2 Rendering within character cells

Selecting the font & style

The Font ID in the Physical font record names the font and its style. In *UKEngineProfile1* the strings carried by in the Font ID are “UK1” and “UK1 Bold” for the plain and bold variants of the font.

Selecting the font size

The parameters xppm and yppm in the Bitmap Size Record describe the size of a set of bitmaps in terms of the size of an Em in pixels. In *UKEngineProfile1* yppm shall match the font size requested in the font attributes of the text object. The parameter xppm can be ignored[1].

1. xppm would only be relevant if bitmaps optimised for different display aspect ratios were employed.

Rendering the character

When rendering a simple binary bitmap character the position of the character cell is quantised from the high resolution measure to the pixel map of the display[1]. Parameters from the "Bitmap glyph program string" such as xPos and yPos determine the placement of the character bitmap within the character cell.

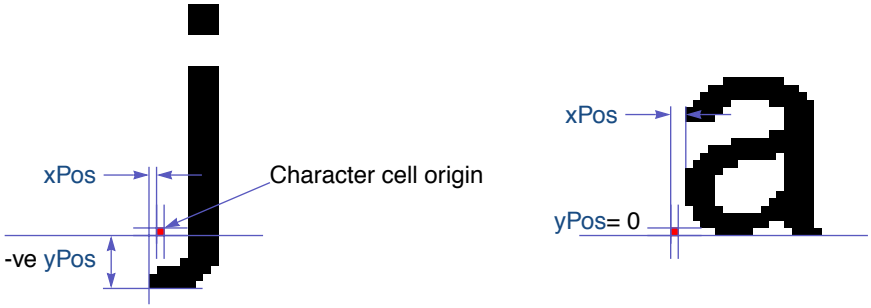


Figure 15. Placing a bitmap within the character cell

The imageData in the Bitmap glyph program string is a compressed pixel map for the character. Each decompressed pixel corresponds exactly to a display pixel.

5.6.2.1 Rendering technology

No restriction is placed on the rendering technology used in a receiver provided that it achieves the deterministic text flow characteristics described previously.

5.7 Entry Fields

All receivers shall provide entry field input for the following character repertoires:

InputType	Set of characters	Comment
numeric	0 - 9 (UTF-8 coded Unicode 0x30 to 0x39)	
alpha		not supported
any		
listed		

Table 29. Characters supported by EntryFields

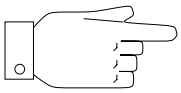
1. More sophisticated implementations may use antialiasing to implement sub-pixel positioning.

5.8 Character repertoire

The set of characters identified in Table 30 is under review to see if it is possible to reduce the number of character shapes that receivers have to store. The candidates for possible exclusion are emphasised in the table.

Unicode value	Glyph	Unicode Name For Character	6937-2	8859-1	Combining character equivalent
0000 to 0008		Non printing control codes			
0009		Horizontal Tabulation	09	09	
000A to 000C					
000D		Carriage Return	0D	0D	
000E to 000F					
0020		Space	20	20	
0021	!	Exclamation Mark	21	21	
0022	"	Quotation Mark	22	22	
0023	#	Number Sign	23	23	
0024	$	Dollar Sign	A4	24	
0025	%	Percent Sign	25	25	
0026	&	Ampersand	26	26	
0027	'	Apostrophe	27	27	
0028	(	Left Parenthesis	28	28	
0029	)	Right Parenthesis	29	29	
002A	*	Asterisk	2A	2A	
002B	+	Plus Sign	2B	2B	
002C	,	Comma	2C	2C	
002D	-	Hyphen-minus	2D	2D	
002E	.	Full Stop	2E	2E	
002F	/	Solidus	2F	2F	
0030	0	Digit Zero	30	30	
0031	1	Digit One	31	31	
0032	2	Digit Two	32	32	
0033	3	Digit Three	33	33	
0034	4	Digit Four	34	34	
0035	5	Digit Five	35	35	
0036	6	Digit Six	36	36	
0037	7	Digit Seven	37	37	
0038	8	Digit Eight	38	38	
0039	9	Digit Nine	39	39	
003A	:	Colon	3A	3A	
003B	;	Semicolon	3B	3B	
003C	<	Less-than Sign	3C	3C	
003D	=	Equals Sign	3D	3D	
003E	>	Greater-than Sign	3E	3E	
003F	?	Question Mark	3F	3F	
0040	@	Commercial At	40	40	
0041	A	Latin Capital Letter A	41	41	
0042	B	Latin Capital Letter B	42	42	
0043	C	Latin Capital Letter C	43	43	

Table 30. Minimum set of characters supported by the engine (Sheet 1 of 6)



Characters highlighted as shown are candidates for exclusion. Service providers should provide comment . Receiver implementers should assume that the full table is required until notified otherwise.

Unicode value	Glyph	Unicode Name For Character	6937-2	8859-1	Combining character equivalent
0044	D	Latin Capital Letter D	44	44	
0045	E	Latin Capital Letter E	45	45	
0046	F	Latin Capital Letter F	46	46	
0047	G	Latin Capital Letter G	47	47	
0048	H	Latin Capital Letter H	48	48	
0049	I	Latin Capital Letter I	49	49	
004A	J	Latin Capital Letter J	4A	4A	
004B	K	Latin Capital Letter K	4B	4B	
004C	L	Latin Capital Letter L	4C	4C	
004D	M	Latin Capital Letter M	4D	4D	
004E	N	Latin Capital Letter N	4E	4E	
004F	O	Latin Capital Letter O	4F	4F	
0050	P	Latin Capital Letter P	50	50	
0051	Q	Latin Capital Letter Q	51	51	
0052	R	Latin Capital Letter R	52	52	
0053	S	Latin Capital Letter S	53	53	
0054	T	Latin Capital Letter T	54	54	
0055	U	Latin Capital Letter U	55	55	
0056	V	Latin Capital Letter V	56	56	
0057	W	Latin Capital Letter W	57	57	
0058	X	Latin Capital Letter X	58	58	
0059	Y	Latin Capital Letter Y	59	59	
005A	Z	Latin Capital Letter Z	5A	5A	
005B	[	Left Square Bracket	5B	5B	
005C	\	Reverse Solidus	5C	5C	
005D	]	Right Square Bracket	5D	5D	
005E	^	Circumflex Accent	5E	5E	
005F	_	Low Line	5F	5F	
0060	`	Grave Accent	60	60	
0061	a	Latin Small Letter A	61	61	
0062	b	Latin Small Letter B	62	62	
0063	c	Latin Small Letter C	63	63	
0064	d	Latin Small Letter D	64	64	
0065	e	Latin Small Letter E	65	65	
0066	f	Latin Small Letter F	66	66	
0067	g	Latin Small Letter G	67	67	
0068	h	Latin Small Letter H	68	68	
0069	i	Latin Small Letter I	69	69	
006A	j	Latin Small Letter J	6A	6A	
006B	k	Latin Small Letter K	6B	6B	
006C	l	Latin Small Letter L	6C	6C	
006D	m	Latin Small Letter M	6D	6D	
006E	n	Latin Small Letter N	6E	6E	
006F	o	Latin Small Letter O	6F	6F	
0070	p	Latin Small Letter P	70	70	
0071	q	Latin Small Letter Q	71	71	
0072	r	Latin Small Letter R	72	72	
0073	s	Latin Small Letter S	73	73	

Table 30. Minimum set of characters supported by the engine (Sheet 2 of 6)

Unicode value	Glyph	Unicode Name For Character	6937-2	8859-1	Combining character equivalent
0074	t	Latin Small Letter T	74	74	
0075	u	Latin Small Letter U	75	75	
0076	v	Latin Small Letter V	76	76	
0077	w	Latin Small Letter W	77	77	
0078	x	Latin Small Letter X	78	78	
0079	y	Latin Small Letter Y	79	79	
007A	z	Latin Small Letter Z	7A	7A	
007B	{	Left Curly Bracket	7B	7B	
007C	|	Vertical Line	7C	7C	
007D	}	Right Curly Bracket	7D	7D	
007E	~	Tilde	7E	7E	
0080 to 009F		Non printing control codes			
00A0		No-break Space	A0	A0	
00A1	¡	Inverted Exclamation Mark	A1	A1	
00A2	¢	Cent Sign	A2	A2	
00A3	£	Pound Sign	A3	A3	
00A4		Currency Sign	24	A4	
00A5	¥	Yen Sign	A5	A5	
00A6	¦	Broken Bar	D7	A6	
00A7	§	Section Sign	A7	A7	
00A8	¨	Diaeresis		A8	
00A9	©	Copyright Sign	D3	A9	
00AA	ª	Feminine Ordinal Indicator	E3	AA	
00AB	«	Left-pointing Double Angle Quotation Mark	AB	AB	
00AC	¬	Not Sign	D6	AC	
00AD		Soft Hyphen	FF	AD	
00AE	®	Registered Sign	D2	AE	
00AF	¯	Macron		AF	
00B0	°	Degree Sign	B0	B0	
00B1	±	Plus-minus Sign	B1	B1	
00B2	2	Superscript Two	B2	B2	
00B3	3	Superscript Three	B3	B3	
00B4	´	Acute Accent		B4	
00B5	µ	Micro Sign	B5	B5	
00B6	¶	Pilcrow Sign	B6	B6	
00B7	·	Middle Dot	B7	B7	
00B8	¸	Cedilla		B8	
00B9	1	Superscript One	D1	B9	
00BA	º	Masculine Ordinal Indicator	EB	BA	
00BB	»	Right-pointing Double Angle Quotation Mark	BB	BB	
00BC	¼	Vulgar Fraction One Quarter	BC	BC	
00BD	½	Vulgar Fraction One Half	BD	BD	
00BE	¾	Vulgar Fraction Three Quarters	BE	BE	
00BF	¿	Inverted Question Mark	BF	BF	

Table 30. Minimum set of characters supported by the engine (Sheet 3 of 6)

Unicode value	Glyph	Unicode Name For Character	6937-2	8859-1	Combining character equivalent
00C0	À	Latin Capital Letter A With Grave		C0	0041 0300
00C1	Á	Latin Capital Letter A With Acute		C1	0041 0301
00C2	Â	Latin Capital Letter A With Circumflex		C2	0041 0302
00C3	Ã	Latin Capital Letter A With Tilde		C3	0041 0303
00C4	Ä	Latin Capital Letter A With Diaeresis		C4	0041 0308
00C5	Å	Latin Capital Letter A With Ring Above		C5	0041 030A
00C6	Æ	Latin Capital Letter Ae	E1	C6	
00C7	Ç	Latin Capital Letter C With Cedilla		C7	0043 0327
00C8	È	Latin Capital Letter E With Grave		C8	0045 0300
00C9	É	Latin Capital Letter E With Acute		C9	0045 0301
00CA	Ê	Latin Capital Letter E With Circumflex		CA	0045 0302
00CB	Ë	Latin Capital Letter E With Diaeresis		CB	0045 0308
00CC	Ì	Latin Capital Letter I With Grave		CC	0049 0300
00CD	Í	Latin Capital Letter I With Acute		CD	0049 0301
00CE	Î	Latin Capital Letter I With Circumflex		CE	0049 0302
00CF	Ï	Latin Capital Letter I With Diaeresis		CF	0049 0308
00D0	Ð	Latin Capital Letter Eth	E2	D0	
00D1	Ñ	Latin Capital Letter N With Tilde		D1	004E 0303
00D2	Ò	Latin Capital Letter O With Grave		D2	004F 0300
00D3	Ó	Latin Capital Letter O With Acute		D3	004F 0301
00D4	Ô	Latin Capital Letter O With Circumflex		D4	004F 0302
00D5	Õ	Latin Capital Letter O With Tilde		D5	004F 0303
00D6	Ö	Latin Capital Letter O With Diaeresis		D6	004F 0308
00D7	×	Multiplication Sign	B4	D7	
00D8	Ø	Latin Capital Letter O With Stroke	E9	D8	
00D9	Ù	Latin Capital Letter U With Grave		D9	0055 0300
00DA	Ú	Latin Capital Letter U With Acute		DA	0055 0301
00DB	Û	Latin Capital Letter U With Circumflex		DB	0055 0302
00DC	Ü	Latin Capital Letter U With Diaeresis		DC	0055 0308
00DD	Ý	Latin Capital Letter Y With Acute		DD	0059 0301
00DE	Þ	Latin Capital Letter Thorn	EC	DE	
00DF	ß	Latin Small Letter Sharp S	FB	DF	
00E0	à	Latin Small Letter A With Grave		E0	0061 0300
00E1	á	Latin Small Letter A With Acute		E1	0061 0301
00E2	â	Latin Small Letter A With Circumflex		E2	0061 0302
00E3	ã	Latin Small Letter A With Tilde		E3	0061 0303
00E4	ä	Latin Small Letter A With Diaeresis		E4	0061 0308
00E5	å	Latin Small Letter A With Ring Above		E5	0061 030A
00E6	æ	Latin Small Letter Ae	F1	E6	
00E7	ç	Latin Small Letter C With Cedilla		E7	0063 0327
00E8	è	Latin Small Letter E With Grave		E8	0065 0300
00E9	é	Latin Small Letter E With Acute		E9	0065 0301
00EA	ê	Latin Small Letter E With Circumflex		EA	0065 0302
00EB	ë	Latin Small Letter E With Diaeresis		EB	0065 0308
00EC	ì	Latin Small Letter I With Grave		EC	0069 0300
00ED	í	Latin Small Letter I With Acute		ED	0069 0301
00EE	î	Latin Small Letter I With Circumflex		EE	0069 0302

Table 30. Minimum set of characters supported by the engine (Sheet 4 of 6)

