Remplacée par une version plus récente

UNION INTERNATIONALE DES TÉLÉCOMMUNICATIONS

UIT-T T.200

(10/96)

SECTEUR DE LA NORMALISATION
DES TÉLÉCOMMUNICATIONS
DE L'UIT

SÉRIE T: TERMINAUX DES SERVICES TÉLÉMATIQUES

Interface de communication programmable pour équipement terminal raccordé au RNIS

Recommandation UIT-T T.200
Remplacée par une version plus récente

(Antérieurement Recommandation du CCITT)

Remplacée par une version plus récente

RECOMMANDATIONS UIT-T DE LA SÉRIE T

TERMINAUX DES SERVICES TÉLÉMATIQUES

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

Remplacée par une version plus récente

SÉRIES DES RECOMMANDATIONS UIT-T

Série A	Organisation du travail de l'UIT-T
Série B	Moyens d'expression: définitions, symboles, classification
Série C	Statistiques générales des télécommunications
Série D	Principes généraux de tarification
Série E	Exploitation générale du réseau, service téléphonique, exploitation des services et facteurs humains
Série F	Services de télécommunication non téléphoniques
Série G	Systèmes et supports de transmission, systèmes et réseaux numériques
Série H	Systèmes audiovisuels et multimédias
Série I	Réseau numérique à intégration de services
Série J	Transmission des signaux radiophoniques, télévisuels et autres signaux multimédias
Série K	Protection contre les perturbations
Série L	Construction, installation et protection des câbles et autres éléments des installations extérieures
Série M	Maintenance: systèmes de transmission, de télégraphie, de télécopie, circuits téléphoniques et circuits loués internationaux
Série N	Maintenance: circuits internationaux de transmission radiophonique et télévisuelle
Série O	Spécifications des appareils de mesure
Série P	Qualité de transmission téléphonique, installations téléphoniques et réseaux locaux
Série Q	Commutation et signalisation
Série R	Transmission télégraphique
Série S	Equipements terminaux de télégraphie
Série T	**Terminaux des services télématiques**
Série U	Commutation télégraphique
Série V	Communications de données sur le réseau téléphonique
Série X	Réseaux pour données et communication entre systèmes ouverts
Série Z	Langages de programmation

Remplacée par une version plus récente

AVANT-PROPOS

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

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

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

La Recommandation UIT-T T.200, que l'on doit à la Commission d'études 8 (1993-1996) de l'UIT-T, a été approuvée par la CMNT (Genève, 9-18 octobre 1996).

_____

NOTE

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

TABLE DES MATIÈRES

Remplacée par une version plus récente

Recommandation T.200

INTERFACE DE COMMUNICATION PROGRAMMABLE POUR ÉQUIPEMENT TERMINAL RACCORDÉ AU RNIS

(Genève, 1996)

L'existence d'applications sera un important facteur de succès du RNIS pour ceux qui utiliseront (surtout par ordinateur personnel) ce type de réseau.

Diverses spécifications ou normes nationales, régionales et de constructeurs sont apparues sur le marché ces dernières années.

L'une de ces spécifications a été élaborée par l'UIT-T et figure ici en appendice. Il est recommandé d'en tenir compte lorsque différentes solutions d'interfaces de programmation de communication seront évaluées aux fins d'implémentation.

Appendice I

Interface de programmation de communication pour équipement terminal raccordé au RNIS

Le présent appendice contient 9 parties:

Partie 1: Architecture générale

Partie 2: Services de base

Partie 3: Architecture de gestion des protocoles dans le plan d'utilisateur

Partie 4: Protocoles de couche un

Partie 5: Protocoles de couche deux

Partie 6: Protocoles de couche trois

Partie 7: Mécanisme d'échange DOS

Partie 8: Mécanisme d'échange Windows

Partie 9: Mécanisme d'échange UNIX

Remplacée par une version plus récente

TABLE DES MATIÈRES

PARTIE 1

Remplacée par une version plus récente

PARTIE 1: ARCHITECTURE GÉNÉRALE

Résumé

La disponibilité d'applications sera un important facteur de succès du RNIS pour ceux qui utiliseront (surtout par ordinateur personnel) ce type de réseau. La présente spécification constitue l'introduction de la série de spécifications qui définit une interface de programmation de communication (PCI, *programming communication interface*) normalisée, qui permettra aux applications d'avoir accès aux services fournis par un RNIS et de les gérer. Elle contient des mécanismes compatibles avec la plupart des protocoles utilisés pour assurer la communication entre applications RNIS.

Après avoir donné un aperçu général du contenu de la série constituée par chacune des parties de la présente spécification, cette partie donne en introduction une description générale de l'interface PCI.

Introduction

La présente interface entre programmes d'application (API, *application programming interface*), appelée interface de programmation de communication par réseau numérique à intégration de services (PCI-RNIS), établie par l'UIT-T pour les RNIS, permet aux applications d'avoir accès aux RNIS et de les gérer. La présente spécification fait partie d'une série de spécifications dont elle constitue l'introduction.

L'interface PCI-RNIS a été définie de façon à offrir, aux fournisseurs d'équipement terminal, une norme qui rendra possible le portage d'applications qui utilisent l'interface PCI-RNIS sur une gamme d'équipements terminaux fondés sur différents systèmes d'exploitation.

L'interface PCI-RNIS a été définie en fonction des besoins des développeurs d'applications et, dans la mesure du possible, élimine la nécessité d'une connaissance approfondie du RNIS. Elle a également été conçue de manière que les futures extensions du RNIS n'aient pas d'incidence sur le fonctionnement des applications existantes.

1 Domaine d'application

La présente partie décrit l'organisation des spécifications relatives à l'interface de programmation de communication par réseau numérique à intégration de services (PCI-RNIS).

Elle décrit la structure de la série de ces spécifications, chacune faisant l'objet d'une brève présentation.

2 Références

[1] Partie 2, *Services de base.*

[2] Partie 3, *Architecture de gestion des protocoles dans le plan d'utilisateur*.

[3] Partie 4, *Protocoles de couche un.*

[4] Partie 5, *Protocoles de couche deux.*

[5] Partie 6, *Protocoles de couche trois.*

[6] Partie 7, *Mécanisme d'échange DOS.*

[7] Partie 8, *Mécanisme d'échange Windows.*

[8] Partie 9, *Mécanisme d'échange UNIX*.

3 Définitions

La présente spécification définit les termes suivants:

3.1 plan d'administration: groupement logique de fonctions pour la gestion des dialogues entre dispositifs d'utilisateur d'interface PCI et dispositifs d'accès réseau (PUF-NAF, *PCI user facility-network access facility*) ainsi que pour l'accès à des ressources en dispositifs d'accès réseau (NAF) locales ou distantes.

3.2 mécanisme d'échange: moyen fourni pour qu'un dispositif PUF puisse échanger des messages avec un dispositif NAF.

3.3 interface RNIS de programmation de communication (PCI-RNIS): interface logicielle orientée RNIS qui offre des possibilités d'accès pour la programmation de la signalisation réseau et de l'échange de données d'utilisateur.

3.4 message: unité d'information transférée de part et d'autre de l'interface PCI-RNIS, entre le dispositif d'accès réseau (NAF) et le dispositif utilisateur d'interface PCI (PUF).

3.5 dispositif d'accès réseau (NAF): unité fonctionnelle située entre l'interface PCI-RNIS et les couches associées au réseau.

3.6 dispositif utilisateur d'interface PCI (PUF): unité fonctionnelle faisant appel à l'interface PCI-RNIS pour accéder à un dispositif NAF, par exemple l'application locale qui utilise l'interface.

3.7 plan d'utilisateur: groupement logique de fonctions d'accès offertes aux protocoles et données d'utilisateur.

3.8 protocole d'utilisateur: protocole exploité conformément aux spécificités du plan d'utilisateur.

4 Abréviations

La présente spécification utilise les abréviations suivantes:

API	interface de programmation d'application (*application programming interface*)
NAF	dispositif d'accès réseau (*network access facility*)
NCO	objet de connexion au réseau (*network connection object*)
PCI	interface de programmation de communication (*programming communication interface*)
PUF	dispositif utilisateur d'interface PCI (*programming communication interface user facility*)
RNIS	réseau numérique à intégration de services

5 Aperçu général de la série des spécifications relatives à l'interface PCI pour RNIS

La présente spécification est destinée à faciliter la compréhension, par les développeurs de logiciels, par les implémenteurs d'applications et par les équipementiers, de l'organisation de la série des spécifications relatives à l'interface PCI-RNIS.

La Figure 1 montre les relations entre les spécifications PCI-RNIS.

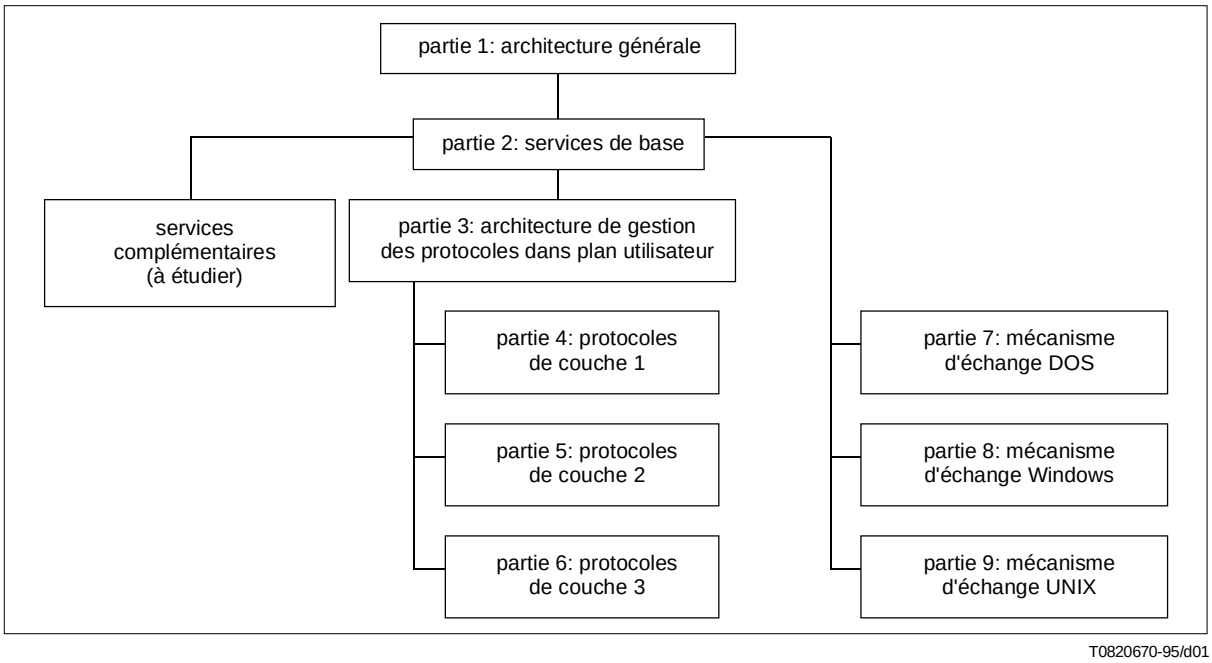


Figure 1 – Organigramme de l'interface PCI-RNIS

– La Partie 2, "Services de base" [1], contient l'ensemble commun d'informations pour toutes les parties. Il convient de lire cette deuxième partie en premier. Elle indique la façon de manipuler un objet de connexion au réseau (NCO, *network connection object*) et l'ensemble général des paramètres qui sont applicables à cet objet ainsi que le moyen d'établir une connexion RNIS. Elle donne la description du mécanisme d'échange afin de faciliter la sélection et l'échange de messages avec un dispositif NAF. En outre, elle montre le mode d'emploi d'équipements externes, par exemple de téléphonie. Les autres parties de la série PCI-RNIS font de fréquentes références au contenu de la présente partie.

NOTE – Un mécanisme d'échange relatif à la désignation du numéro de version est défini dans la série PCI-RNIS. Le numéro de version correspondant à la spécification en question est 2.

– La Partie 3, "Architecture de gestion des protocoles dans le plan d'utilisateur" [2], décrit l'emploi de protocoles du plan d'utilisateur pour un canal D ou B du RNIS. Elle décrit l'emplacement de chaque protocole dans la structure du modèle OSI. Elle décrit également la manière de sélectionner un protocole particulier, au moyen des paramètres NCOtype et UProtocol. Le concept de fonction de coordination est expliqué en détail dans la Partie 3. Elle présente des règles générales sous une forme indépendante des protocoles, par exemple les critères de sélection d'un objet NCO ou la gestion des erreurs.

– La Partie 4, "Protocoles de couche 1" [3], décrit l'emploi des protocoles du plan d'utilisateur situés dans la couche 1, dans un canal B. Elle indique la façon d'accéder en transparence aux octets échangés dans ce canal.

– La Partie 5, "Protocoles de couche 2" [4], décrit l'emploi des protocoles du plan d'utilisateur situés dans la couche deux, dans un canal B. Elle indique la façon d'accéder aux protocoles PPP (point à point), SDLC (commande de liaison de données synchrone), HDLC (commande de liaison de données à haut niveau), avec ou sans mise en trames et protocoles V.110 (bourrage d'octets), ainsi que la manière d'utiliser ces protocoles. Elle montre l'utilisation des messages et des paramètres ainsi que leurs valeurs, le séquencement et le mappage des messages appropriés en mode protocole par protocole, les règles de gestion d'erreurs et certaines informations relatives à la configuration.

– La Partie 6, "Protocoles de couche 3" [5], décrit l'emploi des protocoles du plan d'utilisateur situés dans la couche trois, dans un canal B. Elle indique la façon d'accéder aux protocoles ISO 8208, T.90 et T.70, ainsi que la manière de les utiliser. Elle décrit l'application de la fonction de coordination, l'utilisation des messages et des paramètres ainsi que leurs valeurs, le séquencement et le mappage des messages appropriés en mode protocole par protocole, les règles de gestion d'erreurs et certaines informations relatives à la configuration.

– La Partie 7, "Mécanisme d'échange DOS" [6], contient les informations relatives au système d'exploitation DOS pour la méthode d'échange décrite en [2]. Elle explique la manière d'accéder à un dispositif NAF et de l'utiliser et en quoi consistent les données relatives au DOS, avec un exemple d'implémentation utilisant le langage C.

– La Partie 8, "Mécanisme d'échange Windows" [7], contient les informations relatives au système d'exploitation Windows™ pour la méthode d'échange décrite en [2]. Elle explique la manière d'accéder à un dispositif NAF et de l'utiliser et en quoi consistent les données relatives à Windows, avec un exemple d'implémentation utilisant le langage C.

– La Partie 9, "Mécanisme d'échange UNIX" [8], contient les informations relatives au système d'exploitation UNIX™ pour la méthode d'échange décrite en [2]. Elle explique la manière d'accéder à un dispositif NAF et de l'utiliser et en quoi consistent les données relatives à UNIX, avec un exemple d'implémentation utilisant le langage C.

Remplacée par une version plus récente

TABLE DES MATIÈRES

PARTIE 2

PARTIE 2: SERVICES DE BASE

Résumé

La présente partie de la spécification donne un aperçu général de nature technique et définit les fonctions de base pouvant être offertes par l'interface PCI-RNIS. Elle définit l'architecture de l'interface PCI-RNIS et comporte une définition détaillée des messages et paramètres d'interface PCI qui sont utilisés dans les plans d'administration et de commande de connexion. Elle explique la façon d'utiliser ces messages et ces paramètres par l'intermédiaire d'un mécanisme d'échange générique.

1 Domaine d'application

La présente partie appartient à la série de textes relatifs à l'interface de programmation de communication par réseau numérique à intégration de services (PCI-RNIS) pour accéder aux services RNIS énumérés en [1] et les administrer. Elle spécifie également les "fonctions de base" offertes par l'interface PCI-RNIS.

Les fonctions de base décrites dans la présente partie spécifient:

- une interface entre plans d'administration et de commande pour les applications nécessitant une commande directe des services RNIS;
- les moyens permettant à une application d'avoir accès à plusieurs canaux par de multiples accès RNIS;
- les moyens permettant d'exploiter des applications concurrentes;
- les mécanismes généraux permettant d'appliquer des piles protocolaires multiples et concurrentes, concernant les échanges de données;
- les liens avec des environnements courants en termes de système d'exploitation;
- l'accès aux dispositifs de sécurité via l'interface.

2 Références

[1] Recommandation UIT-T Q.931 (1993), *Spécification de la couche 3 de l'interface usager-réseau RNIS pour la commande de l'appel de base.*

[2] Recommandation UIT-T X.213 (1995), *Technologies de l'information – Interconnexion des systèmes ouverts – Définition du service de réseau.*

[3] Partie 1, *Architecture générale.*

[4] Partie 3, *Architecture de gestion des protocoles dans le plan d'utilisateur.*

[5] Partie 4, *Protocoles de couche un.*

[6] Partie 5, *Protocoles de couche deux.*

[7] Partie 6, *Protocoles de couche trois.*

[8] Partie 7, *Mécanisme d'échange DOS.*

[9] Partie 8, *Mécanisme d'échange Windows.*

[10] Partie 9, *Mécanisme d'échange UNIX*.

On trouvera in fine, sous la rubrique Bibliographie, d'autres références à des publications pouvant faciliter la lecture de la présente partie.

3 Définitions

La présente partie définit les termes suivants:

3.1 **ensemble d'adresses**: ensemble de paramètres contenant des adresses distantes et locales d'utilisation ou de signalisation.

3.2 **plan d'administration**: groupement logique de fonctions pour la gestion des dialogues entre dispositifs d'utilisateur d'interface PCI et dispositifs d'accès réseau (PUF-NAF) ainsi que pour l'accès à des ressources en dispositifs d'accès réseau (NAF) locales ou distantes.

3.3 **ensemble d'attributs**: ensemble de paramètres nécessaires au fonctionnement des protocoles d'utilisation et à la signalisation RNIS.

3.4 **canal B**: voie logique RNIS utilisée pour le transfert de données.

3.5 **plan de commande**: groupement logique de fonctions pour l'accès à la signalisation RNIS.

3.6 **voie D**: voie logique RNIS utilisée pour la signalisation et, dans certains cas, pour le transfert de données.

3.7 **fonction d'échange**: propriété du dispositif PUF, réalisant le processus d'échange.

3.8 **mécanisme d'échange**: moyen fourni pour qu'un dispositif PUF puisse échanger des messages avec un dispositif NAF.

3.9 **accès RNIS**: ensemble de canaux RNIS fourni par un même dispositif d'accès réseau (NAF) pour l'accès aux services RNIS.

3.10 **interface RNIS de programmation de communication (PCI-RNIS)**: interface logicielle orientée RNIS qui offre des possibilités d'accès de programmer la signalisation de réseau et le transfert de données d'utilisateur.

3.11 **message**: unité d'information transférée de part et d'autre de l'interface PCI-RNIS, entre le dispositif d'accès réseau (NAF) et le dispositif utilisateur d'interface PCI (PUF).

3.12 **dispositif d'accès réseau (NAF)**: unité fonctionnelle située entre l'interface PCI-RNIS et les couches associées au réseau.

3.13 **objet de connexion au réseau (NCO,** *network connection object***)**: objet abstrait contenu dans le dispositif NAF, qui doit être créé par le dispositif PUF pour donner accès à la signalisation ou aux données du réseau.

3.14 **couche vide**: couche vide dans le modèle de référence OSI. Une telle couche ne contient aucune fonction et transmet en transparence les requêtes et les réponses aux couches adjacentes.

3.15 **dispositif utilisateur d'interface PCI (PUF,** *PCI user facility***)**: unité fonctionnelle faisant appel à l'interface PCI-RNIS pour accéder à un dispositif NAF, par exemple l'application locale qui utilise l'interface.

3.16 **codage type-longueur-valeur (codage TLV)**: mode de codage utilisé pour la présentation binaire de messages.

3.17 **connexion d'utilisateur**: connexion accessible par l'intermédiaire de la propriété plan d'utilisateur.

3.18 **plan d'utilisateur**: groupement logique de fonctions d'accès offertes aux protocoles et aux données d'utilisateur.

3.19 **protocole utilisateur**: protocole exploité conformément à la propriété de plan d'utilisateur.

4 Abréviations

La présente partie utilise les abréviations suivantes:

API	interface de programmation d'application (*application programming interface*)
CONS	service réseau en mode connexion (*connection oriented network service*)
HLC	compatibilité de couches supérieures (*high layer compatibility*)
IUT	instance sous test (en l'occurrence, couche de protocole soumise à un teste) (*implementation under test)*
LAPB	procédure d'accès à la liaison en mode équilibré (*link access procedure balanced*)
LAPD	procédure d'accès à la liaison sur le canal D (*link access procedure for D-channel*)
LLC	compatibilité de couches inférieures (*low layer compatibility*)
NAF	dispositif d'accès réseau (*network access facility*)
NCO	objet de connexion au réseau (*network connection object*)
PCI	interface de programmation de communication (*programming communication interface*)
PciMPB	bloc de paramètres pour messages d'interface PCI (*Pci message parameter block*)
PUF	dispositif utilisateur d'interface PCI (*PCI user facility*)
RNIS	réseau numérique à intégration de services
TLV	codage type-longueur-valeur (utilisé pour la présentation de messages d'interface PCI-RNIS) (type-length-value coding)
X.25 PLP	protocole de couche paquet X.25 (*X.25 packet layer protocol*)

5 Modèle fonctionnel

5.1 Introduction

Le présent paragraphe décrit le modèle fonctionnel pour l'interface PCI-RNIS, dont il présente l'architecture. Il décrit également les propriétés de l'interface PCI-RNIS et les interactions entre les entités situées autour de cette interface. Il indique le séquencement des messages, pour préciser de quelle façon les entités peuvent échanger des informations.

Le présent paragraphe décrit aussi les identificateurs reconnus par l'interface PCI-RNIS ainsi que le mécanisme de traitement des erreurs qu'elle offre.

5.2 Architecture

L'interface PCI-RNIS est la spécification de la jonction de communication qui se trouve à l'intérieur d'un équipement terminal demandant l'accès à un RNIS. L'emploi de cette interface permet à une entité de couche supérieure d'accéder de manière normalisée aux services offerts par un réseau de type RNIS.

L'interface PCI-RNIS est une jonction logicielle entre un utilisateur de service et un fournisseur de service. En tant que jonction logicielle, l'interface PCI-RNIS se compose de la spécification de l'interface et de la description de la propriété située immédiatement au-dessous de l'interface.

L'interface PCI-RNIS est une spécification d'interface qui est implémentée dans un environnement informatique concret. Cet environnement pose des problèmes tels que l'association des entités et l'échange d'informations entre ces entités. Il en résulte que l'interface PCI-RNIS possède certaines caractéristiques lui permettant de traiter les problèmes posés par son implémentation dans un environnement informatique.

On peut distinguer deux entités de part et d'autre de l'interface PCI-RNIS: l'utilisateur de service et le fournisseur de service. Ces entités sont décrites au 5.2.1, de même que l'interface PCI-RNIS et les échanges d'informations entre ces entités.

5.2.1 Interface PCI-RNIS – Constituants

Le présent sous-paragraphe décrit les constituants fonctionnels qui correspondent à la définition de l'interface PCI-RNIS. La Figure 1 décrit les relations entre ces constituants.

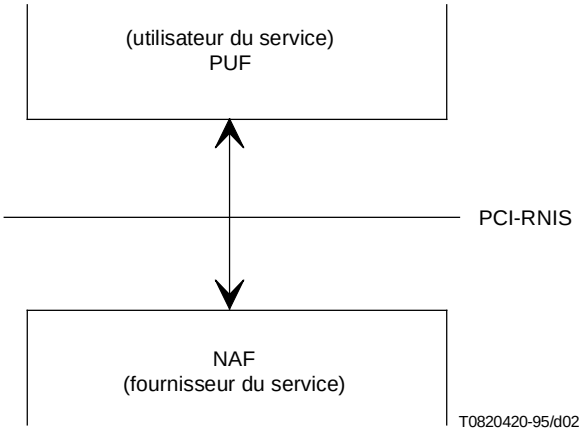


Figure 1 – Représentation fonctionnelle de l'interface PCI-RNIS avec ses constituants environnants

Les constituants fonctionnels qui constituent l'interface PCI-RNIS sont les suivants:

Dispositif PUF

Dans l'ensemble de la présente partie, le terme dispositif d'utilisateur d'interface PCI (PUF, *PCI user facility*) sera utilisé pour désigner l'utilisateur du service. Il englobe toutes les couches fonctionnelles qui utilisent l'interface pour accéder aux services du RNIS.

Dispositif NAF

Le terme dispositif d'accès au réseau (NAF, *network access facility*) est utilisé pour désigner le fournisseur de service par interface PCI-RNIS. Ce fournisseur englobe tous les éléments qui sont nécessaires pour donner accès aux services du RNIS. Ces éléments peuvent être de nature logicielle comme de nature matérielle. Aucune distinction n'est faite à ce propos. Le dispositif NAF se comporte comme s'il représentait les services d'un même accès RNIS.

Interface PCI-RNIS

L'interface PCI-RNIS correspond à la jonction située immédiatement au-dessus du (des) dispositif(s) NAF. L'interface PCI-RNIS possède un certain nombre de fonctions. Tout d'abord, l'interface PCI-RNIS permet l'association entre les dispositifs PUF et NAF. Une fois ces derniers associés, toutes les opérations sont effectuées par un mécanisme d'échange d'informations. Ce mécanisme fait également partie des propriétés de l'interface PCI-RNIS. La Figure 1 décrit comment se situe l'interface PCI-RNIS par rapport aux constituants environnants. Les flèches indiquent les flux d'information.

Messages

L'accès aux fonctions décrites pour l'interface PCI-RNIS s'effectue au moyen de messages. Les dispositifs PUF et NAF utilisent les propriétés du mécanisme d'échange de messages pour transférer leurs messages. Ceux-ci informent les entités des opérations à effectuer ou des résultats des opérations effectuées.

5.2.2 Interface PCI-RNIS – Architecture

L'interface PCI-RNIS possède sa propre structure, décrite dans la Figure 2 et composée de trois plans formant trois propriétés fonctionnelles distinctes. Chaque plan a son propre ensemble de messages. L'interface PCI-RNIS se compose des trois plans suivants:

- **plan de commande**

  le plan de commande se rapporte à la partie sémaphore d'une connexion, c'est-à-dire au dispositif NAF associé à la signalisation par le canal D du RNIS. Ce plan correspond à la propriété offerte par le service du canal D, comme la commande de connexion, la commande de caractéristiques de service, la commande de services complémentaires. Le plan de commande est également chargé de gérer des équipements spéciaux, qui sont accessibles par l'intermédiaire de l'interface PCI-RNIS;

- **plan d'utilisateur**

  le plan d'utilisateur se rapporte à la connexion d'utilisateur, qui peut être soit associée à une connexion par canal B ou être une connexion de transfert de données par le canal D. Ce plan est associé, par l'intermédiaire du dispositif NAF, à la propriété offerte par les services de transfert de données par canaux D et B, c'est-à-dire des services d'échange de données de bout en bout;

- **plan d'administration**

  le plan d'administration ne se rapporte pas au RNIS. Il correspond à la propriété requise pour commander et configurer le plan de commande et le plan d'utilisateur.

La Figure 2 donne une représentation de ces trois plans.

5.2.3 Cas de coordination

L'interface PCI-RNIS propose deux mécanismes pour coordonner les propriétés associées à la signalisation RNIS et à la connexion d'utilisateur.

Dans le cas de la coordination associée au dispositif PUF, celui-ci doit assurer l'établissement d'une connexion d'utilisateur au moyen de la commande d'appel de base offerte par le plan de commande. Le fait que le dispositif PUF puisse utiliser les services complémentaires découle de la commande de la connexion sémaphore.

Dans le cas de la coordination associée au dispositif NAF, une fonction de coordination effectue une opération abstraite, consistant à mapper les primitives du protocole CONS (X.213) [2], situé dans le plan d'utilisateur, avec les primitives du plan de commande et avec les protocoles du plan d'utilisateur. La Partie 3 [4] décrit en détail les conditions et procédures d'emploi de cette fonction de coordination. Etant donné que le dispositif NAF gère la coordination entre connexions sémaphore et d'utilisateur, le dispositif PUF ne doit pas accéder au plan de commande.

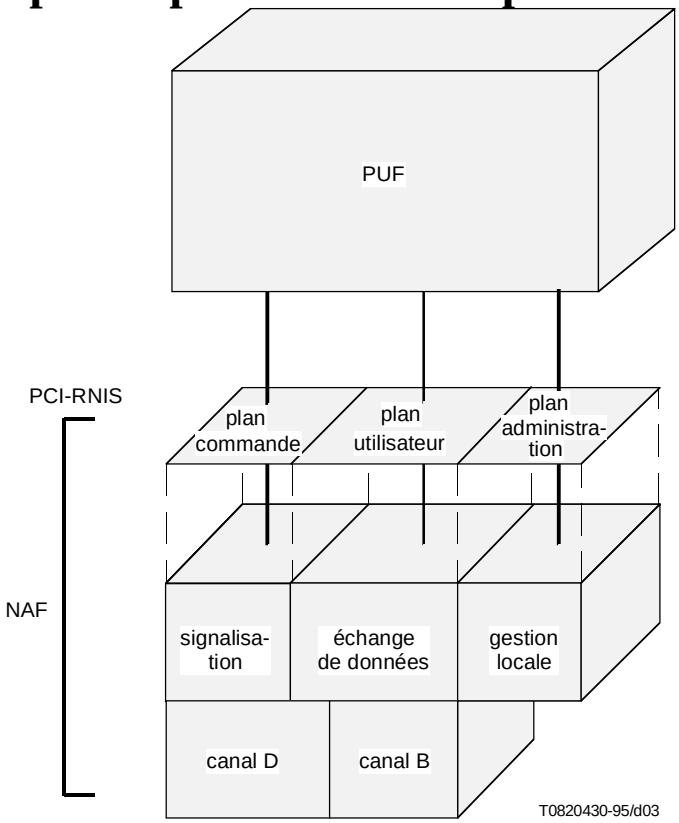


Figure 2 – Relation entre plans et RNIS

5.3 Propriétés

5.3.1 Introduction

Comme décrit au 5.2.2, la propriété d'interface PCI-RNIS est offerte par les trois plans, avec les ensembles de messages associés pour assurer les accès à cette propriété. Le paragraphe 5.5 décrit la façon dont les dispositifs PUF et NAF échangent leurs messages.

Pour accéder à la signalisation RNIS ou transférer des données, le dispositif PUF doit demander au NAF de créer un objet NCO. La création et la destruction des connexions de couche réseau sont les principales fonctions de la propriété de gestion de ressources.

Après avoir effectué les opérations précédentes, le dispositif PUF revient à l'état de "repos" et peut de nouveau accéder à la signalisation RNIS (sauf en cas de coordination par dispositif NAF) ou transférer des données. Les sous-paragraphes 5.3.3 et 5.3.4 décrivent, respectivement, la propriété de gestion de connexion et la propriété de gestion des données.

5.3.2 Gestion de ressource

La propriété de gestion de ressource permet d'utiliser l'interface PCI-RNIS pour la communication. La gestion de ressource contient la propriété de gestion locale. Cette gestion s'applique:

– aux objets de connexion réseau (objets NCO);

– aux équipements externes.

Le plan d'administration de l'interface PCI-RNIS possède la propriété définie pour la gestion de ressource.

La gestion de ressource se développe sur la base de l'objet NCO, qui est nécessaire pour la suite de la communication. Un objet NCO correspond à un objet abstrait contenant toutes les informations de configuration applicables à une connexion d'utilisateur déterminée. Les informations de configuration contenues dans un objet NCO doivent être assignées par le dispositif PUF au moyen d'une des deux principales méthodes suivantes:

a) par une référence à un ensemble d'attributs normalisés;

b) par la spécification de toutes les informations de configuration au cours de la création des objets NCO.

La méthode a) offre au dispositif PUF un moyen simple pour sélectionner les informations appropriées, par référence à un identificateur normalisé d'ensemble d'attributs. Cette méthode vient toutefois au détriment de la flexibilité puisque les ensembles d'attributs sont normalisés et ne peuvent être utilisés que de la manière prévue. L'Annexe C de la présente partie donne la liste des ensembles d'attributs normalisés pour le plan de commande.

La méthode b) donne au dispositif PUF la possibilité de spécifier des informations de configuration pour ses propres besoins, le cas échéant. Cela implique toutefois l'indication d'un grand nombre de détails concernant les paramètres des canaux D et B. Cette méthode doit donc être réservée aux implémenteurs de dispositifs PUF complexes.

5.3.2.1 Ensembles d'attributs

Les ensembles d'attributs servent à grouper d'importants paramètres pour configurer des protocoles d'utilisation, pour exécuter le protocole de signalisation RNIS et pour recueillir certaines informations de gestion se rapportant aux objets NCO (statistiques, coûts, etc.). Les protocoles d'utilisation et la signalisation RNIS sont commandés par l'intermédiaire des propriétés de plan d'utilisateur et de plan de commande. Ceux-ci sont associés à divers ensembles d'attributs, comme suit:

– ensemble d'attributs de signalisation (associé au plan de commande);

– ensemble d'attributs de protocole utilisateur (associé au plan d'utilisateur);

– ensemble d'attributs d'administration (associé au plan d'administration).

L'ensemble d'attributs d'administration n'est pas associé à la création d'un objet NCO mais n'est mis à jour que pendant qu'un objet NCO existe; on peut y accéder à tout moment, au moyen de la propriété de gestion de ressource.

La gestion de ressource offre la propriété de faire référence à des ensembles d'attributs spécifiques au moment de la création d'un objet NCO.

5.3.2.2 Objets de connexion réseau

La propriété de gestion de ressource est applicable:

– à la création d'un objet NCO;

– au groupement d'objets NCO;

– à l'extraction d'informations au sujet d'un objet NCO.

Un objet NCO est un objet abstrait créé par le dispositif NAF en réponse à des requêtes issues du dispositif PUF avant l'établissement d'une connexion.

En règle générale, il existe un seul objet NCO par connexion, indépendamment du type de connexion auquel l'objet est associé. Il peut s'agir d'une connexion sémaphore ou d'une connexion pour transfert de données.

Après création normale d'un objet NCO, un identificateur unique, l'identificateur NCOID, devient disponible. Cet identificateur NCOID doit être fourni au cours d'opérations ultérieures concernant l'établissement d'une connexion et le transfert de données.

Au moment de la création d'un objet NCO, le dispositif PUF peut indiquer que l'objet NCO qui vient d'être créé doit être associé à un autre objet NCO existant déjà.

Ce groupage a pour but de permettre le partage d'un canal lors de l'utilisation d'un protocole de couche Réseau, afin d'insérer plusieurs connexions logiques dans une même voie physique. Le partage est une fonction réservée à un seul dispositif PUF.

La fonction de groupage d'objets dépend du protocole de plan utilisateur. La Partie 3 donne une description détaillée de la condition et des procédures permettant d'utiliser la fonction de groupage d'objets.

Le groupage d'objets NCO est assuré au moyen d'un identificateur de groupe. Un unique identificateur de groupe doit être retourné après création normale d'un objet NCO. Cet identificateur de groupe pourra ensuite être fourni au moment de la création d'un autre objet NCO, qui sera alors groupé avec le premier objet NCO. Si aucun identificateur de groupe n'est fourni, l'objet NCO ne doit pas être groupé. L'identificateur de groupe n'est garanti unique que pour l'interaction entre dispositifs PUF et NAF.

Etant donné que l'identificateur de groupe n'est unique que pour la relation PUF-NAF, plusieurs dispositifs PUF ayant accès au même dispositif NAF ne peuvent pas partager la même connexion.

Pour un appel entrant, le dispositif NAF sélectionne les objets NCO appropriés puis reçoit l'assistance du dispositif PUF pour choisir l'identificateur unique. Cette opération s'effectue au moyen de l'identificateur SelectorID, qui est fourni à la création de l'objet NCO. Cet identificateur permet au dispositif PUF de gérer une liste d'objets NCO que le dispositif NAF traitera de façon exclusive.

En cas d'objet NCO non coordonné (C/U3), le plan d'utilisateur et le plan de commande peuvent avoir des sens différents. Par exemple, le plan d'utilisateur peut être en écoute, tandis que le plan de commande est en appel.

5.3.2.3 Compatibilité avec des équipements externes

L'accès à des équipements externes tels que des postes téléphoniques est donné au dispositif PUF au moyen des propriétés des trois plans.

Tant qu'existe un objet NCO spécifiant, dans ses informations de configuration, un équipement externe, le dispositif NAF doit produire les messages appropriés du plan de commande si l'état de cet équipement externe change.

La gestion des connexions (voir 5.3.3) et la gestion des données (voir 5.3.4) offrent les propriétés de gestion des connexions établies avec ces objets NCO.

Cinq types d'équipements externes sont définis:

1) équipement externe sans commande de décrochage-raccrochage téléphonique. Ce type d'équipement externe ne correspond qu'aux émetteurs-récepteurs combinés. Dans ce cas, le dispositif PUF est chargé de gérer la connexion RNIS;

2) équipement externe avec commande de décrochage-raccrochage téléphonique. Dans ce cas, tous les événements se rapportant au mécanisme de décrochage-raccrochage téléphonique sont communiqués à l'interface PCI-RNIS et le dispositif PUF est chargé de gérer la connexion RNIS;

3) équipement externe avec commande de décrochage-raccrochage téléphonique et capacité de gérer la connexion RNIS. Dans ce cas, tous les événements se rapportant au mécanisme de décrochage-raccrochage téléphonique sont communiqués à l'interface PCI-RNIS;

4) équipement externe avec clavier et avec ou sans commande de décrochage-raccrochage téléphonique. Dans ce cas, tous les événements de numérotation et tous les événements se rapportant au mécanisme de décrochage-raccrochage téléphonique sont communiqués à l'interface PCI-RNIS et le dispositif PUF est chargé de gérer la connexion RNIS;

5) équipement externe avec clavier et avec ou sans commande de décrochage-raccrochage téléphonique, capable de gérer la connexion RNIS. Dans ce cas, tous les événements de numérotation, tous les événements du mécanisme de décrochage-raccrochage téléphonique et toutes les informations sur le statut de la communication sont communiqués à l'interface PCI-RNIS.

Tous ces types d'équipements externes sont connectés au dispositif NAF par une connexion soumise à droits de propriété (hors du domaine d'application de la présente partie) et signalent au dispositif PUF s'ils sont ou non disponibles.

Dans le cas d'équipements externes de type 4 ou de type 5, deux types de numérotation sont possibles:

– numérotation sans chevauchement (en bloc): un seul message du plan de commande, contenant l'adresse de destination complète, est envoyé au dispositif PUF;

– numérotation avec chevauchement (à recouvrement): un seul message du plan de commande par pression de touche est envoyé au dispositif PUF. Au cours d'une communication, des codes à deux tonalités (DTMF) peuvent être envoyés au moyen du clavier.

Les équipements externes du type 3 ont la capacité de traitement autonome des appels entrants lorsque l'ordinateur n'est pas sous tension.

Les équipements externes du type 5 ont la capacité de traitement autonome des appels entrants et des appels sortants lorsque l'ordinateur n'est pas sous tension.

Chaque manipulation effectuée sur le combiné déclenche l'envoi d'un message du plan de commande vers le dispositif PUF. Selon le type d'équipement externe, différents niveaux de message sont envoyés au dispositif PUF:

– *Pour les équipements externes de type 1:*
  - disponibilité/indisponibilité.
– *Pour les équipements externes de type 2 et de type 3:*
  - disponibilité/indisponibilité;
  - combiné raccroché;
  - combiné décroché.
– *Pour les équipements externes de type 4 et de type 5:*
  - disponibilité/indisponibilité;
  - combiné raccroché;
  - combiné décroché;
  - code représentant la touche pressée en cas de numérotation avec chevauchement;
  - table de codes représentant l'adresse de destination complète en cas de numérotation sans chevauchement.

Dans le cas d'équipements externes de type 2 et de type 3, si le dispositif PUF a créé un objet NCO qui spécifie un équipement externe dans son ensemble d'attributs de signalisation, une connexion mettant en œuvre cet équipement externe peut être établie et rompue selon les différents modes suivants:

– *Cas des appels sortants:*
  - l'utilisateur décroche le combiné et le dispositif PUF envoie la signalisation de numérotation avec ou sans chevauchement;
  - le dispositif PUF effectue la numérotation avec ou sans chevauchement et l'utilisateur décroche le combiné.
– *Cas des appels entrants:*
  - l'utilisateur décroche le combiné et le dispositif PUF reçoit un message du plan de commande pour l'en informer;
  - le dispositif PUF répond à l'appel entrant et l'utilisateur décroche le combiné.
– *Cas d'une rupture locale:*
  - l'utilisateur raccroche le combiné et le dispositif PUF reçoit un message du plan de commande pour l'en informer;
  - le dispositif PUF libère la communication et l'utilisateur raccroche le combiné.
– *Cas d'une rupture distante:*
  - le dispositif PUF reçoit un message du plan de commande et l'utilisateur raccroche le combiné.

Dans le cas d'équipements externes de type 4 et de type 5, si le dispositif PUF a créé un objet NCO qui spécifie un équipement externe dans son ensemble d'attributs de signalisation, une connexion mettant en œuvre cet équipement externe peut être établie et rompue selon les différents modes suivants:

– *Cas des appels sortants:*
  - l'utilisateur décroche le combiné et le dispositif PUF envoie la signalisation de numérotation avec ou sans chevauchement;
  - l'utilisateur décroche le combiné, qui envoie au dispositif PUF un message du plan de commande; puis l'utilisateur utilise le clavier de l'équipement externe pour effectuer une numérotation avec chevauchement. Chaque pression de touche envoie au dispositif PUF un message du plan de commande;

- l'utilisateur décroche le combiné, qui envoie au dispositif PUF un message du plan de commande; puis l'utilisateur utilise le clavier de l'équipement externe pour effectuer une numérotation avec chevauchement. Chaque pression de touche envoie au dispositif PUF un message du plan de commande;
- le dispositif PUF effectue la numérotation avec ou sans chevauchement et l'utilisateur décroche le combiné.

– *Cas des appels entrants:*

- l'utilisateur décroche le combiné et le dispositif PUF reçoit un message du plan de commande pour l'informer de cet état;
- le dispositif PUF répond à l'appel entrant et l'utilisateur décroche le combiné.

– *Cas d'une rupture locale:*

- l'utilisateur raccroche le combiné et le dispositif PUF reçoit un message du plan de commande pour l'informer de cet état;
- le dispositif PUF libère la communication et l'utilisateur raccroche le combiné.

– *Cas d'une rupture distante:*

- le dispositif PUF reçoit un message du plan de commande et l'utilisateur raccroche le combiné.

5.3.2.4 Compatibilité avec des fonctions de sécurité

L'accès à des fonctions de sécurité de niveau inférieur à l'interface PCI-RNIS est offert au dispositif PUF par les propriétés du plan d'administration.

Les fonctions de sécurité assurées par l'interface PCI-RNIS couvrent l'emploi d'algorithmes de sécurité sur des connexions.

Le dispositif PUF peut activer et désactiver des fonctions de sécurité pour une connexion spécifique, en envoyant à l'objet NCO de cette connexion des messages du plan d'administration.

5.3.2.5 Compatibilité avec des spécificités de construction

L'accès à des spécificités de construction est offert au dispositif PUF par les propriétés du plan d'administration.

Le dispositif PUF peut accéder aux spécificités de construction en utilisant cette propriété. C'est une manière d'assurer des fonctions supplémentaires, non assurées par l'interface PCI-RNIS.

Les informations échangées entre dispositifs PUF et NAF dépendent de l'implémentation de la spécificité de construction; elles sont donc hors du domaine d'application de la présente partie.

5.3.3 Gestion des connexions

La propriété de gestion des connexions couvre deux aspects:

– l'établissement et la rupture de connexions physiques;

– l'obtention et l'usage de services complémentaires.

Les phases d'établissement et de rupture d'une connexion correspondent à la fonction essentielle de la gestion d'une connexion physique. Les services complémentaires constituent une propriété additionnelle, qui se rapporte à la gestion d'une connexion physique.

Le plan de commande de l'interface PCI-RNIS possède la propriété définie par la gestion de connexions physiques.

5.3.3.1 Etablissement et rupture de la connexion

La seule façon dont un dispositif PUF peut réaliser une connexion consiste à passer dans l'état de "repos" en créant un objet NCO. Il pourra ensuite répondre à une requête de connexion ou attendre une indication de connexion. Une fois la connexion supprimée, le dispositif PUF revient à l'état de "repos" et peut de nouveau utiliser l'objet NCO pour une nouvelle connexion. L'objet NCO devient invalide s'il est détruit ou si le dispositif PUF se désenregistre auprès du dispositif NAF.

Lors de la création d'un objet NCO, le dispositif PUF doit décider du type de connexion qu'il y a lieu de réaliser. L'interface PCI-RNIS offre un accès:

– à une connexion sémaphore associée au protocole de signalisation désigné;

– à une connexion de transfert de données, associée sur option à des protocoles de communication.

Dans le cas des connexions sémaphores, l'interface PCI-RNIS offre la possibilité d'établissement et de rupture de connexions. Cette possibilité est assurée par l'injection d'un seul message en haut de la couche 3 du protocole de signalisation.

Si le dispositif PUF a créé un objet NCO associé à un équipement externe, l'interface PCI-RNIS offre la possibilité d'établissement et de rupture de connexion. Toutes les actions de l'utilisateur concernant l'équipement externe (raccrochage, décrochage, numérotation) sont prises en compte par le sous-système des connexions sémaphores. Par ailleurs, certains équipements externes possèdent la capacité de gérer la signalisation RNIS lorsque le serveur est hors circuit.

Dans le cas de la téléphonie, une propriété supplémentaire peut être offerte. Elle consiste en une rupture temporaire (ou suspension) avec rétablissement ultérieur (reprise) d'une connexion.

Lorsqu'un objet NCO est associé à un dispositif PUF déterminé, celui-ci ne peut pas gérer les connexions établies avec d'autres dispositifs PUF.

5.3.3.2 Compatibilité avec les services complémentaires

Les services complémentaires constituent une propriété additionnelle qui est associée à la gestion des connexions. La description et l'usage des services complémentaires feront l'objet d'un complément d'étude.

Les services complémentaires offerts par la gestion des connexions RNIS sont mis à la disposition du PUF lorsque le cas de la coordination par dispositif PUF est applicable. Le PUF est chargé d'assurer la gestion des connexions et peut donc commander les services complémentaires fournis par la signalisation.

NOTE – L'emploi de services complémentaires lorsque la fonction de coordination est régie par le dispositif NAF fera l'objet d'un complément d'étude.

5.3.4 Gestion des données

La propriété de gestion des données couvre les deux aspects ci-après:

- établissement de connexions de transfert de données sur des connexions physiques déjà établies;
- échange de données.

Le plan d'utilisateur de l'interface PCI-RNIS possède la propriété définie par la gestion des données.

Pour le transfert de données d'utilisateur, l'interface PCI-RNIS donne accès à divers protocoles du plan d'utilisateur, exploités dans la couche Réseau du RNIS. Selon le protocole sélectionné dans le plan d'utilisateur, ce plan donne accès à des connexions de couche Réseau (couche 3), de couche Liaison de données (couche 2) ou de couche Physique transparente (couche 1).

On peut effectuer la sélection du protocole de plan d'utilisateur en utilisant les propriétés de la gestion de ressource (création ou modification d'un objet NCO).

Pour chaque type de connexion, il importe qu'une connexion sémaphore soit établie avant que l'on puisse accéder aux données. En général, à moins qu'un dispositif PUF ne fasse appel à la fonction de coordination offerte par le dispositif NAF, on établit la connexion sémaphore au moyen de la propriété de plan de commande, tandis qu'on établit l'accès aux données au moyen de la propriété de plan d'utilisateur.

Lorsqu'on utilise une connexion avec un protocole transparent du plan d'utilisateur ainsi qu'un objet NCO associé à un équipement externe, les données envoyées sur cette connexion doivent être adressées à l'équipement externe plutôt qu'être utilisées pour produire des messages du plan d'utilisateur.

La Partie 3 [4] décrit en détail les protocoles disponibles avec les messages, séquencements et paramètres correspondants du plan d'utilisateur.

5.4 Correspondance des propriétés avec les plans

Les relations suivantes s'appliquent à la propriété décrite au 5.3:

- le plan d'administration de l'interface PCI-RNIS offre la propriété définie par la gestion de ressource;
- le plan de commande de l'interface PCI-RNIS offre la propriété définie par la gestion de connexion;
- le plan d'utilisateur de l'interface PCI-RNIS offre la propriété définie par la gestion de données.

A l'intérieur de ces plans, chaque propriété est décrite au moyen d'opérations ou de groupes opérationnels.

5.4.1 Caractéristiques facultatives

Lorsqu'on rapporte chaque propriété à chaque plan, il y a certaines opérations ou certains groupes opérationnels que le dispositif NAF n'est pas obligé d'assurer.

Dans la description des plans, des indications sont données quant aux opérations ou groupes opérationnels qui sont obligatoires ou facultatifs.

Le fait que la description autorise des caractéristiques facultatives pour le dispositif NAF n'implique pas que l'interface PCI-RNIS contienne de telles caractéristiques. En tant que spécification, l'interface PCI-RNIS doit permettre l'échange de tous les messages. Le qualificatif facultatif implique ici que ces caractéristiques, fournies par le dispositif NAF, sont offertes au dispositif PUF. Si celui-ci demande une caractéristique qui n'est pas fournie par le dispositif NAF, il doit en être informé.

5.4.2 Plan d'administration

Le plan d'administration donne accès à des opérations qui facilitent la gestion des connexions (par exemple la définition et la gestion des ensembles d'attributs et d'adresses) ainsi que la gestion des objets NCO. En outre, les diverses opérations suivantes sont assurées dans ce plan:

- opération de signalisation d'erreur;
- opération de sécurité;
- opération spécifique d'un constructeur.

Le Tableau 1 donne un aperçu général des opérations assurées dans le plan d'administration.

Tableau 1 – Opérations dans le plan d'administration

Nom de l'opération	But de l'opération
objet NCO Création	créer un objet de connexion réseau
objet NCO Suppression	supprimer un objet de connexion réseau
objet NCO Obtention d'informations	obtenir des informations sur un objet de connexion réseau
erreur	signaler une condition d'erreur associée à un défaut de connexion
sécurité (Note)	manipulation relative à la sécurité
opération spécifique d'un constructeur (Note)	requête d'une propriété spécifique d'un constructeur
changement de protocole (Note)	changement du protocole de plan utilisateur sur le canal B établi
NOTE – Ces groupes opérationnels sont facultatifs pour le dispositif NAF.	

5.4.3 Plan de commande

Le plan de commande donne accès à des opérations qui effectuent la commande d'appel de base dans la signalisation RNIS.

Contrairement au plan d'administration, le plan de commande ne fait pas de séparation précise entre les opérations. On peut toutefois distinguer un certain nombre de groupes opérationnels dans le plan de commande. Le Tableau 2 donne un aperçu général des groupes opérationnels du plan de commande.

5.4.4 Plan d'utilisateur

Le plan d'utilisateur effectue des opérations qui facilitent l'établissement, la libération et l'échange de données de voies logiques de communication. Ce plan fait appel à des services normalisés et à des procédures définies pour l'accès de chaque message d'utilisateur sélectionné. Les Parties 3 à 6 relatives à l'interface PCI-RNIS décrivent en détail les opérations disponibles pour les divers accès possibles (couche Physique transparente, couche Liaison de données, couche Réseau).

Tableau 2 – Opérations dans le plan de commande

Nom du groupe opérationnel	But du groupe opérationnel
établissement d'une connexion	traitement des appels entrants et sortants
rupture de connexion	traitement du retrait de connexions ou du refus d'appels
transfert d'informations d'utilisateur à utilisateur (Note)	échange d'informations d'utilisateur à utilisateur et fourniture des messages de commande pour cet échange
ajournement d'appels (Note)	fourniture des messages de suspension et de reprise d'appels
invocation de service complémentaire (Note)	traitement des invocations de services complémentaires
équipement externe (Note)	indication du statut ou du changement d'état d'un équipement externe
informations additionnelles (Note)	fourniture d'un accès à des informations additionnelles en cours de communication
NOTE – Ces groupes opérationnels sont facultatifs pour le dispositif NAF.	

5.5 Interactions entre dispositifs PUF et NAF

Le présent sous-paragraphe décrit le type de fonctions offertes au dispositif PUF lors de ses interactions avec un dispositif NAF, ainsi que l'ordre dans lequel elles peuvent être utilisées.

