Union internationale des télécommunications

UIT-T

SECTEUR DE LA NORMALISATION
DES TÉLÉCOMMUNICATIONS
DE L'UIT

J.280

(12/2005)

SÉRIE J: RÉSEAUX CÂBLÉS ET TRANSMISSION DES SIGNAUX RADIOPHONIQUES, TÉLÉVISUELS ET AUTRES SIGNAUX MULTIMÉDIAS

Transmission numérique des signaux de télévision

Insertion numérique de programme: interface de programme de collage virtuel

Recommandation UIT-T J.280

Recommandation UIT-T J.280

Insertion numérique de programme: interface de programme de collage virtuel

Résumé

La présente Recommandation spécifie une interface de programme d'application (API, *application program interface*) à partir de laquelle est définie une méthode normalisée de communication entre des serveurs et des colleuses virtuelles en vue de l'insertion d'un contenu dans un multiplex de sortie MPEG-2 d'une colleuse. La souplesse offerte par cette interface API permet de prendre en charge un ou plusieurs serveurs affectés à une ou plusieurs colleuses. L'insertion dans des programmes numériques peut porter sur des contenus tels des annonces publicitaires de durée différente, la substitution de programmes, des annonces de services publics ou des éléments de programme créés par collage virtuel de parties de programme provenant d'un serveur.

Source

Ce texte a été approuvé le 14 décembre 2005 en tant qu'amendement 1 de la Recommandation UIT-T J.280 (2004) par la Commission d'études 9 (2005-2008) de l'UIT-T selon la procédure définie dans la Recommandation UIT-T A.8. Il a été décidé par la suite de republier le texte complet de la Recommandation.

Mots clés

API, collage, colleuse virtuelle, insertion de programme, serveur.

AVANT-PROPOS

L'UIT (Union internationale des télécommunications) est une institution spécialisée des Nations Unies dans le domaine des télécommunications. L'UIT-T (Secteur de la normalisation des télécommunications) est un organe permanent de l'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.

L'Assemblée mondiale de normalisation des télécommunications (AMNT), qui se réunit tous les quatre ans, détermine les thèmes d'étude 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 1 de l'AMNT.

Dans certains secteurs des technologies de l'information qui correspondent à la sphère de compétence de l'UIT-T, les normes nécessaires se préparent en collaboration avec l'ISO et la CEI.

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.

Le respect de cette Recommandation se fait à titre volontaire. Cependant, il se peut que la Recommandation contienne certaines dispositions obligatoires (pour assurer, par exemple, l'interopérabilité et l'applicabilité) et considère que la Recommandation est respectée lorsque toutes ces dispositions sont observées. Le futur d'obligation et les autres moyens d'expression de l'obligation comme le verbe "devoir" ainsi que leurs formes négatives servent à énoncer des prescriptions. L'utilisation de ces formes ne signifie pas qu'il est obligatoire de respecter la Recommandation.

DROITS DE PROPRIÉTÉ INTELLECTUELLE

L'UIT attire l'attention sur la possibilité que l'application ou la mise en œuvre de la présente Recommandation puisse donner lieu à l'utilisation d'un droit de propriété intellectuelle. L'UIT ne prend pas position en ce qui concerne l'existence, la validité ou l'applicabilité des droits de propriété intellectuelle, qu'ils soient revendiqués par un Membre de l'UIT ou par une tierce partie étrangère à la procédure d'élaboration des Recommandations.

A la date d'approbation de la présente Recommandation, l'UIT n'avait pas été avisée de l'existence d'une propriété intellectuelle protégée par des brevets à acquérir pour mettre en œuvre la présente Recommandation. Toutefois, comme il ne s'agit peut-être pas de renseignements les plus récents, il est vivement recommandé aux responsables de la mise en œuvre de consulter la base de données des brevets du TSB.

TABLE DES MATIÈRES

Page

Recommandation UIT-T J.280

Insertion numérique de programme: interface de programme de collage virtuel

1 Domaine d'application

La présente Recommandation spécifie une interface de programme d'application (API, *application program interface*) à partir de laquelle est définie une méthode normalisée de communication entre des serveurs et des colleuses virtuelles (*splicers*) en vue de l'insertion d'un contenu dans un multiplex de sortie MPEG-2 d'une colleuse. La souplesse offerte par cette interface API permet de prendre en charge un ou plusieurs serveurs affectés à une ou plusieurs colleuses. L'insertion dans des programmes numériques peut porter sur des contenus tels des annonces publicitaires de durée différente, la substitution de programmes, des annonces de services publics ou des éléments de programme créés par collage virtuel de parties de programme provenant d'un serveur.

La présente Recommandation ne porte pas sur la commande et le contrôle des opérations de postproduction ou de montage (superposition d'images ou effets de compression d'image), et ne définit pas la façon de réaliser une collure (*splice*) et ne spécifie pas le degré d'invisibilité des collures. De plus, l'importante question de la synchronisation de flux asynchrones à assembler par collage (*to splice*) est un détail d'implémentation qui relève de la colleuse [3].

2 Références

2.1 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. La référence à un document figurant dans la présente Recommandation ne donne pas à ce document, en tant que tel, le statut d'une Recommandation.

[1] Recommandation UIT-T H.222.0 (2000) | ISO/CEI 13818-1:2000, *Technologies de l'information – Codage générique des images animées et du son associé: systèmes*.

[2] Recommandation UIT-T H.262 (2000) | ISO/CEI 13818-2:2000, *Technologies de l'information – Codage générique des images animées et du son associé: données vidéo*.

[3] Recommandation UIT-T J.181 (2004), *Message de repérage d'insertion de programme numérique pour systèmes de télévision par câble*.

2.2 Références informatives

[4] Appendice I à la Recommandation UIT-T J.181 (2004), *Pratiques recommandées et guide d'interprétation*.

[5] IETF RFC 3810 (2004), *Multicast Listener Discovery Version 2 (MLDv2) for IPv6*.

3 Définitions

La présente Recommandation définit les termes suivants:

3.1 connexion API: connexion par connecteur TCP/IP entre un serveur et une colleuse, utilisée pour le transfert des messages API.

3.2 insertions successives: plusieurs sessions temporellement contiguës sans retour au canal primaire entre sessions.

3.3 canal: synonyme de "service" en terminologie DVB ou de "programme" en terminologie MPEG.

3.4 canal d'insertion: canal ou canaux multiplex d'insertion se substituant en totalité ou en partie au canal principal pendant la durée d'un événement collage.

3.5 multiplex d'insertion: source du canal d'insertion. Un multiplex produit par un serveur peut dans certaines circonstances exclure l'information PSI, ainsi il est entendu que ce multiplex peut être un flux de transport de MPEG-2 non conforme.

3.6 multiplex: ensemble d'un ou de plusieurs canaux qui peuvent inclure l'information de service associée. Un multiplex est un flux de transport MPEG-2 à l'exception éventuelle d'un multiplex d'insertion.

3.7 canal de sortie: canal produit en sortie de la colleuse.

3.8 multiplex de sortie: flux de transport MPEG-2 produit par multiplexage d'un ou de plusieurs canaux de sortie. La colleuse doit en permanence faire en sorte que le multiplex de sortie contient une information PSI valide.

3.9 canal primaire: canal multiplex primaire qui est remplacé en partie ou en totalité. Un seul canal primaire peut aboutir à plusieurs canaux de sortie.

3.10 multiplex primaire: source du ou des canaux primaires.

3.11 serveur: dispositif générant un ou plusieurs canaux d'insertion à assembler dans le ou les canaux primaires. Ce dispositif communique avec la colleuse pour ce qui est de savoir quand et quoi assembler.

3.12 session: insertion de contenus (tels des annonces publicitaires de différentes longueurs, des substitutions de programmes, des annonces de services publics ou des éléments de programmes créés par assemblage de parties du programme depuis un serveur). Chaque session est identifiée par un identificateur **SessionID** unique.

3.13 collure de début: collure en début d'insertion. A lieu à l'instant spécifié dans le message **Splice_Request**.

3.14 collure de fin: collure en fin d'insertion. L'instant prévu de fin d'insertion est calculé en additionnant l'instant de début et la durée spécifiés dans le message **Splice_Request**. Cet événement peut toutefois avoir lieu prématurément sous certaines conditions d'erreur.

3.15 colleuse: dispositif qui permet d'assembler par collage un ou plusieurs canaux d'insertion dans un ou plusieurs canaux primaires. Elle peut recevoir des messages de repérage J.181. Ce dispositif communique également avec le serveur pour savoir quand et quoi assembler.

4 Abréviations

La présente Recommandation utilise les abréviations suivantes:

API interface de programme d'application (*application program interface*)

CNN réseau d'actualités par câble (*cable news network*)

DVB-ASI diffusion vidéonumérique – Interface série asynchrone (*digital video broadcast – asynchronous serial interface*)

ID identificateur

ISO Organisation internationale de normalisation (*International Organization for Standardization*)

MLD découverte de trafic multicast (*multicast listener discovery*)

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

MPTS flux de transport multiprogrammes (*multi-program transport stream*)

NCTA *National Cable Television Association*

NTP protocole relatif au temps dans le réseau (*network time protocol*)

PAT table d'association de programmes (*program association table*)

PCR référence d'horloge programme (*program clock reference*)

PID identificateur de paquet (*packet identifier*)

PMT table de contenu de programme (*program map table*)

PSI information spécifique du programme (*program specific information*)

SCTE Société des ingénieurs en télécommunication par câble (*Society of cable telecommunications engineers*)

SPTS flux de transport de programme unique (*single program transport stream*)

TCP/IP protocole de commande de transport/protocole Internet (*transport control protocol/Internet protocol*)

uimsbf entier non signé, bit de plus fort poids en tête (*unsigned integer, most significant bit first*)

UIT Union internationale des télécommunications

UTC temps universel coordonné (*coordinated universal time*)

5 Doit, devrait, pourrait

Dans la présente Recommandation sont exprimés différents degrés de conditionnalité: "doit", "il faut que" indiquant l'obligation; "devrait", "de préférence" exprimant une caractéristique préférable; "pourrait" exprimant une caractéristique éventuelle laissée à la discrétion de l'utilisateur.

6 Introduction

6.1 Schéma fonctionnel

Cette interface API peut être utilisée dans de nombreuses configurations différentes de serveurs et de colleuses. La Figure 1 illustre une configuration à serveur et colleuse uniques, mais il est possible de la transposer en une configuration à plusieurs serveurs et à plusieurs colleuses comme le montre la Figure 2.

