



INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

T.171

(10/96)

SERIES T: TERMINALS FOR TELEMATIC SERVICES

**Protocols for interactive audiovisual services:
coded representation of multimedia and
hypermedia objects**

ITU-T Recommendation T.171

(Previously CCITT Recommendation)

ITU-T T-SERIES RECOMMENDATIONS
TERMINALS FOR TELEMATIC SERVICES

For further details, please refer to ITU-T List of Recommendations.

FOREWORD

The ITU-T (Telecommunication Standardization Sector) is a permanent organ of the International Telecommunication Union (ITU). The ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Conference (WTSC), which meets every four years, establishes the topics for study by the ITU-T Study Groups which, in their turn, produce Recommendations on these topics.

The approval of Recommendations by the Members of the ITU-T is covered by the procedure laid down in WTSC Resolution No. 1 (Helsinki, March 1-12, 1993).

ITU-T Recommendation T.171 was prepared by ITU-T Study Group 8 (1993-1996) and was approved by the WTSC (Geneva, October 9-18, 1996).

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

© ITU 1997

All rights reserved. No part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from the ITU.

CONTENTS

	<i>Page</i>	
0	Introduction.....	1
0.1	Application domains for requirements analysis.....	1
0.2	Multimedia/Hypermedia application requirements.....	2
0.3	Rationale for standardisation of multimedia and hypermedia information.....	4
0.4	T.171 Objectives.....	5
0.5	T.171 concepts.....	6
0.6	The MHEG application interface.....	10
0.7	T.171 extensibility.....	10
1	Scope.....	10
1.1	Specificity of the scope.....	11
1.2	Issues outside T.171 scope.....	11
2	Conformance.....	11
2.1	Profiles.....	11
2.2	Syntax.....	11
2.3	Semantics.....	11
3	Normative References.....	12
4	Definitions.....	12
5	Symbols and abbreviations.....	19
	SECTION 1 – OVERVIEW.....	21
6	T.171 principal feature.....	21
6.1	Interchanging multimedia objects.....	21
6.2	The object-oriented approach.....	21
6.3	Technical features.....	23
7	MHEG Engine Assumptions.....	27
7.1	Handling and Interchange of objects.....	27
7.2	The MHEG application interface.....	28
7.3	Exception handling.....	29
8	Methodology.....	29
8.1	Modularity.....	29
8.2	Methodology of representation of MHEG objects.....	29
	SECTION 2 – GENERIC UTILITY AND USEFUL DEFINITION MECHANISMS.....	33
9	Presentation Mechanism.....	33
9.1	Presentation Space (PS).....	34
9.2	Original Presentation Space (OPS).....	37
9.3	Channel Presentation Space (CPS).....	38
9.4	Relative Presentation Space (RPS).....	39
9.5	CPS mapping.....	39
10	Generic identification mechanism.....	40
10.1	External identification.....	41
10.2	Internal identification.....	43
10.3	Symbolic identification.....	47
11	Generic reference mechanism.....	47
11.1	Generic reference using generic identification.....	47
11.2	Predefined references.....	50
11.3	? reference.....	50

	<i>Page</i>	
12	Generic Value.....	52
	12.1 Generic boolean.....	53
	12.2 Generic numeric.....	53
	12.3 Generic integer.....	53
	12.4 Generic ratio.....	53
	12.5 Generic string.....	54
	12.6 Generic reference.....	54
	12.7 Generic list.....	54
13	Macro Mechanism.....	54
14	Hooks.....	56
15	Extensibility.....	57
	15.1 Catalogues.....	57
	15.2 Addition of New MHEG object classes.....	58
	15.3 Extensibility Provision.....	58
SECTION 3 – OVERVIEW OF MHEG CLASSES.....		58
16	MHEG object classes overview.....	58
17	Structure of MH-Object Class.....	59
	17.1 Class identification.....	59
	17.2 MHEG-ID.....	59
	17.3 General object information.....	59
18	Structure of Action Class.....	60
	18.1 Elementary actions.....	60
	18.2 Basic action object.....	61
	18.3 Nested action object.....	61
	18.4 Macro action object.....	61
19	Structure of Link Class.....	61
	19.1 Link Condition.....	62
	19.2 Link Effect.....	71
	19.3 Basic link object.....	71
	19.4 Nested link object.....	71
	19.5 Macro link object.....	72
20	Structure of Model Class.....	72
21	Structure of Script Class.....	72
22	Structure of Component Class.....	72
23	Structure of Content Class.....	73
24	Structure of Multiplexed Content Class.....	73
25	Structure of Composite Class.....	74
	25.1 Availability Start-up.....	75
	25.2 Availability Close-down.....	76
	25.3 RT-Availability Start-up.....	76
	25.4 Rt-Availability Close-down.....	76
	25.5 Composition Element.....	77
	25.6 Composition example.....	77
26	Structure of Container Class.....	78
	26.1 Container Start-up.....	78
	26.2 Container Close-down.....	78
	26.3 Container Element.....	78

	<i>Page</i>
27	Structure of Descriptor Class 79
27.1	Related Object 79
27.2	Other Descriptor 81
27.3	Readme 81
27.4	System Readable Material 81
27.5	Channel Information 81
27.6	Catalogued Style Information 82
27.7	Cat Ext elementary action info 82
27.8	Cat Ext Attribute Info 82
SECTION 4 – MHEG ENTITIES COMMON BEHAVIOUR 82	
28	MHEG entity behaviour 82
29	MHEG entity state definition 82
29.1	MHEG object availability 83
29.2	Link activation 84
29.3	Channel availability 85
29.4	Rt-object availability 86
29.5	Rt-Component running behaviour 88
29.6	Rt-Component presentation behaviour 89
30	Life cycle of MHEG entities 92
31	General action mechanisms 92
31.1	Action treatment 92
31.2	Processing of link effect 93
31.3	Basic processing of an elementary action 95
31.4	Resolution of target set 95
31.5	Arithmetic precision 95
32	Common action effects and handling 96
32.1	Elementary actions 96
32.2	Get actions 96
32.3	Recommended exception handling 97
33	Postpone Behaviour 97
33.1	Behaviour attributes and statuses 97
33.2	Actions to change the behaviour 97
34	Returnability Behaviour 98
34.1	Behaviour attributes and statuses 98
34.2	Actions to change the behaviour 98
35	Alias Behaviour 98
35.1	Behaviour attributes and statuses 98
35.2	Actions to change the behaviour 98
36	Extensibility Behaviour 99
36.1	Behaviour attributes and statuses 99
36.2	Catalogued Attribute 99
36.3	Actions to change the behaviour 99
36.4	Actions to retrieve the behaviour 100
SECTION 5 – MHEG OBJECTS BEHAVIOUR 101	
37	MHEG Objects Availability Behaviour 101
37.1	Behaviour attributes and statuses 101
37.2	Preparation Status 101
37.3	Actions to change the behaviour 101
37.4	Actions to retrieve the behaviour 104

	<i>Page</i>
38	Link Object Activation Behaviour 104
	38.1 Behaviour attributes and statuses 104
	38.2 Activation Status 105
	38.3 Actions to change the behaviour 105
	38.4 Actions to retrieve the behaviour 105
39	Link Object Abort Behaviour 106
	39.1 Behaviour attributes and statuses 106
	39.2 Actions to change the behaviour 106
40	Content Class Generic Value Storage Behaviour 107
	40.1 Behaviour attributes and statuses 107
	40.2 Data 107
	40.3 Actions to change the behaviour 107
	40.4 Actions to retrieve the behaviour 112
41	Content Class Copy Behaviour 113
	41.1 Behaviour attributes and statuses 113
	41.2 Actions to change the behaviour 113
	SECTION 6 – RT-OBJECTS BEHAVIOUR 114
42	Rt-Objects Availability Behaviour 114
	42.1 Behaviour attributes and statuses 114
	42.2 Rt-Availability Status 114
	42.3 Actions to change the behaviour 114
	42.4 Actions to retrieve the behaviour 116
43	Rt-Objects Running Behaviour 117
	43.1 Behaviour attributes and statuses 117
	43.2 Running Status 117
	43.3 Actions to change the behaviour 117
	43.4 Actions to retrieve the behaviour 119
44	Rt-Script Passing Parameters Behaviour 120
	44.1 Behaviour attributes and statuses 120
	44.2 Actions to change the behaviour 120
45	Rt-Scripts Termination Behaviour 121
	45.1 Behaviour attributes and statuses 121
	45.2 Termination Status 121
	45.3 Actions to retrieve the behaviour 121
46	Sockets Presentation and Structural Dynamism Behaviour 122
	46.1 Behaviour attributes and statuses 122
	46.2 Actions to change the behaviour 122
47	Rt-Composite Navigation Behaviour 124
	47.1 Behaviour attributes and statuses 124
	47.2 Rt-Composite Address 124
	47.3 Navigation Command 124
	47.4 Child 124
	47.5 EmptyChild 124
	47.6 Sibling 124
	47.7 Ancestor 124
	47.8 Actions to retrieve the behaviour 125
48	Rt-Components Rps Assignment Behaviour 125
	48.1 Behaviour attributes and statuses 125
	48.2 RPS Assignment 125
	48.3 Actions to change the behaviour 125
	48.4 Actions to retrieve the behaviour 126

	<i>Page</i>
49	Rt-Components Perceptability Behaviour..... 127
	49.1 Behaviour attributes and statuses..... 127
	49.2 Perceptability 127
	49.3 Presentation Priority 127
	49.4 Actions to change the behaviour..... 127
	49.5 Actions to retrieve the behaviour..... 129
50	Rt-Components Temporal Behaviour 129
	50.1 Behaviour attributes and statuses..... 129
	50.2 OD 130
	50.3 POD 130
	50.4 OVD..... 130
	50.5 PVD 130
	50.6 Temporal Termination..... 130
	50.7 PVD Position 131
	50.8 CTP..... 132
	50.9 GTF..... 133
	50.10 Timestone Status..... 133
	50.11 Timestone ID 133
	50.12 Expected OVD Result..... 133
	50.13 Expected PVD Result 133
	50.14 Actions to change the behaviour..... 133
	50.15 Actions to retrieve the behaviour..... 136
51	Rt-Components Spatial Behaviour 141
	51.1 Behaviour attributes and statuses..... 141
	51.2 OS 141
	51.3 POS 141
	51.4 Aspect Ratio..... 141
	51.5 Resizing Strategy 142
	51.6 OVS 143
	51.7 OAP 143
	51.8 OVS Position 143
	51.9 PVS 144
	51.10 OVS Proj Strategy 144
	51.11 PAP 144
	51.12 PVS Position..... 144
	51.13 GSF..... 144
	51.14 Spatial Control..... 145
	51.15 User Spatial Control 145
	51.16 Expected Axis Result Param..... 145
	51.17 Point Type Param 145
	51.18 Actions to change the behaviour..... 145
	51.19 Actions to retrieve the behaviour..... 151
52	Rt-Components Audible Behaviour 157
	52.1 Behaviour attributes and statuses..... 157
	52.2 OV 157
	52.3 CV..... 157
	52.4 PCV 157
	52.5 GVF 158
	52.6 Actions to change the behaviour..... 158
	52.7 Actions to retrieve the behaviour..... 159

	<i>Page</i>
53	Rt-Mux Stream Choice Behaviour 160
	53.1 Behaviour attributes and statuses 160
	53.2 Stream Choice 161
	53.3 Stream Chosen State 161
	53.4 Stream Identification 161
	53.5 Actions to change the behaviour 161
	53.6 Actions to retrieve the behaviour 161
54	Interaction Behaviour 162
	54.1 Behaviour attributes and statuses 163
	54.2 Interaction Type 163
	54.3 Interaction Status 163
	54.4 Selection Status 163
	54.5 Modification Status 164
	54.6 Interaction Ability 164
	54.7 Selectability 164
	54.8 Modifiability 164
	54.9 Min Interact Required 164
	54.10 Max Interact Required 165
	54.11 Number of Interacted Sockets 165
	54.12 Actions to change the behaviour 166
	54.13 Actions to retrieve the behaviour 167
55	Rt-Components Style Behaviour 169
	55.1 Behaviour attributes and statuses 170
	55.2 Style 170
	55.3 Actions to change the behaviour 170
	55.4 Actions to change the behaviour 170
56	Rt-Contents Anchor Behaviour 171
	56.1 Behaviour attributes and statuses 171
	56.2 Actions to change the behaviour 171
	SECTION 7 – CHANNELS BEHAVIOUR 173
57	Channel Availability Behaviour 173
	57.1 Behaviour attributes and statuses 173
	57.2 Channel Availability Status 173
	57.3 Actions to change the behaviour 173
	57.4 Actions to retrieve the behaviour 174
58	Channel Perceptability Behaviour 175
	58.1 Behaviour attributes and statuses 175
	58.2 Channel Perceptability 175
	58.3 Actions to change the behaviour 175
	58.4 Actions to retrieve the behaviour 176
59	Channel Presentation Space Behaviour 176
	59.1 Behaviour attributes and statuses 176
	59.2 Actions to change the behaviour 176
	SECTION 8 – CHANNELS AND RT-COMPONENTS BEHAVIOUR 177
60	Channels and Rt-Components Events Behaviour 177
	60.1 Behaviour attributes and statuses 177
	60.2 Event 177
	60.3 Event Data 177
	60.4 Actions to change the behaviour 177
	60.5 Actions to retrieve the behaviour 178

SECTION 9 – DETAILED REPRESENTATION OF MHEG OBJECTS	179
61 MH-object class representation attributes	179
61.1 MH-object Class	179
61.2 Class Identification	179
61.3 Class ID	180
61.4 Description.....	180
61.5 Name.....	180
61.6 Owner	180
61.7 Version.....	180
61.8 Date.....	180
61.9 Keywords.....	180
61.10 Copyright	180
61.11 Copyright ID.....	180
61.12 Copyright Number.....	180
61.13 Licence.....	180
61.14 Cache Priority	181
61.15 Comments	181
62 Action class representation attributes	181
62.1 Action Class.....	181
62.2 Synchro Indicator Param	181
62.3 Synchro Indicator Macro	181
62.4 Synchro Indicator.....	182
62.5 Synchronised Action.....	182
62.6 Action Object.....	182
63 Link class representation attributes	182
63.1 Link Class	182
63.2 Link Condition.....	183
63.3 Trigger Condition	183
63.4 Constraint Condition.....	183
63.5 Source Value.....	183
63.6 Comparison Operation.....	183
63.7 Comparison Value	183
63.8 Previous Condition	183
63.9 Current Condition	183
63.10 Comparison Operator.....	183
63.11 Logical Combination	184
63.12 Logical Operator.....	184
63.13 Condition	184
63.14 Link Effect.....	184
63.15 Macro Parameter Resolution	184
63.16 Usage Value.....	184
64 Model class representation attributes	184
64.1 Model Class	184
65 Script class representation attributes	185
65.1 Script Class	185
65.2 Script Classification.....	185
65.3 Script Data	185
65.4 Script Inclusion.....	185
65.5 InterchangedScript.....	185
66 Component class representation attributes	185
66.1 Component Class.....	186

	<i>Page</i>
67	Content class representation attributes 186
67.1	Content Class 186
67.2	Content Data 186
67.3	Data Inclusion 186
68	Multiplexed content class representation attributes 186
68.1	Multiplexed Content Class 187
68.2	Multiplexed Stream 187
69	Composite class representation attributes 187
69.1	Composite Class 187
69.2	Availability Start-up 188
69.3	Availability Close-down 188
69.4	Rt-Availability Start-up 188
69.5	Rt-Availability Close-down 188
69.6	Link Object 188
69.7	Nb of Elements 188
69.8	Composition Element 188
69.9	Element Index 189
69.10	Associated Model 189
69.11	Label 189
70	Container class representation attributes 189
70.1	Container Class 189
70.2	Container Start-up 189
70.3	Container Close-down 190
70.4	Container Element 190
71	Descriptor class representation attributes 190
71.1	Descriptor Class 191
71.2	Related Object 191
71.3	Object Information 191
71.4	Object Size 191
71.5	Class Specific Information 191
71.6	Script Class Information 191
71.7	Content Class Information 192
71.8	Mux Content Class Info 192
71.9	Number of Streams 192
71.10	Stream Information 192
71.11	Alternative Object 192
71.12	Alternative Descriptor Object 192
71.13	Alternative Readme 192
71.14	Offset 192
71.15	Other Descriptor 192
71.16	Readme 192
71.17	System Readable Material 192
71.18	Channel Information 193
71.19	X min 193
71.20	X max 193
71.21	Y min 193
71.22	Y max 193
71.23	Z min 193
71.24	Z max 193
71.25	X Resolution 193
71.26	Y Resolution 193

	<i>Page</i>	
71.27	Z Resolution.....	193
71.28	T Resolution.....	193
71.29	F min.....	194
71.30	F max	194
71.31	Audio Dynamic.....	194
71.32	Channel Media Type.....	194
71.33	Event Mapping.....	194
71.34	Catalogued Style Information	194
71.35	Cat Ext elementary action Info	194
71.36	Cat Ext Attribute Info	194
72	Behaviours.....	194
72.1	Postpone behaviour.....	194
72.2	Returnability behaviour	195
72.3	Alias behaviour.....	196
72.4	Extensibility behaviour.....	196
72.5	Mheg objects availability behaviour.....	197
72.6	Link object activation behaviour	197
72.7	Link object abort behaviour.....	198
72.8	Content class generic value storage behaviour	198
72.9	Content class copy behaviour	199
72.10	Rt-objects availability behaviour	200
72.11	Rt-objects running behaviour	200
72.12	Rt-script passing parameter behaviour.....	201
72.13	Sockets presentation and structural dynamism behaviour	201
72.14	Rt-components rps assignment behaviour	202
72.15	Rt-components perceptability behaviour	202
72.16	Rt-components temporal behaviour.....	203
72.17	Rt-components spatial behaviour.....	205
72.18	Rt-components audible behaviour	207
72.19	Rt-mux stream choice behaviour	208
72.20	Interaction behaviour	209
72.21	Rt-components style behaviour	210
72.22	Rt-contents anchor behaviour	210
72.23	Channel availability behaviour	211
72.24	Channel perceptability behaviour	211
72.25	Channel presentation space behaviour.....	212
72.26	Channels and rt-components events behaviour.....	212
73	Elementary Actions	213
73.1	List of elementary actions.....	213
73.2	MHEG entity, data, stream, macro parameter and identification useful definitions.....	214
73.3	References useful definitions.....	217
73.4	Useful definitions of targets.....	222
73.5	Generic value useful definitions	226
73.6	Evaluated values useful definitions	230
73.7	Hooks.....	238
73.8	Extensibility	238
73.9	Presentation space useful definitions	242
73.10	Constants useful definitions.....	249
73.11	Comparison value constants	256

	<i>Page</i>
Annex A – ASN.1 notations (Level C) Coded representation (Level D)	257
Annex B – Examples of MHEG systems	291
B.1 Example of an MHEG Engine	291
B.2 Application examples	295
Annex C – Interfaces to media Recommendations and Standards	302
C.1 Example of still image content object.....	302
C.2 Example of audio content object	303
Annex D – Hypertext/Hypermedia Support	303
D.1 Introduction	303
D.2 Mechanism for Hypertext/Hypermedia	304
D.3 An example of interrelationships between MHEG and WWW browser	305
Annex E – Examples of spatial behaviours	306
E.1 Example 1	306
E.2 Example 2	311
Annex F – Summary of Object Identifiers	313
Annex G – Index	314

SUMMARY

This Recommendation provides an encoding for multimedia/hypermedia information to be used and interchanged by applications in a wide range of domains.

PROTOCOLS FOR INTERACTIVE AUDIOVISUAL SERVICES: CODED REPRESENTATION OF MULTIMEDIA AND HYPERMEDIA OBJECTS

(Geneva, 1996)

0 Introduction

This Recommendation provides an encoding for multimedia/hypermedia information to be used and interchanged by applications in a wide range of domains.

The design of this Recommendation has been driven by an analysis of application requirements to identify the generic aspects of these applications and incorporate them in a simple encoding.

0.1 Application domains for requirements analysis

In recent years, an explosion of multimedia applications and services has been apparent in many domains. This subclause illustrates the very wide range of audiovisual computer-based applications on which the requirements analysis has been made.

- **Training and education (also known as teletraining):** The extension of data processing and telematics to support audiovisuals makes it a more attractive tool to educators because it improves the interface with the students. There is also a need to exchange audiovisuals between tools and to reuse audiovisuals in other applications. The audiovisuals must be structured in such a way that they can be updated, modified and personalised easily.
- **Simulation and games:** These can be made much more realistic by the use of multiple media in computer-based systems. Examples are flight and maritime simulators; these applications are highly interactive.
- **Sales and advertising:** The association of attractive audiovisuals, showing the products to best advantage, with the stock control and ordering process is a natural commercial evolution.
- **Office information systems, engineering documentation:** There is a long-standing need to be able to integrate drawings, images, audio and video into all types of office documentation. Integrated documents would also provide an attractive means of internal enterprise communication.
- **Culture:** The computer processing of audiovisual information makes it possible to extend the audience for theatres, museums and cultural events. It also opens the possibility of new art forms.
- **Electronic publishing and electronic books:** Much material that is currently published on paper could be enhanced by the integration of audiovisuals and by being available in a hypertext manner over a telecommunication network, for example, tourist guides and yellow pages, which could be a mixed sales and advertising application.
- **Public information:** The use of audiovisual interfaces makes it possible for the public to use computer-based, real-time information systems, for example, a kiosk at which information is accessed through videotex or locally from CD.
- **Computer-supported multimedia cooperative work:** There is a need to form effective teams of people who are geographically separated but want to share text, graphics, image, audio and other information, for example, remote maintenance, joint authorship or multimedia electronic mail.
- **Medical applications:** There is a need for a timely retrieval of audiovisual information, including medical images, medical data and case history, in an integrated way from a remote site. This could be a mixed cooperative work application with different medical experts.

¹⁾ This text is technically aligned with ISO/IEC 13522-1.

- **Interactive television applications:** The emergence of digital television systems on terrestrial, satellite or cable networks, as well as the availability of New modulation and transmission techniques on the telephone network, creates favourable conditions for a New offer of interactive multimedia services based on the merging of television and telematics technologies, providing the users with interactive access to video documents.
- **New classes of applications:** The widespread availability of remote access to audiovisual data processing systems makes possible New classes of applications such as interactive television, customised news programmes and archive materials.

In the following subclauses, the specific needs of communicating multimedia/hypermedia applications from these domains are examined, leading to a list of application requirements. This set of requirements constitutes the global framework of this Recommendation.

0.2 Multimedia/Hypermedia application requirements

In the various domains presented above, interactive multimedia systems are perceived to have increased impact on users through their use of image, video and audio, in addition to traditional text information.

The development of these systems is based upon support for multimedia information in the following areas:

- **Platforms:** Emerging compression standards, such as JPEG (ISO/IEC 10918) or MPEG (ISO/IEC 11172 and ISO/IEC 13818), have enabled the development of dedicated hardware, which, in the future, will be provided as a standard system resource.
- **Storage:** The increasing capacity of magnetic and optical disks, combined with compression techniques, enables the storage of increasing amounts of high bandwidth information.
- **Communication:** The increasing availability and bandwidth of digital transmission media, such as ISDN and other broadband telecommunication and broadcasting networks, enables the appropriate exchange of large amounts of data.

Using these facilities, it is anticipated that multimedia applications will be designed to Run on heterogeneous platforms and will be interconnected to offer multimedia services.

These multimedia applications and services will use large quantities of structured multimedia objects resident in workstations, stored on digital interchange medium and retrieved or distributed from remote sources through a network. These multimedia data will represent a significant investment and it is vital that the information be applicable in a world of rapidly evolving systems and technologies. In particular, it is important that information should be interchangeable between the data structures supported by different applications.

The following application requirements can be identified:

- Multimedia database retrieval.
- Frequent updates of multimedia data.
- Manipulation of a set of data elements.
- Creation of multimedia documents on a range of workstations.
- Composition of multimedia data in time and space.
- Ability to link part of a document with some other part of the same or another document in an open hyperdocument environment.
- Synchronisation.
- Elementary synchronisation: Two objects are synchronised with regard to the same reference origin time (parallel mode) or one with regard to the other (sequential mode).
- Chained synchronisation: A set of objects are presented one after another in the form of a chain.
- Cyclic synchronisation: One or more objects are repetitively presented.
- Conditional synchronisation: The presentation of an object is linked to the satisfaction of a condition.

- Reuse of multimedia data by integration in different documents.
- Exchange of multimedia data between heterogeneous systems.
- Space wide range of users, including closed user groups.
- Real-time interactivity, including acquisition of multimedia data.
- Telecommunication of multimedia data.
- Broadcast of multimedia data.
- Wide range of data volumes and transfer rates.
- Access control, security.
- Tariffing.
- Copyright, licensing.
- Wide range of support materials.
- Use of a wide range of terminals and workstations, including devices with minimal resources.
- Minimal resources: Facilities that the interchange form must provide in order for a given set of objects, running in limited resources environments, to meet their functional specification as determined by the designer.
- Ease of specification: For specifying the minimum recommended resources needed by the MHEG application.
- Ease of application: Minimum requirements of an application should be easily recognised by the negotiation process.
- Flexibility: The mechanism should be reliable across a broad range of system configurations and media formats.
- Informative rather than restrictive: The using application should have the final determination as to whether a given set of objects can be interpreted.
- Extensible/open: The representation should function correctly as New formats, classes, and application-dependent parameters are added.
- Scalability: Trade-off among different application platform capabilities, for example, resolution enhancement relative to presentation time.
- Composition: The minimum resource requirements for a given MHEG application should be combinable in some consistent and predictable manner when applications are combined.
- Graceful degradation: The primary concern of minimal resource systems is the manner in which degradation can be handled; an application should be able to express its policy with regard to this issue, for example, by using scalability and related mechanisms.
- Real-time interchange and presentation.
- Object placement optimisation: Objects that are likely to be accessed simultaneously are adjacent from the standpoint of the access mechanism.
- Progressive access of objects: Images may be retrieved and presented in increasing resolution for systems with significant presentation delay; scalable versions of objects may be represented and retrieved for systems with insufficient resources for full fidelity presentation.
- Partial object retrieval: Large objects may be retrieved in several portions, since the entire content will not be presented at one time.
- Object sequencing: The order in which objects are expected to be presented should be used by the access mechanism when insufficient throughput would lead to unacceptable delays for the whole object to be interchanged.

- Separate retrieval of object description and object content: The object description should be retrieved without necessarily retrieving the content so that the system can use information about a set of objects to optimise the access for this set and resources needed for the access can be prepared.
- Global object index: A table of all objects and their position in an object set should be provided to support fast lookup of objects.
- Object interleaving: Objects that are to be retrieved simultaneously may be interleaved so that large objects do not cause delays for other objects.
- Resource requirements: Those for retrieval and presentation by the target system should be available by lookup rather than by derivation.
- Aggregate retrieval: A collection of objects can be aggregated so that they can be retrieved by one step rather than through a series of requests.
- Ability to reference individual media objects internally and externally.
- Ability to navigate in a hypermedia fashion.
- Ability to support arbitrary hyperlink transversal schemes.

These requirements can be summarised in a set of generic needs:

- application designers and distributors require application portability in a multi-vendor environment;
- applications must handle multimedia information structured in such a way that real-time interactivity (including acquisition of multimedia data), as well as real-time interchange of multimedia data can be ensured;
- applications must compose and synchronise multimedia data in space and time;
- applications must be able to define links between multimedia data elements;
- applications must be able to reuse multimedia data by integration in different contexts;
- applications must be able to frequently update the multimedia data, as well as to manipulate a set of data elements;
- applications must be able to be interpreted on different systems, from minimal resources to non-limited resources systems;
- applications must be interchanged and presented in real-time.

These requirements have led to the definition of this Recommendation for multimedia and hypermedia information.

0.3 Rationale for standardisation of multimedia and hypermedia information

The rationale for a standard defining multimedia and hypermedia information structures is based on the following considerations.

- Standardisation, only at the level of monomedia information is not sufficient to guarantee application portability. Applications do not use the monomedia data separately but must define an associated set of parameters that are necessary for presentation. (Such parameters may include identification of the encoding algorithm applied to the data, decoding parameters relevant to the data, optional attributes to be used for presentation).
- Standardisation, only at the level of monomedia information, is not sufficient for the design and interchange of multimedia and hypermedia information. Multimedia and hypermedia applications rely on abstractions or data structures that provide features such as synchronisation links and logical links between monomedia data.
- The design and management of multimedia and hypermedia applications in distributed environments will be eased if the internal details of the information presentation are masked from the application by the use of appropriate abstractions. The application should deal only with functions such as management of information distribution, scheduling of presentation, management of the user dialogue, and other high-level activities.

This Recommendation aims to provide generic multimedia/hypermedia information structures that fulfil these requirements and additionally are suited to:

- real-time presentation using multimedia presentation and synchronisation facilities available on the application platform;
- real-time interchange using communication facilities available on the application platform;
- final-form representation in which the information is represented and coded for direct presentation without requiring additional processing of its structure.

0.4 T.171 Objectives

0.4.1 Interchange

This Recommendation is intended to provide interchange facilities for various media types. The media data may be encoded according to other international standards, e.g. JPEG for still image, MPEG for video, or may be encapsulated using private and proprietary coding techniques. The interchange units defined by this Recommendation are handled by the MHEG engine under the control of a using application.

In order to support multimedia interchange, as opposed to the interchange of multiple media, this Recommendation provides structures for the composition of different media types within a single unit of interchange.

0.4.2 Presentation

This Recommendation is intended to support final-form presentation of multiple media types. This Recommendation provides facilities for the identification of the coding technique to enable the use of the appropriate presentation resources on a specific platform.

In order to support multimedia presentation, as opposed to the presentation of multiple media, this Recommendation provides structures for the composition of different media types in a presentation. This composition takes the form of time sequencing, spatial positioning, and logical interaction between the media.

In addition, this Recommendation supports interaction with, and modification of, media data and its associated presentation attributes, e.g. size, position, and volume.

Figure 0.1 illustrates the use of this Recommendation as an interchange unit and as an identifier of multimedia information encoded according to several International Standards.

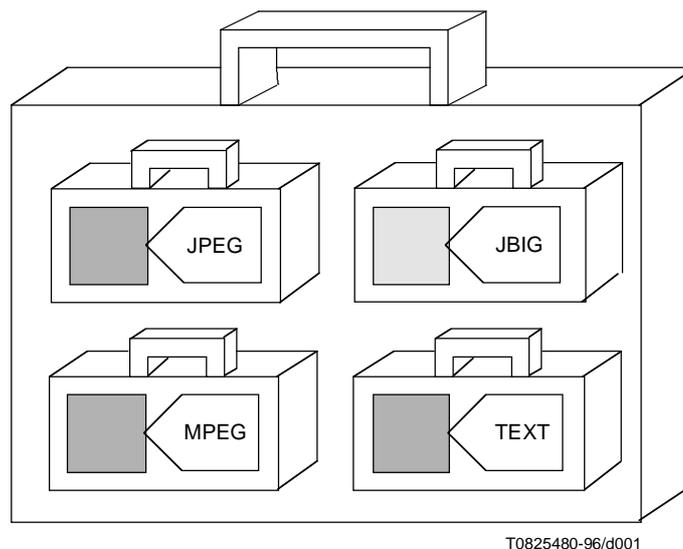


Figure 0.1/T.171 – Recommendation T.171 for interchange and identification of multimedia information

0.4.3 Minimal resources

This Recommendation is intended to support the specification of the minimal resources required to present the encoded data and provides facilities for the interchange of information relating to these resources, which is provided by the information source.

0.4.4 Real time

This Recommendation is intended to ease the real-time interchange of multimedia information and provides facilities to assist a using system to achieve an adequate information flow.

0.5 T.171 concepts

This subclause provides a general introduction to the T.171 concepts. The following concepts are presented:

- **MHEG classes; MHEG objects:** This Recommendation is a standard that defines object classes. From these classes, MHEG objects may be instantiated by the object designer (e.g. a computer program) and interchanged between using applications. Any number of MHEG objects may be instantiated from a given MHEG class.
- **Run time objects (rt-objects):** The MHEG model class is an MHEG abstract class, which provides the following MHEG classes: script, content, multiplexed content, and composite classes. From these model classes, model objects may be instantiated by the object designer and interchanged. In order to reuse the data interchanged in the model objects in different contexts, the object designer is able to create run-time objects, also called rt-objects, from a given model object: for example, the same data in a content object may be presented twice at the same time with different sizes using two rt-content objects, but the content object will be interchanged only once. Any number of rt-objects may be created from a given model object by the object designer.
- **Channels:** This Recommendation defines logical spaces in which the rt-objects are presented by the user.

0.5.1 Object orientation

This Recommendation defines a classification of structures corresponding to units of multimedia/hypermedia information to be interchanged. This classification is based on an analysis of the common behaviour and properties of multimedia/hypermedia information and takes the form of an object-oriented approach to the standardisation. This approach results in autonomous and reusable information structures that are generic to multimedia/hypermedia applications.

In the context of this Recommendation, the term “class” denotes a structure of interchangeable multimedia/hypermedia information from which MHEG objects can be instantiated. This Recommendation makes use of the object-oriented mechanism of deriving subclasses from existing ones, together with the associated concept of inheritance. However, since MHEG objects have to be considered as data rather than executable code, no interference MHEG objects in the form of method signatures is defined.

It should be understood that the use of an object-oriented approach to the definition of MHEG structures, does not imply that a system interpreting these structures must be based upon this approach. While this Recommendation defines an interchange format for multimedia/hypermedia information, it makes no assumptions about the internal representation of MHEG structures or the design of systems, engines, interpreters, tools, or using applications.

The MHEG classes contain a level of complexity that is appropriate for use as generic structures in a wide range of using applications and domains. It is assumed that the semantics associated with the use of MHEG classes are defined at the application level and not at the level of this Recommendation. When using applications interchange multimedia/hypermedia information, they usually rely on interchange Services, which provide the appropriate support for the interchange of data. There are various levels of complexity of data interchange, as shown in Figure 0.2.

In Figure 0.2, applications A and B may be seen as a single application by the user.

Figure 0.2 shows interchange between using applications A and B and identifies the levels at which this interchange takes place.

- Application level: This exchange (Appl) is not covered by this Recommendation. The using application may make use of a script interchange Standard at the next lower layer.
- Script level: This exchange (S) is covered by ISO/IEC 13522-3 and other Standards for hyperdocument and scriptware interchange and may make use of the MHEG object interchange Standard at the next lower layer.
- MHEG object level: This exchange (M) is the subject of this Recommendation. It makes use of Recommendations and Standards for the interchange of content data.
- Non-MHEG content data level: This exchange (C) is addressed by individual monomedia Recommendations and Standards.
- Other protocol element level: This exchange (OPE) of elements such as messages and acknowledgements is required by the application but is not addressed by this Recommendation.

This model also shows the need to address the binding of the script to the MHEG objects and of MHEG objects to the content data.

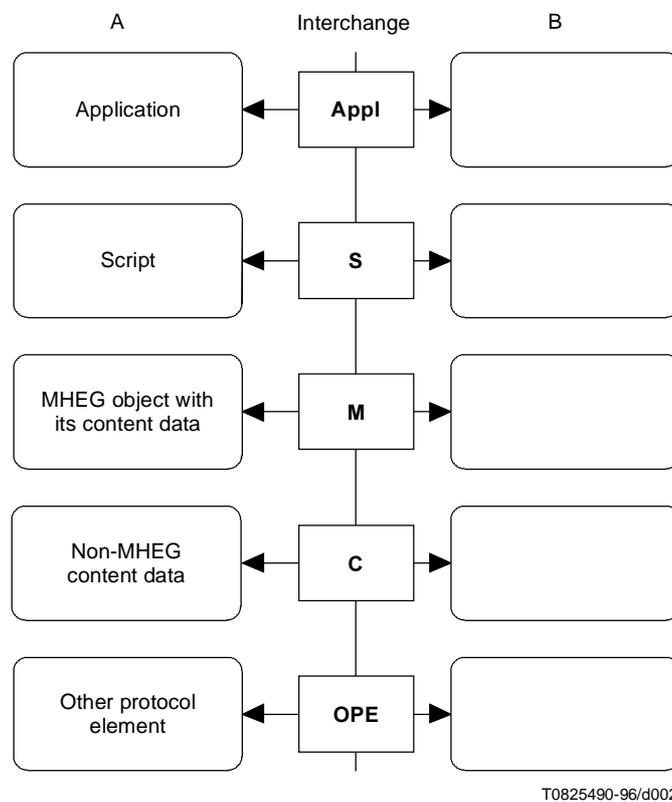


Figure 0.2/T.171 – Multimedia/hypermedia interchange model

0.5.2 Encoding

This Recommendation provides a coded representation of each MHEG class. Instances of these coded representations are MHEG objects and represent the data to be interchanged and presented by multimedia/hypermedia applications.

The base-coded representation makes use of Recommendation X.680 “Specification of Abstract Syntax Notation One” (ASN.1) and Recommendation X.690 “Specification of Basic Encoding Rules for ASN.1” and is described in this Recommendation.

Alternative coded representations may be provided. Their coded representation should be isomorphic to the base-coded representation. These different notations and encoding schemes are alternatives used to express the same and unique representation of the MHEG objects.

0.5.3 Overview of the MHEG classes

The classes defined in this Recommendation can be used to specify:

- objects containing media information;
- relationship between objects;
- dynamic behaviour of objects;
- information to optimise the real-time handling of objects.

The following instantiable classes are defined in this Recommendation.

0.5.3.1 Content class

The content class is a model class; it contains or refers to the coded representation of media information together with a parameter set containing information required for content presentation. This parameter set contains an identification of the coding method and a field for the specification of application-oriented parameters (e.g. fonts and colour table). The content class also specifies the original size, duration and volume of the data. These values are expressed using generic space and temporal units. From content class, multiplexed content objects can be derived; content objects that are not multiplexed are called pure content object or non-mux content object.

0.5.3.2 Multiplexed content class

The multiplexed content class is a model class that is a subclass of the content class. It contains or refers to the coded representation of a multiplexed media data together with a description of each multiplexed stream.

0.5.3.3 Composite class

The composite class is a model class; it provides the support for associating multimedia and hypermedia objects. This mechanism provides a consistent approach to the linking and synchronisation in time, space, of a set of objects. This class also provides a logical structure to describe the list of possible interactions offered to the user, but does not define the interaction facilities provided by the user interface. Such interaction may be achieved in a variety of ways, e.g. graphical user interface and keyboards. This Recommendation does not define the look and feel of multimedia interactive presentations, nor does it propose to change or add concepts to those existing in typical graphical user interfaces. As this Recommendation is generic and independent of platform and implementation, it describes interaction at a virtual level. It is for a using application to apply these mechanisms using its specific look and feel.

0.5.3.4 Action class

This Recommendation provides an initial behaviour for each MHEG object, run-time object (rt-object), and channel, e.g. default position of an rt-content and provides also a means to modify the initial behaviour of each object by defining a list of elementary actions to be applied to the objects. The modification of the behaviour is achieved by interchanging the corresponding elementary actions within action objects. The action objects are used within a link object to describe the link effect.

The action class defines a structure that specifies a synchronised set of elementary actions to be applied to one or more targets.

0.5.3.5 Elementary actions

This Recommendation defines elementary actions that can affect the following behaviours of an MHEG object, an rt-object, or a channel.

- **Preparation:** Actions are provided to control the availability of the MHEG object in the system. For example, Prepare and Destroy actions may be applied to add and remove an MHEG object from the system.
- **Creation of run-time objects (rt-objects):** The New action is provided to create rt-objects (rt-script, rt-component) from a model object (script, component); the Delete action is provided to destroy them.

Presentation: Actions are provided to control the progress of the rt-components in the system. For example, Run and Stop actions may be applied to control the progression of a time-based rt-component.

- **Rendition:** Actions are provided to control the projection of the rt-component on the system. These actions vary according to the media type. For example, Set GTF action to change the presentation speed for time-based media and Set OVS action to change the presentation size for visible media.
- **Interaction:** Actions are provided to control the results of interaction with an rt-component in the system. For example, Set Interaction Ability action specifies the selectability and the modifiability of an rt-component.
- **Activation:** Actions are provided to control the activation of the rt-scripts in the system. For example, Run and Stop actions.

0.5.3.6 Get actions

This Recommendation also defines a means to retrieve the behaviour of an MHEG object, an rt-object or a channel. Actions are provided to get the behaviour attribute or status value of MHEG objects, rt-objects and channels. These actions are called “get actions”. The result of a get action is a generic value or the value “undefined” when an error condition occurs. The get actions are used to express the source and the comparison value of the link condition in a link object or as any parameter of an elementary action.

0.5.3.7 Link class

The link class defines a structure that specifies a set of relationships. Each relationship is defined between one or more sources and one or more targets. The relationship is composed of conditions associated with the sources (link condition) and actions to be applied to the targets (link effect). The actions, which are described in action objects, are to be applied to the targets when the conditions are satisfied.

The source and target can be instances of any MHEG class, including link class and action class. The source and targets can also be any run-time objects.

Instances of the link class are used to specify the time sequencing, spatial positioning, or logical interaction between MHEG objects and run-time objects.

0.5.3.8 Script class

The script class is a model class. It defines a container for complex relationship between MHEG objects, run-time objects and channels, specified in a script language. This Recommendation does not define the script language itself but provides the script class to encapsulate a script and an indication of the language used.

It is assumed that the script language used in a script object is able to reference MHEG objects, rt-objects, and channels, and also to access their attributes. Recommendation T.174 defines an interface to MHEG entities for this purpose and Recommendation T.173 defines a Script interchange Format.

0.5.3.9 Descriptor class

The descriptor class defines a structure for the interchange of resource information about a single or a set of other interchanged objects. The described objects are called “related” objects. The information can be used to facilitate a correspondence between the resources required to present the objects and the resources available to the system, or to perform a negotiation between the source of the MHEG objects and the presentation site.

0.5.3.10 Container class

The container class provides a means to regroup multimedia and hypermedia data in order to interchange them as a whole set.

0.5.4 Run-time objects (rt-objects)

For the purpose of reusing model objects (script, component objects) in different presentations or activation, a clear separation is made between the interchanged model object, which contains the reusable data or composition, and the rt-object corresponding to a specific view of the model data or composition.

The rt-objects (rt-script, rt-content, rt-multiplexed content, and rt-composite objects) are created by the object designer using the MHEG action New. The presentation or activation of an rt-object does not affect the model object; this allows the reuse of a same model object in different rt-objects.

0.5.5 Channels

A channel is a logical space in which the rt-components (rt-content and rt-composite objects) are presented and perceived by the user. The channels are created by the object designer using an MHEG action. The object designer may provide information in the descriptor object to facilitate the mapping by the MHEG engine of the logical channels to the real world.

0.6 The MHEG application interface

This Recommendation does not define an Application Programming Interface (API) for the handling of objects in a system. However, it is assumed that the MHEG engine provides facilities to support the actions defined in this Recommendation for use by the using application.

0.7 T.171 extensibility

The scope of the classes defined in this Recommendation is compatible with their use in a wide range of applications and domains. It is recognised that certain applications may require specific functionality not directly provided by the classes, e.g. a using application may require the use of specialised resources available on a given platform. In this case, additional resources, which are not provided directly by this Recommendation, may be associated with the MHEG objects. This association is defined by using one of the following techniques. These techniques do not change the coded representation of the facilities defined by this Recommendation (see clause 15):

- 1) extension of elementary actions;
- 2) extension of attributes of an MHEG object, an rt-object or a channel;
- 3) extension of data types, classifications, styles, events, channel media types;
- 4) addition of new object classes.

NOTE – This facility may be used for example to create and manipulate new attributes such as payment or colour, or to vary some speech modulation parameter in an audio object.

For 1) through 3), a registration procedure and catalogues are provided by ISO/IEC 13522-4. This allows users to define extensions without significant reduction of portability. This extension allows users to use proprietary or registered catalogues (see clauses 14 and 15.1). An MHEG engine can take any catalogue entry into account and thereby introduces the New extension process; another MHEG engine with no extensions may ignore the extension.

The addition of New object classes implies an appropriate adaptation of MHEG engine for interpretation of the new classes, which limits the usage of MHEG objects of these classes to that MHEG engine.

1 Scope

The scope of this Recommendation is the coded representation of final-form multimedia and hypermedia information objects that will be interchanged as units within or across services and applications, by any means of interchange (e.g. storage, local area network, wide area telecommunication, broadcast telecommunications or broadcast networks). These objects define the structure of a multimedia presentation.

These objects, hereafter called MHEG objects, provide the following functionalities supporting for:

- final-form representation;
- systems with minimal resources;

- interactivity and multimedia synchronisation;
- real-time presentation;
- real-time interchange,

as a common base for many multimedia and hypermedia applications.

This Recommendation defines the specifications of MHEG objects common to all types of coded representation and also defines one type of coded representation, the base-coded representation.

1.1 Specificity of the scope

Since it is expected that a wide range of recommendations, standards, and user applications will be users of this Recommendation, the scope focuses on the generic structuring aspects. This Recommendation recognises the semantics implied by the specification of the MHEG objects but does not enforce any semantic interpretation by the using application.

1.2 Issues outside T.171 scope

The scope excludes any standardisation of models, services, systems, protocols or applications that are likely to make use of MHEG objects. Integration of MHEG objects within these models, services, systems, protocols or applications, or interworking, are defined by other standardisation bodies within JTC1 or ITU-T. The generic part of the interface between MHEG objects and applications was defined through liaison and cooperation with those bodies.

The coded representation of content data is not in the scope of this Recommendation.

2 Conformance

The conformance is evaluated on interchanged objects. There is no consideration of conformance for a system, an engine, a process.

An interchanged object is conformant if it can be correctly interpreted. This means that:

- its ASN.1 encoding is correct (conformance of the concrete syntax);
- its ASN.1 notation is conformant to the grammar specified in this Recommendation (conformance of the abstract syntax).

A conforming MHEG decoder shall recognise and decode all MHEG objects as defined in this Recommendation.

2.1 Profiles

Profiles may be provided by the using applications.

2.2 Syntax

Conformance is on the base representation (this Recommendation). An MHEG object instance is represented using the ASN.1.

Conformance at the level of the coded representation is ensured by the use of international standards for the encoding of the syntax.

2.3 Semantics

This Recommendation provides an encoded representation for expressing an expected behaviour of objects, such as relationship in time and space, but does not define techniques that might handle such relations between the objects.

3 Normative References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; all users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published.

- ITU-T Recommendation X.680 (1994) | ISO/IEC 8824-1:1995, *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation.*
- ITU-T Recommendation X.681 (1994) | ISO/IEC 8824-2:1995, *Information technology – Abstract Syntax Notation One (ASN.1): Information object specification.*
- ITU-T Recommendation X.682 (1994) | ISO/IEC 8824-3:1995, *Information technology – Abstract Syntax Notation One (ASN.1): Constraint specification.*
- ITU-T Recommendation X.683 (1994) | ISO/IEC 8824-4:1995, *Information technology – Abstract Syntax Notation One (ASN.1): Parameterization of ASN.1 specifications.*
- ITU-T Recommendation X.690 (1994) | ISO/IEC 8825-1:1995, *Information technology – ASN.1 encoding rules: Specification of Basic Encoding Rules (BER). Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER).*
- ITU-T Recommendation X.691 (1995) | ISO/IEC 8825-2:1995, *Information technology – ASN.1 encoding rules – Specification of Packed Encoding Rules (PER).*
- ISO/IEC 13522-4:1996, *Information technology – Coding of multimedia and hypermedia information – Part 4: MHEG registration procedure.*

4 Definitions

For the purposes of this Recommendation the following definitions apply.

- Definitions as in Recommendations X.680 and X.690.
- Definitions in this clause.

4.1 abstract class: An MHEG class that cannot be instanced, i.e. no interchanged objects. These classes encapsulate information and provide inheritance of attributes in its subclasses, and also act as a mechanism for specifying which MHEG actions may be applied to which MHEG classes.

4.2 action: An elementary action or a get action.

4.3 action effect: Equal to the concatenation of the MHEG effect and the user effect for all elementary actions. For get actions, it is equal to the MHEG effect.

4.4 action object: An instance of an MHEG class that defines an organised set of elementary actions.

4.5 activate; (active): An elementary action that causes a link object to become active, i.e. only an active link can be fired.

4.6 alias: An attribute that can be associated to an MHEG entity, a data, or a stream. It is used as an alternative identification.

4.7 anchor: A content object containing anchor information rather than media data. Anchor information is used to position an anchor against a hub document.

4.8 attribute: Typed value representing some characteristics of an interchanged MHEG object, an rt-object or a channel.

4.9 author; object designer: A computer program or other devices when the objects are generated by mechanical or electronic means.

4.10 channel: A logical device in which rt-components are positioned for final presentation. Channels are mapped by an MHEG engine to physical devices like screen windows or loudspeakers for making the rt-objects within them perceivable by the user.

- 4.11 coded representation:** Binary representation of the structure and data within an object instance.
- 4.12 component object:** An instance of an MHEG class that represents a content object, a multiplexed content object, or a composite object.
- 4.13 composite object:** An instance of an MHEG class that defines a list of composition elements grouped for presentation. The presentation of a composite object consists of the presentation of its composition elements.
- 4.14 composition elements:** An attribute of the MHEG composite class that defines its presentation and its subhierarchy.
- 4.15 constraint condition:** A part of a link condition describing an attribute and a status value. It is referred as an additional condition if trigger conditions within a same link are satisfied.
- 4.16 container object:** An instance of an MHEG class that defines a list of objects grouped for interchange. It provides a means to group objects without specifying specific relationship.
- 4.17 content object:** An instance of an MHEG class that provides a consistent structure for the interchange of encoded generic value or any presentation data. It is either a non-mux-content or a multiplexed content.
- 4.18 data:** Attribute of content and script object.
- 4.19 default channel:** A channel that is always available to the MHEG engine by the use of a reserved identifier (that is, “Default-Channel”). The elementary actions “New channel” and “Delete channel” are ignored for the default channel. When a root rt-component is not assigned to a specific channel, it is implicitly assigned to the default channel.
- 4.20 delete; delete channel, (not available):** An elementary action that removes an rt-object or a channel from an MHEG engine. The rt-object or the channel is said to be not available.
- 4.21 demultiplex:** A means to extract two or more streams of information from one unique combined encoding scheme to achieve separate presentation or exchange.
- 4.22 descendant:** Any MHEG entity contained in the subhierarchy of which the initial object is the root.
- NOTE – The graph of the descendancy may contain loops. An entity may be a descendant of itself.
- 4.23 descriptor object:** An instance of an MHEG class that defines the associated information to any object. A using application may use this object to negotiate characteristics of MHEG engines.
- 4.24 destroy, not ready, not available:** An elementary action that removes an MHEG object from an MHEG engine. The object is said to be not ready or not available.
- 4.25 elementary action:** Used to modify the behaviour of an MHEG entity within the MHEG engine. Elementary actions are interchanged within action objects. An elementary action is typically composed of a target set, specific parameters defined for each elementary action, and an optional transition duration.
- NOTE – Examples of elementary actions: Prepare (target set), Run (target set, number of performances).
- 4.26 empty socket:** A socket in which no rt-component has been plugged.
- 4.27 event:** A signal generated by some components or the MHEG engine, e.g. key stroke, mouse movement.
- 4.28 final form:** An interchange form intended for presentation without requiring change of the structure of the objects.
- 4.29 fire a link:** Each time a link condition of an active link is satisfied, the link is fired, that is, its link effect is processed.

4.30 generation: A composite object has several sockets where component objects are attached. These component objects has the sibling relationship. Therefore, a composite object is said to define one generation of component objects. Several generations may be made if another composite objects are attached in the sockets. As an rt-composite is created from a composite object, an rt-composite also defines some generations.

4.31 generic value: One of generic boolean, generic numeric, generic integer, generic ratio, generic string, generic reference, or generic list. It may be a constant or an evaluated value.

4.32 get action: Targeted to an MHEG entity and specifies the requested attribute or status value to be evaluated. The result of a get action is to provide a generic value. This generic value is used in the context of the evaluation and is a local value. If the same get action is processed in two different links in parallel, there are two generic values resulting from these two evaluations. These two generic values may be different. A get action can be used and interchanged when a value is allowed, for example, as a parameter of an elementary action or a get action, as a source value or comparison value in a link object.

NOTE – Examples of get actions: Get Preparation Status (target) and Get Running Status (target).

4.33 hook: An attribute of the content class and script class containing encoding and decoding information enabling the use of data. The semantics of these parameters are not defined by this Recommendation but are given by the semantics of the data encoding Standard or Recommendation.

4.34 hypermedia: The ability to access monomedia and multimedia information by navigating across links.

4.35 interchange attribute: An attribute of an interchanged MHEG object.

4.36 interchange medium: The medium used to interchange data: it can be a storage medium, a transmission medium, or a combination.

4.37 interchange value: The value of an interchanged attribute.

4.38 label: A generic string that is associated with an element of a composite. When an rt-composite is created from the composite object, an rt-content is created from this label and automatically plugged into a corresponding socket.

4.39 link condition: Conditions to be satisfied in order to fire a link. It is composed of some trigger conditions, some constraint conditions and some logical operators between them.

4.40 link effect: Effects of a link when it is fired. It is composed of some elementary actions and/or some action objects.

4.41 link object: An instance of an MHEG class that defines spatio-temporal or conditional relationship between MHEG entities. A link object is composed of a link condition and a link effect.

4.42 macro action: An action object that offers the facility to replace the parameters in frequently used action objects.

4.43 macro link: A link object that offers the facility to replace the parameters in frequently used conditional actions.

4.44 macro parameter: A parameter of an action or an attribute of the action class that is replaced by the following information: a macro definition identifier and a default macro usage value.

4.45 macro usage value: Within the link effect of a link object, a usage value can be provided for each macro parameter used within the link effect. This assignment is processed when the link is fired.

4.46 medium (plural media): A means by which information is perceived, expressed, stored, or transmitted.

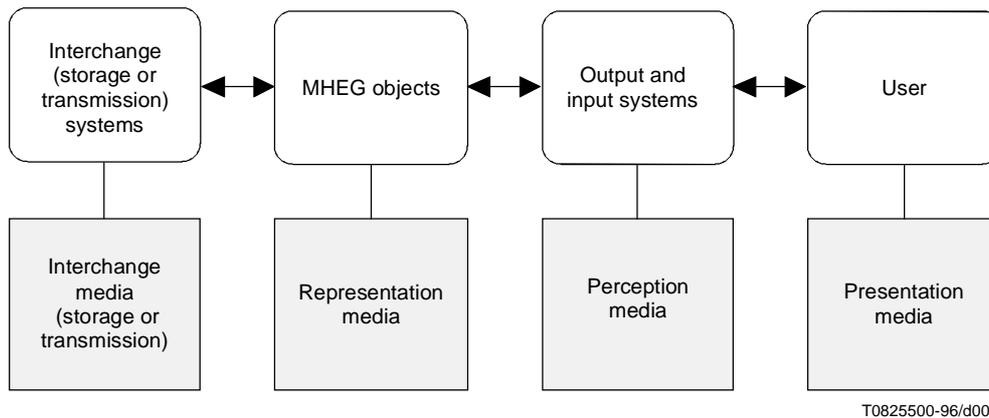


Figure 1/T.171 – Relationships between the different meanings of media

Figure 1 is intended to clarify the relationship between the different meanings of the term “media”. Representation and presentation media are to be differentiated for virtuality and portability (device independence) purposes. A given character string (coded representation) can be presented using different means (screen, paper, loudspeaker after text-to-speech synthesis). The same command can be input by the user through different equivalent means (for example, selection in a list with a mouse, a tactile screen, with digits plus validation key, with a keyboard to input the corresponding character string, or even through a microphone with a voice recognition system allowing recognition of a limited vocabulary including the name of the above-mentioned command.).

NOTE – This is a weak definition because this term has different meanings depending on the context. Thus, the term is to be avoided in its stand-alone form. To be unambiguous, it should be only used in expressions such as perception medium, representation medium, presentation medium, storage medium, transmission medium.

4.47 MH object class: An MHEG object class that is the root of MHEG class hierarchy. It defines attributes common to all other MHEG classes.

4.48 MHEG class: An MHEG object class defined by this Recommendation.

4.49 MHEG effect: The effect caused by the processing of an elementary action or a get action, which changes or retrieves some internal state.

4.50 MHEG engine: A pseudo-mechanism that may consist of a process or a set of processes that interpret MHEG objects encoded according to the encoding specifications of this Recommendation.

NOTE – The way of implementing the MHEG engine is outside the scope of this Recommendation does not define any ingredients inside the MHEG engine.

4.51 MHEG entity; entity: Any MHEG object, rt-object, or channel.

4.52 MHEG object: A coded representation of a multimedia and/or hypermedia object that conforms to this Recommendation.

4.53 minimal resources: A system with minimal buffering capacity, minimal throughput of communication channels, limited computational power.

4.54 model object: An instance of an MHEG class that represents a script object or a component object.

4.55 multimedia representation: The property of handling several types of representation media.

NOTES

1 – The term multimedia is an adjective; it shall be used attached to a noun that provides the context (e.g. multimedia service or application, multimedia terminal, multimedia network, multimedia presentation).

2 – In this Recommendation, multimedia is used in the sense of multiple representation media. Hence, a remote processing service or videotex, using a keyboard and a textual display, is not multimedia, as the interaction involves only one medium (text).

3 – An MHEG object within a multimedia service or application may contain only one representation medium.

4.56 multiplex: A means to combine two or more streams of information into one unique encoding scheme to achieve simultaneous presentation or exchange.

4.57 multiplexed content object: An instance of an MHEG class that provides a single consistent structure for the interchange of any multiplexed presentation data.

4.58 new, new channel, available: An elementary action which causes an rt-object or a channel to become available to the MHEG engine for further processing (e.g. presentation). The rt-object or the channel is said to be available.

4.59 non-mux content object: A pure content object, that is, not a multiplexed content object.

4.60 null: A specific value indicating nothing.

4.61 null data: Null data is generated by the MHEG engine when referenced by the use of a reserved identifier: “Null-Data”. The null data may replace any kind of data. Actions performed on an rt-script/rt-content made from a script object/content object with a null data are not ignored. The user effect of such action is defined by the hook and classification information provided in the model object.

4.62 null MHEG object: A null object is generated by the MHEG engine when referenced by the use of a reserved identifier: “Null-MH”. The null object may replace any object of any MHEG class. An action performed on a null MHEG object causes a null effect.

4.63 null root rt-object: A null root rt-object is generated by the MHEG engine when referenced by the use of a reserved identifier: “Null-Root-Rt”. The null root rt-object may replace any root rt-object. An action performed on a null root rt-object causes a null effect.

4.64 object: A finite, independent, self-defining piece of information that can be manipulated as a whole by MHEG engines and interchanged as one unit.

NOTES

1 – Within the context of this Recommendation, the object can be of multiple types (e.g. action, link, script, content, composite, descriptor, or container object).

2 – “Self-defining” implies that the object contains or refers to the information necessary for its handling.

3 – The definition is not restricted to digitally-coded objects, but may also be applied to analogue objects (e.g. a videodisk sequence).

4.65 object class: Any category of objects that have a specific and homogeneous template (that is, characteristics and behaviour as relevant to the contained information and to their actions modifying their behaviour).

4.66 parent-child relationship: Relationship existing when the parent object is a composite object that contains or refers to a child object.

4.67 perception medium: The nature of the information as perceived by a human being (e.g. speech, noise, music, text, drawings, moving, scenes).

4.68 plug: An elementary action that attaches an rt-component into a socket.

4.69 prepare; prepared; ready: An elementary action that causes an object to become available to the MHEG engine for further processing (e.g. presentation). The object is said to be prepared or ready.

- 4.70 presentation:** The rendition of an rt-component to be perceived by an user.
- 4.71 presentation media:** The means used to reproduce information to a user on an output device (e.g. screen, paper, printer, loudspeaker) or to acquire information from a user with an input device (e.g. keyboard, mouse, microphone, camera).
- 4.72 presentation space:** The space used to present rt-components to the user. It consists of a temporal axis, three dimensional spatial axes and an audible volume range.
- 4.73 projection:** A presentation space of a socket may be mapped into the presentation space of its parent or a channel presentation space, changing its temporal position, size or volume. The mapping is called projection. A root rt-component is always projected into a channel presentation space. A number of elementary actions are used to achieve such projection.
- 4.74 real-time interchange:** The ability to interchange objects between systems within deadlines needed for the presentation schedule to be satisfied. This implies a constant limited delay. Specifically, time-based media and time-synchronised objects have presentation requirements that may not be satisfied by the buffering.
- 4.75 real-time presentation and interaction:** The ability to present objects within a constant delay. Intrinsically time-based (e.g. audio, video) objects or objects that have time-behaviour or time-constraints (e.g. synchronisation) are concerned.
- 4.76 real-time system:** To perform operations in a way that gives the user the impression of happening immediately. A real time system means a system with only constant delay. In a wider sense a limited delay and jitter could be accepted. In the context of certain media (e.g. audio, video), real time implies a presentation that meets the original time base requirements.
- 4.77 render:** The ability to show a certain object in a specific way to the user. A number of elementary actions are used to achieve such specific rendering.
- 4.78 representation medium:** The type of the interchanged data that defines the nature of the information as described by its coded form.

NOTES

1 – Examples of representation media information and their possible coded forms:

- Characters or text: Telex, ASCII, EBCDIC.
- Graphics: CEPT, NAPLPS or CAPTAIN videotex, CGM.
- Audio: Recommendation G.711, MIDI, MPEG Audio.
- Still picture: Fax group 3, JBIG, JPEG.
- Audio visual sequence: CCIR Recommendation 601 plus associated audio: MPEG.

2 – The representation medium is defined independently of the direction of interchange (i.e. to or from the user or between equipment). Each representation medium may be used for both input or output, e.g. character-type representation may be used for both text display and text input from a keyboard; graphics-type representation may be used for both graphic display and graphic input (location) from a mouse. Audio-type or picture-type representations may be used both for reproduction and capture.

- 4.79 representation of an object:** A description of the object structure and its contents.
- 4.80 root rt-component:** A non-plugged rt-component. A root rt-component addresses either a root rt-composite or a root rt-content.
- 4.81 root rt-composite:** An rt-composite that is not plugged.
- 4.82 root rt-content:** An rt-content that is not plugged. A root rt-content addresses either a root rt-non-mux or a root rt-mux.
- 4.83 root rt-mux:** An rt-mux that is not plugged.
- 4.84 root rt-non-mux:** An rt-non-mux that is not plugged.

- 4.85 rt-component:** An rt-component addresses either a root rt-component or an rt-component socket.
- 4.86 rt-component socket:** A socket in which an rt-component is plugged. An rt-component socket addresses either an rt-composite socket or an rt-content socket.
- 4.87 rt-composite:** A run-time object created from a model composite object. The elements of an rt-composite are called sockets. An rt-composite addresses either a root rt-composite or an rt-composite socket.
- 4.88 rt-composite socket:** An rt-composite that is plugged.
- 4.89 rt-content:** A run-time object created from a model content object. An rt-content addresses either a root rt-content or an rt-content socket.
- 4.90 rt-content socket:** An rt-content that is plugged. An rt-content socket addresses either an rt-non-mux socket or an rt-mux socket.
- 4.91 rt-mux:** A run-time content created from a model multiplexed content object. An rt-mux addresses either a root rt-mux or an rt-mux socket.
- 4.92 rt-mux socket:** An rt-mux that is plugged.
- 4.93 rt-non-mux:** A run-time content created from a pure model content object, i.e. not from a multiplexed content object. An rt-non-mux addresses either a root rt-non-mux or an rt-non-mux socket.
- 4.94 rt-non-mux socket:** An rt-non-mux that is plugged.
- 4.95 rt-object, run-time object:** A run-time object created from a model object. For the purpose of reusing model objects in different contexts, a clear separation has been made between the MHEG model interchange object that contains the original definition and the rt-object corresponding to a specific usage of the model. The use of an rt-object does not affect the model object. This allows the reuse of the same model object in different rt-objects. A model object is considered a template. Any number of rt-objects may be created based on instructions given by the author, i.e. action New. This Recommendation defines an initial behaviour of each rt-object and actions that will modify and retrieve this behaviour. The internal representation of the rt-objects is not defined by the standard. Each MHEG engine may have its own internal representation technique. However, this Recommendation defines the identification techniques to reference an rt-object in order to modify or to retrieve its behaviour.

An rt-object addresses either an rt-script or an rt-component.

- 4.96 rt-script:** A run-time object created from a model script object.
- 4.97 run-time process:** An ability to process an interpreted code in an MHEG engine.
- 4.98 script object:** An instance of an MHEG class defining a structure to interchange script data in a specified encoded form.
- 4.99 scriptware; script software:** A process or set of processes that handle scripts.
- 4.100 sequencing:** The ordering of the preparation of objects, as specified by the object designer, that makes best use of the available resources to produce an acceptable presentation.
- 4.101 sibling relationship:** Relationship between sockets of a given generation within an rt-composite.

4.102 socket: The elements of an rt-composite in that rt-components are plugged are called sockets. Therefore, a socket is equivalent to an rt-component that is plugged. Once an rt-component is plugged, it can be referenced only via its socket identification. Different types of sockets are defined depending on the rt-component plugged into the socket: empty socket, rt-content socket (an rt-non-mux socket or an rt-mux socket) and rt-composite socket.

4.103 status: A value representing the current state of an object within the MHEG engine.

NOTE – The processes composing the MHEG engine are outside the scope of this Recommendation. However, this Recommendation requires a mechanism for evaluating the different statuses of objects within the MHEG engine.

4.104 storage media: The means used to store information (e.g. electronic memory, floppy disk, hard disk, optical disk, magnetic tape).

4.105 structure of an object; MHEG object structure: A description of how the information in an object is organised.

4.106 synchronisation: The presentation of rt-components in time and space according to constraints and relationship defined between those objects. The constraints and relationship may be defined explicitly, within link objects, composite objects, script objects or, implicitly, through the nature of the application.

4.107 temporal position: A specified position on the temporal axis of an rt-component or a channel.

4.108 milestone: An identified marker for a temporal position. This marker may be used to trigger links.

4.109 transition duration: An optional parameter that may be provided for certain elementary actions, e.g. Set Current Volume (target set, audible volume, transition duration). A transition duration is expressed in GTU. When a transition duration is specified, the corresponding elementary action is to be processed during the specified duration, e.g. the audible volume value changes gradually from its previous value to the specified value within the specified transition duration.

4.110 transmission media: The means used to transmit information (e.g. twisted pair, coaxial cables, optical fibres, radio links).

4.111 trigger condition: A part of a link condition describing a change of an attribute or status value. If the described change happened, the trigger condition is said to be satisfied.

4.112 user effect: The effect caused by the processing of an elementary action that is perceived by a user.

4.113 using application: An application that makes use of MHEG objects, including ITU-T Recommendations or ISO Standards.

5 Symbols and abbreviations

The following symbols and abbreviations are used in this Recommendation:

AP	Attachment Point
ASN.1	Notation and encoding rules as provided by Recommendations X.680 and X.681.
AVR	Audible Volume Range
BER	Basic ASN.1 Encoding Rules as provided by ITU-T Rec. X.690 ISO/IEC 8825-1.
CC	Current Condition within a link condition
CGSU	Channel original Generic Spatial Unit
CPS	Channel Presentation Space
CRPS	Channel Relative Presentation Space
CTP	Current Temporal Position
CV	Current audible Volume

EA	Elementary Action
GF	Generic Factor
GSF	Generic Spatial Factor
GSU	Generic Spatial Unit
GTF	Generic Temporal Factor
GTU	Generic Temporal Unit
GU	Generic Unit
GUI	Graphical User Interface
GVF	Generic audible Volume Factor
JBIG	Bi-Level Image Encoding Standard provided by Recommendation T.82: Information technology – Digital Compression and Coding of Bi-level Images.
JPEG	Photographic Image Encoding Standard provided by Recommendations T.81, T.83, T.84: Information technology – Digital Compression and Coding of Continuous-tone Still Images.
LC	Link Condition within a link object
LE	Link Effect within a link object
MH	Multimedia Hypermedia
MHEG	Adjective meaning of multimedia and hypermedia information coding scheme provided by ISO/IEC JTC1/SC29/WG12 – Multimedia Hypermedia Experts Group. NOTE – If MHEG is used without any combination of nouns, it means the working group in JTC1. Otherwise, it should be used as an adjective followed by a noun to be unambiguous in its meaning, e.g. MHEG objects, MHEG actions, MHEG classes.
MPEG	Audiovisual encoding standards provided by Recommendation H.262: Information technology – Coding of Moving Pictures and Associated Audio for Digital Storage Media up to about 1.5 Mbit/s and Recommendation H.222: Information technology – Generic Coding of Moving Pictures And Associated Audio.
mux	Multiplexed
OAP	Original visible size Attachment Point
OD	Original Duration
OGSU	Original Generic Spatial Unit
OGTU	Original Generic Temporal Unit
OGU	Original Generic Unit
OGVU	Original Generic audible Volume Unit
OPS	Original Presentation Space
OS	Original Size
OV	Original audible Volume
OVD	Original Visible Duration
OVS	Original Visible Size
PAP	Projected original visible size Attachment Point
PC	Previous Condition within a link condition
PCV	Projected Current audible Volume
POD	Projected Original Duration
POS	Projected Original Size
PRPS	Parent Relative Presentation Space

PS	Presentation Space
PTU	Physical Temporal Unit
PU	Physical Unit
PVD	Projected Visible Duration
PVS	Projected Visible Size
PVU	Physical audible Volume Unit
RGSU	Relative Generic Spatial Unit
RGTU	Relative Generic Temporal Unit
RGU	Relative Generic Unit
RGVU	Relative Generic audible Volume Unit
RPS	Relative Presentation Space
rt	Run-Time
TC	Trigger Condition within a link condition
TD	Transition Duration
VD	Visible Duration

SECTION 1 – OVERVIEW

6 T.171 principal features

6.1 Interchanging multimedia objects

A multimedia object has little use on its own. It becomes useful in a multimedia application. A multimedia application needs some method by which multimedia objects can be described and represented in an application-independent way for interchanging them with other applications. In this context, this Recommendation provides the means of multimedia object interchange in the form of MHEG objects.

6.2 The object-oriented approach

This Recommendation uses some object-oriented techniques like object classes and inheritance. This subclause describes the specific usage of object-oriented concepts within this Recommendation.

6.2.1 Object classes

This Recommendation uses the term of “object class” solely for denoting structures of interchangeable multimedia/hypermedia objects. As such, MHEG classes specify the structure of MHEG objects rather than their interfaces. Especially, MHEG does not specify methods as part of classes (or their interfaces, respectively).

The purpose of this approach is that MHEG objects be considered as information entities that are interpreted by an appropriate process, i.e. the MHEG engine, to realise behaviour.

6.2.2 Subclasses and inheritance

This Recommendation uses the concept of deriving new classes from existing ones. Subclasses inherit all structure from their superclass (i.e. the class they are subclass of) and add new attributes with the aim to add new behaviours or replace inherited ones with the aim to adapt existing behaviours.

6.2.3 Polymorphism

The MHEG engine can perform certain actions on MHEG objects, e.g. prepare them for further usage or set some attribute values. Polymorphism applies to these actions. Polymorphism means that if the application of an action is valid for objects of a certain class, it is also valid for objects of all their subclasses, but the effect may be different for different classes. Polymorphism is a way to represent that the parent has a generic (common) behaviour and that each of the children could have a specific behaviour.

For example, the implementation of a Run action targeted to a run-time content object representing a text is different from the implementation of that targeted to a run-time multiplexed content object representing MPEG bitstream. The generic common behaviour is the presentation (run) of the instance, and the specific part for the video is to present video.

6.2.4 Objects and object life cycles

Three forms of MHEG objects are distinguished within this Recommendation as shown in Figure 2:

- 1) The interchangeable multimedia/hypermedia-information, encoded according to Annex A.
- 2) The representation within the MHEG engine, usable for internal purposes (e.g. MHEG object stored in its native internal representation, computation of link conditions).
- 3) Copies of 2), usable for presentation by the user. These are referred to as rt-objects.

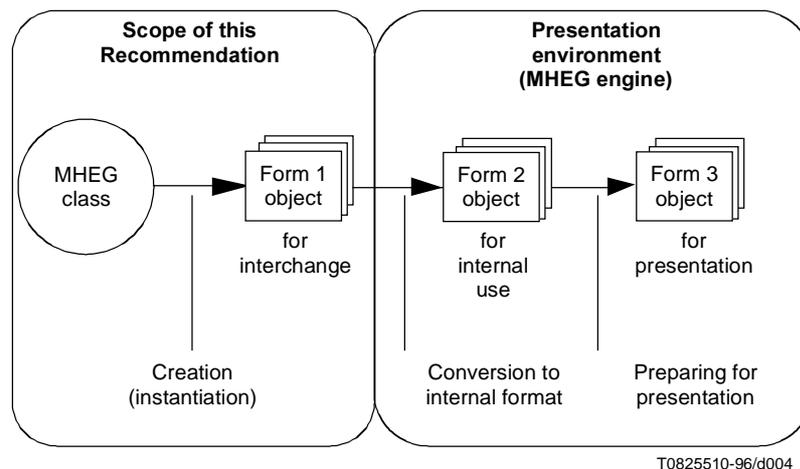


Figure 2/T.171 – The different forms of MHEG objects

Basically, the life cycle of form 1) objects is out of the scope of this Recommendation. Once they are created by some encoding mechanism as instances of concrete MHEG classes, they remain in existence, e.g. in a repository until destroyed by means not covered by this Recommendation. However, if an MHEG object of form 1), which may be a copy of some original object, is received by an MHEG engine, it is converted into form 2). The form 2) remains in existence until a Destroy action is applied to it. The form 3) comes into existence whenever a New action is applied to an appropriate form 2) object (i.e. a model object), resulting in a copy of this object but of form 3), which can be presented and may have its attribute values changed. Form 3) objects are removed from existence by use of a Delete action.

It should be noted that forms 2) and 3) objects live within a presentation environment, i.e. the MHEG engine. This implies that they are assumed to be extinct whenever the presentation environment vanishes.

6.3 Technical features

In this subclause, some technical features achieved by this Recommendation are described.

Because of the generic nature of this Recommendation, nothing prevents an user for combining the following facilities. However, an using application may impose rules to implement a more appropriate architecture.

6.3.1 Object composition for interchange and presentation

Two techniques for object composition are used and provided by this Recommendation. One is the container concept provided by the container class which is used as an information packing tool to convey multimedia and hypermedia information. The other is the presentation composition provided by the composite class which is used as an information presentation tool under the control of a multimedia and hypermedia scenario.

The container class enables the combination of a set of multimedia and hypermedia objects to be interchanged between storage systems and presentation systems. The container class groups a set of objects to be interchanged as a whole, taking into account the needs of minimal resource systems. By using this mechanism, the efficient exchange of multimedia data between heterogeneous systems can be achieved on their needs, for example. When objects in a container are to be used in a presentation, a certain number of preparation action is required before the object can participate in this presentation.

Figure 3 illustrates the use of this Recommendation as an interchange unit and as an identifier of multimedia information encoded according to other International Standards.

The composite class enables the combination of a set of multimedia and hypermedia objects to be presented. It provides a common presentation facility for those objects included in it, and describes presentations varying from the simplest monomedia object to complex multimedia and hypermedia interactions.

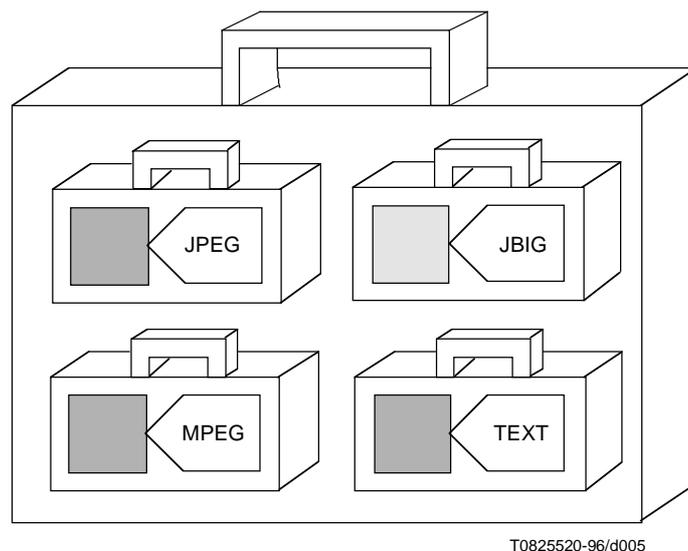


Figure 3/T.171 – Interchange and identification of multimedia information

6.3.2 Run-time objects

For the purpose of reusing objects in different contexts, a clear separation is to be made between an interchange object, called **model object**, and a run-time object, called **rt-object**. The model object corresponds to original reusable information. It is an instance of an MHEG object class containing script data, monomedia, multimedia and/or hypermedia information. An rt-object represents one specific use of the model object. It is used for presentation. Any number of rt-objects may be created from a single model object.

The internal representation of rt-objects is not defined by this Recommendation. Each MHEG engine may have its own internal representation technique. The activation and attribute modification of an rt-object do not affect the original model object. This allows the reuse of the same original model object in different contexts.

NOTE – For example, in a hockey game, two content objects are needed: B (blue team) and R (red team). Content B contains the original drawings of a blue team hockey player and content R contains the original drawings of a red team hockey player. Several rt-contents can be made from these model content objects as follows:

- Rt-contents 1 to 10, corresponding to the 10 hockey players from the blue team, are created from the content B.
- Rt-contents 11 to 20, corresponding to the 10 hockey players from the red team, are created from the content R.

All the information related to the presentation is described for each rt-content. For example, rt-contents 1 to 6 (blue team) and 11 to 16 (red team) have different positions on the rink. Rt-contents 7 to 10 have positions on the blue team bench, and rt-contents 17 to 20 have positions on the red team bench.

The presented size of each rt-content may be different. And the presentation attributes may be evaluated on each rt-content, in order to know at a precise moment if a player (i.e. an rt-content) is in a hospital (not running) or on the ice (running).

6.3.3 Individual behaviour common to all rt-objects created from a given model object

This Recommendation offers a mechanism to describe individual behaviour common to all the rt-objects, that may be created from a given model object. The advantages of this facility are:

- the behaviour is described only once and applies to all the rt-objects;
- the object designer is able to describe a common individual behaviour without knowing the number of rt-objects to be created from the given model object;
- once prepared, the link applies to all rt-objects already created as well as future made rt-objects.

This facility is provided by the “?” tail reference techniques (see 11.3).

6.3.4 Synchronisation

Essentially, multimedia object synchronisation mechanisms are categorised into four classes as follows:

- Elementary synchronisation: Two objects are synchronised either:
 - both with regard to the same reference origin time (parallel mode); or
 - one with regard to the other (sequential mode).
- Chained synchronisation: A set of objects is to be presented one after another in the form of a chain.
- Cyclic synchronisation: One or more objects is to be repetitively presented.
- Conditional synchronisation: The presentation of an object is linked to the satisfaction of a condition.

In this Recommendation, these four classes of synchronisation are provided in a single describing manner, which is the conditional synchronisation.

Multimedia object synchronisation levels are categorised into the following four levels:

- 1) Script: The script may contain complex synchronisation taking into account user replies, calculated values, and the state of system resources, for example:
 - Script synchronisation is not defined by this Recommendation.
- 2) Conditional: The current state of the presented object may trigger a reflex action to another object, e.g. “when the audio has finished, ask the question”.

- 3) Spatio-temporal: Position in time and space of one object may be relative to another, e.g. “show the product name 2 cm above the image, 2 s after the presentation of the image”.
- 4) Intermedia: Close synchronisation within a multiplexed content object, e.g. “lip sync” for movies.

Intermedia synchronisation is provided by other standards such as MPEG.

6.3.5 Links

MHEG links are associative, dynamic, and event driven. There are many types of links; however, they may be categorised into two types as follows:

- 1) Object Synchronisation Link: Describes the synchronisation among multimedia objects.
- 2) Hyperlink: Describes the object relationship in the sense of context. Any related hypermedia objects are linked by this mechanism.

This Recommendation provides both types of links by a single mechanism of object linking within link object.

The set of MHEG links is highly dynamic. Links may be added and removed at any time and may be conditional depending on user activity and previous performance in order to adapt the behaviour during the life of the presentations.

The MHEG links are driven by a change of MHEG entity behaviour, e.g. an object becomes ready, or the volume of an rt-content increases. Each link describes a change that may trigger it and the effect that it will have.

MHEG links are fully resolved and require no further processing other than their direct execution.

6.3.6 Input

In order to support interactive multimedia/hypermedia systems, it is essential to provide some mechanism user input. In this Recommendation, the list of possible interactions offered to the user is described using composite objects. Input is considered a behaviour and is based on selection and modification behaviours of rt-components.

This Recommendation provides facilities to describe the results of the user interaction but does not provide the interaction facilities. Such interaction may be achieved in a variety of ways, e.g. GUIs (Graphical User Interface) and keyboards. This Recommendation does not define the look-and-feel of multimedia interactive presentations, nor does it propose to change or add concepts to those that exist in typical GUIs. As this Recommendation is generic and independent of platform and implementation, it describes interaction at a virtual level.

6.3.7 Event handling

MHEG objects are able to interact with events through event handling mechanisms. An event is generated by either the outer system or the MHEG engine itself.

When an event is generated, the MHEG engine sets the corresponding catalogued event identification to the event attribute and, if needed, associated information to the event data attribute of the rt-object or channel that should receive this event. For example, a mouse click event on a button generated by the GUI can be sent to an rt-object as event number 10 with associated x-y position information.

Event handling mechanisms may be used for general purpose synchronisation and interaction mechanism.

6.3.8 Anchor

In order to support hypertext and hypermedia, this Recommendation provides an anchor mechanism. An anchor is a content object that holds anchor information rather than media data. This anchor information defines an area within another content object data. This area is to be presented on the other data by the GUI with the rendering specified within the objects or using a specified style. As the anchor is interchanged within a content object, rt-content can be created from the content object and all the behaviours of rt-content apply, e.g. running, selection.

6.3.9 Real time

This Recommendation has no control over and makes no assumptions about the quality or capability of the underlying networks through which using applications may interchange MHEG objects. There may already exist some LANs (Local Area Network) that could meet all the capacity requirements for real-time interchange of applications, and future standards may specify some WANs (Wide Area Network) with appropriate capabilities. This Recommendation deals with these uncertain circumstances in two ways:

- This Recommendation allows the author to define a set of specifications that would necessarily have to be satisfied to ensure real-time applications regardless of the interchange media capabilities. In addition, the MHEG object specification provides mechanisms that ease real-time interchange independent of the underlying network capabilities.
- An MHEG descriptor object provides the following:
 - a description of objects to be supported by the system;
 - a description of media encoding;
 - a system-readable material and a read-me mechanism by which communicating applications can negotiate an optimum interchange session. In this regard, the descriptor object provides a placeholder for references to other standards that are needed for allocation of network resources among complementary and competing applications;
 - offset information of objects.

Requirements for real-time capability are satisfied as follows:

- Object placement optimisation and the progressive access of objects are the responsibility of the using application. Object offset information may be used to achieve quick retrieval of a certain object encoding in another object.
- The partial object retrieval, object sequencing, and separate retrieval of object description and object content are features of the Prepare and Destroy actions usage and process.
- The resource requirements are supported by the MHEG descriptor object.
- The object interleaving is supported by the MHEG multiplexed content object. This Recommendation does not provide interleaving facilities but supports the indication of interleaved objects.
- Aggregate retrieval is supported by the MHEG container object.

6.3.10 Object management

6.3.10.1 Object identification

General object identification is needed for copyright, owner identification, and unique identification. This Recommendation provides a description mechanism for this information in an MHEG object. And among this description mechanism, the unique identification mechanism is a key for the processing of MHEG objects inside the MHEG engine.

An internal identifier and an external identifier are provided in order to support unique identification. These identifiers are used to reference objects. The internal identifier is an encoded identifier within an MHEG object and consists of an identifier or an index. The external identifier acts as a connector between MHEG object domain and entities outside of the MHEG domain and consists of a public identifier and a system identifier.

This Recommendation also provides symbolic identification using aliases. This identification may replace any other identifications.

6.3.10.2 Object reference

Reference to local and remote objects, as an alternative to physical inclusion is needed. This Recommendation provides a generic reference mechanism to any entity defined by this Recommendation.

6.3.10.3 Object content

A uniform interface to content data is needed. This Recommendation provides this mechanism in the description of content class.

6.3.10.4 Uniform objects

An uniform view of objects by using applications is needed. This Recommendation provides this uniform view by means of MHEG object hierarchy.

6.3.11 Minimal resources

Extraction of relevant component data on a timely basis for minimal resource systems is considered in this Recommendation. There are several aspects:

- 1) **Inter-object interleaving:** This is outside the scope of this Recommendation because it is a system issue. MHEG does not provide interleaving facilities but supports the indication of interleaved objects by the use of the multiplexed content object.
- 2) **Sequencing:** This is defined in this Recommendation by a specific use of the Prepare action.
- 3) **Degradation and negotiation of required resources:** The descriptor objects provide complementary information about a set of interchanged objects useful for the degradation of some presentation and the negotiation of minimal resources requirements.

6.3.12 Presentation and structuring dynamism

Once a run-time composite, called an rt-composite, is made from an original composite object, the rt-components attached to the sockets may be removed or replaced using the plub action. This allows dynamism in the presentation if the plugged rt-contents are changed and a dynamism in the structure of the rt-composite, if the plugged rt-composites are changed.

6.3.13 Macro action and link

The macro facility provides a general technique for producing efficient coding of frequently used action objects in which only a few values change from one link to another. The same applies on frequent link conditions in which only a few values change from one link to another. This allows the sharing and reuse of complex behaviours. An author may create a catalogue of predefined action object and link object templates.

6.3.14 Static and dynamic assignment of generic value

A generic value may be specified as a constant or as an evaluated result of a get action. Generic values are essential values processed inside an MHEG engine. Generic values can be stored and retrieved from content object data. An evaluated value may be stored after its evaluation, i.e. a constant is stored, e.g. content data = "hello". An evaluated value can also be stored without any processing, i.e. the get action is stored, e.g. content data = Get Preparation Status (target), in which case, the evaluation is performed dynamically each time this generic value is retrieved.

7 MHEG Engine Assumptions

7.1 Handling and Interchange of objects

This Recommendation does not define the way in which an MHEG engine handles the objects, neither the interchanged MHEG objects nor the rt-objects. But it makes some assumptions about the features of MHEG engines in order to be able to define further actions on the objects.

- 1) This Recommendation assumes that the MHEG object, once it has been interchanged, is used by an using application and handled by the MHEG engine.

- 2) This Recommendation makes no assumptions on the internal structure or organisation of the MHEG engine, with these exceptions:
- The MHEG engine is expected to correctly interpret MHEG effect of MHEG action.
 - The MHEG engine is able to evaluate status and attribute values of MHEG entities which are under its control. The changes that occur on the status or attribute value may be used to express conditions in links.
 - When an MHEG engine uses an entry of a catalogue, it is expected that this engine correctly interpret this entry as described in the catalogue.
 - The MHEG engine is able to work with a user interface support system, when it exists, and to obtain from this user interface system the results of the interaction with the user. These results are formalised by the MHEG engine in order to modify the corresponding interaction statuses. For example, the user interface support system indicate to the MHEG engine that a specific element of a given object has been selected, and thus the MHEG engine will be able to modify the corresponding interaction and selection status. This change of status may fire a link and carry on an action as specified in a given MHEG object.

In order to facilitate understanding of the role played by this Recommendation, this clause shows in more detail the role of the objects in the interchange between two systems, as shown in Figure 4.

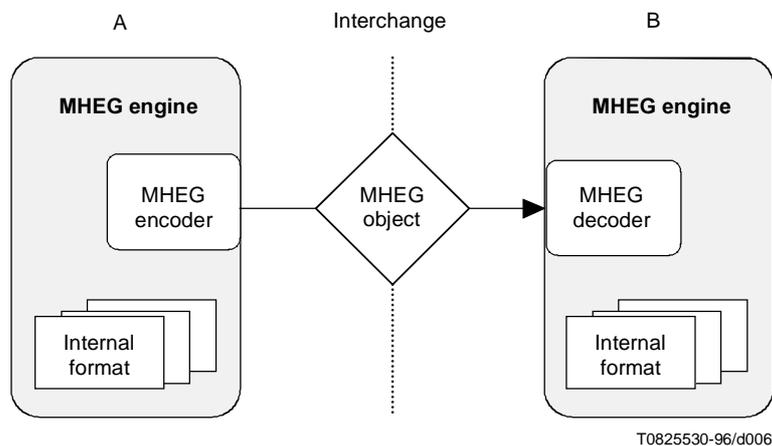


Figure 4/T.171 – Interchange of MHEG objects

The MHEG object is defined only at the interchange point between the using applications A and B. When system A wishes to send an MHEG object to B, it calls an MHEG **encoder**, which converts the internal format used by A to the format defined by this Recommendation.

When B receives an MHEG object, the object is decoded by an MHEG **decoder**. The values within the object are passed by the decoder, which may convert them to the internal format used by B.

NOTES

1 – The internal formats in the using applications A and B may be the same as the MHEG format, but nothing in this Recommendation requires this.

2 – If the exchange between A and B is in both directions, the two systems may have both encoder and decoder.

7.2 The MHEG application interface

The MHEG engine is assumed to provide an interface to the using application. This interface will provide facilities for control of the MHEG engine and is the only access point for the application to MHEG objects within the MHEG engine. This Recommendation does not define the scope or the form of the MHEG engine interface, nor does it define the structure of data passing across this interface.

7.3 Exception handling

This Recommendation describes the recommended reactions of an MHEG engine on abnormal or exceptional situations possibly encountered during execution time. These reactions are neither considered to be minimal nor complete, which means that an MHEG engine can do more (e.g. write to a log file or inform the application) or even less (e.g. stop processing). The described behaviour is recommended but not required for conformance.

NOTE – Specific reactions on error conditions are given with each specific situation throughout the section behaviour of ISO 13522-1.

8 Methodology

This clause describes the methodology used to specify this Recommendation and is provided as a help to the user in understanding this Recommendation. It does not imply that a conforming system or product uses this methodology for either analysis or implementation. This is the case for the general object orientation that is chosen for the design of this Recommendation.

In the context of this Recommendation, an object class denotes a structure of interchangeable multimedia/hypermedia information, from which MHEG objects can be instantiated. This Recommendation restricts the usage of object-oriented concepts to specialisation, inheritance, and polymorphism.

8.1 Modularity

In order to allow an using application to make use of only some of the objects provided for by this Recommendation, the representation itself is in the form of a set of modules organised as follows:

- one module per interchanged object class;
- one module containing all the useful definitions and the abstract classes;
- one module containing all the elementary actions.

NOTES

1 – In the way mentioned above, it is expected that an application process will incur the minimum overhead due to the objects not used.

2 – It is intended that any future extension to the representations provided by this Recommendation (see Figure 5) shall also be isomorphic with respect to the modularity of the base representation.

8.2 Methodology of representation of MHEG objects

This Recommendation provides a description of an MHEG object, a precise definition of the structure of the object, and a base-coded representation for the object. The base coded representation defined in this Recommendation uses ASN.1.

Alternative structure representations may be provided. They are intended to be isomorphic and equivalent for the purposes of this Recommendation (see Figure 5).

NOTE – It is intended that the base transfer syntax provided by this Recommendation and alternative syntaxes provided by other parts of this Recommendation shall be consistent and that it is possible to convert an MHEG object instance from one to another. However, this process is not defined by this Recommendation.

The representation of an MHEG object is specified through four levels:

- description of representation;
- object-oriented definition (structure and semantics);
- notation for the structure of the representation; and
- the coded representation defined by applying encoding rules to the representation (see Figure 5).

The description of the representation provided by this Recommendation includes the structure of the representation and sufficient semantics to allow a concrete understanding of the meaning of an MHEG object. This Recommendation does not provide semantics for a conforming application process making use of this object.

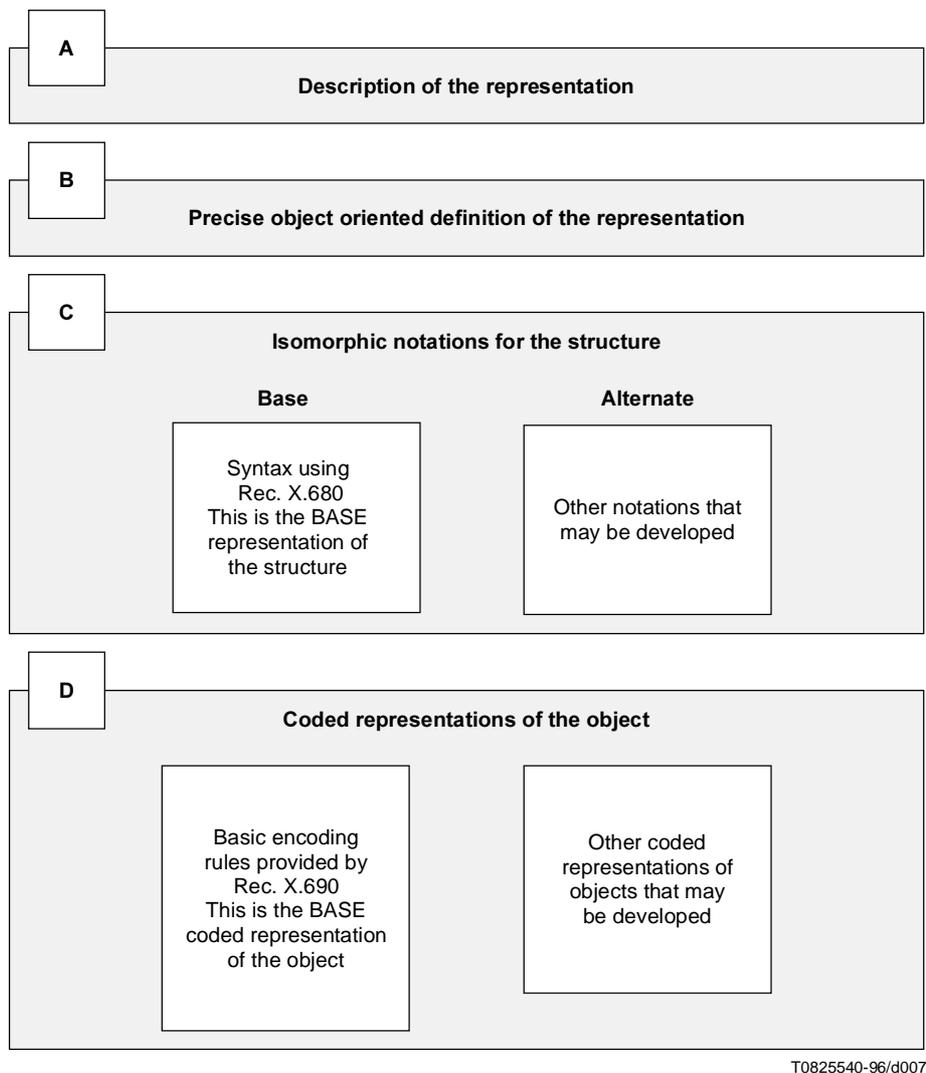


Figure 5/T.171 – Levels of the methodology used by this Recommendation

8.2.1 Level A – Description of representation

Level A provides the description of representation for each useful definition (see Section 2), each MHEG object (see Section 3), and each MHEG entity behaviour (see Sections 4, 5, 6, 7 and 8).