Les propriétés suivantes s'appliquent à toutes les fonctions:

- interaction lancée par le dispositif PUF: seul le dispositif PUF peut demander son association avec le dispositif NAF;
- interaction demandée par appels de fonction du dispositif PUF au dispositif NAF;
- interaction effectuée de manière synchrone. Un dispositif PUF qui demande à un dispositif NAF d'exécuter une fonction doit reprendre à ce dispositif NAF le contrôle du processeur une fois l'appel de fonction exécuté.

Dans l'interaction entre dispositifs PUF et NAF, les phases suivantes peuvent être distinguées:

- *Phase d'enregistrement (d'association)*

  Avant que des dispositifs PUF et NAF puissent échanger des informations, le dispositif PUF établit une association avec le dispositif NAF. Comme, dans un système donné, il est possible que plusieurs dispositifs NAF soient disponibles et qu'ils proviennent de différents constructeurs de NAF, on définit une méthode permettant au dispositif PUF de déterminer quels sont les dispositifs NAF qui sont accessibles dans ce système. Cette phase est appelée phase d'enregistrement. Elle permet d'accéder à une liste de dispositifs NAF accessibles, au moyen des pointeurs de l'interface PCI-RNIS. Ensuite, le dispositif PUF peut déterminer les propriétés du dispositif NAF qui ont été sélectionnées par le pointeur de l'interface PCI-RNIS puis établir une association avec ce dispositif NAF.

- *Phase de conversation*

  Dans cette phase, les dispositifs PUF et NAF peuvent échanger des messages. Cette phase est appelée phase de conversation. Le dispositif PUF contrôle l'échange de messages entre lui-même et le dispositif NAF. C'est-à-dire que le dispositif PUF insère dans le message les paramètres appropriés et le transmet au dispositif NAF pour traitement; sinon, le dispositif PUF demande au dispositif NAF de recevoir un message après lui avoir fourni les ressources nécessaires.

  Un dispositif PUF a deux moyens pour déterminer que le dispositif NAF a un message pour lui. Le plus simple consiste à interroger le dispositif NAF. La deuxième méthode offre au dispositif NAF un moyen rapide pour signaler au dispositif PUF qu'un message lui est destiné. Avec cette méthode, le dispositif PUF autorise explicitement le dispositif NAF à lui signaler la disponibilité d'un message. Elle présente l'avantage d'offrir un mode de fonctionnement efficace, pour les dispositifs PUF sensibles aux performances.

  Cette méthode aidera par exemple les dispositifs PUF qui sont associés à plusieurs dispositifs NAF. Les dispositifs PUF qui utilisent cette méthode sont toutefois de conception plus complexe que ceux qui n'y font pas appel.

– *Phase de désenregistrement (d'association)*

Lorsqu'un dispositif PUF n'a pas besoin d'échanger des messages avec un dispositif NAF, il s'en dissocie. Cette phase est appelée phase de désenregistrement. Elle est importante en termes de gestion de ressource (surtout de mémoire) dans le dispositif NAF. Le dispositif PUF doit effectuer cette dissociation pour garantir une utilisation efficace des ressources du système global.

Le Tableau 3 énumère les fonctions, groupées selon les trois phases ci-dessus.

Tableau 3 – Groupement en phases des fonctions de l'interface PCI-RNIS

Phase	Fonction	But de la fonction
enregistrement (d'association)	PciGetHandles	fournir une liste des dispositifs NAF accessibles et obtenir leurs pointeurs d'interface PCI
	PciGetProperty	fournir des informations détaillées sur un dispositif NAF
	PciRegister	associer le PUF au NAF
conversation	PciPutMessage	transférer un message du PUF au NAF
	PciGetMessage	demander au NAF de recevoir un message, en lui fournissant des ressources
	PciSetSignal	établir un mécanisme permettant au NAF de signaler au PUF qu'un message est disponible
désenregistrement	PciDeregister	dissocier le PUF du NAF

Ces fonctions doivent être utilisées dans un certain ordre. La Figure 3 présente un diagramme de transition d'état pour les appels de fonction de l'interface PCI-RNIS.

Les messages sont transférés entre PUF et NAF au moyen des fonctions PciPutMessage et PciGetMessage.

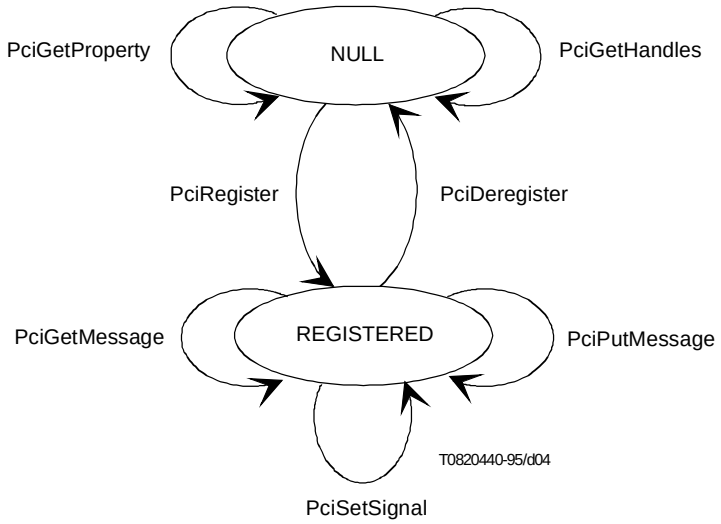


Figure 3 – Ordre des appels de fonction de l'interface PCI-RNIS

5.6 Aperçu général de toutes les interactions

Les Figures 4 et 5 présentent, à titre d'exemple de séquencement des opérations, un aperçu général diachronique des interactions que le dispositif PUF doit assurer pour obtenir une connexion.

Dans ces figures, les conventions suivantes sont utilisées:

– pour toute une figure, les pointillés désignent une interaction facultative;

– dans la partie de la figure qui concerne la phase de conversation, les flèches allant du dispositif PUF au dispositif NAF indiquent l'utilisation du message PciPutMessage tandis que les flèches allant du NAF au PUF indiquent l'utilisation du message PciGetMessage;

– les légendes en petits caractères désignent les messages décrits au paragraphe 7.

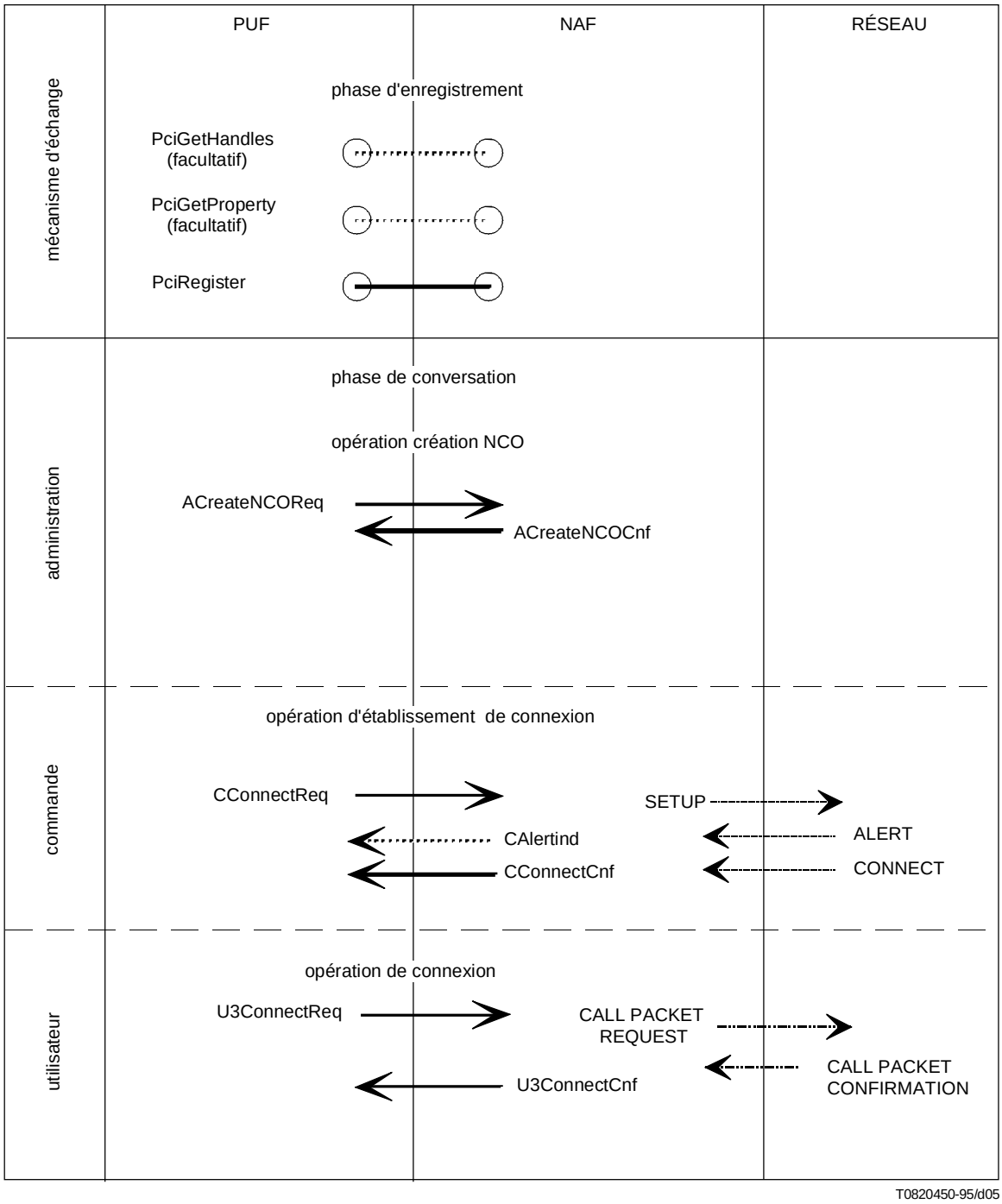


Figure 4 – Exemple de séquence d'opérations – Cas de la coordination par PUF

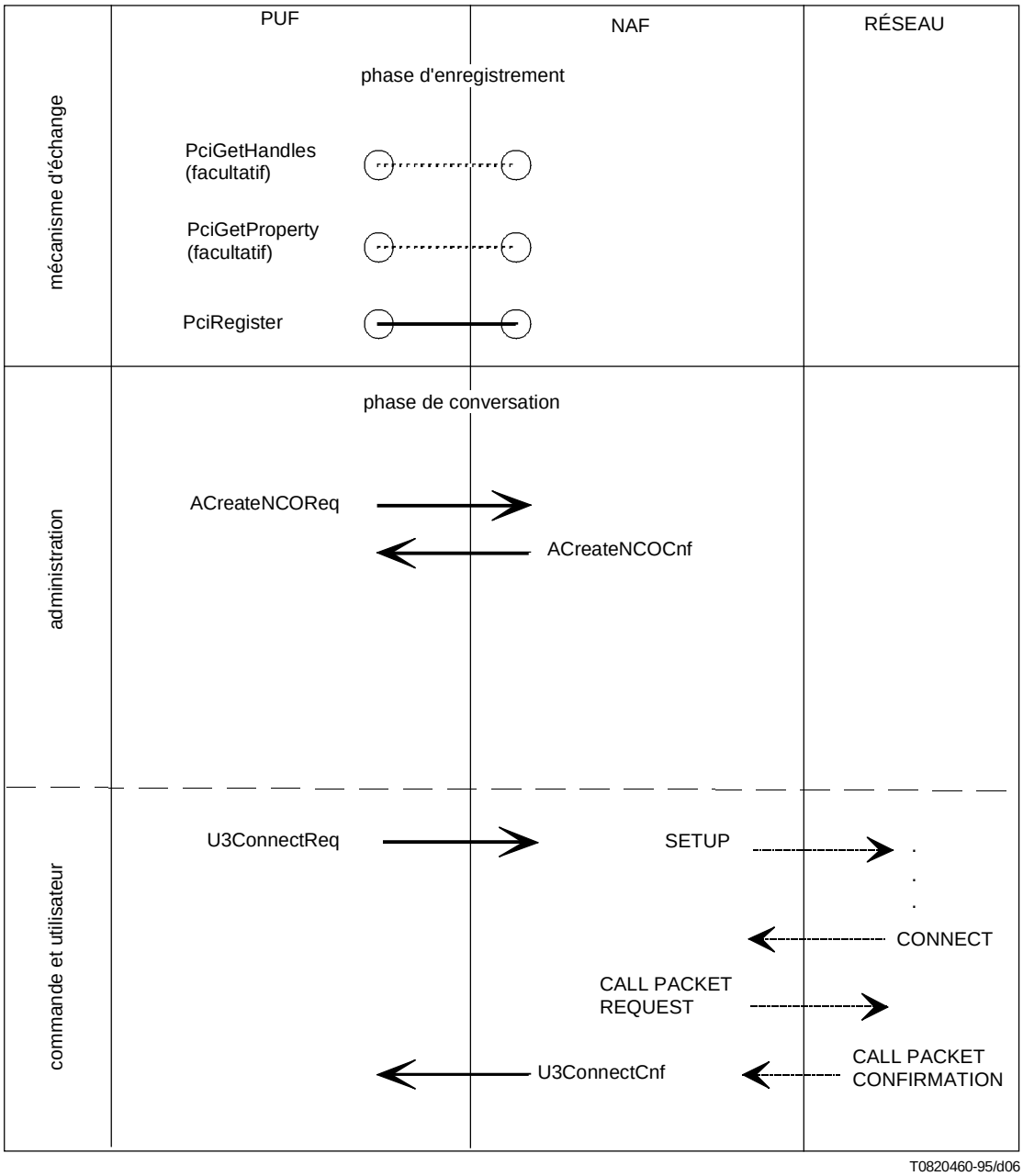


Figure 5 – Exemple de séquence d'opérations – Cas de la coordination par NAF

5.7 Identificateurs

Pour ses opérations, l'interface PCI-RNIS définit des identificateurs. Ces derniers doivent être utilisés par le dispositif PUF pour identifier de manière abstraite des objets concrets ou des connexions réelles.

Le présent sous-paragraphe résume les identificateurs utilisés lors de la spécification d'interface PCI-RNIS. Seule la description fonctionnelle de ces identificateurs sera donnée. Les détails feront l'objet de paragraphes ultérieurs.

Identificateurs se rapportant au mécanisme d'échange (détails au paragraphe 8).

PCI-Handle Cet identificateur est une référence abstraite qui renvoie à un dispositif NAF. Le pointeur ainsi identifié peut être utilisé pour trouver des informations sur le dispositif NAF et pour effectuer un enregistrement auprès de ce dispositif au moyen de la fonction **PciRegister** du mécanisme d'échange.

ExID Cet identificateur est la représentation de l'association d'un dispositif PUF et d'un dispositif NAF. Il est fourni par la fonction **PciRegister** du mécanisme d'échange. Il est nécessaire pour tout appel de fonction PCI-RNIS en rapport avec cette association.

Identificateurs se rapportant à la phase de communication entre dispositifs PUF et NAF (détails au paragraphe 7).

CAttributeName Cet identificateur se rapporte à un ensemble d'attributs statiques de paramètres du plan de commande. Il est représenté par un nom. Cet identificateur sera utilisé ultérieurement pour la création d'un objet de connexion réseau (NCO). La liste complète des ensembles d'attributs statiques est reproduite dans l'Annexe F.

UAttributeName Cet identificateur se rapporte à un ensemble d'attributs de paramètres du plan de commande. Il est représenté par un nom. Cet identificateur sera utilisé ultérieurement pour la création d'un objet de connexion réseau (NCO). La liste complète des ensembles d'attributs statiques est reproduite dans l'Annexe F.

ExtEquipName Cet identificateur se rapporte à un équipement externe. Il est représenté par un nom. Celui-ci peut être obtenu soit implicitement soit au moyen de la fonction PciGetProperty.

NCOID Cet identificateur se rapporte à la connexion désignée en référence par le dispositif PUF. Il sert à celui-ci pour indiquer au dispositif NAF quelle est la connexion visée. Etant donné qu'une connexion correspond toujours à un objet de connexion réseau (NCO), cet identificateur est une référence à cet objet NCO. L'identificateur NCOID est fourni en réponse au message de création d'objet NCO.

GroupID Identificateur abstrait désignant un groupe d'objets NCO. Il dépend du protocole de plan d'utilisateur.

RequestID Cet identificateur désigne le message qui est échangé entre le dispositif PUF et le dispositif NAF dans le plan d'administration. Les réponses suivantes à ce message doivent contenir le même identificateur RequestID pour identifier le message original. Il est permis d'effectuer de multiples transmissions asynchrones vers ce plan.

SelectorID Cet identificateur désigne l'objet NCO se rapportant au message pour un appel entrant, en cas de correspondance avec de multiples objets NCO. Le dispositif NAF ne sélectionnera qu'un seul objet NCO dans cet ensemble abstrait de dispositifs PUF, désigné par la même valeur d'identificateur SelectorID. Cela permet au dispositif PUF de limiter le nombre d'objets NCO sélectionnés par un dispositif NAF et de limiter le nombre de messages émis par le NAF en cas d'appel entrant.

5.8 Traitement des erreurs

5.8.1 Aperçu général

Une information d'erreur est retournée au dispositif PUF au moyen de codes de retour relatifs aux fonctions et d'informations présentes dans les messages. Généralement, les codes de retour relatifs aux fonctions fournissent l'information sur l'erreur produite au moyen d'un transfert de PUF à NAF des paramètres, avec vérification des données correspondant à ces derniers. Les messages contiennent des informations d'erreur reflétant cette vérification des données visées par ces paramètres et le traitement de messages ou d'événements déjà issus des protocoles utilisés.

5.8.2 Traitement des erreurs de fonction

Pour chaque fonction, les valeurs paramétriques fournies sont vérifiées. Si l'une d'elles est jugée erronée, ce fait doit être signalé sous forme de code de retour relatif à une fonction et l'action demandée par cette fonction n'est pas exécutée.

L'examen du paramètre qui a lieu lors de l'invocation d'une fonction par un dispositif PUF (et l'ordre de cette vérification) dépend de la fonction invoquée.

5.8.3 Traitement des erreurs de message

La détection d'erreur s'effectue en deux étapes pendant le traitement d'un message d'interface PCI:

1) lors de l'examen initial du message par le dispositif NAF, pour s'assurer qu'il convient à la suite du traitement. Ce contrôle est d'ordre administratif, de telle sorte que chaque erreur détectée est retournée dans des messages du plan d'administration;

2) lors du traitement du message par le dispositif NAF, la façon dont l'information d'erreur est transmise au dispositif PUF dépend du plan auquel le message appartient et du protocole sous-jacent à ce plan.

L'examen initial du message (et l'ordre de vérification) lorsque ce message vient d'être reçu par un dispositif PUF est le suivant:

a) vérification de la disponibilité du dispositif NAF;

b) vérification de l'identificateur de message:

- message inconnu, non défini par l'interface PCI-RNIS;
- message incompatible, défini par l'interface PCI-RNIS mais non géré par le dispositif NAF.

Dans le cas de messages du plan d'administration, chaque information d'erreur est retournée dans le message de confirmation correspondant. Dans le cas de messages des plans de commande et d'utilisateur, chaque information d'erreur est retournée dans le message AErrorInd du plan d'administration.

La détection d'erreur (et l'ordre de vérification) qui a lieu lors du traitement du message par le dispositif NAF dépend du protocole. L'information d'erreur est retournée par un mécanisme propre au protocole utilisé. Ces mécanismes sont décrits au 7.8.

6 Codage des éléments d'information

Au 7.6, les types utilisés ont les significations suivantes:

–	octet	multiplet de 8 éléments binaires;
–	booléen	octet ayant un ensemble limité de valeurs (0 = FAUX, sinon VRAI);
–	chaîne d'octets	séquence d'octets de longueur variable ou fixe;
–	chaîne IA5	chaîne d'octets dont chacun représente un caractère de l'alphabet IA5.

Chaque paramètre fait l'objet d'un codage TLV (type, longueur, valeur) comme suit:

–	type	= 1 octet;
–	longueur	= 1 octet;
–	valeur	limitée par l'octet.

Les champs insérés dans chaque paramètre sont codés sous forme d'informations structurées, dont l'ordre est défini par celui du paramètre lui-même, comme indiqué au 7.6. Les champs omis réduisent la longueur du paramètre correspondant.

Les valeurs indiquées entre parenthèses sont en décimal.

7 Description des messages d'interface PCI-RNIS

Comme décrit au 5.5: "Interactions entre dispositifs PUF et NAF", l'échange de messages est réalisé au moyen de 2 fonctions: PciPutMessage et PciGetMessage, qui peuvent être appelées dès que le dispositif PUF est associé au dispositif NAF. Etant donné la nature de ces fonctions, que l'on peut utiliser indépendamment l'une de l'autre, la corrélation entre les "messages obtenus" et les "messages émis" doit être assurée par le dispositif PUF. C'est pourquoi les messages de chaque plan contiennent des identificateurs permettant une corrélation entre messages.

Les paragraphes suivants décrivent les messages fournis par chaque plan de l'interface PCI-RNIS, ainsi que les paramètres utilisés en association avec chaque message. Le paragraphe 6: "Codage des éléments d'information" décrit la présentation finale des informations et le codage utilisé pour les opérations et pour les paramètres.

7.1 Conventions

La description des messages, de leurs paramètres et de leurs champs, est indépendante des systèmes matériels et des systèmes d'exploitation.

7.1.1 Conventions relatives aux adresses

Dans la présente partie, les conventions suivantes s'appliquent à toutes les adresses utilisées:

- l'adresse appelée est celle à laquelle l'expéditeur souhaite se connecter;
- l'adresse appelante est l'adresse locale de l'expéditeur.

7.1.2 Conventions relatives aux informations

Les conventions et abréviations suivantes sont utilisées pour la fourniture ou la prescription d'éléments de message, qui peuvent varier:

M = obligatoire (*mandatory*): l'élément doit être fourni;

C = conditionnel: une condition détermine si cet élément doit être fourni. La condition est expliquée sous forme de commentaire associé à l'élément;

O = facultatif (*optional*): cet élément peut ne pas être fourni. Pour l'échange de PUF à NAF, cela implique que le dispositif PUF a le choix entre fournir l'élément et ne pas le fournir. Pour l'échange de NAF à PUF, cela implique que le dispositif NAF ne doit fournir l'élément que si celui-ci est disponible.

Les informations provenant du dispositif NAF correspondent aux informations fournies par le réseau.

7.1.3 Conventions relatives aux messages

Le présent sous-paragraphe présente les conventions utilisées dans les tableaux pour décrire les messages.

Chaque message appartient à une classe. Celle-ci est indiquée avec chaque message. Toutes les classes ne sont pas forcément gérables par un dispositif NAF. Le fournisseur de celui-ci a la possibilité de n'implémenter que certaines classes. Chaque plan contient ses propres classes.

Pour chaque plan, un dispositif PUF ne peut compter que sur la disponibilité des messages de la classe 1 (classe de base). Les autres messages relèvent de classes additionnelles. Si un dispositif NAF implémente une classe additionnelle, tous les messages appartenant à cette classe dans le même plan doivent être fournis.

Le message indique son sens de transfert dans la partie suffixe de son nom. Les messages ayant le suffixe Req ou Rsp sont transférés du PUF au NAF. Les messages ayant le suffixe Ind ou Cnf sont transférés du NAF au PUF. Les identificateurs de message sont exprimés en valeurs décimales.

7.1.4 Conventions relatives aux paramètres

Les conventions suivantes sont utilisées pour la description des paramètres:

- le nom du champ doit être indiqué;
- le type du champ doit être indiqué en valeur décimale;

– l'entité chargée de fournir le contenu du champ doit être indiquée (dans la colonne Sens). Les abréviations suivantes sont utilisées:

  P à la charge du dispositif PUF

  N à la charge du dispositif NAF

  B aussi bien le PUF que le NAF peut fournir le contenu;

– la longueur que le champ doit occuper (en nombre d'octets) peut être indiquée. Le terme octet n'est associé à aucun matériel ou système d'exploitation particulier. L'octet est l'unité d'information de base dans tous les systèmes.

7.1.4.1 Ordonnancement des paramètres

Aucun ordonnancement des paramètres, les uns par rapport aux autres, n'est nécessaire. Le séquencement des tableaux n'a pas de correspondance avec celui des paramètres.

Le 7.6 de la présente partie du présent appendice définit l'ordonnancement des champs à l'intérieur des paramètres.

7.1.4.2 Répétition de paramètres

Les paramètres contenus dans un message peuvent être répétés. Le nombre de répétitions est fixé par le réseau ou par le protocole utilisateur appliqué.

7.1.4.3 Vérification de paramètres

Il n'y a pas lieu que le dispositif NAF effectue un processus particulier de vérification pour les paramètres issus du réseau.

7.1.5 Principes relatifs aux valeurs par défaut

Pour les valeurs des paramètres et de leurs champs internes, des principes relatifs aux valeurs par défaut sont applicables. C'est-à-dire que, le cas échéant, la valeur "par défaut" est indiquée dans la description. Cette indication est suivie de la valeur par défaut proprement dite.

La valeur par défaut ne doit être utilisée que dans l'échange de messages de PUF à NAF. Si un paramètre n'est pas contenu dans un message, il sera remplacé par la valeur fournie au moment de la création de l'objet NCO.

Lors de l'échange de NAF à PUF, seule la valeur réelle doit être indiquée.

7.2 Messages dans le plan d'administration

Les messages du plan d'administration se subdivisent en classes comme suit:

1) messages de gestion d'objets de connexion réseau (NCO) et de signalisation d'erreur;

2) messages de gestion de la sécurité des connexions;

3) messages du constructeur de dispositif NAF;

4) messages de changement de protocole.

Pour la gestion des objets de connexion réseau (NCO), il existe des messages de création et de destruction d'un objet NCO. Au cours de la création d'un objet NCO, des ensembles d'attributs et d'adresses statiques ou dynamiques sont associés à l'objet NCO créé. A la fin de la création d'un objet NCO, un identificateur d'objet NCO (NCOID) devient disponible. Cet identificateur doit être utilisé lors des opérations ultérieurement effectuées dans le plan d'utilisateur ou dans le plan de commande, sur l'objet ainsi créé. L'Annexe F contient une série d'ensembles d'attributs prédéfinis. Un message unique est émis par le dispositif NAF pour signaler des informations d'erreur. Ce message sert à rendre compte de conditions générales d'erreur.

Pour assurer la sécurité des connexions, certains messages permettent d'activer ou de désactiver les mesures de sécurité relatives à une connexion. Ces messages sont facultatifs et peuvent ne pas être assurés par tous les dispositifs NAF. Leur disponibilité doit être indiquée dans la définition des propriétés fournie par le dispositif NAF.

Certains messages permettent d'accéder à des propriétés spécifiques d'un constructeur. Ces messages sont facultatifs et peuvent ne pas être assurés par tous les dispositifs NAF. Leur disponibilité doit être indiquée dans la définition des propriétés fournies par le dispositif NAF. Les informations échangées entre PUF et NAF dépendent de la mise en œuvre de cette caractéristique: elles sont donc hors du domaine d'application de la présente partie.

Certains messages permettent de demander un changement du protocole de plan utilisateur associé à un objet NCO. Ces messages sont facultatifs et peuvent ne pas être assurés par tous les dispositifs NAF. Leur disponibilité doit être indiquée dans la définition des propriétés fournie par le dispositif NAF.

Tous les messages de requête dans le plan d'administration peuvent contenir un identificateur de requête (RequestID). Cet identificateur, s'il est assigné par un dispositif PUF au sujet d'un message de requête, est retourné par le dispositif NAF dans le message de confirmation correspondant.

Le Tableau 4 donne un aperçu général des messages du plan d'administration. Chacun de ces messages est décrit en détail dans les sous-paragraphes suivants.

Tableau 4 – Messages du plan d'administration

Identificateur de message	Classe	Nom du message	But du message
101	1	ACreateNCOReq	demande de création d'un objet de connexion réseau
102	1	ACreateNCOCnf	confirmation de l'opération "CreateNCO"
103	1	ADestroyNCOReq	demande de destruction d'un objet NCO
104	1	ADestroyNCOCnf	confirmation de l'opération "DestroyNCO"
105	1	AGetNCOInfoReq	demande d'information concernant un NCO spécifique
106	1	AGetNCOInfoCnf	confirmation de signalisation d'information concernant un NCO spécifique
108	1	AErrorInd	indication du fait qu'une erreur non relative au protocole s'est produite
109	2	ASecurityReq	demande de lancement/d'arrêt d'algorithme de sécurité
110	2	ASecurityCnf	confirmation de lancement/d'arrêt d'algorithme de sécurité
111	3	AManufacturerReq	demande d'une propriété spécifique d'un constructeur
112	3	AManufacturerInd	fourniture au PUF d'informations associées à la propriété demandée
113	4	AChangeNCOReq	demande de changement relatif à un NCO existant
114	4	AChangeNCOCnf	confirmation de changement d'objet NCO existant

7.2.1 ACreateNCOReq

Classe: 1 (classe de base).

Description: message de demande de création d'un objet de connexion réseau (NCO).

Le dispositif PUF doit fournir un paramètre NCOType indiquant le type d'objet NCO qu'il faut créer. Selon ce type, d'autres paramètres (conditionnels) seront requis. Pour les détails, voir les Tableaux 9 et 10.

Le dispositif PUF peut fournir un unique identificateur de requête (RequestID) qui pourra servir à identifier le message de confirmation correspondant à cette opération.

Paramètres:

Nom	Requis	Commentaire
RequestID	O	demande d'identificateur émise par le PUF
NCOType	M	spécification du type d'objet NCO
CDirection	C	détermination de la manière dont l'objet NCO doit être utilisé, pour le plan de commande. Absent si la valeur du paramètre NCOType est U3; sinon facultatif.
UDirection	C	détermination de la manière dont l'objet NCO doit être utilisé, pour le plan d'utilisateur. Ce paramètre dépend du protocole de plan utilisateur.
CAttributeName	C	nom de l'attribut statique du plan de commande
CAttribute (paramètres)	C	paramètres d'attribut de plan utilisateur mutuellement exclusif avec CAttributeName; pour plus de détails, voir le Tableau 10.
UAttributeName	C	nom de l'attribut statique du plan d'utilisateur
UAttribute (paramètres)	C	paramètres d'attribut du plan d'utilisateur mutuellement exclusif avec UAttributeName; pour plus de détails, voir les spécifications de PCI-RNIS relatives au protocole de plan d'utilisateur.
CAddress (paramètres)	O	adresse dans le plan de commande; pour plus de détails, voir le Tableau 13.
UAddress (paramètres)	O	adresse dans le plan d'utilisateur; pour plus de détails, voir les spécifications de PCI-RNIS relatives au protocole de plan d'utilisateur.
GroupID	C	requis si des objets NCO doivent être groupés. Ce paramètre dépend du protocole du plan utilisateur.
SelectorID	O	aide le NAF à choisir l'objet NCO approprié
CPMessageMask	O	filtre de messages RNIS. Si ce paramètre n'est pas fourni, le PUF reçoit tous les messages du plan de commande.
CPParameterMask	O	filtre de paramètres du plan de commande RNIS. Si ce paramètre n'est pas fourni, le PUF reçoit tous les paramètres du plan de commande.

Remarques: voir également 7.7 sur l'usage des objets NCO.

Message associé: ACreateNCOCnf.

7.2.2 NCOType et spécification de paramètre conditionnel

Le présent sous-paragraphe définit les types d'objets NCO utilisables dans un message ACreateNCOReq (demande de création d'objet NCO dans le plan d'administration).

Quatre types d'objets NCO sont actuellement définis. Ils sont indiqués dans le Tableau 5.

Pour les types d'objets NCO compatibles avec un accès par le plan d'utilisateur, les Tableaux 5 et 6 ne montrent que les formes générales de ces types. La Partie 3 [4] décrit en détail les types d'objets NCO spécifiques que l'on peut utiliser avec chaque protocole du plan d'utilisateur.

Tableau 5 – Types d'objets NCO

NCOType	L'objet NCO autorise le PUF …
C	… à n'accéder qu'à la signalisation
C/U	… à accéder à la signalisation et au plan d'utilisateur (propriété de coordination par dispositif PUF)
U3	… à accéder au plan d'utilisateur U3 avec signalisation confiée au dispositif NAF (propriété de coordination par dispositif NAF)
U3G	… à accéder au plan d'utilisateur pour employer des circuits virtuels additionnels. Cet objet NCO doit être groupé avec un NCO de type U3 ou C/U déjà créé.

Le Tableau 6 montre, en fonction du type d'objet NCO choisi, quels paramètres conditionnels doivent être spécifiés dans le message ACreateNCOReq.

Tableau 6 – Spécification de paramètres conditionnels de message ACreateNCOReq

Type de NCO	Type d'attribut du plan de signalisation	Type d'attribut du plan d'utilisateur	Type d'adresse du plan de signalisation	Type d'adresse du plan d'utilisateur	Identificateur de groupe
C	attribut C		adresse C		
C/U	attribut C	attribut U	adresse C	adresse U	
U3	attribut C	attribut U	adresse C	adresse U	
U3G		attribut U		adresse U	référence au NCO (Note)
NOTE – Si un objet NCO doit être groupé – ce qui ne peut être fait que dans le cas d'un objet de type U3G – il faut que l'identificateur GroupID fasse référence à un objet de type U3 ou C/U déjà créé. La création d'un tel type d'objet NCO doit donc avoir été effectuée normalement pour que l'on puisse disposer de l'identificateur GroupID.					

7.2.3 ACreateNCOCnf

Classe: 1 (classe de base).

Description: message de confirmation de l'opération de création d'objet NCO déjà demandée. Ce message de confirmation peut être mis en correspondance avec le message ACreateNCOReq correct, au moyen de l'identificateur RequestID retourné.

Le message de confirmation peut contenir l'identificateur d'objet (NCOID) qui devra être utilisé lors de futures demandes passant par le plan d'utilisateur ou de commande au sujet de l'objet NCO créé; il peut également contenir l'identificateur de groupe qui devra être utilisé dans de futurs messages ACreateNCOReq, si l'on envisage un groupement selon l'objet NCO créé.

Paramètres:

Nom	Fourni	Commentaire
RequestID	C	fourni si contenu dans message de requête
CompletionStatus	M	statut d'exécution de l'opération de création d'objet NCO par le dispositif NAF
NCOID	C	identificateur d'objet NCO si le paramètre CompletionStatus a la valeur "succès". Sinon absent.
GroupID	C	identificateur de groupe, fourni si l'objet NCO créé était du type C/U3 ou U3 et si le paramètre CompletionStatus a la valeur "succès".

Message associé: ACreateNCOReq.

7.2.4 ADestroyNCOReq

Classe: 1 (classe de base).

Description: détruit un objet NCO déjà créé par le même dispositif PUF. Celui-ci peut fournir un identificateur de requête (RequestID) qui pourra servir à identifier le message de confirmation correspondant à cette opération.

Paramètres:

Nom	Requis	Commentaire
RequestID	O	identificateur de requête, émis par le PUF
NCOID	M	identificateur de l'objet NCO à détruire

NOTE – L'objet NCO ne peut pas être détruit s'il est déjà utilisé pour une connexion établie ou pour une connexion que l'on tente d'établir. Lorsqu'un objet NCO non groupé est détruit, tous les objets NCO groupés selon lui deviennent inutilisables, sauf lorsque l'objet NCO groupé se rapporte à une connexion établie ou que l'on tente d'établir. Dans ce cas, l'objet NCO reste utilisable jusqu'à ce que la connexion en cause soit supprimée. Un objet NCO inutilisable ne peut être détruit qu'au moyen du message ADestroyNCOReq.

Message associé: ADestroyNCOCnf.

7.2.5 ADestroyNCOCnf

Classe: 1 (classe de base).

Description: message de confirmation de l'opération de destruction d'objet déjà demandée. Ce message de confirmation peut être mis en correspondance avec le message ADestroyNCOReq approprié au moyen de l'identificateur RequestID.

Paramètres:

Nom	Fourni	Commentaire
RequestID	C	fourni si contenu dans le message de requête
NCOID	M	identifie l'objet NCO au sujet duquel une opération de destruction a été demandée
CompletionStatus	M	statut d'exécution de l'opération de destruction d'objet NCO par le dispositif NAF

Message associé: ADestroyNCOReq.

7.2.6 AGetNCOInfoReq

Classe: 1 (classe de base).

Description: message de demande d'information sur un objet NCO. Chaque objet NCO est caractérisé par certains attributs (voir au 7.6.47 les paramètres de l'ensemble d'attributs du plan d'administration). Ces attributs sont accessibles à partir du dispositif PUF au moyen de cette demande et de sa confirmation.

Paramètre:

Nom	Requis	Commentaire
NCOID	M	identificateur de l'objet NCO demandé

Message associé: AGetNCOInfoCnf.

7.2.7 AGetNCOInfoCnf

Classe: 1 (classe de base).

Description: message de confirmation envoyé par le NAF au PUF pour répondre à une requête AGetNCOInfoReq. Ce message contient les informations relatives à l'objet NCO demandé (voir au 7.6.47 les paramètres de l'ensemble d'attributs du plan d'administration).

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identificateur de l'objet NCO demandé
CompletionStatus	M	statut d'exécution de l'opération GetNCOInfo du dispositif NAF
AAttribute	C	paramètres de l'ensemble d'attributs du plan d'administration si le paramètre CompletionStatus a la valeur "succès"; absent dans le cas contraire.

Message associé: AGetNCOInfoReq.

7.2.8 AErrorInd

Classe: 1 (classe de base).

Description: ce message se rapporte à la vérification administrative des messages (c'est-à-dire qu'il est indépendant du protocole).

Paramètres:

Nom	Fourni	Commentaire
RequestID	C	fourni si contenu dans le message de requête
CompletionStatus	M	valeur indiquant l'erreur qui s'est produite

Message associé: néant.

7.2.9 ASecurityReq

Classe: 2 (classe additionnelle).

Description: ce message permet au dispositif PUF de lancer un algorithme de sécurité fourni par le dispositif NAF. Le PUF doit fournir l'identificateur NCOID de la connexion à laquelle il souhaite appliquer l'algorithme de sécurité. Le PUF peut, pour l'application de la procédure de sécurité, désigner une connexion quelconque. Le dispositif PUF doit être informé par le NAF – au moyen d'un message ASecurityCnf – s'il est possible d'utiliser l'algorithme de sécurité sur la connexion indiquée.

Le message ASecurityReq n'indique pas comment la connexion est assurée ou quel est le type d'information acheminé par la connexion que l'algorithme de sécurité doit influencer. Le processus de sécurisation de la connexion est sous la dépendance de l'algorithme de sécurité.

Le paramètre Algorithm indique au dispositif NAF quel est l'algorithme de sécurité qui doit être utilisé pour protéger la connexion. Cet algorithme de sécurité est identifié par son nom. Les noms des algorithmes disponibles peuvent être obtenus au moyen des informations relatives aux propriétés, fournies par le dispositif NAF. En utilisant le nom "nosecurity" pour ce paramètre, le dispositif PUF peut indiquer que la sécurité n'est plus requise pour la connexion.

Le paramètre facultatif Key peut être utilisé par le PUF pour donner au NAF des informations utiles à l'algorithme de sécurité. Ce paramètre est facultatif parce que l'algorithme de sécurité n'a pas forcément besoin d'informations spécifiques pour s'activer. Le type d'information à utiliser pour le paramètre Key dépend de l'algorithme de sécurité activé.

Paramètres:

Nom	Requis	Commentaire
RequestID	O	identificateur de demande, émis par le PUF.
NCOID	M	identificateur de la connexion pour laquelle l'algorithme de sécurité doit être activé
Algorithm	M	nom de l'algorithme de sécurité à utiliser
Key	O	clé à utiliser pour l'algorithme de sécurité

Message associé: ASecurityCnf.

7.2.10 ASecurityCnf

Classe: 2 (classe additionnelle).

Description: message de confirmation envoyé au PUF par le NAF dès qu'une demande ASecurityReq est satisfaite. L'identificateur RequestID met ce message de confirmation en correspondance avec la demande ASecurityReq appropriée.

La valeur "succès" du paramètre CompletionStatus indique que l'algorithme de sécurité requis a été activé ou arrêté pour la connexion demandée; sinon la cause de non-activation de l'algorithme de sécurité est retournée. Cette cause dépend de l'algorithme.

Paramètres:

Nom	Fourni	Commentaire
RequestID	C	fourni si contenu dans le message de requête
CompletionStatus	M	statut d'exécution de l'opération ASecurity du NAF

Message associé: ASecurityReq.

7.2.11 AManufacturerReq

Classe: 3 (classe additionnelle).

Description: ce message permet au PUF de demander au NAF de fournir une propriété spécifique du constructeur.

Il permet de gérer une propriété privée non assurée par l'interface PCI-RNIS.

Paramètres:

Nom	Requis	Commentaire
RequestID	M	identificateur de requête
ManufacturerCode	M	identifie le code de constructeur (fourni par celui-ci)

Remarques: les informations sur cette propriété sont obligatoires. Elles ne sont pas fournies sous forme de paramètre du message mais sont contenues dans la base de données.

Message associé: néant.

7.2.12 AManufacturerInd

Classe: 3 (classe additionnelle).

Description: ce message donne à un dispositif PUF des informations spécifiques concernant la propriété requise. Le dispositif NAF n'est autorisé à émettre que des indications de constructeur, lorsque le dispositif PUF a déjà émis au moins une requête concernant une caractéristique privée du constructeur.

Paramètres:

Nom	Fourni	Commentaire
RequestID	M	identificateur de requête
ManufacturerCode	M	identificateur du code constructeur (fourni par celui-ci)
CompletionStatus	O	identificateur du résultat qui est propre au constructeur

Remarques: si des informations sont fournies, elles doivent l'être non pas sous forme de paramètre dans un message mais de données en mémoire tampon.

Message associé: AManufacturerReq.

7.2.13 AChangeNCOReq

Classe: 4 (classe additionnelle).

Description: ce message est utilisé pour modifier le(s) paramètre(s) d'un objet NCO existant.

Seuls les objets NCO de type C, C/U1 ou C/U3 peuvent subir une modification de type (paramètre NCOType). Si l'objet NCO se rapporte à une connexion active, le dispositif NAF ne peut changer de protocole qu'en conditions stables.

Paramètres:

Nom	Requis	Commentaire
RequestID	O	identificateur de requête, émis par le PUF
NCOID	M	objet NCO à modifier
UDirection	O	détermine la manière d'utiliser l'objet NCO dans le plan d'utilisateur
UAttributeName	O	nom de l'attribut statique du plan d'utilisateur
UAttribute parameters	O	paramètres du plan d'utilisateur mutuellement exclusif avec UAttributeName; pour plus de détails, voir la spécification applicable au protocole de plan d'utilisateur du PCI-RNIS.
NCOType	O	spécification du nouveau type d'objet NCO
UAddress parameters	C	adresse dans le plan d'utilisateur; pour plus de détails, voir la spécification applicable au protocole de plan d'utilisateur du PCI-RNIS.

Message associé: AChangeNCOCnf.

7.2.14 AChangeNCOCnf

Classe: 4 (classe additionnelle).

Description: message de confirmation envoyé par le NAF au PUF pour répondre à une demande AChangeNCOReq. Si la confirmation est favorable, les modifications demandées sont immédiatement opérationnelles.

Paramètres:

Nom	Requis	Commentaire
RequestID	O	identificateur de requête, émis par le PUF.
NCOID	M	objet NCO à changer
CompletionStatus	M	statut d'exécution de l'opération de changement d'objet NCO

Message associé: AChangeNCOReq.

7.3 Messages dans le plan de commande

7.3.1 Introduction

7.3.1.1 Classes de messages du plan de commande

Les messages du plan de commande se subdivisent en sept classes, comme suit:

1) établissement d'une connexion et rupture d'une connexion;

2) envoi de messages spécifiques par numérotation avec chevauchement;

3) transfert d'informations d'utilisateur à utilisateur;

4) ajournement d'appels;

5) invocation de fonctionnalité;

6) équipement externe;

7) information additionnelle.

Comme décrit au 7.1.3, toutes ces classes ne sont pas forcément accessibles par l'intermédiaire de l'interface PCI-RNIS. Un dispositif NAF peut choisir de n'implémenter que certaines des catégories ci-dessus. Le mécanisme d'erreur indiquant au dispositif PUF qu'un message n'est pas disponible est décrit au 5.8.

Un dispositif PUF ne peut se fonder que sur la disponibilité des messages de classe 1. La disponibilité des autres classes dépend du dispositif NAF.

Le Tableau 7 donne un aperçu général des messages du plan de commande, avec la classe à laquelle ils appartiennent, et leur identificateur de message.

Tableau 7 – Messages du plan de commande

Identif. de mess.	Classe	Nom du message	But du message
201	1	CAlertReq	indication de la compatibilité avec l'appel entrant
202	1	CAlertInd	le terminal appelé indique qu'il peut traiter un appel
203	1	CConnectReq	lancement d'un appel sortant
204	1	CConnectInd	présence d'un appel entrant
205	1	CConnectRsp	acceptation d'un appel entrant
206	1	CConnectCnf	indication par le terminal appelé de l'acceptation d'un appel sortant
207	1	CDisconnectReq	suppression d'une connexion ou rejet d'un appel entrant
208	1	CDisconnectInd	indication que la connexion a été supprimée ou que l'appel sortant a été rejeté
209	1	CDisconnectRsp	confirmation de la fin d'une connexion
210	1	CDisconnectCnf	indication que l'autre terminal a mis fin à la connexion
212	1	CProgressInd	indication qu'un canal B est connecté
214	1	CStatusInd	indication d'une erreur de protocole
216	2	CSetupAckInd	indication qu'il faut des informations additionnelles pour lancer l'appel sortant
217	2	CConnectInfoReq	envoi d'informations additionnelles pour traiter l'appel
218	2	CProceedingInd	indication qu'aucune information d'établissement ne doit plus être acceptée pour cet appel
219	3	CUserInformationReq	envoi d'informations d'utilisateur à utilisateur
220	3	CUserInformationInd	présentation des informations d'utilisateur à utilisateur reçues
221	3	CCongestionControlReq	application d'opérations de commande de débit (contrôle de flux) à l'échange d'informations d'utilisateur à utilisateur
222	3	CCongestionControlInd	indication de l'opération de commande de débit à appliquer à l'échange d'informations d'utilisateur à utilisateur
223	4	CSuspendReq	suspension d'une connexion
224	4	CSuspendCnf	réponse à la demande de suspension d'une connexion
225	4	CResumeReq	reprise d'une connexion en suspens
226	4	CResumeCnf	réponse à la demande de suspension d'une connexion
228	4	CNotifyInd	indication d'un nouvel état pour une connexion
229	5	CFacilityReq	demande de ressource du réseau
230	5	CFacilityInd	indication de la fourniture d'une ressource par le réseau
232	6	CExtEquipAvailabilityInd	indication que l'équipement externe est ou n'est pas connecté au dispositif NAF
234	6	CExtEquipBlockDiallingInd	indication que l'appel est entièrement lancé par l'équipement externe (numérotation en bloc)
236	6	CExtEquipKeyPressedInd	communication au PUF du code d'une touche activée par pression
238	6	CExtEquipOffHookInd	indication que le combiné est décroché
240	6	CExtEquipOnHookInd	indication que le combiné est raccroché
241	7	CAddInfoReq	demande d'envoi d'informations additionnelles concernant un appel
242	7	CAddInfoInd	indication que des informations additionnelles ont été reçues concernant un appel

7.3.1.2 Séquencement des messages du plan de commande

Les Figures 6, 7, 8 et 9 présentent les diagrammes de transition d'état applicables à une connexion par dispositif PUF.

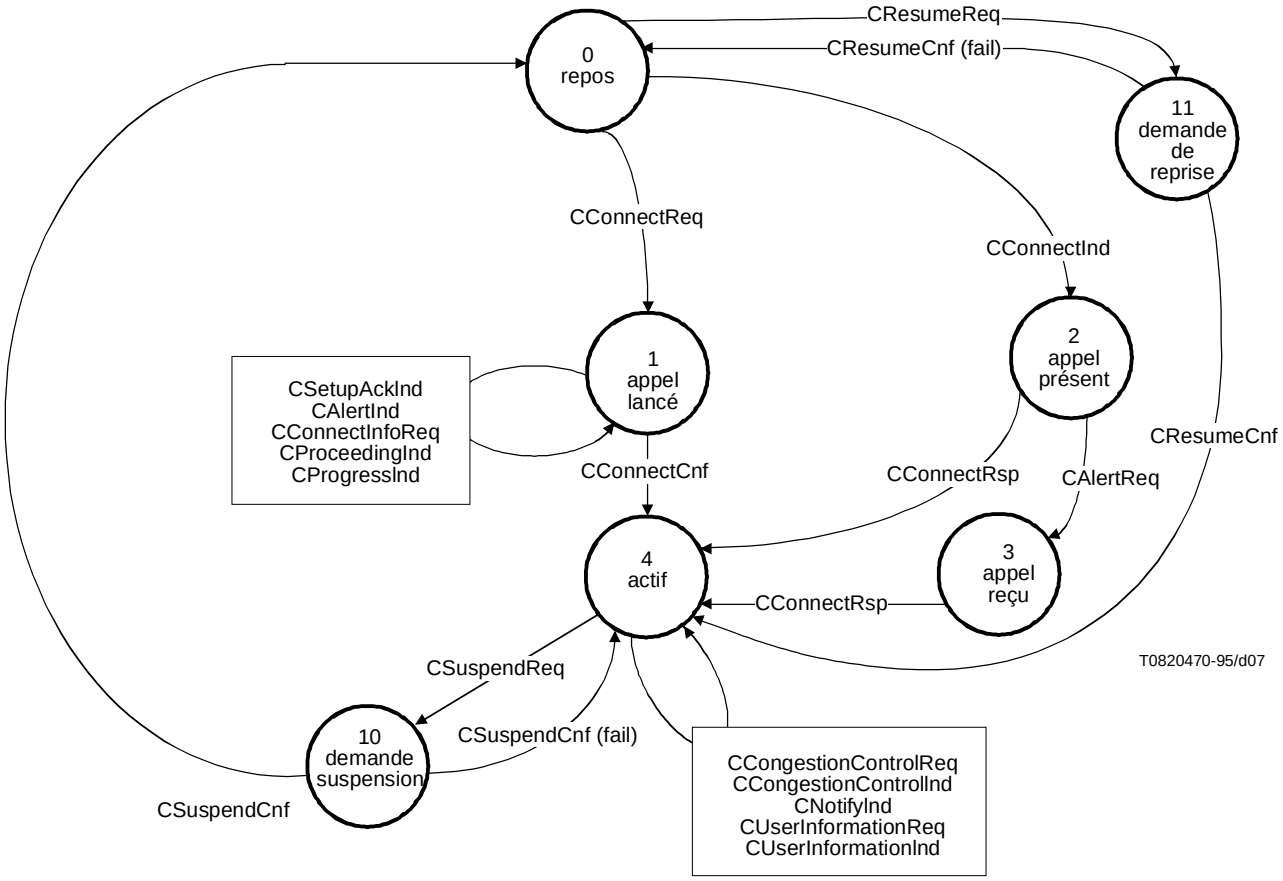


Figure 6 – Diagramme de transition d'état dans le plan de commande sans équipement externe ou avec équipement externe de type 1

NOTE – L'indication CExtEquipAvailabilityInd peut être utilisée dans tous les états. Elle provoque une transition à l'état 0 si l'équipement externe est indisponible.