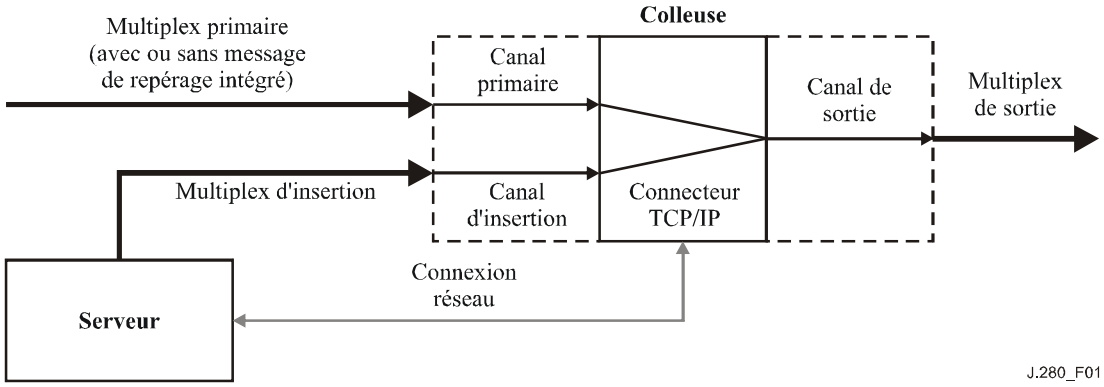


Figure 1/J.280 – Configuration à serveur et à colleuse uniques

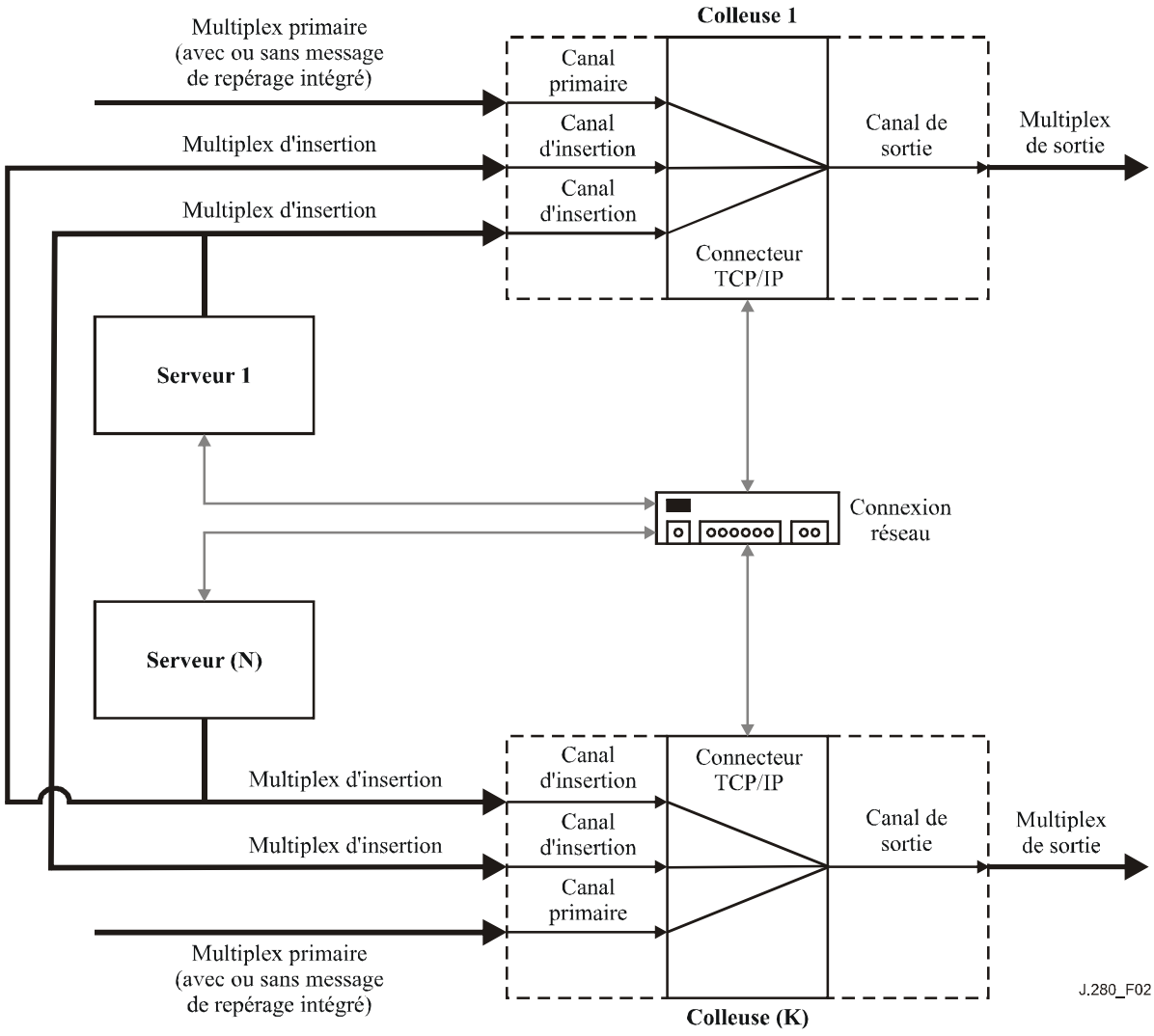


Figure 2/J.280 – Configuration à serveurs et à colleuses multiples

Le modèle de cette interface API comporte une colleuse qui dispose d'une ou de plusieurs entrées multiplex. La colleuse sépare de manière logique le ou les canaux dans le ou les multiplex et présente le ou les canaux à un commutateur. Ce commutateur permet de faire correspondre un canal de sortie quelconque à une entrée quelconque. Dans la configuration initiale, le ou les canaux primaires sont mappés avec le ou les canaux de sortie. Le serveur peut alors ordonner à la colleuse

de passer d'un canal primaire à un canal d'insertion pendant une durée spécifiée. Il peut alors ordonner à la colleuse de passer à un nouveau canal d'insertion après la commutation initiale.

Logiquement, une collure fait intervenir deux canaux d'entrée et un canal de sortie. La colleuse est chargée de l'assemblage de divers flux élémentaires (audio, vidéo et données). L'instant de collure optimal peut correspondre à des instants légèrement différents pour chaque flux élémentaire afin que la colleuse puisse effectuer l'assemblage en offrant la meilleure qualité en sortie. L'assemblage ne se fait pas parfois dans le sens "réseau de programmation" (canal primaire) vers "l'annonce publicitaire" (canal d'insertion) et avec retour vers le réseau de programmation. La colleuse peut assembler le contenu qui est uniquement stocké chez le serveur et qui parvient via un multiplex à entrée unique. Il est possible d'utiliser cette interface API dans une configuration où le serveur dispose d'une seule sortie de flux MPTS qui contient le programme et les éléments à insérer et utilise la colleuse pour créer les collures appropriées dans le contenu.

Cette interface API prend en charge toutes les combinaisons à un ou à plusieurs serveurs communicant avec une ou plusieurs colleuses. Une connexion API distincte est associée à chaque canal de sortie.

Dans certaines configurations, plusieurs serveurs ou plusieurs canaux dans le multiplex d'insertion peuvent être reliés à une colleuse. Dans ces cas, la colleuse dispose de plusieurs connexions API associées à un canal de sortie. A la réception d'un message de repérage J.181 dans un canal primaire, un message **Cue_Request** doit être envoyé au serveur via toutes les connexions API qui ont été établies pour le ou les canaux de sortie associés. Il est également possible que plusieurs connexions API transportent un message **Splice_Request** pour une même insertion au même instant pour un canal de sortie donné.

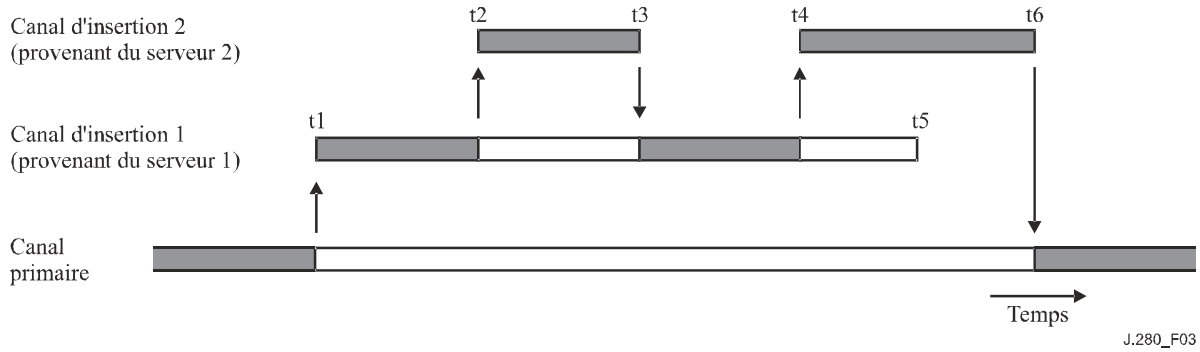
6.2 Arbitrage des priorités

Différents niveaux d'accès sont prévus afin que l'assemblage porte sur le bon canal d'insertion. Il existe dix niveaux différents d'accès numérotés de 0 à 9. Le niveau 9 correspond à la priorité la plus élevée et peut préempter toute connexion de priorité inférieure. L'indicateur **OverridePlaying** dans le message **Splice_Request** indique si une demande d'insertion est honorée lorsque la colleuse est en file d'attente ou si elle exécute une insertion. Si cet indicateur est mis à 1, l'insertion de priorité la plus élevée peut interrompre l'insertion de priorité identique ou inférieure en cours. Si l'indicateur est mis à 0, la colleuse n'interrompra pas l'insertion en cours d'exécution même si la nouvelle demande a une priorité plus élevée.

Pour être valide, le message **Splice_Request** devrait être envoyé au moins trois secondes avant l'instant time( ) de collure. Si cette condition n'est pas remplie, le résultat correspondant au message **Splice_Request** n'est pas déterminé par cette interface API. Si plusieurs serveurs adressent des demandes de collage pour un même instant avec la même priorité, la colleuse appliquera le principe du "premier arrivé, premier servi". Toutes les autres demandes seront rejetées et une indication d'erreur par collision sera envoyée dans le message **Splice_Response** (sauf si l'indicateur **OverridePlaying** est positionné).

Pendant la période qui précède immédiatement le déclenchement d'une insertion, il faut procéder comme suit. Si un message **Splice_Request** de priorité 5 est reçu spécifiant le même instant d'assemblage qu'un même message de priorité 3, une indication d'erreur par collision sera retournée pour la demande de priorité 3. Si un message **Splice_Request** de priorité 7 pour le même instant d'assemblage est reçu plus tard, une indication d'erreur par collision sera retournée pour la demande de priorité 3 et la demande de priorité 7 sera mise dans la file d'attente. Si une deuxième demande de priorité 7 est reçue avec l'indicateur **OverridePlaying** mis à 0, la deuxième demande de priorité 7 recevra une indication d'erreur par collision. Toutefois, si l'indicateur **OverridePlaying** est mis à 1 sur la deuxième demande de priorité 7, la demande de priorité 7 d'origine recevra une indication d'erreur par collision et sera préemptée.

La Figure 3 représente trois entrées de colleuse. La zone ombrée indique quelle source d'entrée sera dirigée vers le canal de sortie à un instant donné.



Instant t1 – Le serveur 1 émet un message **Splice_Request** et commence à envoyer son flux vers la colleuse. La colleuse commute ce flux de canal d'insertion vers le canal de sortie. Dans le message **Splice_Request,** il a été demandé une insertion de durée t1-t5. La colleuse doit envoyer au serveur 1 un message **SpliceComplete_Response** dont l'indicateur **SpliceType** est positionné à Splice_in et le code de résultat à 100, "réussite".

Instant t2 – Le serveur 2 émet un message **Splice_Request** dont l'indicateur **OverridePlaying** est mis à 1 et dont la priorité est au moins égale à celle indiquée dans le précédent message **Splice_Request**. A l'instant spécifié dans le message **Splice_Request**, le flux du canal d'insertion 2 est commuté vers le canal de sortie (se substituant au flux provenant du serveur 1). Dans le message **Splice_Request** du serveur 2 il est demandé une durée d'insertion t2-t3. La colleuse doit envoyer au serveur 1 un message **SpliceComplete_Response** dont l'indicateur **SpliceType** est mis à Splice_out et le code de résultat à 125, "préemption". La colleuse doit envoyer au serveur 2 un message **SpliceComplete_Response** avec un indicateur **SpliceType** mis à Splice_in et un code de résultat mis à 100, " réussite". Si le serveur 1 constate qu'il s'agit d'une préemption par erreur, il peut envoyer un message **Abort_Request** et arrêter alors le flux. Ce comportement n'est pas illustré à la Figure 3.

Instant t3 – La durée affectée à l'insertion est écoulée et la colleuse revient non pas au canal primaire, mais aux éléments provenant du serveur 1 pour les diriger vers le canal de sortie. La colleuse doit envoyer au serveur 1 un message **SpliceComplete_Response** dont l'indicateur **SpliceType** est mis à splice_in (début de collure) et un code de résultat égal à 125, "préemption de canal". La colleuse doit envoyer au serveur 2 un message **SpliceComplete_Response** dont l'indicateur **SpliceType** est positionné à Splice_out (fin de collure) et le code de résultat mis à 100, "réussite".

Instant t4 – Le serveur 2 émet un autre message **Splice_Request** avec l'indicateur **OverridePlaying** mis à 1. A l'instant spécifié dans le message **Splice_Request**, le flux du canal d'insertion 2 est commuté vers le canal de sortie (se substituant ainsi au flux en cours provenant du serveur 1). Dans le message **Splice_Request** du serveur 2 il est demandé une insertion allant de l'instant t4 à l'instant t6. La colleuse envoie au serveur 1 un message **SpliceComplete_Response** avec le fanion **SpliceType** mis à Splice_out et le code Result mis à 125, "canal préempté". La colleuse envoie au serveur 2 un message **SpliceComplete_Response** avec un fanion **SpliceType** mis à Splice_in et le code de résultat mis à 100, "réussite".

Instant t5 – Fin du flux d'insertion du serveur 1, deux parties de ce flux ayant été lues et deux parties ayant été préemptées par le flux du serveur 2.

Instant t6 – La durée finale d'insertion est écoulée et la colleuse revient aux éléments au canal primaire pour les diriger vers le canal de sortie. La colleuse doit envoyer au serveur 2 un message **SpliceComplete_Response** dont l'indicateur **SpliceType** est positionné à Splice_out et un code de résultat mis à 100 "réussite".

Figure 3/J.280 – Fonctionnement de l'indicateur de priorité OverridePlaying

Parfois, plusieurs serveurs veulent scinder un message **Cue_Request**. C'est le cas lorsqu'il y a, par exemple, une fenêtre d'assemblage de 60 secondes dont un serveur veut utiliser les 30 premières secondes et le deuxième les 30 dernières. En fonction des priorités et des instants de réception des messages **Splice_Request**, la colleuse devra indiquer le code de résultat 109 (collision d'assemblage) le cas échéant. La coordination de la capacité des deux serveurs à exécuter cette fonctionnalité n'est pas assurée par l'interface API définie dans la présente Recommandation. Cette fonctionnalité pourrait, par exemple, être mise en œuvre par accord entre les serveurs ou en recourant à une interface API serveur-serveur.

6.3 Fins anormales

Une insertion peut être préemptée à tout instant pendant son exécution par une insertion de priorité plus élevée. Dans ce cas, la colleuse revient à l'insertion préemptée à la fin de l'insertion de priorité plus élevée. S'il est mis fin à l'insertion de priorité plus élevée par un message **Abort_Request**, la

colleuse doit revenir à l'insertion préemptée. Si le canal d'insertion initial n'est plus disponible, la colleuse doit revenir si possible au canal primaire.

Si le serveur demande l'exécution d'une collure sur un canal primaire qui ne dispose pas d'une entrée valide, la colleuse doit exécuter la collure tout en renvoyant le code de résultat 111 (canal primaire introuvable) dans le message **SpliceComplete_Response** adressé au serveur. De même, si après l'exécution d'une collure en provenance d'un canal d'insertion le retour se fait vers un canal primaire qui ne dispose pas d'entrée valide, la colleuse renvoie le code de résultat 111 (canal primaire introuvable).

Dans la colleuse, on pourra ajouter un logiciel qui assurera toujours le retour de la colleuse au canal primaire. Il est hautement souhaitable de disposer d'une colleuse qui, en cas de problème, revienne au canal primaire en présence d'une condition d'erreur susceptible d'amener le canal de sortie à arrêter la transmission. Dans ce cas, la colleuse renverrait alors un message **SpliceComplete_Response** avec un code de résultat 110 (canal d'insertion introuvable).

