UNION INTERNATIONALE DES TÉLÉCOMMUNICATIONS

UIT-T T.174

(10/96)

SECTEUR DE LA NORMALISATION
DES TÉLÉCOMMUNICATIONS
DE L'UIT

SÉRIE T: TERMINAUX DES SERVICES TÉLÉMATIQUES

Interface de programmation d'application pour le système MHEG-1

Recommandation UIT-T T.174

(Antérieurement Recommandation du CCITT)

RECOMMANDATIONS UIT-T DE LA SÉRIE T

TERMINAUX DES SERVICES TÉLÉMATIQUES

Pour plus de détails, voir la Liste des Recommandations de l'UIT-T.

AVANT-PROPOS

L'UIT-T (Secteur de la normalisation des télécommunications) est un organe permanent de l'Union internationale des télécommunications (UIT). Il est chargé de l'étude des questions techniques, d'exploitation et de tarification, et émet à ce sujet des Recommandations en vue de la normalisation des télécommunications à l'échelle mondiale.

La Conférence mondiale de normalisation des télécommunications (CMNT), qui se réunit tous les quatre ans, détermine les thèmes d'études à traiter par les Commissions d'études de l'UIT-T lesquelles élaborent en retour des Recommandations sur ces thèmes.

L'approbation des Recommandations par les Membres de l'UIT-T s'effectue selon la procédure définie dans la Résolution n° 1 de la CMNT (Helsinki, 1er-12 mars 1993).

La Recommandation UIT-T T.174, que l'on doit à la Commission d'études 8 (1993-1996) de l'UIT-T, a été approuvée par la CMNT (Genève, 9-18 octobre 1996).

NOTE

Dans la présente Recommandation, l'expression "Administration" est utilisée pour désigner de façon abrégée aussi bien une administration de télécommunications qu'une exploitation reconnue de télécommunications.

TABLE DES MATIÈRES

RÉSUMÉ

La présente Recommandation spécifie l'interface abstraite de programmation d'application (API, *application programming interface*) pour la manipulation d'objets d'information multimédia et hypermédia, c'est-à-dire l'interface API qui sera fournie par les moteurs MHEG en vue de leur pilotage par des applications MHEG.

INTRODUCTION

La présente Recommandation spécifie l'interface abstraite de programmation d'application (API) pour la manipulation d'objets d'information multimédia et hypermédia, c'est-à-dire l'interface API qui sera fournie par les moteurs MHEG en vue de leur pilotage par des applications MHEG.

La présente Recommandation entre dans un cadre de normalisation plus large qui spécifie l'utilisation de la norme MHEG, dont le but est de permettre la réalisation efficace d'équipements interopérables prenant en charge des services d'information et d'applications multimédias. Ceci implique:

- la spécification de contraintes supplémentaires pour l'utilisation des objets MHEG au sein de systèmes et d'applications réparties utilisant des réseaux de télécommunication;
- la définition d'interfaces API devant être fournie par des architectures de blocs de construction d'applications MHEG;
- la définition de profils MHEG venant compléter la norme MHEG-1 en spécifiant les limitations concernant les représentations codées et en spécifiant complètement le comportement exigé d'un moteur MHEG qui doit être pris en charge pour une catégorie donnée d'applications et/ou d'équipements terminaux;
- la définition d'une représentation pour un script d'échange MHEG;
- la définition de protocoles de bout en bout pour une application MHEG utilisant des services d'information multimédia/hypermédia;
- la spécification de procédures de test de conformité pour ces normes.

Les prescriptions fonctionnelles et techniques pour la présente Recommandation ont été décrites dans le rapport ETR 225.

INTERFACE DE PROGRAMMATION D'APPLICATION POUR LE SYSTÈME MHEG-1

(Genève, 1996)

1 Domaine d'application

La Partie 1 de la norme MHEG (ISO/CEI 13522-1 [1]) est une norme générique qui spécifie la représentation codée des objets d'information multimédia/hypermédia (objets MHEG) qui sont échangés. Ces objets, appelés objets MHEG, sont traités, interprétés et présentés par des moteurs MHEG.

La présente Recommandation spécifie l'interface abstraite de programmation d'application (API, *application programming interface*) pour la manipulation d'objets d'information multimédia et hypermédia, c'est-à-dire l'interface API qui sera fournie par les moteurs MHEG en vue de leur pilotage par des applications MHEG.

Cette interface API obéit aux exigences suivantes:

- elle est indépendante du langage de programmation utilisé pour l'application MHEG;
- elle est indépendante du système d'exploitation sous-jacent;
- elle est indépendante du mécanisme utilisé pour l'échange entre l'utilisateur de l'interface API (l'application MHEG) et le fournisseur de l'interface API (le moteur MHEG), c'est-à-dire des messages qui sont échangés à la suite de l'activation des primitives de l'interface API;
- elle est indépendante du codage concret de ces messages;
- elle est générique et a pour vocation de couvrir toutes les exigences d'application;
- elle peut être testée en vue d'établir sa conformité;
- elle a pour objectif une implémentation aussi facile que possible.

2 Références normatives

La présente Recommandation se réfère à certaines dispositions des Recommandations UIT-T et textes suivants qui de ce fait en sont partie intégrante. Les versions indiquées étaient en vigueur au moment de la publication de la présente Recommandation. Toute Recommandation ou tout texte étant sujet à révision, les utilisateurs de la présente Recommandation sont invités à se reporter, si possible, aux versions les plus récentes des références normatives suivantes. La liste des Recommandations de l'UIT-T en vigueur est régulièrement publiée.

[1] ISO/CEI 13522-1:1997, *Technologies de l'information – Codage de l'information multimédia et hypermédia – Partie 1: Représentation d'objet MHEG – Notation de base (ASN.1).*

[2] Recommandation UIT-T X.290 (1995), *Cadre général et méthodologie des tests de conformité d'interconnexion des systèmes ouverts pour les Recommandations sur les protocoles pour les applications de l'UIT-T – Concepts généraux.*

[3] ETR 173, *Equipements terminaux – Modèle fonctionnel pour les applications multimédias.*

[4] ETR 225, *Equipements terminaux – Interface API et représentation de script pour la norme MHEG – Exigences et cadre général.*

[5] Recommandation X.208 du CCITT (1988), *Spécifications de la syntaxe abstraite numéro un (ASN.1).*

[6] Recommandation X.209 du CCITT (1988), *Spécifications des règles de codage de base pour la notation de syntaxe abstraite numéro un (ASN.1).*

[7] Recommandation UIT-T I.113 (1993), *Glossaire des termes relatifs au RNIS à large bande.*

[8] Recommandation UIT-T I.112 (1993), *Glossaire des termes relatifs au RNIS.*

[9] Recommandation Q.9 du CCITT (1988), *Vocabulaire de termes relatifs à la commutation et à la signalisation.*

[10] ISO/CEI 14750[1)], *Technologies de l'information – Traitement distribué ouvert – Langage de définition d'interface.*

[11] ISO/CEI JTC 1 N 2965:1994, *Guidelines for JTC 1 API Standardisation (Directives pour la normalisation de l'interface API JTC 1).*

3 Définitions et abréviations

3.1 Définitions

Compte tenu de la nature particulière de la présente Recommandation, certains des termes et expressions qu'elle utilise proviennent du glossaire des normes pour les "services de télécommunication" alors que d'autres proviennent du glossaire des normes pour la "technologie du logiciel". Ceci conduit à des termes dont le sens varie en fonction du contexte, c'est-à-dire de l'expression dans laquelle ils sont utilisés. Il en résulte que de nombreuses expressions de ce genre sont définies dans ce sous-paragraphe.

Les définitions des normes suivantes s'appliqueront dans l'ordre décroissant en cas d'ambiguïté:

- ISO/CEI 13522-1 [1] MHEG;
- toute autre norme faisant partie de l'ISO/CEI 13522 [1] MHEG;
- Recommandation I.113 du CCITT [7]: glossaire de termes relatifs au RNIS à large bande;
- Recommandation I.112 du CCITT [8]: glossaire de termes relatifs au RNIS;
- Recommandation Q.9 du CCITT [9]: vocabulaire de termes relatifs à la commutation et à la signalisation.

La présente Recommandation définit les termes suivants:

3.1.1 interface de programmation d'application (API, *application programming interface***)**: frontière à travers laquelle une application logicielle utilise les fonctions d'un langage de programmation pour invoquer des services logiciels. Ces fonctions peuvent inclure des procédures ou opérations, des objets de données partagés ou des résultats de résolution d'identificateurs.

3.1.2 famille de fonctions: groupe de prescriptions fonctionnelles pour l'interface API MHEG constitué de fonctions et de la sémantique associée, et s'appliquant à un même type de cible.

3.1.3 hypermédia: capacité d'accès à des informations monomédias et multimédias au moyen d'une interaction utilisant des liens explicites.

3.1.4 service interactif: service fournissant les moyens utilisés pour un échange bidirectionnel d'information entre utilisateurs ou entre utilisateurs et machines hôtes. Les services interactifs se divisent en trois classes de services: les services conversationnels, les services de messagerie et les services de consultation (Recommandation I.113 [7] du CCITT).

3.1.5 application locale: sous-ensemble logiciel faisant partie de l'application (de télécommunication) et s'exécutant sur l'équipement considéré.

3.1.6 interface API MHEG: interface API, telle qu'elle est définie dans la présente Recommandation, mise à la disposition des applications MHEG par un moteur MHEG pour la manipulation d'objets MHEG.

3.1.7 application MHEG: sous-ensemble logiciel utilisant l'interface API MHEG. Une application MHEG est en conséquence le client d'un moteur MHEG.

3.1.8 moteur MHEG: processus ou ensemble de processus interprétant des objets MHEG codés conformément aux spécifications de codage de l'ISO/CEI 13522-1 [1]: notation de syntaxe abstraite numéro un (ASN.1) pour la première partie, et du langage standard généralisé de balisage (SGML) pour la deuxième partie.

3.1.9 application utilisant le système MHEG: application impliquant l'échange d'objets MHEG, en interne ou avec d'autres applications.

---

1) A publier.

3.1.10 application multimédia et hypermédia: application impliquant la présentation d'informations multimédias à l'utilisateur et la navigation interactive effectuée par l'utilisateur à travers ces informations.

3.1.11 service de consultation d'information multimédia et hypermédia (M&HIRS, *multimedia and hypermedia information retrieval services***)**: ensemble générique de services fournissant à l'utilisateur la capacité d'échanger et d'accéder à des informations multimédias et hypermédias.

3.1.12 application multimédia: application impliquant la présentation à l'utilisateur d'informations multimédias.

3.1.13 multimédia: capacité de manipulation de plusieurs types de média de représentation.

3.1.14 primitive: un des points d'entrée de base mis à la disposition de tout module utilisateur par un module fournisseur et permettant à ce module d'utiliser un ou plusieurs services logiciels offerts par le module fournisseur.

3.1.15 application (logicielle): sous-ensemble logiciel répondant à un ensemble d'exigences de l'utilisateur et destiné à être mis en œuvre par un utilisateur d'ordinateur.

3.1.16 service (logiciel): ensemble de fonctions fournies par un système ou un logiciel (serveur) à un système ou un logiciel client, généralement accessible au moyen d'une interface de programmation d'application.

3.1.17 application (de télécommunication): ensemble d'exigences de l'utilisateur (Recommandation Q.9 du CCITT [9]).

3.1.18 service (de télécommunication): offre d'une Administration à ses clients en vue de satisfaire à une prescription spécifique de télécommunication (Recommandation I.112 [8] du CCITT).

3.1.19 application terminale: sous-ensemble logiciel s'exécutant sur le terminal et effectuant la partie du traitement qui est exigée pour permettre à l'utilisateur d'accéder à l'application au moyen du terminal. L'application terminale est en général le module "maître" situé sur le terminal.

3.1.20 utilisateur: personne ou machine ayant reçu du client la délégation d'utiliser les services et/ou les fonctions d'un réseau de télécommunication (Recommandation I.112 [8] du CCITT).

3.1.21 action (objet): objet fournissant des opérations sur des objets, par exemple la modification de leurs attributs ou de leurs états.

3.1.22 canal: espace logique dans lequel des rt-composants (composants d'exécution) sont positionnés pour la présentation finale. Les canaux sont mappés par le moteur MHEG sur des dispositifs physiques, tels que des fenêtres sur un écran ou un haut-parleur, afin de permettre la perception des rt-objets par l'utilisateur.

3.1.23 (objet) composant: abstraction représentant des objets du type contenant ou composite.

3.1.24 (objet) composite: liste d'éléments de composition regroupés pour la présentation. La présentation d'un objet composite consiste à présenter ses objets composants.

3.1.25 (objet) conteneur: regroupement d'objets ne nécessitant pas la spécification de relations particulières.

3.1.26 (objet) contenu: valeur générique codée, données monomédias ou autres données.

3.1.27 (objet) descripteur: structure permettant l'échange d'une information de ressource concernant un objet échangé ou un ensemble de tels objets.

3.1.28 attribut (IDL): association identifiable entre un objet et une valeur. Un attribut **A** est rendu visible à des clients au moyen du couple d'opérations de lecture et d'écriture **get_A** et **set_A**. Les objets en lecture seulement ne disposent que de l'opération de lecture **get**.

3.1.29 classe (IDL): implémentation pouvant être instanciée en multiples objets de comportement identique. Un objet est une instance de classe. Les types classifient des objets relativement à une implémentation commune.

3.1.30 type de données (IDL): catégorisation des valeurs des arguments d'opérations, couvrant généralement à la fois le comportement et la représentation (il s'agit de la notion de type des langages traditionnels non orientés objets).

3.1.31 instance (IDL): un objet est une instance d'interface s'il fournit les opérations, les signatures et la sémantique spécifiée par cette interface. Un objet est une instance d'implémentation si son comportement est fourni par cette implémentation.

3.1.32 objet (IDL): combinaison d'un état et d'un ensemble de méthodes qui représente d'une manière explicite une abstraction caractérisée par le comportement des demandes s'y rapportant. Un objet est une instance d'implémentation et d'une interface. Un objet modélise une entité du monde réel. Il est implémenté sous la forme d'une entité informatique qui encapsule un état et des opérations (implémentés en interne sous la forme de données et de méthodes) et qui répond à des services de demandeur.

3.1.33 opération (IDL): service demandé. Une opération possède une signature associée qui peut imposer des restrictions à la validité des paramètres.

3.1.34 langage de définition d'interface (IDL): langage spécifiant les types et les objets au moyen de la spécification de leurs interfaces. Le langage IDL fournit un cadre conceptuel pour la description des objets.

3.1.35 (objet) lien: objet définissant une relation spatio-temporelle entre d'autres objets.

3.1.36 macro (objet): objet qui offre le moyen de remplacer les paramètres pour des objets d'action fréquemment utilisés.

3.1.37 mh-objet (objet multimédia/hypermédia): les mh-objets et leurs sous-types correspondent aux objets de type b) définis au 6.2.4 de l'ISO/CEI 13522-1: Technologies de l'information – Codage de l'information multimédia et hypermédia [1], c'est-à-dire des objets utilisables par le moteur MHEG.

3.1.38 action élémentaire (MHEG): attribut de la classe d'action MHEG donnant à un objet l'instruction de réaliser une certaine opération, par exemple de modifier un de ses attributs ou états.

3.1.39 entité MHEG: objet MHEG, rt-objet, donnée de contenu, donnée de script, réceptacle, canal ou autre structure identifiée par la norme MHEG ou à laquelle il y est fait référence.

3.1.40 service d'interprétation MHEG: service qui reçoit en entrée les objets MHEG transférés et les messages émis par le système de présentation, puis les analyse de manière à déclencher la présentation des données contenues conformément à leur sémantique.

3.1.41 service de traitement d'objet MHEG: entité physique créant, traitant et maintenant les structures de données intermédiaires nécessaires à l'implémentation d'un accès client aux objets MHEG et à leurs contenus.

3.1.42 objet MHEG: représentation codée d'une instance de classe d'objets MHEG.

3.1.43 objet modèle: instanciation d'objet à partir des classes MHEG.

3.1.44 notification: primitive émise à sa propre initiative par le serveur afin de transmettre une information au client.

3.1.45 demande: primitive émise à sa propre initiative par le client afin de transmettre une information au serveur.

3.1.46 réponse: primitive émise par le serveur en réponse à une demande de transmission d'information issue du client.

3.1.47 rt-objet (objet d'exécution): Les rt-objets (et leurs sous-types) correspondent aux objets c) tels que ceux-ci sont définis au 6.2.4 de l'ISO/CEI 13522-1: Technologies de l'information – Codage de l'information multimédia et hypermédia [1]; il s'agit donc d'instances de mh-objets utilisables par le processus de présentation.

3.1.48 (objet) script: objet fournissant la structure permettant le transfert de données de script sous une forme codée spécifiée.

3.1.49 réceptacle: élément d'un rt-composite (composite d'exécution). Les rt-composants sont enfichés dans les réceptacles. Différents types de réceptacles sont définis selon le rt-composant qui s'y connecte:

- réceptacle vide, dans lequel est enfiché un rt-composant vide;
- réceptacle présentable, dans lequel est enfiché un rt-contenu ou un rt-contenu multiplexé;
- réceptacle structurel, dans lequel est enfiché un rt-composite.

3.2 Abréviations

La présente Recommandation utilise les abréviations suivantes:

API interface de programmation d'application (*application programming interface*)

ASN.1 notation de syntaxe abstraite numéro un (telle qu'elle est définie dans la Recommandation X.208 [5]) (*abstract syntax notation one*)

BER règles de codage de base (telles qu'elles sont définies dans la Recommandation X.209 [6]) (*basic encoding rules*)

C1 période d'état pour un canal: "non disponible"

C2 période d'état pour un canal: "traitement en cours"

C3 période d'état pour un canal: "disponible"

C4 période d'état pour un canal: "traitement en cours"

CCITT Comité consultatif international télégraphique et téléphonique

CEI Commission électrotechnique internationale

CGS espace générique de canal (*channel generic space*)

CGSU unité d'espace générique de canal (*channel generic space unit*)

CORBA architecture commune de courtage d'objets (*common object request broker architecture*)

DIS projet de norme internationale (*draft international standard*)

EBNF formalisme Backus-Naur étendu (*extended Backus-Naur form*)

ETR rapport technique de l'ETSI (*ETSI technical report*)

ETS norme européenne de télécommunications (*European Telecommunication Standard*)

ETSI Institut européen des normes de télécommunications (*European Telecommunications Standards Institute*)

GSR rapport spatial générique (*generic spatial ratio*)

GTU unité temporelle générique (*generic temporal unit*)

IDL langage de définition d'interface (tel qu'il est défini dans l'ISO/CEI 14750-1 IDL [10]) (*interface definition language*)

IOGTR rapport temporel générique d'espace générique original initial (*initial original generic space generic temporal ratio*)

IOV volume audible original initial (*initial original audible volume*)

ISO Organisation internationale pour la normalisation (*International Organisation For Standardisation*)

JTC comité technique mixte (*joint technical committee*)

MCU unité de commande multipoint (*multipoint control unit*)

MHEG groupe d'experts pour le codage de l'information multimédia et hypermédia (*multimedia and hypermedia information coding experts group*)

MPEG groupe d'experts pour les images animées (*moving picture experts group*)

O1 période d'état d'un mh-objet: "non prêt"

O2 période d'état d'un mh-objet: "en cours de traitement"

O3 période d'état d'un mh-objet: "prêt"

O4 période d'état d'un mh-objet: "en cours de traitement"

OGS espace générique original (*original generic space*)

OGSU unité spatiale générique d'espace générique original (*original generic space generic spatial unit*)

OGTR rapport temporel générique d'espace générique original (*original generic space generic temporal ratio*)

OGTU unité temporelle générique d'espace générique original (*original generic space generic temporal unit*)

OO orienté objet

OSI interconnexion des systèmes ouverts (*open systems interconnection*)

OV volume original (*original volume*)

PRGS espace générique relatif parent (*parent relative generic space*)

PS taille perceptible (*perceptible size*)

PSAP point de rattachement de taille perceptible (*perceptible size attachment point*)

R1 période d'état pour un rt-objet: "non disponible"

R2 période d'état pour un rt-objet: "en cours de traitement"

R3 période d'état pour un rt-objet: "disponible"

R4 période d'état pour un rt-objet: "en cours de traitement"

RGS espace générique relatif (*relative generic space*)

RGSU unité spatiale générique d'espace générique relatif (*relative generic space generic spatial unit*)

RGTU unité temporelle générique d'espace générique relatif (*relative generic space generic temporal unit*)

rt- *préfixe signifiant* "d'exécution" (*run-time*)

SGML langage standardisé généralisé de balisage (*standard generalized markup language*)

SIR représentation d'échange de script (*script interchange representation*)

SSU unité de prise en charge de service (*service support unit*)

UIT-T Union internationale des télécommunications – Secteur de la normalisation des télécommunications

VD durée visible (*visible duration*)

VS taille visible (*visible size*)

VSEAP point externe de rattachement de la taille visible (*visible size external attachment point*)

VSGS espace générique de la taille visible (*visible size generic space*)

VSGSU unité spatiale générique de la taille visible (*visible size generic spatial unit*)

VSIAP point interne de rattachement de la taille visible (*visible size internal attachment point*)

4 Conformité

Une implémentation de la présente Recommandation est une implémentation de moteur MHEG qui fournit à des applications MHEG clientes une ou plusieurs liaisons à un langage pour la prise en charge de l'interface abstraite de programmation d'application (API) définie dans la présente Recommandation.

Une application de la présente Recommandation est une application MHEG utilisant une liaison à un langage pour la prise en charge de l'interface API définie dans la présente Recommandation dans le but de piloter le comportement du moteur MHEG.

Les sous-paragraphes qui suivent énoncent les exigences liées à la conformité à la présente Recommandation en ce qui concerne des implémentations et des applications.

4.1 Conformité d'implémentation

4.1.1 Exigences de conformité

Une implémentation en conformité pour une spécification de liaison à un langage de la présente Recommandation doit satisfaire à l'ensemble des critères suivants:

1) l'implémentation prendra en charge la totalité du comportement exigé défini dans la présente Recommandation;

2) l'implémentation prendra en charge toutes les interfaces exigées, telles que définies dans la spécification de liaison à un langage. Ces interfaces prendront en charge le comportement décrit dans la présente Recommandation et dans la spécification de liaison à un langage;

3) l'implémentation peut fournir des fonctions supplémentaires non exigées par la présente Recommandation ou par la spécification de liaison à un langage. Toute extension non normalisée sera identifiée comme telle dans la documentation. Les extensions non normalisées peuvent, lorsqu'elles sont utilisées, modifier le comportement de fonctions définies dans la présente Recommandation et par la spécification de liaison à un langage. Le document de conformité définira un environnement dans lequel une application peut être exécutée avec le comportement spécifié dans la présente Recommandation ou la spécification de liaison à un langage. Un tel environnement ne doit en aucun cas nécessiter la modification d'une application strictement conforme.

4.1.2 Documentation de conformité

Un document de conformité, contenant l'information donnée ci-dessous, sera fourni pour une implémentation déclarant la conformité avec une spécification de liaison à un langage de la présente Recommandation. Le document de conformité contiendra deux parties. La structure de la première partie sera la même que celle de la présente Recommandation, l'information étant présentée dans les sections, paragraphes et sous-paragraphes subordonnés adéquats. La structure de la deuxième partie sera la même que la structure d'une spécification de liaison à un langage, l'information étant présentée dans les sections, paragraphes et sous-paragraphes adéquats. Le document de conformité ne contiendra pas d'information concernant les extensions de fonctions et de capacités qui se trouvent en dehors du domaine d'application de la présente Recommandation et de la spécification de liaison à un langage.

Le document de conformité identifiera la spécification de liaison à un langage à laquelle se conforme l'implémentation.

Le document de conformité contiendra une déclaration indiquant complètement les noms, numéros et dates des normes indépendantes du langage et des spécifications de liaison à un langage qui s'appliquent.

Le document de conformité indiquera quelles sont les fonctions optionnelles, définies dans la présente Recommandation et dans la spécification de liaison à un langage, qui sont prises en charge par l'implémentation.

Le document de conformité décrira le comportement de l'implémentation pour toutes les fonctions d'implémentation définies dans la présente Recommandation et dans la spécification de liaison à un langage. Cette exigence sera respectée en énumérant ces fonctions et en fournissant soit une référence spécifique vers la documentation du système, soit la syntaxe et la sémantique complètes de ces fonctions. Le document de conformité peut spécifier le comportement de l'implémentation des fonctions pour lesquelles la présente Recommandation ou la spécification de liaison à un langage déclare que les implémentations peuvent varier ou lorsque les fonctions sont identifiées comme non définies ou non spécifiées.

Aucune spécification, autre que celles spécifiées par la présente Recommandation et par la spécification de liaison à un langage, ne figurera dans le document de conformité.

Les expressions "documentera" ou "sera documentée" utilisées dans la présente Recommandation ou dans une spécification de liaison à un langage pour la présente norme signifient que la documentation de la fonction figurera, comme mentionné précédemment, dans le document de conformité, à moins qu'il ne soit fait mention d'une manière explicite de la documentation du système.

La documentation du système contiendra également l'information figurant dans le document de conformité.

4.2 Conformité d'application

Toutes les applications déclarant la conformité à une liaison à langage de la présente Recommandation appartiendront à l'une des catégories définies dans les sous-paragraphes ci-dessous.

4.2.1 Application strictement conforme

Une application est strictement conforme si elle exige uniquement les fonctions obligatoires décrites dans la présente Recommandation, dans la spécification de liaison à un langage et dans les normes de langage qui s'appliquent. Une telle application acceptera un comportement décrit, dans la présente Recommandation ou la spécification de liaison à un langage, comme non spécifié ou comme défini par l'implémentation et acceptera pour les constantes symboliques toute valeur située dans les domaines autorisés par la présente Recommandation et par la spécification de liaison à un langage.

4.2.2 Application conforme

Une application conforme d'une spécification de liaison à un langage pour la présente Recommandation diffère d'une application strictement conforme dans la mesure où elle peut utiliser des facilités optionnelles décrites dans la présente Recommandation, dans la spécification de liaison à un langage et dans les normes de langage applicables, ainsi que des fonctions non normalisées qui sont compatibles avec la norme et la spécification de liaison à un langage. Une telle application documentera complètement ses exigences concernant ces fonctions optionnelles et étendues en plus de la documentation exigée pour une application conforme.

4.3 Méthodes de test

Toute mesure de conformité avec une spécification de liaison à un langage pour la présente Recommandation sera effectuée en utilisant des méthodes de test se conformant à la Recommandation X.290 [2] et à toute autre exigence supplémentaire pouvant être imposée par la spécification de liaison à un langage.

5 Description générale

Ce paragraphe situe l'interface API MHEG au sein de l'architecture d'une application utilisant le système MHEG, puis spécifie et classe les services logiciels que l'interface API MHEG fournira à ses applications clientes.

5.1 Modèle de référence fonctionnel des applications utilisant le système MHEG

5.1.1 Modèle de référence pour les applications multimédias

Les rapports ETR 173 [3] "Modèle fonctionnel pour les applications multimédias" et ETR 225 [4] "Interface API et représentation de script pour la norme MHEG – Exigences et cadre général" définissent un modèle générique de référence qui décrit une architecture fonctionnelle commune pour toutes les applications multimédias utilisant des services de consultation, des services conversationnels et/ou des services de distribution. Ces modèles sont communs à toutes les applications utilisant le système MHEG.

Les Figures 1 à 3 indiquent de quelle manière ce modèle s'applique à des applications utilisant la norme MHEG dans des configurations terminal-terminal et terminal-base de données faisant appel à des communications de point à point ou multipoint.

Les unités fonctionnelles suivantes peuvent être identifiées dans une architecture multimédia/hypermédia:

1) l'agent de présentation;
2) l'agent d'accès;
3) l'interpréteur d'application local;
4) le moteur MHEG;
5) la base d'informations locale.

Les interfaces API suivantes peuvent être identifiées dans une architecture multimédia/hypermédia:

1) les services de l'agent de présentation fournis par l'intermédiaire de l'interface API de présentation;
2) les services du moteur MHEG fournis par l'intermédiaire de l'interface API MHEG;
3) les services de l'agent d'accès fournis par l'intermédiaire de l'interface API d'accès.

Les protocoles de bout en bout suivants peuvent être identifiés dans une architecture multimédia/hypermédia:

1) le protocole de bout en bout entre une application et un moteur MHEG distant;
2) le protocole de bout en bout entre une application et la fonction de service de prise en charge;
3) le protocole de bout en bout entre un agent d'accès et un autre agent d'accès.

Dans une configuration "terminal-hôte et base de données", l'hôte peut utiliser le protocole de bout en bout entre agents d'accès pour faire référence à des objets ou des contenus de la base de données. La base de données possède la même structure que celle indiquée dans la Figure 3.

Dans une configuration terminal-hôte, l'unité de prise en charge de service (SSU, *service support unit*) à laquelle peut être connecté le terminal peut également permettre à l'utilisateur de faire un choix entre plusieurs applications. L'hôte est connecté à l'unité SSU au moyen du réseau d'accès à l'hôte.

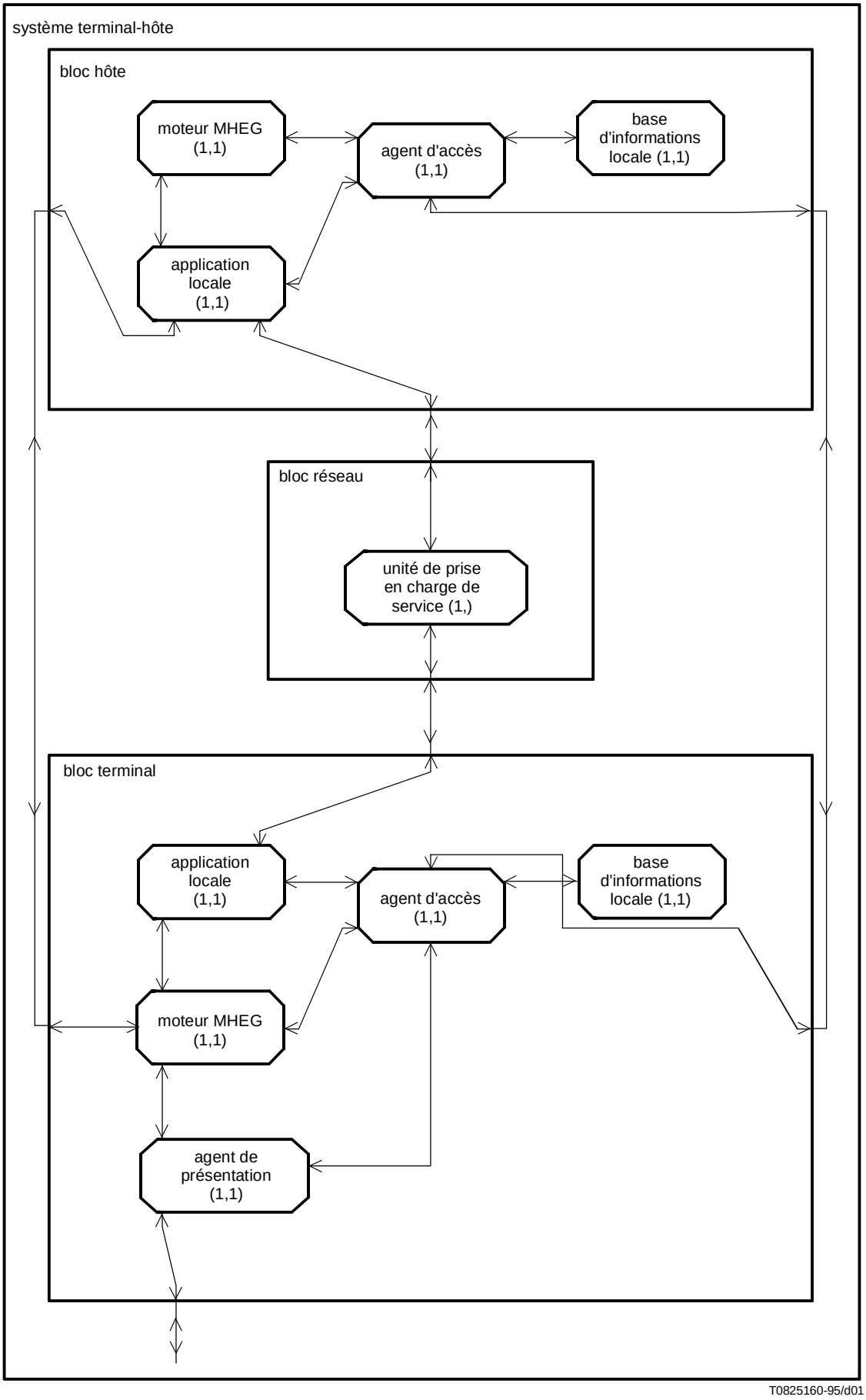


Figure 1/T.174 – Modèle de référence d'architecture d'application pour des configurations terminal-hôte

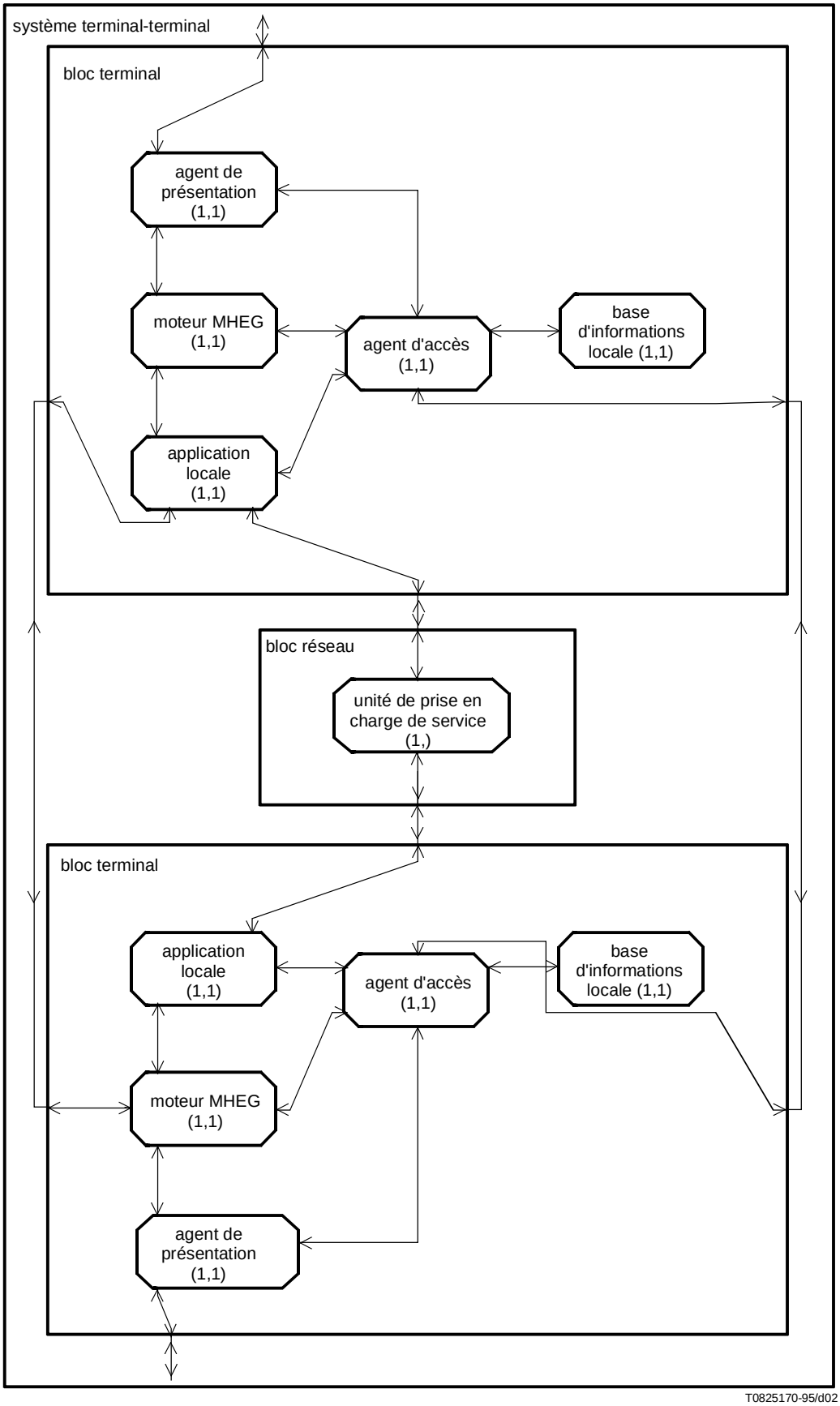


Figure 2/T.174 – Modèle de référence d'architecture d'application pour des configurations terminal-terminal

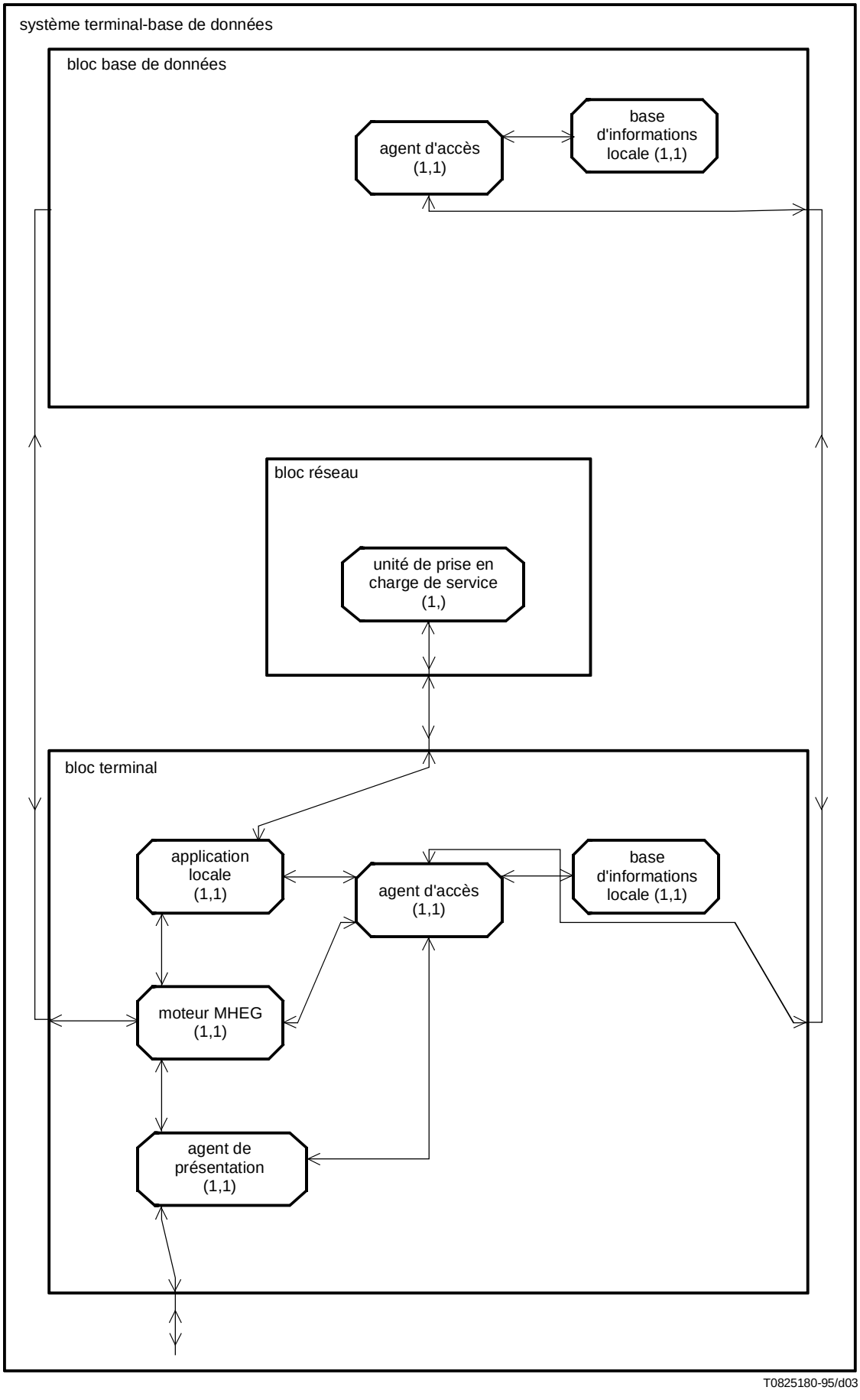


Figure 3/T.174 – Modèle de référence d'architecture d'application pour des configurations terminal-base de données

Dans une configuration terminal-terminal, l'unité SSU à laquelle sont connectés tous les terminaux peut être une unité de commande multipoints (MCU, *multipoint control unit*) qui pilote et supervise l'application et les divers terminaux. Le protocole de bout en bout entre une application et un moteur MHEG distant s'applique pour tous les terminaux. Chaque application terminale peut utiliser le protocole pour communiquer avec une autre application terminale ou un moteur MHEG.

Dans une structure "terminal-base de données" ou "terminal-hôte et base de données", la base de donnée se constitue principalement d'un agent d'accès utilisé pour la localisation des objets référencés.

L'**agent de présentation** fournit un service de présentation de contenu multimédia. Il gère la présentation de données monomédias, réalise le décodage du format de données et gère l'interaction avec l'utilisateur. Il sert également d'interface avec des dispositifs externes tels que des cartes à puce, des enregistreurs vidéo, etc. L'agent de présentation est utilisé par ses clients (dans ce cas le moteur MHEG) au moyen de "l'interface API de présentation" qui isole le logiciel de niveau supérieur des fonctions spécifiques des diverses plates-formes matérielles. L'agent de présentation n'existe que sur un terminal.

L'**agent d'accès** fournit les services de localisation, d'accès et de communication pour les objets MHEG et pour les contenus multimédias. Il rend transparente, pour ses clients, la localisation des divers objets (contenus multimédias, objets MHEG, scripts ou données spécifiques d'application), c'est-à-dire le processus de tentative d'accès à ces objets. Lorsqu'un objet est demandé, l'agent d'accès est tout d'abord en mesure de le localiser (en faisant éventuellement appel à un service de répertoire), puis de le consulter à partir d'un stockage local ou distant (en envoyant éventuellement des demandes d'accès à des services de référentiel et à des équipements distants) et enfin d'en fournir l'accès (en utilisant éventuellement des services de codage et de décodage d'objets et de contenus MHEG). A l'inverse, l'agent d'accès gère aussi bien l'expédition d'objets d'équipements distants, leur enregistrement dans des répertoires et leur stockage dans des dépôts.

L'**application locale** gère la logique de l'application sur une plate-forme donnée. L'application elle-même sera souvent répartie entre plusieurs plates-formes (terminaux, hôtes). L'application locale est un client de l'agent d'accès à travers l'interface API d'accès, du moteur MHEG à travers l'interface API MHEG et de l'agent de présentation à travers l'interface API de présentation. L'application locale peut utiliser un service d'exécution de script fourni par un processeur de script. Les scripts sont des parties d'application qui sont échangées au cours d'une application. Un processeur de script est une unité fonctionnelle qui est en mesure d'exécuter des scripts. Il peut s'agir d'un script (s'il est échangé sous une forme exécutable), d'un interpréteur de langage de script ou d'un interpréteur de représentation d'échange de scripts MHEG (SIR, *script interchange representation*).

Le **moteur MHEG** fournit un service d'interprétation MHEG. Il interprète les objets MHEG, gère leurs liens mutuels, lance des actions et ordonnance la présentation des objets et leur accès. Il est piloté par l'application au moyen de l'interface API MHEG.

La **base d'informations locale** peut être utilisée pour stocker dans l'équipement, d'une manière temporaire ou permanente, des objets, des contenus, des scripts, etc. Il n'est fait aucune hypothèse en ce qui concerne la nature des objets stockés et le dispositif physique de stockage utilisé.

L'**interface API de présentation** permet au moteur MHEG terminal d'accéder au service de présentation de contenu multimédia fourni par un agent de présentation.

L'**interface API MHEG** permet à des applications MHEG d'accéder au service d'interprétation MHEG fourni par un moteur MHEG. L'application cliente de l'interface API MHEG peut être soit une application locale, soit s'exécuter sur un équipement distant. Si l'application est exécutée sur un équipement distant, elle peut accéder à l'interface API MHEG en utilisant le protocole de bout en bout "application – moteur MHEG distant" (9). La présente Recommandation spécifie l'interface API MHEG.

L'**interface API d'accès** permet à des processus (par exemple une application locale, un interpréteur de script, un moteur MHEG ou un agent de présentation) d'accéder à un objet MHEG et à l'emplacement qui le contient, en utilisant les services d'accès et de communication fournis par un agent d'accès.

L'**unité de prise en charge de service** est une unité fonctionnelle qui traite les paramètres de commande spécifiques au service et offre des fonctions qui dépendent du service particulier. Un exemple d'unité de prise en charge de service est l'unité MCU utilisée pour le service de visioconférence. Les fonctions traitées par l'unité de prise en charge de service sont spécifiques du service.

Le **protocole de bout en bout** "**application – moteur MHEG distant**" permet à une application exécutée sur un équipement distant de communiquer avec le moteur MHEG local. Ce protocole permet l'implémentation de configurations terminal-hôte.

Le **protocole de bout en bout "application – unité de prise en charge de service"** permet l'utilisation des services du fournisseur de service par l'application terminale ou par l'utilisateur.

Le **protocole de bout en bout "agent d'accès – agent d'accès"** permet le traitement, la maintenance et l'échange d'objets et de données dans un environnement réparti.

5.1.2 L'interface API MHEG

Le présent sous-paragraphe décrit la terminologie qui s'applique pour l'interface API MHEG.

L'interface API MHEG est l'interface au moyen de laquelle une application MHEG peut piloter un moteur MHEG

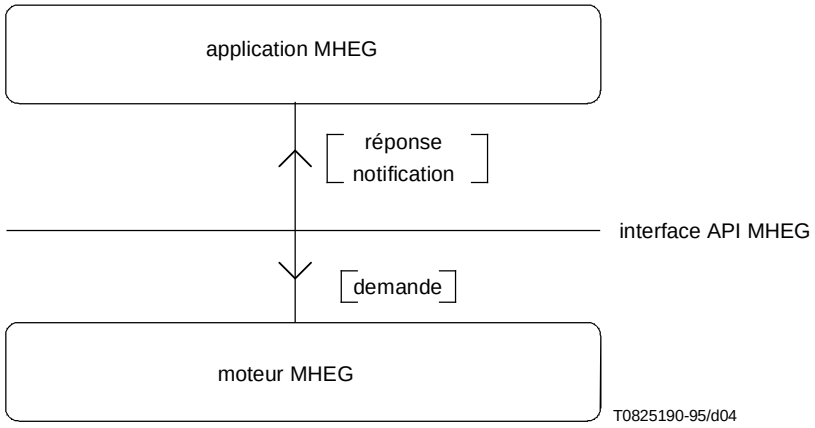


Figure 4/T.174 – L'interface API MHEG

Une interface API se constitue de **primitives**, c'est-à-dire de points d'entrée de base fournis par un module **fournisseur** à tout module **utilisateur** de manière à permettre à l'utilisateur l'accès à des **services logiciels** du fournisseur. Ces modules sont des sous-ensembles de logiciel pouvant utiliser des services fournis par un matériel informatique ou par un autre équipement électronique.

L'**interface API MHEG** fournit l'accès à des services d'**interprétation MHEG** et, d'une manière optionnelle, à des services d'**accès à des objets MHEG**. Les **moteurs MHEG** sont les fournisseurs et les **applications MHEG** sont les utilisateurs. Un moteur MHEG peut fournir des services logiciels à plusieurs applications MHEG. Un moteur MHEG peut donc être considéré comme un serveur, les applications MHEG étant les **clients** des services logiciels.

Une primitive de l'interface API est utilisée pour transférer une certaine information entre l'utilisateur et le fournisseur. Cette information se constitue de commandes et/ou de données. L'information peut être retransmise soit par un client vers son serveur, soit par un serveur vers l'un de ses clients. L'information peut être générée par l'émetteur à sa propre initiative, soit en réponse à une primitive reçue précédemment. Les termes suivants sont utilisés:

- une **demande** est une primitive émise par le client à sa propre initiative afin de transmettre de l'information vers le serveur;
- une **réponse** est une primitive émise par le serveur en réponse à une demande de transmission d'information vers le client;
- une **notification** est une primitive émise par le serveur à sa propre initiative afin de transmettre de l'information vers le serveur.

On peut distinguer différents types de demandes:

- les demandes qui n'exigent pas de réponse sont appelées **demandes asynchrones**;
- les demandes qui n'exigent pas de réponse immédiate sont appelées **demandes synchrones différées**;
- les demandes qui exigent une réponse immédiate et suspendent le traitement fait par le client sont appelées **demandes synchrones**.

Les demandes synchrones ou asynchrones peuvent avoir comme résultat un traitement qui peut à son tour déclencher des notifications.

5.2 Spécifications fonctionnelles de l'interface API MHEG

5.2.1 Spécifications d'utilisation du système MHEG

Le présent sous-paragraphe introduit les règles d'utilisation du système MHEG. Il s'agit de clarifications et d'interprétations de la norme MHEG pour ce qui est de la définition des principales entités citées par la norme MHEG et qui peuvent être traitées par des applications utilisant le système MHEG.

Une compréhension claire des entités en relation avec le système MHEG implique une définition formelle de l'identité des entités qui sont manipulées et/ou transformées par des composants d'une application utilisant le système MHEG, ainsi que la façon de les identifier ou d'y faire référence au sein de divers composants.

5.2.1.1 Définitions

Des sous-paragraphes introduisent le concept d'identité, d'identification et de référence. Les précisions qui suivent complètent la sémantique de la norme MHEG (qui ne fait pas de distinction claire entre référence et identification), et constituent des définitions s'appliquant à l'ensemble de la présente Recommandation.

L'identité d'un objet est l'objet même. La fonction d'identité est définie par la formule Identité(A) = A. Tout objet possède une identité. Deux objets A et B ont la même identité si, et seulement si, A et B sont le même objet.

L'identification d'un objet est un moyen non ambigu de déterminer l'identité d'un objet. Si deux objets A et B ont le même identificateur, ils possèdent la même identité et constituent un seul et même objet. L'identificateur est souvent, mais non nécessairement, contenu dans l'objet. Les objets peuvent avoir plusieurs modes d'identification (avec différents types d'identificateurs), quoique ceci ne soit ni utile ni recommandable. Si un objet n'a pas d'identificateur, il ne peut être identifié bien qu'il possède une identité.

La référence à un objet est un moyen commode d'associer un nom à un objet. Il est nécessaire de résoudre une référence d'une manière ou d'une autre pour déterminer l'objet référencé. Une référence ne peut concerner qu'un seul objet à un instant donné, mais un objet peut être référencé d'une manière multiple. Contrairement à l'identificateur d'un objet, les références ne sont donc pas nécessairement uniques. Il est en outre possible de référencer des objets qui ne sont pas identifiables par ailleurs.

5.2.1.2 Objets MHEG

Un objet MHEG est défini, selon la norme MHEG, comme une représentation codée. Il s'ensuit que les objets MHEG sont des chaînes binaires. L'identité d'un objet MHEG est constituée de sa chaîne binaire. Les objets MHEG sont des objets de "forme a", telle qu'elle est décrite au 6.2.4 de la norme MHEG. Un objet A et un objet B sont identiques si et seulement s'ils **sont** la même succession de bits.

Un objet MHEG n'a pas d'existence physique mais constitue une abstraction (une suite de bits spécifiée) qui peut avoir de nombreuses représentations (sous forme d'objets différents) de types distincts: des objets MHEG échangés, des objets MHEG stockés, des mh-objets, etc. De telles représentations sont traitées par des services logiciels distincts.