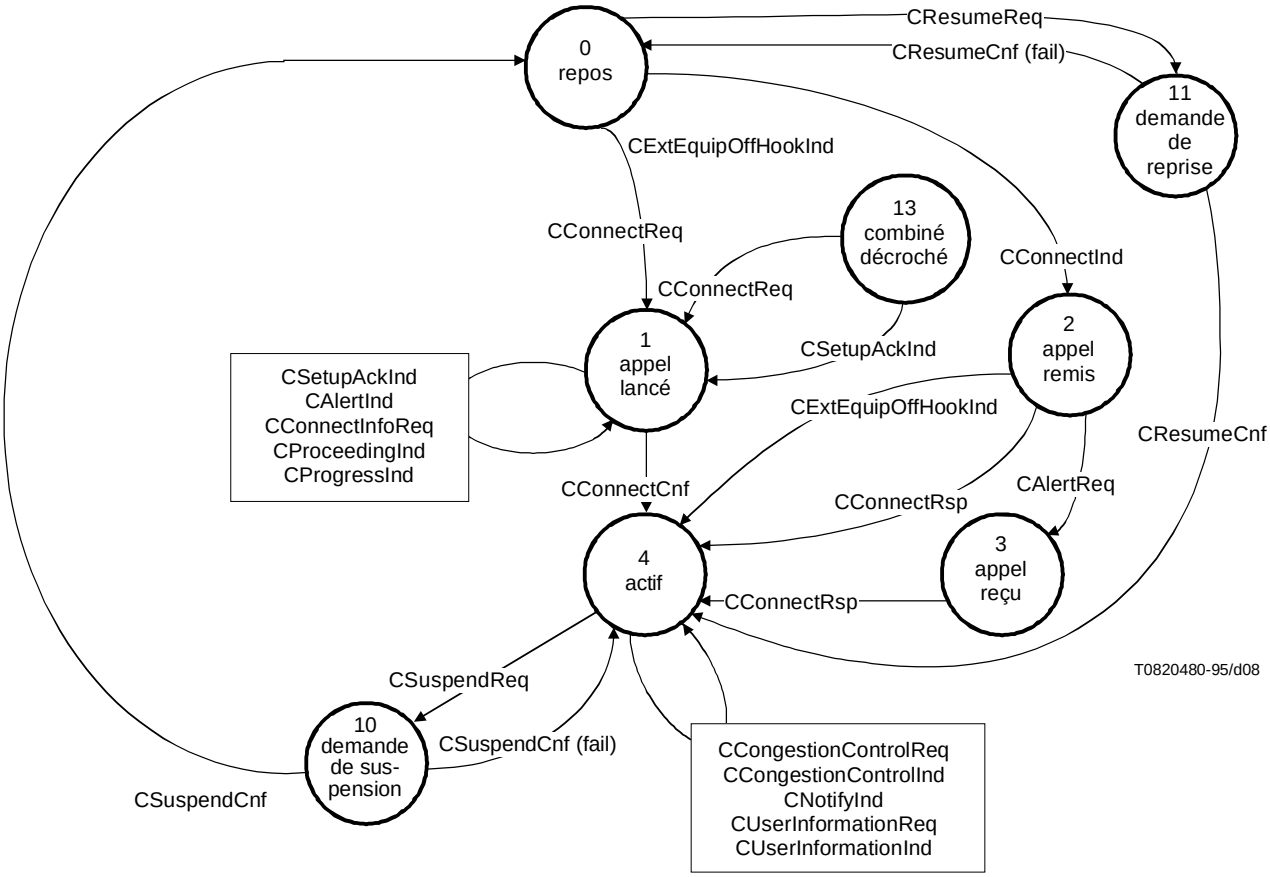


Figure 7 – Diagramme de transition d'état d'un équipement externe de type 2 ou 3 dans le plan de commande

NOTE 1 – L'indication CExtEquipOnHookInd peut être utilisée dans tous les états sauf celui de repos (0). Elle provoque une transition à l'état 0.

NOTE 2 – L'indication CExtEquipAvailabilityInd peut être utilisée dans tous les états. Elle provoque une transition à l'état 0 si l'équipement externe est indisponible.

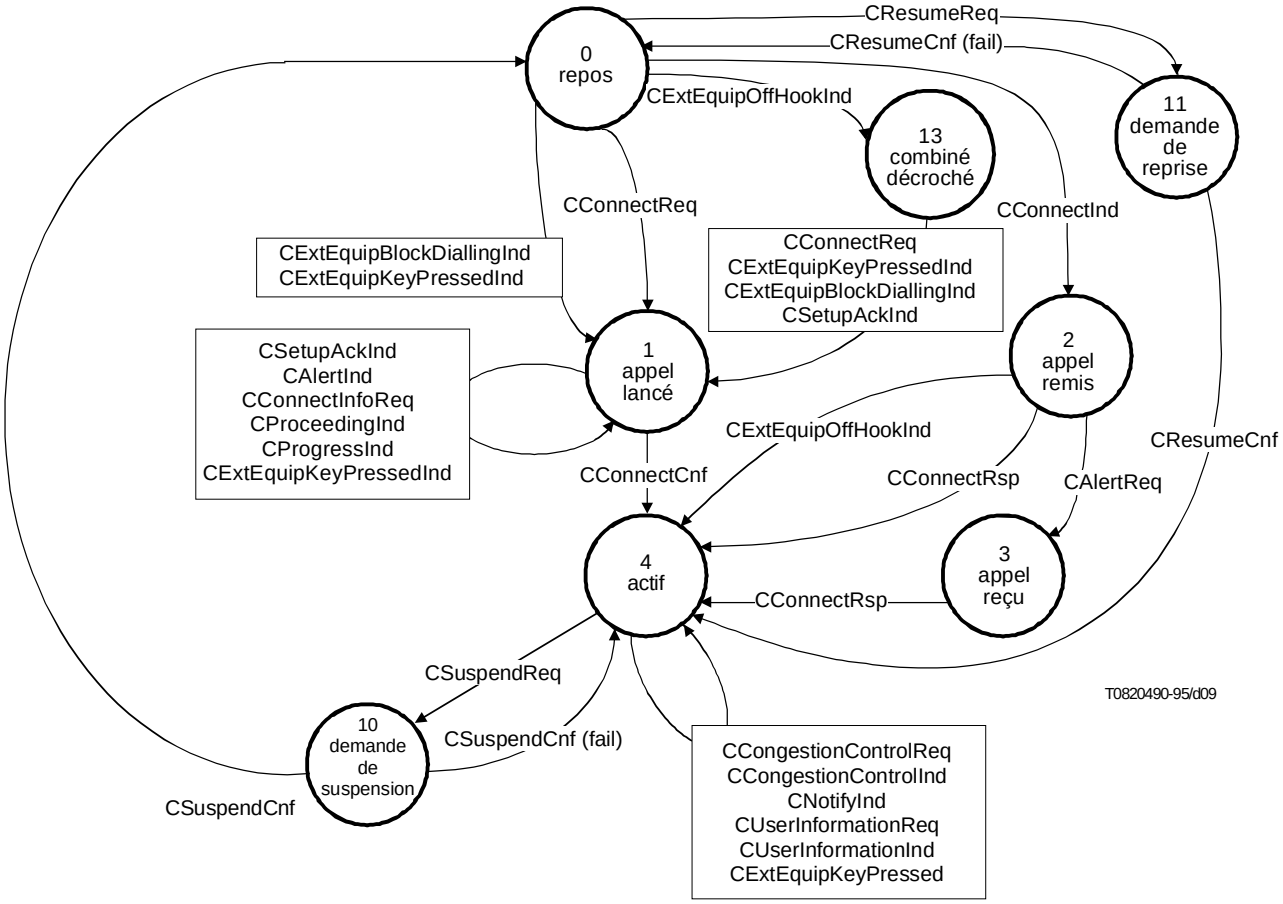


Figure 8 – Diagramme de transition d'état d'un équipement externe de type 4 ou 5 dans le plan de commande

NOTE 1 – L'indication CExtEquipOnHookInd peut être utilisée dans tous les états sauf celui de repos (0). Elle provoque une transition à l'état 0.

NOTE 2 – L'indication CExtEquipAvailabilityInd peut être utilisée dans tous les états. Elle provoque une transition à l'état 0 si l'équipement externe est indisponible.

NOTE 3 – Dans les Figures 6, 7 et 8 ci-dessus, si le dispositif PUF passe à l'état de repos (0) par réception d'une confirmation CSuspendCnf, la connexion est suspendue et peut être réutilisée. L'objet NCO toutefois, décrit toujours l'interaction entre dispositifs PUF et NAF pour cette connexion et ne peut pas être réutilisé. Le dispositif PUF doit réutiliser l'objet NCO lorsque la connexion est reprise ou rompue.

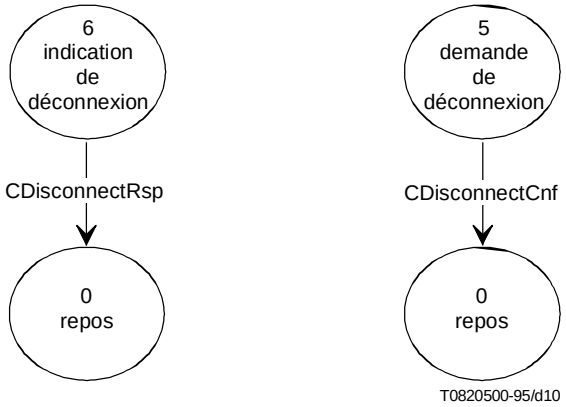


Figure 9 – Diagramme de transition d'état d'une connexion dans le plan de commande – Déconnexion

Remarques: l'indication CDisconnectInd peut être utilisée dans tous les états sauf 0, 5 et 6. Elle provoque une transition à l'état 6.

La demande CDisconnectReq peut être utilisée dans tous les états sauf 0, 5 et 6. Elle provoque une transition à l'état 5.

Les messages associés au réseau et les états intermédiaires complémentaires sont décrits au I.1.

Tout message CStatusInd peut provoquer une transition à l'état 0, s'il est envoyé par le réseau.

Un message CAddInfoReq ou CAddInfoInd ne modifie pas l'état d'un objet NCO.

NOTE 1 – Les Figures 6, 7, 8 et 9 ci-dessus ne donnent aucun renseignement concernant le transfert d'informations d'utilisateur à utilisateur. Ces messages, qui dépendent du niveau de service d'utilisateur à utilisateur, n'ont pas d'incidence sur l'état d'un appel du point de vue PUF.

NOTE 2 – Afin de simplifier l'interface, une fonction de filtrage peut être ajoutée; grâce à cette propriété, le dispositif PUF peut choisir le sous-ensemble de messages à manipuler. Une description détaillée de cette propriété fera l'objet d'une étude complémentaire.

7.3.2 CAlertReq

Classe: 1 (classe de base).

Description: ce message permet à un dispositif PUF d'indiquer sa compatibilité avec un appel entrant.

Paramètres:

Nom	Requis	Commentaire
NCOID	M	identifie l'appel
Facility	O	fonctionnalité – fonctionnement ou informations
ProgressIndicator	O	détails concernant la progression de l'appel
UserToUserInfo	O	informations à échanger entre utilisateurs RNIS

Remarques: la disponibilité du paramètre UserToUserInfo dépend du niveau de service d'utilisateur à utilisateur. On trouvera au 7.3.35 des détails sur les informations d'utilisateur à utilisateur.

Messages associés: CConnectReq, CConnectInd, CConnectRsp, CConnectCnf, CAlertInd.

7.3.3 CAlertInd

Classe: 1 (classe de base).

Description: le dispositif PUF reçoit ce message lorsque le terminal appelé a indiqué sa compatibilité.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie un appel
ChannelIdentification	O	identification du canal utilisé
Facility	O	fonctionnalité – fonctionnement ou informations
ProgressIndicator	O	détails sur la progression de l'appel
Display	O	information fournie par le réseau pour affichage
Signal	O	information fournie par le réseau concernant les tonalités
UserToUserInfo	O	informations à échanger entre utilisateurs RNIS

Remarques: la disponibilité du paramètre UserToUserInfo dépend du niveau de service d'utilisateur à utilisateur. On trouvera au 7.3.35 des détails sur les informations d'utilisateur à utilisateur.

Messages associés: CConnectReq, CConnectInd, CConnectRsp, CConnectCnf, CAlertInd.

7.3.4 CConnectReq

Classe: 1 (classe de base).

Description: ce message est envoyé par le dispositif PUF pour lancer un appel sortant vers l'adresse distante, qui peut être spécifiée dans le message ou dans les paramètres d'adresse utilisés pour créer l'objet NCO cité en référence.

Le dispositif PUF doit spécifier le paramètre BearerCap pour indiquer le type de canal support nécessaire. Ce paramètre doit être inséré dans le message ou avoir été spécifié dans les paramètres d'attribut utilisés pour créer l'objet NCO cité en référence.

Le dispositif PUF peut spécifier les paramètres de compatibilité LLC et HLC afin de préciser quels types de protocoles de couches inférieures ou de couches supérieures doivent être utilisés pour cet appel.

Paramètres:

Nom	Requis	Commentaire
NCOID	M	identifie l'appel. Cette information est fournie par le dispositif PUF.
CallingNumber	O	adresse locale (Note 1)
CallingSubaddress	O	sous-adresse locale (Note 1)
CalledNumber	O	adresse distante (Notes 1 et 2)
CalledSubaddress	O	sous-adresse distante (Notes 1 et 2)
ChannelIdentification	O	paramètre utilisé par le PUF pour indiquer le type de canal demandé. Pour les détails sur les valeurs possibles, voir au 7.6.12 le paramètre ChannelIdentification. Si cet identificateur n'est pas fourni, n'importe quel canal peut être désigné par défaut (Note 1).
BearerCap	O	capacité de transmission requise du canal (Note 1)
LLC	O	élément d'information compatibilité avec les couches inférieures (Note 1)
HLC	O	élément d'information compatibilité avec les couches supérieures (Note 1)
Keypad	O	élément d'information service de clavier
Facility	O	services complémentaires – fonctionnement ou information
UserToUserInfo	O	informations à échanger entre utilisateurs RNIS
NOTE 1 – Cette information peut être fournie lors de la création de l'objet NCO. Si elle est spécifiée aussi bien dans le message que dans l'objet NCO, le paramètre spécifié dans le message est utilisé et le paramètre contenu dans l'objet NCO est négligé. NOTE 2 – Un paramètre CalledNumber ou CalledSubaddress doit être fourni dans le message ou lors de la création de l'objet NCO.		

Remarques: la disponibilité du paramètre UserToUserInfo dépend du niveau de service d'utilisateur à utilisateur. On trouvera au 7.3.35 des détails sur les informations d'utilisateur à utilisateur.

Messages associés: CConnectCnf, CAlertReq, CAlertInd, CConnectInfoReq.

7.3.5 CConnectInd

Classe: 1 (classe de base).

Description: ce message offre un appel entrant à tous les dispositifs PUF appropriés (voir 7.7.1). A ce moment, l'appel est dans la phase d'établissement et aucune connexion n'est encore réalisée.

Le numéro de l'appelant peut être mis à la disposition du dispositif PUF. Dans ce cas, il doit être représenté dans les paramètres CallingNumber et CallingSubaddress.

Le dispositif PUF peut recevoir les paramètres BearerCap, LLC et HLC, qui doivent indiquer:

– quel type de canal support doit être utilisé;

– quel type de protocole de couche inférieure doit être utilisé pour cet appel;

– quel type de protocole de couche supérieure doit être utilisé pour cet appel.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie l'appel. Cet élément d'information est fourni par le NAF.
ChannelIdentification	O	identification du canal utilisé
CallingNumber	O	adresse distante
CallingSubaddress	O	sous-adresse distante
CalledNumber	O	adresse locale
CalledSubaddress	O	sous-adresse locale
BearerCap	O	ressource physique fournie dans le réseau
LLC	O	élément d'information compatibilité avec couches inférieures
HLC	O	élément d'information compatibilité avec couches supérieures
DateTime	O	date et heure
Facility	O	services complémentaires – fonctionnement ou information
Display	O	informations fournies par le réseau pour affichage
Signal	O	information fournie par le réseau concernant les tonalités
UserToUserInfo	O	informations à échanger entre utilisateurs RNIS

Remarques: lorsqu'un dispositif PUF reçoit l'information qu'aucun canal n'est disponible, ce dispositif peut libérer ou suspendre une communication afin de dégager un canal libre, s'il souhaite établir une connexion;

la disponibilité du paramètre UserToUserInfo dépend du niveau de service d'utilisateur à utilisateur. On trouvera au 7.3.35 des détails sur les informations d'utilisateur à utilisateur.

Messages associés: CConnectReq, CConnectRsp, CConnectCnf, CAlertReq, CAlertInd.

7.3.6 CConnectRsp

Classe: 1 (classe de base).

Description: ce message permet à un dispositif PUF d'accepter un appel entrant. Après l'envoi de ce message, le canal est considéré comme établi.

Le dispositif PUF peut fournir une nouvelle valeur pour le paramètre LLC, s'il est en cours de négociation des valeurs de ce paramètre.

Paramètres:

Nom	Requis	Commentaire
NCOID	M	identifie l'appel
ChannelIdentification	O	paramètre utilisé par le PUF pour indiquer le type de canal requis. Pour les détails relatifs aux valeurs possibles, voir le paramètre ChannelIdentification au 7.6.12. Une valeur pourra être indiquée si le canal B choisi par le PUF est différent des canaux présentés par le NAF.
LLC	O	élément d'information compatibilité avec couches inférieures
Facility	O	fonctionnalité – fonctionnement ou information
ProgressIndicator	O	détails concernant la progression de l'appel
UserToUserInfo	O	informations à échanger entre utilisateurs RNIS
ConnectedNumber	O	partie de l'adresse distante
ConnectedSubaddress	O	partie de l'adresse distante

Remarques: la disponibilité du paramètre UserToUserInfo dépend du niveau de service d'utilisateur à utilisateur. On trouvera au 7.3.35 des détails sur les informations d'utilisateur à utilisateur.

Messages associés: CConnectReq, CConnectInd, CConnectCnf, CAlertReq, CAlertInd.

7.3.7 CConnectCnf

Classe: 1 (classe de base).

Description: ce message est la réponse issue de l'appelé, indiquant que celui-ci accepte l'appel. Lorsque le dispositif PUF reçoit ce message, un canal est considéré comme établi.

Si des valeurs sont en cours de négociation pour le paramètre LLC, ce message peut fournir une nouvelle valeur pour ce paramètre.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie l'appel
ChannelIdentification	O	identification du canal utilisé
LLC	O	élément d'information compatibilité avec couches inférieures
DateTime	O	date et heure
Facility	O	fonctionnalité – fonctionnement ou information
Display	O	informations fournies par le réseau pour affichage
ProgressIndicator	O	détails concernant la progression de l'appel
Signal	O	information fournie par le réseau sur les tonalités
UserToUserInfo	O	informations à échanger entre utilisateurs RNIS
ConnectedNumber	O	partie de l'adresse distante
ConnectedSubaddress	O	partie de l'adresse distante

Remarques: la disponibilité du paramètre UserToUserInfo dépend du niveau de service d'utilisateur à utilisateur. On trouvera au 7.3.35 des détails sur les informations d'utilisateur à utilisateur.

Messages associés: CConnectReq, CConnectInd, CConnectRsp, CAlertReq, CAlertInd.

7.3.8 CDisconnectReq

Classe: 1 (classe de base).

Description: ce message permet au dispositif PUF de lancer la rupture d'une connexion ou de refuser un appel.

Ce message doit être confirmé par un message CDisconnectCnf.

Le dispositif PUF peut indiquer la cause de la rupture d'une connexion ou refuser un appel en fournissant le paramètre CauseToNAF.

Paramètres:

Nom	Requis	Commentaire
NCOID	M	identifie l'appel
CauseToNAF	O	raison fournie par le PUF pour déconnecter l'appel. Si cette cause n'est pas fournie par le PUF, la cause #16 "libération normale d'appel" doit être fournie par le NAF.
Facility	O	fonctionnalité – fonctionnement ou information
UserToUserInfo	O	informations à échanger entre utilisateurs RNIS

Remarques: la disponibilité du paramètre UserToUserInfo dépend du niveau de service d'utilisateur à utilisateur. On trouvera au 7.3.35 des détails sur les informations d'utilisateur à utilisateur.

Messages associés: CDisconnectInd, CDisconnectRsp, CDisconnectCnf.

7.3.9 CDisconnectInd

Classe: 1 (classe de base).

Description: ce message informe le dispositif PUF que l'utilisateur distant a lancé la rupture de la connexion ou a rejeté l'appel.

Le dispositif PUF doit confirmer ce message par une réponse CDisconnectRsp.

Le paramètre CauseToPUF doit indiquer la cause de la déconnexion ou du rejet.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie l'appel
CauseToPUF	M	raison pour laquelle l'appel va être déconnecté. Si cette cause n'est pas fournie par le réseau, le NAF doit introduire la cause #16: "libération normale d'appel". Voir également la Remarque.
Facility	O	fonctionnalité – fonctionnement ou information
Display	O	informations fournies par le réseau pour affichage
ProgressIndicator	O	détails concernant la progression de l'appel
Signal	O	information fournie par le réseau concernant les tonalités
UserToUserInfo	O	informations à échanger entre utilisateurs RNIS

Remarques: le réseau ne doit transférer qu'une seule cause au dispositif NAF, de sorte que le dispositif PUF ne reçoive qu'une seule cause.

La disponibilité du paramètre UserToUserInfo dépend du niveau de service d'utilisateur à utilisateur. On trouvera au 7.3.35 des détails sur les informations d'utilisateur à utilisateur.

Messages associés: CDisconnectReq, CDisconnectRsp, CDisconnectCnf.

7.3.10 CDisconnectRsp

Classe: 1 (classe de base).

Description: par ce message, le dispositif PUF confirme qu'une connexion s'est terminée ou qu'un appel a été refusé. Du point de vue du dispositif PUF, le canal est donc libéré et l'identificateur NCOID peut être réutilisé par le NAF.

Ce message est envoyé par le PUF pour confirmer l'indication CDisconnectInd.

Paramètres:

Nom	Requis	Commentaire
NCOID	M	identifie l'appel
Facility	O	fonctionnalié – fonctionnement ou information

Messages associés: CDisconnectReq, CDisconnectInd, CDisconnectCnf.

7.3.11 CDisconnectCnf

Classe: 1 (classe de base).

Description: par ce message, le dispositif PUF est informé que la connexion s'est terminée ou qu'un appel a été refusé, et que le canal a été libéré. L'identificateur NCOID peut alors être réutilisé par le dispositif NAF.

Ce message est la confirmation du message CDisconnectReq, par l'utilisateur distant ou par le réseau.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie l'appel
CauseToPUF	O	raison pour laquelle une demande de service complémentaire a été rejetée par le réseau
Facility	O	fonctionnalité – fonctionnement ou information
Display	O	informations fournies par le réseau pour affichage
Signal	O	information fournie par le réseau concernant les tonalités

Messages associés: CDisconnectReq, CDisconnectInd, CDisconnectRsp.

7.3.12 CProgressInd

Classe: 1 (classe de base).

Description: le dispositif PUF reçoit ce message lorsque l'information est disponible dans le canal B ou en cas d'interfonctionnement. Le canal doit être connecté.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie l'appel
ChannelIdentification	O	identification du canal utilisé
CauseToPUF	O	cause du message
Display	O	informations fournies par le réseau pour affichage
ProgressIndicator	M	détails sur la progression de l'appel
UserToUserInfo	O	informations à échanger entre utilisateurs RNIS

Messages associés: CConnectReq, CConnectInd, CConnectRsp, CConnectCnf, CAlertInd.

7.3.13 CStatusInd

Classe: 1 (classe de base).

Description: par ce message, le dispositif PUF est informé de l'apparition d'une erreur de protocole de signalisation, telle que définie au 7.8.8.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie l'appel
CauseToPUF	M	identifie l'erreur de protocole qui s'est produite

Message associé: néant.

7.3.14 CSetupAckInd

Classe: 2 (classe additionnelle).

Description: le dispositif PUF reçoit ce message lorsqu'il faut des informations additionnelles pour exécuter l'appel, dans le cas d'une numérotation avec chevauchement.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie l'appel
ChannelIdentification	O	identification du canal utilisé
Display	O	informations fournies par le réseau pour affichage
ProgressIndicator	O	détails sur la progression de l'appel

Messages associés: CConnectInfoReq, CConnectReq.

7.3.15 CConnectInfoReq

Classe: 2 (classe additionnelle).

Description: ce message permet à un dispositif PUF d'utiliser la technique de numérotation avec chevauchement pour l'établissement d'une connexion. La numérotation avec chevauchement signifie que le dispositif PUF fournit l'information d'adresse en plusieurs étapes: un message CConnectReq avec information incomplète d'adresse peut être suivi de plusieurs messages CConnectInfoReq, jusqu'à ce que l'adresse soit complète. Ce mécanisme est analogue à la numérotation sur un clavier.

Paramètres:

Nom	Requis	Commentaire
NCOID	M	identifie l'appel
CalledNumber	M	partie de l'adresse distante (Note 1)
NumberComplete	O	indique que ce message contient la dernière partie du numéro appelé, du point de vue d'un PUF (Note 2).
NOTE 1 – A chaque message CConnectInfoReq, le dispositif NAF accumule des informations d'adresse. Le dispositif PUF n'indique pas que l'information d'adresse est complète; on peut déduire cela d'après la réception d'un message CProceedingInd. Une sous-adresse ne peut être spécifiée que dans le message CConnectReq. Cela est dû aux limitations imposées par le protocole de signalisation dans le canal D (message SETUP du réseau). NOTE 2 – Si ce paramètre est inclus, aucun autre message CConnectInfoReq ne sera accepté par le dispositif NAF pour cet appel.		

Remarques: avant l'envoi de ce message, le dispositif PUF doit avoir envoyé un message CConnectReq avec la première partie de l'information sur le numéro appelé.

Messages associés: CConnectReq, CProceedingInd, CSetupAckInd.

7.3.16 CProceedingInd

Classe: 2 (classe additionnelle).

Description: le dispositif PUF reçoit ce message lorsque aucune information d'établissement ne sera plus acceptée, dans le cas de la numérotation avec chevauchement. Etant donné que le réseau peut ne pas envoyer ce message, le dispositif PUF ne peut pas compter sur sa réception.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie l'appel
ChannelIdentification	O	identification du canal utilisé
Display	O	informations fournies par le réseau pour affichage
ProgressIndicator	O	détails sur la progression d'appel

Messages associés: CConnectReq, CConnectInfoReq, CSetupAckInd.

7.3.17 CUserInformationReq

Classe: 3 (classe additionnelle).

Description: ce message permet à un dispositif PUF de demander l'envoi d'informations d'utilisateur à utilisateur sur une connexion établie.

L'état d'appel qui permet au dispositif PUF d'envoyer des informations d'utilisateur à utilisateur dépend du niveau de service d'utilisateur à utilisateur fourni par le réseau ou profilé par abonnement. Voir au 7.3.35 des détails sur les informations d'utilisateur à utilisateur.

Paramètres:

Nom	Requis	Commentaire
NCOID	M	identifie l'appel
MoreData	O	indique à l'entité homologue qu'un autre message d'utilisateur à utilisateur suit
UserToUserInfo	M	informations à échanger entre utilisateurs RNIS

Remarques: ce message n'est disponible que si le niveau de service d'utilisateur à utilisateur est au moins égal à 2. On trouvera au 7.3.35 des détails sur les informations d'utilisateur à utilisateur.

Messages associés: CUserInformationInd, CCongestionControlReq, CCongestionControlInd.

7.3.18 CUserInformationInd

Classe: 3 (classe additionnelle).

Description: ce message permet à un dispositif NAF de présenter au dispositif PUF des informations d'utilisateur à utilisateur reçues au sujet d'une connexion établie.

L'état d'appel qui permet la réception d'informations d'utilisateur à utilisateur dépend du niveau de service d'utilisateur à utilisateur fourni par le réseau ou profilé par abonnement. On trouvera au 7.3.35 des détails sur les informations d'utilisateur à utilisateur.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie l'appel
MoreData	O	si ce paramètre est présent, l'entité homologue indique qu'un autre message d'utilisateur à utilisateur suit.
UserToUserInfo	M	informations échangées entre utilisateurs RNIS

Remarques: ce message n'est disponible que si le niveau de service d'utilisateur à utilisateur est au moins égal à 2. On trouvera au 7.3.35 des détails sur les informations d'utilisateur à utilisateur.

Messages associés: CUserInformationReq, CCongestionControlReq, CCongestionControlInd.

7.3.19 CCongestionControlReq

Classe: 3 (classe additionnelle).

Description: ce message permet à un dispositif PUF d'appliquer des opérations de commande de débit au sujet des informations d'utilisateur à utilisateur fournies par le message CUserInformationInd.

L'opération de commande de débit n'est définie que pour le fonctionnement du côté local de la connexion. La commande de débit fonctionne au moyen du mécanisme prêt/non prêt. La condition initiale pour l'échange d'informations d'utilisateur à utilisateur est prêt. Pour régler la condition de commande de débit, le dispositif PUF doit régler le paramètre CongestionLevel à la valeur appropriée.

Paramètres:

Nom	Requis	Commentaire
NCOID	M	identifie l'appel
CongestionLevel	M	valeur de commande de débit
CauseToNAF	O	paramètre à inclure si une information est perdue
NOTE – Ce message n'est disponible que si un niveau de service d'utilisateur à utilisateur de valeur au moins égale à 2 a été souscrit. Voir au 7.3.35 les détails sur les informations d'utilisateur à utilisateur.		

Remarques: pour la commande de débit indiquée par ce message, la valeur prêt est considérée comme étant le statut initial. La commande de débit dans chaque sens doit être activée indépendamment. Ce message n'a qu'une signification locale.

Messages associés: CUserInformationReq, CUserInformation Ind, CCongestionControlInd.

7.3.20 CCongestionControlInd

Classe: 3 (classe additionnelle).

Description: ce message permet à un dispositif NAF d'indiquer à un dispositif PUF que des opérations de commande de débit ont été appliquées aux informations d'utilisateur à utilisateur fournies par le message UserInformationReq.

L'opération de commande de débit n'est définie que pour le fonctionnement du côté local de la connexion. La commande de débit fonctionne au moyen du mécanisme prêt/non prêt. La condition initiale pour l'échange d'informations d'utilisateur à utilisateur est prêt. Le paramètre CongestionLevel doit indiquer la nouvelle valeur de commande de débit pour le transfert d'informations d'utilisateur à utilisateur vers le dispositif PUF.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie l'appel
CongestionLevel	M	valeur de commande de débit
CauseToPUF	O (Note)	paramètre à inclure en cas de perte d'information
Display	O	informations fournies par le réseau pour affichage
NOTE – Le réseau ne doit transférer qu'une seule cause au NAF, de façon que le PUF ne reçoive qu'une seule cause.		

Remarques: ce message n'est disponible que si le niveau de service d'utilisateur à utilisateur est au moins égal à 2. On trouvera au 7.3.35 des détails sur les informations d'utilisateur à utilisateur.

Pour la commande de débit indiquée par ce message, la valeur prêt est considérée comme étant le statut initial.

Ce message n'a qu'une signification locale. La commande de débit dans chaque sens doit être activée indépendamment.

Messages associés: CUserInformationReq, CUserInformation Ind, CCongestionControlReq.

7.3.21 CSuspendReq

Classe: 4 (classe additionnelle).

Description: ce message permet à un dispositif PUF de suspendre, mais pas de rompre, une connexion.

Après l'envoi de ce message, le dispositif PUF doit être informé de la suspension de la connexion.

Paramètre:

Nom	Requis	Commentaire
NCOID	M	identifie l'appel

Remarques: l'utilisation de ce message en conjonction avec l'application d'un protocole sur la connexion relève de la responsabilité du dispositif PUF.

Lors de la suspension d'une connexion, on ne garantit pas que la connexion pourra être reprise par la suite.

Messages associés: CSuspendCnf, CResumeReq, CResumeCnf, CNotifyInd.

7.3.22 CSuspendCnf

Classe: 4 (classe additionnelle).

Description: ce message est la réponse à un message CSuspendReq. Le dispositif NAF fournit au dispositif PUF le résultat de sa demande de mise en suspens.

Le paramètre Response doit indiquer si la connexion est suspendue.

Si le dispositif PUF reçoit un message CSuspendCnf, la connexion est mise en suspens et peut être réutilisée. L'objet NCO continue toutefois à décrire l'interaction entre PUF et NAF pour cette connexion et ne doit donc pas être réutilisé. Le PUF devra réutiliser l'objet NCO lorsque la connexion sera reprise ou rompue.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie l'appel
CompletionStatus	M	indique le statut de la suspension: – succès si la suspension est acceptée; – échec si la suspension a échoué.
CauseToPUF	C (Note)	obligatoire si la suspension est refusée; indique la cause du refus de suspension. absent en cas de succès.
Display	O	informations fournies par le réseau pour affichage
NOTE – Le réseau ne doit transférer qu'une seule cause au NAF, de façon que le PUF ne reçoive qu'une seule cause.		

Messages associés: CSuspendReq, CResumeReq, CResumeCnf, CNotifyInd.

7.3.23 CResumeReq

Classe: 4 (classe additionnelle).

Description: ce message permet à un dispositif PUF de reprendre, c'est-à-dire de rétablir une connexion mise en suspens.

Après l'envoi de ce message, le PUF doit être informé du rétablissement de la connexion suspendue.

Paramètre:

Nom	Requis	Commentaire
NCOID	M	identifie l'appel

Messages associés: CSuspendReq, CSuspendCnf, CResumeCnf, CNotifyInd.

7.3.24 CResumeCnf

Classe: 4 (classe additionnelle).

Description: ce message est la réponse à un message CResumeReq. Le dispositif NAF fournit au dispositif PUF le résultat de sa demande de reprise.

Le paramètre de réponse doit indiquer si la connexion est reprise.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie l'appel
CompletionStatus	M	indique le statut de la reprise: – succès si l'opération est acceptée; – échec si l'opération a échoué.
CauseToPUF	C (Note)	obligatoire si l'opération est refusée; indique la cause du refus de l'opération. absent en cas de succès.
Display	O	informations fournies par le réseau pour affichage
NOTE – Le réseau ne doit transférer qu'une seule cause au NAF, de façon que le PUF ne reçoive qu'une seule cause.		

Remarques: le résultat de la reprise d'une connexion peut être défavorable (échec d'opération) si le NAF ou le réseau ne dispose pas de ressources (c'est-à-dire de canaux) pour rétablir la connexion.

Messages associés: CSuspendReq, CSuspendCnf, CResumeReq, CNotifyInd.

7.3.25 CNotifyInd

Classe: 4 (classe additionnelle).

Description: ce message est fourni par le dispositif NAF pour indiquer au dispositif PUF un nouvel état de la connexion.

Par exemple, ce message peut être émis si l'utilisateur distant suspend ou reprend une connexion.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie l'appel
NotificationIndicator	M	nouvel état
Display	O	informations fournies par le réseau pour affichage

Messages associés: CSuspendReq, CSuspendCnf, CResumeReq, CResumeCnf.

7.3.26 CFacilityReq

Classe: 5 (classe additionnelle).

Description: ce message permet au dispositif PUF de demander au réseau une ressource qui peut être associée à une connexion établie.

On trouvera au 7.6.26 des détails sur l'utilisation des messages et paramètres relatifs aux ressources et sur le codage du paramètre Facility.

Paramètres:

Nom	Requis	Commentaire
NCOID	O	fourni par le PUF si la ressource est associée à une connexion établie
Facility	M	services complémentaires – fonctionnement ou information
NOTE – Si le PUF fournit un codage transparent de l'élément d'information de ressource, toutes les informations venant à la suite de ce codage transparent devront être renvoyées en transparence.		

Message associé: CFacilityInd.

7.3.27 CFacilityInd

Classe: 5 (classe additionnelle).

Description: ce message fournit au dispositif PUF la capacité de demander au réseau une ressource qui peut être associée à une connexion établie.

On trouvera au 7.6.26 des détails sur l'utilisation des messages et paramètres relatifs aux ressources et sur le codage du paramètre Facility.

Paramètres:

Nom	Fourni	Commentaire
NCOID	O	fourni par le NAF si la ressource est associée à une connexion établie
Facility	M (Note)	services complémentaires – fonctionnement ou information
Display	O	informations fournies par le réseau pour affichage
NOTE – Si le PUF fournit un codage transparent de l'élément d'information de ressource, toutes les informations venant à la suite de ce codage transparent devront être renvoyées en transparence.		

Message associé: CFacilityReq.

7.3.28 CExtEquipAvailabilityInd

Classe: 6 (classe additionnelle).

Description: par ce message, le dispositif PUF est informé de la disponibilité de l'équipement externe. Lorsqu'une connexion est active, le dispositif NAF est chargé d'interrompre la communication si l'équipement externe devient indisponible.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie l'appel. Cet élément d'information est fourni par le NAF.
ExtEquipAvailability	M	indique la disponibilité de l'équipement externe

Message associé: néant.

7.3.29 CExtEquipBlockDiallingInd

Classe: 6 (classe additionnelle).

Description: avec ce message, le dispositif PUF obtient les informations de numérotation introduites par l'utilisateur au moyen du clavier de l'équipement externe en cas de numérotation en bloc. Ce message contient l'adresse et/ou la sous-adresse distante(s) complète(s).

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie l'appel. Cet élément d'information est fourni par le NAF.
ExtEquipBlockDialling	M	fournit au PUF l'adresse et/ou la sous-adresse distante(s) dans le cas où l'équipement externe permet la numérotation en bloc

Message associé: néant.

7.3.30 CExtEquipKeyPressedInd

Classe: 6 (classe additionnelle).

Description: avec ce message, le dispositif PUF obtient les informations de numérotation introduites par l'utilisateur au moyen du clavier de l'équipement externe en cas de numérotation en bloc. Un seul message est envoyé au PUF à chaque pression de touche.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie l'appel. Cet élément d'information est fourni par le NAF.
ExtEquipKeyPressed	M	envoie au PUF le code de la touche pressée si l'équipement externe numérote avec chevauchement

Message associé: néant.

7.3.31 CExtEquipOffHookInd

Classe: 6 (classe additionnelle).

Description: avec ce message, le dispositif PUF est informé du fait que le combiné de l'équipement externe est décroché. Selon le type d'équipement externe et l'état actuel de la connexion, ce message peut être interprété de différentes manières (voir les Figures 6, 7 et 8).

Paramètre:

Nom	Fourni	Commentaire
NCOID	M	identifie l'appel. Cet élément d'information est fourni par le NAF.

Message associé: CExtEquipOnHookInd.

7.3.32 CExtEquipOnHookInd

Classe: 6 (classe additionnelle).

Description: par ce message, le dispositif PUF est informé que le combiné de l'équipement externe est raccroché. Selon le type d'équipement externe et l'état actuel de la connexion, ce message peut être interprété de différentes manières (voir les Figures 6, 7 et 8).

Paramètre:

Nom	Fourni	Commentaire
NCOID	M	identifie l'appel. Cet élément d'information est fourni par le NAF.

Message associé: CExtEquipOffHookInd.

7.3.33 CAddInfoReq

Classe: 7 (classe additionnelle).

Description: ce message permet à un dispositif PUF d'envoyer des informations additionnelles concernant un appel. Les informations de numérotation avec chevauchement sont insérées dans le message CConnectInfoReq (classe 2). Ce message peut être utilisé pour acheminer des informations associées au réseau (par exemple une procédure d'identification particulière).

Paramètres:

Nom	Requis	Commentaire
NCOID	M	identifie l'appel
AdditionalInformation	M	achemine des informations associées au réseau

Message associé: CAddInfoInd.

7.3.34 CAddInfoInd

Classe: 7 (classe additionnelle).

Description: ce message permet à un dispositif NAF d'envoyer des informations additionnelles concernant un appel et fournies par le réseau (par exemple une procédure d'identification particulière).

Paramètres:

Nom	Requis	Commentaire
NCOID	M	identifie l'appel
Display	O	informations fournies par le réseau pour affichage
AdditionalInformation	M	achemine des informations associées au réseau

Message associé: CAddInfoReq.

7.3.35 Echange d'informations d'usager à usager

L'utilisation de l'échange d'informations d'utilisateur à utilisateur dépend du niveau de service d'utilisateur à utilisateur fourni par le réseau ou profilé par abonnement.

Dans la Recommandation Q.931 [1], trois niveaux sont définis pour le service d'utilisateur à utilisateur:

– *service 1*:

  informations d'utilisateur à utilisateur échangées lors de l'établissement et de la phase de libération d'une communication.

– *service 2*:

  informations d'utilisateur à utilisateur échangées lors de l'établissement de la communication.

– *service 3*:

  informations d'utilisateur à utilisateur échangées lorsqu'un appel est dans l'état "active".

Pour le dispositif PUF, l'utilisation du paramètre UserToUserInfo à l'intérieur des messages et de messages UserInformation dépend du niveau de service.

En fonction du niveau de service, on définit comme suit l'usage du paramètre UserToUserInfo et des messages de type UserInformation:

– *niveau de service 1*:

  insertion du paramètre UserToUserInfo dans les messages suivants:

  - CAlertReq;
  - CAlertInd;
  - CConnectReq;
  - CConnectInd;
  - CConnectRsp;
  - CConnectCnf;
  - CDisconnectReq;
  - CDisconnectInd.

– *niveau de service 2*:

  utilisation des messages de type CUserInformation entre les opérations d'émission/réception de messages CAlertReq/Ind et CConnetRsp/Cnf.

– *niveau de service 3*:

  utilisation des messages de type CUserInformation dans l'état "actif" d'une communication.

Ces trois niveaux de service peuvent être utilisés séparément ou selon toute combinaison avec un appel donné.

NOTE – Les services 2 et 3 sont actuellement fournis au moyen de la méthode décrite dans la Recommandation Q.931 [1].

7.4 Implémentation de services complémentaires

Une Recommandation de la série des Recommandations portant sur le PCI-RNIS et décrivant en détail la façon d'utiliser les services complémentaires fera l'objet d'une étude complémentaire.

7.5 Messages dans le plan d'utilisateur

Les messages du plan d'utilisateur donnent accès à différentes piles de protocoles de ce plan. Selon le protocole de plan utilisateur choisi, le plan d'utilisateur donnera accès à une connexion de couche Réseau (couche 3), à une connexion de couche Liaison de données (couche 2) ou à une connexion de couche Physique transparente (couche 1). Une description détaillée des protocoles disponibles et des messages, séquences et paramètres correspondants du plan d'utilisateur se trouve dans les Parties 3 à 6.

7.6 Paramètres des messages

Le présent sous-paragraphe décrit les paramètres utilisés pour les messages du plan d'administration et du plan de commande. Les paramètres pour les messages du plan d'utilisateur sont décrits dans les Parties 3 à 6.

Les paramètres décrits dans le présent sous-paragraphe apparaissent dans l'ordre alphabétique. La numérotation des types est conforme à l'Appendice III.

7.6.1 AdditionInformation

Description: ce paramètre est utilisé pour transmettre des informations additionnelles en provenance/à destination du dispositif PUF. Les informations sont acheminées en transparence à destination/en provenance du réseau. Pour plus de détails, voir la documentation relative au réseau ou au dispositif NAF.

Type: 80.

Champ	Type de champ	Sens	Requis	Commentaire
Value	chaîne IA5	B	M	128 est la longueur maximale. Elle peut être réduite en raison de contraintes dues au réseau.

7.6.2 Algorithm

Description: ce paramètre transmet au NAF le nom de l'algorithme de sécurité à utiliser.

Type: 1.

Champ	Type de champ	Sens	Requis	Commentaire
Algorithm	chaîne IA5	P	M	l'algorithme de sécurité est identifié par son nom. Les noms des algorithmes disponibles peuvent être obtenus au moyen des informations sur les propriétés, fournies par le dispositif NAF. la valeur "nosecurity" de ce paramètre indique que la sécurité n'est plus requise pour la connexion. 16 octets est la longueur maximale
NOTE – Une description détaillée de l'utilisation possible de ce paramètre se trouve dans la Partie 3 [2].				

7.6.3 BearerCap

Description: ce paramètre est utilisé pour transmettre la capacité support à destination/en provenance du PUF et, sur option, des informations de couche 1 si le paramètre LLC décrit au 7.6.31 est fourni dans l'appel.

Type: 3.

Champ	Type de champ	Sens	Requis	Commentaire
BearerCap	Octet-string	B	M	élément d'information capacité support longueur maximale: 12 octets
NOTE – Les valeurs pour ce champ sont définies dans la Recommandation Q.931.				

7.6.4 CalledNumber

Description: ce paramètre est utilisé pour transmettre des détails sur l'adresse appelée, à destination/en provenance du dispositif PUF.

Type: 7.

Champ	Type de champ	Sens	Requis	Commentaire
NumberType	octet	B	M	valeur par défaut (255) – valeur par défaut: type inconnu type inconnu (0) type international (1) type national spécifique (2) type réseau (3) type abonné (4) type abrégé (6)
NumberPlan	octet	N	M	valeur par défaut (255) – valeur par défaut: plan inconnu plan inconnu (0) plan RNIS (1) plan pour données (3) plan pour télex (4) plan national (8) plan de numérotage privé (9)
Number	chaîne IA5	B	C	20 octets est la longueur maximale. Ce champ peut être absent en cas de numérotation avec chevauchement ou s'il est associé au message CConnectReq. Sinon, il est obligatoire.
NOTE – Dans l'échange de messages de PUF à NAF, ce paramètre doit être fourni soit dans l'objet NCO ou dans le message approprié.				

7.6.5 CalledSubaddress

Description: ce paramètre est utilisé pour transmettre la sous-adresse de l'appelé, à destination/en provenance du dispositif PUF.

Type: 8.

Champ	Type de champ	Sens	Requis	Commentaire
NumberType	octet	B	M	valeur par défaut (255) – la valeur par défaut est point nsap point nsap (0) utilisateur (2)
Indicator	octet	B	M	pair (0) impair (1) ce champ n'est significatif que si le champ NumberType est mis à la valeur utilisateur (2). Il indique si le numéro contient un nombre pair ou impair de chiffres BCD (décimaux codés binaires).
Number	chaîne IA5	B	M	20 octets est la longueur maximale

7.6.6 CallingNumber

Description: ce paramètre est utilisé pour transmettre à destination/en provenance du PUF des détails sur l'adresse de l'appelant.

Type: 11.

Champ	Type de champ	Sens	Requis	Commentaire
NumberType	octet	B	M	valeur par défaut (255) – la valeur par défaut est type inconnu type inconnu (0) type international (1) type national spécifique (2) type réseau (3) type abonné (4) type abrégé (6)
NumberPlan	octet	B	M	valeur par défaut (255) – la valeur par défaut est plan inconnu plan inconnu (0) plan RNIS (1) plan pour données (3) plan pour télex (4) plan national (8) plan privé (9)
Presentation	octet	B	M	valeur par défaut (255) – la valeur par défaut est présentation autorisée présentation autorisée (0) présentation interdite (1) présentation indisponible (2) ce champ indique s'il y a lieu de fournir le numéro à l'appelé
Screening	octet	B	M	valeur par défaut (255) – la valeur par défaut est utilisateur non contrôlé utilisateur non contrôlé (0) vérifié par utilisateur (1) assuré par le réseau (3) ce champ indique qu'un certain contrôle a été appliqué au numéro
Number	chaîne IA5	B	M	20 octets est la longueur maximale

NOTE – Si l'on utilise le service complémentaire d'identification d'appel (IA), seules les valeurs "plan (de numérotage) RNIS(/téléphonie)" et "plan inconnu" doivent être autorisées pour le PUF afin d'identifier le plan de numérotage dans l'élément d'information numéro de l'appelant.

Si l'on utilise le service complémentaire d'identification d'appel (IA) et que l'on spécifie un numéro complet, seules les valeurs "type abonné", "type national" et "type international" doivent être autorisées pour le PUF en tant que type de numéro dans l'élément d'information numéro de l'appelant.

Si l'on utilise le service complémentaire d'identification d'appel (IA) et que l'on spécifie un numéro incomplet pour la sélection directe à l'arrivée, seule la valeur "type inconnu" doit être autorisée pour le PUF en tant que type de numéro.

7.6.7 CallingSubaddress

Description: ce paramètre est utilisé pour transmettre à destination/en provenance du PUF des détails de sous-adresse d'appelant.

Type: 12.

Champ	Type de champ	Sens	Requis	Commentaire
NumberType	octet	B	M	valeur par défaut (255) – la valeur par défaut est point nsap point nsap (0) utilisateur (2)
Indicator	octet	B	M	pair (0) impair (1) ce champ n'est significatif que si le type de numéro est mis à utilisateur. Il indique si le numéro contient un nombre impair ou pair de chiffres.
Number	chaîne IA5	B	M	20 octets est la longueur maximale

7.6.8 CAttributeName

Description: ce paramètre est utilisé pour transmettre, en provenance du PUF, le nom d'un ensemble statique d'attributs du plan de commande.

Type: 13.

Champ	Type de champ	Sens	Requis	Commentaire
AttributeName	chaîne IA5	P	M	16 octets est la longueur maximale

7.6.9 CauseToNAF

Description: ce paramètre est utilisé pour transmettre une information de cause du PUF au NAF.

Type: 14.

Champ	Type de champ	Sens	Requis	Commentaire
Cause	octet	P	M	valeur de cause

7.6.10 CauseToPUF

Description: ce paramètre est utilisé pour transmettre une information de cause du NAF au PUF.

Type: 15.

Champ	Type de champ	Sens	Requis	Commentaire
Cause	Octet	N	M	valeur de cause
Standard	Octet	N	M	valeur par défaut (255) – la valeur par défaut est "norme UIT" norme UIT (0) norme internationale (1) norme nationale (2) norme propre au réseau (3)
Location	Octet	N	M	valeur par défaut (255) – la valeur par défaut est "lieu d'utilisateur" lieu d'utilisateur (0) lieu privé (1) lieu public (2) centre de transit (3) lieu public distant (4) lieu privé distant (5) centre international (7) lieu hors réseau (10)
Recommendation	Octet	N	M	valeur par défaut (255) – la valeur par défaut est "Q.931" Q.931 (0) X.21 (3) X.25 (4)
Diagnostics	Octet-string	N	C	le champ de diagnostic dépend de la valeur de cause sa longueur est fixée à 2 octets. l'octet inférieur est l'octet le moins significatif

7.6.11 CDirection

Description: ce paramètre sert à transmettre au dispositif NAF des informations concernant l'utilisation d'un objet NCO particulier, pour le sous-système du plan de commande. Si ce paramètre est absent lors de la création de l'objet NCO, la valeur retenue pour cet objet sera dans les deux sens (3).

Type: 16.

Champ	Type de champ	Sens	Requis	Commentaire
Direction	octet	P	M	écoute (1) appel (2) dans les deux sens (3)

7.6.12 ChannelIdentification

Description: ce paramètre sert à transmettre à destination/en provenance du PUF des informations relatives aux canaux.

Type: 17.

Champ	Type de champ	Sens	Requis	Commentaire
Selection	octet	B	M	aucun canal (0) – aucun canal n'est disponible canal B (1) tout canal (3) – tout canal disponible peut être utilisé canal D (4)
Number	octet	B	O	ce paramètre facultatif est utilisé par le PUF pour sélectionner un canal B particulier. Une valeur de 255 signifie que l'on choisit le premier canal B disponible.

Remarques: pour le message CConnectReq, toutes les valeurs du champ Selection, sauf aucun canal et canal D, sont possibles.

Le champ Number peut être utilisé dans la structure des paramètres de type CAttribute de l'ensemble d'attributs du plan de commande ou dans le message CConnectReq afin de sélectionner un canal B ou D particulier en connexion permanente, si de multiples identificateurs de point d'extrémité du terminal (TEI) sont possibles.

7.6.13 ChargingInfo

Description: ce paramètre est utilisé pour transmettre les informations de taxation, si elles existent, concernant un objet NCO, dans les paramètres de type Attribute de l'ensemble d'attributs du plan d'administration.

Type: 18.

Champ	Type de champ	Sens	Requis	Commentaire
Tag	Octet	N	M	informations de taxation (3) erreur de taxation (4) (voir 7.6.26: codage du champ FacilityTag).
Value	Octet-string	N	C	la longueur et le contenu dépendent du champ Tag. Champ absent si le champ Tag a la valeur "erreur de taxation". (voir 7.6.26: codage du champ FacilityTag).

7.6.14 CompletionStatus

Description: ce paramètre est utilisé pour transmettre au PUF des informations sur l'aboutissement des appels.

Type: 19.

Champ	Type de champ	Sens	Requis	Commentaire
Status	Octet	N	M	valeur du rapport d'achèvement
ErrorSpecific	Octet-string	N	C	la présence de ce champ dépend de la valeur du champ Status. Pour plus de détails, voir 7.8. La longueur doit être comprise entre 0 et 16 octets.

7.6.15 CongestionLevel

Description: ce paramètre est utilisé pour transmettre à destination/en provenance du PUF des détails sur les niveaux d'encombrement.

Type: 20.

Champ	Type de champ	Sens	Requis	Commentaire
Level	octet	B	M	prêt (1) pas prêt (15)

7.6.16 ConnectedNumber

Description: ce paramètre est utilisé pour transmettre au PUF des détails concernant le numéro connecté.

Type: 21.

Champ	Type de champ	Sens	Requis	Commentaire
NumberType	octet	N	M	valeur par défaut: 255 – la valeur par défaut est "type inconnu" inconnu (0) international (1) national (2) réseau (3) abonné (4) abrégé (6)
NumberPlan	octet	N	M	valeur par défaut: 255 – la valeur par défaut est "plan inconnu" plan inconnu (0) plan RNIS (1) plan pour données (3) plan pour télex (4) plan national (8) plan privé (9)
Number	chaîne IA5	N	M	longueur maximale: 20 octets

7.6.17 ConnectedSubaddress

Description: ce paramètre est utilisé pour transmettre au PUF la sous-adresse du numéro connecté.