6.4 Informations nécessaires pour réaliser une collure

La colleuse doit disposer d'informations concernant le canal d'insertion avant que la collure puisse être réalisée sur le canal primaire. Une partie de cette information doit être envoyée dans la connexion API et une autre peut être envoyée dans le multiplex MPEG. Ces informations sont requises avant l'exécution de la collure.

ChannelName sert à identifier le canal de sortie. Il s'agit d'un nom unique assigné à chaque canal de sortie (par exemple, CNN) lors de l'initialisation de la colleuse et il sert au serveur pour déterminer le canal primaire qui sera remplacé par chacun des canaux d'insertion.

Il est nécessaire pour la colleuse de connaître l'identité du canal d'insertion qui servira à réaliser une collure sur le canal primaire, c'est-à-dire: les coordonnées du multiplex d'insertion et l'identité du canal à utiliser dans le multiplex d'insertion. Ces informations figurent dans le message **Splice_Request**.

6.5 Communication

La communication entre le serveur et la colleuse utilise une connexion par connecteur TCP/IP pour chaque canal de sortie. Le connecteur TCP/IP est défini par l'adresse IP et le numéro de port TCP sur le serveur et la colleuse. Les adresses IP sont uniques pour chaque serveur ou colleuse et les numéros de port TCP sont uniques pour chaque programme sur lequel une collure doit être exécutée. Par conséquent, l'ensemble serveur, colleuse et connexion associée à un canal de sortie est identifié de manière unique. Après l'établissement de cette connexion API, celle-ci reste établie jusqu'à ce que l'un des dispositifs mette fin à la connexion API à un instant où une réinitialisation est nécessaire pour commencer une nouvelle collure.

Tous les messages échangés entre la colleuse et le serveur ont en commun un format général dont le détail est donné au § 7.1. Seuls les messages conformes à ce format doivent être utilisés pour la communication entre la colleuse et le serveur. Ce format permet d'utiliser une classe de messages de type "défini par l'utilisateur" qui peuvent servir de modèle pour les messages de données privés entre le serveur et la colleuse, ces messages privés se situant en dehors du domaine d'application de la présente Recommandation.

Tous les messages de demande requièrent une réponse de la colleuse ou du serveur qui dépend du dispositif auquel la demande est adressée. La plupart des messages de réponse indiquent uniquement un résultat et ne contiennent pas d'autres données. Ils sont nécessaires pour garantir au demandeur que le message a été bien reçu et bien interprété. S'il y a des erreurs, le message peut être envoyé de nouveau.

6.6 Complément d'étude

La présente Recommandation spécifie un délai minimal de trois secondes pour la collure entre la réception d'une demande de collure et l'opération réelle de collure. La définition, les paramètres et l'application d'une "demande de collure immédiate" avec un délai très inférieur à ces trois secondes feront l'objet d'un complément d'étude.

7 Syntaxe API

7.1 Syntaxe du message Splicing_API_Message

Les messages, dans l'interface API considérée, ont tous une structure générale de message qui sert à présenter les données contenues dans le message à envoyer. Ainsi, dès que le message est reçu, un utilitaire d'analyse commun stocke le message, détermine la structure des données et s'assure de la bonne réception de ce message.

Tableau 7-1/J.280 – Message Splicing_API_Message

Syntaxe	Octets	Type
Splicing_API_Message {		
MessageID	2	uimsbf
MessageSize	2	uimsbf
Result	2	uimsbf
Result_Extension	2	uimsbf
data( )	*	*
}		

MessageID – Valeur entière qui indique l'identité du message qui va être envoyé (voir le Tableau 7-2).

MessageSize – Taille en octets du champ data( ) envoyé.

Result – Suite donnée au message. Voir l'Appendice I pour les détails concernant les codes de résultat. Pour les messages de demande, ce champ a la valeur 0xFFFF.

Result_Extension – A la valeur 0xFFFF sauf si ce champ sert à envoyer des informations de résultat supplémentaire dans un message de réponse.

data( ) – Structure de données spécifique au message envoyé. Les détails concernant chacun des messages contenant des données sont décrits ci-dessous. La taille de ce champ est égale à celle indiquée dans **MessageSize** et est déterminée par la taille des données ajoutées au message. Les messages n'utilisent pas tous le champ data( ).

Tableau 7-2/J.280 – Valeurs de MessageID

Message ID	Nom du message	Expéditeur	Description
0x0000	General_Response	Colleuse ou serveur	Utilisé pour acheminer des informations asynchrones entre dispositifs. Il n'y a pas de champ data( ) associé à ce message.
0x0001	Init_Request	Serveur	Message initial à destination de la colleuse sur le port 5168.
0x0002	Init_Response	Colleuse	Réponse initiale au serveur sur la connexion établie.

Tableau 7-2/J.280 – Valeurs de MessageID

Message ID	Nom du message	Expéditeur	Description
0x0003	ExtendedData_Request	Serveur	Demande d'informations détaillées relatives à la lecture, formulée à la colleuse.
0x0004	ExtendedData_Response	Colleuse	Réponse spécifique au fournisseur sur les données de lecture élargies provenant de l'événement de lecture demandée.
0x0005	Alive_Request	Serveur	Envoi d'un message de demande d'indication de présence.
0x0006	Alive_Response	Colleuse	Réponse au message de demande d'indication de présence.
0x0007	Splice_Request	Serveur	Demande d'exécution d'une collure à un instant précis.
0x0008	Splice_Response	Colleuse	Réponse pour indiquer que le message **Splice_Request** a bien été reçu et que la colleuse se prépare à effectuer la collure.
0x0009	SpliceComplete_Response	Colleuse	Réponse au début et à la fin du collure.
0x000A	GetConfig_Request	Serveur	Demande visant à connaître la configuration actuelle de collage pour la connexion API considérée.
0x000B	GetConfig_Response	Colleuse	Contient toutes les informations de collure pour la connexion API.
0x000C	Cue_Request	Colleuse	La colleuse envoie la partie informations de repérage au serveur.
0x000D	Cue_Response	Serveur	Accusé de réception de la partie informations de repérage.
0x000E	Abort_Request	Serveur	Demande de retour immédiat au canal primaire ou au canal d'insertion préempté.
0x000F	Abort_Response	Colleuse	Accusé de réception du message **Abort_Request**. Un message **SpliceComplete_Response** doit également être généré si nécessaire.
0x0010-0x7FFF 0xFFFF	Réservé		Plage réservée à une normalisation future.
0x8000-0xFFFE	Défini par l'utilisateur		Plage disponible pour des fonctions définies par l'utilisateur.

7.2 Conventions et prescriptions

1) Chaque message qui contient des données est présenté avec ses champs et types de données ci-dessous. D'autres structures sont indiquées comme étant des fonctions et sont décrites au § 8.

2) Toutes les longueurs de chaîne de caractères ont un espace réservé pour y placer un caractère de fin zéro et doivent se terminer par des caractères zéro. Par exemple, une chaîne définie comme occupant 16 caractères contient au plus 15 caractères de données immédiatement suivis du caractère zéro (0X00). Lorsqu'on constate la présence d'un caractère zéro en analysant une chaîne, les caractères restants de la chaîne sont non définis. La taille définie pour la chaîne est constante et ne variera pas en fonction de la longueur de

la chaîne. La présente Recommandation utilise pour les chaînes des caractères ASCII à 8 bits.

3) Toutes les valeurs de temps sont exprimées en temps UTC.

4) Dans la présente Recommandation les champs entièrement constitués de "1" indiquent une condition DON'T CARE (ne pas prendre en considération). Pour un champ à 4 octets, cette valeur serait 0xFFFFFFFF.

5) Les messages de réponse doivent être envoyés sans retard inutile. En l'absence de réponse dans un délai de 5 secondes, le dispositif qui attend cette réponse doit considérer qu'il y a fin de temporisation. Lorsqu'un serveur suspecte une fin de temporisation, il doit envoyer un message Alive_Request. Si la colleuse ne répond pas comme spécifié dans la présente Recommandation, la connexion associée à ce canal doit être supprimée puis rétablie.

6) Un serveur qui reçoit un message de réponse indiquant un échec d'interprétation d'un message (code d'erreur 123) doit envoyer un message Alive_Request. S'il ne reçoit pas le message Alive_Response approprié, la connexion associée à ce canal doit être supprimée puis rétablie.

7) Le champ Result (résultat) du message Splicing_API_Message est utilisé pour retourner un code de résultat. Plusieurs codes de réponse peuvent être renvoyés au moyen de plusieurs messages **General_Response** à tout moment.

8) La colleuse ou le serveur qui ne peut pas interpréter le message Request doit envoyer un message **General_Response** avec le code de résultat 123.

7.3 Initialisation

La communication initiale commence par la mise en attente de la colleuse sur le port prédéfini 5168 et l'ouverture par le serveur d'une connexion API vers la colleuse. Le serveur envoie un message **Init_Request** vers la colleuse. Le serveur se met alors en attente d'une réponse de la colleuse sur la connexion API ainsi établie. Toutes les communications ultérieures utiliseront cette connexion API. La colleuse ou le serveur peuvent mettre fin aux communications en fermant la connexion API. Chaque dispositif est responsable de la détection et du traitement approprié d'une connexion API fermée. Lorsque la colleuse déclenche l'initialisation du dispositif de mise en attente TCP sur le port 5168, elle doit prévoir un nombre de connexions TCP avec la colleuse au moins égal au triple du nombre de canaux d'insertion. Par exemple, si la colleuse contrôle 70 canaux dont 40 peuvent faire l'objet de collures, elle devrait prévoir 120 (40×3) connexions API simultanées.

7.3.1 Message Init_Request

Le champ data( ) de ce message contient la structure Init_Request_Data suivante. Voir le Tableau 7-3.

Tableau 7-3/J.280 – Init_Request_Data

Syntaxe	Octets	Type
Init_Request_Data {		
Version( )		
ChannelName	32	Chaîne
SplicerName	32	Chaîne
Hardware_Config( )		
pour (i=0; i<N; i++)		
splice_API_descriptor( )		
}		

Version( ) – Voir le § 8.1.

ChannelName – Nom logique donné au canal de sortie associé à la connexion. Il est également utilisé pour vérifier que la connexion API est correcte lorsque la colleuse répond au serveur.

SplicerName – Nom de la colleuse lorsque le serveur utilise l'interface API pour communiquer avec un dispositif qui gère plusieurs colleuses.

Hardware_Config( ) – Voir le § 8.2.

splice_API_descriptor( ) – Descripteur qui doit être conforme à la syntaxe définie au § 8.5. Le descripteur missing_Primary_Channel_action_descripor( ) est un descripteur adapté à cette demande.

7.3.2 Message Init_Response

Après l'envoi du message **Init_Request**, la colleuse répond par un message **Init_Response** sur la connexion API ouverte. Le serveur vérifie que la version envoyée par la colleuse est bien prise en charge et qu'il dispose d'une connexion API avec le bon canal primaire.

Le champ data( ) de ce message contient la structure Init_Response_Data suivante. Voir le Tableau 7-4.

Tableau 7-4/J.280 – Init_Response_Data

Syntaxe	Octets	Type
Init_Response_Data {		
Version( )		
ChannelName	32	String
}		

Version( ) – Voir le § 8.1. La colleuse doit répondre par le numéro de version le plus élevé de l'interface API qu'elle peut prendre en charge.

ChannelName – Retourné au serveur pour indiquer que la connexion correcte a bien été établie.

7.4 Messages de repérage intégrés

Certaines colleuses peuvent disposer de la capacité de réception de messages de repérage intégrés de type Rec. UIT-T J.181. Après leur réception par la colleuse, ces messages de repérage doivent être transmis au serveur. Le message **Cue_Request** est utilisé pour transférer ces messages de repérage du serveur à la colleuse. Lorsqu'elle reçoit un message de repérage, la colleuse envoie au serveur le champ splice_info_section( ) complet en indiquant l'instant de collure au serveur. Le serveur accuse réception du message par un message **Cue_Response**. Le message **Cue_Response** se compose uniquement du message Splicing_API_Message et ne contient pas de données associées data( ) mais peut comporter un code de renvoi. Si le champ splice_info_section( ) est crypté, la colleuse le décryptera avant de l'envoyer au serveur.

Si la colleuse reçoit un message de repérage et constate que celui-ci est altéré, elle doit envoyer un message **General_Message** au serveur avec un code de résultat 117 (Message de repérage non valide). Dans ce cas, la colleuse ne doit pas envoyer de message **Cue_Request**.

7.4.1 Message Cue_Request

Le champ data( ) de ce message contient la structure Cue_Request_Data suivante. Voir le Tableau 7-5.

Tableau 7-5/J.280 – Cue_Request_Data

Syntaxe	Octets	Type

Cue_Request_Data {
time( )
splice_info_section( )
}		

time( ) – désigne l'instant spécifié dans le champ splice_time( ) de la section splice_info_section( ) du message de repérage J.181. Si le mode collage par composantes est spécifié dans la section splice_info_section J.181, time( ) doit se référer à l'instant de collure par défaut décrit en détail au § 7.5.2.1/J.181. Si cette section ne contient pas de champ pts_time( ) nécessitant une traduction comme dans la commande splice_schedule( ), la structure time doit être constituée de "1" uniquement pour indiquer qu'aucun instant n'est spécifié. Il appartient à la colleuse de déterminer la façon de mapper l'instant PTS en temps UTC pour la communication avec le serveur. Cette opération peut varier d'une colleuse à l'autre pour leur permettre de bien gérer leurs tampons internes. Voir le § 8.4 pour la syntaxe de la structure time( ).

splice_info_section( ) – Le détail de cette structure est donné dans la Rec. UIT-T J.181.