Unicode value	Glyph	Unicode Name For Character	6937-2	8859-1	Combining character equivalent
00EF	ï	Latin Small Letter I With Diaeresis		EF	0069 0308
00F0	ð	Latin Small Letter Eth	F3	F0	
00F1	ñ	Latin Small Letter N With Tilde		F1	006E 0303
00F2	ò	Latin Small Letter O With Grave		F2	006F 0300
00F3	ó	Latin Small Letter O With Acute		F3	006F 0301
00F4	ô	Latin Small Letter O With Circumflex		F4	006F 0302
00F5	õ	Latin Small Letter O With Tilde		F5	006F 0303
00F6	ö	Latin Small Letter O With Diaeresis		F6	006F 0308
00F7	÷	Division Sign	B8	F7	
00F8	ø	Latin Small Letter O With Stroke	F9	F8	
00F9	ù	Latin Small Letter U With Grave		F9	0075 0300
00FA	ú	Latin Small Letter U With Acute		FA	0075 0301
00FB	û	Latin Small Letter U With Circumflex		FB	0075 0302
00FC	ü	Latin Small Letter U With Diaeresis		FC	0075 0308
00FD	ý	Latin Small Letter Y With Acute		FD	0079 0301
00FE	þ	Latin Small Letter Thorn	FC	FE	
00FF	ÿ	Latin Small Letter Y With Diaeresis		FF	0079 0308
0111		Latin Small Letter D With Stroke	F2		
0126		Latin Capital Letter H With Stroke	E4		
0127		Latin Small Letter H With Stroke	F4		
0132		Latin Capital Ligature Ij	E6		
0133		Latin Small Ligature Ij	F6		
013F		Latin Capital Letter L With Middle Dot	E7		
0140		Latin Small Letter L With Middle Dot	F7		
0141		Latin Capital Letter L With Stroke	E8		
0142		Latin Small Letter L With Stroke	F8		
0149		Latin Small Letter N Preceded By Apostrophe	EF		
014A		Latin Capital Letter Eng	EE		
014B		Latin Small Letter Eng	FE		
0152	Œ	Latin Capital Ligature Oe	EA		
0153	œ	Latin Small Ligature Oe	FA		
0166		Latin Capital Letter T With Stroke	ED		
0167		Latin Small Letter T With Stroke	FD		
0174	Ŵ	Latin Capital Letter W With Circumflex			0057 0302
0175	ŵ	Latin Small Letter W With Circumflex			0077 0302
0176	Ŷ	Latin Capital Letter Y With Circumflex			0059 0302
0177	ŷ	Latin Small Letter Y With Circumflex			0079 0302
0178	Ÿ	Latin Capital Letter Y With Diaeresis			0059 0308
0300	`	Combining Grave Accent	C1		
0301	´	Combining Acute Accent	C2		
0302	^	Combining Circumflex Accent	C3		
0303	~	Combining Tilde	C4		
0304	¯	Combining Macron	C5		
0306	˘	Combining Breve	C6		
0307	˙	Combining Dot Above	C7		

Table 30. Minimum set of characters supported by the engine (Sheet 5 of 6)

Unicode value	Glyph	Unicode Name For Character	6937-2	8859-1	Combining character equivalent
0308	¨	Combining Diaeresis	C8		
030A	°	Combining Ring Above	CA		
030B	˝	Combining Double Acute Accent	CD		
030C	ˇ	Combining Caron	CF		
0327	¸	Combining Cedilla	CB		
0328	˛	Combining Ogonek	CE		
0332	_	Combining Low Line	CC		
1E80	Ẁ	Latin Capital Letter W With Grave			0057 0300
1E81	ẁ	Latin Small Letter W With Grave			0077 0300
1E82	Ẃ	Latin Capital Letter W With Acute			0057 0301
1E83	ẃ	Latin Small Letter W With Acute			0077 0301
1E84	Ẅ	Latin Capital Letter W With Diaeresis			0057 0308
1E85	ẅ	Latin Small Letter W With Diaeresis			0077 0308
1EF2	Ỳ	Latin Capital Letter Y With Grave			0059 0300
1EF3	ỳ	Latin Small Letter Y With Grave			0079 0300
2014	—	Em Dash	D0		
2018	‘	Left Single Quotation Mark	A9		
2019	’	Right Single Quotation Mark	B9		
201C	“	Left Double Quotation Mark	AA		
201D	”	Right Double Quotation Mark	BA		
20A0	€E	Euro-currency Sign			
2122	™	Trade Mark Sign	D4		
2126	Ω	Ohm Sign	E0		
212A		Kelvin Sign	F0		
215B		Vulgar Fraction One Eighth	DC		
215C		Vulgar Fraction Three Eighths	DD		
215D		Vulgar Fraction Five Eighths	DE		
215E		Vulgar Fraction Seven Eighths	DF		
2190		Leftwards Arrow	AC		
2191		Upwards Arrow	AD		
2192		Rightwards Arrow	AE		
2193		Downwards Arrow	AF		
2309		Right Ceiling	F5		
2669		Quarter Note	D5		

Table 30. Minimum set of characters supported by the engine (Sheet 6 of 6)

6 Receiver Requirements

6.1 Introduction

This section describes the measures that shall be taken by the receiver to ensure good application behaviour.

6.2 Auto Start Application

Whenever a service that includes an MHEG application as a component is selected the MHEG application shall be launched.

If an MHEG application component is added to a service (e.g. a new event starts that has an associated MHEG application) the MHEG application shall be launched.

How's this done?

The receiver looks continuously for the availability of an MHEG application as part of a service. So:

- when a service is selected, any MHEG application is launched
- if an MHEG application appears in a service that previously didn't have an application, the application will be launched.
- if an MHEG application Quits the receiver shall try to launch a new application

See "Application Identification and Boot" on page 94.

6.3 Auto Kill Application

Whenever the receiver changes channel any executing MHEG application is killed.

I.e. if the user interacts with the receivers navigator functions to change channel the current application will be killed. If the new channel has an application this one will be launched. If the application invokes a channel change (via a resident program), and the channel change is successful, then current application will be killed then replaced by any application associated with the new channel. See "Killing Applications" on page 28.

Invocation of different video and audio stream objects by an application is not considered a channel change. So, an application can "preview" a service before selecting it.

6.4 Auto Kill Stream Decoders

Whenever the receiver changes service all stream decoders are stopped and restarted decoding data from the new service.

I.e. if the user interacts with the receiver's navigator functions to change channel the video, audio and subtitle decoders are all stopped and then restarted on the new service.

6.4.1 Stream inheritance by applications

When an MHEG-5 application is launched the Video, Audio and DVB Subtitling stream decoders continue decoding streams from the service containing the application until the application object's preparation phase is complete.

Except where initially active stream objects have been introduced to continue any or all of the Video, Audio and DVB Subtitling stream decoding, the stream decoders will stop when the application object is activated. **When an application chooses to play components from**

the TV service that was active when it launched there shall be no disruption to the stream decoding.

If the user changes service to a service which includes an MHEG application the stream, the decoders will be stopped and restarted with the "default" streams of the new service in the expectation that the MHEG application will take over control of these streams when it is running.

If, within a service, an MHEG-5 application is *launched* by another MHEG-5 application, then it has the opportunity to "inherit" the condition of the stream decoders of the application that launched it.

Service and component content reference

Applications can be "compiled" with explicit service and component references. This is described elsewhere.

Additionally certain generic references are defined:

- The OctetString "rec://def_service"[1] may be used as the original content specification for a Stream object. "rec://def_service" means the service currently select by the receiver.
- The value **'-1'** may be used as the component tag for a Video, Audio or RTGRaphics object. This means the currently selected media component.

For example, an application object might include:

```
{:Stream 1
  :OrigContent "rec://def_service"
  :Shared "true"
  :Multiplex ( {:Audio 2 :ComponentTag -1 } )
}
```

This allows the TV services default audio to continue without disruption. However, Video and Subtitles will stop at the end of application object preparation.

Component tags for RTGraphics

The MHEG-5 component tag is a 32 bit integer. In all cases the least significant byte of the tag corresponds to the DVB SI component tag defined by the a stream identifier descriptor in the PMT of the program.

For an RTGraphics (DVB Subtitle) stream, except when the component tag is -1, the second and third bytes of the MHEG-5 component tag defined the least significant bytes of the required composition_page_id and ancillary_page_id. See Figure 16.

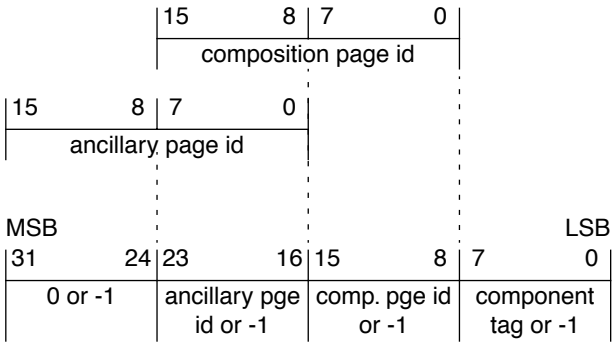


Figure 16. Component tag coding for RTGraphics

1. See "Names within the receiver" on page 101.

6.5 Application Interaction with user Control of Decoders

This section addresses the relationship between user and application control of the stream decoders within the receiver.

6.5.1 Audio Decoder

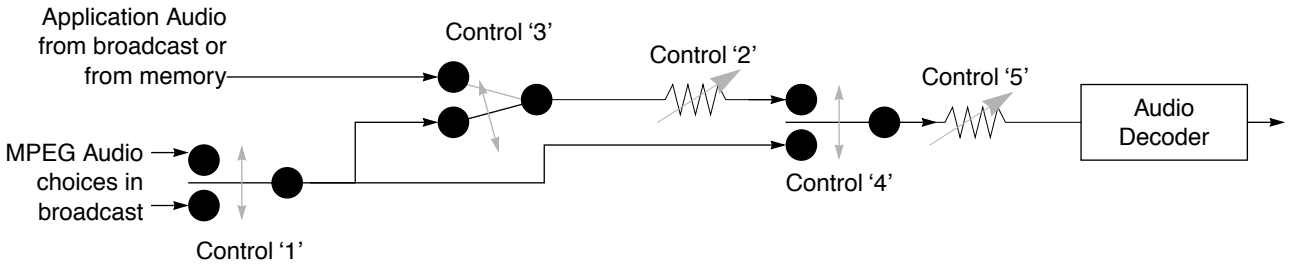


Figure 17. Audio Control - logical model

Controls:

1. This is the user's audio stream preference. Typically this will be implemented as an automatic receiver response to a stored language preference.
2. This is the MHEG application's control of the sound level for any application sourced audio. In *UKEngineProfile1* this control is only required to be on/off.
3. This is the MHEG application's control over whether to replace the programme audio with application sourced audio. See "Stream inheritance by applications" on page 65.
4. This is control is operated automatically by the receiver when an application is executing to give the application certain control of the audio.
5. This is the user's audio volume control operated by the "Mute", "Vol+", & "Vol-" keys on the remote controler.

6.5.2 Subtitle Decoder

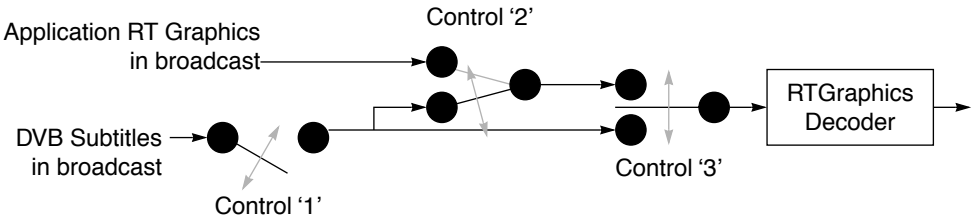


Figure 18. RT Graphics Control - logical model

Controls:

1. The "Subtitles" key on the remote control or a stored user preference for subtitles control this "switch"
2. This is the MHEG application's control over whether to replace the programme subtitling with an application selected subtitle stream. See "Stream inheritance by applications" on page 65.
3. This is control is operated automatically by the receiver when an application is executing to give the application control of the RTGraphics decoder.

6.6 Application impact on stream decoder specification

6.6.1 DVB Subtitle decoder performance

The real-time rendering performance of the DVB Subtitle [2] decoder shall be maintained provided that no part of any visible region within the subtitle page is obscured or transparently overlaid on by any graphics. In these circumstances the real-time performance may degrade in an implementation specific way.

See also "RTGraphics & Overlapping Visibles" on page 42.

6.6.2 Drawing space for RTGraphics

The DVB Subtitle [2] decoder addresses a 720x576 pixel drawing space (i.e. the same drawing space as that used by the MHEG engine). Therefore the position of the RTGraphics box shall always be (0, 0) and its size shall always be (720, 576).

6.6.3 Video decoder performance

The real-time performance of the MPEG Video [15] decoder shall be maintained regardless of the number of visibles that obscure it.

6.6.4 Video Scaling

Video Scaling Reference Model

Logically the behaviour is as follows:

- The receiver's video decoder delivers "full-screen" video (720x576) to the MHEG engine. The ScaleVideo(X,Y) action can be used to scale this to either 100% of the screen area {by using (X,Y) = (720,576)} or 25% of the screen area {by using (X,Y) = (360,288)}. The default scale for a video object is 100%.
- The resolution at which the MPEG video [15] is encoded is transparent to the application. For example, if video is encoded at 352x288 resolution (which ETR 154 [6] regards as a "full screen" format) ScaleVideo(720,576) is logically 100% scaling but invokes the standard x2 horizontal and vertical scaling within the receiver. Similarly, ScaleVideo(360,288) is logically 25% scaling but is actually implemented by suppressing the receiver's normal upsampling response to subsampled video.

The set of video scaling factors that the receiver is required to support is the product of the logical scaled video sizes (100% & 25%) and the full-screen video formats recognised within the "25 Hz SDTV" profile of ETR 154 [6] (720x576, 544 x576, 480x576, 352x576, 352x288).

The ScaleVideo(X,Y) action has no affect on any DVB Subtitles associated with the video.