8.2.2 Level B – Precise object-oriented definition

For each MHEG object class, the precise object-oriented definition is given by the following:

- Class hierarchy: From which class it inherits and by which class(es) it is inherited.
- Usage: Which class(es) it uses and by which class(es) it is used.
- Structure: Attributes it contains, in addition to the inherited attributes.
- Textual description: A short explanation of semantics and behaviour.

The structure of classes is described using a context-free grammar in which the conventions described apply:

- **X ::= I, J, K** X is defined as a sequence of attributes I followed by J followed by K.
- **X?** None or one instance of X may occur.
- **X+** One or more instances of X may occur.
- **X*** Any number of instances of X may occur.
- **(I | J | K)** Exactly one of the set is to occur.
- **(I, J, K)?** The entire sequence may or may not occur.
- **(I, J, K)+** One or more instances of the entire sequence may occur.
- **(I, J, K)*** None or more instances of the entire sequence may occur.

The structure and semantics of each representation attribute of the object (B in Figure 5) are described in latter subclauses with the structure shown in Figure 6.

Attribute name

- *semantics*
- *type OR*
- *structure of subattributes OR*
- *choice between attributes OR*
- *list of a subattribute*

Figure 6/T.171 – Structure of attribute subclause

NOTE

- Each subattribute is described as an attribute.
- Sequence of (A, B): means that subattribute A is followed by subattribute B.
- List of (A, B): means that one or more instances of the entire sequence (A, B) may occur.

8.2.3 Level C – Isomorphic notations for the structure of MHEG objects

The standard provides a set of equivalent notations for the level B description of the structure of an MHEG object. The notations are equivalent in that they are isomorphic with respect to the MHEG object structure described in level B. However, one of the notations is the base notation, which is to be referred to in case of questions of conformity of a product or application process.

The base notation for the representation of MHEG objects structure is provided by ASN.1. This is known as the base representation (C in Figure 5).

NOTE – It is intended that any future additions to this Recommendation of notations for the representation of MHEG objects shall be isomorphic with MHEG object structure described in level B.

8.2.3.1 ASN.1 techniques used in level C

The ASN.1 notation for the structure of each MHEG class is typically as presented in Figure 7.

```
-- MHEG OBJECT CLASS MODULE
© International Organisation for Standardisation, 1995. Permission to copy in any form
is granted for use with conforming MHEG engines and applications as defined in
Recommendation T.171, provided this notice is included in all copies.
ISOMHEG-cl {joint-iso-itu-t(2) mheg (19), version (1), class-identifier (xx)},
DEFINITIONS:=BEGIN
EXPORTS MHEG-Object-Class;
IMPORTS MH-Object-Class FROM ISOMHEG-MH {joint-iso-itu-t(2) mheg (19), version (1),
MH-object (1000)};
MHEG-Object-Class:= SEQUENCE
{
class-identification    OBJECT IDENTIFIER    {joint-iso-itu-t(2),          -- root
                                             mheg (19),                -- arc1
                                             version (1),              -- arc2
                                             class-identifier(xx)     -- arc3
                                             },
COMPONENTS OF MH-object,
-- object class attributes
}
END
```

Figure 7/T.171 – Example of an ASN.1 module used for MHEG object representation

The following ASN.1 techniques are used to represent the MHEG objects:

- The technique of ASN.1 OBJECT IDENTIFIER is used. The object identifiers defined in this Recommendation are summarised in Annex A. The semantics of the object identifier is defined by reference to the following object identifier tree.
- **root**: Origin of the standard. It shall be joint-iso-itu-t(2).
- **arc1**: T.171 identification. It shall be 19. The identification of this Recommendation has been provided by the joint naming authority.
- **arc2**: MHEG Standard version. It is set to 1 for this version of this Recommendation.
- **arc3**: Class identifier. It is a number assigned to each MHEG class.
- The ASN.1 modules are used in this Recommendation to represent the MHEG object classes and superclasses, and also to describe the useful definitions. Each module has an identification, which is expressed with an ASN.1 OBJECT IDENTIFIER. This identification is attached to the definition of the module and is not part of the definition of the MHEG class.

This module OBJECT IDENTIFIER is not encoded within an MHEG object. It is used by another module within the IMPORT/EXPORT facilities in order to identify a module formally.

- The ASN.1 IMPORT/EXPORT and the COMPONENTS OF facilities are used to represent the inheritance of attributes of MHEG object classes. Each superclass defines a SEQUENCE of attributes that are exported. This SEQUENCE is imported by the subclasses and COMPONENTS OF is used in order to inherit these attributes.
- The ASN.1 IMPORT/EXPORT facilities are also used to represent the MHEG useful definition and the elementary actions. The useful definition module exports the definitions and the abstract classes. The elementary action module exports the elementary actions. Each MHEG object class module imports the corresponding definitions when needed.

- Each MHEG object is represented as a sequence of attributes using the ASN.1 SEQUENCE facilities. Each attribute has an arbitrary structure.
- The technique of an ASN.1 OBJECT IDENTIFIER is also used to uniquely identify each MHEG object class. This MHEG object class OBJECT IDENTIFIER is part of the definition of the class and will be encoded in each object instance.

8.2.3.2 Copyright protection

The formal ASN.1 definition is part of the text of this Recommendation and is protected by copyright. In order to facilitate conformance to MHEG, the formal ASN.1 definition in the body and Annex A . This Recommendation may be copied as specified in the following copyright notice:

- © International Organisation for Standardisation, 1995. Permission to copy in any form is granted for use with conforming MHEG engines and applications as defined in Recommendation T.171, provided this notice is included in all copies.

The permission to copy does not apply to any other material in this Recommendation.

NOTE – An attribute is defined by this Recommendation for specifying the copyright notice attached to an MHEG object.

8.2.4 Level D – Coded representation of MHEG objects

This Recommendation provides a set of equivalent coded representations of MHEG objects. The coded representations are equivalent in that they are isomorphic with respect to the level B. However, one of the coded representations is the base-coded representation.

The base coded representation is provided by ASN.1 BER.

NOTE – It is intended that any future additions to this Recommendation of coded representations of MHEG objects shall be of notations isomorphic with the level B.

SECTION 2 – GENERIC UTILITY AND USEFUL DEFINITION MECHANISMS

This Recommendation provides the following mechanisms:

- Presentation mechanism (see clause 9);
- Generic identification mechanism (see clause 10);
- Generic reference mechanism (see clause 11);
- Generic value (see clause 12);
- Macro mechanism (see clause 13);
- Hooks (see clause 14);
- Extensibility (see clause 15).

These generic mechanisms or definitions are used throughout the technical clauses of this Recommendation.

Their coded representation is grouped in a separate ASN.1 module called “useful definitions module”.

When an MHEG object requires a specific definition, its ASN.1 class representation imports the definition from the useful definition module.

9 Presentation Mechanism

This clause describes the general mechanism provided for a Presentation Space (PS) (see 9.1) used to present the rt-components to the user. It is composed of a generic coordinate system in time and space, and it also contains an Audible Volume Range (AVR) for audio expression.

The values used to define the spatial, temporal and audible behaviour of rt-components are expressed in Generic Units (GU), e.g. position, size, duration and volume. This mechanism allows the interpretation of the units with respect to an associated PS.

NOTE – This allows the cohabitation of data coming from different creation tools with different coordinate systems and audible volume ranges.

A PS is associated with each rt-component. It is called Original PS (OPS) (see 9.2). A PS is also associated with each channel. It is called Channel PS (CPS) (see 9.3).

In order to allow placement of children relative to their parent, each rt-component may be projected into a parent PS. For a child, its parent PS is called a Relative PS (RPS) (see 9.4). This parent can also be a child of another parent. So, there is a chain of mapping that ends at CPS: OPS to RPS to RPS...to CPS. At each projection a scaling factor, called Generic Factor (GF), is applied. This GF is defined as a ratio and specifies number of RGU Relative GU (RGU) to be mapped to one Original GU (OGU). Lastly, in order to be perceived by the user, each CPS is mapped by the MHEG engine onto a physical space (see 11.1).

Figure 8 summarises the presentation techniques.

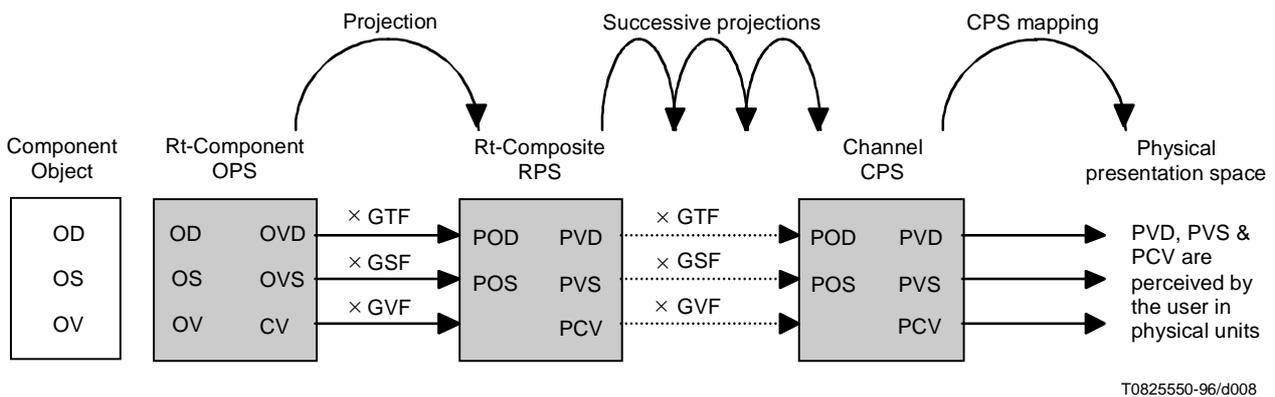


Figure 8/T.171 – Overview of presentation techniques

9.1 Presentation Space (PS)

Each PS is defined as follows:

- a temporal axis T;
- three spatial axes, X, Y and Z;
- an AVR.

9.1.1 Temporal axis

Each PS has one temporal axis: T.

The temporal axis is an interval $[0, T_{EndPoint}]$, where the $T_{EndPoint}$ shall always be greater than or equal to 0. The $T_{EndPoint}$ can be specified as infinite. The point 0 is defined as the origin of the temporal axis.

The number of addressable points within a temporal axis is $(T_{EndPoint} + 1)$.

The GU used along this axis is called Generic Temporal Unit (GTU).

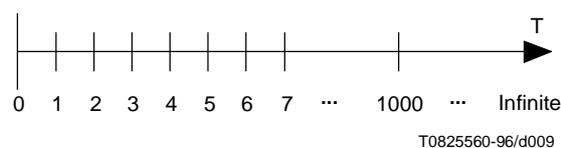


Figure 9/T.171 – The T axis

The rt-component temporal behaviour uses temporal positions and temporal durations, which are expressed in GTU within the PS:

- A temporal position:
 - a point on the T axis;
 - always defined within the interval [0, TEndPoint] on the corresponding T axis.
- A duration:
 - a distance between two temporal positions;
 - defined as the length of an interval [StartPoint, EndPoint];
 - always defined within the interval [0, TEndPoint] on the corresponding T axis;
 - equal to (EndPoint – StartPoint). EndPoint shall be greater than or equal to StartPoint;
 - the number of addressable points within a duration is (EndPoint – StartPoint + 1).

Figure 10 illustrates temporal position and duration attributes.

The temporal scaling factor is called Generic Temporal Factor (GTF). Its value n specifies that one Original GTU (OGTU) is to be mapped in n Relative GTU (RGTU).

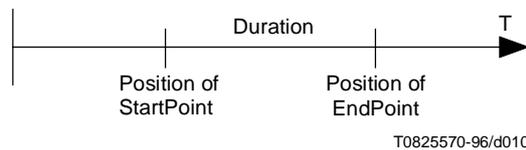


Figure 10/T.171 – Temporal position and duration

9.1.2 Spatial axes

Each PS has three spatial axes — X, Y and Z — that shall be spanning a right-handed orthogonal coordinate system.

Each spatial axis is to be perceived by the user in the direction defined by Figure 11: X from left to right, Y from bottom to top, and Z from display towards the user.

NOTE – Extensions may be provided to allow views from alternative directions, e.g. through additional channel attributes.

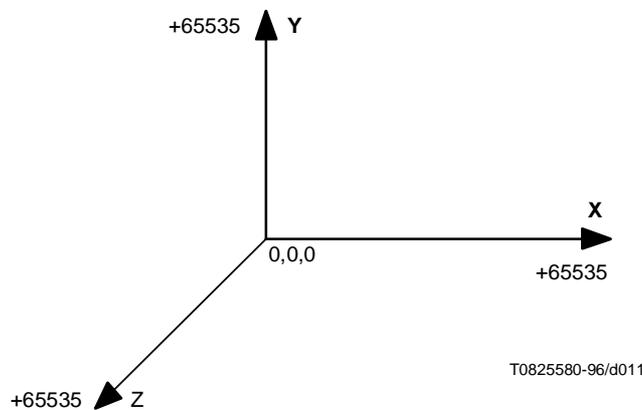


Figure 11/T.171 – The X, Y and Z axes of the presentation space

All instances of such PS shall be colinear relative to each other, meaning that all X axes are parallel, all Y axes are parallel and all Z axes are parallel.

Each spatial axis is of finite length and is an interval [0, SEndPoint], where SEndPoint shall always be greater than or equal to 0. A default value is also provided for SEndPoint and is equal to +65535. The point 0 is defined as the origin of each spatial axis. The point (0, 0, 0) is called the spatial origin.

The number of addressable points within a spatial axis is (SEndPoint + 1).

The GU used along each spatial axis is called Generic Spatial Unit (GSU).

The rt-component spatial behaviour uses spatial positions and sizes, which are expressed in GSU within the PS:

- A spatial position is:
 - a triple of points (x, y, z), one point on each spatial axis;
 - always defined within the interval [0, SEndPoint] of the corresponding spatial axis.
- A size is:
 - a triple of distances (lx, ly, lz), one on each spatial axis, giving the length along each axis in the given order;
 - each distance is defined as the length of an interval [StartPoint, EndPoint] on the corresponding axis;
 - each distance shall be defined within the interval [0, SEndPoint] of the corresponding spatial axis;
 - each distance is equal to (EndPoint – StartPoint). EndPoint shall be greater than or equal to StartPoint;
 - the number of addressable points within each distance is (EndPoint – StartPoint + 1).

Figure 12 illustrates spatial position and size attributes. In this figure, the Z axis is omitted for clarity.

A spatial scaling factor is defined for each spatial axis, which is called Generic Spatial Factor (GSF). Its value n specifies that one Original GSU (OGSU) is to be mapped to n Relative GSU (RGSU).

The GSF may be different for each spatial axis of a PS.

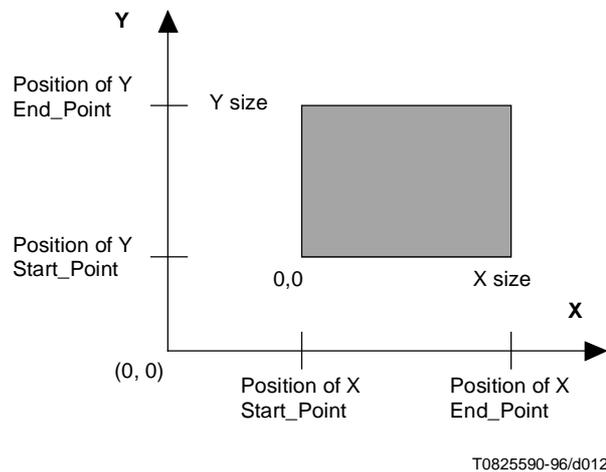


Figure 12/T.171 – Spatial position and size

9.1.3 Audible Volume Range (AVR)

This Recommendation defines one unique AVR. Every audible expression is described within the AVR. The AVR is defined as the interval [0, +255].

The number of addressable points within the AVR is 256.

The unit used along the axis is Generic audible Volume Unit (GVU).

The rt-component audible behaviour uses audible volumes, which are expressed in GVU within the PS. An audible volume is always a point defined within the AVR value provided by this Recommendation.

The audible volume scaling factor is called Generic audible Volume Factor (GVF). Its value n specifies that one Original GVU (OGVU) is to be mapped in n Relative GVU (RGVU).

9.2 Original Presentation Space (OPS)

9.2.1 Initialisation of OS and OD

When an rt-component becomes available, a PS is associated with it, and it is called OPS. The temporal and spatial axes shall be initialised as follows:

- a) The TEndPoint shall be initialised with the Original Duration (OD) value retrieved as follows:
 - 1) if the OD is encoded within the component object, this value shall be used;
 - 2) for an rt-content, if the OD is not encoded within the content object, the OD value may be retrieved from information contained in the hook or in the data itself;
 - 3) otherwise, the OD value shall be set to infinite.
- b) Each SEndPoint shall be initialised with the Original Size (OS) length value on the corresponding axis, which is retrieved as follows:
 - 1) If the OS is encoded within the component object:
 - i) if the length of the OS is provided on an axis, this value shall be used for this spatial axis;
 - ii) otherwise, the length of the OS on this axis shall be set to 0.
 - 2) For an rt-content, if the OS is not encoded within the content object, the OS value may be retrieved from information contained in the hook or in the data itself.

NOTE – For a multiplexed content object, the OS of the global multiplexed data is used. The OS may be encoded individually in each stream within a multiplexed content object. This information should not be used to set the OS of the rt-mux. It is just a help for an MHEG Engine to manipulate and present a stream.

- 3) Otherwise, the OS length value shall be set to 0 on each axis.

The following OGU along each OPS axis are defined:

- the GTU along the OPS temporal axis is called Original GTU (OGTU);
- the GSU along each OPS spatial axis is called Original GSU (OGSU);
- the GVU along the OPS-AVR is called Original GVU (OGVU).

9.2.2 Initialisation of GF

The scaling factors associated with the OPS axes shall be initialised as follows:

- The GTF shall be initialised to 1. It may be changed by the use of the Set GTF action.

NOTE 1 – For example, if the GTF changes from 1 to 10, 1 unit of RGTU lasts 10 times longer than before the change. If the GTF changes from 1 to 1/10, 1 unit of RGTU lasts 10 times shorter than before the change. A GTF value of 0 has the effect of reducing the rt-component duration to an infinitesimal small duration in the RPS, because the whole rt-component duration is perceived within 0 RGTU.

- The GSF on each OPS spatial axis shall be initialised to 1. It may be changed by the use of the Set GSF action.

NOTE 2 – For example, if the GSF changes from 1 to 2 on an axis, 1 OGSU is to be mapped on 2 RGSU on this axis. So the rt-component is perceived twice as large on this axis. A GSF value of 0 on all axes has the effect of reducing the rt-component perception to an infinitesimal small size in the RPS, because the whole rt-component is being perceived within 0 RGSU.

- The GVF shall be initialised to 1. It can be changed by the use of the Set GVF action.

NOTE 3 – For example, if the GVF changes from 1 to 2, 1 OGVU is to be mapped on 2 RGVU. So the rt-component is perceived twice as loud. A GSF value of 0 has the effect reducing the rt-component perception to an infinitesimal small volume in the RPS, because the rt-component volume is being perceived in 0 RGVU.

9.2.3 Initialisation of attributes for rt-components

For each rt-component, the following attributes are defined when the OPS is created:

- OD: Defines the initial duration of the rt-component. Its value is retrieved as explained above.
- Original Visible Duration (OVD): It is a subset of the OD. It defines the part of the OD which is perceived by the user. It shall be initialised to the OD value and may be changed by the use of the Set OVD action.
- OS: Defines the initial size of the rt-component. Its value is retrieved as explained above.
- Original Visible Size (OVS): It is defined as a size and then positioned within the OS. The portion of the OS which fits within the OVS is perceived by the user. The portion of the OS that is outside the OVS is clipped. If the OVS exceeds the limits of the OS, the exceeding portion of the OVS is filled by the background of the rt-component. The OVS shall be initialised to the OS value and positioned at the origin of the OS. The size may be changed by the use of a Set OVS action. The position may be changed by the use of the Set OVS Position action.

NOTE – The background of the rt-component is not defined by ISO/IEC 13522-1. The background may be either retrieved from the hook or the data of the component object. A Set Style action can also be used by the author to set a catalogued background style.

9.2.4 Initialisation of attributes for rt-contents

For each rt-content the following attributes shall be defined when the OPS is created:

- a) OV: Defines the initial volume of the rt-content. Its value is retrieved as follows:
 - 1) if the OV is encoded within the content object, this value is used;
 - 2) if not, the OV value may be retrieved from information contained in the hook or in the data itself;
 - 3) otherwise, the OV value is set to the minimum value of the AVR.
- b) Current Volume (CV): It corresponds to the volume which will be perceived by the user. It is initialised to the OV value and may be changed by the use of the Set CV action.

NOTE – When an rt-mux is responsible for the presentation of several streams, it is up to the demultiplexing process and the GUI to keep the coherence between the duration, size and volume of the rt-mux itself and the duration, size and volume of each stream selected for this rt-mux.

To ensure the final presentation, the duration and the volume that are encoded within each rt-content shall be expressed in 1 ms (PTU) and 1 dB (PVU), respectively. If the duration and the volume are retrieved from the hook, these values are translated and expressed in 1 ms (PTU) and 1 dB (PVU), respectively.

9.3 Channel Presentation Space (CPS)

The CPS shall be initialised as follows:

- 1) The TEndPoint shall be initialised with the CPS Duration value retrieved as follows:
 - a) if the action Set CPS is targeted to this channel and contains a CPS Duration parameter, this value shall be used;
 - b) otherwise, the CPS Duration value is set to infinite.
- 2) Each SEndPoint shall be initialised with the CPS Size length value on the corresponding axis, which is retrieved as follows:
 - a) If the action Set CPS is targeted to this channel and contains a CPS Size parameter, the following applies:
 - i) if the length of the CPS Size is provided on an axis, this value is used for this spatial axis;
 - ii) otherwise, the length of the CPS Size on this axis is set to 0.

- b) Otherwise, the CPS Size length value on each axis is set to the default SEndPoint value provided by this Recommendation.

When different MHEG objects are generated by different authors and take place in the same application, if each of these authors is using the default channel, it is recommended that all the authors are aware of the current CPS definition of the default channel, if it has been changed.

- 3) The AVR is set to the AVR value provided by this Recommendation.

There is no scaling factor associated with the CPS axes.

9.4 Relative Presentation Space (RPS)

The Relative PS of an rt-component is the PS in which the rt-component is assigned and projected. Each rt-component is assigned to only one RPS at a given time. The following RPS are defined:

- 1) Parent RPS (PRPS): A socket may be projected within the OPS of its rt-composite parent, in order to describe a composition in time, space or audio. This can be repeated recursively, if the rt-composite parent is itself an rt-composite socket.
- 2) Channel RPS (CRPS):
 - a) a socket may also be projected directly within a CPS when this socket does not participate to the composition in time, space or audio described by its parent;
 - b) each root rt-composite is projected into a CPS.

Initially, each root rt-component is assigned to the default channel, and each socket is assigned to its parent rt-composite PS. This RPS assignment may be changed by the use of the Set RPS Assignment action.

The following RGU along each RPS axis are defined:

- the GTU along the RPS temporal axis is called Relative GTU (RGTU);
- the GSU along each RPS spatial axis is called Relative GSU (RGSU);
- the GVU along the RPS AVR is called Relative GVU (RGVU).

The attributes defined for each rt-component within the OPS are projected into the assigned RPS as follows:

- Projected OD (POD): Projection of the OD in the RPS and is calculated as follows: $OD \times GTF = POD$.
- Projected Visible Duration (PVD): Projection of the VD is calculated as follows: $OVD \times GTF = PVD$.
- Projected OS (POS): Projection of the OS in the RPS and is calculated as follows: $OS \times GSF = POS$.
- Projected Visible Size (PVS): Projection of the OVS and is calculated as follows: $OVS \times GSF = PVS$. The PVS value can also be set directly by the use of the Set OVS action with the OVS Proj Strategy attribute as “calculated”.

The attribute defined for each rt-content within the OPS is projected into the assigned RPS as follows:

- Projected Current Volume (PCV): Projection of the CV in the RPS and is calculated as follows: $CV \times GVF = PCV$.

9.5 CPS mapping

Each CPS is a virtual space. It is for the MHEG engine to assign a physical PS to a virtual space, and to convert virtual coordinates (GU) to physical one.

Each CPS is assigned to a physical PS by the MHEG engine. The descriptor object may provide information to facilitate this mapping such as:

- An expected media type mapping may be provided to facilitate the assignment to a physical PS. The MHEG engine associates the CPS with physical devices such as TV sets, workstations, cameras, windows, loudspeakers and microphones.

- A recommended resolution for the physical PS axes may be provided for satisfactory rendition: x-resolution, y-resolution, z-resolution and t-resolution.
Authors should take into account the limited possibilities of minimum resource equipment, and not require unduly fine resolutions.
- A recommended audio dynamic range and a recommended minimum and maximum frequency required for playback may be provided to present audible data with an appropriate amplitude.

The virtual coordinates shall be converted to physical coordinates as follows:

- The CPS temporal axis is mapped to a physical temporal axis. The physical temporal units are expressed in milliseconds. 1 CGTU is mapped to 1 PTU.
- The CPS spatial axes are mapped to physical spatial axes. The physical spatial units depend on each physical PS, e.g. a pixel, 1 millimetre. 1 CGSU is mapped to 1 PSU.

In performing the mapping of each spatial axis onto the physical space, the MHEG engine is responsible that the direction of each spatial axis (see Figure 11) will be correctly perceived by the user. Therefore, considering a CPS-Y axis defined as [0, 65535] in the upward direction, it might be required to map this Y axis onto a physical range [0, 479] in the downward direction for a computer display.

- The CPS-AVR is mapped on a physical AVR. The physical audible volume units are expressed in decibels [dB] with respect to the physical AVR's limit. 1 CGVU is mapped to 1 PVU.

10 Generic identification mechanism

MHEG provides generic mechanisms to identify any MHEG entity, data or stream (see Figure 13).

There are three types of identification mechanisms:

- **External identification:** This identification is not encoded within an MHEG object. The external identifications can be decoded for reference purposes without having to decode the MHEG object (see 10.1).
- **Internal identification:** This identification is encoded within the MHEG object. The internal identification cannot be discovered until an MHEG object is decoded (see 10.2).
- **Symbolic identification:** This identification may replace any other external or internal identification (see 10.3).

NOTE – Do not confuse external and internal identification with remote and local objects. A local or remote object may have an external or an internal identification.

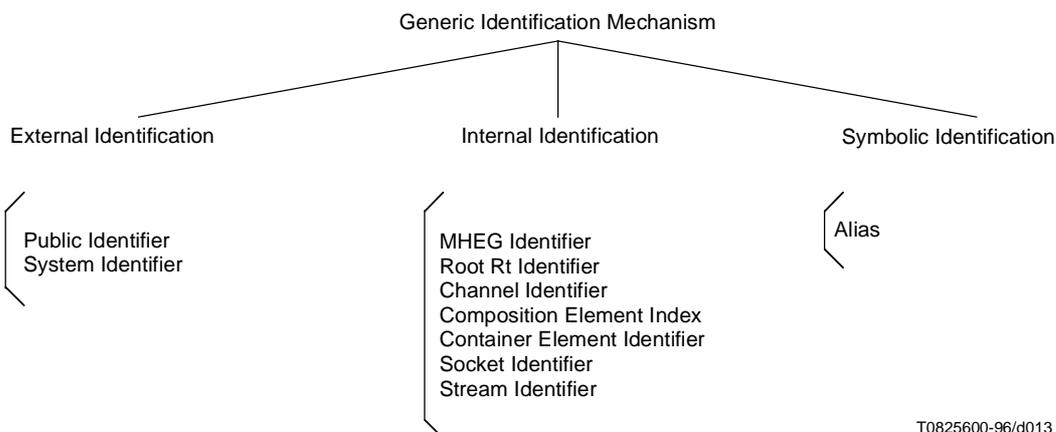


Figure 13/T.171 – Overview of generic identification mechanism

10.1 External identification

This identification mechanism is not defined by this Recommendation and is not encoded within an MHEG object.

NOTE – Distinguish clearly between identification and references. The external identification is not encoded in the MHEG object, but the reference to the MHEG object is encoded within an MHEG object, even if it uses an external reference.

This identification mechanism may be used to identify any MHEG object or data not included in a content or a script object.

This technique is defined in ISO 8879 “Formal Public Identifiers”, and ISO/IEC 9070 “Registration procedures for public text owner identifiers” that provides both ASN.1 and SGML notation for the representation of registered owner identifiers. The external identifier consists of:

- a public identifier (see 10.1.1);
- a system identifier (see 10.1.2).

10.1.1 Public identifier

A public identifier is a character string whose syntax is given by 10.2/ISO 8879. In this Recommendation, a public identifier is encoded as an ASN.1 PrintableString.

NOTE – The encoding method for public identifiers of this Recommendation simplifies that of ISO/IEC 9070 defined in informative Annex D in order to ease implementation.

The example of a public identifier in Figure 14 identifies the image of Lena, the lady with the hat, as it is stored in the system. The example of a public identifier in Figure 15 identifies a remote object requiring telecommunication access. A complete example of a public identifier is given in Figure 16.

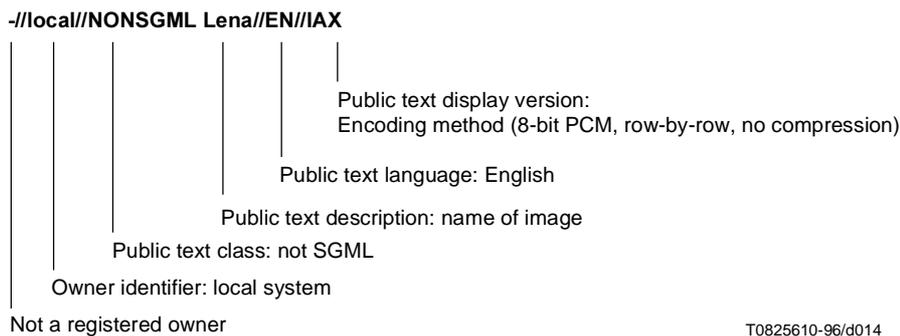


Figure 14/T.171 – Example of a public identifier for a non-registered owner

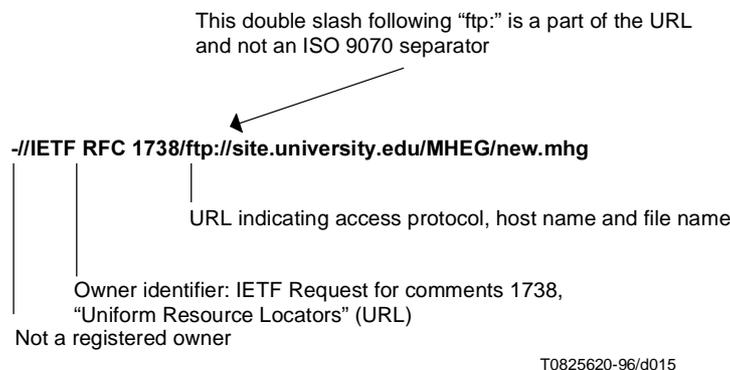


Figure 15/T.171 – Example of a public identifier for a URL (Uniform Resource Locator)

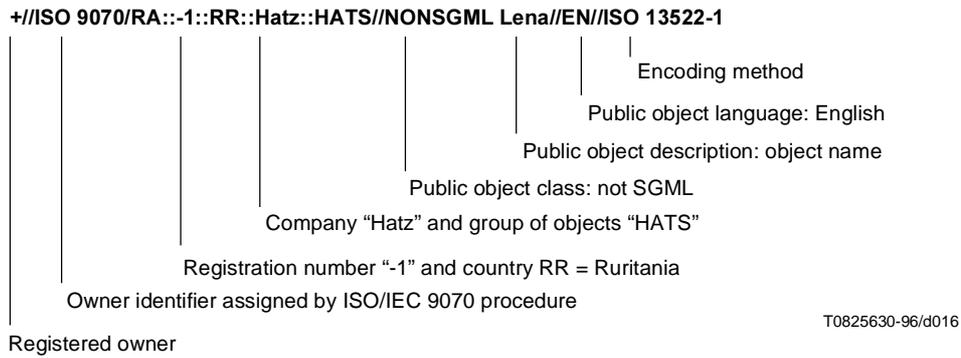


Figure 16/T.171 – Example of a public identifier for a registered owner

In Figure 16, imagine a Ruritanian hat manufacturer called Hatz who applies to the ISO/IEC 9070 Registration Authority and is issued the registration number –1. The company follows the recommended conventions and decides to begin the owner-name components with: “/RR:Hatz”.

The Hatz company decides to set up a group of public MHEG objects containing description of the hats. The additional owner-name component is HATS, thus making the full owner name: ISO/IEC 9070/RA::-1::RR::Hatz::HATS.

10.1.2 System identifier

A system identifier provides a system-dependent means of identifying a particular piece of information within the system. It may also be used to indicate the system that supplies the information. The character set and the syntax of the system identifier is not defined by this Recommendation and depends on the system. This Recommendation encodes the system identifier as an ASN.1 OctetString.

NOTE – The system identifier may be used to describe a database or telecommunications request for an object.

Figure 17 shows that identification mechanisms may be combined. This link object has an internal identification encoded internally using an MHEG identifier, and an external identification encoded externally to the object using an external identifier.

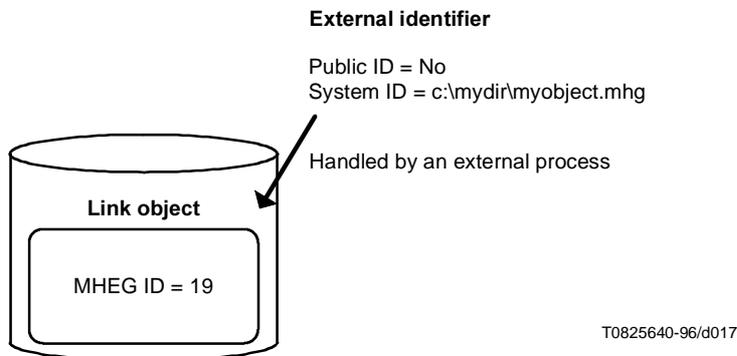


Figure 17/T.171 – Example of a system identifier without public identifier

10.2 Internal identification

This identification mechanism is encoded within MHEG objects.

The following internal identifications are provided in ISO/IEC 13522-1 to identify an outer entity:

- **MHEG Identifier:** Identify any MHEG object (see 17.2);
- **Root Rt Identifier:** Identify any root rt-object (see 10.2.2);
- **Channel Identifier:** Identify any channel (see 10.2.3).

The internal identification is also used to identify an element within a constructed entity. For that purpose, the following is defined:

- An **index:** To provide an identification for an element of a given generation within a constructed entity. Each element is indexed within a generation by sequential integers starting from one. The following notation is used to express index within this Recommendation: `outer_object_identification.i`, where “i” indicates the index within the outer object, e.g. the third element of container object with MHEG ID 1000 is expressed by 1000.3.
- A **tail:** To provide an identification of an element within a hierarchy of constructed entities. A tail is a list of indexes and it designates a path from an outer entity through an element. Each index corresponds to an index of an element within the constructed entity. Successive indexes correspond to nested elements. The last index in the list is the index of the element to be identified. The following notation is used to express tail: `outer_object_identification.i1.i2.i3....in`, where “i1.i2.i3....in” is a list of indexes indicating the path from the outer object through the element.

The following internal identifications are also provided in ISO/IEC 13522-1 to identify an element:

- 1) **Composition Element Index:** Identify any element of a composite (see 10.2.4);
- 2) **Container Element Identification:** Identify any element of a container (see 10.2.5);
- 3) **Socket ID:** Identify any socket (see 10.2.6);
- 4) **Stream ID:** Identify any stream within a multiplexed content object (see 10.2.7).

10.2.1 MHEG identifier

An MHEG identifier may be assigned to each MHEG object. When used, the MHEG Identifier shall be encoded within the MHEG object and shall be composed of the following:

- An optional **application identifier**, which is provided by the application designer. It shall be encoded in this Recommendation as an ASN.1 octet string.
- An **MH number**, which uniquely identifies an MHEG object within an application. The mh number shall be encoded as an integer. When the MHEG identifier is present, it is mandatory.

A reserved MHEG identifier is provided to identify a null MHEG object: “Null-Mh”.

It is the object designer’s responsibility to ensure that MHEG identifiers are unique among the MHEG objects.

The following notation is used to express MHEG identifiers: a1.a2.a3...an-m, where “a1.a2.a3...an” represents the application identifier and m represents the mh number. When the application identifier is omitted, the MHEG identifiers are expressed using the following notation m.

Example: A content object may have the following MHEG identifier: 129.63.24.2-49, and another composite object may have the following MHEG identifier: 700, which means that this composite has no application identifier.

10.2.2 Root Rt-ID

A root rt identifier is assigned to each root rt-object. This identifier is mandatory. This identifier is composed of the following:

- The **model object identification**, which identifies the model object. Any one of the identifications provided by this Recommendation to identify an MHEG object may be used to identify this model object, e.g. MHEG identifier, Container Element Index, External ID, Alias.
- An **rt number**, which is an integer provided by the author.

A reserved root rt identifier is provided to identify a null rt-object: “Null-Root-Rt”.

It is the object designer’s responsibility to ensure that root rt identifiers are unique.

The uniqueness of rt numbers is required at a given instant. An rt number may be reused once this number is again available, i.e. once the rt-object that was using it is deleted. It is not possible to have, at a precise instant, two rt-objects with the same rt number.

The following notation is used to express root rt identifiers: model_object_identification:r, where model_object_identification can be any one of the identification provided by this Recommendation to identify an MHEG object and “r” is the rt number, e.g. the first rt-composite object created from the composite object identified by MHEG identifier: 700 has the following root rt identifier: 700:1. The third rt-content object created from the content object identified by MHEG identifier: 129.63.24.2-49 has the following rt identifier 129.63.24.2-49:3.

10.2.3 Channel identifier

A channel identifier is assigned to each channel. This identifier is mandatory. This identifier is an integer.

A reserved channel identifier is provided to identify the default channel: “Default-Channel”.

It is the object designer’s responsibility to ensure that channel identifiers are unique.

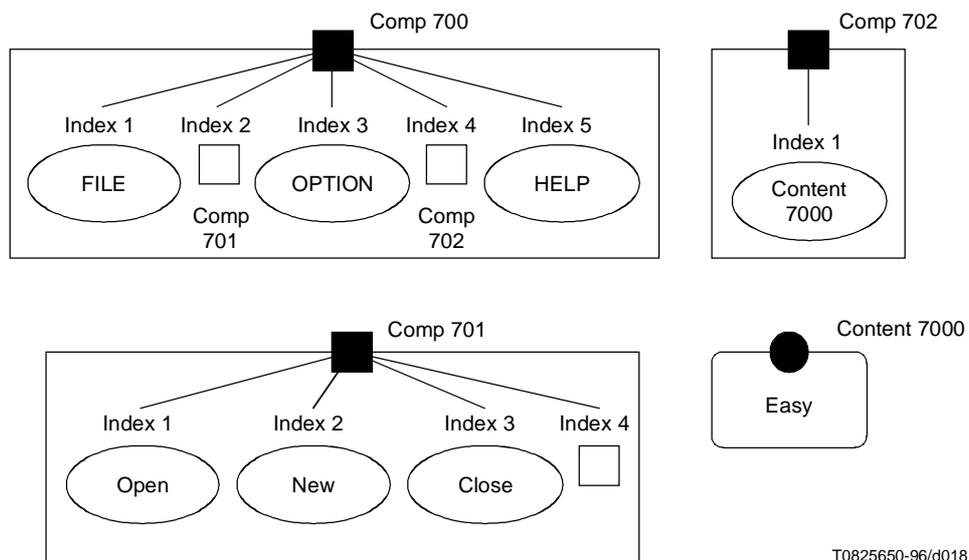
10.2.4 Composition element index

The index facility is used in a composite object to identify each element within the composition.

Each composite object defines one generation. The indexes of elements with non-empty associated models are provided by the author. The missing indexes are automatically provided by the MHEG engine; such elements are considered as elements with an empty associated model.

In Figure 18, the five elements in composite object 700 are identified with indexes 1, 2, 3, 4 and 5. The four elements in composite object 701 are identified with indexes 1, 2, 3 and 4. The single element in composite object 702 is identified by index 1. Comp 701 and Comp 702 that are associated models of Comp 700, and Content 7000 that is an associated model of Comp 702, are referenced rather than included. They are identified by using either their own identifier or the index of the composite elements in which they are associated. Figure 18 does not show the composition behaviour in composite objects.

NOTE – The element index is typically used during the construction of the rt-composite.



T0825650-96/d018

Figure 18/T.171 – Indexing of elements in a composite object

10.2.5 Container element identification

The index facility is used in a container object to identify each element within the container. When a container object contains a hierarchy of embedded container objects, the tail facility is used to identify the embedded elements in the hierarchy of container objects.

Each container object defines one generation. The indexes of elements are automatically provided by the MHEG engine and are indexed in each generation from one, sequentially.

An element in a container object is always an MHEG object. It is possible that a container element, which is an MHEG object, is already assigned another identifier, e.g. an external identifier or an MHEG identifier. In this case, this element has two different identifications, either as an element of a container or as an MHEG object directly.

NOTE – The index or tail within a container is typically used to obtain rapid access to elements of a container. When a Prepare action is targeted to an MHEG object using its identification within a container, e.g. 1000.3, element 3 of container 1000, the object may be picked out of the container with minimal processing by the MHEG engine. If an MHEG identifier is encoded within this object, this identifier is available to the MHEG engine as soon as the object is prepared. It is recommended but not required, that further actions on this object typically use the MHEG identifier rather than the container indexing.

In Figure 19, the elements defined in the container object are MHEG objects; they are identified by an index or a tail within the container, e.g. index 1 of container 800 or tails 1000.8.1 and 800.1 identifies the link object with MHEG-ID 810. An MHEG identifier and an external identifier may be also provided for each object.

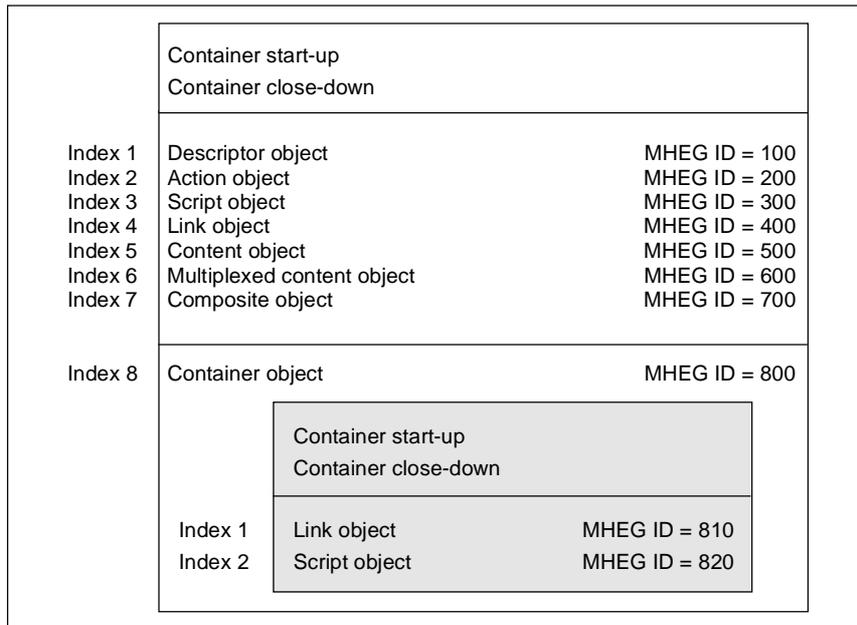
10.2.6 Socket identifier

The index facility is used to identify each element of a given generation of an rt-composite. The tail facility is used in a root rt-composite object to identify each element within the rt-composite.

Each root rt-composite defines a complete hierarchy. In each generation of an rt-composite, the sockets are indexed from one, sequentially and deduced from the composite object model. The tail is used to identify each socket indicating the path from the root rt-composite through the socket.

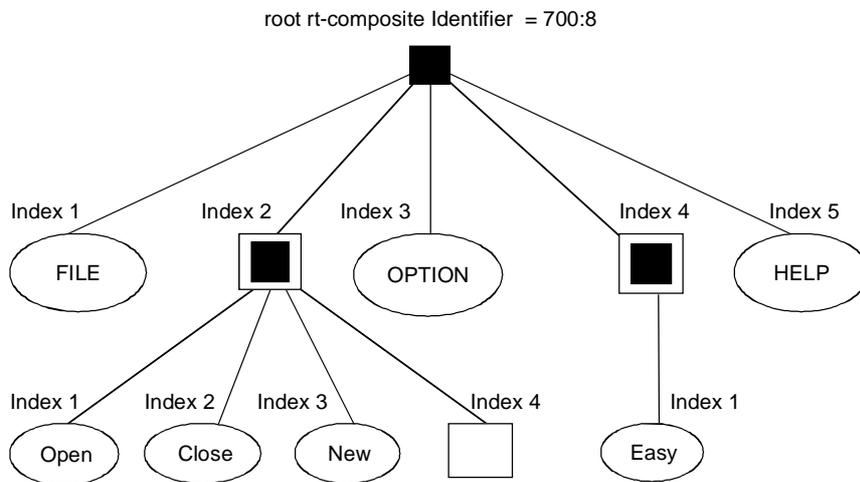
Figure 20 shows an rt-composite that is a result of a New action applied on an rt-composite made from the model composite object 700 shown in Figure 18. The rt number is 8. This rt-composite is made either directly from the composite object 700 or from the same object considered as element 7 in container 1000.

Container Object: MHEG ID = 1000



T0825660-96/d019

Figure 19/T.171 – Indexing of elements in a container object



T0825670-96/d020

Figure 20/T.171 – Indexing of sockets

The sockets in the first generation are identified by indexes 1, 2, 3, 4 and 5 like the elements in the composite 700 or as tails 700:8.1, 700:8.2, 700:8.3, 700:8.4 and 700:8.5.

NOTE – These objects can also be identified by tails: 1000.7:8.1, 1000.7:8.2, 1000.7:8.3, 1000.7:8.4 and 1000.7:8.5, because 700:8 can be identified by 1000.7:8. However, see the Note in Container element identification 10.2.5.

The three sockets in the second generation below index 2 of first generation are identified with indexes 1, 2, 3 like the elements in the composite 701 or as tails: 700:8.2.1, 700:8.2.2 and 700:8.2.3.

The socket in the second generation below index 4 of first generation is identified with index 1 as is the element in the composite 702 or as tail: 700:8.4.1.

10.2.7 Stream identifier

A stream identifier is used to identify each stream within a multiplexed content object. This identifier is expressed as a tail and designs a path from an outer stream to an inner stream within the multiplexed data.

It is the object designer's responsibility to ensure that stream identifiers are unique within the same multiplexed data.

10.3 Symbolic identification

This Recommendation provides a means of assigning a symbolic identification that may be used to replace any other internal or external identification. This symbolic identification is called alias.

Alias provides the following characteristics:

- Addressing entities by another name.
- Regrouping entities on which the same actions are to be applied at the same time.
- Multievaluation of objects using a delayed assignment of alias. A complete application may be described by an author using alias. The real assignment of the entity may be done later dynamically.

This is particularly useful for multilingual/internationalisation, multiresolution media format, multiversion of objects and multiencoded media format. For example, an author provides one object for the different languages, and the actions are described by using aliases. Then, Set Alias actions are used to assign the correct language when needed.

An alias is a string that is used as a convenient alternative for an other identification format. An alias is assigned by an author using the Set Alias action.

NOTES

1 – This alias is different from the object name in the MH-object class.

2 – For example, Set Alias(START, "e:\appli\start-up.mhg"). This alias is an alternative identification for the external ID identified as "e:\appli\start-up.mhg".

3 – For example, Set Alias(OPTION, 700:8.2). This alias is an alternative identification for the socket 2 of root rt-composite number 8 created from composite model 700 (see Figure 20).

A given entity may have several aliases assigned at the same time. A given alias may be assigned to several entities at the same time.

11 Generic reference mechanism

Many MHEG entities, data and streams need to be referenced either from another MHEG object or as a parameter of an action. Targets of action also address the MHEG entities using their reference.

11.1 Generic reference using generic identification

The following references are defined:

- 1) **External ID Reference:** Reference to an MHEG object or a data using its External ID (see 10.1).
- 2) **MHEG ID Reference:** Reference to an MHEG object using its MHEG-ID (see 17.2).

The reference to an MHEG identifier cannot be resolved until the referenced MHEG object is decoded. A using application may provide a mechanism to resolve this reference without decoding the object, e.g. using a table of correspondence between a location of an MHEG object and its MHEG identifier.

- 3) **Root Rt-ID Reference:** Used to reference a root rt-object. It is specified as a reference to the model object followed by a reference to an rt-number. The model object is referenced using one of the references defined for an MHEG object. The reference to an rt number is one of the following:
 - a) An rt number: Used to reference a root rt-object.
 - b) The constant *: Used to reference, at the moment when this reference is solved, all the root rt-objects created from the specified model object.
 - c) The constant ?: Used to reference a single dynamically determined rt number (see 11.3).
For example, taking into account the composite object defined in Figures 18 and 19:
 - i) 700:* represents a reference to all the root rt-composite objects created from this composite object model at the moment of the resolution of this reference;
 - ii) 700:? represents a reference to a dynamically determined root rt-composite object created from this composite object model.
- 4) **Channel ID Reference:** Reference to a channel using its Channel ID (see 10.2.3).
- 5) **Tail Reference:** This reference cannot be used directly by the author; it is part of the Container Element Reference and of the Socket ID Reference. It is defined as one of the following:
 - a) Tail: Used to reference a given element using its tail identification (see 11.1).
 - b) Tail complement: Defined as either **children** or **descendants**. When used alone, it references the child elements, i.e. one generation or all generations (the descendant elements of the outer entity itself).
 - c) Tail followed by a tail complement: The tail complement, in that case, applies to the element referenced using the tail identification. This element shall be itself a constructed entity, i.e. another container object or an rt-composite socket.
- 6) **Container Element Reference:** Reference to an MHEG object using its Container Element. It is specified as a reference to the container object followed by a **Tail Reference**. The container object is referenced using one of the references defined for an MHEG object. The tail reference allows an author to reference:
 - a) a single MHEG object, or an element of the container;
 - b) the children of the outer container object;
 - c) the children of a container object which is an element of the outer container object;
 - d) the descendants of the outer container object;
 - e) the descendants of a container object that is an element of the outer container object.
- 7) NOTE 1 – In Figure 19, 1000.8.children or 800.children represents a reference to the two elements within this embedded container, i.e. link object 810 and multiplexed content object 820.
- 8) NOTE 2 – 1000.descendants represents a reference to all the elements within container 1000 and within the embedded container 800.
- 9) NOTE 3 – The reference to a container element cannot be resolved until the referenced container object is decoded. However, if the author has provided an offset information within the descriptor (see 0.5.3.9) for a container element, the MHEG engine may use this offset information to decode only the specified container element without decoding the rest of the container object.
- 10) NOTE 4 – If an author has provided additional identification to a container element, e.g. an MHEG identifier, this container element may be also referenced using one of the references defined for an MHEG object.
- 11) **Socket ID Reference:** Used to reference a socket. It is specified as a reference to the root rt-composite followed by a **Socket Tail Reference**. The root rt-composite is referenced using a Root Rt ID Reference. The socket tail reference allows an author to reference:
 - a) a single socket;
 - b) the children of the root rt-composite;

- c) the children of an rt-composite socket within the root rt-composite;
 - d) the descendants of the root rt-composite;
 - e) the descendants of an rt-composite socket within the root rt-composite.
- 12) **Socket Tail Reference:** This reference cannot be used directly by the author; it is part of the Socket ID Reference. It is defined as one of the following:
- a) A tail: Used to reference a given socket using its tail identification (see above).
 - b) A socket tail complement: It is defined as one of the following:
 - i) The constants provided by a tail complement, i.e. children or descendants (see above). Used alone, the tail complement applies to the root rt-composite itself.
 - ii) The constant **?child** used to reference a single dynamically determined socket within the root rt-composite itself.
 - iii) The constant **?descendant** used to reference a single dynamically determined socket within the descendant sockets of the root rt-composite itself. When ?child or ?descendant is used, the technique to determine the socket is the same as for ? (see 11.3). The individualisation phase, in that case, is made on each child or descendant socket of the specified root rt-composite.

The ?child and ?descendant facilities may be used as follows: for example, a root rt-composite describes a menu; each socket is an item. When one of the items is selected, the same behaviour is specified. This behaviour is processed individually each time one of the items is selected. This is described in a single link using the ?descendant or ?child facilities. The number of sockets children or descendants may be changed during the life of the menu; the link always applies.
 - c) A tail followed by a socket tail complement: The tail complement, in that case, applies to the socket referenced using the tail identification. This socket shall be an rt-composite socket.
- 13) NOTE 5 – In Figure 20, 700:8.children represents a reference to all the child elements of the root rt-composite (indexes 1 to 5, first generation).
- 14) NOTE 6 – 700:8.2.children represents a reference to all the child sockets of the rt-composite socket 700:8.2 (indexes 1 to 4, 2nd generation).
- 15) NOTE 7 – 700:8.descendants represents a reference to all the descendant sockets of the root rt-composite (all generations).
- 16) NOTE 8 – 700:?.1 represents a reference to rt-content socket (FILE) of a dynamically determined root rt-composite created from composite object model 700.
- 17) NOTE 9 – 700:?.descendants represents a reference to all descendant sockets of a dynamically determined root rt-composite created from composite object model 700.
- 18) NOTE 10 – 700:*.1 represents a reference to the rt-content socket 1 (FILE) of all the root rt-composite objects created from this composite object model at the moment of the resolution of this reference.
- 19) NOTE 11 – 700:8.?descendant represents a reference to a single dynamically determined socket between the descendant socket (all generations) of the root rt-composite.
- 20) NOTE 12 – 700:8.2.?child represents a reference to a single dynamically determined socket between the child socket (open, close, new, index) of the rt-composite socket 700:8.2.
- 21) NOTE 13 – 700:?.?child represents a reference to a single dynamically determined socket between the child socket (one generation) of a dynamically determined root rt-composite created from composite object model 700.
- 22) **Stream ID Reference:** Reference to a stream using its Stream ID (see 10.2.7).
- 23) **Alias Reference:** Reference to an MHEG entity or a data or a stream using one of its assigned aliases (see 10.3). The reference to an alias cannot be resolved until the alias has been assigned.

11.2 Predefined references

The following predefined reference constants value are also defined:

- 1) **Null-Data**: Reference to a null data.
- 2) **Null-Mh**: Reference to a null MHEG object.
- 3) **Null-Root-Rt**: Reference to a null root rt-object.
- 4) **Default-Channel**: Reference to the default channel.
- 5) **“This”**: Local reference to a composite or a container within itself. A composite or a container object may be referenced using any one of the referencing techniques defined for an MHEG object (e.g. MHEG-ID reference, container element reference, alias reference, external identifier reference) or by using the “this” facility provided for local referencing purposes within the composite or the container object itself. “This” shall be used only within link and action object; it shall not be used either within a container element reference or within a component object reference of a composition element. The following applies:
 - a) When the composite or the container object is prepared, “this” takes the value of the composite or the container object identification. When the start-up link, rt start-up and container start-up are activated, the value of “this” is propagated to them.
 - b) The propagation of the value of “this” to a link object has the effect of replacing each occurrence of “this” within the link condition by the value of “this”.
 - c) When such a link fires, the value of “this” shall be propagated to its link effect. The following applies:
 - i) Each occurrence of “this” within the usage value of the macro resolution and within the action object is replaced by the value of “this”. If the action object is a nested action object, any occurrence of “this” within all the embedded action objects is also replaced by the value of “this”.
 - ii) If the link is a nested link object, the value of “this” is also propagated to the embedded link objects.
- 6) NOTE 1 – “This” has the role of a local variable within the scope of the composite or the container itself.
- 7) NOTE 2 – In Figure 18, the availability start-up of the composite object has the following link effect: New this:1, Activate link L1. The link object L1 is defined as follows: TC = this:1.3 becomes selected, LE = Run (this:1.5). When the composite 700 is prepared, the start-up is activated, the root rt-composite number 1 is created and the value this = 700 is propagated to link object L1.

11.3 ? reference

? is used in link and action objects to describe an individual behaviour common to all root rt-objects created from a given model object. The advantages of this facility are:

- the behaviour is described only once and applied to all the root rt-objects created from a same model object;
- the object designer is able to describe an individual behaviour for each root rt-object without knowing the number of root rt-objects that will be created from the given model object;
- once activated, the link applies to all root rt-objects already created as well as future created root rt-objects;
- this facility allows a complete independence between all the root rt-objects created from the same model, i.e. they may be presented independently, the behaviour is an individual behaviour attached to each root rt-object.

The reference “model_object_identification:?” can be used in a link or in an action object wherever an rt-object reference is allowed. However, the following applies:

- 1) If the link object is not an embedded link, the reference “model_object_identification:?” shall be used in the source value of at least one trigger condition in order to assign a usage value to ?.

The composite rt-availability start-up link answers this requirement, its link condition is “this:? becomes available”. All embedded links activated from the rt-availability start-up are active for each new created root rt-composite.

- 2) In an embedded link object, the ? can be used for different model objects, e.g. the following references “model_object_identification1:?”; “model_object_identification2:?” can be used in the same embedded link object.

It is deprecated to use ? with different model object identifications in a non-embedded link object. Because the ? value is resolved when a trigger condition becomes true and only one trigger condition becomes true at a given time. So, in that case only one ? is resolved and all other references are considered as “undefined” references.

The determination of the rt number to be taken into account in the place of each ? is highly dynamic. When a non-embedded link object is activated, the following phases applies:

- 1) Each link object containing a trigger condition with a source value encoded as “model_object_identification:?” is to be evaluated in parallel for each root rt-object created from the indicated model_object_identification. For that purpose, all instances of ? with the same model_object_identification within the link condition are replaced by a given instance of a root rt-object rt number in each parallel evaluation.

During the activation of such a link object, the root rt-objects created from the indicated model_object_identification may vary, i.e. additional root rt-objects may be created or some of them may be destroyed.

- 2) If a trigger condition becomes true for a given root rt-object and if the link condition is satisfied, its rt number becomes the “? usage value”. Each instance of ? with the same model_object_identification is replaced by the “? usage value” in the following places:
 - a) Within the link effect – If the action object defining the link effect is a nested action object, the “? usage value” is also propagated to each instance of the ? with the same model_object_identification within the embedded action objects.
 - b) When an embedded link object is activated from this link object, the following applies:
 - i) a specific instance of the embedded link object handles this “? usage value”;
 - ii) the “? usage value” is propagated to each instance of the ? with the same model_object_identification within this instance of embedded link;
 - iii) if this instance of the embedded link object contains some ? not resolved within a trigger condition, the same phases, i.e. 1) and 2) apply;
 - iv) if some ? remains unresolved after all phases 1) and 2), the corresponding references are to be considered as “undefined”.
- 3) Phase 1 is processed in parallel on all root rt-objects created from the model object. If the link condition of a nested link object becomes true for two root rt-objects in parallel in phase 1, e.g. 700:3 and 700:4, the phase 2 is to be processed for each ? usage value in parallel. That is, if the link effect activates an embedded link, two instances of this embedded link are handled in parallel for both ? usage values of 700:3 and 700:4.

If the link is deactivated, this link cannot be fired for any “? usage value”.

This example takes into account composite object defined in Figure 18:

- Its start-up may have the following link effect: new 700:3, new 700:9.
- Its rt-availability start-up link effect is: activate link object L1.
- Link L1 is an embedded link object, it is defined as follows: TC = 700:?.3 becomes selected, LE = run 700:?.5. This link is used to describe an individual behaviour common to all root rt-composites which may be created from model object 700, i.e. for 700:3, 700:9 but also for any future root rt-composite created from the model 700.

The following applies:

- 1) When the rt-availability start-up is activated, i.e. each time a new root rt-composite is created, an instance of L1 handles this new root rt-composite. The rt number of this new made root rt-composite is considered as the “? usage value” and is transmitted to this instance of L1.

- 2) In fact, there is one instance of link L1 per root rt-composite already created from 700. So we can consider that link L1 is active for all root rt-composite created from 700 and that its trigger is evaluated each time the selection status of the socket 3 of any root rt-composite created from 700 changes.
- 3) When the trigger of an instance of link L1 becomes true for a root rt-composite, e.g. 700:2, the link condition becomes true and the value of this root rt-composite rt number is considered as the “? usage value”. This value is transmitted to the link effect of this instance of link L1. So the fifth socket of 700:2 becomes running.
- 4) When both 700:2.3 and 700:3.3 are selected in parallel, two instances of the link L1 are satisfied in parallel, their link effect is to be performed in parallel. That is, socket 5 of 700:2 and socket 5 of 700:3 become running in parallel.

Following is another example:

- Link object L1 is defined as follows: TC = 700:?3 becomes selected, LE = run (700:?5), activate (link object L2, link object L3).
- Link L2 is an embedded link object defined as follows: LC: 900:3.1 becomes selected AND 700:?2 is running, LE: stop 700:?5 run 700:?2.
- Link L3 is a more complex embedded link object, it is defined as follows: LC: 999:?1 becomes selected AND (700:?5) is running, LE: stop (999:?1, 700:?5) run (999:?2, 700:?2).

The following applies:

- 1) Instances of Link L1 are handled as explained in the previous example.
- 2) Consider now that an instance of L1 link condition is satisfied for 700:7, 7 becomes the “? usage value” for composite 700. This value is transmitted to the link effect of this instance of L1. That is, the socket 5 of 700:7 becomes running as in the previous example, but additionally links L2 and L3 are activated.
- 3) When L2 is activated, an instance of L2 handles the “? usage value” and all appearances of 700:? are replaced by 700:7 in the link effect of this instance. When this instance of link L2 is handled and when its link condition becomes true, i.e. 900:3.1 is selected and 700:7.2 is running, then 700:7.5 is stopped and 700:7.2 becomes running.
- 4) When L3 is activated, and instance of L3 handles the “? usage value” and all appearances of 700:? are replaced by 700:7 in the link effect of this instance. As one ? is remaining in a trigger condition, the instance of link L3 is evaluated in parallel for all root rt-composites created from 999. If the trigger condition of this instance of L3 is valid for 999.9, the “? usage value” 9 is replaced in this instance and so 999:9.1 and 700:7.5 are stopped and 999:7.2 and 700:7.2 becomes running.

12 Generic Value

This clause describes the mechanisms provided by this Recommendation for specifying values in a generic way. The generic values are typically used to express:

- elementary action parameters;
- get action parameters;
- conditions;
- MHEG object attributes.

A generic value may be defined as follows:

- 1) **constant**: A specific constant is also provided to identify an unspecified value: “unspecified”.
The “unspecified” value is used in link condition as a comparison value.
- 2) **evaluated values**: An evaluated value is the result of the process of a get action by the MHEG engine. When the MHEG engine is not able to process a get action, this get action is evaluated as the constant value “undefined”.

The generic values may be stored in and retrieved from the data field of a content object.

The type of a generic value is one of the following:

- Generic boolean (see 12.1);
- Generic numeric (see 12.2);
- Generic integer (see 12.3);
- Generic ratio (see 12.4);
- Generic string (see 12.5);
- Generic reference (see 12.6);
- Generic list (see 12.7).

It is the object designer's responsibility to ensure that the generic values given as parameters of actions conform in type and number to the specification of the parameters as specified in this Recommendation. If the actual parameters do not conform in type or number to what is expected, it is an error and the action shall be ignored.

The type of a generic value determines the following:

- the set of values that it can possibly take;
- common semantics of these values.

12.1 Generic boolean

A generic boolean value has one of the following values: TRUE, FALSE.

The generic boolean value may be specified as follows:

- a boolean constant;
- a result of the evaluation of a get action;
- a result of a logical operation within a link condition;
- a result of a comparison operation within a link condition.

12.2 Generic numeric

A generic numeric value is a numeric. The design of this Recommendation does not limit the numerical values in any way. The values may be integers, reals or complex; however, the coded representations may impose limitations. The ASN.1 syntax defined in this Recommendation supports only integer values.

NOTE – It is recommended that authors assume this limitation.

The generic numeric value may be specified as follows:

- a numeric constant;
- a result of the evaluation of a get action.

12.3 Generic integer

A generic integer value shall be an integer.

The generic integer value may be used as follows:

- an integer constant;
- a result of the evaluation of a get action.

12.4 Generic ratio

A generic ratio value is a pair of integers (m, n) with m being the numerator and n the denominator of a fraction. If n is omitted, it is assumed to be 100, interpreting m as a percentage. n shall be greater than 1.

A generic ratio value may be used as follows:

- a ratio constant;
- a result of the evaluation of a get action.

12.5 Generic string

A generic string value is a string of any number of characters. The design of this Recommendation does not limit the coding scheme of string values in any way. The values may be represented in any code sets. However, the coded representations may impose limitations. The ASN.1 syntax defined in this Recommendation supports only GraphicString.

NOTE 1 – It is recommended that the authors assume this limitation.

GraphicString contains all G sets + SPACE as defined in ASN.1. Such specification allows for international string.

The string is considered as a whole, i.e. the component characters cannot be addressed individually.

NOTE 2 – Typically, the character string is used as a label, e.g. “Help”.

The generic string value may be specified as follows:

- a string constant;
- a result of the evaluation of a get action.

There is a difference between the lowercase and uppercase character.

12.6 Generic reference

A generic reference is one of the references defined by this Recommendation (see Generic reference mechanism).

A generic reference value may be specified as follows:

- a reference constant;
- a result of the evaluation of a get action.

12.7 Generic list

A generic list value is an ordered set, possibly empty, of generic values of any type. Each element of a generic list value may have a distinct type from the other elements of the list. Each element of a generic list is implicitly indexed.

NOTES

1 – An element of a list may be another generic list.

2 – Each element, within a list or a sublist, may be retrieved separately using the Get Data action facility.

3 – The generic list type may be used to construct vectors (same type for all elements of the list) and compound sets (sets of elements of various types).

A generic list value may be specified as follows:

- a list of constants;
- a result of the evaluation of a get action.

13 Macro Mechanism

The macro mechanism provides a general technique for producing efficient coding of frequently used action and link objects in which only a few values are changed from one to another. This allows the sharing and reuse of complex behaviours. An author may create a catalogue of predefined action and link object templates, that are also called macro action object (see 18.4) and macro link object (see 19.5). In a macro action object or a macro link object, at least one attribute is encoded as a macro parameter value.

A macro parameter value is composed of:

- A Macro Def ID: It is either a string or an integer. It is used as a symbol representing the macro parameter. A given Macro Def ID may be used by many macro parameter values.
- A Default usage value: It is a value to be assigned to the attribute encoded as a macro parameter value when the macro usage value assignment is omitted.

Each link object contains an optional set of attributes called “macro parameter resolution” and which is used to resolve the macro parameter value contained in its link effect. Each macro parameter resolution is composed of a Macro Def ID and the corresponding macro usage value to be assigned to the Macro Def ID.

When a link object is fired, the link effect is to be processed. The first step of this process is called the macro resolution phase and is limited to this link effect process.

NOTE 1 – As the same link may be fired several times in parallel, therefore, the macro resolution phase is limited to a given link effect process.

The aim of this phase is to assign a macro usage value to each macro parameter value contained in the link effect. Each macro parameter resolution is propagated to the action object describing the link effect. The following applies:

- if this action is a nested action object, the macro parameter resolution is also propagated to the embedded action object;
- if an embedded link object is activated from this link effect, the macro parameter resolution is propagated to this embedded link object except if this embedded link object itself contains a macro parameter resolution which uses the same Macro Def ID.

During the propagation phase, for each macro parameter value, the following applies:

- The Macro Def ID of the macro parameter value is retrieved from one of the propagated macro parameter resolution: This macro parameter value is replaced by the corresponding macro usage value of this propagated macro parameter resolution.
- NOTE 2 – Even if the usage value is not compatible with the expected type of the attribute encoded as macro parameter value, the MHEG engine is required to assign it. It is for the object designer to ensure the compatibility.
- NOTE 3 – If the usage value is an evaluated value, the macro parameter value is replaced by this Get action without processing it.
- Otherwise, this macro parameter value is replaced by the corresponding default usage value encoded within this macro parameter value. If no default usage value is provided, this macro parameter value is replaced by the value “undefined”.

These abbreviations are used for the following examples:

- MPR Macro Parameter Resolution
- MDI Macro Def ID
- MUV Macro Usage Value
- DUV Default Usage Value (when DUV is omitted, no default usage value is provided).
- ALE Action in Link Effect

Example 1 – Simple link effect.

L1 has following specifications:

- LC1: Button1 becomes selected.
- MPR1: (MDI = TARGET1, MUV = 700:1), (MDI = TARGET2, MUV = 700:2).
- ALE1: New(MDI = TARGET1), New(MDI = TARGET2), Run(MDI = TARGET1), Run(MDI = TARGET2).

If LC1 is satisfied, the macro resolution phase is processed and ALE1 is processed as follows:

- ALE1: New(700:1), New(700:2), Run(700:1), Run(700:2).

Example 2 – Embedded action within link effect.

L2 has following specifications:

- LC2: Button1 becomes selected.
- MPR2: (MDI = SYNC, MUV = serial), (MDI = TARGET1, MUV = 700:1), (MDI = TARGET2, MUV = 700:2).
- ALE2: ActionObject1.

And ActionObject1 is as follows:

- Synchro indicator: SYNC.
- Action set: New(TARGET1), Run(TARGET1), Stop(TARGET2).

If LC2 is satisfied, the macro resolution phase is processed and each MUV is propagated to ActionObject1, and ActionObject1 is processed as follows:

- Synchro indicator: Serial.
- Action set: New(700:1), Run(700:1), Stop(700:2).

If synchro indicator is resolved as “undefined”, the action is not processed.

Example 3 – Embedded links within link effect.

L3 has following specifications:

- LC3: Button1 becomes selected..
- MPR3: (MDI = TARGET1, MUV = 700:1).
- ALE3: New(TARGET1), Activate(L10), Activate(L20).

L10 is as follows:

- LC10: If TARGET1 is “not running”.
- MPR10: None.
- LE10: Run(TARGET1).

And L20 is as follows:

- LC20: If TARGET1 is “not running”.
- MPR20: (MDI = TARGET1, MUV = 999:1).
- ALE20: New(TARGET1), Run(TARGET1).

If LC3 is satisfied, MPR3 is propagated to ALE3, L10 and L20. And L10 is processed as follows:

- LC10: If 700:1 is “not running”.
- LE10: Run(700:1).

However, L20 is processed as follows:

- LC20: If 700:1 is “not running”.
- ALE20: New(999:1), Run(999:1).

14 Hooks

This clause describes the support for the handling of hooks, which are used in content, script and descriptor objects. Hooks provide the information for the type identification of the encoded data. A hook consists of the encoding information that identifies the encoding method and an optional encoding description that may be used for specifying parameters of the encoding method.

Two kinds of hooks are provided, one for the content data and another for the script data:

- 1) **Content hook:** Composed of an identification field for the data encoding standard and a descriptive field:
 - a) The content encoding information contains an identification of the content encoding standard. A wide range of existing encoding standards such as MPEG, G711 or JPEG are maintained in the registered catalogue via the procedure described in ISO/IEC 13522-4. Private encoding formats can also be provided using a proprietary catalogue.
 - b) The content encoding description gives a more precise description of the characteristics needed to decode the content. The semantics of these parameters are not defined by MHEG but are given by the semantics of the data encoding Standard or Recommendation with respect to the using application. Such information may also be provided within the content encoding catalogues.

- 2) **Script hook:** Composed of an identification field for scripting language and a descriptive field:
 - a) Scripting encoding information contains an identification of the script encoding standard. A wide range of existing encoding standards such as C, C++ or Lisp are maintained in the registered catalogue via the procedure described in ISO/IEC 13522-4. Private encoding formats can also be provided using a proprietary catalogue.
 - b) Scripting language description gives a more precise description of the characteristics needed to decode the script. The semantics of these parameters are not defined by MHEG but are given by the semantics of the data encoding Standard or Recommendation with respect to the using application. Such information may also be provided within the script encoding catalogues.

15 Extensibility

Following extensibility is provided by this Recommendation:

- 1) Catalogues (see 15.1);
- 2) Addition of new MHEG object classes (see 15.2);
- 3) Extensibility Provision (see 15.3).

15.1 Catalogues

ISO/IEC 13522-1 provides a set of catalogues to maintain compatibility between MHEG engines and to allow extensibility. New concepts can be made available to all MHEG systems if you register a reference to a format identifier in a registered catalogue or only to one group of applications if you register it in a proprietary catalogue:

- 1) The registered catalogues are maintained in accordance with ISO/IEC 13522-4. Standardised, well-known concepts are listed in the registered catalogue. It is possible to register upcoming concepts in this catalogue.
- 2) Application-specific concepts reside in the proprietary catalogue, which is private to only one set of applications. These catalogues can only be referenced by special MHEG engines. The maintenance of the proprietary catalogue is assured by the using application.

Within each catalogue, an optional description may be associated with each entry to clarify the usage of it.

An MHEG engine can take any catalogue entry into account, thereby, introducing the concept of this entry within its process. When an author provides an MHEG object or an action using such an entry, this MHEG engine will be able to process this object or this action. If the author uses an entry not taken into account by the MHEG engine, the MHEG object or the action will be ignored by this engine.

For each of the following attributes used by this Recommendation, a registered catalogue is provided, and a using application may also maintain a proprietary catalogue for private use:

- **Content encoding** contains a precise description of the content encoding methods that may be used in the content hook, e.g. ISO/IEC 11172-2 (MPEG2 video). A content encoding description may be associated with each entry of this catalogue.
- **Content classification** is provided as an optional assistance in determining the type of content data. It provides information on the type of data. For media data it is used to indicate the perception media, e.g. text, graphics, audio. This information may be used in a negotiation process, a database, or by an MHEG engine to choose a decoder.
- **Script encoding** contains the script languages that may be used in the script hook, e.g. C++, Smalltalk, SMSL. A script encoding description may be associated with each entry of this catalogue.
- **Script classification** is provided as an optional assistance in determining the type of script language. This information may be used in a negotiation process, a database, or by an MHEG engine to choose a script decoder.
- **Media type** contains the medium type that may be used in the descriptor object in order to help the mapping of a channel to a physical device, e.g. text, still image or video.

- **Style** contains the styles that may be used in the set style action, e.g. for user interaction: button, slider, entry field. The styles may depend on the GUI on the platform and on user personalisation by the user. Additional information may be associated with each entry of this catalogue in order to define the style more precisely, e.g. a value range for a slider, an orientation for a menu.
- **Event** contains a list of catalogued event identifiers (e.g. mouse click, keyboard key pressed, remote control command) that may be used in the descriptor object to indicate the expected mapping of event identifiers provided by the author to the catalogued event identifiers. Additional information may be associated with each entry of this catalogue in order to define the event more precisely, e.g. the position of the click.
- **Extended elementary action** contains a list of catalogued elementary actions used in the “catalogued elementary action” action. These catalogued elementary actions are used to extend the number of elementary actions defined by this Recommendation, e.g. draw a line, arithmetic operation. Additional information may be associated with each entry of this catalogue in order to define the behaviour of an action more precisely, e.g. the parameters, the MHEG effect, the allowed periods, additional error conditions.
- **Extended attribute** contains a list of catalogued attributes used in the set catalogued attribute action. These catalogued attributes are used to extend the number of attributes of MHEG entities defined in this Recommendation, e.g. colour, text fonts. Additional information may be associated with each entry of this catalogue in order to define the attribute more precisely, e.g. the type.

15.2 Addition of New MHEG object classes

This Recommendation reserves numbers 0 to 9999 of ASN.1 OBJECT IDENTIFIER arc3. The numbers from 0 to 9999 shall not be used privately. However, a using application may use other numbers to create new object classes, that may be complete new object classes or modifications of certain existing object classes.

15.3 Extensibility Provision

In order to enable future extension of attributes, “Extensibility Provision” encoded as “...” in ASN.1 is provided. “Extensibility Provision” is added to the following places in ASN.1:

- “elementary action” attribute;
- “evaluated value” attribute;
- “description” attribute of MH-object class;
- each leaf class of MHEG object.

And for each “Extensibility Provision” attribute, a list of tags numbered from 0 to 9999 is reserved in order to be used in other Recommendations of the T.170 series. Reserved tags shall not be used privately.

SECTION 3 – OVERVIEW OF MHEG CLASSES

The following clauses present the structure provided by MHEG to interchange multimedia and hypermedia information. The MHEG engine parses this structure.

16 MHEG object classes overview

The object-oriented approach was chosen for the design of the standard because it fits the requirements of active, autonomous and reusable objects. The interchanged object classes have been classified as shown in Figure 21. Each interchanged object belongs to one of the **emphasised** classes. Abstract classes have been added to describe common attributes or to group classes dealing with similar topics. MHEG does not define methods on its classes. Thus, use of the object-oriented paradigm is limited to attribute inheritance.

In Figure 21, the greater-than sign (>) means “has the following subclasses”. Only the instances of classes that have been emphasised may be interchanged. The useful definitions are not considered to be a separate class. They are merely a convenient grouping of utility attributes.

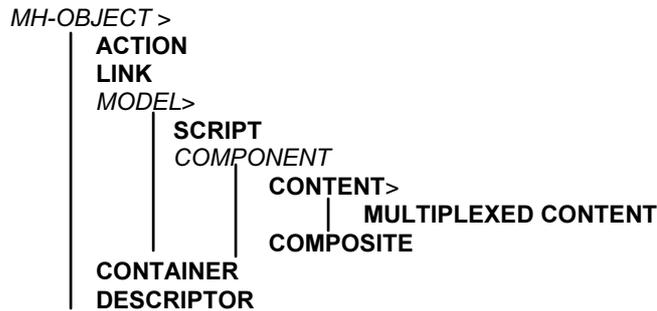


Figure 21/T.171 – MHEG inheritance tree

17 Structure of MH-Object Class

This clause describes the common structure provided for interchanging all the instances of MHEG classes, the MHEG objects, defined by this Recommendation. This mechanism provides a consistent approach to the identification and interchange of MHEG objects.

The MH-Object class provides the following information for the identification of MHEG objects.

17.1 Class identification

The class identification is composed of the identification of the standard, the standard version and a specific identifier unique for each MHEG object class. The class identification attribute is encoded at level C using the ASN.1 technique of OBJECT IDENTIFIER (see 8.2.3.1). For the ease of encoding, this attribute is not encoded within MH-object class at level C. This attribute is encoded individually for each interchanged class in order to assign a unique number for each class instead of inherited directly from MH-object class.

17.2 MHEG-ID

An MHEG-ID may be provided to uniquely identify an interchanged object within the scope of an application.

17.3 General object information

MH-object class also gives supplementary information for an MHEG object as follows:

- 1) Name of the MHEG object.
- 2) Owner of the MHEG object.
- 3) Version of the MHEG object.
- 4) Last modification date of the MHEG object.
- 5) List of keywords qualifying the MHEG object.
- 6) Readable copyright information relative to the MHEG object.
- 7) Copyright identifier used by certain authorised organisation to identify the copyright work type of the MHEG object. For example:
 - a) ISBN (International Standard Book Number) for book;
 - b) ISSN (International Standard Serial Number) for periodical publications;
 - c) ISRC (International Standard Record Code) for sound recording;
 - d) ISAN (International Standard for Audiovisual Number) for audiovisual application.

- 8) Copyright number used by certain authorised organisation to uniquely identify the copyright information assigned to the MHEG object within a copyright identifier scope, e.g. 2-11072557-5 is an ISBN copyright number.
- 9) Readable licence information relative to the usage of the MHEG object.
- 10) Cache priority, which may be used by the MHEG engine as a hint of how to manage the MHEG objects. This is an integer in the range from 0 to 255. The value 0 means that the MHEG engine should delete the object from its memory space entirely if it receives a Destroy action. The value 255 means that the MHEG engine is strongly encouraged to cache the MHEG object within its memory space at the time of a Destroy action. Other values may be interpreted by each MHEG engine independently taking into account the range. However, the MHEG engine is not obliged to follow this instruction. This is only provided as a hint information for the MHEG engine.
- 11) Readable comments relative to the MHEG object.
- 12) Extensibility provision used by other standards which describes an extension of this Recommendation.

18 Structure of Action Class

The action class defines a reusable arrangement of elementary actions. Action objects that are instances of the action class are used in link objects in order to describe their link effects. A given action object may be addressed by many link objects. Several link conditions in different links may be satisfied at the same instant and are assumed to be fired and processed in parallel. Therefore, the processing of actions describing the link effect has a local effect limited to the link itself.

The MHEG action class consists of following information:

- 1) A synchro indicator specifies the type of processing of the synchronised actions. Two values are defined: “parallel” and “serial”. When it is set to “serial”, all the actions within the “synchronised action set” are processed in serial. When it is set to parallel, all these actions are processed in parallel (see 31.3). Since synchronised actions may be processed in parallel, it is the author’s responsibility to take into account any side effects due to the parallelism. Specification of contradictory actions is deprecated.

NOTES

1 – An action object that calls for a parallel Run and stop of an rt-content is ambiguous since the final state is not defined.

2 – An action object that calls in the same serial group for an rt-content to run and stop after a delay of one second is not ambiguous. The rt-content is to be presented for one second.

3 – An MHEG engine is not required to perform true parallel processing, but authors must organise the group as if they were to be processed in parallel.

- 2) A synchronised action set is a set of elementary actions and/or other embedded action objects.

Using this structure, an author is able to describe the following action objects:

- Basic action object (see 18.2);
- Nested action object (see 18.3);
- Macro action object (see 18.4).

18.1 Elementary actions

This Recommendation defines a list of elementary actions that may be included in an action object to modify the behaviour of MHEG entities (see Section 4, “MHEG entities common behaviour”), e.g. Prepare, Run. The structure of each elementary action is as follows:

- 1) A target set is defined as a list of generic references or only one generic reference. Each elementary action is to be processed on the specified target set. If more than one target is specified in the target set, whatever the value of the synchro indicator is, the elementary action is processed in parallel on all the targets specified in the target set. Authors should assume that the MHEG engine applies the elementary action to the multiple targets in parallel.

A common target set can be specified for several elementary actions by using the alias mechanism (see 10.3) or macro parameter mechanism (see clause 13).

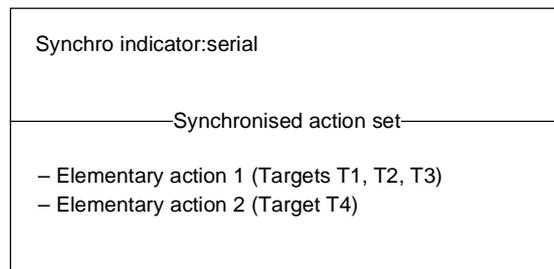
- 2) An optional transition duration may be provided for some elementary actions. A transition duration is expressed in GTU. When a transition duration is specified, the corresponding elementary action is to be processed during this transition duration. When no transition duration is specified, a null duration is taken into account.
- 3) The specific action parameters are defined for each elementary action (see Sections 4, 5, 6, 7 and 8).

18.2 Basic action object

A basic action object contains only elementary actions. It does not contain macro parameters or embedded action objects within its synchronised action set.

NOTE – The using application is responsible for determining the appropriate number of elementary actions .

In Figure 22, the MHEG engine completes the parallel processing of the MHEG effect of elementary action 1 on targets T1, T2 and T3. When completed, the MHEG effect of elementary action 2 is processed on T4.



T0825680-96/d021

Figure 22/T.171 – Example of a basic action object

18.3 Nested action object

In order to describe more complex behaviour, an embedded action object may be contained in the synchronised action set. The embedded action object can be itself a nested action object. A complete hierarchy of action objects can be designed using this mechanism.

NOTE – The using application is responsible for determining the appropriate depth of nesting .

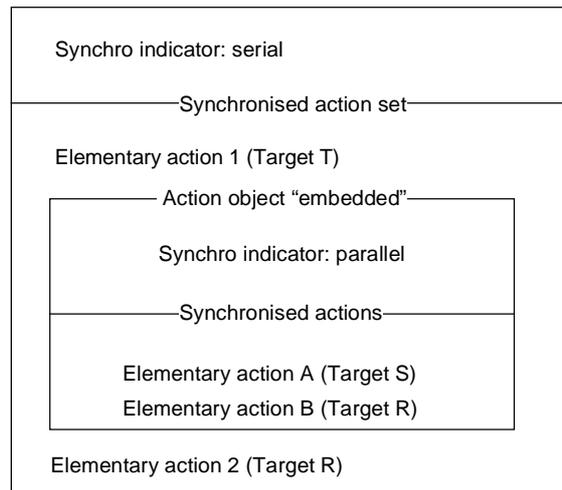
In the example of Figure 23, the MHEG engine processes the MHEG effect of elementary action 1 on target T. When completed, the MHEG engine processes, in parallel, the MHEG effect of elementary action A on target S and the MHEG effect of elementary action B on target R. When completed, the MHEG effect of elementary action 2 is processed on target R.

18.4 Macro action object

In order to produce efficient coding of frequently used action objects in which only a few values change from one link to another, the synchro indicator attribute of an action object and each parameter of an elementary action may be specified as a macro parameter value. A macro action object contains at least one macro parameter value. Each macro parameter value is resolved when the link object which uses this action is fired (see 19.3 for more details).

19 Structure of Link Class

An MHEG link class is defined for specifying spatial, temporal and conditional relations between, and actions upon, MHEG entities. The link contains a condition, which when satisfied evaluates to true or provokes the processing of actions on targets that produce the required effect. The link is said to be fired.



T0825690-96/d022

Figure 23/T.171 – Example of a nested action object

Only those link objects that have been activated can be firable (see 29.2). Several link conditions in different links may be satisfied at the same instant and are assumed to be fired and processed in parallel by the MHEG engine.

The link object is separated from the objects that produce the triggering contexts. The basic behaviour of each MHEG entity (e.g. volume, channel on/off) is part of the entity itself. During the life of the entities, change in the behaviour can be used to trigger a separate link object that calls for the actions to be processed. These actions, in turn, modify the basic behaviour of any other MHEG entity, for example, to increase a volume. This may in turn generate further changes in behaviour and trigger further links.

Each instance of a link class contains a link condition and a link effect.

Using this structure, an author is able to describe the following types of link objects:

- Basic link object (see 19.3);
- Nested link object (see 19.4);
- Macro link object (see 19.5).

19.1 Link condition

The link condition is defined as either a trigger condition (see 19.1.1) or a logical combination of trigger and constraint conditions (see 0).

The link condition is evaluated as follows:

- 1) When a link object becomes active, the link condition is evaluated to false.
- 2) If the link condition is a single trigger condition, the link condition evaluates the value of the trigger condition. If the link condition is a logical combination of conditions, the link condition evaluates the value of the top node of the logical tree combining the conditions (see Figure 25).
- 3) If the link condition is evaluated to true, the link is fired, i.e. its link effect is processed by the MHEG engine (see 31.2). If the link condition is evaluated to false or “undefined”, the link is not fired, i.e. its link effect is not processed.

A link may be fired several times during its activation, each time its link condition becomes true.

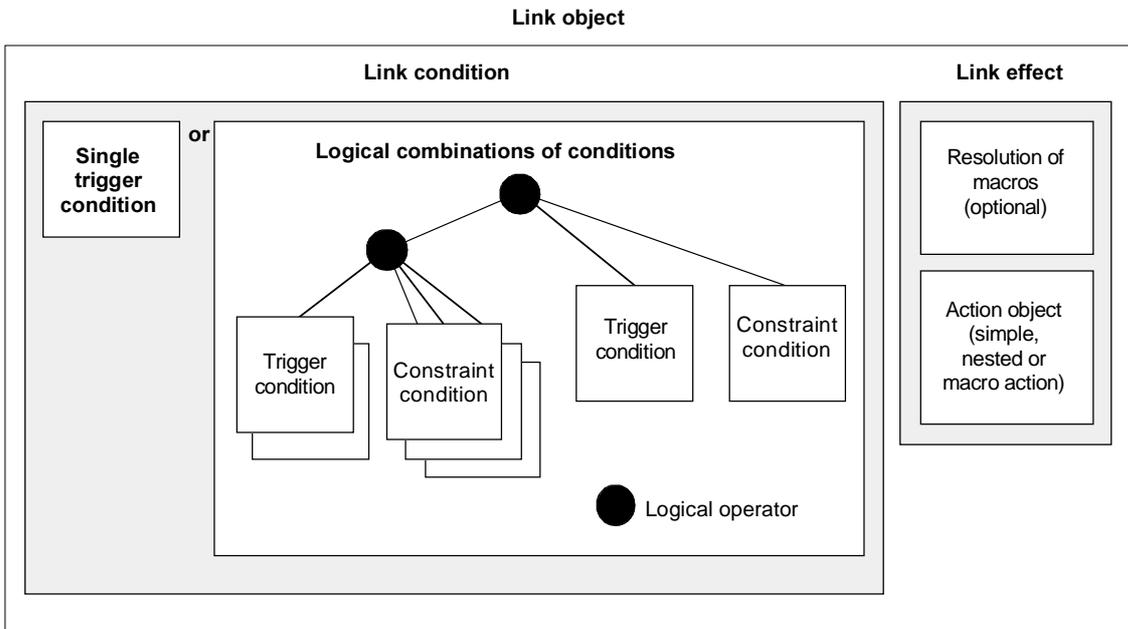


Figure 24/T.171 – Link object structure

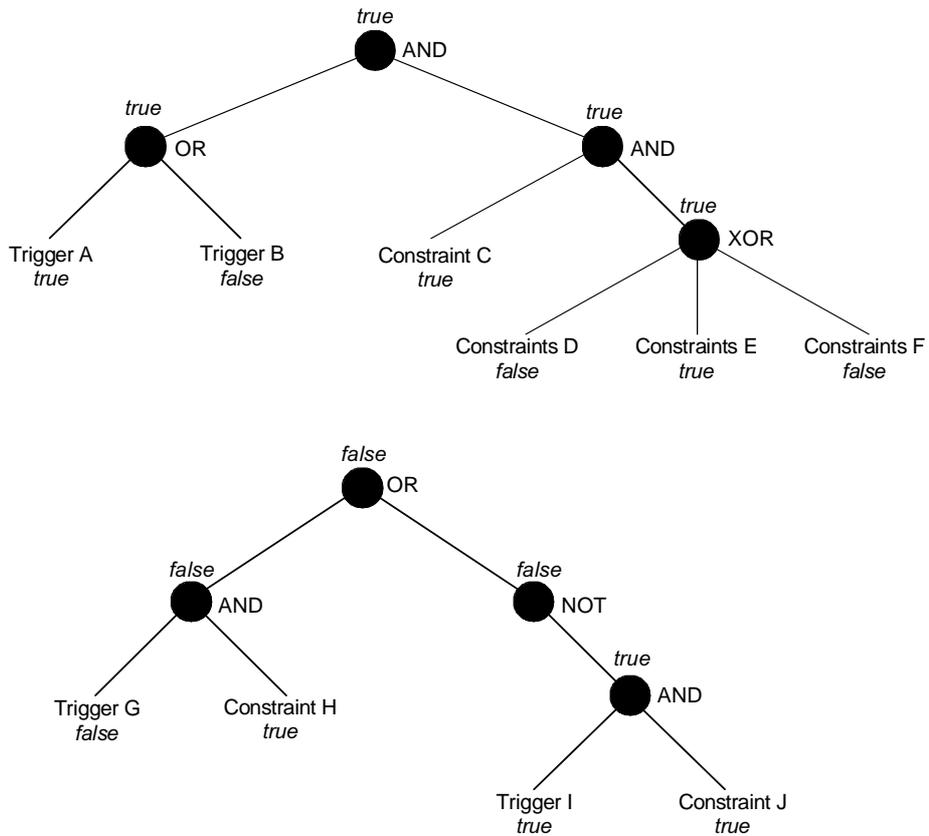


Figure 25/T.171 – Examples of logical tree describing a logical combination of conditions

19.1.1 Trigger condition

A trigger condition describes the occurrence of a change in MHEG entities' behaviour triggering the link. The trigger is described as a change of attribute or status value of an MHEG entity. The following behaviour changes are triggerable:

- 1) Temporal changes: Provoked by timestones or delay.
- 2) Action changes: Consequences of any elementary action.
- 3) Interaction changes: Occur, for example, when the user selects a menu or clicks on a button.
- 4) Any catalogued event changes: Occur, for example, when a system event happens.

The trigger condition is expressed as a transition, i.e. as a double condition on the same source. It is composed of the following:

- a source value;
- a previous condition, which expresses the condition to be satisfied before the transition occurs;
- a current condition, which expresses the expected condition to satisfy after the transition occurs.

A trigger condition evaluates a boolean value or "undefined". The following applies:

- 1) When a link object becomes active, each trigger condition is evaluated to false.
- 2) Evaluates the previous condition of each trigger condition.
- 3) The source value of each trigger condition is an evaluated value on a target attribute or a status, e.g. get preparation status (target). When the value of this attribute or status changes, the following steps are applied for the corresponding trigger condition:
 - a) assigns this value to the current comparison value and evaluates the current condition;
 - b) evaluates the trigger condition by processing the following logical operation: "previous condition" AND "current condition":
 - i) if the result is true and if the link condition is defined as a single trigger condition, the link condition is evaluated to true;
 - ii) if the result is true and if this trigger condition is part of a logical combination of conditions, this logical combination of conditions is to be evaluated (see 19.1.8).
 - c) Evaluates the trigger condition to false.
 - d) Go to 2.

NOTE – Steps 3) a) and 3) d) should be considered as an atomic operation for link, i.e. if a value tested by a trigger condition changes when the MHEG engine is processing steps 3) a) to 3) d), the processing of such a change is delayed until the current processing enters in step 3.

Table 1 shows some examples of trigger conditions. In the example, Get CV (rt-content N) means Get CV action targeted to the rt-content N in order to retrieve the current audible volume (CV), Get Event (rt-content N) means Get Event action targeted to the rt-content N in order to retrieve the number of most recently happened event, and Get Data (content A) means Get Data action targeted to the content object A in order to retrieve the contained generic value contained.