Type: 22.

Champ	Type de champ	Sens	Requis	Commentaire
NumberType	octet	N	M	point nsap (0) utilisateur (1)
Indicator	octet	N	M	pair (0) impair (1) ce champ n'est significatif que si le type de numéro est mis à "utilisateur". Il indique si le numéro contient un nombre impair ou pair de chiffres.
Number	chaîne IA5	N	M	20 octets est la longueur maximale

7.6.18 CPMessageMask

Description: ce paramètre est réglé par le dispositif PUF pour indiquer quels sont les messages du plan de commande qu'il est censé ne pas recevoir du NAF. Tous les messages du plan de commande ne peuvent pas être ainsi filtrés.

Ce paramètre est codé sous forme de champ d'éléments binaires. Sa valeur par défaut est 0 à toutes les positions binaires, ce qui signifie que le PUF recevra tous les messages émis par le réseau.

Type: 73.

Champ	Type de champ	Sens	Requis	Commentaire
CPMessageMask	chaîne d'octets	P	M	longueur fixe: 2 octets bit 1 CAlertInd bit 2 CProgressInd bit 3 CSetupAckInd bit 4 CProceedingInd bit 5 CUserInformationInd bit 6 CCongestionControlInd bit 7 CNotifyInd bit 8 CFacilityInd bits 9 à 16: réservés

7.6.19 CPParameterMask

Description: ce paramètre est réglé par le dispositif PUF pour indiquer quels sont les paramètres contenus dans des messages du plan de commande qu'il est censé ne pas recevoir du NAF. Tous les paramètres de messages du plan de commande ne peuvent pas être ainsi filtrés.

Ce paramètre est codé sous forme de champ d'éléments binaires. Sa valeur par défaut est 0 à toutes les positions binaires, ce qui signifie que le PUF recevra tous les paramètres de messages émis par le réseau dans le plan de commande.

Type: 72.

Champ	Type de champ	Sens	Requis	Commentaire
CPParameterMask	chaîne d'octets	P	M	longueur fixe: 4 octets bit 1 CauseToPUF bit 2 ChannelIdentification bit 3 DateTime bit 4 Display bit 5 Facility bit 6 High Layer Compatibility bit 7 Low Layer Compatibility bit 8 UserToUserInfo bit 9 Signal bit 10 ProgressIndicator bits 11 à 31: réservés

7.6.20 DateTime

Description: ce paramètre est utilisé pour transmettre au dispositif PUF des informations relatives à la date et à l'heure. Ces informations sont fournies par le réseau lors de l'opération d'établissement d'appel ou par le dispositif NAF lors de la création d'objet NCO.

Type: 23.

Champ	Type de champ	Sens	Requis	Commentaire
Year	octet	N	M	0 à 99
Month	octet	N	M	1 à 12
Day	octet	N	M	1 à 31
Hour	octet	N	M	0 à 23
Minute	octet	N	M	0 à 59

7.6.21 Display

Description: ce paramètre sert à transmettre au dispositif PUF des informations d'affichage.

Type: 24.

Champ	Type de champ	Sens	Requis	Commentaire
Information	chaîne IA5	N	M	32 octets est la longueur maximale

7.6.22 ExtEquipAvailability

Description: ce paramètre sert à transmettre les informations relatives à la disponibilité de l'équipement externe.

Type: 25.

Champ	Type de champ	Sens	Requis	Commentaire
Availability	Booléen	N	M	état de l'équipement externe: TRUE – équipement disponible FALSE – équipement indisponible

7.6.23 ExtEquipBlockDialling

Description: ce paramètre sert à transmettre les informations relatives à la numérotation en bloc effectuée au moyen du clavier de l'équipement externe.

Type: 26.

Champ	Type de champ	Sens	Requis	Commentaire
BlockDialling	chaîne IA5	N	M	adresse distante et/ou sous-adresse distante, saisie sur le clavier de l'équipement externe un astérisque (*) sépare les champs d'adresse et de sous-adresse 41 octets est la longueur maximale

7.6.24 ExtEquipKeypressed

Description: ce paramètre sert à transmettre les informations relatives aux touches manipulées sur le clavier de l'équipement externe.

Type: 27.

Champ	Type de champ	Sens	Requis	Commentaire
Keypressed	octet	N	M	informations issues du clavier: (0 à 9) touches numériques (10) touche "*" (11) touche "#" (12) touche "A" (13) touche "B" (14) touche "C" (15) touche "D"

7.6.25 ExtEquipName

Description: ce paramètre sert à transmettre le nom qui identifie un élément d'équipement externe.

Type: 28.

Champ	Type de champ	Sens	Requis	Commentaire
Type	octet	B	M	type1 (1) – équipement externe de type 1 type2 (2) – équipement externe de type 2 type3 (3) – équipement externe de type 3 type4 (4) – équipement externe de type 4 type5 (5) – équipement externe de type 5 les types d'équipement externe sont décrits dans l'Annexe A
Name	chaîne IA5	B	M	longueur maximale: 16 octets "DEFAULT" – utiliser l'équipement externe du type spécifié en premier

7.6.26 Facility

Description: ce paramètre sert à transmettre à destination/en provenance du dispositif PUF des informations relatives aux services complémentaires. Si l'on attend des informations sur un autre service complémentaire que celui qui est défini par les valeurs 1 à 4 du champ FacilityTag, il y a lieu d'utiliser la valeur "codage transparent" (5).

Type: 30.

Tableau 9 – Codage du champ FacilityValue si le champ FacilityTag a la valeur "erreur de taxation"

Sous-champ	Type de champ	Valeur	Commentaire
ChargingError-Cause	octet	non abonné (50)	l'utilisateur n'est pas abonné au service complémentaire d'information de taxation (AOC, *advice of charge*)
		indisponible (63)	le service complémentaire d'information de taxation n'est pas disponible
		non implémenté (69)	le service complémentaire d'information de taxation n'est pas implémenté
		état d'appel non valide (101)	le service complémentaire d'information de taxation est invoqué dans un état d'appel non valide alors qu'il ne peut l'être que par le message CConnectReq
		aucune information de taxation disponible (128)	il n'y a pas d'information de taxation disponible

7.6.27 GroupID

Description: ce paramètre est utilisé pour transmettre l'identificateur de groupe à destination/en provenance du dispositif PUF.

Type: 33.

Champ	Type de champ	Sens	Requis	Commentaire
GroupID	chaîne d'octets	B	M	la valeur est unique pour une relation PUF/NAF 4 octets est la longueur fixe
NOTE – Les Parties 3 à 6 contiennent une description détaillée des utilisations possibles de ce paramètre.				

7.6.28 Compatibilité avec les couches supérieures (HLC, *high layer compatibility*)

Description: ce paramètre est utilisé pour transmettre à destination/en provenance du PUF des informations de compatibilité avec les couches supérieures (HLC).

Type: 34.

Champ	Type de champ	Sens	Requis	Commentaire
Standard	octet	B	M	valeur par défaut (255) – la valeur par défaut est "norme UIT-T" norme UIT-T (0) norme internationale (1) norme nationale (2) norme propre au réseau (3)
Identification	octet	B	M	téléphonie (1) fax G4C1 (33) télétex F184 (36) télétex F220 (40) télétex F200 (49) vidéotex (50) télex (53) messagerie X400 (56) osix200 (65) maintenance (94) gestion (95)
ExtIdentification	octet	B	O	téléphonie (1) fax G4C1 (33) télétex F184 (36) télétex F220 (40) télétex F200 (49) vidéotex (50) télex (53) messagerie X400 (56) osix200 (65)

7.6.29 Key

Description: touche à utiliser pour l'algorithme de sécurité.

Type: 36.

Champ	Type de champ	Sens	Requis	Commentaire
Key	chaîne d'octets	P	M	le paramètre Key est utilisé par le PUF pour donner au NAF des informations relatives à l'algorithme de sécurité longueur maximale: 255

7.6.30 Keypad

Description: ce paramètre est utilisé pour transmettre au NAF des informations sur le service complémentaire de clavier.

Type: 37.

Champ	Type de champ	Sens	Requis	Commentaire
Keypad	chaîne d'octets	P	M	caractères de l'alphabet AI5 à acheminer longueur maximale: 32

7.6.31 Compatibilité avec les couches inférieures (LLC, *low layer compatibility*)

Description: ce paramètre est utilisé pour transmettre à destination/en provenance du PUF un sous-ensemble des informations de compatibilité avec les couches inférieures (LLC). Les informations concernant les détails physiques de la couche 1 dovient être extraites du paramètre BearerCap lorsqu'un message CConnectReq est émis avec un paramètre LLC.

Type: 46.

Champ	Type de champ	Sens	Requis	Commentaire
Negotiation	Booléen	B	M	VRAI – négociation autorisée FAUX – négociation non autorisée
Layer2protocol	octet	B	M	0 à 31 255 = valeur non spécifiée ce champ se rapporte à l'octet 6 de l'élément d'information LLC
layer2optional	octet	B	M	0 à 127 255 = valeur non spécifiée ce champ se rapporte à l'octet 6a de l'élément d'information LLC
Layer3protocol	octet	B	M	0 à 31 255 = valeur non spécifiée ce champ se rapporte à l'octet 7 de l'élément d'information LLC
layer3optional	octet	B	M	0 à 127 255 = valeur non spécifiée ce champ se rapporte à l'octet 7a de l'élément d'information LLC

7.6.32 ManufacturerCode

Description: ce paramètre désigne le constructeur. Il est fourni par celui-ci.

Type: 47.

Champ	Type de champ	Sens	Requis	Commentaire
Value	chaîne d'octets	B	M	identification du constructeur longueur maximale: 255 octets

7.6.33 NCOID

Description: ce paramètre sert à transmettre à destination/en provenance du PUF l'identificateur d'objet NCO.

Type: 49.

Champ	Type de champ	Sens	Requis	Commentaire
Value	chaîne d'octets	B	M	cette valeur est unique pour une relation PUF/NAF la longueur est fixée à 4 octets

7.6.34 NCOType

Description: ce paramètre sert à transmettre au NAF le type d'objet NCO.

Type: 50.

Champ	Type de champ	Sens	Requis	Commentaire
Identifier	octet	B	M	ensemble du plan de commande (1) – accès sémaphore seulement (Note)
NOTE – On trouvera dans les Parties 3 à 6 d'autres valeurs du paramètre NCOType à utiliser s'il y a différents types d'accès par le plan d'utilisateur.				

7.6.35 NotificationIndicator

Description: ce paramètre sert à transmettre au PUF des notifications d'événement réseau. C'est une opération qui peut être suspendue ou reprise.

Type: 51.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	N	M	opération suspendue (1) opération reprise (2) signal d'appel (3)
NOTE – D'autres valeurs dépendent du réseau.				

7.6.36 NumberComplete

Description: ce paramètre est utilisé par le PUF pour indiquer au NAF qu'un numéro appelé est complet.

Type: 79.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	P	M	numéro complet (1)
NOTE – Une seule valeur est disponible.				

7.6.37 ProgressIndicator

Description: ce paramètre sert à transmettre des informations concernant la progression d'un appel téléphonique vers le dispositif PUF.

Type: 53.

Champ	Type de champ	Sens	Requis	Commentaire
Standard	octet	N	M	norme UIT-T (0) norme internationale (1) norme nationale (2) norme propre au réseau (3)
Location	octet	N	M	lieu d'utilisateur (0) lieu privé (1) lieu public (2) centre de transit (3) lieu public distant (4) lieu privé distant (5) centre international (7) lieu hors réseau (10)
Value	octet	N	M	non RNIS (1) – l'appel n'est pas RNIS de bout en bout; d'autres informations peuvent être disponibles dans la bande. destination non RNIS (2) – l'adresse de destination n'est pas RNIS. origine non RNIS (3) – l'adresse d'origine n'est pas RNIS. appel revenu au RNIS (4) – l'appel est revenu au RNIS. information dans la bande (8) – des informations dans la bande ou une structure appropriée sont maintenant disponibles.

7.6.38 RequestID

Description: ce paramètre sert à demander au NAF l'envoi d'un identificateur concernant un message de requête. Ce paramètre est retourné par le NAF au sujet du message de confirmation associé.

Type: 56.

Champ	Type de champ	Sens	Requis	Commentaire
Identifier	chaîne d'octets	B	M	identificateur interne fourni par le PUF longueur fixée à 4 octets

7.6.39 SelectorID

Description: ce paramètre est utilisé par le PUF pour sélectionner l'objet NCO correct lors d'un appel entrant (deuxième étape de la sélection). Le dispositif PUF utilise également ce paramètre pour donner au dispositif NAF une liste d'objets NCO qu'il y a lieu de traiter en exclusivité.

Type: 60.

Champ	Type de champ	Sens	Requis	Commentaire
Identifier	chaîne d'octets	P	M	identificateur interne fourni par le PUF longueur fixée à 4 octets

7.6.40 Signal

Description: ce paramètre est utilisé, sur option, pour acheminer vers le PUF des informations relatives aux tonalités et aux signaux d'alerte.

Type: 81.

Champ	Type de champ	Sens	Requis	Commentaire
Signal	octet	N	M	tonalité d'invitation à numéroter (0) tonalité de retour d'appel (1) tonalité d'interception (2) tonalité d'encombrement (3) tonalité d'occupation (4) tonalité de confirmation (5) tonalité de réponse (6) signal d'appel (7) signal de terminal décroché (8) coupure des tonalités (63) sonnerie – séquence 0 (64) (Note) sonnerie – séquence 1 (65) (Note) sonnerie – séquence 2 (66) (Note) sonnerie – séquence 3 (67) (Note) sonnerie – séquence 4 (68) (Note) sonnerie – séquence 5 (69) (Note) sonnerie – séquence 6 (70) (Note) sonnerie – séquence 7 (71) (Note) coupure des sonneries (79)
NOTE – L'utilisation de ces séquences dépend du réseau.				

7.6.41 TEI

Description: ce paramètre sert à faire accéder une liaison permanente à un commutateur de données en mode paquet (connexion de paquets dans le canal D).

Type: 61.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	B	M	

7.6.42 UProtocol

Description: ce paramètre est utilisé pour sélectionner le protocole de plan d'utilisateur.

Type: 62.

Champ	Type de champ	Sens	Requis	Commentaire
L3Protocol	octet	P	M	valeur par défaut (255) – (Note)
L2Protocol	octet	P	C	obligatoire si le champ L3Protocol a la valeur NULL (4). valeur par défaut (255) – ISO 7776 (Note)
L1Protocol	octet	P	C	obligatoire si le champ L2Protocol a la valeur NULL (8). Absent si le champ L2Protocol est absent. (Note)
NOTE – Les Parties 3 à 6 décrivent en détail l'utilisation et les valeurs possibles pour ce paramètre.				

7.6.43 UAttributeName

Description: ce paramètre est utilisé par le dispositif PUF pour envoyer le nom d'un ensemble statique d'attributs du plan d'utilisateur.

Type: 63.

Champ	Type de champ	Sens	Requis	Commentaire
AttributeName	chaîne IA5	P	M	16 octets est la longueur maximale

7.6.44 UDirection

Description: ce paramètre est utilisé pour envoyer au dispositif NAF des informations concernant l'usage d'un objet NCO particulier, pour le plan d'utilisateur.

Type: 64.

Champ	Type de champ	Sens	Requis	Commentaire
Direction	octet	P	M	écoute (1) appel (2) dans les deux sens (3)
NOTE – Si ce paramètre est absent, le dispositif NAF prend comme valeur celle du paramètre CDirection.				

7.6.45 UserToUserInfo

Description: ce paramètre est utilisé pour transmettre, à destination/en provenance du dispositif PUF, des informations d'utilisateur.

Type: 65.

Champ	Type de champ	Sens	Requis	Commentaire
Discriminator	octet	B	M	format propre à l'utilisateur (0) – le contenu du champ Information est dans un format propre à l'utilisateur. caractères IA5 (4) – le contenu est une chaîne de caractères IA5.
Information	chaîne d'octets	B	M	longueur maximale: 128

Remarques: le champ Discriminator sert à indiquer le format des données contenues dans le champ Information. Des valeurs comprises entre 0 et 256 sont possibles mais peuvent être limitées par le RNIS auquel on accède. Les valeurs définies sont compatibles avec tous les dispositifs NAF.

7.6.46 Paramètres d'ensemble d'attributs (AttributeSet)

Les paramètres d'ensemble d'attributs dépendent du type des paramètres insérés dans la requête ACreateNCO. Les Tableaux 10 et 11 montrent le contenu de ces paramètres.

Tableau 10 – Paramètres de l'ensemble d'attributs de signalisation (CAttributeSet)

Paramètre	Requis	Commentaire
ChannelIdentification	O*	voir 7.6.12
HLC	O	voir 7.6.28
LLC	O	voir 7.6.31
BearerCap	O	voir 7.6.3

Tableau 11 – Paramètres relatifs à l'équipement externe (compris dans l'ensemble d'attributs du plan U)

Paramètre	Requis	Commentaire
ExtEquipName	O	nom de l'équipement externe à utiliser. S'il est fourni, la connexion doit être établie avec l'équipement externe ainsi désigné et aucun message ne doit passer par le plan d'utilisateur. Voir 7.6.25.

Les Parties 3 à 6 contiennent des tableaux décrivant plus en détail les paramètres de l'ensemble d'attributs du plan d'utilisateur (UAttributeSet) que l'on peut utiliser avec la requête ACreateNCO pour les divers protocoles possibles dans ce plan.

7.6.47 Paramètres d'ensemble d'attributs dans le plan d'administration

Les paramètres de l'ensemble d'attributs du plan d'administration servent à collecter certaines informations de gestion au sujet de chaque objet NCO; ces paramètres sont accessibles à tout moment, par l'opération GetNCOInfo. Le Tableau 12 montre le contenu de ce paramètre.

Tableau 12 – Administration Attribute Set (paramètres)

Paramètre	Requis	Commentaire
NCOType	O	voir 7.6.34
CDirection	O	voir 7.6.11
CAttributeName	O	voir 7.6.8
CAttribute (paramètres)	O	voir le Tableau 10
UDirection	O	voir 7.6.44
UAttributeName	O	voir la partie applicable de la spécification du protocole du plan d'utilisateur du PCI-RNIS
UAttribute (paramètres)	O	voir la partie applicable de la spécification du protocole du plan d'utilisateur du PCI-RNIS
CAddress (paramètres)	O	voir le Tableau 13
UAddress (paramètres)	O	voir la partie applicable de la spécification du protocole du plan d'utilisateur du PCI-RNIS
GroupID	O	fourni au moment de la création d'un objet NCO. Voir 7.6.27.
SelectorID	O	voir 7.6.39
ChargingInfo	O	voir 7.6.13
DateTime	O	date et heure de la création d'objet NCO. Voir 7.6.20.
CauseToPUF	O	voir 7.6.10

7.6.48 Paramètres d'ensemble d'adresses (Address set parameter)

Le Tableau 13 montre la structure des paramètres d'un ensemble d'adresses.

Tableau 13 – Paramètres de l'ensemble d'adresses sémaphores (CAddressSet)

Paramètre	Requis	Commentaire
CalledNumber	O	voir 7.6.4 pour la définition de ce paramètre
CalledSubaddress	O	voir 7.6.5 pour la définition de ce paramètre
CallingNumber	O	voir 7.6.6 pour la définition de ce paramètre
CallingSubaddress	O	voir 7.6.7 pour la définition de ce paramètre

Les Parties 3 à 6 contiennent des tableaux décrivant plus en détail les paramètres de l'ensemble d'adresses du plan d'utilisateur (UAddressSet) que le dispositif PUF peut utiliser pour les divers protocoles d'utilisateur possibles.

7.7 Critères de sélection

7.7.1 Sélection d'objet NCO

Afin d'appliquer l'objet NCO approprié à un appel entrant, les conditions suivantes doivent être observées par le dispositif NAF.

Seuls les objets NCO dont les paramètres UDirection et CDirection sont mis à la valeur "appels entrants" ou "dans les deux sens" sont pris en compte pour un appel entrant. Au mieux, l'objet NCO contiendra une définition explicite pour chaque valeur utilisée lors de la vérification. Le niveau d'analogie doit être réglé sur les valeurs vérifiées plutôt que sur les valeurs estimées. Le Tableau 14 résume l'opération de mise en correspondance.

Tableau 14 – Opération de mise en correspondance pour la sélection des objets NCO

Réseau	Objet NCO	Opération	Résultat
fourni	fourni	égal	correspondance
fourni	fourni	non égal	pas de correspondance
fourni	non fourni	(pas d'opération)	correspondance
non fourni	fourni	(pas d'opération)	pas de correspondance
non fourni	non fourni	(pas d'opération)	correspondance

Le dispositif NAF doit diffuser un appel entrant à tous les dispositifs PUF qui ont indiqué leur compatibilité avec un objet NCO. Cet appel entrant doit ensuite être assigné définitivement au dispositif PUF qui accepte le premier l'appel avec le message approprié. Tous les autres dispositifs PUF doivent recevoir une indication de déconnexion. L'emploi de cette procédure implique aussi que les objets NCO coordonnés par dispositif NAF ont une priorité plus élevée que les objets NCO coordonnés par dispositif PUF, car le NAF peut répondre immédiatement à un appel entrant sans impliquer un quelconque dispositif PUF. Dans un tel cas, l'appel n'est pas perçu par les objets NCO non coordonnés.

Si des messages CAlertReq sont envoyés par un dispositif PUF, seul le premier doit être envoyé au réseau. Tous les autres sont ignorés. Si un message CDisconnectReq est envoyé par un dispositif PUF, ce message ne déconnecte pas l'appel, sauf si aucun autre objet NCO n'a été assigné à cet appel. Ce mécanisme donne la possibilité d'établir une connexion avec un objet NCO temporisé.

Le paramètre SelectorID a une incidence sur l'opération de diffusion des appels entrants lorsque l'on sélectionne plusieurs objets NCO par dispositif PUF.

7.7.1.1 Eléments d'information du plan de commande

1) adresse de l'appelé (adresse correcte ou absente);
2) sous-adresse de l'appelé (sous-adresse correcte ou absente);
3) capacités supports (profil correct);
4) compatibilité LLC (voir Note) (correcte ou absente);
5) compatibilité HLC (correcte ou absente).

Ces cinq éléments d'information doivent correspondre aux valeurs d'un objet NCO pour que celui-ci puisse être sélectionné. A la fin du processus de sélection, si plusieurs objets NCO sont possibles, la sélection du deuxième degré doit s'appliquer. Pour choisir un objet NCO, on applique d'abord la fonction de vérification, associée à l'ordre des éléments d'information. Le dernier critère de sélection est le temps. Le dernier objet NCO créé par le dispositif NAF doit être choisi en premier.

Exemple: dans le cas présenté au Tableau 15 ci-dessous, seul l'objet NCO2 doit être choisi car l'objet NCO1 est en attente d'un élément d'information de sous-adresse différent de celui qui lui a été fourni dans l'appel entrant et parce que l'élément d'information attendu, contenu dans l'objet NCO2, correspond aux éléments d'information de l'appel entrant.

Tableau 15 – Correspondance d'un objet NCO avec un appel entrant

Champ	Appel entrant	NCO1	NCO2
adresse de l'appelé	123456789	non fourni	123456789
sous-adresse de l'appelé	1002	1001	non fourni
capacités supports	signaux vocaux	non fourni	signaux vocaux
compatibilité LLC	pas de négociation hors bande	non fourni	non fourni
compatibilité HLC	téléphonie	téléphonie	téléphonie

Les Parties 3 à 6 spécifient, pour les divers protocoles possibles dans le plan d'utilisateur, les éléments d'information propres à chacun des protocoles.

7.7.2 Suite à donner en cas d'indisponibilité d'objet NCO

7.7.2.1 Appel entrant dans le plan de commande

Le dispositif NAF émet une cause de déconnexion #88 "destination incompatible".

7.7.2.2 Appel entrant dans le plan d'utilisateur

Les Parties 3 à 6 spécifient la procédure de déconnexion pour les divers protocoles d'utilisateur possibles.

7.8 Vérification et codes d'erreur

Le présent sous-paragraphe traite de la vérification des erreurs assurée par l'interface PCI-RNIS. On décrira d'abord les méthodes de vérification d'erreur employées par chaque plan. Puis on définira et on décrira, pour chaque plan, les codes de retour relatifs aux fonctions et les codes de retour relatifs aux erreurs.

7.8.1 Plan d'administration

Presque tous les messages du plan d'administration sont émis par paires de requête/confirmation; il n'y a pas de messages d'indication/réponse. Toute erreur détectée dans un message de requête doit être notifiée dans le message de confirmation correspondant.

Dans les messages du plan d'administration, toute erreur détectée doit empêcher l'exécution d'une opération et donc un changement d'état.

Dans le plan d'administration, le message AErrorInd sert à indiquer les erreurs non couvertes par les protocoles qui gèrent les messages du plan de commande et du plan d'utilisateur. Par exemple, ce message servira à informer le dispositif PUF qu'un identificateur NCOID non valide a été spécifié pour un message.

7.8.2 Plan de commande

Si des paramètres à caractère obligatoire (M) font défaut, ou si une erreur de contenu se produit dans un tel paramètre, ou si un paramètre n'est pas reconnu, le dispositif NAF indique cette erreur au dispositif PUF comme indiqué aux 7.8.2.1 à 7.8.2.3.

7.8.2.1 Etat d'invalidité pour les messages

Cet état est indiqué par le message CStatusInd, qui ne provoque aucun changement d'état pour la connexion.

7.8.2.2 Paramètres obligatoires

En cas d'absence de paramètres obligatoires, d'erreur de contenu dans un paramètre obligatoire ou de non-reconnaissance d'un paramètre, le dispositif NAF indique l'erreur au dispositif PUF par un des messages suivants:

- CDisconnectInd pour CConnectReq;
- CDisconnectCnf pour CDisconnectReq;
- CStatusInd pour tout autre message, et aucune opération ou transition d'état n'est effectuée.

7.8.2.3 Erreur de contenu pour un paramètre facultatif

Dans ce cas, le message doit être traité comme si le paramètre facultatif n'était pas présent et le message CStatusInd est envoyé au dispositif PUF pour indiquer le paramètre erroné.

7.8.3 Erreurs dans les demandes de service

Les erreurs relatives à des demandes de service dépendent de la nature du service demandé. Dans le cas du service complémentaire d'information de taxation, les erreurs sont indiquées au moyen d'un message CFacilityInd. Le message qui a produit cette erreur est traité comme si aucun élément d'information relatif aux services n'était présent. Les erreurs spécifiques sont définies dans le Tableau 9.

Lorsqu'un dispositif PUF utilise des services sous la forme transparente, il appartient à ce dispositif de déterminer la manière dont les erreurs seront signalées, et le traitement pouvant être assuré par le réseau.

7.8.4 Plan d'utilisateur

Les erreurs sont traitées conformément aux procédures définies dans les Parties 3 à 6 pour les divers protocoles possibles dans le plan d'utilisateur.

7.8.5 Codes de retour relatifs aux fonctions

Le Tableau 16 définit les codes de retour relatifs aux fonctions.

Tableau 16 – Codes de retour relatifs aux fonctions

Code de retour		Signification
Success	0	fonction correctement exécutée
QueryEntityNotAvailable	128	l'entité interrogée n'est pas disponible ou une erreur s'est produite au cours du dialogue entre le PUF et l'entité interrogée
InvalidSignalNumber	129	le numéro de signalisation spécifié est non valide
InvalidPCIHandle	130	non-identification d'un NAF par le pointeur
NAFnotAvailable	255	le NAF n'est plus disponible et s'est déconnecté à cause d'une erreur. Il s'agit d'un état fixe.
NAFBusy	132	le NAF n'est pas en mesure, pour le moment, de traiter la requête (manque de ressources ou autre motif). La fonction peut ne pas donner un résultat correct si elle est relancée ultérieurement. Il s'agit d'un état temporaire.
MaxPUFsExceeded	133	le NAF ne peut pas gérer un plus grand nombre de dispositifs PUF
InvalidPUFType	134	type de PUF non valide ou incompatible. Le NAF ne gère pas ce type de PUF.
InvalidPCIVersion	135	version non valide ou incompatible d'interface PCI. Le NAF ne gère pas cette version de l'interface PCI.
InvalidExID	136	le NAF ne reconnaît pas l'identificateur de commutateur
InvalidPCIMPB	137	l'adresse du bloc de paramètres de message PCI est incorrecte
InvalidMessageBuffer	138	l'adresse de la mémoire tampon de messages est non valide
InvalidDataBuffer	139	l'adresse de la mémoire tampon de données est non valide
PCIMPBBufferTooSmall	140	la mémoire tampon de blocs PCIMPB est insuffisante ce code est prévu pour les systèmes d'exploitation qui peuvent contrôler la capacité de la mémoire disponible
MessageBufferTooSmall	141	mémoire tampon de messages insuffisante cette mémoire tampon ne répond pas aux prescriptions relatives aux identificateurs de message ou la capacité réelle du tampon des blocs PCIMPB dépasse sa limite supérieure. Dans certains systèmes d'exploitation, ce code peut également indiquer que la capacité maximale du tampon de données dépasse sa limite supérieure.
DataBufferRequired	142	un tampon de données est requis pour ce message
DataBufferTooSmall	143	mémoire tampon de données insuffisante cette mémoire tampon ne répond pas aux prescriptions relatives aux identificateurs de message ou la capacité réelle du tampon des blocs PCIMPB dépasse sa limite supérieure. Dans certains systèmes d'exploitation, ce code peut également indiquer que la capacité maximale du tampon de messages dépasse sa limite supérieure.
PropertyBufferTooSmall	144	la mémoire fournie avec la (les) structure(s) des informations sur la propriété est insuffisante
MessageTooLarge	145	il n'y a pas de limite supérieure quant à la longueur des messages étant donné que les paramètres peuvent être répétés. Si la longueur d'un message dépasse la valeur maximale admissible pour une implémentation, cette valeur est retournée.
InvalidHandlesBuffer	146	l'adresse du tampon de pointeurs PCI est non valide
HandlesBufferTooSmall	147	la capacité du tampon de pointeurs PCI est insuffisante pour contenir tous les pointeurs PCI disponibles
BufferTooSmall	148	la capacité du tampon offert par le PUF est insuffisante pour répondre aux besoins du NAF (code de retour propre au système d'exploitation)
InvalidRegisterInfoStructure	149	au moins un paramètre contenu dans la structure PCIRegisterInfo est non valide (code de retour propre au système d'exploitation)
InvalidOpSysInfoStructure	150	au moins un paramètre contenu dans la structure PCIOpSysInfo est non valide (code de retour propre au système d'exploitation)

7.8.6 Codes de retour relatifs au plan d'administration

Les valeurs suivantes (voir le Tableau 17) sont retournées dans le paramètre CompletionStatus. La colonne intitulée "Information pour le champ ErrorSpecific" indique quelles informations doivent, le cas échéant, être insérées dans le champ ErrorSpecific du paramètre CompletionStatus.

Tableau 17 – Codes de retour relatifs au plan d'administration

Code de retour		Signification	Information spécifique du champ ErrorSpecific
Success	0	opération exécutée correctement	non présente
NAFnotAvailable	255	le NAF n'est plus disponible. Il a été déconnecté en raison d'une erreur. Il s'agit d'un état permanent.	non présente
RessourceNotAvailable	47	code utilisé avec le message de demande de création d'objet NCO pour indiquer qu'une ressource fait défaut (par exemple de la mémoire)	non présente
UndefinedMsgType	95	cet identificateur de message n'est pas défini par l'interface PCI-RNIS	identificateur de message
UnsupportedMsgType	97	cet identificateur de message est défini par l'interface PCI-RNIS mais n'est pas compatible avec ce NAF	identificateur de message
InvalidParameter	99	paramètre non reconnu ou non compatible avec un message	type de paramètre
MissingParameter	96	un paramètre obligatoire fait défaut dans un message	type de paramètre
InvalidParameterLength	182	la longueur d'un paramètre dépasse la limite supérieure autorisée	type de paramètre
InvalidContents	100	un élément du contenu d'un paramètre est non valide. Code utilisé avec le message de confirmation de création d'objet NCO pour signaler des erreurs dans des paramètres servant à définir un objet NCO.	type de paramètre
InvalidNCOID	81	un message a été transmis au NAF avec un identificateur NCOID non valide	valeur du NCOID
NCOIDinUse	183	un identificateur NCOID, déjà utilisé pour une connexion établie/en cours d'établissement, ne peut pas être utilisé dans ce message.	valeur du NCOID
InvalidNCOType	184	un message a été transmis au NAF avec une valeur non valide de type d'objet NCO	valeur de type d'objet NCO
InvalidDirectionType	185	un message a été transmis au NAF avec une valeur de sens non valide	non présente
AttributeNameError	186	utilisation non valide d'un nom d'attribut. Nom inconnu, déjà défini ou ensemble d'attributs de type erroné.	nom d'attribut
ExtraSetError	189	message contenant un nom d'ensemble d'attributs non requis	nom d'attribut
SecurityNotActivated	190	l'algorithme de sécurité demandé n'a pas été activé	valeur spécifique de l'algorithme de sécurité
InvalidCoordValue	191	valeur non valide dans le paramètre NAFCoordination	non présente
InvalidGroupID	192	valeur d'identificateur de groupe non reconnue par le NAF	valeur d'identificateur de groupe
GroupIDError	193	message manquant ou nécessitant un identificateur de groupe	non présente
InvalidExtEquipName	194	nom d'équipement externe inconnu du NAF	non présente
InvalidExtEquipType	195	valeur non valide spécifiée pour le type d'équipement externe	non présente
OperationFailed	196	échec de l'opération demandée	non présente
ManufacturerCodeError	197	erreur dans le code du constructeur	information additionnelle propre au constructeur
FunctionalityNotProvided	198	propriété non fournie par le NAF	non présente

7.8.7 Causes dans le plan de commande

Ces valeurs sont retournées dans le champ "Cause" du paramètre CauseToPUF lorsque celui-ci fait partie d'un message transmis de NAF à PUF.

NOTE – L'abréviation N/A signifie "non applicable".

Tableau 18 – Causes dans le plan de commande

Valeur	Signification selon Q.931 [1]	Signification selon PCI-RNIS	Signal produit par	Diagnostics fournis par le NAF
1	numéro non attribué		RNIS	N/A
2	pas de route vers réseau de transit spécifié		RNIS	N/A
3	pas de route vers destination		RNIS	N/A
6	canal inacceptable		RNIS	N/A
7	appel lancé sur canal déjà établi		RNIS	N/A
16	libération normale d'appel		RNIS	N/A
17	utilisateur occupé		RNIS	N/A
18	pas de réponse de l'utilisateur		RNIS	N/A
19	pas de réponse de l'utilisateur (qui a été alerté)		RNIS	N/A
21	appel rejeté		RNIS	N/A
22	adresse modifiée		RNIS	N/A
26	libération sur non-sélection d'utilisateur		RNIS	N/A
27	destination en dérangement		RNIS	N/A
28	format d'adresse non valide	format d'adresse non valide dans le paramètre	NAF, RNIS	valeur non présente
29	fonctionnalité rejetée	fonctionnalité non fournie par ce NAF	NAF, RNIS	valeur non présente
30	réponse à une demande de statut		RNIS	N/A
31	état normal non spécifié		RNIS	N/A
34	pas de circuit/canal disponible	aucun canal du type demandé n'est disponible pour le moment à partir de ce dispositif NAF	NAF, RNIS	valeur non présente
41	panne temporaire		RNIS	N/A
42	encombrement de l'équipement commutateur		RNIS	N/A
43	rejet d'information d'accès	rejet d'information(s) paramétrique(s)	NAF, RNIS	types de paramètres
44	canal/circuit demandé indisponible	aucun canal du type demandé n'est disponible à partir de ce NAF	NAF, RNIS	valeur non présente
47	ressource indisponible, non spécifiée	équipement externe demandé indisponible	NAF, RNIS	valeur non présente
49	qualité de service indisponible		RNIS	N/A
50	la fonctionnalité demandée par le paramètre Facility n'est pas comprise dans l'abonnement		RNIS	N/A
57	capacité support non autorisée		RNIS	N/A
58	capacité support présentement indisponible		RNIS	N/A

Tableau 18 – Causes dans le plan de commande *(fin)*

Valeur	Signification selon Q.931 [1]	Signification selon PCI-RNIS	Signal produit par:	Diagnostics fournis par le NAF
63	service ou option indisponible, non spécifié		RNIS	N/A
65	non-implémentation du service demandé par capacité support		RNIS	N/A
66	non-fourniture du type de canal	le NAF n'est pas compatible avec ce type de canal	NAF, RNIS	valeur non présente
69	fonctionnalité demandée non implémentée	le NAF n'est pas compatible avec cette fonctionnalité	NAF, RNIS	valeur non présente
79	non-fourniture du service ou de l'option, non spécifié		RNIS	N/A
81	référence d'appel non valide	identificateur NCOID non valide	NAF	valeur non présente
82	le canal identifié n'existe pas	le canal permanent identifié n'est pas défini	NAF	valeur non présente
85	aucune suspension d'appel	l'identificateur NCOID n'identifie pas une connexion en suspens	NAF	valeur non présente
88	destination incompatible		RNIS	N/A
96	absence d'un paramètre obligatoire	absence d'un paramètre obligatoire	NAF	type de paramètre
97	identificateur de message inexistant ou non implémenté sur ce réseau	identificateur de message inexistant ou non implémenté sur ce dispositif NAF	NAF	identificateur de message
98	message incompatible avec état d'appel ou identificateur de message inexistant ou non implémenté	message non compatible avec l'état de l'objet NCO ou identificateur de message inexistant ou non implémenté	NAF	identificateur de message
99	paramètre non valide	paramètre non valide	NAF	type de paramètre
100	non-validité du contenu du paramètre	non-validité du contenu du paramètre	NAF	type de paramètre
101	message incompatible avec état actuel	message incompatible avec état actuel	NAF	identificateur de message
127	interfonctionnement, non spécifié		RNIS	N/A

Ces valeurs sont valides dans le champ Cause du paramètre CauseToNAF lorsque ce paramètre est inséré dans un message transmis de PUF à NAF. Si une valeur non valide est utilisée, elle doit être ignorée et remplacée par une valeur de cause n° 16: libération normale d'appel.

Pour certains RNIS, d'autres valeurs peuvent être fournies dans le sens de NAF à PUF.

Tableau 19 – Contenu du paramètre CauseToNAF

Valeur	Signification
16	libération normale d'appel
21	rejet d'appel
31	appel normal, non spécifié
88	destination incompatible

7.8.8 Causes dans le plan d'utilisateur

Les Parties 3 à 6 indiquent les valeurs retournées comme causes d'achèvement dans les messages du plan d'utilisateur.

8 Méthode d'échange

Le présent paragraphe décrit la méthode d'échange et les fonctions d'échange qui sont utilisées pour réaliser l'échange local d'informations entre un dispositif PUF et un dispositif NAF. Etant donné que l'implémentation des fonctions d'échange dépend du système d'exploitation, ces fonctions sont décrites de manière globale. La méthode d'échange pour un environnement réel est contenue dans les Parties 7 à 9, selon divers systèmes d'exploitation, avec la représentation binaire, certains modules de code en langage de programmation C ainsi que toutes les informations associées à chacun de ces systèmes d'exploitation.

Etant donné que l'implémentation des fonctions d'échange, vues du côté NAF, dépend du système d'exploitation sous-jacent, le code du dispositif PUF pour l'appel de ces fonctions dépendra également du système d'exploitation. Afin d'assurer la portabilité du code source entre différents systèmes d'exploitation, il y aura lieu que le dispositif PUF encapsule le code d'appel du NAF dans une interface fonctionnelle émulant les fonctions génériques d'échange décrites dans le présent paragraphe.

Les fonctions d'échange passent et retournent des valeurs paramétriques qui sont fondées sur les types génériques indiqués dans le Tableau 20 ci-dessous.

Tableau 20 – Types génériques de méthode d'échange

Type générique	Explication
PCI_INTEGER	valeur d'entier signé, exprimée en binaire, couvrant au minimum l'étendue: $-2^{15}+1 \,..\, +2^{15}$.
PCI_BYTEARRAY	table de valeurs d'octet, exprimées en binaire, utilisée pour présenter des caractères. L'extension de signe pour cette valeur d'octet n'est pas définie. Aucune opération arithmétique ne doit être effectuée sur ce type.
PCI_EXID	type dépendant de l'implémentation, utilisé pour présenter l'identificateur d'échange par interface PCI.
PCI_HANDLE	type dépendant du système d'exploitation, utilisé pour présenter les informations relatives aux pointeurs d'interface PCI.
PCI_PROCEDURE	type dépendant du système d'exploitation, utilisé pour présenter des adresses de procédures.

Selon le système d'exploitation, les paramètres sont transmis soit par valeur soit par référence. La façon dont les paramètres sont transmis est définie dans la spécification applicable.

Conventions générales pour cette présentation générique:

- le nom de la fonction est précédé du préfixe "Pci"
- chaque fonction retourne un code d'achèvement. Toute autre valeur que "succès" (0) dans le code d'achèvement indique une erreur.

8.1 Phase d'enregistrement

8.1.1 Aperçu général

Avant qu'un dispositif PUF et un dispositif NAF puissent échanger des informations, le dispositif PUF doit établir une association avec le dispositif NAF. A cette fin, le PUF doit spécifier le pointeur d'interface PCI du NAF auquel il souhaite s'associer.

Pour supporter de nombreuses implémentations de NAF, pouvant provenir de différents constructeurs, on définit une méthode permettant au PUF de déterminer quels sont les NAF qui sont accessibles à partir de l'intérieur d'un système. A cette fin, la fonction facultative[1)] **PciGetHandles** permet au PUF d'obtenir une liste de pointeurs PCI accessibles. Par la suite, le dispositif PUF pourra extraire de cette liste un pointeur PCI. La présentation des pointeurs PCI est décrite dans la documentation propre au système d'exploitation.

1) L'utilisation de cette fonction est facultative pour le PUF mais son implémentation est obligatoire pour le NAF (par construction ou profilage).

Si elle est utilisée, la fonction **PciGetHandles** doit normalement être la toute première fonction lancée par un dispositif PUF car elle rend tous les pointeurs PCI disponibles. L'interfonctionnement avec d'autres fonctions d'échange est décrit à la Figure 10.

Une autre fonction facultative[1)], disponible dans la phase d'enregistrement, est la fonction **PciGetProperty** qui permet au PUF de s'informer des propriétés du NAF. Lors d'un appel, le dispositif PUF doit indiquer le pointeur PCI du NAF recherché. En réponse, le PUF obtient une liste des propriétés statiques du NAF.

Etant donné que les propriétés obtenues contiennent des informations sur des caractéristiques spéciales des NAF, le PUF peut utiliser ces informations pour sélectionner le(s) NAF avec le(s)quel(s) il souhaite établir une association enregistrée. Ces caractéristiques spéciales seront par exemple un combiné ou un dispositif de sécurité.

La seule fonction non facultative de la phase d'enregistrement d'association est la fonction **PciRegister**, qui permet au PUF de s'associer au NAF. Le dispositif PUF doit indiquer le pointeur d'interface PCI désignant le NAF avec lequel il souhaite s'associer. En conséquence, un identificateur deviendra disponible pour l'association entre PUF et NAF. Cet identificateur doit être indiqué lors d'appels de fonction d'échange ultérieurs, au cours des phases de conversation et de désenregistrement de cette association.

Les termes suivants sont utilisés pour la description de la phase d'enregistrement d'association:

Propriété-NAF: information structurée qui décrit les caractéristiques (propriétés) d'un dispositif NAF. Une propriété-NAF est implémentée indépendamment du système d'exploitation, au moyen du codage TLV (voir paragraphe 6). Elle doit donc être codée au moyen du même algorithme que pour le codage des messages. Dans un environnement à dispositifs NAF multiples, un PUF peut utiliser cette information pour sélectionner un NAF spécifique.

Pointeur-PCI: information d'accès à un dispositif NAF – Cette information doit être fournie aux fonctions de la phase d'enregistrement pour trouver un NAF et y accéder. L'implémentation du pointeur-PCI dépend du système d'exploitation. Par exemple, le pointeur-Pci peut être un nom, un chemin de fichier ou une adresse de fonction.

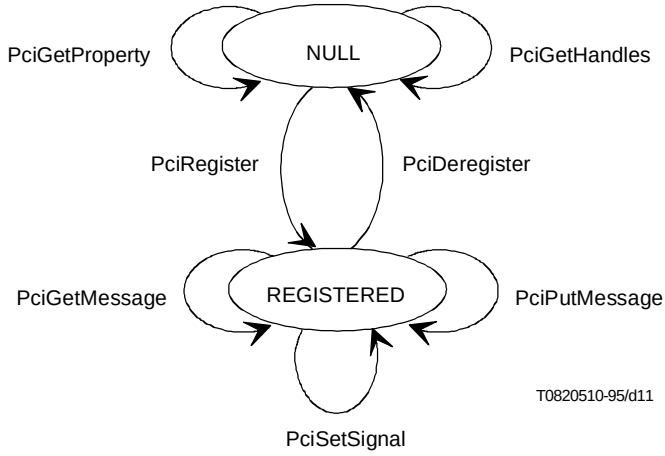


Figure 10 – Ordre des appels de fonction d'échange par interface PCI-RNIS, y compris les fonctions facultatives de la phase d'enregistrement d'association

NOTE – Dans un grand nombre d'environnements NAF, un PUF peut faire appel aux fonctions facultatives PciGetHandles et PciGetProperty, telles que décrites ci-dessus, pour sélectionner le dispositif NAF qui convient le mieux à ses besoins. Pour plus de détails sur la fonction **PciGetProperty**, voir 8.1.3.

1) L'utilisation de cette fonction est facultative pour le PUF mais son implémentation est obligatoire pour le NAF (par construction ou profilage).

8.1.2 PciGetHandles

Cette fonction permet à un dispositif PUF de demander combien de dispositifs NAF sont accessibles et d'obtenir leurs pointeurs-PCI. Au moyen de ces pointeurs, le PUF pourra ultérieurement obtenir la propriété-NAF ou enregistrer son association avec ce NAF.

Nom de la fonction: PciGetHandles

Valeur du retour de fonction: Errorcode (PCI_INTEGER)

Success
QueryEntityNotAvailable
InvalidHandlesBuffer
HandlesBufferTooSmall

Paramètres:

Nom	Type générique	Valeur d'appel ou de retour	Commentaire
MaxHandles	PCI_INTEGER	valeur d'appel	nombre maximal de pointeurs-PCI pouvant être reçus
PCIHandles	Table de PCI_HANDLE	valeur d'appel	tampon de capacité suffisante pour recevoir le nombre maximal (MaxHandles) de pointeurs-PCI
ActualHandles	PCI_INTEGER	valeur de retour	nombre réel de pointeurs-PCI retourné dans le tampon donné

Pour obtenir la liste des pointeurs-PCI disponibles, le PUF doit indiquer une mémoire tampon et sa capacité.

Le dispositif PUF reçoit, copié dans cette mémoire tampon, le nombre réel de pointeurs-PCI. Si ce nombre est supérieur à la capacité du tampon, celui-ci n'est pas rempli et l'erreur HandlesBufferTooSmall est retournée. Dans ce cas, le PUF doit indiquer un autre tampon, plus important, afin d'obtenir la liste complète des pointeurs-PCI.

8.1.3 PciGetProperty

Cette fonction permet à un dispositif PUF d'obtenir la propriété-NAF. Le PUF doit fournir, comme valeur d'appel, un pointeur-PCI. Un dispositif PUF peut obtenir un pointeur-PCI soit au moyen de la fonction facultative **PciGetHandles** ou par d'autres moyens (par exemple une indication locale).

Nom de la fonction: PciGetProperty

Valeur de retour de fonction: Errorcode (PCI_INTEGER)

Success
InvalidPCIHandle
NAFnotAvailable
NAFBusy
PropertyBufferTooSmall

Paramètres:

Nom	Type générique	Valeur d'appel ou de retour	Commentaire
PCIHandle	PCI_HANDLE	valeur d'appel	présentation et valeurs des pointeurs-PCI dépendant du système d'exploitation
MaximumSize	PCI_INTEGER	valeur d'appel	dimension maximale de la propriété permise en retour
NAFProperty	PCI_BYTEARRAY	valeur de retour	retour de propriété, celle-ci étant en codage TLV donc indépendante du système d'exploitation (Tableau 21).
ActualSize	PCI_INTEGER	valeur de retour	dimension réelle de la propriété retournée

Les paramètres de propriété NAF sont indiqués dans le Tableau 21.

Tableau 21 – Paramètres de propriété NAF à codage TLV

Paramètre	Fourni	Codage TLV (Note)			Commentaire et valeur
		ID de type	Longueur	Valeur	
Product	M	1	1..32	octet	chaîne d'octets indiquant le numéro de produit du NAF
Manufacturer	M	2	1..32	octet	chaîne d'octets indiquant le constructeur du dispositif NAF
AccessClass	M	3	1	octet	débit de base (1) ou débit primaire (2)
UserProtocolL3 a)	M	4	1..4	octet	indication des protocoles de couche 3 supportés. Ce paramètre peut servir de critère pour la sélection d'un NAF. Pour la valeur définie, voir 7.6.42.
UserProtocolL2 a)	M	5	1..2	octet	indication des protocoles de couche 2 supportés. Ce paramètre peut servir de critère pour la sélection d'un NAF. Pour la valeur définie, voir 7.6.42.
BChannels	M	6	1	octet	nombre de canaux B
BPermanent	O	7	1	octet	nombre de canaux B permanents
DPermanent	O	8	1	octet	nombre de canaux D permanents
AplaneClass a)	O	9	1	octet	fonctions additionnelles du plan d'administration supportées et identifiées par une classe. Valeurs valides: 2..4.
CPlaneClass a)	O	10	1	octet	fonctions additionnelles du plan de commande supportées et identifiées par une classe. Valeurs valides: 2..7.
SuppService a)	O	11	1..16	octet	spécification de services complémentaires. Pour complément d'étude.
ExtEquipName a)	O	12	2..17	octet	dans l'ordre, type et nom de l'équipement externe. Voir 7.6.25.
AdditionalUserProtocol a)	O	13	1..16	octet	protocoles additionnels du plan d'utilisateur. Pour complément ultérieur.
PCIVersion a)	O	15	1	octet	version d'interface PCI supportée

a) Ce paramètre peut être répété.

NOTE – D'autres spécifications de la série PCI-RNIS peuvent définir d'autres valeurs d'identificateur de type.

8.1.4 PciRegister

Cette fonction permet à un dispositif PUF de s'associer à un dispositif NAF.

Le PUF fournit, en tant que paramètre d'appel, le pointeur PCI du NAF avec lequel il souhaite effectuer un enregistrement d'association. De plus, deux structures sont transmises au sujet de la pile fonctionnelle:

- la structure PCIRegisterInfo;
- la structure PCIOpSysInfo.

La structure PCIRegisterInfo contient des paramètres spécifiques des dispositifs PUF et NAF, qui doivent être transmis entre les deux entités afin d'assurer leur coopération correcte. La structure PCIRegisterInfo est décrite dans le Tableau 22.

La structure PCIOpSysInfo contient des informations propres aux systèmes d'exploitation, qui doivent être échangées entre les dispositifs PUF et NAF.

Tableau 22 – Composition de la structure PCIRegisterInfo

Champ de la structure	Type générique	Valeur d'appel ou de retour	Commentaire
PUFVersion	PCI_INTEGER	valeur d'appel	version d'interface PCI-RNIS que le PUF souhaite utiliser. Valeur pouvant être mise à 0 dans tous les cas (par défaut).
PUFType	PCI_INTEGER	valeur d'appel	type de PUF. Ce paramètre servira à de futures extensions (par exemple pour permettre des types spécifiques de dispositifs PUF, comme des PUF de gestion de réseau). Actuellement, cette valeur doit être mise à zéro.
MaxMsgSize	PCI_INTEGER	valeur de retour	longueur maximale d'un message, que le NAF garantit de traiter: le NAF ne transmettra pas de messages plus longs et ne garantira pas l'acceptation de messages plus longs en provenance du PUF.
NOTE – Le numéro de version du PUF, égal au numéro de révision principal, est défini dans la Partie 1. La valeur du champ PUFType fera l'objet d'extensions ultérieures et est actuellement fixée à 0.			