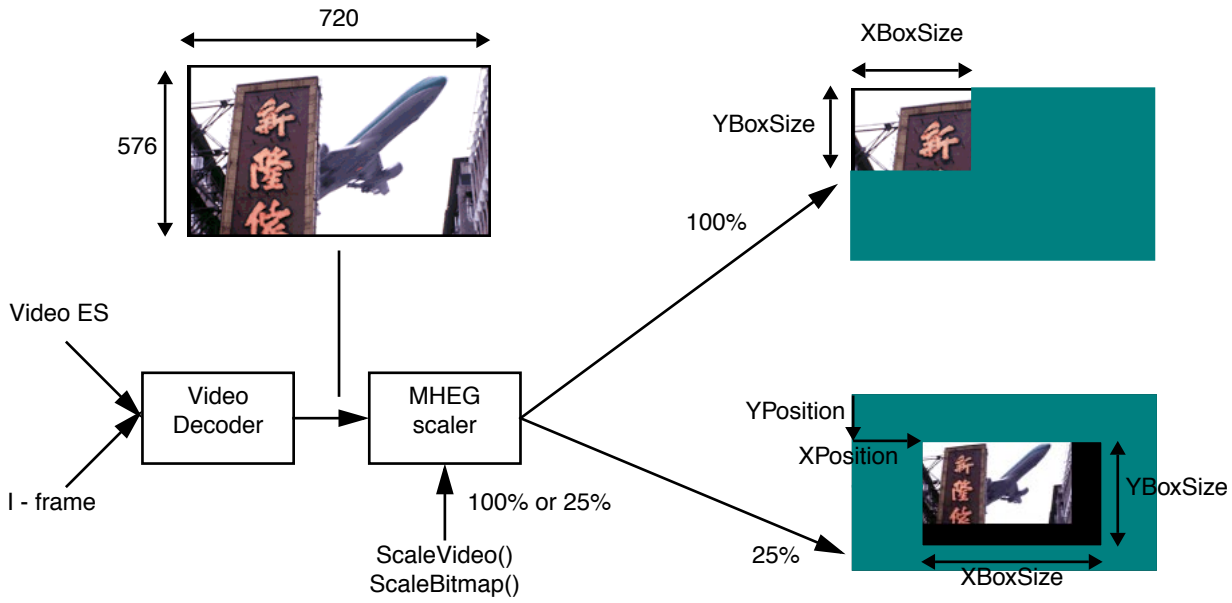


Figure 19. Behaviour of video scaling

Subtitles and scaled video

The ScaleVideo action does NOT scale subtitle graphics (delivered as DVB Subtitle [2] data).

MPEG I-frame scaling

Scaling of MPEG I-frame bitmaps is handled exactly as MPEG video.

6.6.4.1 Video object placement

A video may be placed anywhere within the 720x576 display area subject to the following restrictions:

- The left edge of the video may be offset by any number of pixels from the left edge of the display.
- The top line of the video shall be offset an even number of display lines from the origin of the display area.
- The entirety of the video shall be enclosed within the display area.

  As a consequence, the position of the video object shall be (0,0) when the scaling is 100% size. Non-zero positions shall only be used if the video is scaled to <100%.

Receivers shall be able to display all video windows that meet these criteria.

MPEG I-frame placement

Placement of MPEG I-frame bitmaps is handled exactly as MPEG video.

6.7 Persistent storage

The engine shall provide "persistent" storage for 1024 bytes of data. Storage may be implemented in RAM and may be lost when the receiver is in stand-by or off.

The file name used to access this storage shall be of the form "ram://<name>". It is the responsibility of the broadcasters to arrange a practise for the use of <name> such that there is no accidental collision of file names.

The persistent storage provides a "leaky" file system. Each new file written to the file system pushes out sufficient old files to make room (unless the file over writes a file with the same name).

1. Only the data is stored (not type information)
2. The sizes of the variables when stored are defined in "Limitations on Standard Data-Types" on page 28
3. The set of values is stored in the order they are enumerated in the ASN.1 DER encoding of the StorePersistent InVariables list.
4. The behaviour of ReadPersistent() is undefined unless the set of values is enumerated in the same order as the StorePersistent() that created them.

6.7.1 Storage of file names

The <name> part of the file name "ram://<name>" shall be 8 bytes long. The receiver shall provide storage for at least 32 such file names associated with the "ram://" persistent store.

See "File names in persistent storage" on page 111.

6.7.2 Possible uses of persistent storage (informative)

Start-up Scene reference

An application can store the ObjectReference of a specific scene within another application prior to launching that application. The launched application can retrieve the ObjectReference and TransitionTo the specified scene.

This allows transitions between scenes in two different application. Without this applications could only be launched to start at their root scene.

Return Application reference

An application can store an ObjectReference to itself before launching another application. This allows the launched application to return to the "calling" application (although the Spawn action might be a better way of doing this).

Return Application Start-up Scene reference

A scene can store an ObjectReference to itself before launching another application. This allows the "calling" application to TransitionTo the "calling" scene if it is restarted.

6.8 Reference Decoder Model

A detailed decoder reference will be produced in the future.

Receivers will be expected to implement engines conforming to the reference decoder model with 1 MB of RAM. The reference decoder model will include the storage required to implement the following:

- The content required by the currently executing application.
- The variables, state etc. used by the currently executing application.
- Representation(s) of the application code.
- Persistent file store and directory structures (See "Persistent storage" on page 69).
- The application identifier stack (See "Application Stacking" on page 71).
- Buffers required to receive data from the network (See "Mapping of objects to data carousel modules" on page 89).
- Work space required by decoders used solely by the MHEG-5 engine (e.g. PNG decoding).

6.9 Application Stacking

To support application stacking receivers shall implement an application identifier stack capable of holding at least 5 GroupIdentifiers. See "GroupIndentifier" on page 28. See "Mapping Rules for GroupIdentifier" on page 102.

The application identifier stack behaves as a simple push down stack. When full, a Spawn action will push the oldest GroupIdentifier off the bottom of the stack. If an application terminates and the stack is empty then the receiver shall launch the default application for the currently selected service.

The application identifier stack shall be reset each time there is a service change.

7 Object carousel Profile for UK DTT

7.1 Introduction

The broadcast applications are transmitted using the DSM-CC User-to-User Object Carousels.

This specification is based on the following specifications:

- ISO/IEC 13818-1 [14] - MPEG 2 systems
- ISO/IEC 13818-6 [17] - DSM-CC
- Draft EN 301 192 [5] - DVB specification for data broadcasting
- SI-DAT 382 Rev. 4 [21] - Implementation Guidelines for Databroadcasting

With the constraints and extensions described here.

Work in progress

The section "Application Identification and Boot" on page 94 represents the current state of thinking. The proposal is not yet complete and may be modified following discussions within the DVB.

7.2 Object Carousel Profile

In the following chapter, the message structures of the Object carousels are introduced with associated additional restrictions. Each section contains a table specifying the restrictions on the usage of the fields. The table also indicates the source for these restrictions: the DSM-CC standard, DVB guidelines or a specific restriction for the UK DTT.

For the object carousel messages, also the message syntax is included. In the syntax tables grey shading indicates parts that the broadcaster may put in, but a receiver compliant with this profile may ignore.

7.2.1 DSM-CC Sections

All object carousels messages are transmitted using DSM-CC section format. The DSM-CC Section format is defined in chapter 9.2 of the DSM-CC specification.

The DSM-CC standard provides an option to use either a CRC32 or a checksum for detecting bit errors. For UK DTT, we make the following restriction:

Field	Restrictions	Source
section_syntax_indicator	1 (indicating the use of the CRC32)	UK DTT

Table 31. Restrictions on DSM-CC Section format

The maximum section length is 4096 bytes for all types of sections used in Object Carousels. The section overhead is 12 bytes, leaving a maximum of 4084 bytes of payload per section.

7.2.2 Data Carousel

7.2.2.1 General

The definitions in Table 32 apply to both the dsmccDownloadDataHeader and the similar dsmccMessageHeader.

Field	Restrictions	Source
TransactionId	See "Assignment and use of transactionId values" on page 87	UK DTT
AdaptationLength	The adaptionLength shall be 0, indicating no dsmccAdaptationHeader. The adaptation header is not needed in the broadcast scenario.	UK DTT

Table 32. Restrictions on DSM-CC DownloadData and Message headers

7.2.2.2 DownloadInfoIndication

The DownloadInfoIndication is a message that describes a set of modules and gives the necessary parameters to locate the module and retrieve it.

Field	Restrictions	Source
blockSize	maximum size 4066 (max. section payload - DDB-header size (18)) The recommended blockSize is 4066.	DSM-CC (UK DTT rec.)
windowSize	0 (not used for Object Carousels)	DSM-CC
ackPeriod	0 (not used for Object Carousels)	DSM-CC
tCDownloadWindow	0 (not used for Object Carousels)	DSM-CC
tCDownloadScenario	0 (not used for Object Carousels)	DSM-CC
compatibilityDescriptor(): compatibilityDescriptorLength	0 (no compatibility descriptor for Object Carousels)	DSM-CC
PrivateDataLength	The receiver may ignore the possible contents of the privateData field	DVB

Table 33. Restrictions on the DII

7.2.2.3 DownloadServerInitiate

The DownloadServerInitiate is used in the case of object carousels to provide the object reference to the service gateway (i.e. root directory) of the object carousel.

Field	Restrictions	Source
compatibilityDescriptor(): compatibilityDescriptorLength	0 (no compatibility descriptor for Object Carousels)	DSM-CC
privateData	Contains the ServiceGatewayInfo structure	DSM-CC
serverId	Shall be set to 20 bytes with the value of 0xFF	DVB / UK DTT

Table 34. Restrictions on DSI

7.2.2.4 ModuleInfo

The moduleInfo structure is placed in the moduleInfo field of the DownloadInfoIndication of the data carousel. It contains the information needed to locate the module.

Field	Restrictions	Source
BIOP::ModuleInfo::Taps	The first tap shall have the "use" value 0x0017 (BIOP_OBJECT_USE). The id and selector fields are not used and the receiver may ignore them. The receiver may ignore possible other taps in the list.	DVB
BIOP::ModuleInfo:: UserInfo	The userInfo field contains a loop of descriptors. These are specified in the DVB Data Broadcasting standard. The receiver shall support especially the compressed_module_descriptor (tag 0x09) used to signal that the module is transmitted in compressed form.	DVB / UK DTT

Table 35. Restrictions on the DII moduleInfo field

Syntax	bits	Type	Value	Comment
BIOP::ModuleInfo() {				
moduleTimeOut	32	uimsbf	+	
blockTimeOut	32	uimsbf	+	
minBlockTime	32	uimsbf	+	
taps_count	8	uimsbf	N1	≥ 1
{				
use	16	uimsbf	0x0017	BIOP_OBJECT_USE
id	16	uimsbf	0x0000	user private
assocTag	16	uimsbf	+	
selector_length	8	uimsbf	0x00	
}				
for (j=1; j<N1; j++) {				Possible additional taps that may be ignored by receivers.
use	16	uimsbf	+	
id	16	uimsbf	+	
assocTag	16	uimsbf	+	
selector_length	8	uimsbf	N2	
for (j=0; j<N2; j++) {				
selector_data	8	uimsbf	+	
}				
}				
userInfoLength	8	uimsbf	N3	
for (k=0; k<N3; j++) {				
userInfo_data	8	uimsbf	+	
}				
}				

Table 36. BIOP::ModuleInfo syntax

7.2.2.5 ServiceGatewayInfo

The ServiceGatewayInfo structure is carried in the DownloadServerInitiate message and provides the object reference to the Service Gateway object.

Field	Restrictions	Source
BIOP:: ServiceGatewayInfo:: downloadTaps	The receiver may ignore the downloadTap list.	UK DTT
BIOP:: ServiceGatewayInfo:: serviceContextList	The receiver may ignore the service context list.	UK DTT
BIOP:: ServiceGatewayInfo:: UserInfo	The user info field contains a loop of descriptors. See "Application Identification and Boot" on page 94.	UK DTT

Table 37. Restrictions on the ServiceGatewayInfo

Syntax	bits	Type	Value	Comment
`ServiceGatewayInfo(){`				
`  IOP::IOR()`			+	See Table 48 on page 83
`  downloadTaps_count`	8	uimsbf	N1	software download Taps
`  for (i=0; i<N1; i++) {`				
`    Tap()`	8	uimsbf	+	
`  }`				
`  serviceContextList_count`	8	uimsbf	N2	serviceContextList
`  for (i=0; i<N2; i++) {`				
`    serviceContextList_data`	8	uimsbf	+	
`  }`				
`  userInfoLength`	16	uimsbf	N3	user info
`  for (i=0; i<N3; i++) {`				
`     userInfo_data`	8	uimsbf	+	See Table 60 on page 98.
`  }`				
`}`				

Table 38. ServiceGatewayInfo() syntax

7.2.3 The Object Carousel

7.2.3.1 BIOP Generic Object Message

The BIOP Generic Object Message is a common structure used by all the BIOP (Broadcast Inter-ORB Protocol) messages.

Field	Restrictions	Source
MessageHeader:: byte_order	0 (indicating big-endian byte order)	DVB
MessageSubHeader:: objectKey	Maximum length of the key shall be four bytes.	DVB
MessageSubHeader:: objectKind	The short three-letter aliases shall be used, plus the null-terminator.	DVB
MessageSubHeader:: serviceContextList:	The receiver may ignore the service context list	UK DTT
Access attributes	Access attributes are not transmitted in object carousels	DSM-CC

Table 39. Restrictions on the BIOP Generic Object Message

7.2.3.2 BIOP FileMessage

The BIOP FileMessage is used for carrying file objects.

Field	Restrictions	Source
MessageSubHeader:: ObjectInfo	The first 8 bytes of the ObjectInfo contain the ContentSize attribute. The receiver may skip the possible bytes after the ContentSize attribute.	UK DTT
MessageSubHeader:: ServiceContextList	The receiver may skip the possible serviceContextList structures.	UK DTT

Table 40. Restrictions on the BIOP File Message

Syntax	bits	Type	Value	Comment
BIOP::FileMessage() {				
magic	4x8	uimsbf	0x42494F50	"BIOP"
biop_version.major	8	uimsbf	0x01	BIOP major version 1
biop_version.minor	8	uimsbf	0x00	BIOP minor version 0
byte_order	8	uimsbf	0x00	Big endian byte ordering
message_type	8	uimsbf	0x00	
message_size	32	uimsbf	*	
objectKey_length	8	uimsbf	N1	<= 4
for (i=0; i<N1; i++) {				
objectKey_data	8	uimsbf	+	
}				
objectKind_length	32	uimsbf	0x00000004	
objectKind_data	4x8	uimsbf	0x66696C00	"fil" type_id alias
objectInfo_length	16	uimsbf	N2	
DSM::File::ContentSize	64	uimsbf	+	objectInfo
for (i=0; i<N2-8; i++) {				
objectInfo_data	8	uimsbf	+	
}				
serviceContextList_count	8	uimsbf	N3	serviceContextList
for (i=0; i<N3; i++) {				
serviceContextList_data	8	uimsbf	+	
}				
messageBody_length	32	uimsbf	*	
content_length	32	uimsbf	N4	
for (i=0; i<N4; i++) {				
content_byte	8	uimsbf	+	actual file content
}				
}				

Table 41. BIOP::FileMessage syntax

7.2.3.3 BIOP DirectoryMessage

The BIOP DirectoryMessage is used for carrying the directory objects.