Un objet MHEG peut être identifié par un identificateur MHEG. Les identificateurs sont l'**unique** moyen d'identifier des objets MHEG. La structure et la représentation codée des identificateurs MHEG sont définies par la norme MHEG. L'identificateur MHEG d'un objet MHEG doit être codé au sein de l'objet MHEG. Comme cet attribut est optionnel, certains objets MHEG n'ont pas d'identificateur MHEG. De tels objets ne peuvent être identifiés. La norme MHEG impose, pour la conception des applications qui la mettent en œuvre, la contrainte que deux objets A et B n'auront pas le même identificateur MHEG à moins qu'ils ne soient identiques.

La référence générique de la norme MHEG décrit tous les moyens possibles pour référencer un objet MHEG.

5.2.1.3 mh-objets

Un mh-objet est une représentation interne d'un objet MHEG au sein d'un processus ou d'un système. Un mh-objet n'est pas un objet MHEG. Les mh-objets représentent les objets MHEG "disponibles" au sein d'un moteur MHEG. Les mh-objets sont des objets de "forme b" telle qu'elle est décrite au 6.2.4 de la norme MHEG. Un mh-objet représente un objet MHEG, c'est-à-dire qu'il existe toujours une chaîne binaire correspondant à un mh-objet. Un moteur MHEG n'utilisera pas plus d'un mh-objet pour représenter un objet MHEG.

Il en résulte que les mh-objets traités par des moteurs MHEG peuvent être identifiés au moyen d'identificateurs MHEG. D'autres mécanismes d'identification de mh-objet peuvent être définis par l'application (par exemple comme identification symbolique), dans la mesure où leur représentation interne le permet. Ceci est particulièrement utile lorsque certains des objets MHEG représentés dans un moteur MHEG par des mh-objets ne sont pas identifiables, c'est-à-dire n'ont pas d'identificateur MHEG. Ceci permet d'assurer que tous les mh-objets seront identifiables.

Les mh-objets sont référencés de la même manière que les objets MHEG. Les références à des objets MHEG pour lesquels le moteur MHEG traite des mh-objets sont en général résolues par l'adressage de ces mh-objets.

5.2.1.4 rt-objets (objets d'exécution)

Un rt-objet (*run time object* ou objet d'exécution) est une "instance" (ou copie) d'exécution d'un mh-objet "modèle", créée et traitée par un moteur MHEG dans un but de présentation. Un rt-objet n'est pas un objet MHEG. Les rt-objets représentent les objets MHEG qui sont disponibles sous une forme exécutable au sein d'un moteur MHEG. Les rt-objets sont des objets de "forme c" telle qu'elle est décrite au 6.2.4 de la norme MHEG. Il peut exister zéro ou plusieurs rt-objets qui sont des copies "présentables" d'un mh-objet. Un rt-objet possède toujours comme modèle un et un seul mh-objet.

Les rt-objets peuvent être identifiés au moyen d'identificateurs de rt-objet dont "l'identification d'objet modèle" est définie par la norme MHEG. La norme MHEG décrit la structure et le codage des identificateurs des rt-objets. D'autres mécanismes d'identification de rt-objet peuvent être définis par l'application (par exemple comme identification symbolique), dans la mesure où leur représentation interne le permet. Ceci est particulièrement utile lorsque certains des objets MHEG représentés dans un moteur MHEG par des rt-objets ne sont pas identifiables, c'est-à-dire n'ont pas d'identificateur MHEG. Ceci permet d'assurer que tous les rt-objets seront identifiables.

La référence à des rt-objets peut être faite au moyen de références génériques MHEG.

5.2.1.5 Canaux

Les canaux sont des objets définis par la norme MHEG et traités par les moteurs MHEG. Ils peuvent être identifiés au moyen d'identificateurs de canaux.

5.2.1.6 Objets MHEG échangés

Les objets MHEG échangés sont des représentations d'objets MHEG qui font à un instant donné l'objet d'une communication utilisant un réseau ou un moyen de stockage. Un objet MHEG donné (c'est-à-dire une chaîne binaire) peut être échangé de nombreuses fois entre de nombreux endroits, c'est-à-dire être représenté par de nombreux objets MHEG échangés. Un identificateur MHEG externe peut identifier un objet MHEG échangé et, en conséquence, référencer un objet MHEG en fonction de sa localisation et du moment de l'échange. Il convient toutefois de noter qu'un identificateur MHEG externe ne correspond pas nécessairement à un objet MHEG effectif.

Les objets MHEG stockés sont des représentations d'objets MHEG qui sont en général localisées dans des enregistrements de fichier ou de base de données. Par exemple, un objet MHEG donné (c'est-à-dire une chaîne binaire) peut être stocké en de nombreux endroits, c'est-à-dire être représenté par de nombreux objets stockés. Un identificateur MHEG externe peut identifier un emplacement de stockage pour un objet MHEG et, en conséquence, référencer un objet MHEG compte tenu de son emplacement de stockage.

5.2.2 Description de services en relation avec la norme MHEG

Le présent sous-paragraphe introduit le concept de services en relation avec la norme MHEG, c'est-à-dire de services logiciels généraux permettant le traitement d'entités en relation avec la norme MHEG. Ces services sont fournis par les blocs de construction (composants) de la norme MHEG utilisant une architecture d'application sur laquelle est construite l'architecture MHEG. Les clients et les services ne se connaissent pas nécessairement et peuvent être mis en relation par une entité médiatrice. Dans une telle approche par blocs de construction, les implémentations de services peuvent s'utiliser mutuellement.

Une application MHEG peut utiliser tout ou partie des services suivants en relation avec la norme MHEG:

- le **service d'interprétation MHEG** permettant de gérer le comportement du moteur MHEG, c'est-à-dire l'interprétation et la présentation des objets MHEG;
- le **service d'accès aux objets MHEG** permettant d'accéder aux attributs des objets MHEG "logiques" et de les modifier;
- le **service de communication d'objets MHEG** permettant le transfert d'objets MHEG entre les localisations;

- le **service de localisation d'objets MHEG** permettant de gérer et de résoudre les références à des objets MHEG;
- le **service de traitement d'objets MHEG** permettant de gérer l'accès à des objets MHEG physiques et de les échanger;
- le s**ervice de stockage d'objets MHEG** permettant de stocker et d'accéder aux chaînes binaires des objets MHEG;
- le **service de codage et de décodage d'objets MHEG** permettant de coder ou de décoder des objets MHEG;
- d'autres services de distribution, de présentation et de production en temps réel d'objets MHEG et de représentation peuvent être pris en considération.

Ces services logiciels traitent différentes sortes d'entités, en relation avec la norme MHEG, qui ont des relations fortes mais sont toutefois de types différents (et en conséquences non comparables):

- le service d'interprétation MHEG traite les rt-objets, les réceptacles, les canaux et les mh-objets;
- le service d'accès aux objets MHEG traite les mh-objets;
- le service de traitement des objets MHEG "traite" les objets MHEG. Comme les objets MHEG sont des entités virtuelles et non physiques, ce service fait appel à des services qui traitent les "représentations" physiques des objets MHEG, tels que les services de stockage ou de transport;
- le service de localisation des objets MHEG traite les références MHEG génériques. Il est en mesure de résoudre de telles références en utilisant des cartes de localisation, par exemple de les traduire en identificateurs pouvant être compris par l'application qui en fait la demande;
- le service de communication d'objets MHEG traite les objets MHEG échangés;
- le service de stockage d'objets MHEG traite les objets MHEG stockés et gère les emplacements de stockage, dont le type dépend du mécanisme de stockage sous-jacent qui peut utiliser, par exemple, des enregistrements de fichiers ou de bases de données;
- le service de codage et de décodage fournit les fonctions permettant de transformer des mh-objets en objets MHEG et réciproquement.

L'interface API MHEG se constitue de l'interface fournie par les services suivants, qui seront fournis par les moteurs se conformant à la norme MHEG:

- le service (obligatoire) d'interprétation MHEG;
- le service (optionnel) d'accès aux objets MHEG.

6 Principes de définition de l'interface API

6.1 Satisfaction aux exigences techniques de l'interface API MHEG

Conformément aux recommandations de l'ISO/CEI JTC 1 N 2965 [11]: "Directives pour la normalisation des interfaces API JTC 1", l'interface API MHEG est définie sous la forme d'une spécification abstraite d'interface API, c'est-à-dire d'une description, indépendante de tout langage, de la sémantique d'un ensemble de fonctions faite dans une syntaxe abstraite utilisant des types abstraits de données.

La norme d'interface API MHEG devra respecter les exigences suivantes, conformément aux recommandations du rapport ETR 225:

- portabilité;
- caractère générique;
- faisabilité du test de conformité;
- faisabilité de l'implémentation.

L'exigence de **portabilité** stipule que la norme d'interface API MHEG permettra aux applications MHEG d'utiliser le service de manipulation et d'échange d'objets MHEG fourni par les moteurs MHEG d'une manière qui est indépendante:

- du langage de programmation utilisé pour l'application MHEG;
- du système d'exploitation sous-jacent.

La présente Recommandation satisfait aux exigences de portabilité par la définition d'une spécification abstraite d'interface API.

L'exigence de **caractère générique** stipule que la norme d'interface API MHEG doit fournir une prise en charge adéquate couvrant toutes les exigences communes des applications MHEG.

La présente Recommandation satisfait aux exigences de caractère générique en définissant l'interface API MHEG au niveau le plus élémentaire, c'est-à-dire en définissant des primitives qui correspondent aux actions MHEG élémentaires et des types de données qui correspondent aux types de données MHEG. Ceci assure que le domaine d'utilisation des fonctions de manipulation d'objets mises à la disposition des applications est le plus large possible.

L'exigence de **faisabilité de test de conformité** stipule que la norme d'interface API MHEG doit faciliter autant que possible la vérification de la conformité de moteurs MHEG à la norme d'interface API MHEG, c'est-à-dire la fourniture correcte de cette interface API par un moteur MHEG faisant l'objet d'un test, ainsi que la conformité d'applications MHEG à la norme d'interface API MHEG, c'est-à-dire l'utilisation correcte de cette interface API par une application MHEG faisant l'objet d'un test.

La présente Recommandation satisfait à l'exigence de faisabilité de test de conformité en exprimant d'une manière formelle les exigences concernant les implémentations conformes et les applications conformes, ainsi que par l'utilisation d'une technique de description formelle pour la définition de l'interface API MHEG.

L'exigence de **faisabilité d'implémentation** stipule que la norme d'interface API MHEG doit tenir compte d'exigences de simplicité et de clarté dans sa définition et sa formulation de manière à faciliter le plus possible l'implémentation de moteurs MHEG conformes.

La présente Recommandation satisfait à l'exigence de faisabilité d'implémentation en fournissant des directives d'information permettant de déduire de la spécification abstraite de l'interface API les spécifications d'intégration dans un langage et des règles de codage des messages.

6.2 Utilisation du langage de définition d'interface (IDL) ISO/CEI 14750 [10]

L'interface API MHEG est définie au moyen du langage IDL.

6.2.1 Introduction complète au langage IDL

Le langage IDL constitue une technique de spécification formelle permettant de spécifier les services fournis par des objets utilisés par des applications ou d'autres objets. Bien que la communication par objets dans des environnements répartis soit en principe une technologie qui possède certains rapports avec la définition d'une boîte à outils multimédia de base, la présente Recommandation traite de l'utilisation du langage IDL et du modèle d'objets sous-jacent uniquement sous l'aspect d'une technique formelle de spécification d'interface API indépendante du contexte.

Une application du langage IDL doit être fondée sur un modèle d'objets sous-jacent. Un tel modèle est défini en termes de **types d'objets** prenant en charge des **opérations** qui caractérisent le comportement des objets. Les **objets** sont des instances de types d'objets. Les objets peuvent être identifiés au moyen de **références d'objets**. Les **types non objets** peuvent être instanciés mais ne prennent pas d'opération en charge. Les opérations sont définies par une **signature** qui se constitue d'un nom, d'une liste de types de **paramètres** en entrée ou en sortie et d'une liste de **résultats**. L'ensemble des signatures des opérations définies pour un type constitue l'**interface** de ce type. La création de **sous-types** permet de définir des hiérarchies de types, les sous-types fournissant l'interface de leurs types supérieurs comme partie de leurs propres interfaces. Les **demandes d'opération** peuvent avoir des sémantiques d'exécution diverses, telles que les exécutions synchrones, asynchrones, etc. Les conséquences d'une opération incluent les effets de bord, les résultats et les **exceptions**.

Le langage IDL est utilisé pour décrire les interfaces (c'est-à-dire l'ensemble des opérations) fournies par des objets. Il est constitué de conventions lexicographiques, de directives de prétraitement et d'une grammaire définie selon un formalisme Backus-Naur étendu (EBNF, *extended Backus-Naur form*). Une spécification en langage IDL d'une interface API se constitue de définitions de types de données, de définitions de constantes, de définitions d'interfaces et de définitions de modules.

6.2.2 Le langage de définition d'interface

Le présent sous-paragraphe décrit les principaux concepts nécessaires à la compréhension de la définition de l'interface API MHEG.

Le modèle d'objets fournit une représentation ordonnée du concept et de la terminologie d'objets. Il définit un modèle partiel de traitement qui incorpore les caractéristiques principales des objets tels qu'ils sont réalisés au moyen des technologies présentées.

Le modèle d'objets exposé dans la présente Recommandation est un modèle abstrait dans la mesure où il ne correspond pas à une réalisation directe au moyen d'une technologie donnée.

Un système d'objets fournit des services à ses clients. Le client d'un service est une entité capable de demander ce service.

6.2.2.1 Objets

Un système d'objets contient des entités réputées être des objets. Un objet est une entité encapsulée identifiable fournissant un ou plusieurs services qui peuvent être demandés par un client.

6.2.2.2 Demandes

Les clients demandent des services par l'émission de demandes. Une demande est un événement, c'est-à-dire un fait qui se manifeste à un instant donné. L'information associée à une demande se constitue d'une opération, d'un objet cible, d'un nombre supérieur ou égal à zéro de paramètres (effectifs) et d'un contexte de demande optionnel.

Une *formule de demande* est une description ou une forme qui peut être évaluée ou exécutée à plusieurs reprises afin de générer l'envoi de demandes.

Une *valeur* est tout ce qui est susceptible d'être un paramètre (effectif) valide dans une demande. Une valeur peut identifier un objet à des fins d'exécution de la demande. Une valeur identifiant un objet est appelée un nom d'objet.

Une *référence d'objet* désigne d'une manière fiable un objet donné. D'une manière spécifique, une référence d'objet identifiera le même objet chaque fois que la référence est utilisée dans une demande (avec quelques limitations pragmatiques dans l'espace et le temps). Un objet peut être désigné par des références d'objet multiples et distinctes.

Une demande peut posséder des paramètres utilisés pour transmettre des données à l'objet cible. Elle peut également posséder un contexte d'objet qui fournit une information supplémentaire au sujet de la demande.

Une demande a pour effet l'exécution d'un service au bénéfice d'un client. Une des conséquences de l'exécution d'un service est le renvoi des résultats éventuels définis pour la demande.

Une *exception* est renvoyée si une condition anormale se manifeste durant l'exécution d'une demande. L'exception peut véhiculer des paramètres de retour supplémentaires qui lui sont propres.

Les paramètres d'une demande sont identifiés par leur position. Un paramètre peut être un paramètre d'entrée, de sortie ou d'entrée/sortie. Une demande peut également fournir en retour une unique valeur de résultat en même temps que tout paramètre de sortie.

La sémantique qui suit est valable pour toutes les demandes:

- pour toute valeur d'alias d'un paramètre, il n'est pas garanti si cette valeur est préservée ou supprimée;
- l'ordre d'écriture des alias des paramètres de sortie n'est pas garanti;
- tout paramètre de sortie possède une valeur non définie si une exception est renvoyée;
- la valeur qui peut être renvoyée dans un paramètre d'entrée/sortie peut être limitée par la valeur utilisée en entrée.

6.2.2.3 Types

Un type est une entité identifiable à laquelle est associé un prédicat (fonction mathématique avec un seul argument retournant une valeur booléenne) défini pour un domaine de valeurs. Une valeur satisfait à un type si le prédicat est vrai pour cette valeur. Une valeur qui satisfait à un type est appelée *membre du type*.

Les types sont utilisés dans les signatures afin de limiter les possibilités d'un paramètre ou de caractériser les possibilités d'un résultat.

L'*étendue du type* correspond à l'ensemble de valeurs qui satisfont au type à tout instant particulier.

Un type d'objet est un type dont les membres sont des objets (littéralement, des valeurs identifiant des objets). En d'autres termes, un type d'objet ne peut être satisfait que par (des valeurs identifiant) des objets.

6.2.2.4 Interfaces

Une *interface* est une description d'un ensemble d'opérations possibles dont un client peut demander l'application à un objet. Un objet satisfait à une interface s'il peut être spécifié comme objet cible pour toute demande possible décrite dans l'interface.

Un *type d'interface* est un type qui peut être satisfait par tout objet (littéralement, toute valeur identifiant un objet) qui satisfait à une interface donnée.

6.2.2.5 Opérations

Une *opération* est une entité identifiable qui désigne un service pouvant être demandé.

Une opération est identifiée par un *identificateur d'opération*. Une opération n'est pas une valeur.

Une opération possède une signature qui décrit les valeurs légitimes des paramètres de demande et des résultats renvoyés. Une signature se constitue en particulier des éléments suivants:

- une spécification des paramètres exigés pour cette opération;
- une spécification du résultat de l'opération;
- une spécification des exceptions qui peuvent être activées par une demande pour cette opération et les types de paramètres qui les accompagnent;
- une spécification d'une information additionnelle de contexte pouvant affecter la demande;
- une indication de la sémantique d'exécution que le client doit attendre pour une demande pour l'opération.

Un *paramètre* est caractérisé par son mode et son type. Le mode indique si la valeur doit être transmise du client vers le serveur (entrée), du serveur vers le client (sortie) ou dans les deux sens (entrée-sortie). Le type du paramètre limite les valeurs possibles qui peuvent être transmises dans les directions indiquées par le mode.

Le *résultat de retour* est un paramètre de sortie qui fait l'objet d'une distinction.

Une *exception* indique qu'une demande d'opération n'a pas été exécutée avec succès. Une exception peut être accompagnée par une information supplémentaire spécifique.

Un *contexte de demande* fournit des informations supplémentaires spécifiques de l'opération qui peuvent affecter l'exécution de la demande.

Le modèle objet définit deux styles de *sémantique d'exécution*:

- au plus une fois: une demande d'opération qui renvoie un résultat de succès a été exécutée exactement une fois; si elle renvoie une indication d'exception, elle a été exécutée au moins une fois;
- meilleur effort: une opération exécutée avec le meilleur effort se constitue uniquement d'une demande, c'est-à-dire qu'elle ne peut renvoyer aucun résultat et que le demandeur ne peut pas se synchroniser sur une exécution éventuelle de la demande.

La sémantique d'exécution attendue est associée à une opération. Ceci évite qu'une implémentation de client et d'objet fasse des hypothèses de sémantiques différentes.

Il convient de noter que le client est en mesure d'invoquer une opération avec la sémantique "au moins une fois" d'une manière synchrone ou synchrone différée.

6.2.2.6 Attributs

Une interface peut posséder des attributs. Un attribut est l'équivalent logique de la déclaration d'un couple de fonctions d'accès: une fonction pour lire la valeur de l'attribut et une autre pour positionner cette valeur.

Un attribut peut être en lecture uniquement, auquel cas seule la fonction d'accès en lecture est définie.

6.2.2.7 Comparaison entre sous-types et héritage

Le sous-type est une relation entre des types, basée sur leurs interfaces, définissant les règles qui permettent à des objets d'être considérés comme acceptables dans des contextes qui attendent un autre type. L'héritage est un mécanisme permettant une réutilisation. Beaucoup de systèmes par objets ne font pas de distinction entre sous-type et héritage. Les sous-paragraphes qui suivent définissent séparément les deux concepts mais soulignent explicitement les relations qui les lient.

6.2.2.8 Sous-types

Le modèle objet prend en charge les sous-types pour des types d'objet. D'une manière intuitive, un type est un sous-type d'un autre type si le premier est une spécialisation ou un raffinement du second. Du point de vue de l'exécution, ceci signifie que tout objet du premier type peut être utilisé dans tout contexte qui attend un objet du second type, c'est-à-dire que si S est un sous-type de T, un objet de type S peut être utilisé chaque fois qu'un objet du type T est utilisable. En d'autres termes, les objets du type S sont également du type T. Les sous-types peuvent avoir de multiples parents, ce qui implique qu'un objet qui est une instance du type S est également une instance de tous les hypertypes du type S. Les relations entre types définissent une hiérarchie de types qui peut être représentée par un graphe ordonné sans cycle.

6.2.2.9 Héritage

L'héritage est un mécanisme de notation permettant de définir un type S à partir d'un autre type T. La définition de S hérite de toutes les opérations de T et peut fournir en plus d'autres opérations. D'une manière intuitive, l'héritage signifie que les opérations définies pour T sont également définies pour S et utilisables par lui. L'héritage s'applique aussi bien aux interfaces qu'aux implémentations, ce qui signifie qu'elles peuvent être héritées toutes deux. Le modèle objet traite de l'héritage des interfaces. Il ne spécifie pas ce qui se passe dans le cas d'implémentation d'opérations héritées (par exemple, si elles peuvent être modifiées ou écrasées par un sous-type).

6.2.3 Principes de mappage des interfaces IDL avec les primitives API

L'interface du moteur MHEG se constitue d'un ensemble de primitives d'interface API qui peuvent être organisées en groupe en fonction de leurs entités cibles. La définition de cette interface peut en conséquence être structurée logiquement en fonction du fournisseur de l'opération. Dans le contexte de l'interface API MHEG, les objets qui fournissent des interfaces ne doivent pas nécessairement être réalisés sous la forme d'implémentations distinctes. Il s'agira plus probablement d'entités internes traitées par le moteur MHEG. En ce qui concerne la norme MHEG, la définition de l'interface API MHEG est faite selon une méthodologie par objets sans exiger que les implémentations utilisent des techniques de conception ou de programmation par objets.

Les services ne sont pas définis comme des interfaces devant être fournies par un module, mais comme un ensemble d'opérations qui coopèrent pour fournir un service auquel les clients doivent avoir accès.

L'interface API MHEG se constitue en conséquence d'objets d'interfaces IDL qui fournissent des opérations correspondant aux primitives de l'interface API. L'instance d'un objet pour lequel une opération est demandée correspond au paramètre principal (cible) de la primitive de l'interface API.

6.2.4 Satisfaction aux exigences techniques

L'utilisation du langage IDL contribue à satisfaire aux exigences techniques de portabilité et de faisabilité d'implémentation:

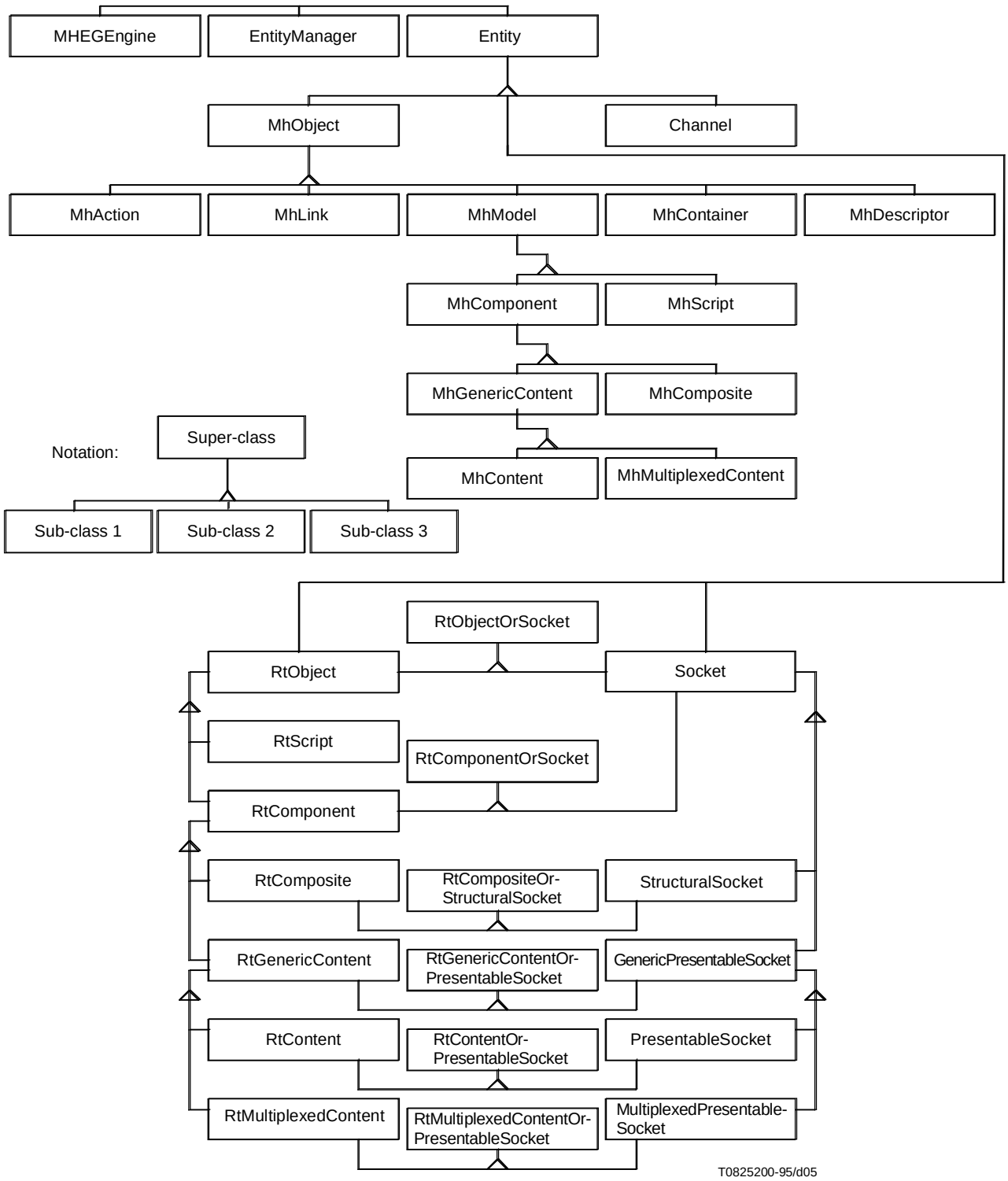
- le langage IDL est indépendant de tout langage de programmation. Des spécifications disponibles publiquement existent pour les liaisons avec les langages C et C++ et d'autres sont à l'étude;
- le langage IDL fournit un langage de description formel complet qui permet une spécification de l'interface API MHEG très précise, lisible et efficace. Ce langage de description formel est en outre également approprié pour une compilation automatique, ce qui signifie que les implémentations de l'interface API MHEG peuvent être générées d'une manière automatique pour un langage et un système d'exploitation donné en utilisant un compilateur IDL adéquat. Ceci peut constituer un facteur essentiel de nature à faciliter l'implémentation et la généralisation de l'utilisation de l'interface API normalisée de préférence à toute interface spécifique.

6.3 Aperçu général de la définition de l'interface API et principes généraux

6.3.1 Le modèle objet de l'interface API MHEG

Le présent sous-paragraphe présente le modèle objet, c'est-à-dire les types d'objets (interface) fournis par l'interface API MHEG et les relations de dérivation de sous-types.

Il peut être noté que les objets décrits ci-dessous sont introduits comme des concepts utiles pour la spécification de l'interface, mais que leur implémentation sous la forme d'objets distincts n'est pas exigée. L'interface API MHEG est spécifiée comme interface API abstraite en termes d'opérations fournies par des objets, mais les implémentations de l'interface API MHEG seront fournies par des implémentations de moteur MHEG.



NOTE 1 – Les objets MhObjects (et leurs sous-types) sont de la forme b) telle qu'elle est décrite au 6.2.4 de l'ISO/CEI 13522-1 "Technologies de l'information – Codage de l'information multimédia et hypermédia" [1], c'est-à-dire qu'il s'agit d'objets disponibles pour le moteur MHEG.

NOTE 2 – Les objets RtObjects (et leurs sous-types) sont de la forme c) telle qu'elle est décrite au 6.2.4 de l'ISO/CEI 13522-1 "Technologies de l'information – Codage de l'information multimédia et hypermédia" [1], c'est-à-dire qu'il s'agit d'instances d'objets MhObject disponibles pour le processus de présentation.

Figure 5/T.174 – Modèle objet

7 Définition de l'interface API MHEG

7.1 Primitives obligatoires

7.1.1 Objet `MHEGEngine`

Les sous-paragraphes qui suivent définissent les opérations de l'objet `MHEGEngine`.

7.1.1.1 Opération `initialiseEngine`

Résumé:

Interface: `MHEGEngine`
Opération: `initialiseEngine`
Résultat: `void`

Description:

Cette opération exécute toute initialisation nécessaire pour l'interface. Elle sera invoquée avant l'invocation de toute autre opération d'interface définie dans la présente Recommandation. Elle peut être invoquée de multiples fois, auquel cas chaque invocation réinitialisera le moteur MHEG.

7.1.1.2 Opération `shutdownEngine`

Résumé:

Interface: `MHEGEngine`
Opération: `shutdownEngine`
Résultat: `void`

Description:

Cette opération supprime tous les objets d'interface générés par un service, qui sont associés à la session en cours. L'opération suivante qui sera acceptée par un moteur MHEG sera une opération `initialiseEngine`.

7.1.1.3 Description IDL

```
interface MHEGEngine    {

     void    initialiseEngine();
     void    shutdownEngine();

};
```

7.1.2 Objet `NotificationManager`

Le sous-paragraphe qui suit définit les opérations de l'objet `NotificationManager`.

7.1.2.1 Opération `getReturnability`

Résumé:

Interface: `NotificationManager`
Opération: `getReturnability`
Résultat: `sequence<unsigned short>`

Description:

Cette opération extrait le comportement de retour du moteur MHEG.

L'opération renvoie une liste de nombres identifiant les notifications disponibles.

7.1.2.2 Opération `getNotification`

Résumé:

Interface:	`NotificationManager`	
Opération:	`getNotification`	
Résultat:	`void`	
Entrée:	`unsigned short`	`notification_number`
Sortie:	`sequence<GenericValue>`	`values`
Sortie:	`sequence<MhObjectReference>`	`objects`
Exception:	`InvalidParameter`	

Description:

Cette opération extrait une notification du moteur MHEG.

Le paramètre `notification_number` identifie la notification. Cette identification peut être le résultat d'une opération `getReturnability`.

Le paramètre `values` spécifie les valeurs renvoyées.

Le paramètre `objects` spécifie les références d'objets renvoyées.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.2.3 Description IDL

```
interface    NotificationManager

     sequence<unsigned short>
         getReturnability();

     void
         getNotification(
            in unsigned short
                notification_number,
            out sequence<GenericValue>
                values,
            out sequence<MhObjectReference>
                objects)
     raises(InvalidParameter);

};
```

7.1.3 Objet `EntityManager`

Les sous-paragraphes qui suivent définissent les opérations de l'objet `EntityManager`.

7.1.3.1 Opération `getAvailableMhObjects`

Résumé:

Interface:	`EntityManager`
Opération:	`getAvailableMhObjects`
Résultat:	`sequence<MHEGIdentifier>`

Description:

Cette opération extrait les mh-objets disponibles pour le moteur MHEG.

Un mh-objet se trouve dans l'un des états "non prêt" (dans la période O1), "en traitement" (dans la période O2 ou O4) ou "prêt" (dans la période O3). Cette opération extrait ceux des mh-objets qui se trouvent en période O3.

L'opération renvoie les identificateurs des mh-objets disponibles.

7.1.3.2 Opération `getAvailableRtObjects`

Résumé:

Interface:	`EntityManager`
Opération:	`getAvailableRtObjects`
Résultat:	`sequence<RtObjectIdentifier>`

Description:

Cette opération extrait les rt-objets disponibles pour le moteur MHEG.

Un rt-objet se trouve dans l'un des états "disponible" (dans la période R1), "en traitement" (dans la période R2 ou R4) ou "disponible" (dans la période R3). Cette opération extrait ceux des rt-objets qui se trouvent en période R3.

L'opération renvoie les identificateurs des rt-objets disponibles.

7.1.3.3 Opération `getAvailableChannels`

Résumé:

Interface:	`EntityManager`
Opération:	`getAvailableChannels`
Résultat:	`sequence<ChannelIdentifier>`

Description:

Cette opération extrait les canaux disponibles pour le moteur MHEG.

Un canal se trouve dans l'un des états "disponible" (dans la période C1), "en traitement" (dans la période C2 ou C4) ou "disponible" (dans la période C3). Cette opération extrait ceux des canaux qui se trouvent en période C3.

L'opération renvoie les identificateurs des canaux disponibles.

7.1.3.4 Opération `releaseAlias`

Résumé:

Interface:	`EntityManager`
Opération:	`releaseAlias`
Résultat:	`void`
Entrée:	`string alias`
Exception:	`InvalidParameter`

Description:

Cette opération permet de libérer un alias. Elle supprime les allocations de cet alias à des entités.

Le paramètre `alias` spécifie la valeur de l'alias libéré.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.3.5 Description IDL

```
interface    EntityManager                          {

     sequence<MHEGIdentifier>
         getAvailableMhObjects();

     sequence<RtObjectIdentifier>
         getAvailableRtObjects();

     sequence<ChannelIdentifier>
         getAvailableChannels();

     void
         release Alias(
             in string
                 alias)
     raises(InvalidParameter);

};
```

7.1.4 Objet `Entity`

Le sous-paragraphe qui suit définit les opérations de l'objet `Entity`.

7.1.4.1 Opération `setAlias`

Résumé:

Interface:	`Entity`
Opération:	`setAlias`
Résultat:	`void`
Entrée:	`string alias`
Exception:	`InvalidTarget`

Description:

Cette opération permet l'assignation d'un alias à une entité.

L'opération `setAlias` déclenche l'exécution de l'action élémentaire "positionner alias" avec comme cible unique l'entité liée.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 34.2.1 de la norme MHEG.

Le paramètre `alias` spécifie la valeur du paramètre "alias" de l'action "positionner alias".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.4.2 Opération `getAlias`

Résumé:

Interface:	`Entity`
Opération:	`getAlias`
Résultat:	`string`
Exception:	`InvalidTarget`

Description:

Cette opération extrait l'alias assigné à une entité.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.4.3 Description IDL

```
interface Entity  {
    void
        setAlias(
            in string
                alias)
    raises(InvalidTarget);

    string
        getAlias()
    raises(InvalidTarget);

};
```

7.1.5 Objet `MhObject`

Le sous-paragraphe qui suit définit les opérations de l'objet `MhObject`. Cet objet est un héritage de l'objet `Entity`.

7.1.5.1 Opération `bind`

Résumé:

Interface:	`MhObject`	
Opération:	`bind`	
Résultat:	`MHEGIdentifier`	
Entrée:	`MhObjectReference`	`mh_object_reference`
Exception:	`AlreadyBound`	
Exception:	`InvalidTarget`	

Description:

Cette opération lie l'instance MhObject (une instance d'objet interface) à un objet MHEG (une entité MHEG).

Le paramètre `mh_object_reference` spécifie la référence de l'objet MHEG.

Cette opération renvoie l'identificateur de l'objet MHEG lié.

L'exception `AlreadyBound` est activée lorsque l'instance d'objet d'interface est déjà liée à une entité MHEG.

L'exception `InvalidTarget` est activée lorsque l'entité MHEG cible n'est pas disponible. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.5.2 Opération `unbind`

Résumé:

Interface:	`MhObject`
Opération:	`unbind`
Résultat:	`void`
Exception:	`NotBound`

Description:

Cette opération annule la liaison entre l'instance de Mh-objet (une instance d'objet interface) et un objet MHEG (une entité MHEG).

L'exception `NotBound` est activée lorsque l'instance d'objet d'interface n'est pas liée à l'entité MHEG.

7.1.5.3 Opération `prepare`

Résumé:

Interface:	`MhObject`	
Opération:	`prepare`	
Résultat:	`MHEGIdentifier`	
Entrée:	`MhObjectReference`	`mh_object_reference`
Exception:	`AlreadyBound`	
Exception:	`InvalidTarget`	

Description:

Cette opération permet la création par le moteur MHEG d'un objet MHEG à partir d'un objet modèle.

L'opération `prepare` déclenche l'exécution de l'action élémentaire "préparer" avec pour cible unique un objet MHEG.

L'effet de l'action sur la cible et les conditions d'erreur qui conduisent à l'activation d'exceptions sont définis au 36.2.1 de la norme MHEG.

Le paramètre `mh_object_reference` spécifie une référence à un objet MHEG.

Cette opération lie implicitement l'instance MhObject (une instance d'objet interface) avec le nouvel objet MHEG préparé (une entité MHEG).

L'opération renvoie l'identificateur du nouvel objet MHEG préparé, lié à l'instance MhObject.

L'exception `AlreadyBound` est activée lorsque l'instance d'objet d'interface est déjà liée à une entité MHEG.

L'exception `InvalidTarget` est activée lorsque l'entité cible MHEG n'est pas disponible. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.5.4 Opération `destroy`

Résumé:

Interface:	`MhObject`
Opération:	`destroy`
Résultat:	`void`
Exception:	`NotBound`
Exception:	`InvalidTarget`

Description:

Cette opération permet la suppression d'un objet MHEG par le moteur MHEG.

L'opération `destroy` déclenche l'exécution de l'action élémentaire "destruction" avec comme cible un objet MHEG unique.

L'effet de l'action sur la cible et les conditions d'erreur qui conduisent à l'activation d'exceptions sont définis au 36.2.2 de la norme MHEG.

Cette opération supprime implicitement la liaison entre l'instance MhObject (une instance d'objet interface) et l'objet MHEG nouvellement détruit (une entité MHEG).

L'exception `NotBound` est activée lorsque l'instance d'objet d'interface n'est pas liée à une entité MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas disponible. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.5.5 Opération `getPreparationStatus`

Résumé:

Interface: `MhObject`
Opération: `getPreparationStatus`
Résultat: `PreparationStatusValue`
Exception: `NotBound`
Exception: `InvalidTarget`

Description:

Cette opération extrait la disponibilité d'un objet MHEG pour le moteur MHEG.

L'opération `getPreparationStatus` déclenche l'exécution de l'action élémentaire "extraire statut de préparation" avec pour cible unique l'objet MHEG lié.

L'effet de l'action sur la cible et les conditions d'erreur qui conduisent à l'activation d'exceptions sont définis au 36.3.1 de la norme MHEG.

L'opération renvoie la disponibilité de l'objet MHEG lié à l'instance MhObject. La valeur renvoyée est soit `NOT_READY`, `PROCESSING` ou `READY`.

Lorsque la valeur renvoyée est `NOT_READY,` l'opération supprime implicitement la liaison de l'instance MhObject (une instance d'objet interface) avec l'objet MHEG (une entité MHEG).

L'exception `NotBound` est activée lorsque l'instance d'objet d'interface n'est pas liée à une entité MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.5.6 Opération `getIdentifier`

Résumé:

Interface: `MhObject`
Opération: `getIdentifier`
Résultat: `MHEGIdentifier`
Exception: `NotBound`

Description:

Cette opération extrait l'identificateur de l'objet MHEG (une entité MHEG) lié à l'instance MhObject (une instance d'objet interface)

L'exception `NotBound` est activée lorsque l'instance d'objet d'interface n'est pas liée à une entité MHEG.

7.1.5.7 Opération `kill`

Résumé:

Interface: `MhObject`
Opération: `kill`
Résultat: `void`

Description:

Cette opération supprime l'instance MhObject (une instance d'objet interface).

7.1.5.8 Description IDL

```
interface    MhObject: Entity                    {

    MHEGIdentifier
        bind(
            in MhObjectReference
                mh_object_reference)
    raises(AlreadyBound, InvalidTarget);
```

```
    void
        unbind()

    raises(NotBound);

    MHEGIdentifier
        prepare(
            in MhObjectReference
                mh_object_reference)
    raises(AlreadyBound, InvalidTarget);

    void
        destroy()
    raises(NotBound, InvalidTarget);

    PreparationStatusValue
        getPreparationStatus()
    raises(NotBound, InvalidTarget);

    MHEGIdentifier
        getIdentifier()
    raises(NotBound);

    void
        kill();

};
```

7.1.6 Objet `MhAction`

Le sous-paragraphe qui suit définit les opérations de l'objet `MhAction.` Cet objet est un héritage de l'objet `MhObject.`

7.1.6.1 Opération `delay`

Résumé:

Interface:	`MhAction`	
Opération:	`delay`	
Résultat:	`void`	
Entrée:	`unsigned short`	`nested_action_number`
Entrée:	`unsigned long`	`delay`
Exception:	`InvalidTarget`	
Exception:	`InvalidParameter`	

Description:

Cette opération permet de différer le traitement d'actions imbriquées au sein de la mh-action.

Le paramètre `nested_action_number` spécifie l'action imbriquée après laquelle le traitement différé doit prendre place.

Le paramètre `delay` spécifie la durée de la suspension exprimée en unités de temps générique (GTU, *generic temporal aunit*).

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.6.2 Description IDL

```
interface    MhAction: MhObject                          {

    void
        delay(
           in unsigned short
               nested_action_number,
           in unsigned long
               delay)
    raises(InvalidTarget, InvalidParameter);

};
```

7.1.7 Objet `MhLink`

Le sous-paragraphe qui suit définit les opérations de l'objet `MhLink`. Cet objet est un héritage de l'objet `MhObject`.

7.1.7.1 Opération `abort`

Résumé:

Interface: `MhLink`
Opération: `abort`
Résultat: `void`
Exception: `InvalidTarget`

Description:

Cette opération interrompt le traitement de toutes les actions qui ont été activées par un objet lien. Les actions qui définissent le lien sont activées et traitées pour chaque occurrence de la condition de lien.

L'opération `abort` déclenche l'exécution de l'action élémentaire "interruption" avec comme cible unique l'objet lié par le lien.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 38.2.1 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.7.2 Description IDL

```
interface MhLink: MhObject                                    {

    void
        abort()
    raises(InvalidTarget);

};
```

7.1.8 Objet `MhModel`

Aucune opération spécifique n'est définie pour l'objet `MhModel`. Cet objet est un héritage de l'objet `MhObject`.

7.1.8.1 Description IDL

```
interface MhModel: MhObject {};
```

7.1.9 Objet `MhComponent`

Aucune opération spécifique n'est définie pour l'objet `MhComponent`. Cet objet est un héritage de l'objet `MhModel`.

7.1.9.1 Description IDL

```
interface MhComponent: MhModel {};
```

7.1.10 Objet `MhGenericContent`

Le sous-paragraphe qui suit définit les opérations de l'objet `MhGenericContent`. Cet objet est un héritage de l'objet `MhComponent`.

7.1.10.1 Opération `copy`

Résumé:

Interface: `MhContent`
Opération: `copy`
Résultat: `void`
Entrée: `sequence<MhObjectReference>  copies`
Exception: `InvalidTarget`
Exception: `InvalidParameter`

Description:

Cette opération spécifie la copie d'un objet contenu "source" dans un ensemble d'objets contenu "copies" ou la copie d'un objet contenu multiplexé "source" dans un ensemble d'objets contenu multiplexés "copies".

L'opération `copy` déclenche l'exécution de l'action élémentaire "copier" avec comme cible unique l'objet contenu lié ou l'objet contenu multiplexé.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 40.2.1 de la norme MHEG.

Le paramètre `copies` spécifie la valeur du paramètre "copies" de l'action "copier".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.10.2 Description IDL

```
interface MhGenericContent: MhComponent                    {

    void
        copy(
            in sequence<MhObjectReference>
                  copies)
    raises(InvalidTarget, InvalidParameter);

};
```

7.1.11 Objet `MhContent`

Le sous-paragraphe qui suit définit les opérations de l'objet `MhContent`. Cet objet est un héritage de l'objet `MhGenericContent`.

7.1.11.1 Opération `setData`

Résumé:

Interface:	`MhContent`	
Opération:	`setData`	
Résultat:	`void`	
Entrée:	`boolean`	`substitution_indicator`
Entrée:	`sequence<DataElement>`	`data_elements`
Exception:	`InvalidTarget`	
Exception:	`InvalidParameter`	

Description:

Cette opération permet de stocker ou de modifier la valeur générique dans des données d'un objet contenu.

L'opération `setData` déclenche l'exécution de l'action élémentaire "positionner données" avec comme cible unique l'objet contenu lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 39.2.1 de la norme MHEG.

Le paramètre `substitution_indicator` spécifie la valeur du paramètre "indicateur de substitution" de l'action "positionner données".

Le paramètre `data_elements` spécifie la valeur du paramètre "éléments de données" de l'action "positionner données".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.11.2 Opération `getData`

Résumé:

Interface:	`MhContent`
Opération:	`getData`
Résultat:	`GenericValue`
Entrée:	`sequence<long>` `element_list_index`
Exception:	`InvalidTarget`
Exception:	`InvalidParameter`

Description:

Cette opération extrait une valeur générique ou un élément d'une liste générique stocké dans les données d'un objet contenu.

L'opération `getData` déclenche l'exécution de l'action élémentaire "extraire données" avec comme cible unique l'objet contenu lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 39.3.1 de la norme MHEG.

L'index `element_list_index` spécifie la valeur du paramètre "index de liste d'éléments" de l'action "extraire données".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet ne constitue pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.11.3 Description IDL

```
interface MhContent: MhGenericContent                    {

    void
        setData(
            in boolean
               substitution_indicator,
            in sequence<DataElement>
                data_elements)
    raises(InvalidTarget, InvalidParameter);

    GenericValue
        getData(
            in sequence<long>
                element_list_index)
    raises(InvalidTarget, InvalidParameter);

};
```

7.1.12 Objet `MhMultiplexedContent`

Le sous-paragraphe qui suit définit les opérations de l'objet `MhMultiplexedContent`. Cet objet est un héritage de l'objet `MhGenericContent`.

7.1.12.1 Opération `setMultiplex`

Résumé:

Interface:	`MhMultiplexedContent`
Opération:	`setMultiplex`
Résultat:	`void`
Entrée:	`sequence<StreamIdentifier>` `stream_list`
Exception:	`InvalidTarget`
Exception:	`InvalidParameter`

Description:

Cette opération spécifie le multiplexage d'une liste d'objets contenu, le résultat étant placé dans un objet contenu multiplexé contenant les données multiplexées.

L'opération `setMultiplex` déclenche l'exécution de l'action élémentaire "positionner multiplex" avec comme cible unique l'objet contenu multiplexé lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 41.2.1 de la norme MHEG.

Le paramètre `stream_list` spécifie la valeur du paramètre "liste de flux" de l'action "positionner multiplex".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.12.2 Opération `setDemultiplex`

Résumé:

Interface:	`MhMultiplexedContent`
Opération:	`setDemultiplex`
Résultat:	`void`
Entrée:	`sequence<StreamIdentifier>  stream_list`
Exception:	`InvalidTarget`
Exception:	`InvalidParameter`

Description:

Cette opération spécifie le démultiplexage d'un flux de données multiples d'un objet contenu multiplexé, par exemple un flux "groupe d'experts pour les images mobiles" (MPEG, *moving picture expert group*), le résultat étant placé dans une liste d'objets contenu qui sont générés s'ils n'existent pas déjà. Chaque objet contenu contient un flux démultiplexé.

L'opération `setDemultiplex` déclenche l'exécution de l'action élémentaire "positionner démultiplex" avec comme cible unique l'objet contenu multiplexé lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 41.2.2 de la norme MHEG.

La liste `stream_list` spécifie la valeur du paramètre "liste de flux" de l'action "positionner démultiplex".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.12.3 Description IDL

```
interface MhMultiplexedContent: MhGenericContent           {

    void
        setMultiplex(
           in sequence<StreamIdentifier>
               stream_list)
    raises(InvalidTarget, InvalidParameter);

    void
        setDemultiplex(
           in sequence<StreamIdentifier>
               stream_list)
    raises(InvalidTarget, InvalidParameter);

};
```

7.1.13 Objet `MhComposite`

Aucune opération spécifique n'est définie pour l'objet `MhComposite`. Cet objet est un héritage de l'objet `MhComponent`.

7.1.13.1 Description IDL

```
interface MhComposite: MhComponent {};
```

7.1.14 Objet `MhScript`

Aucune opération spécifique n'est définie pour l'objet `MhScript`. Cet objet est un héritage de l'objet `MhModel`.

7.1.14.1 Description IDL

```
interface MhScript: MhModel {};
```

7.1.15 Objet `MhContainer`

Aucune opération spécifique n'est définie pour l'objet `MhContainer`. Cet objet est un héritage de l'objet `MhObject`.

7.1.15.1 Description IDL

```
interface MhContainer: MhObject {};
```

7.1.16 Objet `MhDescriptor`

Aucune opération spécifique n'est définie pour l'objet `MhDescriptor`. Cet objet est un héritage de l'objet `MhObject`.

7.1.16.1 Description IDL

```
interface MhDescriptor: MhObject {};
```

7.1.17 Objet `RtObjectOrSocket`

Le sous-paragraphe qui suit définit les opérations de l'objet `RtObjectOrSocket`.

7.1.17.1 Opération `setGlobalBehaviour`

Résumé:

Interface:	`RtObject`	
Opération:	`setGlobalBehaviour`	
Résultat:	`void`	
Entrée:	`GlobalBehaviour`	`global_behaviour`
Exception:	`InvalidTarget`	
Exception:	`InvalidParameter`	

Description:

Cette opération permet la modification du comportement global d'un rt-objet ou d'un réceptacle.

L'opération `setGlobalBehaviour` déclenche l'exécution de l'action élémentaire "positionner comportement global" avec comme cible unique le rt-objet ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 44.2.1 de la norme MHEG.

Le paramètre `global_behaviour` spécifie la valeur du paramètre "comportement global" de l'action "positionner comportement global".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.17.2 Opération `getGlobalBehaviour`

Résumé:

Interface:	`RtObject`
Opération:	`getGlobalBehaviour`
Résultat:	`GenericValue`
Exception:	`InvalidTarget`

Description:

Cette opération extrait toutes les valeurs d'attribut qui constituent le comportement global de chaque rt-objet ou de chaque réceptacle du moteur MHEG.

L'opération `getGlobalBehaviour` déclenche l'exécution de l'action élémentaire "extraire comportement global" avec comme cible unique l'objet rt-objet ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 44.3.1 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.17.3 Opération `run`

Résumé:

Interface: `RtObject`
Opération: `run`
Résultat: `void`
Exception: `InvalidTarget`

Description:

Cette opération permet l'activation du rt-objet ou du réceptacle par le processus en cours.

L'opération `run` déclenche l'exécution de l'action élémentaire "exécuter" avec comme cible unique le rt-objet ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 45.2.1 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.17.4 Opération `stop`

Résumé:

Interface: `RtObject`
Opération: `stop`
Résultat: `void`
Exception: `InvalidTarget`

Description:

Cette opération retire le rt-objet du processus en cours.

L'opération `stop` déclenche l'exécution de l'action élémentaire "stop" avec comme cible unique le rt-objet lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 45.2.2 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.17.5 Description IDL

```
interface RtObjectOrSocket                                    {

    void
        setGlobalBehaviour(
            in GlobalBehaviour
                global_behaviour)
    raises(InvalidTarget, InvalidParameter);

    GenericValue
        getGlobalBehaviour()
    raises(InvalidTarget);

    void
        run()
    raises(InvalidTarget);

    void
        stop()
    raises(InvalidTarget);

};
```

7.1.18 Objet `RtObject`

Le sous-paragraphe qui suit définit les opérations de l'objet `RtObject`. Cet objet est un héritage de l'objet `RtObjectOrSocket` et de l'objet `Entity`.