7.5 Messages relatifs au collage

Après avoir initialisé et configuré la colleuse, le serveur peut émettre un message **Splice_Request** pour déclencher une session. Les deux messages qui sont retournés à la suite du message **Splice_Request** sont les messages **Splice_Response** et **SpliceComplete_Response**. Le serveur doit envoyer un message **Splice_Request** au moins trois secondes avant l'instant spécifié dans le champ time( ) du message **Splice_Request**. La colleuse peut ainsi se configurer et se préparer à effectuer une collure. Le flux du canal d'insertion pour la session considérée doit commencer à être émis dans un délai compris entre 300 et 600 millisecondes avant l'instant time( ) tel qu'il est mesuré à l'entrée de la colleuse. Une référence d'horloge programme (PCR, *program clock reference*) doit être envoyée au moment de la première unité d'accès vidéo du flux de canal d'insertion ou avant ce moment. Le flux vidéo du contenu d'insertion doit commencer par un en-tête de séquence et une trame I. La colleuse doit autoriser la présence d'au moins dix messages **Splice_Request** en file d'attente sur une connexion API donnée. Si la file d'attente de messages de la colleuse est pleine, la colleuse répond avec le code de résultat 114 (file d'attente de la colleuse pleine).

Les détails concernant la connexion physique sont contenus dans le message **Init_Request**. Il y a deux façons de spécifier le canal multiplex d'insertion et les identificateurs de paquets PID à utiliser:

- si l'identificateur **ServiceID** n'est pas 0xFFFF dans le message **Splice_Request**, le champ ServiceID spécifie le numéro de programme dans la table PAT qui pointe sur une table PMT. Les tables PAT et PMT doivent être stables dans le canal d'insertion au moins 200 millisecondes avant l'envoi du message **Splice_Request** et doivent rester stables pendant toute la durée de la session. Elles doivent être des tables MPEG conformes avec des incréments de révision le cas échéant;
- si l'identificateur **ServiceID** est 0xFFFF, on utilise la structure splice_elementary_stream( ) (identificateur de paquets PCR, vidéo, audio et données) du message **Splice_Request**.

NOTE – Si cette méthode est utilisée, l'identificateur ServiceID doit être mis à 0xFFFF. La colleuse doit fournir un flux de transport de type MPEG-2 à la sortie du multiplex bien qu'il ne soit pas nécessaire que le multiplex d'insertion inclue l'information PSI.

L'ordre dans lequel les messages de collure sont envoyés est important. Le premier message envoyé dans le cas d'une séquence d'insertions successives doit utiliser le paramètre time( ), alors que tous les autres messages **Splice_Request** peuvent utiliser le paramètre **PriorSession**. Le numéro

PriorSession doit se référer à une session existante qui n'est pas encore terminée. Si tel n'est pas le cas, une erreur 123 est renvoyée pointant sur le champ **PriorSession** ou time( ).

Le serveur choisit les identificateurs PID des flux élémentaires dans un multiplex d'insertion. Ces identificateurs peuvent ne pas être communs aux sessions adjacentes provenant du même serveur via le même multiplex d'insertion. Cela tient au fait que les flux de sessions adjacentes se chevaucheront parfois légèrement dans le temps en raison des spécifications associées à cette interface API.

7.5.1 Message Splice_Request

Le champ data( ) de ce message contient la structure Splice_Request_Data suivante. Voir le Tableau 7-6.

Tableau 7-6/J.280 – Splice_Request_Data

Syntaxe	Octets	Type
Splice_Request_Data {		
SessionID	4	uimsbf
PriorSession	4	uimsbf
time( )		
ServiceID	2	uimsbf
If (**ServiceID** = 0xFFFF)		
{		
PcrPID	2	uimsbf
PIDCount	4	uimsbf
pour (j=0; j<**PidCount**; j++)		
splice_elementary_stream( )		
}		
Duration	4	uimsbf
SpliceEventID	4	uimsbf
PostBlack	4	uimsbf
AccessType	1	uimsbf
OverridePlaying	1	uimsbf
ReturnToPriorChannel	1	uimsbf
pour (i=0; i<N; i++)		
splice_API_descriptor( )		
}		

SessionID – Identificateur de la session. Utilisé pour distinguer cette demande des autres demandes qui ont été ou qui vont être émises. La présence de plusieurs messages **Splice_Request** parallèles portant le même identificateur **SessionID** n'est pas autorisée. Si le message **ExtendedData_Request** est émis, un message **ExtendedData_Response** doit être reçu pour l'identificateur **SessionID** considéré avant que cet identificateur ne puisse être réutilisé.

PriorSession – Ce champ permet de disposer d'une méthode simplifiée d'exécution d'insertions successives. La valeur de ce champ contiendra l'identificateur **SessionID** de la session qui le précède immédiatement. Une valeur de ce champ égale à 0xFFFFFFFF indique que cette session utilise le champ time( ) pour déclencher son insertion, et non pas l'identificateur **SessionID** de la session précédente. Ce champ aura un identificateur **SessionID** valide uniquement si la session

immédiatement précédente a pour origine le même serveur. Le champ time( ) doit être utilisé et non pas le champ **PriorSession** lorsqu'on crée des insertions successives à partir de plusieurs serveurs.

time( ) – Désigne l'instant de collure pour l'événement considéré. Ce champ sera en général le champ time( ) extrait du message **Cue_Request** renvoyé à la colleuse. Si l'événement n'a pas été déclenché par une demande **Cue_Request**, ce sera l'instant auquel le serveur veut forcer un événement de collage. Ce champ est ignoré si l'élément **PriorSession** n'est pas égal à 0xFFFFFFFF. Si la valeur n'est pas liée au message de repérage J.181, il pourra y avoir variation des instants de collure entre les colleuses en fonction de leur tampon et des modèles de collage. On se reportera au § 8.4 pour la syntaxe de la structure time( ).

ServiceID – Désigne le numéro du programme du canal dans le multiplex d'insertion, qui fera l'objet d'une collure au lieu du canal primaire. Si cet identificateur est égal à 0xFFFF, la structure splice_elementary_stream( ) et l'élément **PIDCount** doivent être présents.

PCR – Indique l'identificateur PID de la référence PCR.

PIDCount – Désigne le nombre d'identificateurs PID présents dans le canal d'insertion (à l'exclusion de l'identificateur PID du PCR).

Durée – Indique le nombre de périodes d'horloge à 90 kHz que le serveur demande à la colleuse d'insérer. Ce champ peut se substituer à la valeur de la durée J.181. Peut être mis à 0 pour indiquer que la colleuse doit se commuter sur le canal d'insertion jusqu'à ce qu'un nouveau message **Splice_Request** arrive.

SpliceEventID – Cet élément est utilisé pour relier l'événement d'insertion au message de repérage J.181 qui peut avoir provoqué l'exécution de la collure. Doit être équivalent à la structure splice_event_id de la commande splice_insert du message de repérage J.181 associé. Doit être le même pour tous les messages **Splice_Request** associés au même message de repérage J.181. Ce champ doit être égal à 0xFFFFFFFF lorsqu'un événement n'a pas été déclenché par un message de repérage J.181.

PostBlack – Nombre de périodes d'horloge à 90 kHz de vidéo noire et de silence audio à reproduire à la fin de la lecture du contenu d'insertion. L'intervalle **PostBlack** suit et n'est pas inclus dans la durée spécifiée par **Duration**. En cas d'absence de demande **PostBlack**, ce champ sera mis à 0. Le champ **PostBlack** ne doit pas être considéré comme faisant partie de l'insertion en cours de lecture pour l'indicateur **OverridePlaying**.

AccessType – Ce champ indique le type d'accès associé à la connexion considérée. Représenté par un entier allant de 0 à 9, la valeur 0 étant la priorité la plus faible et 9 la priorité la plus élevée.

OverridePlaying – Lorsque cet indicateur est mis à 0, ce message **Splice_Request** ne peut avoir la priorité sur une insertion en cours. Si cet indicateur est mis à 1, ce message **Splice_Request** doit avoir la priorité sur toute insertion en cours, de priorité égale ou inférieure. Une insertion en cours a lieu entre les points de début et de fin de collure.

ReturnToPriorChannel – Lorsque cet indicateur a la valeur 0, la colleuse ne doit pas revenir au canal principal ou au canal d'insertion préempté à la fin de l'exécution de la demande contenue dans ce message **Splice_Request**. On attend l'émission d'un nouveau message **Splice_Request** avant la fin de cette insertion. Si tel n'est pas le cas, la colleuse doit cesser la transmission sur ce canal de sortie. Lorsque cet indicateur a la valeur 1, il faut revenir au canal précédent sauf en cas de réception subséquente d'un message **Splice_Request** indiquant autre chose.

splice_API_descriptor( ) – Descripteur qui doit suivre la syntaxe définie au § 8.5. Les champs playback_descriptor( ) et muxpriority_descriptor( ) sont des descripteurs appropriés pour le présent paragraphe.

7.5.2 Message Splice_Response

Le message **Splice_Response** ne contient pas de données et indique que le message **Splice_Request** a été reçu. Le cas échéant, ce message peut contenir un code d'erreur.

7.5.3 Message SpliceComplete_Response

Le message **SpliceComplete_Response** est envoyé en début et en fin d'insertion. Il en est de même dans le cas d'insertions successives. Par exemple, si deux éléments de contenu sont lus, quatre messages **SpliceComplete_Response** sont renvoyés, un au début du premier élément de contenu, un à la fin du premier élément de contenu, un au début du deuxième élément de contenu et un à la fin du deuxième élément de contenu. Le code de résultat dans l'en-tête doit indiquer la raison de l'échec lorsque la collure a échoué afin que le serveur puisse prendre les mesures appropriées. Le début et la fin de collure sont des événements distincts qui doivent être traités comme tels. Si une collure entre deux éléments de contenu échoue, la fin de collure doit comporter une indication d'état "bon" si l'élément de contenu courant est entièrement inséré. Le message **SpliceComplete_Response** doit être envoyé immédiatement après échec d'un événement de collage et ne doit pas attendre la fin de la durée prévue du contenu inséré.

Le champ data( ) de ce message contient la structure SpliceComplete_Response_Data suivante. Voir le Tableau 7-7.

Tableau 7-7/J.280 – SpliceComplete_Response_Data

Syntaxe	Octets	Type
SpliceComplete_Response_Data {		
SessionID	4	uimsbf
SpliceTypeFlag	1	uimsbf
Bitrate	4	uimsbf
PlayedDuration	4	uimsbf
}		

SessionID – Désigne l'identificateur de session utilisé par le message **Splice_Request**.

SpliceTypeFlag – Ce champ doit être mis à 0 pour indiquer un début de collure et à 1 pour indiquer une fin de collure.

Bitrate – Pour une fin de collure, indique le débit binaire moyen de la session. Ce champ est exprimé en bits par seconde (bit/s) en incluant l'en-tête de paquet de transport pour le canal considéré.

PlayedDuration – Pour une fin de collure, indique le nombre de périodes d'horloge à 90 kHz réellement écoulé.

7.6 Messages Alive

Lorsque l'initialisation est terminée, le serveur peut envoyer des messages **Alive_Request** pour s'assurer que la colleuse est toujours présente et active. Chaque message **Alive_Response** contient une indication d'état émanant de la colleuse à destination du serveur, qui sert à indiquer l'état de fonctionnement du dispositif. En l'absence d'activité sur la connexion TCP/IP pendant les 60 secondes précédentes, un message Alive_Request doit être envoyé.

7.6.1 Message Alive_Request

Le champ data( ) du message **Alive_Request** contient la structure Alive_Request_Data suivante. Voir le Tableau 7-8.

Tableau 7-8/J.280 – Alive_Request_Data

Syntaxe	Octets	Type

Alive_Request_Data {
time( )
}		

time( ) – Indique le temps UTC actuel du dispositif d'émission contrôlé aussi proche que possible de l'envoi du message. Est conçu pour être utilisé par la colleuse et le serveur afin de vérifier la bonne synchronisation des deux systèmes. Ne permettrait pas une synchronisation suffisante entre les deux systèmes pour l'exécution d'un collage fiable, mais les réalisateurs peuvent utiliser cette information à leur gré. On se reportera au § 8.4 pour la syntaxe de la structure time( ).

7.6.2 Message Alive_Response

Le champ data( ) du message **Alive_Response** contient la structure Alive_Response_Data suivante. Voir le Tableau 7-9.

Tableau 7-9/J.280 – Alive_Response_Data

Syntaxe	Octets	Type
Alive_Response_Data {		
State	4	uimsbf
SessionID	4	uimsbf
time( )		
}		

State – Décrit l'état du canal de sortie.

Tableau 7-10/J.280 – Etats associés au message Alive_Response

Etat	Description
0x00	Pas de sortie
0x01	Sur le canal primaire
0x02	Sur le canal d'insertion

SessionID – Identificateur **SessionID** de l'insertion en cours. Valide seulement pour **State** = 0x02.

time( ) – Indique le temps UTC actuel du dispositif d'émission contrôlé aussi proche que possible de l'envoi du message. Est conçu pour être utilisé par la colleuse et le serveur afin de vérifiier la bonne synchronisation des deux systèmes. Ne permettrait pas une synchronisation suffisante entre les deux systèmes pour l'exécution d'un collage fiable, mais les réalisateurs peuvent utiliser cette information à leur gré. On se reportera au § 8.4 pour la syntaxe de la structure time( ).

7.7 Messages Data étendus

Il s'agit d'une structure définie par la colleuse pour envoyer des données détaillées au sujet de la l'exécution, en direction du serveur. Après réception du message **SpliceComplete_Response**, les données étendues peuvent être extraites au moyen d'une demande **ExtendedData_Request**. L'identificateur **SessionID** utilisé dans ce message est le même que celui qui a été utilisé pour créer la session en cours et que celui utilisé dans **SpliceComplete_Response**.