En tant que valeur de retour, l'identificateur d'échange (ExID, *exchange identifier*) devient disponible. Il identifie la liaison d'échange entre PUF et NAF. Cet identificateur doit être fourni à l'interface avec d'autres fonctions de la méthode d'échange, au cours de la phase de conversation et de la phase de désenregistrement d'association.

Nom de la fonction: PciRegister

Valeur de retour de fonction: Errorcode (PCI_INTEGER)

Success
InvalidPCIHandle
NAFnotAvailable
NAFBusy
MaxPUFsExceeded
InvalidPUFType
InvalidPUFVersion
InvalidRegisterInfoStructure
InvalidOpSysInfoStructure[1]

Paramètres:

Nom	Type générique	Valeur d'appel ou de retour	Commentaire
PCIHandle	PCI_HANDLE	valeur d'appel	présentation et valeurs des pointeurs-PCI dépendant du système d'exploitation
PCIRegisterInfo	PCIRegisterInfo structure	valeur d'appel	contient des informations propres à l'interaction PUF-NAF, telles que la version et le type du PUF (Tableau 22).
PCIOpSysInfo	PCIOpSysInfo structure	valeur d'appel	contient des informations propres aux systèmes d'exploitation. Pour les détails, voir les Parties 7 à 9.
ExID	PCI_EXID	valeur de retour	identificateur d'échange

[1] Pour plus de détails sur les codes d'erreur (propres aux systèmes d'exploitation), voir les Parties 7 à 9.

8.2 Phase de désenregistrement

Cette phase est la dernière dans l'échange d'informations entre PUF et NAF. Lorsqu'un PUF souhaite supprimer son association avec le NAF, il doit invoquer la fonction **PciDeregister**. L'utilisation de la fonction **PciDeregister** est obligatoire avant la terminaison du dispositif PUF.

Lorsque le dispositif PUF se dissocie au moyen de cette fonction, le NAF doit libérer toutes les ressources éventuellement attribuées pour ce PUF, telles que des connexions déjà existantes.

8.2.1 PciDeregister

Cette fonction dissocie un dispositif PUF d'un dispositif NAF. L'association entre PUF et NAF est désignée par l'identificateur ExID.

Nom de la fonction: PciDeregister

Valeur de retour de fonction: Errorcode (PCI_INTEGER)

Success
InvalidExID
NAFnotAvailable
NAFBusy

Paramètre:

Nom	Type générique	Valeur d'appel ou de retour	Commentaire
ExID	PCI_EXID	valeur d'appel	identificateur d'échange reçu comme résultat d'une fonction PciRegister précédente

Une fois retourné, l'identificateur ExID utilisé devient non valide, même si le code d'erreur retourné indique une erreur lors du désenregistrement d'association. Aucun nouvel accès au NAF n'est possible au moyen de cet identificateur ExID.

8.3 Phase de conversation

Dans la phase de conversation, l'interaction entre le PUF et le NAF se compose d'un échange de messages et de données. Cet échange est assuré par, respectivement, les fonctions d'échange génériques **PciPutMessage** et **PciGetMessage**. Des messages sont envoyés dans les deux sens, un par un et entièrement. Messages et données sont associés. La structure de bloc de paramètres de message PCI (PCI-MPB, *PCI message parameter block*) contient des informations sur les pointeurs de message et de données.

Les messages sont traités par le dispositif NAF de manière asynchrone mais l'exécution des fonctions d'échange est synchrone. Pendant que le dispositif PUF commande l'échange de messages, ceux-ci ne sont émis ou reçus que lorsque le PUF le souhaite.

8.3.1 Emission de messages

La fonction PciPutMessage est fournie au PUF pour envoyer des messages au NAF. Avant d'utiliser cette fonction, le PUF doit insérer dans le bloc PCI-MPB les valeurs appropriées et fournir les adresses du tampon de messages et du tampon de données, ce dernier seulement si le PUF envoie des données associées au message. Le bloc PCI-MPB contient l'identificateur de message et des détails relatifs à l'utilisation des tampons de messages et de données.

8.3.2 Réception de messages

Pour obtenir un message, le PUF émet simplement un appel de fonction **PciGetMessage**. ll peut utiliser cette fonction pour scruter la disponibilité de messages. En retour de fonction, on indique s'il y a eu ou non transfert de message. Pour éviter la scrutation, le dispositif PUF peut choisir d'être informé au moyen d'un mécanisme de type signalisation, dès qu'un message est disponible. Le dispositif NAF informera le PUF de tout événement de disponibilité de messages. Ce mode de fonctionnement améliore la performance globale du système. Cependant, le dispositif PUF doit toujours obtenir le message proprement dit au moyen d'un appel de la fonction **PciGetMessage**.

8.3.3 Réception de messages utilisant la méthode de scrutation

Pour recevoir un message par cette méthode, le PUF doit scruter le NAF à plusieurs reprises afin de déterminer si un message est disponible. Si ce n'est pas le cas, cela doit être indiqué d'une certaine façon.

Pour être en mesure de recevoir un message, le dispositif PUF fournit au dispositif NAF un bloc PCI-MPB correctement constitué, qui doit contenir les adresses d'un tampon de messages et d'un tampon de données, respectivement. La capacité de ces tampons doit être assez importante pour que le message attendu puisse être reçu. La fourniture d'un tampon de données est toutefois facultative car des données ne sont pas associées à tous les messages. Ce sont les indications locales dont dispose le PUF qui détermineront si un tel tampon est nécessaire. Le dispositif NAF indique la longueur totale utilisée pour chaque tampon. Si aucune donnée n'est disponible avec le message, cela sera indiqué par la valeur zéro pour le champ de longueur utilisée. L'absence de message sera indiquée par le type NOMESSAGE (0) dans le champ d'identificateur de message du bloc PCI-MPB.

8.3.4 Réception de messages utilisant la méthode de signalisation

Pour recevoir un message par cette méthode, le dispositif PUF doit d'abord mettre en place un mécanisme permettant au dispositif NAF de lui signaler la disponibilité d'un message. C'est le rôle de la fonction **PciSetSignal**.

Cette méthode permet à un dispositif NAF d'indiquer qu'un message est à la disposition du dispositif PUF avant que celui-ci utilise la fonction **PciGetMessage**. Cette indication n'implique pas le transfert du message entre le PUF et le NAF.

Le NAF avertit le PUF chaque fois qu'un message est disponible et ce jusqu'à ce que la fonction **PciSetSignal** soit utilisée pour supprimer le mécanisme de notification.

Pour recevoir le message du dispositif NAF, le PUF doit utiliser la fonction **PciGetMessage** telle que décrite dans le sous-paragraphe précédent.

Des restrictions peuvent limiter les appels de fonction que le dispositif PUF est autorisé à émettre lorsqu'il traite la notification. Ces restrictions dépendent du système d'exploitation.

8.3.5 Bloc de paramètres pour messages d'interface PCI (PCI-MPB)

Le Tableau 23 montre la structure du bloc de paramètres pour messages d'interface PCI (bloc PCI-MPB).

Tableau 23 – Structure du bloc de paramètres pour messages d'interface PCI (bloc PCI-MPB)

Champ de la structure	Type générique	Explication
MessageID	PCI_INTEGER	identificateur de message – doit être fourni par le PUF lors de l'invocation de la fonction PciPutMessage et qui est à la disposition du PUF au retour de la fonction PciGetMessage
MessageMaximumSize	PCI_INTEGER	longueur maximale d'un message – à fournir lors des appels de fonction PciGetMessage
MessageActualUsedSize	PCI_INTEGER	longueur réelle du message – champ qui doit être fourni par le PUF lors d'appels de la fonction PciPutMessage et qui est à la disposition du PUF au retour de la fonction PciGetMessage
DataMaximumSize	PCI_INTEGER	capacité maximale du tampon de données – ce champ doit être fourni lors d'appels de la fonction PciGetMessage
DataActualUsedSize	PCI_INTEGER	capacité réellement utilisée du tampon de données – ce champ doit être fourni lors d'appels de la fonction PciPutMessage et est à la disposition du PUF au retour de la fonction PciGetMessage

La Figure 11 présente la façon dont les messages sont envoyés ou reçus.

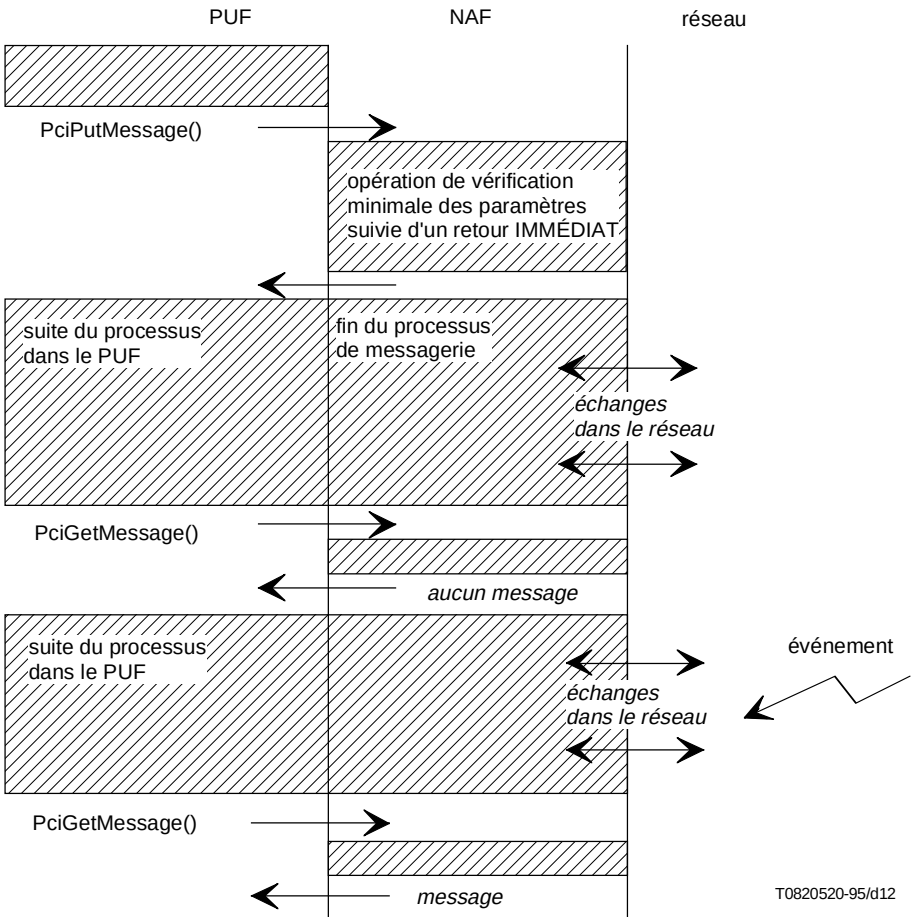


Figure 11 – Processus d'envoi ou de réception de messages

8.3.6 PciPutMessage

Cette fonction permet à un dispositif PUF d'envoyer un message à un dispositif NAF.

Nom de la fonction: PciPutMessage

Valeur de retour de fonction: Errorcode (PCI_INTEGER)

Success
InvalidExID
NAFnotAvailable
NAFBusy
InvalidPCIMPB
InvalidMessageBuffer
PCIMPBTooSmall
MessageBufferTooSmall
DataBufferTooSmall
MessageTooLarge
DataBufferRequired

Paramètres:

Nom	Type générique	Valeur d'appel ou de retour	Commentaire
ExID	PCI_EXID	valeur d'appel	identificateur d'échange reçu à la suite d'une fonction PciRegister précédente
PCIMPB	PCIMPB structure	valeur d'appel	bloc de paramètres pour messages d'interface PCI
Message	PCI_BYTEARRAY	valeur d'appel	message à envoyer au NAF
Data	PCI_BYTEARRAY	valeur d'appel	données associées au message

Le dispositif PUF indique le type de message dans le champ MessageID du bloc PCI-MPB.

Le PUF indique la capacité de chaque mémoire tampon dans les champs de type ActualUsedSize du bloc PCI-MPB, c'est-à-dire dans le champ MessageActualUsedSize dans le cas d'un tampon de messages et dans le champ DataActualUsedSize dans le cas d'un tampon de données.

Il est possible de ne fournir qu'un tampon de messages (sans tampon de données) ou l'inverse. La structure du bloc PCI-MPB est cependant requise dans tous les cas. Pour indiquer l'absence d'un tampon, le dispositif PUF peut spécifier l'absence d'adresse de tampon au lieu de fournir une valeur NULL (0).

8.3.7 PciGetMessage

Cette fonction permet à un dispositif PUF de recevoir un message d'un dispositif NAF.

Nom de la fonction: PciGetMessage

Valeur de retour de fonction: Errorcode (PCI_INTEGER)

Success
InvalidExID
NAFnotAvailable
NAFBusy
InvalidPCIMPB
InvalidMessageBuffer
PCIMPBTooSmall
MessageBufferTooSmall
DataBufferTooSmall
MessageTooLarge

Paramètres:

Nom	Type générique	Valeur d'appel ou de retour	Commentaire
ExID	PCI_EXID	valeur d'appel	identificateur d'échange reçu à la suite d'une fonction PciRegister précédente
PCIMPB	PCIMPB structure	valeur d'appel et valeur de retour	bloc de paramètres pour messages d'interface PCI
Message	PCI_BYTEARRAY	valeur de retour	message reçu du NAF
Data	PCI_BYTEARRAY	valeur de retour	données associées au message

Le PUF est chargé de fournir les mémoires tampons. Si un tampon n'a pas la capacité suffisante pour recevoir le message ou les données issus du NAF, celui-ci doit retourner une erreur.

Le dispositif PUF indique la capacité maximale de chaque tampon dans les champs MaximumSize du bloc PCI-MPB, c'est-à-dire respectivement dans le champ MessageMaximumSize pour le tampon de messages et dans le champ DataMaximumSize pour le tampon de données.

En retour, le dispositif NAF indiquera la capacité de chaque tampon utilisé, dans le champ ActualUsedSize du bloc PCI-MPB.

Pour indiquer l'absence de message, le dispositif NAF insère la valeur NOMESSAGE (0) dans le champ MessageID du bloc PCI-MPB.

8.3.8 PciSetSignal

Cette fonction permet à un dispositif PUF de demander la notification d'un événement. Un événement est constitué par tout message entrant, en provenance du réseau ou du dispositif NAF. Le mécanisme de signalisation restera activé jusqu'à ce que le PUF se dissocie du NAF ou qu'il mette fin explicitement à l'action de notification (voir ci-dessous).

Nom de la fonction: PciSetSignal

Valeur de retour de fonction: Errorcode (PCI_INTEGER)

Success
InvalidExID
NAFnotAvailable
NAFBusy
InvalidSignalNumber

Paramètres:

Nom	Type générique	Valeur d'appel ou de retour	Commentaire
ExID	PCI_EXID	valeur d'appel	identificateur d'échange reçu à la suite d'une fonction PciRegister précédente
Signal	PCI_INTEGER	valeur d'appel	valeur dépendant du système d'exploitation
SignalProcedure	PCI_PROCEDURE	valeur d'appel	valeur et présentation dépendant du système d'exploitation

Le mécanisme réellement utilisé pour cette signalisation dépend du système d'exploitation.

Tout nouvel appel de fonction **PciSetSignal** écrase le précédent.

Le mécanisme de signalisation peut être interrompu en cours de communication par la fourniture d'une valeur NULL (0) au lieu des valeurs des paramètres Signal et SignalProcedure ci-dessus.

9 Sécurité

Le présent paragraphe traite de la sécurité des communications au moyen de l'interface PCI-RNIS.

9.1 Aspects généraux de la sécurité dans les RNIS

Bien que la conception des RNIS n'ait pas prévu la prise en charge des fonctions de sécurité dans les couches inférieures, leur nature numérique en facilite l'introduction. Le déploiement du RNIS dans le réseau public est déjà bien engagé, ce qui limite les modes d'adjonction des fonctions de sécurité.

Du point de vue des applications, les besoins suivants peuvent être répertoriés en ce qui concerne la sécurité:

- garantie de la confidentialité des informations;
- identification (légitimation ou authentification) des participants à la communication;
- assurance de l'intégrité des informations communiquées;
- contrôle de l'accès à des services du réseau et à des équipements ou données de client;
- capacité de prouver à une tierce partie la réalité d'une communication, son contenu et les identités des participants en cause (non-répudiation).

Du point de vue de la sécurité, le RNIS est plus qu'un service de communication dans les couches inférieures. Un RNIS doit tenir également compte des applications, en particulier de leurs exigences en termes de sécurité.

Une base commune de normes de sécurité pour RNIS, visant en particulier l'authentification, la confidentialité et l'intégrité, peut constituer la plate-forme sur laquelle on pourra construire le système de sécurité spécifiquement adapté à diverses applications RNIS. Les techniques nécessaires existent; il ne reste qu'à les adapter aux RNIS et à les intégrer aux normes.

9.2 Sécurité offerte par l'interface PCI-RNIS

Bien qu'aucune norme de sécurité dans les couches inférieures du RNIS ne soit disponible, l'interface PCI-RNIS offre un accès à des fonctions de sécurité pouvant être mises à la disposition du dispositif NAF. Cet accès offre une approche initiale vers l'introduction de la sécurité d'utilisation dans le RNIS.

Le dispositif PUF peut accéder comme suit aux fonctions de sécurité du dispositif NAF:

– *au moyen du service complémentaire d'identification de la ligne appelante (CLIP, calling line identification presentation):*

  Ce service fournit au PUF le numéro RNIS de l'appelant, éventuellement avec des informations de sous-adresse. Le numéro RNIS et l'information de sous-adressage sont fournis par le réseau et peuvent donc être utilisés pour l'identification de l'appelant. Ce service complémentaire offre au PUF un moyen d'identifier le correspondant.

– *au moyen des messages de sécurité suivants dans le plan d'administration:*

  – ASecurityReq

  – ASecurityCnf

  Cet accès de sécurité donne au PUF l'accès à des fonctions de chiffrement et de sécurisation pouvant être assurées par le dispositif NAF. Ces messages permettent d'échanger les informations requises pour utiliser les fonctions de sécurité du NAF. Cet accès de sécurité est un moyen de garantir la confidentialité des informations. On trouvera aux 7.2.9 et 7.2.10 des informations sur l'emploi de ces messages dans le plan d'administration.

9.3 Renforcement de la sécurité offerte par l'interface PCI-RNIS

Etant donné qu'aucune norme n'existe au sujet de la sécurité dans les RNIS, seules des certaines caractéristiques de sécurité peuvent être introduites dans l'interface PCI-RNIS; elles sont décrites au 9.2 ci-dessus.

Malgré l'absence de normes de sécurité dans les RNIS, on peut estimer l'influence qu'elles pourraient avoir sur l'interface PCI-RNIS. On peut distinguer trois modes d'introduction de la sécurité:

1) *fonctions de sécurité sous forme de services complémentaires*

   Ce niveau devrait avoir peu d'influence sur l'interface PCI-RNIS ou sur les dispositifs PUF. Il convient que de tels services complémentaires soient traités comme les autres;

2) *fonctions de sécurité sous forme de protocole spécifique dans le NAF*

   Si un protocole de sécurité fonctionne dans une des couches inférieures, le dispositif PUF peut n'avoir à fournir à ce protocole que les informations de sécurité nécessaires. Cela pourra prendre la forme d'une extension du plan d'administration pour permettre le transfert de ces informations. Plusieurs modes d'extension sont possibles:

   – adjonction d'un message;

   – extension des ensembles d'attributs pour y insérer les informations de sécurité;

   – insertion des informations de sécurité dans les objets NCO;

3) *fonctions de sécurité sous forme d'une nouvelle pile de protocoles de sécurité définie pour les RNIS*

   Si de nouveaux protocoles de sécurité RNIS doivent être établis, l'interface PCI-RNIS doit impérativement être modifiée. De nouveaux protocoles devront sans doute être établis dans le plan d'utilisateur et dans le plan de commande. Le mécanisme d'extension assuré par l'interface PCI-RNIS peut être utilisé pour fournir ces nouveaux protocoles.

Bien que l'on puisse en faire une estimation, l'incidence de l'introduction de la sécurité dans l'interface PCI-RNIS fera l'objet d'une étude ultérieure.

Remplacée par une version plus récente

Annexe A

Téléphonie

La présente annexe décrit les différents types d'équipements externes traités dans la présente partie.

A.1 Equipement externe de type 1

L'équipement externe de type 1 est la forme la plus simple du matériel téléphonique. Il ne comporte ni commande de décrochage ni mécanisme de numérotation. Il ne contient que le dispositif émetteur-récepteur et ne gère pas la signalisation RNIS. Cet équipement est entièrement commandé par le dispositif NAF. Un message particulier du plan de commande indique au dispositif PUF si cet équipement externe est disponible (c'est-à-dire raccordé ou non au dispositif NAF).

Il appartient au dispositif NAF de connecter un canal à ce type d'équipement externe lorsqu'un tel canal devient actif.

Si cet équipement externe est en cours d'utilisation, une demande CConnectReq visant à utiliser cet équipement sera rejetée au moyen d'une indication CDisconnectInd, avec la valeur de cause 47 (ressource indisponible).

Si cet équipement externe est en cours d'utilisation et qu'un appel entrant tente d'utiliser cet équipement, il y a lieu que le NAF envoie au PUF correspondant une indication CConnectInd. Le PUF est alors habilité à rendre l'équipement externe disponible. S'il ne le fait pas, toute tentative de connexion doit être rejetée avec la valeur de cause 47 (ressource indisponible) dans le message CDisconnectInd.

A.2 Equipement externe de type 2

Cet équipement externe contient une commande de décrochage mais pas de mécanisme de numérotation. Il ne gère pas la signalisation RNIS. Il peut fournir certaines informations au PUF quant à l'état du combiné, au moyen de deux messages du plan de commande. Cet équipement externe peut donc provoquer des transitions d'état dans le PUF pour les appels entrants et sortants.

Un message du plan de commande est défini pour indiquer au PUF la disponibilité de l'équipement externe (connecté ou non au NAF).

Il appartient au dispositif NAF de connecter un canal à ce type d'équipement externe lorsqu'un tel canal devient actif. Il appartient au PUF de faire en sorte que la commande de décrochage soit dans l'état désiré au moment où le canal devient actif.

Si cet équipement externe est en cours d'utilisation et qu'un appel entrant tente d'utiliser cet équipement, il y a lieu que le NAF envoie au PUF correspondant une indication CConnectInd. Le PUF est alors habilité à rendre l'équipement externe disponible. S'il ne le fait pas, toute tentative de connexion doit être rejetée avec la valeur de cause 47 (ressource indisponible) dans le message CDisconnectInd.

A.3 Equipement externe de type 3

Cet équipement contient une commande de décrochage mais pas de mécanisme de numérotation. Il est connecté au RNIS et est donc capable de gérer la signalisation RNIS lors de l'arrivée d'un appel entrant, si l'ordinateur est hors tension.

Cet équipement peut fournir certaines informations au PUF sur l'état du combiné, au moyen de deux messages du plan de commande. Cet équipement externe peut donc provoquer des transitions d'état dans le PUF pour les appels entrants et sortants.

Un message particulier du plan de commande indique au dispositif PUF si cet équipement externe est disponible (c'est-à-dire raccordé ou non au dispositif NAF).

Si cet équipement externe est en cours d'utilisation et qu'un appel entrant tente d'utiliser cet équipement, il y a lieu que le NAF envoie au PUF correspondant une indication CConnectInd. Le PUF détermine alors s'il y a lieu de rendre l'équipement externe disponible. S'il ne le fait pas, toute tentative de connexion doit être rejetée avec la valeur de cause 47 (ressource indisponible) dans le message CDisconnectInd.

A.4 Equipement externe de type 4

Ce type d'équipement externe contient un mécanisme de numérotation mais pas de commande de décrochage. Il ne gère pas la signalisation RNIS. Ce type d'équipement permet d'effectuer une numérotation sans (en bloc) ou avec chevauchement. Dans le cas d'une numérotation avec chevauchement, le PUF reçoit, pour chaque touche activée tour à tour sur le clavier, un message contenant le code de cette touche. Dans le cas d'une numérotation en bloc, le PUF reçoit un seul message du plan de commande, contenant l'adresse et/ou la sous-adresse complète(s) du terminal distant.

Si cet équipement externe contient une commande de décrochage, il peut fournir certaines informations au dispositif PUF concernant l'état du combiné, au moyen de deux messages du plan de commande.

Un message particulier du plan de commande indique au dispositif PUF si cet équipement externe est disponible (c'est-à-dire raccordé ou non au dispositif NAF).

Toutes les actions de numérotation et de manipulation de combiné (s'il est présent), effectuées sur ce type d'équipement externe, peuvent provoquer des transitions d'état dans le PUF pour les appels entrants et sortants.

Il appartient au dispositif NAF de connecter un canal à ce type d'équipement externe lorsqu'un tel canal devient actif.

Si cet équipement externe est en cours d'utilisation et qu'un appel entrant tente d'utiliser cet équipement, il y a lieu que le NAF envoie au PUF correspondant une indication CConnectInd. Le PUF détermine alors s'il y a lieu de rendre l'équipement externe disponible. S'il ne le fait pas, toute tentative de connexion doit être rejetée avec la valeur de cause 47 (ressource indisponible) dans le message CDisconnectInd.

A.5 Equipement externe de type 5

Ce type d'équipement externe contient un mécanisme de numérotation mais pas de commande de décrochage. Il est connecté au RNIS et est donc capable de gérer la signalisation RNIS lorsque le serveur (par exemple un ordinateur personnel) est hors tension. L'équipement externe de type 5 permet d'établir des appels sortants et de répondre aux appels entrants, à partir de ce terminal.

Ce type d'équipement permet d'effectuer une numérotation sans (en bloc) ou avec chevauchement. Dans le cas d'une numérotation avec chevauchement, le PUF reçoit, pour chaque touche activée tour à tour sur le clavier, un message contenant le code de cette touche. Dans le cas d'une numérotation en bloc, le PUF reçoit un seul message du plan de commande, contenant l'adresse et/ou la sous-adresse complète(s) du terminal distant.

Si cet équipement externe contient une commande de décrochage, il peut fournir certaines informations au dispositif PUF concernant l'état du combiné, au moyen de deux messages du plan de commande.

Un message particulier du plan de commande indique au dispositif PUF si cet équipement externe est disponible (c'est-à-dire raccordé ou non au dispositif NAF).

Toutes les actions de numérotation et de manipulation de combiné (s'il est présent), de type 5, peuvent provoquer des transitions d'état dans le PUF pour les appels entrants et sortants.

Il appartient au dispositif NAF de connecter un canal à ce type d'équipement externe lorsqu'un tel canal devient actif.

Si cet équipement externe est en cours d'utilisation et qu'un appel entrant tente d'utiliser cet équipement, il y a lieu que le NAF envoie au PUF correspondant une indication CConnectInd. Le PUF détermine alors s'il y a lieu de rendre l'équipement externe disponible. S'il ne le fait pas, toute tentative de connexion doit être rejetée avec la valeur de cause 47 (ressource indisponible) dans le message CDisconnectInd.

Remplacée par une version plus récente

Annexe B

Mappages entre RNIS et messages/paramètres d'interface PCI-RNIS

La présente annexe indique les mappages entre les protocoles utilisés et les messages de l'interface PCI-RNIS.

B.1 Messages du plan de commande

Tableau B.1 – Mappages entre les messages du plan de commande et les messages Q.931

Message d'interface PCI	Message selon Q.931	Sens	Notes
CAlertReq	ALERTING (alerte)	utilisateur-réseau	
CAlertInd	ALERTING (alerte)	réseau-utilisateur	
CConnectReq	SETUP (établissement)	utilisateur-réseau	
CConnectInd	SETUP (établissement)	réseau-utilisateur	
CConnectRsp	CONNECT (connexion)	utilisateur-réseau	
CConnectCnf	CONNECT (connexion)	réseau-utilisateur	
CDisconnectReq	DISCONNECT (déconnexion), RELEASE (libération), RELEASE COMPLETE (fin de libération)	utilisateur-réseau	Note 1
CDisconnectInd	DISCONNECT (déconnexion), RELEASE (libération), RELEASE COMPLETE (fin de libération)	réseau-utilisateur	Note 1
CDisconnectRsp	RELEASE (libération)	utilisateur-réseau	Note 1
CDisconnectCnf	RELEASE (libération), RELEASE COMPLETE (fin de libération)	réseau-utilisateur	Note 1
CProgressInd	PROGRESS (progression)	réseau-utilisateur	
CStatusInd	STATUS (état)	réseau-utilisateur	Note 2
CProceedingInd	CALL PROCEEDING (appel en cours)	réseau-utilisateur	
CSetupAckInd	SETUP ACKNOWLEDGE (accusé de réception d'établissement)	réseau-utilisateur	
CConnectInfoReq	INFORMATION	utilisateur-réseau	
CUserInformationReq	USER INFORMATION (informations d'utilisateur)	utilisateur-réseau	
CUserInformationInd	USER INFORMATION (informations d'utilisateur)	réseau-utilisateur	
CCongestionControlReq	CONGESTION CONTROL (contrôle d'encombrement)	utilisateur-réseau	
CCongestionControlInd	CONGESTION CONTROL (contrôle d'encombrement)	réseau-utilisateur	
CSuspendReq	SUSPEND (suspension)	utilisateur-réseau	
CSuspendCnf	SUSPEND ACKNOWLEDGE (accusé de réception de suspension), SUSPEND REJECT (refus de suspension)	réseau-utilisateur	
CResumeReq	RESUME (reprise)	utilisateur-réseau	
CResumeCnf	RESUME ACKNOWLEDGE (accusé de réception de reprise), RESUME REJECT (refus de reprise)	réseau-utilisateur	
CNotifyInd	NOTIFY (notification)	réseau-utilisateur	

Tableau B.1 – Mappages entre les messages du plan de commande et les messages Q.931 *(fin)*

Message d'interface PCI	Message selon Q.931	Sens	Notes
CFacilityReq	FACILITY (ressource)	utilisateur-réseau	
CFacilityInd	FACILITY (ressource)	réseau-utilisateur	
CAddInfoReq	INFORMATION	utilisateur-réseau	
CAddInfoInd	INFORMATION	réseau-utilisateur	

NOTE 1 – Dans le cas des messages d'interface PCI de type CDisconnect*, le message spécifiquement reçu ou envoyé au RNIS dépend de l'état de l'appel au moment où le message CDisconnect* est reçu du PUF ou envoyé à celui-ci. Selon le message RNIS qui a provoqué l'envoi de l'indication CDisconnectInd, la réponse CDisconnectRsp peut provoquer ou ne pas provoquer l'envoi d'un message au RNIS. La confirmation CDisconnectCnf ne doit pas être mappée à partir d'un message issu du RNIS si la demande CDisconnectReq sert à répondre à l'indication CConnectInd.

NOTE 2 – Ce message d'interface PCI peut être produit par la détection d'une erreur de protocole dans le NAF ou par l'indication d'une erreur de protocole dans un message STATUS reçu du RNIS.

NOTE 3 – Les messages de l'équipement externe ne figurent pas dans le tableau ci-dessus.

B.2 Paramètres du plan de commande

Le Tableau B.2 ci-dessous montre le mappage entre les paramètres du plan de commande et les éléments d'information conformes à la Recommandation Q.931 [1].

Tableau B.2 – Paramètres du plan de commande

Paramètre du plan de commande	Elément d'information selon Q.931 [1]
BearerCap	capacité support
CalledNumber	numéro du demandé
CalledSubaddress	sous-adresse du demandé
CallingNumber	numéro du demandeur
CallingSubaddress	sous-adresse du demandeur
CauseToPUF	cause
CauseToNAF	cause
ChannelIdentification	identification du canal
CongestionLevel	niveau d'encombrement
ConnectedNumber	numéro de l'appelé
ConnectedSubaddress	sous-adresse connectée
DateTime	date/heure
Display	affichage
Facility	fonctionnalité
HLC	compatibilité de couche supérieure
Keypad	clavier
LLC	compatibilité de couche inférieure
NotificationIndicator	indicateur de notification
NumberComplete	fin de numérotation
ProgressIndicator	indicateur de progression
Signal	signal
UserToUserInfo	usager-usager

Remplacée par une version plus récente

Annexe C

Contenu des ensembles d'attributs statiques

La présente annexe contient une description complète des attributs statiques qui sont fournis par un dispositif NAF. Les règles d'établissement de ces attributs sont associées aux protocoles.

C.1 Ensembles d'attributs statiques dans le plan de commande

Les ensembles d'attributs décrits ci-dessous font appel aux conventions suivantes:

nom: à utiliser avec le message ANCOCreateReq

BC: contenu du paramètre BearerCap (capacité support), en octets à codage hexadécimal

LLC: contenu du paramètre LLC (compatibilité avec couches inférieures), en octets à codage hexadécimal (codage décimal entre parenthèses)

HLC: contenu du paramètre HLC (compatibilité avec couches supérieures), (codage décimal entre parenthèses).

C.1.1 Capacité de services supports en mode circuit

C.1.1.1 Signaux de parole

nom: "SPEECH_A-LAW"

BC: 80 90 A3

LLC: non utilisé

HLC: non utilisé

nom "SPEECH_μ-LAW"

BC: 80 90 A2

LLC: non utilisé

HLC: non utilisé

C.1.1.2 Informations numériques sans restriction

nom: "UNRESTRICTED"

BC: 88 90

LLC: non utilisé

HLC: non utilisé

C.1.1.3 Informations numériques avec restriction

nom: "UNRESTRICTED/56"

BC: 88 90 01 8F

LLC: non utilisé

HLC: non utilisé

C.1.1.4 Transfert d'informations audio à 3,1 kHz

nom: "AUDIO_A-LAW"

BC: 90 90 A3

LLC: non utilisé

HLC: non utilisé

nom: "AUDIO_μ-LAW"

BC: 90 90 A2

LLC: non utilisé

HLC: non utilisé

C.1.2 Service support en mode paquet

nom: "D_CHANNEL_HDL"

BC: 88 C0 C6 E6

LLC: non utilisé

HLC: non utilisé

C.1.3 Téléservices

nom: "TELEPHONY_A-LAW"

BC: 80 90 A3

LLC: non utilisé

HLC: standard = 0
identification = 1

nom: "TELEPHONY_μ-LAW"

BC: 80 90 A2

LLC: non utilisé

HLC: standard = 0
identification = 1

nom: "TELEFAX_G4"

BC: 88 90

LLC: selon l'équipement terminal: octet 3a. Non utilisé: octets 4 et 5.
octet 6 (couche 2) = 0D (13)
octet 7 (couche 3) = 07

HLC: standard = 0
identification = 21 (33)

Appendice I

Directives pour le développement de dispositifs NAF

Le corps principal de la présente partie contient la description de l'interface PCI-RNIS, vue du dispositif PUF. Conformément à cette approche, certains points, non directement associés au PUF mais ayant une incidence sur le développement du NAF, ne sont pas décrits. Ces points peuvent être utiles pour le développement du NAF: ils seront donc décrits dans le présent appendice avec des directives pour le développement des NAF conformément au corps de la présente partie.

Le mappage entre le codage du service complémentaire d'information de taxation et le codage spécial utilisé dans l'interface PCI-RNIS est un de ces points non couverts dans le corps de la présente partie.

Le présent appendice tient compte des trois points de vue suivants:

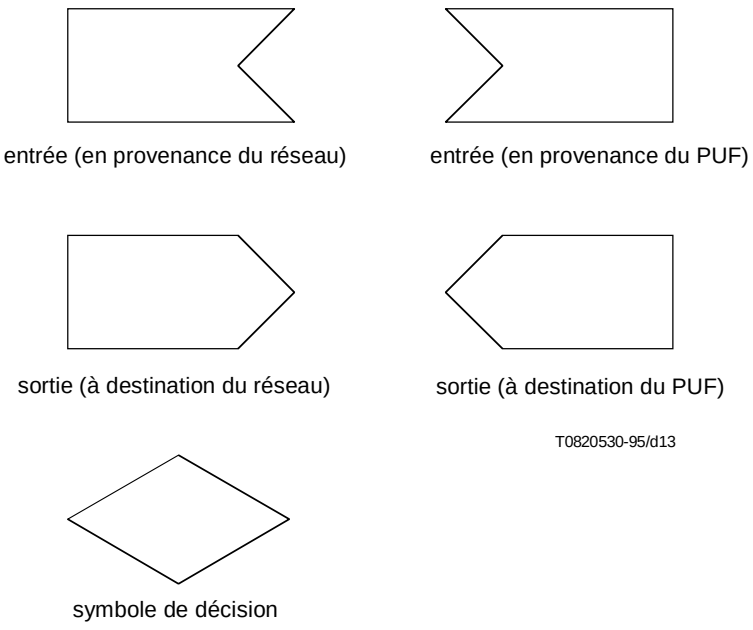
- le présent appendice indique des points additionnels. Le NAF doit être implémenté sur la base de la présente partie. Il y a lieu qu'il implémente l'interface PCI-RNIS de manière que les propriétés décrites soient fournies;
- il convient de donner priorité au corps principal de la présente partie en cas d'incertitude sur un point du présent appendice ou de divergence d'interprétation entre le corps de la présente partie et du présent appendice;
- le présent appendice ne vise pas à imposer de quelconques contraintes quant à l'implémentation d'un dispositif NAF. L'objectif est de donner des directives sur la façon d'élaborer un NAF conformément à la présente partie.

I.1 Diagrammes SDL de dispositif NAF

Les diagrammes SDL ci-après montrent, à titre d'exemple, les états internes de l'étage de commande d'appel d'un dispositif NAF. Ils ne sont donnés qu'à titre explicatif. Par souci de simplification, tous les cas possibles ne sont pas représentés sur ces diagrammes.

Les primitives écrites en lettres capitales sont celles qui sont définies dans la Recommandation Q.931 [1].

Les symboles suivants sont utilisés dans le cadre de cette description. La Recommandation Z.100 donne une description complète des symboles et de leur signification.



Symbole d'état

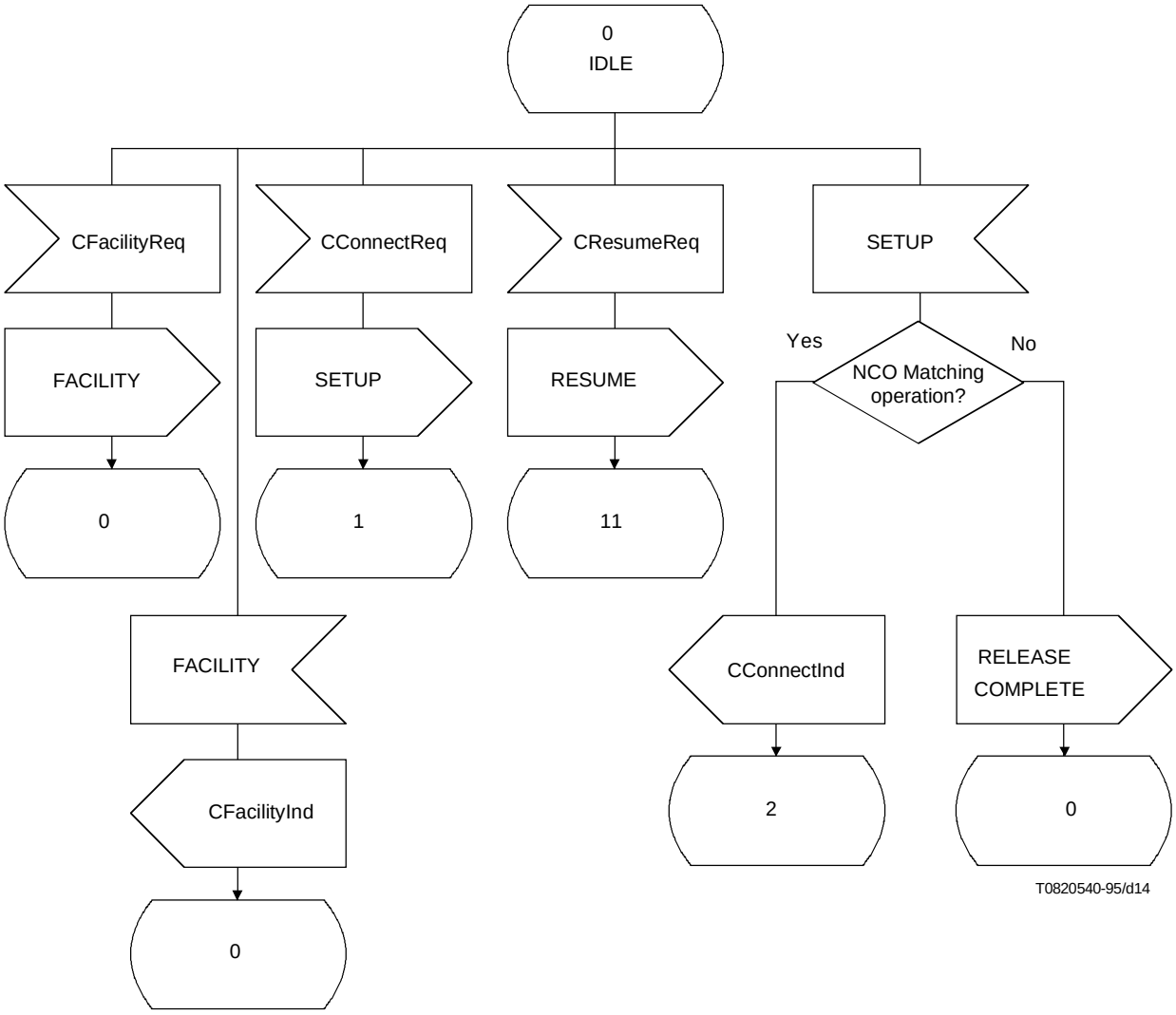


Figure I.1 – Etat de repos (IDLE)

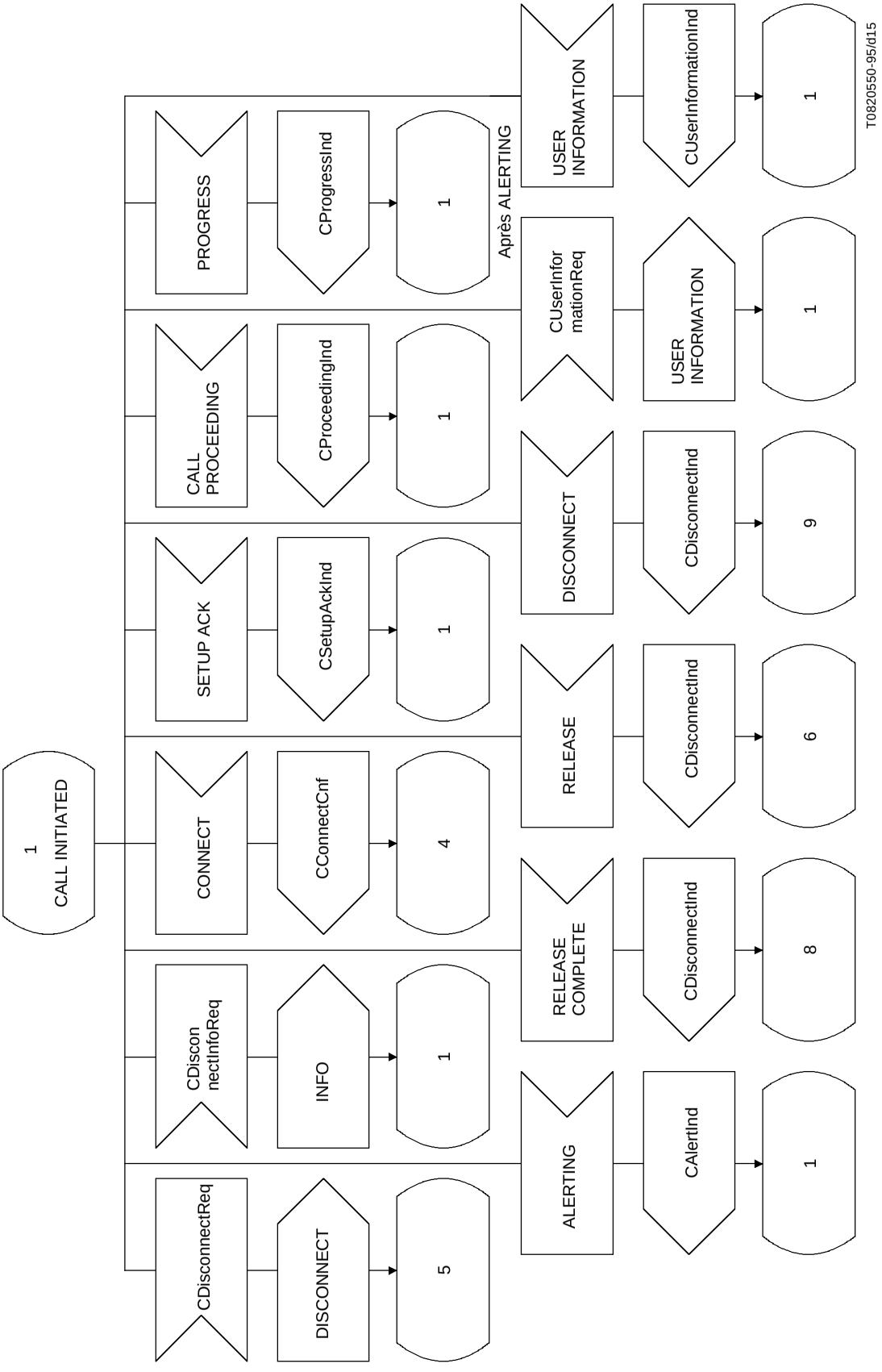


Figure I.2 – Appel lancé (CALL INITIATED)

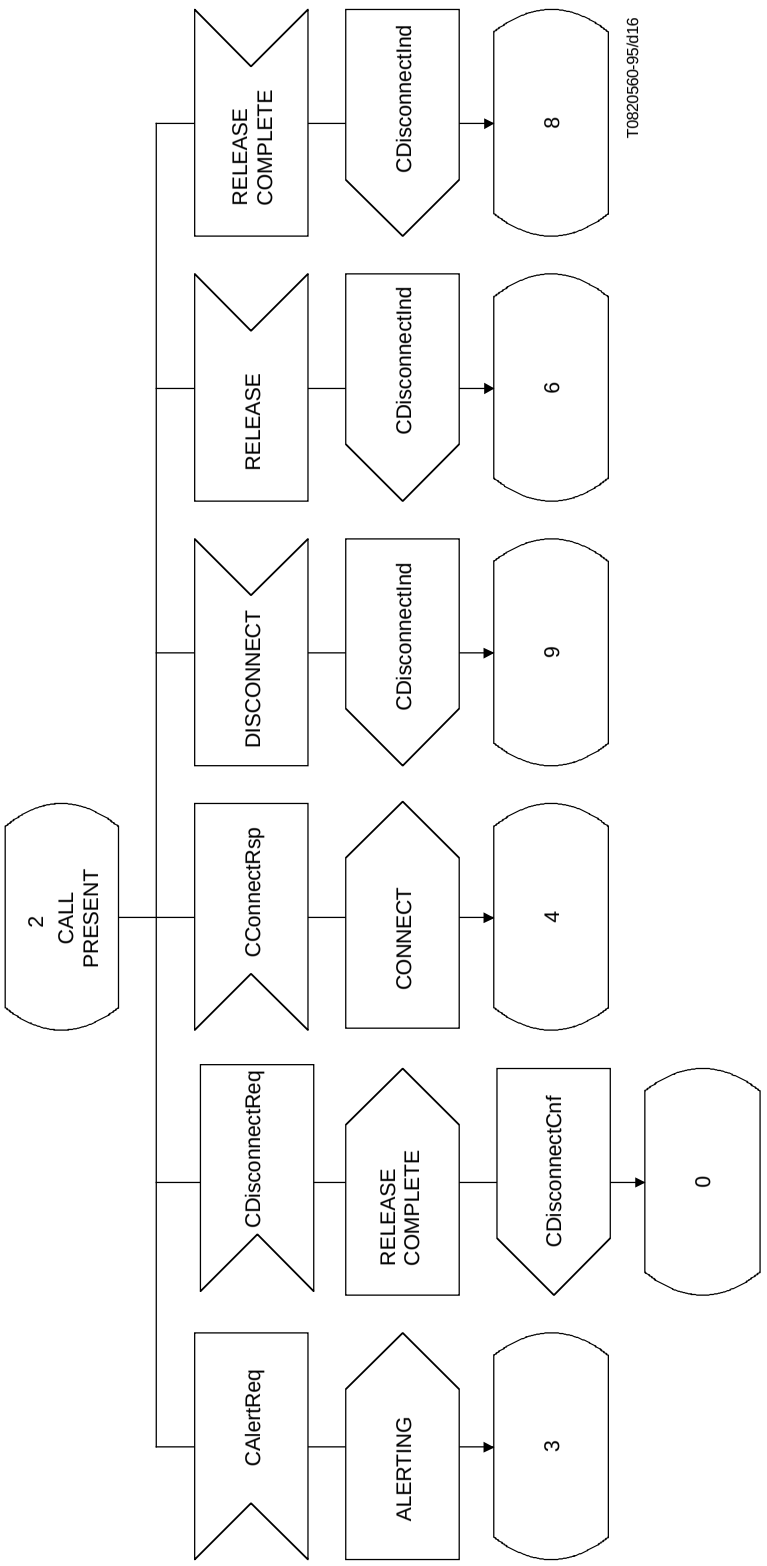


Figure I.3 – Appel présent (CALL PRESENT)

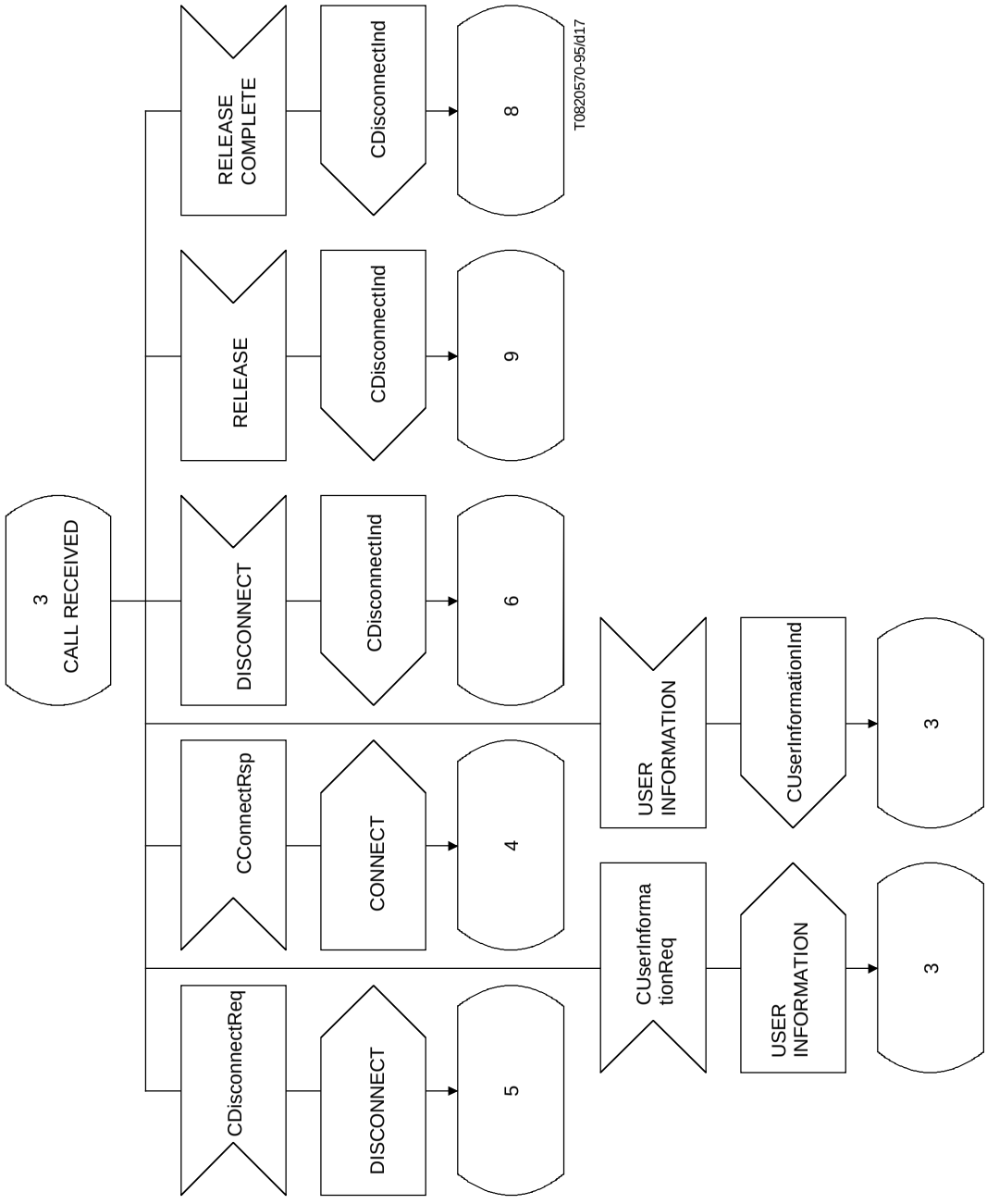


Figure I.4 – Appel reçu (CALL RECEIVED)