7.1.18.1 Opération `bind`

Résumé:

Interface:	`RtObject`	
Opération:	`bind`	
Résultat:	`RtObjectIdentifier`	
Entrée:	`RtObjectReference`	`rt_object_reference`
Exception:	`AlreadyBound`	
Exception:	`InvalidTarget`	

Description:

Cette opération lie l'instance de l'objet RtObject (une instance d'objet interface) à un rt-objet (une entité MHEG).

Le paramètre `rt_object_reference` spécifie la référence du rt-objet.

L'opération renvoie l'identificateur du rt-objet lié.

L'exception `AlreadyBound` est activée lorsque l'instance d'objet d'interface est déjà liée à une entité MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.18.2 Opération `unbind`

Résumé:

Interface:	`RtObject`
Opération:	`unbind`
Résultat:	`void`
Exception:	`NotBound`

Description:

Cette opération supprime la liaison entre l'instance RtObject (une instance d'objet interface) et un rt-objet (une entité MHEG).

L'exception `NotBound` est activée lorsque l'instance d'objet d'interface n'est pas liée à une entité MHEG.

7.1.18.3 Opération `new`

Résumé:

Interface:	`RtObject`	
Opération:	`new`	
Résultat:	`RtObjectIdentifier`	
Entrée:	`RtObjectReference`	`rt_object_reference`
Exception:	`AlreadyBound`	
Exception:	`InvalidTarget`	

Description:

Cette opération permet la création par le moteur MHEG d'un rt-objet à partir d'un objet modèle.

L'opération `new` déclenche l'exécution de l'action élémentaire "nouveau" avec comme cible unique un rt-objet.

L'effet de l'action sur la cible et les conditions d'erreur qui activent des exceptions sont définis au 43.2.1 de la norme MHEG.

Le paramètre `rt_object_reference` spécifie une référence à un rt-objet.

Cette opération lie implicitement l'instance RtObject (une instance d'objet interface) avec le rt-objet (une entité MHEG) nouvellement créé.

L'opération renvoie l'identificateur du rt-objet nouvellement créé qui est lié à l'instance RtObject.

L'exception `AlreadyBound` est activée lorsque l'instance d'objet d'interface est déjà liée à une entité MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.18.4 Opération `delete`

Résumé:

Interface:	`RtObject`
Opération:	`delete`
Résultat:	`void`
Exception:	`NotBound`
Exception:	`InvalidTarget`

Description:

Cette opération permet la suppression d'un rt-objet par le moteur MHEG.

L'opération `delete` déclenche l'exécution de l'action élémentaire "supprimer" avec comme cible unique un rt-objet.

L'effet de l'action sur la cible et les conditions d'erreur qui conduisent à l'activation d'exceptions sont définis au 43.2.2 de la norme MHEG.

Cette opération supprime implicitement la liaison de l'instance RtObject (une instance d'objet interface) avec un rt-objet (une entité MHEG) nouvellement supprimé.

L'exception `NotBound` est activée lorsque l'instance d'objet d'interface n'est pas liée à une entité MHEG.

L'exception `InvalidTarget` est activée lorsque l'entité MHEG cible n'est pas disponible. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.18.5 Opération `getAvailabilityStatus`

Résumé:

Interface:	`RtObject`
Opération:	`getAvailabilityStatus`
Résultat:	`RtAvailabilityStatusValue`
Exception:	`NotBound`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la disponibilité du rt-objet pour le moteur MHEG.

L'opération `getAvailabilityStatus` déclenche l'exécution de l'action élémentaire "extraire statut de disponibilité rt" avec comme cible unique le rt-objet lié.

L'effet de l'action sur la cible, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 43.3.1 de la norme MHEG.

L'opération renvoie la disponibilité du rt-objet lié à l'instance RtObject. La valeur renvoyée est soit `NOT_AVAILABLE`, `PROCESSING` ou `AVAILABLE`.

Si la valeur renvoyée est `NOT_AVAILABLE`, l'opération supprime implicitement la liaison de l'instance RtObject (une instance d'objet interface) avec le rt-objet (une entité MHEG).

L'exception `NotBound` est activée lorsque l'instance d'objet d'interface n'est pas liée à une entité MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.18.6 Opération `getIdentifier`

Résumé:

Interface:	`RtObject`
Opération:	`getIdentifier`
Résultat:	`RtObjectIdentifier`
Exception:	`NotBound`

Description:

Cette opération extrait l'identificateur du rt-objet (une entité MHEG) lié à l'instance RtObject (une instance d'objet interface).

L'exception `NotBound` est activée lorsque l'instance d'objet d'interface n'est pas liée à une entité MHEG.

7.1.18.7 Opération `kill`

Résumé:

Interface: `RtObject`
Opération: `kill`
Résultat: `void`

Description:

Cette opération supprime l'instance RtObject (une instance d'objet interface).

7.1.18.8 Opération `getRunningStatus`

Résumé:

Interface: `RtObject`
Opération: `getRunningStatus`
Résultat: `RunningStatusValue`
Exception: `InvalidTarget`

Description:

Cette opération extrait le statut d'activation par le moteur MHEG de chaque rt-objet et de chaque réceptacle.

L'opération `getRunningStatus` déclenche l'exécution de l'action élémentaire "extraire statut d'exécution" avec comme cible unique le rt-objet ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 45.3.1 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.18.9 Description IDL

```
interface RtObject: Entity                                    {

    RtObjectIdentifier
        bind(
            in RtObjectReference
                rt_object_reference)
    raises(AlreadyBound, InvalidTarget);

    void
       unbind()
    raises(NotBound);

    RtObjectIdentifier
        new(
           in RtObjectReference
               rt_object_reference)
    raises(AlreadyBound, InvalidTarget);

    void
       delete()
    raises(NotBound, InvalidTarget);

    RtAvailabilityStatusValue
        getAvailabilityStatus()
    raises(NotBound, InvalidTarget);

    RtObjectIdentifier
        getIdentifier()
    raises(NotBound);

    void
       kill();
```

```
    RunningStatusValue
        getRunningStatus()
    raises(InvalidTarget);
};
```

7.1.19 Objet `Socket`

Le sous-paragraphe qui suit définit les opérations de l'objet `Socket`. Cet objet est un héritage de l'objet `RtObjectOrSocket` et de l'objet `Entity`.

7.1.19.1 Opération `bind`

Résumé:

Interface:	`Socket`	
Opération:	`bind`	
Résultat:	`SocketIdentification`	
Entrée:	`SocketReference`	`socket_reference`
Exception:	`AlreadyBound`	
Exception:	`InvalidTarget`	

Description:

Cette opération lie l'instance Socket (une instance d'objet interface) avec un réceptacle (une entité MHEG).

Le paramètre `socket_reference` spécifie la référence du réceptacle.

L'opération renvoie l'identificateur du réceptacle lié.

L'exception `AlreadyBound` est activée lorsque l'instance d'objet d'interface est déjà liée à une entité MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet MHEG cible n'est pas disponible. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.19.2 Opération `unbind`

Résumé:

Interface:	`Socket`
Opération:	`unbind`
Résultat:	`void`
Exception:	`NotBound`

Description:

Cette opération supprime le lien entre l'instance Socket (une instance d'objet interface) et un réceptacle (une entité MHEG).

L'exception `NotBound` est activée lorsque l'instance d'objet d'interface n'est pas liée à une entité MHEG.

7.1.19.3 Opération `getIdentifier`

Résumé:

Interface:	`Socket`
Opération:	`getIdentifier`
Résultat:	`SocketIdentification`

Description:

Cette opération extrait l'identificateur du réceptacle (une entité MHEG) lié à l'instance Socket (une instance d'objet interface).

L'exception `NotBound` est activée lorsque l'instance d'objet d'interface n'est pas liée à une entité MHEG.

7.1.19.4 Opération `kill`

Résumé:

Interface:	`Socket`
Opération:	`kill`
Résultat:	`void`

Description:

Cette opération supprime l'instance Socket (une instance d'objet interface).

7.1.19.5 Opération `plug`

Résumé:

Interface:	`Socket`	
Opération:	`plug`	
Résultat:	`void`	
Entrée:	`PlugIn`	`plug_in`
Exception:	`InvalidTarget`	

Description:

Cette opération permet la mise en œuvre dynamique de la présentation et de la structure. Cette opération spécifie l'information à enficher dans un réceptacle. Ce procédé est utilisé pour obtenir une présentation ou une structure différente à partir d'un même modèle d'objet composite.

L'opération `plug` déclenche l'exécution de l'action élémentaire "enficher" avec comme cible unique le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 48.2.1 de la norme MHEG.

Le paramètre `plug_in` spécifie la valeur du paramètre "enficher dans" de l'action "enficher".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.19.6 Opération `setVisibleDurationPosition`

Résumé:

Interface:	`Socket`	
Opération:	`setVisibleDurationPosition`	
Résultat:	`void`	
Entrée:	`VisibleDurationPosition`	`visible_duration_position`
Exception:	`InvalidTarget`	
Exception:	`InvalidParameter`	

Description:

Cette opération spécifie la position dans laquelle attacher la durée visible d'un réceptacle à l'intérieur de la durée perceptible du parent.

L'opération `setVisibleDurationPosition` déclenche l'exécution de l'action élémentaire "positionner la position de la durée visible" avec comme cible unique le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 52.2.4 de la norme MHEG.

Le paramètre `visible_duration_position` spécifie la valeur du paramètre "position temporelle spatiale générique relative du parent" de l'action "positionner la position de la durée visible".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.19.7 Opération `getVisibleDurationPosition`

Résumé:

Interface:	`Socket`
Opération:	`getVisibleDurationPosition`
Résultat:	`unsigned long`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la valeur de la position de la durée visible du réceptacle au sein de l'espace générique relatif de son parent. Cette valeur est exprimée en unités temporelles génériques relatives.

L'opération `getVisibleDurationPosition` déclenche l'exécution de l'action élémentaire "extraire position de durée visible" avec comme cible unique le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 52.3.7 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.19.8 Description IDL

```
interface Socket: Entity, RtObjectOrSocket                {

    SocketIdentification
        bind(
            in SocketReference
               socket_reference)
    raises(AlreadyBound, InvalidTarget);

    void  unbind()
        raises(NotBound);

    SocketIdentification
        getIdentifier();

    void
        kill();

    void
        plug(
            in PlugIn
                plug_in)
    raises(InvalidTarget);

    void
        setVisibleDurationPosition(
            in VisibleDurationPosition
                visible_duration_position)
    raises(InvalidTarget, InvalidParameter);

    unsigned long
        getVisibleDurationPosition()
    raises(InvalidTarget);

};
```

7.1.20 Objet `RtScript`

Le sous-paragraphe qui suit définit les opérations de l'objet `RtScript`. Cet objet est un héritage de l'objet `RtObject`.

7.1.20.1 Opération `setParameters`

Résumé:

Interface:	`RtScript`
Opération:	`setParameters`
Résultat:	`void`
Entrée:	`sequence<Parameter>  parameters`
Exception:	`InvalidTarget`

Description:

Cette opération permet de transmettre des paramètres entre des rt-scripts et d'autres entités MHEG.

L'opération `setParameters` déclenche l'exécution de l'action élémentaire "positionner paramètres" avec comme cible unique le rt-script lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 46.2.1 de la norme MHEG.

Le paramètre `parameters` spécifie la valeur du paramètre "paramètres" de l'action "positionner paramètres".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.20.2 Opération `getTerminationStatus`

Résumé:

Interface:	`RtScript`
Opération:	`getTerminationStatus`
Résultat:	`TerminationStatusValue`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la terminaison du processus de chaque rt-script et la terminaison par le processus de script.

L'opération `getTerminationStatus` déclenche l'exécution de l'action élémentaire "extraire statut de terminaison" avec comme cible unique le rt-script lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 47.3.1 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.20.3 Description IDL

```
interface RtScript: RtObject                                    {

    void
        setParameters(
              in sequence<Parameter>
                  parameters)
    raises(InvalidTarget);

    TerminationStatusValue
        getTerminationStatus()
    raises(InvalidTarget);

};
```

7.1.21 Objet `RtComponentOrSocket`

Le sous-paragraphe qui suit définit les opérations de l'objet `RtComponentOrSocket`.

7.1.21.1 Opération `setRGS`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setRGS`	
Résultat:	`void`	
Entrée:	`ChannelIdentifier`	`channel_identifier`
Exception:	`InvalidTarget`	

Description:

Cette opération assigne un rt-composant ou un réceptacle à un espace générique relatif (RGS, *relative generic space*).

L'opération `setRGS` déclenche l'exécution de l'action élémentaire "positionner RGS" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 50.2.1 de la norme MHEG.

Le paramètre `channel_identifier` spécifie la valeur du paramètre "identificateur de canal" de l'action "positionner RGS".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.2 Opération `getRGS`

Résumé:

Interface:	`RtComponentOrSocket`
Opération:	`getRGS`
Résultat:	`RGSValue`
Exception:	`InvalidTarget`

Description:

Cette opération extrait l'espace RGS assigné à un rt-composant ou à un réceptacle.

L'opération `getRGS` déclenche l'exécution de l'action élémentaire "extraire RGS" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 50.3.1 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.3 Opération `setOpacity`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setOpacity`	
Résultat:	`void`	
Entrée:	`unsigned short`	`opacity_rate`
Entrée:	`unsigned long`	`transition_duration`
Exception:	`InvalidTarget`	

Description:

Cette opération assigne une valeur de taux d'opacité à un rt-composant ou à un réceptacle.

L'opération `setOpacity` déclenche l'exécution de l'action élémentaire "positionner opacité" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 51.2.1 de la norme MHEG.

Le paramètre `opacity_rate` spécifie la valeur du paramètre "taux d'opacité" de l'action "positionner opacité".

Le paramètre `transition_duration` spécifie la valeur du paramètre "durée de transition" de l'action "positionner opacité".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.4 Opération `setPresentationPriority`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setPresentationPriority`	
Résultat:	`void`	
Entrée:	`PresentationPriority`	`presentation_priority`
Entrée:	`unsigned long`	`transition_duration`
Exception:	`InvalidTarget`	
Exception:	`InvalidParameter`	

Description:

Cette opération spécifie la priorité de présentation entre les rt-composants ou les réceptacles assignés à un même espace RGS. L'opération définit la priorité du rt-composant ou du réceptacle par rapport aux autres rt-composants ou réceptacles assignés à un même espace RGS.

L'opération `setPresentationPriority` déclenche l'exécution de l'action élémentaire "positionner priorité de présentation" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 51.2.2 de la norme MHEG.

Le paramètre `presentation_priority` spécifie la valeur du paramètre "priorité de présentation" de l'action "positionner priorité de présentation".

Le paramètre `transition_duration` spécifie la valeur du paramètre "durée de transition" de l'action "positionner priorité de présentation".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.21.5 Opération `getOpacity`

Résumé:

Interface: `RtComponentOrSocket`
Opération: `getOpacity`
Résultat: `unsigned short`
Exception: `InvalidTarget`

Description:

Cette opération extrait la valeur d'opacité d'un rt-composant ou d'un réceptacle.

L'opération `getOpacity` déclenche l'exécution de l'action élémentaire "extraire opacité" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 51.3.1 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.6 Opération `getEffectiveOpacity`

Résumé:

Interface: `RtComponentOrSocket`
Opération: `getEffectiveOpacity`
Résultat: `unsigned short`
Exception: `InvalidTarget`

Description:

Cette opération extrait la valeur de l'opacité effective d'un rt-composant ou d'un réceptacle.

L'opération `getEffectiveOpacity` déclenche l'exécution de l'action élémentaire "extraire opacité effective" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 51.3.2 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.7 Opération `getPresentationPriority`

Résumé:

Interface: `RtComponentOrSocket`
Opération: `getPresentationPriority`
Résultat: `unsigned short`
Exception: `InvalidTarget`

Description:

Cette opération extrait la priorité de présentation d'un rt-composant ou d'un réceptacle.

L'opération `getPresentationPriority` déclenche l'exécution de l'action élémentaire "extraire priorité de présentation" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 51.3.3 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.8 Opération `setVisibleDuration`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setVisibleDuration`	
Résultat:	`void`	
Entrée:	`TemporalPosition`	`initial_temporal_position`
Entrée:	`TemporalPosition`	`terminal_temporal_position`
Exception:	`InvalidTarget`	
Exception:	`InvalidParameter`	

Description:

Cette opération extrait la priorité de présentation d'un rt-composant ou d'un réceptacle.

L'opération `setVisibleDuration` déclenche l'exécution de l'action élémentaire "positionner durée visible" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 51.3.3 de la norme MHEG.

Le paramètre `initial_temporal_position` spécifie la valeur du paramètre "position temporelle initiale" de l'action "positionner durée visible".

Le paramètre `terminal_temporal_position` spécifie la valeur du paramètre "position temporelle finale" de l'action "positionner durée visible".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.21.9 Opération `setTemporalTermination`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setTemporalTermination`	
Résultat:	`void`	
Entrée:	`TemporalTermination`	`temporal_termination`
Exception:	`InvalidTarget`	

Description:

Cette opération spécifie le type de terminaison temporelle lorsque la position temporelle courante dépasse la position temporelle terminale.

L'opération `setTemporalTermination` déclenche l'exécution de l'action élémentaire "positionner terminaison temporelle" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 52.2.2 de la norme MHEG.

Le paramètre `temporal_termination` spécifie la valeur du paramètre "terminaison temporelle" de l'action "positionner terminaison temporelle".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.10 Opération `setCurrentTemporalPosition`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setCurrentTemporalPosition`	
Résultat:	`void`	
Entrée:	`TemporalPosition`	`temporal_position`
Exception:	`InvalidTarget`	
Exception:	`InvalidParameter`	

Description:

Cette opération spécifie la position temporelle actuelle au sein de la durée visible.

L'opération `setCurrentTemporalPosition` déclenche l'exécution de l'action élémentaire "positionner position temporelle" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 52.2.3 de la norme MHEG.

Le paramètre `temporal_position` spécifie la valeur du paramètre "position temporelle" de l'action "positionner position temporelle actuelle".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.21.11 Opération `setSpeed`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setSpeed`	
Résultat:	`void`	
Entrée:	`Speed`	`the_speed`
Entrée:	`unsigned long`	`transition_duration`
Exception:	`InvalidTarget`	
Exception:	`InvalidParameter`	

Description:

Cette opération définit la vitesse de présentation d'un rt-composant ou d'un réceptacle. La vitesse de présentation effective est calculée par le moteur MHEG à partir de cette valeur.

L'opération `setSpeed` déclenche l'exécution de l'action élémentaire "positionner vitesse" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 52.2.5 de la norme MHEG.

Le paramètre `the_speed` spécifie la valeur du paramètre "vitesse" de l'action "positionner vitesse".

Le paramètre `transition_duration` spécifie la valeur du paramètre "durée de transition" de l'action "positionner vitesse".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.21.12 Opération `setTimestones`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setTimestones`	
Résultat:	`void`	
Entrée:	`sequence<Timestone>`	`timestones`
Exception:	`InvalidTarget`	
Exception:	`InvalidParameter`	

Description:

Cette opération spécifie un ensemble complet de marqueurs temporels à l'intérieur de la durée perceptible du présentable.

L'opération `setTimestones` déclenche l'exécution de l'action élémentaire "positionner marque temporelle" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 52.2.6 de la norme MHEG.

Le paramètre `timestones` spécifie la valeur du paramètre "marque temporelle" de l'action "positionner marques temporelles".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.21.13 Opération `getInitialTemporalPosition`

Résumé:

Interface:	`RtComponentOrSocket`
Opération:	`getInitialTemporalPosition`
Résultat:	`unsigned long`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la valeur de la position temporelle initiale du rt-composant ou du réceptacle. Cette valeur est exprimée en unités temporelles génériques d'espace générique d'origine (OGTU, *original generic space generic temporal unit*).

L'opération `getInitialTemporalPosition` déclenche l'exécution de l'action élémentaire "extraire position temporelle initiale" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 52.3.2 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.14 Opération `getTerminalTemporalPosition`

Résumé:

Interface:	`RtComponentOrSocket`
Opération:	`getTerminalTemporalPosition`
Résultat:	`unsigned long`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la valeur de la position temporelle terminale du rt-composant ou d'un réceptacle. Cette valeur est exprimée en unités OGTU.

L'opération `getTerminalTemporalPosition` déclenche l'exécution de l'action élémentaire "extraire position temporelle terminale" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 52.3.3 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.15 Opération `getVDLength`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`getVDLength`	
Résultat:	`unsigned long`	
Entrée:	`GTIndicator`	`gt_indicator`
Exception:	`InvalidTarget`	

Description:

Cette opération extrait la valeur de longueur de la taille visible du rt-composant ou d'un réceptacle exprimée soit en unités OGTU, soit en unités temporelles génériques d'espace générique relatif (RGTU, *relative generic space generic temporal unit*).

L'opération `getVDLength` déclenche l'exécution de l'action élémentaire "extraire longueur taille visible" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 52.3.4 de la norme MHEG.

Le paramètre `gt_indicator` spécifie la valeur du paramètre "indicateur temporel générique" de l'action "extraire longueur taille visible".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.16 Opération `getTemporalTermination`

Résumé:

Interface:	`RtComponentOrSocket`
Opération:	`getTemporalTermination`
Résultat:	`TemporalTermination`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la valeur de la "terminaison temporelle" du rt-composant ou du réceptacle.

L'opération `getTemporalTermination` déclenche l'exécution de l'action élémentaire "extraire terminaison temporelle" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 52.3.5 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.17 Opération `getCurrentTemporalPosition`

Résumé:

Interface:	`RtComponentOrSocket`
Opération:	`getCurrentTemporalPosition`
Résultat:	`unsigned long`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la valeur de la position temporelle courante du rt-composant ou du réceptacle. Cette valeur est exprimée en unités OGTU.

L'opération `getCurrentTemporalPosition` déclenche l'exécution de l'action élémentaire "extraire position temporelle courante" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 52.3.6 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.18 Opération `getSpeedRate`

Résumé:

Interface: `RtComponentOrSocket`
Opération: `getSpeedRate`
Résultat: `short`
Exception: `InvalidTarget`

Description:

Cette opération extrait le rapport de vitesse du rt-composant ou d'un réceptacle. Cette valeur est un pourcentage positif ou négatif. Elle est utilisée pour indiquer la modification de vitesse exigée à partir du rapport temporel générique d'espace générique d'origine (IOGTR, *initial original generic space generic temporal ratio*) ainsi que la direction exigée pour la présentation.

L'opération `getSpeedRate` déclenche l'exécution de l'action élémentaire "extraire rapport de vitesse" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 52.3.8 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.19 Opération `getOGTR`

Résumé:

Interface: `RtComponentOrSocket`
Opération: `getOGTR`
Résultat: `unsigned long`
Exception: `InvalidTarget`

Description:

Cette opération extrait la valeur du rapport temporel générique d'espace générique d'origine (OGTR, *original generic space generic temporal ratio*) du rt-composant ou du réceptacle. Cette valeur est un nombre positif ou nul correspondant au nombre d'unités OGTU devant être mappées en une seconde.

Le paramètre `getOGTR` déclenche l'exécution de l'action élémentaire "extraire OGTR" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 52.3.9 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.20 Opération `getEffectiveSpeedRate`

Résumé:

Interface: `RtComponentOrSocket`
Opération: `getEffectiveSpeedRate`
Résultat: `short`
Exception: `InvalidTarget`

Description:

Cette opération extrait le rapport de vitesse effectif du rt-composant ou du réceptacle. Cette valeur est un pourcentage positif ou négatif. Le rapport de vitesse effectif est utilisé pour calculer la modification de vitesse effective depuis le rapport temporel (IOGTR) ainsi que la direction exigée pour la présentation.

L'opération `getEffectiveSpeedRate` déclenche l'exécution de l'action élémentaire "extraire rapport de vitesse effectif" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 52.3.10 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.21 Opération `getEffectiveOGTR`

Résumé:

Interface:	`RtComponentOrSocket`
Opération:	`getEffectiveOGTR`
Résultat:	`unsigned long`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la valeur du rapport OGTR effectif du rt-composant ou du réceptacle. Cette valeur est un nombre positif ou nul correspondant au nombre d'unités OGTU devant être mappées en une seconde.

L'opération `getEffectiveOGTR` déclenche l'exécution de l'action élémentaire "extraire rapport OGTR effectif" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 52.3.11 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.22 Opération `getTimestoneStatus`

Résumé:

Interface:	`RtComponentOrSocket`
Opération:	`getTimestoneStatus`
Résultat:	`unsigned short`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la valeur du statut de marque temporelle du rt-composant ou du réceptacle.

L'opération `getTimestoneStatus` déclenche l'exécution de l'action élémentaire "extraire statut de marque temporelle" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 52.3.12 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.23 Opération `setPerceptibleSizeProjection`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setPerceptibleSizeProjection`	
Résultat:	`void`	
Entrée:	`PerceptibleSizeProjection`	`perceptible_size_projection`
Entrée:	`unsigned long`	`transition_duration`
Exception:	`InvalidTarget`	
Exception:	`InvalidParameter`	

Description:

Cette opération définit la projection de la taille perceptible dans son espace RGS.

L'opération `setPerceptibleSizeProjection` déclenche l'exécution de l'action élémentaire "positionner projection de taille perceptible" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 53.2.1 de la norme MHEG.

Le paramètre `perceptible_size_projection` spécifie la valeur du paramètre "projection de taille" de l'action "positionner projection de taille perceptible".

Le paramètre `transition_duration` spécifie la valeur du paramètre "durée de transition" de l'action "positionner projection de taille perceptible".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.21.24 Opération `setAspectRatio`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setAspectRatio`	
Résultat:	`void`	
Entrée:	`AspectRatio`	`preserved`
Exception:	`InvalidTarget`	

Description:

Cette opération spécifie l'option prise lors de la projection d'un rt-composant ou d'un réceptacle dans l'espace générique de canal (CGS, *channel generic space*) [au moyen de la chaîne des mappages de l'espace générique original (OGS, *original generic space*) vers l'espace RGS], la relation entre la taille perceptible (PS, *perceptible size*) de l'unité spatiale générique d'espace générique original (OGSU, *original generic space generic spatial unit*), c'est-à-dire les longueurs d'espace OGS et la projection dans l'unité générique d'espace du canal (CGSU, *channel generic space unit*) de la "taille de l'information de contenu" doivent être les mêmes pour chaque axe. Dans ce cas, le facteur de forme est conservé.

L'opération `setAspectRatio` déclenche l'exécution de l'action élémentaire "positionner facteur de forme conservé" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 53.2.3 de la norme MHEG.

Le paramètre `preserved` spécifie la valeur du paramètre "conservé" de l'action "positionner facteur de forme conservé".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.25 Opération `setVisibleSize`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setVisibleSize`	
Résultat:	`void`	
Entrée:	`VSGS`	`the_vsgs`
Entrée:	`VS`	`the_vs`
Entrée:	`unsigned long`	`transition_duration`
Exception:	`InvalidTarget`	
Exception:	`InvalidParameter`	

Description:

Cette opération spécifie la taille visible (VS, *visible size*) définissant quelle est la partie de la taille perceptible perçue par l'utilisateur.

L'opération `setVisibleSize` déclenche l'exécution de l'action élémentaire "positionner taille visible" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 53.2.4 de la norme MHEG.

Le paramètre `the_vsgs` spécifie la valeur du paramètre "vsgs" de l'action "positionner taille visible".

Le paramètre `the_vs` spécifie la valeur du paramètre "vs" de l'action "positionner taille visible".

Le paramètre `transition_duration` spécifie la valeur du paramètre "durée de transition" de l'action "positionner taille visible".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.21.26 Opération `setVisibleSizesAdjustment`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setVisibleSizesAdjustment`	
Résultat:	`void`	
Entrée:	`sequence<AdjustmentAxis>`	`set_of_axes`
Entrée:	`AdjustmentPolicy`	`adjustment_policy`
Entrée:	`unsigned long`	`transition_duration`
Exception:	`InvalidTarget`	

Description:

Cette opération spécifie l'ajustement d'un ensemble de tailles visibles sur un même axe ou un ensemble d'axes. Toutes les tailles visibles à ajuster doivent être assignées au même espace CGS.

L'opération `setVisibleSizesAdjustment` déclenche l'exécution de l'action élémentaire "positionner l'ajustement des tailles visibles" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 53.2.5 de la norme MHEG.

Le paramètre `set_of_axes` spécifie la valeur du paramètre "ensemble d'axes" de l'action "positionner l'ajustement des tailles visibles".

Le paramètre `adjustment_policy` spécifie la valeur du paramètre "politique d'ajustement" de l'action "positionner l'ajustement des tailles visibles".

Le paramètre `transition_duration` spécifie la valeur du paramètre "durée de transition" de l'action "positionner l'ajustement des tailles visibles".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.27 Opération `setBox`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setBox`	
Résultat:	`void`	
Entrée:	`BoxConstants`	`box`
Exception:	`InvalidTarget`	

Description:

Cette opération spécifie si le rt-composant ou le réceptacle est présenté avec une boîte indiquant le pourtour de la taille visible.

L'opération `setBox` déclenche l'exécution de l'action élémentaire "positionner boîte" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 53.2.6 de la norme MHEG.

Le paramètre `box` spécifie la valeur du paramètre "boîte" de l'action "positionner boîte".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.28 Opération `setDefaultBackground`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setDefaultBackground`	
Résultat:	`void`	
Entrée:	`unsigned short`	`background`
Entrée:	`unsigned long`	`transition_duration`
Exception:	`InvalidTarget`	
Exception:	`InvalidParameter`	

Description:

Cette opération spécifie si les zones situées à l'intérieur de la taille visible d'un rt-composant ou d'un réceptacle et qui ne sont pas remplies par le processus de fonctionnement doivent être considérées comme opaques ou transparentes.

L'opération `setDefaultBackground` déclenche l'exécution de l'action élémentaire "positionner arrière-plan par défaut" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 53.2.7 de la norme MHEG.

Le paramètre `background` spécifie la valeur du paramètre "arrière-plan" de l'action "positionner arrière-plan par défaut".

Le paramètre `transition_duration` spécifie la valeur du paramètre "durée de transition" de l'action "positionner arrière-plan par défaut".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.21.29 Opération `setAttachmentPoint`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setAttachmentPoint`	
Résultat:	`void`	
Entrée:	`AttachmentPointType`	`type`
Entrée:	`AttachmentPoint`	`positions`
Exception:	`InvalidTarget`	
Exception:	`InvalidParameter`	

Description:

Cette opération spécifie l'un des points de rattachement suivants: point de rattachement de taille perceptible (PSAP, *perceptible size attachment point*), point de rattachement interne de taille perceptible (VSIAP, *visible size internal attachment point*) ou point de rattachement externe de taille perceptible (VSEAP, *visible size external attachment point*).

L'opération `setAttachmentPoint` déclenche l'exécution de l'action élémentaire "positionner point de rattachement" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 53.2.8 de la norme MHEG.

Le paramètre `type` spécifie la valeur du paramètre "type" de l'action "positionner point de rattachement".

Le paramètre `positions` spécifie la valeur du paramètre "positions" de l'action "positionner point de rattachement".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.21.30 Opération `setAttachmentPointPosition`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setAttachmentPointPosition`	
Résultat:	`void`	
Entrée:	`AttachmentPointType`	`type`
Entrée:	`ReferenceType`	`vseap_reference_point`
Entrée:	`Lengths`	`the_lengths`
Entrée:	`unsigned long`	`transition_duration`
Exception:	`InvalidTarget`	
Exception:	`InvalidParameter`	

Description:

Cette opération spécifie la position du point VSIAP par rapport au point PSAP, ou la position du point VSEAP dans son espace RGS par rapport à l'origine ou par rapport à un autre point VSEAP.

L'opération `setAttachmentPointPosition` déclenche l'exécution de l'action élémentaire "positionner la position du point de rattachement" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 53.2.9 de la norme MHEG.

Le paramètre `type` spécifie la valeur du paramètre "type" de l'action "positionner la position du point de rattachement".

Le paramètre `vseap_reference_point` spécifie la valeur du paramètre "point de référence vseap" de l'action "positionner la position du point de rattachement".

Le paramètre `the_lengths` spécifie la valeur du paramètre "longueurs" de l'action "positionner la position du point de rattachement".

Le paramètre `transition_duration` spécifie la valeur du paramètre "durée de transition" de l'action "positionner la position du point de rattachement".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.21.31 Opération `setVisibleSizesAlignment`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setVisibleSizesAlignment`	
Résultat:	`void`	
Entrée:	`SizeBorder`	`size_border`
Entrée:	`long`	`interval`
Entrée:	`unsigned long`	`transition_duration`
Exception:	`InvalidTarget`	

Description:

Cette opération spécifie l'alignement d'un ensemble de tailles visibles. Les tailles visibles sont alignées sur une bordure et il est possible d'indiquer un intervalle entre deux tailles visibles. Toutes les tailles visibles à aligner doivent être assignées dans un même espace CGS.

L'opération `setVisibleSizesAlignment` déclenche l'exécution de l'action élémentaire "positionner l'alignement des tailles visibles" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 53.2.10 de la norme MHEG.

Le paramètre `size_border` spécifie la valeur du paramètre "bordure de taille" de l'action "positionner l'alignement des tailles visibles".

Le paramètre `interval` spécifie la valeur du paramètre "intervalle" de l'action "positionner l'alignement des tailles visibles".

Le paramètre `transition_duration` spécifie la valeur du paramètre "durée de transition" de l'action "positionner l'alignement des tailles visibles".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.32 Opération `setMovingAbility`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setMovingAbility`	
Résultat:	`void`	
Entrée:	`UserControls`	`moving_ability`
Exception:	`InvalidTarget`	

Description:

Cette opération spécifie si l'utilisateur est en mesure ou non de déplacer les espaces visibles des cibles dans leurs espaces CGS.

L'opération `setMovingAbility` déclenche l'exécution de l'action élémentaire "positionner la capacité de déplacement" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 53.2.11 de la norme MHEG.

Le paramètre `moving_ability` spécifie la valeur du paramètre "capacité de déplacement" de l'action "positionner capacité de déplacement".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.33 Opération `setResizingAbility`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setResizingAbility`	
Résultat:	`void`	
Entrée:	`UserControls`	`resizing_ability`
Exception:	`InvalidTarget`	

Description:

Cette opération spécifie si l'utilisateur est en mesure ou non de redimensionner les espaces visibles des cibles dans leurs espaces CGS.

L'opération `setResizingAbility` déclenche l'exécution de l'action élémentaire "positionner capacité de redimensionnement" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 53.2.12 de la norme MHEG.

Le paramètre `resizing_ability` spécifie la valeur du paramètre "capacité de redimensionnement" de l'action "positionner capacité de redimensionnement".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.34 Opération `setScalingAbility`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setScalingAbility`	
Résultat:	`void`	
Entrée:	`UserControls`	`scaling_ability`
Exception:	`InvalidTarget`	

Description:

Cette opération spécifie si l'utilisateur est en mesure ou non de modifier l'échelle des tailles perceptibles dans leurs espaces CGS.

L'opération `setScalingAbility` déclenche l'exécution de l'action élémentaire "positionner capacité de changement d'échelle" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 53.2.13 de la norme MHEG.

Le paramètre `scaling_ability` spécifie la valeur du paramètre "capacité de changement d'échelle" de l'action "positionner capacité de changement d'échelle".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.35 Opération `setScrollingAbility`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setScrollingAbility`	
Résultat:	`void`	
Entrée:	`UserControls`	`scrolling_ability`
Exception:	`InvalidTarget`	

Description:

Cette opération spécifie si l'utilisateur est en mesure ou non de faire défiler les tailles perceptibles à travers des espaces visibles des cibles dans leurs espaces CGS.

L'opération `setScrollingAbility` déclenche l'exécution de l'action élémentaire "positionner la capacité de défilement" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 53.2.14 de la norme MHEG.

Le paramètre `scrolling_ability` spécifie la valeur du paramètre "capacité de défilement" de l'action "positionner capacité de défilement".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.36 Opération `getGSR`

Résumé:

Interface:	`RtComponentOrSocket`
Opération:	`getGSR`
Résultat:	`unsigned short`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la valeur du rapport spatial générique (GSR, *generic spatial ratio*) de l'espace OGS du rt-composant ou du réceptacle. Ce rapport définit le nombre d'unités OGSU qui doivent être mappées avec une unité spatiale générique d'espace générique relatif (RGSU, *relative generic space generic spatial unit*).

L'opération `getGSR` déclenche l'exécution de l'action élémentaire "extraire rapport GSR" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 53.4.1 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.37 Opération `getPS`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`getPS`	
Résultat:	`SpecifiedPosition`	
Entrée:	`GSIndicator`	`gs`
Exception:	`InvalidTarget`	

Description:

Cette opération extrait la valeur de taille perceptible du rt-composant ou du réceptacle dans une unité OGSU ou une unité RGSU.

L'opération `getPS` déclenche l'exécution de l'action élémentaire "extraire taille perceptible" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 53.4.2 de la norme MHEG.

Le paramètre `gs` spécifie la valeur du paramètre "taille perceptible" de l'action "extraire taille perceptible".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.38 Opération `getAspectRatio`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`getAspectRatio`	
Résultat:	`AspectRatio`	
Exception:	`InvalidTarget`	

Description:

Cette opération extrait la valeur du facteur de forme de la taille perceptible du rt-composant ou du réceptacle.

L'opération `getAspectRatio` déclenche l'exécution de l'action élémentaire "extraire facteur de forme" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 53.4.4 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.39 Opération `getPSAP`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`getPSAP`	
Résultat:	`SpecifiedPosition`	
Entrée:	`PointType`	`point_type`
Exception:	`InvalidTarget`	

Description:

Cette opération extrait la valeur du point de rattachement de la taille perceptible du rt-composant ou du réceptacle.

L'opération `getPSAP` déclenche l'exécution de l'action élémentaire "extraire point de rattachement de taille perceptible" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 53.4.5 de la norme MHEG.

Le paramètre `point_type` spécifie la valeur du paramètre "type de point" de l'action "extraire point PSAP".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.40 Opération `getVSGS`

Résumé:

Interface:	`RtComponentOrSocket`
Opération:	`getVSGS`
Résultat:	`VSGS`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la valeur de l'espace générique de la taille visible (VSGS, *visible size generic space*) du rt-composant ou du réceptacle.

L'opération `getVSGS` déclenche l'exécution de l'action élémentaire "extraire espace VSGS" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 53.4.6 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.41 Opération `getVS`

Résumé:

Interface:	`RtComponentOrSocket`
Opération:	`getVS`
Résultat:	`SpecifiedPosition`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la valeur de la taille visible du rt-composant ou du réceptacle exprimée en unités spatiales génériques de la taille visible (VSGSU, *visible size generic spatial unit*).

L'opération `getVS` déclenche l'exécution de l'action élémentaire "extraire taille visible" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 53.4.7 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.42 Opération `getBox`

Résumé:

Interface:	`RtComponentOrSocket`
Opération:	`getBox`
Résultat:	`BoxConstants`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la boîte de taille visible du rt-composant ou du réceptacle.

L'opération `getBox` déclenche l'exécution de l'action élémentaire "extraire boîte" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 53.4.8 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.43 Opération `getDefaultBackground`

Résumé:

Interface:	`RtComponentOrSocket`
Opération:	`getDefaultBackground`
Résultat:	`unsigned short`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la valeur par défaut de l'arrière-plan de la taille visible du rt-composant ou du réceptacle.

L'opération `getDefaultBackground` déclenche l'exécution de l'action élémentaire "extraire arrière-plan par défaut" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 53.4.9 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.44 Opération `getVSIAP`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`getVSIAP`	
Résultat:	`SpecifiedPosition`	
Entrée:	`PointType`	`point_type`
Exception:	`InvalidTarget`	

Description:

Cette opération extrait la valeur du point VSIAP du rt-composant ou du réceptacle.

L'opération `getVSIAP` déclenche l'exécution de l'action élémentaire "extraire point VSIAP" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 53.4.10 de la norme MHEG.

Le paramètre `point_type` spécifie la valeur du paramètre "type de point" de l'action "extraire point VSIAP".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.45 Opération `getVSIAPPosition`

Résumé:

Interface:	`RtComponentOrSocket`
Opération:	`getVSIAPPosition`
Résultat:	`SpecifiedPosition`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la valeur de la position du point VSIAP du rt-composant ou du réceptacle. Cette valeur est utilisée pour positionner le point VSIAP par rapport au point PSAP.

L'opération `getVSIAPPosition` déclenche l'exécution de l'action élémentaire "extraire la position du point VSIAP" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 53.4.11 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.46 Opération `getVSEAP`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`getVSEAP`	
Résultat:	`SpecifiedPosition`	
Entrée:	`PointType`	`point_type`
Exception:	`InvalidTarget`	

Description:

Cette opération extrait la valeur du point VSEAP du rt-composant ou du réceptacle.

L'opération `getVSEAP` déclenche l'exécution de l'action élémentaire "extraire point VSEAP" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 53.4.12 de la norme MHEG.

Le paramètre `point_type` spécifie la valeur du paramètre "type de point" de l'action "extraire point VSEAP".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.47 Opération `getVSEAPPosition`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`getVSEAPPosition`	
Résultat:	`SpecifiedPosition`	
Entrée:	`ReferencePoint`	`reference_point`
Exception:	`InvalidTarget`	

Description:

Cette opération extrait la valeur de la position du point VSEAP du rt-composant ou du réceptacle par rapport à un point de référence. Cette valeur est utilisée pour positionner le point VSEAP par rapport au point PSAP.

L'opération `getVSEAPPosition` déclenche l'exécution de l'action élémentaire "extraire position du point VSEAP" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 53.4.13 de la norme MHEG.

Le paramètre `reference_point` spécifie la valeur du paramètre "point de référence" de l'action "extraire position du point VSEAP".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.48 Opération `getMovingAbility`

Résumé:

Interface: `RtComponentOrSocket`
Opération: `getMovingAbility`
Résultat: `UserControls`
Exception: `InvalidTarget`

Description:

Cette opération extrait la valeur de la capacité de déplacement du rt-composant ou du réceptacle.

L'opération `getMovingAbility` déclenche l'exécution de l'action élémentaire "extraire capacité de déplacement" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 53.4.14 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.49 Opération `getResizingAbility`

Résumé:

Interface: `RtComponentOrSocket`
Opération: `getResizingAbility`
Résultat: `UserControls`
Exception: `InvalidTarget`

Description:

Cette opération extrait la valeur de la capacité de redimensionnement du rt-composant ou du réceptacle.

L'opération `getResizingAbility` déclenche l'exécution de l'action élémentaire "extraire capacité de redimensionnement" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 53.4.15 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.50 Opération `getScalingAbility`

Résumé:

Interface: `RtComponentOrSocket`
Opération: `getScalingAbility`
Résultat: `UserControls`
Exception: `InvalidTarget`

Description:

Cette opération extrait la valeur de la capacité de mise à l'échelle du rt-composant ou du réceptacle.

L'opération `getScalingAbility` déclenche l'exécution de l'action élémentaire "extraire capacité de mise à l'échelle" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 53.4.16 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.51 Opération `getScrollingAbility`

Résumé:

Interface:	`RtComponentOrSocket`
Opération:	`getScrollingAbility`
Résultat:	`UserControls`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la valeur de la capacité de défilement du rt-composant ou du réceptacle.

L'opération `getScrollingAbility` déclenche l'exécution de l'action élémentaire "extraire capacité de défilement" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 53.4.17 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.52 Opération `setSelectability`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setSelectability`	
Résultat:	`void`	
Entrée:	`unsigned short`	`min_number_of_selections`
Entrée:	`unsigned short`	`max_number_of_selections`
Exception:	`InvalidTarget`	
Exception:	`InvalidParameter`	

Description:

Cette opération assigne une valeur de "nombre minimal de sélections exigées" et une valeur de "nombre maximal de sélections exigées" à un rt-composant ou à un réceptacle. Le moteur MHEG calcule la valeur de "capacité de sélection" du rt-composant ou du réceptacle à partir de ces deux valeurs. La valeur de "capacité de sélection effective" du rt-composant ou du réceptacle est également calculée par le moteur MHEG à partir de cette valeur de "capacité de sélection" et de la valeur de "capacité effective de sélection" du parent du rt-composant ou du réceptacle.

L'opération `setSelectability` déclenche l'exécution de l'action élémentaire "positionner capacité de sélection" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 57.2.1 de la norme MHEG.

Le paramètre `min_number_of_selections` spécifie la valeur du paramètre "nombre minimal de sélections" de l'action "positionner capacité de sélection".

Le paramètre `max_number_of_selections` spécifie la valeur du paramètre "nombre maximal de sélections" de l'action "positionner capacité de sélection".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.21.53 Opération `setSelectionStatus`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setSelectionStatus`	
Résultat:	`void`	
Entrée:	`SelectionStatusValue`	`selection_state`
Exception:	`InvalidTarget`	

Description:

Cette opération assigne une valeur au "statut de sélection" d'un rt-composant ou d'un réceptacle.

L'opération `setSelectionStatus` déclenche l'exécution de l'action élémentaire "positionner statut de sélection" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 57.2.2 de la norme MHEG.

Le paramètre `selection_state` spécifie la valeur du paramètre "statut de sélection" de l'action "positionner statut de sélection".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.54 Opération `setSelectionPresentationEffectResponsibility`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setSelectionPresentationEffectResponsibility`	
Résultat:	`void`	
Entrée:	`Responsibility`	`the_responsibility`
Exception:	`InvalidTarget`	

Description:

Cette opération assigne une valeur à la "responsabilité de sélection d'effet de présentation" d'un rt-composant ou d'un réceptacle. Cet attribut indique, comme cible unique, si c'est le moteur MHEG ou l'auteur qui est responsable de la présentation d'un nouvel état du rt-composant ou du réceptacle.

L'opération `setSelectionPresentationEffectResponsibility` déclenche l'exécution de l'action élémentaire "positionner responsabilité de sélection d'effet de présentation" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 57.2.3 de la norme MHEG.

Le paramètre `the_responsibility` spécifie la valeur du paramètre "responsabilité" de l'action "positionner responsabilité de sélection d'effet de présentation".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.55 Opération `getSelectability`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`getSelectability`	
Résultat:	`void`	
Sortie:	`unsigned short`	`min_mumber_of_selections`
Sortie:	`unsigned short`	`max_mumber_of_selections`
Exception:	`InvalidTarget`	

Description:

Cette opération extrait le "minimum de sélections exigé" et le "maximum de sélections exigé". Si le "maximum de sélections exigé" est égal à 0, la "capacité de sélection" du rt-composant ou du réceptacle est "non sélectable".

L'opération `getSelectability` déclenche l'exécution de l'action élémentaire "extraire capacité de sélection" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 57.3.1 de la norme MHEG.

Le paramètre `min_mumber_of_selections` spécifie la valeur du paramètre "nombre minimal de sélections" de l'action "extraire capacité de sélection".

Le paramètre `max_mumber_of_selections` spécifie la valeur du paramètre "nombre maximal de sélections" de l'action "extraire capacité de sélection".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.56 Opération `getEffectiveSelectability`

Résumé:

Interface: `RtComponentOrSocket`
Opération: `getEffectiveSelectability`
Résultat: `EffectiveSelectability`
Exception: `InvalidTarget`

Description:

Cette opération extrait la valeur de l'attribut "capacité de sélection effective" du rt-composant ou du réceptacle.

L'opération `getEffectiveSelectability` déclenche l'exécution de l'action élémentaire "extraire capacité de sélection effective" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 57.3.2 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.57 Opération `getSelectionStatus`

Résumé:

Interface: `RtComponentOrSocket`
Opération: `getSelectionStatus`
Résultat: `SelectionStatusValue`
Exception: `InvalidTarget`

Description:

Cette opération extrait la valeur du "statut de sélection" du rt-composant ou du réceptacle.

L'opération `getSelectionStatus` déclenche l'exécution de l'action élémentaire "extraire statut de sélection" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 57.3.3 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.58 Opération `getSelectionMode`

Résumé:

Interface: `RtComponentOrSocket`
Opération: `getSelectionMode`
Résultat: `SelectionModeValue`
Exception: `InvalidTarget`

Description:

Cette opération extrait la valeur de l'attribut "mode de sélection" du rt-composant ou du réceptacle.

L'opération `getSelectionMode` déclenche l'exécution de l'action élémentaire "extraire mode de sélection" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 57.3.4 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.59 Opération `getSelectionPresentationEffectResponsibility`

Résumé:

Interface:	`RtComponentOrSocket`
Opération:	`getSelectionPresentationEffectResponsibility`
Résultat:	`Responsibility`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la valeur de l'attribut "responsabilité de sélection du mode de présentation" du rt-composant ou du réceptacle.

L'opération `getSelectionPresentationEffectResponsibility` déclenche l'exécution de l'action élémentaire "extraire responsabilité de sélection du mode de présentation" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 57.3.6 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.60 Opération `setModifiability`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setModifiability`	
Résultat:	`void`	
Entrée:	`unsigned short`	`min_number_of_modifications`
Entrée:	`unsigned short`	`max_number_of_modifications`
Exception:	`InvalidTarget`	
Exception:	`InvalidParameter`	

Description:

Cette opération assigne une valeur de "nombre minimal de modifications exigé" et une valeur de "nombre maximal de modifications exigé" à un rt-composant ou à un réceptacle. Le moteur MHEG calcule la valeur de "capacité de modification" du rt-composant ou du réceptacle à partir de ces deux valeurs. La valeur de "capacité de modification effective" du rt-composant ou du réceptacle est également calculée par le moteur MHEG à partir de cette "capacité de modification" et de la valeur de "capacité effective" du parent du rt-composant ou du réceptacle.

L'opération `setModifiability` déclenche l'exécution de l'action élémentaire "positionner capacité de modification" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 58.2.1 de la norme MHEG.

Le paramètre `min_number_of_modifications` spécifie la valeur du paramètre "nombre minimal de modifications" de l'action "positionner capacité de modification".

Le paramètre `max_number_of_modifications` spécifie la valeur du paramètre "nombre maximal de modifications exigé" de l'action "positionner capacité de modification".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.21.61 **Opération** `setModificationStatus`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setModificationStatus`	
Résultat:	`void`	
Entrée:	`ModificationStatusValue`	`modification_state`
Exception:	`InvalidTarget`	

Description:

Cette opération assigne une valeur au "statut de modification" d'un rt-composant ou d'un réceptacle.

L'opération `setModificationStatus` déclenche l'exécution de l'action élémentaire "positionner statut de modification" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 58.2.2 de la norme MHEG.

Le paramètre `modification_state` spécifie la valeur du paramètre "état de modification" de l'action "positionner statut de modification".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.62 **Opération** `setModificationPresentationEffectResponsibility`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`setModificationPresentationEffectResponsibility`	
Résultat:	`void`	
Entrée:	`Responsibility`	`the_responsibility`
Exception:	`InvalidTarget`	

Description:

Cette opération assigne une valeur à la "responsabilité d'effet de présentation de modification" d'un rt-composant ou d'un réceptacle. Cet attribut indique si c'est le moteur MHEG ou l'auteur qui est responsable de la présentation d'un nouvel état du composant ou du réceptacle.

L'opération `setModificationPresentationEffectResponsibility` déclenche l'exécution de l'action élémentaire "positionner responsabilité d'effet de présentation de modification" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 58.2.3 de la norme MHEG.

Le paramètre `the_responsibility` spécifie la valeur du paramètre "responsabilité" de l'action "positionner responsabilité d'effet de présentation de modification".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.63 **Opération** `getModifiability`

Résumé:

Interface:	`RtComponentOrSocket`	
Opération:	`getModifiability`	
Résultat:	`void`	
Sortie:	`unsigned short`	`min_numbers_of_modifications`
Sortie:	`unsigned short`	`max_numbers_of_modifications`
Exception:	`InvalidTarget`	

Description:

Cette opération extrait le "nombre minimal de modifications exigé" et le "nombre maximal de modifications exigé". Si le "nombre maximal de modifications exigé" est égal à 0, la "capacité de modification" du rt-composant ou du réceptacle est "non modifiable".