7.7.1 Message ExtendedData_Request

Le champ data( ) de ce message contient la structure ExtendedData_Request_Data suivante. Voir le Tableau 7-11.

Tableau 7-11/J.280 – ExtendedData_Request_Data

Syntaxe	Octets	Type
ExtendedData_Request_Data {		
SessionID	4	uimsbf
ExtendedDataType	4	uimsbf
}		

SessionID – Identificateur **SessionID** de la session terminée.

ExtendedDataType – Désigne le type de données contenu dans la réponse demandée émanant de la colleuse, adressée au message **ExtendedData_Response**. Cette valeur peut être mise à 0xFFFFFFFF pour indiquer qu'il s'agit du type données par défaut. La présente Recommandation réserve les valeurs 0x00000000 à 0x7FFFFFFF à une normalisation future. La plage 0x80000000 à 0xFFFFFFFE est uniquement destinée à être utilisée par le fournisseur.

7.7.2 Message ExtendedData_Response

Le serveur doit utiliser le champ **MessageSize** pour déterminer le volume de données qui doit être lu, via le message **ExtendedData_Response**.

Le champ data( ) de ce message contient la structure ExtendedData_Response_Data suivante. Voir le Tableau 7-12.

Tableau 7-12/J.280 – ExtendedData_Response_Data

Syntaxe	Octets	Type
ExtendedData_Response_Data {		
SessionID	4	uimsbf
pour (i=0;i<n;i++)		
splice_API_descriptor( )		
}		

SessionID – Désigne l'identificateur **SessionID** pour lequel ces données sont valides.

splice_API_descriptor( ) – Descripteur au format spécifié au § 8.5 qui est défini par la colleuse.

7.8 Messages Abort

Le serveur peut envoyer un message **Abort_Request** à tout moment; celui-ci provoquera le retour immédiat au canal d'insertion préempté ou au canal primaire. La colleuse doit envoyer un message **Abort_Response** pour accuser réception du message **Abort_Request**. Un message **SpliceComplete_Response** avec un code de résultat 116 (insertion interrompue) est envoyé si le message **Abort_Request** a provoqué une fin d'insertion. S'il n'est pas nécessaire d'avoir une fin de collage, il ne doit pas y avoir de message **SpliceComplete_Response**.

Toutes les insertions successives en instance liées via le champ **PriorSession** du message **Splice_Request** à l'identificateur **SessionID** d'un message **Abort_Request** doivent également être interrompues. Un message d'erreur doit être renvoyé pour chaque **SessionID** interrompu. Par exemple, supposons que trois insertions ont fait l'objet d'un repérage en vue d'être

mises en séquence pendant un certain intervalle de temps – le premier événement est de type temporel – le deuxième événement est lié au premier **SessionID** utilisant **PriorSession**; le troisième élément est lié au deuxième **SessionID** utilisant cet élément **PriorSession**. Dans cet exemple, si le premier événement d'insertion est interrompu, les deux événements d'insertion repérés subséquemment seront également interrompus. Le message d'interruption ne provoque pas l'interruption des insertions qui utilisent une connexion API différente d'un serveur à une colleuse. Pour que la collure suivante puisse avoir lieu sur le canal primaire il faut que la valeur de l'élément **PriorSession** du message de collage soit 0xFFFFFFFF.

7.9 Message Abort_Request

Le champ data( ) de ce message contient la structure Abort_Request_Data suivante. Voir le Tableau 7-13.

Tableau 7-13/J.280 – Abort_Request Data

Syntaxe	Octets	Type

Abort_Request_Data {
SessionID
}
	4	uimsbf

SessionID – Désigne l'identificateur **SessionID** et toutes les sessions subséquentes qui sont liées via le champ **PriorSession** et qui doivent être interrompues.

7.10 Message Abort_Response

Le message **Abort_Response** ne contient aucune donnée et indique que le message **Abort_Request** a été reçu. Le cas échéant, ce message peut contenir un code de résultat.

7.11 Demandes de données de configuration

Les données de configuration en cours pour la connexion API peuvent être retournés. Il faut y inclure certaines des informations se trouvant dans le message **Init_Request**. Le message **GetConfig_Request** ne contient aucune donnée supplémentaire.

7.11.1 Message GetConfig_Request

Le message **GetConfig_Request** ne contient pas de données.

7.11.2 Message GetConfig_Response

Le champ data( ) de ce message contient la structure GetConfig_Response_Data suivante. Voir le Tableau 7-14.

Tableau 7-14/J.280 – GetConfig_Response Data

Syntaxe	Octets	Type

GetConfig_Response_Data {
ChannelName
Hardware_Config( )
TS_program_map_section( )
}
	32	Chaîne

ChannelName – Nom logique donné au canal de sortie associé à la connexion considérée.

Hardware_Config( ) – Voir le § 8.2 pour la syntaxe de la structure Hardware_Config( ).

TS_program_map_section( ) – Il s'agit de la totalité de la section de la table PMT concernant le canal de sortie tel qu'il est défini dans la Rec. UIT-T H.222.0 | ISO/CEI 13818-1. Si la colleuse modifie le tableau PMT, elle doit le signaler au serveur en incluant un code de résultat 128 dans le message **General_Response**.

7.12 Message General_Response

Le message General_Response est utilisé pour acheminer des informations asynchrones entre le serveur et la colleuse. Ce message ne comporte pas de champ data( ) et n'importe quel code de résultat peut être envoyé dans ce message. Il sera en général utilisé pour indiquer au canal de sortie des modifications du tableau PMT ou la présence de messages Request non valides.

8 Structures supplémentaires

8.1 Version

La structure version est utilisée pour tenir à jour l'indication de version dans l'interface API. Cette interface évoluera avec le temps et c'est pourquoi la version est spécifiée dans les messages **Init_Request** et **Init_Response** pour s'assurer que la colleuse prend bien en charge la même version que le serveur.

Tableau 8-1/J.280 – Version( )

Syntaxe	Octets	Type
Version {		
Revision_Num	2	uimsbf
}		

Revision_Num – Ce champ est égal à 1 dans la présente version.

Le serveur et la colleuse devraient activer et vérifier ce champ pour garantir que les deux composantes peuvent fonctionner avec la révision appropriée.

8.2 Hardware_Config

Cette structure décrit une interface matérielle entre le serveur et la colleuse. Il est important pour la colleuse de connaître exactement l'endroit où se trouve connecté le serveur afin que la colleuse sache à quel multiplex il est fait référence. On peut citer comme exemple de cette liaison une connexion DVB-ASI du serveur à la colleuse.

Tableau 8-2/J.280 – Hardware_Config( )

Syntaxe	Octets	Type
Hardware_Config {		
Length	2	uimsbf
Chassis	2	uimsbf
Card	2	uimsbf
Port	2	uimsbf
Logical_Multiplex_Type	2	uimsbf
Logical_Multiplex		uimsbf
}		

Length – Indique la longueur en octets de cette structure qui suit ce champ.

Chassis – Nombre entier indiquant le numéro du châssis de la colleuse auquel le multiplex d'insertion du serveur est connecté. Lorsque la carte est identifiée alphabétiquement, une conversion est faite vers une valeur entière (c'est-à-dire A – 1; B – 2; etc.).

Card – Nombre entier indiquant la carte de la colleuse auquel le multiplex d'insertion du serveur est connecté. Lorsque la carte est identifiée alphabétiquement, une conversion est faite vers une valeur entière (c'est-à-dire A – 1; B – 2; etc.).

Port – Numéro du port matériel auquel le multiplex d'insertion du serveur est connecté.

Logical_Multiplex_Type – Valeur extraite du Tableau 8-3.

Tableau 8-3/J.280 – Type de multiplex logique

Type	Longueur	Nom	Description
0x0000	0	Non utilisé	Le champ **Logical_Multiplex** n'est pas nécessaire pour identifier le multiplex
0x0001	variable	Défini par l'utilisateur	L'utilisation du champ **Logical_Multiplex** n'est pas définie dans la présente Recommandation et doit être "décidée" en commun entre la colleuse et le serveur
0x0002	6	Adresse MAC	Le champ **Logical_Multiplex** contient l'adresse IEEE de la commande d'accès au média du multiplex sous forme d'une adresse de 6 octets
0x0003	6	Adresse IPv4	Les 4 octets de plus fort poids du champ **Logical_Multiplex** contiennent l'adresse IP du multiplex et les 2 octets restants contiennent le numéro de port IP où on peut trouver le multiplex
0x0004	18	Adresse IPv6	Les 16 octets de plus fort poids du champ **Logical_Multiplex** contiennent l'adresse IPV6 du multiplex, et les 2 octets restants contiennent le numéro de port IP où l'on peut trouver le multiplex
0x0005	5	Adresse ATM	Le champ Logical_Multiplex contient les coordonnées du circuit ATM sur lequel le multiplex est acheminé. Les 2 octets de plus fort poids du champ Logical_Multiplex contiennent l'identificateur de conduit virtuel (VPI). Les 2 octets suivants contiennent l'identificateur de canal virtuel (VCI) du circuit. L'octet de plus faible poids contient le numéro de couche AAL
0x0006	variable	Adresse IPv4 avec support du SPTS	Voir la description dans le texte qui suit ce tableau
0x0007	variable	Adresse IPv6 avec support du SPTS	Voir la description dans le texte qui suit ce tableau
0x0008-0xFFFF	variable	Réservé	Réservé à une normalisation future

Logical_Multiplex – Si le **Port** achemine plusieurs multiplex d'insertion sur une seule entrée, ce champ permet à la colleuse de déterminer lequel utiliser lorsqu'elle procède à une collure depuis ce serveur. La signification et le format de ce champ sont définis par le champ **Logical_Multiplex_Type**. Au cas où une définition non normalisée du champ **Logical_Multiplex** est requise, le champ **Logical_Multiplex_Type** doit être mis à 1 pour indiquer qu'il est défini par l'utilisateur.

Type 0x0006 – Adresse IPv4 avec support du flux de transport de programme unique (SPTS, *single program transport stream*)

Le type 0x0006 est utilisé par les serveurs VoD et Ad lorsque le remappage des identificateurs PID est irréalisable ou non souhaitable. Dans ces cas, il est souhaitable d'utiliser un flux SPTS par port UDP.

Tableau 8-4/J.280 – Structure du type 0x0006

Syntaxe	Octets	Type
Type 0x0006 structure {		
number_of_destination_ips	1	uimsbf
pour (j=0; j< **number_of_destination_ips**; j++) {		
dest_ip_address	4	uimsbf
}		
number_of_source_ips	1	uimsbf
pour (j=0; j< **number_of_source_ips**; j++) {		
source_ip_address	4	uimsbf
}		
base_port	2	uimsbf
number_of_ports	1	uimsbf
}		

number_of_destination_ips – Spécifie le nombre d'adresses **dest_ip_address**(s) qui suit. La fourchette applicable va de 1 à 32.

dest_ip_address – Désigne l'adresse IPv4 que la colleuse doit utiliser pour le contenu associé à la collure.

number_of_source_ips – Spécifie le nombre d'adresses **source_ip_address**(s) qui suit. La fourchette applicable va de 0 à 32.

source_ip_address – Désigne l'adresse ou les adresses IPv4 sources que la colleuse peut utiliser dans un groupe (join) du protocole IGMP V3 pour l'adresse ou les adresses **dest_ip_address**(s) multicast associées.

base_port – Désigne le port UDP initial que la colleuse doit utiliser pour le contenu associé à un message Splice_Request avec un instant time( ) spécifié. La fourchette applicable au port UDP de base doit être attribuée par l'IANA.

number_of_ports – Cet octet contient le nombre de ports contigus à réserver. La valeur de number_of_ports peut être comprise entre 1 et 4 et englobe le port de base. Les numéros de port autorisés sont déterminés, dans l'ordre, par le port de base, suivi du port de base +1, puis du port de base +2 et enfin, du port de base +3.

Toutes les demandes Splice_Requests qui utilisent l'instant time( ), doivent utiliser l'adresse IPv4 de base: port de base à moins que le descripteur port_selection_descriptor( ) soit utilisé. La première demande Splice_Request d'un créneau doit utiliser l'instant time( ). Les sessions ultérieures du même créneau qui utilisent aussi l'instant time( ) doivent aussi utiliser l'adresse IPv4 de base: port à moins que le descripteur port_selection_descriptor( ) soit utilisé. Les demandes splice_requests suivantes et ultérieures utilisant le champ PriorSession au lieu de l'instant time( ) doivent utiliser l'adresse IP de base et le port +1, puis le port +2 et ainsi de suite jusqu'à ce que le nombre requis de ports soit utilisé avant un retour au port de base pour la demande splice_request suivante.

Le descripteur port_selection_descriptor( ) peut être utilisé dans n'importe quelle commande Splice_Request qui dispose d'une configuration matérielle avec le champ Logical_Multiplex de type 0x0006 pour modifier le fonctionnement par défaut des ports.

Le port peut être n'importe quelle combinaison d'adresse IPv4: port valable, de type unicast ou multicast. La colleuse doit appliquer un membre join IGMP à un multicast IP.

Type 0x0007 – Adresse IPv6 avec support d'un flux de transport de programme unique (SPTS)

Le type 0x0007 est utilisé par des serveurs VoD et Ad lorsque le remappage des identificateurs PID est irréalisable ou non souhaitable. Dans ces cas, il est souhaitable d'utiliser un flux SPTS par port UDP.