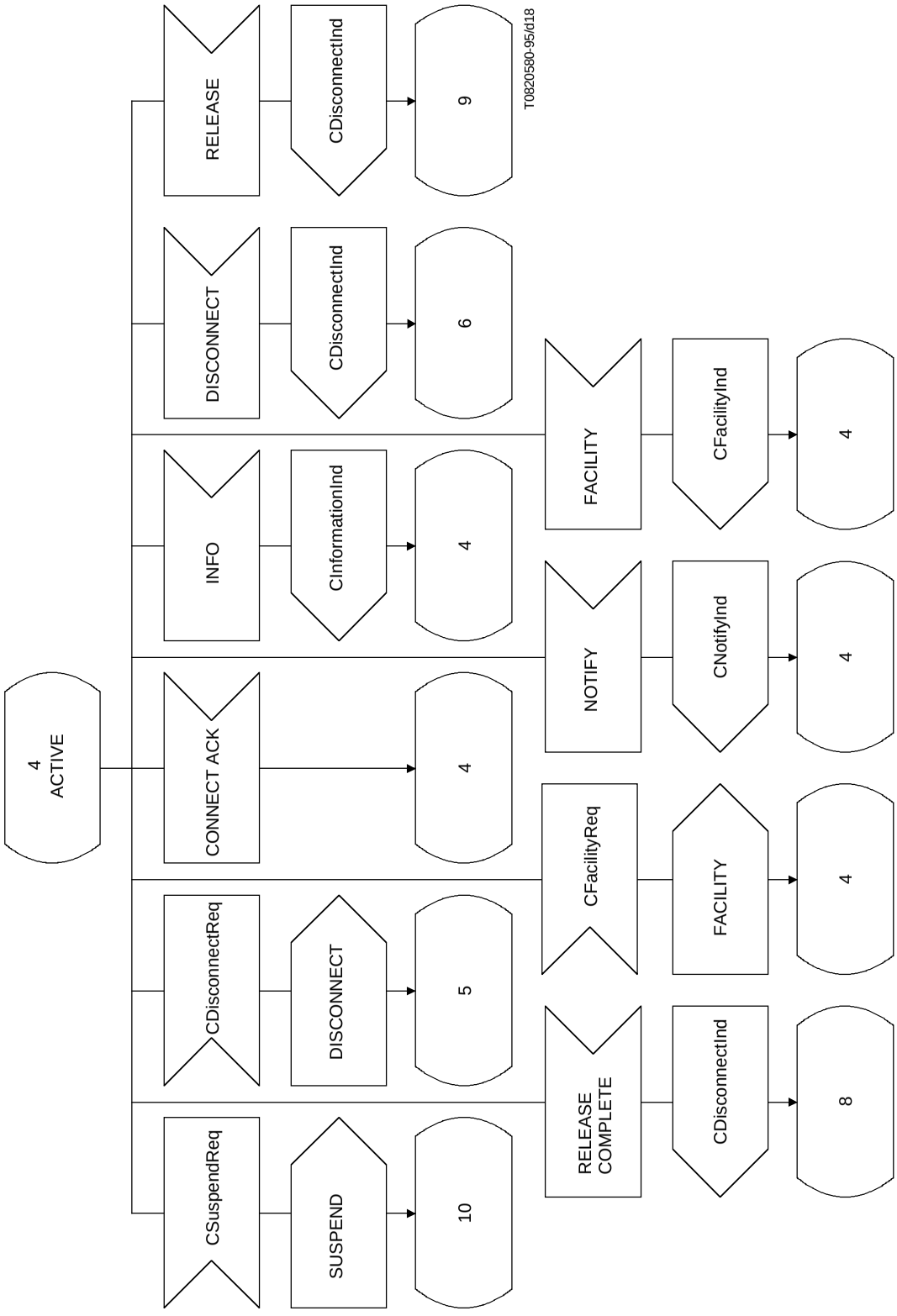


Figure I.5 – Connexion active (ACTIVE)

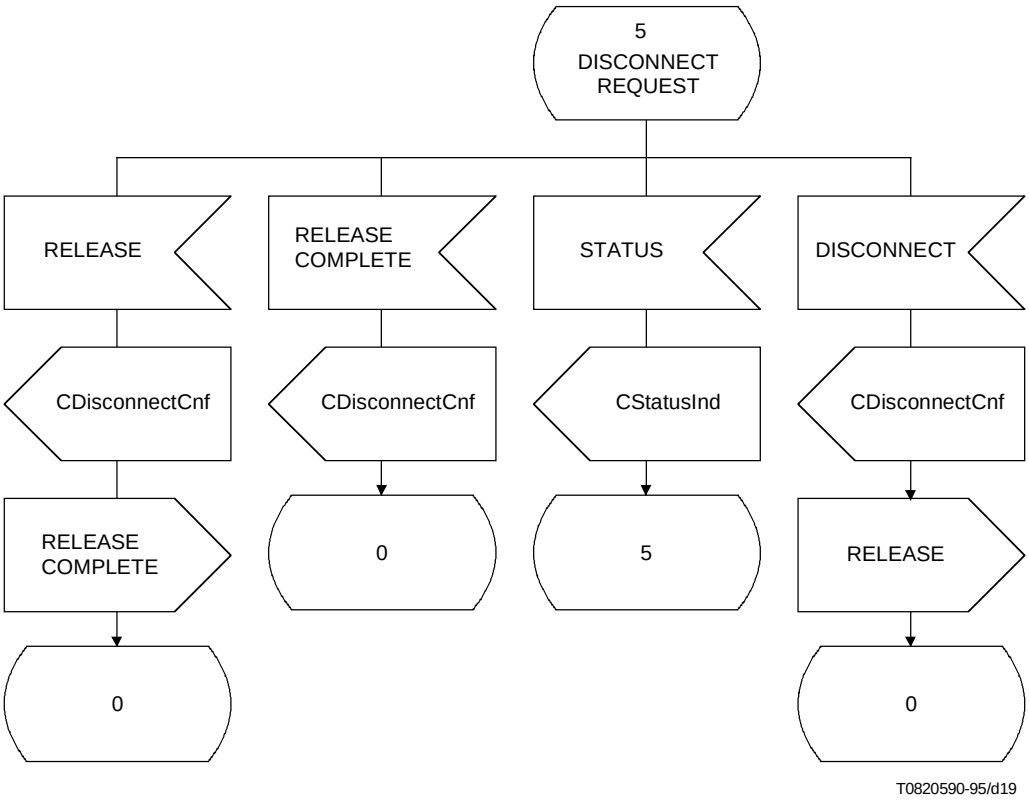


Figure I.6 – Demande de déconnexion (DISCONNECT request)

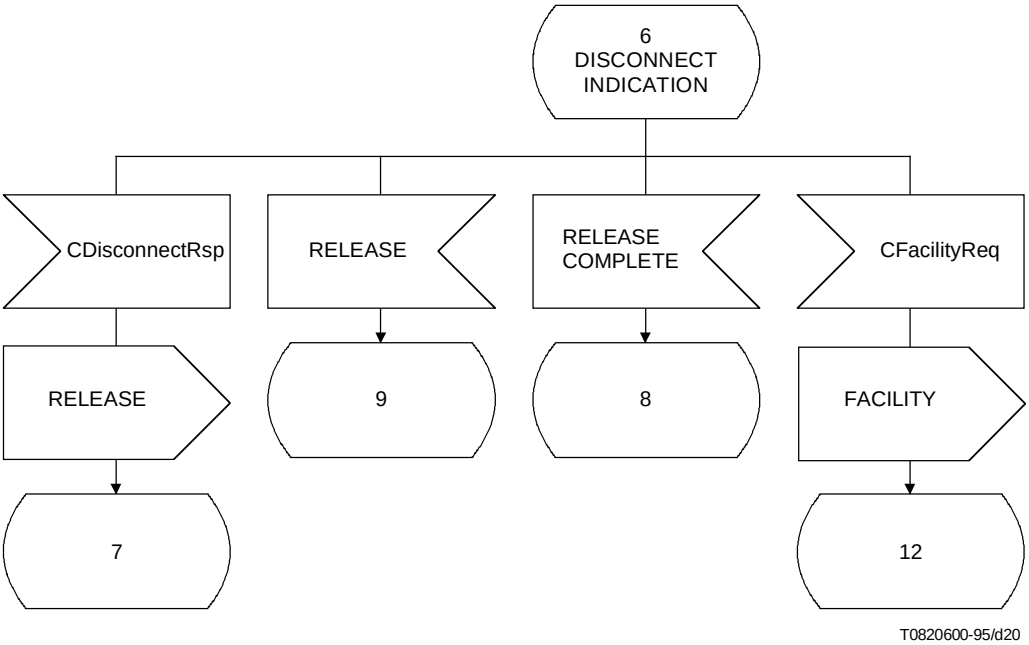


Figure I.7 – Indication de déconnexion (DISCONNECT indication)

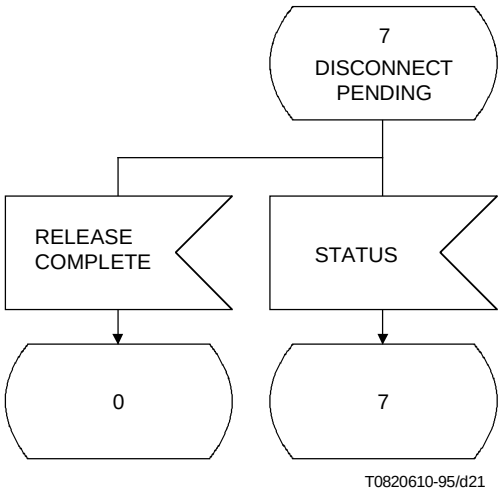


Figure I.8 – Déconnexion imminente (DISCONNECT pending)

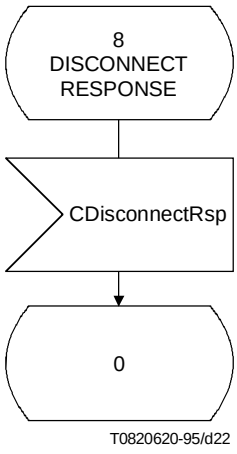


Figure I.9 – Réponse de déconnexion (DISCONNECT response)

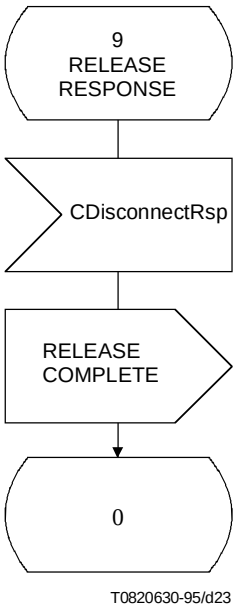


Figure I.10 – Réponse de libération (RELEASE response)

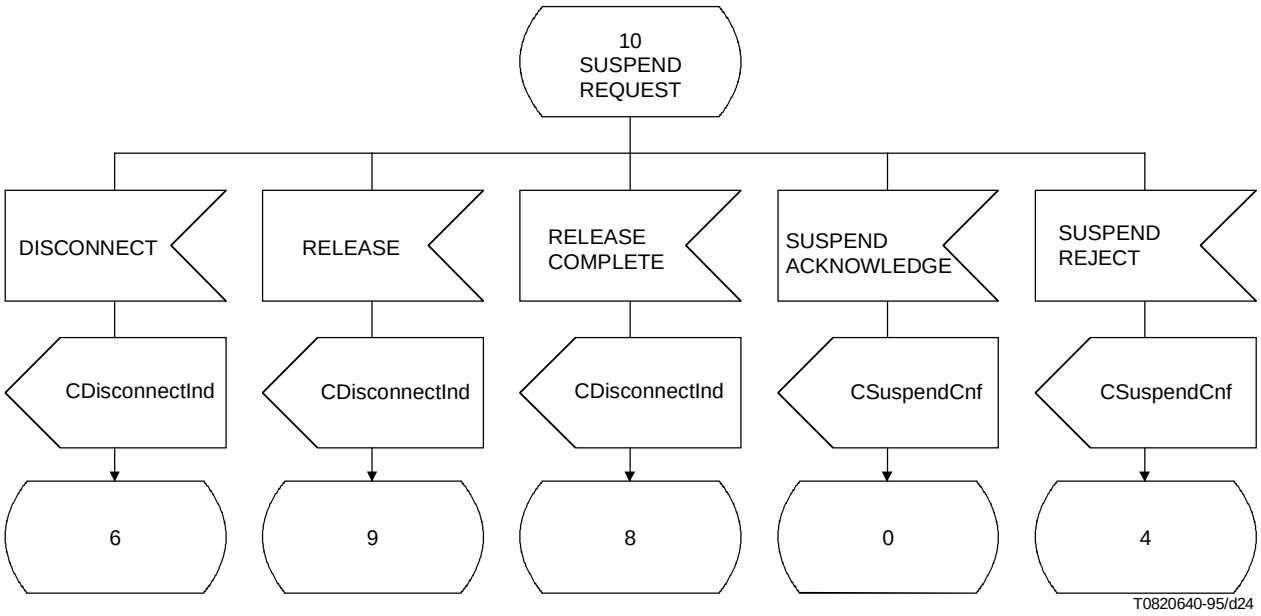


Figure I.11 – Demande de suspension (SUSPEND request)

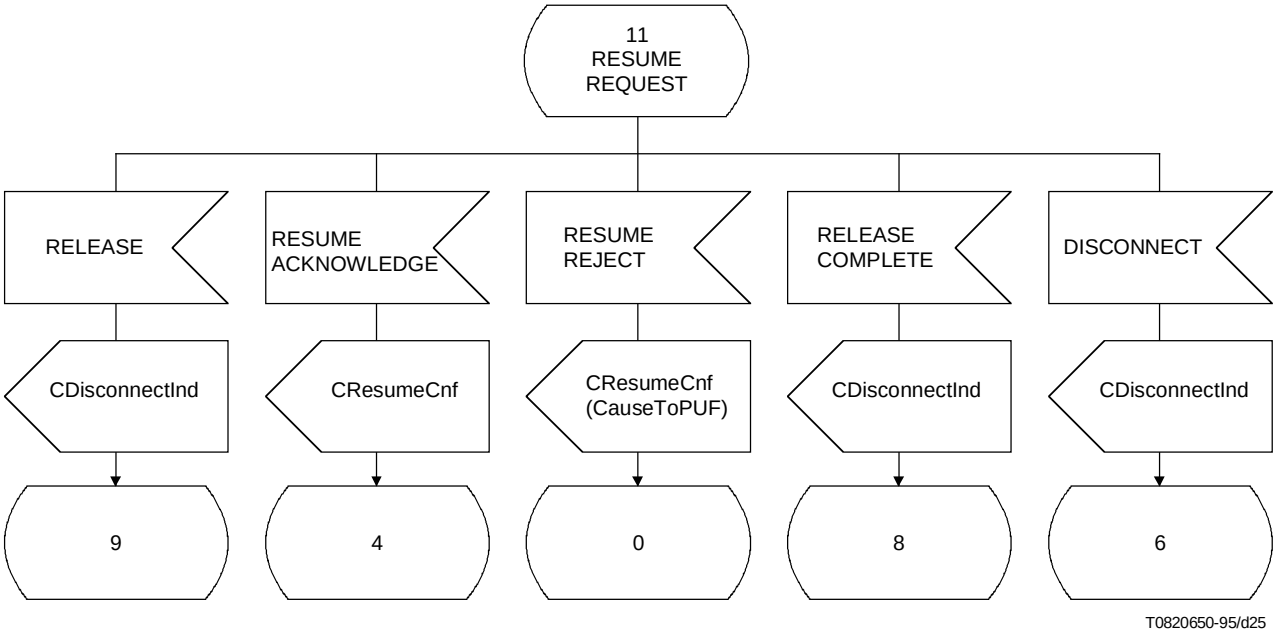


Figure I.12 – Demande de reprise (RESUME request)

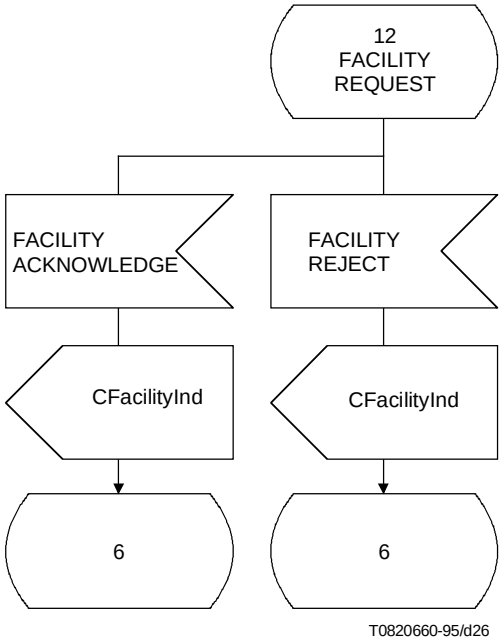


Figure I.13 – Demande de fonctionnalité (FACILITY request)

I.2 Informations fournies par le dispositif NAF

La composition des messages peut varier, en termes d'éléments d'information. Les conventions suivantes s'appliquent à la fourniture d'éléments d'information par le NAF pour le PUF:

– *paramètres obligatoires*

  Ces éléments doivent être fournis.

– *paramètres conditionnels*

  La fourniture de ces éléments dépend de la condition applicable.

– *paramètres facultatifs*

  Ces éléments peuvent être fournis ou ne pas l'être, selon qu'ils sont ou non à la disposition du NAF.

I.3 Suspension/reprise de communications

Le dispositif NAF est appelé à gérer le mappage des identificateurs NCOID avec la référence d'appel requise par le réseau lors de la reprise d'un appel. Une fois la connexion rétablie, il y a lieu que le dispositif NAF assure le mappage de l'identificateur NCOID avec la référence d'appel, qui peut avoir changé, du côté réseau.

I.4 Gestion des erreurs

L'indication d'erreur fournie au PUF ne contient que les informations permettant au PUF de déterminer s'il y a lieu de donner suite ou non à une action particulière. Il est envisagé que le dispositif NAF fournisse des informations plus détaillées sur chaque erreur, d'une manière propre à ces dispositifs. Par exemple, un dispositif NAF peut choisir d'implémenter un journal d'erreurs sous forme de fichier dans lequel il enregistrera des renseignements détaillés concernant une erreur particulière. Ces informations sont nécessaires pour mettre au point un dispositif PUF en étude.

Les sous-paragraphes qui suivent donnent des indications sur les conditions dans lesquelles il y a lieu qu'un NAF renvoie au PUF une erreur spécifique.

En cas de répétition non autorisée d'un paramètre, seul le premier nombre valide de répétitions de ce paramètre sera pris en compte, les répétitions suivantes étant ignorées.

Si le réseau fournit un paramètre facultatif, le dispositif NAF est chargé de le communiquer au dispositif PUF, par le message approprié.

I.4.1 Codes de retour relatifs aux fonctions

Les conditions dans lesquelles il y a lieu d'émettre ces codes sont décrites au 7.8.5.

I.4.2 Plan d'administration

Les conditions dans lesquelles il y a lieu d'émettre ces codes dans le plan d'administration sont décrites au 7.8.6. Pour le message ACreateNCOReq, le nombre de combinaisons paramétriques possibles rend la vérification complexe. Il convient d'y procéder selon l'ordre indiqué dans le Tableau I.1.

Tableau I.1 – Vérification du message ACreateNCOReq

Paramètre	Test	Action
Tous paramètres	non autorisé	erreur de paramètre non valide
	tous valides	continuer
NCOType	manquant	erreur de paramètre manquant
	longueur non valide	erreur de longueur non valide
	valeur non valide	erreur de type NCO non valide
	valeur valide	continuer
Direction	manquant	erreur de paramètre manquant
	longueur non valide	erreur de longueur non valide
	valeur non valide	erreur de type de sens non valide
	valeur valide	continuer

Tableau I.1 – Vérification du message ACreateNCOReq *(fin)*

Paramètre	Test	Action
AttributeName	manquant	erreur d'absence de nom d'attribut
	longueur non valide	erreur de longueur non valide
	non valide	erreur d'invalidité de nom d'attribut
	correct	continuer
Paramètre AttributeContent ou AddressContent	manquant	erreur de paramètre manquant
	longueur non valide	erreur de longueur non valide
	non valide	erreur de contenu non valide
	correct	continuer
NafCoordination	présent mais non requis	erreur de paramètre non valide
	longueur non valide	erreur de longueur non valide
	valeur non valide	erreur de valeur de coordination non valide
	correct	continuer
GroupID	présent mais non requis	erreur d'identificateur de groupe
	manquant	erreur d'identificateur de groupe
	longueur non valide	erreur de longueur non valide
	valeur non valide	erreur d'identificateur de groupe
	correct	continuer
RequestID (si présent)	longueur non valide	erreur de longueur non valide
	présent	traiter le message

I.4.3 Plan de commande

Les erreurs retournées dans le paramètre Cause correspondent à celles de l'élément d'information Cause dans la Recommandation Q.931 [1]. Cette correspondance permet au dispositif NAF de transférer des informations de l'élément d'information Cause dans le paramètre Cause. Dans ce cas, il convient que le dispositif NAF convertisse toutes valeurs d'élément d'information en valeurs de paramètre, comme décrit dans l'Annexe B.

Il convient que le dispositif NAF produise les messages d'erreur suivants à la suite de la vérification de paramètres contenus dans des messages allant de PUF à NAF.

Tableau I.2 – Correspondances du paramètre Cause dans le plan de commande

Valeur	Signification selon Q.931 [1]	Signification pour l'interface PCI	Message émis par	Si reçu du RNIS, traité par
1	numéro non affecté (non attribué)		RNIS	PUF
2	pas d'acheminement à destination du réseau de transit spécifié		RNIS	NAF (Note 1)
3	pas d'acheminement vers la destination		RNIS	PUF
6	canal inacceptable		RNIS	NAF (Note 1)
7	appel attribué et en cours d'établissement dans un canal établi		RNIS	PUF

Tableau I.2 – Correspondances du paramètre Cause dans le plan de commande *(suite)*

Valeur	Signification selon Q.931 [1]	Signification pour l'interface PCI	Message émis par	Si reçu du RNIS, traité par
16	libération normale de l'appel		RNIS	PUF
17	usager occupé		RNIS	PUF
18	pas de réponse d'usager		RNIS	PUF
19	pas de réponse d'usager (usager alerté)		RNIS	PUF
21	refus de l'appel		RNIS	PUF
22	numéro changé		RNIS	PUF
26	libération d'usager non retenu		RNIS	PUF
27	destination en dérangement		RNIS	PUF
28	format du numéro non valide (numéro incomplet)	le paramètre a un format d'adresse non valide	NAF, RNIS	PUF
29	refus de fonctionnalité	fonctionnalité non fournie par ce NAF	NAF, RNIS	PUF
30	réponse à STATUS ENQUIRY		RNIS	NAF
31	normal, non spécifié		RNIS	PUF
34	pas de circuit/canal disponible	aucun canal du type demandé n'est disponible pour le moment à partir de ce NAF	NAF, RNIS	PUF
38	réseau en dérangement		RNIS	NAF (Note 1)
41	dérangement temporaire		RNIS	NAF (Note 1)
42	encombrement de l'équipement de commutation		RNIS	PUF
43	suppression de l'information d'accès		NAF, RNIS	PUF (Note 3)
44	canal/circuit demandé non disponible	aucun canal du type demandé n'est disponible à partir de ce NAF	NAF, RNIS	PUF
47	ressource non disponible, non spécifiée	équipement externe demandé non disponible	NAF, RNIS	PUF
49	qualité de service non disponible		RNIS	PUF
50	fonctionnalité demandée par le paramètre Facilité non profilé par l'abonnement		RNIS	PUF
57	capacité support non autorisée		RNIS	PUF
58	capacité support non disponible actuellement	service demandé par paramètre BearerCap non disponible. Utilisé par un autre PUF	NAF, RNIS	PUF
63	service ou option non disponible, non spécifié		RNIS	PUF
65	service demandé par capacité support non implémenté	service demandé par paramètre BearerCap non fourni par le NAF	NAF, RNIS	PUF
66	type de canal non mis en service	le NAF n'est pas compatible avec ce type de canal	NAF, RNIS	PUF
69	fonctionnalité demandée non implémentée	le NAF n'est pas compatible avec cette fonctionnalité	NAF, RNIS	PUF

Tableau I.2 – Correspondances du paramètre Cause dans le plan de commande *(fin)*

Valeur	Signification selon Q.931 [1]	Signification pour l'interface PCI	Message émis par	Si reçu du RNIS, traité par
70	seule une capacité support à information numérique avec restriction est disponible		RNIS	NAF (Note 1)
79	service ou option non implémenté, non spécifié		RNIS	PUF
81	valeur de référence d'appel non valide	identificateur NCOID non valide	NAF, RNIS	NAF (Note 1)
82	le canal identifié n'existe pas	le canal permanent identifié n'est pas défini	NAF, RNIS	NAF (Note 1)
83	un appel suspendu existe mais cette identité d'appel n'existe pas		RNIS	NAF (Note 1)
85	pas d'appel suspendu	le NCOID n'identifie pas de connexion suspendue	NAF, RNIS	NAF (Note 1)
86	un appel ayant l'identité d'appel demandée a été libéré		RNIS	NAF (Note 1)
88	destination incompatible		RNIS	PUF
91	sélection de réseau de transit non valide		RNIS	NAF (Note 1)
95	message non valide, non spécifié		RNIS	NAF (Note 1)
96	paramètre obligatoire manquant	absence d'un paramètre obligatoire	NAF, RNIS	NAF (Note 1)
97	type de message inexistant ou non implémenté sur ce réseau	identificateur de message inexistant ou non implémenté sur ce NAF	NAF, RNIS	NAF (Note 1)
98	message incompatible avec l'état de l'appel ou type de message inexistant ou non implémenté	message incompatible avec l'état du NCO ou bien identificateur de message inexistant ou non implémenté	NAF, RNIS	NAF (Note 1)
99	paramètre non valide	paramètre non valide	NAF, RNIS	NAF (Note 1)
100	contenu du paramètre non valide	contenu du paramètre non valide	NAF, RNIS	NAF (Note 1)
101	message incompatible avec état actuel	message incompatible avec état actuel	NAF, RNIS	NAF (Note 1)
102	reprise à l'expiration de la temporisation		RNIS	NAF (Note 2)
111	erreur de protocole, non spécifiée		RNIS	NAF (Note 1)
127	interfonctionnement, non spécifié		RNIS	PUF

NOTE 1 – Lorsque des valeurs de cause sont traitées par le NAF, il y a lieu que celui-ci tente d'effectuer une reprise sur erreur. En cas d'échec, il convient qu'il indique sa non-disponibilité aux dispositifs PUF à association enregistrée, au moyen du code d'erreur NAFNotAvailable.

NOTE 2 – Il y a lieu que le dispositif NAF donne la suite appropriée.

NOTE 3 – Si c'est le RNIS qui a produit cette cause, il appartient au NAF de mapper cet élément d'information avec les types de paramètre, dans tout message d'information de diagnostic envoyé au dispositif PUF.

Le Tableau I.3 ci-après montre l'ordre de vérification des messages de type CConnectReq. Les informations sont issues de ce message, plus les ensembles d'attributs et d'adresses associés à l'identificateur obligatoire de la connexion réseau. Ce tableau part du principe que la vérification initiale du message a été effectuée.

Tableau I.3 – Vérification du message CConnectReq

Paramètre	Test	Action
identificateur NCOID	non valide	CStatusInd valeur du paramètre Cause = 81
	valide	continuer
paramètre d'état de message	non valide	CStatusInd valeur du paramètre Cause = 101 diagnostics = MessageID
	valide	combiner paramètres issus des ensembles d'attributs et d'adresses et du message CConnectReq, continuer
paramètres obligatoires	BearerCap manquant	CDisconnectInd valeur du paramètre Cause = 96 diagnostics = BearerCap
	le service demandé par la capacité support n'est pas de type X.25 et le numéro du demandé est manquant	CDisconnectInd valeur du paramètre Cause = 96 diagnostics = CalledNumber
	tous présents	continuer
contenu du paramètre BearerCap	contenu non valide	CDisconnectInd valeur du paramètre Cause = 100 diagnostics = BearerCap
	service non fourni par le NAF	CDisconnectInd valeur du paramètre Cause = 65
	correct	continuer
contenu du paramètre CalledNumber (si présent)	contenu non valide	CDisconnectInd valeur du paramètre Cause = 100 diagnostics = CalledNumber
	correct	continuer
paramètres non reconnus	présents	CDisconnectInd valeur du paramètre Cause = 99 diagnostics = type de paramètre non reconnu
	non présents	continuer
erreur de contenu dans un paramètre facultatif	présent	CStatusInd valeur du paramètre Cause = 100 diagnostics = type de paramètre erroné continuer (ignorer le paramètre)
	non présent	traiter le message

I.5 Configuration d’un dispositif NAF

Le sous-paragraphe suivant contient des informations sur la configuration d'un dispositif NAF. Il est destiné à aider les développeurs de dispositifs NAF et ne vise pas à donner une liste exhaustive des éléments configurables.

I.5.1 Configuration globale

Tableau I.4 – Configuration globale

Paramètre	Valeur par défaut suggérée	Commentaire
nombre de PUF gérés	8	

I.5.2 Paramètres de configuration du système

Tableau I.5 – Paramètres de configuration du système

Paramètre	Valeur par défaut suggérée	Commentaire
DMA		nombre d'adresses d'accès direct en mémoire (DMA) utilisées par l'adaptateur
I/O address		adresse d'entrée/sortie utilisée par l'adaptateur
IRQ		interruptions utilisées par l'adaptateur
DRAM		double adresse d'accès à la mémoire vive pour partage entre l'adaptateur et l'environnement de l'ordinateur ce paramètre peut également indiquer la durée du cycle d'adaptation

I.5.3 Configuration du plan de commande

Tableau I.6 – Configuration du plan de commande

Paramètre	Valeur par défaut suggérée	Commentaire
nombre de canaux D	1	
définition d'un canal D: – type de réseau; – automatique; – fixe + numéro; – taille fenêtre trame (K); – N200; – N201; – N202. temporisateurs: – T200; – T201; – T202; – T203.	1	
nombre de canaux B	2	
nombre de canaux B permanents	0	
liste des identificateurs de canaux B permanents	1..256	
nombre de canaux D permanents (SAPI 16) pour chaque canal D: – automatique; – fixe + numéro; – signalisation d'identité.	0	

I.6 Gestion des mémoires tampons

Les mémoires tampons dont le contenu est transmis de PUF à NAF sont copiées par le NAF vers un espace interne dès que possible, ce qui permet aux dispositifs PUF de réutiliser ces mémoires dès le retour de fonction.

L'instant exact du traitement du message dépend du dispositif NAF et est hors du domaine d'application de la présente partie.

Dans le sens PUF-NAF, le message est transmis en une seule étape, avec les données qui lui sont éventuellement associées. Si l'un des tampons de messages ou de données a une capacité insuffisante pour contenir les informations d'un message complet ou des données, le dispositif NAF doit renvoyer une erreur et le message ne doit pas être acheminé au dispositif PUF. Afin de faciliter la tâche de celui-ci, la longueur du plus grand message est déterminée au cours de la phase d'enregistrement d'association. La capacité d'un tampon de données est étroitement liée au type de connexion et à son protocole. Le dispositif PUF doit consulter les données d'initialisation du protocole du plan utilisateur pour obtenir la valeur correcte du plus long paquet de données possible.

Si un dispositif NAF a besoin de nouveaux tampons internes, il est chargé de les rechercher. Il pourra s'agir d'une opération de configuration, effectuée par le constructeur du dispositif NAF mais hors du domaine d'application de la présente partie. Le constructeur du dispositif NAF peut décrire la façon de réaliser cette opération et les conséquences que l'on peut en attendre.

Appendice II

Echantillon de codeur/décodeur de type-longueur-valeur (TLV)

```
/*
////////////////////////////////////////////////////////////////////////
///
///         ÉCHANTILLONS
///
///         CODEUR ET DÉCODEUR TLV
///
////////////////////////////////////////////////////////////////////////
*/

#include <memory.h>
#include <stdarg.h>

/*
 * Définition des types
 */
typedef int       BOOL;
#define FALSE                   0
#define TRUE                    1

#define LG_MAX_MESSAGE 128

/* Définition des structures */
struct sParameter /* Structure intermédiaire qui reçoit le paramètre à ajouter */
        {
        int iMessageLength;
        char scMessage[LG_MAX_MESSAGE];
        };

/*
```

```
////////////////////////////////////////////////////////////////////////
///
///     Fonction:     AddOctetParameter
///
///     Règle:              Ajouter à un message un paramètre codé en octets
///
///     Paramètres:
///                   pointeur sur la structure qui reçoit le paramètre à ajouter
///                   type de paramètre
///                   valeur de paramètre
///
///     Retour:
///           TRUE:       Succès
///           FALSE:      Erreur en cours de traitement
///
////////////////////////////////////////////////////////////////////////
*/
BOOL AddOctetParameter(   struct sParameter *pMessage, unsigned char cType, unsigned char cValue)
     {
     if (pMessage->iMessageLength + 3 > LG_MAX_MESSAGE) /* Tampon insuffisant */
          {
          /* Traitement de l'erreur de longueur de message */
          return FALSE;
          }/* si */

     /* codage TLV */
     pMessage->scMessage[pMessage->iMessageLength++] = cType;
     pMessage->scMessage[pMessage->iMessageLength++] = 1; /* longueur = 1 pour octet */
     pMessage->scMessage[pMessage->iMessageLength++] = cValue; /* contenu */

     /* Succès */
     return TRUE;
     }/* AddOctetParameter */

/*
////////////////////////////////////////////////////////////////////////
///
///     Fonction:     AddStringParameter
///
///     Règle:        Ajouter un paramètre de type Chaîne (d'octets) dans un message
///
///     Paramètres:
///                   pointeur sur la structure qui reçoit le paramètre à ajouter
///                   type de paramètre
///                   longueur de paramètre
///                   valeur de paramètre (pointeur)
///
///     Retour:
///           TRUE:       Succès
///           FALSE:      Erreur en cours de traitement
///
////////////////////////////////////////////////////////////////////////
*/
BOOL AddStringParameter(  struct sParameter *pMessage,
                          unsigned char cType,
                          int iLg,
                          unsigned char *lpValue)
     {
     if (iLg == 0) return FALSE;
```

```
        if (pMessage->iMessageLength + iLg + 2 > LG_MAX_MESSAGE) /* Tampon insuffisant */
                {
                /* Traiter erreur de longueur de message */
                return FALSE;
                }/* si */

        /* codage TLV */
        pMessage->scMessage[pMessage->iMessageLength++] = cType; /* Ajouter le type */
        pMessage->scMessage[pMessage->iMessageLength++] = iLg; /* Longueur */
        memcpy(pMessage->scMessage+pMessage->iMessageLength, lpValue, iLg); /* Valeur */
        pMessage->iMessageLength += iLg;

        /* Succès */
        return TRUE;
        }/* AddStringParameter */

/*
////////////////////////////////////////////////////////////////////////
///
///     Fonction:       ExtractParameter
///
///     Règle:          Trouver un paramètre spécifique et indiquer son emplacement
///
///     Paramètres:
///                     adresse du message
///                     longueur actuelle du message
///                     type de paramètre recherché
///                     pointeur du pointeur sur l'emplacement de la valeur
///                     pointeur d'un entier indiquant l'emplacement de la longueur du paramètre
///
///     Retour:
///             TRUE: Succès
///             FALSE: Erreur en cours de traitement
///
////////////////////////////////////////////////////////////////////////
*/
BOOL ExtractParameter(  unsigned char *lpMessage,
                        unsigned int iLgMessage, unsigned char cType,
                        unsigned char * *lplpValue, unsigned int *lpiLgValue)
        {
        while (iLgMessage > 0) /* pour tous les paramètres de message */
                {
                if (*lpMessage != cType)
                        {
                        /* traitement du paramètre suivant */
                        iLgMessage -= lpMessage[1] + 2;
                        lpMessage += lpMessage[1] + 2;
                        continue;
                        }/* si */

                /* le type de paramètre met à jour les informations pour l'appelant */
                *lplpValue = lpMessage + 2;
                *lpiLgValue = lpMessage[1];

                /* Succès */
                return TRUE;
                }/* pendant */

        return FALSE;
        }/* ExtractParameter */
```

Remplacée par une version plus récente

Appendice III

Liste des paramètres

Le Tableau III.1 ci-dessous contient la liste complète des paramètres définis dans l'interface PCI-RNIS. La première colonne indique le code (type) du paramètre. La deuxième colonne donne le nom du paramètre. La dernière colonne indique dans quel plan le paramètre est utilisé. Certains paramètres peuvent apparaître dans plusieurs plans.

Tableau III.1 – Liste des paramètres de l'interface PCI-RNIS

Identificateur du paramètre	Nom du paramètre	Plan d'utilisation
1	Algorithm	administration
2	Bcug	utilisateur et administration
3	BearerCap	commande et administration
4	Bit_DQM	commande et administration
5	CalledDTEAddress	utilisateur et administration
6	CalledDTEAddressExt	utilisateur et administration
7	CalledNumber	commande et administration
8	CalledSubaddress	commande et administration
9	CallingDTEAddress	utilisateur et administration
10	CallingDTEAddressExt	utilisateur et administration
11	CallingNumber	commande et administration
12	CallingSubaddress	commande et administration
13	CAttributeName	commande et administration
14	CauseToNAF	commande
15	CauseToPUF	commande et administration
16	CDirection	administration
17	ChannelIdentification	commande
18	ChargingInfo	administration
19	CompletionStatus	administration, commande et utilisateur
20	CongestionLevel	commande
21	ConnectedNumber	commande
22	ConnectedSubaddress	commande
23	DateTime	commande et administration
24	Display	commande
25	ExtEquipAvailability	commande
26	ExtEquipBlockDialling	commande
27	ExtEquipKeyPressed	commande
28	ExtEquipName	commande et administration
29	ExpeditedData	commande
30	Facility	commande
31	FacilityData	utilisateur
32	FastSelect	utilisateur et administration
33	GroupID	administration

Tableau III.1 – Liste des paramètres de l'interface PCI-RNIS *(suite)*

Identificateur du paramètre	Nom du paramètre	Plan d'utilisation
34	HLC	commande et administration
35	IdleFlag	administration
36	Key	administration
37	Keypad	commmande
38	L2ConnectionMode	administration
39	L2FrameSize	administration
40	L2WindowSize	administration
41	L2XID	administration
42	L3ConnectionMode	administration
43	L3IncomingCount	administration
44	L3OutgoingVCCount	administration
45	L3TwoWayCount	administration
46	LLC	commande et administration
47	ManufacturerCode	administration
48	MoreData	commande
49	NCOID	administration, commande et utilisateur
50	NCOType	administration
51	NotificationIndicator	commande
52	PacketSize	utilisateur et administration
53	ProgressIndicator	commande
54	QOSParameters	utilisateur et administration
55	ReadyFlag	utilisateur
56	RequestID	administration
57	ReceiptConfirm	utilisateur
58	RespondingDTEAddress	utilisateur
59	RespondingDTEAddressExt	utilisateur
60	SelectorID	administration
61	TEI	utilisateur et administration
62	UProtocol	utilisateur et administration
63	UAttributeName	administration
64	UDirection	administration
65	UserData	utilisateur
66	UserToUserInfo	contrôle
67	WindowSize	utilisateur et administration
68	X213Cause	utilisateur
69	X213Origin	utilisateur
70	X25Cause	utilisateur
71	X25Diagnostic	utilisateur
72	CPParameterMask	administration
73	CPMessageMask	administration

Tableau III.1 – Liste des paramètres de l'interface PCI-RNIS *(fin)*

Identificateur du paramètre	Nom du paramètre	Plan d'utilisation
74	PPPNegotiation	utilisateur et Administration
75	FlowControlMechanism	administration
76	FlowControlCharacters	administration
77	MomentNumber	administration
78	V110BChannelDisconnection	administration
79	NumberComplete	commande
80	AdditionalInfo	commande
81	Signal	commande

Bibliographie

La présente bibliographie fait référence à des documents qui peuvent présenter une certaine importance pour les développeurs de dispositifs PUF et NAF. Ces documents pourront être utiles lors de la lecture ou de l'implémentation de la présente partie.

– ISO/CEI 9574:1992, *Technologies de l'information – Fourniture du service de réseau OSI en mode connexion par un terminal en mode paquet raccordé à un réseau numérique avec intégration de services* (RNIS).

– ISO/CEI 8878:1992, *Technologies de l'information – Télécommunications et échange d'informations entre systèmes – Utilisation du protocole X.25 pour fournir le service de réseau OSI en mode connexion*.

– Recommandation UIT-T Q.932 (1993), *Procédures génériques pour la commande des services complémentaires RNIS.*

– Recommandations UIT-T Q.951.1, Q.951.2 et Q.951.3 (1992), Q.951.3, Q.951.4, Q.951.5 et Q.951.6 (1993), Q.951.7 (1997), *Description d'étape 3 des services complémentaires d'identification de numéro utilisant le système de signalisation d'abonné numérique n° 1*.

– Recommandations UIT-T Q.953.1 (1992), Q.953.2 (1993), Q.953.3 (1997), et Q.953.4 (1995), *Description d'étape 3 pour les services complémentaires d'aboutissement des appels utilisant le système de signalisation d'abonné numérique n° 1 du RNIS.*

– Recommandation UIT-T X.211 (1995), *Technologies de l'information – Interconnexion des systèmes ouverts – Définition du service physique.*

– Recommandation UIT-T Z.100 (1993), *Langage de description et de spécification du CCITT*.

Remplacée par une version plus récente

TABLE DES MATIÈRES

PARTIE 3

PARTIE 3: ARCHITECTURE DE GESTION DES PROTOCOLES DANS LE PLAN D'UTILISATEUR

Résumé

La présente partie de la série de spécification décrit les aspects généraux concernant la gestion d'accès aux protocoles du plan d'utilisateur qui sont compatibles avec l'interface PCI-RNIS. Elle décrit en particulier l'architecture des protocoles du plan d'utilisateur et les détails du mécanisme de sélection de ces protocoles.

Introduction

L'utilisation de différentes interfaces de programmation par des équipements terminaux connectés au réseau numérique à intégration de services (RNIS) a fait obstacle au développement d'applications communes utilisant le RNIS et a limité le déploiement d'applications RNIS sur des équipements terminaux tels que les ordinateurs personnels.

La présente interface entre programmes d'application (API, *application programming interface*), appelée interface de programmation de communication (PCI, *programming communication interface)* par réseau numérique à intégration de services (PCI-RNIS), établie par l'UIT-T, permet aux applications d'accéder aux services des RNIS et de les gérer. L'interface PCI-RNIS fait l'objet d'une série de spécification, dont la présente partie constitue l'introduction à l'usage des protocoles.

L'interface PCI-RNIS a été définie de façon à offrir aux fournisseurs d'équipement terminal une norme qui permettra de réaliser la portabilité d'applications utilisant l'interface PCI-RNIS sur une gamme d'équipements terminaux tournant sur différents systèmes d'exploitation.

L'interface PCI-RNIS a été définie en fonction des besoins des développeurs d'applications et, dans la mesure du possible, élimine la nécessité d'une connaissance approfondie du RNIS. Elle a également été conçue de façon que les futures extensions du RNIS n'aient pas d'incidence sur le fonctionnement des applications existantes.

1 Domaine d'application

La présente spécification décrit l'accès au plan utilisateur offert par l'interface de programmation de communication pour réseau numérique à intégration de services (PCI-RNIS).

La présente partie décrit la gestion de protocoles d'interface PCI-RNIS assurée dans le plan d'utilisateur. Ce plan offre des opérations qui facilitent l'établissement et/ou la libération de canaux logiques de communication pour l'échange de données. Par ce plan transitent des messages qui permettent d'utiliser les protocoles sous-jacents. Ce plan est associé à la connexion d'utilisateur, qui elle-même peut être associée à un canal B ou, pour une liaison de données, au canal D.

L'interface PCI-RNIS s'insère entre les couches 3 et 4 du modèle de référence OSI. En cas d'accès transparent (par la couche Physique), le dispositif NAF considère les couches 2 et 3 comme nulles. En cas d'accès par la couche Liaison de données, le dispositif NAF considère la couche 3 comme nulle. En cas d'accès par la couche Réseau, les couches 2 et 3 sont implémentées, comme indiqué sur la Figure 1.

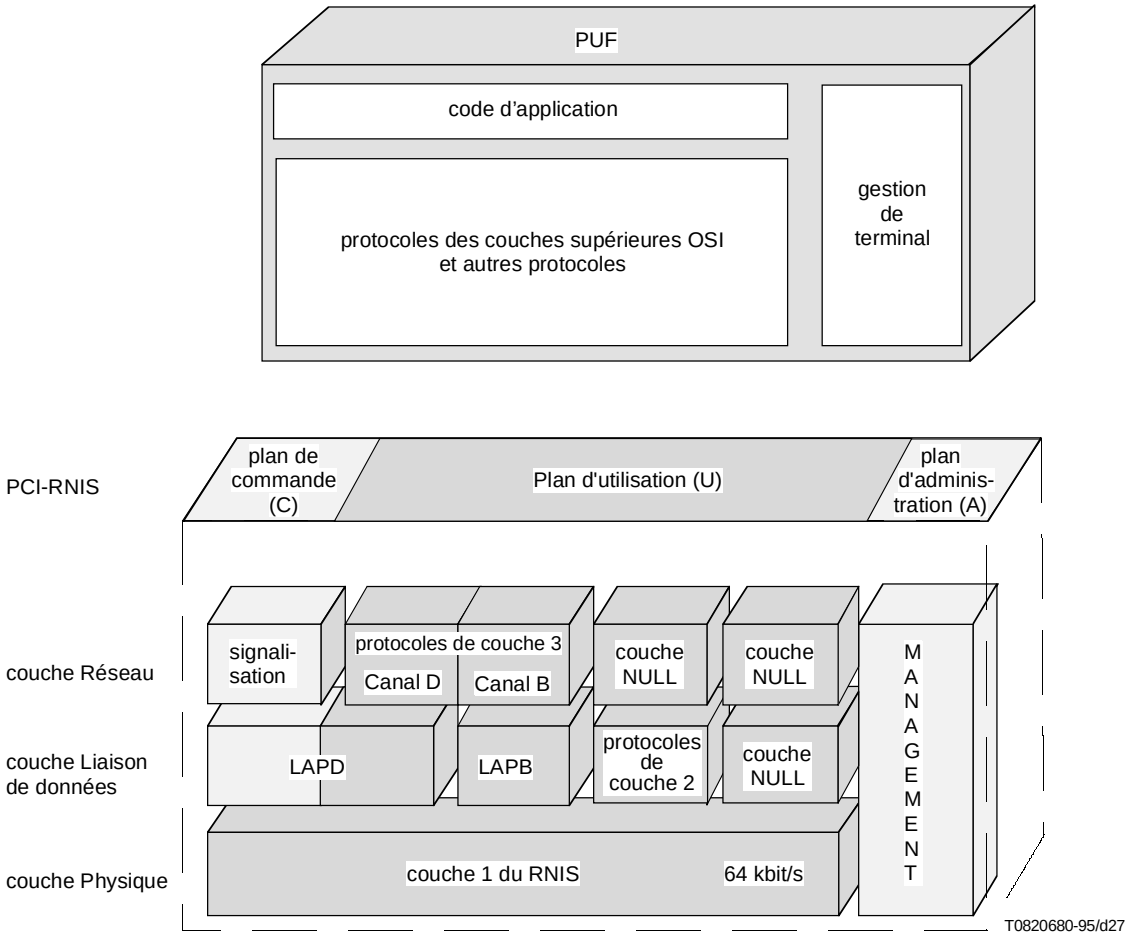


Figure 1 – Insertion dans le modèle OSI

Pour le support de l'accès transparent aux canaux B du RNIS, le plan d'utilisateur propose d'accéder au point d'accès aux services de couche Physique (Ph-SAP, *physical service access point*). Le plan d'utilisateur offre également l'accès à un autre point SAP (SAP, *service access point*). Il fournit les services définis dans la Recommandation X.213 et est donc situé au point d'accès pour le service de réseau (NSAP).

La présente partie spécifie les possibilités d'utilisation de protocoles pour le service de transfert de données. Le plan d'utilisateur offre des services pour divers protocoles, qui sont tous facultatifs et peuvent se subdiviser en trois groupes comme suit:

- protocoles de couche 1;
- protocoles de couche 2;
- protocoles de couche 3.

La présente partie est la première d'une série de Recommandations décrivant le plan d'utilisateur. Chaque groupe de protocoles fait l'objet d'un texte distinct:

Partie 3: mécanisme général, fonctions de coordination et mécanisme de sélection des protocoles

Partie 4: utilisation des protocoles de couche 1:

- accès transparent aux canaux B.

Partie 5: utilisation des protocoles de couche 2:

- ISO/CEI 7776;
- HDLC;
- PPP;
- SDLC;
- V.110.

Partie 6: utilisation des protocoles de couche 3:

- T.90;
- ISO/CEI 8208;
- T.70 NL.

2 Références

[1] Recommandation UIT-T X.213 (1995*), Technologies de l'information – Interconnexion des systèmes ouverts – Définition du service de réseau*.

[2] Partie 1: *Architecture générale*.

[3] Partie 2: *Services de base.*

[4] Partie 4: *Protocoles de couche un*.

[5] Partie 5: *Protocoles de couche deux*.

[6] Partie 6: *Protocoles de couche trois*.

3 Définitions

La présente partie définit les termes suivants:

3.1 plan d'administration: groupement logique de fonctions pour la gestion des dialogues entre dispositifs d'utilisateur d'interface PCI et dispositifs d'accès réseau (PUF-NAF, *PCI user facility-network access facility*) ainsi que pour l'accès à des ressources en dispositifs d'accès réseau (NAF) locales ou distantes.

3.2 ensemble d'attributs: ensemble de paramètres nécessaires au fonctionnement des protocoles d'utilisation et à la signalisation RNIS.

3.3 canal B: voie logique RNIS utilisée pour le transfert de données.

3.4 plan de commande: groupement logique de fonctions pour l'accès à la signalisation RNIS.

3.5 canal D: voie logique RNIS utilisée pour la signalisation et, dans certains cas, pour le transfert de données.

3.6 accès RNIS: ensemble de canaux RNIS fourni par un même dispositif d'accès réseau (NAF) pour l'accès aux services RNIS.

3.7 interface RNIS de programmation de communication (PCI-RNIS): interface logicielle orientée réseau (RNIS) qui offre des possibilités d'accès pour la programmation de la signalisation de réseau et le transfert de données d'utilisateur.

3.8 message: unité d'information transférée de part et d'autre de l'interface PCI-RNIS, entre le dispositif d'accès réseau (NAF) et le dispositif utilisateur d'interface PCI (PUF).

3.9 dispositif d'accès réseau (NAF): unité fonctionnelle située entre l'interface PCI-RNIS et les couches associées au réseau.

3.10 objet de connexion au réseau (NCO): objet abstrait contenu dans le dispositif NAF, qui doit être créé par le dispositif PUF pour donner accès à la signalisation ou aux données du réseau.

3.11 couche vide: couche vide dans le modèle de référence OSI. Une telle couche ne contient aucune fonction et transmet en transparence les requêtes et les réponses aux couches suivantes.

3.12 dispositif utilisateur d'interface PCI (PUF): unité fonctionnelle faisant appel à l'interface PCI-RNIS pour accéder à un dispositif NAF. Ce terme correspond pratiquement à l'application locale qui utilise l'interface.

3.13 connexion d'utilisateur: connexion accessible par l'intermédiaire de la propriété plan d'utilisateur.

3.14 plan d'utilisateur: groupement logique de fonctions d'accès offertes aux protocoles et aux données d'utilisateur.

3.15 protocole utilisateur: protocole exploité conformément à la propriété de plan d'utilisateur.