L'opération `getModifiability` déclenche l'exécution de l'action élémentaire "extraire capacité de modification" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 58.3.1 de la norme MHEG.

Le paramètre `min_numbers_of_modifications` spécifie la valeur du paramètre "nombre minimal de modifications" de l'action "extraire capacité de modification".

Le paramètre `max_numbers_of_modifications` spécifie la valeur du paramètre "nombre maximal de modifications" de l'action "extraire capacité de modification".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.64 Opération `getEffectiveModifiability`

Résumé:

Interface: `RtComponentOrSocket`
Opération: `getEffectiveModifiability`
Résultat: `EffectiveModifiability`
Exception: `InvalidTarget`

Description:

Cette opération extrait la valeur de l'attribut "capacité effective de modification" du rt-composant ou du réceptacle.

Le paramètre `getEffectiveModifiability` déclenche l'exécution de l'action élémentaire "extraire capacité effective de modification" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 58.3.2 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.65 Opération `getModificationStatus`

Résumé:

Interface: `RtComponentOrSocket`
Opération: `getModificationStatus`
Résultat: `ModificationStatusValue`
Exception: `InvalidTarget`

Description:

Cette opération extrait la valeur du "statut de modification" du rt-composant ou du réceptacle.

L'opération `getModificationStatus` déclenche l'exécution de l'action élémentaire "extraire statut de modification" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 58.3.3 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.66 Opération `getModificationMode`

Résumé:

Interface: `RtComponentOrSocket`
Opération: `getModificationMode`
Résultat: `ModificationModeValue`
Exception: `InvalidTarget`

Description:

Cette opération extrait la valeur de l'attribut "mode de modification" du rt-composant ou du réceptacle.

Le paramètre `getModificationMode` déclenche l'exécution de l'action élémentaire "extraire mode de modification" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 58.3.4 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.67 Opération `getModificationPresentationEffectResponsibility`

Résumé:

Interface: `RtComponentOrSocket`
Opération: `getModificationPresentationEffectResponsibility`
Résultat: `Responsibility`
Exception: `InvalidTarget`

Description:

Cette opération extrait la valeur de l'attribut "responsabilité d'effet de présentation de modification" du rt-composant ou du réceptacle.

L'opération `getModificationPresentationEffectResponsibility` déclenche l'exécution de l'action élémentaire "extraire responsabilité d'effet de présentation de modification" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 58.3.6 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.68 Opération `setNoInteractionStyle`

Résumé:

Interface: `RtComponentOrSocket`
Opération: `setNoInteractionStyle`
Résultat: `void`
Exception: `InvalidTarget`

Description:

Cette opération supprime une assignation courante d'un style d'interaction à un rt-composant ou à un réceptacle.

Le paramètre `setNoInteractionStyle` déclenche l'exécution de l'action élémentaire "ne positionner aucun style" avec comme cible unique le rt-composant lié ou le réceptacle lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 59.2.6 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.21.69 Description IDL

```
interface RtComponentOrSocket                                    {

    void
        setRGS(
            in ChannelIdentifier
                channel_identifier)
    raises(InvalidTarget);

    RGSValue
        getRGS()
    raises(InvalidTarget);
```

```
void
    setOpacity(
        in unsigned short
            opacity_rate,
        in unsigned long
            transition_duration)
raises(InvalidTarget);

void
    setPresentationPriority(
        in PresentationPriority
            presentation_priority,
        in unsigned long
            transition_duration)
raises(InvalidTarget, InvalidParameter);

unsigned short
    getOpacity()
raises(InvalidTarget);

unsigned short
    getEffectiveOpacity()
raises(InvalidTarget);

unsigned short
    getPresentationPriority()
raises(InvalidTarget);

void
    setVisibleDuration(
        in TemporalPosition
            initial_temporal_position,
        in TemporalPosition
            terminal_temporal_position)
raises(InvalidTarget, InvalidParameter);

void
    setTemporalTermination(
        in TemporalTermination
            temporal_termination)
raises(InvalidTarget);

void
    setCurrentTemporalPosition(
        in TemporalPosition
            temporal_position)
raises(InvalidTarget, InvalidParameter);

void
    setSpeed(
        in Speed
            the_speed,
        in unsigned long
            transition_duration)
raises(InvalidTarget, InvalidParameter);

void
    setTimestones(
        in sequence<Timestone>
            timestones)
raises(InvalidTarget, InvalidParameter);

unsigned long
    getInitialTemporalPosition()
raises(InvalidTarget);

unsigned long
    getTerminalTemporalPosition()
raises(InvalidTarget);

unsigned long
    getVDLength(
        in GTIndicator
            gt_indicator)
raises(InvalidTarget);
```

```
TemporalTermination
    getTemporalTermination()
raises(InvalidTarget);

unsigned long
    getCurrentTemporalPosition()
raises(InvalidTarget);

short
    getSpeedRate()
raises(InvalidTarget);

unsigned long
    getOGTR()
raises(InvalidTarget);

short
    getEffectiveSpeedRate()
raises(InvalidTarget);

unsigned long
    getEffectiveOGTR()
raises(InvalidTarget);

unsigned short
    getTimestoneStatus()
raises(InvalidTarget);

void
    setPerceptibleSizeProjection(
        in PerceptibleSizeProjection
            perceptible_size_projection,
        in unsigned long
            transition_duration)
raises(InvalidTarget, InvalidParameter);

void
    setAspectRatio(
        in AspectRatio
            preserved)
raises(InvalidTarget);

void
    setVisibleSize(
         in VSGS
             the_vsgs,
         in VS
             the_vs,
         in unsigned long
             transition_duration)
raises(InvalidTarget, InvalidParameter);

void
    setVisibleSizesAdjustment(
         in sequence<AdjustmentAxis>
             set_of_axes,
         in AdjustmentPolicy
             adjustment_policy,
         in unsigned long
             transition_duration)
raises(InvalidTarget);

void
    setBox(
        in BoxConstants
            box)
raises(InvalidTarget);

void
    setDefaultBackground(
        in unsigned short
            background,
        in unsigned long
            transition_duration)
raises(InvalidTarget, InvalidParameter);
```

```
void
    setAttachmentPoint(
        in AttachmentPointType
            type,
        in AttachmentPoint
            positions)
raises(InvalidTarget, InvalidParameter);

void
    setAttachmentPointPosition(
        in AttachmentPointType
            type,
        in ReferenceType
            vseap_reference_point,
        in Lengths
            the_lengths,
        in unsigned long
            transition_duration)
raises(InvalidTarget, InvalidParameter);

void
    setVisibleSizesAlignment(
         in SizeBorder
             size_border,
         in long
             interval,
         in unsigned long
             transition_duration)
raises(InvalidTarget);

void
    setMovingAbility(
        in UserControls
            moving_ability)
raises(InvalidTarget);

void
    setResizingAbility(
        in UserControls
            resizing_ability)
raises(InvalidTarget);

void
    setScalingAbility(
        in UserControls
            scaling_ability)
raises(InvalidTarget);

void
    setScrollingAbility(
        in UserControls
            scrolling_ability)
raises(InvalidTarget);

unsigned short
    getGSR()
raises(InvalidTarget);

SpecifiedPosition
    getPS(
        in GSIndicator
            gs)
raises(InvalidTarget);

AspectRatio
    getAspectRatio()
raises(InvalidTarget);

SpecifiedPosition
    getPSAP(
        in PointType
            point_type)
raises(InvalidTarget);
```

```
VSGS
    getVSGS()
raises(InvalidTarget);

SpecifiedPosition
    getVS()
raises(InvalidTarget);

BoxConstants
    getBox()
raises(InvalidTarget);

unsigned short
    getDefaultBackground()
raises(InvalidTarget);

SpecifiedPosition
    getVSIAP(
        in PointType
            point_type)
raises(InvalidTarget);

SpecifiedPosition
    getVSIAPPosition()
raises(InvalidTarget);

SpecifiedPosition
    getVSEAP(
        in PointType
            point_type)
raises(InvalidTarget);

SpecifiedPosition
    getVSEAPPosition(
        in ReferencePoint
            reference_point)
raises(InvalidTarget);

UserControls
    getMovingAbility()
raises(InvalidTarget);

UserControls
    getResizingAbility()
raises(InvalidTarget);

UserControls
    getScalingAbility()
raises(InvalidTarget);

UserControls
    getScrollingAbility()
raises(InvalidTarget);

void
    setSelectability(
        in unsigned short
            min_number_of_selections,
        in unsigned short
            max_number_of_selections)
raises(InvalidTarget, InvalidParameter);

void
    setSelectionStatus(
        in SelectionStatusValue
            selection_state)
raises(InvalidTarget);

void
    setSelectionPresentationEffectResponsibility(
        in Responsibility
            the_responsibility)
raises(InvalidTarget);
```

```
    void
        getSelectability(
             out unsigned short
                 min_number_of_selections,
             out unsigned short
                 max_number_of_selections)
    raises(InvalidTarget);

    EffectiveSelectability
        getEffectiveSelectability()
    raises(InvalidTarget);

    SelectionStatusValue
        getSelectionStatus()
    raises(InvalidTarget);

    SelectionModeValue
        getSelectionMode()
    raises(InvalidTarget);

    Responsibility
        getSelectionPresentationEffectResponsibility()
    raises(InvalidTarget);

    void
        setModifiability(
            in unsigned short
                min_number_of_modifications,
            in unsigned short
                max_number_of_modifications)
    raises(InvalidTarget, InvalidParameter);

    void
        setModificationStatus(
            in ModificationStatusValue
                modification_state)
    raises(InvalidTarget);

    void
        setModificationPresentationEffectResponsibility(
            in Responsibility
                the_responsibility)
    raises(InvalidTarget);

    void
        getModifiability(
             out unsigned short
                 min_numbers_of_modifications,
             out unsigned short
                 max_numbers_of_modifications)
    raises(InvalidTarget);

    EffectiveModifiability
        getEffectiveModifiability()
    raises(InvalidTarget);

    ModificationStatusValue
        getModificationStatus()
    raises(InvalidTarget);

    ModificationModeValue
        getModificationMode()
    raises(InvalidTarget);

    Responsibility
        getModificationPresentationEffectResponsibility()
    raises(InvalidTarget);

    void
        setNoInteractionStyle()
    raises(InvalidTarget);

};
```

7.1.22 Objet `RtComponent`

Aucune opération spécifique n'est définie pour l'objet `RtComponent.` Cet objet est un héritage de l'objet `RtComponentOrSocket` et de l'objet `RtObject`.

7.1.22.1 Description IDL

```
interface RtComponent: RtComponentOrSocket, RtObject {};
```

7.1.23 Objet `RtCompositeOrStructuralSocket`

Le sous-paragraphe qui suit définit les opérations de l'objet `RtCompositeOrStructuralSocket.`

7.1.23.1 Opération `setResizingStrategy`

Résumé:

Interface:	`RtCompositeOrStructuralSocket`	
Opération:	`setResizingStrategy`	
Résultat:	`void`	
Entrée:	`ResizingStrategy`	`resizing_strategy`
Exception:	`InvalidTarget`	

Description:

Cette opération spécifie la stratégie de redimensionnement de la taille perceptible que doit posséder un composant rt-composite ou un réceptacle structurel en ce qui concerne la modification des tailles visibles des réceptacles fils qui possèdent un espace générique relatif parent (PRGS, *parent relative generic space*).

L'opération `setResizingStrategy` déclenche l'exécution de l'action élémentaire "positionner stratégie de redimensionnement" avec comme cible unique le composant rt-composite lié ou le réceptacle structurel lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 53.2.2 de la norme MHEG.

Le paramètre `resizing_strategy` spécifie la valeur du paramètre "stratégie de redimensionnement" de l'action "positionner stratégie de redimensionnement".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.23.2 Opération `getResizingStrategy`

Résumé:

Interface:	`RtCompositeOrStructuralSocket`
Opération:	`getResizingStrategy`
Résultat:	`ResizingStrategy`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la valeur de stratégie de redimensionnement du composant rt-composite ou du réceptacle structurel.

L'opération `getResizingStrategy` déclenche l'exécution de l'action élémentaire "extraire stratégie de redimensionnement" avec comme cible unique le composant rt-composite lié ou le réceptacle structurel lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 53.4.3 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.23.3 **Opération** `setAudibleCompositionEffect`

Résumé:

Interface:	`RtCompositeOrStructuralSocket`	
Opération:	`setAudibleCompositionEffect`	
Résultat:	`void`	
Entrée:	`unsigned short`	`audible_effect`
Entrée:	`unsigned long`	`transition_duration`
Exception:	`InvalidTarget`	

Description:

Cette opération spécifie l'effet de composition audible d'un composant rt-composite ou d'un réceptacle structurel. Cet effet doit se propager vers leurs réceptacles descendants qui possèdent un espace PRGS. L'opération est utilisée pour calculer le volume d'origine (OV, *original volume*) effectif des réceptacles descendants qui possèdent un espace PRGS.

L'opération `setAudibleCompositionEffect` déclenche l'exécution de l'action élémentaire "positionner effet de composition audible" avec comme cible unique le composant rt-composite lié ou le réceptacle structurel lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 54.2.2 de la norme MHEG.

Le paramètre `audible_effect` spécifie la valeur du paramètre "effet audible" de l'action "positionner effet de composition audible".

Le paramètre `transition_duration` spécifie la valeur du paramètre "durée de transition" de l'action "positionner effet de composition audible".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.23.4 **Opération** `getAudibleCompositionEffect`

Résumé:

Interface:	`RtCompositeOrStructuralSocket`
Opération:	`getAudibleCompositionEffect`
Résultat:	`unsigned short`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la valeur de l'effet de composition audible du composant rt-composite ou d'un réceptacle structurel. Cet effet est exprimé sous la forme d'un pourcentage et utilisé pour déterminer le volume d'origine effectif des réceptacles fils du composant rt-composite ou du réceptacle structurel qui possèdent comme espace PRGS le composant rt-composite ou le réceptacle structurel. Cet effet s'applique d'une manière récursive aux réceptacles structurels descendants qui ont comme espace PRGS le composant rt-composite ou le réceptacle structurel, et ainsi de suite.

L'opération `getAudibleCompositionEffect` déclenche l'exécution de l'action élémentaire "extraire effet de composition audible" avec comme cible unique le composant rt-composite lié ou le réceptacle structurel lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 54.3.3 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.23.5 **Opération** `getNumberOfSelectedSockets`

Résumé:

Interface:	`RtCompositeOrStructuralSocket`
Opération:	`getNumberOfSelectedSockets`
Résultat:	`unsigned short`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la valeur de l'attribut "nombre de réceptacles sélectionnés" du composant rt-composite ou du réceptacle structurel.

L'opération `getNumberOfSelectedSockets` déclenche l'exécution de l'action élémentaire "extraire nombre de réceptacles sélectionnés" avec comme cible unique le composant rt-composite lié ou le réceptacle structurel lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 57.3.5 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.23.6 Opération `getNumberOfModifiedSockets`

Résumé:

Interface:	`RtCompositeOrStructuralSocket`
Opération:	`getNumberOfModifiedSockets`
Résultat:	`unsigned short`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la valeur de l'attribut "nombre de réceptacles modifiés" du composant rt-composite ou du réceptacle structurel.

L'opération `getNumberOfModifiedSockets` déclenche l'exécution de l'action élémentaire "extraire nombre de réceptacles modifiés" avec comme cible unique le composant rt-composite lié ou le réceptacle structurel lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 58.3.5 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.23.7 Opération `setMenuInteractionStyle`

Résumé:

Interface:	`RtCompositeOrStructuralSocket`	
Opération:	`setMenuInteractionStyle`	
Résultat:	`void`	
Entrée:	`Orientation`	`upper_menu_orientation`
Entrée:	`sequence <Association>`	`list_of_associations`
Exception:	`InvalidTarget`	
Exception:	`InvalidParameter`	

Description:

Cette opération assigne le style d'interaction de menu à un composant rt-composite ou à un réceptacle structurel. Cette opération définira un style qui concerne l'ensemble du composant rt-composite ou du réceptacle structurel, c'est-à-dire toutes les générations.

L'opération `setMenuInteractionStyle` déclenche l'exécution de l'action élémentaire "positionner style de menu" avec comme cible unique le composant rt-composite lié ou le réceptacle structurel lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 59.2.4 de la norme MHEG.

Le paramètre `upper_menu_orientation` spécifie la valeur du paramètre "orientation du menu supérieur" de l'action "positionner style de menu".

Le paramètre `list_of_associations` spécifie la valeur du paramètre "liste d'associations" de l'action "positionner style de menu".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.23.8 **Opération** `setScrollingListInteractionStyle`

Résumé:

Interface:	`RtCompositeOrStructuralSocket`	
Opération:	`setScrollingListInteractionStyle`	
Résultat:	`void`	
Entrée:	`PerceptibleReference`	`background`
Entrée:	`unsigned short`	`visible_items_number`
Entrée:	`SocketTail`	`first_item`
Entrée:	`Separator`	`the_separator`
Entrée:	`Orientation`	`the_orientation`
Entrée:	`SliderSide`	`slider_side`
Entrée:	`PerceptibleReference`	`slider`
Entrée:	`PerceptibleReference`	`slider_cursor`
Entrée:	`PerceptibleReference`	`slider_background`
Entrée:	`long`	`slider_min_value`
Entrée:	`long`	`slider_max_value`
Exception:	`InvalidTarget`	
Exception:	`InvalidParameter`	

Description:

Cette opération assigne le style d'interaction de liste déroulante à un composant rt-composite ou à un réceptacle structurel. Cette opération définira le style qui concerne la première génération et uniquement les réceptacles présentables fils du composant rt-composite ou du réceptacle structurel.

L'opération `setScrollingListInteractionStyle` déclenche l'exécution de l'action élémentaire "positionner style de liste déroulante" avec comme cible unique le composant rt-composite lié ou le réceptacle structurel lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 59.2.5 de la norme MHEG.

Le paramètre `background` spécifie la valeur du paramètre "arrière-plan" de l'action "positionner style de liste déroulante".

Le paramètre `visible_items_number` spécifie la valeur du paramètre "nombre d'éléments visibles" de l'action "positionner style de liste déroulante".

Le paramètre `first_item` spécifie la valeur du paramètre "premier élément" de l'action "positionner style de liste déroulante".

Le paramètre `the_separator` spécifie la valeur du paramètre "séparateur" de l'action "positionner style de liste déroulante".

Le paramètre `the_orientation` spécifie la valeur du paramètre "orientation de liste déroulante" de l'action "positionner style de liste déroulante".

Le paramètre `slider_side` spécifie la valeur du paramètre "ascenseur" de l'action "positionner style de liste déroulante".

Les paramètres `slider`, `slider_cursor`, `slider_background`, `slider_min_value`, `slider_max_value` spécifient les valeurs des paramètres de l'action"positionner style ascenseur" qui est contenue dans l'action "positionner style de liste déroulante".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.23.9 Description IDL

```
interface RtCompositeOrStructuralSocket                    {

    void
        setResizingStrategy(
            in ResizingStrategy
                resizing_strategy)
    raises(InvalidTarget);

    ResizingStrategy
        getResizingStrategy()
    raises(InvalidTarget);

    void
        setAudibleCompositionEffect(
            in unsigned short
                audible_effect,
            in unsigned long
                transition_duration)
    raises(InvalidTarget);

    unsigned short
        getAudibleCompositionEffect()
    raises(InvalidTarget);

    unsigned short
        getNumberOfSelectedSockets()
    raises(InvalidTarget);

    unsigned short
        getNumberOfModifiedSockets()
    raises(InvalidTarget);

    void
        setMenuInteractionStyle(
            in Orientation
                upper_menu_orientation,
            in sequence <Association>
                list_of_associations)
    raises(InvalidTarget, InvalidParameter);

    void
        setScrollingListInteractionStyle(
            in PerceptibleReference
                background,
            in unsigned short
                visible_items_number,
            in SocketTail
                first_item,
            in Separator
                the_separator,
            in Orientation
                the_orientation,
            in SliderSide
                slider_side,
            in PerceptibleReference
                slider,
            in PerceptibleReference
                slider_cursor,
            in PerceptibleReference
                slider_background,
            in long
                slider_min_value,
            in long
                slider_max_value)
    raises(InvalidTarget, InvalidParameter);

};
```

7.1.24 Objet RtComposite

Aucune opération spécifique n'est définie pour l'objet `RtComposite`. Cet objet est un héritage de l'objet `RtCompositeOrStructuralSocket` et de l'objet `RtComponent`.

7.1.24.1 Description IDL

```
interface RtComposite: RtCompositeOrStructuralSocket, RtComponent {};
```

7.1.25 Objet `StructuralSocket`

Aucune opération spécifique n'est définie pour l'objet `StructuralSocket.` Cet objet est un héritage de l'objet `RtCompositeOrStructuralSocket` et de l'objet `Socket`.

7.1.25.1 Description IDL

```
interface StructuralSocket: RtCompositeOrStructuralSocket, Socket {};
```

7.1.26 Objet `RtGenericContentOrPresentableSocket`

Le sous-paragraphe qui suit définit les opérations de l'objet `RtGenericContentOrPresentableSocket`.

7.1.26.1 Opération `setAudibleVolume`

Résumé:

Interface:	`RtGenericContentOrPresentableSocket`	
Opération:	`setAudibleVolume`	
Résultat:	`void`	
Entrée:	`AudibleVolume`	`audible_volume`
Entrée:	`unsigned long`	`transition_duration`
Exception:	`InvalidTarget`	
Exception:	`InvalidParameter`	

Description:

Cette action spécifie le volume audible d'un rt-contenu, d'un rt-contenu multiplexé, d'un réceptacle présentable ou d'un réceptacle multiplexé présentable.

L'opération `setAudibleVolume` déclenche l'exécution de l'action élémentaire "positionner volume audible" avec comme cible unique le rt-contenu lié, le rt-contenu multiplexé lié, le réceptacle présentable lié ou le réceptacle multiplexé présentable lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 54.2.1 de la norme MHEG.

Le paramètre `audible_volume` spécifie la valeur du paramètre "volume audible" de l'action "positionner volume audible".

Le paramètre `transition_duration` spécifie la valeur du paramètre "durée de transition" de l'action "positionner volume audible".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.26.2 Opération `getInitialOriginalAudibleVolume`

Résumé:

Interface:	`RtGenericContentOrPresentableSocket`
Opération:	`getInitialOriginalAudibleVolume`
Résultat:	`unsigned long`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la valeur du volume audible original initial du rt-contenu, du rt-contenu multiplexé, du réceptacle présentable ou du réceptacle multiplexé présentable. Ce volume initial est exprimé en unité générique originale de volume audible au sein de l'intervalle appartenant au domaine défini par le domaine original de volume audible.

L'opération `getInitialOriginalAudibleVolume` déclenche l'exécution de l'action élémentaire "extraire volume IOV" avec comme cible unique le rt-contenu lié, le rt-contenu multiplexé lié, le réceptacle présentable lié ou le réceptacle multiplexé présentable lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 54.3.1 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.26.3 Opération `getCurrentOriginalAudibleVolume`

Résumé:

Interface: `RtGenericContentOrPresentableSocket`
Opération: `getCurrentOriginalAudibleVolume`
Résultat: `unsigned long`
Exception: `InvalidTarget`

Description:

Cette opération extrait la valeur du volume audible original courant du rt-contenu, du rt-contenu multiplexé, du réceptacle présentable ou du réceptacle multiplexé présentable. Ce volume courant est exprimé en unités de volume audible générique original au sein de l'intervalle défini par le domaine de volume audible original.

L'opération `getCurrentOriginalAudibleVolume` déclenche l'exécution de l'action élémentaire "extraire volume OV courant" avec comme cible unique le rt-contenu lié, le rt-contenu multiplexé lié, le réceptacle présentable lié ou le réceptacle multiplexé présentable lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 54.3.2 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.26.4 Opération `getEffectiveOriginalAudibleVolume`

Résumé:

Interface: `RtGenericContentOrPresentableSocket`
Opération: `getEffectiveOriginalAudibleVolume`
Résultat: `unsigned long`
Exception: `InvalidTarget`

Description:

Cette opération extrait la valeur du volume audible original effectif du rt-contenu, du rt-contenu multiplexé, du réceptacle présentable ou du réceptacle multiplexé présentable. Ce volume est exprimé en unités de volume audible générique original au sein de l'intervalle défini par le domaine de volume audible original. Il est calculé par le moteur MHEG en utilisant le volume audible original courant et l'effet de composition audible.

L'opération `getEffectiveOriginalAudibleVolume` déclenche l'exécution de l'action élémentaire "extraire volume OV effectif" avec comme cible unique le rt-contenu lié, le rt-contenu multiplexé lié, le réceptacle présentable lié ou le réceptacle multiplexé présentable lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 54.3.4 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.26.5 Opération `getPerceptibleAudibleVolume`

Résumé:

Interface: `RtGenericContentOrPresentableSocket`
Opération: `getPerceptibleAudibleVolume`
Résultat: `unsigned long`
Exception: `InvalidTarget`

Description:

Cette opération extrait la valeur du volume audible perceptible original du rt-contenu, du rt-contenu multiplexé, du réceptacle présentable ou du réceptacle multiplexé présentable dans le canal assigné. Ce volume est exprimé en unités de volume audible générique de canal au sein de l'intervalle défini par le domaine de volume audible du canal. Il est calculé par le moteur MHEG et correspond à une projection du volume audible original effectif dans l'espace générique du canal.

L'opération `getPerceptibleAudibleVolume` déclenche l'exécution de l'action élémentaire "extraire volume audible perceptible" avec comme cible unique le rt-contenu lié, le rt-contenu multiplexé lié, le réceptacle présentable lié ou le réceptacle multiplexé présentable lié.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 54.3.5 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.26.6 Opération `setButtonInteractionStyle`

Résumé:

Interface:	`RtGenericContentOrPresentableSocket`	
Opération:	`setButtonInteractionStyle`	
Résultat:	`void`	
Entrée:	`PresentationState`	`initial_state`
Entrée:	`AlternatePresentation`	`alternate_presentation_1`
Entrée:	`AlternatePresentation`	`alternate_presentation_2`
Entrée:	`AlternatePresentation`	`alternate_presentation_3`
Exception:	`InvalidTarget`	
Exception:	`InvalidParameter`	

Description:

Cette opération assigne le style d'interaction de bouton à un rt-contenu, un rt-contenu multiplexé, un réceptacle présentable ou un réceptacle multiplexé présentable.

L'opération `setButtonInteractionStyle` déclenche l'exécution de l'action élémentaire "positionner style de bouton" avec comme cible unique le rt-contenu lié, le rt-contenu multiplexé lié, le réceptacle présentable lié ou le réceptacle multiplexé présentable lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 59.2.1 de la norme MHEG.

Le paramètre `initial_state` spécifie la valeur du paramètre "état initial" de l'action "positionner style de bouton".

Le paramètre `alternate_presentation_1` spécifie la valeur du paramètre "autre présentation 1" de l'action "positionner style de bouton".

Le paramètre `alternate_presentation_2` spécifie la valeur du paramètre "autre présentation 2" de l'action "positionner style de bouton".

Le paramètre `alternate_presentation_3` spécifie la valeur du paramètre "autre présentation 3" de l'action "positionner style de bouton d'interaction".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.26.7 Description IDL

```
interface RtGenericContentOrPresentableSocket                {

    void
        setAudibleVolume(
            in AudibleVolume
                audible_volume,
            in unsigned long
                transition_duration)
    raises(InvalidTarget, InvalidParameter);

    unsigned long
        getInitialOriginalAudibleVolume()
    raises(InvalidTarget);

    unsigned long
        getCurrentOriginalAudibleVolume()
    raises(InvalidTarget);

    unsigned long
        getEffectiveOriginalAudibleVolume()
    raises(InvalidTarget);

    unsigned long
        getPerceptibleAudibleVolume()
    raises(InvalidTarget);

    void
        setButtonInteractionStyle(
            in PresentationState
                initial_state,
            in AlternatePresentation
                alternate_presentation_1,
            in AlternatePresentation
                alternate_presentation_2,
            in AlternatePresentation
                alternate_presentation_3)
    raises(InvalidTarget, InvalidParameter);

};
```

7.1.27 Objet RtGenericContent

Aucune opération spécifique n'est définie pour l'objet `RtGenericContent.` Cet objet est un héritage de l'objet `RtGenericContentOrPresentableSocket` et de l'objet `RtComponent.`

7.1.27.1 Description IDL

```
interface RtGenericContent: RtGenericContentOrPresentableSocket, RtComponent {};
```

7.1.28 Objet GenericPresentableSocket

Aucune opération spécifique n'est définie pour l'objet `GenericPresentableSocket.` Cet objet est un héritage de l'objet `RtGenericContentOrPresentableSocket` et de l'objet `Socket.`

7.1.28.1 Description IDL

```
interface GenericPresentableSocket: RtGenericContentOrPresentableSocket, Socket
{};
```

7.1.29 Objet RtContentOrPresentableSocket

Le sous-paragaphe qui suit définit les opérations de l'objet `RtContentOrPresentableSocket.`

7.1.29.1 **Opération** `setSliderInteractionStyle`

Résumé:

Interface:	`RtContentOrPresentableSocket`	
Opération:	`setSliderInteractionStyle`	
Résultat:	`void`	
Entrée:	`PerceptibleReference`	`cursor`
Entrée:	`PerceptibleReference`	`background`
Entrée:	`Orientation`	`the_orientation`
Entrée:	`short`	`min_value`
Entrée:	`short`	`max_value`
Exception:	`InvalidTarget`	
Exception:	`InvalidParameter`	

Description:

Cette opération assigne le style d'interaction de curseur de défilement à un rt-contenu ou un réceptacle présentable créé à partir d'un objet contenu modèle qui contient un nombre générique "donnée de contenu".

L'opération `setSliderStyle` déclenche l'exécution de l'action élémentaire "positionner style de curseur de défilement" avec comme cible unique le rt-contenu lié ou le réceptacle présentable lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 59.2.2 de la norme MHEG.

Le paramètre `cursor` spécifie la valeur du paramètre "curseur" de l'action "positionner style de curseur de défilement".

Le paramètre `background` spécifie la valeur du paramètre "arrière-plan" de l'action "positionner style de curseur de défilement".

Le paramètre `the_orientation` spécifie la valeur du paramètre "orientation" de l'action "positionner style de curseur de défilement".

Le paramètre `min_value` spécifie la valeur du paramètre "valeur minimale" de l'action "positionner style de curseur de défilement".

Le paramètre `max_value` spécifie la valeur du paramètre "valeur maximale" de l'action "positionner style de curseur de défilement".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.29.2 **Opération** `setEntryFieldInteractionStyle`

Résumé:

Interface:	`RtContentOrPresentableSocket`	
Opération:	`setEntryFieldInteractionStyle`	
Résultat:	`void`	
Entrée:	`EchoStyle`	`echo_style`
Entrée:	`PerceptibleReference`	`background`
Exception:	`InvalidTarget`	
Exception:	`InvalidParameter`	

Description:

Cette opération assigne le style d'interaction du champ d'entrée à un rt-contenu ou un réceptacle présentable créé à partir d'un modèle d'objet contenu qui contient un nombre générique ou une chaîne générique comme "donnée de contenu".

L'opération `setEntryFieldInteractionStyle` déclenche l'exécution de l'action élémentaire "positionner style de champ d'entrée" avec comme cible unique le rt-contenu lié ou le réceptacle présentable lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 59.2.3 de la norme MHEG.

Le paramètre `echo_style` spécifie la valeur du paramètre "style d'écho" de l'action "positionner style de champ d'entrée".

Le paramètre `background` spécifie la valeur du paramètre "arrière-plan" de l'action "positionner style de champ d'entrée".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.29.3 Description IDL

```
interface RtContentOrPresentableSocket                    {

    void
        setSliderInteractionStyle(
            in PerceptibleReference
                cursor,
            in PerceptibleReference
                background,
            in Orientation
                the_orientation,
            in short
                min_value,
            in short
                max_value)
    raises(InvalidTarget, InvalidParameter);

    void
        setEntryFieldInteractionStyle(
            in EchoStyle
                echo_style,
            in PerceptibleReference
                background)
    raises(InvalidTarget, InvalidParameter);

};
```

7.1.30 Objet `RtContent`

Aucune opération spécifique n'est définie pour l'objet `RtContent`. Cet objet est un héritage de l'objet `RtContentOrPresentableSocket` et de l'objet `RtGenericContent`.

7.1.30.1 Description IDL

```
interface RtContent: RtContentOrPresentableSocket, RtGenericContent {};
```

7.1.31 Objet `PresentableSocket`

Aucune opération spécifique n'est définie pour l'objet `PresentableSocket.` Cet objet est un héritage de l'objet `RtContentOrPresentableSocket` et de l'objet `GenericPresentableSocket.`

7.1.31.1 Description IDL

```
interface PresentableSocket: RtContentOrPresentableSocket,
GenericPresentableSocket {};
```

7.1.32 Objet `RtMultiplexedContentOrPresentableSocket`

Le sous-paragraphe qui suit définit les opérations de l'objet `RtMultiplexedContentOrPresentableSocket.`

7.1.32.1 Opération `setStreamChoice`

Résumé:

Interface:	`RtMultiplexedContentOrPresentableSocket`
Opération:	`setStreamChoice`
Résultat:	`void`
Entrée:	`StreamIdentifier stream_identifier`
Exception:	`InvalidTarget`
Exception:	`InvalidParameter`

Description:

Cette opération spécifie un flux à choisir dans les données multiplexées et associé au rt-contenu multiplexé ou au réceptacle multiplexé présentable. Une fois qu'un flux choisi pour un rt-contenu multiplexé ou un réceptacle multiplexé présentable devient actif, celui-ci est responsable de la présentation de ce flux.

L'opération `setStreamChoice` déclenche l'exécution de l'action élémentaire "positionner choix de flux" avec comme cible unique le rt-contenu multiplexé ou le réceptacle multiplexé présentable.

Le paramètre `stream_identifier` spécifie la valeur du paramètre "choix de flux" de l'action "positionner choix de flux".

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 55.2.1 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.32.2 Opération `getStreamChosen`

Résumé:

Interface:	`RtMultiplexedContentOrPresentableSocket`
Opération:	`getStreamChosen`
Résultat:	`StreamValue`
Exception:	`InvalidTarget`

Description:

Cette opération extrait la valeur du flux choisi pour le rt-contenu multiplexé ou le réceptacle multiplexé présentable.

Le paramètre `getStreamChosen` déclenche l'exécution de l'action élémentaire "extraire flux choisi" avec comme cible unique le rt-contenu multiplexé ou le réceptacle multiplexé présentable.

L'effet de l'action sur la cible, la sémantique des paramètres, le calcul des résultats et les conditions d'erreur qui activent des exceptions sont définis au 55.3.1 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.32.3 Description IDL

```
interface RtMultiplexedContentOrPresentableSocket          {

    void
        setStreamChoice(
            in StreamIdentifier
                stream_identifier)
    raises(InvalidTarget, InvalidParameter);

    StreamValue
        getStreamChosen()
    raises(InvalidTarget);

};
```

7.1.33 Objet RtMultiplexedContent

Aucune opération spécifique n'est définie pour l'objet RtMultiplexedContent. Cet objet est un héritage de l'objet RtMultiplexedContentOrPresentableSocket et de l'objet RtGenericContent.

7.1.33.1 Description IDL

```
interface RtMultiplexedContent: RtMultiplexedContentOrPresentableSocket,
RtGenericContent {};
```

7.1.34 Objet MultiplexedPresentableSocket

Aucune opération spécifique n'est définie pour l'objet MultiplexedPresentableSocket. Cet objet est un héritage de l'objet RtMultiplexedContentOrPresentableSocket et de l'objet GenericPresentableSocket.

7.1.34.1 Description IDL

```
interface MultiplexedPresentableSocket: RtMultiplexedContentOrPresentableSocket,
GenericPresentableSocket {};
```

7.1.35 Objet Channel

Le sous-paragraphe qui suit définit les opérations de l'objet Channel. Cet objet est un héritage de l'objet Entity.

7.1.35.1 Opération bind

Résumé:

Interface:	Channel	
Opération:	bind	
Résultat:	ChannelIdentifier	
Entrée:	ChannelReference	channel_reference
Exception:	AlreadyBound	
Exception:	InvalidTarget	

Description:

Cette opération lie l'instance Channel (une instance d'objet interface) à un canal (une entité MHEG).

Le paramètre channel_reference spécifie la référence du canal.

L'opération renvoie l'identificateur du canal lié.

L'exception AlreadyBound est activée lorsque l'instance d'objet d'interface est déjà liée à une entité MHEG.

L'exception InvalidTarget est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre period renvoie la période dans laquelle se trouve la cible.

7.1.35.2 Opération unbind

Résumé:

Interface:	Channel
Opération:	unbind
Résultat:	void
Exception:	NotBound

Description:

Cette opération supprime le lien entre l'instance Channel (une instance d'objet interface) et un canal (une entité MHEG).

L'exception NotBound est activée lorsque l'instance d'objet d'interface n'est pas liée à une entité MHEG.

7.1.35.3 Opération `new`

Résumé:

Interface:	`Channel`	
Opération:	`new`	
Résultat:	`ChannelIdentifier`	
Entrée:	`ChannelReference`	`channel_reference`
Entrée:	`OriginalDefDeclaration`	`original_definition_declaration`
Exception:	`AlreadyBound`	
Exception:	`InvalidTarget`	

Description:

Cette opération permet la création d'un canal par le moteur MHEG.

L'opération `new` déclenche l'exécution de l'action élémentaire "nouveau canal" avec comme cible unique un canal unique.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 61.2.1 de la norme MHEG.

Le paramètre `channel_reference` spécifie une référence à un canal.

Le paramètre `original_definition_declaration` spécifie la valeur du paramètre "déclaration de définition d'origine" de l'action "nouveau canal".

Cette opération lie l'instance Channel (une instance d'objet interface) avec le nouveau canal créé (une entité MHEG).

L'opération renvoie l'identificateur du nouveau canal créé lié à l'instance Channel.

L'exception `AlreadyBound` est activée lorsque l'instance d'objet d'interface est déjà liée à une entité MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.35.4 Opération `delete`

Résumé:

Interface:	`Channel`
Opération:	`delete`
Résultat:	`void`
Exception:	`NotBound`
Exception:	`InvalidTarget`

Description:

Cette opération permet la suppression d'un canal par le moteur MHEG.

L'opération `delete` déclenche l'exécution de l'action élémentaire "supprimer canal" avec comme cible unique un canal unique.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 61.2.2 de la norme MHEG.

Cette opération supprime implicitement la liaison de l'instance Channel (une instance d'objet interface) avec le nouveau canal supprimé (une entité MHEG).

L'exception `NotBound` est activée lorsque l'instance d'objet d'interface n'est pas liée à une entité MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.35.5 Opération `getRtAvailabilityStatus`

Résumé:

Interface: `Channel`
Opération: `getAvailability`
Résultat: `ChannelStatusValue`
Exception: `NotBound`
Exception: `InvalidTarget`

Description:

Cette opération extrait la disponibilité d'un canal pour le moteur MHEG.

L'opération `getAvailability` déclenche l'exécution de l'action élémentaire "extraire statut de disponibilité du canal" avec comme cible unique le canal lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 61.3.1 de la norme MHEG.

L'opération renvoie la disponibilité du canal lié à l'instance Channel. La valeur renvoyée est `NOT_AVAILABLE`, `PROCESSING` ou `AVAILABLE`.

Lorsque la valeur renvoyée est `NOT_AVAILABLE,` cette opération supprime implicitement la liaison entre l'instance Channel (une instance d'objet interface) et le canal (une entité MHEG).

L'exception `NotBound` est activée lorsque l'instance d'objet d'interface n'est pas liée à une entité MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.35.6 Opération `getIdentifier`

Résumé:

Interface: `Channel`
Opération: `getIdentifier`
Résultat: `ChannelIdentifier`
Exception: `NotBound`

Description:

Cette opération extrait l'identificateur du canal (une entité MHEG) lié à l'instance Channel (une instance d'objet interface).

L'exception `NotBound` est activée lorsque l'instance d'objet d'interface n'est pas liée à une entité MHEG.

7.1.35.7 Opération `kill`

Résumé:

Interface: `Channel`
Opération: `kill`
Résultat: `void`

Description:

Cette opération supprime l'instance Channel (une instance d'objet interface).

7.1.35.8 Opération `setPerceptability`

Résumé:

Interface: `Channel`
Opération: `setPerceptability`
Résultat: `void`
Entrée: `ChannelPerceptabilityValue` `channel_perceptability`
Exception: `InvalidTarget`

Description:

Cette opération permet de mettre un canal en service ou hors service. Elle est utilisée pour activer ou supprimer la perception d'un canal par un utilisateur.

L'opération `setPerceptability` déclenche l'exécution de l'action élémentaire "positionner la perceptibilité du canal" avec comme cible unique le canal lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 62.2.1 de la norme MHEG.

Le paramètre `channel_perceptability` spécifie la valeur du paramètre "capacité de perception" de l'action "positionner la perceptibilité du canal".

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.35.9 Opération `getPerceptability`

Résumé:

Interface: `Channel`
Opération: `getPerceptability`
Résultat: `ChannelPerceptabilityValue`
Exception: `InvalidTarget`

Description:

Cette opération extrait la perceptibilité d'un canal.

L'opération `getPerceptability` déclenche l'exécution de l'action élémentaire "extraire perceptibilité du canal" avec comme cible unique le canal lié.

L'effet de l'action sur la cible, la sémantique des paramètres et les conditions d'erreur qui activent des exceptions sont définis au 62.3.1 de la norme MHEG.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.35.10 Opération `getAssignedPerceptibles`

Résumé:

Interface: `Channel`
Opération: `getAssignedPerceptibles`
Résultat: `sequence<PerceptibleReference>`
Exception: `InvalidTarget`

Description:

Cette opération extrait les éléments perceptibles assignés au canal.

L'opération `getAssignedPerceptibles` n'a pas d'action élémentaire MHEG correspondante. Elle est symétrique de l'action "extraire espace RGS" qui récupère le canal assigné à un élément perceptible.

L'exception `InvalidTarget` est activée lorsque l'instance de l'objet n'est pas une cible valide pour l'exécution normale de l'action. Le membre `period` renvoie la période dans laquelle se trouve la cible.

7.1.35.11 Description IDL

```
interface Channel: Entity                                    {

    ChannelIdentifier
        bind(
            in ChannelReference
                channel_reference)
    raises(AlreadyBound, InvalidTarget);

    void
        unbind()
    raises(NotBound);
```

```
    ChannelIdentifier
        new(
            in ChannelReference
                channel_reference,
            in OriginalDefDeclaration
                original_definition_declaration)
    raises(AlreadyBound, InvalidTarget);

    void
       delete()
    raises(NotBound, InvalidTarget);

    ChannelStatusValue
        getAvailability()
    raises(NotBound, InvalidTarget);

    ChannelIdentifier
        getIdentifier()
    raises(NotBound);

    void
       kill();

    void
       setPerceptability(
           in ChannelPerceptabilityValue
               channel_perceptability)
    raises(InvalidTarget);

    ChannelPerceptabilityValue
        getPerceptability()
    raises(InvalidTarget);

    sequence<PerceptibleReference>
        getAssignedPerceptibles()
    raises(InvalidTarget);

};
```