Table 1/T.171 — Examples of trigger conditions

Trigger source value	Previous condition		Current condition		Semantics	Nature of the Change
	Operator	Previous value	Operator	Current value		
Get CV (rt-content N)	==	10	<	5	rt-content N had CV equal to 10, and decreases to less than five.	Change from a specified previous value to a specified current value.
Get CV (rt-content N)	==	Get Data (content A)	<	Get Data (content B)	rt-content N had CV equal to the value specified in the data of content object A, and changes to a value less than that of the value in content object B.	
Get CV (rt-content N)	Any	“unspecified”	Any	“unspecified”	The CV of rt-content N has changed. The value is not significant.	Change from an "unspecified" value to an "unspecified" value. The trigger is on a change of the source value without regard for its previous and current values.
Get CV (rt-content N)	==	10	Any	“unspecified”	The CV of rt-content N was 10 and has changed. The new value is not significant.	Change from a specified value to an "unspecified" value. The trigger is on a change of the source value without regard for its current value.
Get CV (rt-content N)	Any	“unspecified”	>	10	The CV of rt-content N becomes greater than 10. The previous value is not significant. Note that the previous value may have been greater than 10.	Change from any previous value to a specified current value range.
Get CV (rt-content N)	Any	“unspecified”	==	10	The CV of rt-content N becomes equal to 10. The previous value is not significant. Note that the previous value may have been equal to 10.	Change from an "unspecified" value to a specified current value. The trigger is on a change of the source value whatever its previous value.
Get CV (rt-content N)	<	Get CV (rt-content P)	>	Get CV (rt-content P)	The CV of rt-content N was less than the CV of rt-content P and has become greater.	Change with respect to other rt-contents.
Get Event (rt-content N)	Any	“unspecified”	==	10	The event 10 has occurred on rt-content N. The previous event value is not significant.	Becomes true any rising of the event 10.
Get Event (rt-content N)	==	10	==	10	The previous event occurred to rt-content N was 10. The same event occurs again.	Becomes true for each successive rising of the event 10.
Get CV (rt-content N)	Previous condition is omitted, i.e. it is the negation of the current condition		≥	5	The CV of rt-content N was less than 5 and increases greater than or equal to 5. The previous condition is interpreted as $CV < 5$. The previous value range is $(-\infty, 4)$, the current value range is $(5, +\infty)$. These two value ranges do not overlap and their union is the whole range of integer numbers.	Change from a previous value range to a current value range where the two value ranges do not overlap and where the union of the value ranges is the largest possible value range. In that case, the previous condition is omitted, and so interpreted as the negation of the current condition.

19.1.2 Constraint condition

A constraint condition may be provided within a logical combination of conditions. It expresses a required contextual state at the moment when one of the trigger conditions is satisfied. A constraint condition makes it possible to specify more precisely the context in which the link condition will be satisfied.

A constraint condition is composed of a source value and a current condition which expresses the expected condition to satisfy.

NOTE – Constraints do not require a previous condition because they express contextual states at the moment when the trigger is satisfied.

A constraint condition evaluates a boolean value or “undefined”. It is satisfied when the specified current condition is satisfied for the source value. The constraint conditions are to be evaluated at the instant when one of the trigger conditions becomes true.

Table 2 shows some examples of constraint conditions.

Table 2/T.171 – Examples of constraint conditions

Source Value	Current condition		Semantics
	Operator	Current Value	
Get CV (rt-content N)	<	5	rt-content N has CV value less than five.
Get CV (rt-content N)	<	Get Data (content A)	rt-content N has CV value less than that of the value in content object A.
Get CV (rt-content N)	==	5	rt-content N has CV value equal to five.

19.1.3 Source value

The source value is a value used as the base of the comparison described in the previous condition and the current condition. The presence of the source value in the condition is mandatory.

The source value is always provided by an evaluated value, i.e. it is expressed as a result of a get action.

19.1.4 Comparison value

The comparison value is a value that is to be compared to the source value using the comparison operator in the previous and the current conditions. The presence of the comparison value in the condition is mandatory.

The comparison value is specified as one of the following:

- 1) An evaluated value, i.e. it is expressed as a result of a get action.
- 2) A constant value.
- 3) One of the comparison value constants defined by this Recommendation, e.g. “ready”, “not ready” for the preparation status.
- 4) “Unspecified” value to indicate that the value to be compared with the source is not important; what is important is that the value has changed. The following applies:
 - a) Within a trigger condition:
 - i) Within the previous condition: It means that the trigger condition is on a change of the source value whatever its previous value was.
 - ii) Within the current condition: It means that it is a condition on a change of the source value whatever its current value becomes.

- iii) Within both the previous and the current condition: It means that it is a condition on a change of value whatever its previous value was and whatever its current value becomes.

The current value can be the same as the previous value, but the engine as wilfully set the same value to an attribute or a status. This is particularly useful for successive rising of the same event (see 6.3.7) or timestone (see 72.16.14), i.e. the same event may occur several times, each time the event identifier is set to the corresponding value.

- b) Within a constraint condition: It means that the constraint condition is always evaluated to true.

19.1.5 Previous condition

The previous condition is used in trigger condition only. It is composed of a comparison operator and a previous comparison value.

The previous comparison value is a generic value (see clause 11).

When the previous condition is omitted, the previous condition is to be interpreted as the negation of the indicated current condition, and the following applies:

- the comparison operator of the previous condition is the negation of the comparison operator of the current condition;
- the comparison value of the previous condition is the same as the comparison value of the current condition.

The previous condition is satisfied when the result of the comparison operation between the source value and the previous comparison value using the specified comparison operator is evaluated to true.

19.1.6 Current condition

The current condition is mandatory in a generic condition, and it is composed of a comparison operator and the current comparison value.

The current comparison value is a generic value (see clause 11).

The current condition is satisfied when the result of the comparison operation between the source value and the current comparison value using the specified comparison operator is evaluated to true.

19.1.7 Comparison operator

The comparison operator used in the previous and the current condition is one of the following:

- == Comparison for equality;
- != Comparison for inequality;
- < Comparison for strict inferiority;
- <= Comparison for inferiority or equality;
- > Comparison for strict superiority;
- >= Comparison for superiority or equality.

NOTE 1 – A using application may provide other comparison operators.

A comparison operation is a comparison between the source and a comparison value. The result of the comparison is as defined in Table 3.

Table 3/T.171 – Comparison operations

Comparison operator	Source Value (SV)	Comparison Value (CV)	Result Value (Boolean or “undefined”)	Remarks
==	Generic boolean Generic numeric Generic integer Generic ratio Generic string Generic reference Generic list	Generic boolean Generic numeric Generic integer Generic ratio Generic string Generic reference Generic list	True: if SV == CV False: otherwise	See 1) See 2) See 3) See 4)
!=	Generic boolean Generic numeric Generic integer Generic ratio Generic string Generic reference Generic list	Generic boolean Generic numeric Generic integer Generic ratio Generic string Generic reference Generic list	True: if SV != CV False: otherwise	See 1) See 2) See 3) See 4)
<	Generic numeric Generic integer Generic ratio Generic list of generic numeric / generic integer / generic ratio	Generic numeric Generic integer Generic ratio Generic list of generic numeric / generic integer / generic ratio	True: if SV < CV False: otherwise	See 1) See 4)
<=	Generic numeric Generic integer Generic ratio Generic list of generic numeric / generic integer / generic ratio	Generic numeric Generic integer Generic ratio Generic list of generic numeric / generic integer / generic ratio	True: if SV <= CV False: otherwise	See 1) See 4)
>	Generic numeric Generic integer Generic ratio Generic list of generic numeric / generic integer / generic ratio	Generic numeric Generic integer Generic ratio Generic list of generic numeric / generic integer / generic ratio	True: if SV > CV False: otherwise	See 1) See 4)
>=	Generic numeric Generic integer Generic ratio Generic list of generic numeric / generic integer / generic ratio	Generic numeric Generic integer Generic ratio Generic list of generic numeric / generic integer / generic ratio	True: if SV >= CV False: otherwise	See 1) See 4)
Any	Any	CV type is not the same as SV type	False	See 5)
Any	Any SV type “undefined” “undefined” “undefined”	“undefined” Any CV type “unspecified” “undefined”	“undefined”	See 6)
Any	Any SV type	“unspecified”	True	See 7)

This Recommendation does not define the semantics of operations on types of values different from those presented in Table 3. If such an operation is to be processed by the MHEG engine and if no semantics have been provided by the using application, this operation is evaluated to the value “undefined”.

NOTE 2 – True > False and “Hello” <= “Bye!” are evaluated to “undefined”.

In order to evaluate a result of some operations, the following applies:

- 1) Two generic ratios shall be compared as follows:
 - a) If the denominator is omitted, it is assumed as 100. For example: (10/100) == (10/omitted) is true.

- b) A generic ratio is reduced if it is not irreducible. This reduction should be done only for comparison purposes. The value of the generic ratio still remains as it was before reduction. For example: (10/100) becomes (1/10) for comparison.
 - c) A generic ratio is considered as a list containing two generic integer elements. So, the comparison between two lists applies (see below).
- 2) The comparison between strings is made at the level of each character. The uppercase characters are distinguished from the lowercase characters.

NOTE 3 – It is similar to a comparison between two lists (see below). The strings need to have the same size, and each character needs to be identical.

NOTE 4 – Examples: Comparison of strings: “HELLO” == “HELLO” is true. “hello” == “HELLO” is false. “Hello” != “Hello” is false.

- 3) The comparison of two references is made on the addressed entities. Two different references may address the same entity. In that case, the comparison for equality is evaluated to true. A comparison may be made between references addressing groups of entities. In that case, the comparison is made at the level of the addressed entities. The comparison for equality between two groups not having the same number of entities is evaluated to false. When the MHEG engine is unable to compare two references, the result of the comparison is evaluated to the value “undefined”.

NOTE 5 – This situation may occur when references are not yet resolved by the MHEG engine or when a reference addresses an unknown object.

NOTE 6 – For example: A content object has an MHEG-ID as 1000 and its external ID is “e:\mydir\content.mhg”. 1000 == “e:\mydir\content.mhg” is true. If an alias, e.g. ContentA, is assigned to this content, ContentA == 1000 is true, ContentA != “e:\mydir\content.mhg” is false.

- 4) Two lists shall be compared as follows:
- a) If the two lists have different lengths (i.e. not the same number of elements), the result of the comparison is evaluated to false.
 - b) If the two lists have the same length (i.e. the same number of elements), the comparison is realised at the level of each element. Two elements in the same position of the two lists are compared using the comparison operator:
 - i) If one of the comparisons at the level of elements is evaluated to false, the comparison at the level of the list is evaluated to false.
 - ii) If one of the comparisons at the level of elements is evaluated to “undefined”, the comparison at the level of the list is evaluated to “undefined”.
 - iii) Otherwise the comparison at the list level is evaluated to true. A logical AND operation is processed between the result of each compared elements.
 - c) Two lists of length zero are considered to be strictly equal. For example: comparison of generic list of generic numeric: (2, 3, 5) == (2, 3) is false, (2, 3, 5) == (2, 3, 5) is true, (2, 3, 5) == (2, 4, 5) is false, (2, 3, 5) > (0, 1, 6) is false because 6 > 5, () <= () is true.
- 5) The result of a comparison between non-compatible types is always evaluated to the value false. For example: comparison of non-compatible values: 3 != true is false, 5 != “a” is false.
- 6) The comparison of two values is always gives the value “undefined” if one of the values is an “undefined” value. For example: (2, 3, 5) == (2, undefined, 5) is “undefined”, (2, 3, 5) > (0, 1, undefined) is “undefined”, “undefined” == “undefined” is “undefined”, “undefined” == “unspecified” is “undefined”.
- 7) A given comparison value may be specified by the author as “unspecified” in the place of any comparison value. The comparison of a value with an “unspecified” value always gives true except if the value to be compared is “undefined”.

NOTE 7 – For example: (2, 5, 7) <= (2, unspecified, unspecified) is true, (2, 5, 7) >= (2, unspecified, unspecified) is true, “Hello” == unspecified is true, “undefined” <= “unspecified” is “undefined”.

NOTE 8 – Distinguish clearly between “undefined” and “unspecified” values. The value “undefined” is not provided by an author. It is always a result of a get action with an error or a result of a comparison operation containing a source value or a comparison value evaluated to “undefined”. The value “unspecified” is a wilful omission by the author.

19.1.8 Logical combination

A logical combination of conditions is defined as follows:

- 1) A logical operator defines a logical operator to be applied to the list of conditions.
- 2) A list of conditions. Each condition is either a trigger condition, a constraint condition, or another logical combination of conditions. It is a recursive structure. If the operator NOT is used, the list of conditions shall contain only one condition. If the other operators are used, the list of conditions shall contain at least two conditions.

A logical combination of conditions has the form of a logical tree. The leaves of the logical tree are trigger or constraint conditions. A logical operator is attached to each node.

The two trees described in Figure 25 show different combinations of conditions. Any other types of combinations may be provided. Some combinations are meaningless, for example OR (Trigger Condition K, Trigger Condition L, Constraint Condition M). In that case, the constraint condition is useless.

The evaluation of the logical combination of conditions is as follows:

- 1) Each trigger condition is evaluated as specified in 19.1.1.
- 2) Each time one of the trigger conditions becomes true, the following applies:
 - a) each constraint condition is evaluated as specified in 0;
 - b) then each node is evaluated as follows in a bottom up manner:
 - i) the logical operator attached to the node is applied on the evaluated conditions attached to this node, as specified in 19.1.9;
 - ii) each node evaluates the result of the previous logical operation, i.e. a boolean value or "undefined".
- 3) The link condition evaluates the value of the top node, i.e. the logical tree's root in Figure 25.

19.1.9 Logical operator

The logical operator is used within a logical operation. The following logical operators are defined:

- 1) AND: logical and;
- 2) OR: logical or;
- 3) XOR: logical exclusive or;
- 4) NOT: logical negation.

A logical operation is composed of a logical operator and a list of operands. The result of the logical operation is evaluated as follows when the operator NOT is used:

- 1) If the operator NOT is used with one operand, the result is as follows:

a) NOT True	evaluates:	False
b) NOT False	evaluates:	True
c) NOT "undefined"	evaluates:	"undefined"
- 2) Otherwise the result is "undefined".

The result of the logical operation is evaluated as follows when operator NOT is used:

- 1) If the operator AND, OR or XOR is used with n operands, n greater than 2, the logical operation is decomposed in n-1 operations, that are evaluated from left to right.
For example: AND (c1, c2, c3, ..., cn-1, cn) is decomposed as
AND(...AND(AND(AND(c1, c2), c3), ...), cn)
- 2) If the operator AND is used on two operands, the result is as follows:

a) True AND True	evaluates:	True
b) False AND False	evaluates:	False
c) "undefined" AND "undefined"	evaluates:	"undefined"
d) True AND False	evaluates:	False
e) True AND "undefined"	evaluates:	"undefined"
f) False AND "undefined"	evaluates:	"undefined"

- 3) If the operator OR is used on two operands, the result is as follows:
 - a) True OR True evaluates: True
 - b) False OR False evaluates: False
 - c) “undefined” OR “undefined” evaluates: “undefined”
 - d) True OR False evaluates: True
 - e) True OR “undefined” evaluates: True
 - f) False OR “undefined” evaluates: False
- 4) If the operator XOR is used on two operands, the result is as follows:
 - a) True XOR True evaluates: False
 - b) False XOR False evaluates: False
 - c) “undefined” XOR “undefined” evaluates: “undefined”
 - d) True XOR False evaluates: True
 - e) True XOR “undefined” evaluates: “undefined”
 - f) False XOR “undefined” evaluates: “undefined”
- 5) If the operator AND, OR or XOR is used on one operand, the result is “undefined”.

19.2 Link Effect

The link effect contains the following information:

- 1) An optional macro resolution used to assign a usage value to each macro parameter. The macro resolution is defined as a list of macro parameter resolutions. Each macro parameter resolution is defined as:
 - a) macro Def ID identifies one of the macro parameters;
 - b) macro usage value specified as a generic value.
- 2) An action, which may be a Basic action object (see 18.2), a nested action object (see 18.3) or a macro action object (see 18.4).

The link effect is processed as specified in 31.2 when the link condition becomes satisfied, i.e. it becomes evaluated to true.

19.3 Basic link object

A basic link object is defined as a combination of a basic link condition and a basic link effect. The combinations that can be made can form a simple link object or a more complex one.

A basic link condition does not contain any macro parameter. The following basic link conditions can be defined:

- 1) TC;
- 2) simple logical combination of conditions, e.g. TC AND CC, TC1 OR TC2;
- 3) complex logical combination of TC and CC, e.g. NOT (((TC1 AND CC1) OR (TC2 AND NOT (CC3 AND CC4 AND CC5))) XOR (TC3 AND CC6)).

A basic link effect does not contain any Activate actions (targeted to link objects). The following basic link effects can be defined:

- 1) no macro parameter resolution and a basic action object;
- 2) no macro parameter resolution and a nested action object;
- 3) a set of macro parameter resolution and a macro action.

19.4 Nested link object

In order to describe more complex behaviour, a nested link object may be provided by an author. A nested link object contains an Activate elementary action within its link effect. The target of this Activate elementary action is called embedded link object. When the nested link object fires, the embedded link object is activated.

The embedded link object can be itself a nested link object. A complete hierarchy of link objects can be designed using this mechanism.

NOTE – For example, considering the link objects L1, L2 and L3 as follows:

- L1 has LE1 as (run Image1, activate L2);
- L2 has LE2 as (run Image2, activate L3);
- L3 has LE3 as (run Content6).

If the link condition of L1 is satisfied, presentation of Image1 begins and embedded L2 is activated each time due to the LE1. If the link condition of L2 is satisfied after this activation of L2, presentation of Image2 begins and embedded L3 is activated due to the LE2. And so on.

The links L2 and L3 are called embedded links from the L1 point of view. The link L3 is called embedded link from the L2 point of view.

19.5 Macro link object

In order to produce efficient coding of frequently used link objects in which only a few values change from one link condition to another, a macro link object is an embedded link object which contains at least one macro parameter value within its link condition. Each macro parameter value is resolved when it is activated from a nested link object (see 6.3.13).

NOTE – The macro link object needs to be an embedded link object, otherwise the macro parameter value will never be resolved, so the link condition described will never be satisfied.

20 Structure of Model Class

The model objects, instance of model classes, may be interchanged within or across using applications. A model object is considered as a template object. From this model, rt-objects may be created based on instructions given by the author.

The model class is an abstract class inherited by script class and component class.

Any number of rt-objects may be created from a given model object. The activation of an rt-object does not affect the model object. This allows the reuse of the same model object in different contexts, i.e. different rt-objects.

The internal representation of the rt-objects is not defined by this Recommendation. Each MHEG engine will have its own internal representation technique.

NOTE – The model class contains no interchange attribute.

21 Structure of Script Class

An MHEG script class is defined for specifying complex conditional actions upon MHEG entities. The script objects, instances of script class, may be interchanged within or across applications. A script object is as a model object. From this model, rt-scripts may be created based on instructions given by the author.

The MHEG script class provides the following information for the interchange of scripts.

- Optional script classification (see clause 14): Provides as an assistance in determining the type of the script data.
- Script hook (see clause 14): Identifies the scripting language and describes the encoding and decoding information enabling the use of the script data. It is composed of an identification and a description of the scripting language.
- Script data: Inclusion or reference of the encoded script itself or to a “Null-Data”.

22 Structure of Component Class

The component class is an abstract class inherited by content class and composite class.

The component objects, instances of component classes, may be interchanged within or across using applications.

A component object is a model object. From this model, rt-components may be created based on instructions given by the author.

In addition to the attributes inherited from model class, the component class defines an optional “OPS initialisation” attribute, which is composed of an OD and an OS. This attribute is used to initialise the OPS of each rt-component created from this component object. The OD and OS are used to initialise the OPS-T axis and X, Y, Z axes lengths. The AVR is set to the range value provided by this Recommendation. The OD and OS are provided as an optional assistance to the MHEG engine. It is for the object designer to ensure that this information is compatible with similar information that may be contained in the hook or in the data.

23 Structure of Content Class

This clause describes the support provided by this Recommendation for the handling of encoded data. This mechanism is common to all types of encoded data and provides a consistent approach to the identification of the type and the interchange of the following:

- Data that may be presented to, and perceived by, the user, e.g. graphics, audio data.
- Data that cannot be presented to the user in an immediate way.
- Null data that may or may not be presented to the user depending on information given in hook or classification attributes or depending on the styles applied, e.g. a transparent area with a rectangle around the area.
- Generic value stored in the data – This value may be presented to the user depending on information given in hook or classification attributes or depending on the styles applied, e.g. a numeric may be presented as a pure numeric or as a slider, a string as a pure string or as a button. This value may be also used to store a value to be retrieved later, e.g. for a comparison in a link condition or for another attribute assignment. Specific get data and set data actions allow to retrieve and to modify this value, e.g. Set CV [target set, Get Data (content target)].

A content object is a model object. From this model, rt-contents may be created based on instructions given by the author.

An MHEG content class is defined that provides the following information for the type identification and interchange of encoded data:

- Optional data classification provided as an assistance in determining the type of the data. The classification is either proprietary or registered, e.g. generic value, graphics or video (see clause 14).
- Content Hook describes the encoding and decoding information enabling the use of the encoded data (see clause 14).
- Optional OV provided as an assistance in determining the current audible volume encoded within the data. When provided, this OV shall be defined within the AVR (see 9.1.3).
- Content Data – Inclusion or reference to a generic value, an encoded data provided by other Recommendations and Standards and by this Recommendation or a “Null Data”. The generic value in this Recommendation is encoded in ASN.1 using the syntax defined for Generic Value in the Useful Definition module.

24 Structure of Multiplexed Content Class

This clause describes the support provided by this Recommendation for the handling of multiplexed data. This mechanism extends the mechanism provided by content class.

An MHEG multiplexed content class provides a list of ordered streams in addition to the information provided by content class. It describes the streams contained in the multiplexed data.

Each stream is composed of the following:

- 1) Stream identifier – Each stream has a unique identification within a multiplex.
- 2) Optional catalogued content classification – It is provided as an assistance in determining the type of this stream (see clause 14).
- 3) Content hook – It describes the encoding and decoding information enabling the use of this stream (see clause 14).

NOTE – An example of a multiplexed content object:

- *Data Hook:*
 - MPEG-System
- *Data:*
 - Multiplexed data
- *Streams:*
 - Stream identifier: 2.1, stream hook: MPEG2-Video
 - Stream identifier: 2.4, stream hook: MPEG2-Audio
 - Stream identifier: 2.5, stream hook: MPEG2-Audio

25 Structure of Composite Class

This class provides the support for associating multimedia and hypermedia objects. This mechanism provides a consistent approach to the synchronisation in time and space and linking of a set of objects.

This class provides also the logical structure to describe the list of possible interactions offered to the user but does not define the interaction facilities provided by the user interface. Such interaction may be achieved in a variety of ways, for example, Graphical User Interfaces, keyboards, etc. This Recommendation does not define the look and feel of multimedia interactive presentations, or does it propose to change or add concepts to those that exist in typical Graphical User Interfaces. As this Recommendation is generic and independent of platform and implementation, it describes interaction at a virtual level. It is for a using application to apply these mechanisms using its specific look and feel.

NOTE 1 – In this way, this Recommendation achieves a consistent interface between an MHEG engine and the user interaction services or other applications, while still allowing the author to maintain the local look and feel.

Each composite object provides the means to describe one generation of information and its behaviour. Each generation is made up of a list of elements. The element is the basic building block of the composite class. Each element provides also a means to point to another generation defined by another composite object. Thus, a recursive structure is built up with the repetitive use of composite object and pointers in their elements.

Each generation represents a unit in the logic of the presentation intended by the author.

There are typically intrinsic relationships between sibling elements of a same generation.

There are also typical intrinsic relationships of propagation between generations.

NOTE 2 – A typical use of a composite object is to define one element for each choice to be offered to the user in the main menu. A second composite is used to describe the choices of the submenu.

NOTE 3 – Another typical use of a composite object is to define elements that are to be presented simultaneously.

The MHEG composite class provides the following information in addition to a generic identification of the composition:

- 1) Composition behaviour: This set of behaviours may describe the sequencing and interrelationships between the elements of this composite and the hyperlinking between different kinds of information. The composition behaviour contains the following information:
 - a) Predefined behaviour for specifying initial behaviours – The following start-up and close-down are defined:
 - i) Availability start-up: When the composite object is prepared.
 - ii) Availability close-down: When the composite object is destroyed.

- iii) rt-availability start-up: When one rt-composite is created from the composite object.
 - iv) rt-availability close-down: When one rt-composite is deleted.
 - b) A set of links for specifying conditional actions.
 - c) A set of actions – Actions and links are used to describe the behaviour of each element and the interrelationships between parent and siblings.
- 2) The number of elements contained in the composition defined by this composite.
 - 3) Composition elements – Each composite defines one generation. One generation is composed of the list of all the sibling elements.

Empty elements are not represented in Figure 26.

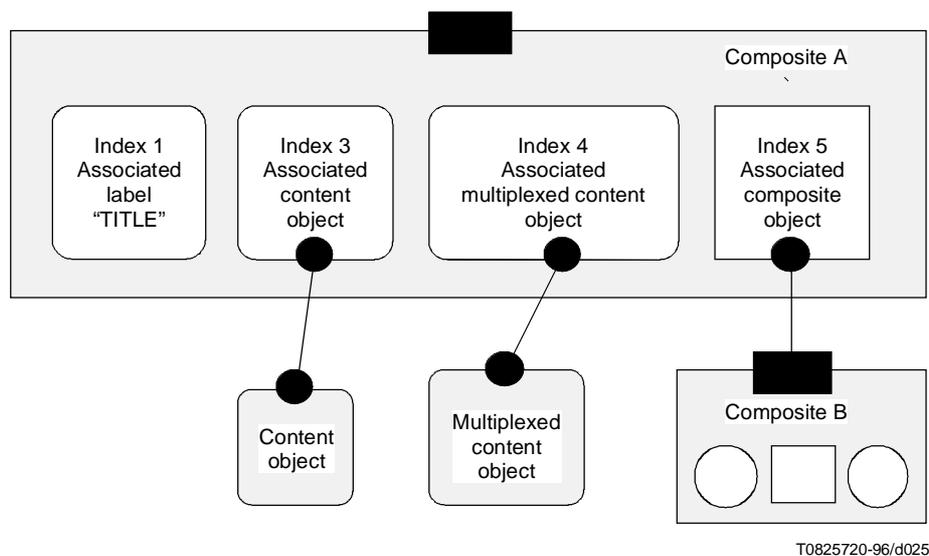


Figure 26/T.171 – Example of a composite object structure

25.1 Availability Start-up

The availability start-up is a link object that always has the same link condition: when the preparation status of the composite object becomes ready [i.e. Get Preparation Status (“this”) becomes ready]. The link effect may be used for additional preparation effect of the composite. It is either a personalised link effect provided by the author or one of the following default link effects:

- Automatic start-up 1: Activate all the included or referenced links within the specific behaviour.
- Automatic start-up 2: Prepare all the referenced associated model objects.
- Automatic start-up 3: Prepare all the referenced action and link objects in specific behaviour.
- Automatic start-up 4: Activate all the included or referenced links within the specific behaviour, and prepare all the referenced associated model objects.
- Automatic start-up 5: Prepare all the referenced associated model objects, and prepare all the referenced action and link objects in specific behaviour.

When the availability start-up is omitted, there is no start-up effect, i.e. the availability start-up is inhibited.

NOTE – The availability start-up is a link. It is not to be confused with a constructor. It is processed only when the composite object’s preparation status becomes “ready”.

25.2 Availability Close-down

The availability close-down is a link object that always has the same link condition: when the preparation status of the composite object becomes not ready. i.e. Get Preparation Status (“this”) becomes not ready. The link effect may be used for additional destruction effect of the composite. It is either a personalised link effect provided by the author or one of the following default link effects:

- *Automatic Close-down 1:*
 - destruction of referenced link and action objects in the specific behaviour;
 - destruction of referenced action objects describing link effect in one of the link objects;
 - destruction of nested action objects referenced in action objects included or referenced in composition behaviour;
 - when completed, destruction of the availability close-down link itself.
- *Automatic Close-down 2:*
 - destruction of referenced associated components within the composition elements;
 - when completed, destruction of the availability close-down link itself.
- *Automatic Close-down 3:*
 - destruction of referenced link and action objects in the specific behaviour;
 - destruction of referenced action objects describing link effect in one of the link objects;
 - destruction of nested action objects referenced in action objects included or referenced in composition behaviour;
 - destruction of referenced associated components within the composition elements;
 - when completed, destruction of the availability close-down link itself.

When the availability close-down is omitted, there is no close-down effect, i.e. the availability close-down is inhibited.

NOTE – The availability close-down is a link. It is not to be confused with a destructor. It is processed only when the composite object’s preparation status becomes “not ready”.

25.3 RT-Availability Start-up

The rt-availability start-up is a link object that always has the same link condition: when an rt-composite is created from this composite, i.e. Get Rt-Availability Status (“this”:?) becomes available. The link effect may be used for additional preparation effect of the rt-composite. It is either a personalised link effect provided by the author or the following default link effect:

- *Automatic Rt-Start-up:* Run action is targeted to this rt-composite.

When the rt-availability start-up is omitted, there is no start-up effect, i.e. the rt-availability start-up is inhibited.

NOTE – The availability start-up is a link. It is not to be confused with a constructor of an rt-component. It is processed only when the rt-composite object’s Rt-Availability Status becomes “available” to each rt-composite created from this composite object.

25.4 Rt-Availability Close-down

The rt-availability close-down is a link object that always has the same link condition: when an rt-composite created from this composite becomes deleted, i.e. Get Rt-Availability Status (“this”:?) becomes not available. The link effect may be used for additional destruction effect of the rt-composite. It is always a personalised link effect provided by the author. When the rt-availability close-down is omitted, there is no close-down effect, i.e. the rt-availability close-down is inhibited.

NOTE – The availability close-down is a link. It is not to be confused with a distracter of an rt-component. It is processed only when the rt-composite object’s rt-availability status becomes “not available” to each rt-composite created from this composite object.

25.5 Composition Element

Each element is composed of the following:

- 1) A composition **element index** – Each element has a unique identification within a composite object.
- 2) A **associated model** – The associated model may be used to associate information to be presented to the user. The model is either a label, a content, a composite, or an empty model:
 - a) An label is a generic string – A label may be presented to the user as an interaction item following the style applied to the rt-composite. For example, a label may correspond to a title of a menu or push button.
 - b) A content represents a terminal element that is either a content object or a multiplexed content object.
 - c) A composite allows the build-up of a parent-child relationship through successive generations. It is the author’s responsibility to avoid the appearance of cycle in the composite object, i.e. an element addressing an ascendant.
 - d) An empty model – No model is associated with the element.

When an rt-composite is created from the composite object, the MHEG engine creates rt-objects from the associated models described in the composition (an rt-content from an associated label or an associated content, an rt-composite from an associated composite, a “Null-Root-Rt” from an empty associated model). These rt-objects are plugged into the corresponding sockets within the rt-composite.

25.6 Composition example

Figure 27 shows an example of a menu using the root rt-composite 700:8 in Figure 20 created from composite 700 in Figure 18. Composite 700 defines a logical structure of composition independently of its presentation. In this example, a style menu has been applied to 700:8 in order to represent different generations of menus and submenus. The following applies:

- The rt-content sockets: “FILE”, “OPTION”, “HELP” are presented in the principal menu.
- The rt-composite sockets 700:8.2 and 700:8.4 represents the submenus.
- The rt-content sockets: “open”, “close” and “new” are presented when the first menu is activated.
- The rt-content socket: “easy” is presented when the second menu is activated.

The presentation of this menu is not defined by this Recommendation; it is part of the look and feel of the presentation system.

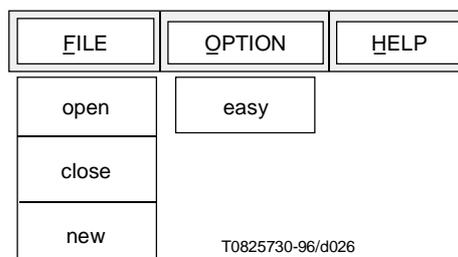


Figure 27/T.171 – Example of menu structure with submenus

26 Structure of Container Class

This clause describes the support provided for regrouping other MHEG objects. This regroupment is intended to facilitate the interchange in order to obtain a unique set of interchange. Once interchanged, it is ensured that all the included objects have also been interchanged.

This may also be used by the object designer to regroup information participating in a same application.

The container object provides the following facilities:

- an optional predefined behaviour composed of a Container Start-up (see 26.1) and Container Close-down (see 26.2);
- a list of Container Elements (see 26.3)

26.1 Container Start-up

The container start-up is a link object that always has the same link condition: when the preparation status of the container object becomes ready, i.e. Get Preparation Status (“this”) becomes ready. The link effect may be used for additional preparation effect of the container. It is either a personalised link effect provided by the author or one of the following default link effects:

- Automatic Container Start-up 1: Prepare action is targeted to the included objects.
- Automatic Container Start-up 2: Prepare action is targeted to the included and referenced objects.
- Automatic Container Start-up 3: Activate link objects included.
- Automatic Container Start-up 4: Activate link objects included and referenced.
- Automatic Container Start-up 5: Prepare action is targeted to the included objects; Activate link objects included.
- Automatic Container Start-up 6: Prepare action is targeted to the included objects; Activate link objects included and referenced.
- Automatic Container Start-up 7: Prepare action is targeted to the included and referenced objects; Activate link objects included.
- Automatic Container Start-up 8: Prepare action is targeted to the included and referenced objects; Activate link objects included and referenced.

When the container start-up is omitted, there is no start-up effect, i.e. the container start-up is inhibited.

NOTE – The container start-up is a link. It is not to be confused with a constructor. It is processed only when the container object’s preparation status becomes ready.

26.2 Container Close-down

The container close-down is a link object that always has the same link condition: when the preparation status of the container object becomes not ready, i.e. Get Preparation Status (“this”) = not ready. The link effect may be used for additional destruction effect of the container. It is either a personalised link effect provided by the author or one of the following default link effects:

- Automatic Container Close-down;
- Destruction of referenced objects;
- when completed, destruction of the availability close-down link itself.

When the container close-down is omitted, there is no close-down effect, i.e. the container close-down is inhibited.

NOTE – The availability close-down is a link. It is not to be confused with a distracter. It is processed only when the container object’s preparation status becomes not ready.

26.3 Container Element

This is an ordered list of container elements. Each container element can be an instance of any other MHEG classes that may be included or referenced to. It may be one of the following:

- action object;
- link object;

- script object;
- content object;
- multiplexed content object;
- composite object;
- container object;
- descriptor object.

A container element index is assigned to each container element. Each element has a unique identification within a container: the elements are all indexed from 1 to n by the MHEG engine.

27 Structure of Descriptor Class

This clause describes the support provided for the description of other objects that are interchanged. This description is intended to facilitate the negotiation, installation, operation and management of applications that use MHEG objects.

It is recognised that different presentation systems may have widely differing capabilities. By interchanging and interpreting descriptor objects, it is possible to adapt the resources of the presentation system to the requirements of the described objects. The descriptor objects allow MHEG engines to determine whether or not they are able to proceed with a presentation and allow using applications to determine the resources that would be needed for a presentation of a set of objects.

This Recommendation does not require that all objects that are interchanged be the subject of a descriptor object. If no descriptor is associated with an object, it is for the using application to define the description of the object. It is the responsibility of the author to ensure consistency between the descriptor objects and the described objects. This Recommendation does not provide mechanisms to handle errors or inconsistencies in descriptor objects.

A number of facilities ranging from the informal to the formal may be provided by the descriptor object:

- set of related objects (see 27.1);
- other descriptor (see 27.2);
- readme (see 27.3);
- system readable material (see 27.4);
- set of channel informations (see 27.5);
- set of catalogued style informations (see 27.6);
- set of catalogued extended elementary action informations (see 27.7);
- set of Cat Ext Attribute Info (see 27.8).

27.1 Related Object

A set of related objects may be provided in order to specify the scope of the descriptor object. If this set is omitted, the scope of the descriptor object is defined by the using application. In this case, in general, the scope should include all objects belonging to the application. Each related object attribute provides information about a single related object; it is specified as a list of the following attributes:

- Object reference: Identifies the related object to which this object information applies.
- Object information: Optional. It defines more precisely the related object with the following attributes:
 - Object size: Indicates the size of the encoded related object in octets (optional).
 - Class ID: Identifies the class of the related object.
 - A set of class-specific attributes: This is a choice of attributes sets, which carry object specific information on the related object:
 - specific attributes on a related script object (see 27.1.1);
 - specific attributes on a related content object (see 27.1.2);
 - specific attributes on a related multiplexed content object (see 27.1.3).
 - Offset: It provides a relative position in octets of an encoded object that is included in another encoded object (see 27.1.5).

This Recommendation does not define how to handle contradictory descriptions, if a given related object is described more than once.

27.1.1 Script Class Information

This provides a set of attributes describing the related script object. The following information may be provided:

- Script classification: Informs the executing system about the classification of the related script object (see 15.1).
- Script hook: Informs the executing system about the script languages used by the related script object. The executing system may use this information to prepare the corresponding script interpreter or to reject this script object if this script interpreter is not available (see clause 14).

27.1.2 Content Class Information

This provides a set of attributes describing the related content object. The following information may be provided:

- Content classification: Informs the executing system about the classification of the related content object (see 15.1).
- Content hook: Informs the executing system about the encoding used for the data of the related content. The executing system may use this information to prepare the corresponding decoder (see clause 14).
- Alternative objects: Provides a facility to indicate alternative content objects that may be used instead of this related content (see 27.1.4).

27.1.3 Mux Content Class Info

This provides a set of attributes describing the related multiplexed content object. The following information may be provided:

- Content class information: Informs the executing system about the whole multiplexed data of the related multiplexed content object, i.e. classification, hook, alternative objects (see 27.1.2).
- Number of streams: Reflects the number of streams contained within the multiplexed data of the related object.
- Set of stream information: Each stream information is composed of the following attributes:
 - Stream identifier: Identifies a stream within the related multiplexed content object.
 - Content class information: A single stream can be regarded as a single content object; therefore, the attribute set that describes a related content is used, i.e. classification, hook, alternative objects (see 27.1.2).

27.1.4 Alternative Object

This provides information about possible alternative content objects that may replace the related object.

Alternative content objects may differ in the encoding or in any other attributes from the related object. The using application may use this facility to adapt the data to be presented to the local environment. It is up to the author of the descriptor to ensure the integrity of alternative objects in the context of the related object.

The alternative objects are described as a list of alternative objects. Each alternative object is composed of the following attributes:

- A reference to an alternative content object.
- An alternative hook: Specifying the hook of the alternative object.
- An alternative descriptor object: Reference to the descriptor object describing the alternative object. It is optional.
- An alternative readme: Optional and specified as a free format string. It may be used by the using application to identify the nature of the alternative object. For example, in a multi-language application, the readme may carry an identification of the language of the alternative object. It is up to the using application to interpret the meaning of the readme.

27.1.5 Offset

An encoded MHEG object may be included within another encoded MHEG object. In order to retrieve this included MHEG object without decoding from the beginning, offset information may be provided. The offset indicates the first octet of the included MHEG object in level-D representation.

27.2 Other Descriptor

A set of references to other descriptor objects may be provided and used by the author to connect different descriptor objects. The connected descriptor objects may describe different objects, and the scope of the descriptors may differ. This Recommendation does not define how to handle contradictory descriptions if the scope of two or more descriptors overlap.

This facility provides a mechanism to create a structured set of descriptors. Possible structures are linked lists or trees. It is in the responsibility of the author to define a useful structure.

27.3 Readme

This informal text provides for a human being with the information and encouragement needed to proceed with the installation and operation of the set of related MHEG objects. The choice of words is left to the author, who may use the readme to point to further sources of information and support.

27.4 System Readable Material

A using application may place information in the descriptor object which it finds useful. This Recommendation does not define either this information or the encoding used by the application. An author or a using application may include information in the descriptor intended for the MHEG engine. However, the author should be aware that this information may not be portable to other MHEG systems.

27.5 Channel Information

A set of channel information may be provided by the author in order for the MHEG engine to map more easily each channel on the real environment. Each channel may be described as follows:

- Channel ID: Identifies the channel to be described.
- x-min, x-max, y-min, y-max, z-min, z-max: Describe for each spatial axis the minimum and maximum values required for the device associated with this channel.
- x-resolution, y-resolution, z-resolution: Describe for each spatial axis the resolution for the device associated with this channel. This is the number of addressable spatial physical units.
- t-resolution: Describes the temporal resolution for the device associated with this channel. This is the number of addressable temporal physical units within one second.
- f-min, f-max: Define the minimum and maximum presentable frequency required for that channel if audible media are presented on this channel. The units are “hertz”.
- Audio-dynamic: Defines the audio dynamic required for that channel if audible media are presented on this channel. The units are “dB”.
- Media types: Specifies the types of the media of the rt-components assigned on this channel. This allows the MHEG engine to assign channels to physical devices. It is one of the catalogued values that can be either registered or proprietary. However, if the presentation system does not support the requested device, the engine may ignore the media type.
- Events mapping: Specifies the expected events that may occur on this channel or on the rt-components assigned to this channel. It is also possible to assign a catalogued event to each event. This allows an MHEG engine to map the event defined by the author to a real event that may occur in its environment. The catalogued event can be either registered or proprietary. If no mapping on a catalogued event is provided, it is considered that the event identifier provided by the author directly map on catalogued event identifier. If this is not the case, it is for the using application to define how to map the event identifier on a real event. This Recommendation does not define how to handle contradictory descriptions if a given event identifier is mapped more than once.

This Recommendation does not define how to handle contradictory descriptions if a given channel is described more than once.

27.6 Catalogued Style Information

A set of catalogued styles may be provided in order to inform the executing system about the catalogued styles used in the Set Style actions interchanged in related action objects. The catalogued style can be either registered or proprietary.

27.7 Cat Ext elementary action info

A set of catalogued extended elementary actions may be provided in order to inform the executing system about the catalogued extended elementary actions used in the Catalogued elementary action interchanged in related action objects. The catalogued extended elementary actions can be either registered or proprietary.

27.8 Cat Ext Attribute Info

A set of Cat Ext Attributes may be provided in order to inform the executing system about the Cat Ext Attributes used in the Set Catalogued Attribute actions interchanged in related action and link objects. The Cat Ext Attributes can be either registered or proprietary.

SECTION 4 – MHEG ENTITIES COMMON BEHAVIOUR

This Recommendation defines the expected behaviour and the initial expected behaviour of the MHEG objects, rt-objects and channels through behaviour attributes and their status values. This Recommendation also defines actions to change and retrieve each expected behaviour. The processing effects of these actions that are perceptible to the user are called “user effects”, and the processing effects whose results are only available to the MHEG engine are called “MHEG effect”. Only the MHEG effect is defined by this Recommendation. The user effect is perceived by the user via a certain GUI that decides look and feel of MHEG entities.

28 MHEG entity behaviour

The initial behaviour of each MHEG entity provided by this Recommendation may be changed by the interchanging the following instances of MHEG classes:

- Action class used to interchange a set of actions with target references to MHEG objects, rt-objects or channels;
- Link class used to interchange a set of actions and to specify conditions for those actions;
- Script class used to interchange scripts that describe complex behaviours.

29 MHEG entity state definition

The complete life of an MHEG object, an rt-object or a channel is defined by the following series of diagrams. As some actions are only valid for a certain period of an MHEG entity, the definition of MHEG entity life is used to verify the validity of an action. Two description methods are used as follows:

- 1) Timing diagrams describe the principal behaviour along with the time as shown in Figure 28. This figure shows a status change of a certain MHEG entity. First, this is in period i. After targeting a certain action to this entity, there is some MHEG effect caused by an MHEG engine or a GUI, but still the entity remains in period i. This means that a Get action targeted to the entity during the processing of an MHEG effect always retrieves the value that was attached to the period (the value in period i). After finishing the processing of the MHEG effect, the entity enters in period j. If the period j has some user effect, the appearance of the entity is changed and perceived by a user. The duration of MHEG effect and possible user effect are shown as in Figure 28.
- 2) Finite state diagrams describe the possible state transitions depending on actions as shown in Figure 29. In this figure, two ellipses (State1, State2) represent states, and 3 boxes (action x, action y, action z) stand for actions that may modify states. The initial status is State1. If action x is targeted, the state remains State1. If action y is targeted after that, the state is changed to State2. And action z causes a retransition to State1. Action x and y are not valid and ignored if the state is State2, and action z is not valid and ignored if the state is State1.

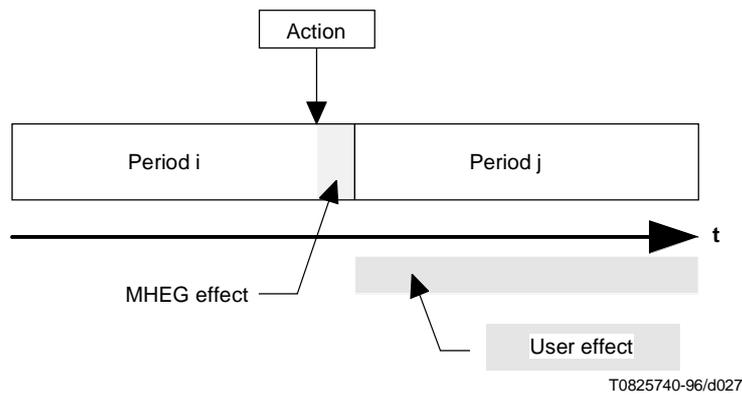


Figure 28/T.171 – Example of timing diagram

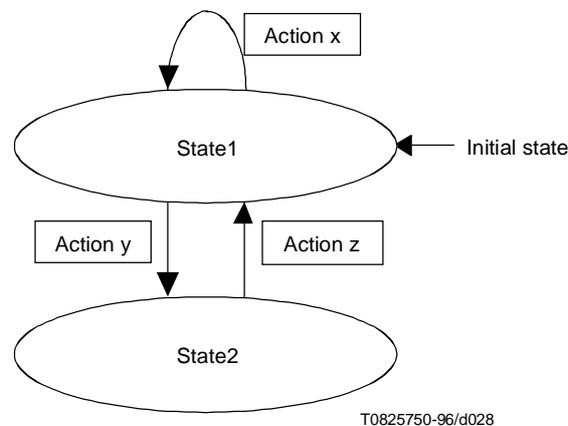


Figure 29/T.171 – Example of finite state diagrams

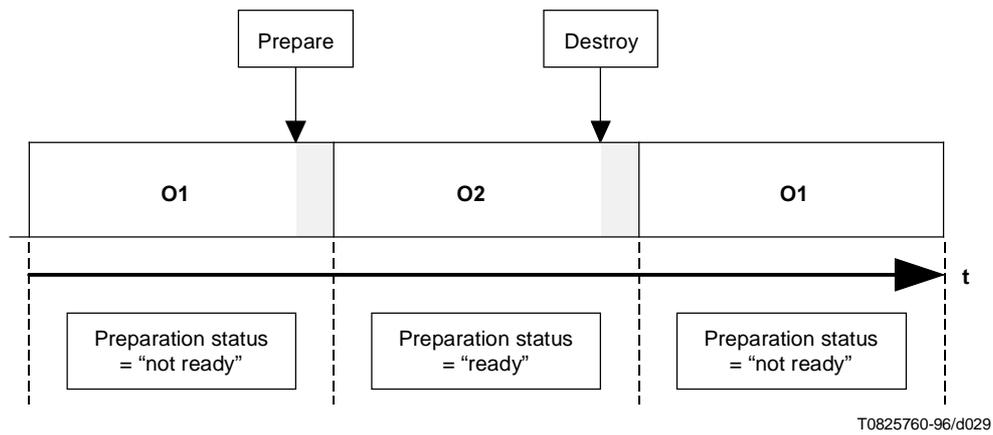
Several periods of MHEG entities are defined for the following behaviours:

- MHEG object availability (see 29.1);
- Link object activation (see 29.2);
- Channel availability (see 29.3);
- RT-object availability (see 29.4);
- Rt-Component running behaviour (see 29.5);
- Rt-Component presentation behaviour (see 29.6).

29.1 MHEG object availability

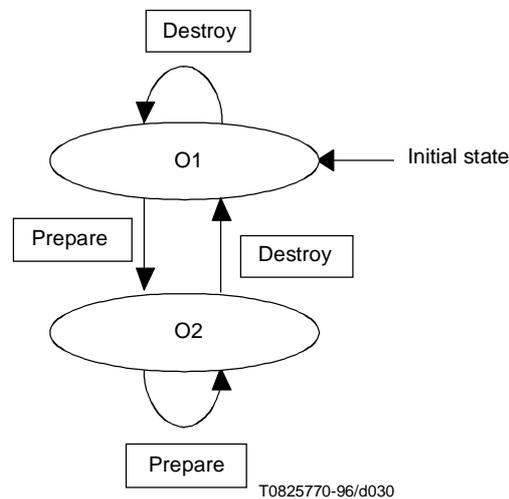
Figures 30 and 31 show MHEG object availability. Two states are defined as follows:

- **O1:** The MHEG object is not known to the MHEG engine. Even if it is not created, the MHEG object is said to be in period O1. During O1, all MHEG objects except link objects have only one attribute: preparation status which is equal to “not ready”. Link objects additionally have the activation status which is equal to “inactive”. Any other object attributes described in this Recommendation do not exist. Any Get actions evaluate “undefined” to these non-existing attributes.
- **O2:** The MHEG object is available to the MHEG engine. The preparation status of the MHEG object is “ready”. The object has been decoded. All other object attributes have been created and set to their corresponding interchanged values.



T0825760-96/d029

Figure 30/T.171 – Timing diagram of MHEG object availability



T0825770-96/d030

Figure 31/T.171 – Finite state diagram of MHEG object availability

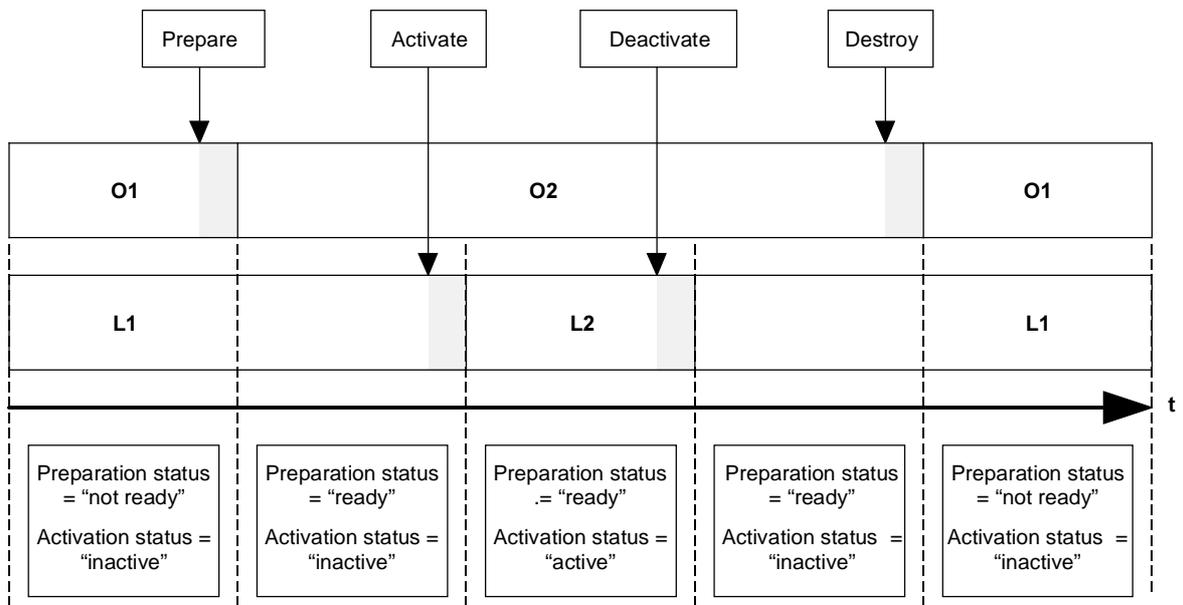
29.2 Link activation

Figures 32 and 33 show link activation. Link objects have additionally MHEG object availability states because they are also MHEG objects. The two figures also show the relationship between MHEG object availability and link activation. Two states are defined for link activation as follows:

- **L1:** The activation status of the link object is “inactive”. This link cannot be fired.
- **L2:** The activation status of the link object is “active”. The link object has been activated. During L2, each time the link condition becomes true, the link is fired. The Link Abort action does not affect these states.

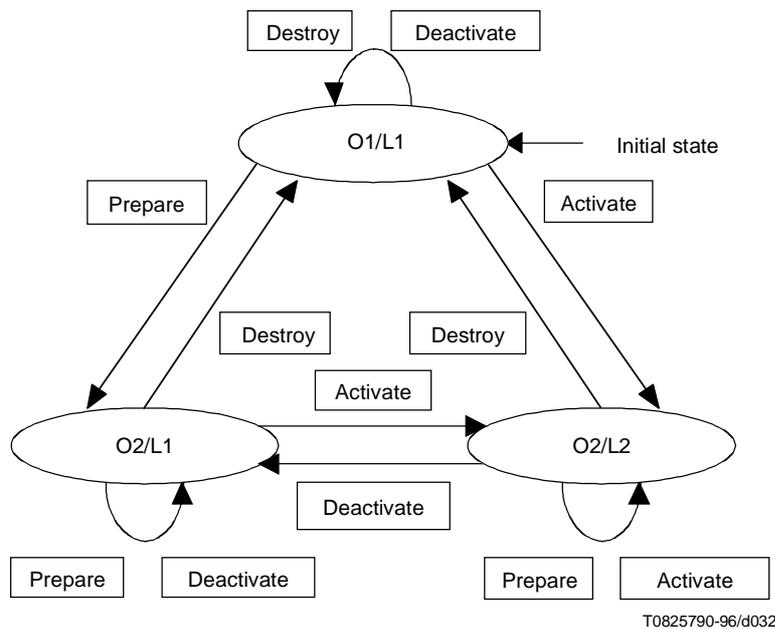
An Activate action targeted to a link whose preparation status is “not ready” implicitly executes a Prepare action.

A Destroy action may also be targeted in L2. Then the link object enters directly in period L1/O1.



T0825780-96/d031

Figure 32/T.171 – Timing diagram of link object availability and activation



T0825790-96/d032

Figure 33/T.171 – Finite state diagram of link object availability and activation

29.3 Channel availability

Figures 34 and 35 show channel availability. Two states are defined as follows:

- **C1:** The channel is not yet created, and not available to the MHEG engine.
- **C2:** The channel is available to the MHEG engine.

The default channel can neither be created nor be deleted. It is always available to the MHEG engine (i.e. the default channel is always in state C2). Both New channel and Delete channel are ignored for the default channel.

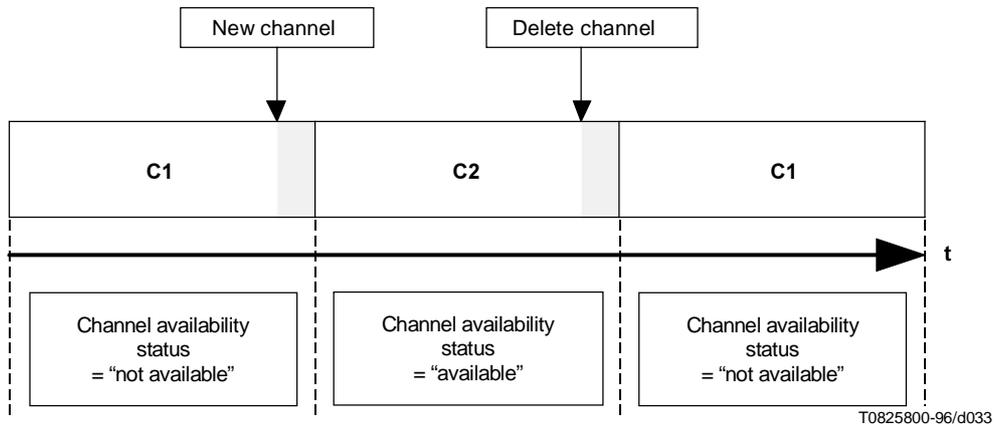


Figure 34/T.171 – Timing diagram of channel availability

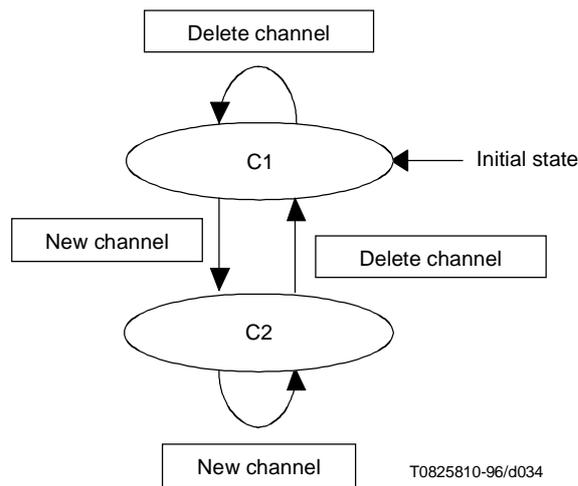


Figure 35/T.171 – Finite state diagram of channel availability

29.4 Rt-object availability

Figures 36 and 37 show rt-object availability. They also show the relationship between rt-object availability status and MHEG object availability status belonging to the model object from which the rt-object is derived. Two states are defined as follows:

- **R1:** The model object is prepared (O2), the rt-object is not created, it is not available.
- **R2:** The rt-object is available to the MHEG engine. During this period the rt-object may be presented, using a Run action.

A New action in O1/R1 prepares the model object implicitly. And a Destroy action in O2/R2 deletes all rt-objects created from this model implicitly.

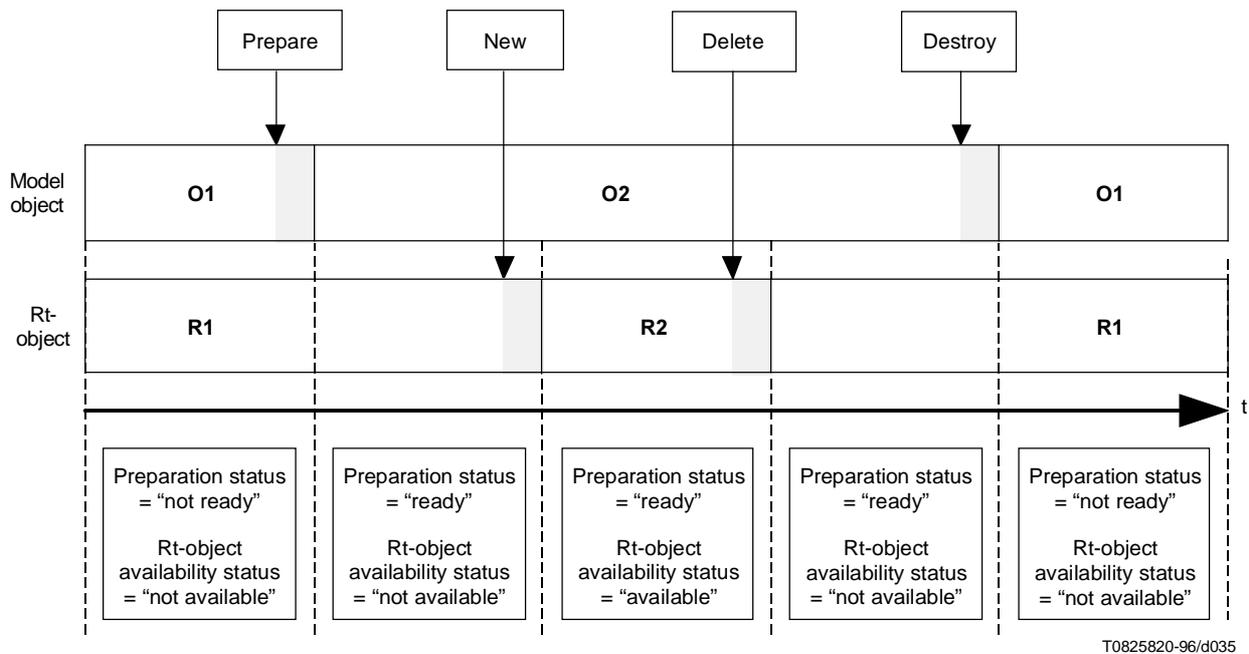


Figure 36/T.171 – Timing diagram of rt-object availability

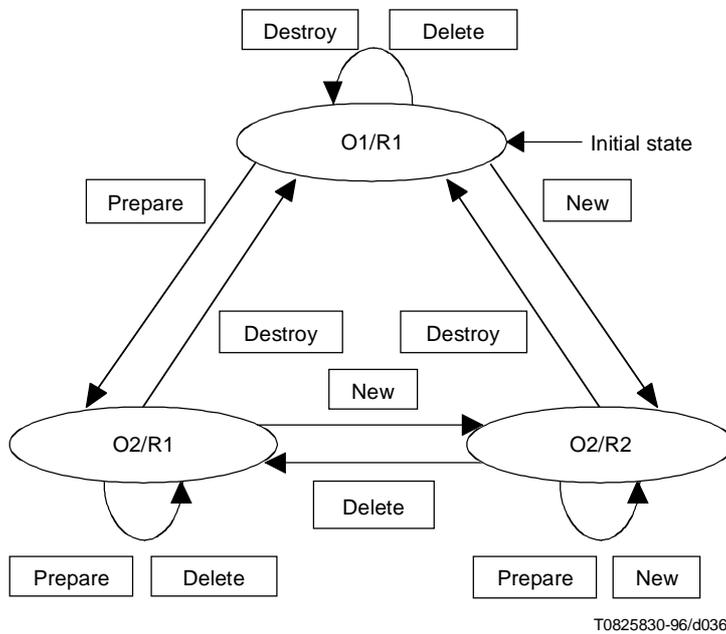


Figure 37/T.171 – Timing diagram of rt-object availability

29.5 Rt-Component running behaviour

Figures 38 and 39 show rt-component running behaviour. Once an rt-component is “available” (i.e. in R2), it is ready for presentation. Two states are defined as follows:

- **R2:** The rt-component is not running. It is not presented to the user.
- **R3:** The rt-component is running. It may be presented to the user. During this period, there may be a user effect, which is the result of the running of the rt-component to the user with the specified behaviour attributes. The user effect may be dependent of each MHEG engine and GUI.

If a Run action is targeted to a non-existing rt-component (i.e. the rt-object is in R1), an implicit New action is targeted to this rt-component before processing the Run.

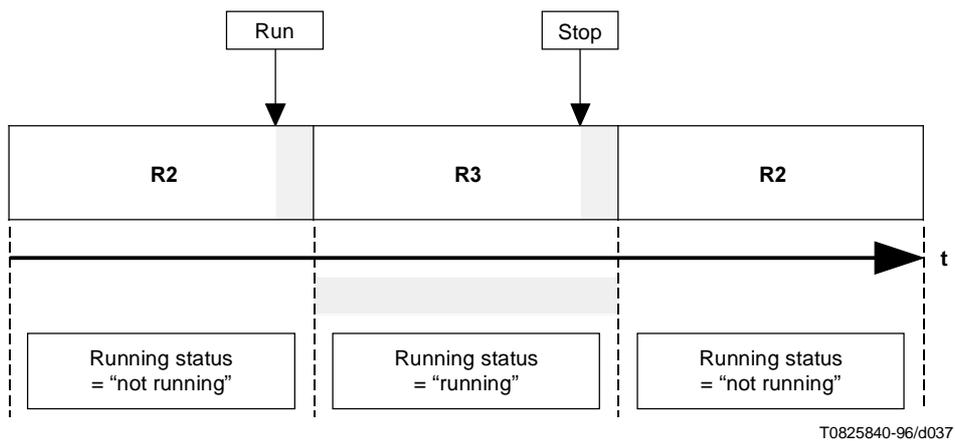


Figure 38/T.171 – Timing diagram of rt-component that is “running”

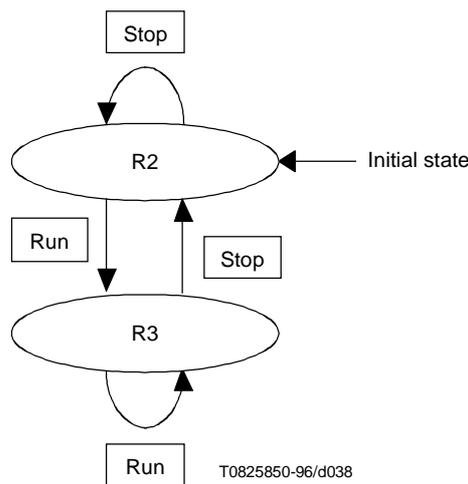


Figure 39/T.171 – Running behaviour finite state diagram

29.6 Rt-Component presentation behaviour

While an rt-component is running (i.e. the rt-component is in R3), presentation behaviour attributes may be changed by targeting certain Set actions.

Figure 40 shows a typical example of a presentation behaviour attribute. In this figure, a presentation behaviour attribute called “A” associated with a certain rt-component is changed from value V0 to V1 during R3 of the rt-component. A user effect may be associated with this attribute value changing. After stopping this rt-component, the attribute A remains as V1 independent of the running status and any other attributes.

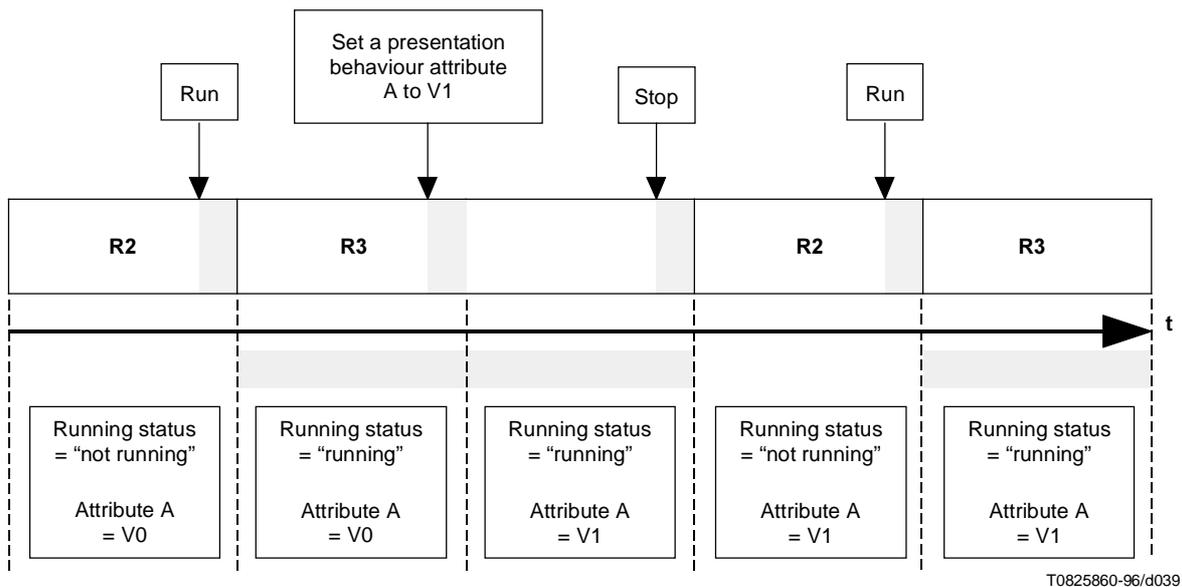


Figure 40/T.171 – Presentation behaviour attributes changing

Figure 41 shows a presentation behaviour attribute changing with a certain transition duration. When an rt-component is running, a certain Set action is targeted to it in order to change the presentation behaviour attribute A gradually. In this case, one additional state period R3.TD is defined as follows:

- R3.TD:** The rt-component running status is “running”. The MHEG engine is processing the attribute changing with the transition duration caused by a certain Set action. The duration of R3.TD is equal to the transition duration specified in the Set action. During R3.TD, the value of the specified attribute is changing gradually. When such an action is in a serial synchronised group of actions, the following action in the list is to be processed after the completion of R3.TD. A user effect may be associated with R3.TD. It shows the impact of the transition duration on the attribute value perception. The user may gradually perceive the change of the presentation behaviour attribute value from the previous value to the specified value. The user effect may be dependent of each MHEG engine and GUI.

The MHEG effect of a Set action with some transition duration is composed of two parts:

- the initial processing of the Set action in R3; and
- the consecutive processing of the attribute changing gradually in R3.TD.

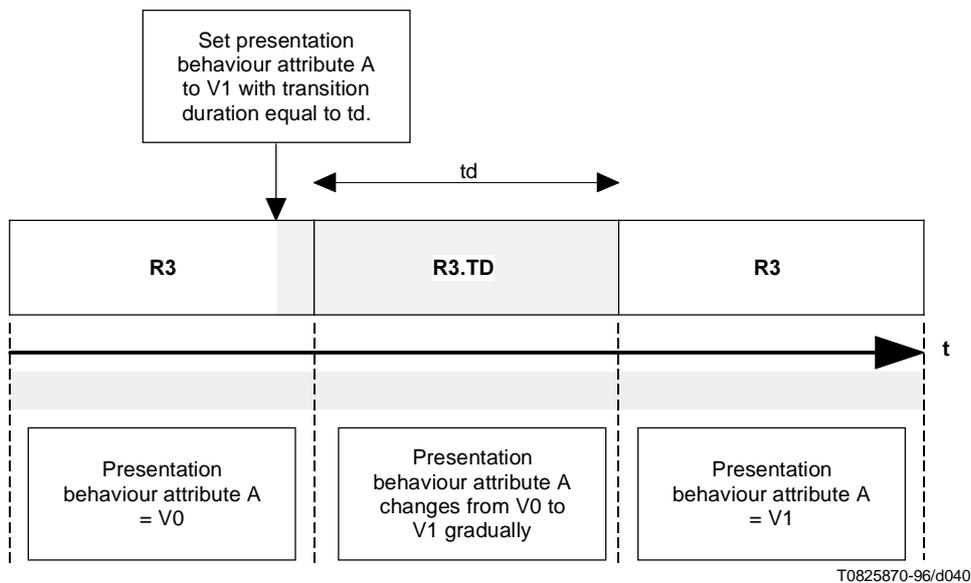


Figure 41/T.171 – Timing diagram of Set actions with transition duration

When an MHEG effect of a transition duration is under process and a Stop action is issued on this rt-component, the MHEG effect of the transition duration is terminated after the MHEG effect of the Stop action is completed. The presentation behaviour attribute value takes the specified value in the Set action as if no transition duration was specified. Figure 42 shows this situation.

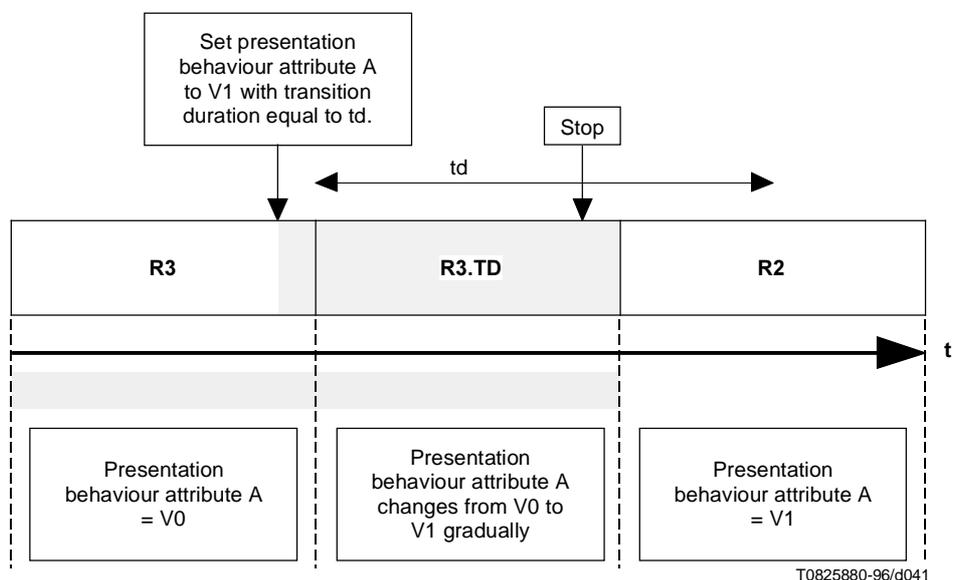


Figure 42/T.171 – Timing diagram of a Set action with transition duration when a Stop action occurs

When an MHEG effect of a transition duration is under process and a Set action is issued on the same attribute, the MHEG effect of the transition duration is terminated. The presentation behaviour attribute value takes the value specified by the last Set action. If the last Set action has no transition duration, the MHEG effect is processed as described in Figure 40. If it has transition duration, the MHEG effect is processed as described in Figure 41. Figure 43 shows an example of the former case.

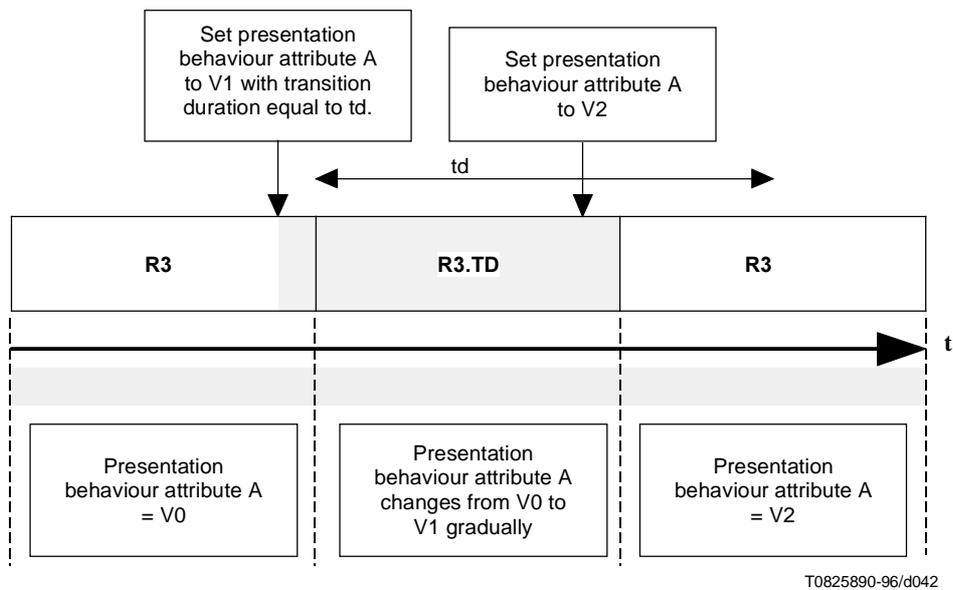


Figure 43/T.171 – Timing diagram of an interrupted set action with transition duration

Figure 44 shows presentation behaviour attribute changing while an rt-component is “not running”. The rt-component has a presentation behaviour attribute A equal to V0 while it is in R2. When a Set action to change the presentation behaviour attribute A is targeted to this rt-component, the MHEG engine processes the MHEG effect of this Set action. If a transition duration is specified, it is set to 0 meaning that the transition duration is ignored. After processing the MHEG effect, the presentation behaviour attribute A is equal to V1. If a Run action targeted after that, the rt-component is presented with the presentation behaviour attribute A equal to V1.

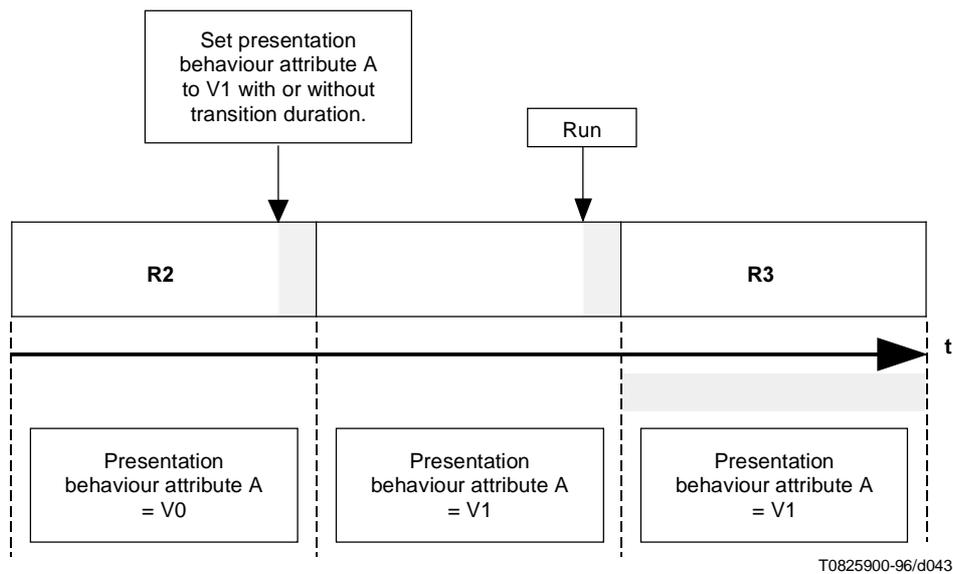


Figure 44/T.171 – Timing diagram of set actions on an rt-component that is “not running”

30 Life cycle of MHEG entities

MHEG entities are created and destroyed by using elementary actions. The period between those two actions makes the life of an MHEG entity. During the life, some attributes associated with the MHEG entity are used.

The principal rules between an MHEG entity life and associated attributes are as follows:

- Even if such an MHEG entity does not exist, preparation status (see 29.1), rt-availability status (see 29.4) or channel availability status (see 29.3) exists and is always available. The preparation status is set to “not ready”, the rt-availability status is set to “not ready”, and the channel availability status is set to “not available”.
- If a Prepare action (see 29.1), a New action (29.4) or a New Channel action (see 29.3) is targeted, the MHEG engine is required to instantiate the specified MHEG entity with its corresponding attributes which are described in different behaviours. The presentation status is set to “ready”, the rt-availability status is set to “ready” and the channel availability status is set to “available”.
- If a Destroy action (see 29.1), a Delete action (see 29.4) or a Delete Channel action (see 29.3) is targeted, the MHEG engine is required to remove the specified MHEG entity with its corresponding associated attributes except for the preparation status, rt-availability status and channel availability status that are set to “not ready”, “not ready” and “not available”, respectively.

Therefore, if an MHEG entity is destroyed or deleted, only the preparation status, the rt-availability status or the channel availability status is changed. And if there is a link whose condition is “when the preparation status becomes “not ready””, “when the rt-availability status becomes “not ready”” or “when the channel availability status becomes “not available””, it shall be triggered.

On the other hand, other associated attributes with the destroyed or deleted MHEG entity are completely removed without changing their status values. For example, if there is an rt-component which running status is “running”, its running status is removed without changing value. That means the rt-component has been deleted while it was “running”. The running status is no more available after deletion. If there is a link which condition is “when the running status becomes “not running””, it shall not be triggered at the time of the deletion.

31 General action mechanisms

This clause summarises the general action mechanisms and its treatment to be processed by the MHEG engine. This mechanism is applied for all other behaviours processing as well as common behaviours processing.

31.1 Action treatment

Actions are to be processed satisfying following conditions:

- If an MHEG effect of an action contains a list of subeffects, these subeffects are to be performed by the MHEG engine as if they were a single uninterruptable and indivisible operation.
- The effect of processing an action on a set of two or more targets is to propagate the action to each target within the set of targets. These actions are processed in parallel. The MHEG effect of the action ends when the MHEG effects of all propagated actions end.
- In certain cases, in order to process an action, a proceeding action is implicitly executed. This proceeding action may produce a change of status. A link triggered on this change of status is fired before the processing of the specified action ends.

For example, when an rt-object is made from a model object which is not prepared, this Recommendation specifies that an implicit Prepare action is to be targeted to the model object before the New action. A link whose condition is “when a preparation status of the model becomes ready” is to be triggered when the model object enters in O2, and not when the rt-object enters R2.

31.2 Processing of link effect

When the link is fired, i.e. the link is activated and the link condition is satisfied, the MHEG engine processes the link effect. Each time the link is fired, the link effect is processed in two phases as follows:

- The macro resolution phase: If some macros are in this link (see clause 13).
- The action process phase: It consists on processing the action object included or referenced in the link effect. This action object may be a basic action, a nested action or a macro action.

The link effect is processed as summarised in Figure 45.

The action process begins when the link condition is satisfied and the macro resolution accomplished, when the action is a macro action. Each action object is a recursive construction containing a group of synchronised actions accompanied by a synchro indicator.

The synchro indicator specifies if the synchronised actions in the group are to be processed in parallel or in serial. The action group process depends on the synchro indicator. If a synchro indicator attribute is specified as a macro and if this macro is resolved as the value "undefined", the entire synchronised action set is not processed.

```
BEGIN ACTION PROCESS of action object within the link effect
  IF synchro indicator == serial
    FOR J = 1 TO number_of_synchronised_actions
      DO
        IF synchronised_action J == elementary_action
          PERFORM target resolution of elementary_action J
          PERFORM the action effect of elementary_action J on the target set
        ELSE IF synchronised_action J == action_object
          PERFORM RECURSIVE ACTION PROCESS of action_object J
        ENDIF
        WAIT FOR the end of MHEG effect of synchronised_action J
      ELSE IF synchro_indicator == parallel
        FOR J = 1 TO number_of_synchronised_actions
          DO SIMULTANEOUSLY
            IF synchronised_action J == elementary_action
              PERFORM target resolution of elementary_action J
              PERFORM the action effect of elementary_action J on the target set
            ELSE IF synchronised_action J == action_object
              PERFORM RECURSIVE ACTION PROCESS on action_object J
            ENDIF
          ENDIF
        WAIT FOR the end of MHEG effect of each action in each "synchronised actions"
        WAIT FOR the end of user effect of each action 'run' in each "synchronised actions"
        DURING that time IF a target is "running" AND
          IF its current temporal position == its terminal temporal position
            PERFORM a Stop action on this target
        END OF ACTION PROCESS of action object within the link effect
```

Figure 45/T.171 – Action process phase

31.2.1 Serial action processing

If all the elementary actions contained in a serial action group are specified to be processed, each elementary action is processed sequentially after the MHEG effect of the previous elementary action is completed.

Figure 46 shows an example. In this figure, 2 elementary actions are included in a link effect. After finishing the MHEG effect of elementary action 1, elementary action 2 is processed.

The duration of the MHEG effect for a serial action list is the sum of duration of the MHEG effects for the serial actions.

If a Run action is contained within the group, the user effect of the Run action continues even after the MHEG effects of the group are completed, except if a Stop action is performed.

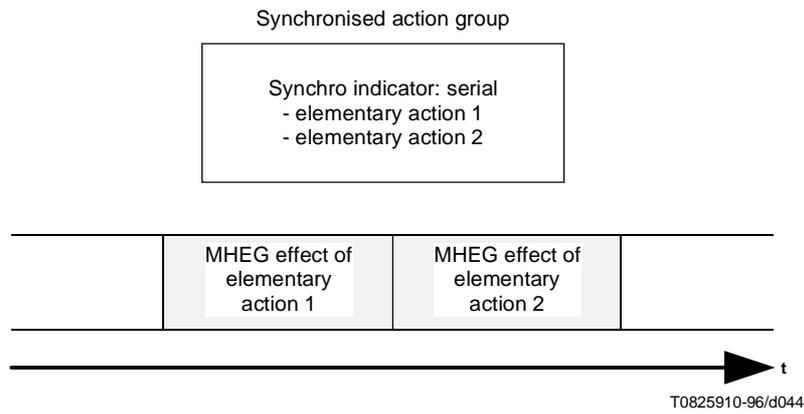


Figure 46/T.171 – Example of processing of serial actions

31.2.2 Parallel action processing

If all the elementary actions contained in a parallel action group are specified to be processed, each elementary action is processed simultaneously. The MHEG effect of the parallel action group ends when the MHEG effects of all the parallel actions end.

When a parallel group is imbricated in a serial group, the MHEG engine is required to wait for the end of the MHEG effect of the set of parallel actions before processing the next action in the serial group.

Figure 47 shows an example of parallel processing. Before processing the elementary action 4, the MHEG engine waits for the end of the MHEG effect of the action group, i.e. the end of the MHEG effect of elementary action 2.

The duration of the MHEG effect for a set of parallel actions is the longest MHEG effect among the parallel actions.

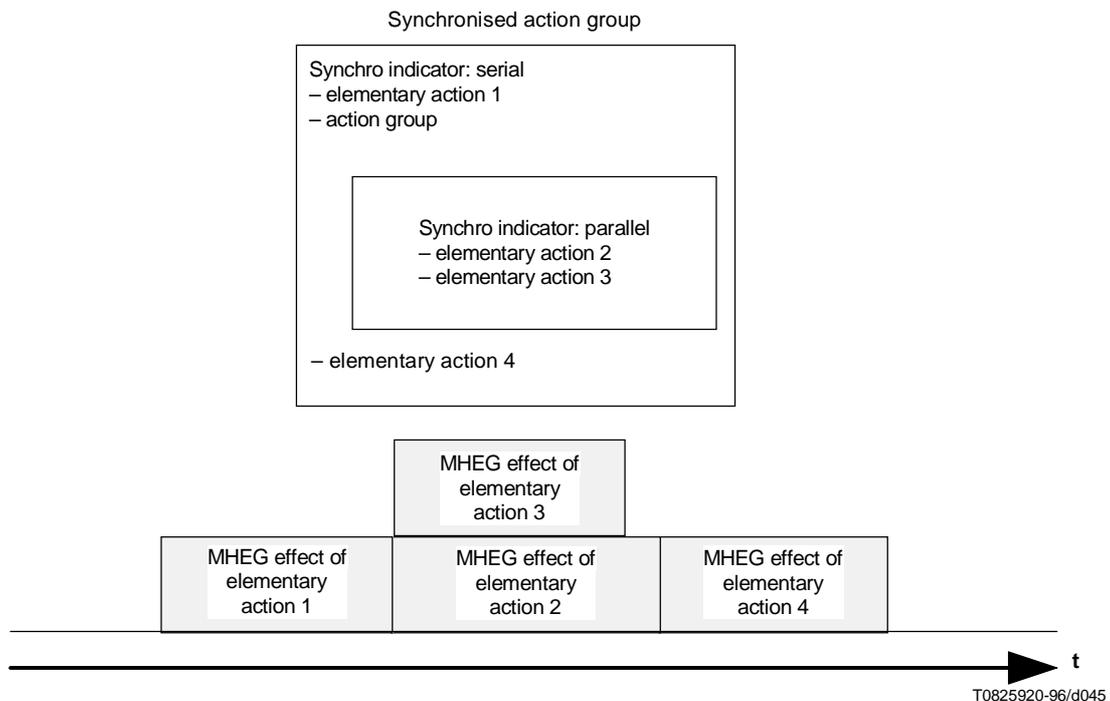


Figure 47/T.171 – Example of processing parallel actions

31.3 Basic processing of an elementary action

When the MHEG engine processes an elementary action on a single target, the following effects are obtained:

- **MHEG effect without transition duration:** This begins at the instance when the action is targeted, and it ends when the action has been taken into account by the MHEG engine. This typically results in a change of status or behaviour attribute value. See some examples in Figures 30, 32, 34, 36, 38, 40 and 44.
- **User effect without transition duration:** This effect exists only if the running status is “running”. This effect begins at the instance when the MHEG effect of the Run action ends. This user effect lasts until the MHEG effect of the Stop action ends. The user effect is perceptible by the user during the “running”. Each time a presentation attribute of the target is changed, if the target is “running”, the user effect immediately reflects this change. See some examples in Figures 38 and 40.
- **MHEG Effect with transition duration:** This begins at the instance when an action specifying a non-null transition duration is targeted to an entity. It ends when the action has been taken into account by the MHEG engine. If the target is not running, the transition duration is set to zero and ignored. See some examples in Figures 41, 42 and 43.
- **User Effect with transition duration:** This effect exists only when a target is running and a set action with a non-null transition duration is targeted to it. This effect begins at the instance when the MHEG effect of the transition duration begins. The effect of the action is perceptible to the user gradually from the previous value to the specified value. Once the MHEG effect of the transition duration ends, the user effect with transition duration ends. See examples in Figures 41, 42 and 43.

31.4 Resolution of target set

A single target (e.g. an MHEG object, a container element, an rt-object, a socket element, a channel) or multiple targets (e.g. some targets, an alias addressing several entities, all the rt-objects made from a model object, all the children of a specified entity) may be specified for each elementary action. Multiple targets are specified as a list of targets. The action effect for each target within a list should be processed in parallel.

NOTES

- 1 – Authors should not rely on the order in which the parallel actions are executed.
- 2 – An MHEG engine is not required to perform true parallel processing.

Processing an action on an undefined target has no effect.

31.5 Arithmetic precision

Percentage and ratio may be specified in some of elementary actions’ parameters, for example, Set GTF, Set OVD. And ratio is also used in mapping OPS to RPS.

The general rule to calculate percentage and ratio is that full precision operation should be taken up to the final presentation of objects. Rounding should be done only for the final presentation of objects, i.e. in the final chain of OPS to CPS at channels.

NOTES

- 1 – If an rt-object with CV 17 is connected to rt-composite A with GVF 1/3 that is connected to rt-composit B with GTF 3 and rt-composite B is connected to a channel, the final volume to be mapped to a physical device is 17 (Round($17 \times 1/3 \times 3$)) and not 18 (Round(Round($17 \times 1/3$) $\times 3$)).
- 2 – In above example with changing GTF of rt-composite B to 2, the final volume is 11 (Round($17 \times 1/3 \times 2$)).
- 3 – A profile of this Recommendation or a using application may define another arithmetic precision and rounding policies.

32 Common action effects and handling

This clause describes common action effects and handling for all elementary actions and get actions applied for all behaviours throughout this Recommendation. The actual process of elementary actions and get actions is outside the scope of this Recommendation, and should be provided by the MHEG engine.

32.1 Elementary actions

Common action effects and handling to all elementary actions are as follows:

- If the MHEG engine is unable to perform the required MHEG effect of an elementary action, it is recommended, but not required, that the MHEG engine notify the using application of it.
- If the effect of targeting an elementary action to an MHEG object, an rt-object or a channel is not described in this Recommendation, this action is ignored by the MHEG engine.
- If the whole set of targets becomes no more available during the processing of an elementary action (i.e. all the targets within the set are destroyed or deleted), the process of this action is aborted.
- If a reference to a target of an elementary action except for Prepare, Activate, New and New Channel actions is invalid, the elementary action is ignored (i.e. with a null effect and completed immediately).
- If a target within a set of targets for an elementary action does not exist, this action is ignored only for that target.
- If the reference specification defines an empty set of MHEG objects, rt-objects or channels, the reference is ignored.
- If the reference is ignored, the MHEG engine behaves as if no entity is addressed.
- If a Prepare, New or New Channel action is applied to an MHEG entity which has an identifier already assigned to an MHEG entity currently available to the MHEG engine, that action is ignored.
- If an elementary action is targeted to an rt-object in period R2 with a transition duration parameter, the specified transition parameter value is ignored and the elementary action is performed as if the transition duration is set to 0.
- If an elementary action is targeted to an rt-object in period R3 with a transition duration parameter, the specified transition parameter value is interpreted by the GTU assigned for the target. The presentation of the target is being gradually changed depending on the elementary action and changing presentation attribute.
- If a transition duration parameter is specified as a negative value, it is set to 0.
- This Recommendation does not define the behaviour of the MHEG engine that is executing a sequential synchronised actions group while one or more actions fail.

If some user effect is associated with an elementary action, it is explained in the corresponding clause of the elementary action. If there is nothing about the user effect explained in the clause, no user effect is associated with the elementary action.

32.2 Get actions

Common action effects and handling to all get actions are as follows:

- If a reference to a target of a get action except for Get Preparation Status, Get Activation Status, Get Rt-availability Status and Get Channel Availability actions is invalid, the result is to be evaluated to “undefined”.
- If an MHEG engine is unable to evaluate a get action, this action is evaluated to “undefined”.
- If a reference to multiple entities (e.g. children tail, descendant tail) is specified, this reference is ignored. The target of a get action needs to address a single entity.
- If the reference is ignored, the MHEG engine behaves as if no entity is addressed.

32.3 Recommended exception handling

This Recommendation does not define any exception handling within the MHEG engine, but recommends following treatments on some situations:

- 1) If an object whose encoding does not conform to this Recommendation is provided to the MHEG engine, ignoring the whole object is recommended.
- 2) If non-existing entities are referenced except for targets of elementary actions and get actions, following are recommended depending on the invalid reference:
 - a) in a link condition, respective conditions are evaluated to “undefined”;
 - b) otherwise (e.g. in a composition element), they are replaced by null objects.

These replacement behaviours are withdrawn as soon as the referenced object comes into existence.

- 3) If an object receives an action for which the corresponding behaviour is not defined by this Recommendation, ignoring this action is recommended. For example, Delete action targeted to a content object, a Set action with an unsupported value targeted to an rt-object.

The behaviour may be defined in an extension to this Recommendation defined by another standard or a using application.

- 4) If other exceptions, e.g. request for a presentation aspect not supported by the presentation environment, or runtime failures like insufficient memory, happen, graceful degradation and sensible fall backs are encouraged.

33 Postpone Behaviour

This behaviour postpones the process of any actions. Initially, each action is processed immediately. However, using this facility, any action may be postponed.

33.1 Behaviour attributes and statuses

None.

33.2 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following action:

- Delay action (see 33.2.1).

33.2.1 Delay action

This action inserts a delay between two actions of a serial synchronised action set within an action object.

This action has the following parameters:

- Temporal Unit Ref Param;
- Duration Param.

This action may be targeted during any period.

33.2.1.1 Delay action effect

The MHEG effect is as follows:

- 1) If the temporal unit reference is specified as an rt-component, the duration of the delay is expressed in OGTU belonging to the rt-component.
When the temporal unit reference indicates an rt-content, the delay is interpreted following the rules of the OPS-RPS-CPS chain exactly like any other temporal attributes.
- 2) Otherwise (the temporal unit reference is specified as default GF), the duration of the delay is expressed in PTU.
- 3) If the Delay action is in serial synchronised actions, the MHEG engine waits during the specified duration before processing the next action.
- 4) The MHEG effect ends when the duration of the delay is expired.

33.2.1.2 Delay additional error conditions

- if the Duration Parameter parameter value is a negative value, it is set to the value 0;
- if the specified rt-component does not exist, the retrieved GTF is interpreted as the default-GF.

34 Returnability Behaviour

This behaviour returns information to a using application or external entities outside of the MHEG engine, e.g. a script engine, a GUI, a server.

34.1 Behaviour attributes and statuses

None.

34.2 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following action:

- Return action (see 34.2.1).

34.2.1 Return action

This action enables to return information to a using application or external entities outside of the MHEG engine. Generic values or information contained in content objects may be passed.

This action has the following parameters:

- Set of Return Target Param;
- Return Indicator Param;
- Set of Returned Generic Value Param;
- Set of Content Object Ref Param.

This action may be targeted during any period.

34.2.1.1 Return action effect

The MHEG effect of the Return action is as follows:

Return Indicator Param, Returned Generic Value Param and Content Object Ref Param are passed to the entity specified by the Return Target Parameters. The encoding format of the returned information is not defined by this Recommendation.