Field	Restrictions	Source
MessageSubHeader:: ObjectInfo	The receiver may skip the possible bytes in the objectInfo field.	UK DTT
MessageSubHeader:: ServiceContextList	The receiver may skip the possible serviceContextList structures.	UK DTT
BIOP::Name	The name shall contain exactly one NameComponent.	UK DTT
BIOP::Binding:: bindingType	Either “ncontext” (in the case of a Directory object) or “nobject” (in the case of a File or a Stream object). Binding type “composite” shall not be used.	DVB

Table 42. Restrictions on the BIOP Directory Message

Syntax	bits	Type	Value	Comment
BIOP::DirectoryMessage() {				
magic	4x8	uimsbf	0x42494F50	“BIOP”
biop_version.major	8	uimsbf	0x01	BIOP major version 1
biop_version.minor	8	uimsbf	0x00	BIOP minor version 0
byte_order	8	uimsbf	0x00	big endian byte ordering
message_type	8	uimsbf	0x00	
message_size	32	uimsbf	*	
objectKey_length	8	uimsbf	N1	<= 4
for (i=0; i<N1; i++) {				
objectKey_data	8	uimsbf	+	
}				
objectKind_length	32	uimsbf	0x00000004	
objectKind_data	4x8	uimsbf	0x64697200	“dir” type_id alias
objectInfo_length	16	uimsbf	N2	objectInfo
for (i=0; i<N2; i++) {				
objectInfo_data	8	uimsbf	+	
}				
serviceContextList_count	8	uimsbf	0	serviceContextList
for (i=0; i<N3; i++) {				
serviceContextList_data	8	uimsbf	+	
}				
messageBody_length	32	uimsbf	*	
bindings_count	16	uimsbf	N4	

Table 43. BIOP::DirectoryMessage syntax (Sheet 1 of 2)

Syntax	bits	Type	Value	Comment
for (i=0; i<N4; i++) {				Binding
BIOP::Name() {				
nameComponents_count	8	uimsbf	1	
for (i=0; i<N5; i++) {				
id_length	8	uimsbf	N6	NameComponent id
for (j=0; j<N6; j++) {				
id_data	8	uimsbf	+	
}				
kind_length	8	uimsbf	N7	NameComponent kind
for (j=0; j<N7; j++) {				
kind_data	8	uimsbf	+	as type_id (see Table 4-4)
}				
}				
}				
BindingType	8	uimsbf	+	0x01 for nobject 0x02 for ncontext
IOP::IOR()			+	objectRef see Table 48 on page 83
objectInfo_length	16	uimsbf	0	
for (j=0; j<N8; j++) {				
objectInfo_byte	8	uimsbf	+	
}				
}				
}				

Table 43. BIOP::DirectoryMessage syntax (Sheet 2 of 2)

7.2.3.4 BIOP StreamMessage

There are two versions of stream messages. The BIOP StreamMessage is used for carrying the stream objects that don't use DSM-CC Stream events. The BIOP StreamEventMessage is used for carrying stream objects that include a stream carrying the DSM-CC Stream events.

Field	Restrictions	Source
MessageSubHeader:: ObjectInfo	The ObjectInfo field contains the Stream Info structure and optionally other data after the Stream Info structure. The receiver may ignore the possible other data after the Stream Info structure.	UK DTT
MessageSubHeader:: ServiceContextList	The receiver may skip the possible serviceContextList structures.	UK DTT

Table 44. Restrictions on the BIOP Stream Message

Syntax	bits	Type	Value	Comment
BIOP::StreamMessage() {				
magic	4x8	uimsbf	0x42494F50	"BIOP"
biop_version.major	8	uimsbf	0x01	BIOP major version 1
biop_version.minor	8	uimsbf	0x00	BIOP minor version 0
byte_order	8	uimsbf	0x00	big endian byte ordering
message_type	8	uimsbf	0x00	
message_size	32	uimsbf	*	
objectKey_length	8	uimsbf	N1	<= 4
for (i=0; i<N1; i++) {				
objectKey_data	8	uimsbf	+	

Table 45. BIOP::StreamMessage syntax (Sheet 1 of 2)

Syntax	bits	Type	Value	Comment
}				
objectKind_length	32	uimsbf	0x00000004	
objectKind_data	8	uimsbf	0x73747200	“str” type_id alias
objectInfo_length	16	uimsbf	N6	
DSM::Stream::Info_T {		uimsbf		objectInfo
aDescription_length	8	uimsbf	N2	aDescription
for (i=0; i<N2; i++) {				
aDescription_bytes	8	uimsbf	+	
}				
duration.aSeconds	32	simsbf	+	AppNPT seconds
duration.aMicroSeconds	16	uimsbf	+	AppNPT micro seconds
audio	8	uimsbf	+	
video	8	uimsbf	+	
data	8	uimsbf	+	
}				
for (i=0; i<N6-(N2+10); i++) {				
objectInfo_byte	8	uimsbf	+	
}				
serviceContextList_count	8	uimsbf	0	serviceContextList
for (i=0; i<N3; i++) {				
serviceContextList_data	8	uimsbf	+	
}				
messageBody_length	32	uimsbf	*	
taps_count	8	uimsbf	N4	
for (i=0; i<N4; i++) {				
use	16	uimsbf	+	see Table 4-12 in DVB Guidelines for Data Broadcasting
id	16	uimsbf	0x0000	undefined
assocTag	16	uimsbf	+	
selector_length	8	uimsbf	0x00	no selector
}				
}				

Table 45. BIOP::StreamMessage syntax (Sheet 2 of 2)

Syntax	bits	Type	Value	Comment
BIOP::StreamEventMessage() {				
magic	4x8	uimsbf	0x42494F50	“BIOP”
version.major	8	uimsbf	0x01	BIOP major version 1
version.minor	8	uimsbf	0x00	BIOP minor version 0
byte_order	8	uimsbf	0x00	big endian byte ordering
message_type	8	uimsbf	*	
message_size	32	uimsbf	*	
objectKey_length	8	uimsbf	N1	
for (i=0; i<N1; i++) {				
objectKey_data	8	uimsbf	+	
}				
objectKind_length	32	uimsbf	0x00000004	
objectKind_data	4x8	uimsbf	0x73746500	“ste” type_id alias
objectInfo_length	16	uimsbf	N6	
DSM::Stream::Info_T {		uimsbf		
aDescription_length	8	uimsbf	N2	aDescription
for (i=0; i<N2; i++) {				

Table 46. BIOP::StreamEventMessage syntax (Sheet 1 of 2)

Syntax	bits	Type	Value	Comment
aDescription_bytes	8	uimsbf	+	see BIOP::StreamMessage()
}				
duration.aSeconds	32	simsbf	+	see BIOP::StreamMessage()
duration.aMicroSeconds	16	uimsbf	+	see BIOP::StreamMessage()
audio	8	uimsbf	+	see BIOP::StreamMessage()
video	8	uimsbf	+	see BIOP::StreamMessage()
data	8	uimsbf	+	see BIOP::StreamMessage()
}				
DSM::Event::EventList_T {				
eventNames_count	16	uimsbf	N3	
for (i=0; i<N3; i++) {				
eventName_length	8	uimsbf	N4	
for (j=0; j<N4; j++) {				
eventName_data	8	uimsbf	+	(including zero terminator)
}				
}				
}				
for (i=0; i<N6-(N2+10); i++) {				
objectInfo_byte	8	uimsbf	+	
}				
serviceContextList_count	8	uimsbf	N3	
for (i=0; i<N3; i++) {				
serviceContextList_data	8	uimsbf	+	
}				
messageBody_length	32	uimsbf	*	
taps_count	8	uimsbf	N5	
for (i=0; i<N5; i++) {				
use	16	uimsbf	+	see Table 4-12 in DVB Guidelines for Data Broadcasting
id	16	uimsbf	0x0000	undefined
assocTag	16	uimsbf	+	
selector_length	8	uimsbf	0x00	no selector
}				
eventIds_count	8	uimsbf	N3	(= eventNames_count)
for (i=0; i<N3; i++) {				
eventId	16	uimsbf	+	
}				
}				

Table 46. BIOP::StreamEventMessage syntax (Sheet 2 of 2)

7.2.3.5 BIOP Interoperable Object References

The Interoperable Object References (IOR) are references to objects and contain the necessary information to locate the object. The IOR structure may contain different options to be able to point to objects that can be reached via different types of connections. For this receiver profile, the use of IORs is limited to references to objects carried in broadcast object carousels. For object carousels, there are two types of object references: one to be used to

reference objects carried in the same object carousel and one to be used to reference objects in other object carousels.

Field	Restrictions	Source
IOP::IOR::type_id	Contains the objectKind of the referenced object. A short three-letter aliases shall be used, plus a null-terminator.	UK DTT
IOP::IOR:: taggedProfileList	There shall be at least 1 taggedProfile included in an IOR. For objects carried in a broadcast object carousel, the first taggedProfile shall be either a TAG_BIOP profile or a TAG_LITE_OPTIONS. If the first tagged profile is some other profile, the object is not carried in a broadcast object carousel and the receiver shall ignore the object.	UK DTT

Table 47. Restrictions on the BIOP IOR

Syntax	bits	Type	Value	Comment
IOP::IOR {				
type_id_length	32	uimsbf	N1	
for (i=0; i<N1; i++) {				
type_id_byte	8	uimsbf	+	Short alias type_id (e.g. “dir”)
}				
taggedProfiles_count	32	uimsbf	N2	Profile bodies
profileId_tag	32	uimsbf	+	For objects in broadcast carousels: either TAG_BIOP or TAG_LITE_OPTIONS
profile_data_length	32	uimsbf	N3	
for (I=0; i<N3; i++) {				
profile_data_byte	8	uimsbf		For objects in broadcast carousels: either BIOPProfileBody or LiteOptionsProfileBody
}				
for (n=0; n<N2-1;n++) {				Receiver may ignore other profiles (2...N1) if present
IOP::taggedProfile() {				
profileId_tag	32	uimsbf	+	
profile_data_length	32	uimsbf	N3	
for (i=0; i<N3; i++) {				
profile_data_byte	8	uimsbf		
}				
}				
}				
}				

Table 48. IOP::IOR syntax

BiopProfileBody

The BiopProfileBody is used for references to objects within the same object carousel.

Field	Restrictions	Source
BiopProfileBody:: byte_order	0 (indicating big-endian byte order)	DVB
BiopProfileBody:: LiteOptionComponents	The list shall contain a exactly 1 BiopObjectLocation and exactly 1 DsmConnectionBinder as the first two components in that order. The receiver may ignore possible other components in the list.	UK DTT
DsmConnectionBinder	For objects carried in the broadcast object carousel, the first Tap shall be of type BIOP_DELIVERY_PARA_USE. If there is another type of tap in the first position, the receiver may ignore this object reference, as it is a reference for object accessed using another type of protocol (e.g. for return channel use). The receiver may ignore possible other taps in the list.	UK DTT
BIOP::Tap	In the BIOP_DELIVER_PARA_USE tap, the id field is not used and may be ignored by the receiver.	UK DTT
DsmConnectionBinder	For objects carried in the broadcast object carousel, the first Tap shall be of type BIOP_DELIVERY_PARA_USE. If there is another type of tap in the first position, the receiver may ignore this object reference, as it is a reference for object accessed using another type of protocol (e.g. for return channel use). The receiver may ignore possible other taps in the list.	UK DTT

Table 49. Restrictions on the BIOP Profile Body

Syntax	bits	Type	Value	Comment
BIOPProfileBody {				
profileId_tag	32	uimsbf	0x49534F06	TAG_BIOP (BIOP Profile Body)
profile_data_length	32	uimsbf	*	
profile_data_byte_order	8	uimsbf	0x00	big endian byte order
liteComponents_count	8	uimsbf	N1	
BIOP::ObjectLocation {				
componentId_tag	32	uimsbf	0x49534F50	TAG_ObjectLocation
component_data_length	8	uimsbf	*	
carouselId	32	uimsbf	+	
moduleId	16	uimsbf	+	
version.major	8	uimsbf	0x01	BIOP protocol major version 1
version.minor	8	uimsbf	0x00	BIOP protocol minor version 0
objectKey_length	8	uimsbf	N2	<= 4
for (k=0; k<N2; k++) {				
objectKey_data	8	uimsbf	+	
}				
}				
DSM::ConnBinder {				
componentId_tag	32	uimsbf	0x49534F40	TAG_ConnBinder
component_data_length	8	uimsbf	N4	
taps_count	8	uimsbf	N3	
BIOP::Tap {				
use	16	uimsbf	0x0016	For objects carried in the broadcast object carousel: BIOP_DELIVERY_PARA_USE If there is another type of tap in the first position, the receiver may ignore this object reference, as it is a reference for object accessed using another type of protocol (e.g. for return channel use).
Id	16	uimsbf	0x0000	user private
assocTag	16	uimsbf	+	
selector_length	8	uimsbf	0x0A	
selector_type	16	uimsbf	0x0001	
transactionId	32	uimsbf	*	
timeout	32	uimsbf	*	
}				
for (n=0; n<N4-18; n++) {				The receiver may skip over the possible additional taps
additional_tap_byte	8	uimsbf		
}				
}				
for (n=0;n<N6;n++) {				N6=N1-2
BIOP::LiteOptionComponent{				
componentId_tag	32	uimsbf	+	
component_data_length	8	uimsbf	N7	
for (i=0; i<N7; i++) {				
component_data_byte	8	uimsbf		
}				
}				
}				
}				

Table 50. BIOP Profile Body syntax

Lite Options Profile Body

The LiteOptionsProfileBody is used for making links to objects carried in other object carousels. For this receiver profile, the use of the LiteOptionsProfileBody is limited to be used only within the same transport stream. Thus, the use of the object carousel will not require tuning to a new transport stream.