7.1.36 Définition de paramètre

Les sous-paragraphes qui suivent définissent les paramètres utilisés par les primitives obligatoires.

```
//=========================================================================
typedef sequence<long> ApplicationIdentifier;

// Corresponding MHEG datatype: Object-Number
//=========================================================================
typedef long ObjectNumber;

// Interface: MhObject Operation: bind
// Interface: MhObject Operation: prepare
// Interface: MhObject Operation: getIdentifier
// Corresponding MHEG datatype: MHEG-Identifier
//=========================================================================
struct MHEGIdentifier {
    sequence<ApplicationIdentifier,1>
        application_identifier;
    ObjectNumber
        object_number;
};

// Corresponding MHEG datatype: Public-Identifier
//=========================================================================
typedef string PublicIdentifier;

// Corresponding MHEG datatype: System-Identifier
//=========================================================================
typedef string SystemIdentifier;
```

```
// Corresponding MHEG datatype: External-Long-Identifier
//=========================================================================
struct ExternalLongIdentifier {
    PublicIdentifier
        public_identifier;
    SystemIdentifier
        system_identifier;
};

// Corresponding MHEG datatype: Alias
//=========================================================================
typedef string Alias;

// Corresponding MHEG datatype: Container-Child-Reference
//=========================================================================
enum ContainerChildReference {
    CHILD,
    DESCENDANT
};

// Interface: MhObject Operation: getPreparationStatus
// Corresponding MHEG datatype: Preparation-Status-Value
//=========================================================================
enum PreparationStatusValue {
    READY,
    NOT_READY,
    PROCESSING
};

// Interface: MhMultiplexedContent Operation: setMultiplex
// Interface: MhMultiplexedContent Operation: setDemultiplex
// Interface: RtMultiplexedContentOrPresentableSocket Operation: setStreamChoice
// Corresponding MHEG datatype: Stream-Identifier
//=========================================================================
typedef sequence<long> StreamIdentifier;

// Corresponding MHEG datatype: Rt-Dynamic-Reference
//=========================================================================
enum RtDynamicReference {
    QUESTION_MARK,
    STAR
};

// Interface: RtObject Operation: getAvailabilityStatus
// Corresponding MHEG datatype: Rt-Availibility-Status-Value
//=========================================================================
enum RtAvailabilityStatusValue {
    RT_AVAILIBILITY_STATUS_VALUE_AVAILABLE,
    RT_AVAILIBILITY_STATUS_VALUE_NOT_AVAILABLE,
    RT_AVAILIBILITY_STATUS_VALUE_PROCESSING
};

// Interface: RtObject Operation: getRunningStatus
// Corresponding MHEG datatype: Running-Status-Value
//=========================================================================
enum RunningStatusValue {
    RUNNING_STATUS_VALUE_RUNNING,
    RUNNING_STATUS_VALUE_NOT_RUNNING,
    RUNNING_STATUS_VALUE_PROCESSING
};

// Interface: RtScript Operation: getTerminationStatus
// Corresponding MHEG datatype: Termination-Status-Value
//=========================================================================
enum TerminationStatusValue {
    TERMINATED,
    NOT_TERMINATED
};

// Interface: RtComponentOrSocket Operation: setRGS
// Interface: Channel Operation: getIdentifier
// Corresponding MHEG datatype: Channel-Identifier
//=========================================================================
typedef long ChannelIdentifier;
```

```
// Corresponding MHEG datatype: Priority-Level
//=========================================================================
enum PriorityLevel {
    INCREMENT_PRIORITY,
    DECREMENT_PRIORITY
};

// Interface: RtComponentOrSocket Operation: setVisibleDuration
// Corresponding MHEG datatype: Temporal-Position
//=========================================================================
enum TemporalPositionTag { SPECIFIED_TEMPORAL_POINT_TAG,
LOGICAL_TEMPORAL_PD_POINT_TAG };
union TemporalPosition
switch (TemporalPositionTag){
    case SPECIFIED_TEMPORAL_POINT_TAG:
        long
            specified_temporal_point;
    case LOGICAL_TEMPORAL_PD_POINT_TAG:
        long
            logical_temporal_PD_point;
};

// Interface: RtComponentOrSocket Operation: setCurrentTemporalPosition
// Corresponding MHEG datatype: Current-Temporal-Position
//=========================================================================
enum CurrentTemporalPositionTag {
CURRENT_TEMPORAL_POSITION_SPECIFIED_TEMPORAL_POINT_TAG,
CURRENT_TEMPORAL_POSITION_LOGICAL_TEMPORAL_VD_POINT_TAG };
union CurrentTemporalPosition
switch (CurrentTemporalPositionTag){
    case CURRENT_TEMPORAL_POSITION_SPECIFIED_TEMPORAL_POINT_TAG:
        long
            specified_temporal_point;
    case CURRENT_TEMPORAL_POSITION_LOGICAL_TEMPORAL_VD_POINT_TAG:
        long
            logical_temporal_vd_point;
};

// Interface: RtComponentOrSocket Operation: setTemporalTermination
// Interface: RtComponentOrSocket Operation: getTemporalTermination
// Corresponding MHEG datatype: Temporal-Termination
//=========================================================================
enum TemporalTermination {
    TEMPORAL_TERMINATION_FREEZE,
    TEMPORAL_TERMINATION_STOP
};

// Interface: RtComponentOrSocket Operation: setSpeed
// Corresponding MHEG datatype: Speed
//=========================================================================
enum SpeedTag { SPECIFIED_OGTR_TAG, SPEED_RATE_TAG, SCALING_FACTOR_TAG };
union Speed
switch (SpeedTag){
    case SPECIFIED_OGTR_TAG:
        long
            specified_OGTR;
    case SPEED_RATE_TAG:
        long
            speed_rate;
    case SCALING_FACTOR_TAG:
        long
            scaling_factor;
};

// Corresponding MHEG datatype: Timestone-Position
//=========================================================================
enum TimestonePositionTag { TIMESTONE_POSITION_SPECIFIED_TEMPORAL_POINT_TAG,
TIMESTONE_POSITION_LOGICAL_TEMPORAL_PD_POINT_TAG,
TIMESTONE_POSITION_LOGICAL_TEMPORAL_VD_POINT_TAG };
union TimestonePosition
switch (TimestonePositionTag){
    case TIMESTONE_POSITION_SPECIFIED_TEMPORAL_POINT_TAG:
        long
            specified_temporal_point;
```

```
    case TIMESTONE_POSITION_LOGICAL_TEMPORAL_PD_POINT_TAG:
        long
            logical_temporal_PD_point;
    case TIMESTONE_POSITION_LOGICAL_TEMPORAL_VD_POINT_TAG:
        long
            logical_temporal_VD_point;
};

// Interface: RtComponentOrSocket Operation: getVDLength
// Corresponding MHEG datatype: GT-Indicator
//=========================================================================
enum GTIndicator {
    OGTU,
    RGTU
};

// Corresponding MHEG datatype: Perceptible-Projection
//=========================================================================
enum PerceptibleProjectionTag { SPECIFIED_SIZE_TAG, IOGSR_SCALING_FACTOR_TAG,
COGSR_SCALING_FACTOR_TAG };
union PerceptibleProjection
switch (PerceptibleProjectionTag){
    case SPECIFIED_SIZE_TAG:
        long
            specified_size;
    case IOGSR_SCALING_FACTOR_TAG:
        long
            iogsr_scaling_factor;
    case COGSR_SCALING_FACTOR_TAG:
        long
            cogsr_scaling_factor;
};

// Interface: RtComponentOrSocket Operation: setAspectRatioPreserved
// Interface: RtComponentOrSocket Operation: getAspectRatio
// Corresponding MHEG datatype: Aspect-Ratio
//=========================================================================
enum AspectRatio {
    PRESERVED,
    NOT_PRESERVED
};

// Interface: RtComponentOrSocket Operation: setVisibleSize
// Interface: RtComponentOrSocket Operation: getVSGS
// Corresponding MHEG datatype: VSGS
//=========================================================================
enum VSGS {
    THIS,
    RELATIVE
};

// Corresponding MHEG datatype: Size-Attribute
//=========================================================================
enum SizeAttributeTag { SIZE_ATTRIBUTE_SPECIFIED_SIZE_TAG,
SIZE_ATTRIBUTE_IVS_RELATIVE_TAG, SIZE_ATTRIBUTE_CVS_RELATIVE_TAG };
union SizeAttribute
switch (SizeAttributeTag){
    case SIZE_ATTRIBUTE_SPECIFIED_SIZE_TAG:
        long
            specified_size;
    case SIZE_ATTRIBUTE_IVS_RELATIVE_TAG:
        long
            ivs_relative;
    case SIZE_ATTRIBUTE_CVS_RELATIVE_TAG:
        long
            cvs_relative;
};
```

```
// Interface: RtComponentOrSocket Operation: setVisibleSizesAdjustment
// Corresponding MHEG datatype: Adjustment-Axis
//=========================================================================
enum AdjustmentAxis {
    X_AXIS,
    Y_AXIS,
    Z_AXIS
};

// Corresponding MHEG datatype: Sub-Socket-Reference
//=========================================================================
enum SubSocketReference {
    SUB_SOCKET_REFERENCE_CHILD,
    SUB_SOCKET_REFERENCE_DESCENDANT,
    SUB_SOCKET_REFERENCE_QUESTION_MARK_CHILD,
    SUB_SOCKET_REFERENCE_QUESTION_MARK_DESCENDANT
};

// Interface: RtComponentOrSocket Operation: setBox
// Interface: RtComponentOrSocket Operation: getBox
// Corresponding MHEG datatype: Box-Constants
//=========================================================================
enum BoxConstants {
    PRESENTED,
    NOT_PRESENTED
};

// Interface: RtComponentOrSocket Operation: setAttachmentPointPosition
// Corresponding MHEG datatype: Reference-Type
//=========================================================================
enum ReferenceType {
    VSIAP,
    VSEAP
};

// Interface: RtComponentOrSocket Operation: setAttachmentPoint
// Interface: RtComponentOrSocket Operation: setAttachmentPointPosition
// Corresponding MHEG datatype: Attachment-Point-Type
//=========================================================================
enum AttachmentPointType {
    ATTACHMENT_POINT_TYPE_PSAP,
    ATTACHMENT_POINT_TYPE_VSIAP,
    ATTACHMENT_POINT_TYPE_VSEAP
};

// Interface: RtComponentOrSocket Operation: setVisibleSizesAlignment
// Corresponding MHEG datatype: Size-Border
//=========================================================================
enum SizeBorder {
    TOP,
    BOTTOM,
    RIGHT,
    LEFT,
    UPPER_Z,
    LOWER_Z,
    CENTER_X,
    CENTER_Y,
    CENTER_Z
};

// Interface: RtComponentOrSocket Operation: setMovingAbility
// Interface: RtComponentOrSocket Operation: setResizingAbility
// Interface: RtComponentOrSocket Operation: setScalingAbility
// Interface: RtComponentOrSocket Operation: setScrollingAbility
// Interface: RtComponentOrSocket Operation: getMovingAbility
// Interface: RtComponentOrSocket Operation: getResizingAbility
// Interface: RtComponentOrSocket Operation: getScalingAbility
// Interface: RtComponentOrSocket Operation: getScrollingAbility
// Corresponding MHEG datatype: User-Controls
//=========================================================================
enum UserControls {
    ALLOWED,
    NOT_ALLOWED
};
```

```
// Interface: RtComponentOrSocket Operation: getPS
// Corresponding MHEG datatype: GS-Indicator
//=========================================================================
enum GSIndicator {
    OGSU,
    RGSU
};

// Interface: RtComponentOrSocket Operation: getPSAP
// Interface: RtComponentOrSocket Operation: getVSIAP
// Corresponding MHEG datatype: Point-Type
//=========================================================================
enum PointType {
    RELATIVE_POINT,
    ABSOLUTE_POINT
};

// Interface: RtComponentOrSocket Operation: setSelectionStatus
// Interface: RtComponentOrSocket Operation: getSelectionStatus
// Corresponding MHEG datatype: Selection-Status-Value
//=========================================================================
enum SelectionStatusValue {
    SELECTED,
    NOT_SELECTED
};

// Interface: RtComponentOrSocket Operation:
//   setSelectionPresentationEffectResponsibility
// Interface: RtComponentOrSocket Operation:
//   getSelectionPresentationEffectResponsibility
// Interface: RtComponentOrSocket Operation:
//   setModificationPresentationEffectResponsibility
// Interface: RtComponentOrSocket Operation:
//   getModificationPresentationEffectResponsibility
// Corresponding MHEG datatype: Responsibility
//=========================================================================
enum Responsibility {
    MHEG_ENGINE,
    AUTHOR
};

// Interface: RtComponentOrSocket Operation: getEffectiveSelectability
// Corresponding MHEG datatype: Effective-Selectability
//=========================================================================
enum EffectiveSelectability {
    EFFECTIVELY_SELECTABLE,
    EFFECTIVELY_NOT_SELECTABLE
};

// Interface: RtComponentOrSocket Operation: setModificationStatus
// Interface: RtComponentOrSocket Operation: getModificationStatus
// Corresponding MHEG datatype: Modification-Status-Value
//=========================================================================
enum ModificationStatusValue {
    MODIFIED,
    MODIFYING,
    NOT_MODIFIED
};

// Interface: RtComponentOrSocket Operation: getEffectiveModifiability
// Corresponding MHEG datatype: Effective-Modifiability
//=========================================================================
enum EffectiveModifiability {
    EFFECTIVELY_MODIFIABLE,
    EFFECTIVELY_NOT_MODIFIABLE
};
```

```
// Interface: RtCompositeOrStructuralSocket Operation: setResizingStrategy
// Interface: RtCompositeOrStructuralSocket Operation: getResizingStrategy
// Corresponding MHEG datatype: Resizing-Strategy
//=========================================================================
enum ResizingStrategy {
    FIXED,
    MINIMUM,
    GROWS_ONLY
};

// Interface: RtCompositeOrStructuralSocket Operation: setMenuInteractionStyle
// Interface: RtCompositeOrStructuralSocket Operation:
//   setScrollingListInteractionStyle
// Interface: RtContentOrPresentableSocket Operation: setSliderInteractionStyle
// Corresponding MHEG datatype: Orientation
//=========================================================================
enum Orientation {
    HORIZONTAL,
    VERTICAL
};

// Corresponding MHEG datatype: Presentation-Persistence
//=========================================================================
enum PresentationPersistence {
    PERSISTENT,
    NOT_PERSISTENT
};

// Interface: RtCompositeOrStructuralSocket Operation:
//   setScrollingListInteractionStyle
// Corresponding MHEG datatype: Slider-Side
//=========================================================================
enum SliderSide {
    SIDE1,
    SIDE2
};

// Interface: RtGenericContentOrPresentableSocket Operation: setAudibleVolume
// Corresponding MHEG datatype: Audible-Volume
//=========================================================================
enum AudibleVolumeTag { SPECIFIED_VOLUME_TAG, LOGICAL_VOLUME_TAG,
IOV_SCALING_FACTOR_TAG, OV_SCALING_FACTOR_TAG };
union AudibleVolume
switch (AudibleVolumeTag){
    case SPECIFIED_VOLUME_TAG:
        long
            specified_volume;
    case LOGICAL_VOLUME_TAG:
        long
            logical_volume;
    case IOV_SCALING_FACTOR_TAG:
        long
            iov_scaling_factor;
    case OV_SCALING_FACTOR_TAG:
        long
            ov_scaling_factor;
};

// Interface: RtGenericContentOrPresentableSocket Operation:
//   setButtonInteractionStyle
// Corresponding MHEG datatype: Presentation-State
//=========================================================================
enum PresentationState {
    SELECTABLE_NOT_SELECTED,
    SELECTABLE_SELECTED,
    NOT_SELECTABLE_SELECTED,
    NOT_SELECTABLE_NOT_SELECTED
};

// Corresponding MHEG datatype: Echo-Mode
//=========================================================================
enum EchoMode {
    ITSELF,
    HIDDEN
};
```

```
// Interface: RtContentOrPresentableSocket Operation:
//   setEntryFieldInteractionStyle
// Corresponding MHEG datatype: Echo-Style
//=========================================================================
enum EchoStyleTag { MODE_TAG, SPECIFIED_TAG };
union EchoStyle
switch (EchoStyleTag){
    case MODE_TAG:
        EchoMode
            mode;
    case SPECIFIED_TAG:
        string
            specified;
};

// Corresponding MHEG datatype: Channel-Reference
//=========================================================================
enum ChannelReferenceTag { CHANNEL_IDENTIFIER_TAG, ALIAS_TAG,
NULL_CHANNEL_REFERENCE_TAG };
union ChannelReference
switch (ChannelReferenceTag){
    case CHANNEL_IDENTIFIER_TAG:
        ChannelIdentifier
            channel_identifier;
    case ALIAS_TAG:
        Alias
            alias;
};

// Corresponding MHEG datatype: Interval
//=========================================================================
struct Interval {
    sequence<long,1>
        start_point;
    sequence<long,1>
        end_point;
};

// Corresponding MHEG datatype: Generic-Volume-Range
//=========================================================================
struct GenericVolumeRange {
    sequence<long,1>
        maximum_volume;
    sequence<long,1>
        minimum_volume;
};

// Interface: Channel Operation: new
// Corresponding MHEG datatype: Original-Def-Declaration
//=========================================================================
struct OriginalDefDeclaration {
    sequence<long,1>
        generic_temporal_ratio;
    sequence<Interval,1>
        x_axis_interval;
    sequence<Interval,1>
        y_axis_interval;
    sequence<Interval,1>
        z_axis_interval;
    sequence<GenericVolumeRange,1>
        audible_volume_range_declaration;
};

// Interface: Channel Operation: getAvailability
// Corresponding MHEG datatype: Channel-Status-ValueCHANNEL-STATUS-VALUE-
//=========================================================================
enum ChannelStatusValue {
    CHANNEL_STATUS_VALUE_AVAILABLE,
    CHANNEL_STATUS_VALUE_NOT_AVAILABLE,
    CHANNEL_STATUS_VALUE_PROCESSING
};
```

```
// Interface: Channel Operation: setPerceptability
// Interface: Channel Operation: getPerceptability
// Corresponding MHEG datatype: Channel-Perceptability-Values
//=====================================================================
enum ChannelPerceptabilityValue {
    ON,
    OFF
};

// Interface: NotificationManager Operation: getNotification
// Interface: MhContent Operation: getData
// Corresponding MHEG datatype: Generic-Value
//=====================================================================
enum GenericValueTag { BOOLEAN_FIELD_TAG, NUMERIC_TAG, STRING_FIELD_TAG,
GENERIC_LIST_TAG, UNSPECIFIED_TAG };
union GenericValue
switch (GenericValueTag){
    case BOOLEAN_FIELD_TAG:
        boolean
            boolean_field;
    case NUMERIC_TAG:
        long
            numeric;
    case STRING_FIELD_TAG:
        string
            string_field;
    case GENERIC_LIST_TAG:
        sequence<GenericValue>
            generic_list;
};

// Corresponding MHEG datatype: Generic-String
//=====================================================================
enum GenericStringTag { GENERIC_STRING_CONSTANT_TAG,
GENERIC_STRING_UNSPECIFIED_TAG };
union GenericString
switch (GenericStringTag){
    case GENERIC_STRING_CONSTANT_TAG:
        string
            constant;
};

// Interface: Socket Operation: setVisibleDurationPosition
// Corresponding MHEG datatype: Visible-Duration
//=====================================================================
enum VisibleDurationPositionTag {
VISIBLE_DURATION_POSITION_SPECIFIED_TEMPORAL_POINT_TAG,
VISIBLE_DURATION_POSITION_LOGICAL_TEMPORAL_PD_POINT_TAG,
VISIBLE_DURATION_POSITION_LOGICAL_TEMPORAL_VD_POINT_TAG };
union VisibleDurationPosition
switch (VisibleDurationPositionTag){
    case VISIBLE_DURATION_POSITION_SPECIFIED_TEMPORAL_POINT_TAG:
        long
            specified_temporal_point;
    case VISIBLE_DURATION_POSITION_LOGICAL_TEMPORAL_PD_POINT_TAG:
        long
            logical_temporal_PD_point;
    case VISIBLE_DURATION_POSITION_LOGICAL_TEMPORAL_VD_POINT_TAG:
        long
            logical_temporal_VD_point;
};

// Interface: RtComponentOrSocket Operation: getRGS
// Corresponding MHEG datatype: none
//=====================================================================
enum RGSValueTag { RGS_VALUE_CHANNEL_IDENTIFIER_TAG, RGS_VALUE_NULL_CHANNEL_TAG,
RGS_VALUE_PRGS_TAG };
union RGSValue
switch (RGSValueTag){
    case RGS_VALUE_CHANNEL_IDENTIFIER_TAG:
        ChannelIdentifier
            channel_identifier;
};
```

```
// Corresponding MHEG datatype: Generic-Numeric
//=========================================================================
enum GenericNumericTag { GENERIC_NUMERIC_CONSTANT_TAG,
GENERIC_NUMERIC_UNSPECIFIED_TAG };
union GenericNumeric
switch (GenericNumericTag){
    case GENERIC_NUMERIC_CONSTANT_TAG:
        long
            constant;
};

// Interface: RtComponentOrSocket Operation: getSelectionMode
// Corresponding MHEG datatype: none
//=========================================================================
enum SelectionModeValueTag { USER_INTERACTION_TAG, NO_SELECTION_TAG,
MHEG_ACTION_TAG, USING_APPLICATION_ACTION_TAG };
union SelectionModeValue
switch (SelectionModeValueTag){
    case USER_INTERACTION_TAG:
        unsigned long
            user_interaction;
};

// Interface: RtComponentOrSocket Operation: getModificationMode
// Corresponding MHEG datatype: none
//=========================================================================
enum ModificationModeValueTag { MODIFICATION_MODE_VALUE_USER_INTERACTION_TAG,
MODIFICATION_MODE_VALUE_NO_MODIFICATION_TAG,
MODIFICATION_MODE_VALUE_MHEG_ACTION_TAG,
MODIFICATION_MODE_VALUE_USING_APPLICATION_ACTION_TAG,
MODIFICATION_MODE_VALUE_CHILD_TAG };
union ModificationModeValue
switch (ModificationModeValueTag){
    case MODIFICATION_MODE_VALUE_USER_INTERACTION_TAG:
        unsigned long
            user_interaction;
};

// Corresponding MHEG datatype: External-Identifier
//=========================================================================
enum ExternalIdentifierTag { EXTERNAL_LONG_ID_TAG, PUBLIC_ID_TAG,
SYSTEM_ID_TAG };
union ExternalIdentifier
switch (ExternalIdentifierTag){
    case EXTERNAL_LONG_ID_TAG:
        ExternalLongIdentifier
            external_long_id;
    case PUBLIC_ID_TAG:
        PublicIdentifier
            public_id;
    case SYSTEM_ID_TAG:
        SystemIdentifier
            system_id;
};

// Corresponding MHEG datatype: Container-Tail
//=========================================================================
struct ContainerTail {
    sequence<long>
        indexes;
    enum ContainerTailTag { INDEX_TAG, CONTAINER_CHILD_REF_TAG } tag;
    union ContainerTail
    switch (ContainerTailTag){
        case INDEX_TAG:
            long
                index;
        case CONTAINER_CHILD_REF_TAG:
            ContainerChildReference
                container_child_ref;
    } end;
};
```

```
// Corresponding MHEG datatype: Specified-Sizes
//=========================================================================
struct SpecifiedSizes {
    sequence<GenericNumeric,1>
        x_axis_length;
    sequence<GenericNumeric,1>
        y_axis_length;
    sequence<GenericNumeric,1>
        z_axis_length;
};

// Corresponding MHEG datatype: Attachment-Attribute
//=========================================================================
enum AttachmentAttributeTag { SPECIFIED_POSITION_TAG, LOGICAL_POSITION_TAG };
union AttachmentAttribute
switch (AttachmentAttributeTag){
    case SPECIFIED_POSITION_TAG:
        GenericNumeric
            specified_position;
    case LOGICAL_POSITION_TAG:
        GenericNumeric
            logical_position;
};

// Corresponding MHEG datatype: Length-Attribute
//=========================================================================
enum LengthAttributeTag { SPECIFIED_LENGTH_TAG, RELATIVE_LENGTH_TAG };
union LengthAttribute
switch (LengthAttributeTag){
    case SPECIFIED_LENGTH_TAG:
        GenericNumeric
            specified_length;
    case RELATIVE_LENGTH_TAG:
        GenericNumeric
            relative_length;
};

// Interface: RtComponentOrSocket Operation: getPS
// Interface: RtComponentOrSocket Operation: getPSAP
// Interface: RtComponentOrSocket Operation: getVS
// Interface: RtComponentOrSocket Operation: getVSIAP
// Interface: RtComponentOrSocket Operation: getVSIAPPosition
// Interface: RtComponentOrSocket Operation: getVSEAP
// Interface: RtComponentOrSocket Operation: getVSEAPPosition
// Corresponding MHEG datatype: Specified-Position
//=========================================================================
struct SpecifiedPosition {
    GenericNumeric
        x_point;
    GenericNumeric
        y_point;
    GenericNumeric
        z_point;
};

// Interface: RtComponentOrSocket Operation: setPresentationPriority
// Corresponding MHEG datatype: Presentation-Priority
//=========================================================================
enum PresentationPriorityTag { GENERIC_NUMERIC_TAG, PRIORITY_LEVEL_TAG };
union PresentationPriority
switch (PresentationPriorityTag){
    case GENERIC_NUMERIC_TAG:
        GenericNumeric
            generic_numeric;
    case PRIORITY_LEVEL_TAG:
        PriorityLevel
            priority_level;
};
```

```
// Interface: RtComponentOrSocket Operation: setTimestones
// Corresponding MHEG datatype: Timestone
//=========================================================================
struct Timestone {
    long
        timestone_identifier;
    TimestonePosition
        timestone_position;
};

// Interface: RtComponentOrSocket Operation: setVisibleSize
// Corresponding MHEG datatype: none
//=========================================================================
enum VSTag { X_SIZE_ATTRIBUTE_TAG, Y_SIZE_ATTRIBUTE_TAG, Z_SIZE_ATTRIBUTE_TAG };
union VS
switch (VSTag){
    case X_SIZE_ATTRIBUTE_TAG:
        SizeAttribute
            x_size_attribute;
    case Y_SIZE_ATTRIBUTE_TAG:
        SizeAttribute
            y_size_attribute;
    case Z_SIZE_ATTRIBUTE_TAG:
        SizeAttribute
            z_size_attribute;
};

// Interface: RtComponentOrSocket Operation: setAttachmentPoint
// Corresponding MHEG datatype: none
//=========================================================================
struct AttachmentPoint {
    sequence<AttachmentAttribute,1>
        x_attachment;
    sequence<AttachmentAttribute,1>
        y_attachment;
    sequence<AttachmentAttribute,1>
        z_attachment;
};

// Interface: RtComponentOrSocket Operation: setAttachmentPointPosition
// Corresponding MHEG datatype: Lengths
//=========================================================================
struct Lengths {
    sequence<LengthAttribute,1>
        x_length;
    sequence<LengthAttribute,1>
        y_length;
    sequence<LengthAttribute,1>
        z_length;
};

// Interface: RtMultiplexedContentOrPresentableSocket Operation: getStreamChosen
// Corresponding MHEG datatype: none
//=========================================================================
enum StreamValueTag { STREAM_IDENTIFIER_TAG, NO_STREAM_CHOSEN_TAG };
union StreamValue
switch (StreamValueTag){
    case STREAM_IDENTIFIER_TAG:
        StreamIdentifier
            stream_identifier;
};

// Interface: MhContent Operation: setData
// Corresponding MHEG datatype: Data-Element
//=========================================================================
struct DataElement {
    sequence<long>
        element_list_index;
    GenericValue
        generic_value;
};
```

```
// Interface: NotificationManager Operation: getNotification
// Interface: MhObject Operation: bind
// Interface: MhObject Operation: prepare
// Interface: MhGenericContent Operation: copy
// Corresponding MHEG datatype: Mh-Object-Reference
//=========================================================================
struct MhObjectReference {
enum MhObjectReferenceHeadTag { MHEG_IDENTIFIER_TAG, EXTERNAL_IDENTIFIER_TAG,
ALIAS_TAG, NULL_OBJECT_REF_TAG } head_tag;
union MhObjectReferenceHead
switch (MhObjectReferenceHeadTag){
    case MHEG_IDENTIFIER_TAG:
        MHEGIdentifier
            mheg_identifier;
    case EXTERNAL_IDENTIFIER_TAG:
        ExternalIdentifier
            external_identifier;
    case ALIAS_TAG:
        Alias
            alias;
} head;
enum MhObjectReferenceTailTag { CONTAINER_ELEMENT_REFERENCE_TAG,
OTHER_REFERENCE_TAG } tail_tag;
union MhObjectReferenceTail
switch (MhObjectReferenceTailTag){
    case CONTAINER_ELEMENT_REFERENCE_TAG:
        ContainerTail
            container_tail;
} tail;
};

// Interface: RtComponentOrSocket Operation: setPerceptibleSizeProjection
// Corresponding MHEG datatype: Perceptible-Size-Projection
//=========================================================================
struct PerceptibleSizeProjection {
    sequence<PerceptibleProjection,1>
        x_perceptible_size_projection;
    sequence<PerceptibleProjection,1>
        y_perceptible_size_projection;
    sequence<PerceptibleProjection,1>
        z_perceptible_size_projection;
};

// Corresponding MHEG datatype: Rt-Object-Number-Reference
//=========================================================================
enum RtObjectNumberReferenceTag { RT_OBJECT_NUMBER_TAG, RT_DYNAMIC_REFERENCE_TAG
};
union RtObjectNumberReference
switch (RtObjectNumberReferenceTag){
    case RT_OBJECT_NUMBER_TAG:
        long
            rt_object_number;
    case RT_DYNAMIC_REFERENCE_TAG:
        RtDynamicReference
            rt_dynamic_reference;
};

// Interface: RtObject Operation: bind
// Interface: RtObject Operation: new
// Corresponding MHEG datatype: Rt-Object-Reference
//=========================================================================
struct RtObjectReference {
    MhObjectReference
        model_object_reference;
    RtObjectNumberReference
        rt_object_number_reference;
};
```

```
// Corresponding MHEG datatype: Rt-Reference
//=========================================================================
enum RtReferenceTag { RT_REFERENCE_RT_OBJECT_REFERENCE_TAG,
RT_REFERENCE_ALIAS_TAG, RT_REFERENCE_NULL_RT_OBJECT_TAG };
union RtReference
switch (RtReferenceTag){
    case RT_REFERENCE_RT_OBJECT_REFERENCE_TAG:
        RtObjectReference
            rt_object_reference;
    case RT_REFERENCE_ALIAS_TAG:
        Alias
            alias;
};

// Corresponding MHEG datatype: Socket-Tail
//=========================================================================
struct SocketTail {
    sequence<long>
        indexes;
    enum SocketTailTag { INDEX_TAG, SUB_SOCKET_REF_TAG } tag;
    union SocketTail
    switch (SocketTailTag){
        case INDEX_TAG:
            long
                index;
        case SUB_SOCKET_REF_TAG:
            SubSocketReference
                sub_socket_ref;
    } end;
};

// Corresponding MHEG datatype: Indexed-Child-Socket
//=========================================================================
struct IndexedChildSocket {
    long
        index;
    SocketTail
        tail;
};

// Interface: Socket Operation: bind
// Interface: Socket Operation: getIdentification
// Corresponding MHEG datatype: Socket-Identification
//=========================================================================
struct SocketIdentification {
    RtReference
        rt_composite_reference;
    SocketTail
        socket_tail;
};

// Interface: Socket Operation: bind
// Corresponding MHEG datatype: Socket-Reference
//=========================================================================
enum SocketReferenceTag { SOCKET_REFERENCE_SOCKET_IDENT_TAG,
SOCKET_REFERENCE_ALIAS_TAG };
union SocketReference
switch (SocketReferenceTag){
    case SOCKET_REFERENCE_SOCKET_IDENT_TAG:
        SocketIdentification
            socket_ident;
    case SOCKET_REFERENCE_ALIAS_TAG:
        Alias
            alias;
};

// Corresponding MHEG datatype: Rt-Object-Socket-Reference
//=========================================================================
enum RtObjectSocketReferenceTag { RT_REFERENCE_TAG, SOCKET_REFERENCE_TAG };
union RtObjectSocketReference
switch (RtObjectSocketReferenceTag){
    case RT_REFERENCE_TAG:
        RtReference
            rt_reference;
```

```
    case SOCKET_REFERENCE_TAG:
        SocketReference
            socket_reference;
};

// Interface: RtCompositeOrStructuralSocket Operation:
//   setScrollingListInteractionStyle
// Interface: RtContentOrPresentableSocket Operation: setSliderInteractionStyle
// Interface: RtContentOrPresentableSocket Operation:
//   setEntryFieldInteractionStyle
// Interface: Channel Operation: getAssignedPerceptibles
// Corresponding MHEG datatype: Perceptible-Reference
//=========================================================================
enum PerceptibleReferenceTag { RT_COMPONENT_REFERENCE_TAG,
RT_SOCKET_REFERENCE_TAG };
union PerceptibleReference
switch (PerceptibleReferenceTag){
    case RT_COMPONENT_REFERENCE_TAG:
        RtReference
            rt_component_reference;
    case RT_SOCKET_REFERENCE_TAG:
        SocketReference
            rt_socket_reference;
};

// Interface: RtCompositeOrStructuralSocket Operation:
//   setScrollingListInteractionStyle
// Corresponding MHEG datatype: Separator
//=========================================================================
enum SeparatorTag { NO_TAG, YES_DEFAULT_TAG, SEPARATOR_PIECE_TAG };
union Separator
switch (SeparatorTag){
    case SEPARATOR_PIECE_TAG:
        PerceptibleReference
            separator_piece;
};

// Interface: RtCompositeOrStructuralSocket Operation: setMenuInteractionStyle
// Corresponding MHEG datatype: Association
//=========================================================================
struct Association {
    sequence<SocketReference,1>
        title;
    sequence<Separator,1>
        separator;
    sequence<SocketReference,1>
        submenu;
    sequence<PresentationPersistence,1>
        submenu_presentation_persistence;
    sequence<Orientation,1>
        submenu_orientation;
};

// Interface: RtSocket Operation: plug
// Corresponding MHEG datatype: Plug-In
//=========================================================================
enum PlugInTag { PLUG_IN_RT_COMPONENT_REFERENCE_TAG,
PLUG_IN_COMPONENT_REFERENCE_TAG, PLUG_IN_LABEL_TAG };
union PlugIn
switch (PlugInTag){
    case PLUG_IN_RT_COMPONENT_REFERENCE_TAG:
        RtObjectReference
            rt_component_reference;
    case PLUG_IN_COMPONENT_REFERENCE_TAG:
        MhObjectReference
            component_reference;
    case PLUG_IN_LABEL_TAG:
        GenericString
            label;
};
```

```
// Interface: RtComponentOrSocket Operation: getVSEAPPosition
// Corresponding MHEG datatype: none
//=========================================================================
enum ReferencePointTag { VSEAP_POSITION_ORIGIN_RGS_TAG,
VSEAP_POSITION_ORIGIN_CGS_TAG, VSEAP_POSITION_SAME_RGS_COMPONENT_TAG,
VSEAP_POSITION_SAME_CGS_COMPONENT_TAG, VSEAP_POSITION_SPECIFIED_POSITION_TAG };
union ReferencePoint
switch (ReferencePointTag){
    case VSEAP_POSITION_SAME_RGS_COMPONENT_TAG:
        RtObjectSocketReference
            same_RGS_component;
    case VSEAP_POSITION_SAME_CGS_COMPONENT_TAG:
        RtObjectSocketReference
            same_CGS_component;
    case VSEAP_POSITION_SPECIFIED_POSITION_TAG:
        SpecifiedPosition
            specified_position;
};

// Interface: RtScript Operation: setParameters
// Corresponding MHEG datatype: Parameter
//=========================================================================
enum ParameterTag { GENERIC_VALUE_TAG, MH_OBJECT_REFERENCE_TAG };
union Parameter
switch (ParameterTag){
    case GENERIC_VALUE_TAG:
        GenericValue
            generic_value;
    case MH_OBJECT_REFERENCE_TAG:
        MhObjectReference
            mh_object_reference;
};

// Interface: RtObjectOrSocket Operation: setGlobalBehaviour
// Corresponding MHEG datatype: Global-Behaviour
//=========================================================================
enum GlobalBehaviourTag { GLOBAL_BEHAVIOUR_RT_REFERENCE_TAG,
GLOBAL_BEHAVIOUR_GENERIC_LIST_TAG, GLOBAL_BEHAVIOUR_UNSPECIFIED_TAG };
union GlobalBehaviour
switch (GlobalBehaviourTag){
    case GLOBAL_BEHAVIOUR_RT_REFERENCE_TAG:
        RtReference
            rt_reference;
    case GLOBAL_BEHAVIOUR_GENERIC_LIST_TAG:
        GenericValue
            generic_list;
};

// Interface: RtComponentOrSocket Operation: setVisibleSizesAdjustment
// Corresponding MHEG datatype: Adjustment-PolicyADJUSTMENT-POLICY-
//=========================================================================
enum AdjustmentPolicyTag { ADJUSTMENT_POLICY_COMPONENT_REFERENCE_TAG,
ADJUSTMENT_POLICY_SPECIFIED_TAG, ADJUSTMENT_POLICY_GREATEST_TAG,
ADJUSTMENT_POLICY_SMALLEST_TAG };
union AdjustmentPolicy
switch (AdjustmentPolicyTag){
    case ADJUSTMENT_POLICY_COMPONENT_REFERENCE_TAG:
        RtObjectSocketReference
            component_reference;
    case ADJUSTMENT_POLICY_SPECIFIED_TAG:
        SpecifiedSizes
            specified;
};
```

```
// Interface: RtObject Operation: bind
// Interface: RtObject Operation: new
// Interface: RtObject Operation: getIdentifier
// Corresponding MHEG datatype: none
//=========================================================================
struct RtObjectIdentifier {
    MHEGIdentifier
        model_object_id;
    long
        rt_object_number;
};

// Interface: RtGenericContentOrPresentableSocket Operation:
//   setButtonInteractionStyle
// Corresponding MHEG datatype: Alternate-Presentation-State
//=========================================================================
struct AlternatePresentation {
    PresentationState
        presentation_state;
    PerceptibleReference
        perceptible_target;
};
```

7.1.37 Exceptions

7.1.37.1 Exception `InvalidTarget`

Description:

L'exception `InvalidTarget` est activée lorsque l'entité MHEG n'est pas disponible. Le membre `period` identifie la période de la cible.

7.1.37.2 Exception `InvalidParameter`

Description:

L'exception `InvalidParameter` est activée lorsque la valeur de l'un des paramètres interdit l'exécution normale de l'action. Le membre `completion_status` indique si l'action a été réalisée (avec une valeur par défaut attribuée au paramètre inadéquat) ou non. Le membre `parameter_number` donne le rang du paramètre non valide.

7.1.37.3 Exception `NotBound`

Description:

L'exception `NotBound` est activée lorsque l'instance d'objet d'interface n'est pas liée à une entité MHEG.

7.1.37.4 Exception `AlreadyBound`

Description:

L'exception `AlreadyBound` est activée lorsque l'instance d'objet d'interface est déjà liée à une entité MHEG. Le membre `entity_identifier` identifie l'entité de limite.

7.1.37.5 Définition IDL

```
exception InvalidTarget {
    unsigned short period;
};

enum CompletionStatus { YES, NO};

exception InvalidParameter {
    CompletionStatus completion_status;
    unsigned short period;
};

typedef long EntityIdentifier
exception AlreadyBound {
    EntityIdentifier entity_identifier;
};

exception NotBound {};
```

7.2 Primitives optionnelles

Les objets suivants seront utiliés pour modifier les objets MHEG de forme b:

- mhAction
- mhComposite
- mhContainer
- mhContent
- mhDescriptor
- mhLink
- mhMultiplexedContent
- mhScript