34.2.1.2 Return additional error conditions

None.

35 Alias Behaviour

This behaviour provides a symbolic identification to any generic references. Initially, no alias is given to any references.

35.1 Behaviour attributes and statuses

None.

35.2 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following actions:

- Set Alias action (see 35.2.1).

35.2.1 Set Alias action

This action enables the assignment of aliases to any generic references.

This action has the following parameters:

- Set of Target Param;
- Set of Alias Spec Param.

This action may be targeted during any period.

35.2.1.1 Set Alias action effect

This effect of this action is as follows:

- 1) according to the Update Command, one or several aliases are added, removed or replaced to a target;
- 2) if a target is not accessible at the moment of this action, the alias is assigned to the reference as if the target were accessible.

35.2.1.2 Set Alias additional error conditions

None.

36 Extensibility Behaviour

This behaviour allows the process of a catalogued elementary action, or the modification or retrieval of a catalogued attribute value (see clause 15).

The main purpose of the catalogued elementary actions is to provide the behaviour that cannot be sensibly controlled like attributes, e.g. draw line, perform some arithmetic operations or stop the MHEG engine.

On the other hand, the main purpose of the catalogued attributes is to deal with the information that can be expressed by attributes, e.g. presentation attributes like font size or channel attributes like current number of assigned rt-components.

The author should distinguish the differences.

Style behaviour is also a kind of extension that is devoted for presentation and interaction. For example, content format (encoding data) conversion is a feature of catalogued elementary actions, while presenting a content in different ways is a feature of the style behaviour (a music code playing in a graphical way or audible way).

36.1 Behaviour attributes and statuses

This behaviour is expressed using the following attribute or status:

- Catalogued Attribute (see 36.2).

36.2 Catalogued Attribute

A catalogue entry which represents the list of informations that can be, for instance, presentation attributes, font size or channel attributes. The catalogued attribute may be used as a kind of tail to navigate into the hierarchy of the catalogues.

36.3 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following actions:

- Catalogued Elementary Action action (see 36.3.1);
- Set Catalogued Attribute action (see 36.3.2).

36.3.1 Catalogued Elementary Action action

This action enables to use an extended elementary action which is registered or proprietary.

This action has the following parameters:

- Set of Target Param;
- Catalogued Extended EA Param;
- Set of Elementary Action Param.

This period is described in the catalogue where the extended EA is retrieved.

36.3.1.1 Catalogued Elementary action effect

The effect of this action is described in the catalogue where the extended elementary action is retrieved.

36.3.1.2 Catalogued Elementary Action additional error conditions

- The additional errors are also described in the catalogue where the extended elementary action is retrieved.

36.3.2 Set Catalogued Attribute action

This action enables to set a value of a catalogued attribute for the target.

This action has the following parameters:

- Set of Target Param.
- Cat Ext Attribute Param.
- Ext Attribute Value Param: A generic value to be assigned to a Cat Ext Attribute.
- Transition Duration Param: A generic value to be assigned to a Cat Ext Attribute.

This period is described in the catalogue where the extended EA is retrieved.

36.3.2.1 Set Catalogued Attribute action effect

The action effect depends on the chosen attributes and is described in the corresponding catalogue.

36.3.2.2 Set Catalogued Attribute additional error conditions

- The additional errors are also described in the catalogue where the extended elementary action is retrieved.

36.4 Actions to retrieve the behaviour

This Recommendation defines means to get this behaviour by the use of the following action:

- Get Catalogued Attribute action (see 36.4.1).

36.4.1 Get Catalogued Attribute action

This action enables to retrieve a value of a catalogued attribute for the target.

This action has the following parameters:

- Target Param.
- Cat Ext Attribute Param: A generic value to be assigned to a Cat Ext Attribute.

This period is described in the catalogue where the extended EA is retrieved.

36.4.1.1 Get Catalogued Attribute action effect

The value of the catalogued attribute is retrieved.

36.4.1.2 Get Catalogued Attribute additional error conditions

- The additional errors are also described in the catalogue where the extended elementary action is retrieved.

SECTION 5 – MHEG OBJECTS BEHAVIOUR

37 MHEG Objects Availability Behaviour

The behaviour is used to express the availability of an MHEG object to the MHEG engine. This behaviour is common for all MHEG objects.

37.1 Behaviour attributes and statuses

This behaviour is expressed using the following attribute or status:

- Preparation Status (see 37.2).

37.2 Preparation Status

The availability of each MHEG object to the MHEG engine may be evaluated. The result of this evaluation is set in the preparation status. The following applies (see Figure 31).

- 1) When an MHEG object is not available to the MHEG engine, its preparation status is evaluated to “not ready”, i.e. the MHEG object is in period O1. Initially an MHEG object is in that state.

NOTE 1 – For example, in that state, the object has not been interchanged to the receiving system, some pre-processing is required to prepare an object for presentation, etc.

- 2) Otherwise, its preparation status is evaluated to “ready”, i.e. the MHEG object is in period O2.

NOTE 2 – An MHEG engine may offer to the using application more detailed diagnostic and information messages, for example, to explain that an object cannot be located, or to indicate error conditions. These messages are not defined by this Recommendation but may be the subject of other standards.

37.3 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following actions:

- Prepare action (see 37.3.1);
- Destroy action (see 37.3.2).

37.3.1 Prepare action

This action makes the MHEG object available to the MHEG engine.

NOTES

1 – For example, retrieval of an audiovisual sequence from a disk may require so much time that it may be efficient to start loading it before it is needed.

2 – A using application or a specific profile shall define one concrete behaviour of this action. For example, considering a content object which has large data may be loaded either within the Prepare MHEG effect or outside of it, depending on the adapted strategy.

This action has the following parameter:

- Set of MH-Target Param.

This action may be targeted during the period O1.

37.3.1.1 Prepare action effect

If the preparation status of the target is “not ready”, the following applies:

- 1) Make the target available to the MHEG engine.
- 2) Process the specific effects of the Prepare action on the following targets:
 - a) Action object: Implicit Prepare actions are targeted to the embedded (included or referenced) action objects in a nested action object.
 - b) Link object: Implicit Prepare actions are targeted to the action objects (included or referenced) describing the link effect.

The preparation of a link object does not imply the preparation of any targets of the get actions and elementary actions considered in this link.

- c) Script object: The script data (included or referenced) is made available to the MHEG engine.

Object availability to an MHEG engine is not defined by this Recommendation. For example, the script data may be compiled into executable code and linked to the MHEG engine or it may be in an interpreted form requiring a separate process.

- d) Content object: The content data (included or referenced) is made available to the MHEG engine.

Object availability to an MHEG engine is not defined by this Recommendation. For example, the content data may be in the memory of the MHEG engine or available when required via a telecom or broadcast network.

- e) Composite object:

i) Composite Prepare actions are targeted to the links and action objects included in the specific behaviour.

ii) Implicit Prepare actions are targeted to the associated model objects included in the composition elements.

iii) Implicit Activate actions are targeted to the “availability start-up”, “rt-availability start-up”, “availability close-down” and “rt-availability close-down” links.

- 3) NOTE 1 – The object designer may specify an explicit preparation and activation of the link and action objects only when there is a need to include that behaviour in the MHEG engine, or in order to take into account the needs of minimal resource systems.

- 4) NOTE 2 – The object designer may specify an explicit preparation of the referenced associated model objects only when there is a need to create an rt-composite. This takes into account the needs of minimal resource systems:

- a) Container object: Implicit Activate actions are targeted to the “container start-up” and “container close-down” links.

The object designer may specify, within these links, an explicit sequencing of preparation of the objects, and so make these objects available to the MHEG engine when needed, in order to take into account the needs of minimal resource systems.

- b) Descriptor object: The descriptor information (included or referenced) is made available to the MHEG engine.

It is expected that the MHEG engine evaluates the information provided by the descriptor during the execution of the Prepare action. It depends on the implementation of the engine and the execution context how the information is handled.

- 5) Set the preparation status of the target to “ready”. The target enters in period O2. These additional effects are to be processed on the following targets:

- a) Content object: The following behaviours are available:

i) Generic value storage behaviour;

ii) Copy behaviour (see 72.9.1).

- b) Composite object: “Availability start-up” is fired, i.e. its link effect is processed.

- c) Container object: “Container start-up” is fired, i.e. its link effect is processed.

37.3.1.2 Prepare additional error conditions

- If one of the targets is an indexed element within a container object, the outer container object needs to be prepared. If it is not prepared, the action is ignored for that target.
- If one of the targets is not an MHEG object in O1, this action is ignored for that target.

37.3.2 Destroy action

This action removes an MHEG object from the MHEG engine in order to release resources (e.g. memory, bandwidth, processing power) in the MHEG engine.

This action has the following parameter:

- Set of MH-Target Param.

This action may be targeted during the period O2.

37.3.2.1 Destroy action effect

If the preparation status of the target is “ready”, the following applies:

- 1) Process the specified effects of the Destroy action on the following targets:
 - a) Action object:
 - i) Implicit Destroy actions are targeted to the embedded (included or referenced) action objects in a nested action object.

NOTE 1 – Object designers should be aware of side effects that may occur if these objects are referenced in other objects.
 - ii) If an action object to be destroyed is under processing, the processing is interrupted.

The way in which the processing is interrupted is not defined by this Recommendation. It is the responsibility of the MHEG engine.
 - iii) If a nested action object is destroyed in an action object, this nested action object is replaced by a null object.

NOTE 2 – Processing a null object has a null effect.
 - b) Link object:
 - i) If the link effect of the target is under process, an implicit Link Abort action is targeted to the link.

As a link may be fired several times during its life, several link effects may be under process at the same time. All of these link effects are concerned.
 - ii) An implicit Destroy action is targeted to each included or referenced action object.

NOTE 3 – Object designers should be aware of side effects that may occur if this action object is referenced in other objects.

NOTE 4 – The destruction of a link does not imply the destruction of any target for that link.
 - c) Model object: Implicit Delete action is targeted to all the existing rt-objects created from this model object. The rt-availability status for each rt-object is changed to “not available”. Other behaviours are destroyed.

NOTE 5 – Object designers should be aware that the rt-objects disappear if the original model object is destroyed.

NOTE 6 – It is not defined by this Recommendation how to handle the disappearance of the rt-objects. It is left to the MHEG engine.
 - d) Script object: The script data (included or referenced) is destroyed.

Object designers should be aware of side effects that may occur if this script data is referenced in another script object.
 - e) Content object: The content data (included or referenced) is destroyed.

Object designers should be aware of side effects that may occur if this content object data is referenced in another content object.
 - f) Composite object:
 - i) Implicit Destroy actions are targeted to the “availability start-up”, “rt-availability start-up” and “rt-availability close-down”;
 - ii) Implicit Destroy actions are targeted to the links and action objects included in the specific behaviour;

- iii) Implicit Destroy actions are targeted to the included component objects associated with the composition elements;
 - iv) Associated model objects and labels are also destroyed.
 - g) Container object: An implicit Destroy action is targeted to the “availability start-up” link.
- 2) Set the preparation status of the target to “not ready”. The target enters in period O1. These additional effects are to be processed on the following targets:
 - a) Link object: The activation behaviour is changed to “inactivate”.
 - b) Model object: The rt-availability behaviour is changed to “not available”.
 - c) Composite object: availability close-down link is fired, i.e. its link effect is processed. After the link effect, availability close-down link is destroyed.
 - d) Container object: Availability close-down link is fired, i.e. its link effect is processed. After the link effect, availability close-down link is destroyed.
 - 3) Destroy the target.

37.3.2.2 Destroy additional error conditions

- If one of the targets is not an MHEG object in O2, this action is ignored for that target.

37.4 Actions to retrieve the behaviour

This Recommendation defines means to get this behaviour by the use of the following actions:

- Get Preparation Status action (see 37.4.1).

37.4.1 Get Preparation Status action

This action gets the preparation status of each MHEG object.

This action has the following parameter:

- MH-Target Param.

This action may be targeted during the periods O1 and O2.

37.4.1.1 MHEG effect

This action evaluates the preparation status of the target. Following applies:

- 1) if the target is in period O1, the preparation status is evaluated to “not ready”;
- 2) otherwise (i.e. the target is in period O2), the preparation status is evaluated to “ready”.

37.4.1.2 Get Preparation Status additional error conditions

None.

38 Link Object Activation Behaviour

This behaviour controls the processing of a link object in the MHEG engine. All link objects have this behaviour. Active link objects are surveyed by the MHEG engine. Whenever a link condition becomes true, the corresponding link effect is performed.

38.1 Behaviour attributes and statuses

This behaviour is expressed using the following attribute or status:

- Activation Status (see 38.2).

38.2 Activation Status

After preparing, the activity of each link object to the MHEG engine may be evaluated. The result of this evaluation is set in the activation status.

When a link object is allowed to be fired, its activation status is set to “activate”, i.e. the link object is in period L2.

Otherwise, its activation status is set to “inactive”, i.e. the link object is in period L1.

38.3 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following actions:

- Activate action (see 38.3.1);
- Deactivate action (see 38.3.2).

38.3.1 Activate action

This action activates link objects, i.e. they may be fired.

This action has the following parameter:

- Set of Link Target Param.

This action may be targeted during the period L1.

38.3.1.1 Activate action targeted to a single link object

The MHEG effect of this action is as follows:

- If the target is in period O2/L1, the activation status of the target is set to “active”.
- If the target is in period O1, an implicit Prepare action is targeted to this target. After preparing, the activation status of the target is set to “active”.

A link object may be fired only if its preparation status is “ready” and its activation status is “active”, i.e. the link object is in period O2/L2. Once active, the link effect may be processed several times (i.e. each time the link condition becomes true) until the link object is set to “inactive”.

38.3.1.2 Activate additional error conditions

- If one of the targets is not a link object in O1 O2/L1, this action is ignored for that target.

38.3.2 Deactivate action

This action deactivates link objects, i.e. they cannot be fired.

This action has the following parameter:

- Set of Link Target Param.

This action may be targeted during the period L2.

38.3.2.1 Deactivate action effect

The MHEG effect of this action is as follows:

- If the target is in period L2, the activation status of the target is set to “inactive”.

38.3.2.2 Deactivate additional error conditions

- If one of the targets is not a link object in L2, this action is ignored for that target.

38.4 Actions to retrieve the behaviour

This Recommendation defines means to get this behaviour by the use of the following action:

- Get Activation Status action (see 38.4.1).

38.4.1 Get Activation Status action

This action retrieves the activation status value of a link object.

This action has the following parameter:

- Link Target Param.

This action may be targeted during the periods L1 and L2.

38.4.1.1 Get activation status targeted to a link object

The MHEG effect of this action is to retrieve the activation status of the target. The following applies:

- 1) if the target is in period L1, the activation status evaluates to “inactive”;
- 2) if the target is in period L2, the activation status evaluates to “active”.

38.4.1.2 Get Activation Status additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not a link object, this action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target.

39 Link Object Abort Behaviour

This behaviour aborts the processing of all the actions that are contained in a link object.

39.1 Behaviour attributes and statuses

None.

39.2 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following action:

- Link Abort action (see 39.2.1).

39.2.1 Link Abort action

This action aborts the processing of all the actions contained in a link object.

This action has the following parameter:

- Set of Link Target Param.

This action may be targeted during the period L2.

39.2.1.1 Link Abort action effect

The effect of the action is as follows:

- 1) The elementary actions currently processed within the link effect of the target are aborted. These elementary actions are directly contained in the link effect or in a nested action object. The rest of the elementary actions contained within the link effect or some nested action objects are not processed.
- 2) If the aborted elementary action is processing the MHEG effect changing an attribute value, the following applies:
 - a) If it is without a transition duration, the value is not changed.
 - b) Otherwise (with a transition duration), the value becomes the current changing value.

What is aborted is the process of the transition duration. That is why the presentation attribute value on which the changing with transition duration is under processing stops its changing at its current value. This is a different effect from a Stop action targeted during that time (see Figure 42).

- 3) If the aborted elementary action is a Run action, an implicit Stop action is targeted to the same target of the Run action.

If the aborted elementary action has a set of targets, the effect depends on the processing state of this elementary action on each target.

NOTES

1 – Suppose one Prepare action is targeted to both Object A and Object B within the link L1, and the MHEG effect of the Prepare action for Object A is already terminated (i.e. Object A is in period O2), but the MHEG effect of that for Object B is still under processing (i.e. Object B is in period O1). In this case, if a Link Abort action is targeted to L1, the preparation status of Object A is evaluated as “ready”, however, that of Object B is “not ready”.

2 – A Link Abort action does not affect a link activation status. A link can fire even if a Link Abort action targeted to it is being processed. Only the actions originated by a link before the Link Abort action is targeted to such link are affected by the Link Abort action.

3 – It is advised to limit the use of Link Abort actions as much as possible. Its main role is to terminate the processing of actions that are performing MHEG effects with transition durations in R3.TD.

39.2.1.2 Link Abort additional error conditions

- If one of the targets is not a link object in L2, this action is ignored for that target.

40 Content Class Generic Value Storage Behaviour

This behaviour is of storage and retrieval of a generic value within the data of a content object.

40.1 Behaviour attributes and statuses

This behaviour is expressed using the following attribute or status:

- Data (see 40.2).

40.2 Data

A generic value may be stored in a content object and may be retrieved from a content object for further processing. By using the generic value storage behaviour, a content object may be used as a variable holder.

40.3 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following actions:

- Set Data action (see 40.3.1);
- Add action (see 40.3.2);
- Subtract action (see 40.3.3).

40.3.1 Set Data action

This action enables to store or to modify a generic value in the data field of a content object.

This action has the following parameters:

- Set of Content Target Param;
- Substitution Indicator Param;
- Set of Data Element Param.

This action may be targeted during the period O2.

40.3.1.1 Set Data targeted to a single content object

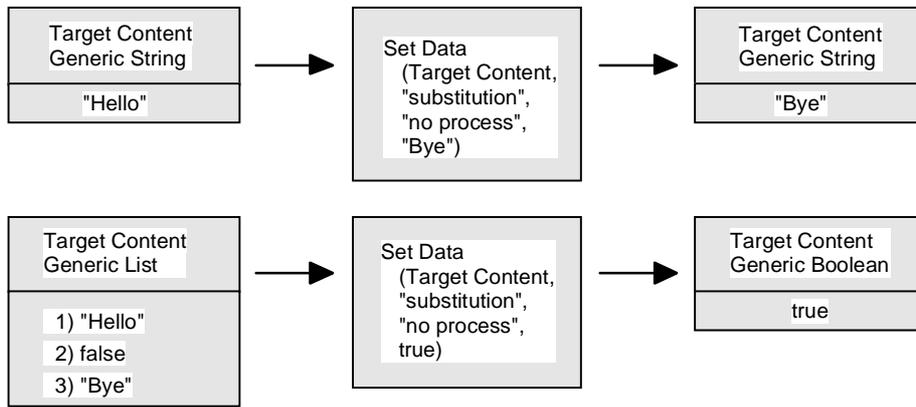
If a set of data element parameters is provided, the elements in this set are interpreted sequentially. For each data element parameter, the following effect applies:

- 1) If the generic list element ID parameter is omitted:
 - a) And if the substitution indicator parameter is equal to “substitution”, the previously stored generic value is totally substituted by the specified generic value (see Figure 48).
 - b) Otherwise (substitution indicator parameter is “no substitution”):
 - i) If the target contains a generic list, the specified generic value is added as a new element of the generic list in the target. If the target contains n elements, the new value is indexed as n + 1th element (see Figure 49).
 - ii) Otherwise, this action is ignored.
- 2) Otherwise (generic list element ID parameter is specified):
 - a) If the target does not store a generic list:
 - i) If the substitution indicator parameter is “substitution”, the previously stored value is discarded and new generic list is to be created. The specified generic value is stored in the specified generic list element ID, and other elements until the last path numeric in the specified generic list element ID should be filled with “unspecified” (see Figure 50).
 - ii) Otherwise, this action is ignored.
 - b) Otherwise (the target is a generic list):
 - i) If the specified generic list element ID exists in the target:
 - If the substitution indicator is “substitution”, the specified generic value is stored in the specified element ID, overwriting the previous value (see Figure 51).
 - Otherwise (the substitution indicator is “no substitution”), this action is ignored.
 - ii) If the specified generic list element ID does not exist in the target and it is possible to extend generic list element ID as specified, the generic list of the target is extended in order to store the specified generic value. Empty elements created by this extension are filled with “unspecified”. In this case, the substitution indicator is ignored (see Figure 52).
 - iii) Otherwise (the specified generic list element ID does not exist and it is impossible to extend), this action is ignored (see Figure 53).

The order of the data element parameters is important, especially if no generic list element ID is specified for the generic value with “no substitution” and the target has a generic list.

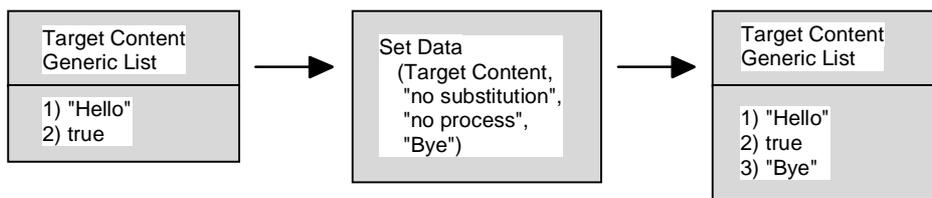
40.3.1.2 Set Data additional error conditions

- if one of the targets is not a content object in O2, this action is ignored for that target;
- if the hook of the targeted content object does not specify generic value, this action is ignored;
- if a set data contains an element value which is specified as a Get Data action applied to the same target content object, this element value is replaced by an “unspecified” value;
- if the target is not a non-mux content, this action is ignored for that target.



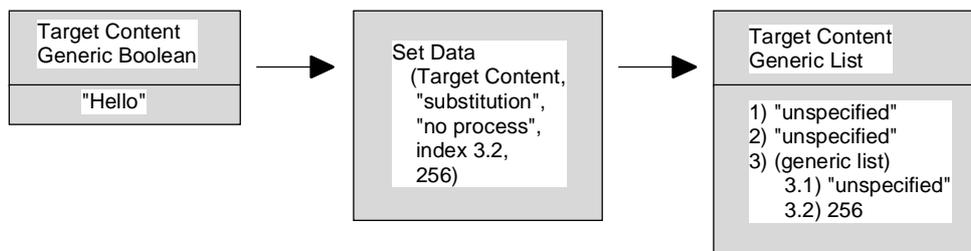
T0825930-96/d046

Figure 48/T.171 – Generic value substitution



T0825940-96/d047

Figure 49/T.171 – Addition of an element to generic list



T0825950-96/d048

Figure 50/T.171 – Creation of list

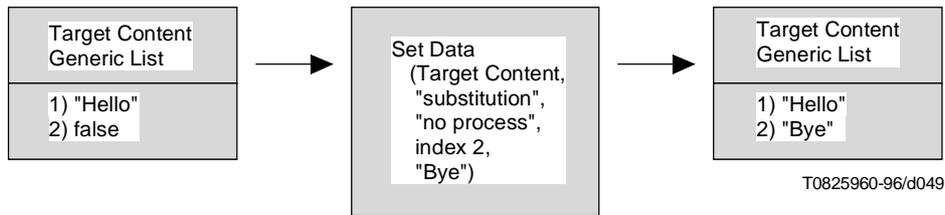


Figure 51/T.171 – Substitute of an element in a generic list

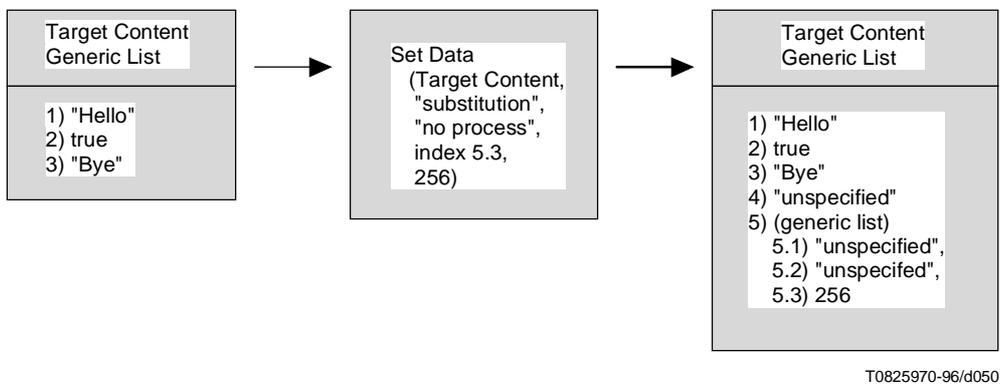


Figure 52/T.171 – Extension of a list

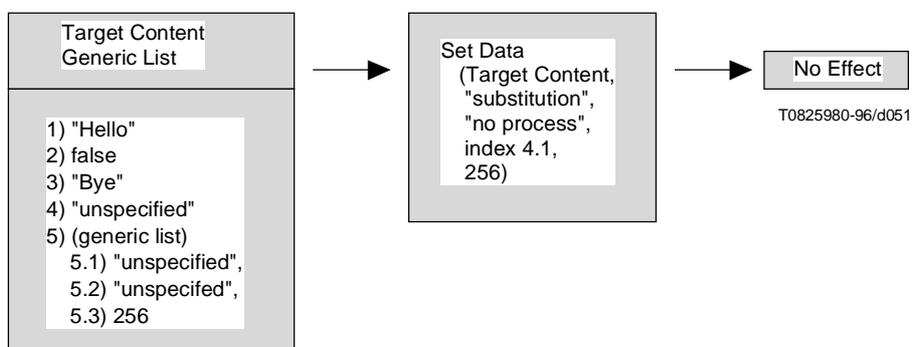


Figure 53/T.171 – Incompatible generic list element ID

40.3.2 Add action

This action provides an arithmetic operation that adds generic numeric, generic integer or generic ratio to the generic value contained in the data field of the target.

This action has the following parameters:

- Set of Content Target Param;
- Generic List Elt ID Param;
- Generic Value Param.

This action may be targeted during the period O2.

40.3.2.1 Add action effect

If the generic list element ID parameter is omitted, this action should be targeted to a content object which does not contain a generic list. Otherwise, this action should be targeted to a content object which contains a generic list and the specified generic list element ID should exist.

The generic value type of the target content data and the specified generic value should be the same, and they are either generic numerics, generic integers or generic ratios.

This action has the following effects:

- 1) if the generic value parameter is provided, the content data or the element data specified by the generic list element ID parameter is increased by the generic value parameter;
- 2) otherwise, the content data or the element data specified by the generic list element ID parameter is incremented by 1 (generic numeric 1, generic integer 1 or generic ratio 1 depending on the generic value type in the target content object).

40.3.2.2 Add additional error conditions

- if one of the targets is not a content object in O2, this action is ignored for that target;
- if the generic list element ID parameter is omitted and the target has a generic list, this action is ignored;
- if the generic list element ID parameter is provided and the target does not have a generic list, this action is ignored;
- if the generic list element ID parameter is provided and the target does not have a specified element, this action is ignored;
- if the generic value type of the target content data and the specified generic value are different, this action is ignored;
- if the generic value type of the target content data is neither generic numerics, generic integers nor generic ratio, this action is ignored;
- if the target is not a non-mux content, this action is ignored for that target.

40.3.3 Subtract action

This action provides an arithmetic operation that subtracts generic numeric, generic integer or generic ratio from the generic value contained in the data field of the target.

This action has the following parameters:

- Set of Content Target Param;
- Generic List Elt ID Param;
- Generic Value Param.

This action may be targeted during the period O2.

40.3.3.1 Subtract targeted to a single content object

If the generic list element ID parameter is omitted, this action should be targeted to a content object which does not contain a generic list. Otherwise, this action should be targeted to a content object which contains a generic list and the specified generic list element ID should exist.

The generic value type of the target content data and the specified generic value should be of the same type, and they are either generic numerics, generic integers or generic ratios.

This action has the following effects:

- 1) if the generic value parameter is provided, the content data or the element data specified by the generic list element ID parameter is decreased by the generic value parameter;
- 2) otherwise, the content data or the element data specified by the generic list element ID parameter is decremented by 1 (generic numeric 1, generic integer 1 or generic ratio 1 depending on the generic value type in the target content object).

40.3.3.2 Subtract additional error conditions

- if one of the targets is not a content object in O2, this action is ignored for that target;
- if the generic list element ID parameter is omitted and the target has a generic list, this action is ignored;
- if the generic list element ID parameter is provided and the target does not have a generic list, this action is ignored;
- if the generic list element ID parameter is provided and the target does not have a specified element, this action is ignored;
- if the generic value type of the target content data and the specified generic value are different, this action is ignored;
- if the generic value type of the target content data is neither generic numerics, generic integers nor generic ratio, this action is ignored;
- if the target is not a non-mux content, this action is ignored for that target.

40.4 Actions to retrieve the behaviour

This Recommendation defines means to get this behaviour by the use of the following action:

- Get Data action (see 40.4.1).

40.4.1 Get Data action

This action retrieves a generic value or an element of a generic list stored in a content object.

This action has the following parameters:

- Content Target Param;
- Generic List Elt ID Param.

This action may be targeted during the period O2.

40.4.1.1 Get Data targeted to a content object

The MHEG effect of this action is to retrieve a generic value or an element of a generic list stored in a content object. This action evaluates a generic value as follows:

- 1) if generic list element ID parameter is not specified, this action evaluates the whole generic value stored in the target content object;
- 2) otherwise, this action evaluates the generic value stored in the place specified by the generic list element ID parameter within the target content object.

40.4.1.2 Get Data additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if one of the targets is not a content object in O2, this action is ignored for that target;
- if the target is not a content object which does not contain a generic value, e.g. a still picture, a video, this action is ignored for that target;

- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if the element list index is an incompatible element list index, i.e. it indicates a non-existent path within the data, this action returns “undefined”;
- if the target is not a non-mux content, this action returns “undefined”.

41 Content Class Copy Behaviour

This behaviour generates new content objects from an existing one.

This behaviour enables to copy a model content object to other independent content objects. The copied content objects have independent identifications provided by the object designer. They are completely independent of the original content object, i.e. any modification to the original does not affect the copied content objects.

41.1 Behaviour attributes and statuses

None.

41.2 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following action:

- Copy action (see 41.2.1).

41.2.1 Copy action

This action specifies a target content object as a source of the copy operation and a set of content objects as destinations of the copy operation.

This action has the following parameters:

- Content Target Param;
- Set of Destination Param.

This action may be targeted during the period O2.

41.2.1.1 Copy action effect

The effect of this action is as follows:

- 1) If the content object specified as a destination parameter does not exist, the specified content object is created. If the destination is specified by an MHEG identifier, a content object containing the MHEG identifier is created.
- 2) All attributes of the target content object except the MHEG identifier attribute are copied in each destination content object.

A target content object may be specified by a container element which is the content object. However, a destination content object shall not be specified by a container element even if it is a content object.

No rt-contents shall exist for a destination content object at the time of a Copy action.

41.2.1.2 Copy additional error conditions

- if one of the targets is not a content object in O2, this action is ignored for that target;
- if rt-contents are created from a content object referenced in a destination parameter, this action is ignored for that destination;
- if a content object referenced in a destination parameter is specified as a container element reference, this action is ignored for that destination;
- if the target is not a non-mux content, this action is ignored for that target.

SECTION 6 – RT-OBJECTS BEHAVIOUR

42 Rt-Objects Availability Behaviour

This behaviour describes the availability for running for rt-objects by the MHEG engine. All rt-objects have this behaviour.

42.1 Behaviour attributes and statuses

This behaviour is expressed using the following attribute or status:

- Rt-Availability Status (see 42.2).

42.2 Rt-Availability Status

The rt-availability status value of each rt-object should evaluate to “available” if the rt-object is available for running for the MHEG engine, i.e. the rt-object is created from a model object and in period R2. The rt-availability status value of each rt-object should be evaluated to “not-available” if the rt-object is not created or deleted, i.e. the rt-object is in period R1.

Initially, the rt-availability status of each rt-object is “not available”.

42.3 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following actions:

- New action (see 42.3.1);
- Delete action (see 42.3.2).

42.3.1 New action

This action creates rt-objects from model objects. This is used to obtain a kind of view of the model object for running and possibly for perception by the user. The rt-object creation process is outside the scope of this Recommendation and is provided by the MHEG engine.

This action has the following parameter:

- Set of Rt-Target Param.

This action may be targeted during the period R1.

42.3.1.1 New action effect

The MHEG effect is processed immediately. There is no associated user effect. The MHEG effect of this action is as follows:

- 1) Construct the targeted rt-object.
It is for the MHEG engine to decide if data copying of the model object is required.
- 2) Assign the rt-object identifier. This is the target of the New action.
- 3) Perform specific effects of the New action on the following objects:
 - a) Rt-script:
 - i) Passing parameter behaviour is initialised.
 - ii) Rt-script termination behaviour is set to “not terminated”.
 - b) Rt-Component:
 - i) Presentation behaviours are initialised.
 - ii) Interaction behaviour is initialised.

- c) Rt-Composite:
 - i) Construct each component within the rt-composite.
Each composite object represents one generation. The construction is recursive when a composite object is associated with a composition element. With this facility, the MHEG engine combines all the generations to construct a complete rt-composite.
 - ii) Target a Plug action to each socket in the rt-composite – The data to be plugged is the data found in the associated model of the corresponding element. When an empty model is associated, a null rt-object is to be plugged in the corresponding socket.
- 4) NOTE 1 – The Plug action is targeted automatically by the MHEG engine. The author need not specify this action. This is an advantage of the associated model facility in the composite object.
- 5) NOTE 2 – Strictly speaking, a label or a component object cannot be plugged directly into a socket. When a label is associated, the MHEG engine is expected to create a content object with the label as data and then create an rt-content from this content. When a component is associated, the MHEG engine processes an implicit New action to create an rt-component from that component, and then plugs that rt-component into the socket.
- 6) NOTE 3 – The New action is propagated to the next generation in the rt-composite through the effect of the Plug action:
 - sockets presentation and structural dynamism behaviour is initialised;
 - presentation behaviours are initialised.
- 7) The running behaviour is initialised to “not running”.
- 8) Set the rt-availability status of the targeted rt-object to “available”. The created rt-object enters in period R2.
- 9) Rt-availability start-up link in a composite object is fired.

When the original model object is in O1, an implicit Prepare action is targeted to the model object, and then the New action is applied as above.

42.3.1.2 New additional error conditions

- if in one of the targets, the original object identification addresses an object which is not a model object, this action is ignored for this target;
- if one of the targets does not address an rt-object, this action is ignored for this target;
- if one of the targets is not an rt-object in R1, this action is ignored for that target.

42.3.2 Delete action

This action removes rt-objects from the MHEG engine. This may be used to release resources in the MHEG engine. This action does not affect model objects from which rt-objects are created.

This action has the following parameter:

- Set of Rt-Target Param.

This action may be targeted during the periods R2, R3 and R3.TD.

42.3.2.1 Delete action effect

There is no direct user effect associated; however, there may be an indirect user effect in certain cases. The MHEG effect of this action is as follows:

- 1) Perform specific effects of this action to the following targets:
 - a) *Rt-script*:
 - i) Passing parameters behaviour is destroyed.
 - ii) Termination behaviour is destroyed.
 - b) *Rt-Component*:
 - i) Presentation behaviours are destroyed.
 - ii) Interaction behaviour is destroyed.

c) *Rt-Composite*:

- i) Sockets presentation and structural dynamism behaviour is destroyed.

The Plug action defined in this behaviour is no longer applied to sockets. This behaviour is no longer possible.

- ii) Presentation behaviour is destroyed.

- iii) Propagates the Delete action to each rt-component plugged in each socket of this rt-composite. It destroys the complete rt-composite when the propagation is completed.

- iv) If a label is plugged in a socket, destroys the dynamically created content object used for the Plug action process.

- 2) Destroys running behaviour.

- 3) Destroys the targeted rt-object.

- 4) Set the rt-availability status of the targeted rt-object to “not available”. The rt-object enters in period R1.

- 5) Rt-availability close-down link of the corresponding composite object is fired.

42.3.2.2 Delete additional error conditions

- if in one of the targets, the original object identification addresses an object which is not a model object, this action is ignored for this target;
- if in one of the targets, the original object identification addresses a model object which is not in period O2, this action is ignored for that target because the model object is not yet available;
- if one of the targets is not an rt-object in R2, R3, and R3.TD, this action is ignored for that target.

42.4 Actions to retrieve the behaviour

This Recommendation defines means to get this behaviour by the use of the following action:

- Get Rt-Availability Status action (see 42.4.1).

42.4.1 Get Rt-Availability Status action

This action retrieves the availability status of each rt-object for the MHEG engine.

This action has the following parameter:

- Rt-Target Param.

This action may be targeted during the periods R1, R2, R3 and R3.TD.

42.4.1.1 Get Rt-availability Status targeted to rt-object

The MHEG effect of this action is as follows:

- 1) if the targeted rt-object is in period R1, the rt-availability status is evaluated to “not available”;
- 2) otherwise, it is evaluated to “available”.

42.4.1.2 Get Rt-Availability Status additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not an rt-object, this action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if in one of the targets, the original object identification addresses a model object which is not in period O2, this action is ignored for that target because the model object is not yet available.

43 Rt-Objects Running Behaviour

This behaviour is used to run and stop rt-objects.

43.1 Behaviour attributes and statuses

This behaviour is expressed using the following attribute or status:

- Running Status (see 43.2).

43.2 Running Status

An rt-object may be run by the MHEG engine at different moments in time. When an rt-component is running, it may be presented to the user. When an rt-script is running, the script itself is under processing.

The running of each rt-object for the MHEG engine may be evaluated.

Initially, the running status of each rt-object is evaluated to “not running” and the rt-object is in period R2. The running status of each rt-object is evaluated to “running” if the rt-object is running and its presentation or processing is controlled by the MHEG engine.

43.3 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following actions:

- Run action (see 43.3.1);
- Stop action (see 43.3.2).

43.3.1 Run action

This action enables the presentation and processing of the rt-object for the MHEG engine.

This action has the following parameters:

- Set of Rt-Target Param;
- Number of Performances Param.

This action may be targeted during the periods R1 and R2.

43.3.1.1 MHEG effect of Run targeted to a single rt-object

If the rt-availability status of the target is “not available”, an implicit New action is targeted at this rt-object. If the target is in period R2, the MHEG effect is processed immediately as follows:

- 1) Place the target under the control of the presentation or processing system.
- 2) Set number of completed performances to 0.
- 3) Perform specific effects as follows:
 - Rt-Component: The current temporal position is positioned at the current temporal position of its RPS.
- 4) NOTE 1 – By default, when an rt-component is created, its current temporal position is set to the start point of its OVD, i.e. its initial temporal position.
- 5) NOTE 2 – The Run action starts the presentation of the rt-component at the current temporal position of the target where it has been left by a previous Run action, i.e. the current temporal position is not reset by targeting the Run action.
- 6) Set the running status to “running”. The rt-object enters in period R3.
- 7) If the current temporal position reaches the terminal temporal position, the number of completed performances is increased by 1.
- 8) If the specified number of performances is not equal to the number of completed performances, the current temporal position is set to the initial temporal position and presentation continues from the beginning of the OVD.
- 9) The temporal termination of the target affects if the specified number of performances is equal to the number of completed performances.

43.3.1.2 User effect of Run targeted to a single rt-object

The user effect of this action is as follows:

- 1) Rt-script: The user effect is described within the script data. At the end of the process of an rt-script, an implicit stop action is targeted at the rt-script by the MHEG engine.

NOTE 1 – In that case, the termination status is set to “terminated”. The process of the rt-script has been completed until the end of the script.
- 2) Rt-Component:
 - a) The current temporal position of the target progresses within the OVD of the target taking into account the current GTF value. All presentation behaviour attributes may affect the user effect (e.g. position, size, audible volume).
 - b) When the terminal temporal position (i.e. the end of OVD) of the target is crossed after the specified number of performances, the following applies:
 - If the temporal termination of the target is equal to “freeze”, the current GTF value is memorised and the GTF value is temporarily set to 0 by the MHEG engine. The GTF of the target becomes 0. The current temporal position remains constant at the terminal temporal position value. The rt-component is still “running”.
- 3) NOTE 1 – If the information presented is a video, the last frame may be displayed. If it is an audio, no sound may be perceived. If it is a still picture, it may remain presented.
- 4) NOTE 2 – If the current temporal position is changed to another position, the memorised GTF values is resumed.
 - Otherwise (the temporal termination of the target is equal to “stop”), an implicit Stop action is targeted to the rt-component.
- 5) NOTE 3 – By default, the terminal temporal position is set to the rt-component original duration. If this duration is infinite, an explicit Stop action should be targeted to that rt-component to stop the user effect of the Run action.
- 6) NOTE 4 – If a timestone is positioned on the terminal temporal position, it is crossed as many times as the specified number of performance just when the current temporal position of rt-component crosses it. If an rt-component is set to “freeze” and there is such a timestone, the timestone is crossed just at the time the current temporal position arrives on the terminal temporal position every time.
- 7) Rt-Composite:
 - a) All the user effects for rt-component applies.
 - b) Each time the current temporal position of an rt-composite crosses a temporal position where a child socket is attached, an implicit Run action is to be targeted to that socket.

A socket may be positioned on a temporal position on its parent T axis (PRPS) using a Set PVD Position action.
 - c) If the terminal temporal position of an rt-composite is crossed, a Set CTP action is targeted to each socket that is still running to set it to its terminal temporal position.
- 8) NOTE 5 – The VD of the sockets that are running and not at their terminal temporal position are effectively shortened when the terminal temporal position of the parent is crossed. Whatever their positions, if the parent is terminated, they are terminated too.
- 9) NOTE 6 – They are positioned at their terminal temporal position so that they restart at their initial temporal position when they become running again.

43.3.1.3 Run additional error conditions

- If the target is not a socket whose parent is not in period R3 or R3.TD, this action is ignored for that target. The parent needs to be running to set the children or descendants running.
- If one of the targets is not an rt-object in R1 or R2, this action is ignored for that target.
- If the number of performance is less than 1, this action is ignored for the whole target set.

43.3.2 Stop action

This action sets the running status to “not running” and stops possible user effects.

This action has the following parameter:

- Set of Rt-Target Param.

This action may be targeted during the periods R3 and R3.TD.

43.3.2.1 Stop targeted to a single rt-object

The action effect of this action is as follows:

- 1) Perform specific effects to following objects:
 - a) Rt-Component: The progression of the current temporal position is stopped and it remains at its current value.
 - b) Rt-Composite: The Stop action is propagated to all child sockets, i.e. to all descendant sockets recursively.
- 2) NOTE 1 – A socket cannot be running if its parent is not running.
- 3) NOTE 2 – All sockets, that are assigned to this composition space and that are directly assigned to a channel, are stopped.
- 4) Stop possible user effects of the target.
- 5) Set the running status to “not running”. The rt-object enters in period R2, and all the user effects are stopped.

43.3.2.2 Stop additional error conditions

- If one of the targets is not an rt-object in R3 or R3.TD, this action is ignored for that target.
- If the target is not a socket whose parent is not in period R3 or R3.TD, this action is ignored for that target. The children or descendants cannot be running when the parent is not running. The Stop action is useless in this case.

43.4 Actions to retrieve the behaviour

This Recommendation defines means to get this behaviour by the use of the following action:

- Get Running Status action (see 43.4.1).

43.4.1 Get Running Status action

This action gets the running status of each rt-object.

This action has the following parameter:

- Rt-Target Param.

This action may be targeted during the periods R2, R3 and R3.TD.

43.4.1.1 MHEG effect

This action evaluates the running status as follows:

- 1) if the target is in period R3 or R3.TD, the running status is evaluated “running”;
- 2) otherwise, it is “not running”.

43.4.1.2 Get Running Status additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if one of the targets is not an rt-object in R2, R3 or R3.TD, this action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if in one of the targets, the original object identification addresses a model object which is not in period O2, this action is ignored for that target because the model object is not yet available.

44 Rt-Script Passing Parameters Behaviour

This behaviour is used to pass parameters to rt-script.

44.1 Behaviour attributes and statuses

None.

44.2 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following action:

- Set Parameters action (see 44.2.1).

44.2.1 Set Parameters action

This action enables to pass parameters to rt-scripts and to return information back from the rt-script processing. Any number of rt-script parameters may be defined for a given rt-script.

Each rt-script parameter is specified as a generic value or a content object. Since a generic value is a result of a Get action or a constant, using a generic value is similar to “call by value” However, using a content object is similar to “call by reference” since a content object may contain or refer to a generic value, image, video and so on. This action may be targeted at any time during the life of an rt-script even when an rt-script is under processing of a script engine, and may also be targeted several times during the life of the rt-script.

The semantics, the parameter order, the type (input, output or input/output) and any usage of each parameter shall be defined by the object designer and shall be supported by a using application and a script engine.

No passing parameter may be specified. It may be used to signal or to notify the script engine or the MHEG engine. The semantics of this usage depends on the using application.

This action has the following parameters:

- Set of Rt-Script Target Param;
- Set of Passing Param.

This action may be targeted during the periods R2, R3 and R3.TD.

44.2.1.1 Set Parameter action effect

There is no direct user effect associated with this action. However, there may be an indirect user effect in certain cases.

The MHEG effect of this action is to pass the specified passing parameters to the target.

Content objects in passing parameters may be used to return information from a script engine to the MHEG engine. Therefore, if a content object is used as a parameter from which rt-contents are created and running, there may be indirect user effects by an rt-script due to the data modification within the content object. The presentation of such rt-content may be modified.

Suppose an rt-script is in charge of the access to a database to retrieve information and the results are set into a content object. The result may be used to complete a form filling, for example, as follows:

- An rt-script RS1 is created from a model script object MS1: New (RS1).
- A content object CO1 and two generic values of integers are passed to RS1 as parameters: Set Parameters (RS1, CO1, 13, 15).
- RS1 is executed: Run (RS1).
- The execution of RS1 uses the parameters set by the Set Parameters action during the process. After finishing the script, the execution result is stored in CO1.
- The value in CO1 may be retrieved by using a link condition for further processing in the MHEG engine: “When the running status of RS1 becomes “not running””, Get Data (CO1).

44.2.1.2 Set Parameters additional error conditions

- if one of the targets is not an rt-script in R2, R3 or R3.TD, this action is ignored for that target;
- if the reference to content object used to pass parameters is not accessible or does not address a content object, this parameter is replaced by the value “undefined”.

45 Rt-Scripts Termination Behaviour

This behaviour evaluates the complete termination of an rt-script process inside a script engine for the MHEG engine.

45.1 Behaviour attributes and statuses

This behaviour is expressed using the following attribute or status:

- Termination Status (see 45.2).

45.2 Termination Status

This attribute specifies the execution status of an rt-script. It is either “terminated” or “not terminated”. If the execution of an rt-script is completed, the termination status is evaluated to “terminated”. Otherwise, it is evaluated to “not terminated”.

After the creation of an rt-script, the termination status is initialised “not terminated”. After receiving a Run action, the rt-script is executed by a script engine. When it is terminated (i.e. its termination status is changed from “not terminated” to “terminated”), an implicit Stop action is targeted to the rt-script. If a Stop action is targeted during the execution of the rt-script, it becomes “not running” and “not terminated”. Only the rt-objects that are “not terminated” may be rerun. If the same script is to be executed, two different rt-scripts should be made and run. There is no direct relationship between the running status and the termination status.

45.3 Actions to retrieve the behaviour

This Recommendation defines means to get this behaviour by the use of the following action:

- Get Termination Status action (see 45.3.1).

45.3.1 Get Termination Status action

This action retrieves the termination status of the target.

This action has the following parameter:

- Rt-Script Target Param.

This action may be targeted during the periods R2, R3 and R3.TD.

45.3.1.1 Get Termination Status action effect

The effect of this action is as follows:

- 1) If the process of the target rt-script is not terminated, the Termination Status should be evaluated to “not terminated”;
- 2) Otherwise, it should be evaluated to “terminated”.

45.3.1.2 Get Termination Status additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if one of the targets is not an rt-script in R2, R3 or R3.TD, this action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target.

46 Sockets Presentation and Structural Dynamism Behaviour

This behaviour attaches or detaches an rt-component to a socket. This may be done at any time during the life of the rt-composite once the rt-composite is created.

This behaviour has the following particularities:

- 1) Attachment of an rt-content to a socket. This socket becomes an rt-content socket, and the rt-content is perceived through the socket.
- 2) Attachment of an rt-composite to a socket. This socket becomes an rt-composite socket. A new set of generations is attached to the socket. If each generation contains a set of rt-content sockets, a new depth of presentation may be perceived.
- 3) Replacement of an rt-content in an rt-content socket. A new rt-content may be perceived.
- 4) Replacement of an rt-composite in an rt-composite socket. A generation and all its descendants are replaced by a new set of generations. A new generation may be perceived.
- 5) Detachment of an rt-component from a socket. The socket becomes an empty socket. No presentation is perceived.

Initially, each socket has an rt-component plugged into it. This is a null rt-object or an rt-component created from the associated model information exchanged within the composite.

46.1 Behaviour attributes and statuses

None.

46.2 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following action:

- Plug action (see 46.2.1).

46.2.1 Plug action

This action enables dynamic presentation and structuring of the sockets. It specifies the information to be plugged into a socket. This is used to obtain a different presentation or structure from the same composite object model.

The parent of the target should be in R2, R3 or R3.TD

This action has the following parameters:

- Set of Socket Target Param;
- Plug In Param.

This action may be targeted during any period.

46.2.1.1 Plug action effect

The MHEG effect of this action is as follows:

- 1) If the specified socket exists, the following applies:
 - a) If a null rt-object is to be plugged in:
 - i) An implicit Delete action is targeted to the rt-component currently plugged into the targeted socket.
 - ii) A null rt-object is attached to the targeted socket. The socket becomes an empty socket.
 - iii) If the running status of a previous plugged-in socket was “running”, an implicit Run action is targeted to the socket.
 - iv) Any previous user effects may cease.
 - b) If an rt-component is to be plugged in:
 - i) A copy of the specified rt-component to be plugged in is made. This new rt-component has no explicit identification and is referenced only via the socket addressing in which it is to be plugged.

- 2) NOTE 1 – It is for the MHEG engine to assign this identification and ensure that it is unique. It should not be possible for an author to use the same value.
- 3) NOTE 2 – For an rt-content, a New action may be performed on the model object and then a transfer of behaviour may be performed from the targeted rt-content to this newly created rt-content.
- 4) NOTE 3 – For an rt-composite, it is not possible to perform a New action on the model composite object in order to make a copy. The created one is not always the same as the targeted rt-composite due to the structural and presentation dynamism behaviour of sockets that may be applied to the rt-composite.
 - ii) If the running status of the new made rt-component is “running”, it is set to “not running”.
 - iii) An implicit Delete action is targeted to the rt-component currently plugged into the targeted socket.
 - iv) The created new rt-content is attached to the targeted socket with its currently associated behaviours.
 - v) If the running status of previous plugged in socket was “running”, an implicit Run action is targeted to the socket.
 - vi) Any previous user effects may be modified as a result of the Plug action.
- c) If a component object is to be plugged in:
 - i) An rt-component is created from this model component object. This rt-component has no explicit identification and is referenced only via the socket addressing in which it is to be plugged.
 - ii) This new made rt-component is attached to the targeted socket with its currently associated behaviours.
 - iii) If the running status of previous plugged in socket was “running”, an implicit Run action is targeted to the socket.
 - iv) Any previous user effects may be modified as a result of the Plug action.
- d) If a label is to be plugged in:
 - i) A content object is dynamically created. This content object has no explicit identification and is referenced only via the socket in which an rt-content created from this content object is to be plugged.
 - ii) Apply all steps in the case as a component object is to be plugged in.
- 5) Otherwise (the specified socket does not exist), the specified socket is to be created, and the following applies:
 - a) If the first index of the socket address does not exist, the children sockets are completed with empty sockets until this socket index. For example, if the number of sockets is n and the first index of the targeted socket address is m ($n < m$), new sockets should be indexed from $n + 1$ to m .
 - b) If the second index of the socket address is provided:
 - i) If the first index indicates an rt-composite and the second index does not exist, the children sockets of this second generation are completed with empty sockets as described in the first index case.
 - ii) If the first index does not indicate an rt-composite:
 - An implicit Delete action is targeted to the rt-content currently plugged-in there.
 - A new rt-composite is created with the number of sockets equal to the second index. Each socket within this rt-composite is created as an empty socket. This new rt-composite has no explicit identification and is referenced only via the socket addressing in which it is to be plugged.
 - This new rt-composite is plugged into the socket addressed by the first index.

- c) Above steps are repeated until the last index of the targeted socket address.
- d) Once socket creations are completed, the target addresses an empty socket and the steps for existing socket address described above is applied.

46.2.1.2 Plug additional error conditions

- if one of the targets does not address a socket that exists or does not exist, this action is ignored for that target;
- if one of the targets is a socket with its parent in period R1, this action is ignored for that target.

47 Rt-Composite Navigation Behaviour

This behaviour provides a navigation through the trees formed by the rt-composites.

47.1 Behaviour attributes and statuses

This behaviour is expressed using the following attributes or statuses:

- Rt-composite address (see 47.2);
- Navigation Command (see 47.3);
- Child (see 47.4);
- EmptyChild (see 47.5);
- Sibling (see 47.6);
- Ancestor (see 47.7).

47.2 Rt-Composite Address

An address referencing an rt-composite.

47.3 Navigation Command

It gives a navigation path. It is a choice among “Child”, “EmptyChild”, “Sibling” or “Ancestor”.

47.4 Child

Navigation path to reach a child. It is expressed as an integer $N > 0$ for N-th child, “last” which specifies the last indexed child or “random” which specifies randomised indexed child.

47.5 EmptyChild

Navigation path to reach an empty socket. It is expressed as an integer $N > 0$ for N-th empty socket or “last” which specifies the last empty socket in a composite.

47.6 Sibling

Navigation path to reach a relative sibling. It is expressed as an integer for relative sibling, e.g. (1 = previous, +1 = next, 0 = current).

47.7 Ancestor

Navigation path to reach an ancestor. It is expressed as an integer $N \geq 0$ for N-th previous generation or “root” which specifies the root. Any ancestor of root is root.

47.8 Actions to retrieve the behaviour

This Recommendation defines means to get this behaviour by the use of the following action:

- Get Rt-Composite Address action (see 47.8.1).

47.8.1 Get Rt-Composite Address action

This action provides means to rise to the root, to descend to leaves and to explore the intermediate levels (nodes) in a composition tree.

This action has the following parameters:

- Rt-Composite Target Param;
- Navigation Command Param.

This action may be targeted during the periods R2, R3 and R3.TD.

47.8.1.1 Get Rt-Composite Address effect

The result of the action is a generic reference addressing an rt-composite specified by the navigation command parameter.

NOTES

- 1 – Clearly this reference is suitable for use as a target of other actions.
- 2 – It may be stored in a content object with Set Data action (see 72.8.1) to be used later.

This action may be recursively used.

47.8.1.2 Get Rt-Composite Address additional error conditions

- if one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target;
- if the target is not an rt-component, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if in one of the targets, the original object identification addresses an object which is not a model object, this action is ignored for this target;
- navigating to a non-existent socket returns “undefined”;
- if the specified navigation command parameter is invalid, “undefined” is returned.

48 Rt-Components Rps Assignment Behaviour

This behaviour assigns rt-components to an RPS, i.e. a composition space or a channel.

48.1 Behaviour attributes and statuses

This behaviour is expressed using the following attribute or status:

- RPS Assignment (see 48.2).

48.2 RPS Assignment

An RPS is a composition space or a channel.

48.3 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following action:

- Set RPS Assignment action (see 48.3.1).

48.3.1 Set RPS Assignment action

This action assigns an rt-component to an RPS.

This action has the following parameters:

- Set of Rt-Component Target Param;
- RPS Assignment Param.

This action may be targeted during the periods R2, R3 and R3.TD.

48.3.1.1 Set RPS Assignment action effect

The MHEG effect of this action is as follows:

- 1) if the RPS parameter is specified as a reference to a channel, the current RPS assignment of this target is changed to the specified channel;
- 2) otherwise, the current RPS assignment of this target is changed to the PRPS for an rt-component target and to the default channel for a root rt-component target.

After this action, the old OPS-RPS relationship is destroyed and a new one is established. Therefore, previous PVD position mapping and previous PVS mapping of the target are completely destroyed and reinitialised.

48.3.1.2 Set RPS Assignment additional error conditions

- if one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target;
- if the previous RPS assignment and the specified RPS assignment are the same, this action is ignored for that target;
- if the specified reference to a channel does not address an available channel (i.e. the channel is not in period C2), this reference is replaced by the default channel reference and the action should be applied for the default channel.

48.4 Actions to retrieve the behaviour

This Recommendation defines means to get this behaviour by the use of the following action:

- Get RPS Assignment action (see 48.4.1).

48.4.1 Get RPS Assignment action

This action retrieves the RPS assignment to an rt-component and a root rt-component.

This action has the following parameter:

- Rt-Component Target Param.

This action may be targeted during the periods R2, R3 and R3.TD.

48.4.1.1 Get RPS Assignment action effect

The MHEG effect of this action is to retrieve the RPS assigned to the target. The following applies:

- 1) if the target is assigned to a channel, this channel identifier is retrieved;
- 2) If the target is assigned to PRPS, the value “prps” is retrieved.

48.4.1.2 Get RPS Assignment additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not an rt-component, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for this target.

49 Rt-Components Perceptability Behaviour

This behaviour is used to represent perceptability of rt-components and root rt-components for the MHEG engine.

Rt-Components and root rt-components concerned by this behaviour may be either “visible” or “audible”.

49.1 Behaviour attributes and statuses

This behaviour is expressed using the following attributes or statuses:

- Perceptability (see 49.2);
- Presentation Priority (see 49.3).

49.2 Perceptability

The perceptability value is defined in the interval between 0% to 100% represented by a generic ratio.

Each rt-content or root rt-content has a perceptability. How to perceive an rt-content or a root rt-content is influenced by this value. If the perceptability value is set to zero, an rt-content or a root rt-content is not perceived. If the perceptability value is set to 100%, the original perception of an rt-content or a root rt-content is perceived. How to perceive an rt-content or a root rt-content on other perceptability values depends on the MHEG engine.

NOTE – The intensity of a picture may be controlled from 0% to 100%.

Each rt-composite or root rt-composite also has a perceptability. How to perceive its descendants is influenced by this value. If the perceptability value is set to zero, all descendants are not perceived. If the perceptability value is set to 100%, the original perception of all descendants taking into account their own perceptability values is perceived. How to perceive all descendants on other perceptability values depends on the MHEG engine.

Initially, all rt-objects shall have perceptability values as 100%.

49.3 Presentation Priority

All rt-components assigned to a same channel have priorities relative to another. The presentation order is determined in one RPS. If this RPS is of an rt-composite, this may have also a presentation priority relative to other rt-components presented in a same RPS.

A presentation priority is defined by an integer from 0 to 255, where 0 represents the highest priority in the presentation stacking order and 255 represents the lowest priority in it.

If two rt-components have the same presentation priority values in the same RPS, the stacking order is defined by Run actions targeted to these rt-components. That means that the rt-component to which a Run action is targeted most recently has higher presentation priority. If the Run actions are targeted in parallel for those rt-components, the stacking order depends on implementation. However, there are no affections for their presentation priority values.

Initially, all rt-components have a presentation priority value 0.

49.4 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following actions:

- Set Perceptability action (see 49.4.1);
- Set Presentation Priority action (see 49.4.2).

49.4.1 Set Perceptability action

This action enables to change the perceptability of an rt-component or a root rt-component.

This action has the following parameters:

- Set of Rt-Component Target Param.
- Perceptability Param: Specifies the perceptability to be assigned to an rt-component.

- Transition Duration Param: Specifies the perceptability to be assigned to an rt-component.

This action may be targeted during the periods R2, R3 and R3.TD.

49.4.1.1 Set Perceptability action effect

The Set Perceptability action effect is as follows:

- 1) set the perceptability of the targeted rt-component as specified;
- 2) if the target is running, there may be a user effect associated with this action;
- 3) if the transition duration parameter is specified, the perceptability is changed gradually.

49.4.1.2 Set Perceptability additional error conditions

- if one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target;
- if the presentation priority parameter value is not in interval (0%, 100%), it shall be set to the value 0;
- if the transition duration is negative, it is set to zero.

49.4.2 Set Presentation Priority action

This action enables to change the presentation priority of an rt-component or a root rt-component.

This action has the following parameters:

- Set of Rt-Component Target Param;
- Presentation Priority Param;
- Transition Duration Param.

This action may be targeted during the periods R2, R3 and R3.TD.

49.4.2.1 Set Presentation Priority action effect

The effect of Set Presentation Priority action is as follows:

- 1) Set the presentation priority of the targeted rt-component is set as specified.
- 2) If the specified presentation priority is an integer, the presentation priority of the target is set to this value.
- 3) If the specified presentation priority is “up-priority”, the presentation priority of the target is decreased by 1. If the value is 0, the value remains 0.
- 4) Otherwise (the specified presentation priority is “down-priority”), the presentation priority of the target is increased by 1. If the value is 255, the value remains 255.

If the target is running, there may be a user effect associated with this action.

If the presentation priority is equal to “up-priority” or “down-priority”, the transition parameter is ignored.

The use of a transition duration parameter allows an rt-component to appear with a gradual presentation priority. Typically, this is accompanied with a special effect such as a change of the stacking order of the presentation progressively.

49.4.2.2 Set Presentation Priority additional error conditions

- if one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target;
- if the presentation priority parameter value is not in interval (0, 255), it is set to the value 0;
- if the transition duration is negative, it is set to zero.

49.5 Actions to retrieve the behaviour

This Recommendation defines means to get this behaviour by the use of the following actions:

- Get Perceptability action (see 49.5.1);
- Get Presentation Priority action (see 49.5.2).

49.5.1 Get Perceptability action

This action retrieves the perceptability value of an rt-component.

This action has the following parameter:

- Rt-Component Target Param.

This action may be targeted during the periods R2, R3 and R3.TD.

49.5.1.1 Get Perceptability action effect

This action evaluates a generic ratio within the interval (0, 1) that is the current value of the perceptability.

49.5.1.2 Get Perceptability additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not an rt-component, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target.

49.5.2 Get Presentation Priority action

This action retrieves the presentation priority value of an rt-component.

This action has the following parameter:

- Rt-Component Target Param.

This action may be targeted during the periods R2, R3 and R3.TD.

49.5.2.1 Get Presentation Priority action effect

This action evaluates a generic integer within the interval (0, 255) that is the current value of the presentation priority.

49.5.2.2 Get Presentation Priority additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not an rt-component, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target.

50 Rt-Components Temporal Behaviour

This behaviour is used to express the presentability of rt-components for the MHEG engine in time.

50.1 Behaviour attributes and statuses

This behaviour is expressed using the following attributes or statuses:

- OD (see 50.2);
- POD (see 50.3);
- OVD (see 50.4);

- PVD (see 50.5);
- Temporal Termination (see 50.6);
- PVD Position (see 50.7);
- CTP (see 50.8);
- GTF (see 50.9);
- Timestone Status (see 50.10);
- Timestone ID (see 50.11);
- Expected OVD Result (see 50.12);
- Expected PVD Result (see 50.13).

50.2 OD

Each rt-component and channel has an OD initialised by the process described in the original presentation space. Each OD for an rt-component is set only once at the time of creation and shall not be changed.

50.3 POD

Each OD is projected to its RPS. The projection is POD. The POD of an rt-component is calculated by the GTF assigned to the rt-component as $POD = OD \times GTF$.

50.4 OVD

Each rt-component has an OVD which is defined as a subset of the OD. Only the part specified as the OVD is perceived by the user. The portion of the OD that is outside the OVD is clipped. The OVD is specified by the initial temporal position and the terminal temporal position that are positions within the OD. Both points are specified by the OGTU in the OPS.

By default, the initial temporal position of the OVD is initialised to 0 (i.e. the origin of the OD), and the terminal temporal position of the OVD is initialised to the length of the OD (i.e. the end point of the OD).

The terminal temporal position may be specified smaller than the initial temporal position (see Figure 56). The length of the OVD is always defined starting from the initial temporal position to the terminal temporal position taking into account the direction of the T axis.

If the terminal temporal position is smaller than the initial temporal position, the VD length is as follows:

- [OD end point (OVD initial temporal position) + (OVD terminal temporal position) OD start point (0)]

Otherwise:

- OVD terminal temporal position (OVD initial temporal position).

50.5 PVD

The projection of each OVD in its RPS results in the PVD. The PVD of an rt-component is calculated by the GTF assigned to the rt-component as $PVD = OVD \times GTF$.

50.6 Temporal Termination

The temporal termination allows an rt-composite to indicate the choice of the termination process when the presentation of the rt-composite is finished run (see 72.11.1).

This attribute is either “freeze” or “stop”. Initially, the temporal termination is set to “freeze”.

50.7 PVD Position

The OVD of an rt-composite is projected to its RPS as PVD. The length of the PVD is controlled by the GTF value of the rt-composite, and the position of the PVD is controlled implicitly or explicitly by an elementary action as follows:

- 1) The PVD position of a root rt-component is implicitly determined by a Run action targeted to the root rt-component.
- 2) After the Run action, the CTP of the target is positioned to its current RPS (CPS) temporal position in the temporal life of the RPS. The PVD of the target is positioned to the RPS taking into account the CTP.
- 3) The CTP of the target progresses taking into account its GTF value.
- 4) The PVD position of an rt-component is determined as follows:
- 5) Initially, the PVD of an rt-component is not positioned to its RPS-OD. If a Run action is targeted to such an rt-component, the CTP of the target is implicitly positioned to its current RPS temporal position in the temporal life of the RPS.
- 6) Once an rt-component is positioned to its RPS-OD by using a Set PVD Position action, the following applies (see Figure 54):
- 7) The PVD of the rt-component is attached to the RPS-OD tightly. Therefore, the CTP of the rt-component is calculated from the CTP of its RPS automatically and shall not be changed by a Set CTP action.
- 8) If the parent CTP exists outside the child PVD, the CTP of an rt-component is “undefined”.
- 9) If the parent of the rt-composite is running and the parent CTP enters in an rt-component PVD, the running status of the rt-component is set to “running” and presentation begins taking into account the playing direction of the parent (forward or reverse). The CTPs of children progress synchronously with the parent CTP.
- 10) If the parent of the rt-composite is running and the parent CTP gets out of an rt-component PVD, the running status of the rt-component is set to “not running” and its presentation ends.
- 11) If a part of the PVD outbounds the OVD of its RPS, such part is clipped and shall not be assigned to RPS. The presentation of such PVD may begin or end at the middle of the PVD corresponding to the edge of parent OVD.
- 12) The binding of the PVD of the rt-component to its RPS is effective until it is explicitly removed.

NOTE – PVD positions are used to realise temporal composition. As the position is set to the rt-composite object of the RPS, the rt-component is automatically started each time the rt-composite is made “running”.

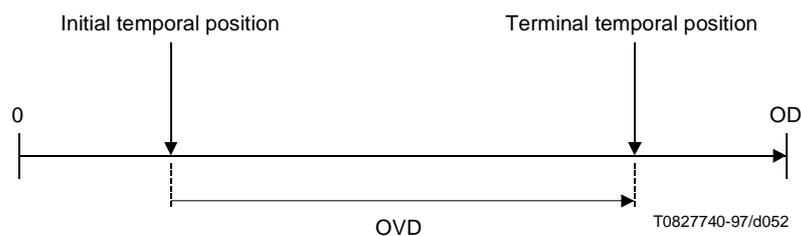


Figure 54/T.171 – Forward presentation

50.8 CTP

Each rt-component has a CTP within its OVD interval. Initially, the CTP is set to the initial temporal position of the OVD.

The CTP is affected by a Run action and Stop action as follows:

- 1) After a Run action, the CTP progresses at the current GTF (see 50.9) in the PVD of final OPS-RPS chains until the terminal temporal position of the PVD is reached or a Stop action is performed.

NOTES

- 1 – If the GTF is equal to 0, the CTP remains at its current position.
 - 2 – “Progress” means “increment” if the GTF is positive. It means “decrement” if the GTF is negative.
- 2) After a Run action, if the CTP passes the end point of the OD, the CTP is set to zero, i.e. the start point of OD (see Figure 55).
 - 3) After a Run action, if the CTP passes the zero point (i.e. the start point of OD), the CTP is set to the end point of the OD (see Figure 56).
 - 4) After a Stop action, the CTP remains at its current value.

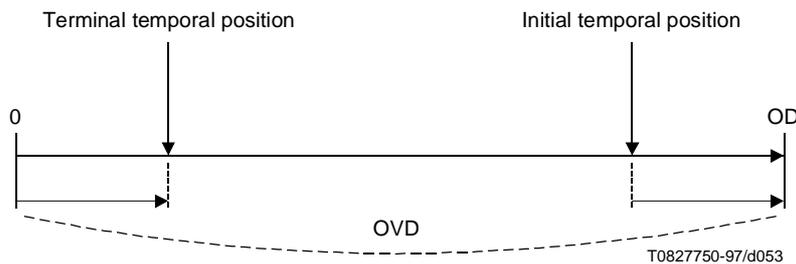


Figure 55/T.171 – Forward presentation with wrap-around

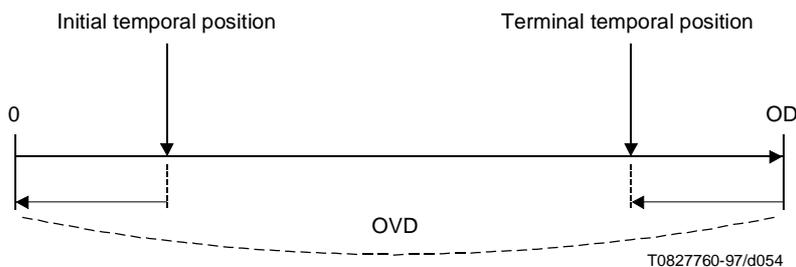


Figure 56/T.171 – Backward presentation with wrap-around

50.9 GTF

The presentation speed of an rt-component is controlled by its current GTF. The GTF defines the number of RGTU corresponding to one OGTU. A GTF value may be negative to express that the playback direction is reverse. In this case, the CTP of an rt-component is progressed reversibly if it is running.

NOTE – Channels do not have GTFs. No temporal factors are available for the mapping to CPS.

50.10 Timestone Status

A timestone is a marker in a temporal position. An rt-component may have any number of timestones defined along its OD. A timestone is composed of:

- A timestone identifier: A generic integer that value is provided by the author. The value 0 is reserved to express that no timestone has ever encountered for the target.
- A temporal position: Expressed in OGTU within the interval of the OD.

As a timestone contains a temporal position and is defined in the OPS, the perception of this point may differ on the RPS T-axis.

NOTE – For example, if the GTF value is doubled, the PVD in the RPS is halved and the CTP progresses in double speed. In this case, the temporal position of OPS looks closer in the RPS.

Timestones are crossed by the CTP when an rt-component is running. The timestone status is defined as a last timestone identifier crossed by the CTP.

Initially, the timestone status is 0, and no timestones are provided on an rt-component. Timestones are specified by the author using a Set Timestones action.

50.11 Timestone ID

A generic integer. The value 0 is reserved.

50.12 Expected OVD Result

Choice among “initial-temporal-position”, “terminal-temporal-position” or “duration”.

50.13 Expected PVD Result

Choice among “initial-temporal-position”, “terminal-temporal-position” or “duration”.

50.14 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following actions:

- Set OVD action (see 50.14.1);
- Set CTP action (see 50.14.2);
- Set Temporal Termination action (see 50.14.3);
- Set PVD Position action (see 50.14.4);
- Set GTF action (see 50.14.5);
- Set Timestones action (see 50.14.6).

50.14.1 Set OVD action

This action specifies an OVD within an OD.

This action has the following parameters:

- Set of Rt-Component Target Param.
- Initial Point Spec Param;
- Terminal Point Spec Param.

This action may be targeted during the periods R2, R3 and R3.TD.

50.14.1.1 Set OVD effect

The initial temporal position and the terminal temporal position may be specified as follows:

- 1) By an absolute value in the OGTU of the target.
- 2) By a relative value relatively to the OD. The value is interpreted as $OD \times (\text{specified value})$. If the calculated value is outside the OD, the value is set to 0 (if negative) or OD (if greater than OD). If the specified value does not indicate an addressable point, this value shall be rounded.

If the target is running, there may be a user effect associated, i.e. the perception duration may be shortened or extended.

50.14.1.2 Set OVD additional error conditions

- if one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target;
- if one of the temporal position parameters specified as an absolute point is not in the interval (0, OD), this action is ignored for the whole target set.

50.14.2 Set CTP action

This action specifies the CTP within the OVD.

This action has the following parameters:

- Set of Rt-Component Target Param.
- Current Point Spec Param.

This action may be targeted during the periods R2, R3 and R3.TD.

50.14.2.1 Set CTP effect

The CTP may be specified as follows:

- 1) If the position is specified as an absolute value, the CTP is set to this value.
- 2) If the position is specified as a relative value, this value is interpreted as a relative value to the OVD of the target. The CTP is set to $OVD \times (\text{specified value})$.
- 3) If the position is specified as an original point factor, the CTP is set to (initial temporal position) \times (specified value).
- 4) Otherwise (the position is specified as a current point factor), the CTP is set to $CTP \times (\text{specified value})$.
- 5) In above cases, if the calculated value is outside the OVD, the value is set to the nearest point within the OVD.
- 6) If the specified point does not indicate an addressable point, this value shall be rounded.

If the target is running, there may be a user effect associated. If the playback is forward and the CTP is set to a smaller one, this has the effect of replaying a part of the target. If the playback is forward and the CTP is set to a greater one, this has the effect of skipping a part of the target.

A Set CTP action does not affect any timestamp status. If the target is replayed, a previously passed timestamp may be crossed again. If the target is skipped forward, timestamps within this skipped part are not crossed.

50.14.2.2 Set CTP additional error conditions

- if one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target;
- if the temporal position parameter specified as an absolute point is not in the interval of OVD, this action is ignored for the whole target set;
- if one of the targets is assigned to its RPS by a Set PVD Position action, this action is ignored for that target.

50.14.3 Set Temporal Termination action

This action specifies the type of temporal termination if the CTP reaches the end of presentation.

This action has the following parameters:

- Set of Rt-Target Param.
- Temporal Termination Param.

This action may be targeted during the periods R2, R3 and R3.TD.

50.14.3.1 Set Temporal Termination effect

This action sets the temporal termination of the target. If the target is at the end of the presentation and the running status is “running”, there may be a user effect changing the previous temporal termination presentation to the specified new one.

50.14.3.2 Set Temporal Termination additional error conditions

- If one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target.

50.14.4 Set PVD Position action

This action specifies the position within the OD of the parent where to attach the PVD of a socket.

This action has the following parameters:

- Set of Socket Target Param.
- Temporal Position Param: Specifies temporal position either by an absolute value along a temporal axis or a relative value against a temporal duration.

This action may be targeted during the period R2.

50.14.4.1 Set PVD Position effect

The PVD position may be specified as follows:

- 1) By an absolute value in the RGTU of the target.
- 2) By a relative value relatively to the OD of the parent. The value is interpreted as $OD \times (\text{specified value})$. If the calculated value is outside the OD of the parent, the value is set to 0 (if negative) or OD (if greater than OD). If the specified point does not indicate an addressable point, this value shall be rounded.

If the parent of the target is running, there may be a user effect associated. After performing this action, if the CTP of the parent enters the PVD of the target, this target is to be played.

50.14.4.2 Set PVD Position additional error conditions

- if one of the targets is not an rt-component in R2, this action is ignored for that target;
- if the temporal position parameter specified as an absolute point is not in the interval of the parent OD, this action is ignored for the whole target set;
- if the temporal position parameter specified as a relative point is not in the interval of the parent OD, this action is ignored for the whole target set;
- if the target does not have a PRPS, this action is ignored.

50.14.5 Set GTF action

This action defines the presentation speed of an rt-component.

This action has the following parameters:

- Set of Rt-Component Target Param.
- GTF Param: Generic ratio or default-GF.

This action may be targeted during the periods R2, R3 and R3.TD.

50.14.5.1 Set GTF effect

The GTF value of the target is set to the specified value.

If the target is running, there may be a user effect associated with this action. That is the change of the target presentation speed.

50.14.5.2 Set GTF additional error conditions

- If one of the targets is not rt-component in R2, R3 or R3.TD, this action is ignored for that target.

50.14.6 Set Timestones action

This action specifies a set of timestones to be set within the OD of the target.

This action has the following parameters:

- Set of Rt-Component Target Param.
- Set of Timestone Spec Param: Generic ratio or default-GF.

This action may be targeted during the periods R2, R3 and R3.TD.

50.14.6.1 Set Timestones action effect

The effect of this action is as follows:

- 1) According to the update command within the timestone specification parameter, specified timestones are added, removed or replaced.
- 2) If an absolute value is specified as a timestone position, the value is interpreted as the position in the OD.
- 3) If a relative value is specified as a timestone position, the value, $OD \times (\text{specified value})$, is calculated and used.
- 4) A same timestone may be repeated several times or infinitely. This is specified by the “number of repetitions”. Number of repetitions is valid only for “add” command and ignored for other commands. If the number of repetitions is specified greater than 1, this means the corresponding timestones should be repeated every specified time position as many as specified. For example, Set Timestones (target, ((1, 5, 10), add)) action repeats ten times to add timestone 1 every 5 GTUs from the beginning of the OD.

50.14.6.2 Set Timestones additional error conditions

- If one of the targets is not rt-component in R2, R3 or R3.TD, this action is ignored for that target.
- If the timestone position parameter is not in the interval of the OD, this timestone is ignored. All other timestones are applied.
- If the number of repetitions is less than 1 with “add” command, this timestone is ignored. All other timestones are applied.

50.15 Actions to retrieve the behaviour

This Recommendation defines means to get this behaviour by the use of the following actions:

- Get OD action (see 50.15.1);
- Get POD action (see 50.15.2);
- Get OVD action (see 50.15.3);
- Get PVD action (see 50.15.4);
- Get CTP action (see 50.15.5);

- Get Temporal Termination action (see 50.15.6);
- Get PVD Position action (see 50.15.7);
- Get GTF action (see 50.15.8);
- Get Timestone Status action (see 50.15.9).

50.15.1 Get OD action

This action retrieves the OD of the target expressed in OGTU.

This action has the following parameter:

- Rt-Component Target Param.

This action may be targeted during the periods R2, R3 and R3.TD.

50.15.1.1 Get OD action effect

The MHEG effect of this action is to retrieve the OD value of the target expressed in OGTU.

50.15.1.2 Get OD additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not an rt-component, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target.

50.15.2 Get POD action

This action retrieves the POD of the target expressed in RGTU.

This action has the following parameter:

- Rt-Component Target Param.

This action may be targeted during the periods R2, R3 and R3.TD.

50.15.2.1 Get POD action effect

The MHEG effect of this action is to retrieve the POD value of the target expressed in RGTU.

50.15.2.2 Get POD additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not an rt-component, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target.

50.15.3 Get OVD action

This action retrieves the OVD of the target expressed in OGTU.

This action has the following parameters:

- Rt-Component Target Param.
- Expected OVD Result Param: Generic ratio or default-GF.

This action may be targeted during the periods R2, R3 and R3.TD.

50.15.3.1 Get OVD action effect

The MHEG effect of this action is to retrieve the OVD value of the target expressed in OGTU as follows:

- 1) if the expected OVD result parameter is “initial-temporal position”, this action retrieves the initial temporal position of the OVD;
- 2) if the expected OVD result parameter is “terminal temporal position”, this action retrieves the terminal temporal position of the OVD;
- 3) otherwise (the expected OVD result parameter is “duration”), this action retrieves the duration of the OVD.

50.15.3.2 Get OVD additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not an rt-component, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target.

50.15.4 Get PVD action

This action retrieves the PVD of the target expressed in RGTU.

This action has the following parameters:

- Rt-Component Target Param.
- Expected PVD Result Param: Generic ratio or default-GF.

This action may be targeted during the periods R2, R3 and R3.TD.

50.15.4.1 Get PVD action effect

The MHEG effect of this action is to retrieve the PVD value of the target expressed in RGTU as follows:

- 1) if the expected PVD result parameter is “initial-temporal position”, this action retrieves the initial temporal position of the PVD;
- 2) if the expected PVD result parameter is “terminal temporal position”, this action retrieves the terminal temporal position of the PVD;
- 3) otherwise (the expected PVD result parameter is “duration”), this action retrieves the duration of the PVD.

50.15.4.2 Get PVD additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not an rt-component, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target.

50.15.5 Get CTP action

This action retrieves the CTP value of the target.

This action has the following parameter:

- Rt-Component Target Param.

This action may be targeted during the periods R2, R3 and R3.TD.

50.15.5.1 Get CTP action effect

The MHEG effect of this action is to retrieve the CTP value of the target. This action evaluates a temporal position expressed in OGTU.

50.15.5.2 Get CTP additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not an rt-component, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target;
- if a target is positioned to its PRPS by a Set PVD Position action, this action returns “undefined” for that target.

50.15.6 Get Temporal Termination action

This action retrieves the temporal termination value of the target.

This action has the following parameter:

- Rt-Component Target Param.

This action may be targeted during the periods R2, R3 and R3.TD.

50.15.6.1 Get Temporal Termination action effect

The MHEG effect of this action is to retrieve the temporal termination value of the target, that is “freeze” or “stop”.

50.15.6.2 Get Temporal Termination additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not an rt-component, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target.

50.15.7 Get PVD Position action

This action retrieves the PVD position value of the target within its PRPS.

This action has the following parameter:

- Rt-Component Target Param.

This action may be targeted during the periods R2, R3 and R3.TD.

50.15.7.1 Get PVD Position action effect

The MHEG effect of this action is to retrieve the PVD position value of the target within its PRPS. This action evaluates a temporal position expressed in RGTU.

50.15.7.2 Get PVD Position additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not an rt-component, the action is ignored for that target;

- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target;
- if the target is not positioned to its PRPS, this action returns “undefined”.

50.15.8 Get GTF action

This action retrieves the GTF value of the target.

This action has the following parameter:

- Rt-Component Target Param.

This action may be targeted during the periods R2, R3 and R3.TD.

50.15.8.1 Get GTF action effect

The MHEG effect of this action is to retrieve the GTF value of the target. This action evaluates a generic ratio.

50.15.8.2 Get GTF additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not an rt-component, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target.

50.15.9 Get Timestone Status action

This action retrieves the timestone status value of the target.

This action has the following parameter:

- Rt-Component Target Param.

This action may be targeted during the periods R2, R3 and R3.TD.

50.15.9.1 Get Timestone Status action effect

The MHEG effect of this action is to retrieve the timestone status value of the target. This action evaluates an integer corresponding to the timestone identifier of the last timestone crossed by the CTP. If no timestones have ever been crossed, 0 is returned.

50.15.9.2 Get Timestone Status additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not an rt-component, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target.

51 Rt-Components Spatial Behaviour

The behaviour describes presentability in space of rt-components for the MHEG engine.

51.1 Behaviour attributes and statuses

This behaviour is expressed using the following attributes or statuses:

- OS (see 51.2),
- POS (see 51.3),
- Aspect Ratio (see 51.4),
- Resizing Strategy (see 51.5),
- OVS (see 51.6),
- OAP (see 51.7),
- OVS Position (see 51.8),
- PVS (see 51.9),
- OVS Proj Strategy (see 51.10),
- PAP (see 51.11),
- PVS Position (see 51.12),
- GSF (see 51.13),
- Spatial Control (see 51.14),
- User Spatial Control (see 51.15),
- Expected Axis Result Param (see 51.16),
- Point Type Param (see 51.17).

51.2 OS

Each rt-component and each channel has an assigned OS that is defined in its OPS. Therefore, the OS is measured in the OGSU (see Original presentation space for initialisation of each OS).

Each OS for an rt-content is set only once and shall not be changed. However, the OS of an rt-composite may be changed resizing strategy (see 51.5).

51.3 POS

Each OS of an rt-component is projected to its RPS. The projection is called POS. Therefore, the POS is measured in the RGSU. It is calculated as $POS = OS \times GSF$ on each axis.

51.4 Aspect Ratio

The aspect ratio is the ratio among the width, the height and the depth along the spatial axes.

The projection of OS is controlled by GSF and the relationship, $POS = OS \times GSF$, is always respected. However, in order to preserve the aspect ratio of an rt-component, how to fill the POS is specified independently of the GSF. The value of the aspect ratio attribute is either “preserved” or “not-preserved”. This attribute controls the aspect ratio preserving only for one OS-POS mapping. If the rt-component is to be presented in a channel preserving the aspect ratio, every aspect ratio attribute through the OPS-RPS chains from the rt-component to the channel should be set to “preserved”.

For preserving the aspect ratio, the following applies:

- “Preserved”: The rt-component is scaled preserving the aspect ratio. This scaling is independent of the assigned GSF, and the rt-component is scaled so that at least one of the width, height or depth becomes to fit to the POS. Then, the POS is filled with this scaled rt-component. The scaled rt-component may not entirely fit the POS, i.e. some portion of the POS may not be filled by the scaled rt-component. If on a given axis, the scaled rt-component is smaller than the size of the POS, the scaled rt-component is centred on the interval of such axis. How the portions of POS that are not covered by the scaled rt-component are filled is either defined by a (registered) presentation style or implementation dependent.
- “Not preserved”: The POS should be filled entirely with the rt-component taking into account the GSF on each axis.

Initially, the aspect ratio attribute value is set to “preserved”.

NOTES

1 – The POS are not affected by the aspect ratio attribute value, i.e. the aspect ratio of the OS and the aspect ratio of the POS may be different on each axis even if the aspect ratio attribute value is “preserved”.

2 – If an axis has a zero value in the OS, such axis is not taken into account in the scaling of the rt-component.

51.5 Resizing Strategy

The OS for each rt-composite may be modified in order to accommodate the modification in size and position of the PVSs of the child rt-components. This is controlled by a resizing strategy attribute. One of the following three attribute values may be set to an rt-composite:

- “Fixed”: The OS of an rt-composite should remain as the initial size of the OS. The OS cannot be changed until another attribute value is assigned. If this attribute is changed from another value to “fixed” and the previous OS is different from the initial size due to the resizing at that time, the OS should be reset to the initial size.
- “Minimum”: The OS of an rt-composite should have a minimum size that is enough to contain all the PVSs of the child rt-components and greater than or equal to the initial OS of the target on each axis.
- “Grows only”: The OS of an rt-composite should not be reduced. If a part of an rt-component PVS is positioned outside the OS, the OS becomes the minimum size on each axis that is big enough to include every PVS. However, it is never reduced if the OS is bigger than the actual minimum size enough for every PVS.

The initial value is “fixed”.

The portions of PVSs that outbound the OS of an rt-composite are clipped. However, the values of such PVSs does not change. This situation happens only if the value is “fixed”.

If some portions of a PVS outbound the OS of an rt-composite and its resizing strategy is “minimum” or “grows only”, the PAP position of the PVS may be changed due to the resizing. The MHEG engine should take into account this change.

NOTES

1 – The values of such PVSs remain the same because these rt-components may be assigned later to another RPS or may become completely perceivable due to the change of the parent OS.

2 – The size considerations described above should be done independently on each axis. For example, in the case of “minimum”, if the initial value of the rt-composite OS is $x = 10$ and $y = 10$, and the rt-composite has one rt-component whose PVS is $x = 5$ and $y = 20$, the OS of the rt-composite becomes $x = 10$ and $y = 20$.

If the aspect ratio attribute is set to “preserved” for an rt-composite, the size of the data information may be a subset of the OS in order to preserve the aspect ratio.

51.6 OVS

The OVS is a subset of the OS for an rt-component, and is positioned relatively to the OS. The portion of the OS that fits within the OVS is to be presented to the user. The portion of the OS that is outside of the OVS is clipped. If the OVS exceeds the limits of the OS, the exceeding portion of the OVS is filled with a background which depends on the implementation or is specified by an extensibility facility.

The OVS may be specified by one of the following methods (see also OVS Proj Strategy):

- 1) Directly defined in the OPS: The OVS is defined within the OPS by a Set OVS action with OVS Proj Strategy attribute as “calculated”. In this case, a change of the rt-component GSF affects the PVS because the OVS looks like a constant [$PVS = OVS \text{ (constant)} \times GSF$]. However, this change does not affect the PAP and the PVS position.
- 2) Indirectly defined in the RPS: The PVS is defined within the RPS, rather than the OVS, by a Set OVS action with OVS Proj Strategy attribute as “fixed”. The OVS is automatically calculated by $OVS = PVS / GSF$ on each axis. In this case, the OVS may change dynamically depending on the GSF value. And a change of the rt-component GTF does not affect the PVS because the PVS looks like a constant [$PVS \text{ (constant)} = OVS \times GSF$]. This GTF change does not affect the OAP (see 51.7) and the OVS position (see 51.8) to the OVS, but the OVS.

NOTE 1 – However, it affects some part of the OS becoming visible.

The OVS is initialised to the size of the corresponding OS in the OPS using OGSU.

NOTE 2 – Typically, the OVS directly defined in the OPS is used to perform real scaling of an rt-component. Therefore, the PVS and the POS are varied depending on the GTF of the rt-component.

NOTE 3 – Typically, the OVS indirectly defined in the RPS is used to scale only the OS of an rt-component that is projected and perceived through the constant PVS.

51.7 OAP

An OAP defines an attachment position within the OVS. This attachment position with the OVS position is used to position the OVS within its OS. Initially, the OAP is set to the origin of the OVS, i.e. (0, 0, 0).

The OAP may be specified by an absolute position within the OVS, or a relative position relatively to the OVS origin expressed by a generic ratio.

If an OVS is specified indirectly by a Set OVS action with OVS Proj Strategy attribute as “fixed”, the OVS may be dynamically changed depending on the corresponding GSF value. In this case, the OAP shall not be affected by the changing OVS. It should keep the current value. If the value is specified by an absolute one, it should be kept. If the value is specified by a relative one, the relative value should be kept.

51.8 OVS Position

An OVS position defines a position relatively to the origin of the OS where the corresponding OAP should be positioned. The OVS position with the OAP is used to position the OVS within its OS. Initially, the OVS position is set to the origin of the OS, i.e. (0, 0, 0).

The OVS may be specified by an absolute position within the OS, or a relative position relatively to the OS expressed by a generic ratio.

If an OVS is specified indirectly by a Set OVS action with OVS Proj Strategy attribute as “fixed”, the OVS may be dynamically changed depending on the corresponding GSF value. In this case, the OVS position should not be affected by the changing OVS. It should keep the current value. If the value is specified by an absolute value, it should be kept. If the value is specified by a relative one, it should be kept.

51.9 PVS

The PVS is a projection of the OVS of an rt-component to its RPS, and is positioned relatively to the OS of the PRS.

The PVS may be specified by one of the following methods:

- Directly defined in the RPS: The PVS is defined within the RPS by a Set OVS action with OVS Proj Strategy attribute as “fixed”. In this case, the PVS is constant and the corresponding OVS is automatically calculated.
- Indirectly defined in the OPS: The OVS is defined within the OPS by a Set OVS action with OVS Proj Strategy attribute as “calculated”, a Set OAP action and a Set OVS Position action. In this case, the PVS is calculated according to the GSF value and the corresponding OVS is constant.

The PVS is initialised by the OVS initial value and the GSF initial value.

51.10 OVS Proj Strategy

Each rt-component has an OVS Proj Strategy attribute. This attribute defines how to project an OVS to the corresponding PVS. The relationship of the OVS and the PVS, $PVS = OVS \times GSF$ on each axis, is always respected.

This attribute has either “fixed” or “calculated”. The following applies:

- if the OVS Proj Strategy is equal to “fixed”, the PVS is always fixed;
- if the OVS changes, the GSF is recalculated;
- if the GSF changes, the OVS is recalculated;
- otherwise (“calculated”), the PVS is always calculated;
- if the OVS changes, the PVS is recalculated;
- if the GSF changes, the PVS is recalculated.

51.11 PAP

A PAP defines an attachment position within the PVS. This attachment position with the PVS position is used to position the PVS within the OS of the RPS. Initially, the PAP is set to the origin of the PVS, i.e. (0, 0, 0).

The PAP may be specified as an absolute position within the PVS, or a relative position relatively to the PVS origin expressed by a generic ratio.

51.12 PVS Position

A PVS position defines a position relatively to the origin of the OS of the RPS or to another PVS within the same RPS, where the corresponding PAP should be positioned. The PVS position with the PAP is used to position the PVS within the OS of the RPS. Initially, the PVS position is set to the origin of the OS of RPS, i.e. (0, 0, 0).

The PVS position may be specified by an absolute position within the OS, or a relative position relatively to the OS expressed by a generic ratio.

51.13 GSF

The projection of the OPS to RPS is controlled by its GSF value. The GSF defines the number of RGSU corresponding to one OGSU for each axis.

51.14 Spatial Control

Moving, resizing, scaling and scrolling an rt-component may be allowed or not to a user independently of the elementary action controls. This attribute indicates that those functions are allowed or not with the combination of the user spatial control attribute value. The spatial control attribute has one of the following values:

- “moving”: indicating the move of the PVS.
- “resizing”: indicating the resize of the PVS.
- “scaling”: indicating the scale of the OS.
- “scrolling”: indicating the scroll of the OS.

NOTE 1 – Scrolling may be allowed for each rt-component and for the parent of the rt-components independently. The scrolling always refers to the PS of the rt-component.

Initially, those user controls are “not allowed”.

It is the responsibility of the MHEG engine with the user interface to make the user knowledgeable of being allowed to perform a given control and to provide the necessary supports (e.g. scroll bars, resizing handles).

For example, if the scrolling is allowed to the rt-component, a mechanism is to be offered by the MHEG engine to operate a scrolling of the OS as perceived through the PVS. It is the responsibility of the MHEG engine to decide which look-and-feel has to be adopted for the scroll mechanism and to decide whether such a mechanism has to be attached to the rt-component.

If the user control is not allowed, it is the responsibility of the MHEG engine to make the user knowledgeable about that (e.g. scroll bars are not displayed, or pressing scroll keys produce a beep).

NOTE 2 – Typically, a scroll mechanism attached to an rt-component may be implemented through a couple of scroll bars.

NOTE 3 – Typically, a scroll mechanism not attached to an rt-component may be implemented with keys or arrows.

51.15 User Spatial Control

This attribute is used with the spatial control attributes to indicate whether a certain user spatial control is allowed or not.

51.16 Expected Axis Result Param

Choice among “x”, “y”, “z” or “xyz”.

51.17 Point Type Param

Choice between “absolute” and “relative”.

51.18 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following actions:

- Set Aspect Ratio action (see 51.18.1);
- Set Resizing Strategy action (see 51.18.2);
- Set OVS Proj Strategy action (see 51.18.3);
- Set OVS action (see 51.18.4);
- Set OAP action (see 51.18.5);
- Set OVS Position action (see 51.18.6);
- Set PAP action (see 51.18.7);
- Set PVS Position action (see 51.18.8);
- Set GSF action (see 51.18.9);
- Set User Spatial Control action (see 51.18.10).