Field	Restrictions	Source
LiteOptionsProfileBody:: profile_data_byte_order	0 (indicating big-endian byte order)	DVB
LiteOptionsProfileBody:: LiteOptionComponents	The list shall contain a ServiceLocation component as the first component. The receiver may ignore possible other components in the list.	UK DTT
Dsm::ServiceLocation	For objects carried in the broadcast object carousel, the service domain NSAP address shall follow the Carousel NSAP address format. If there is another type of NSAP address, the receiver may ignore this object reference, as it is a reference for object accessed using another type of protocol (e.g. for return channel use). The carousel NSAP address shall point to an object carousel in the same transport stream. If the NSAP address points to another transport stream, the receiver may ignore the object reference.	UK DTT
Dsm::ServiceLocation:: InitialContext	The receiver may ignore the initial context	UK DTT

Table 51. Restrictions on the Lite Options Profile Body

Syntax	bits	Type	Value	Comment
LiteOptionsProfileBody {				
profileId_tag	32	uimsbf	0x49534F05	TAG_LITE_OPTIONS (Lite Options Profile Body)
profile_data_length	32	uimsbf	*	
profile_data_byte_order	8	uimsbf	0x00	big endian byte order
component_count	8	uimsbf	N1	
DSM::ServiceLocation {				
componentId_tag	32	uimsbf	0x49534F46	TAG_ServiceLocation
component_data_length	32	uimsbf	*	
serviceDomain_length	8	uimsbf	0x14	Length of carousel NSAP address
serviceDomain_data()	160	uimsbf	+	DVBcarouselNSAPaddress, see Table 4-8 in DVB Guidelines for Data Broadcasting
CosNaming::Name() {				pathName
nameComponents_count	32	uimsbf	N2	
for (i=0; i<N2; i++) {				
id_length	32	uimsbf	N3	NameComponent id
for (j=0; j<N3 j++) {				
id_data	8	uimsbf	+	
}				
kind_length	32	uimsbf	N4	NameComponent kind
for (j=0; j<N4 j++) {				
kind_data	8	uimsbf	+	as type_id (see Table 4-4 in DVB Guidelines for Data Broadcasting)
}				
}				

Table 52. Syntax of Options Profile Body with ServiceLocation component. (Sheet 1 of 2)

Syntax	bits	Type	Value	Comment
}				
initialContext_length	32	uimsbf	N5	
for (n=0; n<N5 n++) {				
InitialContext_data_byte	8	uimsbf		
}				
}				
for (n=0;n<N6;n++) {				N6=N1-1
BIOP::LiteOptionComponent{				
componentId_tag	32	uimsbf	+	
component_data_length	8	uimsbf	N7	
for (i=0; i<N7; i++) {				
component_data_byte	8	uimsbf		
}				
}				
}				
}				

Table 52. Syntax of Options Profile Body with ServiceLocation component. (Sheet 2 of 2)

7.2.4 Assignment and use of transactionId values

The use of the transactionId in the object carousel is inherited from its use as defined by the DSM-CC specification, and as such it can appear somewhat complex. The transactionId has a dual role, providing both identification and versioning mechanisms for control messages, i.e. DownloadInfoIndication and DownloadServerInitiate messages. The transactionId should uniquely identify a download control message within a data carousel, however it should be “incremented” whenever any field of the message is modified.

Note: The term “incremented” is used in the DSM-CC specification. Within the scope of the UK DTT object carousel this should be interpreted as “changed”.

An object carousel are carried on top the data carousels and may be distributed over multiple data carousels. By a data carousel used below the object carousel, we mean in this specification a set of DownloadInfoIndication message transmitted on a single PID and the DownloadDataBlock messages carrying the modules described in the DownloadInfoIndication messages. The DownloadDataBlock messages may be spread on other elementary streams than the DownloadInfoIndication messages. The DownloadServerInitiate message in the context of object carousels is considered to be part of the top level of the object carousel and not associated with any data carousel.

When a module is changed, the version number of the module needs to be changed. This implies that the DownloadInfoIndication message that references the module needs to be also updated. Since the DownloadInfoIndication is updated, the transactionId needs to be also changed. However, the transactionId of the DownloadInfoIndication message is used in other messages also, but the need to change the other messages should specifically be avoided and the implications of updating a module should be limited to the module itself and the DownloadInfoIndication that references the module. Therefore, additional rules on the usage of the transactionId have been specified as follows.

The transactionId has been split up into a number of sub-fields defined in Table 53. This reflects the dual role of the transactionId (outlined above) and constraints imposed to reduce the effects of updating a module. However, to increase interoperability the assignment of the

transactionId has been designed to be independent of the expected filtering in target receivers.

Bits	Value	Sub-field	Description
0	User-defined	Updated flag	This must be toggled every time the control message is updated
1-15	User-defined	Identification	This must and can only be all zeros for the DownloadServerInitiate message. All other control messages must have one or more non-zero bit(s).
16-29	User-defined	Version	This must be incremented/changed every time the control message is updated.
30-31	Bit 30 - zero Bit 31 - non-zero	Originator	This is defined in the DSM-CC specification [17] as 0x02 if the transactionId has been assigned by the network - in a broadcast scenario this is implicit.

Table 53. Sub-fields of the transactionId

Due to the role of the transactionId as a versioning mechanism, any change to a control message will cause the transactionId of that control message to be incremented. Any change to a Module will necessitate incrementing its moduleVersion field. This change must be reflected in the corresponding field in the description of the Module in the DownloadInfoIndication message(s) that describes it. Since a field in the DownloadInfoIndication message is changed its transactionId must be incremented to indicate a new version of the message. Also, any change in the DownloadServerInitiate message implies that its transactionId must also be incremented. However, when the transactionId is divided into subfields as specified above, updating a message will change only the Version part of the transactionId while the Identification part remains the same.

Since the transactionId is used also for identifying the messages when referencing the messages in other structures, it is very desirable that these referenced would not need to be updated every time the control message is update. Therefore the following rule shall be applied when locating the messages based on the references:

When locating a message based on the transactionId value used for referencing the message, only the Identification part (bits 1...15) shall be matched.

Using this rule, the implications of updating a module can be limited to the module itself and the DownloadInfoIndication message describing the module. Also, this implies that if a receiver wants to find out if a particular module that it has retrieved earlier has changed, it needs to filter the DownloadInfoIndication message that described that module and check if it has been changed.

7.2.5 Mapping of objects to data carousel modules

The DSM-CC Object Carousels allow one or more objects to be carried in one module of the data carousel. In order to optimize the performance and memory requirements two additional requirements are specified:

- When mapping objects to modules of a data carousel, only closely related objects (e.g. objects of one MHEG scene) should be put into one module. Objects that are not closely related should not be put into the same module. If in the process of retrieving an object from the carousel a receiver acquires a module containing multiple objects, it should attempt to cache these since the expectation should be that the other objects are related to the object requested and probably will be needed soon.
- The size of a module that contains multiple objects should not exceed [64] kB when decompressed[1]. For modules containing only a single object, there is obviously no limit for the size (except what is determined by the memory in the receivers and the size of the length fields).

7.2.6 Compression of modules

The modules may be transmitted either in uncompressed or compressed form. If the module is transmitted in compressed form, this is signalled by including the compressed_module_descriptor in the userInfo field of the moduleInfo in the DownloadInfoIndication message.

Presence of the compressed_module_descriptor indicates that the data in the module has the "zlib" structure as defined in RFC1950.

Table 54 shows the syntax of the compressed_module_descriptor:

	No. of bytes	Mnemonic	Value
compressed_module_descriptor(){			
descriptor_tag	1	uimsbf	0x09
descriptor_length	1	uimsbf	
original_size	4	uimsbf	
}			

Table 54. compressed_module_descriptor

Presence of the compressed_module_descriptor indicates that the data in the module has the "zlib" structure as defined in RFC 1950. Table 55 shows the syntax of the ZLIB structure.

	No. of bytes	Value	
zlib structure(){			
compression_method	1		
flags_check	1		
compressed_data	n		
check value	4		
}			

Table 55. zlib structure

The receiver shall support the Deflate compression algorithm as specified in RFC 1951. This is signalled setting the most significant nibble of the compression_method to 0x8 (i.e.

1. I.e. when the file has been decompressed from the file transport but before the content decoding has started.

compression_method is 1000xxxx). The receiver is not required to support other compression algorithms.

7.3 Tag Mapping

7.3.1 MHEG-5 ComponentTags to DSM-CC association_tags

The MHEG-5 ComponentTag is mapped to the least significant byte of the association_tag as defined by ISO/IEC 13818-6 [17]. The most significant byte of the association_tag may take any value (0xXX). The value encoded in the least significant byte (LSB) of the association_tag shall be equal to the MHEG-5 ComponentTag value. In addition, the value of the component_tag defined in the stream_identifier_descriptor specified by DVB shall also take the same value as the MHEG-5 ComponentTag. A stream_identifier_descriptor in the descriptor loop of a PID is equivalent with an association_tag_descriptor for that PID with an association_tag value of MSB=0xXX and LSB=<component_tag> and a use value of 0x0100.

7.3.2 DSM-CC association_tags to DVB component_tags

The component_tag in a PMT's stream_identifier_descriptor is used to relate SI service component information with an elementary stream without directly referring to a PID value. Likewise, the association_tag in the DSM-CC's association_tag_descriptor is used to relate (part of) an object carousel with an elementary stream without directly referencing a PID value. To avoid duplicating component_- and association_tags, the DVB specification defines a mapping between the two. The mapping of the 8-bit component_tag to the 16-bit association_tag is as follows: (ref. paragraph 8.3)

> "it is assumed that in the absence of an association_tag_descriptor, the component_tag field of the stream_identifier_descriptor is the Least Significant Byte of the referenced association_tag value."

Similarly, the DVB Guideline is: (ref. 4.7.7.3)

> "A stream_identifier_descriptor in the descriptor loop of a PID shall be equivalent with an association_tag_descriptor for that PID with an association_tag value of LSB=<component_tag> and a use value of 0x0100."

Note that this definition provides the flexibility to distribute the object carousel over multiple elementary streams and over multiple services while still allowing the use of the same component_tag value in the different PMTs.

See also "Component tags for RTGraphics" on page 66.

The following figure illustrates the decision tree for identifying the elementary stream(s) by which the object carousel is distributed:

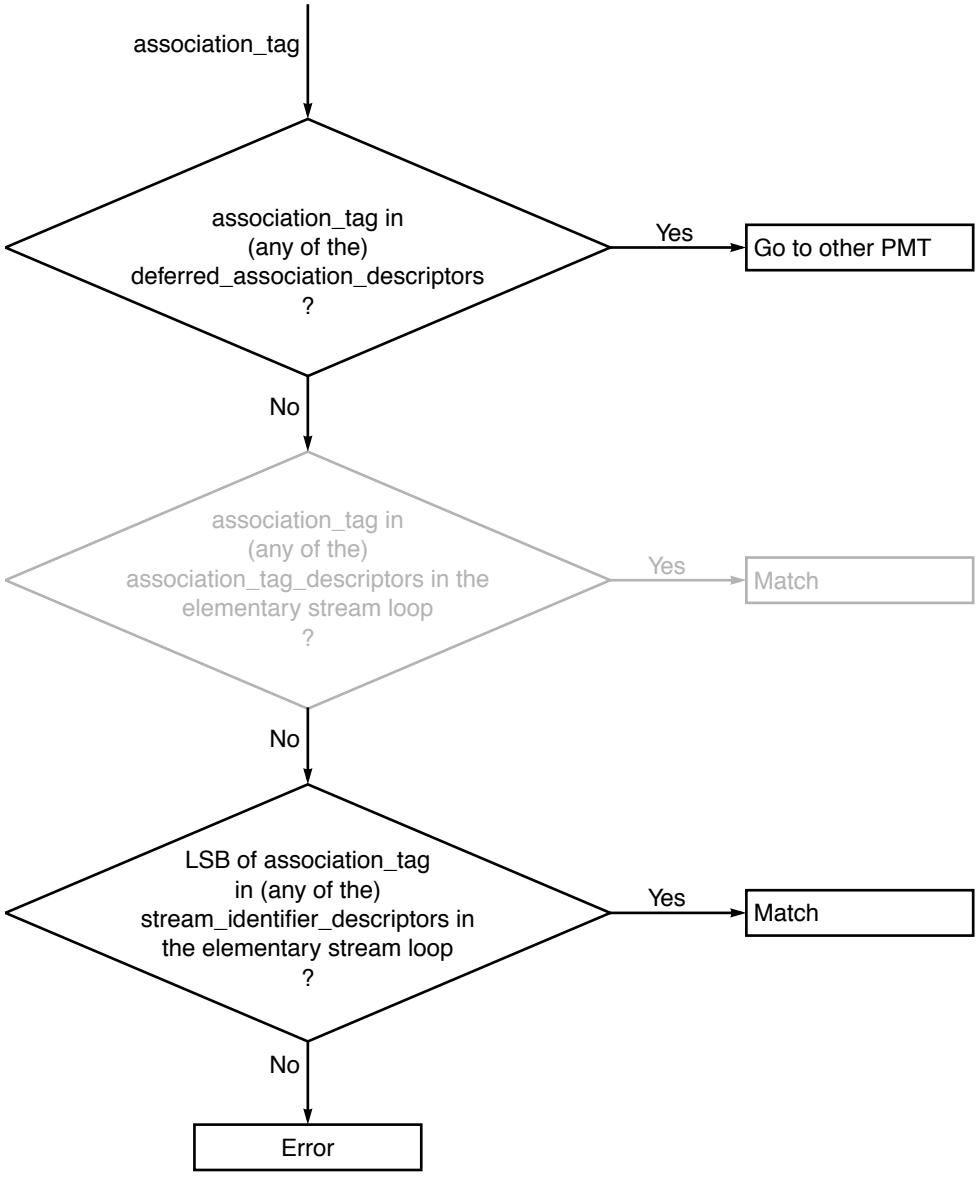


Figure 20. Object Carousel ES identification decision tree

In UK DTT, the stream_identifier_descriptor shall always be used for assigning a component_tag for the elementary streams. Use of association_tag_descriptors is not required. If the association_tag_descriptor is optionally used, a stream_identifier_descriptor shall still be present and the tag values shall be set consistently in each descriptor. This restriction simplifies the decision tree above so that the second decision can be skipped.

7.4 Example of an Object Carousel

The figure below illustrates an object carousel that is distributed over three elementary streams belonging to the same service.