Tableau 8-5/J.280 – Structure du type 0x0007

Syntaxe	Octets	Type
Type 0x0007 structure {		
number_of_destination_ips	1	uimsbf
pour (j=0; j< **number_of_destination_ips**; j++) {		
dest_ip_address	16	uimsbf
}		
number_of_source_ips	1	uimsbf
pour (j=0; j< **number_of_source_ips**; j++) {		
source_ip_address	16	uimsbf
}		
base_port	2	uimsbf
number_of_ports	1	uimsbf
}		

number_of_destination_ips – Spécifie le nombre d'adresses **dest_ip_address**(s) qui suit. La fourchette applicable va de 1 à 32.

dest_ip_address – Désigne l'adresse IPv6 que la colleuse doit utiliser pour le contenu associé à la collure.

number_of_source_ips – Spécifie le nombre d'adresses **source_ip_address**(s) qui suit. La fourchette applicable va de 0 à 32.

source_ip_address – Désigne l'adresse ou les adresses IPv6 sources que la colleuse peut utiliser dans un groupe (join) du protocole MLD V2 pour l'adresse ou les adresses **dest_ip_address**(s) multicast associées.

base_port – Désigne le port UDP initial que la colleuse doit utiliser pour le contenu associé à un message Splice_Request avec un instant time( ) spécifié. La fourchette applicable au port UDP de base doit être attribuée par l'IANA.

number_of_ports – Cet octet contient le nombre de ports contigus à réserver. La valeur de number_of_ports peut être comprise entre 1 et 4 et englobe le port de base. Les numéros de port autorisés sont déterminés, dans l'ordre, par le port de base, suivi du port de base +1, puis du port de base +2 et enfin, du port de base +3.

Toutes les demandes Splice_Requests qui utilisent l'instant time( ) doivent utiliser l'adresse IPv6 de base: port à moins que le descripteur port_selection_descriptor( ) soit utilisé. La première demande Splice_Request d'un créneau doit utiliser l'instant time( ). Les sessions ultérieures du même créneau

qui utilisent aussi l'instant time( ) doivent aussi utiliser l'adresse IPv6 de base: port à moins que le descripteur port_selection_descriptor( ) soit utilisé. Les demandes splice_requests suivantes et ultérieures utilisant le champ PriorSession au lieu de l'instant time( ) doivent utiliser l'adresse IP de base et le port +1, puis le port +2 et ainsi de suite jusqu'à ce que le nombre requis de ports soit utilisé avant un retour au port de base pour la demande splice_request suivante.

Le descripteur port_selection_descriptor( ) peut être utilisé dans n'importe quelle commande Splice_Request qui dispose d'une configuration matérielle avec le champ Logical_Multiplex de type 0x0007 pour modifier le fonctionnement par défaut des ports.

Le port peut être n'importe quelle combinaison d'adresse IPv6: port valable, de type unicast ou multicast. La colleuse doit appliquer un membre join MLD à un multicast IP.

8.3 splice_elementary_stream( )

Les identificateurs de paquet (PID) sont des identificateurs de parties du flux de transport vidéo, audio, données, etc. Cette structure est utilisée pour décrire l'un des éléments du programme dans le flux MPTS. Le message **Splice_Request** peut contenir une structure splice_elementary_stream( ) pour chaque composante du flux de transport (sauf pour l'identificateur PID PCR). Les **StreamType**s sont fondés sur les définitions du tableau PMT MPEG.

La présente Recommandation ne définit pas le mappage de plusieurs identificateurs PID audio/vidéo/données en identificateurs PID de sortie. Elle ne définit pas non plus le comportement de la colleuse lorsque plusieurs pistes audio sont présentes ou manquantes dans le canal d'insertion comparativement au canal primaire.

Tableau 8-6/J.280 – splice_elementary_stream( )

Syntaxe	Octets	Type
splice_elementary_stream {		
Length	1	uimsbf
PID	2	uimsbf
StreamType	2	uimsbf
AvgBitrate	4	uimsbf
MaxBitrate	4	uimsbf
MinBitrate	4	uimsbf
HResolution	2	uimsbf
VResolution	2	uimsbf
pour (i=0;i<N;i++)		
descriptor( )		
}		

L'identificateur PID PCR est obligatoire.

Length – Longueur totale de la structure splice_elementary_stream( ).

PID – Numéro d'identificateur PID utilisé. Il s'agit d'un champ de 2 octets (16 bits) contenant le PID de 13 bits justifié à droite comme un entier à 16 bits (0x0000 à 0x1FFF).

StreamType – Type d'identificateur PID (audio, vidéo, etc.). Ce chiffre correspond à la spécification du tableau PMT défini dans la Rec. UIT-T H.222.0 | ISO/CEI 13818-1.

AvgBitrate – Débit (bit/s) pour l'identificateur PID moyenné sur tout l'élément de contenu, qui a été utilisé pour le codage du contenu. Il est mis à la valeur 0xFFFFFFFF lorsque le débit utilisé pour le codage n'est pas connu.

MaxBitrate – Débit binaire maximal associé à l'identificateur PID. Il est mis à 0xFFFFFFFF lorsque le débit n'est pas connu.

MinBitrate – Débit binaire minimal associé à l'identificateur PID. Il est mis à 0xFFFFFFFF lorsque le débit n'est pas connu.

HResolution – Largeur en pixels des images vidéo utilisant cet identificateur PID. Il doit être mis à 0xFFFF lorsque l'identificateur PID ne contient pas d'image vidéo ou lorsque le serveur ne peut pas fournir cette valeur.

VResolution – Hauteur en pixels des images vidéo qui utilisent cet identificateur PID. Il doit être mis à 0xFFFF lorsque l'identificateur PID ne contient pas d'image vidéo ou lorsque le serveur ne peut pas fournir cette valeur.

descriptor( ) – Descripteur valide quelconque utilisé dans un tableau PMT. Lorsqu'il y a plusieurs identificateurs PID audio, la présence des descripteurs de langue tels qu'ils sont définis dans la Rec. UIT-T H.222.0 | ISO/CEI 13818-1, est obligatoire.

8.4 Définition du champ time( )

La structure time est utilisée dans la présente Recommandation pour définir divers instants d'exécution des collures.

Tableau 8-7/J.280 – time( )

Syntaxe	Octets	Type
time {		
Seconds	4	uimsbf
MicroSeconds	4	uimsbf
}		

Seconds – Nombre de secondes écoulées depuis le 1er janvier 1970, 12 h 00 UTC.

MicroSeconds – Décalage en microsecondes du champ **Seconds**.

8.5 Définition du champ splice_API_descriptor( )

Il s'agit d'un modèle permettant d'ajouter des descripteurs dans un message quelconque défini dans le présente Recommandation. Les messages **Splice_Request, ExtendedData_Response** et **Init_Request** peuvent utiliser des descripteurs. L'utilisation des descripteurs dans les messages définis dans la présente Recommandation est optionnelle. Le format général des descripteurs utilisés ici est donné dans le Tableau 8-8.

Tableau 8-8/J.280 – splice_api_descriptor( )

Syntaxe	Octets	Type
splice_API_descriptor {		
Splice_Descriptor_Tag	1	uimsbf
Descriptor_Length	1	uimsbf
Splice_API_Identifier	4	uimsbf
pour (i=0;i<n;i++)		
Private_Byte	1	uimsbf
}		

Splice_Descriptor_Tag – Valeur d'étiquette allant de 0x00 à 0xFF désignant le descripteur spécifique utilisé. Ces valeurs sont destinées à être utilisées dans la présente Recommandation. Le fournisseur pourra utiliser un identificateur **Splice_API_Identifier** spécifique pour élargir la fourchette de valeurs d'étiquette, cette méthode d'ajout de descripteurs propres au fournisseur étant aussi plus fiable.

Descriptor_Length – Indique la longueur en octets du descripteur qui suit ce champ. La taille des descripteurs est limitée à 256 octets, cette valeur est ainsi limitée à 254.

Splice_API_Identifier – Désigne un identificateur de l'organisation qui a défini le descripteur considéré. Pour tous les descripteurs figurant dans la présente Recommandation, la valeur de cet identificateur est 0x53415049 ("SAPI" ASCII). Elle a été choisie de manière à ne pas être en conflit avec des descripteurs de tout autre identificateur connu.

Private_Byte – La suite de ce descripteur est affectée à des champs de données comme exigé par le descripteur en cours de définition.

8.5.1 Définition du champ playback_descriptor( )

Le champ playback_descriptor( ) est une implémentation du champ splice_API_descriptor( ) qui est destiné à être utilisé dans le message **Splice_Request**.

Le critère d'interruption porte sur le débit de lecture défini comme étant le débit binaire du canal de sortie moyenné sur une période d'une seconde. La valeur recommandée du glissement de la fenêtre de moyennage est une seconde ou moins.

Tableau 8-9/J.280 – playback_descriptor( )

Syntaxe	Octets	Type
playback_descriptor {		
Splice_Descriptor_Tag	1	uimsbf
Descriptor_Length	1	uimsbf
Splice_API_Identifier	4	uimsbf
BitrateRule	1	uimsbf
MinPlaybackRate	4	uimsbf
}		

Splice_Descriptor_Tag – 0x01.

DescriptorLength – 0x09.

Splice_API_Identifier – 0x53415049, "SAPI" ASCII.

BitrateRule – Indicateur des règles applicables à l'élément **MinPlaybackRate**.

Tableau 8-10/J.280 – Valeurs de BitrateRule

BitrateRule	Description
0x00	Ignorer **MinPlaybackRate**
0x01	Retourner le code de résultat 127 immédiatement en utilisant le message **General_Response** si le débit de lecture devient inférieur au **MinPlaybackRate** mais ne pas provoquer d'interruption
0x02	Interruption si le débit de lecture devient inférieur à **MinPlaybackRate**
0x03	Annuler la session avant l'opération début de collure si la colleuse constate que le débit **MinPlaybackRate** ne sera pas respecté. La colleuse enverra un message **SpliceComplete_Response** ou **General_Response** avec un code de résultat 127

MinPlaybackRate – Désigne le débit binaire minimal total du canal de sortie moyenné sur une seconde pendant la durée du collage, avec lequel la lecture est possible avant que l'indicateur **BitrateRule** ne soit actionné. La valeur 0 indique qu'il n'y a pas de débit minimal.

8.5.2 Définitions du champ muxpriority_descriptor( )

Le champ muxpriority_descriptor( ) est une implémentation du descripteur splice_API_descriptor( ) qui est destiné à être utilisé dans le message **Splice_Request**.

Tableau 8-11/J.280 – muxpriority_descriptor( )

Syntaxe	Octets	Type
muxpriority_descriptor {		
Splice_Descriptor_Tag	1	uimsbf
Descriptor_Length	1	uimsbf
Splice_API_Identifier	4	uimsbf
MuxPriorityValue	1	uimsbf
}		

Splice_Descriptor_Tag – 0x02.

Descriptor_Length – 0x05.

Splice_API_Identifier – 0x53415049, "SAPI" ASCII.

MuxPriorityValue – Ce nombre va de 1 à 10 (1 étant la valeur la plus faible, 5 la valeur moyenne et 10 la valeur la plus élevée). Ce nombre modifie la valeur **MuxPriorityValue** stockée du premier canal dans la colleuse. La valeur **MuxPriorityValue** d'une valeur de 5 ne modifiera pas la priorité des canaux de sortie. Une valeur inférieure à 5 se soustraira du niveau de priorité du canal de sortie et une valeur supérieure à 5 s'ajoutera à la priorité des canaux de sortie.

L'utilisation de **MuxPriorityValue** ne garantira pas que le contenu est lu avec un certain niveau de qualité. L'effet réel de **MuxPriorityValue** dépend de la configuration de tous les multiplex "collés" et de la baisse de débit binaire total du multiplex que la colleuse doit obtenir à un instant donné. Cela dépendra également du mode de fonctionnement de la colleuse et, par conséquent, ce champ dépendra fortement du fournisseur de la colleuse.

8.5.3 Définitions du champ missing_Primary_Channel_action_descriptor( )

Ce champ est une implémentation du champ splice_API_descriptor( ) qui est destiné à être utilisé dans le message **Init_Request**.

Si le canal primaire est définitivement coupé pour une raison quelconque pendant une insertion, cela peut se traduire au niveau du décodeur par le gel de la dernière image insérée en fin d'insertion.

Ce descripteur permet de demander à la colleuse d'insérer une vidéo noire et un silence audio afin de réinitialiser le tampon du décodeur, lorsque le canal primaire n'est plus présent alors qu'il devait normalement devenir la source audio/vidéo de sortie.

Tableau 8-12/J.280 – missing_Primary_Channel_action_descriptor ( )

Syntaxe	Octets	Type
missing_Primary_Channel_action_descriptor {		
Splice_Descriptor_Tag	1	uimsbf
Descriptor_Length	1	uimsbf
Splice_API_Identifier	4	uimsbf
MissingPrimaryChannelAction	1	uimsbf
}		

Splice_Descriptor_Tag – 0x03.

DescriptorLength – 0x05.

Splice_API_Identifier – 0x53415049, "SAPI" ASCII.

MissingPrimaryChannelAction – Ce paramètre peut prendre trois valeurs possibles à savoir: 0, 1 et 2. La valeur 0 signifie "rien". La valeur 1 signifie "insérer une trame I noire et une trame de silence audio". La valeur 2 signifie "continuer de transmettre une trame noire et une trame silence jusqu'au retour du signal primaire".

8.5.4 Définitions du champ port_selection_descriptor( )

Le champ port_selection_descriptor( ) est une implémentation du champ splice_API_descriptor( ) qui doit être utilisée uniquement dans le message **Splice_Request** lorsque le champ Logical_Multiplex de type 0x0006 ou 0x0007 est utilisé dans la configuration matérielle. Si le serveur envoie un champ port_selection_descriptor( ) au cours d'une séquence d'insertions, il doit continuer à envoyer ce champ jusqu'à ce que le message Splice_Request de type temporel soit émis.