Annexe A

Définition IDL complète de l'interface de programmation d'application MHEG

```
//=========================================================================
typedef sequence<long> ApplicationIdentifier;

// Corresponding MHEG datatype: Object-Number
//=========================================================================
typedef long ObjectNumber;

// Interface: MhObject Operation: bind
// Interface: MhObject Operation: prepare
// Interface: MhObject Operation: getIdentifier
// Corresponding MHEG datatype: MHEG-Identifier
//=========================================================================
struct MHEGIdentifier {
    sequence<ApplicationIdentifier,1>
        application_identifier;
    ObjectNumber
        object_number;
};

// Corresponding MHEG datatype: Public-Identifier
//=========================================================================
typedef string PublicIdentifier;

// Corresponding MHEG datatype: System-Identifier
//=========================================================================
typedef string SystemIdentifier;

// Corresponding MHEG datatype: External-Long-Identifier
//=========================================================================
struct ExternalLongIdentifier {
    PublicIdentifier
        public_identifier;
    SystemIdentifier
        system_identifier;
};

// Corresponding MHEG datatype: Alias
//=========================================================================
typedef string Alias;

// Corresponding MHEG datatype: Container-Child-Reference
//=========================================================================
enum ContainerChildReference {
    CHILD,
    DESCENDANT
};
```

```
// Interface: MhObject Operation: getPreparationStatus
// Corresponding MHEG datatype: Preparation-Status-Value
//=========================================================================
enum PreparationStatusValue {
    READY,
    NOT_READY,
    PROCESSING
};

// Interface: MhMultiplexedContent Operation: setMultiplex
// Interface: MhMultiplexedContent Operation: setDemultiplex
// Interface: RtMultiplexedContentOrPresentableSocket Operation: setStreamChoice
// Corresponding MHEG datatype: Stream-Identifier
//=========================================================================
typedef sequence<long> StreamIdentifier;

// Corresponding MHEG datatype: Rt-Dynamic-Reference
//=========================================================================
enum RtDynamicReference {
    QUESTION_MARK,
    STAR
};

// Interface: RtObject Operation: getAvailabilityStatus
// Corresponding MHEG datatype: Rt-Availibility-Status-Value
//=========================================================================
enum RtAvailabilityStatusValue {
    RT_AVAILABILITY_STATUS_VALUE_AVAILABLE,
    RT_AVAILABILITY_STATUS_VALUE_NOT_AVAILABLE,
    RT_AVAILABILITY_STATUS_VALUE_PROCESSING
};

// Interface: RtObject Operation: getRunningStatus
// Corresponding MHEG datatype: Running-Status-Value
//=========================================================================
enum RunningStatusValue {
    RUNNING_STATUS_VALUE_RUNNING,
    RUNNING_STATUS_VALUE_NOT_RUNNING,
    RUNNING_STATUS_VALUE_PROCESSING
};

// Interface: RtScript Operation: getTerminationStatus
// Corresponding MHEG datatype: Termination-Status-Value
//=========================================================================
enum TerminationStatusValue {
    TERMINATED,
    NOT_TERMINATED
};

// Interface: RtComponentOrSocket Operation: setRGS
// Interface: Channel Operation: getIdentifier
// Corresponding MHEG datatype: Channel-Identifier
//=========================================================================
typedef long ChannelIdentifier;

// Corresponding MHEG datatype: Priority-Level
//=========================================================================
enum PriorityLevel {
    INCREMENT_PRIORITY,
    DECREMENT_PRIORITY
};

// Interface: RtComponentOrSocket Operation: setVisibleDuration
// Corresponding MHEG datatype: Temporal-Position
//=========================================================================
enum TemporalPositionTag { SPECIFIED_TEMPORAL_POINT_TAG,
LOGICAL_TEMPORAL_PD_POINT_TAG };
union TemporalPosition
switch (TemporalPositionTag){
    case SPECIFIED_TEMPORAL_POINT_TAG:
        long
            specified_temporal_point;
```

```
    case LOGICAL_TEMPORAL_PD_POINT_TAG:
        long
            logical_temporal_PD_point;
};

// Interface: RtComponentOrSocket Operation: setCurrentTemporalPosition
// Corresponding MHEG datatype: Current-Temporal-Position
//=========================================================================
enum CurrentTemporalPositionTag {
CURRENT_TEMPORAL_POSITION_SPECIFIED_TEMPORAL_POINT_TAG,
CURRENT_TEMPORAL_POSITION_LOGICAL_TEMPORAL_VD_POINT_TAG };
union CurrentTemporalPosition
switch (CurrentTemporalPositionTag){
    case CURRENT_TEMPORAL_POSITION_SPECIFIED_TEMPORAL_POINT_TAG:
        long
            specified_temporal_point;
    case CURRENT_TEMPORAL_POSITION_LOGICAL_TEMPORAL_VD_POINT_TAG:
        long
            logical_temporal_vd_point;
};

// Interface: RtComponentOrSocket Operation: setTemporalTermination
// Interface: RtComponentOrSocket Operation: getTemporalTermination
// Corresponding MHEG datatype: Temporal-Termination
//=========================================================================
enum TemporalTermination {
    TEMPORAL_TERMINATION_FREEZE,
    TEMPORAL_TERMINATION_STOP
};

// Interface: RtComponentOrSocket Operation: setSpeed
// Corresponding MHEG datatype: Speed
//=========================================================================
enum SpeedTag { SPECIFIED_OGTR_TAG, SPEED_RATE_TAG, SCALING_FACTOR_TAG };
union Speed
switch (SpeedTag){
    case SPECIFIED_OGTR_TAG:
        long
            specified_OGTR;
    case SPEED_RATE_TAG:
        long
            speed_rate;
    case SCALING_FACTOR_TAG:
        long
            scaling_factor;
};

// Corresponding MHEG datatype: Timestone-Position
//=========================================================================
enum TimestonePositionTag { TIMESTONE_POSITION_SPECIFIED_TEMPORAL_POINT_TAG,
TIMESTONE_POSITION_LOGICAL_TEMPORAL_PD_POINT_TAG,
TIMESTONE_POSITION_LOGICAL_TEMPORAL_VD_POINT_TAG };
union TimestonePosition
switch (TimestonePositionTag){
    case TIMESTONE_POSITION_SPECIFIED_TEMPORAL_POINT_TAG:
        long
            specified_temporal_point;
    case TIMESTONE_POSITION_LOGICAL_TEMPORAL_PD_POINT_TAG:
        long
            logical_temporal_PD_point;
    case TIMESTONE_POSITION_LOGICAL_TEMPORAL_VD_POINT_TAG:
        long
            logical_temporal_VD_point;
};


// Interface: RtComponentOrSocket Operation: getVDLength
// Corresponding MHEG datatype: GT-Indicator
//=========================================================================
enum GTIndicator {
      OGTU,
      RGTU
};
```

```
// Corresponding MHEG datatype: Perceptible-Projection
//=========================================================================
enum PerceptibleProjectionTag { SPECIFIED_SIZE_TAG, IOGSR_SCALING_FACTOR_TAG,
COGSR_SCALING_FACTOR_TAG };
union PerceptibleProjection
switch (PerceptibleProjectionTag){
    case SPECIFIED_SIZE_TAG:
        long
            specified_size;
    case IOGSR_SCALING_FACTOR_TAG:
        long
            iogsr_scaling_factor;
    case COGSR_SCALING_FACTOR_TAG:
        long
            cogsr_scaling_factor;
};

// Interface: RtComponentOrSocket Operation: setAspectRatioPreserved
// Interface: RtComponentOrSocket Operation: getAspectRatio
// Corresponding MHEG datatype: Aspect-Ratio
//=========================================================================
enum AspectRatio {
    PRESERVED,
    NOT_PRESERVED
};

// Interface: RtComponentOrSocket Operation: setVisibleSize
// Interface: RtComponentOrSocket Operation: getVSGS
// Corresponding MHEG datatype: VSGS
//=========================================================================
enum VSGS {
    THIS,
    RELATIVE
};

// Corresponding MHEG datatype: Size-Attribute
//=========================================================================
enum SizeAttributeTag { SIZE_ATTRIBUTE_SPECIFIED_SIZE_TAG,
SIZE_ATTRIBUTE_IVS_RELATIVE_TAG,
SIZE_ATTRIBUTE_CVS_RELATIVE_TAG };
union SizeAttribute
switch (SizeAttributeTag){
    case SIZE_ATTRIBUTE_SPECIFIED_SIZE_TAG:
        long
            specified_size;
    case SIZE_ATTRIBUTE_IVS_RELATIVE_TAG:
        long
            ivs_relative;
    case SIZE_ATTRIBUTE_CVS_RELATIVE_TAG:
        long
            cvs_relative;
};

// Interface: RtComponentOrSocket Operation: setVisibleSizesAdjustment
// Corresponding MHEG datatype: Adjustment-Axis
//=========================================================================
enum AdjustmentAxis {
    X_AXIS,
    Y_AXIS,
    Z_AXIS
};

// Corresponding MHEG datatype: Sub-Socket-Reference
//=========================================================================
enum SubSocketReference {
    SUB_SOCKET_REFERENCE_CHILD,
    SUB_SOCKET_REFERENCE_DESCENDANT,
    SUB_SOCKET_REFERENCE_QUESTION_MARK_CHILD,
    SUB_SOCKET_REFERENCE_QUESTION_MARK_DESCENDANT
};
```

```
// Interface: RtComponentOrSocket Operation: setBox
// Interface: RtComponentOrSocket Operation: getBox
// Corresponding MHEG datatype: Box-Constants
//=========================================================================
enum BoxConstants {
    PRESENTED,
    NOT_PRESENTED
};

// Interface: RtComponentOrSocket Operation: setAttachmentPointPosition
// Corresponding MHEG datatype: Reference-Type
//=========================================================================
enum ReferenceType {
    VSIAP,
    VSEAP
};

// Interface: RtComponentOrSocket Operation: setAttachmentPoint
// Interface: RtComponentOrSocket Operation: setAttachmentPointPosition
// Corresponding MHEG datatype: Attachment-Point-Type
//=========================================================================
enum AttachmentPointType {
    ATTACHMENT_POINT_TYPE_PSAP,
    ATTACHMENT_POINT_TYPE_VSIAP,
    ATTACHMENT_POINT_TYPE_VSEAP
};

// Interface: RtComponentOrSocket Operation: setVisibleSizesAlignment
// Corresponding MHEG datatype: Size-Border
//=========================================================================
enum SizeBorder {
    TOP,
    BOTTOM,
    RIGHT,
    LEFT,
    UPPER_Z,
    LOWER_Z,
    CENTER_X,
    CENTER_Y,
    CENTER_Z
};

// Interface: RtComponentOrSocket Operation: setMovingAbility
// Interface: RtComponentOrSocket Operation: setResizingAbility
// Interface: RtComponentOrSocket Operation: setScalingAbility
// Interface: RtComponentOrSocket Operation: setScrollingAbility
// Interface: RtComponentOrSocket Operation: getMovingAbility
// Interface: RtComponentOrSocket Operation: getResizingAbility
// Interface: RtComponentOrSocket Operation: getScalingAbility
// Interface: RtComponentOrSocket Operation: getScrollingAbility
// Corresponding MHEG datatype: User-Controls
//=========================================================================
enum UserControls {
    ALLOWED,
    NOT_ALLOWED
};

// Interface: RtComponentOrSocket Operation: getPS
// Corresponding MHEG datatype: GS-Indicator
//=========================================================================
enum GSIndicator {
    OGSU,
    RGSU
};

// Interface: RtComponentOrSocket Operation: getPSAP
// Interface: RtComponentOrSocket Operation: getVSIAP
// Corresponding MHEG datatype: Point-Type
//=========================================================================
enum PointType {
    RELATIVE_POINT,
    ABSOLUTE_POINT
};
```

```
// Interface: RtComponentOrSocket Operation: setSelectionStatus
// Interface: RtComponentOrSocket Operation: getSelectionStatus
// Corresponding MHEG datatype: Selection-Status-Value
//=========================================================================
enum SelectionStatusValue {
    SELECTED,
    NOT_SELECTED
};

// Interface: RtComponentOrSocket Operation:
//   setSelectionPresentationEffectResponsibility
// Interface: RtComponentOrSocket Operation:
//   getSelectionPresentationEffectResponsibility
// Interface: RtComponentOrSocket Operation:
//   setModificationPresentationEffectResponsibility
// Interface: RtComponentOrSocket Operation:
//   getModificationPresentationEffectResponsibility
// Corresponding MHEG datatype: Responsibility
//=========================================================================
enum Responsibility {
    MHEG_ENGINE,
    AUTHOR
};

// Interface: RtComponentOrSocket Operation: getEffectiveSelectability
// Corresponding MHEG datatype: Effective-Selectability
//=========================================================================
enum EffectiveSelectability {
    EFFECTIVELY_SELECTABLE,
    EFFECTIVELY_NOT_SELECTABLE
};

// Interface: RtComponentOrSocket Operation: setModificationStatus
// Interface: RtComponentOrSocket Operation: getModificationStatus
// Corresponding MHEG datatype: Modification-Status-Value
//=========================================================================
enum ModificationStatusValue {
    MODIFIED,
    MODIFYING,
    NOT_MODIFIED
};

// Interface: RtComponentOrSocket Operation: getEffectiveModifiability
// Corresponding MHEG datatype: Effective-Modifiability
//=========================================================================
enum EffectiveModifiability {
    EFFECTIVELY_MODIFIABLE,
    EFFECTIVELY_NOT_MODIFIABLE
};

// Interface: RtCompositeOrStructuralSocket Operation: setResizingStrategy
// Interface: RtCompositeOrStructuralSocket Operation: getResizingStrategy
// Corresponding MHEG datatype: Resizing-Strategy
//=========================================================================
enum ResizingStrategy {
    FIXED,
    MINIMUM,
    GROWS_ONLY
};

// Interface: RtCompositeOrStructuralSocket Operation: setMenuInteractionStyle
// Interface: RtCompositeOrStructuralSocket Operation:
//   setScrollingListInteractionStyle
// Interface: RtContentOrPresentableSocket Operation: setSliderInteractionStyle
// Corresponding MHEG datatype: Orientation
//=========================================================================
enum Orientation {
    HORIZONTAL,
    VERTICAL
};
```

```
// Corresponding MHEG datatype: Presentation-Persistence
//=========================================================================
enum PresentationPersistence {
    PERSISTENT,
    NOT_PERSISTENT
};

// Interface: RtCompositeOrStructuralSocket Operation:
//   setScrollingListInteractionStyle
// Corresponding MHEG datatype: Slider-Side
//=========================================================================
enum SliderSide {
    SIDE1,
    SIDE2
};

// Interface: RtGenericContentOrPresentableSocket Operation: setAudibleVolume
// Corresponding MHEG datatype: Audible-Volume
//=========================================================================
enum AudibleVolumeTag { SPECIFIED_VOLUME_TAG, LOGICAL_VOLUME_TAG,
IOV_SCALING_FACTOR_TAG,
OV_SCALING_FACTOR_TAG };
union AudibleVolume
switch (AudibleVolumeTag){
    case SPECIFIED_VOLUME_TAG:
        long
            specified_volume;
    case LOGICAL_VOLUME_TAG:
        long
            logical_volume;
    case IOV_SCALING_FACTOR_TAG:
        long
            iov_scaling_factor;
    case OV_SCALING_FACTOR_TAG:
        long
            ov_scaling_factor;
};

// Interface: RtGenericContentOrPresentableSocket Operation:
//   setButtonInteractionStyle
// Corresponding MHEG datatype: Presentation-State
//=========================================================================
enum PresentationState {
    SELECTABLE_NOT_SELECTED,
    SELECTABLE_SELECTED,
    NOT_SELECTABLE_SELECTED,
    NOT_SELECTABLE_NOT_SELECTED
};

// Corresponding MHEG datatype: Echo-Mode
//=========================================================================
enum EchoMode {
    ITSELF,
    HIDDEN
};

// Interface: RtContentOrPresentableSocket Operation:
//   setEntryFieldInteractionStyle
// Corresponding MHEG datatype: Echo-Style
//=========================================================================
enum EchoStyleTag { MODE_TAG, SPECIFIED_TAG };
union EchoStyle
switch (EchoStyleTag){
    case MODE_TAG:
        EchoMode
            mode;
    case SPECIFIED_TAG:
        string
            specified;
};
```

```
// Corresponding MHEG datatype: Channel-Reference
//=====================================================================
enum ChannelReferenceTag { CHANNEL_IDENTIFIER_TAG, ALIAS_TAG,
NULL_CHANNEL_REFERENCE_TAG };
union ChannelReference
switch (ChannelReferenceTag){
    case CHANNEL_IDENTIFIER_TAG:
        ChannelIdentifier
            channel_identifier;
    case ALIAS_TAG:
        Alias
            alias;
};

// Corresponding MHEG datatype: Interval
//=====================================================================
struct Interval {
    sequence<long,1>
        start_point;
    sequence<long,1>
        end_point;
};

// Corresponding MHEG datatype: Generic-Volume-Range
//=====================================================================
struct GenericVolumeRange {
    sequence<long,1>
        maximum_volume;
    sequence<long,1>
        minimum_volume;
};

// Interface: Channel Operation: new
// Corresponding MHEG datatype: Original-Def-Declaration
//=====================================================================
struct OriginalDefDeclaration {
    sequence<long,1>
        generic_temporal_ratio;
    sequence<Interval,1>
        x_axis_interval;
    sequence<Interval,1>
        y_axis_interval;
    sequence<Interval,1>
        z_axis_interval;
    sequence<GenericVolumeRange,1>
        audible_volume_range_declaration;
};

// Interface: Channel Operation: getAvailability
// Corresponding MHEG datatype: Channel-Status-ValueCHANNEL-STATUS-VALUE-
//=====================================================================
enum ChannelStatusValue {
    CHANNEL_STATUS_VALUE_AVAILABLE,
    CHANNEL_STATUS_VALUE_NOT_AVAILABLE,
    CHANNEL_STATUS_VALUE_PROCESSING
};

// Interface: Channel Operation: setPerceptability
// Interface: Channel Operation: getPerceptability
// Corresponding MHEG datatype: Channel-Perceptability-Values
//=====================================================================
enum ChannelPerceptabilityValue {
    ON,
    OFF
};

// Interface: NotificationManager Operation: getNotification
// Interface: MhContent Operation: getData
// Corresponding MHEG datatype: Generic-Value
//=====================================================================
enum GenericValueTag { BOOLEAN_FIELD_TAG, NUMERIC_TAG, STRING_FIELD_TAG,
GENERIC_LIST_TAG,
UNSPECIFIED_TAG };
union GenericValue
```

```
switch (GenericValueTag){
    case BOOLEAN_FIELD_TAG:
        boolean
            boolean_field;
    case NUMERIC_TAG:
        long
            numeric;
    case STRING_FIELD_TAG:
        string
            string_field;
    case GENERIC_LIST_TAG:
        sequence<GenericValue>
            generic_list;
};

// Corresponding MHEG datatype: Generic-String
//=========================================================================
enum GenericStringTag { GENERIC_STRING_CONSTANT_TAG,
GENERIC_STRING_UNSPECIFIED_TAG };
union GenericString
switch (GenericStringTag){
    case GENERIC_STRING_CONSTANT_TAG:
        string
            constant;
};

// Interface: Socket Operation: setVisibleDurationPosition
// Corresponding MHEG datatype: Visible-Duration
//=========================================================================
enum VisibleDurationPositionTag {
VISIBLE_DURATION_POSITION_SPECIFIED_TEMPORAL_POINT_TAG,
VISIBLE_DURATION_POSITION_LOGICAL_TEMPORAL_PD_POINT_TAG,
VISIBLE_DURATION_POSITION_LOGICAL_TEMPORAL_VD_POINT_TAG };
union VisibleDurationPosition
switch (VisibleDurationPositionTag){
    case VISIBLE_DURATION_POSITION_SPECIFIED_TEMPORAL_POINT_TAG:
        long
            specified_temporal_point;
    case VISIBLE_DURATION_POSITION_LOGICAL_TEMPORAL_PD_POINT_TAG:
        long
            logical_temporal_PD_point;
    case VISIBLE_DURATION_POSITION_LOGICAL_TEMPORAL_VD_POINT_TAG:
        long
            logical_temporal_VD_point;
};

// Interface: RtComponentOrSocket Operation: getRGS
// Corresponding MHEG datatype: none
//=========================================================================
enum RGSValueTag { RGS_VALUE_CHANNEL_IDENTIFIER_TAG, RGS_VALUE_NULL_CHANNEL_TAG,
RGS_VALUE_PRGS_TAG };
union RGSValue
switch (RGSValueTag){
    case RGS_VALUE_CHANNEL_IDENTIFIER_TAG:
        ChannelIdentifier
            channel_identifier;
};

// Corresponding MHEG datatype: Generic-Numeric
//=========================================================================
enum GenericNumericTag { GENERIC_NUMERIC_CONSTANT_TAG,
GENERIC_NUMERIC_UNSPECIFIED_TAG };
union GenericNumeric
switch (GenericNumericTag){
    case GENERIC_NUMERIC_CONSTANT_TAG:
        long
            constant;
};
```

```
// Interface: RtComponentOrSocket Operation: getSelectionMode
// Corresponding MHEG datatype: none
//=========================================================================
enum SelectionModeValueTag { USER_INTERACTION_TAG, NO_SELECTION_TAG,
MHEG_ACTION_TAG,
USING_APPLICATION_ACTION_TAG };
union SelectionModeValue
switch (SelectionModeValueTag){
    case USER_INTERACTION_TAG:
        unsigned long
            user_interaction;
};

// Interface: RtComponentOrSocket Operation: getModificationMode
// Corresponding MHEG datatype: none
//=========================================================================
enum ModificationModeValueTag { MODIFICATION_MODE_VALUE_USER_INTERACTION_TAG,
MODIFICATION_MODE_VALUE_NO_MODIFICATION_TAG,
MODIFICATION_MODE_VALUE_MHEG_ACTION_TAG,
MODIFICATION_MODE_VALUE_USING_APPLICATION_ACTION_TAG,
MODIFICATION_MODE_VALUE_CHILD_TAG };
union ModificationModeValue
switch (ModificationModeValueTag){
    case MODIFICATION_MODE_VALUE_USER_INTERACTION_TAG:
        unsigned long
            user_interaction;
};

// Corresponding MHEG datatype: External-Identifier
//=========================================================================
enum ExternalIdentifierTag { EXTERNAL_LONG_ID_TAG, PUBLIC_ID_TAG, SYSTEM_ID_TAG
};
union ExternalIdentifier
switch (ExternalIdentifierTag){
    case EXTERNAL_LONG_ID_TAG:
        ExternalLongIdentifier
            external_long_id;
    case PUBLIC_ID_TAG:
        PublicIdentifier
            public_id;
    case SYSTEM_ID_TAG:
        SystemIdentifier
            system_id;
};

// Corresponding MHEG datatype: Container-Tail
//=========================================================================
struct ContainerTail {
    sequence<long>
        indexes;
    enum ContainerTailTag { INDEX_TAG, CONTAINER_CHILD_REF_TAG } tag;
    union ContainerTail
    switch (ContainerTailTag){
        case INDEX_TAG:
            long
                index;
        case CONTAINER_CHILD_REF_TAG:
            ContainerChildReference
                container_child_ref;
      } end;
};

// Corresponding MHEG datatype: Specified-Sizes
//=========================================================================
struct SpecifiedSizes {
    sequence<GenericNumeric,1>
        x_axis_length;
    sequence<GenericNumeric,1>
        y_axis_length;
    sequence<GenericNumeric,1>
        z_axis_length;
};
```

```
// Corresponding MHEG datatype: Attachment-Attribute
//=========================================================================
enum AttachmentAttributeTag { SPECIFIED_POSITION_TAG, LOGICAL_POSITION_TAG };
union AttachmentAttribute
switch (AttachmentAttributeTag){
    case SPECIFIED_POSITION_TAG:
        GenericNumeric
            specified_position;
    case LOGICAL_POSITION_TAG:
        GenericNumeric
            logical_position;
};

// Corresponding MHEG datatype: Length-Attribute
//=========================================================================
enum LengthAttributeTag { SPECIFIED_LENGTH_TAG, RELATIVE_LENGTH_TAG };
union LengthAttribute
switch (LengthAttributeTag){
    case SPECIFIED_LENGTH_TAG:
        GenericNumeric
            specified_length;
    case RELATIVE_LENGTH_TAG:
        GenericNumeric
            relative_length;
};

// Interface: RtComponentOrSocket Operation: getPS
// Interface: RtComponentOrSocket Operation: getPSAP
// Interface: RtComponentOrSocket Operation: getVS
// Interface: RtComponentOrSocket Operation: getVSIAP
// Interface: RtComponentOrSocket Operation: getVSIAPPosition
// Interface: RtComponentOrSocket Operation: getVSEAP
// Interface: RtComponentOrSocket Operation: getVSEAPPosition
// Corresponding MHEG datatype: Specified-Position
//=========================================================================
struct SpecifiedPosition {
    GenericNumeric
        x_point;
    GenericNumeric
        y_point;
    GenericNumeric
        z_point;
};

// Interface: RtComponentOrSocket Operation: setPresentationPriority
// Corresponding MHEG datatype: Presentation-Priority
//=========================================================================
enum PresentationPriorityTag { GENERIC_NUMERIC_TAG, PRIORITY_LEVEL_TAG };
union PresentationPriority
switch (PresentationPriorityTag){
    case GENERIC_NUMERIC_TAG:
        GenericNumeric
            generic_numeric;
    case PRIORITY_LEVEL_TAG:
        PriorityLevel
            priority_level;
};

// Interface: RtComponentOrSocket Operation: setTimestones
// Corresponding MHEG datatype: Timestone
//=========================================================================
struct Timestone {
    long
        timestone_identifier;
    TimestonePosition
        timestone_position;
};
```

```
// Interface: RtComponentOrSocket Operation: setVisibleSize
// Corresponding MHEG datatype: none
//=========================================================================
enum VSTag { X_SIZE_ATTRIBUTE_TAG, Y_SIZE_ATTRIBUTE_TAG, Z_SIZE_ATTRIBUTE_TAG };
union VS
switch (VSTag){
    case X_SIZE_ATTRIBUTE_TAG:
        SizeAttribute
            x_size_attribute;
    case Y_SIZE_ATTRIBUTE_TAG:
        SizeAttribute
            y_size_attribute;
    case Z_SIZE_ATTRIBUTE_TAG:
        SizeAttribute
            z_size_attribute;
};

// Interface: RtComponentOrSocket Operation: setAttachmentPoint
// Corresponding MHEG datatype: none
//=========================================================================
struct AttachmentPoint {
    sequence<AttachmentAttribute,1>
        x_attachment;
    sequence<AttachmentAttribute,1>
        y_attachment;
    sequence<AttachmentAttribute,1>
        z_attachment;
};

// Interface: RtComponentOrSocket Operation: setAttachmentPointPosition
// Corresponding MHEG datatype: Lengths
//=========================================================================
struct Lengths {
    sequence<LengthAttribute,1>
        x_length;
    sequence<LengthAttribute,1>
        y_length;
    sequence<LengthAttribute,1>
        z_length;
};

// Interface: RtMultiplexedContentOrPresentableSocket Operation: getStreamChosen
// Corresponding MHEG datatype: none
//=========================================================================
enum StreamValueTag { STREAM_IDENTIFIER_TAG, NO_STREAM_CHOSEN_TAG };
union StreamValue
switch (StreamValueTag){
    case STREAM_IDENTIFIER_TAG:
        StreamIdentifier
            stream_identifier;
};

// Interface: MhContent Operation: setData
// Corresponding MHEG datatype: Data-Element
//=========================================================================
struct DataElement {
      sequence<long>
         element_list_index;
      GenericValue
         generic_value;
};

// Interface: NotificationManager Operation: getNotification
// Interface: MhObject Operation: bind
// Interface: MhObject Operation: prepare
// Interface: MhGenericContent Operation: copy
// Corresponding MHEG datatype: Mh-Object-Reference
//=========================================================================
struct MhObjectReference {
enum MhObjectReferenceHeadTag { MHEG_IDENTIFIER_TAG, EXTERNAL_IDENTIFIER_TAG,
ALIAS_TAG, NULL_OBJECT_REF_TAG } head_tag;
union MhObjectReferenceHead
```

```
switch (MhObjectReferenceHeadTag){
    case MHEG_IDENTIFIER_TAG:
        MHEGIdentifier
            mheg_identifier;
    case EXTERNAL_IDENTIFIER_TAG:
        ExternalIdentifier
            external_identifier;
    case ALIAS_TAG:
        Alias
            alias;
} head;
enum MhObjectReferenceTailTag { CONTAINER_ELEMENT_REFERENCE_TAG,
OTHER_REFERENCE_TAG
} tail_tag;
union MhObjectReferenceTail
switch (MhObjectReferenceTailTag){
    case CONTAINER_ELEMENT_REFERENCE_TAG:
        ContainerTail
            container_tail;
} tail;
};
// Interface: RtComponentOrSocket Operation: setPerceptibleSizeProjection
// Corresponding MHEG datatype: Perceptible-Size-Projection
//=========================================================================
struct PerceptibleSizeProjection {
    sequence<PerceptibleProjection,1>
        x_perceptible_size_projection;
    sequence<PerceptibleProjection,1>
        y_perceptible_size_projection;
    sequence<PerceptibleProjection,1>
        z_perceptible_size_projection;
};
// Corresponding MHEG datatype: Rt-Object-Number-Reference
//=========================================================================
enum RtObjectNumberReferenceTag { RT_OBJECT_NUMBER_TAG, RT_DYNAMIC_REFERENCE_TAG
};
union RtObjectNumberReference
switch (RtObjectNumberReferenceTag){
    case RT_OBJECT_NUMBER_TAG:
        long
            rt_object_number;
    case RT_DYNAMIC_REFERENCE_TAG:
        RtDynamicReference
            rt_dynamic_reference;
};
// Interface: RtObject Operation: bind
// Interface: RtObject Operation: new
// Corresponding MHEG datatype: Rt-Object-Reference
//=========================================================================
struct RtObjectReference {
    MhObjectReference
        model_object_reference;
    RtObjectNumberReference
        rt_object_number_reference;
};
// Corresponding MHEG datatype: Rt-Reference
//=========================================================================
enum RtReferenceTag { RT_REFERENCE_RT_OBJECT_REFERENCE_TAG,
RT_REFERENCE_ALIAS_TAG,
RT_REFERENCE_NULL_RT_OBJECT_TAG };
union RtReference
switch (RtReferenceTag){
    case RT_REFERENCE_RT_OBJECT_REFERENCE_TAG:
        RtObjectReference
            rt_object_reference;
    case RT_REFERENCE_ALIAS_TAG:
        Alias
            alias;
};
```

```
// Corresponding MHEG datatype: Socket-Tail
//=========================================================================
struct SocketTail {
    sequence<long>
        indexes;
    enum SocketTailTag { INDEX_TAG, SUB_SOCKET_REF_TAG } tag;
    union SocketTail
    switch (SocketTailTag){
        case INDEX_TAG:
            long
                index;
        case SUB_SOCKET_REF_TAG:
            SubSocketReference
                sub_socket_ref;
    } end;
};

// Corresponding MHEG datatype: Indexed-Child-Socket
//=========================================================================
struct IndexedChildSocket {
    long
        index;
    SocketTail
        tail;
};

// Interface: Socket Operation: bind
// Interface: Socket Operation: getIdentification
// Corresponding MHEG datatype: Socket-Identification
//=========================================================================
struct SocketIdentification {
    RtReference
        rt_composite_reference;
    SocketTail
        socket_tail;
};

// Interface: Socket Operation: bind
// Corresponding MHEG datatype: Socket-Reference
//=========================================================================
enum SocketReferenceTag { SOCKET_REFERENCE_SOCKET_IDENT_TAG,
SOCKET_REFERENCE_ALIAS_TAG };
union SocketReference
switch (SocketReferenceTag){
    case SOCKET_REFERENCE_SOCKET_IDENT_TAG:
        SocketIdentification
            socket_ident;
    case SOCKET_REFERENCE_ALIAS_TAG:
        Alias
            alias;
};

// Corresponding MHEG datatype: Rt-Object-Socket-Reference
//=========================================================================
enum RtObjectSocketReferenceTag { RT_REFERENCE_TAG, SOCKET_REFERENCE_TAG };
union RtObjectSocketReference
switch (RtObjectSocketReferenceTag){
    case RT_REFERENCE_TAG:
        RtReference
            rt_reference;
    case SOCKET_REFERENCE_TAG:
        SocketReference
            socket_reference;
};

// Interface: RtCompositeOrStructuralSocket Operation:
//   setScrollingListInteractionStyle
// Interface: RtContentOrPresentableSocket Operation: setSliderInteractionStyle
// Interface: RtContentOrPresentableSocket Operation:
//   setEntryFieldInteractionStyle
// Interface: Channel Operation: getAssignedPerceptibles
// Corresponding MHEG datatype: Perceptible-Reference
```

```
//=========================================================================
enum PerceptibleReferenceTag { RT_COMPONENT_REFERENCE_TAG,
RT_SOCKET_REFERENCE_TAG };
union PerceptibleReference
switch (PerceptibleReferenceTag){
    case RT_COMPONENT_REFERENCE_TAG:
        RtReference
            rt_component_reference;
    case RT_SOCKET_REFERENCE_TAG:
        SocketReference
            rt_socket_reference;
};

// Interface: RtCompositeOrStructuralSocket Operation:
//   setScrollingListInteractionStyle
// Corresponding MHEG datatype: Separator
//=========================================================================
enum SeparatorTag { NO_TAG, YES_DEFAULT_TAG, SEPARATOR_PIECE_TAG };
union Separator
switch (SeparatorTag){
    case SEPARATOR_PIECE_TAG:
        PerceptibleReference
            separator_piece;
};

// Interface: RtCompositeOrStructuralSocket Operation: setMenuInteractionStyle
// Corresponding MHEG datatype: Association
//=========================================================================
struct Association {
    sequence<SocketReference,1>
        title;
    sequence<Separator,1>
        separator;
    sequence<SocketReference,1>
        submenu;
    sequence<PresentationPersistence,1>
        submenu_presentation_persistence;
    sequence<Orientation,1>
        submenu_orientation;
};

// Interface: RtSocket Operation: plug
// Corresponding MHEG datatype: Plug-In
//=========================================================================
enum PlugInTag { PLUG_IN_RT_COMPONENT_REFERENCE_TAG,
PLUG_IN_COMPONENT_REFERENCE_TAG,
PLUG_IN_LABEL_TAG };
union PlugIn
switch (PlugInTag){
    case PLUG_IN_RT_COMPONENT_REFERENCE_TAG:
        RtObjectReference
            rt_component_reference;
    case PLUG_IN_COMPONENT_REFERENCE_TAG:
        MhObjectReference
            component_reference;
    case PLUG_IN_LABEL_TAG:
        GenericString
            label;
};

// Interface: RtComponentOrSocket Operation: getVSEAPPosition
// Corresponding MHEG datatype: none
//=========================================================================
enum ReferencePointTag { VSEAP_POSITION_ORIGIN_RGS_TAG,
VSEAP_POSITION_ORIGIN_CGS_TAG,
VSEAP_POSITION_SAME_RGS_COMPONENT_TAG, VSEAP_POSITION_SAME_CGS_COMPONENT_TAG,
VSEAP_POSITION_SPECIFIED_POSITION_TAG };
union ReferencePoint
switch (ReferencePointTag){
    case VSEAP_POSITION_SAME_RGS_COMPONENT_TAG:
        RtObjectSocketReference
            same_RGS_component;
```

```
    case VSEAP_POSITION_SAME_CGS_COMPONENT_TAG:
        RtObjectSocketReference
            same_CGS_component;
    case VSEAP_POSITION_SPECIFIED_POSITION_TAG:
        SpecifiedPosition
            specified_position;
};

// Interface: RtScript Operation: setParameters
// Corresponding MHEG datatype: Parameter
//=========================================================================
enum ParameterTag { GENERIC_VALUE_TAG, MH_OBJECT_REFERENCE_TAG };
union Parameter
switch (ParameterTag){
    case GENERIC_VALUE_TAG:
        GenericValue
            generic_value;
    case MH_OBJECT_REFERENCE_TAG:
        MhObjectReference
            mh_object_reference;
};

// Interface: RtObjectOrSocket Operation: setGlobalBehaviour
// Corresponding MHEG datatype: Global-Behaviour
//=========================================================================
enum GlobalBehaviourTag { GLOBAL_BEHAVIOUR_RT_REFERENCE_TAG,
GLOBAL_BEHAVIOUR_GENERIC_LIST_TAG,
GLOBAL_BEHAVIOUR_UNSPECIFIED_TAG };
union GlobalBehaviour
switch (GlobalBehaviourTag){
    case GLOBAL_BEHAVIOUR_RT_REFERENCE_TAG:
        RtReference
            rt_reference;
    case GLOBAL_BEHAVIOUR_GENERIC_LIST_TAG:
        GenericValue
            generic_list;
};

// Interface: RtComponentOrSocket Operation: setVisibleSizesAdjustment
// Corresponding MHEG datatype: Adjustment-PolicyADJUSTMENT-POLICY-
//=========================================================================
enum AdjustmentPolicyTag { ADJUSTMENT_POLICY_COMPONENT_REFERENCE_TAG,
ADJUSTMENT_POLICY_SPECIFIED_TAG, ADJUSTMENT_POLICY_GREATEST_TAG,
ADJUSTMENT_POLICY_SMALLEST_TAG
};
union AdjustmentPolicy
switch (AdjustmentPolicyTag){
    case ADJUSTMENT_POLICY_COMPONENT_REFERENCE_TAG:
        RtObjectSocketReference
            component_reference;
    case ADJUSTMENT_POLICY_SPECIFIED_TAG:
        SpecifiedSizes
            specified;
};

// Interface: RtObject Operation: bind
// Interface: RtObject Operation: new
// Interface: RtObject Operation: getIdentifier
// Corresponding MHEG datatype: none
//=========================================================================
struct RtObjectIdentifier {
    MHEGIdentifier
        model_object_id;
    long
        rt_object_number;
};

// Interface: RtGenericContentOrPresentableSocket Operation:
//   setButtonInteractionStyle
// Corresponding MHEG datatype: Alternate-Presentation-State
//=========================================================================
struct AlternatePresentation {
    PresentationState
        presentation_state;
```

```
    PerceptibleReference
        perceptible_target;
};

// Exceptions
//============================================================================
exception InvalidTarget {
    unsigned short period;
};

enum CompletionStatus { YES, NO};

exception InvalidParameter {
    CompletionStatus completion_status;
    unsigned short period;
};

typedef long EntityIdentifier;

exception AlreadyBound {
    EntityIdentifier entity_identifier;
};

exception NotBound {};

interface MHEGEngine {

    void
        initialiseEngine();
    void
        shutdownEngine();

};

interface NotificationManager {

    sequence<unsigned short>
        getReturnability();

    void
        getNotification(
            in unsigned short
                notification_number,
            out sequence<GenericValue>
                values,
            out sequence<MhObjectReference>
                objects)
    raises(InvalidParameter);

};

interface    EntityManager {

    sequence<MHEGIdentifier>
        getAvailableMhObjects();

    sequence<RtObjectIdentifier>
        getAvailableRtObjects();

    sequence<ChannelIdentifier>
        getAvailableChannels();

    void
        releaseAlias(
            in string
                alias)
    raises(InvalidParameter);

};

interface Entity {

    void
        setAlias(
            in string
                alias)
    raises(InvalidTarget);
```

```
    string
        getAlias()
    raises(InvalidTarget);
};

interface MhObject: Entity {

    MHEGIdentifier
        bind(
            in MhObjectReference
                mh_object_reference)
    raises(AlreadyBound, InvalidTarget);

    void
        unbind()
    raises(NotBound);

    MHEGIdentifier
        prepare(
            in MhObjectReference
                mh_object_reference)
    raises(AlreadyBound, InvalidTarget);

    void
        destroy()
    raises(NotBound, InvalidTarget);

    PreparationStatusValue
        getPreparationStatus()
    raises(NotBound, InvalidTarget);

    MHEGIdentifier
        getIdentifier()
    raises(NotBound);

    void
        kill();

};

interface MhAction: MhObject {

    void
        delay(
            in unsigned short
                nested_action_number,
            in unsigned long
                delay)
    raises(InvalidTarget, InvalidParameter);

};

interface MhLink: MhObject {

    void
        abort()
    raises(InvalidTarget);
};

interface MhModel: MhObject {};

interface MhComponent: MhModel {};

interface MhGenericContent: MhComponent {

    void
        copy(
            in sequence<MhObjectReference>
                copies)
    raises(InvalidTarget, InvalidParameter);
};
```

```
interface MhContent: MhGenericContent {

    void
        setData(
            in boolean
                substitution_indicator,
            in sequence<DataElement>
                data_elements)
    raises(InvalidTarget, InvalidParameter);

    GenericValue
        getData(
            in sequence<long>
                element_list_index)
    raises(InvalidTarget, InvalidParameter);

};

interface MhMultiplexedContent: MhGenericContent {

    void
        setMultiplex(
            in sequence<StreamIdentifier>
                stream_list)
    raises(InvalidTarget, InvalidParameter);

    void
        setDemultiplex(
            in sequence<StreamIdentifier>
                stream_list)
    raises(InvalidTarget, InvalidParameter);

};

interface MhComposite: MhComponent {};

interface MhScript: MhModel {};

interface MhContainer: MhObject {};

interface MhDescriptor: MhObject {};

interface RtObjectOrSocket {

    void
        setGlobalBehaviour(
            in GlobalBehaviour
                global_behaviour)
    raises(InvalidTarget, InvalidParameter);

    GenericValue
        getGlobalBehaviour()
    raises(InvalidTarget);

    void
        run()
    raises(InvalidTarget);

    void
        stop()
    raises(InvalidTarget);

};

interface RtObject: Entity {

    RtObjectIdentifier
        bind(
            in RtObjectReference
                rt_object_reference)
    raises(AlreadyBound, InvalidTarget);

    void
        unbind()
    raises(NotBound);
```

```
    RtObjectIdentifier
        new(
            in RtObjectReference
                rt_object_reference)
    raises(AlreadyBound, InvalidTarget);

    void
        delete()
    raises(NotBound, InvalidTarget);

    RtAvailabilityStatusValue
        getAvailabilityStatus()
    raises(NotBound, InvalidTarget);

    RtObjectIdentifier
        getIdentifier()
    raises(NotBound);

    void
        kill();

    RunningStatusValue
        getRunningStatus()
    raises(InvalidTarget);

};

interface Socket: Entity, RtObjectOrSocket {

    SocketIdentification
        bind(
            in SocketReference
                socket_reference)
    raises(AlreadyBound, InvalidTarget);

    void
        unbind()
        raises(NotBound);

    SocketIdentification
        getIdentifier();
    void
        kill();

    void
        plug(
            in PlugIn
                plug_in)
    raises(InvalidTarget);

    void
        setVisibleDurationPosition(
            in VisibleDurationPosition
                visible_duration_position)
    raises(InvalidTarget, InvalidParameter);

    unsigned long
        getVisibleDurationPosition()
    raises(InvalidTarget);

};

interface RtScript: RtObject {

    void
        setParameters(
            in sequence<Parameter>
                parameters)
    raises(InvalidTarget);

    TerminationStatusValue
        getTerminationStatus()
    raises(InvalidTarget);

};

interface RtComponentOrSocket {
```

```
void
    setRGS(
        in ChannelIdentifier
            channel_identifier)
raises(InvalidTarget);

RGSValue
    getRGS()
raises(InvalidTarget);

void
    setOpacity(
        in unsigned short
            opacity_rate,
        in unsigned long
            transition_duration)
raises(InvalidTarget);

void
    setPresentationPriority(
        in PresentationPriority
            presentation_priority,
        in unsigned long
            transition_duration)
raises(InvalidTarget, InvalidParameter);

unsigned short
    getOpacity()
raises(InvalidTarget);

unsigned short
    getEffectiveOpacity()
raises(InvalidTarget);

unsigned short
    getPresentationPriority()
raises(InvalidTarget);

void
    setVisibleDuration(
        in TemporalPosition
            initial_temporal_position,
        in TemporalPosition
            terminal_temporal_position)
raises(InvalidTarget, InvalidParameter);

void
    setTemporalTermination(
        in TemporalTermination
            temporal_termination)
raises(InvalidTarget);

void
    setCurrentTemporalPosition(
        in TemporalPosition
            temporal_position)
raises(InvalidTarget, InvalidParameter);

void
    setSpeed(
        in Speed
            the_speed,
        in unsigned long
            transition_duration)
raises(InvalidTarget, InvalidParameter);

void
    setTimestones(
        in sequence<Timestone>
            timestones)
raises(InvalidTarget, InvalidParameter);

unsigned long
    getInitialTemporalPosition()
raises(InvalidTarget);
```

```
unsigned long
    getTerminalTemporalPosition()
raises(InvalidTarget);

unsigned long
    getVDLength(
        in GTIndicator
            gt_indicator)
raises(InvalidTarget);

TemporalTermination
    getTemporalTermination()
raises(InvalidTarget);

unsigned long
    getCurrentTemporalPosition()
raises(InvalidTarget);

short
    getSpeedRate()
raises(InvalidTarget);

unsigned long
    getOGTR()
raises(InvalidTarget);

short
    getEffectiveSpeedRate()
raises(InvalidTarget);

unsigned long
    getEffectiveOGTR()
raises(InvalidTarget);

unsigned short
    getTimestoneStatus()
raises(InvalidTarget);

void
    setPerceptibleSizeProjection(
        in PerceptibleSizeProjection
            perceptible_size_projection,
        in unsigned long
            transition_duration)
raises(InvalidTarget, InvalidParameter);

void
    setAspectRatio(
        in AspectRatio
            preserved)
raises(InvalidTarget);

void
    setVisibleSize(
        in VSGS
            the_vsgs,
        in VS
            the_vs,
        in unsigned long
            transition_duration)
raises(InvalidTarget, InvalidParameter);

void
    setVisibleSizesAdjustment(
        in sequence<AdjustmentAxis>
            set_of_axes,
        in AdjustmentPolicy
            adjustment_policy,
        in unsigned long
            transition_duration)
raises(InvalidTarget);

void
    setBox(
        in BoxConstants
            box)
raises(InvalidTarget);
```

```
void
    setDefaultBackground(
        in unsigned short
            background,
        in unsigned long
            transition_duration)
raises(InvalidTarget, InvalidParameter);

void
    setAttachmentPoint(
        in AttachmentPointType
            type,
        in AttachmentPoint
            positions)
raises(InvalidTarget, InvalidParameter);

void
    setAttachmentPointPosition(
        in AttachmentPointType
            type,
        in ReferenceType
            vseap_reference_point,
        in Lengths
            the_lengths,
        in unsigned long
            transition_duration)
raises(InvalidTarget, InvalidParameter);

void
    setVisibleSizesAlignment(
        in SizeBorder
            size_border,
        in long
            interval,
        in unsigned long
            transition_duration)
raises(InvalidTarget);

void
    setMovingAbility(
        in UserControls
            moving_ability)
raises(InvalidTarget);

void
    setResizingAbility(
        in UserControls
            resizing_ability)
raises(InvalidTarget);

void
    setScalingAbility(
        in UserControls
            scaling_ability)
raises(InvalidTarget);

void
    setScrollingAbility(
        in UserControls
            scrolling_ability)
raises(InvalidTarget);

unsigned short
    getGSR()
raises(InvalidTarget);

SpecifiedPosition
    getPS(
        in GSIndicator
            gs)
raises(InvalidTarget);

AspectRatio
    getAspectRatio()
raises(InvalidTarget);
```

```
SpecifiedPosition
    getPSAP(
        in PointType
            point_type)
raises(InvalidTarget);

VSGS
    getVSGS()
raises(InvalidTarget);

SpecifiedPosition
    getVS()
raises(InvalidTarget);

BoxConstants
    getBox()
raises(InvalidTarget);

unsigned short
    getDefaultBackground()
raises(InvalidTarget);

SpecifiedPosition
    getVSIAP(
        in PointType
            point_type)
raises(InvalidTarget);

SpecifiedPosition
    getVSIAPPosition()
raises(InvalidTarget);

SpecifiedPosition
    getVSEAP(
        in PointType
            point_type)
raises(InvalidTarget);

SpecifiedPosition
    getVSEAPPosition(
        in ReferencePoint
            reference_point)
raises(InvalidTarget);

UserControls
    getMovingAbility()
raises(InvalidTarget);

UserControls
    getResizingAbility()
raises(InvalidTarget);

UserControls
    getScalingAbility()
raises(InvalidTarget);

UserControls
    getScrollingAbility()
raises(InvalidTarget);

void
    setSelectability(
        in unsigned short
            min_number_of_selections,
        in unsigned short
            max_number_of_selections)
raises(InvalidTarget, InvalidParameter);

void
    setSelectionStatus(
        in SelectionStatusValue
            selection_state)
raises(InvalidTarget);

void
    setSelectionPresentationEffectResponsibility(
        in Responsibility
            the_responsibility)
```

```
    raises(InvalidTarget);

    void
        getSelectability(
            out unsigned short
                min_number_of_selections,
            out unsigned short
                max_number_of_selections)
    raises(InvalidTarget);

    EffectiveSelectability
        getEffectiveSelectability()
    raises(InvalidTarget);

    SelectionStatusValue
        getSelectionStatus()
    raises(InvalidTarget);

    SelectionModeValue
        getSelectionMode()
    raises(InvalidTarget);

    Responsibility
        getSelectionPresentationEffectResponsibility()
    raises(InvalidTarget);

    void
        setModifiability(
            in unsigned short
                min_number_of_modifications,
            in unsigned short
                max_number_of_modifications)
    raises(InvalidTarget, InvalidParameter);

    void
        setModificationStatus(
            in ModificationStatusValue
                modification_state)
    raises(InvalidTarget);

    void
        setModificationPresentationEffectResponsibility(
            in Responsibility
                the_responsibility)
    raises(InvalidTarget);

    void
        getModifiability(
            out unsigned short
                min_numbers_of_modifications,
            out unsigned short
                max_numbers_of_modifications)
    raises(InvalidTarget);

    EffectiveModifiability
        getEffectiveModifiability()
    raises(InvalidTarget);

    ModificationStatusValue
        getModificationStatus()
    raises(InvalidTarget);

    ModificationModeValue
        getModificationMode()
    raises(InvalidTarget);

    Responsibility
        getModificationPresentationEffectResponsibility()
    raises(InvalidTarget);

    void
        setNoInteractionStyle()
    raises(InvalidTarget);

};
```

```
interface   RtComponent: RtComponentOrSocket, RtObject {};

interface   RtCompositeOrStructuralSocket {

    void
        setResizingStrategy(
            in ResizingStrategy
                resizing_strategy)
    raises(InvalidTarget);

    ResizingStrategy
        getResizingStrategy()
    raises(InvalidTarget);

    void
        setAudibleCompositionEffect(
            in unsigned short
                audible_effect,
            in unsigned long
                transition_duration)
    raises(InvalidTarget);

    unsigned short
        getAudibleCompositionEffect()
    raises(InvalidTarget);

    unsigned short
        getNumberOfSelectedSockets()
    raises(InvalidTarget);

    unsigned short
        getNumberOfModifiedSockets()
    raises(InvalidTarget);

    void
        setMenuInteractionStyle(
            in Orientation
                upper_menu_orientation,
            in sequence <Association>
                list_of_associations)
    raises(InvalidTarget, InvalidParameter);

    void
        setScrollingListInteractionStyle(
            in PerceptibleReference
                background,
            in unsigned short
                visible_items_number,
            in SocketTail
                first_item,
            in Separator
                the_separator,
            in Orientation
                the_orientation,
            in SliderSide
                slider_side,
            in PerceptibleReference
                slider,
            in PerceptibleReference
                slider_cursor,
            in PerceptibleReference
                slider_background,
            in long
                slider_min_value,
            in long
                slider_max_value)
    raises(InvalidTarget, InvalidParameter);

};
interface RtComposite: RtCompositeOrStructuralSocket, RtComponent {};

interface StructuralSocket: RtCompositeOrStructuralSocket, Socket {};

interface RtGenericContentOrPresentableSocket {
```

```
    void
        setAudibleVolume(
            in AudibleVolume
                audible_volume,
            in unsigned long
                transition_duration)
    raises(InvalidTarget, InvalidParameter);

    unsigned long
        getInitialOriginalAudibleVolume()
    raises(InvalidTarget);

    unsigned long
        getCurrentOriginalAudibleVolume()
    raises(InvalidTarget);

    unsigned long
        getEffectiveOriginalAudibleVolume()
    raises(InvalidTarget);

    unsigned long
        getPerceptibleAudibleVolume()
    raises(InvalidTarget);

    void
        setButtonInteractionStyle(
            in PresentationState
                initial_state,
            in AlternatePresentation
                alternate_presentation_1,
            in AlternatePresentation
                alternate_presentation_2,
            in AlternatePresentation
                alternate_presentation_3)
    raises(InvalidTarget, InvalidParameter);

};
interface  RtGenericContent: RtGenericContentOrPresentableSocket, RtComponent {};

interface  GenericPresentableSocket: RtGenericContentOrPresentableSocket, Socket
{};

interface RtContentOrPresentableSocket {

    void
        setSliderInteractionStyle(
            in PerceptibleReference
                cursor,
            in PerceptibleReference
                background,
            in Orientation
                the_orientation,
            in short
                min_value,
            in short
                max_value)
    raises(InvalidTarget, InvalidParameter);

    void
        setEntryFieldInteractionStyle(
            in EchoStyle
                echo_style,
            in PerceptibleReference
                background)
    raises(InvalidTarget, InvalidParameter);

};

interface RtContent: RtContentOrPresentableSocket, RtGenericContent {};

interface PresentableSocket: RtContentOrPresentableSocket,
GenericPresentableSocket {};

interface RtMultiplexedContentOrPresentableSocket {
```

```
    void
        setStreamChoice(
            in StreamIdentifier
                stream_identifier)
    raises(InvalidTarget, InvalidParameter);

    StreamValue
        getStreamChosen()
    raises(InvalidTarget);
};

interface RtMultiplexedContent: RtMultiplexedContentOrPresentableSocket,
RtGenericContent {};

interface MultiplexedPresentableSocket:  RtMultiplexedContentOrPresentableSocket,
GenericPresentableSocket {};

interface Channel: Entity {

    ChannelIdentifier
        bind(
            in ChannelReference
                channel_reference)
    raises(AlreadyBound, InvalidTarget);

    void
        unbind()
    raises(NotBound);

    ChannelIdentifier
        new(
            in ChannelReference
                channel_reference,
            in OriginalDefDeclaration
                original_definition_declaration)
    raises(AlreadyBound, InvalidTarget);

    void
        delete()
    raises(NotBound, InvalidTarget);

    ChannelStatusValue
        getAvailability()
    raises(NotBound, InvalidTarget);

    ChannelIdentifier
        getIdentifier()
    raises(NotBound);

    void
        kill();

    void
        setPerceptability(
            in ChannelPerceptabilityValue
                channel_perceptability)
    raises(InvalidTarget);

    ChannelPerceptabilityValue
        getPerceptability()
    raises(InvalidTarget);

    sequence<PerceptibleReference>
        getAssignedPerceptibles()
    raises(InvalidTarget);

};

module x {

interface EvaluatedValue;

// Corresponding MHEG datatype: RGS-Constants
//=====================================================================
enum RGSConstants {
    NULL_CHANNEL,
    PRGS
};
```

```
// Corresponding MHEG datatype: Stream-Chosen
//=========================================================================
enum StreamChosen {
    NO_STREAM_CHOSEN
};

// Corresponding MHEG datatype: Selection-Mode-Constants
//=========================================================================
enum SelectionModeConstants {
    NO_SELECTION,
    MHEG_ACTION,
    USING_APPLICATION_ACTION
};

// Corresponding MHEG datatype: Modification-Mode-Constants
//=========================================================================
enum ModificationModeConstants {
    MODIFICATION_MODE_CONSTANTS_NO_MODIFICATION,
    MODIFICATION_MODE_CONSTANTS_MHEG_ACTION,
    MODIFICATION_MODE_CONSTANTS_USING_APPLICATION_ACTION,
    MODIFICATION_MODE_CONSTANTS_CHILD
};

// Corresponding MHEG datatype: Comparison-Value-Constant
//=========================================================================
enum ComparisonValueConstantTag { TEMPORAL_TERMINATION_TAG,
RESIZING_STRATEGY_TAG, VSGS_TAG, RESPONSIBILITY_TAG,
PREPARATION_STATUS_VALUE_TAG, RT_AVAILABILITY_STATUS_VALUE_TAG,
RUNNING_STATUS_VALUE_TAG, TERMINATION_STATUS_VALUE_TAG, RGS_CONSTANTS_TAG,
USER_CONTROLS_TAG, ASPECT_RATIO_TAG, BOX_CONSTANTS_TAG, STREAM_CHOSEN_TAG,
EFFECTIVE_SELECTABILITY_TAG, SELECTION_STATUS_VALUE_TAG,
SELECTION_MODE_CONSTANTS_TAG, EFFECTIVE_MODIFIABILITY_TAG,
MODIFICATION_STATUS_VALUE_TAG, MODIFICATION_MODE_CONSTANTS_TAG,
CHANNEL_STATUS_VALUE_TAG, CHANNEL_PERCEPTABILITY_VALUES_TAG };
union ComparisonValueConstant
switch (ComparisonValueConstantTag){
    case TEMPORAL_TERMINATION_TAG:
        TemporalTermination
            temporal_termination;
    case RESIZING_STRATEGY_TAG:
        ResizingStrategy
            resizing_strategy;
    case VSGS_TAG:
        VSGS
            vsgs;
    case RESPONSIBILITY_TAG:
        Responsibility
            responsibility;
    case PREPARATION_STATUS_VALUE_TAG:
        PreparationStatusValue
            preparation_status_value;
    case RT_AVAILABILITY_STATUS_VALUE_TAG:
        RtAvailabilityStatusValue
            rt_availability_status_value;
    case RUNNING_STATUS_VALUE_TAG:
        RunningStatusValue
            running_status_value;
    case TERMINATION_STATUS_VALUE_TAG:
        TerminationStatusValue
            termination_status_value;
    case RGS_CONSTANTS_TAG:
        RGSConstants
            rgs_constants;
    case USER_CONTROLS_TAG:
        UserControls
            user_controls;
    case ASPECT_RATIO_TAG:
        AspectRatio
            aspect_ratio;
    case BOX_CONSTANTS_TAG:
        BoxConstants
            box_constants;
```

```
    case STREAM_CHOSEN_TAG:
        StreamChosen
            stream_chosen;
    case EFFECTIVE_SELECTABILITY_TAG:
        EffectiveSelectability
            effective_selectability;
    case SELECTION_STATUS_VALUE_TAG:
        SelectionStatusValue
            selection_status_value;
    case SELECTION_MODE_CONSTANTS_TAG:
        SelectionModeConstants
            selection_mode_constants;
    case EFFECTIVE_MODIFIABILITY_TAG:
        EffectiveModifiability
            effective_modifiability;
    case MODIFICATION_STATUS_VALUE_TAG:
        ModificationStatusValue
            modification_status_value;
    case MODIFICATION_MODE_CONSTANTS_TAG:
        ModificationModeConstants
            modification_mode_constants;
    case CHANNEL_STATUS_VALUE_TAG:
        ChannelStatusValue
            channel_status_value;
    case CHANNEL_PERCEPTABILITY_VALUES_TAG:
        ChannelPerceptabilityValue
            channel_perceptability_value;
};

// Corresponding MHEG datatype: Data-Reference
//=========================================================================
enum DataReferenceTag { EXTERNAL_IDENTIFIER_TAG,ALIAS_TAG };
union DataReference
switch (DataReferenceTag){
    case EXTERNAL_IDENTIFIER_TAG:
        ExternalIdentifier
            external_identifier;
    case ALIAS_TAG:
        Alias
            alias;
};

// Corresponding MHEG datatype: Reference
//=========================================================================
enum ReferenceTag { MHEG_OBJECT_REF_TAG, RT_OBJECT_REF_TAG, CHANNEL_ID_TAG,
SOCKET_IDENT_REF_TAG, NULL_REF_TAG };
union Reference
switch (ReferenceTag){
    case MHEG_OBJECT_REF_TAG:
        MhObjectReference
            mh_object_ref;
    case RT_OBJECT_REF_TAG:
        RtObjectReference
            rt_object_ref;
    case CHANNEL_ID_TAG:
        ChannelIdentifier
            channel_id;
    case SOCKET_IDENT_REF_TAG:
        SocketIdentification
            socket_ident_ref;
};

// Corresponding MHEG datatype: Generic-Reference
//=========================================================================
enum GenericReferenceTag { REFERENCE_TAG, EVALUATED_VALUE_TAG, UNSPECIFIED_TAG };
union GenericReference
switch (GenericReferenceTag){
    case REFERENCE_TAG:
        Reference
            reference;
    case EVALUATED_VALUE_TAG:
        EvaluatedValue
            evaluated_value;
};
```

```
// Corresponding MHEG datatype: Macro-Def-Id
//=========================================================================
enum MacroDefIdTag { STRING_FIELD_TAG, NUMERIC_TAG };
union MacroDefId
switch (MacroDefIdTag){
    case STRING_FIELD_TAG:
        string
            string_field;
    case NUMERIC_TAG:
        long
            numeric;
};

// Corresponding MHEG datatype: Target-Macro
//=========================================================================
struct TargetMacro {
    MacroDefId
        macro_def_id;
    sequence<GenericReference,1>
        generic_reference;
};

// Corresponding MHEG datatype: Target-Parameter
//=========================================================================
enum TargetParameterTag { GENERIC_REFERENCE_TAG, TARGET_MACRO_TAG };
union TargetParameter
switch (TargetParameterTag){
    case GENERIC_REFERENCE_TAG:
        GenericReference
            generic_reference;
    case TARGET_MACRO_TAG:
        TargetMacro
            target_macro;
};

// Corresponding MHEG datatype: Targets-Parameter
//=========================================================================
typedef sequence<TargetParameter> TargetsParameter;

// Corresponding MHEG datatype: Mh-Target-Macro
//=========================================================================
struct MhTargetMacro {
    MacroDefId
        macro_def_id;
    sequence<MhObjectReference,1>
        mh_object_reference;
};

// Corresponding MHEG datatype: Mh-Target-Parameter
//=========================================================================
enum MhTargetParameterTag { MH_REFERENCE_TAG, EVALUATED_TARGET_TAG,
MH_TARGET_MACRO_TAG };
union MhTargetParameter
switch (MhTargetParameterTag){
    case MH_REFERENCE_TAG:
        MhObjectReference
            mh_reference;
    case EVALUATED_TARGET_TAG:
        EvaluatedValue
            evaluated_target;
    case MH_TARGET_MACRO_TAG:
        MhTargetMacro
            mh_target_macro;
};

// Corresponding MHEG datatype: Mh-Targets-Parameter
//=========================================================================
typedef sequence<MhTargetParameter> MhTargetsParameter;
```

```
// Corresponding MHEG datatype: Rt-Target-Macro
//=========================================================================
struct RtTargetMacro {
    MacroDefId
        macro_def_id;
    sequence<RtReference,1>
        rt_reference;
};

// Corresponding MHEG datatype: Rt-Target-Parameter
//=========================================================================
enum RtTargetParameterTag { RT_TARGET_PARAMETER_RT_REFERENCE_TAG,
RT_TARGET_PARAMETER_EVALUATED_TARGET_TAG, RT_TARGET_PARAMETER_RT_TARGET_MACRO_TAG
};
union RtTargetParameter
switch (RtTargetParameterTag){
    case RT_TARGET_PARAMETER_RT_REFERENCE_TAG:
        RtReference
            rt_reference;
    case RT_TARGET_PARAMETER_EVALUATED_TARGET_TAG:
        EvaluatedValue
            evaluated_target;
    case RT_TARGET_PARAMETER_RT_TARGET_MACRO_TAG:
        RtTargetMacro
            rt_target_macro;
};

// Corresponding MHEG datatype: Rt-Targets-Parameter
//=========================================================================
typedef sequence<RtTargetParameter> RtTargetsParameter;

// Corresponding MHEG datatype: Socket-Target-Macro
//=========================================================================
struct SocketTargetMacro {
    MacroDefId
        macro_def_id;
    sequence<SocketReference,1>
        socket_reference;
};

// Corresponding MHEG datatype: Socket-Target-Parameter
//=========================================================================
enum SocketTargetParameterTag { SOCKET_TARGET_PARAMETER_SOCKET_REFERENCE_TAG,
SOCKET_TARGET_PARAMETER_EVALUATED_TARGET_TAG,
SOCKET_TARGET_PARAMETER_SOCKET_TARGET_MACRO_TAG };
union SocketTargetParameter
switch (SocketTargetParameterTag){
    case SOCKET_TARGET_PARAMETER_SOCKET_REFERENCE_TAG:
        SocketReference
            socket_reference;
    case SOCKET_TARGET_PARAMETER_EVALUATED_TARGET_TAG:
        EvaluatedValue
            evaluated_target;
    case SOCKET_TARGET_PARAMETER_SOCKET_TARGET_MACRO_TAG:
        SocketTargetMacro
            socket_target_macro;
};

// Corresponding MHEG datatype: Socket-Targets-Parameter
//=========================================================================
typedef sequence<SocketTargetParameter> SocketTargetsParameter;

// Corresponding MHEG datatype: Rt-Socket-Target-Macro
//=========================================================================
struct RtSocketTargetMacro {
    MacroDefId
        macro_def_id;
    sequence<RtObjectSocketReference,1>
        rt_socket_reference;
};
```

```
// Corresponding MHEG datatype: Rt-Socket-Target-Parameter
//=========================================================================
enum RtSocketTargetParameterTag {
RT_SOCKET_TARGET_PARAMETER_RT_SOCKET_REFERENCE_TAG,
RT_SOCKET_TARGET_PARAMETER_EVALUATED_TARGET_TAG,
RT_SOCKET_TARGET_PARAMETER_RT_SOCKET_TARGET_MACRO_TAG };
union RtSocketTargetParameter
switch (RtSocketTargetParameterTag){
    case RT_SOCKET_TARGET_PARAMETER_RT_SOCKET_REFERENCE_TAG:
        RtObjectSocketReference
            rt_socket_reference;
    case RT_SOCKET_TARGET_PARAMETER_EVALUATED_TARGET_TAG:
        EvaluatedValue
            evaluated_target;
    case RT_SOCKET_TARGET_PARAMETER_RT_SOCKET_TARGET_MACRO_TAG:
        RtSocketTargetMacro
            rt_socket_target_macro;
};

// Corresponding MHEG datatype: Rt-Socket-Targets-Parameter
//=========================================================================
typedef sequence<RtSocketTargetParameter> RtSocketTargetsParameter;

// Corresponding MHEG datatype: Channel-Target-Macro
//=========================================================================
struct ChannelTargetMacro {
    MacroDefId
        macro_def_id;
    sequence<ChannelReference,1>
        channel_reference;
};

// Corresponding MHEG datatype: Channel-Target-Parameter
//=========================================================================
enum ChannelTargetParameterTag { CHANNEL_TARGET_PARAMETER_CHANNEL_REFERENCE_TAG,
CHANNEL_TARGET_PARAMETER_EVALUATED_TARGET_TAG,
CHANNEL_TARGET_PARAMETER_CHANNEL_TARGET_MACRO_TAG };
union ChannelTargetParameter
switch (ChannelTargetParameterTag){
    case CHANNEL_TARGET_PARAMETER_CHANNEL_REFERENCE_TAG:
        ChannelReference
            channel_reference;
    case CHANNEL_TARGET_PARAMETER_EVALUATED_TARGET_TAG:
        EvaluatedValue
            evaluated_target;
    case CHANNEL_TARGET_PARAMETER_CHANNEL_TARGET_MACRO_TAG:
        ChannelTargetMacro
            channel_target_macro;
};

// Corresponding MHEG datatype: Channel-Targets-Parameter
//=========================================================================
typedef sequence<ChannelTargetParameter> ChannelTargetsParameter;

// Corresponding MHEG datatype: Element-List-Index-Macro
//=========================================================================
struct ElementListIndexMacro {
    MacroDefId
        macro_def_id;
    sequence<sequence<GenericNumeric>,1>
        generic_numeric_list;
};

// Corresponding MHEG datatype: Element-List-Index-Parameter
//=========================================================================
enum ElementListIndexParameterTag { GENERIC_NUMERIC_LIST_TAG,
ELEMENT_LIST_INDEX_MACRO_TAG };
union ElementListIndexParameter
switch (ElementListIndexParameterTag){
    case GENERIC_NUMERIC_LIST_TAG:
        sequence<GenericNumeric>
            generic_numeric_list;
```

```
    case ELEMENT_LIST_INDEX_MACRO_TAG:
        ElementListIndexMacro
            element_list_index_macro;
};

// Corresponding MHEG datatype: Get-Data
//=========================================================================
struct GetData {
    MhTargetParameter
        content_target_parameter;
    sequence<ElementListIndexParameter,1>
        element_list_index_parameter;
};

// Corresponding MHEG datatype: GT-Indicator-Macro
//=========================================================================
struct GTIndicatorMacro {
    MacroDefId
        macro_def_id;
    sequence<GTIndicator,1>
        gt_indicator;
};

// Corresponding MHEG datatype: GT-Indicator-Parameter
//=========================================================================
enum GTIndicatorParameterTag { GT_INDICATOR_TAG, GT_INDICATOR_MACRO_TAG };
union GTIndicatorParameter
switch (GTIndicatorParameterTag){
    case GT_INDICATOR_TAG:
        GTIndicator
            gt_indicator;
    case GT_INDICATOR_MACRO_TAG:
        GTIndicatorMacro
            gt_indicator_macro;
};

// Corresponding MHEG datatype: Get-PD
//=========================================================================
struct GetPD {
    RtSocketTargetParameter
        target;
    sequence<GTIndicatorParameter,1>
        gt_indicator;
};

// Corresponding MHEG datatype: Get-VD-Length
//=========================================================================
struct GetVDLength {
    RtSocketTargetParameter
        target;
    GTIndicatorParameter
        gt_indicator;
};

// Corresponding MHEG datatype: GS-Indicator-Macro
//=========================================================================
struct GSIndicatorMacro {
    MacroDefId
        macro_def_id;
    sequence<GSIndicator,1>
        gs_indicator;
};

// Corresponding MHEG datatype: GS-Indicator-Parameter
//=========================================================================
enum GSIndicatorParameterTag { GS_INDICATOR_TAG, GS_INDICATOR_MACRO_TAG };
union GSIndicatorParameter
switch (GSIndicatorParameterTag){
    case GS_INDICATOR_TAG:
        GSIndicator
            gs_indicator;
    case GS_INDICATOR_MACRO_TAG:
        GSIndicatorMacro
            gs_indicator_macro;
};
```

```
// Corresponding MHEG datatype: Perceptible-Size-Parameter
//=========================================================================
struct PerceptibleSizeParameter {
    RtSocketTargetParameter
        target;
    sequence<GSIndicatorParameter,1>
        generic_space;
};

// Corresponding MHEG datatype: Point-Type-Macro
//=========================================================================
struct PointTypeMacro {
    MacroDefId
        macro_def_id;
    sequence<PointType,1>
        point_type;
};

// Corresponding MHEG datatype: Point-Type-Parameter
//=========================================================================
enum PointTypeParameterTag { POINT_TYPE_TAG, POINT_TYPE_MACRO_TAG };
union PointTypeParameter
switch (PointTypeParameterTag){
    case POINT_TYPE_TAG:
        PointType
            point_type;
    case POINT_TYPE_MACRO_TAG:
        PointTypeMacro
            point_type_macro;
};


// Corresponding MHEG datatype: AP-Parameter
//=========================================================================
struct APParameter {
      RtSocketTargetParameter
         target;
      sequence<PointTypeParameter,1>
         type;
};


// Corresponding MHEG datatype: Reference-Point-Macro
//=========================================================================
struct ReferencePointMacro {
      MacroDefId
         macro_def_id;
      sequence<ReferencePoint,1>
         reference_point;
};


// Corresponding MHEG datatype: Reference-Point-Parameter
//=========================================================================
enum ReferencePointParameterTag { REFERENCE_POINT_TAG, REFERENCE_POINT_MACRO_TAG
};
union ReferencePointParameter
switch (ReferencePointParameterTag){
      case REFERENCE_POINT_TAG:
         ReferencePoint
            reference_point;
      case REFERENCE_POINT_MACRO_TAG:
         ReferencePointMacro
            reference_point_macro;
};
```

```
// Corresponding MHEG datatype: VSEAP-Position-Parameter
//=========================================================================
struct VSEAPPositionParameter {
       RtSocketTargetParameter
          target;
       sequence<ReferencePointParameter,1>
          reference_point;
};


// Corresponding MHEG datatype: Generic-Value
//=========================================================================
enum GenericValueTag { GENERIC_VALUE_BOOLEAN_FIELD_TAG,
GENERIC_VALUE_NUMERIC_TAG, GENERIC_VALUE_STRING_FIELD_TAG,
GENERIC_VALUE_GENERIC_LIST_TAG, GENERIC_VALUE_REFERENCE_TAG,
GENERIC_VALUE_UNSPECIFIED_TAG, GENERIC_VALUE_EVALUATED_VALUE_TAG };
union GenericValue
switch (GenericValueTag){
       case GENERIC_VALUE_BOOLEAN_FIELD_TAG:
          boolean
             boolean_field;
       case GENERIC_VALUE_NUMERIC_TAG:
          long
             numeric;
       case GENERIC_VALUE_STRING_FIELD_TAG:
          string
             string_field;
       case GENERIC_VALUE_GENERIC_LIST_TAG:
          sequence<GenericValue>
             generic_list;
       case GENERIC_VALUE_REFERENCE_TAG:
          Reference
             reference;
       case GENERIC_VALUE_EVALUATED_VALUE_TAG:
          EvaluatedValue
             evaluated_value;
};
// Corresponding MHEG datatype: Generic-Boolean
//=========================================================================
enum GenericBooleanTag { GENERIC_BOOLEAN_CONSTANT_TAG,
GENERIC_BOOLEAN_EVALUATED_VALUE_TAG, GENERIC_BOOLEAN_UNSPECIFIED_TAG };
union GenericBoolean
switch (GenericBooleanTag){
    case GENERIC_BOOLEAN_CONSTANT_TAG:
        boolean
           constant;
    case GENERIC_BOOLEAN_EVALUATED_VALUE_TAG:
        EvaluatedValue
           evaluated_value;
};
// Corresponding MHEG datatype: Generic-Numeric
//=========================================================================
enum GenericNumericTag { GENERIC_NUMERIC_CONSTANT_TAG,
GENERIC_NUMERIC_EVALUATED_VALUE_TAG, GENERIC_NUMERIC_UNSPECIFIED_TAG };
union GenericNumeric
switch (GenericNumericTag){
    case GENERIC_NUMERIC_CONSTANT_TAG:
        long
           constant;
    case GENERIC_NUMERIC_EVALUATED_VALUE_TAG:
        EvaluatedValue
           evaluated_value;
};
// Corresponding MHEG datatype: Generic-String
//=========================================================================
enum GenericStringTag { GENERIC_STRING_CONSTANT_TAG,
GENERIC_STRING_EVALUATED_VALUE_TAG, GENERIC_STRING_UNSPECIFIED_TAG };
union GenericString
switch (GenericStringTag){
```

```
    case GENERIC_STRING_CONSTANT_TAG:
        string
            constant;
    case GENERIC_STRING_EVALUATED_VALUE_TAG:
        EvaluatedValue
            evaluated_value;
};

// Corresponding MHEG datatype: Value-Macro
//=========================================================================
struct ValueMacro {
    MacroDefId
        macro_def_id;
    sequence<GenericValue,1>
        generic_value;
};

// Corresponding MHEG datatype: Boolean-Macro
//=========================================================================
struct BooleanMacro {
    MacroDefId
        macro_def_id;
    sequence<GenericBoolean,1>
        generic_boolean;
};

// Corresponding MHEG datatype: Numeric-Macro
//=========================================================================
struct NumericMacro {
    MacroDefId
        macro_def_id;
    sequence<GenericNumeric,1>
        generic_numeric;
};

// Corresponding MHEG datatype: String-Macro
//=========================================================================
struct StringMacro {
    MacroDefId
        macro_def_id;
    sequence<GenericString,1>
        generic_string;
};

// Corresponding MHEG datatype: Boolean-Parameter
//=========================================================================
enum BooleanParameterTag { GENERIC_BOOLEAN_TAG, BOOLEAN_MACRO_TAG };
union BooleanParameter
switch (BooleanParameterTag){
    case GENERIC_BOOLEAN_TAG:
        GenericBoolean
            generic_boolean;
    case BOOLEAN_MACRO_TAG:
        BooleanMacro
            boolean_macro;
};

// Corresponding MHEG datatype: Numeric-Parameter
//=========================================================================
enum NumericParameterTag { GENERIC_NUMERIC_TAG, NUMERIC_MACRO_TAG };
union NumericParameter
switch (NumericParameterTag){
    case GENERIC_NUMERIC_TAG:
        GenericNumeric
            generic_numeric;
    case NUMERIC_MACRO_TAG:
        NumericMacro
            numeric_macro;
};
```

```
// Corresponding MHEG datatype: String-Parameter
//=========================================================================
enum StringParameterTag { GENERIC_STRING_TAG, STRING_MACRO_TAG };
union StringParameter
switch (StringParameterTag){
    case GENERIC_STRING_TAG:
        GenericString
            generic_string;
    case STRING_MACRO_TAG:
        StringMacro
            string_macro;
};

// Corresponding MHEG datatype: Value-Parameter
//=========================================================================
enum ValueParameterTag { GENERIC_VALUE_TAG, VALUE_MACRO_TAG };
union ValueParameter
switch (ValueParameterTag){
    case GENERIC_VALUE_TAG:
        GenericValue
            generic_value;
    case VALUE_MACRO_TAG:
        ValueMacro
            value_macro;
};

// Corresponding MHEG datatype: Generic-Volume-Range
//=========================================================================
struct GenericVolumeRange {
    sequence<long,1>
        maximum_volume;
    sequence<long,1>
        minimum_volume;
};

// Corresponding MHEG datatype: Original-Size
//=========================================================================
struct OriginalSize {
    sequence<long,1>
        x_length;
    sequence<long,1>
        y_length;
    sequence<long,1>
        z_length;
};

// Corresponding MHEG datatype: Original-Perception
//=========================================================================
struct OriginalPerception {
    sequence<long,1>
        initial_GTR;
    sequence<long,1>
        original_duration;
    sequence<OriginalSize,1>
        original_size;
    sequence<GenericVolumeRange,1>
        audible_volume_range;
    sequence<long,1>
        original_volume;
};

// Corresponding MHEG datatype: Content-Encoding-Identification
//=========================================================================
enum ContentEncodingIdentificationTag { MHEG_CONTENT_CATALOGUE_TAG,
PROPRIETARY_CONTENT_CATALOGUE_TAG };
union ContentEncodingIdentification
switch (ContentEncodingIdentificationTag){
    case MHEG_CONTENT_CATALOGUE_TAG:
        long
            mheg_content_catalogue;
    case PROPRIETARY_CONTENT_CATALOGUE_TAG:
        long
            proprietary_content_catalogue;
};
```

```
// Corresponding MHEG datatype: Content-Hook
//=========================================================================
struct ContentHook {
    ContentEncodingIdentification
        content_encoding_identification;
    string
        content_encoding_description;
};

// Corresponding MHEG datatype: Script-Encoding-Identification
//=========================================================================
enum ScriptEncodingIdentificationTag { MHEG_SCRIPT_CATALOGUE_TAG,
PROPRIETARY_SCRIPT_CATALOGUE_TAG };
union ScriptEncodingIdentification
switch (ScriptEncodingIdentificationTag){
    case MHEG_SCRIPT_CATALOGUE_TAG:
        long
            mheg_script_catalogue;
    case PROPRIETARY_SCRIPT_CATALOGUE_TAG:
        long
            proprietary_script_catalogue;
};

// Corresponding MHEG datatype: Script-Hook
//=========================================================================
struct ScriptHook {
    ScriptEncodingIdentification
        script_encoding_identification;
    string
        script_encoding_description;
};

// Corresponding MHEG datatype: Content-Classification
//=========================================================================
enum ContentClassificationTag { MHEG_CONTENT_CLASSIFICATION_TAG,
PROPRIETARY_CONTENT_CLASSIFICATION_TAG };
union ContentClassification
switch (ContentClassificationTag){
    case MHEG_CONTENT_CLASSIFICATION_TAG:
        long
            mheg_content_classification;
    case PROPRIETARY_CONTENT_CLASSIFICATION_TAG:
        long
            proprietary_content_classification;
};

// Corresponding MHEG datatype: Script-Classification
//=========================================================================
enum ScriptClassificationTag { MHEG_SCRIPT_CLASSIFICATION_TAG,
PROPRIETARY_SCRIPT_CLASSIFICATION_TAG };
union ScriptClassification
switch (ScriptClassificationTag){
    case MHEG_SCRIPT_CLASSIFICATION_TAG:
        long
            mheg_script_classification;
    case PROPRIETARY_SCRIPT_CLASSIFICATION_TAG:
        long
            proprietary_script_classification;
};

// Corresponding MHEG datatype: Evaluated-Value
//=========================================================================
interface EvaluatedValue {
enum EvaluatedValueTag { GET_PREPARATION_STATUS_TAG, GET_DATA_TAG,
GET_RT_AVAILABILITY_STATUS_TAG, GET_GLOBAL_BEHAVIOUR_TAG, GET_RUNNING_STATUS_TAG,
GET_TERMINATION_STATUS_TAG, GET_RGS_TAG, GET_OPACITY_TAG,
GET_EFFECTIVE_OPACITY_TAG, GET_PRESENTATION_PRIORITY_TAG, GET_PD_TAG,
GET_INITIAL_TEMPORAL_POSITION_TAG, GET_TERMINAL_TEMPORAL_POSITION_TAG,
GET_VD_LENGTH_TAG, GET_CURRENT_TEMPORAL_POSITION_TAG, GET_VD_POSITION_TAG,
GET_SPEED_RATE_TAG, GET_OGTR_TAG, GET_EFFECTIVE_SPEED_RATE_TAG,
```

```
GET_EFFECTIVE_OGTR_TAG, GET_TIMESTONE_STATUS_TAG, GET_GSR_TAG, GET_PS_TAG,
GET_RESIZING_STRATEGY_TAG, GET_ASPECT_RATIO_TAG, GET_PSAP_TAG, GET_VSGS_TAG,
GET_VS_TAG, GET_BOX_TAG, GET_DEFAULT_BACKGROUND_TAG, GET_VSIAP_TAG,
GET_VSIAP_POSITION_TAG, GET_VSEAP_TAG, GET_VSEAP_POSITION_TAG,
GET_MOVING_ABILITY_TAG, GET_RESIZING_ABILITY_TAG, GET_SCALING_ABILITY_TAG,
GET_SCROLLING_ABILITY_TAG, GET_IOV_TAG, GET_CURRENT_OV_TAG,
GET_AUDIBLE_COMPOSITION_EFFECT_TAG, GET_EFFECTIVE_OV_TAG, GET_PERCEPTIBLE_OV_TAG,
GET_STREAM_CHOSEN_TAG, GET_SELECTABILITY_TAG, GET_EFFECTIVE_SELECTABILITY_TAG,
GET_SELECTION_STATUS_TAG, GET_SELECTION_MODE_TAG,
GET_NUMBER_SELECTED_SOCKETS_TAG,
GET_SELECTION_PRESENTATION_EFFECT_RESPONSIBILITY_TAG, GET_MODIFIABILITY_TAG,
GET_EFFECTIVE_MODIFIABILITY_TAG, GET_MODIFICATION_STATUS_TAG,
GET_MODIFICATION_MODE_TAG, GET_NUMBER_MODIFIED_SOCKETS_TAG,
GET_MODIFICATION_PRESENTATION_EFFECT_RESPONSIBILITY_TAG,
GET_CHANNEL_AVAILABILITY_STATUS_TAG, GET_CHANNEL_PERCEPTABILITY_TAG };
union EvaluatedValue
switch (EvaluatedValueTag){
    case GET_PREPARATION_STATUS_TAG:
        MhTargetParameter
            get_preparation_status;
    case GET_DATA_TAG:
        GetData
            get_data;
    case GET_RT_AVAILABILITY_STATUS_TAG:
        RtTargetParameter
            get_rt_availability_status;
    case GET_GLOBAL_BEHAVIOUR_TAG:
        RtTargetParameter
            get_global_behaviour;
    case GET_RUNNING_STATUS_TAG:
        RtTargetParameter
            get_running_status;
    case GET_TERMINATION_STATUS_TAG:
        RtTargetParameter
            get_termination_status;
    case GET_RGS_TAG:
        RtSocketTargetParameter
            get_RGS;
    case GET_OPACITY_TAG:
        RtSocketTargetParameter
            get_opacity;
    case GET_EFFECTIVE_OPACITY_TAG:
        RtSocketTargetParameter
            get_effective_opacity;
    case GET_PRESENTATION_PRIORITY_TAG:
        RtSocketTargetParameter
            get_presentation_priority;
    case GET_PD_TAG:
        GetPD
            get_PD;
    case GET_INITIAL_TEMPORAL_POSITION_TAG:
        RtSocketTargetParameter
            get_initial_temporal_position;
    case GET_TERMINAL_TEMPORAL_POSITION_TAG:
        RtSocketTargetParameter
            get_terminal_temporal_position;
    case GET_VD_LENGTH_TAG:
        GetVDLength
            get_VD_length;
    case GET_CURRENT_TEMPORAL_POSITION_TAG:
        RtSocketTargetParameter
            get_current_temporal_position;
    case GET_VD_POSITION_TAG:
        SocketTargetParameter
            get_VD_position;
    case GET_SPEED_RATE_TAG:
        RtSocketTargetParameter
            get_speed_rate;
```

```
case GET_OGTR_TAG:
    RtSocketTargetParameter
        get_OGTR;
case GET_EFFECTIVE_SPEED_RATE_TAG:
    RtSocketTargetParameter
        get_effective_speed_rate;
case GET_EFFECTIVE_OGTR_TAG:
    RtSocketTargetParameter
        get_effective_OGTR;
case GET_TIMESTONE_STATUS_TAG:
    RtSocketTargetParameter
        get_timestone_status;
case GET_GSR_TAG:
    RtSocketTargetParameter
        get_GSR;
case GET_PS_TAG:
    PerceptibleSizeParameter
        get_PS;
case GET_RESIZING_STRATEGY_TAG:
    RtSocketTargetParameter
        get_resizing_strategy;
case GET_ASPECT_RATIO_TAG:
    RtSocketTargetParameter
        get_aspect_ratio;
case GET_PSAP_TAG:
    APParameter
        get_PSAP;
case GET_VSGS_TAG:
    RtSocketTargetParameter
        get_VSGS;
case GET_VS_TAG:
    RtSocketTargetParameter
        get_VS;
case GET_BOX_TAG:
    RtSocketTargetParameter
        get_box;
case GET_DEFAULT_BACKGROUND_TAG:
    RtSocketTargetParameter
        get_default_background;
case GET_VSIAP_TAG:
    APParameter
        get_VSIAP;
case GET_VSIAP_POSITION_TAG:
    RtSocketTargetParameter
        get_VSIAP_position;
case GET_VSEAP_TAG:
    APParameter
        get_VSEAP;
case GET_VSEAP_POSITION_TAG:
    VSEAPPositionParameter
        get_VSEAP_position;
case GET_MOVING_ABILITY_TAG:
    RtSocketTargetParameter
        get_moving_ability;
case GET_RESIZING_ABILITY_TAG:
    RtSocketTargetParameter
        get_resizing_ability;
case GET_SCALING_ABILITY_TAG:
    RtSocketTargetParameter
        get_scaling_ability;
case GET_SCROLLING_ABILITY_TAG:
    RtSocketTargetParameter
        get_scrolling_ability;
case GET_IOV_TAG:
    RtSocketTargetParameter
        get_IOV;
case GET_CURRENT_OV_TAG:
    RtSocketTargetParameter
        get_current_OV;
```

```
    case GET_AUDIBLE_COMPOSITION_EFFECT_TAG:
        RtSocketTargetParameter
            get_audible_composition_effect;
    case GET_EFFECTIVE_OV_TAG:
        RtSocketTargetParameter
            get_effective_OV;
    case GET_PERCEPTIBLE_OV_TAG:
        RtSocketTargetParameter
            get_perceptible_OV;
    case GET_STREAM_CHOSEN_TAG:
        RtSocketTargetParameter
            get_stream_chosen;