4 Abréviations

La présente partie utilise les abréviations suivantes:

API	interface de programmation d'application (*application programming interface*)
CONS	service réseau en mode connexion (*connection oriented network service*)
LAPB	procédure d'accès à la liaison en mode équilibré (*link access procedure balanced*)
LAPD	procédure d'accès à la liaison sur canal D (*link access procedure for D-channel*)
NSAP	point d'accès aux services de la couche réseau (*network layer – service access point)*
NAF	dispositif d'accès réseau (*network access facility*)
NCO	objet de connexion au réseau (*network connection object*)
PCI	interface de programmation de communication (*programming communication interface*)
Ph-SAP	point d'accès au service physique (*physical layer – service access point*)
PUF	dispositif utilisateur d'interface PCI (*PCI user facility*)
RNIS	réseau numérique à intégration de services
SAP	point d'accès au service (*service access point*)

5 Guide de lecture

5.1 Guide du lecteur

La présente partie est destinée aux développeurs de logiciels, aux réalisateurs d'applications et aux constructeurs d'équipement, qui y trouveront la description de l'usage général des protocoles du plan d'utilisateur.

5.2 Mode d'emploi de la présente partie

Les lecteurs qui:

- ont besoin d'un aperçu général rapide de la gestion des protocoles du plan d'utilisateur le trouveront dans la présente partie;
- envisagent d'implémenter une application au moyen de l'interface PCI-RNIS, devront lire les autres Parties [4], [5] et [6] selon le protocole examiné;
- ont l'intention de construire une carte ou un équipement d'adaptation au RNIS utilisant l'interface PCI-RNIS, devront se reporter à l'Appendice I et lire les autres Parties [4], [5] et [6] selon le protocole examiné. Les diagrammes SDL et la configuration font l'objet de ces parties spécifiques.

Le Tableau 1 donne une liste qui décrit le contenu général de la présente partie.

Tableau 1 – Table des matières de la présente partie

Paragraphe, Appendice	Contenu
paragraphe 1	domaine d'application de la présente partie, décrivant son objet
paragraphe 2	références
paragraphe 3	définitions de termes utilisés dans cette partie
paragraphe 4	définitions des abréviations utilisées dans cette partie
paragraphe 5	aperçu général
paragraphe 6	présentation générale de l'architecture d'extension pour les protocoles
Appendice I	directives pour le développement de dispositifs NAF
Appendice II	informations essentielles sur les protocoles d'utilisation

La présente partie donne:

– la définition des accès pour les messages (voir le paragraphe 2);

– la liste des protocoles supportés et la méthode de sélection (voir le paragraphe 3);

– les informations relatives à la fonction de coordination (voir le paragraphe 4);

– les critères de sélection d'objets NCO communs (voir le paragraphe 5);

– les principes de vérification d'erreur (voir le paragraphe 6);

– le contenu d'un ensemble d'attributs (voir le paragraphe 7).

6 Architecture de gestion des protocoles du plan d'utilisateur

6.1 Introduction

La gestion de données implique les fonctions suivantes:

– établissement de liaisons de données sur des connexions physiques déjà établies, si nécessaire;

– échange de données.

Le plan d'utilisateur de l'interface PCI-RNIS assure les fonctions définies par la gestion des données.

Trois ensembles de messages sont actuellement définis dans le plan d'utilisateur. Le premier permet d'accéder aux protocoles du plan d'utilisateur à l'interface avec les services de couche Réseau OSI. Le deuxième donne accès aux protocoles à l'interface avec les services de couche Liaison de données. Le troisième offre une interface transparente par laquelle le dispositif PUF implémente le protocole qui doit être appliqué dans la connexion.

Pour les deux premiers types d'accès, il importe de disposer d'une connexion sémaphore avant de pouvoir accéder à des données. En général, l'établissement de cette connexion sémaphore sera réalisé au moyen des propriétés du plan de commande, décrites en [3], tandis que l'établissement de l'accès données sera réalisé, au besoin, au moyen des propriétés du plan d'utilisateur.

Les sous-paragraphes qui suivent expliqueront les différents modes d'accès pour les messages, autorisés par l'interface PCI-RNIS.

6.2 Accès pour les messages

6.2.1 Accès à la couche Physique (accès transparent)

L'interface PCI-RNIS autorise un accès transparent pour les messages, par les couches 2 et 3 (couches vides). Elle donne donc accès direct à la couche Physique du RNIS, avec commande synchronisée au niveau des octets sur les canaux B. Le service support offert par le réseau (paramètre de capacité support) n'est pas limité aux services numériques. Le paramètre BearerCapability peut par exemple avoir la valeur "signaux vocaux".

Comme tout accès pour messages, celui-ci comporte son propre ensemble d'opérations. Le Tableau 2 donne un aperçu général des opérations effectuées dans le plan d'utilisateur pour cet accès messages.

Etant donné la nature de cet accès, seules y sont assurées les opérations permettant une commande directe du flux d'octets; aucun protocole d'utilisateur n'est activé. L'établissement d'une connexion sémaphore permet donc d'obtenir l'accès transparent aux données. Contrairement à l'accès par la couche Réseau, chaque connexion sémaphore n'autorise qu'une seule connexion de données.

Lorsque l'on utilise une connexion à accès transparent avec un objet NCO du type C associé à un équipement externe, les données envoyées sur cette connexion doivent être destinées à cet équipement externe plutôt qu'être utilisées pour produire des messages à accès transparent. Ce cas est hors du domaine d'application de la présente partie.

Tableau 2 – Opérations dans le plan utilisateur pour l'accès en transparence

Nom de l'opération	But de l'opération
données	transfert de données
erreur	indication de l'apparition d'une erreur

6.2.2 Accès à la couche Liaison de données

L'interface PCI-RNIS autorise l'accès des messages par la couche Liaison de données. Elle donne accès transparent à la couche 3 (couche vide) et donc accès direct à la couche Liaison de données du RNIS.

Cet accès pour messages comporte son propre ensemble d'opérations. Selon le protocole d'utilisateur, ces opérations seront disponibles ou indisponibles. Le Tableau 3 donne un aperçu général des opérations du plan d'utilisateur pour ces accès de messages.

Tableau 3 – Opérations du plan d'utilisateur pour l'accès à la couche Liaison de données

Nom de l'opération	But de l'opération
connexion	établissement d'une connexion d'homologue à homologue utilisateur
données	échange de données par une connexion d'utilisateur établie, donc avec commande de débit assurée par le protocole sous-jacent.
déconnexion	rupture de la connexion
prêt à recevoir	commande du débit normal des données
erreur	indication de l'apparition d'une erreur

6.2.3 Accès à la couche Réseau

L'interface PCI-RNIS autorise l'accès des messages par la couche Réseau. Elle donne accès aux protocoles du plan d'utilisateur tournant dans la couche Réseau du RNIS. Elle donne donc accès à une connexion de couche Réseau.

Cet accès pour messages comporte son propre ensemble d'opérations. Selon le protocole d'utilisateur, ces opérations seront disponibles ou indisponibles. Le Tableau 4 donne un aperçu général des opérations du plan d'utilisateur pour ces accès de messages.

L'ensemble d'opérations est fondé sur la Recommandation X.213 [1].

L'interface PCI-RNIS offre, pour certains protocoles, une fonction de coordination qui supprime la nécessité que le dispositif PUF fasse appel aux propriétés correspondantes du plan de commande. Cette fonction de coordination, disponible sur demande du PUF, établit implicitement une connexion sémaphore lorsqu'une connexion d'utilisateur est demandée.

Tableau 4 – Opérations du plan d'utilisateur pour l'accès à la couche Réseau

Nom de l'opération	But de l'opération
connexion	établissement d'une connexion d'homologue à homologue utilisateur
données	échange de données par une connexion d'utilisateur établie, donc avec commande de débit assurée par le protocole sous-jacent.
données exprès	échange de données par une connexion d'utilisateur établie, sans commande de débit assurée par le protocole sous-jacent.
accusé de réception de données	accusé de réception de données par une connexion d'utilisateur établie
réinitialisation	suppression du transfert de données
déconnexion	rupture de la connexion
prêt à recevoir (Note)	commande du débit normal des données
NOTE – Cette opération n'est pas fondée sur la Recommandation X.213 [1].	

6.3 Protocoles

6.3.1 Protocoles supportés dans le plan d'utilisateur

Il existe différents protocoles auxquels on peut accéder de différentes manières dans le plan d'utilisateur. L'un d'eux peut être sélectionné lors de la création d'un objet NCO.

Le Tableau 5 résume l'utilisation des protocoles définis.

Tableau 5 – Protocoles du plan d'utilisateur supportés	Couche
couche Réseau selon la Recommandation T.90	3
ISO/CEI 8208	3
couche Réseau selon la Recommandation T.70 NL	3
couche 3 vide avec accès au protocole ISO/CEI 7776 par la couche 2	2
couche 3 vide avec accès transparent à la mise en trames HDLC	2
couche 3 vide avec accès transparent à la mise en trames HDLC avec indication d'erreur	2
PPP (protocole point à point)	2
SDLC (protocole de liaison de données synchrone)	2
V.110 asynchrone (Note)	2
V.110 synchrone (Note)	2
accès transparent aux canaux B avec mise en trames des octets par le réseau	1
NOTE – Un accès selon la Recommandation V.110 est offert à un PUF par la couche 2 mais il y a d'autres moyens d'utiliser ce protocole.	

6.3.2 Sélection des protocoles

La sélection des protocoles est effectuée au cours de la création de l'objet NCO, au moyen de deux paramètres: NCOType et UProtocol (voir en [3] la description de la fonction ACreateNCOReq).

6.3.2.1 Paramètre NCOType

Description: ce paramètre sert à transmettre au dispositif NAF le type d'objet de connexion réseau

Type: 50.

Champ	Type de champ	Sens	Requis	Commentaire
identifier	octet	P	M	C (1) – accès à la connexion sémaphore seulement (Note) U3 (2) – accès utilisateur à la couche Réseau avec coordination sémaphore par NAF (fonction de coordination NAF) C/U (3) – accès utilisateur à la connexion sémaphore et à la couche Réseau, à la couche Liaison ou à la couche Physique. U3/G (4) – accès utilisateur à la couche Réseau pour des circuits virtuels additionnels. Ce NCO doit être groupé avec un NCO de type U3 ou C/U déjà créé.
NOTE – Les objets NCO de type C sont hors du domaine d'application des Parties 3 à 6.				

6.3.2.2 Paramètre UProtocol

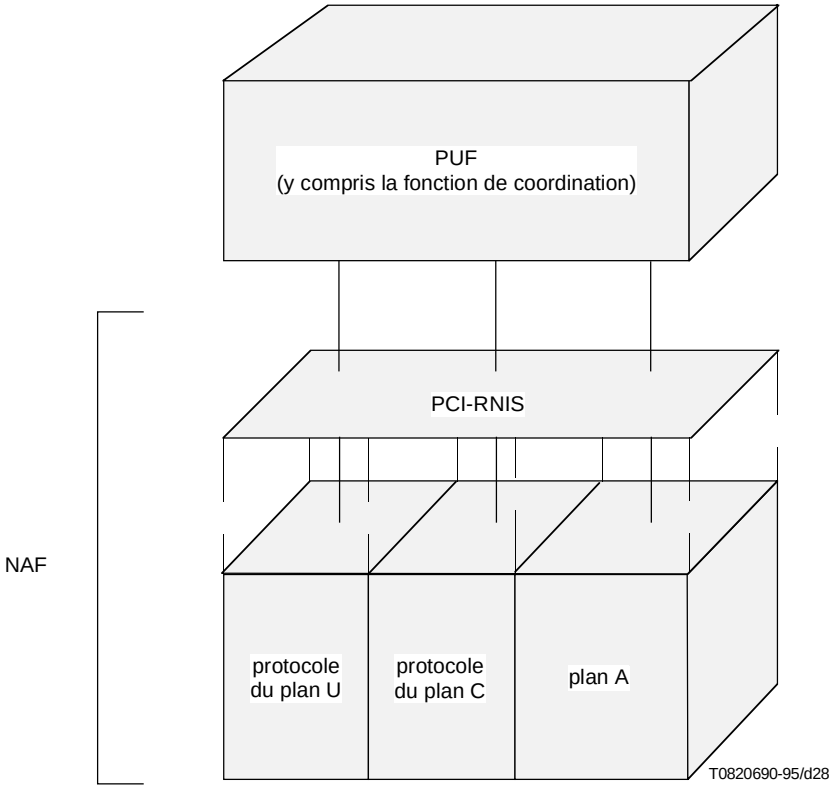
Description: ce paramètre sert à sélectionner le protocole du plan utilisateur. Si sa longueur a la valeur 3, le premier octet indique le protocole de couche 3 demandé, le deuxième octet indique le protocole de couche 2 demandé et le troisième octet indique le protocole de couche 1 demandé.

Type: 62.

Champ	Type de champ	Sens	Requis	Commentaire
L3Protocol	octet	P	M	défaut (255) – T.90 T.90 (1) ISO/CEI 8208 (2) T.70 NL (3) NULL (4)
L2Protocol	octet	P	C (Note 1)	défaut (255) – ISO/CEI 7776 ISO/CEI 7776 (1) transparence en mode trame (2) transparence en mode trame avec indication d'erreur (3) PPP (4) SDLC (5) V.110 asynchrone (6) V.110 synchrone (7) NULL (8)
L1Protocol	octet	P	C (Note 2)	défaut (255) – accès transparent avec mise en trame des octets par le réseau accès transparent avec mise en trame des octets par le réseau (1)
NOTE 1 – Obligatoire si le champ L3Protocol a la valeur NULL. NOTE 2 – Obligatoire si les champs L3Protocol et L2Protocol ont la valeur NULL.				

6.4 Fonction de coordination

L'interface PCI-RNIS donne directement accès à la connexion sémaphore et à la connexion d'utilisateur, en association avec les canaux D et B du RNIS. Un dispositif PUF utilisant cette méthode doit effectuer l'établissement d'une connexion d'utilisateur au moyen de la commande d'appel de base offerte par le plan de commande. La coordination entre connexions sémaphores et d'utilisateur n'est assurée que par le dispositif PUF. La Figure 2 montre la fonction de coordination assurée par le dispositif PUF. C'est grâce à la commande des connexions sémaphores, que le dispositif PUF peut faire appel aux services complémentaires.



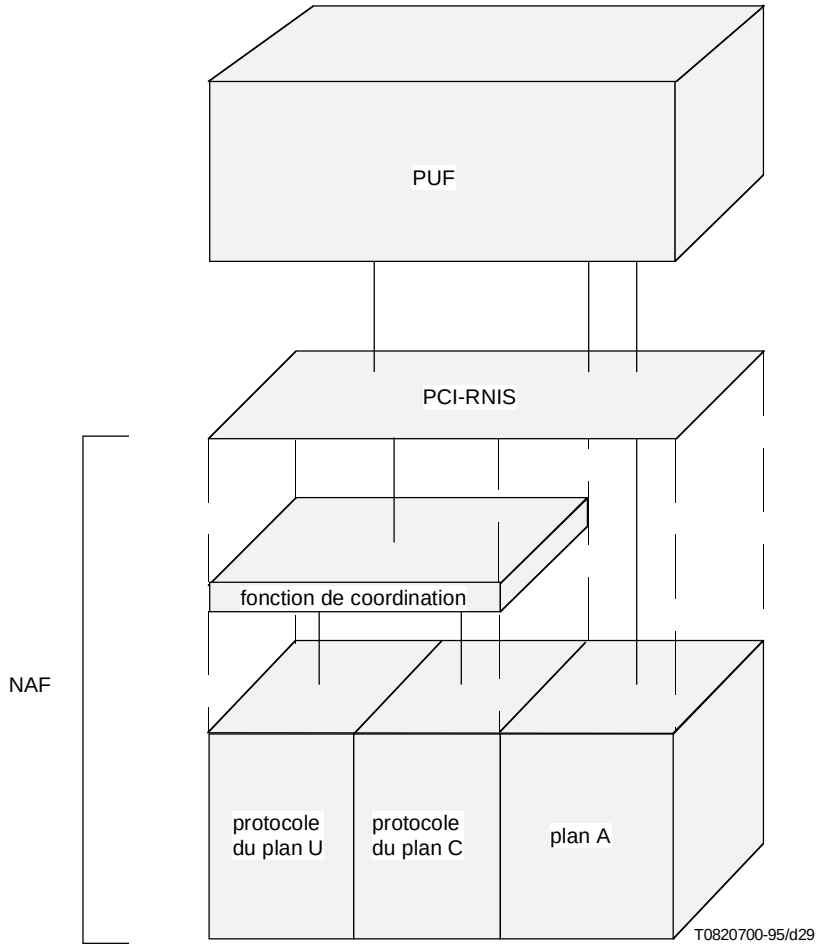
NOTE – L'existence d'une fonction de coordination à l'intérieur du dispositif PUF est hors du domaine d'application de la présente partie.

Figure 2 – Coordination par dispositif PUF

Le dispositif PUF peut recevoir un service réseau en mode connexion ISO (CONS, *connection-oriented network service*) tel que défini dans la Recommandation X.213. Cette abstraction est gérée par une fonction de coordination qui mappe les primitives du service CONS dans le plan U en correspondance avec les primitives des protocoles des plans C et U. Cette fonction de coordination ne peut être utilisée qu'avec les protocoles de plan U conformes à la Recommandation X.213. La fonction de coordination est assurée dans le cadre d'un dispositif NAF. Etant donné que celui-ci gère la coordination entre connexions sémaphores et utilisateur, le dispositif PUF ne doit pas accéder au plan C. La Figure 3 montre la fonction de coordination assurée par le dispositif NAF.

La fonction de coordination n'a pas d'incidence sur le plan d'administration.

Même si la fonction de coordination est utilisée par le dispositif PUF, les protocoles de couche 2 et de couche 3 utilisés sont les protocoles sélectionnés pour les accès à la couche Réseau et à la couche Liaison de données.



NOTE – La fonction de coordination n'est définie que pour les protocoles du plan U conformes à la Recommandation X.213.

Figure 3 – NAF coordination

Pour établir une connexion dont la coordination sera assurée par le dispositif NAF, le dispositif PUF transmet le message suivant:

ACreateNCOReq, pour un objet NCO de type U3, avec les informations applicables.

Une connexion peut alors être demandée au moyen du message UConnectReq. Tous les autres messages du plan U restent utilisables par le dispositif PUF. Aucun message du plan de commande ne peut être utilisé en combinaison avec un objet NCO de type U3. Pour les diagrammes de transition d'état, voir les Parties 4 à 6 [4], [5] et [6].

La fonction de coordination peut ne pas être disponible pour tous les protocoles du plan U: sa disponibilité sera donc indiquée dans chaque partie applicable à un protocole d'utilisateur.

6.5 Critères de sélection

6.5.1 Sélection d'objets NCO – Eléments d'information dans le plan U

Afin d'appliquer l'objet NCO correct lors d'un appel entrant, le dispositif NAF fait appel à divers critères de sélection, dont le mécanisme et les éléments d'information dans le plan C sont décrits dans la Partie 2 [3].

Certains éléments du plan U peuvent également s'appliquer lors de la sélection d'un objet NCO. Les éléments utiles sont spécifiques du protocole utilisé. Par exemple, dans le cas du protocole ISO/CEI 8208, les éléments d'information du plan U sont les suivants:

- négociation de la longueur des paquets;
- négociation de la longueur des fenêtres.

Les autres Parties 4 à 6 décrivent les éléments d'information à utiliser dans le plan U.

6.5.2 Action en cas d'indisponibilité d'objet NCO – Appel entrant dans le plan U

Le dispositif NAF émet un message de déconnexion avec la cause spécifique au protocole. La cause exacte est précisée dans le sous-paragraphe relatif à chaque protocole.

6.6 Contrôle d'erreur dans le plan d'utilisateur

Une information d'erreur de message A est retournée dans le message d'erreur du plan A. Pour plus de détails sur le traitement des erreurs de message, voir la Partie 2 [3].

La détection des erreurs protocolaires intervient après le contrôle administratif et le mécanisme utilisé pour renvoyer l'erreur dépend du protocole. Ces mécanismes sont décrits dans les sous-paragraphes décrivant chaque protocole.

Un paramètre à contenu non valide et des données d'utilisateur de longueur non valide sont considérés comme des erreurs protocolaires.

6.7 Ensembles d'attributs du plan d'utilisateur

Les ensembles d'attributs servent à grouper des paramètres importants pour l'exécution de protocoles d'utilisation. Pour ce plan, il existe un certain nombre d'ensembles d'attributs, qui sont définis dans chaque sous-paragraphe ([4], [5] et [6]) applicable aux protocoles.

Appendice I

Directives pour le développement d'un dispositif NAF

Le corps principal de la présente partie contient la description de l'interface PCI-RNIS, vue du dispositif PUF. Conformément à cette approche, certains points, non directement associés au PUF mais ayant une incidence sur le développement du NAF, ne sont pas décrits. Ces points peuvent être utiles pour le développement du NAF: ils seront donc décrits dans le présent appendice avec des directives pour le développement des NAF conformément au corps de la présente partie.

Le présent appendice tient compte des trois points de vue suivants:

- le présent appendice indique des points additionnels. Le NAF doit être implémenté sur la base de la présente partie. Il y a lieu qu'il implémente l'interface PCI-RNIS de manière que les propriétés décrites soient fournies;
- il convient de donner priorité au corps principal de la présente partie en cas d'incertitude sur un point du présent appendice ou de divergence d'interprétation entre le corps de la partie et du présent appendice;
- le présent appendice ne vise pas à imposer de quelconques contraintes quant à l'implémentation d'un dispositif NAF. L'objectif est de donner des directives sur la façon d'élaborer un NAF conformément à la présente partie.

I.1 Gestion des erreurs dans le plan d'utilisateur

Le traitement des erreurs pour ce plan est défini pour chaque protocole dans la Partie ([4], [5] et [6]) correspondant à ce protocole.

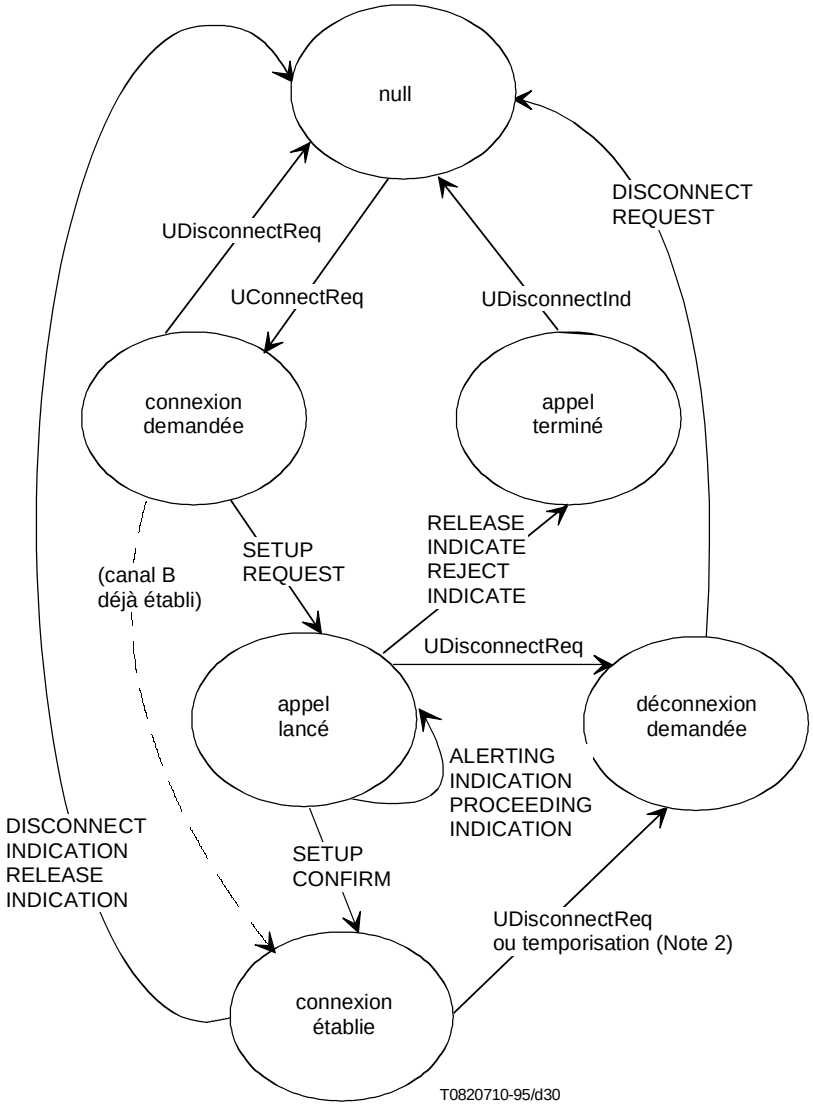
I.2 Configuration d'un dispositif NAF

La configuration globale, la configuration du système et la configuration du plan de commande sont décrites dans la Partie 2.

La configuration du plan d'utilisateur est décrite dans les Parties 4 à 6 [4], [5] et [6], selon le protocole.

I.3 Fonction de coordination pour un appel sortant du plan d'utilisateur

La Figure I.1 montre l'établissement d'une connexion du plan de commande. Les états indiqués sont internes au dispositif NAF.



NOTE 1

– Les événements indiqués en capitales sont des primitives décrites dans la Recommandation Q.931;

– Les événements indiqués en lettres mixtes sont des messages d'interface PCI dans le plan U;

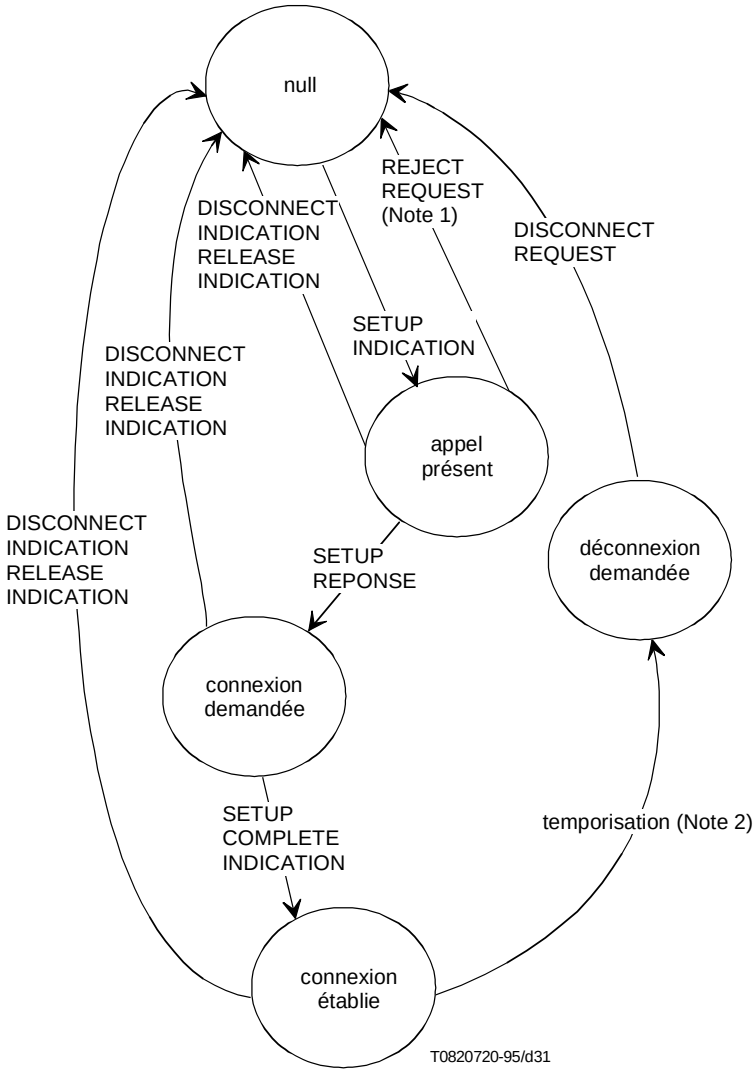
– Les états indiqués sont internes au dispositif NAF.

NOTE 2 – La déconnexion – immédiatement ou sur temporisation – de la liaison RNIS par le NAF à la suite de la déconnexion de la dernière liaison dans le plan U relève de la conception du dispositif NAF.

Figure I.1 – Fonction de coordination d'un appel entrant et établissement d'un canal

I.4 Fonction de coordination – Appel RNIS entrant

Le diagramme de transition d'état ci-dessous (voir la Figure I.2) montre l'établissement de la connexion dans le plan de commande. Les états indiqués sont internes au dispositif NAF.



NOTE 1

– Les événements indiqués en lettres capitales sont des primitives décrites dans la Recommandation Q.931;

– Les états indiqués sont internes au dispositif NAF.

NOTE 2 – La déconnexion – immédiatement ou sur temporisation – de la liaison RNIS par le NAF à la suite de la déconnexion de la dernière liaison dans le plan U relève de la conception du dispositif NAF.

NOTE 3 – Le dispositif NAF peut rejeter la demande de connexion RNIS.

Figure I.2 – Appel RNIS entrant et fonction de coordination

Remplacée par une version plus récente

Appendice II

Protocoles d'utilisateur

Le présent appendice résume les informations essentielles pour l'emploi des protocoles du plan U. Les valeurs entre parenthèses sont codées en décimal.

Protocole	Type de NCO	Protocole U	Sens U	Fonction de coordination
accès transparent	. C/U	. NULL (4) . NULL (8) . accès transparent (1)	«les deux» ou absent	non
V.110	. C/U	. NULL (4) . V.110 (6 ou 7) . (...)	«les deux» ou absent	non
PPP	. C/U	. NULL (4) . PPP (4) . (...)	«les deux» ou absent	non
SDLC	. C/U	. NULL (4) . SDLC (5) . (...)	«les deux» ou absent	non
HDLC	. C/U	. NULL (4) . HDLC (2 ou 3) . (...)	«les deux» ou absent	non
ISO/CEI 7776	. C/U	. NULL (4) . ISO/CEI 7776 (1) . (...)	«les deux» ou absent	non
T.70	. C/U	. T.70 (3) . / . (...)	«les deux» ou absent	non
ISO/CEI 8208	. C/U . U3 . U3/G	. ISO/CEI 8208 (2) . (...) . (...)	utilisé	oui
T.90	. C/U . U3 . U3/G	. T.90 (1) . ISO/CEI 7776 (1) . (...)	utilisé	oui
NOTE – Les points de suspension (...) signifient qu'il n'y a pas de valeur fixée.				

Remplacée par une version plus récente

TABLE DES MATIÈRES

PARTIE 4

PARTIE 4: PROTOCOLES DE COUCHE UN

Résumé

La présente partie de la spécification décrit les procédures, messages et paramètres utilisés pour accéder aux protocoles du plan d'utilisateur qui assurent un service de communication dans la couche 1.

Introduction

L'utilisation de différentes interfaces de programmation par des équipements terminaux connectés au réseau numérique à intégration de services (RNIS) a fait obstacle au développement d'applications communes utilisant le RNIS et a limité le déploiement d'applications RNIS sur des équipements terminaux tels que les ordinateurs personnels.

La présente interface d'application (API, *application programming interface*), appelée interface de programmation de communication (PCI, *programming communication interface*) par réseau numérique à intégration des services (PCI-RNIS), établie par l'UIT-T, permet aux applications d'accéder aux services des RNIS et de les gérer. L'interface PCI-RNIS fait l'objet d'une série de spécifications, dont la présente partie constitue la description de l'usage des protocoles de couche 1.

L'interface PCI-RNIS a été définie de façon à offrir aux fournisseurs d'équipement terminal une norme qui permettra de réaliser la portabilité d'applications utilisant l'interface PCI-RNIS sur une gamme d'équipements terminaux tournant sur différents systèmes d'exploitation.

L'interface PCI-RNIS a été définie en fonction des besoins des développeurs d'applications et, dans la mesure du possible, élimine la nécessité d'une connaissance approfondie du RNIS. Elle a également été conçue de façon que les futures extensions du RNIS n'aient pas d'incidence sur le fonctionnement des applications existantes.

Remplacée par une version plus récente

1 Domaine d'application

La présente partie décrit les protocoles de couche 1 offerts par l'interface de programmation de communication pour réseau numérique à intégration de services (PCI-RNIS). Il fait partie de la spécification PCI-RNIS.

Ce projet décrit les éléments spécifiques (messages, paramètres, ensembles d'attributs, etc.) qui se rapportent aux protocoles du plan U dans la couche 1. La présente partie traite du protocole d'accès transparent à un canal B par le plan d'utilisateur. La description d'autres protocoles utilisateurs de couche 1 fera l'objet de futures Recommandations UIT-T.

2 Références

[1] Partie 1, *Architecture générale.*

[2] Partie 2, *Services de base.*

[3] Partie 3, *Architecture de gestion des protocoles du plan d'utilisateur.*

3 Définitions

La présente partie définit les termes suivants:

3.1 ensemble d'attributs: ensemble de paramètres nécessaires au fonctionnement des protocoles d'utilisation et à la signalisation RNIS.

3.2 canal B: voie logique RNIS utilisée pour le transfert de données.

3.3 plan de commande: groupement logique de fonctions pour l'accès à la signalisation RNIS.

3.4 canal D: voie logique RNIS utilisée pour la signalisation et, dans certains cas, pour le transfert de données.

3.5 accès RNIS: ensemble de canaux RNIS fourni par un même dispositif d'accès réseau (NAF) pour l'accès aux services RNIS.

3.6 interface RNIS de programmation de communication (PCI-RNIS): interface logicielle orientée RNIS qui offre des possibilités de programmer la signalisation de réseau et l'échange de données d'utilisateur.

3.7 message: unité d'information transférée de part et d'autre de l'interface PCI-RNIS, entre le dispositif d'accès réseau (NAF) et le dispositif utilisateur d'interface PCI (PUF).

3.8 dispositif d'accès réseau (NAF): unité fonctionnelle située entre l'interface PCI-RNIS et les couches associées au réseau.

3.9 objet de connexion au réseau (NCO): objet abstrait contenu dans le dispositif NAF, qui est créé par le dispositif PUF pour donner accès à la signalisation ou aux données du réseau.

3.10 couche vide: couche vide dans le modèle de référence OSI. Une telle couche ne contient aucune fonction et transmet en transparence les requêtes et les réponses aux couches adjacentes.

3.11 dispositif utilisateur d'interface PCI (PUF): unité fonctionnelle faisant appel à l'interface PCI-RNIS pour accéder à un dispositif NAF. Ce terme correspond par exemple à l'application locale qui utilise l'interface.

3.12 connexion d'utilisateur: connexion accessible par l'intermédiaire de la propriété plan d'utilisateur.

3.13 plan d'utilisateur: groupement logique de fonctions d'accès offertes aux protocoles et aux données d'utilisateur.

3.14 protocole utilisateur: protocole exploité conformément à la propriété de plan d'utilisateur.

4 Abréviations

La présente partie utilise les abréviations suivantes:

API	interface de programmation d'application (*application programming interface*)
LAPB	procédure d'accès à la liaison en mode équilibré (*link access procedure – balanced*)
LAPD	procédure d'accès à la liaison sur le canal D (*link access procedure for D-channel*)
NAF	dispositif d'accès réseau (*network access facility*)

NCO objet de connexion au réseau (*network connection object*)

PCI interface de programmation de communication (*programming communication interface*)

PUF dispositif utilisateur d'interface PCI (*PCI user facility*)

RNIS réseau numérique à intégration de services

SAP point d'accès au service (*service access point*)

5 Guide de lecture

5.1 Guide du lecteur

La présente partie est destinée aux développeurs de logiciels, aux réalisateurs d'applications et aux constructeurs d'équipement, qui y trouveront la description de l'usage général des protocoles d'utilisateur dans la couche 1.

5.2 Mode d'emploi de la présente partie

Les lecteurs qui:

- ont besoin d'un aperçu général rapide des protocoles du plan d'utilisateur le trouveront dans la Partie 3 [3];
- envisagent d'implémenter une application au moyen d'un protocole d'utilisateur de couche 1 par l'interface PCI-RNIS devront lire la présente spécification, dont le paragraphe 6 traite de l'usage de ce type de protocole;
- ont l'intention de construire une carte ou un équipement d'adaptation au RNIS utilisant l'interface PCI-RNIS pour un protocole de couche 1 devront consulter la présente spécification, dont le paragraphe 6 traite de l'usage de ce type de protocole et dont l'Appendice I (informatif) décrit les valeurs de configuration par défaut.

Le Tableau 1 donne une liste qui décrit le contenu général de la présente partie.

Tableau 1 – Table des matières de la présente Recommandation

Paragraphe, annexe, appendice	Contenu
paragraphe 1	domaine d'application de la présente partie, décrivant son objet
paragraphe 2	références
paragraphe 3	définitions de termes utilisés dans la présente partie
paragraphe 4	définitions d'abréviations utilisées dans la présente partie
paragraphe 5	aperçu général
paragraphe 6	accès transparent
Appendice I	valeurs de configuration par défaut (à titre d'information)

La présente partie donne, pour chaque protocole supporté:

- la description des messages disponibles dans le plan U (voir le paragraphe 2);
- la description des paramètres utiles dans le plan U (voir le paragraphe 3);
- le diagramme de transition d'état (voir le paragraphe 4);
- les informations relatives à la fonction de coordination (voir le paragraphe 5);
- les critères de sélection d'éventuels objets NCO spécifiques (voir le paragraphe 6);
- le traitement et les codes d'erreur spécifique (voir le paragraphe 7);
- la définition d'un ensemble d'attributs (voir le paragraphe 8).

L'Appendice I indique les valeurs par défaut éventuelles d'une configuration de protocole.

6 Accès transparent aux canaux B avec mise en trame des octets par le réseau

6.1 Introduction

Le présent paragraphe traite des protocoles d'accès transparent à un canal B avec mise en trame des octets par le réseau. Le paramètre BearerCap indique si le canal B est exploité à 64 kbit/s ou à 56 kbit/s.

Pour cet accès, le dispositif NAF considère les couches 2 et 3 comme des couches de type vide, comme indiqué dans la Figure 1.

L'emplacement dans le modèle OSI du protocole d'accès transparent à 64 kbit/s est également représenté dans la Figure 1.

La Partie 2 de l'Annexe 2 donne une description générale des conventions utilisées.

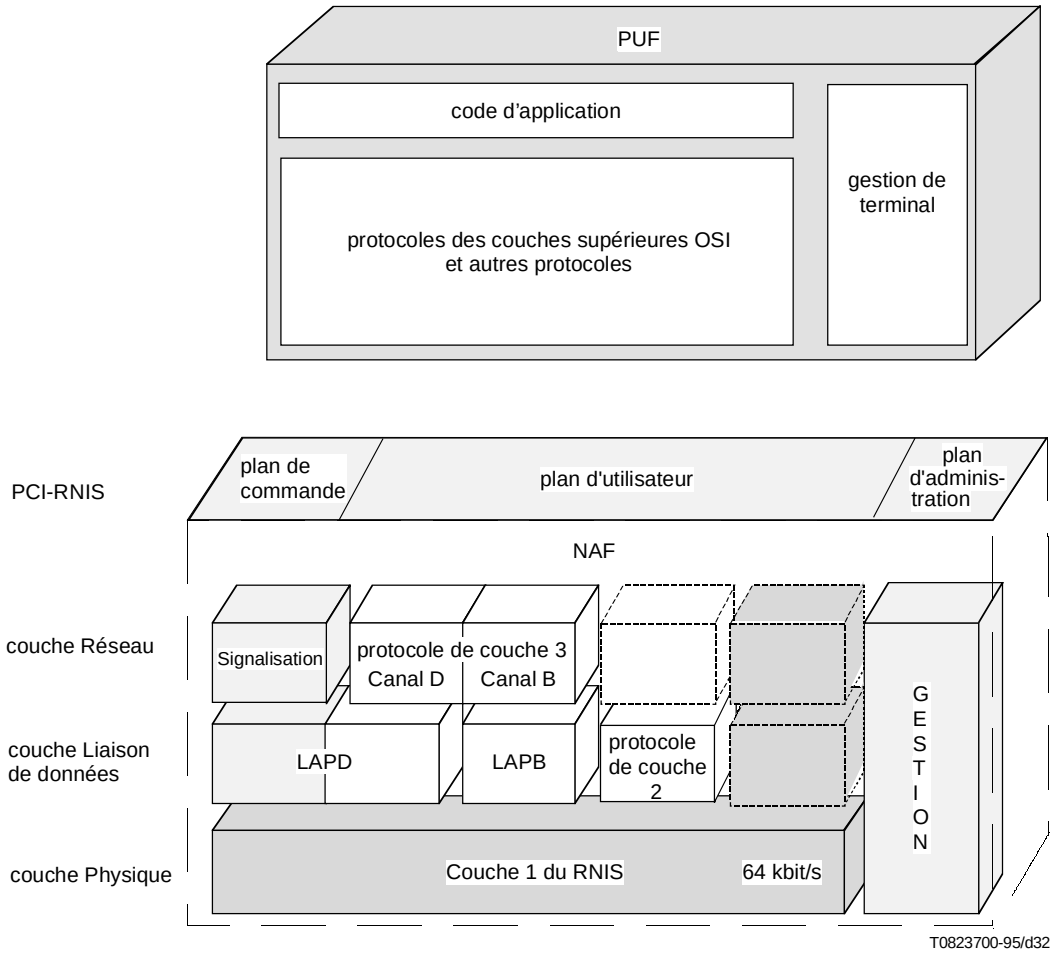


Figure 1 – Emplacement dans le modèle OSI

6.2 Messages

Le Tableau 2 donne un aperçu général des messages du plan U.

Tableau 2 – Aperçu général des messages du plan U

Identificateur du message	Classe	Nom du message	But du message
307	1	UDataReq	demande de transfert de données
308	1	UDataInd	indication d'arrivée de données transférées
319	1	UErrorInd	indication d'erreur

6.2.1 Message UDataReq

Classe: 1 (classe de base)

Description: ce message permet à un dispositif PUF d'envoyer des données *transparentes* sur le canal B. Par défaut, ces données sont envoyées sous forme d'un train d'octets sans aucun protocole. La synchronisation utilisée sur le canal B est en mode caractères. Lorsqu'il n'y a plus de données à traiter, le dispositif NAF envoie l'octet IdleFlag, inséré dans l'ensemble d'attributs utilisé pour cette connexion.

Paramètre:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion du plan de commande

Remarque: les données à envoyer sont obligatoires. Elles ne sont pas codées sous forme de paramètre du message.

Les données obligatoires doivent être envoyées dans le tampon de données

Message associé: néant.

6.2.2 Message UDataInd

Classe: 1 (classe de base)

Description: ce message indique à un dispositif PUF que des données *transparentes* ont été reçues sur le canal B. Ces données sont reçues sous forme d'un train d'octets sans aucun protocole ni message de commande. Le paramètre IdleFlag, fourni comme caractère de bourrage par défaut dans l'ensemble d'attributs, n'est pas extrait des données reçues.

Paramètre:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion du plan de commande

Remarque: les données reçues sont toujours fournies mais elles ne sont pas codées sous forme de paramètre du message.

Les données sont fournies dans le tampon de données. Celui-ci, dans ce cas, est obligatoire.

Message associé: UErrorInd.

6.2.3 Message UErrorInd

Classe: 1 (classe de base)

Description: ce message indique à un dispositif PUF qu'une erreur s'est produite.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion du plan de commande
Cause	M	identifie le type d'erreur

Message associé: néant.

6.3 Paramètres insérés dans les messages

Ce sous-paragraphe décrit les paramètres de ce plan. Le Tableau 3 résume les paramètres utilisés.

Tableau 3 – Aperçu général des paramètres du plan U

Identificateur du paramètre	Nom du paramètre	Utilisé dans des messages du plan U	Utilisé dans un ensemble d'attributs du plan U	Autre usage
35	IdleFlag		X	
50	NCOType			X
62	UProtocol		X	
63	UAttributeName			X
64	UDirection			X
68	Cause	X		

6.3.1 Paramètre IdleFlag

Description: ce paramètre contient l'octet de fanion qui doit être envoyé par le dispositif NAF lorsque l'accès d'utilisateur est à l'état de repos.

Type: 35.

Champ	Type de champ	Sens	Requis	Commentaire
IdleFlag	octet	P	M	octet de fanion

6.3.2 Paramètre NCOType

Description: ce paramètre sert à transmettre au dispositif NAF le type d'objet de connexion réseau.

Type: 50.

Champ	Type de champ	Sens	Requis	Commentaire
Identifier	octet		M	C/U (3) – accès sémaphore et accès utilisateur transparent

6.3.3 Paramètre UProtocol

Description: ce paramètre est utilisé pour sélectionner le protocole de plan U. Le premier octet contient le protocole de couche 3 demandé, le deuxième octet contient le protocole de couche 2 demandé et le troisième octet contient le protocole de couche 1 demandé.

Type: 62.

Champ	Type de champ	Sens	Requis	Commentaire
L3Protocol	octet	P	M	NULL (4)
L2Protocol	octet	P	M	NULL (8)
L1Protocol	octet	P	M	accès transparent au canal B (1)

Remarque: d'autres valeurs (pour d'autres protocoles) sont indiquées en [3].

6.3.4 Paramètre UAttributeName

Description: ce paramètre sert au dispositif PUF à transmettre le nom d'un ensemble statique d'attributs du plan U.

Type: 63.

Champ	Type de champ	Sens	Requis	Commentaire
AttributeName	chaîne IA5	P	M	16 est la longueur maximale

6.3.5 Paramètre UDirection

Description: ce paramètre sert à transmettre au dispositif NAF des informations concernant l'usage d'un objet NCO particulier dans le plan U.

Type: 64.

Champ	Type de champ	Sens	Requis	Commentaire
Direction	octet	P	O	dans les deux sens (3)

6.3.6 Cause

Description: ce paramètre sert à transmettre au dispositif PUF des informations de cause pour la déconnexion.

Type: 68.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	N	M	210 – débordement (de capacité)

6.4 Diagramme de transition d'état

Les messages du plan U ne modifie pas l'état de la connexion.

6.5 Fonction de coordination

La fonction de coordination ne peut pas être utilisée avec un protocole d'accès transparent à un canal B par le plan U.

6.6 Critères de sélection

Aucun paramètre spécifique n'est utilisé. Les critères généraux de sélection d'objet NCO sont indiqués en [2].

6.7 Traitement des erreurs spécifiques

Les erreurs sont traitées comme suit: en cas de débordement de données entrantes, le PUF reçoit le message UErrorInd.

6.8 Attributs statiques

6.8.1 Paramètres de type AttributeSet

Tableau 4 – Paramètres d'ensemble d'attributs du plan U (UAttributeSet)

Paramètre	Requis	Commentaire
IdleFlag	C	octet de fanion à envoyer à l'état de repos. Voir 6.3.1.
UProtocol	O	voir 6.3.3

Remarque: ces paramètres ne peuvent être utilisés que lors de la création d'un objet NCO contenant des informations relevant du plan de commande. Pour plus de détails, voir en [2] le sous-paragraphe "opération ACreateNCO".

Si des paramètres sont omis, les valeurs par défaut décrites dans l'Appendice I doivent être utilisées.

6.8.2 Contenu d'un ensemble d'attributs statiques

nom:	U-TRANSPARENT
protocole U:	64 kbit/s
fanion de repos:	0xFF

Appendice I

Configuration

I.1 Accès transparent à un canal B

Tableau I.1 – Configuration du plan U pour l'accès transparent aux canaux B

Paramètre	Valeur par défaut suggérée	Commentaire
valeur par défaut du fanion de repos	0xFF	

Remplacée par une version plus récente

TABLE DES MATIÈRES

PARTIE 5

PARTIE 5: PROTOCOLES DE COUCHE DEUX

Résumé

La présente partie de la spécification décrit les procédures, messages et paramètres utilisés pour accéder aux protocoles du plan d'utilisateur qui assurent un service de communication dans la couche 2.

Introduction

L'utilisation de différentes interfaces de programmation par des équipements terminaux connectés au réseau numérique à intégration de services (RNIS) a fait obstacle au développement d'applications communes utilisant le RNIS et a limité le déploiement d'applications RNIS sur des équipements terminaux tels que les ordinateurs personnels.

La présente interface entre programmes d'application (API, *application programming interface*), appelée interface de programmation de communication (PCI, *programming communication interface*) par réseau numérique à intégration de services (PCI-RNIS), établie par l'UIT-T, permet aux applications d'accéder aux services des RNIS et de les gérer. L'interface PCI-RNIS fait l'objet d'une série de spécifications, dont la présente partie constitue la description de l'usage des protocoles de couche 2.

L'interface PCI-RNIS a été définie de façon à offrir aux fournisseurs d'équipement terminal une norme qui permettra de réaliser la portabilité d'applications utilisant l'interface PCI-RNIS sur une gamme d'équipements terminaux tournant sur différents systèmes d'exploitation.

L'interface PCI-RNIS a été définie en fonction des besoins des développeurs d'application et, dans la mesure du possible, élimine la nécessité d'une connaissance approfondie du RNIS. Elle a également été conçue de façon que les futures extensions du RNIS n'aient pas d'incidence sur le fonctionnement des applications existantes.

1 Domaine d'application

La Partie 5 décrit les protocoles de couche 2 offerts par l'interface de programmation de communication pour réseau numérique à intégration de services (PCI-RNIS). Il fait partie de la spécification PCI-RNIS.

Ce projet décrit les éléments spécifiques (messages, paramètres, ensembles d'attributs, etc.) qui se rapportent aux protocoles du plan U dans la couche 2. La présente partie traite des protocoles suivants du plan d'utilisateur: ISO/CEI 7776, HDLC avec ou sans indication d'erreur, PPP, SDLC et V.110. La description d'autres protocoles utilisateurs de couche 2 fera l'objet d'études ultérieures.

2 Références

[1] Partie 1, *Architecture générale.*

[2] Partie 2, *Services de base.*

[3] Partie 3, *Architecture de gestion des protocoles du plan d'utilisateur.*

[4] ISO/CEI 7776:1995, *Technologies de l'information – Télécommunications et échange d'informations entre systèmes – Procédures de commande à haut niveau – Description des procédures de liaison de données ETTD compatibles X.25 LAPB.*

[5] Recommandation UIT-T V.110 (1996), *Connexion au réseau numérique à intégration de services d'équipements terminaux de traitement de données munis d'interfaces du type défini dans les Recommandations de la série V.*

3 Définitions

La présente partie définit les termes suivants:

3.1 ensemble d'attributs: ensemble de paramètres nécessaires au fonctionnement des protocoles d'utilisation et à la signalisation RNIS.

3.2 canal B: voie logique RNIS utilisée pour le transfert de données.

3.3 plan de commande: groupement logique de fonctions pour l'accès à la signalisation RNIS.

3.4 canal D: voie logique RNIS utilisée pour la signalisation et, dans certains cas, pour le transfert de données.

3.5 accès RNIS: ensemble de canaux RNIS fourni par un même dispositif d'accès réseau (NAF) pour l'accès aux services RNIS.

3.6 interface RNIS de programmation de communication (PCI-RNIS): interface logicielle orientée réseau (RNIS) qui offre des possibilités d'accès pour programmation de la signalisation réseau et du transfert de données d'utilisateur.

3.7 message: unité d'information transférée de part et d'autre de l'interface PCI-RNIS, entre le dispositif d'accès réseau (NAF) et le dispositif utilisateur d'interface PCI (PUF).

3.8 dispositif d'accès réseau (NAF): unité fonctionnelle située entre l'interface PCI-RNIS et les couches associées au réseau.

3.9 objet de connexion (au) réseau (NCO): objet abstrait contenu dans le dispositif NAF, qui doit être créé par le dispositif PUF pour donner accès à la signalisation ou aux données du réseau.

3.10 couche vide: couche vide dans le modèle de référence OSI. Une telle couche ne contient aucune fonction et transmet en transparence les requêtes et les réponses aux couches suivantes.

3.11 dispositif utilisateur d'interface PCI (PUF): unité fonctionnelle faisant appel à l'interface PCI-RNIS pour accéder à un dispositif NAF. Ce terme correspond pratiquement à l'application locale qui utilise l'interface.

3.12 connexion d'utilisateur: connexion accessible par l'intermédiaire de la propriété plan d'utilisateur.

3.13 plan d'utilisateur: groupement logique de fonctions d'accès offertes aux protocoles et aux données d'utilisateur.

3.14 protocole utilisateur: protocole exploité conformément à la propriété de plan d'utilisateur.

4 Abréviations

La présente partie utilise les abréviations suivantes:

API	interface de programmation d'application (*application programming interface*)
LAPB	procédure d'accès à la liaison en mode équilibré (*link access procedure balanced*)
LAPD	procédure d'accès à la liaison sur le canal D (*link access procedure for D-channel*)
NAF	dispositif d'accès réseau (*network access facility*)
NCO	objet de connexion au réseau (*network connection object*)
PCI	interface de programmation de communication (*programming communication interface*)
PUF	dispositif utilisateur d'interface PCI (*PCI user facility*)
RNIS	réseau numérique à intégration de services

5 Guide de lecture

5.1 Guide du lecteur

La présente partie est destinée aux développeurs de logiciels, aux réalisateurs d'applications et aux constructeurs d'équipement, qui y trouveront la description de l'usage général des protocoles d'utilisateur dans la couche 2.