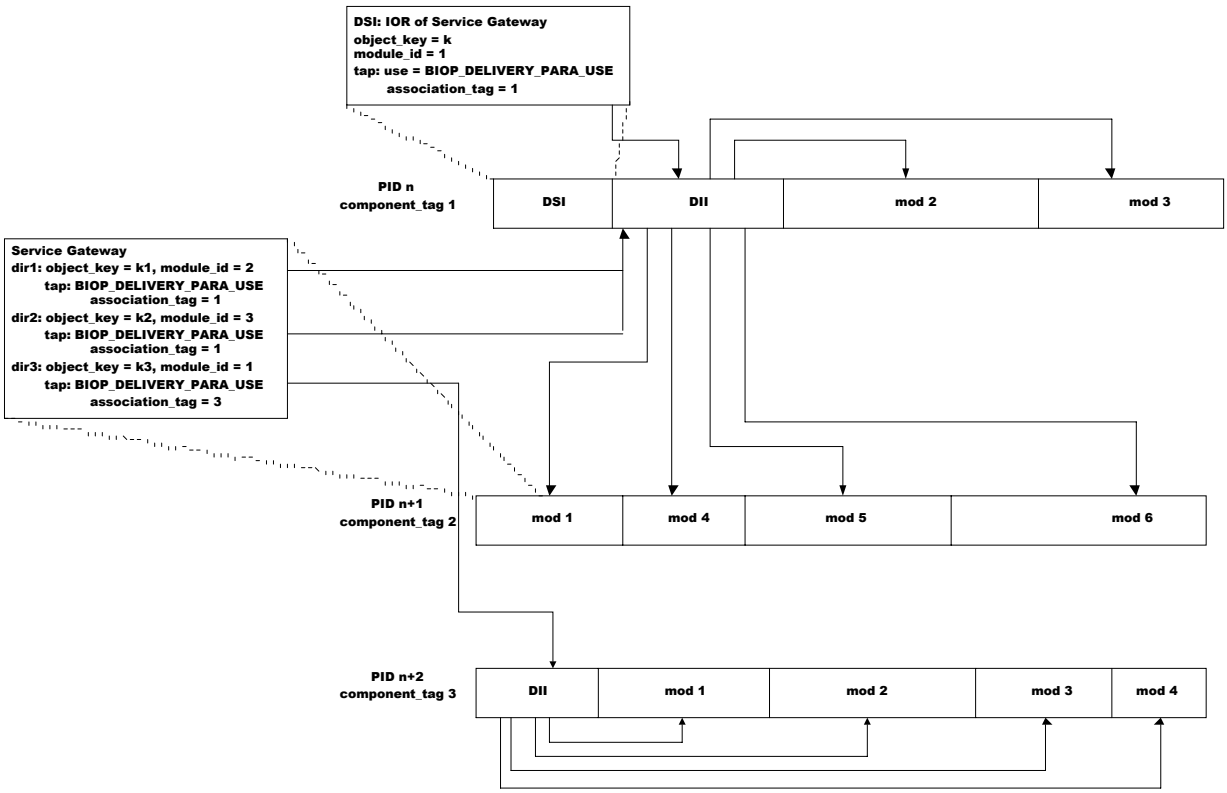


Figure 21. Example carousel

The DownloadServerInitiate (DSI) message is carried on the first elementary stream. It contains the object reference that points to the service gateway. The tap with the BIOP_DELIVERY_PARA_USE points to a DownloadInfoIndication (DII) message that provides the information about the module and the location where the module is being broadcasted. In the example, the service gateway object is in the module number 1 that is carried on the second elementary stream (indicated by a BIOP_OBJECT_USE tap structure in the DII message).

The Service Gateway object is a root directory that, in this example, references three subdirectories. Taps with BIOP_DELIVERY_PARA_USE are used in the object references of the subdirectories to provide links to the modules via the DownloadInfoIndication (DII) message. The two first subdirectories “dir1” and “dir2” are referenced in the DII message that is carried in the first elementary stream. The third subdirectory is referenced in the DII message carried in the third elementary stream.

In this example, the two first elementary streams carry the messages of one logical data carousel while the third elementary stream carries the messages of another logical data carousel. All these belong to the same object carousel. In the example, the third elementary stream contains the objects in the “dir3” subdirectory and the objects in the “dir1” and “dir2” subdirectories are distributed over the first and second elementary stream.

It is important to note that the third elementary stream may originate from a completely separate source than the first two elementary streams. The directory hierarchy and objects contained in the third elementary stream are “mounted” in the root directory by providing the “dir3” directory entry with the appropriate location information.

This type of structure could be used, for example, in a national information service that contains some regional parts. The common national parts could be carried in this example case on the two first elementary streams that are distributed unmodified in the whole country. The regional parts are carried in the third elementary stream that is locally inserted at each region. From the application's point of view, the common national parts are in the "dir1" and "dir2" subdirectories while the regional parts are in the "dir3" subdirectory.

Another example where this type of structure could be used is if the service contains multiple independent applications. In this case, each application could be placed in its own subdirectory and these subdirectories might be carried as separate data carousels on different elementary streams.

7.5 Application Identification and Boot

7.5.1 Introduction

This section covers the identification and boot of an application by a receiver. The term "application" in this section of the specification will describe the use of the broadcast stream. So, the application defined in this specification can be described as:

> UK DTT Profile Version 1 of MHEG-5 delivered using the DVB Object Carousel Revision 1

The mechanism specified is designed to rely only on information in the PMT, i.e. there is no need to access the SDT or EIT to be able to identify and boot the application.

This section is still in development. The proposal is not yet complete and may be modified following discussions within the DVB.

7.5.2 Information in the PMT

7.5.2.1 Generic Application Identification

Any application delivered within a UK DTT multiplex (whether or not covered in this specification) shall be identified by the use of the **application_descriptor** in the PMT. If present, it shall be in the descriptor loop for the elementary stream (SI component) from which the application may be booted.

Note: This descriptor loop is usually referred to as the second descriptor loop of the PMT.

A generic form of the **application_descriptor** which receivers shall be able to parse is as follows:

	Size (bytes)	Value	Comment
application_descriptor{			
descriptor_tag	1	0x80	[a]
descriptor_length	1	N1*(2+N2+N3)	
for(i=0; i<N1; j++){			
application_type_id	4[b]		Registered centrally
application_type_id_revision	1		Registered locally
application_specific_data_length	1	N3	
for(j=0; j<N3; j++){			
application_specific_data_byte	1		See Table 57
}			
}			
}			

Table 56. Application descriptor

a] The value of descriptor_tag is a working value allocated from the DVB user private range which may be subject to change.

b] The size of the application_type_id field may be subject to change.

Only one instance of the descriptor shall be included in an single descriptor loop since it is capable of identifying multiple applications if necessary. Instances of the descriptor may be associated with as many elementary streams in a single MPEG Program as is necessary.

The **application_type_id** is a registered value that is used to identify the application. This value will be allocated by a central registration body[1] and will remain fixed.

Knowing that the delivery profile is DVB Object Carousel and that presentation format is MHEG-5 still does not guarantee that the receiver can run the application, e.g. images may be encoded as PNG when the receiver only supports GIF. The receiver needs to know that it is the "UK DTT Profile Version 1" which implicitly defines the other information. Thus a registered number is sufficient. It may (or may not) be sensible to construct the number from a series of subfields, so that some information can be extracted from it if necessary.

The **application_type_id_revision** is free to be updated by the specifying body of the application without consulting the registration body.

The use of the **application_specific_data** shall be defined by the specifying body at the time of **application_type_id** registration. A receiver may skip over the **application_specific_data** for an **application_type_id** that it does not recognise by virtue of the **application_specific_data_length** field.

7.5.2.2 Specific Application Identification

For the application defined in this specification the following shall be used:

	Size (bytes)	Value	Comment
:			See Table 56
application_type_id		0x8000[a]	
application_type_id_revision		0x01	
application_specific_data{			
transaction_id[b]	4		
timeout	4		
private_data_length	1	N1	
for(i=0; i<N1; i++){			
private_data_byte	1		
}			
}			
:			

Table 57. Application specific data

a] The value for application_type_id is an arbitrary choice and may be subject to change.

b] The inclusion of the transaction_id and timeout fields is still under consideration.

Receivers may ignore the contents of **private_data**.

The U-U Object Carousel may be spread over more than one elementary stream. However, the DSI messages that allow the receiver to boot the application shall only be broadcast on a single elementary stream. Thus only one **application_descriptor** shall be present in an MPEG Program for this application.

1. Whether this central body is UK DTT or DVB is an open issue.

7.5.2.3 Other Related PMT Descriptors

Descriptor	Comment
carousel_id_descriptor	Receivers may ignore this descriptor.
data_broadcast_descriptor	Receivers may ignore this descriptor.
data_broadcast_id_descriptor	Receivers may ignore this descriptor.
association_tag_descriptor	Receivers shall interpret this descriptor as defined, except for the **selector** and **private_data** fields which may be ignored.
deferred_association_tag_descriptor	Receivers shall interpret this descriptor as defined.
stream_identifier_descriptor	Receivers shall interpret this descriptor as defined.

Table 58. Other PMT descriptors

7.5.2.4 Example Use of application_descriptor

The following example shows an application where the U-U Object Carousel has been spread over two elementary streams. DSI messages are only broadcast on **elementary_stream_1**.

	Size (bytes)	Value	
PMT{			
< various fields >			
first_descriptor_loop{			
}			
elementary_stream_loop{			
elementary_stream_1{			
< various fields >			
second_descriptor_loop{			
application_descriptor(){			See Table 56
descriptor_tag	1	0x80	
descriptor_length	1	0x0E	
application_type_id	4	0x8000	
application_type_id_revision	1	0x01	
application_specific_data_length	1	N3 = 9	
transaction_id	4	0xFFFFFFFF	
timeout	4	0xFFFFFFFF	
private_data_length	1	N1 = 0	
}			
stream_identifier_descriptor(){			Defined by DVB
descriptor_tag	1		
descriptor_length	1		
component_tag	1		
}			
}			
}			

Table 59. Signalling in the PMT (Sheet 1 of 2)

	Size (bytes)	Value	
elementary_stream_2{			
< various fields >			
second_descriptor_loop{			
stream_identifier_descriptor(){			Defined by DVB
descriptor_tag	1		
descriptor_length	1		
component_tag	1		
}			
}			
}			
}			
}			

Table 59. Signalling in the PMT (Sheet 2 of 2)

7.5.3 Information in the DSI

7.5.3.1 Specifying an Initial Object

An initial object, which in this specification is the Application object to launch, may be explicitly defined for reasons of application authoring convenience or to support multiple languages.

It is not mandatory to explicitly specify an initial object. If no initial object is specified then default values will be used. See "No Explicit Initial Object Identified" on page 99.

Explicit initial objects may be specified using an application initial object descriptor in the **userInfo** field of the ServiceGatewayInfo message (see page 76) as well as in the DSI message. The initial object value to use (when there is more than one choice) may be determined by the use of ISO language codes.

To specify a default choice or a single, language independent choice the ISO language code "und" should be used.

No order of preference shall be implied by the order of choices in the loop.

	Size (bytes)	Value	
application_initial_object_descriptor(){			
descriptor_tag	1	0x81[a]	
descriptor_length	1	N1	$N1 = 1 + \sum_{j=0}^{j<N2} 4 + N3(j)$
num_initial_objects	1	N2	
for(j=0; j<N2; j++){			
ISO_639_language_code	3		
initial_object_length	1	N3	
for(k=0; k<N3; k++){			
initial_object_byte	1		
}			
}			
}			

Table 60. Initial object specification descriptor

a] The value for application_initial_object_descriptor is an arbitrary choice and may be subject to change.

The **initial_object_length** shall include a null terminator.

The **initial_object** field may include a path.

7.5.3.2 Example Use of DSI userInfo Field

	Size (bytes)	Value	
DSI {			
< various fields >			
privateData_length	2		
privateData{			
ServiceGatewayInfo{			See Table 38 on page 76
< various fields >			
userInfo_length	2	0x0000	
userInfo{			Empty
}			
}			
}			
}			

Table 61. Initial Object Not Explicitly Specified

	Size (bytes)	Value	
DSI {			
< various fields >			
privateData_length	2		
privateData{			
ServiceGatewayInfo{			See Table 38 on page 76
< various fields >			
userInfo_length	2		
application_initial_object_descriptor()			See Table 60 on page 98
}			
}			
}			

Table 62. Single, Language Independent Initial Object Explicitly Specified

7.5.4 Locating the Initial Object

No Explicit Initial Object Identified

If no appropriate **initial_object** field can be identified in the DSI message then the receiver shall use the following default names to locate the initial Application object in the Service-Gateway object (in the following order):

"~/a"

"~/startup"

According to DAVIC the initial object is "~/startup". However, it would be sensible to support a shorter alternative to this to save bitrate. "~/s" as an abbreviation of "startup" could be confused with an abbreviation of "scene", so "~/a" as an abbreviation of "application" is probably better.

Explicit Initial Object Identified

If an appropriate **initial_object** field can be identified from the DSI message then the receiver shall use this to locate the initial Application object.

If the **initial_object** field identifies a File object, this shall be the initial object.

If the **initial_object** field identifies a Directory object, the receiver shall use the default names to locate the initial Application object in this Directory object.

If the **initial_object** field produces no match in either of the above steps, it shall be discarded and the receiver shall use the default names to locate the initial Application object in the ServiceGateway.

So for **initial_object**:

"my_app"

The receiver shall work down the following list (in the order shown):

"~/my_app"

"~/my_app/a"

"~/my_app/startup"

"~/a"

"~/startup"

7.5.5 Steps Required to Identify and Boot Application

1. Tune to the transport stream of the service (MPEG Program).
2. Acquire the PAT.
3. Acquire the PMT of the service using the information in the PAT.
4. Search through the second descriptor loops in the PMT, i.e. the descriptor loop for each elementary stream, for an instance of the **application_descriptor** containing the appropriate **application_type_id** and **application_type_id_revision** values. This identifies the elementary stream on which DSI messages are broadcast.
5. If there is no match, there is no associated application in the service.
6. Acquire the DSI from this elementary stream.
7. Acquire the DII message (DII1) that identifies the module (Module1) containing the ServiceGateway object, using information in the DSI message (IOR1).
8. Acquire Module1 using information in DII1.
9. Extract the ServiceGateway object from Module1 using information in IOR1.
10. Get the name of the initial File object (in this case containing an MHEG-5 Application object) using explicit information in the DSI message and/or default values. The location of this File object is relative to the ServiceGateway object.
11. If the name points to subdirectories follow the whole path, filtering the appropriate Directory objects (essentially repeating steps 6...8 in a more generic fashion as required).
12. Acquire the DII message (DII2) that identifies the module (Module2) containing the initial File object, using information in the ServiceGateway object (IOR2).
13. Acquire Module2 using information in DII2.
14. Extract the initial File object from Module2 using information in IOR2.