51.18.1 Set Aspect Ratio action

This action specifies whether the ratio between the OS in the OGSU and the scaled rt-component in the POS of the RGSU is to be preserved for each axis in performing projection of an rt-component OS to the POS.

This action has the following parameters:

- Set of Rt-Component Target Param;
- Aspect Ratio Param.

This action may be targeted during the periods R2, R3 and R3.TD.

51.18.1.1 Set Aspect Ratio action effect

The action effect of this action is as follows:

- 1) if the aspect ratio parameter is “preserved”, the aspect ratio of the scaled rt-component should be preserved;
- 2) otherwise, the rt-component should fit the POS entirely.

This effect continues until the target is destroyed or this action is targeted again with a different value. The continuous effect means that the scaled rt-component within the POS should be recalculated each time the POS is changed in order to maintain the aspect ratio preserved or in order to fit the POS.

If the target is running, there is a user effect associated, i.e. this area where the data information is presented may be stretched to fit the entire POS or squeezed to a subset of it.

51.18.1.2 Set Aspect Ratio additional error conditions

- If one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target.

51.18.2 Set Resizing Strategy action

This action specifies the OS resizing strategy of an rt-composite regarding the modification of the PVSs of the child rt-components positioned in the OS of the rt-composite.

This action has the following parameters:

- Set of Rt-Composite Target Param;
- Resizing Strategy Param.

This action may be targeted during the periods R2, R3 and R3.TD.

51.18.2.1 Set Resizing Strategy effect

The action effect is as follows:

- 1) depending on the resizing strategy parameter value (“fixed”, “minimum” or “grows only”), the OS of the target rt-composite may be changed;
- 2) if “minimum” or “grows only” is assigned, the OS of the rt-composite is recalculated each time the child PVSs are modified and each time the OS does not fit the specified strategy.

This effect continues until the target is destroyed or this action is targeted again with a different value.

51.18.2.2 Set Resizing Strategy additional error conditions

- If one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target.

51.18.3 Set OVS Proj Strategy action

This action set the OVS Proj Strategy to an rt-component.

This action has the following parameters:

- Set of Rt-Component Target Param;
- OVS Proj Strategy Param.

This action may be targeted during the periods R2, R3 and R3.TD.

51.18.3.1 Set OVS Proj Strategy effect

The effect of this action is as follows:

- 1) set the OVS Proj Strategy attribute to either “fixed” or “calculated” as specified;
- 2) if the OVS Proj Strategy attribute is set to “fixed”, the size specification parameter in the Set OVS action is interpreted as RGSU and the Set OVS action sets the PVS;
- 3) otherwise, the size specification parameter in the Set OVS action is interpreted as OGSU and the Set OVS action sets the OVS directly.

51.18.3.2 Set OVS Proj Strategy additional error conditions

- If one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target.

51.18.4 Set OVS action

This action sets the OVS directly or indirectly depending on the OVS Proj Strategy attribute of the target.

This action has the following parameters:

- Set of Rt-Component Target Param;
- Size Spec Param;
- Transition Duration Param.

This action may be targeted during the periods R2, R3 and R3.TD.

51.18.4.1 Set OVS action effect

The effect of this action is as follows:

- 1) The size specification parameter may be specified by an absolute value expressed in GSU or a relative value expressed in generic ratio.
- 2) If the OVS Proj Strategy attribute is equal to “fixed”:
 - a) The specified values in the size specification parameter are interpreted as in the RPS of the target. Absolute values are considered as expressed in RGSU. Relative values are considered as expressed relatively to the POS of the target.
 - b) The PVS of the target is directly defined by the interpreted values.
 - c) The OVS is automatically calculated by the current GSF of the target.
 - d) If the OVS or the GSU is changed after applying this action, the new GSU or the new OVS is recalculated and is set in order to hold $PVS = OVS \times GSU$ on each axis.
- 3) Otherwise:
 - a) The specified values in the size specification parameter are interpreted as in the OPS of the target. Absolute values are considered as expressed in OGSU. Logical values are considered as expressed relatively to the OS of the target.
 - b) The OVS of the target is directly defined by the interpreted values.
 - c) The PVS is automatically calculated by the current GSF of the target.
 - d) If the OVS or the GSU is changed after applying this action, the new PVS is recalculated in order to hold $PVS = OVS \times GSU$ on each axis.

If a size specification on an axis is omitted, 100% is applied.

If a non-zero transition duration is specified, the OVS or the PVS is gradually changed depending on the OVS Proj Strategy attribute. There is a user effect associated with this action, i.e. the presented content within the PVS is changed gradually or the PVS is changed gradually. The change of the OVS or the PVS does not affect the OAP, the OVS position, the PAP and the PVS position.

51.18.4.2 Set OVS additional error conditions

- if one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target;
- if a size specification on an axis is omitted, 100% is applied;
- if the size specification parameter value is a negative value, this action is ignored for the whole target set;
- if the transition duration is negative, it is set to zero;
- if an original point factor or a current point factor is used, this action is ignored for all the targets.

51.18.5 Set OAP action

This action sets the OAP of an rt-component. The OAP shall be specified within the OVS.

This action has the following parameters:

- Set of Rt-Component Target Param.
- OAP Param: May be specified as an absolute value or a relative value.

This action may be targeted during the periods R2, R3 and R3.TD.

51.18.5.1 Set OAP action effect

The OAP may be set to absolute values or relative values to the OVS depending on the OAP parameter.

There may be a user effect changing the presenting portion of the OS.

51.18.5.2 Set OAP additional error conditions

- if one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target;
- if one of the OAP parameters is specified as an absolute value and it is not in the interval (0, OVS) on an axis, this parameter is set to 0;
- if one of the OAP parameters is specified as a relative value and it is not in the interval (0, 100%) on an axis, this parameter is set to 0%.

51.18.6 Set OVS Position action

This action specifies the position of the OVS relatively to the OS.

This action has the following parameters:

- Set of Rt-Component Target Param.
- OVS Position Param: Allows the specification of a macro parameter instead of a specified position value for the OVS relatively to the OS.
- Transition Duration Param: Allows the specification of a macro parameter instead of a specified position value for the OVS relatively to the OS.

This action may be targeted during the periods R2, R3 and R3.TD.

51.18.6.1 Set OVS Position action effect

The OVS position may be set to absolute values in the OPS or relative values to the OS depending on the OAP parameter. The effect of this action is as follows:

- Set the OVS position of the target.

There may be a user effect changing the presenting portion of the OS.

If a non-zero transition duration is specified, the OVS position is gradually changed. There is a user effect associated with this action, i.e. the presenting part of the OS is gradually changed. The change of the OVS position does not affect the OAP.

NOTES

1 – Any values are allowed for the OVS position parameter in absolute values and relative values.

2 – If some portions of the OS are not within the OVS, they are not perceived by the user. Note that the OVS position may position the OVS completely outside the OS. In this case, the entire OS is not perceived.

51.18.6.2 Set OVS Position additional error conditions

- if one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target;
- if the transition duration is negative, it is set to zero.

51.18.7 Set PAP action

This action sets the PAP of an rt-component. The PAP shall be specified within the PVS.

This action has the following parameters:

- Set of Rt-Component Target Param.
- PAP Param: May be specified as an absolute value or a relative value.

This action may be targeted during the periods R2, R3 and R3.TD.

51.18.7.1 Set PAP action effect

The PAP may be set to absolute values or relative values to the PVS depending on the PAP parameter. The effect of this action is as follows.

- Set the PAP of the target.

There may be a user effect changing the presenting position within the RPS.

51.18.7.2 Set PAP additional error conditions

- if one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target;
- if one of the PAP parameters is specified as an absolute value and it is not in the interval (0, PVS) on an axis, this parameter is set to 0;
- if one of the PAP parameters is specified as a relative value and it is not in the interval (0, 100%) on an axis, this parameter is set to 0%.

51.18.8 Set PVS Position action

This action specifies the position of the PVS relatively to the OS of the RPS.

This action has the following parameters:

- Set of Rt-Component Target Param.
- PVS Position Param: May be specified as an absolute value or a relative value.
- Transition Duration Param: May be specified as an absolute value or a relative value.

This action may be targeted during the periods R2, R3 and R3.TD.

51.18.8.1 Set PVS Position action effect

The PVS position may be set to absolute values in the RPS or relative values to the OS of the RPS depending on the PAP parameter. The effect of this action is as follows:

- Set the PVS position of the target.

There may be a user effect changing the presenting position within the RPS.

If a non-zero transition duration is specified, the PVS position is gradually changed. There is a user effect associated with this action, i.e. the presenting position of the PVS is gradually changed. The change of the PVS position does not affect the PAP.

Any values are allowed for the PVS position parameter in absolute values and relative values.

51.18.8.2 Set PVS Position additional error conditions

- if one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target;
- if the transition duration is negative, it is set to zero.

51.18.9 Set GSF action

This action sets the GSF of the target.

This action has the following parameters:

- Set of Rt-Component Target Param.
- GSF Param: a generic ratio.
- Transition Duration Param: A generic ratio.

This action may be targeted during the periods R2, R3 and R3.TD.

51.18.9.1 Set GSF action effect

If the target is running, there may be a user effect. The presenting rt-component may be changed in its size.

If a non-zero transition duration is specified, the size of the target is gradually changing up to the specified GSF.

51.18.9.2 Set GSF additional error conditions

- if one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target;
- if the GSF parameter value is a negative value, it is set to the value 0;
- if the transition duration is negative, it is set to zero.

51.18.10 Set User Spatial Control action

This action specifies whether the user is able to control some rt-component objects according to the specified functions (see 11.3).

This action has the following parameters:

- Set of Rt-Component Target Param.
- Set of Spatial Control Param: A generic ratio.
- User Spatial Control Param.

This action may be targeted during the periods R2, R3 and R3.TD.

51.18.10.1 Set User Spatial Control action effect

The effect of this action is as follows:

- Specified spatial control parameters are set to “allowed” or “not allowed” according to the user spatial control parameter.

If one of the spatial control functions is allowed, the user may interact the rt-component to perform the function.

51.18.10.2 Set User Spatial Control additional error conditions

- if one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target;
- if the target is not an rt-component, the action is ignored for that target.

51.19 Actions to retrieve the behaviour

This Recommendation defines means to get this behaviour by the use of the following actions:

- Get OS action (see 51.19.1);
- Get POS action (see 51.19.2);
- Get Aspect Ratio action (see 51.19.3);
- Get Resizing Strategy action (see 51.19.4);
- Get OVS Proj Strategy action (see 51.19.5);
- Get OVS action (see 51.19.6);
- Get OAP action (see 51.19.7);
- Get OVS Position action (see 51.19.8);
- Get PVS action (see 51.19.9);
- Get PAP action (see 51.19.10);
- Get PVS Position action (see 51.19.11);
- Get GSF action (see 51.19.12);
- Get User Spatial Control action (see 51.19.13).

51.19.1 Get OS action

This action retrieves the OS of the target.

This action has the following parameters:

- Rt-Component Target Param;
- Expected Axis Result Param.

This action may be targeted during the periods R2, R3 and R3.TD.

51.19.1.1 Get OS action effect

The effect of this action is as follows:

- 1) if the expected axis result parameter is “x”, “y” or “z”, the corresponding size of the OS on that axis is returned as a generic integer;
- 2) otherwise (“xyz”), all the sizes of the OS are returned as a generic list containing three generic integers.

The return value is measured in the OGSU of the target.

51.19.1.2 Get OS additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if the transition duration is negative, it is set to zero.

51.19.2 Get POS action

This action retrieves the POS of the target.

This action has the following parameters:

- Rt-Component Target Param;
- Expected Axis Result Param.

This action may be targeted during the periods R2, R3 and R3.TD.

51.19.2.1 Get POS action effect

The effect of this action is as follows:

- 1) if the expected axis result parameter is “x”, “y” or “z”, the corresponding size of the POS on that axis is returned as a generic integer;
- 2) otherwise (“xyz”), all the sizes of the POS are returned as a generic list containing three generic integers.

The return value is measured in the RGSU of the target.

51.19.2.2 Get POS additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if the transition duration is negative, it is set to zero.

51.19.3 Get Aspect Ratio action

This action retrieves the aspect ratio attribute of the target.

This action has the following parameter:

- Rt-Component Target Param.

This action may be targeted during the periods R2, R3 and R3.TD.

51.19.3.1 Get Aspect Ratio action effect

This action evaluates the aspect ratio attribute of the target and returns it.

51.19.3.2 Get Aspect Ratio additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if the transition duration is negative, it is set to zero.

51.19.4 Get Resizing Strategy action

This action retrieves the resizing strategy attribute of the target.

This action has the following parameter:

- Rt-Composite Target Param.

This action may be targeted during the periods R2, R3 and R3.TD.

51.19.4.1 Get Resizing Strategy action effect

This action evaluates the resizing strategy attribute of the target and returns it.

51.19.4.2 Get Resizing Strategy additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if the transition duration is negative, it is set to zero.

51.19.5 Get OVS Proj Strategy action

This action retrieves the OVS Proj Strategy attribute of the target.

This action has the following parameter:

- Rt-Component Target Param.

This action may be targeted during the periods R2, R3 and R3.TD.

51.19.5.1 Get OVS Proj Strategy action effect

This action evaluates the OVS Proj Strategy attribute of the target and returns it.

51.19.5.2 Get OVS Proj Strategy additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if the transition duration is negative, it is set to zero.

51.19.6 Get OVS action

This action retrieves the OVS of the target.

This action has the following parameters:

- Rt-Component Target Param;
- Point Type Param;
- Expected Axis Result Param.

This action may be targeted during the periods R2, R3 and R3.TD.

51.19.6.1 Get OVS action effect

The effect of this action is as follows:

- 1) if the point type parameter is “absolute”, this action evaluates the OVS as absolute values in the OGSU, and returns generic integers;
- 2) otherwise (“relative”), this action evaluates the OVS as relative values relatively to the corresponding OS, and returns generic ratios;
- 3) if the expected axis result parameter is “x”, “y” or “z”, the corresponding size of the OVS on that axis is returned;
- 4) otherwise (“xyz”), all the sizes of the OVS are returned as a generic list containing three elements.

51.19.6.2 Get OVS additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if the transition duration is negative, it is set to zero.

51.19.7 Get OAP action

This action retrieves the OAP of the target.

This action has the following parameters:

- Rt-Component Target Param;
- Point Type Param;
- Expected Axis Result Param.

This action may be targeted during the periods R2, R3 and R3.TD.

51.19.7.1 Get OAP action effect

The effect of this action is as follows:

- 1) if the point type parameter is “absolute”, this action evaluates the OAP as absolute values in the OGSU within the OVS and returns generic integers;
- 2) otherwise (“relative”), this action evaluates the OAP as relative values relatively to the corresponding OVS and returns generic ratios;
- 3) if the expected axis result parameter is “x”, “y” or “z”, the corresponding point of the OAP on that axis is returned;
- 4) otherwise (“xyz”), all the points of the OAP are returned as a list containing three elements.

51.19.7.2 Get OAP additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if the transition duration is negative, it is set to zero.

51.19.8 Get OVS Position action

This action retrieves the OVS position of the target.

This action has the following parameters:

- Rt-Component Target Param;
- Point Type Param;
- Expected Axis Result Param.

This action may be targeted during the periods R2, R3 and R3.TD.

51.19.8.1 Get OVS Position action effect

The effect of this action is as follows:

- 1) if the point type parameter is “absolute”, this action evaluates the OVS position as absolute values in the OGSU and returns generic integers;
- 2) otherwise (“relative”), this action evaluates the OVS position as relative values relatively to the corresponding OS and returns generic ratios;
- 3) if the expected axis result parameter is “x”, “y” or “z”, the corresponding length of the OVS position on that axis is returned;
- 4) otherwise (“xyz”), all the lengths of the OVS position are returned as a list containing three elements.

51.19.8.2 Get OVS Position additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if the transition duration is negative, it is set to zero.

51.19.9 Get PVS action

This action retrieves the PVS of the target.

This action has the following parameters:

- Rt-Component Target Param;
- Point Type Param;
- Expected Axis Result Param.

This action may be targeted during the periods R2, R3 and R3.TD.

51.19.9.1 Get PVS action effect

The effect of this action is as follows:

- 1) if the point type parameter is “absolute”, this action evaluates the PVS as absolute values in the RGSU of the target and returns generic integers;
- 2) otherwise (“relative”), this action evaluates the PVS as relative values relatively to the OS of the parent and returns generic ratios;
- 3) if the expected axis result parameter is “x”, “y” or “z”, the corresponding size of the PVS on that axis is returned;
- 4) otherwise (“xyz”), all the sizes of the PVS are returned as a list containing three elements.

51.19.9.2 Get PVS additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if the transition duration is negative, it is set to zero.

51.19.10 Get PAP action

This action retrieves the PAP of the target.

This action has the following parameters:

- Rt-Component Target Param;
- Point Type Param;
- Expected Axis Result Param.

This action may be targeted during the periods R2, R3 and R3.TD.

51.19.10.1 Get PAP action effect

The effect of this action is as follows:

- 1) if the point type parameter is “absolute”, this action evaluates the PAP as absolute values in the RGSU of the target within the PVS and returns generic integers;
- 2) otherwise (“relative”), this action evaluates the PAP as relative values relatively to the corresponding PVS and returns generic ratios;
- 3) if the expected axis result parameter is “x”, “y” or “z”, the corresponding point of the PAP on that axis is returned;
- 4) otherwise (“xyz”), all the points of the PAP are returned as a list containing three elements.

51.19.10.2 Get PAP additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if the transition duration is negative, it is set to zero.

51.19.11 Get PVS Position action

This action retrieves the PVS position of the target.

This action has the following parameters:

- Rt-Component Target Param;
- Point Type Param;
- Expected Axis Result Param.

This action may be targeted during the periods R2, R3 and R3.TD.

51.19.11.1 Get PVS Position action effect

The effect of this action is as follows:

- 1) if the point type parameter is “absolute”, this action evaluates the PVS position as absolute values in the RGSU of the target and returns generic integers;
- 2) otherwise (“relative”), this action evaluates the PVS position as relative values relatively to the OS of the parent and returns generic ratios;
- 3) if the expected axis result parameter is “x”, “y” or “z”, the corresponding length of the PVS position on that axis is returned;
- 4) otherwise (“xyz”), all the lengths of the PVS position are returned as a list containing three elements.

51.19.11.2 Get PVS Position additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if the transition duration is negative, it is set to zero.

51.19.12 Get GSF action

This action retrieves the GSF of the target.

This action has the following parameters:

- Rt-Component Target Param;
- Expected Axis Result Param.

This action may be targeted during the periods R2, R3 and R3.TD.

51.19.12.1 Get GSF action effect

The effect of this action is as follows:

- 1) if the expected axis result parameter is “x”, “y” or “z”, the corresponding GSF of the target on that axis is returned as a generic integer;
- 2) otherwise (“xyz”), all the GSFs of the target are returned as a list containing three generic integers.

51.19.12.2 Get GSF additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if the transition duration is negative, it is set to zero.

51.19.13 Get User Spatial Control action

This action retrieves the user spatial control attribute value.

This action has the following parameters:

- Rt-Component Target Param;
- Spatial Control Param.

This action may be targeted during the periods R2, R3 and R3.TD.

51.19.13.1 Get User Spatial Control effect

The effect of this action is as follows:

- Depending on the specified spatial control parameter (“moving”, “resizing”, “scaling” or “scrolling”), this action returns “allowed” or “not allowed”.

51.19.13.2 Get User Spatial Control additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if the transition duration is negative, it is set to zero.

52 Rt-Components Audible Behaviour

The behaviour describes the audible presentation of rt-components. This Recommendation defines an initial value of this behaviour and means to modify it and to retrieve its value by using actions.

52.1 Behaviour attributes and statuses

This behaviour is expressed using the following attributes or statuses:

- OV (see 52.2);
- CV (see 52.3);
- PCV (see 52.4);
- GVF (see 52.5).

52.2 OV

Each rt-content has an OV that is defined within the AVR. However, there is no OV for an rt-composite. The OV is always measured in the GVU, and shall be in this range (see “Original Presentation Space” for initialisation of each OV).

Each OV for an rt-content is set only once and shall not be changed.

52.3 CV

Each rt-content has a CV that is initialised as the OV. The CV may be changed by using the Set CV action; however, the CV shall be within the AVR.

52.4 PCV

The CV is projected to its RPS. That is called the PCV. The PCV is calculated by the GVF belonging to its RPS (PRPS or CPS) as $CV \times GVF$.

NOTE – Each rt-component (rt-content and rt-composite) has a GTF and a GSF. However, rt-contents do not have GVF. Rt-Composites and channels have GVF. Instead of having a GVF, an rt-content has a CV that may be changed.

52.5 GVF

The projection of the CV in the OPS to the PCV in the RPS is controlled by the GVF belonging to the RPS. The GVF defines the number of RGVU corresponding to one OGVU Original presentation space.

52.6 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following actions:

- Set CV action (see 52.6.1),
- Set GVF action (see 52.6.2).

52.6.1 Set CV action

This action sets the CV of the target as specified value.

This action has the following parameters:

- Set of Rt-Content Target Param.
- CV Param: Specified as an absolute value, a relative value, an original point factor or a current point factor.
- Transition Duration Param: Specified as an absolute value, a relative value, an original point factor or a current point factor.

This action may be targeted during the periods R2, R3 and R3.TD.

52.6.1.1 Set CV action effect

This effect of this action is as follows:

- 1) If the CV is specified as an absolute value, this value is set to the CV of the target.
- 2) If the CV is specified as a relative value, this value is interpreted as a relative value to the AVR (0, 255) and the audible volume is calculated by $255 \times (\text{specified value})$. The calculated value is set to the CV of the target.
- 3) If the CV is specified as an original point factor, the value, $OV \times (\text{specified value})$, is calculated and set to the CV of the target.
- 4) Otherwise (the CV is specified as a current point factor), the value, $(\text{current CV}) \times (\text{specified value})$, is calculated and set to the CV.
- 5) In above case, if the calculated value is outside the AVR, the value is set to 0 (if negative) or 255 (if greater than 255).

If a non-zero transition duration is specified, the CV is gradually changed to the specified value.

52.6.1.2 Set CV additional error conditions

- if one of the targets is not an rt-content in R2, R3 or R3.TD, this action is ignored for that target;
- if the CV parameter defined as an absolute parameter value is not in interval of the AVR, this action is ignored for the whole target set;
- if the transition duration is negative, it is set to zero.

52.6.2 Set GVF action

This action specifies the GVF of an rt-composite or a channel.

This action has the following parameters:

- GVF Target.
- GVF Param: GVF value to be assigned.
- Transition Duration Param: GVF value to be assigned.

This action may be targeted during the periods R2, R3, R3.TD and C2.

52.6.2.1 Set GVF action effect

The effect of this action is as follows:

- 1) the GVF of the target is set to the specified value;
- 2) if the target is an rt-composite, all the PCVs of descendants are recalculated by this value;
- 3) otherwise (the target is a channel), all the PCVs of rt-components assigned to this channel are recalculated by this value.

If a non-zero transition duration is specified, the PCVs of the targets are gradually changed.

52.6.2.2 Set GVF additional error conditions

- if one of the targets is not an rt-composite or a channel in R2, R3, R3.TD or C2, this action is ignored for that target;
- if the transition duration is negative, it is set to zero.

52.7 Actions to retrieve the behaviour

This Recommendation defines means to get this behaviour by the use of the following actions:

- Get OV action (see 52.7.1);
- Get CV action (see 52.7.2);
- Get PCV action (see 52.7.3);
- Get GVF action (see 52.7.4).

52.7.1 Get OV action

This action retrieves the OV of the target.

This action has the following parameter:

- Rt-Content Target Param.

This action may be targeted during the periods R2, R3 and R3.TD.

52.7.1.1 Get OV action effect

The effect of this action is as follows:

- Retrieve the OV of the target expressed in the OGVU.

52.7.1.2 Get OV additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if one of the targets is not an rt-content in R2, R3 or R3.TD, this action is ignored for that target.

52.7.2 Get CV action

This action retrieves the CV of the target.

This action has the following parameter:

- Rt-Content Target Param.

This action may be targeted during the periods R2, R3 and R3.TD.

52.7.2.1 Get CV action effect

The effect of this action is as follows:

- Retrieve the CV of the target expressed in the OGVU.

52.7.2.2 Get CV additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if one of the targets is not an rt-content in R2, R3 or R3.TD, this action is ignored for that target.

52.7.3 Get PCV action

This action retrieves the PCV of the target.

This action has the following parameter:

- Rt-Content Target Param.

This action may be targeted during the periods R2, R3 and R3.TD.

52.7.3.1 Get PCV action effect

The effect of this action is as follows:

- Retrieve the PCV of the target expressed in the RGVU.

52.7.3.2 Get PCV additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if one of the targets is not an rt-content in R2, R3 or R3.TD, this action is ignored for that target.

52.7.4 Get GVF action

This action retrieves the GVF of the target.

This action has the following parameters:

- Rt-Composite Target Param;
- Channel Target Param.

This action may be targeted during the periods R2, R3 or R3.TD and C2.

52.7.4.1 Get GVF action effect

The effect of this action is as follows:

- Retrieve the GVF of the target as a generic ratio.

52.7.4.2 Get GVF additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if one of the targets is not an rt-component or a channel in R2, R3, R3.TD or C2, this action is ignored for that target.

53 Rt-Mux Stream Choice Behaviour

This behaviour describes the choices of the stream within an rt-mux. Once streams are chosen, the rt-mux is responsible for these chosen streams and ensure their presentation.

53.1 Behaviour attributes and statuses

This behaviour is expressed using the following attributes or statuses:

- Stream Choice (see 53.2),
- Stream Chosen State (see 53.3).

53.2 Stream Choice

An rt-mux is responsible for the complete multiplexed data presentation or some specified streams presentation individually chosen within the multiplexed data.

The stream choice attribute holds the chosen stream IDs as a list.

Initially, all streams are chosen for an rt-mux, i.e. the rt-mux is responsible for the presentation of the complete multiplexed data. Thus, this attribute holds every stream ID.

Once several streams are chosen for an rt-mux, it is responsible for the presentation of these chosen streams when it becomes “running”. In this case, typically the demultiplexing process is pipelined with the presentation of them, and therefore no new data is generated from the demultiplexing.

53.3 Stream Chosen State

Each stream in an rt-mux has a stream chosen state. If it is chosen, the state is “chosen”. Otherwise, the state is “not chosen”.

Each rt-mux also has a stream chosen state. If all the streams of it are chosen, the state is “chosen”. If some streams of it are chosen, the state is “partially chosen”. Otherwise (no streams are chosen), the state is “not chosen”.

53.4 Stream Identification

Choice between a stream identifier or “all streams”.

53.5 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following action:

- Set Stream Choice action (see 53.5.1).

53.5.1 Set Stream Choice action

The action chooses the streams that are to be presented within a target rt-mux.

This action has the following parameters:

- Set of Rt-Mux Target Param.
- Set of Stream Spec Param: GVF value to be assigned.

This action may be targeted during the periods R2, R3 and R3.TD.

53.5.1.1 Set Stream Choice action effect

The effect of this action is as follows:

- 1) according to the specified update command, streams are added, removed or replaced with a new set;
- 2) if a “remove” command is used and no streams are specified, all streams currently chosen are changed to “not chosen”.

The result of this action is to change stream chosen states of some streams and the rt-mux. There may be a user effect if the target is running.

53.5.1.2 Set Stream Choice additional error conditions

- If one of the targets is not an rt-mux in <validperiod>, this action is ignored for that target.

53.6 Actions to retrieve the behaviour

This Recommendation defines means to get this behaviour by the use of the following actions:

- Get Stream Choice action (see 53.6.1),
- Get Stream Chosen State action (see 53.6.2).

53.6.1 Get Stream Choice action

This action retrieves a list of chosen stream IDs of the target.

This action has the following parameter:

- Rt-Mux Target Param.

This action may be targeted during the periods R2, R3 and R3.TD.

53.6.1.1 Get Stream Choice action effect

The effect of this action is as follows:

- A list of chosen stream IDs are retrieved as a generic list.

53.6.1.2 Get Stream Choice additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if one of the targets is not an rt-mux in R2, R3 or R3.TD, this action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target.

53.6.2 Get Stream Chosen State action

This action retrieves the stream state of the target.

This action has the following parameters:

- Rt-Mux Target Param.
- Stream Identification Param: GVF value to be assigned.

This action may be targeted during the periods R2, R3 and R3.TD.

53.6.2.1 Get Stream Chosen State action effect

The effect of this action is as follows:

- 1) if a stream ID is specified as a value of the stream identifier parameter, the stream state of the stream is retrieved;
- 2) otherwise (i.e. “all streams” is specified as a value of the stream identifier parameter), the stream state of the target rt-mux is retrieved.

53.6.2.2 Get Stream Chosen State additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if one of the targets is not an rt-mux in R2, R3 or R3.TD, this action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target.

54 Interaction Behaviour

The behaviour describes the interaction between rt-components and the user. Through this behaviour, the MHEG engine and MHEG entities interactive to the user. All rt-components may be interactive. Initially, each rt-component is not interactive.

Through the event behaviour, the MHEG engine and the MHEG entities are also interact to the user. However, this interaction behaviour is dedicated especially to user’s selection and modification that are frequently used.

54.1 Behaviour attributes and statuses

This behaviour is expressed using the following attributes or statuses:

- Interaction Type (see 54.2),
- Interaction Status (see 54.3),
- Selection Status (see 54.4),
- Modification Status (see 54.5),
- Interaction Ability (see 54.6),
- Selectability (see 54.7),
- Modifiability (see 54.8),
- Min Interact Required (see 54.9),
- Max Interact Required (see 54.10),
- Number of Interacted Sockets (see 54.11).

54.2 Interaction Type

There are two interaction types for rt-components. One is “selection” concerning the user selection, and the other one is “modification” concerning the user modification. The two selection types are independent of each other so that one rt-component may have one of the interaction types of both simultaneously.

The MHEG engine should manage the selection and the modification of rt-components with the help of the user interface. By the value of following interaction attributes, the MHEG engine is able to know the user interactions to rt-components.

The user may interact to rt-components only if they are running. If an rt-component is not running, the user cannot interact to it and any states concerning the interaction behaviour shall not be changed.

The two types of interactions are controlled and retrieved by the same set of elementary actions and get actions specifying the interaction type as a parameter.

54.3 Interaction Status

Each rt-component may have the selection status indicating the user selection on it, and each rt-component may also have the modification status indicating the user modification to it.

The interaction status of an rt-component is set only if it is “running”. The interaction status is set explicitly by a Set Interaction Status action or implicitly by the user interaction.

How to interact to rt-components is up to the MHEG engine and the user interface.

NOTES

1 – Rt-contents may be selected by a mouse positioning and click, and may be modified by a mouse positioning and keyboard input.

2 – Not only the visible rt-contents but also some audible rt-contents and multiplexed contents may be selected or modified. How to do that depends on the MHEG engine implementation and the using application.

The interaction status may be perceptible (visibly or audibly) to show the current status of an rt-component to the user. However, it is an MHEG engine feature and this Recommendation does not specify any methods for that.

54.4 Selection Status

The selection status is “selected” or “not selected”. It indicates the results of the user selection.

Initially, each rt-component is “not selected”.

54.5 Modification Status

The modification status is “modified”, “not modified” or “modifying”. It indicates the results of the user modification.

Initially, each rt-component is “not modified”.

54.6 Interaction Ability

The two interaction types are allowed individually to assign each rt-component. The allowance of the selection is called selectability, and the allowance of the modification is called modifiability.

The selectability tells the MHEG engine whether this rt-component is an item for selection. And the modifiability tells the MHEG engine whether this rt-component is an item for modification. If an rt-component is allowed for selection and modification, the user may select or modify it.

The interaction ability may be perceptible (visibly or audibly) to show the interaction possibility of an rt-component to the user. However, it is the MHEG engine feature and this Recommendation does not specify any methods for that.

54.7 Selectability

Each rt-component has the selectability that is “selectable” or “not selectable”. This attribute is indirectly specified by a value but a value of the maximum interaction required attribute for the selection.

The selectability attribute is calculated as follows:

- If the maximum interaction required attribute for the selection is 0, it is “not selectable”.

Otherwise, it is “selectable”.

If an rt-component is “selectable”, the user may interact to it and may select it. Otherwise, the user selection is inhibited.

54.8 Modifiability

Each rt-component has the modifiability that is “modifiable” or “not modifiable”. This attribute is not directly specified by a value but a value of the maximum interaction required attribute value for the modification.

The modifiability attribute is calculated as follows:

- If the maximum interaction required attribute for the modification is 0, it is “not modifiable”

Otherwise, it is “modifiable”.

If an rt-component is “modifiable”, the user may interact to it and may modify it. Otherwise, the user modification is inhibited.

54.9 Min Interact Required

The selection and modification status for an rt-content is directly determined by a user interaction because the user can directly select and modify the rt-content. However, the selection and modification status for an rt-composite is not directly determined by a user interaction. The selection and modification status for an rt-composite are dynamically determined as follows:

- If a Set Interaction Status action is targeted to an rt-composite, the rt-composite is set to the specified status regardless of the min interact required attribute, the max interact required attribute and some possible “modifying” child rt-components.

NOTE 1 – It is an author’s intention whether to keep consistency among those attribute values if an explicit Set Interaction Status action is used.

If one of the rt-components in an rt-composite is “modifying”, the rt-composite is determined as “modifying”.

If a number of selected or modified rt-components in this rt-composite is greater than or equal to the min interact required attribute value, and if it is less than or equal to the max interact required attribute value, and if the user interaction process is done, then the rt-composite is determined as “selected” or “modified”.

NOTE 2 – The MHEG engine should provide some mechanism that checks the end of user interaction process. It is not an automatic process to determine the interaction status. The user interaction may continue until the number of selected or modified rt-components becomes the value of the max interact required attribute even if the number is greater than the min interact required attribute value.

Otherwise, the rt-composite is “not selected” or “not modified”.

Therefore, the min interact required and max interact required attributes are provided for the MHEG engine as a hint to determine the interaction status of rt-composite.

The number of interacted rt-components in an rt-composite may be changed dynamically due to a Plug action. It is the MHEG engine responsibility to keep the consistency among the children statuses, the parent status, the min interact required and the max interact required attributes for the parent.

NOTE 3 – For example, if a selected rt-component is plugged into an rt-composite that has the max interact required attribute 5 and the number of selected sockets 6, the MHEG engine starts the user interaction process and asks the user to deselect one among the selected 6 sockets in order to satisfy the max interact required condition.

NOTE 4 – For example, if the min interact required attribute is 2 and one of the selected sockets is unplugged, the MHEG engine starts the user interaction process and asks the user to select at least one more socket.

NOTE 5 – For example, if the min interact required attribute is 5 and there are 4 sockets plugged into an rt-composite, the MHEG engine continues the user interaction process until some other rt-components are plugged in and at least 5 sockets are selected.

NOTE 6 – If an rt-composite is not running, some inconsistency among those attributes may exist. Because the user may interact only if the rt-composite is running, the MHEG engine may ask the user later again to interact for that rt-composite if another Run action is targeted. Suppose that an rt-composite with one minute duration and “stop” for the temporal termination attribute is running and its min interact required is set to 1. In this case, if the user does not interact to the sockets of the rt-composite while it is running, the number of interacted sockets are less than the min interact required. If it is set to “running” again, the user interaction process begins again. This may be considered as a timer mechanism of the user interaction.

The min interact required attribute is valid only for rt-composites. The value for an rt-content is always ignored and has no meaning.

If the value of the min interact required attribute of an rt-composite is greater than or equal to 1, it means at least one interacted child is needed to change the interaction status of the parent rt-composite.

Initially, the attribute is set to 0.

54.10 Max Interact Required

The max interact required attribute is used to determine the interaction ability of an rt-component.

The attribute for an rt-composite also gives a hint to the MHEG engine to control the number of interacted children. If the value of the max interact required attribute is greater than or equal to 1, it means that one interacted child is the maximum number of interacted children allowed for the rt-composite.

The value of the max interact required shall be greater than or equal to the value of the min interact required for each interaction type.

The MHEG engine shall prevent the number of interacted rt-components greater than the specified value of the max interact required.

Initially, the attribute is set to 0.

54.11 Number of Interacted Sockets

The number of interacted children may be evaluated. The number of interacted sockets attribute stores the value, and it may be retrieved by a Get Number of Interacted Sockets action.

Initially, no sockets are interacted. Therefore, the attribute is set to 0.

54.12 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following actions:

- Set Interaction Ability action (see 54.12.1),
- Set Interaction Status action (see 54.12.2).

54.12.1 Set Interaction Ability action

This action sets the interaction ability to the target.

This action has the following parameters:

- Set of Rt-Component Target Param.
- Interaction Type Param: GVF value to be assigned.
- Min Interact Required Param: GVF value to be assigned.
- Max Interact Required Param: GVF value to be assigned.

This action may be targeted during the periods R2, R3 and R3.TD.

54.12.1.1 Set Interaction Ability action effect

According to the specified interaction type, the selection or the modification of the target is set. The effect of this action is as follows:

- 1) the min interact required attribute is set to the specified value;
- 2) the max interact required attribute is set to the specified value.

The min interact required attribute value has no meaning for rt-contents.

54.12.1.2 Set Interaction Ability additional error conditions

- if one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target;
- if the min interact required parameter value is a negative value, it is set to the value 0;
- if the max interact required parameter value is a negative value, it is set to the value 0;
- if the min interact required is greater than the max interact required, this action is ignored for the whole target set;
- if the min interact required parameter value is not in interval (0, number of child sockets of the target), this action is ignored for the whole target set;
- if the max interact required parameter value is not in interval (0, number of child sockets of the target), this action is ignored for the whole target set;
- if the current number of interacted sockets is greater than the specified max interact required, this action is ignored for that target.

54.12.2 Set Interaction Status action

This action sets the interaction status of the target.

This action has the following parameters:

- Set of Rt-Component Target Param.
- Interaction Type Param: GVF value to be assigned.
- Interaction Status Param: GVF value to be assigned.

This action may be targeted during the periods R3 and R3.TD.

54.12.2.1 Set Interaction Status action effect

The effect of this action is to set the interaction status according to the specified interaction type as follows:

- 1) if the specified interaction type is “selection”, the interaction status of the target is set to either “selected” or “not selected”;
- 2) otherwise (the specified interaction type is “modification”), the interaction status of the target is set to either “modified”, “not modified” or “modifying”.

This action sets the interaction status regardless of any other related attribute values. It is for the author to decide whether to keep consistency among attribute values or not.

54.12.2.2 Set Interaction Status additional error conditions

- If one of the targets is not an rt-component in R3 or R3.TD, this action is ignored for that target.

54.13 Actions to retrieve the behaviour

This Recommendation defines means to get this behaviour by the use of the following actions:

- Get Interaction Ability action (see 54.13.1),
- Get Min Interact Required action (see 54.13.2),
- Get Max Interact Required action (see 54.13.3),
- Get Interaction Status action (see 54.13.4),
- Get Number of Interacted Sockets action (see 54.13.5).

54.13.1 Get Interaction Ability action

This action retrieves the interaction ability of the target.

This action has the following parameters:

- Rt-Component Target Param.
- Interaction Type Param: GVF value to be assigned.

This action may be targeted during the periods R2, R3 and R3.TD.

54.13.1.1 Get Interaction Ability action effect

According to the specified interaction type, the effect of this action is as follows:

- 1) if the interaction type is “selection”, this action returns “selectable” or “not selectable”;
- 2) otherwise (the interaction type is “modification”), this action returns “modifiable” or “not modifiable”.

54.13.1.2 Get Interaction Ability additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not an rt-component, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if one of the targets is not rt-component in R2, R3, R3.TD, this action is ignored for that target.

54.13.2 Get Min Interact Required action

This action retrieves the min interact required of the target.

This action has the following parameters:

- Rt-Component Target Param.
- Interaction Type Param: GVF value to be assigned.

This action may be targeted during the periods R2, R3 and R3.TD.

54.13.2.1 Get Min Interact Required action effect

According to the specified interaction type, the value of the min interact required attribute for the selection or the modification is retrieved.

54.13.2.2 Get Min Interact Required additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not an rt-component, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if one of the targets is not rt-component in R2, R3 or R3.TD, this action is ignored for that target.

54.13.3 Get Max Interact Required action

This action retrieves the max interact required of the target.

This action has the following parameters:

- Rt-Component Target Param.
- Interaction Type Param: GVF value to be assigned.

This action may be targeted during the periods R2, R3 and R3.TD.

54.13.3.1 Get Max Interact Required action effect

According to the specified interaction type, the value of the max interact required attribute for the selection or the modification is retrieved.

54.13.3.2 Get Max Interact Required additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not an rt-component, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if one of the targets is not rt-component in R2, R3 or R3.TD, this action is ignored for that target.

54.13.4 Get Interaction Status action

This action retrieves the interaction status of the target.

This action has the following parameters:

- Rt-Component Target Param.
- Interaction Type Param: GVF value to be assigned.

This action may be targeted during the periods R2, R3 and R3.TD.

54.13.4.1 Get Interaction Status action effect

According to the specified interaction type, the effect of this action is as follows:

- 1) if the interaction type is “selection”, the selection status of the target is returned;
- 2) otherwise (the interaction type is “modification”), the modification status of the target is returned.

54.13.4.2 Get Interaction Status additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not an rt-component, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if one of the targets is not rt-component in R2, R3 or R3.TD, this action is ignored for that target.

54.13.5 Get Number of Interacted Sockets action

This action retrieves the number of interacted sockets of the target.

This action has the following parameters:

- Rt-Composite Target Param.
- Interaction Type Param: GVF value to be assigned.

This action may be targeted during the periods R2, R3 and R3.TD.

54.13.5.1 Get Number of Interacted Sockets action effect

According to the specified interaction type, the number of selected or modified sockets is retrieved.

54.13.5.2 Get Number of Interacted Sockets additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not an rt-component, the action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target;
- if one of the targets is not rt-composite in R2, R3 or R3.TD, this action is ignored for that target.

55 Rt-Components Style Behaviour

This behaviour describes the style of rt-component. Each rt-component may have a style. A style may change the user perception of rt-contents.

A style is a way of presentation, and may be compared to a predefined behaviour or style of presentation. Applying a style to a target avoids the author describing this behaviour or presentation. The MHEG engine uses the information described in the style to directly map a style to a facility provided by the user interface if possible. If the user interface does not provide such a facility, the MHEG engine is able to interpret the style with the description of the predefined behaviour provided for each style.

The following are some examples of the style behaviour:

- an rt-non-mux containing a generic integer may be displayed as a graphical slider with the style of “slider”;
- an rt-non-mux containing audio may be listened to through a speaker as audio, or may be displayed as wave in graphical way.

55.1 Behaviour attributes and statuses

This behaviour is expressed using the following attribute or status:

- Style (see 55.2).

55.2 Style

A style is a catalogued entry, either registered or proprietary. This attribute gives a logical style to the rt-component. How the style is perceived by the user depends on the MHEG engine and the GUI.

55.3 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following action:

- Set Style action (see 55.3.1).

55.3.1 Set Style action

This action assigns a style to the target.

This action has the following parameters:

- Set of Rt-Component Target Param.
- Catalogued Style Param.
- Additional Information Param: It may be used by a specific style to interchange some additional information to be associated with, for example, a value range for a slider style. The number of generic values required and the semantics of those generic values, depend on a style.

This action may be targeted during the periods R2, R3 and R3.TD.

55.3.1.1 Set Style action effect

The effect of this action is as follows:

- Depending on the specified style param and the additional information param, a style is assigned to the target.
- The previous style applied to the target is removed.

If a Run action is targeted to the target after applying this action, the target is presented with the current specified style. If the target is running at the time of applying this action, there may be a user effect associated with this action.

NOTE – The MHEG engine may directly map the target on a style facility provided by its user interface. However, certain attributes and statuses may be used in link objects, e.g. the selection status of the target, the position and size of the target. They have to be updated by the MHEG engine according to the style and coherently with the current user interaction.

55.3.1.2 Set Style additional error conditions

- If one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target.

55.4 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following actions:

- Get Style action (see 55.4.1).

55.4.1 Get Style action

This action retrieves the current style associated with the target.

This action has the following parameter:

- Rt-Component Target Param.

This action may be targeted during the periods R2, R3 and R3.TD.

55.4.1.1 Get Style action effect

The effect of this action is as follows:

- This action evaluates a list corresponding to the catalogued styles that are applied to the target.

55.4.1.2 Get Style additional error conditions

- If one of the targets is not an rt-component in R2, R3 or R3.TD, this action is ignored for that target.

56 Rt-Contents Anchor Behaviour

This behaviour describes the anchor function of an rt-content. An anchor is an rt-content created from a content object, which contains anchor information.

An anchor is characterised as follows:

- An anchor is distinguished from a normal rt-content by the optional data classification attribute of the content object. If this classification attribute is “Anchor”, then this content object has anchor information inside, and rt-contents created from this content object may be used as anchors.
- An anchor may be used to create a hotspot on another rt-component. The content data contained in an anchor is used to attach the anchor to another rt-content, e.g. starting point and end point, rectangular area, polygon area. This content data is called anchor information. If an rt-content to which an anchor is attached is modified in its presentation (e.g. resize, move), the anchor should follow its modification and change its position or shape automatically. The semantics of this anchor information is not defined by this Recommendation. It depends on rt-contents to which anchors are attached, the MHEG engine or the application.
- The hook in an anchor is used to indicate the semantics of anchor information. It provides an encoding format which is used to describe the rendering of the anchor, the attachment points in the rt-content which the anchor is attached to, and the semantic of the anchor. Since the encoding format is registered, the use of the hook is the way to have a generic implementation of anchor mechanism.
- The anchor information is used by the GUI or some other system that supports the rendering of rt-contents, and appropriate rendering of an anchor is to be done by it.
- As an anchor is an rt-content, its selection status tells the MHEG engine that the hotspot specified by this anchor is selected by a user action. A Run action or a Stop action may make an anchor active or inactive. A Set Interaction Ability action may also make an anchor active or inactive.
- An anchor has preparation behaviour, rt-availability behaviour, running behaviour, Interaction behaviour and rt-contents anchor behaviour, but no other behaviours. If such behaviours that should not be applied to an anchor are applied, these behaviours are ignored by the anchor.

Figure 57 shows an example of using anchors. In this figure, three anchors (rt-content 1, rt-content 2, rt-content 3) are attached to the hub document (rt-content A). Depending on the anchor information, each anchor specifies the shape and the position of the hotspot within the hub document. A rendering system displays anchors in an appropriate way. If the user selects one of the anchors, the link objects associated with these anchors are fired and lead the user to another presentation.

56.1 Behaviour attributes and statuses

None.

56.2 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following action:

- Attach Anchor action (see 56.2.1).

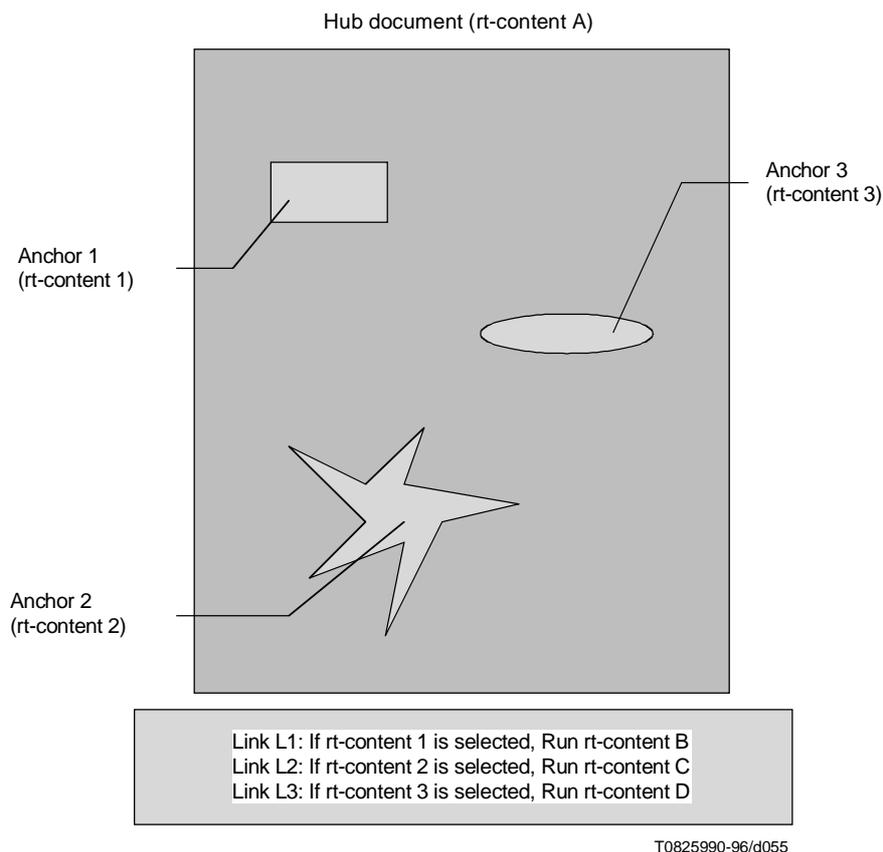


Figure 57/T.171 – Example of anchors

56.2.1 Attach Anchor action

This action attaches anchors to the target.

This action has the following parameters:

- Set of Rt-Content Target Param.
- Set of Anchor Spec Param: It may be used by a specific style to interchange some additional information to be associated with. For example, a value range for a slider style. How many generic values are required the semantics of those generic values, depends on a style.

This action may be targeted during the periods R2, R3 and R3.TD.

56.2.1.1 Attach Anchor action effect

The effect of this action is as follows:

- Depending on the specified update command, specified anchors are added, removed or replaced.

56.2.1.2 Attach Anchor additional error conditions

- if one of the targets is not an rt-content in R2, R3 or R3.TD, this action is ignored for that target;
- if one of the anchors is not an anchor (i.e. containing no catalogued classification attribute as “Anchor”), this action is ignored for that anchor.

SECTION 7 – CHANNELS BEHAVIOUR

57 Channel Availability Behaviour

This behaviour describes the availability of channels. Each channel has this behaviour.

57.1 Behaviour attributes and statuses

This behaviour is expressed using the following attribute or status:

- Channel Availability Status (see 57.2).

57.2 Channel Availability Status

The availability of each channel to the MHEG engine may be evaluated. The channel availability status indicates whether a channel is available. This status has one of the two values, “available” or “not available”.

The channel availability status should be evaluated to “not available” if the channel cannot be activated by the MHEG engine, i.e. the channel is in period C1 (see 29.3, “Channel availability”).

NOTE 1 – “Cannot” means that the channel has not been formally created or has been deleted. It does not mean that equipment required to perform the processing is not operational.

For example, the channel has not yet been created by the MHEG engine, i.e. a New action has not yet been targeted.

The channel availability status should be evaluated to “available” if the channel is activated by the MHEG engine, i.e. the channel has been created by the MHEG engine and is in the period C2.

NOTE 2 – The MHEG engine may offer to the using application a more detailed diagnostic and information message. For example, to explain that a channel cannot be created, or to indicate error conditions. These messages are not defined by this Recommendation, but may be defined by another standard.

Initially, each channel expect for the default channel is “not available” to the MHEG engine.

57.3 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following actions:

- New Channel action (see 57.3.1);
- Delete Channel action (see 57.3.2).

57.3.1 New Channel action

This action creates channels. This is used to obtain a logical perception space through which the assigned rt-components are perceived.

The channel creation process is outside the scope of this Recommendation and is provided by the MHEG engine.

This action has the following parameter:

- Set of Channel Target Param.

This action may be targeted during the period C1.

57.3.1.1 New Channel action effect

The MHEG effect of this action is as follows:

- 1) Construct the target channel (see 9.3).
- 2) Assign a channel identifier (target of this action) to the constructed channel.
- 3) Initialise the channel perceptability behaviour as “off”.
- 4) Set the channel availability status of the target to “available”. The created channel enters in period C2.

57.3.1.2 New Channel additional error conditions

- if one of the targets is not a channel in C1, this action is ignored for that target;
- if an existing channel identifier is used in one of the targets, this action is ignored for that target;
- if one of the targets is the default channel, this action is ignored for that target.

57.3.2 Delete Channel action

This action removes channels from the MHEG engine. This may be used to release resources in the MHEG engine.

This action has the following parameter:

- Set of Channel Target Param.

This action may be targeted during the period C2.

57.3.2.1 Delete Channel action effect

The effect of this action is as follows:

- 1) Destroy the channel perceptability behaviour.
Object designer should be aware that if the channel is “on”, the channel and all associated rt-components disappear. How to handle the disappearance is not defined by this Recommendation. It is left to the MHEG engine.
- 2) Implicit Stop actions are targeted to all rt-components assigned to the targets.
- 3) All rt-components assigned to the target are deassigned, and a Set Channel Assignment action targeted to the default channel is implicitly targeted to those rt-components.
- 4) NOTE 1 – The behaviours of those rt-components remain identical after the deassignment.
- 5) NOTE 2 – Destroy the target and its CPS.

Set the channel availability status of the target to “not available”. The channel enters in period C1.

57.3.2.2 Delete Channel additional error conditions

- if one of the targets is not a channel in C2, this action is ignored for that target;
- if one of the targets is the default channel, this action is ignored for that target.

57.4 Actions to retrieve the behaviour

This Recommendation defines means to get this behaviour by the use of the following action:

- Get Channel Availability Status action (see 57.4.1).

57.4.1 Get Channel Availability Status action

This action retrieves the channel availability status of the target.

This action has the following parameter:

- Channel Target Param.

This action may be targeted during the periods C1 and C2.

57.4.1.1 Get Channel action effect

The effect of this action is as follows:

- 1) if the target is in period C1, this action evaluates “not available”;
- 2) otherwise (the target is in period C2), this action evaluates “available”.

57.4.1.2 Get Channel Availability Status additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if the target is not a channel, this action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target.

58 Channel Perceptability Behaviour

This behaviour describes the perceptability of channels. All channels have this behaviour.

58.1 Behaviour attributes and statuses

This behaviour is expressed using the following attribute or status:

- Channel Perceptability (see 58.2).

58.2 Channel Perceptability

Each channel may be turned on and off. If a channel is turned on, all the rt-components assigned to this channel are perceived by the user. If a channel is turned off, all the rt-components assigned to this channel are no more perceived by the user.

NOTE – Rt-Components may be assigned and positioned to a non-perceptible channel. If the channel is turned on, all the assigned rt-components appear as specified previously.

The channel perceptability has one of the two values, “on” or “off”.

Initially, the default channel is turned on. So the channel perceptability state is “on”. However, all other channels are turned off. So their channel perceptability is “off”.

58.3 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following action:

- Set Channel Perceptability action (see 58.3.1).

58.3.1 Set Channel Perceptability action

This action changes the channel perceptability of a target.

This action has the following parameters:

- Set of Channel Target Param;
- Channel Perceptability Param.

This action may be targeted during the period C2.

58.3.1.1 Set Channel Perceptability action effect

The effect of this action is as follows:

- Set the channel perceptability of the target as specified
 - 1) if the target is set to “off”, the target is turned off;
 - 2) otherwise (the target is set to “on”), the target is turned on.

There is no effect if the specified value is the same as the current value.

58.3.1.2 Set Channel Perceptability additional error conditions

- If one of the targets is not a channel in C2, this action is ignored for that target.

58.4 Actions to retrieve the behaviour

This Recommendation defines means to get this behaviour by the use of the following action:

- Get Channel Perceptability action (see 58.4.1).

58.4.1 Get Channel Perceptability action

This action retrieves the channel perceptability of the target.

This action has the following parameter:

- Channel Target Param.

This action may be targeted during the period C2.

58.4.1.1 Get Channel Perceptability action effect

This action evaluates the current value of the channel perceptability attribute, and returns “on” or “off” depending on the state.

58.4.1.2 Get Channel Perceptability additional error conditions

- the target addresses multiple entities, e.g. use of an alias which addresses a list of targets, or use of the “*” wildcard instead of a given rt-component number, or use of children or descendants tails, the action is ignored for that target;
- if one of the targets is not a channel in C2, this action is ignored for that target;
- if the target is not accessible, e.g. the reference cannot be solved, the action is ignored for that target.

59 Channel Presentation Space Behaviour

This behaviour describes the channel presentation space. At the creation of channels, each presentation space of a channel is initialised to the default values. These values may be changed after creation.

59.1 Behaviour attributes and statuses

None.

59.2 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following action:

- Set CPS action (see 59.2.1).

59.2.1 Set CPS action

This action specifies the CPS of the target channel.

This action has the following parameters:

- Set of Channel Target Param;
- CPS Initialisation Param.

This action may be targeted during the period C2.

59.2.1.1 Set CPS action effect

The effect of this action is as follows:

- 1) The temporal axis of the target CPS is set to the specified CPS duration. If it is omitted, the temporal axis is set to infinite.
- 2) The spatial axis of the target CPS is set to the specified CPS size. If it is omitted on a certain axis, this spatial axis is set to 0.

There may be a user effect. If the spatial axes of the CPS are shrunk, some rt-components projected to this CPS may become invisible.

59.2.1.2 Set CPS additional error conditions

- If one of the targets is not a channel in C2, this action is ignored for that target.

SECTION 8 – CHANNELS AND RT-COMPONENTS BEHAVIOUR

60 Channels and Rt-Components Events Behaviour

This behaviour describes the events that may occur on channels and rt-components.

The behaviour allows the MHEG engine to get events that may arise not only from the MHEG engine but also from some systems or system components outside of it.

60.1 Behaviour attributes and statuses

This behaviour is expressed using the following attributes or statuses:

- Event (see 60.2);
- Event Data (see 60.3).

60.2 Event

An event is identified by a catalogued entry that is a list. An event identifier 0 is reserved to represent the status that no event has occurred.

The event attribute holds the event identifier that has occurred most recently to a channel or an rt-component. Initially, each event attribute is set to 0. This behaviour may be changed by the occurrence of an event.

60.3 Event Data

An event data attribute stores the information associated with the event most recently received by a channel or rt-component if the event requests to store additional information. The value stored in an event data attribute is a generic value. Its valid structure and allowed values are catalogued associated with an event.

Initially, the value of an event data attribute is set to a specific value depending on the encoding of the MHEG object. In this Recommendation, it is a NULL in ASN.1.

60.4 Actions to change the behaviour

This Recommendation defines means to modify this behaviour by the use of the following action:

- Set Event action (see 60.4.1).

60.4.1 Set Event action

This action sends an event to the target by the intention of the author. This action may be implicitly targeted if the MHEG engine or some systems generate an event and if this event is catalogued.

This action has the following parameters:

- Set of Rt-Component Channel Tg Param.
- Event Param.
- Event Data Param: Specifies associated data with an event.

This action may be targeted during the periods R2, R3, R3.TD and C2.

60.4.1.1 Set Event action effect

The effect of this action is as follows:

- 1) set the event attribute of the target as specified;
- 2) if the event data is associated with the event, the event data attribute is filled with it;
- 3) otherwise (the event data is omitted), the event data attribute is set to NULL.

It is an author's responsibility to keep the consistency between the event attribute and the event data attribute if this action is explicitly used.

It is the MHEG engine or the using application responsibility to keep this consistency if this action is implicitly used.

60.4.1.2 Set Event additional error conditions

- If one of the targets is not an rt-component or a channel in R2, R3, R3.TD or C2, this action is ignored for that target.

60.5 Actions to retrieve the behaviour

This Recommendation defines means to get this behaviour by the use of the following actions:

- Get Event action (see 60.5.1).
- Get Event Data action (see 60.5.2).

60.5.1 Get Event action

This action retrieves the event that happened most recently to the target.

This action has the following parameter:

- Rt-Component Channel Tg Param: Specifies associated data with an event.

This action may be targeted during the periods R2, R3, R3.TD and C2.

60.5.1.1 Get Event action effect

The effect of this action is as follows:

- The event attribute is elevated and stored value is returned.

60.5.1.2 Get Event additional error conditions

- If one of the targets is not an rt-component or a channel in R2, R3, R3.TD or C2, this action is ignored for that target.

60.5.2 Get Event Data action

This action retrieves the event data of the target that is associated with the most recently happened event.

This action has the following parameter:

- Rt-Component Channel Tg Param: Specifies associated data with an event.

This action may be targeted during the periods R2, R3, R3.TD and C2.

60.5.2.1 Get Event Data action effect

The effect of this action is as follows:

- The event data attribute is evaluated and stored value is returned.

60.5.2.2 Get Event Data additional error conditions

- If one of the targets is not an rt-component or a channel in R2, R3, R3.TD or C2, this action is ignored for that target.

61 MH-object class representation attributes

Table 4/T.171 – Overview of MH-object class representation attributes

<p>MH-object Class ::= Class Identification, Mheg ID?, Description?</p> <p>Class Identification ::= #joint-iso-itu-t, #mheg, #version, Class ID</p> <p>Class ID ::= #action-class-ID / #link-class-ID / #script-class-ID / #content-class-ID / #multiplexed-content-class-ID / #composite-class-ID / #container-class-ID / #descriptor-class-ID / Extensibility Provision</p> <p>Description ::= Name?, Owner?, Version?, Date?, Keywords?, Copyright?, Copyright ID?, Copyright Number?, Licence?, Cache Priority?, Comments?, Extensibility Provision</p> <p>Name ::= String</p> <p>Owner ::= String</p> <p>Version ::= String</p> <p>Date ::= §UTCTime</p> <p>Keywords ::= String*</p> <p>Copyright ::= String</p> <p>Copyright ID ::= §OCTET STRING</p> <p>Copyright Number ::= §OCTET STRING</p> <p>Licence ::= String</p> <p>Cache Priority ::= Integer</p> <p>Comments ::= String</p>

61.1 MH-object Class

- The MH-object class provides the structure for the identification and interchange of all MHEG objects. It provides also the generic addressing mechanism. MH-object class is an abstract class providing a consistent approach to the precise identification and interchange of all MHEG objects.

Inherits from: NONE

Addresses: NONE

Inherited by: Action, Link, Model, Container and Descriptor classes

Addressed by: NONE

- Sequence of (Class Identification, Mheg ID?, Description?).

61.2 Class Identification

- Identification of the class by:
MHEG Standard: joint-iso-itu (2) mheg (19)
MHEG Standard version: (1)
Class identifier corresponding to the interchanged object
- Sequence of (joint-iso-itu, mheg, version, Class ID).

61.3 Class ID

- Identification of the MHEG object class of the interchanged object.
- Choice between (action-class-ID, link-class-ID, script-class-ID, content-class-ID, multiplexed-content-class-ID, composite-class-ID, container-class-ID, descriptor-class-ID, Extensibility Provision).

61.4 Description

- More precise description of an MHEG interchanged object.
- Sequence of (Name?, Owner?, Version?, Date?, Keywords?, Copyright?, Copyright ID?, Copyright Number?, Licence?, Cache Priority?, Comments?, Extensibility Provision).

61.5 Name

- Name of the MHEG object.
- String.

61.6 Owner

- Identification of the owner of the MHEG object.
- String.

61.7 Version

- Identification of the version of the MHEG object.
- String.

61.8 Date

- Latest modification date of the MHEG object. In the base notation, ASN.1, ISO format Universal Time (UTC) is used.
- UTCTime.

61.9 Keywords

- List of keywords qualifying the MHEG object.
- List of (String).

61.10 Copyright

- Content of the copyright attached to the MHEG object.
- String.

61.11 Copyright ID

- Work type code identifier.
- OCTET STRING.

61.12 Copyright Number

- Unique identification code of each work.
- OCTET STRING.

61.13 Licence

- Information concerning the licence attached to the MHEG object.
- String.

61.14 Cache Priority

- It provides a facility for memory management and is used by the MHEG engine for optimisation of Prepare and Destroy actions. The value should be within the interval (0, 255).
- Integer.

61.15 Comments

- Free format comments attached to the MHEG object.
- String.

62 Action class representation attributes

Table 5/T.171 – Overview of Action class representation attributes

<p>Action Class ::= MH-object Class, Synchro Indicator Param, Synchronised Action+, Extensibility Provision</p> <p>Synchro Indicator Param ::= Synchro Indicator Synchro Indicator Macro</p> <p>Synchro Indicator Macro ::= Macro Def ID, Synchro Indicator?</p> <p>Synchro Indicator ::= <i>#serial</i> / <i>#parallel</i></p> <p>Synchronised Action ::= Elementary Action Action Object</p> <p>Action Object ::= Action Object Reference Action Class</p>
--

62.1 Action Class

- The action class provides a consistent structure for the interchange of sets of actions. It also provides a consistent structure for the interchange of basic action object, nested action object. Action objects, instances of action class, are used to interchange an organised set of elementary actions.

Inherits from: MH-object Class

Addresses: Action Class

Inherited by: NONE

Addressed by: Link Class, Composite Class, Container Class

- Sequence of (MH-object Class, Synchro Indicator Param, Synchronised Action+, Extensibility Provision).

62.2 Synchro Indicator Param

- Parameter of Synchro Indicator.
- Choice between (Synchro Indicator, Synchro Indicator Macro).

62.3 Synchro Indicator Macro

- Allows the specification of a macro parameter instead of a specified Synchro Indicator value.
- Sequence of (Macro Def ID, Synchro Indicator?).

62.4 Synchro Indicator

- This parameter specifies if the following group of synchronised actions are to be performed in parallel or serially.
- Choice between (serial, parallel).

62.5 Synchronised Action

- Defines a recursive structure to express combination of actions. Each synchronised action within this group is either a single elementary action or another nested action object.
- Choice between (Elementary Action, Action Object).

62.6 Action Object

- The action object can be referenced or included in the object.
- Choice between (Action Object Reference, Action Class).

63 Link class representation attributes

Table 6/T.171 – Overview of Link class representation attributes

Link Class ::= MH-object Class, Link Condition, Link Effect, Extensibility Provision
Link Condition ::= Trigger Condition Logical Combination
Trigger Condition ::= Source Value, Previous Condition?, Current Condition
Constraint Condition ::= Source Value, Current Condition
Source Value ::= Evaluated Value
Comparison Operation ::= Comparison Operator, Comparison Value
Comparison Value ::= Generic Value Comparison Value Constant <i>#unspecified</i>
Previous Condition ::= Comparison Operation
Current Condition ::= Comparison Operation
Comparison Operator ::= <i>#equal</i> / <i>#not-equal</i> / <i>#greater</i> / <i>#greater-equal</i> / <i>#less</i> / <i>#less-equal</i>
Logical Combination ::= Logical Operator, Condition+
Logical Operator ::= <i>#and</i> / <i>#or</i> / <i>#xor</i> / <i>#not</i>
Condition ::= Trigger Condition Constraint Condition Logical Combination
Link Effect ::= Macro Parameter Resolution*, Action Object
Macro Parameter Resolution ::= Macro Def ID, Usage Value
Usage Value ::= Generic Value

63.1 Link Class

- The Link class provides a consistent structure for the interchange of conditional actions targeted to MHEG objects, run-time objects or channels. MHEG link objects, instances of Link class, may be interchanged. A link object is directional and connects one or more sources with one or more targets. The actions defined within the link are processed on the indicated targets only if the conditions are satisfied.

Inherits from: MH-object class

Addresses: Action class

Inherited by: NONE

Addressed by: Composite class, Container Class

- Sequence of (MH-object Class, Link Condition, Link Effect, Extensibility Provision).

63.2 Link Condition

- Defines the condition of the link. The link effect is to be performed each time the link condition becomes evaluated to true.
- Choice between (Trigger Condition, Logical Combination).

63.3 Trigger Condition

- Defines a trigger condition which is described as a change of an attribute or status value.
- Sequence of (Source Value, Previous Condition?, Current Condition).

63.4 Constraint Condition

- Defines a constraint condition which tests a current state of a value.
- Sequence of (Source Value, Current Condition).

63.5 Source Value

- Identifies the status or attribute of an object which is to be compared in a previous or a current condition. The evaluated value is the result of a get action targeted on a single object. Otherwise the source value is “undefined”.
- Evaluated Value.

63.6 Comparison Operation

- Describes a comparison operation to be performed on source value in order to fulfil a condition.
- Sequence of (Comparison Operator, Comparison Value).

63.7 Comparison Value

- Specification of the value to be compared to the source value using the comparison operator. It is for the object designer to ensure that the type of the current comparison value is compatible with the source value.
- Choice between (Generic Value, Comparison Value Constant, unspecified).

63.8 Previous Condition

- Defines the Previous Condition on the source value.
If the Previous Condition is omitted, it means that the Previous Condition is evaluated as: NOT Current Condition.
- Comparison Operation.

63.9 Current Condition

- Defines the Current Condition on the source value.
- Comparison Operation.

63.10 Comparison Operator

- Specifies the operation to be made between the source value and the comparison value.
- Choice between (equal, not-equal, greater, greater-equal, less, less-equal).

63.11 Logical Combination

- Defines a tree structure to express combination of conditions. The Logical operator constitutes a node of the logical tree of conditions. The set of conditions constitutes the leaves of this node. When the logical operator is NOT, only one condition is allowed. Any other operator needs at least two conditions.
- Sequence of (Logical Operator, Condition+).

63.12 Logical Operator

- Defines the logical operation to be applied between the conditions of the node.
- Choice between (and, or, xor, not).

63.13 Condition

- Defines the conditions on which the logical operation is to be applied.
- Choice between (Trigger Condition, Constraint Condition, Logical Combination).

63.14 Link Effect

- Defines the macro resolution providing a usage value to each macro parameter and the action object to be performed when the result of the link is fired.
- Sequence of (Macro Parameter Resolution*, Action Object).

63.15 Macro Parameter Resolution

- Associates a usage value to a macro Def ID.
- Sequence of (Macro Def ID, Usage Value).

63.16 Usage Value

- Assigns a usage value to a macro Def ID. It is for the object designer to ensure that the usage value is compatible with the parameter which has been specified using a macro Def ID.
- Generic Value.

64 Model class representation attributes

Table 7/T.171 – Overview of Model class representation attributes

Model Class ::= MH-object Class

64.1 Model Class

- Model Class is an abstract class. The script class and the component class inherit this class. Model class is an abstract class.
Inherits from: MH-object class
Addresses: NONE
Inherited by: Script Class, Component Class
Addressed by: NONE
- MH-object Class.

65 Script class representation attributes

Table 8/T.171 – Overview of Script class representation attributes

<p>Script Class ::= Model Class, Script Classification?, Script Hook, Script Data, Extensibility Provision</p> <p>Script Classification ::= Catalogued Script Classification</p> <p>Script Data ::= Script Inclusion Data Reference</p> <p>Script Inclusion ::= <i>\$BIT STRING</i> / <i>\$OCTET STRING</i> / InterchangedScript</p> <p>InterchangedScript ::= <i>#Imported from ITU-T Rec. T.173</i></p>
--

65.1 Script Class

- The Script class provides a consistent structure for the interchange of complex actions on MHEG entities. MHEG script objects, instances of Script class, may be interchanged. A script object contains an indication of the scripting language used and the encoded script itself. It is assumed that the scripting language used in a script object is able to reference MHEG entities and access their attributes.
Inherits from: Model class
Addresses: NONE
Inherited by: NONE
Addressed by: Container class
- Sequence of (Model Class, Script Classification?, Script Hook, Script Data, Extensibility Provision).

65.2 Script Classification

- One of the catalogued script classification, it may be a registered one or a proprietary one.
- Catalogued Script Classification.

65.3 Script Data

- Encoded script according to the scripting language identified by the script hook. This script data may be included in the script object itself or referenced by using the mechanism of external identifier or alias. A null script data may be also used.
- Choice between (Script Inclusion, Data Reference).

65.4 Script Inclusion

- The script data is included in the interchanged script object.
- Choice between (BIT STRING, OCTET STRING, InterchangedScript).

65.5 InterchangedScript

- The Interchanged Script syntax is provided by Recommendation T.173.
- Imported from Recommendation T.173.

66 Component class representation attributes

Table 9/T.171 – Overview of Component class representation attributes

<p>Component Class ::= Model Class, OPS Initialisation?</p>
--

66.1 Component Class

- Component Class is an abstract class. The content class and the composite class inherit this class. Component class is an abstract class.
Inherits from: Model class
Addresses: NONE
Inherited by: Content Class, Composite Class
Addressed by: NONE
- Sequence of (Model Class, OPS Initialisation?).

67 Content class representation attributes

Table 10/T.171 – Overview of Content class representation attributes

<p>Content Class ::= Component Class, Cat Content Classification?, Content Hook, OV?, Content Data, Extensibility Provision</p> <p>Content Data ::= Data Inclusion Data Reference</p> <p>Data Inclusion ::= §BIT STRING / §OCTET STRING / Generic Value</p>
--

67.1 Content Class

- The content class provides a consistent structure for the interchange of encoded data. MHEG content objects, instances of Content class, may be interchanged. A content object provides the identification of the type, and the interchange of data.
Inherits from: Component Class
Addresses: NONE
Inherited by: Multiplexed Content Class
Addressed by: Composite Class, Container Class
- Sequence of (Component Class, Cat Content Classification?, Content Hook, OV?, Content Data, Extensibility Provision).

67.2 Content Data

- Encoded data provided by other Recommendations and standards or by proprietary format. This data is included in the content object itself or referenced by using the mechanism of alias or external identifier offered. A Null-Data can also be specified.
The content data is not limited to encoding schemes provided by International Standards. The reference to content data facility in this Recommendation provides a mechanism for reference to content data conforming to any standard including private standards.
- Choice between (Data Inclusion, Data Reference).

67.3 Data Inclusion

- The data is included in the interchanged content object.
- Choice between (BIT STRING, OCTET STRING, Generic Value).

68 Multiplexed content class representation attributes

Table 11/T.171 – Overview of Multiplexed content class representation attributes

<p>Multiplexed Content Class ::= Content Class, Multiplexed Stream+, Extensibility Provision</p> <p>Multiplexed Stream ::= Stream ID, Cat Content Classification?, Content Hook?</p>
--

68.1 Multiplexed Content Class

- The multiplexed content class provides a consistent structure for the exchange of multiplexed media data. The multiplexed content class is a model class which is a subclass of the content class. It contains or refers to the coded representation of a multiplexed media data together with a description of each multiplexed stream. A Multiplexed Content Class represents the association between a Content Class and an ordered list of streams describing the streams contained in the multiplexed data.

Inherits from: Content class

Addresses: NONE

Inherited by: NONE

Addressed by: Composite, Container classes

- Sequence of (Content Class, Multiplexed Stream+, Extensibility Provision).

68.2 Multiplexed Stream

- Defines a stream of the multiplexed data and specific information.
- Sequence of (Stream ID, Cat Content Classification?, Content Hook?).

69 Composite class representation attributes

Table 12/T.171 – Overview of Composite class representation attributes

Composite Class ::=

Component Class, Availability Start-up?, Availability Close-down?, Rt- Availability Start-up?, Rt-Availability Close-down?, Action Object*, Link Object*, Nb Of Elements, Composition Element*, Extensibility Provision

Availability Start-up ::=

Link Effect | #automatic-start-up-1 | #automatic-start-up-2 | #automatic start-up-3 | #automatic-start-up-4 | #automatic-start-up-5

Availability Close-down ::=

Link Effect | #automatic-close-down-1 | #automatic-close-down-2 | #automatic-close-down-3

Rt-Availability Start-up ::= Link Effect | #automatic-rt-start-up

Rt-Availability Close-down ::= Link Effect

Link Object ::= Link Class | Link Object Reference

Nb Of Elements ::= Integer

Composition Element ::= Index, Associated Model

Element Index ::= Index

Associated Model ::=

Component Object Reference | Content Class | Multiplexed Content Class | Composite Class | Label

Label ::= String

69.1 Composite Class

- The composite class provides a consistent approach to the synchronisation in time and space, linking, encapsulation of a set of objects and interchange of these objects. MHEG composite objects, instances of composite class, may be interchanged. A composite object addresses actions, links to describe the initial and dynamic behaviour of the component objects and the rt-objects created from these composite objects.

Inherits from: Component class

Addresses: Link, Action, Content, Multiplexed Content and Composite classes

Inherited by: NONE

Addressed by: Composite class

- Sequence of (Component Class, Availability Start-up?, Availability Close-down?, Rt-Availability Start-up?, Rt-Availability Close-down?, Action Object*, Link Object*, Nb of Elements, Composition Element*, Extensibility Provision).

69.2 Availability Start-up

- The composite availability start-up may be used for additional preparation effect of the composite. It is either a personalised link effect provided by the author or one of the default link effects provided by this Recommendation.
- Choice between (Link Effect, automatic-start-up-1, automatic-start-up-2, automatic start-up-3, automatic-start-up-4, automatic-start-up-5).

69.3 Availability Close-down

- The composite close-down start-up may be used for additional destruction effect of the composite. It is either a personalised link effect provided by the author or one of the default link effects provided by this Recommendation.
- Choice between (Link Effect, automatic-close-down-1, automatic-close-down-2, automatic-close-down-3).

69.4 Rt-Availability Start-up

- The rt-availability start-up may be used for additional creation effect of the rt-composite. It is either a personalised link effect provided by the author or the default link effect provided by this Recommendation.
- Choice between (Link Effect, automatic-rt-start-up).

69.5 Rt-Availability Close-down

- The rt-availability close-down may be used for additional destruction effect of the rt-composite. It is always a personalised link effect provided by the author.
- Link Effect.

69.6 Link Object

- Link object used to describe the behaviour of each element and the interrelationships between parent and siblings. The link object can be referenced or included in the composite object.
- Choice between (Link Class, Link Object Reference).

69.7 Nb of Elements

- Indicates the number of elements within the composition.
- Integer.

69.8 Composition Element

- Defines the internal structure of the composite. This structure provides a support for created rt-composites.

If no elements are included, the MHEG engine deduces that the composite has “Nb of Elements” empty elements.

If the last element has an index lower than “Nb of Elements”, the MHEG engine deduces that empty elements are repeated from the last element until the “Nb of Element” element.

Each element has an index and refers a model component object or a label. The component object may be interchanged included in this composite object or externally. The label is always included.

Empty elements are not encoded but are implicitly generated by the MHEG engine.

- Sequence of (Index, Associated Model).

69.9 Element Index

- Index of the associated model unique within a composite.
- Index.

69.10 Associated Model

- The element addresses either a component object or a label.
- Choice between (Component Object Reference, Content Class, Multiplexed Content Class, Composite Class, Label).

69.11 Label

- A label is a string, it can be provided as an element of a composite.
- String.

70 Container class representation attributes

Table 13/T.171 – Overview of Container class representation attributes

<p>Container Class ::= MH-object Class, Container Start-up?, Container Close-down?, Container Element+, Extensibility Provision</p> <p>Container Start-up ::= Link Effect <i>#automatic-container-start-up-1</i> <i>#automatic-container-start-up-2</i> <i>#automatic-container-start-up-3</i> <i>#automatic-container-start-up-4</i> <i>#automatic-container-start-up-5</i> <i>#automatic-container-start-up-6</i> <i>#automatic-container-start-up-7</i> <i>#automatic-container-start-up-8</i></p> <p>Container Close-down ::= Link Effect <i>#automatic-container-close-down</i></p> <p>Container Element ::= MH-Reference Action Class Link Class Script Class Content Class Multiplexed Content Class Composite Class Container Class Descriptor Class</p>
--

70.1 Container Class

- The container class provides a consistent structure for the handling of multimedia and hypermedia information.

This regroupment is intended to facilitate the interchange in order to obtain in a unit set of interchange. Defines the elements of the container to be exchanged. The indexes of the components do not require to be encoded because the elements are numbered from 1 to n by the MHEG engine.

Inherits from: MH Object class

Addresses: Link, Action, Script, Content, Multiplexed Content, Composite, Descriptor and Container classes

Inherited by: NONE

Addressed by: NONE

- Sequence of (MH-object Class, Container Start-up?, Container Close-down?, Container Element+, Extensibility Provision).

70.2 Container Start-up

- The container start-up may be used for additional preparation effect of the container. It is either a personalised link effect provided by the author or one of the default link effects provided by this Recommendation.
- Choice between (Link Effect, *automatic-container-start-up-1*, *automatic-container-start-up-2*, *automatic-container-start-up-3*, *automatic-container-start-up-4*, *automatic-container-start-up-5*, *automatic-container-start-up-6*, *automatic-container-start-up-7*, *automatic-container-start-up-8*).

70.3 Container Close-down

- The container availability close-down may be used for additional destruction effect of the container. It is either a personalised link effect provided by the author or the default link effect provided by this Recommendation.
- Choice between (Link Effect, automatic-container-close-down).

70.4 Container Element

- An element can be included or referenced.
- Choice between (MH-Reference, Action Class, Link Class, Script Class, Content Class, Multiplexed Content Class, Composite Class, Container Class, Descriptor Class).

71 Descriptor class representation attributes

Table 14/T.171 – Overview of Descriptor class representation attributes

<p>Descriptor Class ::= MH-object Class, Related Object*, Other Descriptor*, Readme?, System Readable Material?, Channel Information*, Catalogued Style Information*, Cat Ext elementary action Info*, Cat Ext Attribute Info*, Extensibility Provision</p> <p>Related Object ::= MH-Reference, Object Information?</p> <p>Object Information ::= Object Size?, Class ID, Class Specific Information?, Offset?</p> <p>Object Size ::= Integer</p> <p>Class Specific Information ::= Script Class Information Content Class Information Mux Content Class Info</p> <p>Script Class Information: ::= Script Classification?, Script Hook?</p> <p>Content Class Information ::= Cat Content Classification?, Content Hook?, Alternative Object*</p> <p>Mux Content Class Info ::= Content Class Information, Number Of Streams?, Stream Information*</p> <p>Number Of Streams ::= Integer</p> <p>Stream Information ::= Stream ID, Content Class Information</p> <p>Alternative Object ::= Content Object Ref, Content Hook?, Alternative Descriptor Object?, Alternative Readme?</p> <p>Alternative Descriptor Object ::= Descriptor Object Reference</p> <p>Alternative Readme ::= String</p> <p>Offset::=Integer</p> <p>Other Descriptor ::= Descriptor Object Reference</p> <p>Readme::=String</p> <p>System Readable Material ::= <i>\$BIT STRING</i> / <i>\$OCTET STRING</i></p> <p>Channel Information ::= Channel ID, X min?, X max?, Y min?, Y max?, Z min?, Z max?, X Resolution?, Y Resolution?, Z Resolution?, T Resolution?, F min?, F max?, Audio Dynamic?, Channel Media Type*, Event Mapping*</p> <p>X min: ::= Integer</p> <p>X max ::= Integer</p> <p>Y min ::= Integer</p> <p>Y max ::= Integer</p>

Table 14/T.171 – Overview of Descriptor class representation attributes (concluded)

Z min ::= Integer
Z max ::= Integer
X Resolution ::= Integer
Y Resolution ::= Integer
Z Resolution ::= Integer
T Resolution ::= Integer
F min ::= Integer
F max ::= Integer
Audio Dynamic ::= Integer
Channel Media Type ::= Catalogued Media Type
Event Mapping ::= Event, Catalogued Event?
Catalogued Style Information ::= Catalogued Style
Cat Ext elementary action Info ::= Catalogued Extended EA
Cat Ext Attribute Info ::= Cat Ext Attribute

71.1 Descriptor Class

- The descriptor class defines a description of objects that are to be interchanged. The aim is to facilitate the negotiation, installation, operation and management of applications that interchange MHEG objects. MHEG descriptor objects, instances of descriptor class, may be interchanged. A descriptor object describes the support provided for the description of other objects that are interchanged.
- Sequence of (MH-object Class, Related Object*, Other Descriptor*, Readme?, System Readable Material?, Channel Information*, Catalogued Style Information*, Cat Ext elementary action Info*, Cat Ext Attribute Info*, Extensibility Provision).

71.2 Related Object

- Specifies the scope of the descriptor, i.e. the list of MHEG objects concerned by this descriptor.
- Sequence of (MH-Reference, Object Information?).

71.3 Object Information

- Defines more precisely the related object.
- Sequence of (Object Size?, Class ID, Class Specific Information?, Offset?).

71.4 Object Size

- Size of the related encoded object in octets.
- Integer.

71.5 Class Specific Information

- Object specific information.
- Choice between (Script Class Information, Content Class Information, Mux Content Class Info).

71.6 Script Class Information

- Carries information specific to the script class.
- Sequence of (Script Classification?, Script Hook?).

71.7 Content Class Information

- Carries information specific to the content class.
- Sequence of (Cat Content Classification?, Content Hook?, Alternative Object*).

71.8 Mux Content Class Info

- Carries information specific to the multiplexed content class.
- Sequence of (Content Class Information, Number of Streams?, Stream Information*).

71.9 Number of Streams

- Reflects the number of sub-streams.
- Integer.

71.10 Stream Information

- Further information belonging to stream.
- Sequence of (Stream ID, Content Class Information).

71.11 Alternative Object

- Provides information about possible alternative objects that an engine can use instead of the related object.
- Sequence of (Content Object Ref, Content Hook?, Alternative Descriptor Object?, Alternative Readme?).

71.12 Alternative Descriptor Object

- References the alternative descriptor where the alternative objects are described.
- Descriptor Object Reference.

71.13 Alternative Readme

- Identifies the nature of the alternative object.
- String.

71.14 Offset

- Provides a relative position in octets of an encoded object which is included in another encoded object.
- Integer.

71.15 Other Descriptor

- Specifies a reference to another descriptor object.
- Descriptor Object Reference.

71.16 Readme

- Provides readable text for human users.
- String.

71.17 System Readable Material

- Provides information for using applications.
- Choice between (BIT STRING, OCTET STRING).

71.18 Channel Information

- Contains information to handle a channel.
- Sequence of (Channel ID, X min?, X max?, Y min?, Y max?, Z min?, Z max?, X Resolution?, Y Resolution?, Z Resolution?, T Resolution?, F min?, F max?, Audio Dynamic?, Channel Media Type*, Event Mapping*).

71.19 X min

- Defines the minimum X-axis value for the device on which this channel is to be mapped.
- Integer.

71.20 X max

- Defines the maximum X-axis value for the device on which this channel is to be mapped.
- Integer.

71.21 Y min

- Defines the minimum Y-axis value for the device on which this channel is to be mapped.
- Integer.

71.22 Y max

- Defines the maximum Y-axis value for the device on which this channel is to be mapped.
- Integer.

71.23 Z min

- Defines the minimum Z-axis value for the device on which this channel is to be mapped.
- Integer.

71.24 Z max

- Defines the maximum Z-axis value for the device on which this channel is to be mapped.
- Integer.

71.25 X Resolution

- Number of addressable spatial physical units on X axis.
- Integer.

71.26 Y Resolution

- Number of addressable spatial physical units on Y axis.
- Integer.

71.27 Z Resolution

- Number of addressable spatial physical units on Z axis.
- Integer.

71.28 T Resolution

- Number of addressable spatial physical units in one second on T axis.
- Integer.

71.29 **F min**

- Defines the minimum presentable frequency for the device on which this channel is to be mapped.
- Integer.

71.30 **F max**

- Defines the maximum presentable frequency for the device on which this channel is to be mapped.
- Integer.

71.31 **Audio Dynamic**

- Defines the audio dynamic for the device on which the channel is to be mapped. The units are dB.
- Integer.

71.32 **Channel Media Type**

- Specifies the type of media of the channel, it is one of the catalogued value which can be either registered or proprietary.
- Catalogued Media Type.

71.33 **Event Mapping**

- Specifies the expected events and their mapping on a catalogued event.
- Sequence of (Event, Catalogued Event?).

71.34 **Catalogued Style Information**

- Specifies which style is used by the related objects.
- Catalogued Style.

71.35 **Cat Ext elementary action Info**

- Specifies which catalogued elementary action is used by the related objects.
- Catalogued Extended EA.

71.36 **Cat Ext Attribute Info**

- Specifies which extended attribute is used by the related objects.
- Cat Ext Attribute.

72 **Behaviours**

72.1 **Postpone behaviour**

Table 15/T.171 – Overview of Postpone behaviour

Delay ::= Temporal Unit Ref Param, Duration Param

Temporal Unit Ref Param ::= Temporal Unit Ref | Temporal Unit Ref Macro

Temporal Unit Ref Macro ::= Macro Def ID, Temporal Unit Ref?

Temporal Unit Ref ::= Rt-Component Reference | default-GF

Duration Param ::= Generic Integer Param

72.1.1 Delay

- This action enables to delay the process of further actions.
- Sequence of (Temporal Unit Ref Param, Duration Param).

72.1.2 Temporal Unit Ref Param

- Parameter of Temporal Unit Ref.
- Choice between (Temporal Unit Ref, Temporal Unit Ref Macro).

72.1.3 Temporal Unit Ref Macro

- Allows the specification of a macro parameter instead of a specified Temporal Unit Ref value.
- Sequence of (Macro Def ID, Temporal Unit Ref?).

72.1.4 Temporal Unit Ref

- Defines how to retrieve the GTF in order to interpret the delay.
- Choice between (Rt-Component Reference, default-GF).

72.1.5 Duration Param

- The duration is expressed in GTU.
- Generic Integer Param.

72.2 Returnability behaviour

Table 16/T.171 – Overview of Returnability behaviour

Return ::=

Return Target Param+, Return Indicator Param, Returned Generic Value Param*, Content Object Ref Param*

Return Indicator Param ::= Generic Numeric Param

Returned Generic Value Param ::= Generic Value Param

Content Object Ref Param ::= Content Object Ref | Content Object Ref Macro

Content Object Ref Macro ::= Macro Def ID, Content Object Ref?

72.2.1 Return

- This action enable to return information to a using application or external entities outside the MHEG engine.
- Sequence of (Return Target Param+, Return Indicator Param, Returned Generic Value Param*, Content Object Ref Param*).

72.2.2 Return Indicator Param

- Identification for returning information.
- Generic Numeric Param.

72.2.3 Returned Generic Value Param

- A list of generic values to be returned.
- Generic Value Param.

72.2.4 Content Object Ref Param

- A list of content objects to be returned.
- Choice between (Content Object Ref, Content Object Ref Macro).

72.2.5 Content Object Ref Macro

- Allows the specification of a macro parameter instead of a specified Content Object Ref value.
- Sequence of (Macro Def ID, Content Object Ref?).

72.3 Alias behaviour

Table 17/T.171 – Overview of Alias behaviour

Set Alias ::= Target Param+, Alias Spec Param+

Alias Spec Param ::= Alias Spec | Alias Spec Macro

Alias Spec Macro ::= Macro Def ID, Alias Spec?

Alias Spec ::= Alias+, Update Command

72.3.1 Set Alias

- This action enables the assignment of aliases to any generic references.
- Sequence of (Target Param+, Alias Spec Param+).

72.3.2 Alias Spec Param

- Parameter of Alias Spec.
- Choice between (Alias Spec, Alias Spec Macro).

72.3.3 Alias Spec Macro

- Allows the specification of a macro parameter instead of a specified Alias Spec value.
- Sequence of (Macro Def ID, Alias Spec?).

72.3.4 Alias Spec

- Aliases and an Update Command to be applied.
- Sequence of (Alias+, Update Command).

72.4 Extensibility behaviour

Table 18/T.171 – Overview of Extensibility behaviour

Catalogued Elementary Action ::=

Target Param+, Catalogued Extended EA Param, Elementary Action Param*

Catalogued Extended EA Param ::= Catalogued Extended EA | Catalogued Extended EA Macro

Catalogued Extended EA Macro ::= Macro Def ID, Catalogued Extended EA?

Elementary Action Param ::= Generic Value Param

Set Catalogued Attribute ::=

Target Param+, Cat Ext Attribute Param, Ext Attribute Value Param, Transition Duration Param?

Ext Attribute Value Param ::= Generic Value Param

72.4.1 Catalogued Elementary Action

- This action enables to use an extended elementary action which is registered or proprietary.
- Sequence of (Target Param+, Catalogued Extended EA Param, Elementary Action Param*).

72.4.2 Catalogued Extended EA Param

- Parameter of Catalogued Extended EA.
- Choice between (Catalogued Extended EA, Catalogued Extended EA Macro).

72.4.3 Catalogued Extended EA Macro

- Allows the specification of a macro parameter instead of a specified Catalogued Extended EA value.
- Sequence of (Macro Def ID, Catalogued Extended EA?).

72.4.4 Elementary Action Param

- Used to interchange the specific parameters of the catalogued extended elementary action.
- Generic Value Param.

72.4.5 Set Catalogued Attribute

- This action enables to set a value of a catalogued attribute for the target.
- Sequence of (Target Param+, Cat Ext Attribute Param, Ext Attribute Value Param, Transition Duration Param?).

72.4.6 Ext Attribute Value Param

- A generic value to be assigned to a Cat Ext Attribute.
- Generic Value Param.

72.5 Mheg objects availability behaviour

Table 19/T.171 – Overview of Mheg objects availability behaviour

Prepare ::= MH-Target Param+

Destroy ::= MH-Target Param+

72.5.1 Prepare

- This action makes the MHEG object available to the MHEG engine.
- List of (MH-Target Param).

72.5.2 Destroy

- Removes an MHEG object from the MHEG engine in order to release resources.
- List of (MH-Target Param).

72.6 Link object activation behaviour

Table 20/T.171 – Overview of Link object activation behaviour

Activate ::= Link Target Param+

Deactivate ::= Link Target Param+

72.6.1 Activate

- Make the link object active.
- List of (Link Target Param).

72.6.2 Deactivate

- Makes the link object inactive.
- List of (Link Target Param).

72.7 Link object abort behaviour

Table 21/T.171 – Overview of Link object abort behaviour

Link Abort ::= Link Target Param+
--

72.7.1 Link Abort

- Aborts processing of links specified in the target set.
- List of (Link Target Param).

72.8 Content class generic value storage behaviour

Table 22/T.171 – Overview of Content class generic value storage behaviour

Set Data ::= Content Target Param+, Substitution Indicator Param, Data Element Param*
--

Substitution Indicator Param ::= Substitution Indicator Substitution Indicator Macro

Substitution Indicator Macro ::= Macro Def ID, Substitution Indicator?

Substitution Indicator ::= <i>#substitution</i> <i>#no-substitution</i>
--

Data Element Param ::= Data Element Data Element Macro

Data Element Macro ::= Macro Def ID, Data Element?

Data Element ::= Process Indicator, Generic List Elt ID?, Generic Value
--

Process Indicator ::= <i>#process</i> <i>#no-process</i>

Add ::= Content Target Param+, Generic List Elt ID Param?, Generic Value Param?
--

Substract ::= Content Target Param+, Generic List Elt ID Param?, Generic Value Param?
--

72.8.1 Set Data

- This action enables to store or to modify the generic value in a content object.
- Sequence of (Content Target Param+, Substitution Indicator Param, Data Element Param*).