Il est possible d'utiliser le champ port_selection_descriptor( ) pour modifier le fonctionnement par défaut des ports ou sélectionner une nouvelle combinaison d'adresse IPv4 ou IPv6: port, créée de manière dynamique.

La colleuse crée de manière dynamique un port de destination si l'adresse **ps_ip_address** n'était pas définie dans la configuration matérielle. Si l'adresse **ps_ip_address** est de type multicast, la colleuse doit envoyer une demande de regroupement IGMP ou MLD dans les 400 millisecondes qui suivent l'arrivée du message Splice_Request. Le temps de latence pour l'établissement d'un groupe multicast doit être inférieur à 2 secondes comme cela est indiqué ci-après:

– arrivée du message Splice_Request au moins 3 secondes (voir § 7.5);

– le flux doit être émis moins de 600 millisecondes avant l'instant time (voir § 7.5);

– la colleuse doit émettre le message de demande de regroupement IGMP ou de regroupement MLD dans un délai inférieur à 400 millisecondes.

Tableau 8-13/J.280 – IPv4 port_selection_descriptor ( )

Syntaxe	Octets	Type
port_selection_descriptor {		
Splice_Descriptor_Tag	1	uimsbf
Descriptor_Length	1	uimsbf
Splice_API_Identifier	4	uimsbf
ps_ip_address	4	uimsbf
ps_port	2	uimsbf
ps_number_of_source_ip	1	uimsbf
pour (j=0; j< **ps_number_of_source_ip**; j++) {		
ps_source_ip_address	4	uimsbf
}		
}		

Splice_Descriptor_Tag – 0x04.

Descriptor_Length – Variable.

Splice_API_Identifier – 0x53415049, ASCII "SAPI".

ps_ip_address – Désigne l'adresse de protocole Internet IPv4 que la colleuse doit utiliser pour le contenu associé à la collure. Si cette combinaison d'adresse: port diffère de l'adresse figurant dans le tableau du champ Logical_Mux_Type 0x0006, il faut considérer qu'il s'agit d'une demande d'établissement dynamique de port.

ps_port – Désigne le port UDP que la colleuse doit utiliser pour le contenu associé à la collure. Ce numéro de port doit l'emporter sur le numéro donné par la méthode de sélection automatique de port du champ Logical_Multiplex_Type 0x0006.

ps_number_of_source_ip – Spécifie le nombre d'adresses **ps_source_ip_address**(s) qui suit. La fourchette valable va de 0 à 32.

ps_source_ip_address – Désigne la ou les adresses IPv4 sources que la colleuse doit utiliser dans un groupe (join) IGMP V3 pour l'adresse **ps_ip_address** multicast associée.

Tableau 8-14/J.280 – IPv6 port_selection_descriptor ( )

Syntaxe	Octets	Type
port_selection_descriptor {		
Splice_Descriptor_Tag	1	uimsbf
Descriptor_Length	1	uimsbf
Splice_API_Identifier	4	uimsbf
ps_ip_address	16	uimsbf
ps_port	2	uimsbf
ps_number_of_source_ip	1	uimsbf
pour (j=0; j< **ps_number_of_source_ip**; j++) {		
ps_source_ip_address	16	uimsbf
}		
}		

Splice_Descriptor_Tag – 0x05.

Descriptor_Length – Variable.

Splice_API_Identifier – 0x53415049, "SAPI" ASCII.

ps_ip_address – Désigne l'adresse de protocole Internet IPv6 que la colleuse doit utiliser pour le contenu associé à la collure. Si cette combinaison d'adresse: port diffère de l'adresse figurant dans le tableau du champ Logical_Mux_Type 0x0007, il faut considérer qu'il s'agit d'une demande d'établissement dynamique de port.

ps_port – Désigne le port UDP que la colleuse doit utiliser pour le contenu associé à la collure. Ce numéro de port doit l'emporter sur le numéro donné par la méthode de sélection automatique de port du champ Logical_Multiplex_Type 0x0007.

ps_number_of_source_ip – Spécifie le nombre d'adresses ps_source_ip_address(s) qui suit. La fourchette valable va de 0 à 32.

ps_source_ip_address – Désigne la ou les adresses IPv6 sources que la colleuse doit utiliser dans un groupe (join) IGMP V3 ou dans un groupe (join) MLD pour l'adresse ps_ip_address multicast associée.

9 Synchronisation temporelle

Une synchronisation temporelle est nécessaire en raison du transfert de temps entre le serveur et la colleuse. Le délai de transmission d'un message TCP/IP est assez imprévisible et dépend des autres machines présentes dans le réseau. En synchronisant ces machines, le temps peut être transféré entre deux machines sans se préoccuper des délais normaux de transmission sur le réseau, ce qui permet d'obtenir un collage très précis. Une méthode possible, non obligatoire, consiste à utiliser le protocole relatif au temps dans le réseau (NTP, *network time protocol*) pour maintenir la synchronisation entre le serveur et la colleuse. Les serveurs maintiennent en principe déjà une certaine synchronisation temporelle qui leur permet d'assurer le service NTP, la colleuse étant un client NTP. Un serveur NTP d'un système hôte commun du réseau peut également être utilisé étant donné que cette configuration existe généralement dans la tête du système à câble disposant d'une infrastructure de réseau.

Le système de synchronisation temporel doit pouvoir maintenir la synchronisation entre la colleuse et le serveur à ±15 ms près. On suppose que cette précision (c'est-à-dire dans les limites d'une période de trame vidéo) est suffisante pour assurer le bon fonctionnement serveur-colleuse dans la présente Recommandation. Le système peut utiliser les messages **Alive_Request/Alive_Response** pour s'assurer de la bonne synchronisation des deux dispositifs et pour signaler la perte de synchronisation à l'opérateur.

Le flux binaire représentant le canal primaire subit divers retards qui peuvent être dus au collage en amont, aux liaisons à satellite et à d'autres processus de transmission et de conditionnement; au total, ces retards peuvent aller de quelques millisecondes à plusieurs secondes. Une référence de synchronisation des flux permettant de compenser ces retards, peut être extraite des valeurs de la référence PCR acheminée dans les flux de transport MPEG-2. Toutefois, ces retards n'affectent pas la précision d'un message de repérage intégré dans le canal primaire. Le message de repérage utilise la référence PCR pour indiquer l'instant d'insertion correct, il maintient ainsi la précision d'origine relativement au contenu.

Le serveur fournissant le contenu du canal d'insertion ne connaît que le temps d'horloge (UTC, *coordinated universal time*) et les fenêtres d'insertion pour lesquelles il a été programmé ont le temps d'horloge pour référence. Toutefois, la colleuse doit indiquer l'instant réel exact où il faut commencer à délivrer le contenu.

Lorsque la colleuse reçoit le flux binaire de programme, ce flux a subi tous les retards auxquels il a pu être soumis. La colleuse peut prendre la référence PCR et la rapporter au temps d'horloge, puis envoyer un message au serveur spécifiant le temps UTC exact auquel il doit commencer à envoyer ce flux. Au niveau de la colleuse, le canal d'insertion provenant du serveur est alors exactement synchronisé avec le canal primaire, ce qui permet alors d'obtenir une collure parfaite. Tout retard additionnel qui se produit dans la colleuse n'a aucun effet, étant donné que les flux binaires d'entrée sont synchronisés.

10 Enchaînement des événements

10.1 Flux de signaux de collage DPI

Les Figures 4 et 5 donnent des détails spécifiques concernant l'utilisation et l'enchaînement des divers messages admis au niveau de l'interface API définie dans la présente Recommandation. Ces exemples ne sont pas limitatifs du point de vue des messages API.

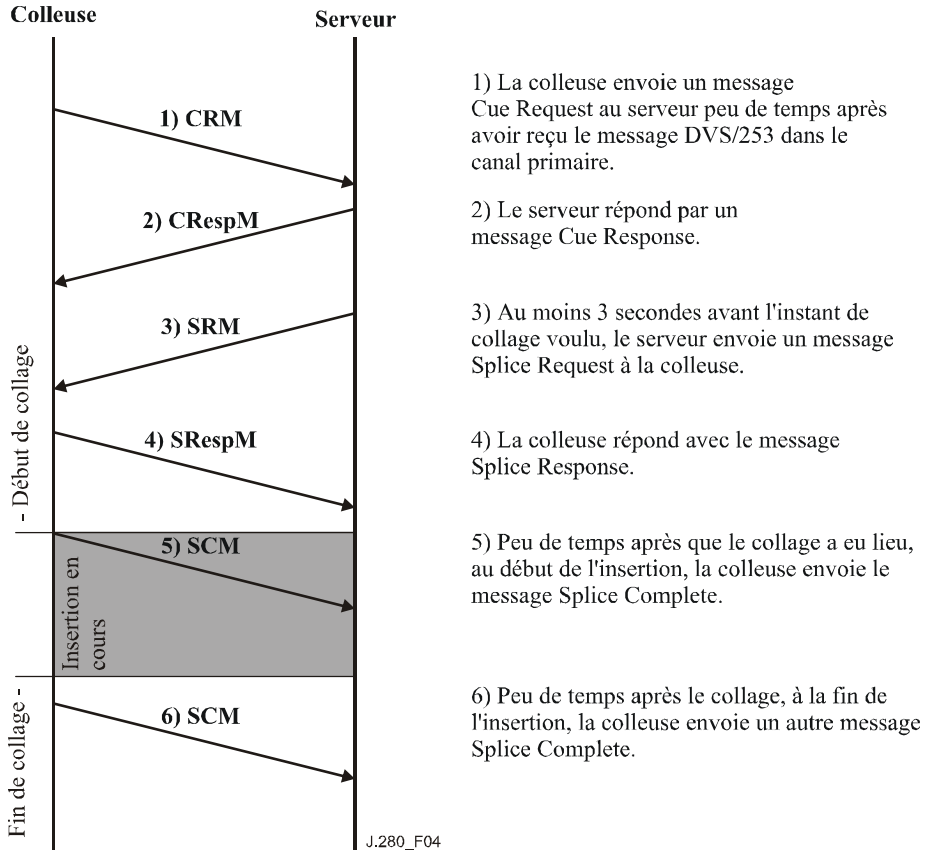


Figure 4/J.280 – Collage associé à un seul événement

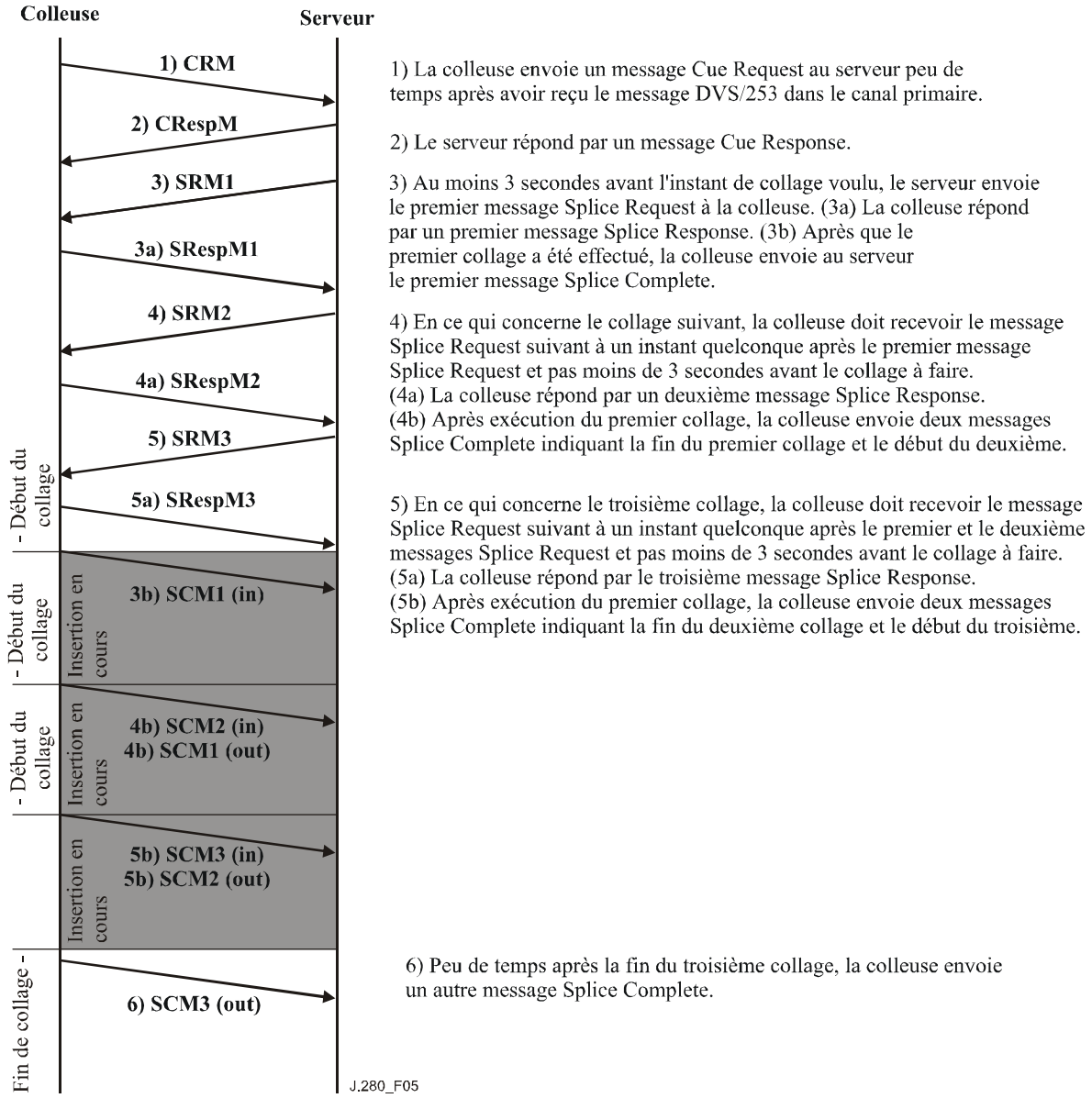


Figure 5/J.280 – Collage associé à plusieurs événements

10.2 Chronogramme de déclenchement du collage DPI

La Figure 6 contient un exemple de chronogramme des événements conduisant au début de l'insertion d'un programme (ou d'une annonce publicitaire). Dans la pratique, les chronogrammes peuvent différer de celui qui est présenté sur la figure. L'intervalle de temps utilisé permet un arbitrage de priorités comme cela est exposé au § 6.2. Le mode de fonctionnement avec messages de repérage J.181 est également montré.