8 Name Mapping

8.1 Names within the receiver

Name Format	Use	Comment
rec://def_service	ContentReference for a Stream object	This content reference identifies the receivers currently selected service. See “Stream inheritance by applications” on page 65.
ram://<name>	Name space for persistent storage.	See “Persistent storage” on page 69

Table 63. Names within the receiver

8.2 MHEG-5 Component tags

See “Component tags for RTGraphics” on page 66.

See “Tag Mapping” on page 90.

8.3 DAVIC definitions

The text here is an extract from DAVIC 1.3 Part 09 [27] provided for information. The formal definition of these resident programs remains the DAVIC document *except where noted*.

See also:

- “Non-presented text” on page 45
- “Type conversion” on page 46
- “GroupIndentifier” on page 28

8.3.1 Stream Events and Normal Play Time Mapping

The DSM-CC StreamEvent interface provides the possibility to carry private data in the data structure for the event, in the form of the PrivateDataByte field. These bytes shall be mapped one-to-one on the StreamEventTag of the MHEG-5 event StreamEvent.

The MHEG-5 internal attribute CounterPosition of the Stream class shall also be mapped on the value of the DSM-CC Normal Play Time of the corresponding stream. The counter position shall represent a millisecond precision. Mapping from Normal Play Time to CounterPosition shall take place by rounding the value to the nearest millisecond.

8.3.2 Namespace Mapping

When an application starts, it is assumed that a service gateway has been located and attached to, so that there is exactly one name space within which the application objects are located. Within that name space, a service has also been located. That service is a DSM-CC directory; within it, there can be other directories, files, and streams.

Furthermore, it is assumed that each object belongs to exactly one application. This assumption is necessary to allow for unambiguous object references.

The high-level API differentiates between three types of retrieved data:

- objects that comply to the high-level API definition,
- the content (such as bitmaps or text) of those objects, and
- streams (such as video and audio).

For accessing the various objects of an application on the server side, the DSM-CC Directory, File and Stream objects are used. Note that the server, in this context, does not have to be a physical server, but could be implemented, for example, as a broadcast carousel in a pure-broadcast topology.

Each file is either a Scene object, an Application object, or the content data of an Ingredient object. Each Scene object, Application object and content data is stored in a separate file.

8.3.3 MHEG-5 Object References

MHEG-5 objects can be exchanged in two ways: either the object is exchanged as a DSM-CC file object or within another high-level API object. The former method is used for Applications and Scenes, the latter for all other objects contained in a scene or application object.

MHEG-5 references objects by an ObjectReference, consisting of an optional byte string GroupIdentifier, followed by an integer, the ObjectNumber.

For the mapping on DSM-CC, the following additional rules are defined:

1. Each Application and Scene object shall have in its GroupIdentifier a byte string which maps on the name of the DSM-CC file which contains that object. These objects shall have their ObjectNumber set to 0.
2. Each application shall have exactly one Application object. That object shall be contained in a DSM-CC File object with the name ‘startup’. Only one Application object shall be contained in each such file.
3. Ingredient Objects other than Application and Scene objects may

- either leave out the GroupIdentifier, in which case it is assumed to be a string which maps on the name of a DSM-CC file which contains the object (Application or Scene) of which this object is a part,
- or fill in the GroupIdentifier with such a string.

4. Such objects shall have their ObjectNumber set to a value which is unique within that Scene.
5. For the GroupIdentifier, the mapping rules defined in the following Clause apply.

8.3.3.1 Mapping Rules for GroupIdentifier

All GroupIdentifiers are ASCII strings. They are composed of four components:

```
Source   Path Origin   Path Filename
```

The Source component is optional, and specifies the data source to be used to retrieve the data. Each source identification is terminated with a “:”. The default source identification is “DSM:” (in upper case). DAVIC 1.3 does not specify any further data sources, but the use of other source identifications is permitted.

The Path Origin is either ‘//’ or ‘/’. If the path origin is ‘//’ then the following path and filename are to be interpreted as an absolute path starting from the root of the current service gateway to which the runtime is attached. If the path origin is ‘/’ then the following path and filename is to be interpreted as a relative path, starting from the directory that contains the current Application object.

The Path component is a (possible empty) sequence of directory names, each followed by a '/' character. Each directory name is to be used to delimit directory references (of the depth type).

The Filename component identifies the DSMCC object within the directory that is identified by the preceding Source, Path Origin and Path components.

For instance, if the GroupIdentifier is 'DSM://apps/otherAppl', it is mapped on the DSM-CC name 'otherAppl' in the directory 'apps' of the service gateway to which the runtime has attached. If the GroupIdentifier is 'DSM:/scenes/myScene', it is mapped on the DSM-CC name 'myScene' in the directory 'scenes' of the directory where the Application object was found.

8.3.3.2 Shorthand Notation

DAVIC defines shorthand notations for references to objects accessible from the default Data Source. In DAVIC 1.3 the default Data Source is a DSM-CC service gateway. The rationale behind the shorthand notation is compatibility with previous versions of the DAVIC specification.

There are two abbreviations allowed. The first abbreviation is '~/' (standard ASCII tilde + slash), which may be used to specify the default Data Source and a relative Path Origin; use in DAVIC 1.3 means 'DSM:' for the Data Source and '/' for Path Origin. This abbreviation can therefore be used to refer to objects relative to the current application.

The other abbreviation is '/' (standard ASCII slash), which may be used to specify the default Data Source and an absolute Path Origin; use in DAVIC 1.3 means 'DSM:' for the Source and '//' for Path Origin. This abbreviation can therefore be used to refer to objects with an absolute path from the root of the current service gateway to which the runtime is attached.

For instance, an object, referred to as 'DSM:/scenes/myScene' may be referred to as '~/scenes/myScene', while the object referred to as 'DSM://apps/otherAppl' may be referred to as '/apps/otherAppl'.

8.3.3.3 Other Service Gateways

It is possible to refer to objects in another service gateway, as DSMCC objects have built-in indirection to the actual location of the data. An object with the path "DSM://Mheg5/Banking/Welcome" can refer to a DSMCC object whose actual data is only accessible from another service gateway. Normal access to such an object will result in a 'service transfer exception', which will be followed by an attempt to attach to that service gateway.

8.3.4 MHEG-5 Content References

MHEG-5 has a separate way of referencing the actual content of objects belonging to the Ingredient class. This is done by way of a ContentReference. The ContentReference consists of an optional PublicIdentifier followed by a SystemIdentifier, which is a byte string. The following rule shall apply.

For the SystemIdentifier, the exact same mapping shall be used as for the GroupIdentifier above. The PublicIdentifier shall not be used.

8.3.4.1 DSMCC Stream Objects

Note that the mapping of ContentReference and GroupIdentifier allow references to both DSMCC File and DSMCC Stream objects. Although this is possible, applications shall not refer to DSMCC Stream objects using a GroupIdentifier. GroupIdentifiers are always expected to refer to DSMCC File objects.

ContentReferences of content of MHEG-5 Stream objects may refer to both DSMCC File and DSMCC Stream objects. If such a ContentReference refers to a DSMCC File object, the

MHEG-5 Stream object shall have its Storage attribute set to ‘memory’. If the ContentReference refers to a DSMCC Stream object, the MHEG-5 Stream object shall have its Storage attribute set to ‘stream’.

ContentReferences of content of MHEG-5 Ingredients, other than Stream objects must always refer to a DSMCC File object.

8.3.5 URL Format for Access to Broadcast Services

8.3.5.1 Introduction

DAVIC defines a specific Uniform Resource Locator (URL) format to access broadcast services. This URL format provides a general addressing mechanism intended to access broadcast services from e.g. JAVA and HTML. Note that URLs are commonly used in JAVA for addressing resources and other objects.

DAVIC broadcast networks carry the Service Information (SI) which contains globally unique parameters for locating the services in the broadcast networks. The URL format defined by DAVIC to access such services is based on these parameters as they provide an addressing mechanism in a physical network independent way. The same services may be carried simultaneously in many physical networks, but the parameters in the SI will remain the same and thus they can be used by the clients to locate the services regardless of the actual physical network.

DAVIC defined the following general format of the URLs :

<protocol>://<“server”>/<dir1>/.../<file>

The protocol (scene) part of the URL identifies that it is a broadcast service. The “server” part of the URL points to the service as the services are the basic element that is carried in the broadcast networks. The rest of the URL specifies the individual component inside a service. The format of the last part is dependent on the type of the service (this part is not needed if the URL points to the whole service).

8.3.5.2 Numerical format

The combination of original_network_id, transport_stream_id and service_id identifies a service in a DVB / DAVIC broadcast network globally uniquely. Thus, the first part of the URL in a numerical form can be following:

UK modification

dvb://<original_network_id>.**[<transport_stream_id>]**[.<service_id>[.<component_tag>][;<event_id>]][/<...>]

UK modification

So, URLs specifying a service might be of the form:

dvb://abcd.1234.5679

or

dvb://abcd..5679

See ETR 211 [8]. Services are unique within the scope of a single original network ID. So, transport stream ID is not required to uniquely specify a service within an original network. The transport stream id (0x1234 in this case) is optional in URLs and can be left empty.

In terrestrial networks the same service may be available to the receiver on more than one transport stream. Where a URL does not define the transport stream ID the receiver is free to choose which transport stream ID it selects when obtaining a service.

The <original_network_id>, <transport_stream_id>, <service_id>, <event_id> and <component_tag> are the values represented as hexadecimal strings without any “0x” prefix (for example, “4d2e”).

Depending on the service the following part of the URL (“/<...>“) may contain a more detailed specifier that identifies the desired part of the service. For example, if the service contains a data carousel, the last part of the URL may contain the file name.

9 MHEG-5 Authoring Guidelines

9.1 Introduction

This section describes the measures that should be taken by the broadcaster to ensure good application behaviour.

9.2 Avoiding confusion with navigator functions

9.2.1 Use of user inputs

Requirement

The user shall be able to “surf” through services with MHEG applications without a change in the user interface behaviour.

Selecting & Entering

Two concepts are introduced *selecting* the MHEG application and *entering* it.

When the user first navigates to a service that is wholly or partly implemented as an MHEG application it is “selected”. If the user interacts with the MHEG aspects of that service they “enter” it.

Until the user “enters” the application the channel change facility of the receiver should not be modified by the application.

Limiting Input methods on first selection

Applications should be authored so that none of the user inputs normally associated with changing channel are used by the application until the user has “entered” the application.

Doing it

The first scene of the application should:

- **Use an input register that excludes the ““Entered” Group” of keys (i.e. only the ““Selected” Group”)**
- **Require the user to actively select an EntryField, or any other Interactible, before setting its InteractionStatus to True.**

These measures mean that when an application is first “selected” it makes no use of the ““Entered” Group” of keys. So, these keys retain their usual receiver specific functions (such as “go to favourite channel”).

An entry field may be included in the first scene (e.g. to allow the user to “jump” to a specified scene). However, it is not initially active, and so won’t absorb numeric input. One of the ““Selected” Group” must be used activate the entry field (or to navigate to a scene where the entry field is active). Again this preserves the “go to favourite channel” function of the ““Entered” Group” of keys.

Within these constraints the author is free to use the ““Selected” Group” of keys to navigate from this first scene.

Starting Applications “Entered”

Through use of “Persistent storage” an application can launch a second application that transitions immediately to an interior “entered” scene. This should only be done after the user has “entered” the first application.

9.3 Use of the “Text” function

The “Text” user input event has a specific meaning that should be observed by all applications to avoid confusing the user. It means “toggle” the visibility of the MHEG aspect of the service.

9.3.1 The traditional “teletext” key

In Figure 22 “Text” toggles between conventional TV and a full screen MHEG application. This is very similar to the behaviour of today’s “teletext” button.

An available elaboration is that a single “teletext” service can be shared by a number of TV services.

Entering

In this case the first scene of the “Auto Start Application” associated with the service produces no visible effect. This first “invisible” scene responds to “Text” either by transitioning to another scene within the same application or by launching a different application.

Leaving

If the application is part of the service then the “return” when “Text” is used a second time can be implemented in various ways:

- TransitionTo the first scene
- Quitting the application (which will automatically be restarted on the first scene)
- Launching the application (which will automatically start on the first scene)
- Service change (resident program) to the information application

If the information application is NOT part of the service (i.e. the “invisible” application launched it) then the options are for the visible application to:

- Launch the invisible application
- Service change (resident program) to the invisible application

If there is a one-to-one relationship between TV service (with invisible “springboard” application) and the information service then the return link can be explicitly coded into the application.

If more than one “TV service” directs its viewers to a single information service (as might be the case with BBC services funnelling their viewers to a single Ceefax service) then each of the “invisible” applications should store appropriate information in persistent storage to allow returning to the point of origin (See “Persistent storage” on page 37).

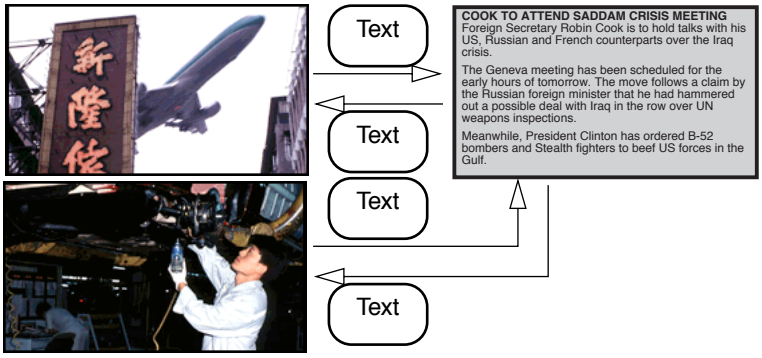


Figure 22. “Text” accesses “teletext”

9.3.2 Accessing Additional Programme Information

In Figure 23 the service by default has a visual prompt that more information is available. Using “Red” reveals the additional information. Using “Text” restores the original presentation (alternatively a coloured key might be used to take the viewer back to the original presentation and the “Text” key might take the viewer to a full-screen text service).