    case GET_SELECTABILITY_TAG:
        RtSocketTargetParameter
            get_selectability;
    case GET_EFFECTIVE_SELECTABILITY_TAG:
        RtSocketTargetParameter
            get_effective_selectability;
    case GET_SELECTION_STATUS_TAG:
        RtSocketTargetParameter
            get_selection_status;
    case GET_SELECTION_MODE_TAG:
        RtSocketTargetParameter
            get_selection_mode;
    case GET_NUMBER_SELECTED_SOCKETS_TAG:
        RtSocketTargetParameter
            get_number_selected_sockets;
    case GET_SELECTION_PRESENTATION_EFFECT_RESPONSIBILITY_TAG:
        RtSocketTargetParameter
            get_selection_presentation_effect_responsibility;
    case GET_MODIFIABILITY_TAG:
        RtSocketTargetParameter
            get_modifiability;
    case GET_EFFECTIVE_MODIFIABILITY_TAG:
        RtSocketTargetParameter
            get_effective_modifiability;
    case GET_MODIFICATION_STATUS_TAG:
        RtSocketTargetParameter
            get_modification_status;
    case GET_MODIFICATION_MODE_TAG:
        RtSocketTargetParameter
            get_modification_mode;
    case GET_NUMBER_MODIFIED_SOCKETS_TAG:
        RtSocketTargetParameter
            get_number_modified_sockets;
    case GET_MODIFICATION_PRESENTATION_EFFECT_RESPONSIBILITY_TAG:
        RtSocketTargetParameter
            get_modification_presentation_effect_responsibility;
    case GET_CHANNEL_AVAILABILITY_STATUS_TAG:
        ChannelTargetParameter
            get_channel_availability_status;
    case GET_CHANNEL_PERCEPTABILITY_TAG:
        ChannelTargetParameter
            get_channel_perceptability;
};
attribute EvaluatedValue evaluated_value;
};

// Corresponding MHEG datatype: Delay-Targets
//=========================================================================
enum DelayTargetsTag { PERCEPTIBLE_TARGETS_TAG, NULL_CHANNEL_TAG };
union DelayTargets
switch (DelayTargetsTag){
    case PERCEPTIBLE_TARGETS_TAG:
        sequence<RtObjectSocketReference>
            perceptible_targets;
};
```

```
// Corresponding MHEG datatype: Delay-Targets-Macro
//=========================================================================
struct DelayTargetsMacro {
    MacroDefId
        macro_def_id;
    sequence<DelayTargets,1>
        delay_targets;
};

// Corresponding MHEG datatype: Delay-Targets-Parameter
//=========================================================================
enum DelayTargetsParameterTag { DELAY_TARGETS_TAG, DELAY_TARGETS_MACRO_TAG };
union DelayTargetsParameter
switch (DelayTargetsParameterTag){
    case DELAY_TARGETS_TAG:
        DelayTargets
            delay_targets;
    case DELAY_TARGETS_MACRO_TAG:
        DelayTargetsMacro
            delay_targets_macro;
};

// Corresponding MHEG datatype: Delay
//=========================================================================
struct Delay {
    sequence<DelayTargetsParameter,1>
        targets_parameter;
    NumericParameter
        duration;
};

// Corresponding MHEG datatype: Values
//=========================================================================
typedef sequence<GenericValue> Values;

// Corresponding MHEG datatype: Values-Macro
//=========================================================================
struct ValuesMacro {
    MacroDefId
        macro_def_if;
    sequence<Values,1>
        values;
};

// Corresponding MHEG datatype: Returned-Values-Parameter
//=========================================================================
enum ReturnedValuesParameterTag { VALUES_TAG, VALUES_MACRO_TAG };
union ReturnedValuesParameter
switch (ReturnedValuesParameterTag){
    case VALUES_TAG:
        Values
            values;
    case VALUES_MACRO_TAG:
        ValuesMacro
            values_macro;
};

// Corresponding MHEG datatype: Return
//=========================================================================
struct Return {
    NumericParameter
        return_indicator_param;
    sequence<ReturnedValuesParameter,1>
        returned_values_param;
    sequence<MhTargetsParameter,1>
        returned_objects_param;
};

// Corresponding MHEG datatype: Set-Alias
//=========================================================================
struct SetAlias {
    sequence<TargetsParameter,1>
        targets_parameter;
```

```
    StringParameter
        given_alias;
};

// Corresponding MHEG datatype: Data-Element
//=========================================================================
struct DataElement {
    sequence<ElementListIndexParameter,1>
        element_list_index_param;
    sequence<BooleanParameter,1>
        process_indicator_param;
    sequence<ValueParameter,1>
        value_parameter;
};

// Corresponding MHEG datatype: Set-Data
//=========================================================================
struct SetData {
       sequence<MhTargetsParameter,1>
          content_targets_param;
       sequence<BooleanParameter,1>
          substitution_indicator_param;
       sequence<sequence<DataElement>,1>
          data_elements;
};

// Corresponding MHEG datatype: Copy
//=========================================================================
struct Copy {
    sequence<MhTargetParameter,1>
        source;
    MhTargetsParameter
        copies;
};

// Corresponding MHEG datatype: Stream
//=========================================================================
struct Stream {
    sequence<long>
        stream_id;
    MhTargetParameter
        content_target;
};

// Corresponding MHEG datatype: Set-Multiplex
//=========================================================================
struct SetMultiplex {
    sequence<MhTargetsParameter,1>
        multiplex_targets;
    sequence<Stream>
        streams;
};

// Corresponding MHEG datatype: Set-Demultiplex
//=========================================================================
struct SetDemultiplex {
    sequence<MhTargetParameter,1>
        multiplex_targets;
    sequence<Stream>
        multiplex;
};

// Corresponding MHEG datatype: Global-Behaviour
//=========================================================================
enum GlobalBehaviourTag { GLOBAL_BEHAVIOUR_GET_GLOBAL_BEHAVIOUR_TAG,
GLOBAL_BEHAVIOUR_GET_DATA_TAG, GLOBAL_BEHAVIOUR_UNSPECIFIED_TAG };
union GlobalBehaviour
switch (GlobalBehaviourTag){
    case GLOBAL_BEHAVIOUR_GET_GLOBAL_BEHAVIOUR_TAG:
        RtTargetParameter
            get_global_behaviour;
```

```
    case GLOBAL_BEHAVIOUR_GET_DATA_TAG:
        GetData
            get_data;
};

// Corresponding MHEG datatype: Global-Behaviour-Macro
//=========================================================================
struct GlobalBehaviourMacro {
    MacroDefId
        macro_def_id;
    sequence<GlobalBehaviour,1>
        global_behaviour;
};

// Corresponding MHEG datatype: Global-Behaviour-Parameter
//=========================================================================
enum GlobalBehaviourParameterTag { GLOBAL_BEHAVIOUR_TAG,
GLOBAL_BEHAVIOUR_MACRO_TAG };
union GlobalBehaviourParameter
switch (GlobalBehaviourParameterTag){
    case GLOBAL_BEHAVIOUR_TAG:
        GlobalBehaviour
            global_behaviour;
    case GLOBAL_BEHAVIOUR_MACRO_TAG:
        GlobalBehaviourMacro
            global_behaviour_macro;
};

// Corresponding MHEG datatype: Set-Global-Behaviour
//=========================================================================
struct SetGlobalBehaviour {
    sequence<RtSocketTargetsParameter,1>
        rt_sockets_targets_param;
    sequence<GlobalBehaviourParameter,1>
        global_behaviour_param;
};

// Corresponding MHEG datatype: Parameter
//=========================================================================
enum ParameterTag { PARAMETER_GENERIC_VALUE_TAG, PARAMETER_CONTENT_TARGET_TAG };
union Parameter
switch (ParameterTag){
    case PARAMETER_GENERIC_VALUE_TAG:
        ValueParameter
            generic_value;
    case PARAMETER_CONTENT_TARGET_TAG:
        MhTargetParameter
            content_target;
};

// Corresponding MHEG datatype: Parameters-Macro
//=========================================================================
struct ParametersMacro {
    MacroDefId
        macro_def_id;
    sequence<sequence<Parameter>,1>
        parameters;
};

// Corresponding MHEG datatype: Parameters-Parameter
//=========================================================================
enum ParametersParameterTag { PARAMETERS_TAG, PARAMETERS_MACRO_TAG };
union ParametersParameter
switch (ParametersParameterTag){
    case PARAMETERS_TAG:
        sequence<Parameter>
            parameters;
    case PARAMETERS_MACRO_TAG:
        ParametersMacro
            parameters_macro;
};
```

```
// Corresponding MHEG datatype: Set-Parameters
//=========================================================================
struct SetParameters {
    sequence<RtTargetsParameter,1>
        rt_script_targets_parameter;
    ParametersParameter
        parameters;
};

// Corresponding MHEG datatype: Plug-In
//=========================================================================
enum PlugInTag { RT_COMPONENT_REFERENCE_TAG, COMPONENT_REFERENCE_TAG, LABEL_TAG
};
union PlugIn
switch (PlugInTag){
    case RT_COMPONENT_REFERENCE_TAG:
        RtObjectReference
            rt_component_reference;
    case COMPONENT_REFERENCE_TAG:
        MhObjectReference
            component_reference;
    case LABEL_TAG:
        GenericString
            label;
};

// Corresponding MHEG datatype: Plug-In-Macro
//=========================================================================
struct PlugInMacro {
    MacroDefId
        macro_def_id;
    sequence<PlugIn,1>
        plug_in;
};

// Corresponding MHEG datatype: Plug-In-Parameter
//=========================================================================
enum PlugInParameterTag { PLUG_IN_TAG, PLUG_IN_MACRO_TAG };
union PlugInParameter
switch (PlugInParameterTag){
    case PLUG_IN_TAG:
        PlugIn
            plug_in;
    case PLUG_IN_MACRO_TAG:
        PlugInMacro
            plug_in_macro;
};

// Corresponding MHEG datatype: Plug
//=========================================================================
struct Plug {
    sequence<SocketTargetsParameter,1>
        socket_targets_parameter;
    PlugInParameter
        plug_in_parameter;
};

// Corresponding MHEG datatype: Set-RGS
//=========================================================================
struct SetRGS {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_param;
    sequence<ChannelTargetParameter,1>
        rgs_parameter;
};

// Corresponding MHEG datatype: Set-Opacity
```

```
//=========================================================================
struct SetOpacity {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_param;
    sequence<NumericParameter,1>
        opacity_rate;
    sequence<NumericParameter,1>
        transition_duration_param;
};

// Corresponding MHEG datatype: Presentation-Priority
//=========================================================================
enum PresentationPriorityTag { PRESENTATION_PRIORITY_GENERIC_NUMERIC_TAG,
PRESENTATION_PRIORITY_PRIORITY_LEVEL_TAG };
union PresentationPriority
switch (PresentationPriorityTag){
    case PRESENTATION_PRIORITY_GENERIC_NUMERIC_TAG:
        GenericNumeric
            generic_numeric;
    case PRESENTATION_PRIORITY_PRIORITY_LEVEL_TAG:
        PriorityLevel
            priority_level;
};

// Corresponding MHEG datatype: Presentation-Priority-Macro
//=========================================================================
struct PresentationPriorityMacro {
    MacroDefId
        macro_def_id;
    sequence<PresentationPriority,1>
        presentation_priority;
};

// Corresponding MHEG datatype: Presentation-Priority-Parameter
//=========================================================================
enum PresentationPriorityParameterTag { PRESENTATION_PRIORITY_TAG,
PRESENTATION_PRIORITY_MACRO_TAG };
union PresentationPriorityParameter
switch (PresentationPriorityParameterTag){
    case PRESENTATION_PRIORITY_TAG:
        PresentationPriority
            presentation_priority;
    case PRESENTATION_PRIORITY_MACRO_TAG:
        PresentationPriorityMacro
            presentation_priority_macro;
};

// Corresponding MHEG datatype: Set-Presentation-Priority
//=========================================================================
struct SetPresentationPriority {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_param;
    sequence<PresentationPriorityParameter,1>
        presentation_priority_param;
    sequence<NumericParameter,1>
        transition_duration_param;
};

// Corresponding MHEG datatype: Temporal-Position-Macro
//=========================================================================
struct TemporalPositionMacro {
    MacroDefId
        macro_def_id;
    sequence<TemporalPosition,1>
        temporal_position;
};

// Corresponding MHEG datatype: Temporal-Position-Parameter
//=========================================================================
enum TemporalPositionParameterTag { TEMPORAL_POSITION_TAG,
TEMPORAL_POSITION_MACRO_TAG };
union TemporalPositionParameter
```

```
switch (TemporalPositionParameterTag){
    case TEMPORAL_POSITION_TAG:
        TemporalPosition
            temporal_position;
    case TEMPORAL_POSITION_MACRO_TAG:
        TemporalPositionMacro
            temporal_position_macro;
};

// Corresponding MHEG datatype: Set-Visible-Duration
//=========================================================================
struct SetVisibleDuration {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    sequence<TemporalPositionParameter,1>
        initial_temporal_position_parameter;
    sequence<TemporalPositionParameter,1>
        terminal_temporal_position_parameter;
};

// Corresponding MHEG datatype: Temporal-Termination-Macro
//=========================================================================
struct TemporalTerminationMacro {
    MacroDefId
        macro_def_id;
    sequence<TemporalTermination,1>
        temporal_termination;
};

// Corresponding MHEG datatype: Temporal-Termination-Parameter
//=========================================================================
enum TemporalTerminationParameterTag {
TEMPORAL_TERMINATION_PARAMETER_TEMPORAL_TERMINATION_TAG,
TEMPORAL_TERMINATION_PARAMETER_TEMPORAL_TERMINATION_MACRO_TAG };
union TemporalTerminationParameter
switch (TemporalTerminationParameterTag){
    case TEMPORAL_TERMINATION_PARAMETER_TEMPORAL_TERMINATION_TAG:
        TemporalTermination
            temporal_termination;
    case TEMPORAL_TERMINATION_PARAMETER_TEMPORAL_TERMINATION_MACRO_TAG:
        TemporalTerminationMacro
            temporal_termination_macro;
};

// Corresponding MHEG datatype: Set-Temporal-Termination
//=========================================================================
struct SetTemporalTermination {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    TemporalTerminationParameter
        temporal_termination_parameter;
};

// Corresponding MHEG datatype: Current-Temporal-Position-Macro
//=========================================================================
struct CurrentTemporalPositionMacro {
    MacroDefId
        macro_def_id;
    sequence<CurrentTemporalPosition,1>
        current_temporal_termination;
};

// Corresponding MHEG datatype: Current-Temporal-Position-Parameter
//=========================================================================
enum CurrentTemporalPositionParameterTag { CURRENT_TEMPORAL_POSITION_TAG,
CURRENT_TEMPORAL_POSITION_MACRO_TAG };
union CurrentTemporalPositionParameter
switch (CurrentTemporalPositionParameterTag){
    case CURRENT_TEMPORAL_POSITION_TAG:
        CurrentTemporalPosition
            current_temporal_position;
    case CURRENT_TEMPORAL_POSITION_MACRO_TAG:
        CurrentTemporalPositionMacro
            current_temporal_position_macro;
};
```

```
// Corresponding MHEG datatype: Set-Current-Temporal-Position
//=========================================================================
struct SetCurrentTemporalPosition {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    sequence<CurrentTemporalPositionParameter,1>
        temporal_position_parameter;
};

// Corresponding MHEG datatype: Visible-Duration-Macro
//=========================================================================
struct VisibleDurationPositionMacro {
    MacroDefId
        macro_def_id;
    sequence<VisibleDurationPosition,1>
        visible_duration_position;
};

// Corresponding MHEG datatype: Visible-Duration-Parameter
//=========================================================================
enum VisibleDurationPositionParameterTag { VISIBLE_DURATION_POSITION_TAG,
VISIBLE_DURATION_POSITION_MACRO_TAG };
union VisibleDurationPositionParameter
switch (VisibleDurationPositionParameterTag){
    case VISIBLE_DURATION_POSITION_TAG:
        VisibleDurationPosition
            visible_duration_position;
    case VISIBLE_DURATION_POSITION_MACRO_TAG:
        VisibleDurationPositionMacro
            visible_duration_position_macro;
};

// Corresponding MHEG datatype: Set-Visible-Duration-Position
//=========================================================================
struct SetVisibleDurationPosition {
    sequence<SocketTargetsParameter,1>
        socket_targets;
    sequence<VisibleDurationPositionParameter,1>
        visible_duration_position_param;
};

// Corresponding MHEG datatype: Speed-Macro
//=========================================================================
struct SpeedMacro {
    MacroDefId
        macro_def_id;
    sequence<Speed,1>
        speed;
};

// Corresponding MHEG datatype: Speed-Parameter
//=========================================================================
enum SpeedParameterTag { SPEED_TAG, SPEED_MACRO_TAG };
union SpeedParameter
switch (SpeedParameterTag){
    case SPEED_TAG:
        Speed
            speed;
    case SPEED_MACRO_TAG:
        SpeedMacro
            speed_macro;
};

// Corresponding MHEG datatype: Set-Speed
//=========================================================================
struct SetSpeed {
    RtSocketTargetsParameter
        perceptible_targets_parameter;
    sequence<SpeedParameter,1>
        speed_parameter;
    sequence<NumericParameter,1>
        transition_duration_parameter;
};
```

```
// Corresponding MHEG datatype: Timestone-Macro
//=========================================================================
struct TimestoneMacro {
    MacroDefId
        macro_def_id;
    sequence<Timestone,1>
        timestone;
};

// Corresponding MHEG datatype: Timestone-Parameter
//=========================================================================
enum TimestoneParameterTag { TIMESTONE_TAG, TIMESTONE_MACRO_TAG };
union TimestoneParameter
switch (TimestoneParameterTag){
    case TIMESTONE_TAG:
        Timestone
            timestone;
    case TIMESTONE_MACRO_TAG:
        TimestoneMacro
            timestone_macro;
};

// Corresponding MHEG datatype: Set-Timestones
//=========================================================================
struct SetTimestones {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    sequence<sequence<TimestoneParameter>,1>
        timestones_parameter;
};

// Corresponding MHEG datatype: Perceptible-Size-Projection-Macro
//=========================================================================
struct PerceptibleSizeProjectionMacro {
    MacroDefId
        macro_def_id;
    sequence<PerceptibleSizeProjection,1>
        perceptible_size_projection;
};

// Corresponding MHEG datatype: Perceptible-Size-Projection-Parameter
//=========================================================================
enum PerceptibleSizeProjectionParameterTag { PERCEPTIBLE_SIZE_PROJECTION_TAG,
PERCEPTIBLE_SIZE_PROJECTION_MACRO_TAG };
union PerceptibleSizeProjectionParameter
switch (PerceptibleSizeProjectionParameterTag){
    case PERCEPTIBLE_SIZE_PROJECTION_TAG:
        PerceptibleSizeProjection
            perceptible_size_projection;
    case PERCEPTIBLE_SIZE_PROJECTION_MACRO_TAG:
        PerceptibleSizeProjectionMacro
            perceptible_size_projection_macro;
};

// Corresponding MHEG datatype: Set-Perceptible-Size-Projection
//=========================================================================
struct SetPerceptibleSizeProjection {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    PerceptibleSizeProjectionParameter
        perceptible_size_projection;
    sequence<NumericParameter,1>
        transition_duration_parameter;
};

// Corresponding MHEG datatype: Resizing-Strategy-Macro
//=========================================================================
struct ResizingStrategyMacro {
    MacroDefId
        macro_def_id;
    sequence<ResizingStrategy,1>
        resizing_strategy;
};
```

```
// Corresponding MHEG datatype: Resizing-Strategy-Parameter
//=========================================================================
enum ResizingStrategyParameterTag {
RESIZING_STRATEGY_PARAMETER_RESIZING_STRATEGY_TAG,
RESIZING_STRATEGY_PARAMETER_RESIZING_STRATEGY_MACRO_TAG };
union ResizingStrategyParameter
switch (ResizingStrategyParameterTag){
    case RESIZING_STRATEGY_PARAMETER_RESIZING_STRATEGY_TAG:
        ResizingStrategy
            resizing_strategy;
    case RESIZING_STRATEGY_PARAMETER_RESIZING_STRATEGY_MACRO_TAG:
        ResizingStrategyMacro
            resizing_strategy_macro;
};

// Corresponding MHEG datatype: Set-Resizing-Strategy
//=========================================================================
struct SetResizingStrategy {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    sequence<ResizingStrategyParameter,1>
        resizing_strategy_parameter;
};

// Corresponding MHEG datatype: Set-Aspect-Ratio-Preserved
//=========================================================================
struct SetAspectRatioPreserved {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    BooleanParameter
        preserved;
};

// Corresponding MHEG datatype: VSGS-Macro
//=========================================================================
struct VSGSMacro {
    MacroDefId
        macro_def_id;
    sequence<VSGS,1>
        vsgs;
};

// Corresponding MHEG datatype: VSGS-Parameter
//=========================================================================
enum VSGSParameterTag { VSGS_PARAMETER_VSGS_TAG, VSGS_PARAMETER_VSGS_MACRO_TAG };
union VSGSParameter
switch (VSGSParameterTag){
    case VSGS_PARAMETER_VSGS_TAG:
        VSGS
            vsgs;
    case VSGS_PARAMETER_VSGS_MACRO_TAG:
        VSGSMacro
            vsgs_macro;
};

// Corresponding MHEG datatype: Size-Attribute-Macro
//=========================================================================
struct SizeAttributeMacro {
    MacroDefId
        macro_def_id;
    sequence<SizeAttribute,1>
        size_attribute;
};

// Corresponding MHEG datatype: Size-Attribute-Parameter
//=========================================================================
enum SizeAttributeParameterTag { SIZE_ATTRIBUTE_TAG, SIZE_ATTRIBUTE_MACRO_TAG };
union SizeAttributeParameter
switch (SizeAttributeParameterTag){
    case SIZE_ATTRIBUTE_TAG:
        SizeAttribute
            size_attribute;
```

```
    case SIZE_ATTRIBUTE_MACRO_TAG:
        SizeAttributeMacro
            size_attribute_macro;
};

// Corresponding MHEG datatype: VS-Parameter
//=========================================================================
enum VSParameterTag { X_SIZE_ATTRIBUTE_PARAMETER_TAG,
Y_SIZE_ATTRIBUTE_PARAMETER_TAG, Z_SIZE_ATTRIBUTE_PARAMETER_TAG };
union VSParameter
switch (VSParameterTag){
    case X_SIZE_ATTRIBUTE_PARAMETER_TAG:
        SizeAttributeParameter
            x_size_attribute_parameter;
    case Y_SIZE_ATTRIBUTE_PARAMETER_TAG:
        SizeAttributeParameter
            y_size_attribute_parameter;
    case Z_SIZE_ATTRIBUTE_PARAMETER_TAG:
        SizeAttributeParameter
            z_size_attribute_parameter;
};

// Corresponding MHEG datatype: Set-Visible-Size
//=========================================================================
struct SetVisibleSize {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    VSGSParameter
        vsgs_parameter;
    VSParameter
        vs_parameter;
    sequence<NumericParameter,1>
        transition_duration_parameter;
};

// Corresponding MHEG datatype: Adjustment-Axis-Macro
//=========================================================================
struct AdjustmentAxisMacro {
    MacroDefId
        macro_def_id;
    sequence<AdjustmentAxis,1>
        adjustment_axis;
};

// Corresponding MHEG datatype: Adjustment-Axis-Parameter
//=========================================================================
enum AdjustmentAxisParameterTag { ADJUSTMENT_AXIS_TAG, ADJUSTMENT_AXIS_MACRO_TAG
};
union AdjustmentAxisParameter
switch (AdjustmentAxisParameterTag){
    case ADJUSTMENT_AXIS_TAG:
        AdjustmentAxis
            adjustment_axis;
    case ADJUSTMENT_AXIS_MACRO_TAG:
        AdjustmentAxisMacro
            adjustment_axis_macro;
};

// Corresponding MHEG datatype: Adjustment-Policy-Macro
//=========================================================================
struct AdjustmentPolicyMacro {
    MacroDefId
        macro_def_id;
    sequence<AdjustmentPolicy,1>
        adjustment_policy;
};

// Corresponding MHEG datatype: Adjustment-Policy-Parameter
//=========================================================================
enum AdjustmentPolicyParameterTag { ADJUSTMENT_POLICY_TAG,
ADJUSTMENT_POLICY_MACRO_TAG };
union AdjustmentPolicyParameter
switch (AdjustmentPolicyParameterTag){
```

```
    case ADJUSTMENT_POLICY_TAG:
        AdjustmentPolicy
            adjustment_policy;
    case ADJUSTMENT_POLICY_MACRO_TAG:
        AdjustmentPolicyMacro
            adjustment_policy_macro;
};

// Corresponding MHEG datatype: Set-Visible-Sizes-Adjustment
//=========================================================================
struct SetVisibleSizesAdjustment {
    sequence<AdjustmentAxisParameter>
        adjustment_axis_set;
    AdjustmentPolicyParameter
        adjustment_policy;
    sequence<NumericParameter,1>
        transition_duration_parameter;
};

// Corresponding MHEG datatype: Set-Box
//=========================================================================
struct SetBox {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    sequence<BooleanParameter,1>
        box_parameter;
};

// Corresponding MHEG datatype: Set-Default-Background
//=========================================================================
struct SetDefaultBackground {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    sequence<NumericParameter,1>
        background_parameter;
    sequence<NumericParameter,1>
        transition_duration;
};

// Corresponding MHEG datatype: Attachment-Point-Type-Macro
//=========================================================================
struct AttachmentPointTypeMacro {
    MacroDefId
        macro_def_id;
    sequence<AttachmentPointType,1>
        attachment_point_type;
};

// Corresponding MHEG datatype: Attachment-Point-Type-Parameter
//=========================================================================
enum AttachmentPointTypeParameterTag { ATTACHMENT_POINT_TYPE_TAG,
ATTACHMENT_POINT_TYPE_MACRO_TAG };
union AttachmentPointTypeParameter
switch (AttachmentPointTypeParameterTag){
    case ATTACHMENT_POINT_TYPE_TAG:
        AttachmentPointType
            attachment_point_type;
    case ATTACHMENT_POINT_TYPE_MACRO_TAG:
        AttachmentPointTypeMacro
            attachment_point_type_macro;
};

// Corresponding MHEG datatype: Attachment-Attribute-Macro
//=========================================================================
struct AttachmentAttributeMacro {
    MacroDefId
        macro_def_id;
    sequence<AttachmentAttribute,1>
        attachment_attribute;
};
```

```
// Corresponding MHEG datatype: Attachment-Attribute-Parameter
//=========================================================================
enum AttachmentAttributeParameterTag { ATTACHMENT_ATTRIBUTE_TAG,
ATTACHMENT_ATTRIBUTE_MACRO_TAG };
union AttachmentAttributeParameter
switch (AttachmentAttributeParameterTag){
    case ATTACHMENT_ATTRIBUTE_TAG:
        AttachmentAttribute
            attachment_attribute;
    case ATTACHMENT_ATTRIBUTE_MACRO_TAG:
        AttachmentAttributeMacro
            attachment_attribute_macro;
};

// Corresponding MHEG datatype: Attachment-Point-Parameter
//=========================================================================
struct AttachmentPointParameter {
    sequence<AttachmentAttributeParameter,1>
        x_attachment_parameter;
    sequence<AttachmentAttributeParameter,1>
        y_attachment_parameter;
    sequence<AttachmentAttributeParameter,1>
        z_attachment_parameter;
};

// Corresponding MHEG datatype: Set-Attachment-Point
//=========================================================================
struct SetAttachmentPoint {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    sequence<AttachmentPointTypeParameter,1>
        attachment_point_type_param;
    sequence<AttachmentPointParameter,1>
        attachment_point_param;
};

// Corresponding MHEG datatype: Reference-Type-Macro
//=========================================================================
struct ReferenceTypeMacro {
    MacroDefId
        macro_def_id;
    sequence<ReferenceType,1>
        reference_type;
};

// Corresponding MHEG datatype: Reference-Type-Parameter
//=========================================================================
enum ReferenceTypeParameterTag { REFERENCE_TYPE_TAG, REFERENCE_TYPE_MACRO_TAG };
union ReferenceTypeParameter
switch (ReferenceTypeParameterTag){
    case REFERENCE_TYPE_TAG:
        ReferenceType
            reference_type;
    case REFERENCE_TYPE_MACRO_TAG:
        ReferenceTypeMacro
            reference_type_macro;
};

// Corresponding MHEG datatype: VSEAP-Reference-Point
//=========================================================================
enum VSEAPReferencePointTag { ORIGIN_TAG, COMPONENT_TAG };
union VSEAPReferencePoint
switch (VSEAPReferencePointTag){
    case COMPONENT_TAG:
        SocketTargetParameter
            component;
};

// Corresponding MHEG datatype: VSEAP-Reference-Point-Macro
//=========================================================================
struct VSEAPReferencePointMacro {
    MacroDefId
        macro_def_id;
```

```
    sequence<VSEAPReferencePoint,1>
        vseap_reference;
};

// Corresponding MHEG datatype: VSEAP-Reference-Parameter
//=========================================================================
enum VSEAPReferenceParameterTag { VSEAP_REFERENCE_TAG, VSEAP_REFERENCE_MACRO_TAG
};
union VSEAPReferenceParameter
switch (VSEAPReferenceParameterTag){
    case VSEAP_REFERENCE_TAG:
        VSEAPReferencePoint
            vseap_reference;
    case VSEAP_REFERENCE_MACRO_TAG:
        VSEAPReferencePointMacro
            vseap_reference_macro;
};

// Corresponding MHEG datatype: Length-Attribute-Macro
//=========================================================================
struct LengthAttributeMacro {
    MacroDefId
        macro_def_id;
    sequence<LengthAttribute,1>
        length_attribute;
};

// Corresponding MHEG datatype: Length-Attribute-Parameter
//=========================================================================
enum LengthAttributeParameterTag { LENGTH_ATTRIBUTE_TAG,
LENGTH_ATTRIBUTE_MACRO_TAG };
union LengthAttributeParameter
switch (LengthAttributeParameterTag){
    case LENGTH_ATTRIBUTE_TAG:
        LengthAttribute
            length_attribute;
    case LENGTH_ATTRIBUTE_MACRO_TAG:
        LengthAttributeMacro
            length_attribute_macro;
};

// Corresponding MHEG datatype: Lengths-Parameter
//=========================================================================
struct LengthsParameter {
    sequence<LengthAttributeParameter,1>
        x_length_parameter;
    sequence<LengthAttributeParameter,1>
        y_position_parameter;
    sequence<LengthAttributeParameter,1>
        z_position_parameter;
};

// Corresponding MHEG datatype: Set-Attachment-Point-Position
//=========================================================================
struct SetAttachmentPointPosition {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    sequence<ReferenceTypeParameter,1>
        attachment_type_parameter;
    sequence<VSEAPReferenceParameter,1>
        vseap_reference_parameter;
    sequence<LengthsParameter,1>
        positions_parameter;
    sequence<NumericParameter,1>
        transition_duration_parameter;
};

// Corresponding MHEG datatype: Size-Border-Macro
//=========================================================================
struct SizeBorderMacro {
    MacroDefId
        macro_def_id;
    sequence<SizeBorder,1>
        size_border;
};
```

```
// Corresponding MHEG datatype: Size-Border-Parameter
//=========================================================================
enum SizeBorderParameterTag { SIZE_BORDER_TAG, SIZE_BORDER_MACRO_TAG };
union SizeBorderParameter
switch (SizeBorderParameterTag){
    case SIZE_BORDER_TAG:
        SizeBorder
            size_border;
    case SIZE_BORDER_MACRO_TAG:
        SizeBorderMacro
            size_border_macro;
};

// Corresponding MHEG datatype: Set-Visible-Sizes-Alignment
//=========================================================================
struct SetVisibleSizesAlignment {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    sequence<SizeBorderParameter,1>
        size_border_parameter;
    sequence<NumericParameter,1>
        margin_parameter;
    sequence<NumericParameter,1>
        transition_duration;
};

// Corresponding MHEG datatype: User-Controls-Macro
//=========================================================================
struct UserControlsMacro {
    MacroDefId
        macro_def_id;
    sequence<UserControls,1>
        user_controls;
};

// Corresponding MHEG datatype: User-Controls-Parameter
//=========================================================================
enum UserControlsParameterTag { USER_CONTROLS_PARAMETER_USER_CONTROLS_TAG,
USER_CONTROLS_PARAMETER_USER_CONTROLS_MACRO_TAG };
union UserControlsParameter
switch (UserControlsParameterTag){
    case USER_CONTROLS_PARAMETER_USER_CONTROLS_TAG:
        UserControls
            user_controls;
    case USER_CONTROLS_PARAMETER_USER_CONTROLS_MACRO_TAG:
        UserControlsMacro
            user_controls_macro;
};

// Corresponding MHEG datatype: Set-Moving-Ability
//=========================================================================
struct SetMovingAbility {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    sequence<UserControlsParameter,1>
        allowed;
};

// Corresponding MHEG datatype: Set-Resizing-Ability
//=========================================================================
struct SetResizingAbility {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    sequence<UserControlsParameter,1>
        allowed;
};

// Corresponding MHEG datatype: Set-Scaling-Ability
//=========================================================================
struct SetScalingAbility {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    sequence<UserControlsParameter,1>
        allowed;
};
```

```
// Corresponding MHEG datatype: Set-Scrolling-Ability
//=========================================================================
struct SetScrollingAbility {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    sequence<UserControlsParameter,1>
        allowed;
};

// Corresponding MHEG datatype: Audible-Volume-Macro
//=========================================================================
struct AudibleVolumeMacro {
    MacroDefId
        macro_def_id;
    sequence<AudibleVolume,1>
        audible_volume;
};

// Corresponding MHEG datatype: Audible-Volume-Parameter
//=========================================================================
enum AudibleVolumeParameterTag { AUDIBLE_VOLUME_TAG, AUDIBLE_VOLUME_MACRO_TAG };
union AudibleVolumeParameter
switch (AudibleVolumeParameterTag){
    case AUDIBLE_VOLUME_TAG:
        AudibleVolume
            audible_volume;
    case AUDIBLE_VOLUME_MACRO_TAG:
        AudibleVolumeMacro
            audible_volume_macro;
};

// Corresponding MHEG datatype: Set-Audible-Volume
//=========================================================================
struct SetAudibleVolume {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    sequence<AudibleVolumeParameter,1>
        audible_volume_parameter;
    sequence<NumericParameter,1>
        transition_duration_parameter;
};

// Corresponding MHEG datatype: Set-Audible-Composition-Effect
//=========================================================================
struct SetAudibleCompositionEffect {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    sequence<NumericParameter,1>
        audible_effect_parameter;
    sequence<NumericParameter,1>
        transition_duration_parameter;
};

// Corresponding MHEG datatype: Stream-Identifier-Macro
//=========================================================================
struct StreamIdentifierMacro {
    MacroDefId
        macro_def_id;
    sequence<StreamIdentifier,1>
        stream_identifier;
};

// Corresponding MHEG datatype: Stream-Identifier-Parameter
//=========================================================================
enum StreamIdentifierParameterTag { STREAM_IDENTIFIER_TAG,
STREAM_IDENTIFIER_MACRO_TAG };
union StreamIdentifierParameter
switch (StreamIdentifierParameterTag){
    case STREAM_IDENTIFIER_TAG:
        StreamIdentifier
            stream_identifier;
    case STREAM_IDENTIFIER_MACRO_TAG:
        StreamIdentifierMacro
            stream_identifier_macro;
};
```

```
// Corresponding MHEG datatype: Set-Stream-Choice
//=========================================================================
struct SetStreamChoice {
    RtSocketTargetsParameter
        rt_multiplexed_content_socket_param;
    sequence<StreamIdentifierParameter,1>
        stream_identifier_parameter;
};

// Corresponding MHEG datatype: Set-Selectability
//=========================================================================
struct SetSelectability {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    sequence<NumericParameter,1>
        min_number_selections;
    sequence<NumericParameter,1>
        max_number_selections;
};

// Corresponding MHEG datatype: Selection-State-Macro
//=========================================================================
struct SelectionStateMacro {
    MacroDefId
        macro_def_id;
    sequence<SelectionStatusValue,1>
        modification_state;
};

// Corresponding MHEG datatype: Selection-State-Parameter
//=========================================================================
enum SelectionStateParameterTag { SELECTION_STATE_MACRO_TAG, SELECTION_STATE_TAG
};
union SelectionStateParameter
switch (SelectionStateParameterTag){
    case SELECTION_STATE_MACRO_TAG:
        SelectionStateMacro
            selection_state_macro;
    case SELECTION_STATE_TAG:
        SelectionStatusValue
            selection_state;
};

// Corresponding MHEG datatype: Set-Selection-Status
//=========================================================================
struct SetSelectionStatus {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    sequence<SelectionStateParameter,1>
        selection_state_parameter;
};

// Corresponding MHEG datatype: Responsibility-Macro
//=========================================================================
struct ResponsibilityMacro {
    MacroDefId
        macro_def_id;
    sequence<Responsibility,1>
        responsibility;
};

// Corresponding MHEG datatype: Responsibility-Parameter
//=========================================================================
enum ResponsibilityParameterTag { RESPONSIBILITY_PARAMETER_RESPONSIBILITY_TAG,
RESPONSIBILITY_PARAMETER_RESPONSIBILITY_MACRO_TAG };
union ResponsibilityParameter
switch (ResponsibilityParameterTag){
    case RESPONSIBILITY_PARAMETER_RESPONSIBILITY_TAG:
        Responsibility
            responsibility;
    case RESPONSIBILITY_PARAMETER_RESPONSIBILITY_MACRO_TAG:
        ResponsibilityMacro
            responsibility_macro;
};
```

```
// Corresponding MHEG datatype: Set-Selection-Presentation-Effect-Responsibility
//=========================================================================
struct SetSelectionPresentationEffectResponsibility {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    sequence<ResponsibilityParameter,1>
        selection_responsibility;
};

// Corresponding MHEG datatype: Set-Modifiability
//=========================================================================
struct SetModifiability {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    sequence<NumericParameter,1>
        min_number_modifications;
    sequence<NumericParameter,1>
        max_number_modifications;
};

// Corresponding MHEG datatype: Modification-State-Macro
//=========================================================================
struct ModificationStateMacro {
    MacroDefId
        macro_def_id;
    sequence<ModificationStatusValue,1>
        modification_state;
};

// Corresponding MHEG datatype: Modification-State-Parameter
//=========================================================================
enum ModificationStateParameterTag { MODIFICATION_STATE_MACRO_TAG,
MODIFICATION_STATE_TAG };
union ModificationStateParameter
switch (ModificationStateParameterTag){
    case MODIFICATION_STATE_MACRO_TAG:
        ModificationStateMacro
            modification_state_macro;
    case MODIFICATION_STATE_TAG:
        ModificationStatusValue
            modification_state;
};

// Corresponding MHEG datatype: Set-Modification-Status
//=========================================================================
struct SetModificationStatus {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    sequence<ModificationStateParameter,1>
        modification_state_parameter;
};

// Corresponding MHEG datatype: Set-Modification-Presentation-Effect-
//   Responsibility
//=========================================================================
struct SetModificationPresentationEffectResponsibility {
    sequence<RtSocketTargetsParameter,1>
        perceptible_targets_parameter;
    sequence<ResponsibilityParameter,1>
        modification_responsibility;
};

// Corresponding MHEG datatype: Presentation-State-Macro
//=========================================================================
struct PresentationStateMacro {
    MacroDefId
        macro_def_id;
    sequence<PresentationState,1>
        presentation_state;
};
```

```
// Corresponding MHEG datatype: Presentation-State-Parameter
//=========================================================================
enum PresentationStateParameterTag { PRESENTATION_STATE_MACRO_TAG,
PRESENTATION_STATE_TAG };
union PresentationStateParameter
switch (PresentationStateParameterTag){
    case PRESENTATION_STATE_MACRO_TAG:
        PresentationStateMacro
            presentation_state_macro;
    case PRESENTATION_STATE_TAG:
        PresentationState
            presentation_state;
};

// Corresponding MHEG datatype: Alternate-Presentation-State-Macro
//=========================================================================
struct AlternatePresentationMacro {
    MacroDefId
        macro_def_id;
    sequence<AlternatePresentation,1>
        alternate_presentation_state;
};

// Corresponding MHEG datatype: Alternate-Presentation-State-Parameter
//=========================================================================
enum AlternatePresentationParameterTag { ALTERNATE_PRESENTATION_MACRO_TAG,
ALTERNATE_PRESENTATION_TAG };
union AlternatePresentationParameter
switch (AlternatePresentationParameterTag){
    case ALTERNATE_PRESENTATION_MACRO_TAG:
        AlternatePresentationMacro
            alternate_presentation_macro;
    case ALTERNATE_PRESENTATION_TAG:
        AlternatePresentation
            alternate_presentation;
};

// Corresponding MHEG datatype: Set-Button-Style
//=========================================================================
struct SetButtonStyle {
    sequence<RtSocketTargetsParameter,1>
        targets;
    sequence<PresentationStateParameter,1>
        initial_state;
    sequence<AlternatePresentationParameter,1>
        alternate_presentation1;
    sequence<AlternatePresentationParameter,1>
        alternate_presentation2;
    sequence<AlternatePresentationParameter,1>
        alternate_presentation3;
};

// Corresponding MHEG datatype: Orientation-Macro
//=========================================================================
struct OrientationMacro {
    MacroDefId
        macro_def_id;
    sequence<Orientation,1>
        orientation;
};

// Corresponding MHEG datatype: Orientation-Parameter
//=========================================================================
enum OrientationParameterTag { ORIENTATION_TAG, ORIENTATION_MACRO_TAG };
union OrientationParameter
switch (OrientationParameterTag){
    case ORIENTATION_TAG:
        Orientation
            orientation;
    case ORIENTATION_MACRO_TAG:
        OrientationMacro
            orientation_macro;
};
```

```
// Corresponding MHEG datatype: Set-Slider-Style
//=========================================================================
struct SetSliderStyle {
    sequence<RtSocketTargetsParameter,1>
        targets;
    sequence<RtSocketTargetParameter,1>
        cursor;
    sequence<RtSocketTargetParameter,1>
        background;
    sequence<OrientationParameter,1>
        orientation;
    sequence<NumericParameter,1>
        min_value;
    NumericParameter
        max_value;
};

// Corresponding MHEG datatype: Echo-Style-Macro
//=========================================================================
struct EchoStyleMacro {
    MacroDefId
        macro_def_id;
    sequence<EchoStyle,1>
        echo_style;
};

// Corresponding MHEG datatype: Echo-Style-Parameter
//=========================================================================
enum EchoStyleParameterTag { ECHO_STYLE_MACRO_TAG, ECHO_STYLE_TAG };
union EchoStyleParameter
switch (EchoStyleParameterTag){
    case ECHO_STYLE_MACRO_TAG:
        EchoStyleMacro
            echo_style_macro;
    case ECHO_STYLE_TAG:
        EchoStyle
            echo_style;
};

// Corresponding MHEG datatype: Set-Entry-Field-Style
//=========================================================================
struct SetEntryFieldStyle {
    sequence<RtSocketTargetsParameter,1>
        targets;
    sequence<EchoStyleParameter,1>
        echo;
    sequence<RtSocketTargetParameter,1>
        background;
};

// Corresponding MHEG datatype: Association-Macro
//=========================================================================
struct AssociationMacro {
    MacroDefId
        macro_def_id;
    sequence<Association,1>
        association;
};

// Corresponding MHEG datatype: Association-Parameter
//=========================================================================
enum AssociationParameterTag { ASSOCIATION_TAG, ASSOCIATION_MACRO_TAG };
union AssociationParameter
switch (AssociationParameterTag){
    case ASSOCIATION_TAG:
        Association
            association;
    case ASSOCIATION_MACRO_TAG:
        AssociationMacro
            association_macro;
};
```

```
// Corresponding MHEG datatype: Set-Menu-Style
//=========================================================================
struct SetMenuStyle {
    sequence<RtSocketTargetsParameter,1>
        targets;
    sequence<OrientationParameter,1>
        upper_menu_orientation;
    sequence<AssociationParameter>
        associations;
};

// Corresponding MHEG datatype: Socket-Tail-Macro
//=========================================================================
struct SocketTailMacro {
    MacroDefId
        macro_def_id;
    sequence<SocketTail,1>
        socket_tail;
};

// Corresponding MHEG datatype: Socket-Tail-Parameter
//=========================================================================
enum SocketTailParameterTag { SOCKET_TAIL_TAG, SOCKET_TAIL_MACRO_TAG };
union SocketTailParameter
switch (SocketTailParameterTag){
    case SOCKET_TAIL_TAG:
        SocketTail
            socket_tail;
    case SOCKET_TAIL_MACRO_TAG:
        SocketTailMacro
            socket_tail_macro;
};

// Corresponding MHEG datatype: Separator-Macro
//=========================================================================
struct SeparatorMacro {
    MacroDefId
        macro_def_id;
    sequence<Separator,1>
        separator;
};

// Corresponding MHEG datatype: Separator-Parameter
//=========================================================================
enum SeparatorParameterTag { SEPARATOR_MACRO_TAG, SEPARATOR_TAG };
union SeparatorParameter
switch (SeparatorParameterTag){
    case SEPARATOR_MACRO_TAG:
        SeparatorMacro
            separator_macro;
    case SEPARATOR_TAG:
        Separator
            separator;
};

// Corresponding MHEG datatype: Slider-Side-Macro
//=========================================================================
struct SliderSideMacro {
    MacroDefId
        macro_def_id;
    sequence<SliderSide,1>
        slider_side;
};

// Corresponding MHEG datatype: Slider-Side-Parameter
//=========================================================================
enum SliderSideParameterTag { SLIDER_SIDE_TAG, SLIDER_SIDE_MACRO_TAG };
union SliderSideParameter
switch (SliderSideParameterTag){
    case SLIDER_SIDE_TAG:
        SliderSide
            slider_side;
```

```
    case SLIDER_SIDE_MACRO_TAG:
        SliderSideMacro
            slider_side_macro;
};

// Corresponding MHEG datatype: Set-Slider-Style-Macro
//=========================================================================
struct SetSliderStyleMacro {
    MacroDefId
        macro_def_id;
    sequence<SetSliderStyle,1>
        set_slider_style;
};

// Corresponding MHEG datatype: Set-Slider-Style-Parameter
//=========================================================================
enum SetSliderStyleParameterTag { SET_SLIDER_STYLE_MACRO_TAG,
SET_SLIDER_STYLE_ACTION_TAG };
union SetSliderStyleParameter
switch (SetSliderStyleParameterTag){
    case SET_SLIDER_STYLE_MACRO_TAG:
        SetSliderStyleMacro
            set_slider_style_macro;
    case SET_SLIDER_STYLE_ACTION_TAG:
        SetSliderStyle
            set_slider_style_action;
};

// Corresponding MHEG datatype: Set-Scrolling-List-Style
//=========================================================================
struct SetScrollingListStyle {
    sequence<RtSocketTargetsParameter,1>
        targets;
    sequence<RtSocketTargetParameter,1>
        background;
    sequence<NumericParameter,1>
        visible_items_number;
    sequence<SocketTailParameter,1>
        first_item;
    sequence<SeparatorParameter,1>
        separator;
    sequence<OrientationParameter,1>
        orientation;
    sequence<SliderSideParameter,1>
        slider_side;
    sequence<SetSliderStyleParameter,1>
        set_slider_style_action;
};

// Corresponding MHEG datatype: Original-Def-Declaration-Macro
//=========================================================================
struct OriginalDefDeclarationMacro {
    MacroDefId
        macro_def_id;
    sequence<OriginalDefDeclaration,1>
        original_definition;
};

// Corresponding MHEG datatype: Original-Def-Declaration-Parameter
//=========================================================================
enum OriginalDefDeclarationParameterTag { ORIGINAL_DEFINITION_TAG,
ORIGINAL_DEFINITION_MACRO_TAG };
union OriginalDefDeclarationParameter
switch (OriginalDefDeclarationParameterTag){
    case ORIGINAL_DEFINITION_TAG:
        OriginalDefDeclaration
            original_definition;
    case ORIGINAL_DEFINITION_MACRO_TAG:
        OriginalDefDeclarationMacro
            original_definition_macro;
};
```

```
// Corresponding MHEG datatype: New-Channel
//=========================================================================
struct NewChannel {
    sequence<ChannelTargetsParameter,1>
        channel_targets_parameter;
    sequence<OriginalDefDeclarationParameter,1>
        original_definition;
};

// Corresponding MHEG datatype: Channel-Perceptability-Values-Macro
//=========================================================================
struct ChannelPerceptabilityValueMacro {
    MacroDefId
        macro_def_id;
    sequence<ChannelPerceptabilityValue,1>
        channel_perceptability_value;
};

// Corresponding MHEG datatype: Perceptability-Parameter
//=========================================================================
enum PerceptabilityParameterTag { CHANNEL_PERCEPTABILITY_VALUE_TAG,
CHANNEL_PERCEPTABILITY_VALUE_MACRO_TAG };
union PerceptabilityParameter
switch (PerceptabilityParameterTag){
    case CHANNEL_PERCEPTABILITY_VALUE_TAG:
        ChannelPerceptabilityValue
            channel_perceptability_value;
    case CHANNEL_PERCEPTABILITY_VALUE_MACRO_TAG:
        ChannelPerceptabilityValueMacro
            channel_perceptability_value_macro;
};

// Corresponding MHEG datatype: Set-Channel-Perceptability
//=========================================================================
struct SetChannelPerceptability {
    sequence<ChannelTargetsParameter,1>
        channel_targets_parameter;
    sequence<PerceptabilityParameter,1>
        perceptability_parameter;
};

// Corresponding MHEG datatype: Elementary-Action
//=========================================================================
enum ElementaryActionTag { DELAY_TAG, RETURN_TAG, SET_ALIAS_TAG, PREPARE_TAG,
DESTROY_TAG, ABORT_TAG, SET_DATA_TAG, COPY_TAG, SET_MULTIPLEX_TAG,
SET_DEMULTIPLEX_TAG, NEW_TAG, DELETE_TAG, SET_GLOBAL_BEHAVIOUR_TAG, RUN_TAG,
STOP_TAG, SET_PARAMETERS_TAG, PLUG_TAG, SET_RGS_TAG, SET_OPACITY_TAG,
SET_PRESENTATION_PRIORITY_TAG, SET_VISIBLE_DURATION_TAG,
SET_TEMPORAL_TERMINATION_TAG, SET_CURRENT_TEMPORAL_POSITION_TAG,
SET_VISIBLE_DURATION_POSITION_TAG, SET_SPEED_TAG, SET_TIMESTONES_TAG,
SET_PERCEPTIBLE_SIZE_PROJECTION_TAG, SET_RESIZING_STRATEGY_TAG,
SET_ASPECT_RATIO_PRESERVED_TAG, SET_VISIBLE_SIZE_TAG,
SET_VISIBLE_SIZES_ADJUSTMENT_TAG, SET_BOX_TAG, SET_DEFAULT_BACKGROUND_TAG,
SET_ATTACHMENT_POINT_TAG, SET_ATTACHMENT_POINT_POSITION_TAG,
SET_VISIBLE_SIZES_ALIGNMENT_TAG, SET_MOVING_ABILITY_TAG,
SET_RESIZING_ABILITY_TAG, SET_SCALING_ABILITY_TAG, SET_SCROLLING_ABILITY_TAG,
SET_AUDIBLE_VOLUME_TAG, SET_AUDIBLE_COMPOSITION_EFFECT_TAG,
SET_STREAM_CHOICE_TAG, SET_SELECTABILITY_TAG, SET_SELECTION_STATUS_TAG,
SET_SELECTION_PRESENTATION_EFFECT_RESPONSIBILITY_TAG, SET_MODIFIABILITY_TAG,
SET_MODIFICATION_STATUS_TAG,
SET_MODIFICATION_PRESENTATION_EFFECT_RESPONSIBILITY_TAG, SET_BUTTON_STYLE_TAG,
SET_SLIDER_STYLE_TAG, SET_ENTRY_FIELD_STYLE_TAG, SET_MENU_STYLE_TAG,
SET_SCROLLING_LIST_STYLE_TAG, SET_NO_STYLE_TAG, NEW_CHANNEL_TAG,
DELETE_CHANNEL_TAG, SET_CHANNEL_PERCEPTABILITY_TAG };
union ElementaryAction
switch (ElementaryActionTag){
    case DELAY_TAG:
        Delay
            delay;
    case RETURN_TAG:
        Return
            return;
```

```
case SET_ALIAS_TAG:
    SetAlias
        set_alias;
case PREPARE_TAG:
    sequence<MhTargetsParameter,1>
        prepare;
case DESTROY_TAG:
    sequence<MhTargetsParameter,1>
        destroy;
case ABORT_TAG:
    sequence<MhTargetsParameter,1>
        abort;
case SET_DATA_TAG:
    SetData
        set_data;
case COPY_TAG:
    Copy
        copy;
case SET_MULTIPLEX_TAG:
    SetMultiplex
        set_multiplex;
case SET_DEMULTIPLEX_TAG:
    SetDemultiplex
        set_demultiplex;
case NEW_TAG:
    sequence<RtTargetsParameter,1>
        new;
case DELETE_TAG:
    sequence<RtTargetsParameter,1>
        delete;
case SET_GLOBAL_BEHAVIOUR_TAG:
    SetGlobalBehaviour
        set_global_behaviour;
case RUN_TAG:
    sequence<RtSocketTargetsParameter,1>
        run;
case STOP_TAG:
    sequence<RtSocketTargetsParameter,1>
        stop;
case SET_PARAMETERS_TAG:
    SetParameters
        set_parameters;
case PLUG_TAG:
    Plug
        plug;
case SET_RGS_TAG:
    SetRGS
        set_RGS;
case SET_OPACITY_TAG:
    SetOpacity
        set_opacity;
case SET_PRESENTATION_PRIORITY_TAG:
    SetPresentationPriority
        set_presentation_priority;
case SET_VISIBLE_DURATION_TAG:
    SetVisibleDuration
        set_visible_duration;
case SET_TEMPORAL_TERMINATION_TAG:
    SetTemporalTermination
        set_temporal_termination;
case SET_CURRENT_TEMPORAL_POSITION_TAG:
    SetCurrentTemporalPosition
        set_current_temporal_position;
case SET_VISIBLE_DURATION_POSITION_TAG:
    SetVisibleDurationPosition
        set_visible_duration_position;
case SET_SPEED_TAG:
    SetSpeed
        set_speed;
case SET_TIMESTONES_TAG:
    SetTimestones
        set_timestones;
```

```
case SET_PERCEPTIBLE_SIZE_PROJECTION_TAG:
    SetPerceptibleSizeProjection
        set_perceptible_size_projection;
case SET_RESIZING_STRATEGY_TAG:
    SetResizingStrategy
        set_resizing_strategy;
case SET_ASPECT_RATIO_PRESERVED_TAG:
    SetAspectRatioPreserved
        set_aspect_ratio_preserved;
case SET_VISIBLE_SIZE_TAG:
    SetVisibleSize
        set_visible_size;
case SET_VISIBLE_SIZES_ADJUSTMENT_TAG:
    SetVisibleSizesAdjustment
        set_visible_sizes_adjustment;
case SET_BOX_TAG:
    SetBox
        set_box;
case SET_DEFAULT_BACKGROUND_TAG:
    SetDefaultBackground
        set_default_background;
case SET_ATTACHMENT_POINT_TAG:
    SetAttachmentPoint
        set_attachment_point;
case SET_ATTACHMENT_POINT_POSITION_TAG:
    SetAttachmentPointPosition
        set_attachment_point_position;
case SET_VISIBLE_SIZES_ALIGNMENT_TAG:
    SetVisibleSizesAlignment
        set_visible_sizes_alignment;
case SET_MOVING_ABILITY_TAG:
    SetMovingAbility
        set_moving_ability;
case SET_RESIZING_ABILITY_TAG:
    SetResizingAbility
        set_resizing_ability;
case SET_SCALING_ABILITY_TAG:
    SetScalingAbility
        set_scaling_ability;
case SET_SCROLLING_ABILITY_TAG:
    SetScrollingAbility
        set_scrolling_ability;
case SET_AUDIBLE_VOLUME_TAG:
    SetAudibleVolume
        set_audible_volume;
case SET_AUDIBLE_COMPOSITION_EFFECT_TAG:
    SetAudibleCompositionEffect
        set_audible_composition_effect;
case SET_STREAM_CHOICE_TAG:
    SetStreamChoice
        set_stream_choice;
case SET_SELECTABILITY_TAG:
    SetSelectability
        set_selectability;
case SET_SELECTION_STATUS_TAG:
    SetSelectionStatus
        set_selection_status;
case SET_SELECTION_PRESENTATION_EFFECT_RESPONSIBILITY_TAG:
    SetSelectionPresentationEffectResponsibility
        set_selection_presentation_effect_responsibility;
case SET_MODIFIABILITY_TAG:
    SetModifiability
        set_modifiability;
case SET_MODIFICATION_STATUS_TAG:
    SetModificationStatus
        set_modification_status;
case SET_MODIFICATION_PRESENTATION_EFFECT_RESPONSIBILITY_TAG:
    SetModificationPresentationEffectResponsibility
        set_modification_presentation_effect_responsibility;
case SET_BUTTON_STYLE_TAG:
    SetButtonStyle
        set_button_style;
```

```
    case SET_SLIDER_STYLE_TAG:
        SetSliderStyle
            set_slider_style;
    case SET_ENTRY_FIELD_STYLE_TAG:
        SetEntryFieldStyle
            set_entry_field_style;
    case SET_MENU_STYLE_TAG:
        SetMenuStyle
            set_menu_style;
    case SET_SCROLLING_LIST_STYLE_TAG:
        SetScrollingListStyle
            set_scrolling_list_style;
    case SET_NO_STYLE_TAG:
        sequence<RtSocketTargetsParameter,1>
            set_no_style;
    case NEW_CHANNEL_TAG:
        NewChannel
            new_channel;
    case DELETE_CHANNEL_TAG:
        sequence<ChannelTargetsParameter,1>
            delete_channel;
    case SET_CHANNEL_PERCEPTABILITY_TAG:
        SetChannelPerceptability
            set_channel_perceptability;
};

// Corresponding MHEG datatype: Keywords
//=============================================================================
typedef sequence<string> Keywords;

// Corresponding MHEG datatype: Description
//=============================================================================
struct Description {
    sequence<string,1>
        name;
    sequence<string,1>
        owner;
    sequence<string,1>
        version;
    sequence<string,1>
        date;
    sequence<Keywords,1>
        keywords;
    sequence<string,1>
        copyright;
    sequence<string,1>
        licence;
    sequence<string,1>
        comment;
};

// Corresponding MHEG datatype: Class-Identifier
//=============================================================================
enum ClassIdentifier {
    ACTION_CLASS,
    LINK_CLASS,
    SCRIPT_CLASS,
    CONTENT_CLASS,
    MULTIPLEXED_CONTENT_CLASS,
    COMPOSITE_CLASS,
    CONTAINER_CLASS,
    DESCRIPTOR_CLASS
};

// Corresponding MHEG datatype: OBJECTIDENTIFIER
//=============================================================================
struct OBJECTIDENTIFIER {
unsigned short root;
unsigned short arc1;
unsigned short arc2;
unsigned short arc3;
};
```

```
// Corresponding MHEG datatype: Synchro-Indicator
//=========================================================================
enum SynchroIndicator {
    SERIAL,
    PARALLEL
};

// Corresponding MHEG datatype: Synchro-Indicator-Macro
//=========================================================================
struct SynchroIndicatorMacro {
    MacroDefId
        macro_def_id;
    sequence<SynchroIndicator,1>
        synchro_indicator;
};

// Corresponding MHEG datatype: Synchro-Indicator-Parameter
//=========================================================================
enum SynchroIndicatorParameterTag { SYNCHRO_INDICATOR_TAG,
SYNCHRO_INDICATOR_MACRO_TAG };
union SynchroIndicatorParameter
switch (SynchroIndicatorParameterTag){
    case SYNCHRO_INDICATOR_TAG:
        SynchroIndicator
            synchro_indicator;
    case SYNCHRO_INDICATOR_MACRO_TAG:
        SynchroIndicatorMacro
            synchro_indicator_macro;
};

// Corresponding MHEG datatype: Performances
//=========================================================================
typedef long Performances;

// Corresponding MHEG datatype: Performances-Macro
//=========================================================================
struct PerformancesMacro {
    MacroDefId
        macro_def_id;
    sequence<Performances,1>
        performances;
};

// Corresponding MHEG datatype: Performances-Parameter
//=========================================================================
enum PerformancesParameterTag { PERFORMANCES_TAG, PERFORMANCES_MACRO_TAG };
union PerformancesParameter
switch (PerformancesParameterTag){
    case PERFORMANCES_TAG:
        Performances
            performances;
    case PERFORMANCES_MACRO_TAG:
        PerformancesMacro
            performances_macro;
};

interface ActionClass;

// Corresponding MHEG datatype: Action
//=========================================================================
enum ActionTag { ACTION_REFERENCE_TAG, ACTION_CLASS_TAG };
union Action
switch (ActionTag){
    case ACTION_REFERENCE_TAG:
        MhObjectReference
            action_reference;
    case ACTION_CLASS_TAG:
        ActionClass
            action_class;
};

// Corresponding MHEG datatype: Synchronized-Action
//=========================================================================
enum SynchronizedActionTag { ELEMENTARY_ACTION_TAG, ACTION_TAG };
```

```
union SynchronizedAction
switch (SynchronizedActionTag){
    case ELEMENTARY_ACTION_TAG:
        ElementaryAction
            elementary_action;
    case ACTION_TAG:
        Action
            action;
};
```

```
// Corresponding MHEG datatype: Action-Class
//=========================================================================
interface ActionClass {
attribute OBJECTIDENTIFIER
        object_identification;
attribute sequence<MHEGIdentifier,1>
        mheg_identifier;
attribute sequence<Description,1>
        the_description;
attribute sequence<SynchroIndicatorParameter,1>
        synchro_indicator_parameter;
attribute sequence<TargetsParameter,1>
        target_set_parameter;
attribute sequence<PerformancesParameter,1>
        performances_parameter;
attribute sequence<SynchronizedAction>
        synchronized_actions;
};
```

```
// Corresponding MHEG datatype: Logical-Operator
//=========================================================================
enum LogicalOperator {
    AND,
    OR,
    XOR,
    NAND,
    NOR,
    NXOR
};
```

```
// Corresponding MHEG datatype: Comparison-Operator
//=========================================================================
enum ComparisonOperator {
    EQUAL,
    NOT_EQUAL,
    GREATER,
    GREATER_EQUAL,
    LESS,
    LESS_EQUAL
};
```

```
// Corresponding MHEG datatype: Comparison-Value
//=========================================================================
enum ComparisonValueTag { COMPARISON_VALUE_GENERIC_VALUE_TAG,
COMPARISON_VALUE_COMPARISON_VALUE_CONSTANT_TAG };
union ComparisonValue
switch (ComparisonValueTag){
    case COMPARISON_VALUE_GENERIC_VALUE_TAG:
        GenericValue
            generic_value;
    case COMPARISON_VALUE_COMPARISON_VALUE_CONSTANT_TAG:
        ComparisonValueConstant
            comparison_value_constant;
};
```

```
// Corresponding MHEG datatype: Evaluated-Condition
//=========================================================================
struct EvaluatedCondition {
    ComparisonOperator
        comparison_operator;
    ComparisonValue
        comparison_value;
};
```

```
// Corresponding MHEG datatype: Previous-Condition
//=========================================================================
enum PreviousConditionTag { EVALUATED_CONDITION_TAG, NEGATION_TAG };
union PreviousCondition
switch (PreviousConditionTag){
    case EVALUATED_CONDITION_TAG:
        EvaluatedCondition
            evaluated_condition;
};

// Corresponding MHEG datatype: Generic-Condition
//=========================================================================
struct GenericCondition {
    EvaluatedValue
        source_value;
    sequence<PreviousCondition,1>
        previous_condition;
    EvaluatedCondition
        current_condition;
};

// Corresponding MHEG datatype: Link-Condition
//=========================================================================
enum LinkConditionTag { LOGICAL_COMBINATION_TAG, GENERIC_CONDITION_TAG };
union LinkCondition
switch (LinkConditionTag){
    case LOGICAL_COMBINATION_TAG:
        struct LogicalCombination {
            sequence<LogicalOperator,1>                  logical_operator;
            sequence<LinkCondition>                      conditions;
            } logical_combination;
    case GENERIC_CONDITION_TAG:
        GenericCondition
            generic_condition;
};

// Corresponding MHEG datatype: Macro-Resolution-Parameter
//=========================================================================
struct MacroResolutionParameter {
    MacroDefId
        macro_def_id;
    sequence<GenericValue,1>
        usage_value;
};

// Corresponding MHEG datatype: Link-Effect
//=========================================================================
struct LinkEffect {
    sequence<sequence<MacroResolutionParameter>,1>
        macro_resolution;
    Action
                        action;
};

// Corresponding MHEG datatype: Link-Class
//=========================================================================
interface LinkClass {
attribute OBJECTIDENTIFIER
        object_identification;
attribute sequence<MHEGIdentifier,1>
        mheg_identifier;
attribute sequence<Description,1>
        the_description;
attribute sequence<LinkCondition,1>
        link_condition;
attribute LinkEffect
            link_effect;
};

// Corresponding MHEG datatype: Script-Inclusion
//=========================================================================
enum ScriptInclusionTag { OCTETSTRING_TAG, BITSTRING_TAG };
union ScriptInclusion
```

```
switch (ScriptInclusionTag){
    case OCTETSTRING_TAG:
        string
            octetstring;
    case BITSTRING_TAG:
        string
            bitstring;
};

// Corresponding MHEG datatype: Script-Data
//=========================================================================
enum ScriptDataTag { SCRIPT_INCLUSION_TAG, DATA_REFERENCE_TAG };
union ScriptData
switch (ScriptDataTag){
    case SCRIPT_INCLUSION_TAG:
        ScriptInclusion
            script_inclusion;
    case DATA_REFERENCE_TAG:
        DataReference
            data_reference;
};

// Corresponding MHEG datatype: Script-Class
//=========================================================================
interface ScriptClass {
attribute OBJECTIDENTIFIER
        object_identification;
attribute sequence<MHEGIdentifier,1>
        mheg_identifier;
attribute sequence<Description,1>
        the_description;
attribute sequence<ScriptClassification,1>
        script_classification;
attribute ScriptHook
            script_hook_information;
attribute ScriptData
            script_data;
};

// Corresponding MHEG datatype: Data-Inclusion
//=========================================================================
enum DataInclusionTag { DATA_INCLUSION_OCTET_STRING_TAG,
DATA_INCLUSION_BIT_STRING_TAG };
union DataInclusion
switch (DataInclusionTag){
    case DATA_INCLUSION_OCTET_STRING_TAG:
        string
            octet_string;
    case DATA_INCLUSION_BIT_STRING_TAG:
        string
            bit_string;
};

// Corresponding MHEG datatype: Content-Data
//=========================================================================
enum ContentDataTag { CONTENT_DATA_DATA_INCLUSION_TAG,
CONTENT_DATA_DATA_REFERENCE_TAG };
union ContentData
switch (ContentDataTag){
    case CONTENT_DATA_DATA_INCLUSION_TAG:
        DataInclusion
            data_inclusion;
    case CONTENT_DATA_DATA_REFERENCE_TAG:
        DataReference
            data_reference;
};

// Corresponding MHEG datatype: Content-Class
//=========================================================================
interface ContentClass {
attribute OBJECTIDENTIFIER
        object_identification;
```

```
attribute MHEGIdentifier
        mheg_identifier;
attribute Description
            the_description;
attribute ContentClassification
        classification;
attribute OriginalPerception
        original_perception;
attribute ContentHook
            content_hook;
attribute ContentData
            content_data;
};

// Corresponding MHEG datatype: Multiplexed-Stream
//=============================================================================
struct MultiplexedStream {
    sequence<sequence<long>,1>
        stream_identifier;
    sequence<ContentClassification,1>
        stream_classification;
    sequence<OriginalPerception,1>
        stream_original_perception;
    ContentHook
                    hook_stream;
};

// Corresponding MHEG datatype: Multiplexed-Content-Class
//=============================================================================
interface MultiplexedContentClass {
attribute OBJECTIDENTIFIER
        object_identification;
attribute sequence<MHEGIdentifier,1>
        mheg_identifier;
attribute sequence<Description,1>
        the_description;
attribute sequence<ContentClassification,1>
        mux_classification;
attribute sequence<OriginalPerception,1>
        original_perception;
attribute ContentHook
            mux_hook;
attribute ContentData
            mux_data;
attribute sequence<sequence<MultiplexedStream>,1>
        multiplexed_streams;
};

// Corresponding MHEG datatype: Link
//=============================================================================
enum LinkTag { LINK_REFERENCE_TAG, LINK_CLASS_TAG };
union Link
switch (LinkTag){
    case LINK_REFERENCE_TAG:
        MhObjectReference
            mh_object_reference;
    case LINK_CLASS_TAG:
        LinkClass
            link;
};

// Corresponding MHEG datatype: Specific-Behaviour
//=============================================================================
enum SpecificBehaviourTag { ACTIONS_TAG, LINKS_TAG };
union SpecificBehaviour
switch (SpecificBehaviourTag){
    case ACTIONS_TAG:
        sequence<Action>
            actions;
    case LINKS_TAG:
        sequence<Link>
            links;
};
```

```
// Corresponding MHEG datatype: Availability-Start-Up
// Corresponding MHEG datatype: Availability-Close-Down
// Corresponding MHEG datatype: Rt-Availability-Start-Up
//=========================================================================
enum PredefinedBehaviourTag { PREDEFINED_BEHAVIOUR_LINK_CLASS_TAG,
PREDEFINED_BEHAVIOUR_INHIBIT_TAG };
union PredefinedBehaviour
switch (PredefinedBehaviourTag){
    case PREDEFINED_BEHAVIOUR_LINK_CLASS_TAG:
        LinkClass
            link_class;
};

// Corresponding MHEG datatype: Predefined-Behaviour
//=========================================================================
struct PredefinedBehaviours {
    sequence<PredefinedBehaviour,1>
        availability_start_up;
    sequence<PredefinedBehaviour,1>
        availability_close_down;
    sequence<PredefinedBehaviour,1>
        rt_availability_start_up;
    sequence<LinkClass,1>
        rt_availability_close_down;
};

// Corresponding MHEG datatype: Composition-Behaviour
//=========================================================================
struct CompositionBehaviour {
    PredefinedBehaviours
        predefined_behaviours;
    SpecificBehaviour
        specific_behaviour;
};

interface CompositeClass;

// Corresponding MHEG datatype: Component
//=========================================================================
enum ComponentTag { MH_OBJECT_REFERENCE_TAG, CONTENT_TAG,
MULTIPLEXED_CONTENT_TAG, COMPOSITE_TAG };
union Component
switch (ComponentTag){
    case MH_OBJECT_REFERENCE_TAG:
        MhObjectReference
            mh_object_reference;
    case CONTENT_TAG:
        ContentClass
            content;
    case MULTIPLEXED_CONTENT_TAG:
        MultiplexedContentClass
            multiplexed_content;
    case COMPOSITE_TAG:
        CompositeClass
            composite;
};

// Corresponding MHEG datatype: Associated-Model
//=========================================================================
enum AssociatedModelTag { ASSOCIATED_MODEL_COMPONENT_TAG,
ASSOCIATED_MODEL_ASSOCIATED_LABEL_TAG };
union AssociatedModel
switch (AssociatedModelTag){
    case ASSOCIATED_MODEL_COMPONENT_TAG:
        Component
            component;
    case ASSOCIATED_MODEL_ASSOCIATED_LABEL_TAG:
        string
            associated_label;
};
```

```
// Corresponding MHEG datatype: Element
//=========================================================================
struct Element {
    long
        element_index;
    AssociatedModel
        associated_model;
};

// Corresponding MHEG datatype: Composition
//=========================================================================
struct Composition {
    long
        nb_elements;
    sequence<Element>
        elements;
};

// Corresponding MHEG datatype: Composite-Class
//=========================================================================
interface CompositeClass {
attribute OBJECTIDENTIFIER
        object_identification;
attribute sequence<MHEGIdentifier,1>
        mheg_identifier;
attribute sequence<Description,1>
        the_description;
attribute sequence<CompositionBehaviour,1>
        composition_behaviour;
attribute sequence<OriginalPerception,1>
        original_perception;
attribute Composition
            elements;
};

// Corresponding MHEG datatype: Container-Default-Behaviour
//=========================================================================
struct ContainerDefaultBehaviour {
    sequence<PredefinedBehaviour,1>
        availability_start_up;
    sequence<PredefinedBehaviour,1>
        availability_close_down;
};

interface ContainerClass;
interface DescriptorClass;

// Corresponding MHEG datatype: Mh-Element
//=========================================================================
enum MhElementTag { MH_ELEMENT_MH_OBJECT_REFERENCE_TAG, MH_ELEMENT_ACTION_TAG,
MH_ELEMENT_LINK_TAG, MH_ELEMENT_SCRIPT_TAG, MH_ELEMENT_CONTENT_TAG,
MH_ELEMENT_MUX_CONTENT_TAG, MH_ELEMENT_COMPOSITE_TAG, MH_ELEMENT_CONTAINER_TAG,
MH_ELEMENT_DESCRIPTOR_TAG };
union MhElement
switch (MhElementTag){
    case MH_ELEMENT_MH_OBJECT_REFERENCE_TAG:
        MhObjectReference
            mh_object_reference;
    case MH_ELEMENT_ACTION_TAG:
        ActionClass
            action;
    case MH_ELEMENT_LINK_TAG:
        LinkClass
            link;
    case MH_ELEMENT_SCRIPT_TAG:
        ScriptClass
            script;
    case MH_ELEMENT_CONTENT_TAG:
        ContentClass
            content;
    case MH_ELEMENT_MUX_CONTENT_TAG:
        MultiplexedContentClass
            mux_content;
```

```
    case MH_ELEMENT_COMPOSITE_TAG:
        CompositeClass
            composite;
    case MH_ELEMENT_CONTAINER_TAG:
        ContainerClass
            container;
    case MH_ELEMENT_DESCRIPTOR_TAG:
        DescriptorClass
            descriptor;
};

// Corresponding MHEG datatype: Container-Class
//=========================================================================
interface ContainerClass {
attribute OBJECTIDENTIFIER
        object_identification;
attribute sequence<MHEGIdentifier,1>
        mheg_identifier;
attribute sequence<Description,1>
        the_description;
attribute ContainerDefaultBehaviour
        container_default_behaviour;
attribute sequence<MhElement>
        container_elements;
};

// Corresponding MHEG datatype: Delay-Bounds
//=========================================================================
struct DelayBounds {
    long
        lower_value;
    long
        upper_value;
};

// Corresponding MHEG datatype: Degradation
//=========================================================================
enum Degradation {
    GUARANTEED,
    BEST_EFFORT,
    NOT_GUARANTEED
};

// Corresponding MHEG datatype: Quality-Of-Service
//=========================================================================
struct QualityOfService {
    sequence<Degradation,1>
        degradation;
    sequence<boolean,1>
        reliability;
    sequence<DelayBounds,1>
        delay_bounds;
    sequence<long,1>
        jitter_bounds;
};

// Corresponding MHEG datatype: Script-Class-Information
//=========================================================================
struct ScriptClassInformation {
    sequence<ScriptClassification,1>
        script_classification;
    sequence<ScriptHook,1>
        script_hook;
};

// Corresponding MHEG datatype: Alternative-Object
//=========================================================================
struct AlternativeObject {
    MhObjectReference
        content_or_mux_object;
    sequence<ContentHook,1>
        alternative_hook;
    sequence<MhObjectReference,1>
        alternative_descriptor;
```

```
    sequence<string,1>
        alternative_keyword;
};

// Corresponding MHEG datatype: Content-Class-Information
//=========================================================================
struct ContentClassInformation {
    sequence<ContentClassification,1>
        content_classification;
    sequence<ContentHook,1>
        content_hook;
    sequence<sequence<AlternativeObject>,1>
        alternative_object;
};

// Corresponding MHEG datatype: Stream-Information
//=========================================================================
struct StreamInformation {
    sequence<StreamIdentifier,1>
        stream_identifier;
    sequence<ContentClassInformation,1>
        content_class_information;
};

// Corresponding MHEG datatype: Multiplexed-Content-Class-Information
//=========================================================================
struct MultiplexedContentClassInformation {
    sequence<ContentClassification,1>
        multiplexed_content_classification;
    sequence<ContentHook,1>
        multiplexed_content_hook;
    sequence<long,1>
        stream_number;
    sequence<sequence<StreamInformation>,1>
        stream_information;
    sequence<sequence<AlternativeObject>,1>
        alternative_object;
};

// Corresponding MHEG datatype: Class-Specific
//=========================================================================
enum ClassSpecificTag { SCRIPT_CLASS_INFORMATION_TAG,
CONTENT_CLASS_INFORMATION_TAG, MULTIPLEXED_CONTENT_CLASS_INFORMATION_TAG };
union ClassSpecific
switch (ClassSpecificTag){
    case SCRIPT_CLASS_INFORMATION_TAG:
        ScriptClassInformation
            script_class_information;
    case CONTENT_CLASS_INFORMATION_TAG:
        ContentClassInformation
            content_class_information;
    case MULTIPLEXED_CONTENT_CLASS_INFORMATION_TAG:
        MultiplexedContentClassInformation
            multiplexed_content_class_information;
};

// Corresponding MHEG datatype: Object-Information
//=========================================================================
struct ObjectInformation {
    sequence<long,1>
        object_size;
    sequence<ClassIdentifier,1>
        class_identifier;
    sequence<ClassSpecific,1>
        class_specific;
    sequence<QualityOfService,1>
        quality_of_service;
};

// Corresponding MHEG datatype: Related-Object
//=========================================================================
struct RelatedObject {
    MhObjectReference
        object_reference;
```

```
    sequence<ObjectInformation,1>
        object_information;
};

// Corresponding MHEG datatype: System-Readable-Material
//=========================================================================
enum SystemReadableMaterialTag { SYSTEM_READABLE_MATERIAL_BITSTRING_TAG,
SYSTEM_READABLE_MATERIAL_OCTETSTRING_TAG };
union SystemReadableMaterial
switch (SystemReadableMaterialTag){
    case SYSTEM_READABLE_MATERIAL_BITSTRING_TAG:
        string
            bitstring;
    case SYSTEM_READABLE_MATERIAL_OCTETSTRING_TAG:
        string
            octetstring;
};

// Corresponding MHEG datatype: General-Information
//=========================================================================
struct GeneralInformation {
    sequence<string,1>
        readme;
    sequence<SystemReadableMaterial,1>
        system_readable_material;
};

// Corresponding MHEG datatype: Media-Type
//=========================================================================
enum MediaType {
    AUDIBLE,
    LEFT_AUDIBLE,
    RIGHT_AUDIBLE,
    VISIBLE
};

// Corresponding MHEG datatype: Predefined-Selection-Mode
//=========================================================================
enum PredefinedSelectionMode {
    MOUSE_BUTTON_1_DOWN,
    MOUSE_BUTTON_2_DOWN,
    MOUSE_BUTTON_3_DOWN,
    MOUSE_BUTTONS_1_2_DOWN,
    MOUSE_BUTTONS_1_3_DOWN,
    MOUSE_BUTTONS_2_3_DOWN,
    MOUSE_BUTTON_1_UP,
    MOUSE_BUTTON_2_UP,
    MOUSE_BUTTON_3_UP,
    MOUSE_BUTTONS_1_2_UP,
    MOUSE_BUTTONS_1_3_UP,
    MOUSE_BUTTONS_2_3_UP,
    MOUSE_SINGLE_CLICK_ON_BUTTON_1,
    MOUSE_SINGLE_CLICK_ON_BUTTON_2,
    MOUSE_SINGLE_CLICK_ON_BUTTON_3,
    MOUSE_SINGLE_CLICK_ON_BUTTONS_12,
    MOUSE_SINGLE_CLICK_ON_BUTTONS_13,
    MOUSE_SINGLE_CLICK_ON_BUTTONS_23,
    MOUSE_DOUBLE_CLICK_ON_BUTTON_1,
    MOUSE_DOUBLE_CLICK_ON_BUTTON_2,
    MOUSE_DOUBLE_CLICK_ON_BUTTON_3,
    MOUSE_DOUBLE_CLICK_ON_BUTTONS_12,
    MOUSE_DOUBLE_CLICK_ON_BUTTONS_13,
    MOUSE_DOUBLE_CLICK_ON_BUTTONS_23,
    MOUSE_TRIPLE_CLICK_ON_BUTTON_1,
    MOUSE_TRIPLE_CLICK_ON_BUTTON_2,
    MOUSE_TRIPLE_CLICK_ON_BUTTON_3,
    MOUSE_TRIPLE_CLICK_ON_BUTTONS_12,
    MOUSE_TRIPLE_CLICK_ON_BUTTONS_13,
    MOUSE_TRIPLE_CLICK_ON_BUTTONS_23,
    MOUSE_ENTER,
    MOUSE_LEAVE,
    MOUSE_POINTER_MOTION,
```

```
    MOUSE_BUTTON_1_MOTION,
    MOUSE_BUTTON_2_MOTION,
    MOUSE_BUTTON_3_MOTION,
    MOUSE_BUTTONS_1_2_MOTION,
    MOUSE_BUTTONS_1_3_MOTION,
    MOUSE_BUTTONS_2_3_MOTION,
    KEY_PRESS,
    KEY_RELEASE
};

// Corresponding MHEG datatype: Physical-Selection-Mode
//=========================================================================
enum PhysicalSelectionModeTag { PRIVATE_MODE_TAG, PREDEFINED_TAG };
union PhysicalSelectionMode
switch (PhysicalSelectionModeTag){
    case PRIVATE_MODE_TAG:
        long
            private_mode;
    case PREDEFINED_TAG:
        PredefinedSelectionMode
            predefined;
};

// Corresponding MHEG datatype: Selection-Mode
//=========================================================================
struct SelectionMode {
    long
        logical_selection_mode;
    sequence<PhysicalSelectionMode,1>
        physical_selection_mode;
};

// Corresponding MHEG datatype: Predefined-Modification-Mode
//=========================================================================
enum PredefinedModificationMode {
    MEDIA_EDITOR
};

// Corresponding MHEG datatype: Physical-Modification-Mode
//=========================================================================
enum PhysicalModificationModeTag { PHYSICAL_MODIFICATION_MODE_NUMERIC_TAG,
PHYSICAL_MODIFICATION_MODE_PREDEFINED_TAG };
union PhysicalModificationMode
switch (PhysicalModificationModeTag){
    case PHYSICAL_MODIFICATION_MODE_NUMERIC_TAG:
        long
            numeric;
    case PHYSICAL_MODIFICATION_MODE_PREDEFINED_TAG:
        PredefinedModificationMode
            predefined;
};

// Corresponding MHEG datatype: Modification-Mode
//=========================================================================
struct ModificationMode {
    long
        logical_modification_mode;
    sequence<PhysicalModificationMode,1>
        physical_modification_mode;
};

// Corresponding MHEG datatype: Channel-Information
//=========================================================================
struct ChannelInformation {
    sequence<ChannelIdentifier,1>
        channel_id;
    sequence<long,1>
        x_min;
    sequence<long,1>
        x_max;
    sequence<long,1>
        y_min;
```

```
    sequence<long,1>
        y_max;
    sequence<long,1>
        z_min;
    sequence<long,1>
        z_max;
    sequence<long,1>
        x_granularity;
    sequence<long,1>
        y_granularity;
    sequence<long,1>
        z_granularity;
    sequence<long,1>
        t_granularity;
    sequence<long,1>
        f_min;
    sequence<long,1>
        f_max;
    sequence<long,1>
        audio_dynamic;
    sequence<MediaType,1>
        media_type;
    sequence<SelectionMode,1>
        selection_mode;
    sequence<ModificationMode,1>
        modification_mode;
};

// Corresponding MHEG datatype: Interaction-Style-Information
//=============================================================================
enum InteractionStyleInformation {
    BUTTON,
    SLIDER,
    ENTRY_FIELD,
    MENU,
    SCROLLING_LIST
};

// Corresponding MHEG datatype: Descriptor-Class
//=============================================================================
interface DescriptorClass {
attribute OBJECTIDENTIFIER
        object_identification;
attribute sequence<MHEGIdentifier,1>
        mheg_identifier;
attribute sequence<Description,1>
        the_description;
attribute sequence<sequence<RelatedObject>,1>
        related_objects;
attribute sequence<sequence<MhObjectReference>,1>
        other_descriptors;
attribute sequence<GeneralInformation,1>
        general_information;
attribute sequence<sequence<ChannelInformation>,1>
        channel_information;
attribute sequence<sequence<InteractionStyleInformation>,1>
        interaction_styles_information;
};

}; // end of module
```