Dans la figure, les droites en trait gras noir indiquent le flux d'information MPEG sur la droite canal primaire et sur la droite canal d'insertion. Les droites en trait fin noir indiquent que l'information MPEG ne circule pas à l'instant considéré ou est sans importance (c'est-à-dire non sélectionnée pour apparaître dans le canal de sortie).

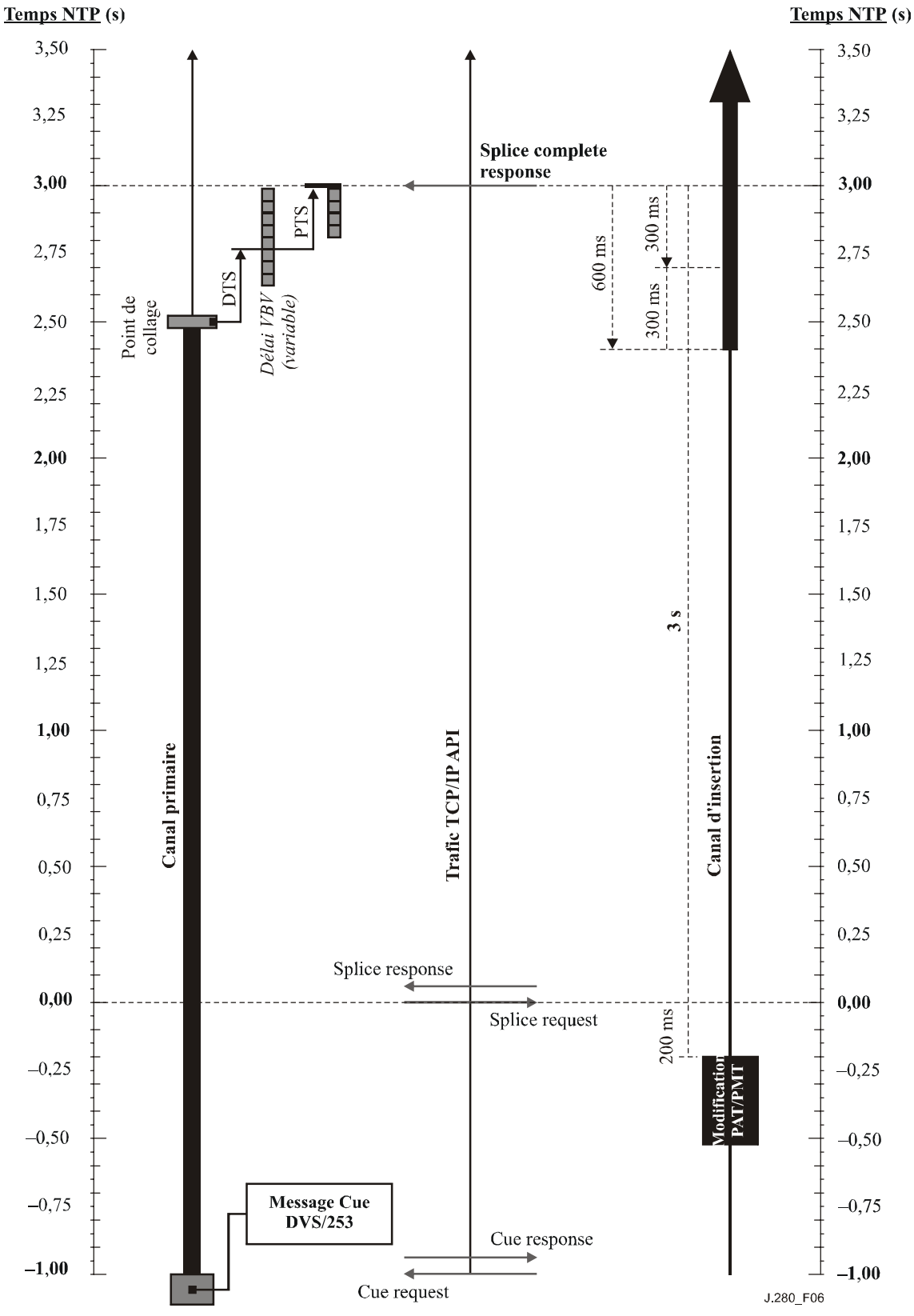


Figure 6/J.280 – Chronogramme de déclenchement d'un collage DPI

Appendice I

Codes de résultat

Code	Nom du résultat	Description	Message de réponse
100	Réponse de succès		Tous
101	Echec inconnu		Tous
102	Version non valide	Le serveur et la colleuse utilisent des versions différentes de cette API.	**Init_Response**
103	Accès refusé	Eventuel problème de licence	**Init_Response**
104	**ChannelName** non valide/inconnu	Eventuelle erreur de configuration	**Init_Response**
105	Connexion physique non valide	Eventuelle erreur de configuration	**Init_Response**
106	Configuration non trouvée	La colleuse n'a pas pu déterminer la configuration pour cette connexion	**GetConfig_Response**
107	Configuration non valide	Un ou plusieurs paramètres dans cette configuration associée à cette connexion ne sont pas valides	**GetConfig_Response**
108	Echec de collure – échec inconnu		**SpliceComplete_Response**
109	Collision de collure	Une priorité identique ou supérieure est déjà fixée pour le collure	**Splice_Response** **SpliceComplete_Response**
110	Canal d'insertion introuvable	Cette erreur doit être renvoyée si le canal d'insertion est absent au début d'une collure	**SpliceComplete_Response**
111	Canal primaire introuvable	Cette erreur doit être renvoyée si le canal primaire est absent aux instants de début ou de fin de collage	**SpliceComplete_Response**
112	Message **Splice_Request** trop tardif	Le message **Splice_Request** n'a pas été reçu suffisamment à temps (3 secondes) pour permettre à la colleuse de déclencher la réalisation de la collure	**Splice_Response**
113	Point de collage non trouvé	La colleuse n'a pas pu trouver un point valide où réaliser la collure dans le canal primaire	**SpliceComplete_Response**
114	File d'attente de collage pleine	Messages **Splice_Request** en suspens trop nombreux	**Splice_Response**
115	Lecture de session suspecte	La colleuse a détecté des différences vidéo ou audio qui peuvent avoir affecté la lecture	**SpliceComplete_Response**
116	Insertion interrompue	Un message **Abort_Request** a provoqué la fin de collure	**SpliceComplete_Response**
117	Message de repérage non valide	La colleuse ou le serveur n'a pu analyser le message de repérage	**General_Response** **Cue_Response**

Code	Nom du résultat	Description	Message de réponse
118	Le dispositif de collage n'existe pas	**SplicerName** non trouvé	**Init_Response**
119	**Init_Request** refusé	Refus de la colleuse d'autoriser le serveur à se connecter	**Init_Response**
120	**MessageID** inconnu	Utiliser le message Splicing_API_Message pour envoyer une réponse au demandeur. Renvoyer le **MessageID** inconnu	Tous
121	**SessionID** non valide	La colleuse ne connaît pas le **SessionID** spécifié	**Splice_Response** **Abort_Response** **ExtendedData_Response**
122	La session n'a pas pris fin	La colleuse n'a pas pu lire la durée totale. Inclut le cas où le serveur n'a pas fourni un contenu suffisant	**SpliceComplete_Response**
123	Request Message data( ) non valide	La colleuse ou le serveur n'ont pas pu bien interpréter un champ dans le message de demande. La position du champ non valide est renvoyée dans le champ **Result_Extension** du message Splicing_API_Message	Tous
124	Descripteur non implémenté	La colleuse ne comprend pas ou n'implémente pas actuellement le descripteur demandé	Réponses à tous les messages autorisant des descripteurs
125	Préemption de canal	Ce code de résultat est utilisé pour indiquer à l'insertion en cours de lecture qu'elle a été préemptée avec un état Splice-out ou qu'elle est revenue à l'état Splice_in	**SpliceComplete_Response**
126	Canal d'insertion ayant démarré précocement	Cette erreur peut être signalée si le canal d'insertion a commencé précocement et que la colleuse n'a pas été en mesure de déterminer le début correct du flux d'insertion	**SpliceComplete_Response**
127	Débit de lecture en dessous du seuil	Voir playback_descriptor( ) pour de plus amples détails	**SpliceComplete_Response**
128	Table PMT modifiée	Utilisé pour indiquer au serveur que le tableau PMT associé à ce canal primaire a été modifié	**General_Response**
129	Taille de message non valide	Ce message n'a pas la longueur correcte correspondant à la présente Recommandation	Tous
130	Syntaxe de message non valide	Les champs définis dans la présente Recommandation ne se trouvent pas dans la fourchette valide	Tous
131	Erreur par collision de port	La colleuse n'a pas pu utiliser l'adresse IP: combinaison de ports spécifiée qui a été demandée. La combinaison est utilisée ou n'est pas valable sur cette colleuse	**Init_Response** **General_Response**
NOTE – Tous les codes de résultat peuvent être utilisés dans le message **General_Response**.			

Appendice II

Exemple d'utilisation des champs Logical_Multiplex Type 0x0006 et port_selection_descriptor( )

II.1 Exemple informatif 1

L'exemple qui suit illustre l'utilisation de messages multiples Splice_Requests en séquence ainsi que l'incrémentation des numéros de port entre les demandes subséquentes en l'absence des champs port_selection_descriptors (voir le § 8.2).

Tous les ports sont établis statiquement à partir du message Init_Request.

Message IP de base: port = 192.168.134.9:2000

Nombre de ports = 4.

Les événements ci-après se produisent de manière séquentielle dans le temps au cours d'un seul et même créneau:

1) message Splice_Request avec time( ) set, le serveur utilise le port 2000;
2) message Splice_Request avec l'élément PriorSession, le serveur utilise le port 2001;
3) message Splice_Request avec l'élément PriorSession, le serveur utilise le port 2002;
4) message Splice_Request avec l'élément PriorSession, le serveur utilise le port 2003;
5) message Splice_Request avec l'élément PriorSession, le serveur utilise le port 2000;
6) message Splice_Request avec l'élément PriorSession, port du champ port_selection_descriptor = 2000, le serveur utilise le port 2000;
7) message Splice_Request avec l'élément PriorSession, port du champ port_selection_descriptor = 2003, le serveur utilise le port 2003.

Créneau suivant:

1) message Splice_Request avec time( ) set, le serveur utilise le port 2000.

II.2 Exemple informatif 2

Utilisation du champ port_selection_descriptor( ) pour la création dynamique d'un port. Un port de base est créé de manière statique dans le message Init_Request message.

Message IP de base: port = 192.168.134.9:3000

Nombre de ports = 1.

Les événements ci-après se produisent de manière séquentielle dans le temps au cours d'un seul et même créneau:

1) message Splice_Request avec time( ) set, le serveur utilise le port 3000;
2) message Splice_Request avec l'élément PriorSession, message IP port_selection_descriptor = 192.168.134.9 port = 2010, le serveur crée et utilise la séquence 192.168.134.9:2010;
3) message Splice_Request avec l'élément PriorSession, message IP port_selection_descriptor = 239.192.0.2 port = 2010, le serveur crée et utilise la séquence 239.192.0.2:2010.

Créneau suivant:

1) message Splice_Request avec time( ) set, le serveur utilise le port 2000.

A noter qu'il n'est pas nécessaire d'avoir le même numéro de port avec le message IP de base car chaque créneau peut assigner de manière dynamique un numéro de port comme cela est décrit dans cet exemple.

BIBLIOGRAPHIE

Liste de documents bibliographiques

- KAR (M.), NARASIMHAN (S.), PRODAN (R.): Local Commercial Insertion in the Digital Headend, *Proceedings of NCTA 2000 Conference*, Nouvelle-Orléans, Etats-Unis.
- Cable Television Laboratories: Cable Advertising, white paper, Louisville, CO, mars 1997.

Acquisitions bibliographiques

- The National Cable Television Association, 1724 Massachusetts Ave., NW, Washington, D.C. 20036-1969; Téléphone: 202-775-3669; URL: http://www.ncta.com
- CableLabs, 400 Centennial Parkway, Louisville, CO 80027; Téléphone: 303-661-9100; Télécopie: 303-661-9199; URL: http://www.cablelabs.com

SÉRIES DES RECOMMANDATIONS UIT-T

Série A	Organisation du travail de l'UIT-T
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	**Réseaux câblés et 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	Gestion des télécommunications y compris le RGT et maintenance des réseaux
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 de données, communication entre systèmes ouverts et sécurité
Série Y	Infrastructure mondiale de l'information, protocole Internet et réseaux de prochaine génération
Série Z	Langages et aspects généraux logiciels des systèmes de télécommunication

Imprimé en Suisse
Genève, 2006