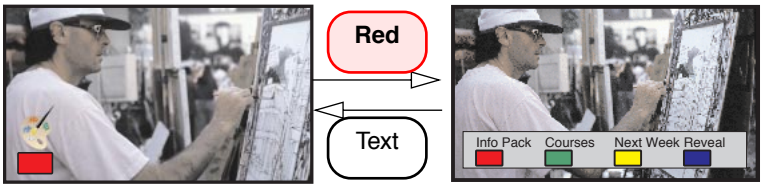


Figure 23. “Text” provides more information

The initial presentation includes a visual prompt that more information is available. Exactly how this is used is the broadcaster’s choice. For example, this graphic might be visible throughout the programme. Alternatively, it be shown for short periods e.g. after the programme is first selected and at the end of each “item” within the programme.

9.3.3 “Text” has no effect

In Figure 24 the TV service has no MHEG application. In this case “Text” is not required to have any effect. However, the receiver may *optionally* provide a response.

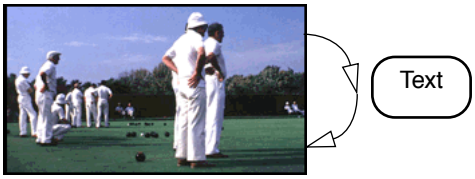


Figure 24. TV with no MHEG application

In Figure 25 a predominantly MHEG service has no related TV service revert to. Alternatively, the application may have been launched by a user channel change and so has no knowledge of any previous service to which it should return. In either case “Text” cannot return the user to a “logical” TV service.

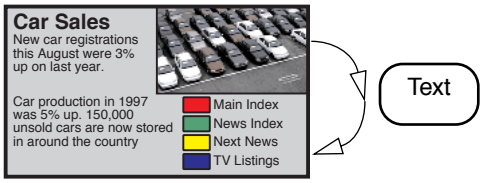


Figure 25. MHEG service has no TV alternate

In the case shown in Figure 25 it may be appropriate for the application to toggle between “selected” and “entered” conditions (even if there is no visible change in the display) to allow the user to navigate to another service using the numeric keys.

9.4 Changing Applications at Programme (Event) boundaries

In some cases MHEG applications will be associated with a particular TV programme. So, as the schedule progresses applications must die and new ones replace them.

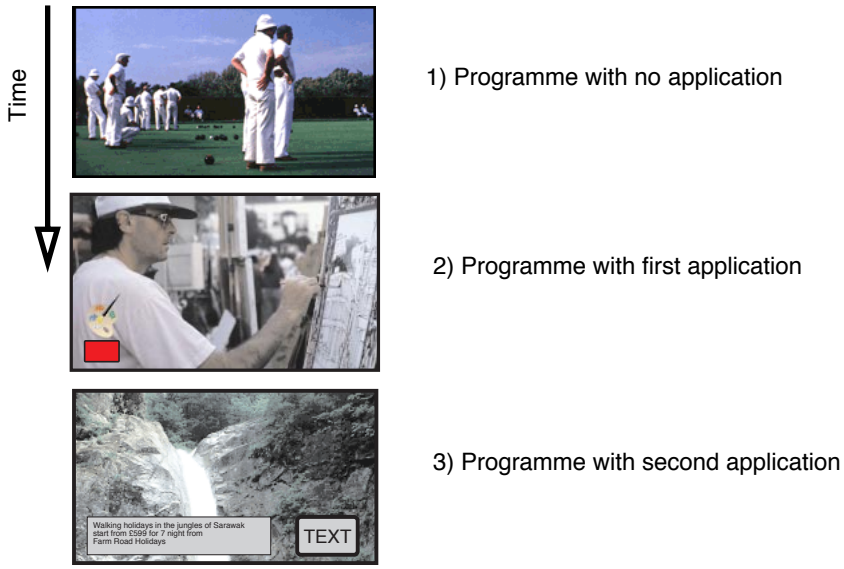


Figure 26. A progression of applications through the schedule

Various strategies are available to create the effect seen in Figure 26. In Figure 27 the receiver's auto start behaviour is used to start an appropriate application at the start of each programme.

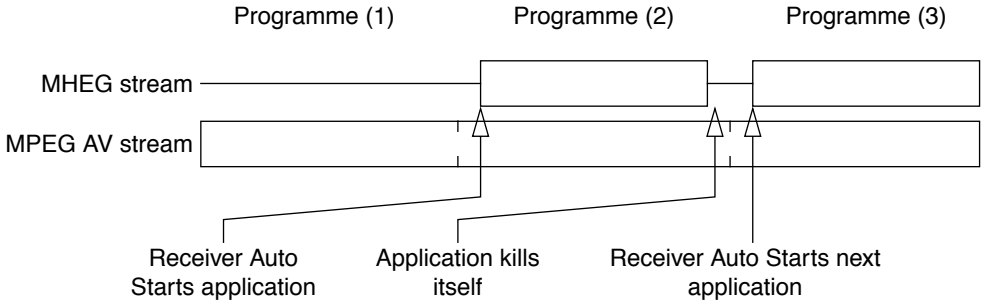


Figure 27. Dead man's shoes method

In Figure 28 the MHEG application carousel operates continuously periodically being updated with new applications. Applications explicitly launch their successor.

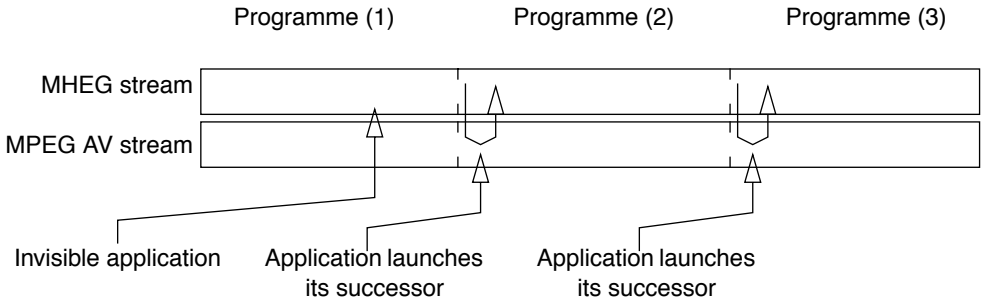


Figure 28. Baton passing method

The receiver's auto start behaviour is still required to start the first application of the day and to start applications when viewers join the service.

9.5 MHEG streams

Each program (service) may contain *at most* one MHEG application designated as an autostart application.

There may be a number of other non-autostart applications. These are always Launched by other applications.

A TS may contain any number of MHEG services.

9.6 Use of the display

9.6.1 Visible Area

Typically a 5% border region is lost due to monitor overscan. This leaves a central 648x518 area of the graphics plane which will normally be visible provided that the user does not configure their equipment in an unusual way. For example, additional area may be lost if a 4/3 service is "zoomed" to file a 16/9 display.

9.6.2 Scene Aspect Ratio

Where an MHEG application is associated with one or more video streams, the scene AspectRatio shall be equivalent to the aspect ratio of the video.

9.7 Use of stream decoders

Receivers conforming to *UKEngineProfile1* are modelled as providing just one of each of the following decoders:

- MPEG video or still picture
- MPEG audio
- DVB Subtitle

To obtain deterministic behaviour author's should not build applications that attempt to activate more than one of each type of decoder. E.g. applications should Stop a running Stream object using MPEG video before Running a Bitmap object with MPEG I-frame content.

See "Numbers of Objects" on page 43.

9.8 Missed events

Applications are not guaranteed to receive all events. For example, stream events and timer events may be missed while an application is paused as a consequence of losing priority access to the display. See "Graphics priority" on page 44.

9.9 File naming

Name length

The length of file references is limited. See "GroupIndentifier" on page 28.

Name character coding

The character set for file references is limited. See "Non-presented text" on page 45.

File names in persistent storage

The <name> part of the file name "ram://<name>" used to access files in persistent storage (See "Persistent storage" on page 69) shall be managed to avoid accidental file name collision between the applications of different service providers.

The length of <name> is defined in "Storage of file names" on page 70. The first characters of these names are allocated to multiplex operators as shown in Table 64 where 'xxxxx'

indicates characters that may be allocated by the multiplex operator to services and applications as they choose.

<name> format	multiplex operator
BBCxxxxx	BBC
D34xxxxx	Digital 3/4
SDNxxxxx	SDN
BDBxxxxx	BDB

Table 64. Format of file names for persistent storage

Where an application (or co-operating applications) is (are) delivered by more than one multiplex operator the application author shall be responsible for liaising with the multiplex operators to obtain a <name> allocated within the space of one operator.

Further names “roots” may be allocated following mutual agreement between all the multiplex operators.

9.10 Text and HyperText encoding

Mark-up coding for Text and HyperText objects is described in “Text Objects” on page 54 and “HyperText Objects” on page 55 is functionally equivalent to a small subset of HTML re-coded to improve transmission efficiency. A simple translation process between the two formats is possible.

HTML mark-up	broadcast mark-up codes
	0x09[a]
	0x20[b]
	0xC2 0xA0
 	0x0D
<P>	
</P>	0x0D 0x0D
<B> [text in bold] </B>	0x1B 0x42 0x00 [text in bold] 0x1B 0x42
<FONT COLOR="#rrggbbtt"> [coloured text] </FONT>	0x1B 0x43 0x04 0xrr 0xgg 0xbb 0xtt [coloured text] 0x1B 0x43
<A HREF="tag bytes"> [anchor text] </A>	0x1B 0x41 0xnn tag_bytes [anchor text] 0x1B 0x41
<	0x3C
>	0x3E
&[c]	0x26

Table 65. Text object mark-up codes

a] Tab characters have meaning See “Tabulation” on page 52.

b] All space characters are significant.

c] And so on. I.e. the HTML “named character entities” can be directly represented with a simple character code where they exist in the “Character repertoire” on page 58.

10 References

10.1 ETSI

[1]	ETS 300 468	Digital broadcasting systems for television, sound and data services; Specification for Service Information (SI) in Digital Video Broadcasting (DVB) systems.
[2]	ETS 300 743	Digital broadcasting systems for television, sound and data services; *Subtitling systems*.
[3]	prETS 300801	Digital Video Broadcasting (DVB); DVB interaction channel through the Public Switched Telecommunications System (PSTN) / Integrated Services Digital Network (ISDN).
[4]	prETS 300802	Digital Video Broadcasting (DVB); Network Independent Protocols for DVB Interactive Services.
[5]	Draft EN 301 192	Digital Video Broadcasting (DVB); DVB specification for data broadcasting
[6]	ETR 154	Digital broadcasting systems for television; Implementation Guidelines for the use of MPEG-2 Systems, Video and Audio in Satellite and Cable Broadcasting Applications.
[7]	ETR 162	Digital broadcasting systems for television, sound and data services; Allocation of Service Information (SI) codes for Digital Video Broadcasting (DVB) systems.
[8]	ETR 211	Digital broadcasting systems for television, sound and data services; Guidelines on the implementation and usage Service Information (SI). August 1997 Second Edition.

10.2 CENELEC

[9]	prEN 50201	Interfaces for DVB-IRDs
[10]	prEN 50221	Common Interface Specification for Conditional Access and other Digital Video Broadcasting Decoder Applications

10.3 ISO/IEC

[11]	ISO/IEC 11172-2	MPEG 1 video
[12]	ISO/IEC 11172-3	MPEG 1 audio
[13]	ISO/IEC 13522-5	MHEG-5 Information technology - Coding of multimedia and hypermedia information: Support for Base-Level Interactive Applications.
[14]	ISO/IEC 13818-1	Information technology - Generic coding of moving pictures and associated audio information: Systems. ISO/IEC 13818-1:1996(E)
[15]	ISO/IEC 13818-2	Information technology - Generic coding of moving pictures and associated audio information: Video.
[16]	ISO/IEC 13818-3	Information technology - Generic coding of moving pictures and associated audio information: Audio.
[17]	ISO/IEC 13818-6	Information technology - Generic coding of moving pictures and associated audio information: Extensions for Digital Storage Media Command and Control.
[18]	ISO 10646-1	Information Technology - Universal Multiple-Octet Coded Character Set (UCS) - Part 1: Architecture and Basic Multilingual Plane. Also see “The Unicode Standard, Version 2.0”, The Unicode Consortium published by Addison-Wesley. ISBN 0-201-48345-9.

10.4 DVB

[19]	DVB A008	DVB Commercial requirements for asymmetric interactive services supporting broadcast to the home with narrow band return channels.
[20]	DVB V2 SI 244	DVB Data Broadcasting Services - Commercial Requirements.
[21]	SI-DAT 382 Rev. 4	Implementation Guidelines for Databroadcasting

10.5 ITU

[22]	ITU-T X.680	INFORMATION TECHNOLOGY – ABSTRACT SYNTAX NOTATION ONE (ASN.1): SPECIFICATION OF BASIC NOTATION Also published as ISO/IEC International Standard 8824-1. ASN.1
[23]	ITU-T X.690	INFORMATION TECHNOLOGY – ASN.1 ENCODING RULES: SPECIFICATION OF BASIC ENCODING RULES (BER), CANONICAL ENCODING RULES (CER) AND DISTINGUISHED ENCODING RULES (DER) Also published as ISO/IEC International Standard 8825-1. DER

10.6 Apple Corporation Inc.

[24] AIFF-C — AIFF-C Audio Interchange File Format, version C, allowing for Compression.

10.7 W3C

[25] HTML 3.2 — HyperText Mark-up Language reference specification, by Dave Raggett, 14-Jan-1997

available at http://www.w3.org/TR/REC-html32.html

[26] PNG — Portable Network Graphics Specification, Version 1.0, 01-October-1996

available at http://www.w3.org/TR/REC-png.html

10.8 DAVIC

[27] DAVIC 1.3 Part 09 — DAVIC 1.3 Specification Part 09, Information Representation, Revision 6.1

[28] DAVIC 14B94R10 — DAVIC 1.4 Baseline document 94, Revision 1.0, “MHEG-5 Resident Programs to access SI.”