SÉRIES DES RECOMMANDATIONS UIT-T

Série A	Organisation du travail de l'UIT-T
Série B	Moyens d'expression: définitions, symboles, classification
Série C	Statistiques générales des télécommunications
Série D	Principes généraux de tarification
Série E	Exploitation générale du réseau, service téléphonique, exploitation des services et facteurs humains
Série F	Services de télécommunication non téléphoniques
Série G	Systèmes et supports de transmission, systèmes et réseaux numériques
Série H	Systèmes audiovisuels et multimédias
Série I	Réseau numérique à intégration de services
Série J	Transmission des signaux radiophoniques, télévisuels et autres signaux multimédias
Série K	Protection contre les perturbations
Série L	Construction, installation et protection des câbles et autres éléments des installations extérieures
Série M	Maintenance: systèmes de transmission, de télégraphie, de télécopie, circuits téléphoniques et circuits loués internationaux
Série N	Maintenance: circuits internationaux de transmission radiophonique et télévisuelle
Série O	Spécifications des appareils de mesure
Série P	Qualité de transmission téléphonique, installations téléphoniques et réseaux locaux
Série Q	Commutation et signalisation
Série R	Transmission télégraphique
Série S	Equipements terminaux de télégraphie
Série T	**Terminaux des services télématiques**
Série U	Commutation télégraphique
Série V	Communications de données sur le réseau téléphonique
Série X	Réseaux pour données et communication entre systèmes ouverts
Série Z	Langages de programmation