72.8.2 Substitution Indicator Param

- Parameter of Substitution Indicator.
- Choice between (Substitution Indicator, Substitution Indicator Macro).

72.8.3 Substitution Indicator Macro

- Allows the specification of a macro parameter instead of a specified Substitution Indicator value.
- Sequence of (Macro Def ID, Substitution Indicator?).

72.8.4 Substitution Indicator

- Allowed values for the substitution indicator.
- Choice between (substitution, no-substitution).

72.8.5 Data Element Param

- Parameter of Data Element.
- Choice between (Data Element, Data Element Macro).

72.8.6 Data Element Macro

- Allows the specification of a macro parameter instead of a specified Data Element value.
- Sequence of (Macro Def ID, Data Element?).

72.8.7 Data Element

- Indicates a data to be stored.
- Sequence of (Process Indicator, Generic List Elt ID?, Generic Value).

72.8.8 Process Indicator

- Indicates a data to be stored after evaluated or not.
- Choice between (process, no-process).

72.8.9 Add

- Adds a generic numeric, generic integer or generic ratio to a content data.
- Sequence of (Content Target Param+, Generic List Elt ID Param?, Generic Value Param?).

72.8.10 Subtract

- Offer to subtract generic numeric, generic integer or generic ratio from a content data.
- Sequence of (Content Target Param+, Generic List Elt ID Param?, Generic Value Param?).

72.9 Content class copy behaviour

Table 23/T.171 – Overview of Content class copy behaviour

Copy ::= Content Target Param, Destination Param+

Destination Param ::= Content Target Param

72.9.1 Copy

- This action specifies a target content object as a source of the copy operation and a set of content objects as destinations of the copy operation.
- Sequence of (Content Target Param, Destination Param+).

72.9.2 Destination Param

- Content objects to which the target data are to be copied.
- Content Target Param.

72.10 Rt-objects availability behaviour

Table 24/T.171 – Overview of Rt-objects availability behaviour

<p>New ::= Rt-Target Param+</p> <p>Delete ::= Rt-Target Param+</p>
--

72.10.1 New

- Creates rt-objects from model objects for the MHEG engine.
- List of (Rt-Target Param).

72.10.2 Delete

- Removes rt-objects from the MHEG engine. This may be used to release resources.
- List of (Rt-Target Param).

72.11 Rt-objects running behaviour

Table 25/T.171 – Overview of Rt-objects running behaviour

<p>Run ::= Rt-Target Param+, Number Of Performances Param?</p> <p>Number Of Performances Param ::= Number Of Performances Number Of Performances Macro</p> <p>Number Of Performances Macro ::= Macro Def ID, Number Of Performances?</p> <p>Number Of Performances ::= Generic Integer <i>#infinite</i></p> <p>Stop ::= Rt-Target Param+</p>

72.11.1 Run

- Enables the presentation of rt-objects by the presentation process.
- Sequence of (Rt-Target Param+, Number of Performances Param?).

72.11.2 Number of Performances Param

- Parameter of Number of Performances.
- Choice between (Number of Performances, Number of Performances Macro).

72.11.3 Number Of Performances Macro

- Allows the specification of a macro parameter instead of a specified Number of Performances value.
- Sequence of (Macro Def ID, Number of Performances?).

72.11.4 Number of Performances

- Specifies the number of performances.
- Choice between (Generic Integer, infinite).

72.11.5 Stop

- Sets the running status to “not running” and stops possible user effects.
- List of (Rt-Target Param).

72.12 Rt-script passing parameter behaviour

Table 26/T.171 – Overview of Rt-script passing parameter behaviour

Set Parameters ::= Rt-Script Target Param+, Passing Param*

Passing Param ::= Passing | Passing Macro

Passing Macro ::= Macro Def ID, Passing?

Passing ::= Generic Value | Content Object Ref

72.12.1 Set Parameters

- Offers means to pass parameters to an rt-script.
- Sequence of (Rt-Script Target Param+, Passing Param*).

72.12.2 Passing Param

- Parameter of Passing.
- Choice between (Passing, Passing Macro).

72.12.3 Passing Macro

- Allows the specification of a macro parameter instead of a specified Passing value.
- Sequence of (Macro Def ID, Passing?).

72.12.4 Passing

- Parameters to be passed to an rt-script.
- Choice between (Generic Value, Content Object Ref).

72.13 Sockets presentation and structural dynamism behaviour

Table 27/T.171 – Overview of Sockets presentation and structural dynamism behaviour

Plug ::= Socket Target Param+, Plug In Param

Plug In Param ::= Plug In | Plug In Macro

Plug In Macro ::= Macro Def ID, Plug In?

Plug In ::= Rt-Component Reference | Component Object Reference | Label | Evaluated Reference

72.13.1 Plug

- Attaches an rt-component or a label to a socket.
- Sequence of (Socket Target Param+, Plug In Param).

72.13.2 Plug In Param

- Parameter of Plug In.
- Choice between (Plug In, Plug In Macro).

72.13.3 Plug In Macro

- Allows the specification of a macro parameter instead of a specified Plug In value.
- Sequence of (Macro Def ID, Plug In?).

72.13.4 Plug In

- Specifies the information to be plugged into a socket.
- Choice between (Rt-Component Reference, Component Object Reference, Label, Evaluated Reference).

72.14 Rt-components rps assignment behaviour

Table 28/T.171 – Overview of Rt-components RPS assignment behaviour

Set RPS Assignment ::= Rt-Component Target Param+, RPS Assignment Param

RPS Assignment Param ::= RPS Assignment | RPS Assignment Macro

RPS Assignment Macro ::= Macro Def ID, RPS Assignment?

72.14.1 Set RPS Assignment

- Sets the RPS assignment of the target.
- Sequence of (Rt-Component Target Param+, RPS Assignment Param).

72.14.2 RPS Assignment Param

- Parameter of RPS Assignment.
- Choice between (RPS Assignment, RPS Assignment Macro).

72.14.3 RPS Assignment Macro

- Allows the specification of a macro parameter instead of a specified RPS Assignment value.
- Sequence of (Macro Def ID, RPS Assignment?).

72.15 Rt-components perceptability behaviour

Table 29/T.171 – Overview of Rt-components perceptability behaviour

Set Perceptability ::= Rt-Component Target Param+, Perceptability Param, Transition Duration Param?

Perceptability Param ::= Generic Ratio Param

Set Presentation Priority ::= Rt-Component Target Param+, Presentation Priority Param, Transition Duration Param?

Presentation Priority Param ::= Presentation Priority | Presentation Priority Macro

Presentation Priority Macro ::= Macro Def ID, Presentation Priority?

72.15.1 Set Perceptability

- This action enables to change the perceptability of an rt-component or a root rt-component.
- Sequence of (Rt-Component Target Param+, Perceptability Param, Transition Duration Param?).

72.15.2 Perceptability Param

- Perceptability to be assigned to rt-component.
- Generic Ratio Param.

72.15.3 Set Presentation Priority

- This action enables to change the presentation priority of an rt-component or a root rt-component.
- Sequence of (Rt-Component Target Param+, Presentation Priority Param, Transition Duration Param?).

72.15.4 Presentation Priority Param

- Parameter of Presentation Priority.
- Choice between (Presentation Priority, Presentation Priority Macro).

72.15.5 Presentation Priority Macro

- Allows the specification of a macro parameter instead of a specified Presentation Priority value.
- Sequence of (Macro Def ID, Presentation Priority?).

72.16 Rt-components temporal behaviour

Table 30/T.171 – Overview of Rt-components temporal behaviour

Set OVD ::= Rt-Component Target Param+, Initial Point Spec Param, Terminal Point Spec Param

Set CTP ::= Rt-Component Target Param+, Current Point Spec Param

Set Temporal Termination ::= Rt-Target Param+, Temporal Termination Param

Temporal Termination Param ::= Temporal Termination | Temporal Termination Macro

Temporal Termination Macro ::= Macro Def ID, Temporal Termination?

Set PVD Position ::= Socket Target Param+, Temporal Position Param

Temporal Position Param ::= Point Spec Param

Set GTF ::= Rt-Component Target Param+, GTF Param

GTF Param ::= GF Param

Set Timestones ::= Rt-Component Target Param+, Timestone Spec Param+

Timestone Spec Param ::= Timestone Spec | Timestone Spec Macro

Timestone Spec Macro ::= Macro Def ID, Timestone Spec?

Timestone Spec ::= Timestone+, Update Command

Timestone ::= Timestone ID, Timestone Position, Number Of Repetitions

Timestone Position ::= Point Spec

Number Of Repetitions ::= Generic Integer | *#infinite*

72.16.1 Set OVD

- The first Point Spec Param specifies the initial temporal position of the OVD. The second specifies the terminal temporal position of the OVD.
- Sequence of (Rt-Component Target Param+, Initial Point Spec Param, Terminal Point Spec Param).

72.16.2 Set CTP

- This action specifies the CTP within the OVD.
- Sequence of (Rt-Component Target Param+, Current Point Spec Param).

72.16.3 Set Temporal Termination

- Specifies the action to be performed once the terminal temporal position of the rt-component is reached.
- Sequence of (Rt-Target Param+, Temporal Termination Param).

72.16.4 Temporal Termination Param

- Parameter of Temporal Termination.
- Choice between (Temporal Termination, Temporal Termination Macro).

72.16.5 Temporal Termination Macro

- Allows the specification of a macro parameter instead of a specified Temporal Termination value.
- Sequence of (Macro Def ID, Temporal Termination?).

72.16.6 Set PVD Position

- Binds child PVDs to their parent OVD.
- Sequence of (Socket Target Param+, Temporal Position Param).

72.16.7 Temporal Position Param

- Specifies temporal position either by an absolute value along a temporal axis or a relative value against a temporal duration.
- Point Spec Param.

72.16.8 Set GTF

- This action defines the presentation speed of an rt-component.
- Sequence of (Rt-Component Target Param+, GTF Param).

72.16.9 GTF Param

- Generic ratio or default-GF.
- GF Param.

72.16.10 Set Timestones

- Sets timestones.
- Sequence of (Rt-Component Target Param+, Timestone Spec Param+).

72.16.11 Timestone Spec Param

- Parameter of Timestone Spec.
- Choice between (Timestone Spec, Timestone Spec Macro).

72.16.12 Timestone Spec Macro

- Allows the specification of a macro parameter instead of a specified Timestone Spec value.
- Sequence of (Macro Def ID, Timestone Spec?).

72.16.13 Timestone Spec

- A set of timestones and the update commands to be applied.
- Sequence of (Timestone+, Update Command).

72.16.14 Timestone

- A pair of a timestone identifier and a timestone position and the number of repetitions.
- Sequence of (Timestone ID, Timestone Position, Number of Repetitions).

72.16.15 Timestone Position

- Specified by point specification.
- Point Spec.

72.16.16 Number Of Repetitions

- Specifies the number of repetitions.
- Choice between (Generic Integer, infinite).

72.17 Rt-components spatial behaviour

Table 31/T.171 – Overview of Rt-components spatial behaviour

<p>Set Aspect Ratio ::= Rt-Component Target Param+, Aspect Ratio Param</p> <p>Aspect Ratio Param ::= Aspect Ratio Aspect Ratio Macro</p> <p>Aspect Ratio Macro ::= Macro Def ID, Aspect Ratio?</p> <p>Set Resizing Strategy ::= Rt-Composite Target Param+, Resizing Strategy Param</p> <p>Resizing Strategy Param ::= Resizing Strategy Resizing Strategy Macro</p> <p>Resizing Strategy Macro ::= Macro Def ID, Resizing Strategy?</p> <p>Set OVS Proj Strategy ::= Rt-Component Target Param+, OVS Proj Strategy Param</p> <p>OVS Proj Strategy Param ::= OVS Proj Strategy OVS Proj Strategy Macro</p> <p>OVS Proj Strategy Macro ::= Macro Def ID, OVS Proj Strategy?</p> <p>Set OVS ::= Rt-Component Target Param+, Size Spec Param, Transition Duration Param?</p> <p>Set OAP ::= Rt-Component Target Param+, OAP Param</p> <p>OAP Param ::= Spatial Position Spec Param</p> <p>Set OVS Position ::= Rt-Component Target Param+, OVS Position Param, Transition Duration Param?</p> <p>OVS Position Param ::= Spatial Position Spec Param</p> <p>Set PAP ::= Rt-Component Target Param+, PAP Param</p> <p>PAP Param ::= Spatial Position Spec Param</p> <p>Set PVS Position ::= Rt-Component Target Param+, PVS Position Param, Transition Duration Param?</p> <p>PVS Position Param ::= Spatial Position Spec Param</p> <p>Set GSF ::= Rt-Component Target Param+, GSF Param, Transition Duration Param?</p> <p>GSF Param ::= GF Param</p> <p>Set User Spatial Control ::= Rt-Component Target Param+, Spatial Control Param+, User Spatial Control Param</p> <p>User Spatial Control Param ::= User Spatial Control User Spatial Control Macro</p> <p>User Spatial Control Macro ::= Macro Def ID, User Spatial Control?</p>
--

72.17.1 Set Aspect Ratio

- Specifies whether the OS is projected with the aspect ratio preserved or not.
- Sequence of (Rt-Component Target Param+, Aspect Ratio Param).

72.17.2 Aspect Ratio Param

- Specifies the aspect ratio preservation.
- Choice between (Aspect Ratio, Aspect Ratio Macro).

72.17.3 Aspect Ratio Macro

- Allows the specification of a macro parameter instead of a specified Aspect Ratio value.
- Sequence of (Macro Def ID, Aspect Ratio?).

72.17.4 Set Resizing Strategy

- Specifies the OS resizing strategy of an rt-composite.
- Sequence of (Rt-Composite Target Param+, Resizing Strategy Param).

72.17.5 Resizing Strategy Param

- Specifies the resizing strategy.
- Choice between (Resizing Strategy, Resizing Strategy Macro).

72.17.6 Resizing Strategy Macro

- Allows the specification of a macro parameter instead of a specified Resizing Strategy value.
- Sequence of (Macro Def ID, Resizing Strategy?).

72.17.7 Set OVS Proj Strategy

- Specifies the OVS Proj Strategy. Choice between “fixed” or “calculated”.
- Sequence of (Rt-Component Target Param+, OVS Proj Strategy Param).

72.17.8 OVS Proj Strategy Param

- Parameter of OVS Proj Strategy.
- Choice between (OVS Proj Strategy, OVS Proj Strategy Macro).

72.17.9 OVS Proj Strategy Macro

- Allows the specification of a macro parameter instead of a specified OVS Proj Strategy value.
- Sequence of (Macro Def ID, OVS Proj Strategy?).

72.17.10 Set OVS

- Set the OVS directly or indirectly depending on the OVS Proj Strategy attribute.
- Sequence of (Rt-Component Target Param+, Size Spec Param, Transition Duration Param?).

72.17.11 Set OAP

- Sets the OAP.
- Sequence of (Rt-Component Target Param+, OAP Param).

72.17.12 OAP Param

- May be specified as an absolute value or a relative value.
- Spatial Position Spec Param.

72.17.13 Set OVS Position

- Sets the OVS position.
- Sequence of (Rt-Component Target Param+, OVS Position Param, Transition Duration Param?).

72.17.14 OVS Position Param

- May be specified as an absolute value or a relative value.
- Spatial Position Spec Param.

72.17.15 Set PAP

- Sets the PAP.
- Sequence of (Rt-Component Target Param+, PAP Param).

72.17.16 PAP Param

- May be specified as an absolute value or a relative value.
- Spatial Position Spec Param.

72.17.17 Set PVS Position

- Sets the PVS position.
- Sequence of (Rt-Component Target Param+, PVS Position Param, Transition Duration Param?).

72.17.18 PVS Position Param

- May be specified as an absolute value or a relative value.
- Spatial Position Spec Param.

72.17.19 Set GSF

- Sets GSF.
- Sequence of (Rt-Component Target Param+, GSF Param, Transition Duration Param?).

72.17.20 GSF Param

- A generic ratio.
- GF Param.

72.17.21 Set User Spatial Control

- Sets one of the user spatial controls.
- Sequence of (Rt-Component Target Param+, Spatial Control Param+, User Spatial Control Param).

72.17.22 User Spatial Control Param

- Choice between “allowed” and “not allowed”.
- Choice between (User Spatial Control, User Spatial Control Macro).

72.17.23 User Spatial Control Macro

- Allows the specification of a macro parameter instead of a specified User Spatial Control value.
- Sequence of (Macro Def ID, User Spatial Control?).

72.18 Rt-components audible behaviour

Table 32/T.171 – Overview of Rt-components audible behaviour

Set CV ::= Rt-Content Target Param+, CV Param, Transition Duration Param?

CV Param ::= Current Point Spec Param

Set GVF ::= GVF Target, GVF Param, Transition Duration Param?

GVF Target ::= Rt-Composite Targets Param | Channel Targets Param

Rt-Composite Targets Param ::= Rt-Composite Target Param+

Channel Targets Param ::= Channel Target Param+

GVF Param ::= GF Param

72.18.1 Set CV

- Specifies the CV of an rt-content.
- Sequence of (Rt-Content Target Param+, CV Param, Transition Duration Param?).

72.18.2 CV Param

- Specified as an absolute value, a relative value, an original point factor or a current point factor.
- Current Point Spec Param.

72.18.3 Set GVF

- This action specifies the GVF of an rt-composite or a channel.
- Sequence of (GVF Target, GVF Param, Transition Duration Param?).

72.18.4 GVF Target

- Targets for GVF.
- Choice between (Rt-Composite Targets Param, Channel Targets Param).

72.18.5 Rt-Composite Targets Param

- Targets for GVF.
- List of (Rt-Composite Target Param).

72.18.6 Channel Targets Param

- Targets for GVF.
- List of (Channel Target Param).

72.18.7 GVF Param

- GVF value to be assigned.
- GF Param.

72.19 Rt-mux stream choice behaviour

Table 33/T.171 – Overview of Rt-mux stream choice behaviour

Set Stream Choice ::= Rt-Mux Target Param+, Stream Spec Param*

Stream Spec Param ::= Stream Spec | Stream Spec Macro

Stream Spec Macro ::= Macro Def ID, Stream Spec?

Stream Spec ::= Stream ID Reference*, Update Command

72.19.1 Set Stream Choice

- This action specifies a list of streams to be chosen in the multiplexed data and to be assigned to the target.
- Sequence of (Rt-Mux Target Param+, Stream Spec Param*).

72.19.2 Stream Spec Param

- Parameter of Stream Spec.
- Choice between (Stream Spec, Stream Spec Macro).

72.19.3 Stream Spec Macro

- Allows the specification of a macro parameter instead of a specified Stream Spec value.
- Sequence of (Macro Def ID, Stream Spec?).

72.19.4 Stream Spec

- Combination of a list of stream ID and an update command.
- Sequence of (Stream ID Reference*, Update Command).

72.20 Interaction behaviour

Table 34/T.171 – Overview of Interaction behaviour

Interaction Status Param ::= Interaction Status | Interaction Status Macro

Interaction Status Macro ::= Macro Def ID, Interaction Status?

Min Interact Required Param ::= Min Interact Required | Min Interact Required Macro

Min Interact Required Macro ::= Macro Def ID, Min Interact Required?

Max Interact Required Param ::= Max Interact Required | Max Interact Required Macro

Max Interact Required Macro ::= Macro Def ID, Max Interact Required?

Set Interaction Ability ::=

Rt-Component Target Param+, Interaction Type Param, Min Interact Required Param, Max Interact Required Param

Set Interaction Status ::= Rt-Component Target Param+, Interaction Type Param, Interaction Status Param

72.20.1 Interaction Status Param

- Parameter of Interaction Status.
- Choice between (Interaction Status, Interaction Status Macro).

72.20.2 Interaction Status Macro

- Allows the specification of a macro parameter instead of a specified Interaction Status value.
- Sequence of (Macro Def ID, Interaction Status?).

72.20.3 Min Interact Required Param

- Parameter of Min Interact Required.
- Choice between (Min Interact Required, Min Interact Required Macro).

72.20.4 Min Interact Required Macro

- Allows the specification of a macro parameter instead of a specified Min Interact Required value.
- Sequence of (Macro Def ID, Min Interact Required?).

72.20.5 Max Interact Required Param

- An integer greater than or equal to 0 to indicate the maximum interaction required for an rt-component.
- Choice between (Max Interact Required, Max Interact Required Macro).

72.20.6 Max Interact Required Macro

- Allows the specification of a macro parameter instead of a specified Max Interact Required value.
- Sequence of (Macro Def ID, Max Interact Required?).

72.20.7 Set Interaction Ability

- This action may be targeted to rt-components to initialise or modify their interactivity behaviour.
- Sequence of (Rt-Component Target Param+, Interaction Type Param, Min Interact Required Param, Max Interact Required Param).

72.20.8 Set Interaction Status

- This action assigns a value to the selection status of an rt-component.
- Sequence of (Rt-Component Target Param+, Interaction Type Param, Interaction Status Param).

72.21 Rt-components style behaviour

Table 35/T.171 – Overview of Rt-components style behaviour

Set Style ::= Rt-Component Target Param+, Catalogued Style Param, Additional Information Param?

Catalogued Style Param ::= Catalogued Style | Catalogued Style Macro

Catalogued Style Macro ::= Macro Def ID, Catalogued Style?

Additional Information Param ::= Generic Value Param

72.21.1 Set Style

- Specifies a style to be applied to a target.
- Sequence of (Rt-Component Target Param+, Catalogued Style Param, Additional Information Param?).

72.21.2 Catalogued Style Param

- Parameter of Catalogued Style.
- Choice between (Catalogued Style, Catalogued Style Macro).

72.21.3 Catalogued Style Macro

- Allows the specification of a macro parameter instead of a specified Catalogued Style value.
- Sequence of (Macro Def ID, Catalogued Style?).

72.21.4 Additional Information Param

- Specifies some additional information required for a style.
- Generic Value Param.

72.22 Rt-contents anchor behaviour

Table 36/T.171 – Overview of Rt-contents anchor behaviour

Attach Anchor ::= Rt-Content Target Param+, Anchor Spec Param+

Anchor Spec Param ::= Anchor Spec | Anchor Spec Macro

Anchor Spec Macro ::= Macro Def ID, Anchor Spec?

Anchor Spec ::= Anchor+, Update Command

Anchor ::= Rt-Content Reference | Evaluated Reference

72.22.1 Attach Anchor

- Attaches anchors to rt-contents.
- Sequence of (Rt-Content Target Param+, Anchor Spec Param+).

72.22.2 Anchor Spec Param

- Parameter of Anchor Spec.
- Choice between (Anchor Spec, Anchor Spec Macro).

72.22.3 Anchor Spec Macro

- Allows the specification of a macro parameter instead of a specified Anchor Spec value.
- Sequence of (Macro Def ID, Anchor Spec?).

72.22.4 Anchor Spec

- A pair of anchors and the update command.
- Sequence of (Anchor+, Update Command).

72.22.5 Anchor

- Specifies an anchor.
- Choice between (Rt-Content Reference, Evaluated Reference).

72.23 Channel availability behaviour

Table 37/T.171 – Overview of Channel availability behaviour

New Channel ::= Channel Target Param+

Delete Channel ::= Channel Target Param+

72.23.1 New Channel

- This action creates channels.
- List of (Channel Target Param).

72.23.2 Delete Channel

- This action removes channels.
- List of (Channel Target Param).

72.24 Channel perceptability behaviour

Table 38/T.171 – Overview of Channel perceptability behaviour

Set Channel Perceptability ::= Channel Target Param+, Channel Perceptability Param

Channel Perceptability Param ::= Channel Perceptability | Channel Perceptability Macro

Channel Perceptability Macro ::= Macro Def ID, Channel Perceptability?

72.24.1 Set Channel Perceptability

- This action changes the channel perceptability of a target.
- Sequence of (Channel Target Param+, Channel Perceptability Param).

72.24.2 Channel Perceptability Param

- Parameter of Channel Perceptability.
- Choice between (Channel Perceptability, Channel Perceptability Macro).

72.24.3 Channel Perceptability Macro

- Allows the specification of a macro parameter instead of a specified Channel Perceptability value.
- Sequence of (Macro Def ID, Channel Perceptability?).

72.25 Channel presentation space behaviour

Table 39/T.171 – Overview of Channel presentation space behaviour

<p>Set CPS ::= Channel Target Param+, CPS Initialisation Param?</p> <p>CPS Initialisation Param ::= CPS Initialisation CPS Initialisation Macro</p> <p>CPS Initialisation Macro ::= Macro Def ID, CPS Initialisation?</p> <p>CPS Initialisation ::= CPS Duration?, CPS Size?</p> <p>CPS Duration ::= Integer <i>#infinite</i> / Evaluated Integer</p> <p>CPS Size ::= Size Evaluated List</p>

72.25.1 Set CPS

- This action specifies an CPS of the target.
- Sequence of (Channel Target Param+, CPS Initialisation Param?).

72.25.2 CPS Initialisation Param

- Parameter of CPS Initialisation.
- Choice between (CPS Initialisation, CPS Initialisation Macro).

72.25.3 CPS Initialisation Macro

- Allows the specification of a macro parameter instead of a specified CPS Initialisation value.
- Sequence of (Macro Def ID, CPS Initialisation?).

72.25.4 CPS Initialisation

- Specifies the duration and the size of the CPS.
- Sequence of (CPS Duration?, CPS Size?).

72.25.5 CPS Duration

- Specifies the duration of the CPS.
- Choice between (Integer, infinite, Evaluated Integer).

72.25.6 CPS Size

- Specifies the spatial size of the CPS.
- Choice between (Size, Evaluated List).

72.26 Channels and rt-components events behaviour

Table 40/T.171 – Overview of Channels and rt-components events behaviour

<p>Set Event ::= Rt-Component Channel Tg Param+, Event Param, Event Data Param?</p> <p>Event Param ::= Generic List Param</p> <p>Event Data Param ::= Generic Value Param</p>
--

72.26.1 Set Event

- The event identifier attribute of the target is set to the specified value.
- Sequence of (Rt-Component Channel Tg Param+, Event Param, Event Data Param?).

72.26.2 Event Param

- Specifies an event.
- Generic List Param.

72.26.3 Event Data Param

- Specifies the data associated with an event.
- Generic Value Param.

73 Elementary Actions

73.1 List of elementary actions

Table 41/T.171 – Overview of list of elementary actions

Elementary Action ::=

Set Event | Set CPS | Set Channel Perceptability | New Channel | Delete Channel | Attach Anchor | Get Style | Set Style | Set Interaction Ability | Set Interaction Status | Set Stream Choice | Set CV | Set GVF | Set Aspect Ratio | Set Resizing Strategy | Set OVS | Set OAP | Set OVS Position | Set PAP | Set PVS Position | Set GSF | Set User Spatial Control | Set OVD | Set CTP | Set Temporal Termination | Set PVD Position | Set GTF | Set Timestones | Set Perceptability | Set Presentation Priority | Set RPS Assignment | Plug | Set Parameters | Run | Stop | New | Delete | Copy | Set Data | Add | Subtract | Link Abort | Activate | Deactivate | Prepare | Destroy | Catalogued Elementary Action | Set Catalogued Attribute | Set Alias | Return | Delay | Extensibility Provision

73.1.1 Elementary Action

- This Recommendation defines a list of elementary actions that may be included in an action object to modify the behaviour of MHEG entities.
- Choice between (Set Event, Set CPS, Set Channel Perceptability, New Channel, Delete Channel, Attach Anchor, Get Style, Set Style, Set Interaction Ability, Set Interaction Status, Set Stream Choice, Set CV, Set GVF, Set Aspect Ratio, Set Resizing Strategy, Set OVS, Set OAP, Set OVS Position, Set PAP, Set PVS Position, Set GSF, Set User Spatial Control, Set OVD, Set CTP, Set Temporal Termination, Set PVD Position, Set GTF, Set Timestones, Set Perceptability, Set Presentation Priority, Set RPS Assignment, Plug, Set Parameters, Run, Stop, New, Delete, Copy, Set Data, Add, Subtract, Link Abort, Activate, Deactivate, Prepare, Destroy, Catalogued Elementary Action, Set Catalogued Attribute, Set Alias, Return, Delay, Extensibility Provision).

73.2 MHEG entity, data, stream, macro parameter and identification useful definitions

Table 42/T.171 – Overview of MHEG entity, data, stream and macro parameter, identification useful definitions

Using-Application ::= <i>\$NULL</i>
Alias ::= String
Identifier ::= Integer
Index ::= Integer
Tail Param ::= Tail Tail Macro
Tail Macro ::= Macro Def ID, Tail?
Tail ::= Index*
External ID ::= Public ID?, System ID?
Public ID ::= <i>\$PrintableString</i>
System ID ::= <i>\$OCTET STRING</i>
Null-Data ::= <i>\$NULL</i>
Mheg ID ::= Application ID?, Mh Number
Application ID ::= <i>\$OCTET STRING</i>
Mh Number ::= Identifier
Null-Mh ::= <i>\$NULL</i>
Stream ID ::= Tail
Rt-Number ::= Identifier
Null-Root-Rt ::= <i>\$NULL</i>
Socket Identification ::= Socket ID Alias
Socket ID ::= Root Rt-Composite Reference, Tail
Channel ID ::= Identifier
Default-Channel ::= <i>\$NULL</i>
Macro Def ID ::= String Integer

73.2.1 Using-Application

- The using application is identified by a reserved identifier: “Using-Application”.
- NULL.

73.2.2 Alias

- Identification of an MHEG entity, a data or a stream. It is a sequence of alphanumeric characters which is used as a convenient substitute for the other addressing mechanism. It is for the using application to map this name to a physical address.
- String.

73.2.3 Identifier

- Identification of an MHEG entity.
- Integer.

73.2.4 Index

- Identification of an inner element. The indexes are numbered from 1 to n by default.
- Integer.

73.2.5 Tail Param

- Parameter of Tail.
- Choice between (Tail, Tail Macro).

73.2.6 Tail Macro

- Allows the specification of a macro parameter instead of a specified Tail value.
- Sequence of (Macro Def ID, Tail?).

73.2.7 Tail

- Identification of an inner element within a hierarchy. The list of indexes designs a path from an outer entity to an inner element.
- List of (Index).

73.2.8 External ID

- Identification of an external information (e.g. an MHEG object, a content data) as defined for by ISO 9070 and ISO 8879.

The technique for reference to external information is based on:

The techniques used in ISO 8879 for “formal public identifiers”.

International Standard ISO 9070 “Registration procedures for public text owner identifier” which provides both ASN.1 and SGML notation for the representation of registered owner identifiers.

The presence of at least one of the two fields is mandatory.

- Sequence of (Public ID?, System ID?).

73.2.9 Public ID

- Character string whose syntax is given by subclause 10.2 of ISO 8879:1986.
Recommendation T.171 uses a simplification of ISO 9070 Informative Annex D shall be used for the encoding of the public identifier.
- PrintableString.

73.2.10 System ID

- The system identifier may be used to identify a particular file within the system. It may also be used to indicate the system which supplies the object. This information may be used to describe a database or telecommunications request for an information.
- OCTET STRING.

73.2.11 Null-Data

- The null data is identified by a reserved identifier: “Null-Data”.
- NULL.

73.2.12 Mheg ID

- Identification of an MHEG interchanged object.
- Sequence of (Application ID?, Mh Number).

73.2.13 Application ID

- Identification of the application supplying the object. The semantics of the identification are defined by the using application.
- OCTET STRING.

73.2.14 MH Number

- Unique identification of the object within the application. A reserved MH-number is used to address a null object: “Null-MH”.
- Identifier.

73.2.15 Null-MH

- The null MHEG object is identified by a reserved identifier: “Null-MH”.
- NULL.

73.2.16 Stream ID

- Unique identification of a stream within a multiplex.
- Tail.

73.2.17 Rt Number

- Unique identification of a root rt-object created from a given model object.
- Identifier.

73.2.18 Null-Root-Rt

- The null root rt-object is identified by a reserved identifier: “Null-Root-Rt”.
- NULL.

73.2.19 Socket Identification

- Identification of a socket.
- Choice between (Socket ID, Alias).

73.2.20 Socket ID

- Identifier of a socket, i.e. an rt-composite element.
- Sequence of (Root Rt-Composite Reference, Tail).

73.2.21 Channel ID

- Identification of a channel.
- Identifier.

73.2.22 Default-Channel

- The default channel is identified by a reserved identifier: “Default-Channel”.
- NULL.

73.2.23 Macro Def ID

- Identification of a macro parameter within an action.
- Choice between (String, Integer).

73.3 References useful definitions

Table 43/T.171 – Overview of References useful definitions

Reference ::= Null-Data Mheg ID Reference External ID Reference Container Element Reference Null-Mh This Stream ID Reference Root Rt-ID Reference Null-Root-Rt Socket ID Reference Channel ID Reference Default-Channel Alias Reference
Alias Reference ::= Alias
Mheg ID Reference ::= Mheg ID
External ID Reference ::= External ID
Tail Reference ::= Tail?, Tail Complement?
Tail Complement ::= #children / #descendants
Data Reference ::= External ID Null-Data Alias Reference
Mh-Reference ::= Mheg ID Reference External ID Reference Container Element Reference Null-Mh This Alias Reference
Action Object Reference ::= Mh-Reference
Link Object Reference ::= Mh-Reference
Model Object Reference ::= Mh-Reference
Script Object Reference ::= Mh-Reference
Component Object Reference ::= Mh-Reference
Content Object Ref ::= Mh-Reference
Non-Mux Content Object Ref ::= Mh-Reference
Multiplexed Content Object Ref ::= Mh-Reference
Stream ID Reference ::= Stream ID
Composite Object Reference ::= Mh-Reference
Container Object Reference ::= Mh-Reference
Container Element Reference ::= Container Object Reference, Tail Reference
This ::= \$NULL
Descriptor Object Reference ::= Mh-Reference
Root Rt-Reference ::= Root Rt-ID Reference Null-Root-Rt Alias Reference
Root Rt-ID Reference ::= Model Object Reference, Rt-Number Reference
Rt-Number Reference ::= Rt-Number #question-mark / #star
Rt-Reference ::= Root Rt-ID Reference Null-Root-Rt Socket ID Reference Alias Reference
Rt-Script Reference ::= Rt-Script ID Reference Null-Root-Rt Alias Reference
Rt-Script ID Reference ::= Script Object Reference, Rt-Number Reference
Root Rt-Component Reference ::= Root Rt-Component ID Ref Null-Root-Rt Alias Reference
Root Rt-Component ID Ref ::= Component Object Reference, Rt-Number Reference
Rt-Component Reference ::= Root Rt-Component ID Ref Null-Root-Rt Socket ID Reference Alias Reference
Root Rt-Content Reference ::= Root Rt-Content ID Ref Null-Root-Rt Alias Reference
Root Rt-Content ID Ref ::= Content Object Ref, Rt-Number Reference

Table 43/T.171 – Overview of References useful definitions (*concluded*)

Rt-Content Reference ::= Root Rt-Content ID Ref Null-Root-Rt Socket ID Reference Alias Reference
Root Rt-Mux Reference ::= Root Rt-Mux ID Reference Null-Root-Rt Alias Reference
Root Rt-Mux ID Reference ::= Multiplexed Content Object Ref, Rt-Number Reference
Rt-Mux Reference ::= Root Rt-Mux ID Reference Null-Root-Rt Socket ID Reference Alias Reference
Root Rt-Composite Reference ::= Root Rt-Composite ID Ref Null-Root-Rt Alias Reference
Root Rt-Composite ID Ref ::= Composite Object Reference, Rt-Number Reference
Rt-Composite Reference ::= Root Rt-Composite ID Ref Null-Root-Rt Socket ID Reference Alias Reference
Socket Reference ::= Socket ID Reference Alias Reference
Socket ID Reference ::= Root Rt-Composite Reference, Socket Tail Reference
Socket Tail Reference ::= Tail?, Socket Tail Complement?
Socket Tail Complement ::= Tail Complement <i>#question-mark-child</i> <i>#question-mark-descendant</i>
Channel Reference ::= Channel ID Reference Default-Channel Alias Reference
Channel ID Reference ::= Channel ID
Rt-Component Channel Ref ::= Rt-Component Reference Channel Reference

73.3.1 Reference

- A reference is used to address an MHEG entity, a data or a stream.
- Choice between (Null-Data, Mheg ID Reference, External ID Reference, Container Element Reference, Null-Mh, This, Stream ID Reference, Root Rt-ID Reference, Null-Root-Rt, Socket ID Reference, Channel ID Reference, Default-Channel, Alias Reference).

73.3.2 Alias Reference

- Reference to a single or multiple MHEG entity or data or stream using the assigned alias.
- Alias.

73.3.3 Mheg ID Reference

- Reference to an MHEG object using its MHEG identifier.
- Mheg ID.

73.3.4 External ID Reference

- Reference to an MHEG entity or a data using its external identifier.
- External ID.

73.3.5 Tail Reference

- Reference to a given element using its tail identification, a tail complement may be provided. The tail can be omitted, in which case the tail complement is applied to the outer entity itself.
- Sequence of (Tail?, Tail Complement?).

73.3.6 Tail Complement

- The tail complement is used to reference the children (one generation) or the descendants (all generations) of the element referenced using its tail or of the outer entity if the tail is omitted.
- Choice between (children, descendants).

73.3.7 Data Reference

- Reference to a data of a content or a script object using one of its possible identification.
- Choice between (External ID, Null-Data, Alias Reference).

73.3.8 MH-Reference

- Reference to an MHEG object using one of its possible identifications.
- Choice between (Mheg ID Reference, External ID Reference, Container Element Reference, Null-MH, This, Alias Reference).

73.3.9 Action Object Reference

- Reference to an action object.
- MH-Reference.

73.3.10 Link Object Reference

- Reference to a link object.
- MH-Reference.

73.3.11 Model Object Reference

- Reference to a model object. When the model object to be referenced is a composite object, “this” may be used within link and action objects to express local addressing within the composite or the container object.
- MH-Reference.

73.3.12 Script Object Reference

- Reference to a script object.
- MH-Reference.

73.3.13 Component Object Reference

- Reference to a component object.
- MH-Reference.

73.3.14 Content Object Ref

- Reference to a content object or a multiplexed content object.
- MH-Reference.

73.3.15 Non-Mux Content Object Ref

- Reference to a non-mux content object, i.e. a pure content object.
- MH-Reference.

73.3.16 Multiplexed Content Object Ref

- Reference to a multiplexed content object.
- MH-Reference.

73.3.17 Stream ID Reference

- Reference to a stream using its stream identifier.
- Stream ID.

73.3.18 Composite Object Reference

- Reference to a composite object, “this” may be used within link and action objects to express local addressing facilities within the composite object.
- MH-Reference.

73.3.19 Container Object Reference

- Reference to a container object, “this” can be used within link and action objects for local addressing facilities within the composite or the container object.
- MH-Reference.

73.3.20 Container Element Reference

- Reference to a container element using a reference to the container object defining this element followed by a reference to the element itself.
- Sequence of (Container Object Reference, Tail Reference).

73.3.21 This

- When the MHEG entity to be referenced is a composite or a container object, “this” can be used within link and action objects for local addressing facilities within the composite or the container object itself.
- NULL.

73.3.22 Descriptor Object Reference

- Reference to a descriptor object.
- MH-Reference.

73.3.23 Root Rt-Reference

- Reference to a root rt-object using one of its possible identifications.
- Choice between (Root Rt-ID Reference, Null-Root-Rt, Alias Reference).

73.3.24 Root Rt-ID Reference

- Reference to a root rt-identifier using a reference to the model object from which it is created and a reference to the rt-number assigned to it.
- Sequence of (Model Object Reference, Rt-Number Reference).

73.3.25 Rt-Number Reference

- Reference to a specified root rt-object (rt-number) or to a dynamically determined root rt-object (?) or to all root rt-objects created from a given model object (*).
- Choice between (Rt-Number, question-mark, star).

73.3.26 Rt-Reference

- Reference to an rt-object, i.e. either a root (rt-script, root rt-component) or a socket.
- Choice between (Root Rt-ID Reference, Null-Root-Rt, Socket ID Reference, Alias Reference).

73.3.27 Rt-Script Reference

- Reference to an rt-script using one of its possible identifications.
- Choice between (Rt-Script ID Reference, Null-Root-Rt, Alias Reference).

73.3.28 Rt-Script ID Reference

- Reference to an rt-script identifier using a reference to the model object from which it is created and a reference to the rt-number assigned to it.
- Sequence of (Script Object Reference, Rt-Number Reference).

73.3.29 Root Rt-Component Reference

- Reference to a root rt-component using one of its possible identifications.
- Choice between (Root Rt-Component ID Ref, Null-Root-Rt, Alias Reference).

73.3.30 Root Rt-Component ID Ref

- Reference to a root rt-component identifier using a reference to the model object from which it is created and a reference to the rt-number assigned to it.
- Sequence of (Component Object Reference, Rt-Number Reference).

73.3.31 Rt-Component Reference

- Reference to an rt-component, i.e. either a root or a socket.
- Choice between (Root Rt-Component ID Ref, Null-Root-Rt, Socket ID Reference, Alias Reference).

73.3.32 Root Rt-Content Reference

- Reference to a root rt-content using one of its possible identifications.
- Choice between (Root Rt-Content ID Ref, Null-Root-Rt, Alias Reference).

73.3.33 Root Rt-Content ID Ref

- Reference to a root rt-content identifier using a reference to the model object from which it is created and a reference to the rt-number assigned to it.
- Sequence of (Content Object Ref, Rt-Number Reference).

73.3.34 Rt-Content Reference

- Reference to an rt-content, i.e. either a root or a socket.
- Choice between (Root Rt-Content ID Ref, Null-Root-Rt, Socket ID Reference, Alias Reference).

73.3.35 Root Rt-Mux Reference

- Reference to a root rt-mux using one of its possible identifications.
- Choice between (Root Rt-Mux ID Reference, Null-Root-Rt, Alias Reference).

73.3.36 Root Rt-Mux ID Reference

- Reference to a root rt-mux identifier using a reference to the model object from which it is created and a reference to the rt-number assigned to it.
- Sequence of (Multiplexed Content Object Ref, Rt-Number Reference).

73.3.37 Rt-Mux Reference

- Reference to an rt-mux, i.e. either a root or a socket.
- Choice between (Root Rt-Mux ID Reference, Null-Root-Rt, Socket ID Reference, Alias Reference).

73.3.38 Root Rt-Composite Reference

- Reference to a root rt-composite using one of its possible identifications.
- Choice between (Root Rt-Composite ID Ref, Null-Root-Rt, Alias Reference).

73.3.39 Root Rt-Composite ID Ref

- Reference to a root rt-composite identifier using a reference to the model object from which it is created and a reference to the rt-number assigned to it.
- Sequence of (Composite Object Reference, Rt-Number Reference).

73.3.40 Rt-Composite Reference

- Reference to an rt-composite, i.e. either a root or a socket.
- Choice between (Root Rt-Composite ID Ref, Null-Root-Rt, Socket ID Reference, Alias Reference).

73.3.41 Socket Reference

- Reference to a socket, using one of its possible identifications. There is no specific reference to a null socket because a null socket, also called an empty socket, is referenced as any other sockets.
- Choice between (Socket ID Reference, Alias Reference).

73.3.42 Socket ID Reference

- Reference to a socket identifier using a reference to the rt-composite defining this socket followed by a reference to the socket itself. A socket is an element of an rt-composite.
- Sequence of (Root Rt-Composite Reference, Socket Tail Reference).

73.3.43 Socket Tail Reference

- Reference to a given socket using its tail identification, a tail complement may be provided. The tail can be omitted, in that case the socket tail complement is applied to the root rt-composite itself.
- Sequence of (Tail?, Socket Tail Complement?).

73.3.44 Socket Tail Complement

- The socket tail complement is used to reference the children or descendants or a single dynamically determined socket within the children (?child), or a single dynamically determined socket within the descendants (? descendant) of the socket referenced by the tail or of the root rt-composite if the tail is omitted.
- Choice between (Tail Complement, question-mark-child, question-mark-descendant).

73.3.45 Channel Reference

- Reference to a Channel Identifier using one of its possible identifications.
- Choice between (Channel ID Reference, Default-Channel, Alias Reference).

73.3.46 Channel ID Reference

- Reference to a channel using its channel identifier.
- Channel ID.

73.3.47 Rt-Component Channel Ref

- Reference to an rt-component or to a channel using one of its possible identifications.
- Choice between (Rt-Component Reference, Channel Reference).

73.4 Useful definitions of targets

Table 44/T.171 – Overview of useful definitions of targets

Target Param ::= Target Target Macro
Target Macro ::= Macro Def ID, Target?
Target ::= Generic Reference
Mh-Target Param ::= Mh-Target Mh-Target Macro
Mh-Target Macro ::= Macro Def ID, Mh-Target?
Mh-Target ::= Mh-Reference Evaluated Reference
Link Target Param ::= Mh-Target Param
Content Target Param ::= Mh-Target Param
Rt-Target Param ::= Rt-Target Rt-Target Macro
Rt-Target Macro ::= Macro Def ID, Rt-Target?
Rt-Target ::= Rt-Reference Evaluated Reference
Rt-Script Target Param ::= Rt-Script Target Rt-Script Target Macro
Rt-Script Target Macro ::= Macro Def ID, Rt-Script Target?
Rt-Script Target ::= Rt-Script Reference Evaluated Reference

Table 44/T.171 – Overview of useful definitions of targets (concluded)

Rt-Component Target Param ::= Rt-Component Target Rt-Component Target Macro
Rt-Component Target Macro ::= Macro Def ID, Rt-Component Target?
Rt-Component Target ::= Rt-Component Reference Evaluated Reference
Rt-Content Target Param ::= Rt-Content Target Rt-Content Target Macro
Rt-Content Target Macro ::= Macro Def ID, Rt-Content Target?
Rt-Content Target ::= Rt-Content Reference Evaluated Reference
Rt-Mux Target Param ::= Rt-Mux Target Rt-Mux Target Macro
Rt-Mux Target Macro ::= Macro Def ID, Rt-Mux Target?
Rt-Mux Target ::= Rt-Mux Reference Evaluated Reference
Rt-Composite Target Param ::= Rt-Composite Target Rt-Composite Target Macro
Rt-Composite Target Macro ::= Macro Def ID, Rt-Composite Target?
Rt-Composite Target ::= Rt-Composite Reference Evaluated Reference
Socket Target Param ::= Socket Target Socket Target Macro
Socket Target Macro ::= Macro Def ID, Socket Target?
Socket Target ::= Socket Reference Evaluated Reference
Return Target Param ::= Generic Integer Param Using-Application
Channel Target Param ::= Channel Target Channel Target Macro
Channel Target Macro ::= Macro Def ID, Channel Target?
Channel Target ::= Channel Reference Evaluated Reference
Rt-Component Channel Tg Param ::= Rt-Component Channel Tg Rt-Component Channel Tg Macro
Rt-Component Channel Tg Macro ::= Macro Def ID, Rt-Component Channel Tg?
Rt-Component Channel Tg ::= Rt-Component Channel Ref Evaluated Reference

73.4.1 Target Param

- Parameter of Target.
- Choice between (Target, Target Macro).

73.4.2 Target Macro

- Allows the specification of a macro parameter instead of a specified Target value.
- Sequence of (Macro Def ID, Target?).

73.4.3 Target

- A target of an action addresses an MHEG entity using its reference.
- Generic Reference.

73.4.4 MH-Target Param

- Parameter of MH-Target.
- Choice between (MH-Target, MH-Target Macro).

73.4.5 MH-Target Macro

- Allows the specification of a macro parameter instead of a specified MH-Target value.
- Sequence of (Macro Def ID, MH-Target?).

73.4.6 MH-Target

- An MH-target of an action addresses an MHEG object using its reference.
- Choice between (MH-Reference, Evaluated Reference).

73.4.7 Link Target Param

- A link target of an action addresses a link object using its reference.
- MH-Target Param.

73.4.8 Content Target Param

- A content target of an action addresses a content object using its reference.
- MH-Target Param.

73.4.9 Rt-Target Param

- Parameter of Rt-Target.
- Choice between (Rt-Target, Rt-Target Macro).

73.4.10 Rt-Target Macro

- Allows the specification of a macro parameter instead of a specified Rt-Target value.
- Sequence of (Macro Def ID, Rt-Target?).

73.4.11 Rt-Target

- An rt-target of an action addresses an rt-object using its reference.
- Choice between (Rt-Reference, Evaluated Reference).

73.4.12 Rt-Script Target Param

- Parameter of Rt-Script Target.
- Choice between (Rt-Script Target, Rt-Script Target Macro).

73.4.13 Rt-Script Target Macro

- Allows the specification of a macro parameter instead of a specified Rt-Script Target value.
- Sequence of (Macro Def ID, Rt-Script Target?).

73.4.14 Rt-Script Target

- An rt-script target of an action addresses an rt-script object using its reference.
- Choice between (Rt-Script Reference, Evaluated Reference).

73.4.15 Rt-Component Target Param

- Parameter of Rt-Component Target.
- Choice between (Rt-Component Target, Rt-Component Target Macro).

73.4.16 Rt-Component Target Macro

- Allows the specification of a macro parameter instead of a specified Rt-Component Target value.
- Sequence of (Macro Def ID, Rt-Component Target?).

73.4.17 Rt-Component Target

- An rt-component target of an action addresses an rt-component object using its reference.
- Choice between (Rt-Component Reference, Evaluated Reference).

73.4.18 Rt-Content Target Param

- Parameter of Rt-Content Target.
- Choice between (Rt-Content Target, Rt-Content Target Macro).

73.4.19 Rt-Content Target Macro

- Allows the specification of a macro parameter instead of a specified Rt-Content Target value.
- Sequence of (Macro Def ID, Rt-Content Target?).

73.4.20 Rt-Content Target

- An rt-content target of an action addresses an rt-content object using its reference.
- Choice between (Rt-Content Reference, Evaluated Reference).

73.4.21 Rt-Mux Target Param

- Parameter of Rt-Mux Target.
- Choice between (Rt-Mux Target, Rt-Mux Target Macro).

73.4.22 Rt-Mux Target Macro

- Allows the specification of a macro parameter instead of a specified Rt-Mux Target value.
- Sequence of (Macro Def ID, Rt-Mux Target?).

73.4.23 Rt-Mux Target

- An rt-mux target of an action addresses an rt-mux object using its reference.
- Choice between (Rt-Mux Reference, Evaluated Reference).

73.4.24 Rt-Composite Target Param

- Parameter of Rt-Composite Target.
- Choice between (Rt-Composite Target, Rt-Composite Target Macro).

73.4.25 Rt-Composite Target Macro

- Allows the specification of a macro parameter instead of a specified Rt-Composite Target value.
- Sequence of (Macro Def ID, Rt-Composite Target?).

73.4.26 Rt-Composite Target

- An rt-composite target of an action addresses an rt-composite object using its reference.
- Choice between (Rt-Composite Reference, Evaluated Reference).

73.4.27 Socket Target Param

- Parameter of Socket Target.
- Choice between (Socket Target, Socket Target Macro).

73.4.28 Socket Target Macro

- Allows the specification of a macro parameter instead of a specified Socket Target value.
- Sequence of (Macro Def ID, Socket Target?).

73.4.29 Socket Target

- A socket target of an action addresses socket using its reference.
- Choice between (Socket Reference, Evaluated Reference).

73.4.30 Return Target Param

- When provided it is up to the implementation to map the parameter to the appropriate recipient.
- Choice between (Generic Integer Param, Using-Application).

73.4.31 Channel Target Param

- Parameter of Channel Target.
- Choice between (Channel Target, Channel Target Macro).

73.4.32 Channel Target Macro

- Allows the specification of a macro parameter instead of a specified Channel Target value.
- Sequence of (Macro Def ID, Channel Target?).

73.4.33 Channel Target

- A channel target of an action addresses a channel using its reference.
- Choice between (Channel Reference, Evaluated Reference).

73.4.34 Rt-Component Channel Tg Param

- Parameter of Rt-Component Channel Tg.
- Choice between (Rt-Component Channel Tg, Rt-Component Channel Tg Macro).

73.4.35 Rt-Component Channel Tg Macro

- Allows the specification of a macro parameter instead of a specified Rt-Component Channel Tg value.
- Sequence of (Macro Def ID, Rt-Component Channel Tg?).

73.4.36 Rt-Component Channel Tg

- An rt-component or channel target of an action addresses an rt-component or a channel using its reference.
- Choice between (Rt-Component Channel Ref, Evaluated Reference).

73.5 Generic value useful definitions

Table 45/T.171 – Overview of Generic value useful definitions

Generic Value Param ::= Generic Value | Generic Value Macro

Generic Value Macro ::= Macro Def ID, Generic Value?

Generic Value ::= Value | Evaluated Value

Value ::= Boolean | Numeric | Ratio | String | List | Reference

Generic Boolean Param ::= Generic Boolean | Generic Boolean Macro

Generic Boolean Macro ::= Macro Def ID, Generic Boolean?

Generic Boolean ::= Boolean | Evaluated Boolean

Boolean ::= *\$BOOLEAN*

Generic Numeric Param ::= Generic Numeric | Generic Numeric Macro

Generic Numeric Macro ::= Macro Def ID, Generic Numeric?

Generic Numeric ::= Numeric | Evaluated Numeric

Numeric ::= *\$INTEGER*

Generic Integer Param ::= Generic Integer | Generic Integer Macro

Generic Integer Macro ::= Macro Def ID, Generic Integer?

Generic Integer ::= Integer | Evaluated Integer

Integer ::= *\$INTEGER*

Generic Ratio Param ::= Generic Ratio | Generic Ratio Macro

Generic Ratio Macro ::= Macro Def ID, Generic Ratio?

Generic Ratio ::= Ratio | Evaluated Ratio

Table 45/T.171 – Overview of Generic value useful definitions (concluded)

Ratio ::= Numerator, Denominator?
Numerator ::= Integer
Denominator ::= Integer
Generic String Param ::= Generic String Generic String Macro
Generic String Macro ::= Macro Def ID, Generic String?
Generic String ::= String Evaluated String
String ::= <i>\$GraphicString</i>
Generic Reference Param ::= Generic Reference Generic Reference Macro
Generic Reference Macro ::= Macro Def ID, Generic Reference?
Generic Reference ::= Reference Evaluated Reference
Generic List Param ::= Generic List Generic List Macro
Generic List Macro ::= Macro Def ID, Generic List?
Generic List ::= List Evaluated List
List ::= Value*
Generic List Elt ID Param ::= Generic List Elt ID Generic List Elt ID Macro
Generic List Elt ID Macro ::= Macro Def ID, Generic List Elt ID?
Generic List Elt ID ::= Tail

73.5.1 Generic Value Param

- Parameter of Generic Value.
- Choice between (Generic Value, Generic Value Macro).

73.5.2 Generic Value Macro

- Allows the specification of a macro parameter instead of a specified Generic Value value.
- Sequence of (Macro Def ID, Generic Value?).

73.5.3 Generic Value

- Provides a mechanism to specify values in a generic way. The generic values are typically used to express elementary action parameters, conditions or MHEG object attribute.
- Choice between (Value, Evaluated Value).

73.5.4 Value

- Specifications of all possible values supported by this Recommendation.
- Choice between (Boolean, Numeric, Ratio, String, List, Reference).

73.5.5 Generic Boolean Param

- Parameter of Generic Boolean.
- Choice between (Generic Boolean, Generic Boolean Macro).

73.5.6 Generic Boolean Macro

- Allows the specification of a macro parameter instead of a specified Generic Boolean value.
- Sequence of (Macro Def ID, Generic Boolean?).

73.5.7 Generic Boolean

- A generic boolean shall have one of the following values: true, false, a result of a get action.
- Choice between (Boolean, Evaluated Boolean).

73.5.8 Boolean

- A boolean is encoded in this Recommendation as an ASN.1 BOOLEAN.
- BOOLEAN.

73.5.9 Generic Numeric Param

- Parameter of Generic Numeric.
- Choice between (Generic Numeric, Generic Numeric Macro).

73.5.10 Generic Numeric Macro

- Allows the specification of a macro parameter instead of a specified Generic Numeric value.
- Sequence of (Macro Def ID, Generic Numeric?).

73.5.11 Generic Numeric

- A generic numeric has one of the following values: a numeric, a result of a get action.
- Choice between (Numeric, Evaluated Numeric).

73.5.12 Numeric

- The design of this Recommendation does not limit the numerical values in any way. The values may be integers, reals or complex. However the coded representations may impose limitations. The ASN.1 syntax defined in this Recommendation encodes numeric as an ASN.1 INTEGER, and it is recommended that authors assume this limitation.
- INTEGER.

73.5.13 Generic Integer Param

- Parameter of Generic Integer.
- Choice between (Generic Integer, Generic Integer Macro).

73.5.14 Generic Integer Macro

- Allows the specification of a macro parameter instead of a specified Generic Integer value.
- Sequence of (Macro Def ID, Generic Integer?).

73.5.15 Generic Integer

- A generic integer has one of the following values: an integer, a result of a get action.
- Choice between (Integer, Evaluated Integer).

73.5.16 Integer

- An integer is encoded in this Recommendation as an ASN.1 INTEGER.
- INTEGER.

73.5.17 Generic Ratio Param

- Parameter of Generic Ratio.
- Choice between (Generic Ratio, Generic Ratio Macro).

73.5.18 Generic Ratio Macro

- Allows the specification of a macro parameter instead of a specified Generic Ratio value.
- Sequence of (Macro Def ID, Generic Ratio?).

73.5.19 Generic Ratio

- A generic ratio has one of the following values: a ratio, a result of a get action.
- Choice between (Ratio, Evaluated Ratio).

73.5.20 Ratio

- A ratio value is a pair of integers (m, n) with m the numerator and n the denominator of a fraction. If the denominator is omitted, it is assumed to be 100, thus interpreting the numerator as a percentage.
The numerator shall be greater than 1 and the denominator may be negative.
- Sequence of (Numerator, Denominator?).

73.5.21 Numerator

- Numerator of a fraction.
- Integer.

73.5.22 Denominator

- Denominator of a fraction.
- Integer.

73.5.23 Generic String Param

- Parameter of Generic String.
- Choice between (Generic String, Generic String Macro).

73.5.24 Generic String Macro

- Allows the specification of a macro parameter instead of a specified Generic String value.
- Sequence of (Macro Def ID, Generic String?).

73.5.25 Generic String

- A generic string has one of the following values: a string, a result of a get action.
- Choice between (String, Evaluated String).

73.5.26 String

- A string is encoded in this Recommendation as an ASN.1 Graphic String. This allows for international string specification.
- GraphicString.

73.5.27 Generic Reference Param

- Parameter of Generic Reference.
- Choice between (Generic Reference, Generic Reference Macro).

73.5.28 Generic Reference Macro

- Allows the specification of a macro parameter instead of a specified Generic Reference value.
- Sequence of (Macro Def ID, Generic Reference?).

73.5.29 Generic Reference

- A generic reference has one of the following values: a reference, a result of a get action.
- Choice between (Reference, Evaluated Reference).

73.5.30 Generic List Param

- Parameter of Generic List.
- Choice between (Generic List, Generic List Macro).

73.5.31 Generic List Macro

- Allows the specification of a macro parameter instead of a specified Generic List value.
- Sequence of (Macro Def ID, Generic List?).

73.5.32 Generic List

- A generic list has one of the following values: a list, a result of a get action.
- Choice between (List, Evaluated List).

73.5.33 List

- A list of values.
- List of (Value).

73.5.34 Generic List Elt ID Param

- Parameter of Generic List Elt ID.
- Choice between (Generic List Elt ID, Generic List Elt ID Macro).

73.5.35 Generic List Elt ID Macro

- Allows the specification of a macro parameter instead of a specified Generic List Elt ID value.
- Sequence of (Macro Def ID, Generic List Elt ID?).

73.5.36 Generic List Elt ID

- Identification of a generic value which is an element of a generic list.
- Tail.

73.6 Evaluated values useful definitions

Table 46/T.171 – Overview of Evaluated values useful definitions

Get Event Data ::= Rt-Component Channel Tg Param
Get Event ::= Rt-Component Channel Tg Param
Get Channel Perceptability ::= Channel Target Param
Get Channel Availability Status ::= Channel Target Param
Get Style ::= Rt-Component Target Param
Get Number Of Interacted Sockets ::= Rt-Composite Target Param, Interaction Type Param
Get Interaction Status ::= Rt-Component Target Param, Interaction Type Param
Get Max Interact Required ::= Rt-Component Target Param, Interaction Type Param
Get Min Interact Required ::= Rt-Component Target Param, Interaction Type Param
Get Interaction Ability ::= Rt-Component Target Param, Interaction Type Param
Get Stream Chosen State ::= Rt-Mux Target Param, Stream Identification Param
Get Stream Choice ::= Rt-Mux Target Param
Get GVF ::= Rt-Composite Target Param Channel Target Param
Get PCV ::= Rt-Content Target Param
Get CV ::= Rt-Content Target Param
Get OV ::= Rt-Content Target Param

Table 46/T.171 – Overview of Evaluated values useful definitions (cont.)

Get User Spatial Control ::= Rt-Component Target Param, Spatial Control Param
Get GSF ::= Rt-Component Target Param, Expected Axis Result Param
Get PVS Position ::= Rt-Component Target Param, Point Type Param, Expected Axis Result Param
Get PAP ::= Rt-Component Target Param, Point Type Param, Expected Axis Result Param
Get PVS ::= Rt-Component Target Param, Point Type Param, Expected Axis Result Param
Get OVS Position ::= Rt-Component Target Param, Point Type Param, Expected Axis Result Param
Get OAP ::= Rt-Component Target Param, Point Type Param, Expected Axis Result Param
Get OVS ::= Rt-Component Target Param, Point Type Param, Expected Axis Result Param
Get OVS Proj Strategy ::= Rt-Component Target Param
Get Resizing Strategy ::= Rt-Composite Target Param
Get Aspect Ratio ::= Rt-Component Target Param
Get POS ::= Rt-Component Target Param, Expected Axis Result Param
Get OS ::= Rt-Component Target Param, Expected Axis Result Param
Get Timestone Status ::= Rt-Component Target Param
Get GTF ::= Rt-Component Target Param
Get PVD Position ::= Rt-Component Target Param
Get Temporal Termination ::= Rt-Component Target Param
Get CTP ::= Rt-Component Target Param
Get PVD ::= Rt-Component Target Param, Expected PVD Result Param
Get OVD ::= Rt-Component Target Param, Expected OVD Result Param
Get POD ::= Rt-Component Target Param
Get OD ::= Rt-Component Target Param
Get Presentation Priority ::= Rt-Component Target Param
Get Perceptability ::= Rt-Component Target Param
Get RPS Assignment ::= Rt-Component Target Param
Get Rt-Composite Address ::= Rt-Composite Target Param, Navigation Command Param
Get Termination Status ::= Rt-Script Target Param
Get Running Status ::= Rt-Target Param
Get Rt-Availability Status ::= Rt-Target Param
Get Data ::= Content Target Param, Generic List Elt ID Param?
Get Activation Status ::= Link Target Param
Get Preparation Status ::= Mh-Target Param
Get Catalogued Attribute ::= Target Param, Cat Ext Attribute Param
Get Boolean ::= *\$NULL*
Get Numeric ::= Get Integer

Table 46/T.171 – Overview of Evaluated values useful definitions (concluded)

<p>Get Integer ::=</p> <p>Get Preparation Status Get Activation Status Get Rt-Availability Status Get Running Status Get Termination Status Get Presentation Priority Get OD Get POD Get PVD Get CTP Get Temporal Termination Get PVD Position Get Timestone Status Get Aspect Ratio Get Resizing Strategy Get OVS Proj Strategy Get User Spatial Control Get CV Get PCV Get Stream Chosen State Get Min Interact Required Get Max Interact Required Get Number Of Interacted Sockets Get Channel Availability Status Get Channel Perceptability</p> <p>Get Ratio ::= Get Perceptability</p> <p>Get String ::= \$NULL</p> <p>Get Reference ::= Get Rt-Composite Address Get RPS Assignment</p> <p>Get List ::= Get Catalogued Attribute Get Stream Choice Get Style Get Event</p> <p>Get Any ::=</p> <p>Get Data Get OVD Get GTF Get OS Get POS Get OVS Get OAP Get OVS Position Get PVS Get PAP Get PVS Position Get GSF Get GVF Get Interaction Ability Get Interaction Status Get Event Data</p> <p>Evaluated Value ::=</p> <p>Get Boolean Get Numeric Get Ratio Get String Get Reference Get List Get Any Extensibility Provision</p> <p>Evaluated Boolean ::= Get Boolean Get Any</p> <p>Evaluated Numeric ::= Get Numeric Get Integer Get Any</p> <p>Evaluated Integer ::= Get Integer Get Any</p> <p>Evaluated Ratio ::= Get Ratio Get Any</p> <p>Evaluated String ::= Get String Get Any</p> <p>Evaluated Reference ::= Get Reference Get Any</p> <p>Evaluated List ::= Get List Get Any</p>
--

73.6.1 Get Event Data

- Retrieves the event data of the target.
- Rt-Component Channel Tg Param.

73.6.2 Get Event

- Retrieves the event of the target.
- Rt-Component Channel Tg Param.

73.6.3 Get Channel Perceptability

- Retrieves the channel perceptability of the target.
- Channel Target Param.

73.6.4 Get Channel Availability Status

- Retrieves the channel availability status.
- Channel Target Param.

73.6.5 Get Style

- Retrieves the current style associated with the target.
- Rt-Component Target Param.

73.6.6 Get Number of Interacted Sockets

- Retrieves the number of interacted sockets attribute value of the target.
- Sequence of (Rt-Composite Target Param, Interaction Type Param).

73.6.7 Get Interaction Status

- Retrieves the interaction status value of the target. This action evaluates “selected” or “not selected” for the selection, and “modified”, “modifying” or “not modified” for the modification.
- Sequence of (Rt-Component Target Param, Interaction Type Param).

73.6.8 Get Max Interact Required

- Retrieves the max interact required attribute value of the target.
- Sequence of (Rt-Component Target Param, Interaction Type Param).

73.6.9 Get Min Interact Required

- Retrieves the min interact required attribute value of the target.
- Sequence of (Rt-Component Target Param, Interaction Type Param).

73.6.10 Get Interaction Ability

- Retrieves the interaction ability of the target.
- Sequence of (Rt-Component Target Param, Interaction Type Param).

73.6.11 Get Stream Chosen State

- Retrieves the stream state.
- Sequence of (Rt-Mux Target Param, Stream Identification Param).

73.6.12 Get Stream Choice

- Retrieves a list of stream IDs chosen for the target.
- Rt-Mux Target Param.

73.6.13 Get GVF

- Retrieves the GVF.
- Choice between (Rt-Composite Target Param, Channel Target Param).

73.6.14 Get PCV

- Retrieves the PCV.
- Rt-Content Target Param.

73.6.15 Get CV

- Retrieves the CV.
- Rt-Content Target Param.

73.6.16 Get OV

- Retrieves the OV.
- Rt-Content Target Param.

73.6.17 Get User Spatial Control

- Retrieves the user spatial control attribute value.
- Sequence of (Rt-Component Target Param, Spatial Control Param).

73.6.18 Get GSF

- Retrieves the GSF.
- Sequence of (Rt-Component Target Param, Expected Axis Result Param).

73.6.19 Get PVS Position

- Retrieves the PVS position.
- Sequence of (Rt-Component Target Param, Point Type Param, Expected Axis Result Param).

73.6.20 Get PAP

- Retrieves the PAP.
- Sequence of (Rt-Component Target Param, Point Type Param, Expected Axis Result Param).

73.6.21 Get PVS

- Retrieves the PVS.
- Sequence of (Rt-Component Target Param, Point Type Param, Expected Axis Result Param).

73.6.22 Get OVS Position

- Retrieves the OVS position.
- Sequence of (Rt-Component Target Param, Point Type Param, Expected Axis Result Param).

73.6.23 Get OAP

- Retrieves the OAP.
- Sequence of (Rt-Component Target Param, Point Type Param, Expected Axis Result Param).

73.6.24 Get OVS

- Retrieves the OVS.
- Sequence of (Rt-Component Target Param, Point Type Param, Expected Axis Result Param).

73.6.25 Get OVS Proj Strategy

- Retrieves the OVS Proj Strategy attribute.
- Rt-Component Target Param.

73.6.26 Get Resizing Strategy

- Retrieves the resizing strategy attribute.
- Rt-Composite Target Param.

73.6.27 Get Aspect Ratio

- Retrieves the aspect ratio attribute.
- Rt-Component Target Param.

73.6.28 Get POS

- Retrieves the POS.
- Sequence of (Rt-Component Target Param, Expected Axis Result Param).

73.6.29 Get OS

- Retrieves the OS.
- Sequence of (Rt-Component Target Param, Expected Axis Result Param).

73.6.30 Get Timestone Status

- Retrieves the timestone status value.
- Rt-Component Target Param.

73.6.31 Get GTF

- Retrieves the GTF of the target.
- Rt-Component Target Param.

73.6.32 Get PVD Position

- Retrieves the PVD position attached to its PRPS.
- Rt-Component Target Param.

73.6.33 Get Temporal Termination

- Offers to retrieve the temporal termination of the target.
- Rt-Component Target Param.

73.6.34 Get CTP

- Offers to retrieve the CTP of the target.
- Rt-Component Target Param.

73.6.35 Get PVD

- Retrieves the PVD, initial temporal position or terminal temporal position of the target.
- Sequence of (Rt-Component Target Param, Expected PVD Result Param).

73.6.36 Get OVD

- Retrieve the OVD, initial temporal position or terminal temporal position of the target.
- Sequence of (Rt-Component Target Param, Expected OVD Result Param).

73.6.37 Get POD

- Retrieves the POD of the target.
- Rt-Component Target Param.

73.6.38 Get OD

- Retrieves the OD of the target.
- Rt-Component Target Param.

73.6.39 Get Presentation Priority

- Retrieves the presentation priority.
- Rt-Component Target Param.

73.6.40 Get Perceptability

- Retrieves the perceptability.
- Rt-Component Target Param.

73.6.41 Get RPS Assignment

- Retrieves the RPS assignment to an rt-component and a root rt-component.
- Rt-Component Target Param.

73.6.42 Get Rt-Composite Address

- Offers means to rise to the root, to descend to leave and to explore the intermediate levels (nodes) in a composition tree.
- Sequence of (Rt-Composite Target Param, Navigation Command Param).

73.6.43 Get Termination Status

- Retrieves the termination status of an rt-script. The result is “terminated” or “not terminated”.
- Rt-Script Target Param.

73.6.44 Get Running Status

- Retrieves the running status of an rt-object or a socket. The result is “running” or “not running”.
- Rt-Target Param.

73.6.45 Get Rt-Availability Status

- Provides means to retrieve the availability status of an rt-object. The result is “available” or “not available”.
- Rt-Target Param.

73.6.46 Get Data

- This action retrieves a generic value or an element of a generic list stored in a content object.
- Sequence of (Content Target Param, Generic List Elt ID Param?).

73.6.47 Get Activation Status

- This action retrieves the activation status value of a link object.
- Link Target Param.

73.6.48 Get Preparation Status

- Retrieves the preparation status of an MHEG object. The result is “ready” or “not ready”.
- MH-Target Param.

73.6.49 Get Catalogued Attribute

- Retrieves the value of an extended catalogued attribute. The result is a generic value which is described in the corresponding catalogue.
- Sequence of (Target Param, Cat Ext Attribute Param).

73.6.50 Get Boolean

- A get boolean is one of the result of a get action that returns a boolean.
- NULL.

73.6.51 Get Numeric

- A get numeric is one of the result of a get action that returns a numeric.
- Get Integer.

73.6.52 Get Integer

- A get integer is one of the result of a get action that returns an integer.
- Choice between (Get Preparation Status, Get Activation Status, Get Rt-Availability Status, Get Running Status, Get Termination Status, Get Presentation Priority, Get OD, Get POD, Get PVD, Get CTP, Get Temporal Termination, Get PVD Position, Get Timestone Status, Get Aspect Ratio, Get Resizing Strategy, Get OVS Proj Strategy, Get User Spatial Control, Get CV, Get PCV, Get Stream Chosen State, Get Min Interact Required, Get Max Interact Required, Get Number of Interacted Sockets, Get Channel Availability Status, Get Channel Perceptability).

73.6.53 Get Ratio

- A get ratio is one of the result of a get action that returns a ratio.
- Get Perceptability.

73.6.54 Get String

- A get string is one of the result of a get action that returns a string.
- NULL.

73.6.55 Get Reference

- A get reference is one of the result of a get action that returns a reference.
- Choice between (Get Rt-Composite Address, Get RPS Assignment).

73.6.56 Get List

- A get list is one of the result of a get action that returns a list.
- Choice between (Get Catalogued Attribute, Get Stream Choice, Get Style, Get Event).

73.6.57 Get Any

- A get any is one of the result of a get action that returns any kind of value.
- Choice between (Get Data, Get OVD, Get GTF, Get OS, Get POS, Get OVS, Get OAP, Get OVS Position, Get PVS, Get PAP, Get PVS Position, Get GSF, Get GVF, Get Interaction Ability, Get Interaction Status, Get Event Data).

73.6.58 Evaluated Value

- An evaluated value is one of the result of a get action.
- Choice between (Get Boolean, Get Numeric, Get Ratio, Get String, Get Reference, Get List, Get Any, Extensibility Provision).

73.6.59 Evaluated Boolean

- An evaluated boolean is one of the result of a get action resulting to a boolean or any.
- Choice between (Get Boolean, Get Any).

73.6.60 Evaluated Numeric

- An evaluated numeric is one of the result of a get action resulting to a numeric, an integer or any.
- Choice between (Get Numeric, Get Integer, Get Any).

73.6.61 Evaluated Integer

- An evaluated integer is one of the result of a get action resulting to an integer or any.
- Choice between (Get Integer, Get Any).

73.6.62 Evaluated Ratio

- An evaluated ratio is one of the result of a get action resulting to a ratio or any.
- Choice between (Get Ratio, Get Any).

73.6.63 Evaluated String

- An evaluated string is one of the result of a get action resulting to a string or any.
- Choice between (Get String, Get Any).

73.6.64 Evaluated Reference

- An evaluated reference is one of the result of a get action resulting to a reference or any.
- Choice between (Get Reference, Get Any).

73.6.65 Evaluated List

- An evaluated list is one of the result of a get action resulting to a list or any.
- Choice between (Get List, Get Any).

73.7 Hooks

Table 47/T.171 – Overview of Hooks

Content Hook ::= **Catalogued Content Encoding?, Content Encoding Description?**

Content Encoding Description ::= *\$OCTET STRING*

Script Hook ::= **Catalogued Script Encoding?, Script Encoding Description?**

Script Encoding Description ::= *\$OCTET STRING*

73.7.1 Content Hook

- Encoding and decoding information enabling the use of the encoded media or non-media data. This information contains an identification field for the media encoding standard and a field for parameters associated with the encoding.

The semantics of these fields are not defined by MHEG but will be under control of MHEG Format Identifier Registration Authority.

- Sequence of (Catalogued Content Encoding?, Content Encoding Description?).

73.7.2 Content Encoding Description

- Additional information associated with reference to specific encoding/decoding techniques. The semantics of this description can be provided in the content encoding catalogue.
- OCTET STRING.

73.7.3 Script Hook

- Encoding and decoding information enabling the use of the encoded scripting language. This information contains an identification field for the script encoding standard and a field for parameters associated with the encoding MHEG Format Identifier Registration Authority.
- Sequence of (Catalogued Script Encoding?, Script Encoding Description?).

73.7.4 Script Encoding Description

- Additional information associated with reference to specific encoding/decoding techniques. The semantics of this description can be provided in the script encoding catalogue.
- OCTET STRING.

73.8 Extensibility

Table 48/T.171 – Overview of Extensibility

Catalogued Content Encoding ::= **Registered Content Encoding | Proprietary Content Encoding**

Registered Content Encoding ::= *#Imported from ITU-T Rec. T.171 cat*

Proprietary Content Encoding ::= **Prop Cat Entry ID**

Catalogued Script Encoding ::= **Registered Script Encoding | Proprietary Script Encoding**

Registered Script Encoding ::= *#Imported from ITU-T Rec. T.171 cat*

Proprietary Script Encoding ::= **Prop Cat Entry ID**

Cat Content Classification ::= **Registered Content Classification | Prop Content Classification**

Registered Content Classification ::= *#Imported from ITU-T Rec. T.171 cat*

Table 48/T.171 – Overview of Extensibility (concluded)

Prop Content Classification ::= Prop Cat Entry ID
Catalogued Script Classification ::= Registered Script Classification Prop Script Classification
Registered Script Classification ::= <i>#Imported from ITU-T Rec. T.171 cat</i>
Prop Script Classification ::= Prop Cat Entry ID
Catalogued Media Type ::= Registered Media Type Proprietary Media Type
Registered Media Type ::= <i>#Imported from ITU-T Rec. T.171 cat</i>
Proprietary Media Type ::= Prop Cat Entry ID
Catalogued Style ::= Registered Style Proprietary Style
Registered Style ::= <i>#Imported from ITU-T Rec. T.171 cat</i>
Proprietary Style ::= Prop Cat Entry ID
Catalogued Event ::= Registered Event Proprietary Event
Registered Event ::= <i>#Imported from ITU-T Rec. T.171 cat</i>
Proprietary Event ::= Prop Cat Entry ID
Catalogued Extended EA ::= Registered Extended Elementary Action Proprietary Extended Elementary Action
Registered Extended Elementary Action ::= <i>#Imported from ITU-T Rec. T.171 cat</i>
Proprietary Extended Elementary Action ::= Prop Cat Entry ID
Cat Ext Attribute ::= Registered Extended Attribute Proprietary Extended Attribute
Registered Extended Attribute ::= <i>#Imported from ITU-T Rec. T.171 cat</i>
Proprietary Extended Attribute ::= Prop Cat Entry ID
Prop Cat Entry ID ::= Tail \$OCTET STRING
Extensibility Provision ::= Pointpointpoint
Pointpointpoint ::= <i>#INTEGER ::= 9999</i>

73.8.1 Catalogued Content Encoding

- Identification of the encoding of the content data. Two types of catalogues are offered: a registered catalogue and a proprietary catalogue.
- Choice between (Registered Content Encoding, Proprietary Content Encoding).

73.8.2 Registered Content Encoding

- Identification of the encoding and decoding techniques within a registered content catalogue.
The registered content catalogue is under control of MHEG Format Identifier Registration Authority as specified by ISO/IEC 13522-4.
- Imported from ISO/IEC 13522-4 cat module.

73.8.3 Proprietary Content Encoding

- Identification of the encoding and decoding techniques within a proprietary content catalogue.
The using application is responsible for the resolution of references to proprietary content catalogue.
- Prop Cat Entry ID.

73.8.4 Catalogued Script Encoding

- Identification of the encoding of the scripting language. Two types of catalogues are offered: a registered catalogue and a proprietary catalogue.
- Choice between (Registered Script Encoding, Proprietary Script Encoding).

73.8.5 Registered Script Encoding

- Identification of the scripting language within a registered script catalogue.
The registered script catalogue is under control of the MHEG Format Identifier Registration Authority as specified by ISO/IEC 13522-4.
- Imported from ISO/IEC 13522-4 cat module.

73.8.6 Proprietary Script Encoding

- Identification of the scripting language within a proprietary script catalogue.
- Prop Cat Entry ID.

73.8.7 Cat Content Classification

- Identification of data type. For media data, the content classification is used to indicate the perception media, e.g. text, graphics, audio. This information may be used by a negotiation process, a database, or by an MHEG engine to choose a decoder. It is provided as an optional assistance in determining the type of the data. In case of incompatibility between the hook and the classification information, the hook is to be preferred.
- Choice between (Registered Content Classification, Prop Content Classification).

73.8.8 Registered Content Classification

- Identification of the type of classification within the registered content catalogue.
The Mheg Content classification is under control of the MHEG Format Identifier Registration Authority as specified by ISO/IEC 13522-4.
- Imported from ISO/IEC 13522-4 cat module.

73.8.9 Prop Content Classification

- Identification of the data type within the proprietary content catalogue.
- Prop Cat Entry ID.

73.8.10 Catalogued Script Classification

- Identification of script type.
- Choice between (Registered Script Classification, Prop Script Classification).

73.8.11 Registered Script Classification

- Identification of the type of classification within the registered script catalogue.
The registered script classification is under control of the MHEG Format Identifier Registration Authority as specified by ISO/IEC 13522-4.
- Imported from ISO/IEC 13522-4 cat module.

73.8.12 Prop Script Classification

- Identification of the script type within the proprietary script classification catalogue.
- Prop Cat Entry ID.

73.8.13 Catalogued Media Type

- Identification of media type.
- Choice between (Registered Media Type, Proprietary Media Type).

73.8.14 Registered Media Type

- Identification of the type of classification within the registered media type catalogue.
The registered media type classification is under control of the MHEG Format Identifier Registration Authority as specified by ISO/IEC 13522-4.
- Imported from ISO/IEC 13522-4 cat module.

73.8.15 Proprietary Media Type

- Identification of the media type within the proprietary media type classification catalogue.
- Prop Cat Entry ID.

73.8.16 Catalogued Style

- Identification of style.
- Choice between (Registered Style, Proprietary Style).

73.8.17 Registered Style

- Identification of the type of classification within the registered style catalogue.
The registered style classification is under control of the MHEG Format Identifier Registration Authority as specified by ISO/IEC 13522-4.
- Imported from ISO/IEC 13522-4 cat module.

73.8.18 Proprietary Style

- Identification of the style type within the proprietary style classification catalogue.
- Prop Cat Entry ID.

73.8.19 Catalogued Event

- Identification of event.
- Choice between (Registered Event, Proprietary Event).

73.8.20 Registered Event

- Identification of the type of classification within the registered event catalogue.
The registered event classification is under control of the MHEG Format Identifier Registration Authority as specified by ISO/IEC 13522-4.
- Imported from ISO/IEC 13522-4 cat module.

73.8.21 Proprietary Event

- Identification of the event type within the proprietary event classification catalogue.
- Prop Cat Entry ID.

73.8.22 Catalogued Extended EA

- Identification of catalogued extended elementary action. Either registered one or proprietary one may be used.
- Choice between (Registered Extended Elementary Action, Proprietary Extended Elementary Action).

73.8.23 Registered Extended Elementary Action

- Identification of the type of classification within the registered catalogued elementary action catalogue.
The registered catalogued elementary action classification is under control of the MHEG Format Identifier Registration Authority as specified by ISO/IEC 13522-4.
- Imported from ISO/IEC 13522-4 cat module.

73.8.24 Proprietary Extended Elementary Action

- Identification of the catalogued elementary action type within the proprietary catalogued elementary action classification catalogue.
- Prop Cat Entry ID.

73.8.25 Cat Ext Attribute

- Identification of extended attribute.
- Choice between (Registered Extended Attribute, Proprietary Extended Attribute).

73.8.26 Registered Extended Attribute

- Identification of the type of classification within the registered extended attribute catalogue.
The registered extended attribute classification is under control of the MHEG Format Identifier Registration Authority as specified by ISO/IEC 13522-4.
- Imported from ISO/IEC 13522-4 cat module.

73.8.27 Proprietary Extended Attribute

- Identification of the extended attribute type within the proprietary extended attribute classification catalogue.
- Prop Cat Entry ID.

73.8.28 Prop Cat Entry ID

- Identification of an entry in a catalogue hierarchy.
- Choice between (Tail, OCTET STRING).

73.8.29 Extensibility Provision

- Means to provide an extensibility of the syntax for a using application.
- Point point point.

73.9 Presentation space useful definitions

Table 49/T.171 – Overview of Presentation space useful definitions

```
OPS Initialisation ::= OD?, OS?
min-AVR ::= #INTEGER ::= 0
max-AVR ::= #INTEGER ::= 255
GF Param ::= GF | GF Macro
GF Macro ::= Macro Def ID, GF?
GF ::= Generic Ratio | default-GF
default-GF ::= #INTEGER ::= 1
X GF ::= GF
Y GF ::= GF
Z GF ::= GF
Length ::= Integer
Point ::= Integer | #terminal
Temporal Position ::= Point
Size ::= X Spatial Length?, Y Spatial Length?, Z Spatial Length?
X Spatial Length ::= Spatial Length
Y Spatial Length ::= Spatial Length
Z Spatial Length ::= Spatial Length
Spatial Length ::= Length | default-spatial-length
default-spatial-length ::= #INTEGER ::= 65536
Size Spec Param ::= Size Spec | Size Spec Macro
Size Spec Macro ::= Macro Def ID, Size Spec?
Size Spec ::= Lengths Spec | Get List | Get Any
```

Table 49/T.171 – Overview of Presentation space useful definitions (concluded)

Lengths Spec ::= **X Length Spec?, Y Length Spec?, Z Length Spec?**
X Length Spec ::= **Length Spec**
Y Length Spec ::= **Length Spec**
Z Length Spec ::= **Length Spec**
Length Spec Param ::= **Length Spec | Length Spec Macro**
Length Spec Macro ::= **Macro Def ID, Length Spec?**
Length Spec ::= **Length | Relative Length | Get Integer | Get Ratio | Get Any**
Relative Length ::= **Ratio**
Spatial Position ::= **X Point?, Y Point?, Z Point?**
X Point ::= **Point**
Y Point ::= **Point**
Z Point ::= **Point**
Spatial Position Spec Param ::= **Spatial Position Spec | Spatial Position Spec Macro**
Spatial Position Spec Macro ::= **Macro Def ID, Spatial Position Spec?**
Spatial Position Spec ::= **Points Spec | Get List | Get Any**
Points Spec ::= **X Point Spec?, Y Point Spec?, Z Point Spec?**
X Point Spec ::= **Point Spec**
Y Point Spec ::= **Point Spec**
Z Point Spec ::= **Point Spec**
Point Spec Param ::= **Point Spec | Point Spec Macro**
Point Spec Macro ::= **Macro Def ID, Point Spec?**
Point Spec ::= **Point | Relative Point | Get Integer | Get Ratio | Get Any**
Current Point Spec Param ::= **Current Point Spec | Current Point Spec Macro**
Current Point Spec Macro ::= **Macro Def ID, Current Point Spec?**
Current Point Spec ::= **Point | Relative Point | Original Point Factor | Current Point Factor | Get Integer | Get Ratio | Get Any**
Relative Point ::= **Ratio**
Original Point Factor ::= **Ratio**
Current Point Factor ::= **Ratio**
Duration Spec ::= **Length Spec**
Initial Point Spec Param ::= **Initial Point Spec | Initial Point Spec Macro**
Initial Point Spec Macro ::= **Macro Def ID, Initial Point Spec?**
Terminal Point Spec Param ::= **Terminal Point Spec | Terminal Point Spec Macro**
Terminal Point Spec Macro ::= **Macro Def ID, Terminal Point Spec?**
Initial Point Spec ::= **Point Spec**
Terminal Point Spec ::= **Point Spec**
Transition Duration Param ::= **Generic Integer Param**

73.9.1 OPS Initialisation

- Provides the information to build an OPS of an rt-component or a channel. It provides an OD and an OS used to determine the OPS temporal and spatial axes lengths. The AVR is set to the range value provided by this Recommendation.
- Sequence of (OD?, OS?).

73.9.2 min-AVR

- Minimum AVR value provided by this Recommendation.
- INTEGER ::= 0.

73.9.3 max-AVR

- Maximum AVR value provided by this Recommendation.
- INTEGER ::= 255.

73.9.4 GF Param

- Parameter of GF.
- Choice between (GF, GF Macro).

73.9.5 GF Macro

- Allows the specification of a macro parameter instead of a specified GF value.
- Sequence of (Macro Def ID, GF?).

73.9.6 GF

- Specifies the GF of an rt-component or a channel.
- Choice between (Generic Ratio, default-GF).

73.9.7 default-GF

- Default GF value provided by this Recommendation.
- INTEGER ::= 1.

73.9.8 X GF

- Express the GF on the X axis.
- GF.

73.9.9 Y GF

- Express the GF on the Y axis.
- GF.

73.9.10 Z GF

- Express the GF on the Z axis.
- GF.

73.9.11 Length

- Specifies a length of an interval [Start Point, EndList Point]. It is equal to EndList-Point – Start-Point + 1.
- Integer.

73.9.12 Point

- Specifies a position within a duration or a size. It is expressed either in GTU or in GSU. The value 0 corresponds to the initial position of the corresponding duration or size. A reserved value “terminal” corresponds to the terminal position of the corresponding duration or size.
- Choice between (Integer, terminal).

73.9.13 Temporal Position

- Specifies a temporal position. It is expressed in GTU. The position is always specified within a duration.
- Point.

73.9.14 Size

- Specifies a size by providing the length on each axis: X, Y, Z.
- Sequence of (X Spatial Length?, Y Spatial Length?, Z Spatial Length?).

73.9.15 X Spatial Length

- Provides the length on X axis.
- Spatial Length.

73.9.16 Y Spatial Length

- Provides the length on Y axis.
- Spatial Length.

73.9.17 Z Spatial Length

- Provides the length on Z axis.
- Spatial Length.

73.9.18 Spatial Length

- Specifies the length of a spatial axis. It is expressed in GSU.
- Choice between (Length, default-spatial-length).

73.9.19 default-spatial-length

- Default spatial axis length value provided by this Recommendation.
- INTEGER ::= 65536.

73.9.20 Size Spec Param

- Parameter of Size Spec.
- Choice between (Size Spec, Size Spec Macro).

73.9.21 Size Spec Macro

- Allows the specification of a macro parameter instead of a specified Size Spec value.
- Sequence of (Macro Def ID, Size Spec?).

73.9.22 Size Spec

- This elementary action parameter specifies different means to define a size. It is expressed as a length on each axis: x, y, z.
- Choice between (Lengths Spec, Get List, Get Any).

73.9.23 Lengths Spec

- Expressed as a length on each axis: x, y, z.
- Sequence of (X Length Spec?, Y Length Spec?, Z Length Spec?).

73.9.24 X Length Spec

- Provides the length on X axis.
- Length Spec.

73.9.25 Y Length Spec

- Provides the length on Y axis.
- Length Spec.

73.9.26 Z Length Spec

- Provides the length on Z axis.
- Length Spec.

73.9.27 Length Spec Param

- Parameter of Length Spec.
- Choice between (Length Spec, Length Spec Macro).

73.9.28 Length Spec Macro

- Allows the specification of a macro parameter instead of a specified Length Spec value.
- Sequence of (Macro Def ID, Length Spec?).

73.9.29 Length Spec

- Each length may be specified as an absolute length, a relative length or as a result of a get integer, get ratio or get any actions.
- Choice between (Length, Relative Length, Get Integer, Get Ratio, Get Any).

73.9.30 Relative Length

- Defines a length as a ratio of an interval.
- Ratio.

73.9.31 Spatial Position

- Specifies a spatial position by providing a point on each axis: X, Y, Z.
- Sequence of (X Point?, Y Point?, Z Point?).

73.9.32 X Point

- Provides the position on X axis.
- Point.

73.9.33 Y Point

- Provides the position on Y axis.
- Point.

73.9.34 Z Point

- Provides the position on Z axis.
- Point.

73.9.35 Spatial Position Spec Param

- Parameter of Spatial Position Spec.
- Choice between (Spatial Position Spec, Spatial Position Spec Macro).

73.9.36 Spatial Position Spec Macro

- Allows the specification of a macro parameter instead of a specified Spatial Position Spec value.
- Sequence of (Macro Def ID, Spatial Position Spec?).

73.9.37 Spatial Position Spec

- This elementary action parameter specifies different means to define a spatial position. It is expressed as a length on each axis: x, y, z.
- Choice between (Points Spec, Get List, Get Any).

73.9.38 Points Spec

- Define a position. It is expressed as a length on each axis: x, y, z.
- Sequence of (X Point Spec?, Y Point Spec?, Z Point Spec?).

73.9.39 X Point Spec

- Length on X axis.
- Point Spec.

73.9.40 Y Point Spec

- Length on Y axis.
- Point Spec.

73.9.41 Z Point Spec

- Length on Z axis.
- Point Spec.

73.9.42 Point Spec Param

- Parameter of Point Spec.
- Choice between (Point Spec, Point Spec Macro).

73.9.43 Point Spec Macro

- Allows the specification of a macro parameter instead of a specified Point Spec value.
- Sequence of (Macro Def ID, Point Spec?).

73.9.44 Point Spec

- Each point may be specified as an absolute point, a relative point within a length or as a result of a get integer, get ratio or get any actions. An integer is interpreted as an absolute value. A ratio is interpreted as a relative value.
- Choice between (Point, Relative Point, Get Integer, Get Ratio, Get Any).

73.9.45 Current Point Spec Param

- Parameter of Current Point Spec.
- Choice between (Current Point Spec, Current Point Spec Macro).

73.9.46 Current Point Spec Macro

- Allows the specification of a macro parameter instead of a specified Current Point Spec value.
- Sequence of (Macro Def ID, Current Point Spec?).

73.9.47 Current Point Spec

- Each current point may be specified as an absolute point, a relative point within a length, a factor of the original point value, a factor of the current point value or as a result of a get integer, get ratio or get any actions. An integer is interpreted as an absolute value. A ratio is interpreted as a relative value.
- Choice between (Point, Relative Point, Original Point Factor, Current Point Factor, Get Integer, Get Ratio, Get Any).

73.9.48 Relative Point

- Specified as a ratio to be mapped on an interval.
- Ratio.

73.9.49 Original Point Factor

- Scaling Factor in relation to the original value of this point. It is defined as a positive ratio which is used as a multiplier of the original value of this point.
- Ratio.

73.9.50 Current Point Factor

- Scaling Factor in relation to the current value of this point. It is defined as a positive ratio which is used as a multiplier of the current value of this point.
- Ratio.

73.9.51 Duration Spec

- This elementary action parameter specifies different means to define a duration.
- Length Spec.

73.9.52 Initial Point Spec Param

- Parameter of Initial Point Spec.
- Choice between (Initial Point Spec, Initial Point Spec Macro).

73.9.53 Initial Point Spec Macro

- Allows the specification of a macro parameter instead of a specified Initial Point Spec value.
- Sequence of (Macro Def ID, Initial Point Spec?).

73.9.54 Terminal Point Spec Param

- Parameter of Terminal Point Spec.
- Choice between (Terminal Point Spec, Terminal Point Spec Macro).

73.9.55 Terminal Point Spec Macro

- Allows the specification of a macro parameter instead of a specified Terminal Point Spec value.
- Sequence of (Macro Def ID, Terminal Point Spec?).

73.9.56 Initial Point Spec

- Initial position on the temporal axis.
- Point Spec.

73.9.57 Terminal Point Spec

- Terminal position on the temporal axis.
- Point Spec.

73.9.58 Transition Duration Param

- An optional parameter which can be provided for certain elementary actions within the presentation behaviour. It is expressed in GTU. When a transition duration is specified, the corresponding elementary action is to be processed during the specified duration, e.g. for set audible volume, the audible volume value changes gradually from its previous value to the specified value within the specified transition duration.
- Generic Integer Param.

73.10 Constants useful definitions

Table 50/T.171 – Overview of Constants useful definitions

Event Data ::= Value
Event ::= List
Channel Perceptability ::= <i>#on / #off</i>
Channel Availability Status ::= <i>#available / #not-available</i>
Style ::= List
Max Interact Required ::= Integer
Min Interact Required ::= Integer
Modifiability ::= <i>#modifiable / #not-modifiable</i>
Selectability ::= <i>#selectable / #not-selectable</i>
Modification Status ::= <i>#modified / #not-modified / #modifying</i>
Selection Status ::= <i>#selected / #not-selected</i>
Interaction Status ::= Selection Status Modification Status
Interaction Type ::= <i>#selection / #modification</i>
Interaction Type Macro ::= Macro Def ID, Interaction Type?
Interaction Type Param ::= Interaction Type Interaction Type Macro
Stream Identification ::= Stream ID #all-streams
Stream Identification Macro ::= Macro Def ID, Stream Identification?
Stream Identification Param ::= Stream Identification Stream Identification Macro
Stream Chosen State ::= <i>#chosen / #not-chosen / #partially-chosen</i>
Stream Choice ::= Generic List
GVF ::= GF
PCV ::= Integer min-AVR max-AVR
CV ::= Integer min-AVR max-AVR
OV ::= Integer min-AVR max-AVR
Point Type Param ::= <i>#absolute / #relative</i>
Spatial Control Macro ::= Macro Def ID, Spatial Control?
Spatial Control Param ::= Spatial Control Spatial Control Macro
Expected Axis Result Param ::= <i>#x / #y / #z / #xyz</i>
User Spatial Control ::= <i>#allowed / #not-allowed</i>
Spatial Control ::= <i>#moving / #resizing / #scaling / #scrolling</i>
GSF ::= X GF?, Y GF?, Z GF?
PVS Position ::= Spatial Position Spec
PAP ::= Spatial Position Spec
OVS Proj Strategy ::= <i>#fixed / #calculated</i>
OVS Position ::= Spatial Position Spec
OAP ::= Spatial Position Spec
OVS ::= Size Spec

Table 50/T.171 – Overview of Constants useful definitions (concluded)

Resizing Strategy ::= <i>#fixed / #minimum / #grows-only</i>
Aspect Ratio ::= <i>#preserved / #not-preserved</i>
OS ::= Size
Expected PVD Result ::= <i>#initial-temporal-position / #terminal-temporal-position / #duration</i>
Expected PVD Result Macro ::= Macro Def ID, Expected PVD Result?
Expected PVD Result Param ::= Expected PVD Result Expected PVD Result Macro
Expected OVD Result ::= <i>#initial-temporal-position / #terminal-temporal-position / #duration</i>
Expected OVD Result Macro ::= Macro Def ID, Expected OVD Result?
Expected OVD Result Param ::= Expected OVD Result Expected OVD Result Macro
Timestone ID ::= Generic Integer
Temporal Termination ::= <i>#freeze / #stop</i>
OD ::= Integer #infinite
Presentation Priority ::= Generic Integer #up-priority / #down-priority
Perceptability ::= Generic Ratio
RPS Assignment ::= Channel Reference Evaluated Reference #prps
Ancestor ::= Integer #root
Sibling ::= Integer
EmptyChild ::= Integer #last
Child ::= Generic Integer #last / #random
Navigation Command ::= Child EmptyChild Sibling Ancestor
Navigation Command Macro ::= Macro Def ID, Navigation Command?
Navigation Command Param ::= Navigation Command Navigation Command Macro
Termination Status ::= <i>#terminated / #not-terminated</i>
Running Status ::= <i>#running / #not-running</i>
Rt-Availability Status ::= <i>#available / #not-available</i>
Activation Status ::= <i>#active / #inactive</i>
Preparation Status ::= <i>#ready / #not-ready</i>
Cat Ext Attribute Macro ::= Macro Def ID, Cat Ext Attribute?
Cat Ext Attribute Param ::= Cat Ext Attribute Cat Ext Attribute Macro
Update Command ::= <i>#add / #remove / #replace</i>

73.10.1 Event Data

- Specifies an associated data with an event.
- Value.

73.10.2 Event

- Specifies an event.
- List.

73.10.3 Channel Perceptability

- Specifies the channel perceptability. Choice between “on” and “off”.
- Choice between (on, off).

73.10.4 Channel Availability Status

- Indicates whether a channel is available.
- Choice between (available, not-available).

73.10.5 Style

- A catalogued entry, either registered or proprietary that gives a logical style to an rt-component.
- List.

73.10.6 Max Interact Required

- An integer greater than or equal to 0 to indicate the maximum interaction required for an rt-component.
- Integer.

73.10.7 Min Interact Required

- A integer greater than or equal to 0 to indicate the minimum interaction required for an rt-component.
- Integer.

73.10.8 Modifiability

- Indicates whether an rt-component is modifiable.
- Choice between (modifiable, not-modifiable).

73.10.9 Selectability

- Specifies a value of the maximum interaction required attribute for the selection of an rt-component.
- Choice between (selectable, not-selectable).

73.10.10 Modification Status

- Indicates the result of the user modification.
- Choice between (modified, not-modified, modifying).

73.10.11 Selection Status

- Indicates the results of the user selection.
- Choice between (selected, not-selected).

73.10.12 Interaction Status

- Indicates if an rt-component may be modified or selected.
- Choice between (Selection Status, Modification Status).

73.10.13 Interaction Type

- Specifies the interaction type.
- Choice between (selection, modification).

73.10.14 Interaction Type Macro

- Allows the specification of a macro parameter instead of a specified Interaction Type value.
- Sequence of (Macro Def ID, Interaction Type?).

73.10.15 Interaction Type Param

- Parameter of Interaction Type.
- Choice between (Interaction Type, Interaction Type Macro).

73.10.16 Stream Identification

- Choice between a stream identifier or “all streams”.
- Choice between (Stream ID, all-streams).

73.10.17 Stream Identification Macro

- Allows the specification of a macro parameter instead of a specified Stream Identification value.
- Sequence of (Macro Def ID, Stream Identification?).

73.10.18 Stream Identification Param

- Parameter of Stream Identification.
- Choice between (Stream Identification, Stream Identification Macro).

73.10.19 Stream Chosen State

- Indicates stream chosen state for each stream within an rt-mux, and for each rt-mux.
- Choice between (chosen, not-chosen, partially-chosen).

73.10.20 Stream Choice

- A list of chosen stream IDs in an rt-mux.
- Generic List.

73.10.21 GVF

- Specifies the GVF to be applied during the mapping of the OPS into its RPS.
- GF.

73.10.22 PCV

- Specifies the PCV of a content object that is a projection of the CV to its parent RPS.
- Choice between (Integer, min-AVR, max-AVR).

73.10.23 CV

- Specifies the CV of a content object. It is initialised by the OV and may be changed by using the Set CV action.
- Choice between (Integer, min-AVR, max-AVR).

73.10.24 OV

- Specifies the OV of a content object. The OV shall be defined within the AVR. It is used to initialise the CV of each rt-content created from this content.
- Choice between (Integer, min-AVR, max-AVR).

73.10.25 Point Type Param

- Choice between “absolute” and “relative”.
- Choice between (absolute, relative).

73.10.26 Spatial Control Macro

- Allows the specification of a macro parameter instead of a specified Spatial Control value.
- Sequence of (Macro Def ID, Spatial Control?).

73.10.27 Spatial Control Param

- Choice among “moving”, “resizing”, “scaling” and “scrolling”.
- Choice between (Spatial Control, Spatial Control Macro).

73.10.28 Expected Axis Result Param

- Choice among “x”, “y”, “z” or “xyz”.
- Choice between (x, y, z, xyz).

73.10.29 User Spatial Control

- Indicates functions of user spatial controls are allowed or not.
- Choice between (allowed, not-allowed).

73.10.30 Spatial Control

- Indicates functions of user spatial controls.
- Choice between (moving, resizing, scaling, scrolling).

73.10.31 GSF

- A set of three GF used for each axis.
- Sequence of (X GF?, Y GF?, Z GF?).

73.10.32 PVS Position

- Specifies the Position of the PVS relatively to the origin of the OS or to another PVS.
- Spatial Position Spec.

73.10.33 PAP

- Specifies the PAP of the PVS. It is used to position the PVS within the RPS.
- Spatial Position Spec.

73.10.34 OVS Proj Strategy

- Specifies the projection strategy of the OVS to the PVS.
- Choice between (fixed, calculated).

73.10.35 OVS Position

- Specifies the Position of the OVS relatively to the origin of the OS.
- Spatial Position Spec.

73.10.36 OAP

- Specifies the OAP of the OVS. It is used to position the OVS within its OS.
- Spatial Position Spec.

73.10.37 OVS

- Specifies the OVS of an rt-component. It is specified as a subset of the OS.
- Size Spec.

73.10.38 Resizing Strategy

- Defines the resizing strategy of an rt-composite.
- Choice between (fixed, minimum, grows-only).

73.10.39 Aspect Ratio

- Defines if the aspect ratio of an rt-component is to be preserved.
- Choice between (preserved, not-preserved).

73.10.40 OS

- Specifies the OS of a component object or a channel. It is used to initialise the OS of an rt-component. It is also used to build the spatial axes of the OPS, of each rt-component created from this component, or the CPS.
- Size.

73.10.41 Expected PVD Result

- Choice between (initial-temporal-position, terminal-temporal-position, duration).

73.10.42 Expected PVD Result Macro

- Allows the specification of a macro parameter instead of a specified Expected PVD Result value.
- Sequence of (Macro Def ID, Expected PVD Result?).

73.10.43 Expected PVD Result Param

- Parameter of Expected PVD Result.
- Choice between (Expected PVD Result, Expected PVD Result Macro).

73.10.44 Expected OVD Result

- Choice between (initial-temporal-position, terminal-temporal-position, duration).

73.10.45 Expected OVD Result Macro

- Allows the specification of a macro parameter instead of a specified Expected OVD Result value.
- Sequence of (Macro Def ID, Expected OVD Result?).

73.10.46 Expected OVD Result Param

- Parameter of Expected OVD Result.
- Choice between (Expected OVD Result, Expected OVD Result Macro).

73.10.47 Timestone ID

- A generic integer. The value 0 is reserved.
- Generic Integer.

73.10.48 Temporal Termination

- Determines the presentation when a target reaches to the terminal temporal position.
- Choice between (freeze, stop).

73.10.49 OD

- Specifies the OD of a component object or a channel. It is used to initialise the OD of an rt-component. It is also used to build the temporal axis of the OPS, of each rt-component created from this component, or the CPS.
- Choice between (Integer, infinite).

73.10.50 Presentation Priority

- Determines the presentation stacking order of the rt-components assigned in a same RPS.
- Choice between (Generic Integer, up-priority, down-priority).

73.10.51 Perceptability

- A ratio which defines the perceptability of an rt-component. If Perceptability Status is set to zero rt-content is not perceived. If it is set to 100 the original perception of the rt-content is perceived.
- Generic Ratio.

73.10.52 RPS Assignment

- RPS is either a channel reference, an evaluated reference or PRPS.
- Choice between (Channel Reference, Evaluated Reference, prps).

73.10.53 Ancestor

- Navigation path to reach an ancestor.
- Choice between (Integer, root).

73.10.54 Sibling

- Navigation path to reach a relative sibling.
- Integer.

73.10.55 EmptyChild

- Navigation path to reach an empty socket.
- Choice between (Integer, last).

73.10.56 Child

- Navigation path to reach a child.
- Choice between (Generic Integer, last, random).

73.10.57 Navigation Command

- It gives a navigation path.
- Choice between (Child, EmptyChild, Sibling, Ancestor).

73.10.58 Navigation Command Macro

- Allows the specification of a macro parameter instead of a specified Navigation Command value.
- Sequence of (Macro Def ID, Navigation Command?).

73.10.59 Navigation Command Param

- Parameter of Navigation Command.
- Choice between (Navigation Command, Navigation Command Macro).

73.10.60 Termination Status

- Represents the process of an rt-script.
- Choice between (terminated, not-terminated).

73.10.61 Running Status

- Indicates if an rt-object is running or not.
- Choice between (running, not-running).

73.10.62 Rt-Availability Status

- Indicates if the rt-object has been created from model object or not.
- Choice between (available, not-available).

73.10.63 Activation Status

- Indicates if a link object has been activated or not.
- Choice between (active, inactive).

73.10.64 Preparation Status

- Indicates the availability of MHEG objects to the MHEG engine.
- Choice between (ready, not-ready).

73.10.65 Cat Ext Attribute Macro

- Allows the specification of a macro parameter instead of a specified Cat Ext Attribute value.
- Sequence of (Macro Def ID, Cat Ext Attribute?).

73.10.66 Cat Ext Attribute Param

- Parameter of Cat Ext Attribute.
- Choice between (Cat Ext Attribute, Cat Ext Attribute Macro).

73.10.67 Update Command

- When an attribute of an MHEG entity takes a list of values, this parameter is provided in order to update this list of values. It is used in Set Alias action, Set Timestones action, Set Stream Choice action and Attach Anchor action. It is always used as Attribute Specification = List of (List of attribute value, Update Command)*.

The following applies:

If the update command is equal to “replace”, the list of current attribute values of the target is removed and replaced by the list of attribute values provided as parameters of the elementary action.

If the update command is equal to “add”, the list of attribute values provided as parameters of the elementary action is added to the current attribute values list of the target. It is ignored to add an attribute which already exists in the current attribute values list.

If the update type is equal to “remove”, the list of attribute values currently provided as parameter of the elementary action is removed from the current attribute values list of the target. It is ignored to remove an attribute which does not exist in the current attribute values list.

Combination of “add” and “remove” update commands may be used within a given attribute specification list. In that case, each command should be evaluated according to the order of the list.

“replace” update command should not be used with “add” or/and “remove” within a given attribute specification list. If it is used with them, the elementary action shall be ignored.

- Choice between (add, remove, replace).

73.11 Comparison value constants

Table 51/T.171 – Overview of Comparison value constants

Comparison Value Constant ::=

Preparation Status | Activation Status | Rt-Availability Status | Running Status | Termination Status | #prps / #infinite / Temporal Termination | Aspect Ratio | Resizing Strategy | OVS Proj Strategy | User Spatial Control | Stream Chosen State | Selection Status | Modification Status | Channel Availability Status | Channel Perceptability

73.11.1 Comparison Value Constant

- Specification of the constants defined by this Recommendation which can be used as a comparison value within a link condition of a link object.
- Choice between (Preparation Status, Activation Status, Rt-Availability Status, Running Status, Termination Status, prps, infinite, Temporal Termination, Aspect Ratio, Resizing Strategy, OVS Proj Strategy, User Spatial Control, Stream Chosen State, Selection Status, Modification Status, Channel Availability Status, Channel Perceptability).

Annex A

ASN.1 notations (Level C) Coded representation (Level D)

Useful definitions

-- *Copyright statement:*

-- -----

-- *(c) International Organization for Standardization 1996.*

-- *Permission to copy in any form is granted for use with conforming MHEG Engines and applications*

-- *as defined in ITU-T Recommendation T.171 provided this notice is included in all copies.*

--

ISOMHEG-ud {joint-iso-itu-t(2) mheg(19) version(1) useful-definitions(9)}

DEFINITIONS

IMPLICIT TAGS

::= BEGIN

EXPORTS

Aspect-Ratio,
Cat-Content-Classification,
Cat-Ext-Attribute,
Cat-Ext-Attribute-Param,
Catalogued-Event,
Catalogued-Extended-Elementary-Action,
Catalogued-Media-Type,
Catalogued-Script-Classification,
Catalogued-Style,
Channel-Perceptability,
Channel-Target-Param,
Class-ID,
Comparison-Value-Constant,
Component-Class,
Content-Hook,
Current-Point-Spec-Param,
Data-Reference,
Evaluated-Integer,
Evaluated-List,
Evaluated-Reference,
Evaluated-Value,
Generic-Integer,
Generic-Integer-Param,
Generic-List-Elc-ID-Param,
Generic-List-Param,
Generic-Numeric-Param,
Generic-Ratio-Param,
Generic-Value,
Generic-Value-Param,
GF-Param,
Initial-Point-Spec-Param,
Interaction-Status,
Interaction-Type-Param,
Macro-Def-ID,
Mh-object-Class,
Mh-Reference,
Mh-Target-Param,
Model-Class,
OVS-Proj-Strategy,
Point-Spec,
Point-Spec-Param,
Presentation-Priority,
Resizing-Strategy,
Return-Target-Param,
RPS-Assignment,
Rt-Component-Channel-Tg-Param,

Rt-Component-Reference,
 Rt-Component-Target-Param,
 Rt-Composite-Target-Param,
 Rt-Content-Reference,
 Rt-Content-Target-Param,
 Rt-Mux-Target-Param,
 Rt-Script-Target-Param,
 Rt-Target-Param,
 Script-Hook,
 Size,
 Size-Spec-Param,
 Socket-Target-Param,
 Spatial-Control-Param,
 Spatial-Position-Spec-Param,
 Target-Param,
 Temporal-Termination,
 Terminal-Point-Spec-Param,
 Update-Command,
 User-Spatial-Control,
 Value,
 OV;

IMPORTS

Registered-Content-Encoding,
 Registered-Script-Encoding,
 Registered-Content-Classification,
 Registered-Script-Classification,
 Registered-Media-Type,
 Registered-Style,
 Registered-Event,
 Registered-Extended-Elementary-Action,
 Registered-Extended-Attribute FROM ISOMHEG-cat {joint-iso-itu-t(2) mheg(19) version(1) catalogues(12)};

```

Mh-object-Class ::= SEQUENCE      {
mheg-id                          Mheg-ID OPTIONAL,
description                       [0] Description OPTIONAL
}
Class-ID ::= CHOICE              {
action-class-id                   [0] INTEGER { action-class-id (1)},
link-class-id                     [1] INTEGER { link-class-id (2)},
script-class-id                   [2] INTEGER { script-class-id (3)},
content-class-id                  [3] INTEGER { content-class-id (4)},
multiplexed-content-class-id      [4] INTEGER { multiplexed-content-class-id (5)},
composite-class-id                [5] INTEGER { composite-class-id (6)},
container-class-id                [6] INTEGER { container-class-id (7)},
descriptor-class-id               [7] INTEGER { descriptor-class-id (8)},
...
}
Description ::= SEQUENCE         {
name                              GraphicString OPTIONAL,
owner                             [0] GraphicString OPTIONAL,
version                           [1] GraphicString OPTIONAL,
date                              UTCTime OPTIONAL,
keywords                           SEQUENCE OF GraphicString OPTIONAL,
copyright                          [2] GraphicString OPTIONAL,
copyright-id                       OCTET STRING OPTIONAL,
copyright-number                   [3] OCTET STRING OPTIONAL,
licence                            [4] GraphicString OPTIONAL,
cache-priority                     INTEGER OPTIONAL,
comments                           [5] GraphicString OPTIONAL,
...
}
Model-Class ::= SEQUENCE        {
COMPONENTS OF Mh-object-Class
}
Component-Class ::= SEQUENCE   {
COMPONENTS OF Model-Class,
ops-initialisation               [1] OPS-Initialisation OPTIONAL
}
  
```

```

Tail-Param ::= CHOICE
tail
tail-macro
}
Tail-Macro ::= SEQUENCE
macro-def-id
tail
}
External-ID ::= SEQUENCE
public-id
system-id
}
Mheg-ID ::= SEQUENCE
application-id
mh-number
}
Socket-Identification ::= CHOICE
socket-id
alias
}
Socket-ID ::= SEQUENCE
root-rt-composite-reference
tail
}
Macro-Def-ID ::= CHOICE
string
integer
}
Reference ::= CHOICE
null-data
mheg-id-reference
external-id-reference
container-element-reference
null-mh
this
stream-id-reference
root-rt-id-reference
null-root-rt
socket-id-reference
channel-id-reference
default-channel
alias-reference
}
Tail-Reference ::= SEQUENCE
tail
tail-complement
}
Tail-Complement ::= ENUMERATED
children (1),
descendants (2)
}
Data-Reference ::= CHOICE
external-id
null-data
alias-reference
}
Mh-Reference ::= CHOICE
mheg-id-reference
external-id-reference
container-element-reference
null-mh
this
alias-reference
}
Container-Element-Reference ::= SEQUENCE
container-object-reference
tail-reference
}

```

```

Root-Rt-Reference ::= CHOICE      {
root-rt-id-reference           Root-Rt-ID-Reference,
null-root-rt                   NULL,
alias-reference                 GraphicString
}
Root-Rt-ID-Reference ::= SEQUENCE {
model-object-reference         Mh-Reference,
rt-number-reference           Rt-Number-Reference
}
Rt-Number-Reference ::= CHOICE   {
rt-number                      INTEGER,
question-mark                  [0] INTEGER { question-mark (1)},
star                           [1] INTEGER { star (2)}
}
Rt-Reference ::= CHOICE         {
root-rt-id-reference           Root-Rt-ID-Reference,
null-root-rt                   NULL,
socket-id-reference            [0] Socket-ID-Reference,
alias-reference                 GraphicString
}
Rt-Script-Reference ::= CHOICE  {
rt-script-id-reference         Rt-Script-ID-Reference,
null-root-rt                   NULL,
alias-reference                 GraphicString
}
Rt-Script-ID-Reference ::= SEQUENCE {
script-object-reference        Mh-Reference,
rt-number-reference           Rt-Number-Reference
}
Root-Rt-Component-Reference ::= CHOICE {
root-rt-component-id-ref       Root-Rt-Component-ID-Ref,
null-root-rt                   NULL,
alias-reference                 GraphicString
}
Root-Rt-Component-ID-Ref ::= SEQUENCE {
component-object-reference      Mh-Reference,
rt-number-reference           Rt-Number-Reference
}
Rt-Component-Reference ::= CHOICE {
root-rt-component-id-ref       [10] Root-Rt-Component-ID-Ref,
null-root-rt                   [11] NULL,
socket-id-reference            [12] Socket-ID-Reference,
alias-reference                 [13] GraphicString
}
Root-Rt-Content-Reference ::= CHOICE {
root-rt-content-id-ref         Root-Rt-Content-ID-Ref,
null-root-rt                   NULL,
alias-reference                 GraphicString
}
Root-Rt-Content-ID-Ref ::= SEQUENCE {
content-object-ref             Mh-Reference,
rt-number-reference           Rt-Number-Reference
}
Rt-Content-Reference ::= CHOICE {
root-rt-content-id-ref         Root-Rt-Content-ID-Ref,
null-root-rt                   NULL,
socket-id-reference            [0] Socket-ID-Reference,
alias-reference                 GraphicString
}
Root-Rt-Mux-Reference ::= CHOICE {
root-rt-mux-id-reference       Root-Rt-Mux-ID-Reference,
null-root-rt                   NULL,
alias-reference                 GraphicString
}
Root-Rt-Mux-ID-Reference ::= SEQUENCE {
multiplexed-content-object-ref Mh-Reference,
rt-number-reference           Rt-Number-Reference
}

```

```

Rt-Mux-Reference ::= CHOICE      {
root-rt-mux-id-reference      Root-Rt-Mux-ID-Reference,
null-root-rt                  NULL,
socket-id-reference           [0] Socket-ID-Reference,
alias-reference                GraphicString
}
Root-Rt-Composite-Reference ::= CHOICE  {
root-rt-composite-id-ref      Root-Rt-Composite-ID-Ref,
null-root-rt                  NULL,
alias-reference                GraphicString
}
Root-Rt-Composite-ID-Ref ::= SEQUENCE  {
composite-object-reference    Mh-Reference,
rt-number-reference           Rt-Number-Reference
}
Rt-Composite-Reference ::= CHOICE      {
root-rt-composite-id-ref      [28] Root-Rt-Composite-ID-Ref,
null-root-rt                  [29] NULL,
socket-id-reference           [30] Socket-ID-Reference,
alias-reference                [31] GraphicString
}
Socket-Reference ::= CHOICE           {
socket-id-reference           Socket-ID-Reference,
alias-reference                GraphicString
}
Socket-ID-Reference ::= SEQUENCE      {
root-rt-composite-reference    Root-Rt-Composite-Reference,
socket-tail-reference          Socket-Tail-Reference
}
Socket-Tail-Reference ::= SEQUENCE    {
tail                           SEQUENCE OF INTEGER OPTIONAL,
socket-tail-complement          Socket-Tail-Complement OPTIONAL
}
Socket-Tail-Complement ::= CHOICE     {
tail-complement                Tail-Complement,
question-mark-child             INTEGER { question-mark-child (1)},
question-mark-descendant        [0] INTEGER { question-mark-descendant (2)}
}
Channel-Reference ::= CHOICE          {
channel-id-reference            [34] INTEGER,
default-channel                 [35] NULL,
alias-reference                 [36] GraphicString
}
Rt-Component-Channel-Ref ::= CHOICE   {
rt-component-reference          [41] Rt-Component-Reference,
channel-reference               [42] Channel-Reference
}
Target-Param ::= CHOICE               {
target                          Generic-Reference,
target-macro                     [45] Target-Macro
}
Target-Macro ::= SEQUENCE             {
macro-def-id                    Macro-Def-ID,
target                           Generic-Reference OPTIONAL
}
Mh-Target-Param ::= CHOICE            {
mh-target                       [2] Mh-Target,
mh-target-macro                  [3] Mh-Target-Macro
}
Mh-Target-Macro ::= SEQUENCE         {
macro-def-id                    Macro-Def-ID,
mh-target                        Mh-Target OPTIONAL
}
Mh-Target ::= CHOICE                  {
mh-reference                     Mh-Reference,
evaluated-reference              Evaluated-Reference
}

```

```

Rt-Target-Param ::= CHOICE      {
rt-target          [5] Rt-Target,
rt-target-macro    [6] Rt-Target-Macro
}
Rt-Target-Macro ::= SEQUENCE    {
macro-def-id       Macro-Def-ID,
rt-target          Rt-Target OPTIONAL
}
Rt-Target ::= CHOICE           {
rt-reference       Rt-Reference,
evaluated-reference Evaluated-Reference
}
Rt-Script-Target-Param ::= CHOICE {
rt-script-target   [8] Rt-Script-Target,
rt-script-target-macro [9] Rt-Script-Target-Macro
}
Rt-Script-Target-Macro ::= SEQUENCE {
macro-def-id       Macro-Def-ID,
rt-script-target   Rt-Script-Target OPTIONAL
}
Rt-Script-Target ::= CHOICE     {
rt-script-reference Rt-Script-Reference,
evaluated-reference Evaluated-Reference
}
Rt-Component-Target-Param ::= CHOICE {
rt-component-target Rt-Component-Target,
rt-component-target-macro [15] Rt-Component-Target-Macro
}
Rt-Component-Target-Macro ::= SEQUENCE {
macro-def-id       Macro-Def-ID,
rt-component-target Rt-Component-Target OPTIONAL
}
Rt-Component-Target ::= CHOICE {
rt-component-reference Rt-Component-Reference,
evaluated-reference [14] Evaluated-Reference
}
Rt-Content-Target-Param ::= CHOICE {
rt-content-target   [26] Rt-Content-Target,
rt-content-target-macro [27] Rt-Content-Target-Macro
}
Rt-Content-Target-Macro ::= SEQUENCE {
macro-def-id       Macro-Def-ID,
rt-content-target   Rt-Content-Target OPTIONAL
}
Rt-Content-Target ::= CHOICE {
rt-content-reference Rt-Content-Reference,
evaluated-reference Evaluated-Reference
}
Rt-Mux-Target-Param ::= CHOICE {
rt-mux-target       [45] Rt-Mux-Target,
rt-mux-target-macro [46] Rt-Mux-Target-Macro
}
Rt-Mux-Target-Macro ::= SEQUENCE {
macro-def-id       Macro-Def-ID,
rt-mux-target      Rt-Mux-Target OPTIONAL
}
Rt-Mux-Target ::= CHOICE {
rt-mux-reference    Rt-Mux-Reference,
evaluated-reference Evaluated-Reference
}
Rt-Composite-Target-Param ::= CHOICE {
rt-composite-target Rt-Composite-Target,
rt-composite-target-macro [33] Rt-Composite-Target-Macro
}
Rt-Composite-Target-Macro ::= SEQUENCE {
macro-def-id       Macro-Def-ID,
rt-composite-target Rt-Composite-Target OPTIONAL
}

```

```

Rt-Composite-Target ::= CHOICE {
rt-composite-reference      Rt-Composite-Reference,
evaluated-reference        [32] Evaluated-Reference
}
Socket-Target-Param ::= CHOICE {
socket-target              Socket-Target,
socket-target-macro        [0] Socket-Target-Macro
}
Socket-Target-Macro ::= SEQUENCE {
macro-def-id              Macro-Def-ID,
socket-target              Socket-Target OPTIONAL
}
Socket-Target ::= CHOICE {
socket-reference          Socket-Reference,
evaluated-reference      Evaluated-Reference
}
Return-Target-Param ::= CHOICE {
generic-integer-param    Generic-Integer-Param,
using-application        NULL
}
Channel-Target-Param ::= CHOICE {
channel-target            Channel-Target,
channel-target-macro      [38] Channel-Target-Macro
}
Channel-Target-Macro ::= SEQUENCE {
macro-def-id              Macro-Def-ID,
channel-target            Channel-Target OPTIONAL
}
Channel-Target ::= CHOICE {
channel-reference        Channel-Reference,
evaluated-reference      [37] Evaluated-Reference
}
Rt-Component-Channel-Tg-Param ::= CHOICE {
rt-component-channel-tg  Rt-Component-Channel-Tg,
rt-component-channel-tg-macro [44] Rt-Component-Channel-Tg-Macro
}
Rt-Component-Channel-Tg-Macro ::= SEQUENCE {
macro-def-id              Macro-Def-ID,
rt-component-channel-tg  Rt-Component-Channel-Tg OPTIONAL
}
Rt-Component-Channel-Tg ::= CHOICE {
rt-component-channel-ref  Rt-Component-Channel-Ref,
evaluated-reference      [43] Evaluated-Reference
}
Generic-Value-Param ::= CHOICE {
generic-value            Generic-Value,
generic-value-macro      [50] Generic-Value-Macro
}
Generic-Value-Macro ::= SEQUENCE {
macro-def-id              Macro-Def-ID,
generic-value            Generic-Value OPTIONAL
}
Generic-Value ::= CHOICE {
value                    Value,
evaluated-value          Evaluated-Value
}
Value ::= CHOICE {
boolean                  BOOLEAN,
numeric                  INTEGER,
ratio                    Ratio,
string                   GraphicString,
list                     [0] SEQUENCE OF Value,
reference                 [1] Reference
}
Generic-Boolean-Param ::= CHOICE {
generic-boolean          Generic-Boolean,
generic-boolean-macro    Generic-Boolean-Macro
}

```

```

Generic-Boolean-Macro ::= SEQUENCE    {
macro-def-id           Macro-Def-ID,
generic-boolean       Generic-Boolean OPTIONAL
}
Generic-Boolean ::= CHOICE           {
boolean               BOOLEAN,
evaluated-boolean    Evaluated-Boolean
}
Generic-Numeric-Param ::= CHOICE     {
generic-numeric      Generic-Numeric,
generic-numeric-macro Generic-Numeric-Macro
}
Generic-Numeric-Macro ::= SEQUENCE   {
macro-def-id         Macro-Def-ID,
generic-numeric      Generic-Numeric OPTIONAL
}
Generic-Numeric ::= CHOICE           {
numeric              INTEGER,
evaluated-numeric    Evaluated-Numeric
}
Generic-Integer-Param ::= CHOICE     {
generic-integer      Generic-Integer,
generic-integer-macro Generic-Integer-Macro
}
Generic-Integer-Macro ::= SEQUENCE   {
macro-def-id         Macro-Def-ID,
generic-integer      Generic-Integer OPTIONAL
}
Generic-Integer ::= CHOICE           {
integer              INTEGER,
evaluated-integer    Evaluated-Integer
}
Generic-Ratio-Param ::= CHOICE       {
generic-ratio        Generic-Ratio,
generic-ratio-macro [0] Generic-Ratio-Macro
}
Generic-Ratio-Macro ::= SEQUENCE     {
macro-def-id         Macro-Def-ID,
generic-ratio        Generic-Ratio OPTIONAL
}
Generic-Ratio ::= CHOICE             {
ratio                Ratio,
evaluated-ratio      Evaluated-Ratio
}
Ratio ::= SEQUENCE                   {
numerator            INTEGER,
denominator          INTEGER OPTIONAL
}
Generic-String-Param ::= CHOICE      {
generic-string       Generic-String,
generic-string-macro Generic-String-Macro
}
Generic-String-Macro ::= SEQUENCE    {
macro-def-id         Macro-Def-ID,
generic-string       Generic-String OPTIONAL
}
Generic-String ::= CHOICE            {
string               GraphicString,
evaluated-string     Evaluated-String
}
Generic-Reference-Param ::= CHOICE   {
generic-reference    Generic-Reference,
generic-reference-macro [45] Generic-Reference-Macro
}

```

```

Generic-Reference-Macro ::= SEQUENCE {
macro-def-id          Macro-Def-ID,
generic-reference     Generic-Reference OPTIONAL
}
Generic-Reference ::= CHOICE {
reference             Reference,
evaluated-reference  Evaluated-Reference
}
Generic-List-Param ::= CHOICE {
generic-list         Generic-List,
generic-list-macro  [1] Generic-List-Macro
}
Generic-List-Macro ::= SEQUENCE {
macro-def-id        Macro-Def-ID,
generic-list        Generic-List OPTIONAL
}
Generic-List ::= CHOICE {
list                SEQUENCE OF Value,
evaluated-list     Evaluated-List
}
Generic-List-Elt-ID-Param ::= CHOICE {
generic-list-elt-id  SEQUENCE OF INTEGER,
generic-list-elt-id-macro [0] Generic-List-Elt-ID-Macro
}
Generic-List-Elt-ID-Macro ::= SEQUENCE {
macro-def-id        Macro-Def-ID,
generic-list-elt-id SEQUENCE OF INTEGER OPTIONAL
}
Get-Number-Of-Interacted-Sockets ::= SEQUENCE {
rt-composite-target-param  Rt-Composite-Target-Param,
interaction-type-param     Interaction-Type-Param
}
Get-Interaction-Status ::= SEQUENCE {
rt-component-target-param  Rt-Component-Target-Param,
interaction-type-param     Interaction-Type-Param
}
Get-Max-Interact-Required ::= SEQUENCE {
rt-component-target-param  Rt-Component-Target-Param,
interaction-type-param     Interaction-Type-Param
}
Get-Min-Interact-Required ::= SEQUENCE {
rt-component-target-param  Rt-Component-Target-Param,
interaction-type-param     Interaction-Type-Param
}
Get-Interaction-Ability ::= SEQUENCE {
rt-component-target-param  Rt-Component-Target-Param,
interaction-type-param     Interaction-Type-Param
}
Get-Stream-Chosen-State ::= SEQUENCE {
rt-mux-target-param       Rt-Mux-Target-Param,
stream-identification-param Stream-Identification-Param
}
Get-GVF ::= CHOICE {
rt-composite-target-param  Rt-Composite-Target-Param,
channel-target-param       Channel-Target-Param
}
Get-User-Spatial-Control ::= SEQUENCE {
rt-component-target-param  Rt-Component-Target-Param,
spatial-control-param      Spatial-Control-Param
}
Get-GSF ::= SEQUENCE {
rt-component-target-param  Rt-Component-Target-Param,
expected-axis-result-param Expected-Axis-Result-Param
}
Get-PVS-Position ::= SEQUENCE {
rt-component-target-param  Rt-Component-Target-Param,
point-type-param          Point-Type-Param,
expected-axis-result-param Expected-Axis-Result-Param
}

```

Get-PAP ::= SEQUENCE	{
rt-component-target-param	Rt-Component-Target-Param,
point-type-param	Point-Type-Param,
expected-axis-result-param	Expected-Axis-Result-Param
}	
Get-PVS ::= SEQUENCE	{
rt-component-target-param	Rt-Component-Target-Param,
point-type-param	Point-Type-Param,
expected-axis-result-param	Expected-Axis-Result-Param
}	
Get-OVS-Position ::= SEQUENCE	{
rt-component-target-param	Rt-Component-Target-Param,
point-type-param	Point-Type-Param,
expected-axis-result-param	Expected-Axis-Result-Param
}	
Get-OAP ::= SEQUENCE	{
rt-component-target-param	Rt-Component-Target-Param,
point-type-param	Point-Type-Param,
expected-axis-result-param	Expected-Axis-Result-Param
}	
Get-OVS ::= SEQUENCE	{
rt-component-target-param	Rt-Component-Target-Param,
point-type-param	Point-Type-Param,
expected-axis-result-param	Expected-Axis-Result-Param
}	
Get-POS ::= SEQUENCE	{
rt-component-target-param	Rt-Component-Target-Param,
expected-axis-result-param	Expected-Axis-Result-Param
}	
Get-OS ::= SEQUENCE	{
rt-component-target-param	Rt-Component-Target-Param,
expected-axis-result-param	Expected-Axis-Result-Param
}	
Get-PVD ::= SEQUENCE	{
rt-component-target-param	Rt-Component-Target-Param,
expected-pvd-result-param	Expected-PVD-Result-Param
}	
Get-OVD ::= SEQUENCE	{
rt-component-target-param	Rt-Component-Target-Param,
expected-ovd-result-param	Expected-OVD-Result-Param
}	
Get-Rt-Composite-Address ::= SEQUENCE	{
rt-composite-target-param	Rt-Composite-Target-Param,
navigation-command-param	Navigation-Command-Param
}	
Get-Data ::= SEQUENCE	{
content-target-param	Mh-Target-Param,
generic-list-elt-id-param	Generic-List-Elt-ID-Param OPTIONAL
}	
Get-Catalogued-Attribute ::= SEQUENCE	{
target-param	Target-Param,
cat-ext-attribute-param	Cat-Ext-Attribute-Param
}	
Get-Integer ::= CHOICE	{
get-preparation-status	Mh-Target-Param,
get-activation-status	[4] Mh-Target-Param,
get-rt-availability-status	Rt-Target-Param,
get-running-status	[7] Rt-Target-Param,
get-termination-status	Rt-Script-Target-Param,
get-presentation-priority	Rt-Component-Target-Param,
get-od	[16] Rt-Component-Target-Param,
get-pod	[17] Rt-Component-Target-Param,
get-pvd	[18] Get-PVD,
get-ctp	[19] Rt-Component-Target-Param,
get-temporal-termination	[20] Rt-Component-Target-Param,
get-pvd-position	[21] Rt-Component-Target-Param,

get-timestone-status	[22] Rt-Component-Target-Param,
get-aspect-ratio	[23] Rt-Component-Target-Param,
get-resizing-strategy	Rt-Composite-Target-Param,
get-ovs-proj-strategy	[24] Rt-Component-Target-Param,
get-user-spatial-control	[25] Get-User-Spatial-Control,
get-cv	Rt-Content-Target-Param,
get-pcv	[34] Rt-Content-Target-Param,
get-stream-chosen-state	[35] Get-Stream-Chosen-State,
get-min-interact-required	[36] Get-Min-Interact-Required,
get-max-interact-required	[37] Get-Max-Interact-Required,
get-number-of-interacted-sockets	[38] Get-Number-Of-Interacted-Sockets,
get-channel-availability-status	[39] Channel-Target-Param,
get-channel-perceptability	[40] Channel-Target-Param
}	
Get-Reference ::= CHOICE	{
get-rt-composite-address	[9] Get-Rt-Composite-Address,
get-rps-assignment	Rt-Component-Target-Param
}	
Get-List ::= CHOICE	{
get-catalogued-attribute	[44] Get-Catalogued-Attribute,
get-stream-choice	Rt-Mux-Target-Param,
get-style	[47] Rt-Component-Target-Param,
get-event	[48] Rt-Component-Channel-Tg-Param
}	
Get-Any ::= CHOICE	{
get-data	[16] Get-Data,
get-ovd	[17] Get-OVD,
get-gtf	[18] Rt-Component-Target-Param,
get-os	[19] Get-OS,
get-pos	[20] Get-POS,
get-ovs	[21] Get-OVS,
get-oap	[22] Get-OAP,
get-ovs-position	[23] Get-OVS-Position,
get-pvs	[24] Get-PVS,
get-pap	[25] Get-PAP,
get-pvs-position	[26] Get-PVS-Position,
get-gsf	[27] Get-GSF,
get-gvf	Get-GVF,
get-interaction-ability	[39] Get-Interaction-Ability,
get-interaction-status	[40] Get-Interaction-Status,
get-event-data	Rt-Component-Channel-Tg-Param
}	
Evaluated-Value ::= CHOICE	{
get-boolean	NULL,
get-numeric	Get-Integer,
get-ratio	[41] Rt-Component-Target-Param,
get-string	[42] NULL,
get-reference	[43] Get-Reference,
get-list	Get-List,
get-any	[49] Get-Any,
...	
}	
Evaluated-Boolean ::= CHOICE	{
get-boolean	NULL,
get-any	Get-Any
}	
Evaluated-Numeric ::= CHOICE	{
get-numeric	Get-Integer,
get-integer	[0] Get-Integer,
get-any	[1] Get-Any
}	
Evaluated-Integer ::= CHOICE	{
get-integer	Get-Integer,
get-any	[41] Get-Any
}	
Evaluated-Ratio ::= CHOICE	{
get-ratio	Rt-Component-Target-Param,
get-any	Get-Any
}	

```

Evaluated-String ::= CHOICE      {
get-string          NULL,
get-any            Get-Any
}
Evaluated-Reference ::= CHOICE  {
get-reference      Get-Reference,
get-any           Get-Any
}
Evaluated-List ::= CHOICE      {
get-list          Get-List,
get-any          [0] Get-Any
}
Content-Hook ::= SEQUENCE      {
catalogued-content-encoding  Catalogued-Content-Encoding OPTIONAL,
content-encoding-description [0] OCTET STRING OPTIONAL
}
Script-Hook ::= SEQUENCE      {
catalogued-script-encoding  Catalogued-Script-Encoding OPTIONAL,
script-encoding-description [0] OCTET STRING OPTIONAL
}
Catalogued-Content-Encoding ::= CHOICE  {
registered-content-encoding  Registered-Content-Encoding,
proprietary-content-encoding Prop-Cat-Entry-ID
}
Catalogued-Script-Encoding ::= CHOICE  {
registered-script-encoding  Registered-Script-Encoding,
proprietary-script-encoding Prop-Cat-Entry-ID
}
Cat-Content-Classification ::= CHOICE  {
registered-content-classification  Registered-Content-Classification,
prop-content-classification       Prop-Cat-Entry-ID
}
Catalogued-Script-Classification ::= CHOICE  {
registered-script-classification  Registered-Script-Classification,
prop-script-classification       Prop-Cat-Entry-ID
}
Catalogued-Media-Type ::= CHOICE  {
registered-media-type           Registered-Media-Type,
proprietary-media-type         Prop-Cat-Entry-ID
}
Catalogued-Style ::= CHOICE      {
registered-style               Registered-Style,
proprietary-style             Prop-Cat-Entry-ID
}
Catalogued-Event ::= CHOICE      {
registered-event               Registered-Event,
proprietary-event             Prop-Cat-Entry-ID
}
Catalogued-Extended-Elementary-Action ::= CHOICE  {
registered-extended-elementary-action  Registered-Extended-Elementary-Action,
proprietary-extended-elementary-action Prop-Cat-Entry-ID
}
Cat-Ext-Attribute ::= CHOICE      {
registered-extended-attribute  Registered-Extended-Attribute,
proprietary-extended-attribute Prop-Cat-Entry-ID
}
Prop-Cat-Entry-ID ::= CHOICE      {
tail          SEQUENCE OF INTEGER,
octet-string  OCTET STRING
}
OPS-Initialisation ::= SEQUENCE  {
od          OD OPTIONAL,
os          Size OPTIONAL
}

```

```

GF-Param ::= CHOICE
gf
gf-macro
}
GF-Macro ::= SEQUENCE
macro-def-id
gf
}
GF ::= CHOICE
generic-ratio
default-gf
}
Point ::= CHOICE
integer
terminal
}
Size ::= SEQUENCE
x-spatial-length
y-spatial-length
z-spatial-length
}
Spatial-Length ::= CHOICE
length
default-spatial-length
}
Size-Spec-Param ::= CHOICE
size-spec
size-spec-macro
}
Size-Spec-Macro ::= SEQUENCE
macro-def-id
size-spec
}
Size-Spec ::= CHOICE
lengths-spec
get-list
get-any
}
Lengths-Spec ::= SEQUENCE
x-length-spec
y-length-spec
z-length-spec
}
Length-Spec-Param ::= CHOICE
length-spec
length-spec-macro
}
Length-Spec-Macro ::= SEQUENCE
macro-def-id
length-spec
}
Length-Spec ::= CHOICE
length
relative-length
get-integer
get-ratio
get-any
}
Spatial-Position ::= SEQUENCE
x-point
y-point
z-point
}
Spatial-Position-Spec-Param ::= CHOICE
spatial-position-spec
spatial-position-spec-macro
}
{
GF,
[0] GF-Macro
}
{
Macro-Def-ID,
GF OPTIONAL
}
{
Generic-Ratio,
INTEGER { default-gf (1)}
}
{
INTEGER,
[0] INTEGER { terminal (1)}
}
{
Spatial-Length OPTIONAL,
[1] Spatial-Length OPTIONAL,
[2] Spatial-Length OPTIONAL
}
{
INTEGER,
[0] INTEGER { default-spatial-length (65536)}
}
{
Size-Spec,
[0] Size-Spec-Macro
}
{
Macro-Def-ID,
Size-Spec OPTIONAL
}
{
Lengths-Spec,
Get-List,
[50] Get-Any
}
{
[2]Length-Spec OPTIONAL,
[0] Length-Spec OPTIONAL,
[1] Length-Spec OPTIONAL
}
{
Length-Spec,
[0] Length-Spec-Macro
}
{
Macro-Def-ID,
Length-Spec OPTIONAL
}
{
INTEGER,
Ratio,
Get-Integer,
[46] Rt-Component-Target-Param,
[45] Get-Any
}
{
Point OPTIONAL,
[1] Point OPTIONAL,
[2] Point OPTIONAL
}
{
Spatial-Position-Spec,
[0] Spatial-Position-Spec-Macro
}

```

```

Spatial-Position-Spec-Macro ::= SEQUENCE {
macro-def-id           Macro-Def-ID,
spatial-position-spec  Spatial-Position-Spec OPTIONAL
}
Spatial-Position-Spec ::= CHOICE {
points-spec           Points-Spec,
get-list             Get-List,
get-any             [51] Get-Any
}
Points-Spec ::= SEQUENCE {
x-point-spec        [3] Point-Spec OPTIONAL,
y-point-spec        [1] Point-Spec OPTIONAL,
z-point-spec        [2] Point-Spec OPTIONAL
}
Point-Spec-Param ::= CHOICE {
point-spec          Point-Spec,
point-spec-macro    [1] Point-Spec-Macro
}
Point-Spec-Macro ::= SEQUENCE {
macro-def-id        Macro-Def-ID,
point-spec          Point-Spec OPTIONAL
}
Point-Spec ::= CHOICE {
point              Point,
relative-point     Ratio,
get-integer        Get-Integer,
get-ratio          [46] Rt-Component-Target-Param,
get-any           [45] Get-Any
}
Current-Point-Spec-Param ::= CHOICE {
current-point-spec  Current-Point-Spec,
current-point-spec-macro [3] Current-Point-Spec-Macro
}
Current-Point-Spec-Macro ::= SEQUENCE {
macro-def-id        Macro-Def-ID,
current-point-spec  Current-Point-Spec OPTIONAL
}
Current-Point-Spec ::= CHOICE {
point              Point,
relative-point     Ratio,
original-point-factor [1] Ratio,
current-point-factor [2] Ratio,
get-integer        [47] Get-Integer,
get-ratio          [46] Rt-Component-Target-Param,
get-any           [45] Get-Any
}
Initial-Point-Spec-Param ::= CHOICE {
initial-point-spec  Point-Spec,
initial-point-spec-macro [1] Initial-Point-Spec-Macro
}
Initial-Point-Spec-Macro ::= SEQUENCE {
macro-def-id        Macro-Def-ID,
initial-point-spec  Point-Spec OPTIONAL
}
Terminal-Point-Spec-Param ::= CHOICE {
terminal-point-spec  Point-Spec,
terminal-point-spec-macro [1] Terminal-Point-Spec-Macro
}
Terminal-Point-Spec-Macro ::= SEQUENCE {
macro-def-id        Macro-Def-ID,
terminal-point-spec  Point-Spec OPTIONAL
}
Channel-Perceptability ::= ENUMERATED {
on (1),
off (2)
}

```

```

Channel-Availability-Status ::= ENUMERATED    {
available (1),
not-available (2)
}
Modifiability ::= ENUMERATED    {
modifiable (1),
not-modifiable (2)
}
Selectability ::= ENUMERATED    {
selectable (1),
not-selectable (2)
}
Modification-Status ::= ENUMERATED    {
modified (1),
not-modified (2),
modifying (3)
}
Selection-Status ::= ENUMERATED {
selected (1),
not-selected (2)
}
Interaction-Status ::= CHOICE      {
selection-status           Selection-Status,
modification-status       [0] Modification-Status
}
Interaction-Type ::= ENUMERATED    {
selection (1),
modification (2)
}
Interaction-Type-Macro ::= SEQUENCE {
macro-def-id               Macro-Def-ID,
interaction-type           Interaction-Type OPTIONAL
}
Interaction-Type-Param ::= CHOICE  {
interaction-type           Interaction-Type,
interaction-type-macro    Interaction-Type-Macro
}
Stream-Identification ::= CHOICE   {
stream-id                 SEQUENCE OF INTEGER,
all-streams               INTEGER { all-streams (1)}
}
Stream-Identification-Macro ::= SEQUENCE {
macro-def-id             Macro-Def-ID,
stream-identification    Stream-Identification OPTIONAL
}
Stream-Identification-Param ::= CHOICE {
stream-identification    Stream-Identification,
stream-identification-macro [0] Stream-Identification-Macro
}
Stream-Chosen-State ::= ENUMERATED {
chosen (1),
not-chosen (2),
partially-chosen (3)
}
PCV ::= CHOICE                {
integer                   INTEGER,
min-avr                  [0] INTEGER { min-avr (0)},
max-avr                  [1] INTEGER { max-avr (255)}
}
CV ::= CHOICE                 {
integer                   INTEGER,
min-avr                  [0] INTEGER { min-avr (0)},
max-avr                  [1] INTEGER { max-avr (255)}
}
OV ::= CHOICE                 {
integer                   INTEGER,
min-avr                  [0] INTEGER { min-avr (0)},
max-avr                  [1] INTEGER { max-avr (255)}
}

```

```

Point-Type-Param ::= ENUMERATED      {
absolute (1),
relative (2)
}
Spatial-Control-Macro ::= SEQUENCE    {
macro-def-id          Macro-Def-ID,
spatial-control       Spatial-Control OPTIONAL
}
Spatial-Control-Param ::= CHOICE     {
spatial-control       Spatial-Control,
spatial-control-macro Spatial-Control-Macro
}
Expected-Axis-Result-Param ::= ENUMERATED  {
x (1),
y (2),
z (3),
xyz (4)
}
User-Spatial-Control ::= ENUMERATED      {
allowed (1),
not-allowed (2)
}
Spatial-Control ::= ENUMERATED  {
moving (1),
resizing (2),
scaling (3),
scrolling (4)
}
GSF ::= SEQUENCE                {
x-gf          GF OPTIONAL,
y-gf          [0] GF OPTIONAL,
z-gf          [1] GF OPTIONAL
}
OVS-Proj-Strategy ::= ENUMERATED  {
fixed (1),
calculated (2)
}
Resizing-Strategy ::= ENUMERATED  {
fixed (1),
minimum (2),
grows-only (3)
}
Aspect-Ratio ::= ENUMERATED      {
preserved (1),
not-preserved (2)
}
Expected-PVD-Result ::= ENUMERATED  {
initial-temporal-position (1),
terminal-temporal-position (2),
duration (3)
}
Expected-PVD-Result-Macro ::= SEQUENCE  {
macro-def-id          Macro-Def-ID,
expected-pvd-result    Expected-PVD-Result OPTIONAL
}
Expected-PVD-Result-Param ::= CHOICE   {
expected-pvd-result    Expected-PVD-Result,
expected-pvd-result-macro Expected-PVD-Result-Macro
}
Expected-OVD-Result ::= ENUMERATED      {
initial-temporal-position (1),
terminal-temporal-position (2),
duration (3)
}
Expected-OVD-Result-Macro ::= SEQUENCE  {
macro-def-id          Macro-Def-ID,
expected-ovd-result    Expected-OVD-Result OPTIONAL
}

```

```

Expected-OVD-Result-Param ::= CHOICE      {
expected-ovd-result          Expected-OVD-Result,
expected-ovd-result-macro    Expected-OVD-Result-Macro
}
Temporal-Termination ::= ENUMERATED      {
freeze (1),
stop (2)
}
OD ::= CHOICE                          {
integer                       INTEGER,
infinite                       [0] INTEGER { infinite (1) }
}
Presentation-Priority ::= CHOICE        {
generic-integer               Generic-Integer,
up-priority                    [1] INTEGER { up-priority (1) },
down-priority                  [0] INTEGER { down-priority (2) }
}
RPS-Assignment ::= CHOICE              {
channel-reference              Channel-Reference,
evaluated-reference           [0] Evaluated-Reference,
prps                          INTEGER { prps (1) }
}
Ancestor ::= CHOICE                    {
integer                        [4] INTEGER,
root                          [5] INTEGER { root (1) }
}
EmptyChild ::= CHOICE                  {
integer                        [1] INTEGER,
last                          [2] INTEGER { last (1) }
}
Child ::= CHOICE                       {
generic-integer                [6] Generic-Integer,
last                          INTEGER { last (1) },
random                        [0] INTEGER { random (2) }
}
Navigation-Command ::= CHOICE          {
child                          Child,
emptychild                     EmptyChild,
sibling                        [3] INTEGER,
ancestor                       Ancestor
}
Navigation-Command-Macro ::= SEQUENCE   {
macro-def-id                   Macro-Def-ID,
navigation-command             Navigation-Command OPTIONAL
}
Navigation-Command-Param ::= CHOICE     {
navigation-command             Navigation-Command,
navigation-command-macro       Navigation-Command-Macro
}
Termination-Status ::= ENUMERATED      {
terminated (1),
not-terminated (2)
}
Running-Status ::= ENUMERATED          {
running (1),
not-running (2)
}
Rt-Availability-Status ::= ENUMERATED   {
available (1),
not-available (2)
}
Activation-Status ::= ENUMERATED        {
active (1),
inactive (2)
}

```

```

Preparation-Status ::= ENUMERATED    {
ready (1),
not-ready (2)
}
Cat-Ext-Attribute-Macro ::= SEQUENCE    {
macro-def-id                Macro-Def-ID,
cat-ext-attribute            Cat-Ext-Attribute OPTIONAL
}
Cat-Ext-Attribute-Param ::= CHOICE    {
cat-ext-attribute            Cat-Ext-Attribute,
cat-ext-attribute-macro      [0] Cat-Ext-Attribute-Macro
}
Update-Command ::= ENUMERATED    {
add (1),
remove (2),
replace (3)
}
Comparison-Value-Constant ::= CHOICE    {
preparation-status            Preparation-Status,
activation-status              [0] Activation-Status,
rt-availability-status         [1] Rt-Availability-Status,
running-status                 [2] Running-Status,
termination-status             [3] Termination-Status,
prps                           INTEGER { prps (1) },
infinite                       [4] INTEGER { infinite (2) },
temporal-termination           [5] Temporal-Termination,
aspect-ratio                   [6] Aspect-Ratio,
resizing-strategy               [7] Resizing-Strategy,
ovs-proj-strategy               [8] OVS-Proj-Strategy,
user-spatial-control           [9] User-Spatial-Control,
stream-chosen-state            [10] Stream-Chosen-State,
selection-status               [11] Selection-Status,
modification-status            [12] Modification-Status,
channel-availability-status     [13] Channel-Availability-Status,
channel-perceptability         [14] Channel-Perceptability
}
END

Action Class
-- Copyright statement:
-- -----
-- (c) International organization for Standardization 1996.
-- Permission to copy in any form is granted for use with conforming MHEG Engines and applications
-- as defined in ITU-T Recommendation T.171 provided this notice is included in all copies.
--

ISOMHEG-ac {joint-iso-itu-t(2) mheg(19) version(1) action-class(1)}
DEFINITIONS
    IMPLICIT TAGS
    ::= BEGIN

EXPORTS
    Action-Object,
    Action-Class;

IMPORTS
    Mh-object-Class,
    Macro-Def-ID,
    Mh-Reference FROM ISOMHEG-ud {joint-iso-itu-t(2) mheg(19) version(1) action-class(9)}
    Elementary-Action FROM ISOMHEG-ea {joint-iso-itu-t(2) mheg(19) version(1) action-class(10)};

Action-Class ::= SEQUENCE    {
OBJECT IDENTIFIER DEFAULT {joint-iso-itu-t(2) mheg(19) version(1) action-class(1)},
COMPONENTS OF Mh-object-Class,
synchro-indicator-param        [1] Synchro-Indicator-Param,
synchronised-action-list       SEQUENCE OF Synchronised-Action,
...
}

```

```

Synchro-Indicator-Param ::= CHOICE      {
synchro-indicator          Synchro-Indicator,
synchro-indicator-macro    Synchro-Indicator-Macro
}
Synchro-Indicator-Macro ::= SEQUENCE    {
macro-def-id                Macro-Def-ID,
synchro-indicator          Synchro-Indicator OPTIONAL
}
Synchro-Indicator ::= ENUMERATED      {
serial (1),
parallel (2)
}
Synchronised-Action ::= CHOICE        {
elementary-action          Elementary-Action,
action-object              [60] Action-Object
}
Action-Object ::= CHOICE               {
action-object-reference    [0] Mh-Reference,
action-class                [1] Action-Class
}
}
END

```

Link Class

-- *Copyright statement:*

-- (c) International organization for Standardization 1996.

-- Permission to copy in any form is granted for use with conforming MHEG Engines and applications

-- as defined in ITU-T Recommendation T.171 provided this notice is included in all copies.

--

ISOMHEG-lk {joint-iso-itu-t(2) mheg(19) version(1) link-class(2)}

DEFINITIONS

IMPLICIT TAGS

::= BEGIN

EXPORTS

Link-Class,
Link-Effect;

IMPORTS

Mh-object-Class,
Evaluated-Value,
Generic-Value,
Comparison-Value-Constant,
Macro-Def-ID FROM ISOMHEG-ud {joint-iso-itu-t(2) mheg(19) version(1) useful-definitions(9)}
Action-Object FROM ISOMHEG-ac {joint-iso-itu-t(2) mheg(19) version(1) action-class(1)};

```

Link-Class ::= SEQUENCE      {
OBJECT IDENTIFIER DEFAULT {joint-iso-itu-t(2) mheg(19) version(1) link-class(2)},
COMPONENTS OF Mh-object-Class,
link-condition              [1] Link-Condition,
link-effect                  Link-Effect,
...
}

```

```

Link-Condition ::= CHOICE    {
trigger-condition          Trigger-Condition,
logical-combination        [0] Logical-Combination
}

```

```

Trigger-Condition ::= SEQUENCE {
source-value              Evaluated-Value,
previous-condition        Comparison-Operation OPTIONAL,
current-condition         [0] Comparison-Operation
}

```

```

Constraint-Condition ::= SEQUENCE {
source-value              Evaluated-Value,
current-condition         Comparison-Operation
}

```

```

Comparison-Operation ::= SEQUENCE    {
comparison-operator      Comparison-Operator,
comparison-value        Comparison-Value
}
Comparison-Value ::= CHOICE    {
generic-value           Generic-Value,
comparison-value-constant [50] Comparison-Value-Constant,
unspecified            [51] INTEGER { unspecified (1)}
}
Comparison-Operator ::= ENUMERATED    {
equal (1),
not-equal (2),
greater (3),
greater-equal (4),
less (5),
less-equal (6)
}
Logical-Combination ::= SEQUENCE    {
logical-operator        Logical-Operator,
condition-list          SEQUENCE OF Condition
}
Logical-Operator ::= ENUMERATED    {
and (1),
or (2),
xor (3),
not (4)
}
Condition ::= CHOICE    {
trigger-condition      Trigger-Condition,
constraint-condition   [0] Constraint-Condition,
logical-combination    [1] Logical-Combination
}
Link-Effect ::= SEQUENCE    {
macro-parameter-resolution-list SEQUENCE OF Macro-Parameter-Resolution OPTIONAL,
action-object          Action-Object
}
Macro-Parameter-Resolution ::= SEQUENCE    {
macro-def-id           Macro-Def-ID,
usage-value            Generic-Value
}
END

Script Class
-- Copyright statement:
-- -----
-- (c) International organization for Standardization 1996.
-- Permission to copy in any form is granted for use with conforming MHEG Engines and applications
-- as defined in ITU-T Recommendation T.171 provided this notice is included in all copies.
--

ISOMHEG-sc {joint-iso-itu-t(2) mheg(19) version(1) script-class(3)}
DEFINITIONS
    IMPLICIT TAGS
    ::= BEGIN

EXPORTS
    Script-Class;

IMPORTS
    Model-Class,
    Catalogued-Script-Classification,
    Script-Hook,
    Data-Reference FROM ISOMHEG-ud {joint-iso-itu-t(2) mheg(19) version(1) useful-definitions(9)}
    InterchangedScript FROM ISOMHEG-sir {joint-iso-itu-t(2) mheg(19) version(1) script-interchange-representation(11)};

```

```

Script-Class ::= SEQUENCE      {
OBJECT IDENTIFIER DEFAULT {joint-iso-itu-t(2) mheg(19) version(1) script-class(3)},
COMPONENTS OF Model-Class,
script-classification          [1] Catalogued-Script-Classification OPTIONAL,
script-hook                    [2] Script-Hook,
script-data                    Script-Data,
...
}
Script-Data ::= CHOICE        {
script-inclusion                 Script-Inclusion,
data-reference                 Data-Reference
}
Script-Inclusion ::= CHOICE     {
bit-string                     BIT STRING,
octet-string                   OCTET STRING,
interchangedscript             InterchangedScript
}
END

Content Class
-- Copyright statement:
-- -----
-- (c) International organization for Standardization 1996.
-- Permission to copy in any form is granted for use with conforming MHEG Engines and applications
-- as defined in ITU-T Recommendation T.171 provided this notice is included in all copies.
--

ISOMHEG-ct {joint-iso-itu-t(2) mheg(19) version(1) content-class(4)}
DEFINITIONS
    IMPLICIT TAGS
    ::= BEGIN

EXPORTS
    Content,
    Content-Class;

IMPORTS
    OV,
    Component-Class,
    Cat-Content-Classification,
    Content-Hook,
    Data-Reference,
    Generic-Value FROM ISOMHEG-ud {joint-iso-itu-t(2) mheg(19) version(1) content-class(9)};

Content-Class ::= SEQUENCE      {
OBJECT IDENTIFIER DEFAULT {joint-iso-itu-t(2) mheg(19) version(1) content-class(4)},
COMPONENTS OF Content,
...
}

Content ::= SEQUENCE            {
COMPONENTS OF Component-Class,
cat-content-classification      [2] Cat-Content-Classification OPTIONAL,
content-hook                    [3] Content-Hook,
ov                              [51] OV OPTIONAL,
content-data                    Content-Data
}

Content-Data ::= CHOICE        {
data-inclusion                   Data-Inclusion,
data-reference                  [50] Data-Reference
}
Data-Inclusion ::= CHOICE       {
bit-string                     BIT STRING,
octet-string                   OCTET STRING,
generic-value                   Generic-Value
}
END

```

Multiplexed Content Class

-- *Copyright statement:*

-- -----

-- (c) International organization for Standardization 1996.

-- Permission to copy in any form is granted for use with conforming MHEG Engines and applications

-- as defined in ITU-T Recommendation T.171 provided this notice is included in all copies.

--

ISOMHEG-mu {joint-iso-itu-t(2) mheg(19) version(1) multiplexed-content-class(5)}

DEFINITIONS

IMPLICIT TAGS

::= BEGIN

EXPORTS

Multiplexed-Content-Class;

IMPORTS

Content FROM ISOMHEG-ct {joint-iso-itu-t(2) mheg(19) version(1) content-class(4)}

Cat-Content-Classification,

Content-Hook FROM ISOMHEG-ud {joint-iso-itu-t(2) mheg(19) version(1) useful-definitions(9)};

Multiplexed-Content-Class ::= SEQUENCE {

OBJECT IDENTIFIER DEFAULT {joint-iso-itu-t(2) mheg(19) version(1) multiplexed-content-class(5)},

COMPONENTS OF Content,

multiplexed-stream-list SEQUENCE OF Multiplexed-Stream,

...

}

Multiplexed-Stream ::= SEQUENCE {

stream-id SEQUENCE OF INTEGER,

cat-content-classification Cat-Content-Classification OPTIONAL,

content-hook [0] Content-Hook OPTIONAL

}

END

Composite Class

-- *Copyright statement:*

-- -----

-- (c) International organization for Standardization 1996.

-- Permission to copy in any form is granted for use with conforming MHEG Engines and applications

-- as defined in ITU-T Recommendation T.171 provided this notice is included in all copies.

--

ISOMHEG-co {joint-iso-itu-t(2) mheg(19) version(1) composite-class(6)}

DEFINITIONS

IMPLICIT TAGS

::= BEGIN

EXPORTS

Composite-Class;

IMPORTS

Component-Class,

Mh-Reference FROM ISOMHEG-ud {joint-iso-itu-t(2) mheg(19) version(1) useful-definition(9)}

Link-Effect,

Link-Class FROM ISOMHEG-lk {joint-iso-itu-t(2) mheg(19) version(1) link-class(2)}

Action-Object FROM ISOMHEG-ac {joint-iso-itu-t(2) mheg(19) version(1) action-class(1)}

Content-Class FROM ISOMHEG-ct {joint-iso-itu-t(2) mheg(19) version(1) content-class(4)}

Multiplexed-Content-Class FROM ISOMHEG-mu {joint-iso-itu-t(2) mheg(19) version(1) multiplexed-content-class(5)};

Composite-Class ::= SEQUENCE {

OBJECT IDENTIFIER DEFAULT {joint-iso-itu-t(2) mheg(19) version(1) composite-class(6)},

COMPONENTS OF Component-Class,

availability-start-up [14] Availability-Start-up OPTIONAL,

availability-close-down Availability-Close-down OPTIONAL,

rt-availability-start-up Rt-Availability-Start-up OPTIONAL,

```

rt-availability-close-down      [10] Link-Effect OPTIONAL,
action-object-list             [11] SEQUENCE OF Action-Object OPTIONAL,
link-object-list               [12] SEQUENCE OF Link-Object OPTIONAL,
nb-of-elements                 [13] INTEGER,
composition-element-list       SEQUENCE OF Composition-Element OPTIONAL,
...
}
Availability-Start-up ::= CHOICE {
link-effect                    Link-Effect,
automatic-start-up-1           INTEGER { automatic-start-up-1 (1)},
automatic-start-up-2           [0] INTEGER { automatic-start-up-2 (2)},
automatic-start-up-3           [1] INTEGER { automatic-start-up-3 (3)},
automatic-start-up-4           [2] INTEGER { automatic-start-up-4 (4)},
automatic-start-up-5           [3] INTEGER { automatic-start-up-5 (5)}
}
Availability-Close-down ::= CHOICE {
link-effect                    [4] Link-Effect,
automatic-close-down-1         [5] INTEGER { automatic-close-down-1 (1)},
automatic-close-down-2         [6] INTEGER { automatic-close-down-2 (2)},
automatic-close-down-3         [7] INTEGER { automatic-close-down-3 (3)}
}
Rt-Availability-Start-up ::= CHOICE {
link-effect                    [8] Link-Effect,
automatic-rt-start-up          [9] INTEGER { automatic-rt-start-up (1)}
}
Link-Object ::= CHOICE {
link-class                     Link-Class,
link-object-reference          [0] Mh-Reference
}
Composition-Element ::= SEQUENCE {
index                         INTEGER,
associated-model               Associated-Model
}
Associated-Model ::= CHOICE {
component-object-reference     Mh-Reference,
content-class                  [3] Content-Class,
multiplexed-content-class      [4] Multiplexed-Content-Class,
composite-class                [5] Composite-Class,
label                          [6] GraphicString
}
END

```

Container Class

-- *Copyright statement:*

-- -----

-- *(c) International organization for Standardization 1996.*

-- *Permission to copy in any form is granted for use with conforming MHEG Engines and applications*

-- *as defined in ITU-T Recommendation T.171 provided this notice is included in all copies.*

--

ISOMHEG-cr {joint-iso-itu-t(2) mheg(19) version(1) container-class(7)}

DEFINITIONS

IMPLICIT TAGS

::= BEGIN

EXPORTS;

IMPORTS

Mh-object-Class,

Mh-Reference FROM ISOMHEG-ud {joint-iso-itu-t(2) mheg(19) version(1) useful-definition(9)}

Link-Effect,

Link-Class FROM ISOMHEG-lk {joint-iso-itu-t(2) mheg(19) version(1) link-class(2)}

Action-Class FROM ISOMHEG-ac {joint-iso-itu-t(2) mheg(19) version(1) action-class(1)}

Script-Class FROM ISOMHEG-sc {joint-iso-itu-t(2) mheg(19) version(1) script-class(3)}

Content-Class FROM ISOMHEG-ct {joint-iso-itu-t(2) mheg(19) version(1) content-class(4)}

Multiplexed-Content-Class FROM ISOMHEG-mu {joint-iso-itu-t(2) mheg(19) version(1) multiplexed-content-class(5)}

Composite-Class FROM ISOMHEG-co {joint-iso-itu-t(2) mheg(19) version(1) composite-class(6)}

Descriptor-Class FROM ISOMHEG-de {joint-iso-itu-t(2) mheg(19) version(1) descriptor-class(8)};

```

Container-Class ::= SEQUENCE      {
OBJECT IDENTIFIER DEFAULT {joint-iso-itu-t(2) mheg(19) version(1) container-class(7)},
COMPONENTS OF Mh-object-Class,
container-start-up                [10] Container-Start-up OPTIONAL,
container-close-down              Container-Close-down OPTIONAL,
container-element-list            [9] SEQUENCE OF Container-Element,
...
}

Container-Start-up ::= CHOICE     {
link-effect                       Link-Effect,
automatic-container-start-up-1    INTEGER { automatic-container-start-up-1 (1)},
automatic-container-start-up-2    [0] INTEGER { automatic-container-start-up-2 (2)},
automatic-container-start-up-3    [1] INTEGER { automatic-container-start-up-3 (3)},
automatic-container-start-up-4    [2] INTEGER { automatic-container-start-up-4 (4)},
automatic-container-start-up-5    [3] INTEGER { automatic-container-start-up-5 (5)},
automatic-container-start-up-6    [4] INTEGER { automatic-container-start-up-6 (6)},
automatic-container-start-up-7    [5] INTEGER { automatic-container-start-up-7 (7)},
automatic-container-start-up-8    [6] INTEGER { automatic-container-start-up-8 (8)}
}

Container-Close-down ::= CHOICE  {
link-effect                       [7] Link-Effect,
automatic-container-close-down    [8] INTEGER { automatic-container-close-down (1)}
}

Container-Element ::= CHOICE     {
mh-reference                      Mh-Reference,
action-class                      [3] Action-Class,
link-class                        [4] Link-Class,
script-class                      [5] Script-Class,
content-class                    [6] Content-Class,
multiplexed-content-class        [7] Multiplexed-Content-Class,
composite-class                  [8] Composite-Class,
container-class                  [9] Container-Class,
descriptor-class                 [10] Descriptor-Class,
...
}
END

```

Descriptor Class

-- *Copyright statement:*

-- -----

-- (c) International organization for Standardization 1996.

-- Permission to copy in any form is granted for use with conforming MHEG Engines and applications

-- as defined in ITU-T Recommendation T.171 provided this notice is included in all copies.

--

ISOMHEG-de {joint-iso-itu-t(2) mheg(19) version(1) descriptor-class(8)}

DEFINITIONS

IMPLICIT TAGS

::= BEGIN

EXPORTS

Descriptor-Class;

IMPORTS

Mh-object-Class,

Mh-Reference,

Catalogued-Style,

Catalogued-Extended-Elementary-Action,

Cat-Ext-Attribute,

Class-ID,

Catalogued-Script-Classification,

Script-Hook,

Cat-Content-Classification,

Content-Hook,

Catalogued-Media-Type,

Value,

Catalogued-Event FROM ISOMHEG-ud {joint-iso-itu-t(2) mheg(19) version(1) useful-definitions(9)};

```

Descriptor-Class ::= SEQUENCE {
OBJECT IDENTIFIER DEFAULT {joint-iso-itu-t(2) mheg(19) version(1) descriptor-class(8)},
COMPONENTS OF Mh-object-Class,
related-object-list [1] SEQUENCE OF Related-Object OPTIONAL,
other-descriptor-list [2] SEQUENCE OF Mh-Reference OPTIONAL,
readme GraphicString OPTIONAL,
system-readable-material System-Readable-Material OPTIONAL,
channel-information-list [3] SEQUENCE OF Channel-Information OPTIONAL,
catalogued-style-information-list [4] SEQUENCE OF Catalogued-Style OPTIONAL,
cat-ext-elementary-action-info-list [5] SEQUENCE OF Catalogued-Extended-Elementary-Action OPTIONAL,
cat-ext-attribute-info-list [6] SEQUENCE OF Cat-Ext-Attribute OPTIONAL,
...
}
Related-Object ::= SEQUENCE {
mh-reference Mh-Reference,
object-information Object-Information OPTIONAL
}
Object-Information ::= SEQUENCE {
object-size INTEGER OPTIONAL,
class-id Class-ID,
class-specific-information Class-Specific-Information OPTIONAL,
offset INTEGER OPTIONAL
}
Class-Specific-Information ::= CHOICE {
script-class-information Script-Class-Information,
content-class-information [0] Content-Class-Information,
mux-content-class-info [1] Mux-Content-Class-Info
}
Script-Class-Information ::= SEQUENCE {
script-classification Catalogued-Script-Classification OPTIONAL,
script-hook [0] Script-Hook OPTIONAL
}
Content-Class-Information ::= SEQUENCE {
cat-content-classification Cat-Content-Classification OPTIONAL,
content-hook [0] Content-Hook OPTIONAL,
alternative-object-list [1] SEQUENCE OF Alternative-Object OPTIONAL
}
Mux-Content-Class-Info ::= SEQUENCE {
content-class-information Content-Class-Information,
number-of-streams INTEGER OPTIONAL,
stream-information-list SEQUENCE OF Stream-Information OPTIONAL
}
Stream-Information ::= SEQUENCE {
stream-id SEQUENCE OF INTEGER,
content-class-information Content-Class-Information
}
Alternative-Object ::= SEQUENCE {
content-object-ref Mh-Reference,
content-hook Content-Hook OPTIONAL,
alternative-descriptor-object [0] Mh-Reference OPTIONAL,
alternative-readme GraphicString OPTIONAL
}
System-Readable-Material ::= CHOICE {
bit-string BIT STRING,
octet-string OCTET STRING
}
Channel-Information ::= SEQUENCE {
channel-id INTEGER,
x-min INTEGER OPTIONAL,
x-max [0] INTEGER OPTIONAL,
y-min [1] INTEGER OPTIONAL,
y-max [2] INTEGER OPTIONAL,
z-min [3] INTEGER OPTIONAL,
z-max [4] INTEGER OPTIONAL,
x-resolution [5] INTEGER OPTIONAL,
y-resolution [6] INTEGER OPTIONAL,

```

```

z-resolution          [7] INTEGER OPTIONAL,
t-resolution          [8] INTEGER OPTIONAL,
f-min                 [9] INTEGER OPTIONAL,
f-max                 [10] INTEGER OPTIONAL,
audio-dynamic         [11] INTEGER OPTIONAL,
channel-media-type-list SEQUENCE OF Catalogued-Media-Type OPTIONAL,
event-mapping-list    [12] SEQUENCE OF Event-Mapping OPTIONAL
}
Event-Mapping ::= SEQUENCE {
event                SEQUENCE OF Value,
catalogued-event     Catalogued-Event OPTIONAL
}
END

```

Elementary Action Class

-- *Copyright statement:*

-- -----

-- (c) International organization for Standardization 1996.

-- Permission to copy in any form is granted for use with conforming MHEG Engines and applications

-- as defined in ITU-T Recommendation T.171 provided this notice is included in all copies.

--

ISOMHEG-ea {joint-iso-itu-t(2) mheg(19) version(1) elementary-actions(10)}

DEFINITIONS

IMPLICIT TAGS

::= BEGIN

EXPORTS

Elementary-Action;

IMPORTS

Generic-Integer-Param,
Macro-Def-ID,
Rt-Component-Reference,
Return-Target-Param,
Generic-Numeric-Param,
Generic-Value-Param,
Mh-Reference,
Target-Param,
Update-Command,
Catalogued-Extended-Elementary-Action,
Cat-Ext-Attribute-Param,
Mh-Target-Param,
Generic-Value,
Generic-List-Elt-ID-Param,
Rt-Target-Param,
Generic-Integer,
Rt-Script-Target-Param,
Socket-Target-Param,
Evaluated-Reference,
Rt-Component-Target-Param,
RPS-Assignment,
Generic-Ratio-Param,
Presentation-Priority,
Initial-Point-Spec-Param,
Terminal-Point-Spec-Param,
Current-Point-Spec-Param,
Temporal-Termination,
Point-Spec-Param,
GF-Param,
Point-Spec,
Aspect-Ratio,
Rt-Composite-Target-Param,
Resizing-Strategy,
OVS-Proj-Strategy,
Size-Spec-Param,
Spatial-Position-Spec-Param,
Spatial-Control-Param,
User-Spatial-Control,

Rt-Content-Target-Param,
Channel-Target-Param,
Rt-Mux-Target-Param,
Interaction-Status,
Interaction-Type-Param,
Catalogued-Style,
Rt-Content-Reference,
Channel-Perceptability,
Size,
Evaluated-Integer,
Evaluated-List,
Rt-Component-Channel-Tg-Param,
Generic-List-Param FROM ISOMHEG-ud {joint-iso-itu-t(2) mheg(19) version(1) useful-definitions(9)};

```

Delay ::= SEQUENCE {
temporal-unit-ref-param      Temporal-Unit-Ref-Param,
duration-param              Generic-Integer-Param
}
Temporal-Unit-Ref-Param ::= CHOICE {
temporal-unit-ref          Temporal-Unit-Ref,
temporal-unit-ref-macro   Temporal-Unit-Ref-Macro
}
Temporal-Unit-Ref-Macro ::= SEQUENCE {
macro-def-id              Macro-Def-ID,
temporal-unit-ref        Temporal-Unit-Ref OPTIONAL
}
Temporal-Unit-Ref ::= CHOICE {
rt-component-reference    Rt-Component-Reference,
default-gf                INTEGER { default-gf (1) }
}
Return ::= SEQUENCE {
return-target-param-list  SEQUENCE OF Return-Target-Param,
return-indicator-param    Generic-Numeric-Param,
returned-generic-value-param-list SEQUENCE OF Generic-Value-Param OPTIONAL,
content-object-ref-param-list [0] SEQUENCE OF Content-Object-Ref-Param OPTIONAL
}
Content-Object-Ref-Param ::= CHOICE {
content-object-ref        Mh-Reference,
content-object-ref-macro  [3] Content-Object-Ref-Macro
}
Content-Object-Ref-Macro ::= SEQUENCE {
macro-def-id              Macro-Def-ID,
content-object-ref        Mh-Reference OPTIONAL
}
Set-Alias ::= SEQUENCE {
target-param-list         SEQUENCE OF Target-Param,
alias-spec-param-list     SEQUENCE OF Alias-Spec-Param
}
Alias-Spec-Param ::= CHOICE {
alias-spec                Alias-Spec,
alias-spec-macro          [0] Alias-Spec-Macro
}
Alias-Spec-Macro ::= SEQUENCE {
macro-def-id              Macro-Def-ID,
alias-spec                Alias-Spec OPTIONAL
}
Alias-Spec ::= SEQUENCE {
alias-list                SEQUENCE OF GraphicString,
update-command            Update-Command
}
Catalogued-Elementary-Action ::= SEQUENCE {
target-param-list         SEQUENCE OF Target-Param,
catalogued-extended-elementary-action-param Cat-Ext-Elementary-Action-Param,
elementary-action-param-list SEQUENCE OF Generic-Value-Param OPTIONAL
}
Cat-Ext-Elementary-Action-Param ::= CHOICE {
cat-ext-elementary-action Catalogued-Extended-Elementary-Action,
cat-ext-elementary-action-macro [0] Cat-Ext-Elementary-Action-Macro
}

```

```

Cat-Ext-Elementary-Action-Macro ::= SEQUENCE {
macro-def-id          Macro-Def-ID,
catalogued-extended-elementary-action  Catalogued-Extended-Elementary-Action OPTIONAL
}
Set-Catalogued-Attribute ::= SEQUENCE {
target-param-list    SEQUENCE OF Target-Param,
cat-ext-attribute-param  Cat-Ext-Attribute-Param,
ext-attribute-value-param  Generic-Value-Param,
transition-duration-param  Generic-Integer-Param OPTIONAL
}
Set-Data ::= SEQUENCE {
content-target-param-list  SEQUENCE OF Mh-Target-Param,
substitution-indicator-param  Substitution-Indicator-Param,
data-element-param-list    SEQUENCE OF Data-Element-Param OPTIONAL
}
Substitution-Indicator-Param ::= CHOICE {
substitution-indicator    Substitution-Indicator,
substitution-indicator-macro  Substitution-Indicator-Macro
}
Substitution-Indicator-Macro ::= SEQUENCE {
macro-def-id          Macro-Def-ID,
substitution-indicator  Substitution-Indicator OPTIONAL
}
Substitution-Indicator ::= ENUMERATED {
substitution (1),
no-substitution (2)
}
Data-Element-Param ::= CHOICE {
data-element          Data-Element,
data-element-macro    [0] Data-Element-Macro
}
Data-Element-Macro ::= SEQUENCE {
macro-def-id          Macro-Def-ID,
data-element          Data-Element OPTIONAL
}
Data-Element ::= SEQUENCE {
process-indicator      Process-Indicator,
generic-list-elt-id    SEQUENCE OF INTEGER OPTIONAL,
generic-value          [0] Generic-Value
}
Process-Indicator ::= ENUMERATED {
process (1),
no-process (2)
}
Add ::= SEQUENCE {
content-target-param-list  SEQUENCE OF Mh-Target-Param,
generic-list-elt-id-param  Generic-List-Elt-ID-Param OPTIONAL,
generic-value-param        [1] Generic-Value-Param OPTIONAL
}
Substract ::= SEQUENCE {
content-target-param-list  SEQUENCE OF Mh-Target-Param,
generic-list-elt-id-param  Generic-List-Elt-ID-Param OPTIONAL,
generic-value-param        [1] Generic-Value-Param OPTIONAL
}
Copy ::= SEQUENCE {
content-target-param      Mh-Target-Param,
destination-param-list    SEQUENCE OF Mh-Target-Param
}
Run ::= SEQUENCE {
rt-target-param-list      SEQUENCE OF Rt-Target-Param,
number-of-performances-param  [0] Number-Of-Performances-Param OPTIONAL
}
Number-Of-Performances-Param ::= CHOICE {
number-of-performances    Number-Of-Performances,
number-of-performances-macro  [1] Number-Of-Performances-Macro
}

```

```

Number-Of-Performances-Macro ::= SEQUENCE {
macro-def-id          Macro-Def-ID,
number-of-performances [0] Number-Of-Performances OPTIONAL
}
Number-Of-Performances ::= CHOICE {
generic-integer       Generic-Integer,
infinite              [0] INTEGER { infinite (1) }
}
Set-Parameters ::= SEQUENCE {
rt-script-target-param-list SEQUENCE OF Rt-Script-Target-Param,
passing-param-list       SEQUENCE OF Passing-Param OPTIONAL
}
Passing-Param ::= CHOICE {
passing                Passing,
passing-macro          Passing-Macro
}
Passing-Macro ::= SEQUENCE {
macro-def-id          Macro-Def-ID,
passing               Passing OPTIONAL
}
Passing ::= CHOICE {
generic-value         [1] Generic-Value,
content-object-ref    [0] Mh-Reference
}
Plug ::= SEQUENCE {
socket-target-param-list SEQUENCE OF Socket-Target-Param,
plug-in-param          Plug-In-Param
}
Plug-In-Param ::= CHOICE {
plug-in                Plug-In,
plug-in-macro          [45] Plug-In-Macro
}
Plug-In-Macro ::= SEQUENCE {
macro-def-id          Macro-Def-ID,
plug-in               Plug-In OPTIONAL
}
Plug-In ::= CHOICE {
rt-component-reference Rt-Component-Reference,
component-object-reference Mh-Reference,
label                  [3] GraphicString,
evaluated-reference    [4] Evaluated-Reference
}
Set-RPS-Assignment ::= SEQUENCE {
rt-component-target-param-list SEQUENCE OF Rt-Component-Target-Param,
rps-assignment-param          RPS-Assignment-Param
}
RPS-Assignment-Param ::= CHOICE {
rps-assignment          RPS-Assignment,
rps-assignment-macro    RPS-Assignment-Macro
}
RPS-Assignment-Macro ::= SEQUENCE {
macro-def-id          Macro-Def-ID,
rps-assignment        RPS-Assignment OPTIONAL
}
Set-Perceptability ::= SEQUENCE {
rt-component-target-param-list SEQUENCE OF Rt-Component-Target-Param,
perceptability-param          Generic-Ratio-Param,
transition-duration-param      Generic-Integer-Param OPTIONAL
}
Set-Presentation-Priority ::= SEQUENCE {
rt-component-target-param-list SEQUENCE OF Rt-Component-Target-Param,
presentation-priority-param      Presentation-Priority-Param,
transition-duration-param        Generic-Integer-Param OPTIONAL
}
Presentation-Priority-Param ::= CHOICE {
presentation-priority          Presentation-Priority,
presentation-priority-macro    Presentation-Priority-Macro
}

```

```

Presentation-Priority-Macro ::= SEQUENCE {
macro-def-id          Macro-Def-ID,
presentation-priority Presentation-Priority OPTIONAL
}
Set-OVD ::= SEQUENCE {
rt-component-target-param-list SEQUENCE OF Rt-Component-Target-Param,
initial-point-spec-param      Initial-Point-Spec-Param,
terminal-point-spec-param     Terminal-Point-Spec-Param
}
Set-CTP ::= SEQUENCE {
rt-component-target-param-list SEQUENCE OF Rt-Component-Target-Param,
current-point-spec-param      Current-Point-Spec-Param
}
Set-Temporal-Termination ::= SEQUENCE {
rt-target-param-list          SEQUENCE OF Rt-Target-Param,
temporal-termination-param    Temporal-Termination-Param
}
Temporal-Termination-Param ::= CHOICE {
temporal-termination          Temporal-Termination,
temporal-termination-macro    Temporal-Termination-Macro
}
Temporal-Termination-Macro ::= SEQUENCE {
macro-def-id          Macro-Def-ID,
temporal-termination Temporal-Termination OPTIONAL
}
Set-PVD-Position ::= SEQUENCE {
socket-target-param-list SEQUENCE OF Socket-Target-Param,
temporal-position-param Point-Spec-Param
}
Set-GTF ::= SEQUENCE {
rt-component-target-param-list SEQUENCE OF Rt-Component-Target-Param,
gtf-param                      GF-Param
}
Set-Timestones ::= SEQUENCE {
rt-component-target-param-list SEQUENCE OF Rt-Component-Target-Param,
timestone-spec-param-list      SEQUENCE OF Timestone-Spec-Param
}
Timestone-Spec-Param ::= CHOICE {
timestone-spec              Timestone-Spec,
timestone-spec-macro       [0] Timestone-Spec-Macro
}
Timestone-Spec-Macro ::= SEQUENCE {
macro-def-id          Macro-Def-ID,
timestone-spec        Timestone-Spec OPTIONAL
}
Timestone-Spec ::= SEQUENCE {
timestone-list          SEQUENCE OF Timestone,
update-command          Update-Command
}
Timestone ::= SEQUENCE {
timestone-id            Generic-Integer,
timestone-position      Point-Spec,
number-of-repetitions   Number-Of-Repetitions
}
Number-Of-Repetitions ::= CHOICE {
generic-integer          Generic-Integer,
infinite                 [0] INTEGER { infinite (1) }
}
Set-Aspect-Ratio ::= SEQUENCE {
rt-component-target-param-list SEQUENCE OF Rt-Component-Target-Param,
aspect-ratio-param            Aspect-Ratio-Param
}
Aspect-Ratio-Param ::= CHOICE {
aspect-ratio              Aspect-Ratio,
aspect-ratio-macro        Aspect-Ratio-Macro
}

```

```

Aspect-Ratio-Macro ::= SEQUENCE {
macro-def-id          Macro-Def-ID,
aspect-ratio          Aspect-Ratio OPTIONAL
}
Set-Resizing-Strategy ::= SEQUENCE {
rt-composite-target-param-list  SEQUENCE OF Rt-Composite-Target-Param,
resizing-strategy-param         Resizing-Strategy-Param
}
Resizing-Strategy-Param ::= CHOICE {
resizing-strategy          Resizing-Strategy,
resizing-strategy-macro    Resizing-Strategy-Macro
}
Resizing-Strategy-Macro ::= SEQUENCE {
macro-def-id          Macro-Def-ID,
resizing-strategy      Resizing-Strategy OPTIONAL
}
Set-OVS-Proj-Strategy ::= SEQUENCE {
rt-component-target-param-list  SEQUENCE OF Rt-Component-Target-Param,
ovs-proj-strategy-param         OVS-Proj-Strategy-Param
}
OVS-Proj-Strategy-Param ::= CHOICE {
ovs-proj-strategy          OVS-Proj-Strategy,
ovs-proj-strategy-macro    OVS-Proj-Strategy-Macro
}
OVS-Proj-Strategy-Macro ::= SEQUENCE {
macro-def-id          Macro-Def-ID,
ovs-proj-strategy      OVS-Proj-Strategy OPTIONAL
}
Set-OVS ::= SEQUENCE {
rt-component-target-param-list  SEQUENCE OF Rt-Component-Target-Param,
size-spec-param                Size-Spec-Param,
transition-duration-param       Generic-Integer-Param OPTIONAL
}
Set-OAP ::= SEQUENCE {
rt-component-target-param-list  SEQUENCE OF Rt-Component-Target-Param,
oap-param                      Spatial-Position-Spec-Param
}
Set-OVS-Position ::= SEQUENCE {
rt-component-target-param-list  SEQUENCE OF Rt-Component-Target-Param,
ovs-position-param             Spatial-Position-Spec-Param,
transition-duration-param       Generic-Integer-Param OPTIONAL
}
Set-PAP ::= SEQUENCE {
rt-component-target-param-list  SEQUENCE OF Rt-Component-Target-Param,
pap-param                      Spatial-Position-Spec-Param
}
Set-PVS-Position ::= SEQUENCE {
rt-component-target-param-list  SEQUENCE OF Rt-Component-Target-Param,
pvs-position-param             Spatial-Position-Spec-Param,
transition-duration-param       Generic-Integer-Param OPTIONAL
}
Set-GSF ::= SEQUENCE {
rt-component-target-param-list  SEQUENCE OF Rt-Component-Target-Param,
gsf-param                      GF-Param,
transition-duration-param       Generic-Integer-Param OPTIONAL
}
Set-User-Spatial-Control ::= SEQUENCE {
rt-component-target-param-list  SEQUENCE OF Rt-Component-Target-Param,
spatial-control-param-list      SEQUENCE OF Spatial-Control-Param,
user-spatial-control-param     User-Spatial-Control-Param
}
User-Spatial-Control-Param ::= CHOICE {
user-spatial-control          User-Spatial-Control,
user-spatial-control-macro    User-Spatial-Control-Macro
}
User-Spatial-Control-Macro ::= SEQUENCE {
macro-def-id          Macro-Def-ID,
user-spatial-control  User-Spatial-Control OPTIONAL
}

```

```

Set-CV ::= SEQUENCE {
rt-content-target-param-list SEQUENCE OF Rt-Content-Target-Param,
cv-param Current-Point-Spec-Param,
transition-duration-param Generic-Integer-Param OPTIONAL
}
Set-GVF ::= SEQUENCE {
gvf-target GVF-Target,
gvf-param GF-Param,
transition-duration-param Generic-Integer-Param OPTIONAL
}
GVF-Target ::= CHOICE {
rt-composite-targets-param SEQUENCE OF Rt-Composite-Target-Param,
channel-targets-param [0] SEQUENCE OF Channel-Target-Param
}
Set-Stream-Choice ::= SEQUENCE {
rt-mux-target-param-list SEQUENCE OF Rt-Mux-Target-Param,
stream-spec-param-list SEQUENCE OF Stream-Spec-Param OPTIONAL
}
Stream-Spec-Param ::= CHOICE {
stream-spec Stream-Spec,
stream-spec-macro [0] Stream-Spec-Macro
}
Stream-Spec-Macro ::= SEQUENCE {
macro-def-id Macro-Def-ID,
stream-spec Stream-Spec OPTIONAL
}
Stream-Spec ::= SEQUENCE {
stream-id-reference-list SEQUENCE OF SEQUENCE OF INTEGER OPTIONAL,
update-command Update-Command
}
Interaction-Status-Param ::= CHOICE {
interaction-status Interaction-Status,
interaction-status-macro Interaction-Status-Macro
}
Interaction-Status-Macro ::= SEQUENCE {
macro-def-id Macro-Def-ID,
interaction-status Interaction-Status OPTIONAL
}
Min-Interact-Required-Param ::= CHOICE {
min-interact-required INTEGER,
min-interact-required-macro Min-Interact-Required-Macro
}
Min-Interact-Required-Macro ::= SEQUENCE {
macro-def-id Macro-Def-ID,
min-interact-required INTEGER OPTIONAL
}
Max-Interact-Required-Param ::= CHOICE {
max-interact-required INTEGER,
max-interact-required-macro Max-Interact-Required-Macro
}
Max-Interact-Required-Macro ::= SEQUENCE {
macro-def-id Macro-Def-ID,
max-interact-required INTEGER OPTIONAL
}
Set-Interaction-Ability ::= SEQUENCE {
rt-component-target-param-list SEQUENCE OF Rt-Component-Target-Param,
interaction-type-param Interaction-Type-Param,
min-interact-required-param Min-Interact-Required-Param,
max-interact-required-param Max-Interact-Required-Param
}
Set-Interaction-Status ::= SEQUENCE {
rt-component-target-param-list SEQUENCE OF Rt-Component-Target-Param,
interaction-type-param Interaction-Type-Param,
interaction-status-param Interaction-Status-Param
}

```

```

Set-Style ::= SEQUENCE      {
rt-component-target-param-list  SEQUENCE OF Rt-Component-Target-Param,
catalogued-style-param          Catalogued-Style-Param,
additional-information-param    Generic-Value-Param OPTIONAL
}
Catalogued-Style-Param ::= CHOICE  {
catalogued-style                Catalogued-Style,
catalogued-style-macro          [0] Catalogued-Style-Macro
}
Catalogued-Style-Macro ::= SEQUENCE {
macro-def-id                    Macro-Def-ID,
catalogued-style                Catalogued-Style OPTIONAL
}
Attach-Anchor ::= SEQUENCE      {
rt-content-target-param-list    SEQUENCE OF Rt-Content-Target-Param,
anchor-spec-param-list         SEQUENCE OF Anchor-Spec-Param
}
Anchor-Spec-Param ::= CHOICE    {
anchor-spec                     Anchor-Spec,
anchor-spec-macro               [0] Anchor-Spec-Macro
}
Anchor-Spec-Macro ::= SEQUENCE  {
macro-def-id                    Macro-Def-ID,
anchor-spec                     Anchor-Spec OPTIONAL
}
Anchor-Spec ::= SEQUENCE      {
anchor-list                     SEQUENCE OF Anchor,
update-command                 Update-Command
}
Anchor ::= CHOICE              {
rt-content-reference            Rt-Content-Reference,
evaluated-reference            Evaluated-Reference
}
Set-Channel-Perceptability ::= SEQUENCE {
channel-target-param-list      SEQUENCE OF Channel-Target-Param,
channel-perceptability-param  Channel-Perceptability-Param
}
Channel-Perceptability-Param ::= CHOICE {
channel-perceptability         Channel-Perceptability,
channel-perceptability-macro   Channel-Perceptability-Macro
}
Channel-Perceptability-Macro ::= SEQUENCE {
macro-def-id                    Macro-Def-ID,
channel-perceptability          Channel-Perceptability OPTIONAL
}
Set-CPS ::= SEQUENCE          {
channel-target-param-list      SEQUENCE OF Channel-Target-Param,
cps-initialisation-param      CPS-Initialisation-Param OPTIONAL
}
CPS-Initialisation-Param ::= CHOICE {
cps-initialisation             CPS-Initialisation,
cps-initialisation-macro       [0] CPS-Initialisation-Macro
}
CPS-Initialisation-Macro ::= SEQUENCE {
macro-def-id                    Macro-Def-ID,
cps-initialisation              CPS-Initialisation OPTIONAL
}
CPS-Initialisation ::= SEQUENCE {
cps-duration                   CPS-Duration OPTIONAL,
cps-size                       [2] CPS-Size OPTIONAL
}
CPS-Duration ::= CHOICE      {
integer                        INTEGER,
infinite                       [0] INTEGER { infinite (1)},
evaluated-integer              [1] Evaluated-Integer
}

```

```

CPS-Size ::= CHOICE
size
evaluated-list
}
Set-Event ::= SEQUENCE
rt-component-channel-tg-param-list
event-param
event-data-param
}
Elementary-Action ::= CHOICE
set-event
set-cps
set-channel-perceptability
new-channel
delete-channel
attach-anchor
get-style
set-style
set-interaction-ability
set-interaction-status
set-stream-choice
set-cv
set-gvf
set-aspect-ratio
set-resizing-strategy
set-ovs
set-oap
set-ovs-position
set-pap
set-pvs-position
set-gsf
set-user-spatial-control
set-ovd
set-ctp
set-temporal-termination
set-pvd-position
set-gtf
set-timestamps
set-perceptability
set-presentation-priority
set-rps-assignment
plug
set-parameters
run
stop
new
delete
copy
set-data
add
substract
link-abort
activate
deactivate
prepare
destroy
catalogued-elementary-action
set-catalogued-attribute
set-alias
return
delay
...
}
END
{
Size,
Evaluated-List
SEQUENCE OF Rt-Component-Channel-Tg-Param,
Generic-List-Param,
Generic-Value-Param OPTIONAL
Set-Event,
[0] Set-CPS,
[1] Set-Channel-Perceptability,
[59] SEQUENCE OF Channel-Target-Param,
[2] SEQUENCE OF Channel-Target-Param,
[3] Attach-Anchor,
Rt-Component-Target-Param,
[4] Set-Style,
[5] Set-Interaction-Ability,
[6] Set-Interaction-Status,
[7] Set-Stream-Choice,
[8] Set-CV,
[9] Set-GVF,
[16] Set-Aspect-Ratio,
[17] Set-Resizing-Strategy,
[18] Set-OVS,
[19] Set-OAP,
[20] Set-OVS-Position,
[21] Set-PAP,
[22] Set-PVS-Position,
[23] Set-GSF,
[24] Set-User-Spatial-Control,
[25] Set-OVD,
[26] Set-CTP,
[27] Set-Temporal-Termination,
[28] Set-PVD-Position,
[29] Set-GTF,
[30] Set-Timestamps,
[31] Set-Perceptability,
[32] Set-Presentation-Priority,
[33] Set-RPS-Assignment,
[39] Plug,
[40] Set-Parameters,
[41] Run,
[42] SEQUENCE OF Rt-Target-Param,
[43] SEQUENCE OF Rt-Target-Param,
[44] SEQUENCE OF Rt-Target-Param,
[45] Copy,
[46] Set-Data,
[47] Add,
[48] Substract,
[49] SEQUENCE OF Mh-Target-Param,
[50] SEQUENCE OF Mh-Target-Param,
[51] SEQUENCE OF Mh-Target-Param,
[52] SEQUENCE OF Mh-Target-Param,
[53] SEQUENCE OF Mh-Target-Param,
[54] Catalogued-Elementary-Action,
[55] Set-Catalogued-Attribute,
[56] Set-Alias,
[57] Return,
[58] Delay,

```

Annex B

Examples of MHEG systems

This Recommendation does not define the structure of the MHEG engine, nor how it handles the interchanged objects. In this annex, an example of the MHEG engine is shown.

B.1 Example of an MHEG Engine

Figure B.1 shows an implementation example of the MHEG engine and the relationship between MHEG engine, application, presentation services and access services. The MHEG engine is responsible for all processes within itself and the interworking with other modules. The MHEG engine may use the facilities provided by an operating system and system management services, which are not presented in this figure.

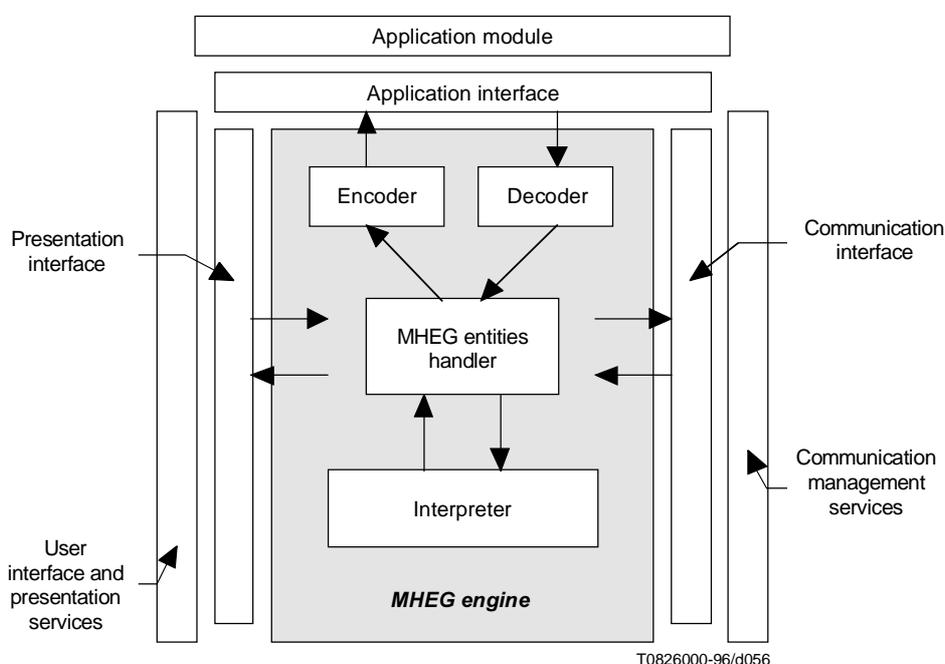


Figure B.1/T.171 – Example of MHEG engine

The MHEG engine may be composed of several modules as shown in Figure B.1.

B.1.1 MHEG decoder

This module converts MHEG objects encoded in ASN.1 to the internal format of the MHEG engine in order to handle MHEG objects.

B.1.2 MHEG encoder

This module formats internal MHEG objects into ASN.1 data so that they can be interchanged with another system. It is used when the MHEG engine sends an MHEG object to another system. This module may be optionally required in, for example, an authoring system.

B.1.3 MHEG entities handler

This module handles MHEG objects, rt-objects, channels in their internal formats, allocates entities and controls the memory management.

B.1.3.1 Reference resolution

The MHEG entities handler may also resolve MHEG entities' references. The process of resolving a reference may be as shown in Figure B.2. This figure shows an example of the decision process to resolve references of MHEG entities.

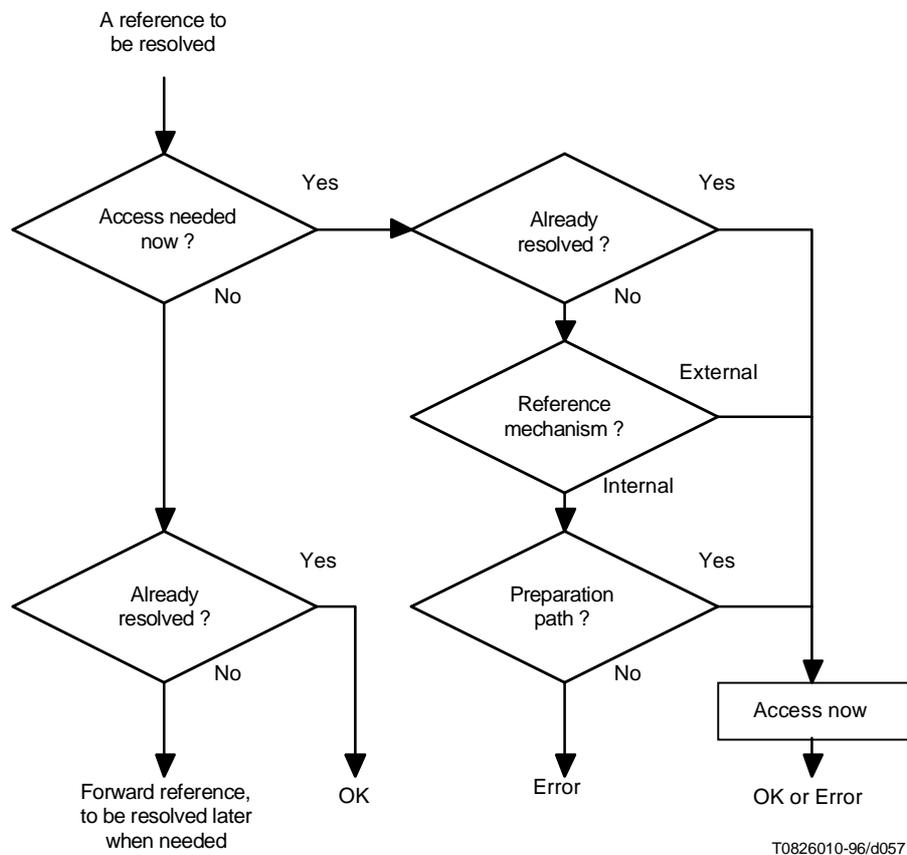


Figure B.2/T.171 – Reference resolution

When the MHEG engine needs to resolve a reference, two cases may arise as shown in Figure B.2:

- 1) The entity does not need to be accessed immediately. For example, if there is an MHEG entity as a target of an action specified in the link effect of a certain link object and a Prepare action is targeted to this link object, it is not required to prepare the MHEG object at this moment. If the entity is not available for the MHEG engine, the reference to this entity may be left for later completion. This is called “forward reference”. If the entity is accessible for the MHEG engine at the time of request, no further processing is needed.
- 2) The entity needs to be accessed immediately. For example, if an action object is referenced in a link object and a Prepare action is targeted to this link object, it is required also to prepare the referenced

action object. In this case, the access, decoding and preparation of the entity is needed. Two cases apply as follows:

- 3) If the referenced entity is not resolved yet, the MHEG engine tries to solve the reference. Two cases apply as follows:
- 4) If the entity is referenced using an external identification, it is assumed that the MHEG engine recognises and resolves the external identification. The reference is considered as resolved.
- 5) If the entity is referenced using an internal identification, the MHEG engine tries to resolve the reference. Two cases may apply:
- 6) If a preparation path is available for the MHEG engine, the reference is considered as resolved. The preparation path is not defined by this Recommendation and may be dependent on the using application, the MHEG engine and the site practice. For example, a preparation path may indicate the whereabouts of the container which includes the required objects, or it may be represented in a form of a SQL request, a server address, a file specification, etc.
- 7) If the MHEG engine is not able to resolve the reference, it is an error.
- 8) If the referenced entity is considered as accessible for the MHEG engine, the MHEG engine tries to access the object using the communication services.

If no entity is addressed while resolving the reference, no information is addressed.

Depending on the context, this may generate an error condition. For example, if the target of the Prepare action is not accessible, the target remains not available to the MHEG engine. It is the MHEG engine's responsibility to signal this condition to the using application. For example, if a link condition is described on a non-existing MHEG entity, this condition cannot be satisfied now, but it is not an error. This link is not fired at this time; however, it may be fired later.

B.1.4 MHEG interpreter

This module processes MHEG entities according to this Recommendation. The MHEG interpreter may contain various processes, however; the following processes may be identified.

B.1.4.1 Preparation process

This process prepares any MHEG entities before their processing in the MHEG engine. For example, retrieving a content data of an audiovisual sequence from a disk may require so long time that it may be efficient to start loading it before it is needed.

The preparation process may interpret the following elementary actions: Prepare action, Destroy action and Get Preparation Status action.

B.1.4.2 Creation process of rt-objects and channels

This process creates any rt-objects and channels.

The creation process may interpret the following elementary actions: New action, Delete action, Get Rt-Availability Status action, New Channel action, Delete Channel action and Get Channel-Availability Status action.

B.1.4.3 Activation process

This process activates rt-components to be passed to a presentation process and rt-scripts to be passed to a script engine.

This process may interpret the following elementary actions: Run action, Stop action and Get Running Status.

B.1.4.4 Script process

This process interprets a script data as described in a script object. Parameters may be exchanged in both ways between the MHEG engine and the script process.

This process may interpret the following elementary action: Set Parameters action.

B.1.4.5 Presentation process

This process handles the presentation of rt-components. This process may interwork with the user interface and presentation services.

This process may interpret all the elementary actions concerning the presentation. These actions may affect dynamically the rendition of rt-components and channel assignment, such as temporal behaviours, spatial behaviours, audible behaviours. And this process may interpret styles as specified by attributes.

A using application may extend the presentation facilities to its specific renditions, e.g. colour, character fonts. This specific rendition may be interchanged within MHEG objects by using extensions and descriptions in descriptor objects. It is for the using application to ensure that the presentation process available for the MHEG engine is able to present MHEG objects with these specific renditions.

B.1.4.6 Interaction process

This process interacts with the user, and interworks with the GUI tools existing on the system. This process may handle the interaction behaviour, e.g. selection and modification.

B.1.4.7 Link process

This process evaluates in parallel all the conditions on MHEG entities described in the link conditions of the activated link objects, and processes the actions described in a link effect if the corresponding link condition is satisfied.

B.1.5 User interface and presentation services

This module is in charge of the multimedia presentation to the user and of the data acquisition from the user. The presentation module may be also used by the application module to present directly using application information.

B.1.6 Application module

It is the actual using application which handles, exchanges and manages MHEG entities. The using application handles MHEG entities through the interface provided by this Recommendation (by actions applied to each MHEG entity), possibly through the extensions using a proprietary catalogue. The using application may also need to access directly to the user interface.

The following are possible commands and data facilities for the MHEG engine interface that may be used by the application module.

- start the MHEG engine;
- stop the MHEG engine;
- supply MHEG entities or references to MHEG entities to the MHEG engine;
- accept MHEG entities or references to MHEG entities from the MHEG engine;
- start the processing of a specified object;
- stop the processing of a specified object;
- facilities equivalent to those provided by elementary actions;
- facilities to access data within the MHEG engine, e.g. attributes, status values, engine statuses;
- facilities to accept data from the MHEG engine, e.g. as the result of a Return action;
- handle system management information indicating the result from the above facilities, i.e. errors.

B.1.7 Communication management module

This module handles the transmission of entities received or/and transmitted, and is in charge of the data access required for the MHEG engine. All data may be stored in local repositories (e.g. CD drive) or in remote repositories (e.g. database accessed through a network). This module is also in charge of the information transmission from the MHEG engine to the using application.

The protocol to interchange is not defined in this Recommendation. During the interchanging process, some specific identification may be added to MHEG entities.

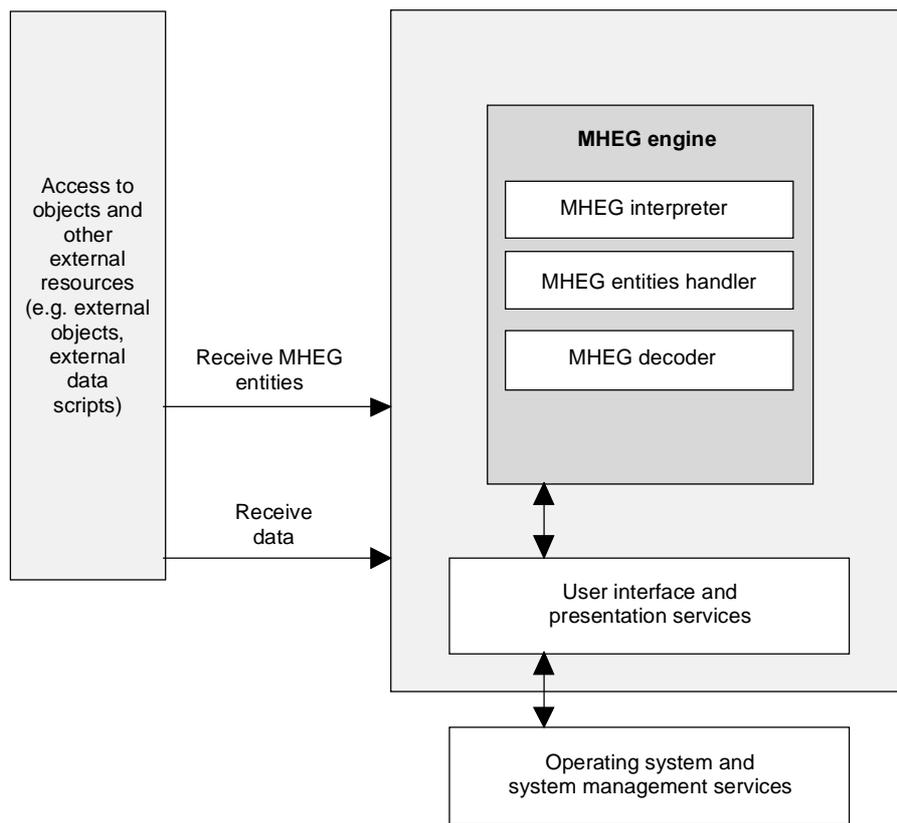
The communication module may be also used by the application module to directly access pieces of software, scripts, external data and so on.

B.2 Application examples

In this subclause, various usages and configurations of the multimedia/hypermedia systems, such as training applications, passive/interactive presentation of multimedia information, authoring applications, groupware applications, are shown.

B.2.1 Passive presentation system

Figure B.3 shows a simple multimedia presentation system that is purely passive and only to receive and present MHEG entities. The MHEG entities are sent by a certain external process, and are received through the communication module and presented without any interaction with the user.



T0826020-96/d058

Figure B.3/T.171 – Configuration example of passive presentation system

B.2.2 Enhanced passive presentation system

Figure B.4 shows a multimedia presentation system that is passive with some support of a using application for some specific operations. In this configuration, there is no interaction with the user. However, the MHEG engine may access MHEG entities under the control of the using application.

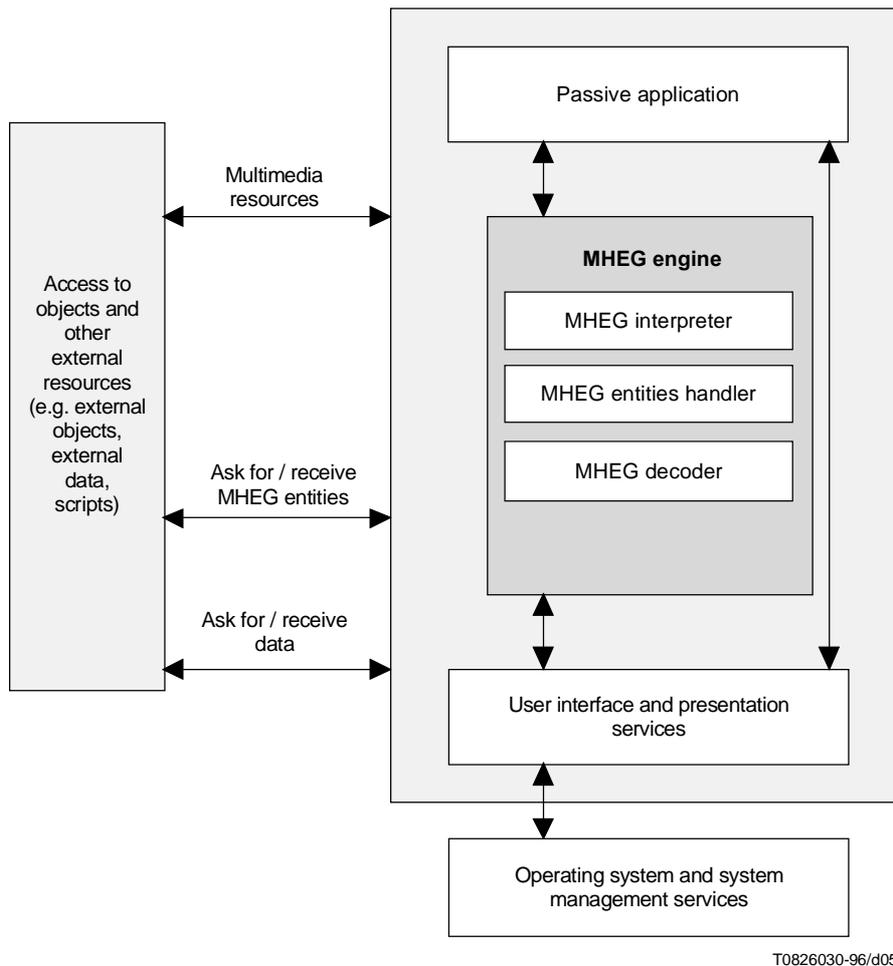


Figure B.4/T.171 – Configuration example of enhanced passive presentation system

B.2.3 Interactive presentation system

Figure B.5 shows a multimedia/hypermedia presentation system that is fully interactive with the user.

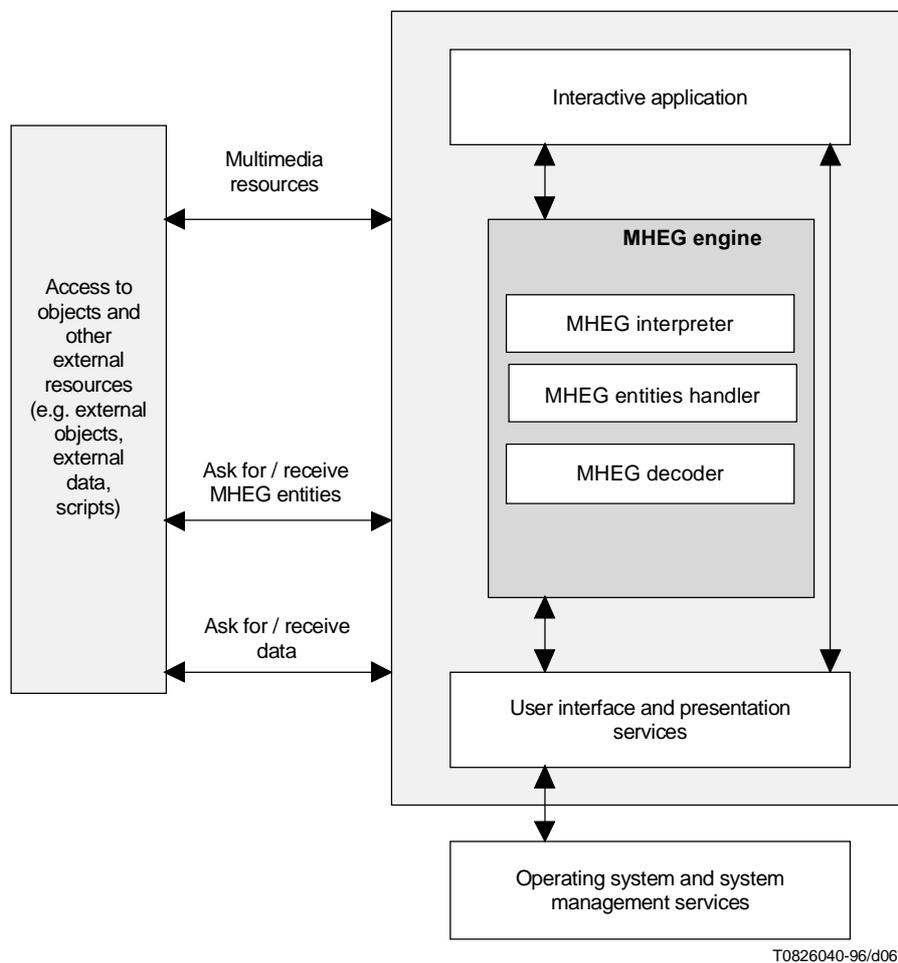


Figure B.5/T.171 – Configuration example of interactive presentation system

B.2.4 Interactive telematic system

Figure B.6 shows an interactive telematic system in the educational environment given in Recommendation F.740. In this system, a student may react to the multimedia presentation and interact to the hypermedia. Depending on the interaction, MHEG entities are exchanged between the terminal and the server. Complex interaction or data process may be done by a using application which may use a script.

The MHEG engine analyses the responses of the student, and it may request further MHEG entities, or content data required by the MHEG entities that are already received from the server. The formal responses of the student are returned to the server as some MHEG objects that may be received by the more sophisticated system used by a teacher.

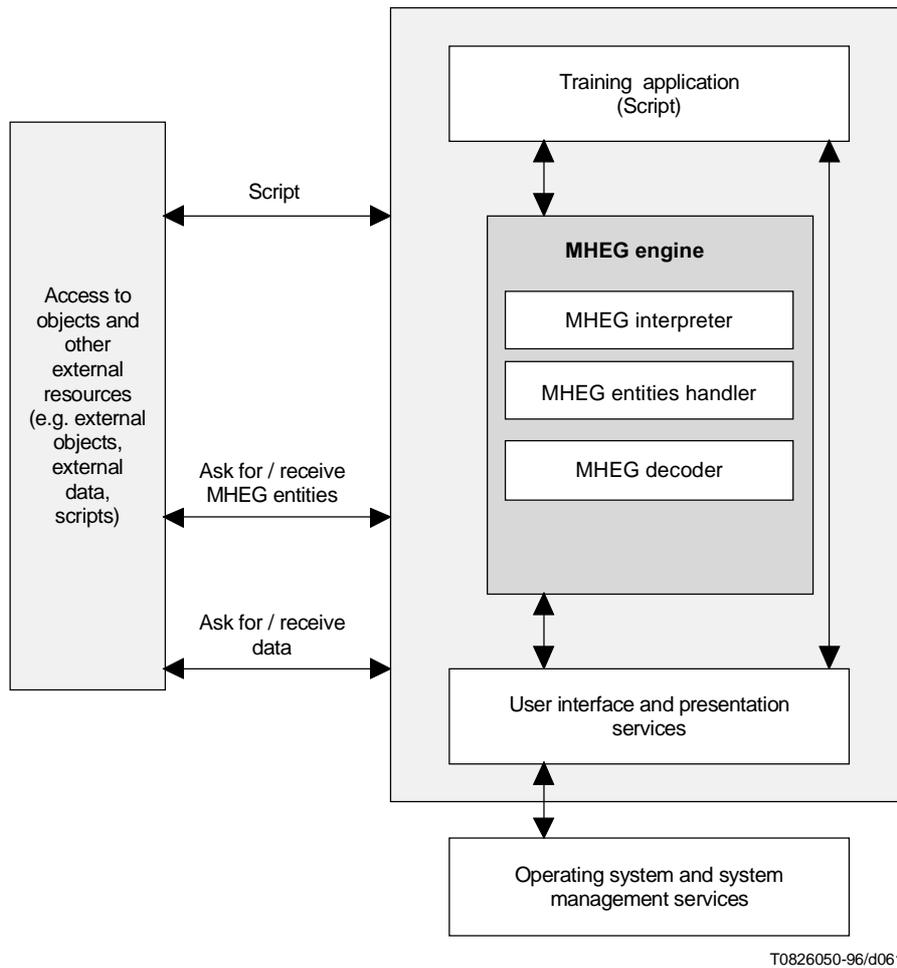
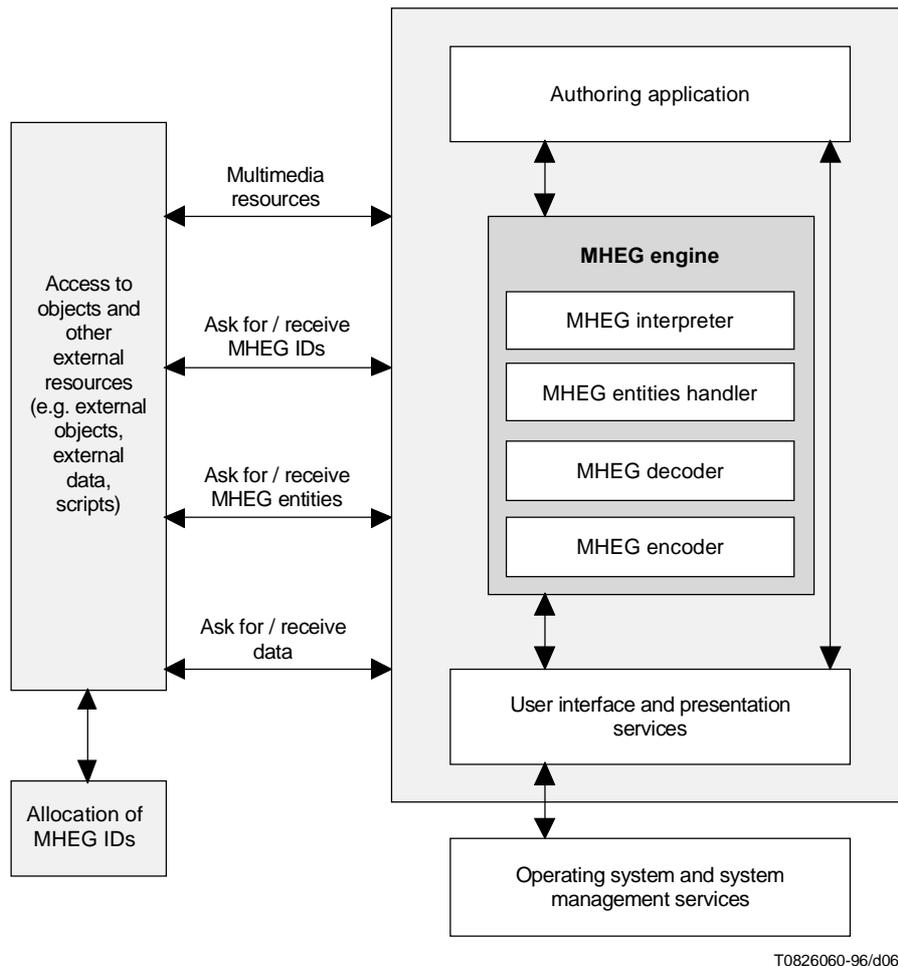


Figure B.6/T.171 – Configuration example of interactive telematic service

B.2.5 Authoring system

Figure B.7 shows an authoring system that creates MHEG objects and relies on several services of the MHEG engine for the dialogue with the author, such as the allocation of MHEG IDs to the objects and the exchange of MHEG objects.

In this system, the MHEG engine as well as the application can access a specific module which is in charge of allocating unique IDs to each MHEG object at the authoring time.



T0826060-96/d062

Figure B.7/T.171 – Configuration example for authoring system

B.2.6 Medical System

A media system retrieving some client data, storing it back to the database and interworking between doctors through a network is a typical example of a multimedia/hypermedia information system and a cooperative work system.

The following usages typically apply:

- Database access and update (see Figure B.8): Through a local area network, doctors can access and update electrical medical chart database, in which medical information (e.g. text, image, sound, video) is stored. Doctors can see it on display and update the records as if they were working on paper documents. When the doctors input the request to consult a medical chart, a user application asks for the corresponding MHEG object to the MHEG engine. The MHEG engine gets the MHEG object from the database and presents it in cooperation with the communication manager and the MHEG decoder.
- Cooperative work: Doctors and specialists in local and remote hospitals interchange medical charts as well as opinions on difficult cases. For this purpose, they need also a live video communication to establish a more friendly relationship among them.
- Text input by a doctor: Figure B.9 shows the information flow corresponding to the text input by a doctor in a local hospital, which is presented on the display of another doctor located in a remote hospital.
- Video input from a camera: Figure B.10 shows the information flow corresponding to the video input from a camera located in a remote hospital to the doctor's display in the local hospital.

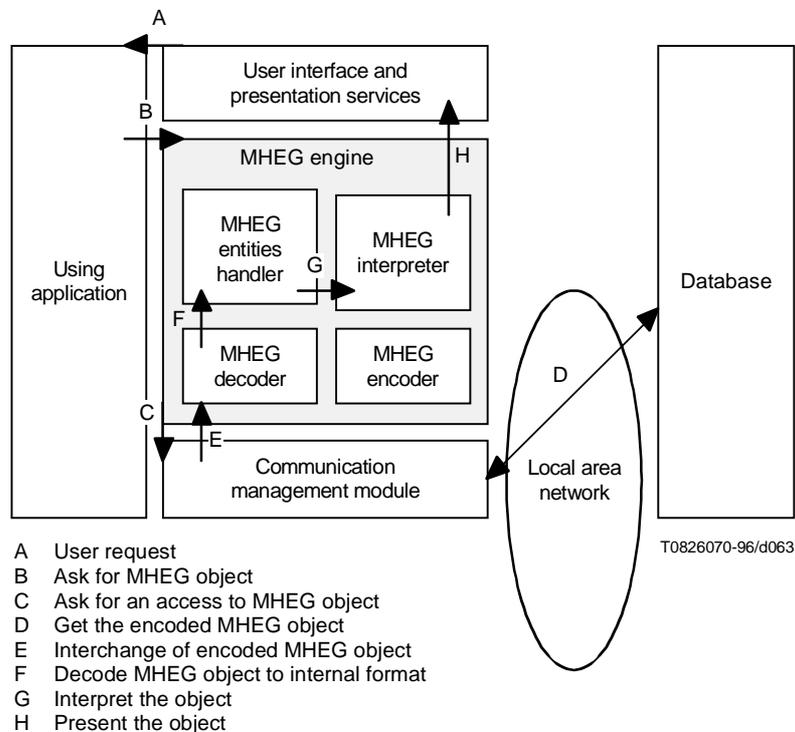
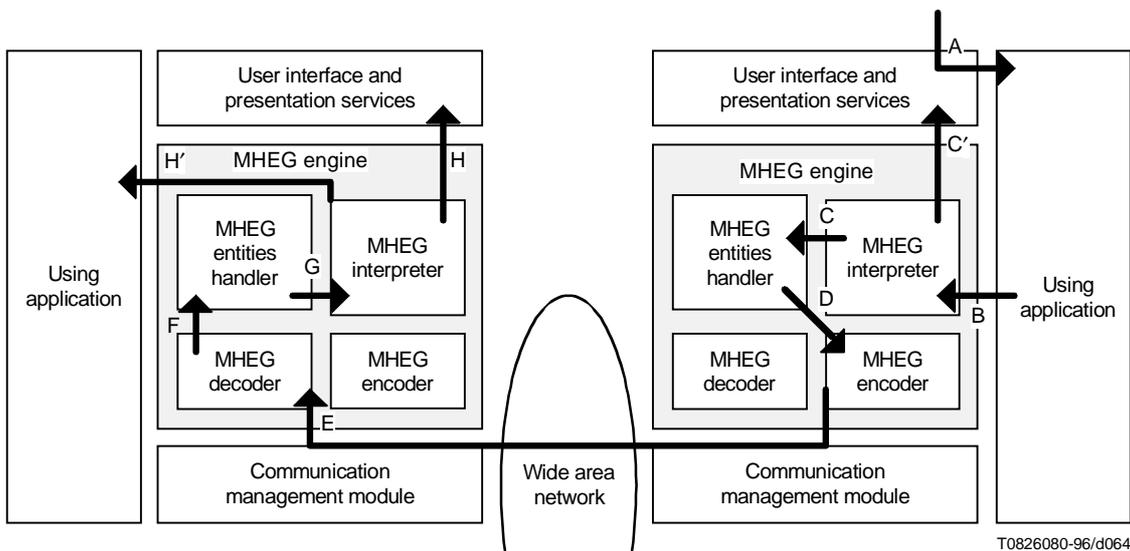
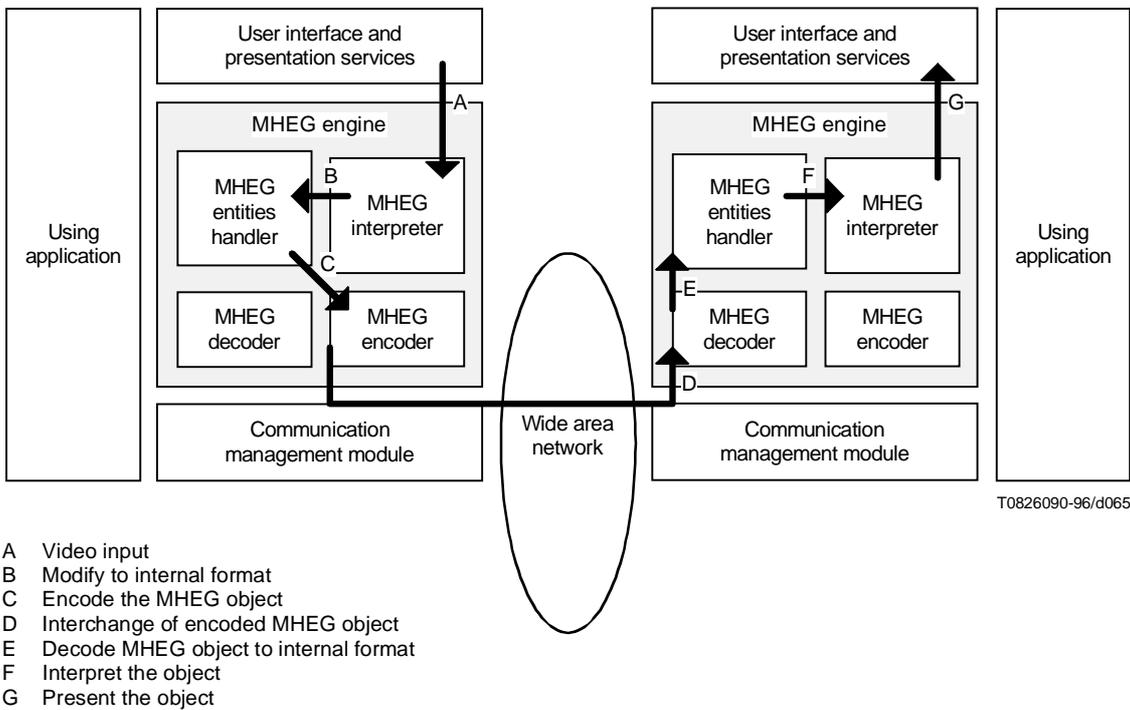


Figure B.8/T.171 – Procedure of database access and update



- A Text input
- B Ask for modifying corresponding MHEG object
- C Modify to internal format
- C' Present the updated object
- D Encode the MHEG object
- E Interchange of encoded MHEG object
- F Decode MHEG object to internal format
- G Interpret the object
- H Present the object
- H' Indicate the modification to using application

Figure B.9/T.171 – Information flow of text input



- A Video input
- B Modify to internal format
- C Encode the MHEG object
- D Interchange of encoded MHEG object
- E Decode MHEG object to internal format
- F Interpret the object
- G Present the object

Figure B.10/T.171 – Information flow of video input

Annex C

Interfaces to media Recommendations and Standards

C.1 Example of still image content object

This example shows a content object which contains an encoded still image as a content data. The detailed information of this content data is as follows and is shown in Figure C.1:

- 256 (x) × 128 (y) pixels;
- 150 dpi (x) × 150 dpi (y);
- 256 grey levels (8 bits/pixel);
- JPEG compression, baseline and hierarchical (4 levels).

The content object may contain the following information:

- 1) Content data: The corresponding encoded data.
- 2) Data classification: Still picture.
- 3) Content original perception: Not encoded. This information is to be retrieved from the hook and the content data.
- 4) Content hook information:
 - 5) Content encoding identification: ISO 10918 (JPEG) in the registered catalogue.
 - 6) Content encoding description:
 - 7) Byte 1: Luminance picture.
 - 8) Bytes 2 and 3: y axis pixel density (150 dpi).
 - 9) Bytes 4 and 5: x axis pixel density (150 dpi).
 - 10) Byte 6: 8 bits/pixel.
 - 11) Bytes 7 and 8: 256 (x) pixels.
 - 12) Bytes 9 and 10: 128 (y) pixels.
 - 13) Bytes 11 and 12: baseline and hierarchical.
 - 14) Bytes 13 and 128: 128 bytes of comments, “casa mia”.



T0826100-96/d066

Figure C.1/T.171 – Still picture

C.2 Example of audio content object

This example shows a content object which contains audio information as a content data. The detailed information of this content data is as follows:

- 1) Duration: 10 s.
- 2) Sampling frequency: 48 kHz.
- 3) Stereo mode.
- 4) MPEG audio, layer 2.
- 5) Rate: 256 kbit/s.
- 6) Length: 352 000 bytes.
- 7) Copyrighted by Deutsch Gramophon.
- 8) “A baroque music piece from Haydn”.

The content object may contain the following information:

- 1) Content data: The corresponding encoded data is included in the content data.
- 2) Data classification: Audio.
- 3) Content original perception: Not encoded. This information is to be retrieved from the hook and the content data.
- 4) Content hook information:
 - 5) Content encoding identification: ISO 11172, MPEG-Audio and Annex E/T.101 in the registered catalogue.
 - 6) Content encoding description:
 - 7) Byte 1: Rate in kbit/s: 256.
 - 8) Byte 2: JPEG-audio layer: 2.
 - 9) Byte 3: Mode stereo.
 - 10) Byte 4: Sampling frequency in kHz, 48.
 - 11) Byte 5: Duration in seconds, 10.
 - 12) Bytes 6 to 10: Length in bytes, 352 000.
 - 13) Byte 11: Copyrighted, yes.
 - 14) Bytes 12 and 128: 128 bytes of comments: “a baroque music piece from Haydn, copyrighted by Deutsch Gramophon”.

Annex D

Hypertext/Hypermedia Support

D.1 Introduction

This Recommendation provides a variety of ways to realise hypertext/hypermedia systems, i.e. hyperlinks. How to realise them completely depends on a certain implementation. The aim of this annex is to show several possible ways and provide some hints for implementers.

The essential points to realise hypertext/hypermedia systems is how to provide so-called anchor and traverse mechanism. The anchor is a specific entity which indicates that some other entities are connected to a specific place or area of a certain entity. The place or area which an anchor is associated with is called a hotspot. A hotspot may be rendered in a specific way by a presentation system in order to indicate that an anchor is attached, e.g. words with underline, words with different colour. The traverse mechanism allows you to explore another entity presentation through an anchor which links two entities.

D.2 Mechanism for Hypertext/Hypermedia

Hypertext/hypermedia mechanisms provided by this annex are categorised into 3 types by using:

- transparent object;
- anchor attached to an entity;
- event handling.

D.2.1 Transparent Object

Using a content object, a transparent object can be created and it can be used as an anchor. The typical method may be as follows:

- 1) create a content object with some data, e.g. bitmap;
- 2) create an rt-content from this model;
- 3) set its perceptability attribute to 0;
- 4) set its selectability as selectable;
- 5) attach this rt-content to an MHEG object with certain positioning corresponding to the place where an anchor should be placed;
- 6) create a link with the condition 'if this rt-content is selected' run other MHEG object, e.g. display another text.

The transparent object can be considered as an anchor. Its hotspot can be set by some positioning elementary actions. And the traverse is specified by the associated link to the transparent rt-content object.

The content data for this transparent object can be anything as far as the hook indicates what is inside. At least, the content should contain something which indicates the size of this object because this size is used as a size of the hotspot. If the perceptability is set to 0, the appearance of this object completely depends on the GUI, i.e. how to display a selectable object to a screen. It might be possible to set the perceptability not to 0 but around 0. In this case, the object can be perceived as semi-transparent so that a user can distinguish the hotspot. Positioning this rt-content to a certain MHEG object is something considerable. If the MHEG object changes its presentation, e.g. moving, resizing, change font, the attached rt-content should follow its change. As this annex defines the final form presentation, this might not be so big a problem in certain cases.

D.2.2 Anchor attached to an Entity

Using the anchor defined by this Recommendation, an anchor mechanism can be achieved in a straightforward way. The typical way may be as follows:

- 1) create an anchor;
- 2) create an rt-object from this anchor;
- 3) set its selectability as selectable;
- 4) attach this rt-object (rt-anchor) to an MHEG object with the elementary action "attach anchor";
- 5) create a link with the condition 'if this rt-content is selected' run other MHEG object, e.g. display another text.

The anchor object contains anchor information depending on the content which the anchor is attached to. It is a kind of addressing scheme and tells where the hotspot for this anchor is. The hook can be used to indicate which addressing scheme is used in this anchor. Its hotspot is to be set by using a GUI interpreting this anchor information. Its presentation relies on the GUI, too.

As there is no difference between an anchor and a content object but its presentation, any attributes for a content object can be used that may control some behaviours of an anchor.

If an MHEG object to which the anchor is attached, changes its presentation, the anchor follows the position specified by the address information, because the GUI knows the changing of the presentation and can change the anchor position appropriately.

D.2.3 Event Handling

An MHEG engine can receive some events from the system outside the engine. This event handling mechanism can be also used for realisation of hypermedia and hypertext. The typical way may be as follows:

- 1) declare necessary events within a descriptor;
- 2) create a content object corresponding to the descriptor;
- 3) create an rt-content from this model;
- 4) create a link with the condition, if some event happens, run other MHEG object, e.g. display another text.

The event attribute and associated event data attribute can control the hyperlink mechanism.

For example, in order to realise a bitmap depending on the position on which the user pointing device, e.g. a mouse, is clicked, the following is an example:

- 1) declare "mouse click" event within a descriptor;
- 2) create a content object containing a bitmap;
- 3) create an rt-content from it;
- 4) create a link which condition is "if the mouse click event happens on this rt-object, run a certain MHEG object depending on the position information stored in the event data attribute".

This is just an example. Since the event mechanism gives a general purpose method to synchronise and interact both between the MHEG engine and outer systems and between MHEG objects, beside them, many other possible ways can be realised.

D.2.4 Other Possibility

A composite object can be made with a number of content objects. Each content object holds a piece of information or a unit of information which makes complete information together as aggregated as a composite. For example, a text object can be made as a composite object with a number of contents which have one word for each. In this case, each content object may be a hotspot if every content object is arranged appropriately and is set as selectable.

This method might be difficult and complicated for a complex object.

D.2.5 Implementation and Cost

In a real application, not only the above described methods may be useful, but also some combination of them. In addition, this annex provides means for even other realisations of hyperlink functionality. It is recommended to choose an appropriate method depending on the cost of realisation and system requirement.

D.3 An example of interrelationships between MHEG and WWW browser

WWW stands for World Wide Web which is a hypermedia system and currently used widely in the Internet. It uses HyperText Markup Language (HTML), for hypertext markup, and Universal Resource Locator (URL), for identification of WWW objects. A WWW browser which is associated with a presentation system is responsible for handling any anchors that appear in the hypertext written in HTML.

WWW uses an object identification technique called URL different from that used in this Recommendation which is the public identifier defined by ISO/IEC 9070. Therefore, there are two issues to be considered:

- anchors in an HTML document which points to MHEG objects;
- anchors in an MHEG object which points to HTML documents.

In order to harmonise both issues, ISO/IEC 9070, public identifier, may be a solution.

If public identification is used in this case, it looks like:

$$\text{Public ID} = +\text{IETF}://\langle\text{Protocol}\rangle://\text{xxx.yyy.edu}/\dots/\text{zzz.mhg}$$

where zzz.mhg, MIME type associated; <Protocol>, e.g. T.176, ftp, http.

IETF should ask the registration authority of ISO/IEC 9070 for registration of its identifier. And a MIME type for this Recommendation should be registered in IETF. It is an ITU-T responsibility to register T.176 protocol.

And public identification decoder should be in the system, which is shared by both MHEG engine and WWW browsers. This decoder has the intelligence to resolve the reference. If the reference is a WWW object, then this information is passed to the WWW browser, and if the information is an MHEG object, it is passed to the MHEG engine. On the other hand, if the reference is an MHEG object, then this information is passed to the MHEG engine, and the MHEG engine passes the information for a GUI.

Annex E

Examples of spatial behaviours

E.1 Example 1

This example shows some snapshots of the presentation evolution with an rt-composite composed of two sockets in which two rt-contents are plugged in. They are displayed in the OS of their parent, i.e. their PRGS.

E.1.1 Structure of rt-composite and initial settings

Figure E.1 presents the rt-composite structure and the OSs of the rt-components.

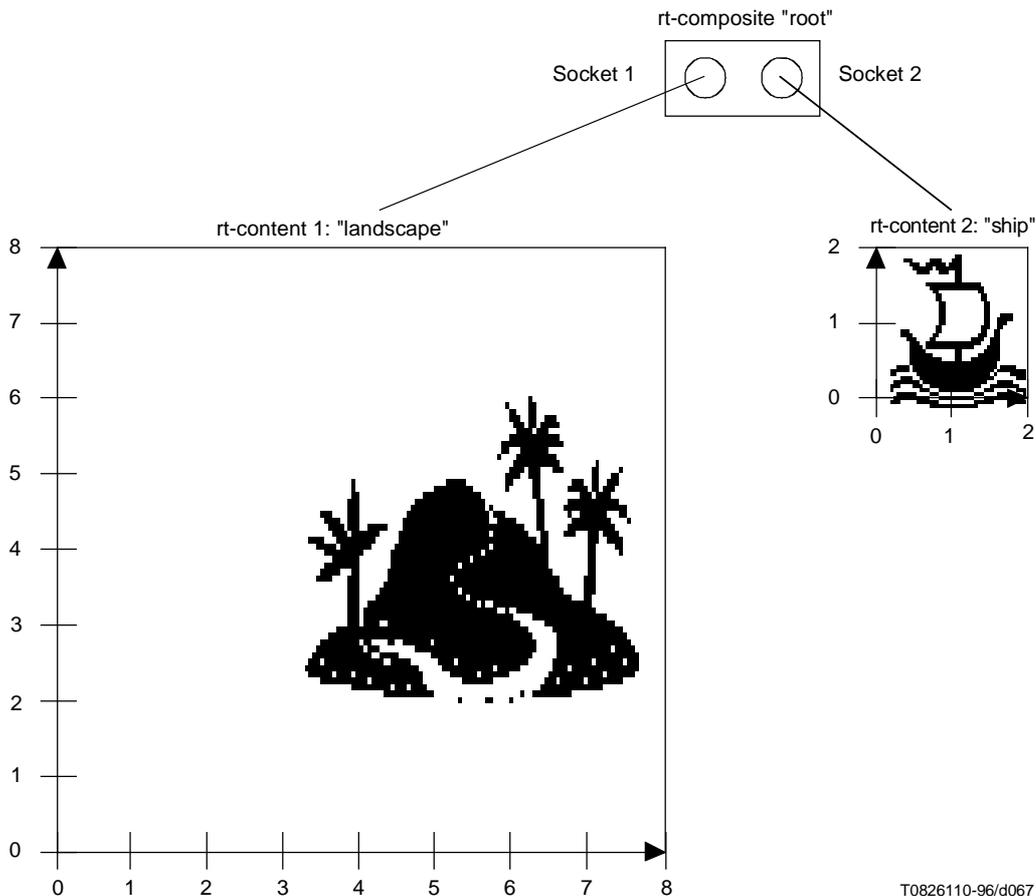


Figure E.1/T.171 – Rt-composite and rt-content OS

The OS of the rt-composite is composed of the following sockets. The initial settings of these three rt-objects are shown in Table E.1.

Table E.1/T.171 – Initial settings of Example 1

	OS	RPS assignment	Resizing strategy	OVS projection strategy	GSF	Presentation priority
rt-composite “root”	8 × 8	Channel	Fixed	Fixed	1.0	0
rt-content 1 “landscape”	8 × 8	Parent	–	Calculated	1.0	1
rt-content 2 “ship”	2 × 2	Parent	–	Calculated	1.0	0

E.1.2 Initial presentation in CGS

Figure E.2 shows the mapping of PVSs of two rt-contents to their parent OS.

No actions are applied for the rt-content 1. All default values are applied for setting OVS, OAP, OVS position, PAP and PVS position.

For rt-content 2, the following actions are applied:

- Set PAP (rt-content 2, x: 1, y: 0);
- Set PVS Position (rt-content 2, x: 1, y: 4).

No actions to defined OVS, OAP and OVS position are applied. All default values are applied for that.

The rt-content 1 is overlaid on the rt-content 2 as it has a higher presentation priority.

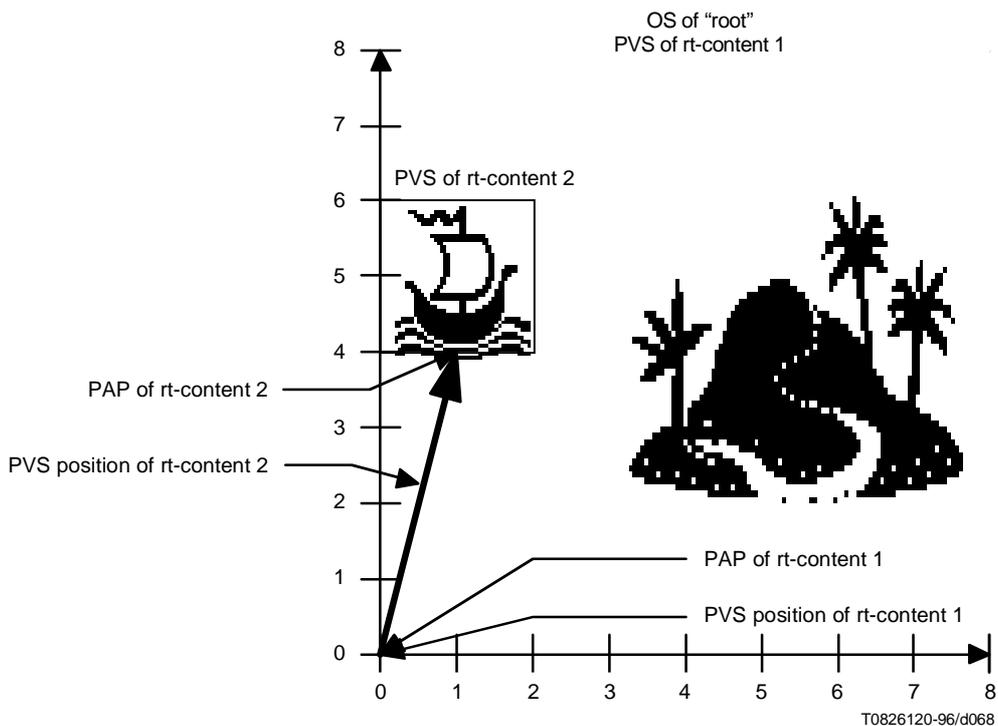


Figure E.2/T.171 – PVS setting of rt-contents to “root” OS

Figure E.3 shows the OS mapping of the rt-composite to its CPS.

Following actions are applied:

- Set OVS (root, x: 4, y: 4);
- Set OAP (root, x: 50%, y: 50%);
- Set OVS Position (root, x: 2, y: 4);
- Set PVS Position (root, x: 2, y: 4).

A Set PAP action is not used. The default value for the PAP of root is used.

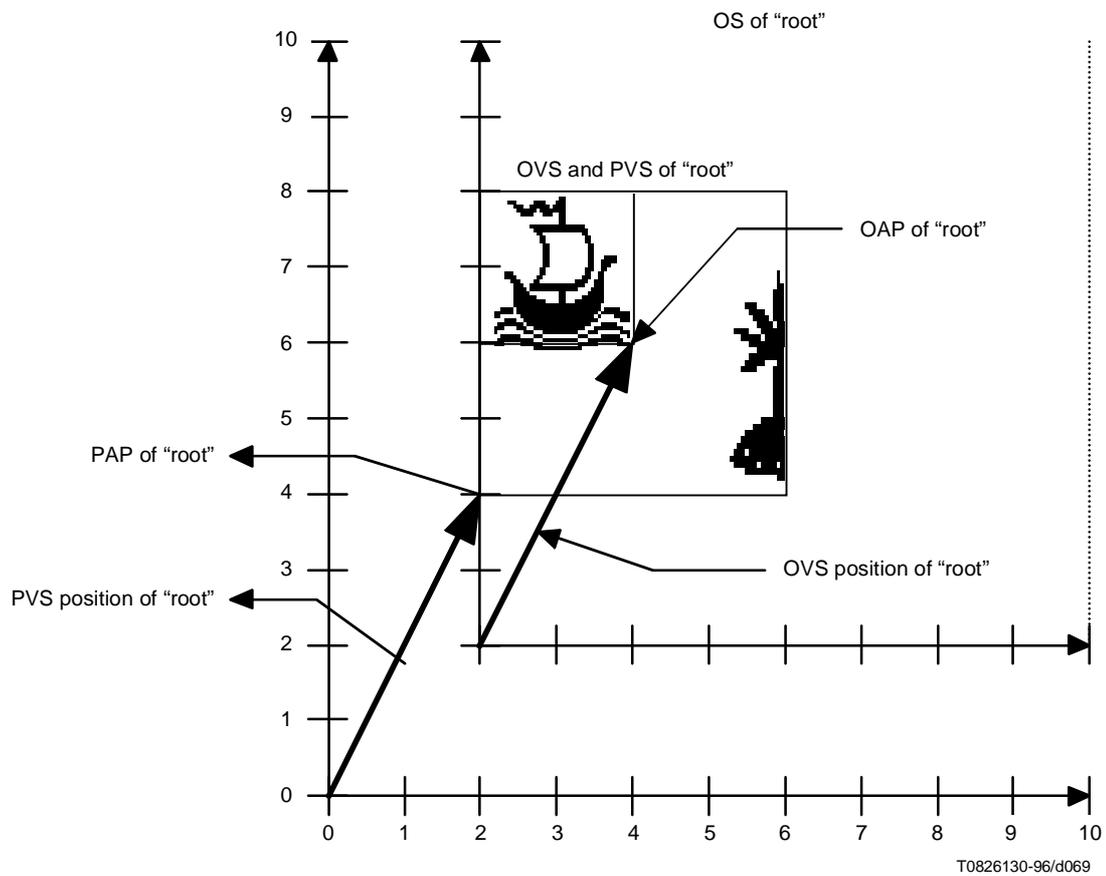


Figure E.3/T.171 – PVS setting of "root" to CPS

E.1.3 Scrolling

Figure E.4 shows the effect of the following action targeted to the rt-composite "root".

- Set OVS Position (root, x: 3, y: 4).

The effect of this action is to scroll the OVS of the rt-composite relatively to its OS. Other positions still remain as before.

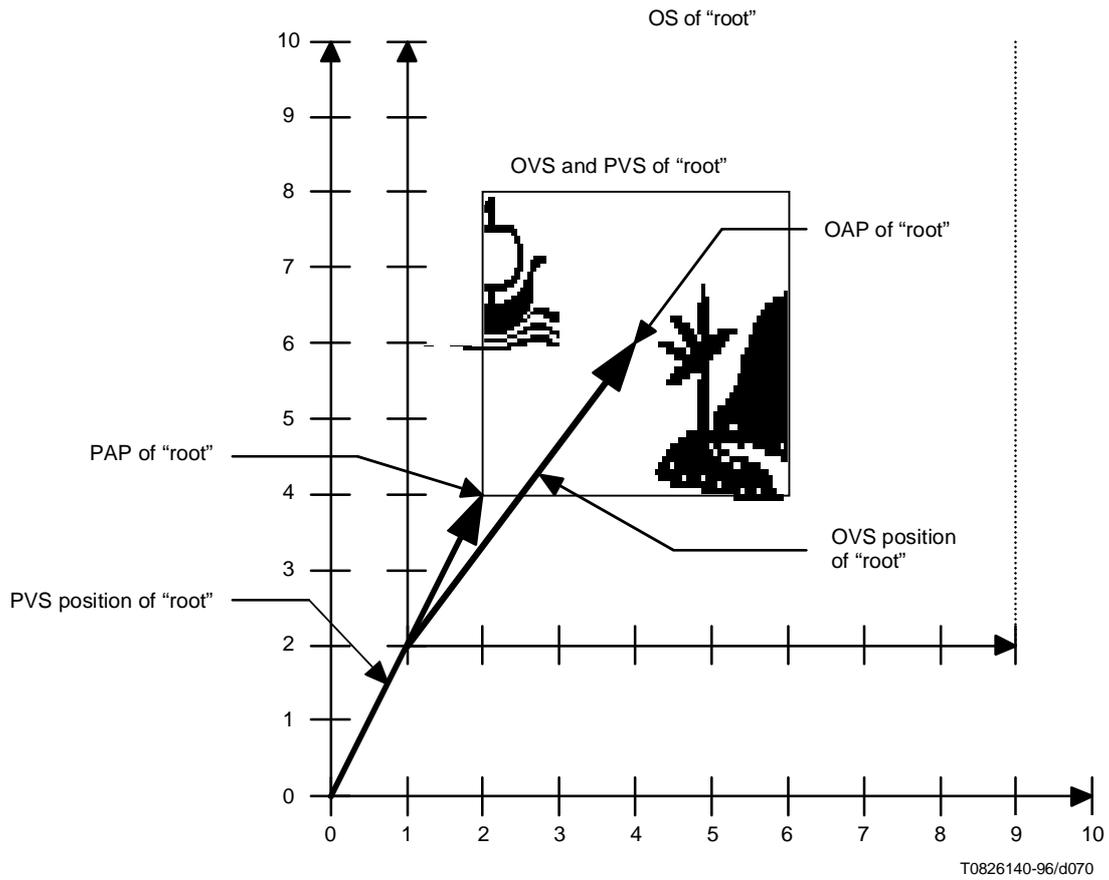


Figure E.4/T.171 – Scrolling of the rt-composite

E.1.4 Scaling

Figure E.5 shows the effect of the following actions targeted to the rt-composite “root”.

- Set OAP (root, x: 0, y: 0);
- Set OVS Position (root, x: 0, y: 0);
- Set GSF (root, x: 0.5, y: 0.5).

The PVS of the rt-composite “root” is not affected by the Set GSF action because the OVS projection strategy is set to “fixed”.

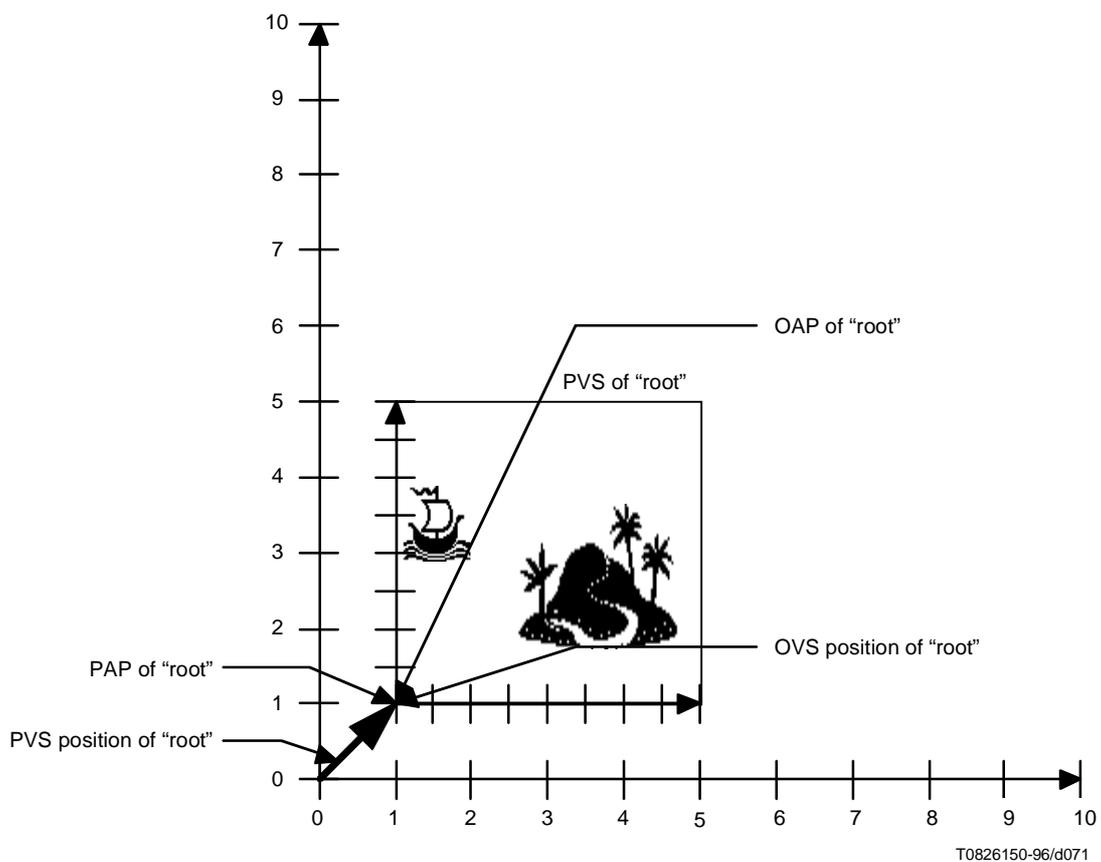


Figure E.5/T.171 – Scaling of the rt-composite

E.1.5 Socket scaling

Figure E.6 shows the effect of the following actions targeted to the rt-content 1.

- Set GSF (rt-content 1, x: 2.0, y: 2.0);
- Set PVS Position (rt-content 1, x: 3, y: 0).

As the PAP is set to x: 1 and y: 0 previously, the PAP is not affected to the GSF change of the rt-content 1. If the PAP should remain relatively as specified before independent of the GSF change, the PAP should be specified relatively.

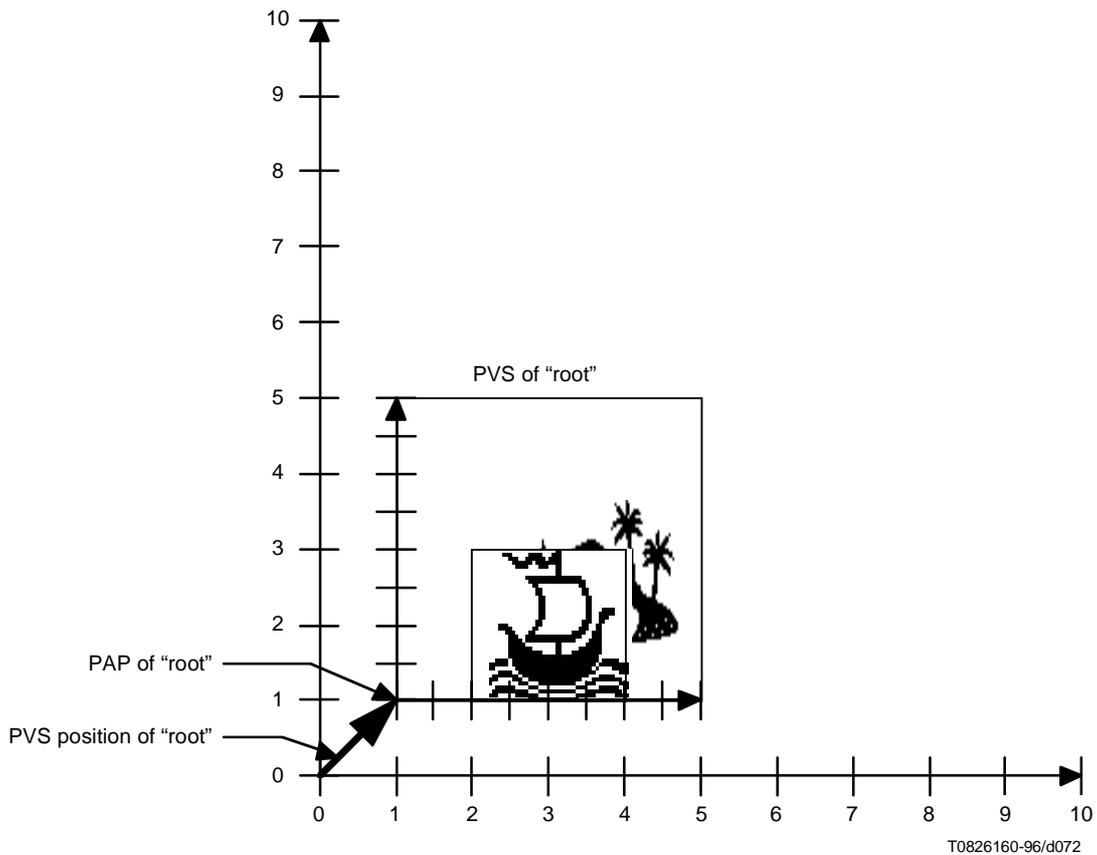


Figure E.6/T.171 – Scaling and Moving of rt-content 1

E.2 Example 2

This example shows a typical pull-down menu situation where a submenu is assigned directly to a channel. Such a feature enables the direct control of a submenu without constraining an element to be presented in the RPS of their parent.

E.2.1 Structure of rt-composite and initial settings

Figure E.7 shows the composition structure of “menu”. Rt-content “file” and “option” compose a top menu. Rt-content “open”, “close” and “new” compose a submenu of “file”.

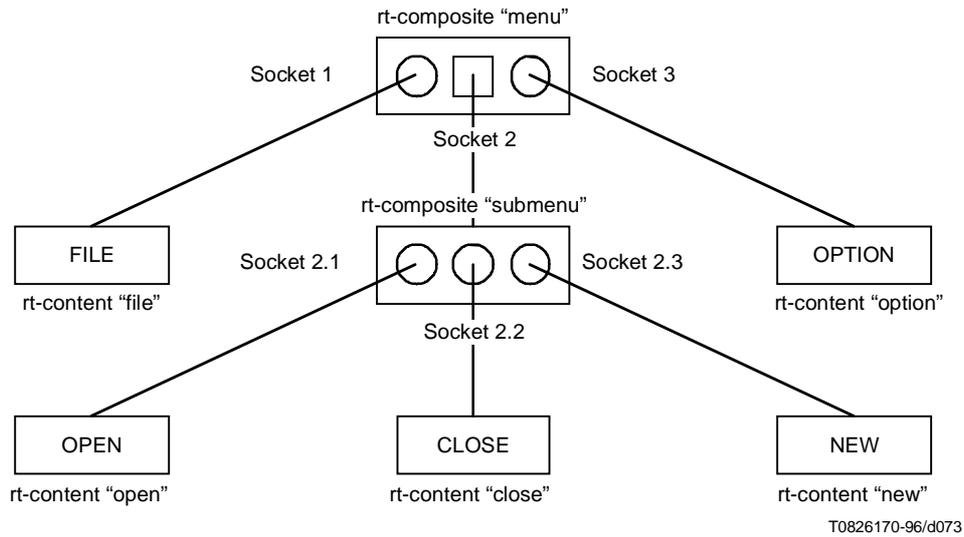


Figure E.7/T.171 – The structure of “menu”

E.2.2 Initial presentation in CGS

Figure E.8 shows the initial presentation of menu in a channel. Rt-composite “menu” and rt-composite “submenu” are directly presented in the channel.

Rt-content “file” and “option” are projected to rt-composite “menu”. The PVS of the “menu” is positioned in the channel as shown in Figure E.8. Rt-content “open”, “close” and “new” are projected to rt-composite “submenu”. The PVS of the “submenu” is positioned in the channel as shown in Figure E.8.

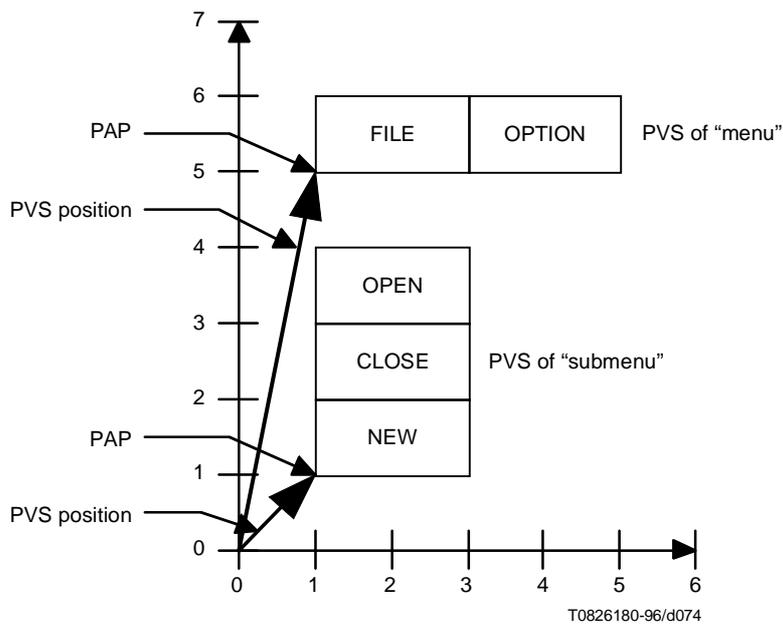


Figure E.8/T.171 – Initial presentation of menu

Table E.2/T.171 – Initial settings of Example 2

	RPS	Resizing strategy
rt-composite “menu”	CGS	Minimum
rt-content “file”	Parent	–
rt-content “option”	Parent	–
rt-composite “submenu”	CGS	Minimum
rt-composite “open”	Parent	–
rt-composite “close”	Parent	–
rt-composite “new”	Parent	–

Annex F

Summary of Object Identifiers

The object identifiers have the form {root arc1 arc2 arc3} where:

- **root** = 2, joint-iso-itu-t standard.
- **arc1** = 19, MHEG standard identification.
- **arc2** = 1, version of the standard.
- **arc3** = number of a class as defined in MH-Object class.

Table F.1/T.171 – Summary of object Identifier

ASN.1 module name	ASN.1 module identifier value	MHEG class identifier	Mnemonic	Class identified
ISOMHEG-mh	None	None	mh	Mh-object
ISOMHEG-ac	{2 19 1 1}	{2 19 1 1}	ac	Action
ISOMHEG-lk	{2 19 1 2}	{2 19 1 2}	lk	Link
ISOMHEG-mo	None	None	mo	Model
ISOMHEG-sc	{2 19 1 3}	{2 19 1 3}	sc	Script
ISOMHEG-cp	None	None	cp	Component
ISOMHEG-ct	{2 19 1 4}	{2 19 1 4}	ct	Content
ISOMHEG-mu	{2 19 1 5}	{2 19 1 5}	mu	Multiplexed Content
ISOMHEG-co	{2 19 1 6}	{2 19 1 6}	co	Composite
ISOMHEG-cr	{2 19 1 7}	{2 19 1 7}	cr	Container
ISOMHEG-de	{2 19 1 8}	{2 19 1 8}	de	Descriptor
ISOMHEG-ud	{2 19 1 9}	None	ud	Useful Definition
ISOMHEG-ea	{2 19 1 10}	None	ea	Elementary Action

Annex G

Index

Index of the clauses

Action-Class	274	Cat-Ext-Elementary-Action-Param	283
Action-Object	275		
Activation-Status	273	Channel-Availability-Status	271
Add	284		
Alias-Spec	283	Channel-Information	281
Alias-Spec-Macro	283		
Alias-Spec-Param	283	Channel-Perceptability	270
Alternative-Object	281		
Ancestor	273	Channel-Perceptability-Macro	289
Anchor	289		
Anchor-Spec	289	Channel-Perceptability-Param	289
Anchor-Spec-Macro	289		
Anchor-Spec-Param	289	Channel-Reference	261
Aspect-Ratio	272		
Aspect-Ratio-Macro	287	Channel-Target	263
Aspect-Ratio-Param	286		
Associated-Model	279	Channel-Target-Macro	263
Attach-Anchor	289		
Availability-Close-down	279	Channel-Target-Param	263
Availability-Start-up	279		
Catalogued-Content-Encoding	268	Child	273
Catalogued-Elementary-Action	283		
Catalogued-Event	268	Class-ID	258
Catalogued-Extended-Elementary-Action	268		
Catalogued-Media-Type	268	Class-Specific-Information	281
Catalogued-Script-Classification	268		
Catalogued-Script-Encoding	268	Comparison-Operation	276
Catalogued-Style	268		
Catalogued-Style-Macro	289		
Catalogued-Style-Param	289		
Cat-Content-Classification	268		
Cat-Ext-Attribute	268		
Cat-Ext-Attribute-Macro	274		
Cat-Ext-Attribute-Param	274		
Cat-Ext-Elementary-Action-Macro	284		

Comparison-Operator	276	Evaluated-Boolean	267
Comparison-Value	276		
Comparison-Value-Constant	274		
Component-Class	258		
Composite-Class	278	Evaluated-Integer	267
Composition-Element	279		
Condition	276		
Constraint-Condition	275		
Container-Class	280		
Container-Close-down	280	Evaluated-List	268
Container-Element	280		
Container-Element-Reference	259		
Container-Start-up	280		
Content	277	Evaluated-Numeric	267
Content-Class	277		
Content-Class-Information	281		
Content-Data	277		
Content-Hook	268		
Content-Object-Ref-Macro	283	Evaluated-Ratio	267
Content-Object-Ref-Param	283		
Copy	284		
CPS-Duration	289		
CPS-Initialisation	289	Evaluated-Reference	268
CPS-Initialisation-Macro	289		
CPS-Initialisation-Param	289		
CPS-Size	290		
Current-Point-Spec	270		
Current-Point-Spec-Macro	270	Evaluated-String	268
Current-Point-Spec-Param	270		
CV	271		
Data-Element	284		
Data-Element-Macro	284	Evaluated-Value	267
Data-Element-Param	284		
Data-Inclusion	277		
Data-Reference	259		
Delay	283		
Description	258	Event-Mapping	282
Descriptor-Class	281		
Elementary-Action	290		
EmptyChild	273		

Expected-Axis-Result-Param	272	Get-Interaction-Ability	265
Expected-OVD-Result	272		
Expected-OVD-Result-Macro	272		
Expected-OVD-Result-Param	273		
Expected-PVD-Result	272	Get-Interaction-Status	265
Expected-PVD-Result-Macro	272		
Expected-PVD-Result-Param	272		
External-ID	259		
Generic-Boolean	264		
Generic-Boolean-Macro	264	Get-List	267
Generic-Boolean-Param	263		
Generic-Integer	264		
Generic-Integer-Macro	264		
Generic-Integer-Param	264	Get-Max-Interact-Required	265
Generic-List	265		
Generic-List-Elt-ID-Macro	265		
Generic-List-Elt-ID-Param	265		
Generic-List-Macro	265		
Generic-List-Param	265	Get-Min-Interact-Required	265
Generic-Numeric	264		
Generic-Numeric-Macro	264		
Generic-Numeric-Param	264		
Generic-Ratio	264	Get-Number-Of-Interacted-Sockets	265
Generic-Ratio-Macro	264		
Generic-Ratio-Param	264		
Generic-Reference	265		
Generic-Reference-Macro	265	Get-OAP	266
Generic-Reference-Param	264		
Generic-String	264		
Generic-String-Macro	264		
Generic-String-Param	264		
Generic-Value	263	Get-OS	266
Generic-Value-Macro	263		
Generic-Value-Param	263		
Get-Any	267		
Get-Catalogued-Attribute	266	Get-OVD	266
Get-Data	266		
Get-GSF	265		
Get-GVF	265		
Get-Integer	266		

Get-OVS	266	Mh-Reference	259
Get-OVS-Position	266		
Get-PAP	266		
Get-POS	266		
Get-PVD	266	Mh-Target	261
Get-PVS	266		
Get-PVS-Position	265		
Get-Reference	267		
Get-Rt-Composite-Address	266		
Get-Stream-Chosen-State	265	Mh-Target-Macro	261
Get-User-Spatial-Control	265		
GF	269		
GF-Macro	269		
GF-Param	269	Mh-Target-Param	261
GSF	272		
GVF-Target	288		
Initial-Point-Spec-Macro	270		
Initial-Point-Spec-Param	270		
Interaction-Status	271	Min-Interact-Required-Macro	288
Interaction-Status-Macro	288		
Interaction-Status-Param	288		
Interaction-Type	271		
Interaction-Type-Macro	271	Min-Interact-Required-Param	288
Interaction-Type-Param	271		
Length-Spec	269		
Length-Spec-Macro	269		
Length-Spec-Param	269		
Lengths-Spec	269	Model-Class	258
Link-Class	275		
Link-Condition	275		
Link-Effect	276		
Link-Object	279	Modifiability	271
Logical-Combination	276		
Logical-Operator	276		
Macro-Def-ID	259		
Macro-Parameter-Resolution	276		
Max-Interact-Required-Macro	288	Modification-Status	271
Max-Interact-Required-Param	288		
Mheg-ID	259		
Mh-object-Class	258		

Multiplexed-Content-Class	278	Resizing-Strategy	272
Multiplexed-Stream	278		
Mux-Content-Class-Info	281		
Navigation-Command	273		
Navigation-Command-Macro	273	Resizing-Strategy-Macro	287
Navigation-Command-Param	273		
Number-Of-Performances	285		
Number-Of-Performances-Macro	285		
Number-Of-Performances-Param	284		
Number-Of-Repetitions	286	Resizing-Strategy-Param	287
Object-Information	281		
OD	273		
OPS-Initialisation	268		
OV	271	Return	283
OVS-Proj-Strategy	272		
OVS-Proj-Strategy-Macro	287		
OVS-Proj-Strategy-Param	287		
Passing	285		
Passing-Macro	285	Return-Target-Param	263
Passing-Param	285		
PCV	271		
Plug	285		
Plug-In	285	Root-Rt-Component-ID-Ref	260
Plug-In-Macro	285		
Plug-In-Param	285		
Point	269		
Point-Spec	270	Root-Rt-Component-Reference	260
Point-Spec-Macro	270		
Point-Spec-Param	270		
Points-Spec	270		
Point-Type-Param	272		
Preparation-Status	274	Root-Rt-Composite-ID-Ref	261
Presentation-Priority	273		
Presentation-Priority-Macro	286		
Presentation-Priority-Param	285		
Process-Indicator	284		
Prop-Cat-Entry-ID	268	Root-Rt-Composite-Reference	261
Ratio	264		
Reference	259		
Related-Object	281		

Root-Rt-Content-ID-Ref	260	Rt-Target-Param	262
Root-Rt-Content-Reference	260		
Root-Rt-ID-Reference	260		
Root-Rt-Mux-ID-Reference	260		
Root-Rt-Mux-Reference	260	Run	284
Root-Rt-Reference	260		
RPS-Assignment	273		
RPS-Assignment-Macro	285		
RPS-Assignment-Param	285		
Rt-Availability-Start-up	279	Running-Status	273
Rt-Availability-Status	273		
Rt-Component-Channel-Ref	261		
Rt-Component-Channel-Tg	263		
Rt-Component-Channel-Tg-Macro	263	Script-Class	277
Rt-Component-Channel-Tg-Param	263		
Rt-Component-Reference	260		
Rt-Component-Target	262		
Rt-Component-Target-Macro	262	Script-Class-Information	281
Rt-Component-Target-Param	262		
Rt-Composite-Reference	261		
Rt-Composite-Target	263		
Rt-Composite-Target-Macro	262		
Rt-Composite-Target-Param	262	Script-Data	277
Rt-Content-Reference	260		
Rt-Content-Target	262		
Rt-Content-Target-Macro	262		
Rt-Content-Target-Param	262		
Rt-Mux-Reference	261	Script-Hook	268
Rt-Mux-Target	262		
Rt-Mux-Target-Macro	262		
Rt-Mux-Target-Param	262		
Rt-Number-Reference	260	Script-Inclusion	277
Rt-Reference	260		
Rt-Script-ID-Reference	260		
Rt-Script-Reference	260		
Rt-Script-Target	262		
Rt-Script-Target-Macro	262	Selectability	271
Rt-Script-Target-Param	262		
Rt-Target	262		
Rt-Target-Macro	262		

Selection-Status	271	Socket-Reference	261
Set-Alias	283		
Set-Aspect-Ratio	286		
Set-Catalogued-Attribute	284		
Set-Channel-Perceptability	289	Socket-Tail-Complement	261
Set-CPS	289		
Set-CTP	286		
Set-CV	288		
Set-Data	284		
Set-Event	290		
Set-GSF	287	Socket-Tail-Reference	261
Set-GTF	286		
Set-GVF	288		
Set-Interaction-Ability	288		
Set-Interaction-Status	288		
Set-OAP	287	Socket-Target	263
Set-OVD	286		
Set-OVS	287		
Set-OVS-Position	287		
Set-OVS-Proj-Strategy	287		
Set-PAP	287	Socket-Target-Macro	263
Set-Parameters	285		
Set-Perceptability	285		
Set-Presentation-Priority	285		
Set-PVD-Position	286		
Set-PVS-Position	287	Socket-Target-Param	263
Set-Resizing-Strategy	287		
Set-RPS-Assignment	285		
Set-Stream-Choice	288		
Set-Style	289		
Set-Temporal-Termination	286	Spatial-Control	272
Set-Timestones	286		
Set-User-Spatial-Control	287		
Size	269		
Size-Spec	269		
Size-Spec-Macro	269	Spatial-Control-Macro	272
Size-Spec-Param	269		
Socket-ID	259		
Socket-Identification	259		
Socket-ID-Reference	261		

Spatial-Control-Param	272	Timestone-Spec-Macro	286
Spatial-Length	269		
Spatial-Position	269		
Spatial-Position-Spec	270		
Spatial-Position-Spec-Macro	270		
Spatial-Position-Spec-Param	269	Timestone-Spec-Param	286
Stream-Chosen-State	271		
Stream-Identification	271		
Stream-Identification-Macro	271		
Stream-Identification-Param	271		
Stream-Information	281		
Stream-Spec	288	Trigger-Condition	275
Stream-Spec-Macro	288		
Stream-Spec-Param	288		
Substitution-Indicator	284		
Substitution-Indicator-Macro	284		
Substitution-Indicator-Param	284		
Substract	284	Update-Command	274
Synchro-Indicator	275		
Synchro-Indicator-Macro	275		
Synchro-Indicator-Param	275		
Synchronised-Action	275		
System-Readable-Material	281	User-Spatial-Control	272
Tail-Complement	259		
Tail-Macro	259		
Tail-Param	259		
Tail-Reference	259		
Target-Macro	261		
Target-Param	261	User-Spatial-Control-Macro	287
Temporal-Termination	273		
Temporal-Termination-Macro	286		
Temporal-Termination-Param	286		
Temporal-Unit-Ref	283		
Temporal-Unit-Ref-Macro	283	User-Spatial-Control-Param	287
Temporal-Unit-Ref-Param	283		
Terminal-Point-Spec-Macro	270		
Terminal-Point-Spec-Param	270		
Termination-Status	273		
Timestone	286		
Timestone-Spec	286	Value	263

Index of the behaviours

Behaviours	Status and Attributes	Elementary Actions	Get Actions
Postpone behaviour		Delay (Temporal Unit Ref Param, ...)	
Returnability behaviour		Return (Return Target Param+, ...)	
Alias behaviour		Set Alias (Target Param+, ...)	
Extensibility behaviour	Catalogued Attribute@Cat Ext Attribute Param@Cat Ext Attribute Macro@	Catalogued Elementary Action (Target Param+, ...) Set Catalogued Attribute (Target Param+, ...)	Get Catalogued Attribute@
Mheg objects availability behaviour	Preparation Status@	Prepare (MH-Target Param+) Destroy (MH-Target Param+)	Get Preparation Status@
Link object activation behaviour	Activation Status@	Activate (Link Target Param+) Deactivate (Link Target Param+)	Get Activation Status@
Link object abort behaviour		Link Abort (Link Target Param+)	
Content class generic value storage behaviour	Data@	Set Data (Content Target Param+, ...) Add (Content Target Param+, ...) Substract (Content Target Param+, ...)	Get Data@
Content class copy behaviour		Copy (Content Target Param, ...)	
Rt-objects availability behaviour	Rt-Availability Status@	New (Rt-Target Param+) Delete (Rt-Target Param+)	Get Rt-Availability Status@
Rt-objects running behaviour	Running Status@	Run (Rt-Target Param+, ...) Stop (Rt-Target Param+)	Get Running Status@
Rt-script passing parameter behaviour		Set Parameters (Rt-Script Target Param+, ...)	
Rt-scripts termination behaviour	Termination Status@		Get Termination Status@
Sockets presentation and structural dynamism behaviour		Plug (Socket Target Param+, ...)	
Rt-composite navigation behaviour	Rt-Composite Address@Navigation Command Param@Navigation Command Macro@Navigation Command@ Child@Empty Child@Sibling@ Ancestor@		Get Rt-Composite Address@
Rt-components rps assignment behaviour	RPS Assignment@	Set RPS Assignment (Rt-Component Target Param+, ...)	Get RPS Assignment@
Rt-components perceptability behaviour	Perceptability@Presentation Priority@	Set Perceptability (Rt-Component Target Param+, ...) Set Presentation Priority (Rt-Component Target Param+, ...)	Get Perceptability Get Presentation Priority@

Index of the behaviours (continued)

Behaviours	Status and Attributes	Elementary Actions	Get Actions
Rt-components temporal behaviour	OD@POD@OVD@PVD@ Temporal Termination@PVD Position@CTP@GTF@Timestone Status@Timestone ID@Expected OVD Result Param@Expected OVD Result Macro@Expected OVD Result@Expected PVD Result Param@Expected PVD Result Macro@Expected PVD Result@	Set OVD (Rt-Component Target Param+, ...) Set CTP (Rt-Component Target Param+, ...) Set Temporal Termination (Rt-Target Param+, ...) Set PVD Position (Socket Target Param+, ...) Set GTF (Rt-Component Target Param+, ...) Set Timestones (Rt- Component Target Param+, ...)	Get OD Get POD Get OVD Get PVD Get CTP Get Temporal Termination Get PVD Position Get GTF Get Timestone Status@
Rt-components spatial behaviour	OS@POS@Aspect Ratio@Resizing Strategy@OVS@OAP@OVS Position@PVS@OVS Proj Strategy@PAP@PVS Position@GSF@Spatial Control@User Spatial Control@Expected Axis Result Param@Spatial Control Param@Spatial Control Macro@Point Type Param@	Set Aspect Ratio (Rt- Component Target Param+, ...) Set Resizing Strategy (Rt-Composite Target Param+, ...) Set OVS (Rt-Component Target Param+, ...) Set OAP (Rt-Component Target Param+, ...) Set OVS Position (Rt- Component Target Param+, ...) Set PAP (Rt-Component Target Param+, ...) Set PVS Position (Rt- Component Target Param+, ...) Set GSF (Rt-Component Target Param+, ...) Set User Spatial Control (Rt-Component Target Param+, ...)	Get OS Get POS Get Aspect Ratio Get Resizing Strategy Get OVS Proj Strategy Get OVS Get OAP Get OVS Position Get PVS Get PAP Get PVS Position Get GSF Get User Spatial Control@
Rt-components audible behaviour	OV@CV@PCV@GVF@	Set CV (Rt-Content Target Param+, ...) Set GVF (GVF Target, ...)	Get OV Get CV Get PCV Get GVF@
Rt-mux stream choice behaviour	Stream Choice@Stream Chosen State@Stream Identification Param@Stream Identification Macro@Stream Identification@	Set Stream Choice (Rt- Mux Target Param+, ...)	Get Stream Choice Get Stream Chosen State@
Interaction behaviour	Interaction Type Param@Interaction Type Macro@Interaction Type@Interaction Status@Selection Status@Modification Status@Interaction Ability@Selectability@Modifiability @Min Interact Required@Max Interact Required@Number Of Interacted Sockets@	Set Interaction Ability (Rt-Component Target Param+, ...) Set Interaction Status (Rt-Component Target Param+, ...)	Get Interaction Ability Get Min Interact Required Get Max Interact Required Get Interaction Status Get Number Of Interacted Sockets@
Rt-components style behaviour	Style	Set Style (Rt-Component Target Param+, ...) Get Style (Rt-Component Target Param)	
Rt-contents anchor behaviour		Attach Anchor (Rt- Content Target Param+, ...)	
Channel availability behaviour	Channel Availability Status	New Channel (Channel Target Param+) Delete Channel (Channel Target Param+)	Get Channel Availability Status

Index of the behaviours (concluded)

Behaviours	Status and Attributes	Elementary Actions	Get Actions
Channel perceptability behaviour	Channel Perceptability	Set Channel Perceptability (Channel Target Param+, ...)	Get Channel Perceptability
Channel presentation space behaviour		Set CPS (Channel Target Param+, ...)	
Channels and rt-components events behaviour	Event Event Data	Set Event (Rt-Component Channel Tg Param+, ...)	Get Event Get Event Data

ITU-T RECOMMENDATIONS SERIES

- Series A Organization of the work of the ITU-T
- Series B Means of expression: definitions, symbols, classification
- Series C General telecommunication statistics
- Series D General tariff principles
- Series E Overall network operation, telephone service, service operation and human factors
- Series F Non-telephone telecommunication services
- Series G Transmission systems and media, digital systems and networks
- Series H Audiovisual and multimedia systems
- Series I Integrated services digital network
- Series J Transmission of television, sound programme and other multimedia signals
- Series K Protection against interference
- Series L Construction, installation and protection of cables and other elements of outside plant
- Series M Maintenance: international transmission systems, telephone circuits, telegraphy, facsimile and leased circuits
- Series N Maintenance: international sound programme and television transmission circuits
- Series O Specifications of measuring equipment
- Series P Telephone transmission quality, telephone installations, local line networks
- Series Q Switching and signalling
- Series R Telegraph transmission
- Series S Telegraph services terminal equipment
- Series T Terminals for telematic services**
- Series U Telegraph switching
- Series V Data communication over the telephone network
- Series X Data networks and open system communication
- Series Z Programming languages