5.2 Mode d'emploi de la présente partie

Les lecteurs qui:

- ont besoin d'un aperçu général rapide des protocoles du plan d'utilisateur le trouveront dans la Partie 3;
- envisagent d'implémenter une application au moyen d'un protocole d'utilisateur de couche 2 par l'interface PCI-RNIS devront lire la présente partie, dont les paragraphes 6 à 11 traitent de l'usage de ce type de protocole;
- ont l'intention de construire une carte ou un équipement d'adaptation au RNIS utilisant l'interface PCI-RNIS pour un protocole de couche 2 devront lire la présente partie, dont les paragraphes 6 à 11 traitent de l'usage de ce type de protocole et dont les Appendices I et II (informatifs) décrivent les valeurs de configuration par défaut ainsi que les diagrammes SDL pour dispositifs NAF.

Le Tableau 1 énumère les principaux paragraphes de la présente partie.

Tableau 1 – Table des matières de la présente partie

Paragraphe, annexe, appendice	Contenu
paragraphe 1	domaine d'application de la présente partie, décrivant son objet
paragraphe 2	références
paragraphe 3	définitions des termes utilisés dans la présente partie
paragraphe 4	définitions des abréviations utilisées dans la présente partie
paragraphe 5	donne un aperçu général
paragraphe 6	protocole ISO/CEI 7776
paragraphe 7	protocole HDLC
paragraphe 8	protocole HDLC avec indication d'erreur
paragraphe 9	protocole PPP
paragraphe 10	protocole SDLC
paragraphe 11	protocole V.110
Appendice I	valeurs de configuration par défaut (pour information)
Appendice II	diagrammes SDL des dispositifs NAF (pour information)

La présente partie donne, pour chaque protocole pris en charge:

- la description des messages disponibles dans le plan U (paragraphe 2);
- la description des paramètres utiles dans le plan U (paragraphe 3);
- le diagramme de transition d'état (paragraphe 4);
- les informations relatives à la fonction de coordination (paragraphe 5);
- les critères de sélection d'éventuels objets NCO spécifiques (paragraphe 6);
- le traitement et les codes d'erreur spécifique (paragraphe 7);
- la définition d'un ensemble d'attributs (paragraphe 8).

L'Appendice I indique les valeurs par défaut éventuelles d'une configuration de protocole.

L'Appendice II montre, sous forme de diagrammes SDL, la plupart des situations d'un dispositif NAF.

6 Protocole ISO/CEI 7776

6.1 Introduction

Le présent paragraphe traite du protocole ISO/CEI 7776.

Le plan d'utilisateur (U) fournit les services nécessaires au protocole ISO/CEI 7776 au moyen d'autres protocoles de ce plan, sur une connexion par canal B. Pour ce type d'accès, le dispositif NAF considère la couche 3 comme étant vide (voir la Figure 1).

La Figure 1 montre comment le protocole ISO/CEI 7776 s'insère dans le modèle OSI.

On trouvera en [2] la description générale des conventions.

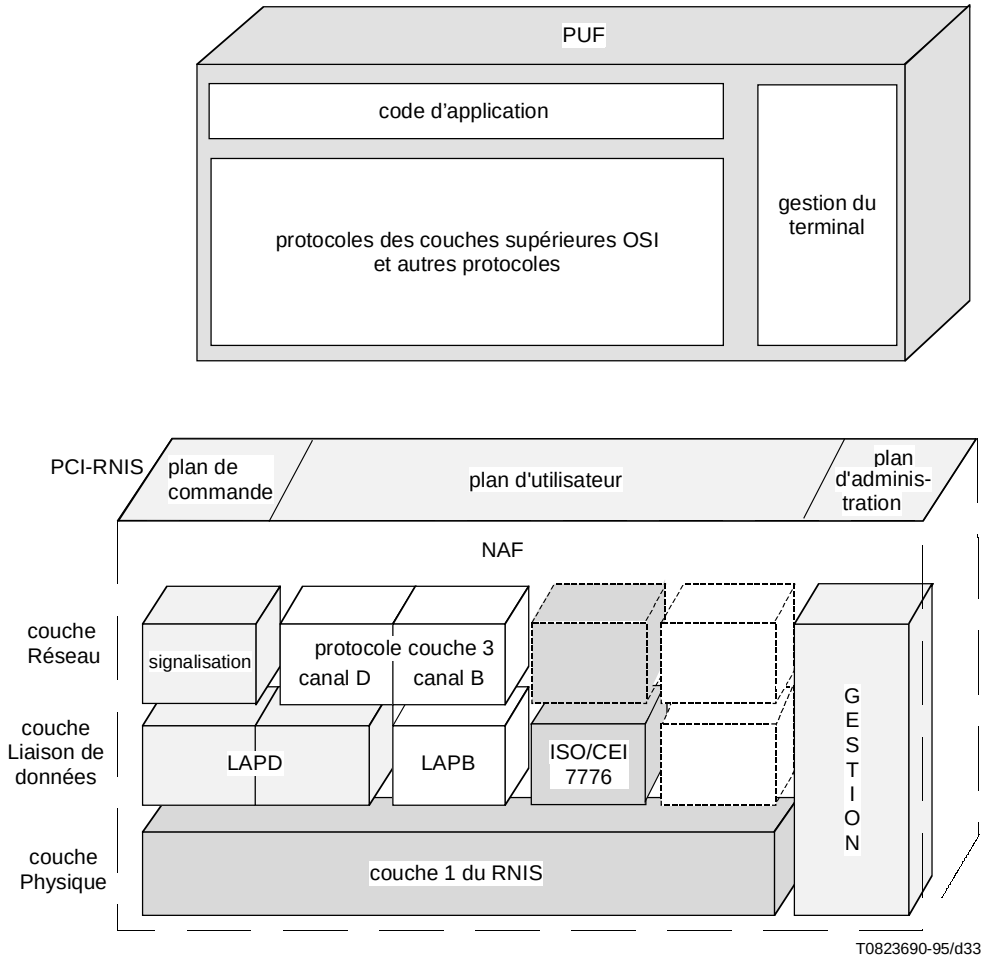


Figure 1 – Emplacement dans le modèle OSI

6.2 Messages

Les messages du plan d'utilisateur donnent accès à des piles protocolaires conformes à l'ISO/CEI 7776. Les paragraphes suivants énumèrent et décrivent brièvement les messages relevant du plan d'utilisateur. Le Tableau 2 donne un aperçu général de ces messages.

Tableau 2 – Apercu général des messages dans le plan U

Identificateur de message	Classe	Nom du message	But du message
301	1	UConnectReq	demande d'établissement d'une connexion dans le plan d'utilisateur
302	1	UConnectInd	indication d'une demande d'établissement de connexion dans le plan d'utilisateur
303	1	UConnectRsp	indication de l'acceptation d'établissement d'une connexion dans le plan d'utilisateur
304	1	UConnectCnf	confirmation de l'établissement d'une connexion dans le plan d'utilisateur
305	1	UDisconnectReq	demande de suppression d'une connexion du plan U
306	1	UDisconnectInd	indication de la suppression d'une connexion du plan U
307	1	UDataReq	demande de transfert de données sur une connexion d'utilisateur établie
308	1	UDataInd	indication de l'arrivée de données transférées sur une connexion d'utilisateur établie
317	1	UReadyToReceiveReq	message effectuant la commande de débit pour une connexion d'utilisateur
318	1	UReadyToReceiveInd	message indiquant le statut de la commande de débit pour une connexion du plan U

6.2.1 UConnectReq

Classe: 1 (classe de base)

Description: ce message permet à un dispositif PUF de demander l'établissement d'une connexion d'utilisateur.

Paramètre:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Message associé: UConnectCnf.

6.2.2 UConnectInd

Classe: 1 (classe de base)

Description: ce message informe un dispositif PUF de l'entrée d'une demande visant à établir une connexion d'utilisateur.

Paramètre:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Message associé: UConnectRsp

6.2.3 UConnectRsp

Classe: 1 (classe de base)

Description: ce message permet à un PUF d'accepter l'établissement d'une connexion d'utilisateur.

Paramètre:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Message associé: UConnectInd.

6.2.4 UConnectCnf

Classe: 1 (classe de base)

Description: ce message informe le PUF de l'établissement d'une connexion d'utilisateur.

Paramètre:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Message associé: UConnectReq.

6.2.5 UDisconnectReq

Classe: 1 (classe de base)

Description: ce message permet à un PUF de supprimer une connexion U.

Paramètre:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Message associé: néant.

6.2.6 UDisconnectInd

Classe: 1 (classe de base)

Description: ce message informe un PUF de la suppression d'une connexion U.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur
Origin	M	identifie l'initiateur de la suppression de la connexion d'utilisateur
Cause	M	identifie la raison de la suppression de la connexion d'utilisateur

Message associé: néant.

6.2.7 UDataReq

Classe: 1 (classe de base)

Description: ce message permet à un PUF d'envoyer un paquet de données, dont la longueur sera limitée à celle qui est définie au moment de la création de l'objet NCO pour les paquets de données dans la couche 2.

Paramètre:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Remarque: les données à envoyer sont obligatoires. Elles ne sont pas envoyées sous forme de paramètres du message.

Les données obligatoires doivent être enregistrées dans le tampon de données.

Message associé: néant.

6.2.8 UDataInd

Classe: 1 (classe de base)

Description: ce message indique à un dispositif PUF la présence de données reçues dans un paquet de données, dont la longueur sera limitée à celle qui est définie au moment de la création de l'objet NCO pour les paquets de données dans la couche 2.

Paramètre:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Remarque: les données reçues sont toujours fournies mais pas sous la forme d'un paramètre de ce message.

Les données sont reçues dans le tampon de données. Celui-ci est alors obligatoire.

Message associé: néant.

6.3 Paramètres contenus dans les messages

Le présent sous-paragraphe décrit les paramètres utilisés dans le protocole ISO/CEI 7776. Le Tableau 3 résume ces paramètres, dans l'ordre alphabétique.

Tableau 3 – Aperçu général des paramètres du plan U

Identificateur du paramètre	Nom du paramètre	Usage dans messages du plan U	Usage dans ensemble d'attributs U	Autre usage
38	L2ConnectionMode		X	
40	L2WindowSize		X	
41	L2XID		X	
50	NCOType			X
62	UProtocol		X	
63	UAttributeName			X
64	UDirection			X
68	Cause	X		
69	Origin	X		

6.3.1 L2ConnectionMode

Description: ce paramètre n'est utilisé que s'il n'est pas défini dans le champ de valeur du paramètre L2XID. Il sert à transmettre au NAF des détails relatifs au mode de connexion dans la couche.

Type: 38.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	P	M	dte (1) – l'ETTD remplit le rôle de station secondaire pour la liaison (non négociable) dce (2) – l'ETCD remplit le rôle de station primaire pour la liaison (non négociable) auto (3) – le rôle de station pour la liaison est négociable par échange d'identificateurs XID

6.3.2 L2WindowSize

Description: ce paramètre n'est utilisé que s'il n'est pas défini dans le champ de valeur du paramètre L2XID. Il sert à transmettre au NAF des détails sur la longueur de fenêtre de couche 2 (paramètre L2WindowSize).

Type: 40.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	P	M	longueur de fenêtre

6.3.3 L2XID

Description: ce paramètre est utilisé pour transmettre des détails sur la valeur du paramètre L2XID et sur son usage. Le champ d'information de l'identificateur XID peut comporter des valeurs ayant priorité sur certains paramètres définis par ailleurs.

Type: 41.

Champ	Type de champ	Sens	Requis	Commentaire
Use	octet	P	M	envoi (1) – envoi du XID. adaptation (2) – adaptation du XID avec le XID reçu. S'il n'y a pas correspondance des XID, la connexion n'est pas établie.
Value	chaîne d'octets	P	M	valeur du XID (identificateur et signature). longueur maximale: 64 octets

6.3.4 NCOType

Description: ce paramètre sert à transmettre au NAF le type d'objet NCO.

Type: 50.

Champ	Type de champ	Sens	Requis	Commentaire
Identifier	octet	P	M	C/U (3) – signalisation et accès utilisateur à la couche Réseau

6.3.5 UProtocol

Description: ce paramètre sert à sélectionner le protocole correct dans le plan d'utilisateur.

Type: 62.

Champ	Type de champ	Sens	Requis	Commentaire
L3Protocol	octet	P	M	NULL (4)
L2Protocol	octet	P	M	ISO/CEI 7776 (0)
L1Protocol	octet	P	O	défaut (255) – accès transparent au canal B

Remarque: d'autres valeurs possibles (pour d'autres protocoles) sont indiquées dans la Partie 2 [2].

6.3.6 UAttributeName

Description: ce paramètre sert au PUF à transmettre le nom d'un ensemble statique d'attributs du plan d'utilisateur.

Type: 63

Champ	Type de champ	Sens	Requis	Commentaire
AttributeName	chaîne IA5	P	M	la longueur maximale est 16 octets

6.3.7 UDirection

Description: ce paramètre sert à transmettre au NAF des informations concernant l'usage d'un objet NCO particulier, pour le plan U.

Type: 64.

Champ	Type de champ	Sens	Requis	Commentaire
sens	octet	P	M	dans les deux sens (3)

6.3.8 Cause

Description: ce paramètre sert à transmettre au PUF des informations de cause pour la déconnexion.

Type: 68.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	N	M	valeur indiquée dans le Tableau 4

6.3.9 Origin

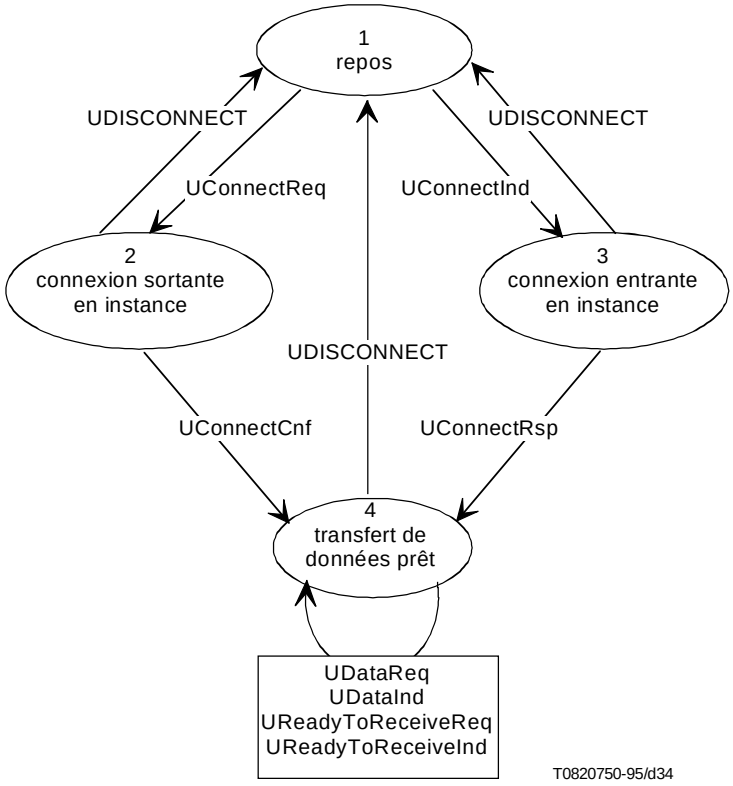
Description: ce paramètre sert à transmettre au PUF les informations relatives à l'origine de la déconnexion.

Type: 69.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	N	M	non définie (1) fournisseur du NAF (2) utilisateur distant (3)

6.4 Diagramme de transition d'état

La Figure 2 montre les différents états par lesquels une connexion du plan U peut passer, sous la forme de messages du plan U présentés dans l'ordre où ils doivent être utilisés.



NOTE – Lorsque le message UDISCONNECT apparaît, il peut prendre la forme UDisconnectReq ou UDisconnectInd.

Figure 2 – Aperçu général des messages du plan U

6.5 Fonction de coordination

La fonction de coordination ne peut pas être utilisée avec les protocoles du plan U faisant appel à l'ISO/CEI 7776.

6.6 Critères de sélection

Aucun paramètre propre au protocole ISO/CEI 7776 n'est utilisé. Les critères généraux de sélection des objets NCO sont indiqués en [2].

6.7 Traitement et codes d'erreur spécifiques

En cas de longueur non valide du paramètre UserData dans le message UDataReq, le dispositif PUF doit recevoir le message UDisconnectInd.

Le Tableau 4 indique les valeurs possibles du paramètre Cause.

Tableau 4 – Valeurs du paramètre Cause

Code de retour		Signification	Information d'erreur spécifique
Undefined	220	situation d'erreur non définie	absente
DiscNorm	241	déconnexion – état normal	absente
InvalidSequence	244	connexion rejetée – séquencement des numéros de trames invalide (état transitoire)	absente
FrameTooBig	245	connexion rejetée – réception d'une trame de longueur supérieure à la valeur indiquée dans le NCO (état fixe)	absente

6.8 Attributs statiques

6.8.1 Paramètres d'ensemble d'attributs

Tableau 5 – Paramètres d'ensemble d'attributs du plan U (UAttributeSet)

Paramètre	Requis	Commentaire
Uprotocol	O	voir la remarque et 6.5
L2ConnectionMode	O	voir la remarque et 6.1
L2WindowSize	O	voir la remarque et 6.2
L2XID	O	voir la remarque et 6.3

Remarque: ces paramètres ne peuvent être utilisés que pendant la création d'un objet NCO contenant des informations sur le plan de commande. Pour plus de détails, voir en [2] le sous-paragraphe "opération ACreateNCO".

Si des paramètres sont omis, le dispositif NAF doit utiliser les valeurs par défaut qui sont décrites dans l'Appendice I.

6.8.2 Contenu d'un ensemble d'attributs

Nom:	U_ISO7776
L2FrameSize:	128
L2WindowSize:	7
L2ConnectionMode:	auto
L2XID:	envoi et correspondance

7 Protocole HDLC

7.1 Introduction

Le présent paragraphe traite du protocole HDLC (commande de liaison de données à haut niveau).

Pour cet accès, le dispositif NAF considère la couche 3 comme vide (NULL), comme indiqué dans la Figure 3.

La Figure 3 montre comment le protocole HDLC s'insère dans le modèle OSI.

On trouvera en [2] la description générale des conventions.

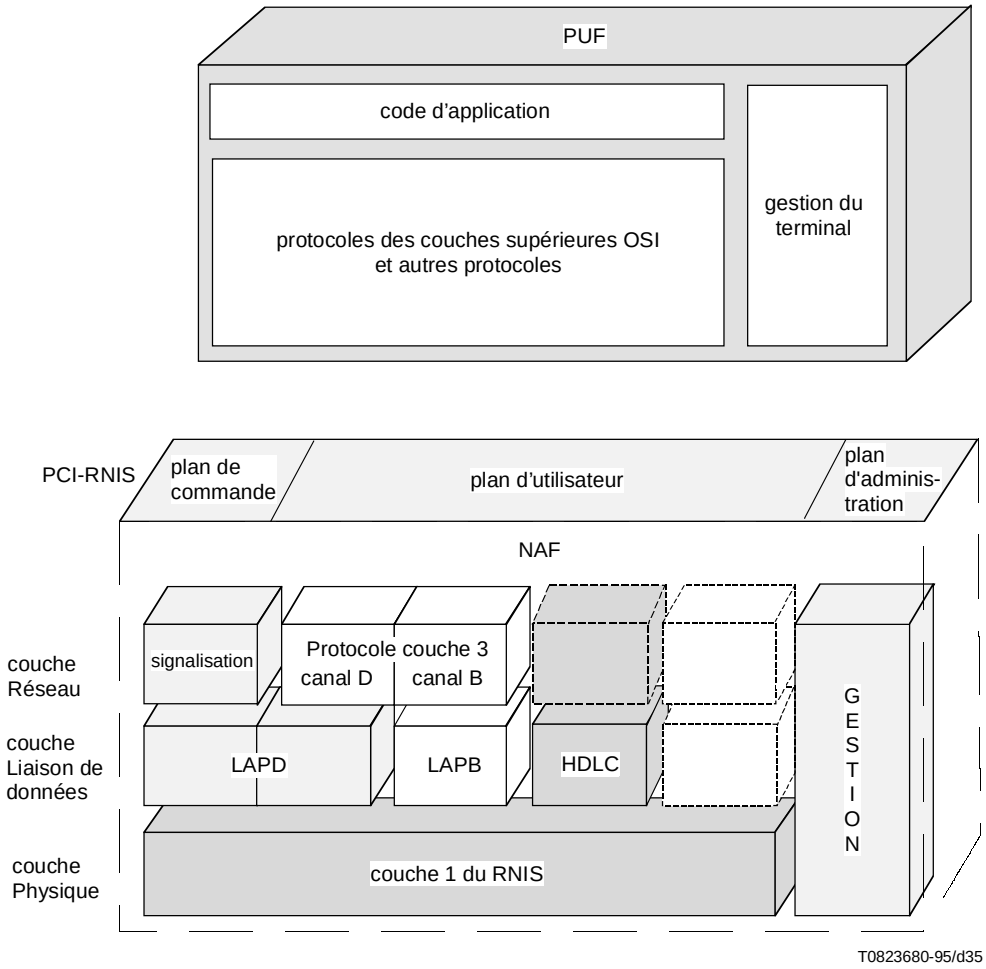


Figure 3 – Emplacement dans le modèle OSI

7.2 Messages

Les messages du plan d'utilisateur donnent accès aux piles protocolaires. Les paragraphes suivants énumèrent et décrivent brièvement les messages relevant du plan d'utilisateur. Le Tableau 6 donne un aperçu général de ces messages.

Tableau 6 – Aperçu général des messages du plan U

Identificateur du message	Classe	Nom du message	But du message
307	1	UDataReq	demande de transfert de données sur une connexion U établie
308	1	UDataInd	indication de l'arrivée de données transférées sur une connexion U établie

7.2.1 UDataReq

Classe: 1 (classe de base)

Description: ce message permet à un dispositif PUF d'envoyer un paquet de données, dont la longueur est limitée par la valeur maximale autorisée par l'interface PCI-RNIS, c'est-à-dire 4096 octets.

Paramètre:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Remarque: les données à envoyer sont obligatoires. Elles ne sont pas envoyées sous forme de paramètres du message.

Les données obligatoires doivent être enregistrées dans le tampon de données.

Message associé: néant.

7.2.2 UDataInd

Classe: 1 (classe de base)

Description: ce message indique à un dispositif PUF la présence de données reçues dans un paquet de données, dont la longueur est limitée par la valeur maximale autorisée par l'interface PCI-RNIS, c'est-à-dire 4096 octets.

Paramètre:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Remarque: les données reçues sont toujours fournies mais pas sous la forme d'un paramètre de ce message.

Les données sont reçues dans le tampon de données. Celui-ci est alors obligatoire.

Message associé: néant.

7.3 Paramètres contenus dans les messages

Le présent sous-paragraphe décrit les paramètres utilisés dans le protocole HDLC. Le Tableau 7 résume ces paramètres, dans l'ordre alphabétique.

Tableau 7 – Aperçu général des paramètres du plan U

Identificateur du paramètre	Nom du paramètre	Usage dans messages du plan U	Usage dans ensemble d'attributs U	Autre usage
50	NCOType			X
62	UProtocol		X	
63	UAttributeName			X
64	UDirection			X

7.3.1 NCOType

Description: ce paramètre sert à transmettre au NAF le type d'objet NCO.

Type: 50.

Champ	Type de champ	Sens	Requis	Commentaire
Identifier	octet		M	C/U (3) – signalisation et accès utilisateur à la couche Réseau

7.3.2 UProtocol

Description: ce paramètre sert à sélectionner le protocole correct dans le plan d'utilisateur.

Type: 62.

Champ	Type de champ	Sens	Requis	Commentaire
L3Protocol	octet	P	M	NULL (4)
L2Protocol	octet	P	M	HDLC (2)
L1Protocol	octet	P	O	défaut (255) – accès transparent au canal B

Remarque: d'autres valeurs possibles (pour d'autres protocoles) sont indiquées en [3].

7.3.3 UAttributeName

Description: ce paramètre sert au PUF à transmettre le nom d'un ensemble statique d'attributs du plan d'utilisateur.

Type: 63.

Champ	Type de champ	Sens	Requis	Commentaire
AttributeName	chaîne IA5	P	M	la longueur maximale est de 16

7.3.4 UDirection

Description: ce paramètre sert à transmettre au NAF des informations concernant l'usage d'un objet NCO particulier, pour le plan U.

Type: 64.

Champ	Type de champ	Sens	Requis	Commentaire
Direction	octet	P	O	dans les deux sens (3)

7.4 Diagramme de transition d'état

Les messages du plan U ne modifient pas l'état de la connexion.

7.5 Fonction de coordination

La fonction de coordination ne peut pas être utilisée avec ce protocole du plan U.

7.6 Critères de sélection

Aucun paramètre spécifique n'est utilisé. Les critères généraux de sélection des objets NCO sont indiqués en [2].

7.7 Traitement et codes d'erreur spécifiques

Le traitement des erreurs n'est pas disponible à l'interface pour ce protocole.

7.8 Attributs statiques

7.8.1 Paramètres d'ensemble d'attributs

Tableau 8 – Paramètres d'ensemble d'attributs du plan U (UAttributeSet)

Paramètre	Requis	Commentaire
Uprotocol	O	voir la remarque et 7.3.2

Remarque: ces paramètres ne peuvent être utilisés que pendant la création d'un objet NCO contenant des informations sur le plan de commande. Pour plus de détails, voir en [2] le sous-paragraphe "opération ACreateNCO".

Si des paramètres sont omis, le dispositif NAF doit utiliser les valeurs par défaut qui sont décrites dans l'Appendice I.

7.8.2 Contenu d'un ensemble d'attributs

Nom:	U_HDLC
UProtocol:	HDLC

8 Protocole HDLC avec indication d'erreur

8.1 Introduction

Le présent paragraphe traite du protocole HDLC (commande de liaison de données à haut niveau) avec indication d'erreur.

Pour cet accès, le dispositif NAF considère la couche 3 comme vide (NULL), comme indiqué dans la Figure 4.

La Figure 4 montre comment ce protocole HDLC s'insère dans le modèle OSI.

On trouvera en [2] la description générale des conventions.

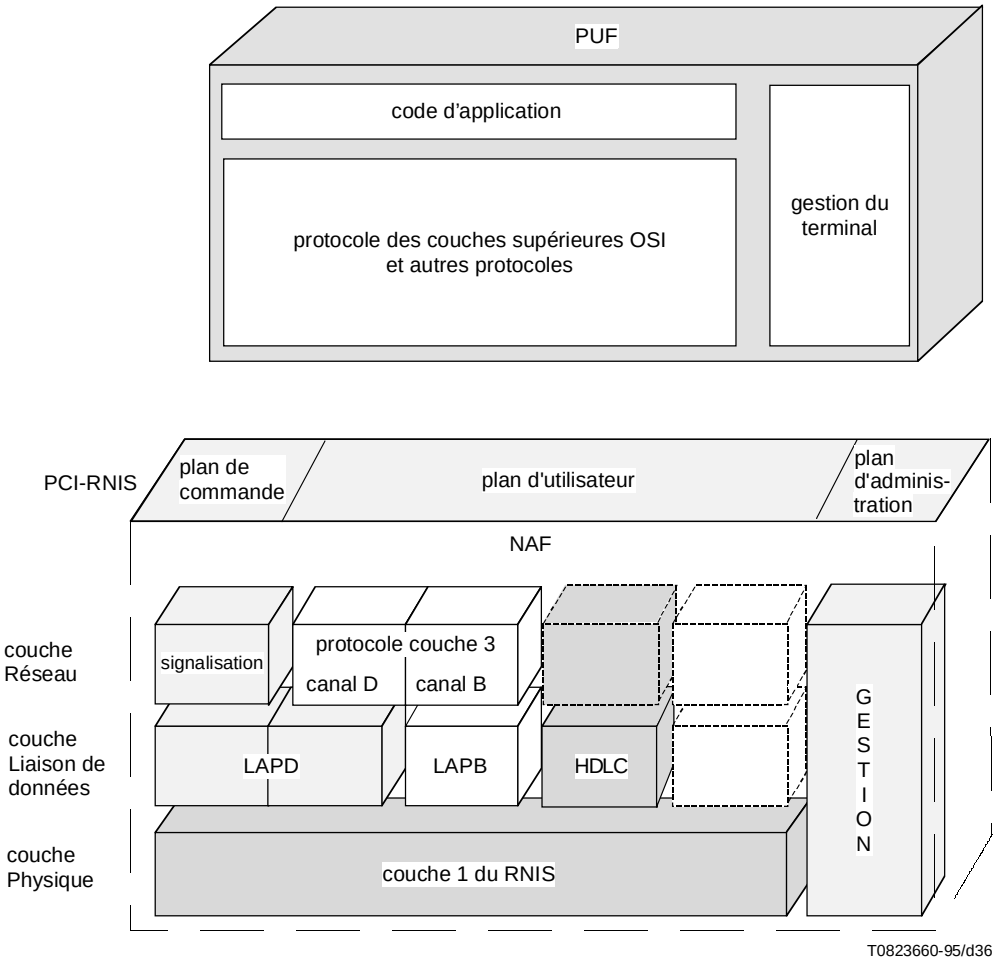


Figure 4 – Emplacement dans le modèle OSI

8.2 Messages

Les messages du plan d'utilisateur donnent accès aux piles protocolaires. Les paragraphes suivants énumèrent et décrivent brièvement les messages relevant du plan d'utilisateur. Le Tableau 9 donne un aperçu général de ces messages.

Tableau 9 – Vue d'ensemble des messages du plan U

Identificateur du message	Classe	Nom du message	But du message
307	1	UDataReq	demande de transfert de données sur une connexion U établie
308	1	UDataInd	indication de l'arrivée de données transférées sur une connexion U établie

8.2.1 UDataReq

Classe: 1 (classe de base)

Description: ce message permet à un dispositif PUF d'envoyer un paquet de données, dont la longueur est limitée par la valeur maximale autorisée par l'interface PCI-RNIS, c'est-à-dire 4096 octets.

Paramètre:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Remarque: les données à envoyer sont obligatoires. Elles ne sont pas envoyées sous forme de paramètres du message.

Les données obligatoires doivent être enregistrées dans le tampon de données.

Message associé: néant.

8.2.2 UDataInd

Classe: 1 (classe de base)

Description: ce message indique à un dispositif PUF la présence de données reçues dans un paquet de données, dont la longueur est limitée par la valeur maximale autorisée par l'interface PCI-RNIS, c'est-à-dire 4096 octets.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur
Cause	O	identifie le type d'erreur

Remarque: les données reçues sont toujours fournies mais pas sous la forme d'un paramètre de ce message.

Les données sont reçues dans le tampon de données. Celui-ci est alors obligatoire.

Message associé: néant.

8.3 Paramètres contenus dans les messages

Le présent sous-paragraphe décrit les paramètres utilisés dans le protocole HDLC avec indication d'erreur. Le Tableau 10 résume ces paramètres, dans l'ordre alphabétique.

Tableau 10 – Aperçu général des paramètres du plan U

Identificateur du paramètre	Nom du paramètre	Usage dans messages du plan U	Usage dans ensemble d'attributs U	Autre usage
50	NCOType			X
62	UProtocol		X	
63	UAttributeName			X
64	UDirection			X
68	Cause	X		

8.3.1 NCOType

Description: ce paramètre sert à transmettre au NAF le type d'objet NCO.

Type: 50.

Champ	Type de champ	Sens	Requis	Commentaire
Identifier	octet		M	C/U (3) – signalisation et accès utilisateur à la couche Réseau

8.3.2 UProtocol

Description: ce paramètre sert à sélectionner le protocole correct dans le plan d'utilisateur.

Type: 62.

Champ	Type de champ	Sens	Requis	Commentaire
L3Protocol	octet	P	M	NULL (4)
L2Protocol	octet	P	M	protocole HDLC avec indication d'erreur (3)
L1Protocol	octet	P	O	défaut (255) – accès transparent au canal B

8.3.3 UAttributeName

Description: ce paramètre sert au PUF à transmettre le nom d'un ensemble statique d'attributs du plan d'utilisateur.

Type: 63.

Champ	Type de champ	Sens	Requis	Commentaire
AttributeName	chaîne IA5	P	M	la longueur maximale est de 16

8.3.4 UDirection

Description: ce paramètre sert à transmettre au NAF des informations concernant l'usage d'un objet NCO particulier, pour le plan U.

Type: 64.

Champ	Type de champ	Sens	Requis	Commentaire
Direction	octet	P	O	dans les deux sens (3)

8.3.5 Cause

Description: ce paramètre sert à transmettre des informations de cause à destination/en provenance du PUF.

Type: 68.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	B	M	210 – débordement 211 – erreur de mise en trame

8.4 Diagramme de transition d'état

Les messages du plan U ne modifient pas l'état de la connexion.

8.5 Fonction de coordination

La fonction de coordination ne peut pas être utilisée avec ce protocole du plan U.

8.6 Critères de sélection

Aucun paramètre spécifique n'est utilisé. Les critères généraux de sélection des objets NCO sont indiqués en [2].

8.7 Traitement et codes d'erreur spécifiques

En cas de longueur non valide du paramètre UserData dans un message UDataReq, le dispositif PUF reçoit le message UDataInd avec le paramètre Cause.

8.8 Attributs statiques

8.8.1 Paramètres de l'ensemble statique d'attributs

Tableau 11 – Paramètres d'ensemble d'attributs du plan U (UAttributeSet)

Paramètre	Requis	Commentaire
UProtocol	O	voir la remarque et 8.3.2

Remarque: ces paramètres ne peuvent être utilisés que pendant la création d'un objet NCO contenant des informations sur le plan de commande. Pour plus de détails, voir en [2] le sous-paragraphe "opération ACreateNCO".

Si des paramètres sont omis, le dispositif NAF doit utiliser les valeurs par défaut qui sont décrites dans l'Appendice I.

8.8.2 Contenu de l'ensemble statique d'attributs

Nom:	U_HDLC_E
UProtocol:	HDLC

9 Protocole PPP

9.1 Introduction

Le présent paragraphe traite du protocole PPP (point à point).

Le plan d'utilisateur (U) fournit les services du protocole PPP au moyen des protocoles de ce plan, sur une connexion par canal B. Pour cet accès, le dispositif NAF considère la couche 3 comme NULL, comme indiqué dans la Figure 5.

La Figure 5 montre comment ce protocole HDLC s'insère dans le modèle OSI.

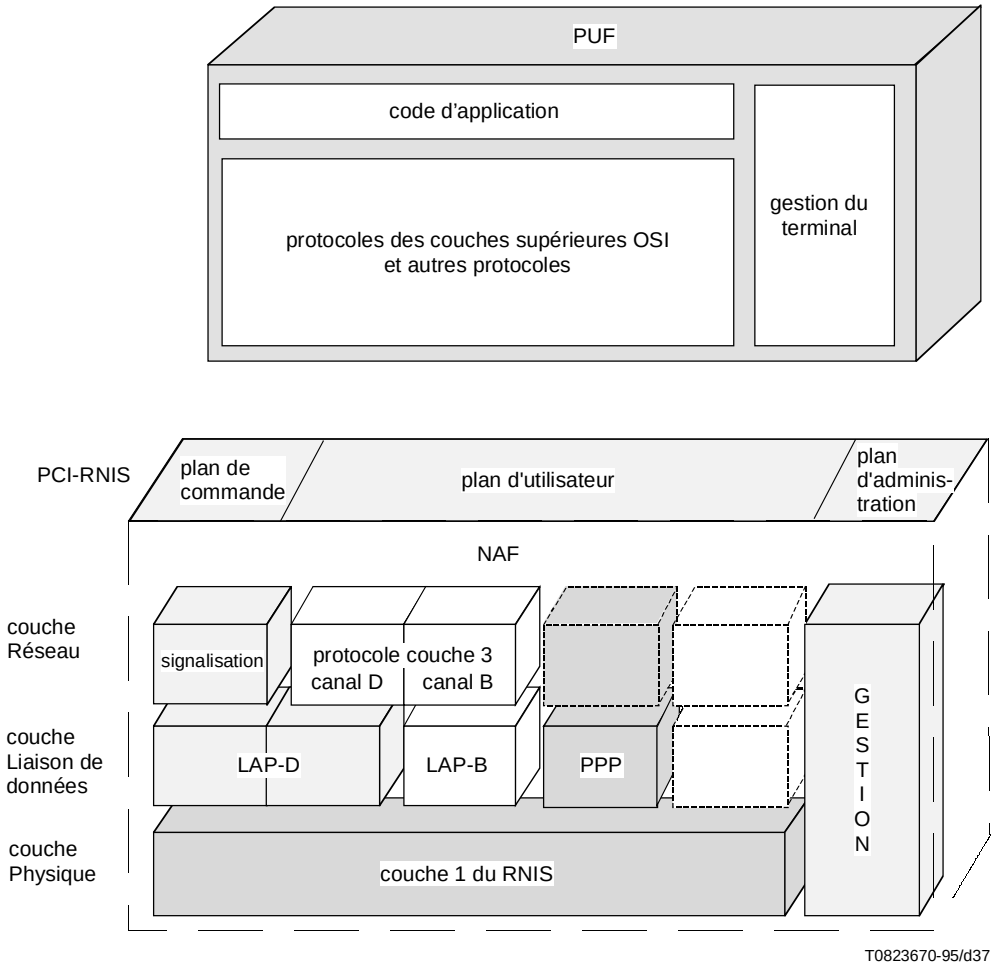


Figure 5 – Emplacement dans le modèle OSI

NOTE – Le protocole PPP est défini comme un ensemble de protocoles que l'on peut subdiviser en deux groupes:

- protocoles de commande de liaison (LCP, *link control protocols*), chargés de l'établissement, de la configuration et des essais de la connexion de couche Liaison de données;
- protocoles de commande de couche Réseau (NCP, *network control protocols*), chargés de l'établissement et de la configuration des différents protocoles de couche Réseau.

L'implémentation du protocole PPP dans l'interface PCI-RNIS n'est applicable qu'aux protocoles de commande de liaison (LCP) (RFC 1548), de contrôle de qualité de liaison PPP (RFC 1333) et d'authentification PPP (RFC 1334) ainsi qu'aux extensions des protocoles PPP de type LCP (RFC 1570).

Les protocoles de type NCP qui gèrent les problèmes relatifs à la configuration des protocoles de couche Réseau sont définis dans des documents spécifiques et sont donc hors du domaine d'application de la présente spécification.

On trouvera en [2] la description générale des conventions.

9.2 Messages

Les messages du plan U donnent accès aux piles protocolaires de type PPP. Le Tableau 12 ci-dessous énumère et décrit brièvement les messages du plan U applicables, dont il donne un aperçu général.

Tableau 12 – Aperçu général des messages du plan U

Identificateur du message	Classe	Nom du message	But du message
301	1	UConnectReq	demande d'établissement d'une connexion d'utilisateur
302	1	UConnectInd	indication de l'établissement d'une connexion d'utilisateur
303	1	UConnectRsp	indication de l'acceptation de l'établissement d'une connexion d'utilisateur
304	1	UConnectCnf	confirmation de l'établissement d'une connexion d'utilisateur
305	1	UDisconnectReq	demande de suppression d'une connexion d'utilisateur
306	1	UDisconnectInd	indication de la suppression d'une connexion d'utilisateur
307	1	UDataReq	demande de transfert de données sur une connexion d'utilisateur établie
308	1	UDataInd	indication de l'arrivée de données transférées sur une connexion d'utilisateur établie
319	1	UErrorInd	indication d'erreur

9.2.1 UConnectReq

Classe: 1 (classe de base)

Description: ce message permet à un dispositif PUF de lancer l'établissement d'une connexion d'utilisateur.

Paramètres:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion d'utilisateur
PPPNegotiation	M	indique la valeur demandée

Message associé: UConnectCnf.

9.2.2 UConnectInd

Classe: 1 (classe de base)

Description: ce message informe un dispositif PUF de l'entrée d'une demande d'établissement d'une connexion d'utilisateur.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur
PPPNegotiation	M	indique la valeur proposée pour cette connexion d'utilisateur

Message associé: UConnectRsp.

9.2.3 UConnectRsp

Classe: 1 (classe de base)

Description: ce message permet à un dispositif PUF d'accepter l'établissement d'une connexion d'utilisateur.

Paramètre:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Message associé: UConnectInd.

9.2.4 UConnectCnf

Classe: 1 (classe de base)

Description: ce message informe le dispositif PUF qu'une connexion d'utilisateur a été établie.

Paramètre:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Message associé: UConnectReq.

9.2.5 UDisconnectReq

Classe: 1 (classe de base)

Description: ce message permet à un dispositif PUF de supprimer une connexion d'utilisateur.

Paramètres:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion d'utilisateur
PPPCause	O	cause invoquée par le protocole PPP pour supprimer la connexion d'utilisateur

Message associé: néant.

9.2.6 UDisconnectInd

Classe: 1 (classe de base)

Description: ce message informe un dispositif PUF du fait qu'une connexion d'utilisateur a été supprimée.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur
PPPOrigin	M	identifie l'initiateur de la suppression de la connexion d'utilisateur
PPPCause	O	cause invoquée par le protocole PPP pour supprimer la connexion d'utilisateur
PPPDiagnostic	C	information complémentaire pour le paramètre PPPCause. Paramètre facultatif si PPPCause est fourni

Message associé: néant.

9.2.7 UDataReq

Classe: 1 (classe de base)

Description: ce message permet à un dispositif PUF d'envoyer un paquet de données, dont la longueur est limitée à la valeur négociée lors de l'établissement de la connexion d'utilisateur.

Paramètre:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Remarque: les données à envoyer sont obligatoires. Elles ne sont pas envoyées sous forme de paramètres du message.

Les données obligatoires doivent être enregistrées dans le tampon de données.

Le champ d'adresse est mis à "11111111" (adressage toutes stations) et le champ de commande est mis à "00000011" (information non numérotée) avec le bit P/F mis à zéro. La séquence FCS est insérée par le NAF en transparence avec le fanion terminant chaque bloc de données.

Message associé: néant.

9.2.8 UDataInd

Classe: 1 (classe de base)

Description: ce message indique à un dispositif PUF la présence de données reçues. La longueur d'un paquet de données est limitée à la valeur négociée au cours de l'établissement de la connexion d'utilisateur.

Paramètre:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Remarque: les données reçues sont toujours fournies, mais pas sous forme de paramètre du message.

Ces données sont enregistrées dans le tampon de données qui, dans ce cas, est obligatoire.

Le champ d'adresse est mis à "11111111" (adressage toutes stations) et le champ de commande est mis à "00000011" (information non numérotée) avec le bit P/F mis à zéro. La séquence FCS est insérée par le NAF en transparence avec le fanion terminant chaque bloc de données.

Message associé: néant.

9.2.9 UErrorInd

Classe: 1 (classe de base).

Description: ce message indique à un dispositif PUF qu'une erreur s'est produite.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion dans le plan de commande
PPPCause	M	identifie le type d'erreur

Message associé: néant.

9.3 Paramètres contenus dans les messages

Le présent sous-paragraphe décrit les paramètres pour le protocole PPP dans le plan d'utilisateur. Le Tableau 13 résume ces paramètres.

Tableau 13 – Aperçu général des paramètres du plan U

Identificateur du paramètre	Nom du paramètre	Usage dans messages du plan U	Usage dans ensemble d'attributs U	Autre usage
50	NCOType			X
62	UProtocol		X	
63	UAttributeName			X
64	UDirection			X
69	PPPOrigin	X		
70	PPPCause	X		
71	PPPDiagnostic	X		
74	PPPNegotiation	X	X	

9.3.1 NCOType

Description: ce paramètre sert à transmettre au dispositif NAF le type de l'objet de connexion réseau.

Type: 50.

Champ	Type de champ	Sens	Requis	Commentaire
Identifier	octet		M	C/U (3) – signalisation et accès d'utilisateur à la couche Réseau

9.3.2 UProtocol

Description: ce paramètre est utilisé pour sélectionner le protocole dans le plan d'utilisateur.

Type: 62.

Champ	Type de champ	Sens	Requis	Commentaire
L3Protocol	octet	P	M	NULL (4)
L2Protocol	octet	P	M	PPP (4)
L1Protocol	octet	P	O	défaut (255) – accès transparent aux canaux B

Remarque: d'autres valeurs possibles (pour d'autres protocoles) sont indiquées en [3].

9.3.3 UAttributeName

Description: ce paramètre sert au dispositif PUF pour transmettre le nom d'un ensemble statique d'attributs du plan U.

Type: 63.

Champ	Type de champ	Sens	Requis	Commentaire
AttributeName	chaîne IA5	P	M	la longueur maximale est de 16 octets

9.3.4 UDirection

Description: ce paramètre sert à transmettre au NAF, pour le plan U, des informations concernant l'usage d'un objet NCO particulier.

Type: 64.

Champ	Type de champ	Sens	Requis	Commentaire
Direction	octet	P	O	dans les deux sens (3)

9.3.5 PPPCause

Description: ce paramètre sert à transmettre, à destination/en provenance du PUF, des informations de cause pour le paramètre PPPCause.

Type: 70.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	B	M	valeur fournie dans le Tableau 14

9.3.6 PPPDiag

Description: ce paramètre sert à transmettre des informations de diagnostic PPP associées à une cause du protocole PPP.

Type: 71.

Champ	Type de champ	Sens	Requis	Commentaire
DiagType	octet	N	M	indique le type de diagnostic associé à la cause PPP ConfNoConverging (0)
NoConvergingDiag	chaîne d'octets	N	M	diagnostic associé à la cause ConfNoConverging (Note)
NOTE – Ces éléments sont présentés dans l'ordre où ils ont été définis pour le message contenant le paramètre PPPNegotiation (voir PPPNegotiation). Par ailleurs, les bits correspondant aux options non acquittées sont activés et ceux qui correspondent aux options acquittées sont remis à zéro.				

9.3.7 PPPNegotiation

Description: ce paramètre sert à indiquer la négociation à effectuer par le protocole PPP.

Type: 74.

Champ		Type de champ	Sens	Requis	Commentaire
PPPNegiotiation	Usage	chaîne d'octets	B	M	ce champ indique si les valeurs suivantes doivent être incluses longueur fixée à 2 octets (Note 1)
	MRUlocal	chaîne d'octets	B	C	ce champ indique la taille maximale de l'unité de réception de l'homologue local défaut (0) longueur fixée à 2 octets (Note 2)
	MRUremote	chaîne d'octets	N	C	ce champ indique la taille maximale de l'unité de réception de l'homologue distant longueur fixée à 2 octets (Note 2)
	Authentproto	octet	B	C	ce champ indique le type d'authentification à effectuer. Ces valeurs s'excluent mutuellement défaut (0) PAP (1) CHAP (2) (Note 2)
	Qualityproto	chaîne d'octets	B	C	ce champ indique la valeur de la période de suivi pour le protocole de contrôle qualité défaut (0) longueur fixée à 4 octets (Note 2)
	Magicnumber	chaîne d'octets	B	C	ce champ indique la valeur du nombre magique à utiliser. défaut (0) longueur fixée à 4 octets (Note 2)
	Protocolcomp	octet	B	C	ce champ indique si la compression du champ de protocole doit être activée (Note 2)
	Addresscomp	octet	B	C	ce champ indique si la compression du champ d'adresse doit être activée (Note 2)
	FCSAlternatives	chaîne d'octets	B	C	ce champ indique la valeur du format de la séquence FCS à utiliser défaut (0) longueur fixée à 4 octets (Note 2)

Champ		Type de champ	Sens	Requis	Commentaire
PPPNegiotiation *(suite)*	SelfDescPadding	chaîne d'octets	B	C	ce champ indique la valeur du bourrage autodescripteur à utiliser défaut (0) longueur fixée à 4 octets
	CallBack	octet	B	C	ce champ indique si l'option de rétroappel doit être activée (Note 2)
	CompoundFrame	octet	B	C	ce champ indique si l'option de trames composites est à activer (Note 2)
NOTE 1 – Chaque bit de cet indicateur correspond à une option placée dans l'ordre indiqué dans ce tableau (c'est-à-dire que le premier élément binaire se rapporte à l'option MRUlocal, le second à l'option MRUremote, le troisième à l'option Authentproto, etc.). Lorsque l'indicateur relatif à une option n'est pas activé, cela signifie que l'option de protocole PPP ne doit pas être négociée. NOTE 2 – Avant de définir un paramètre de négociation, le dispositif PUF doit, au moyen de la fonction PciGetProperty (voir la Partie 2 [2]), vérifier si la propriété est assurée par le dispositif NAF.					

9.3.8 PPPOrigin

Description: ce paramètre est utilisé pour transmettre, à destination/en provenance du PUF, des informations relatives à l'origine du protocole PPP.

Type: 69.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	B	M	indéfinie (1) fournisseur du NAF (2) utilisateur du PUF (3)

9.4 Diagramme de transition d'état

La Figure 6 montre les différents états par lesquels une connexion du plan U peut passer, sous la forme de messages du plan U présentés dans l'ordre où ils doivent être utilisés.

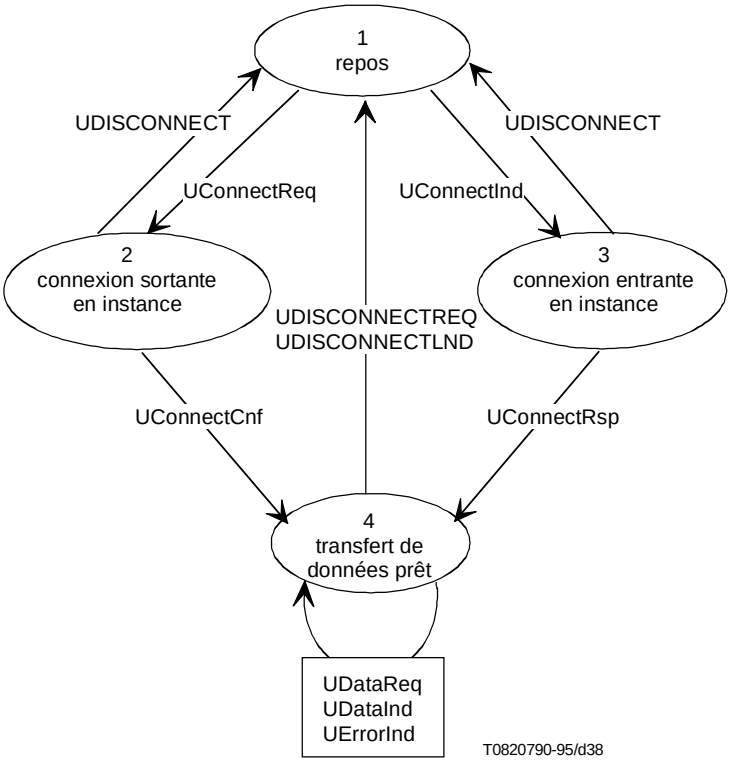


Figure 6 – Aperçu général des messages du plan U

9.5 Fonction de coordination

La fonction de coordination ne peut pas être utilisée avec le protocole du plan d'utilisateur associé au protocole PPP.

9.6 Critères de sélection

Aucun paramètre spécifique n'est utilisé pour le protocole PPP. Les critères généraux de sélection des objets NCO sont indiqués en [2].

9.7 Traitement et codes d'erreur spécifiques

Les erreurs sont traitées comme suit.

9.7.1 Erreurs

En cas d'envoi d'informations de rejet du protocole par le terminal distant, le message UErrorInd est envoyé au dispositif PUF.

En cas de longueur non valide du paramètre UserData dans le message UDataReq, les données sont ignorées.

9.7.2 Causes

Ces valeurs peuvent être spécifiées et sont retournées dans le paramètre PPPCause.

Tableau 14 – Valeurs du paramètre PPPCause

Code à retourner		Signification	Information d'erreur spécifique
Undefined	220	situation d'erreur indéfinie	absente
DiscNorm	241	déconnexion – état normal	absente
ConfNoConverging	244	connexion rejetée – non-réponse du serveur (état transitoire)	absente
Hostunreachable	245	connexion rejetée – incompatibilité des configurations (état fixe)	absente
Protocol Error	212	erreur de protocole	absente

9.8 Attributs statiques

9.8.1 Paramètres de l'ensemble statique d'attributs

Tableau 15 – Paramètres d'ensemble d'attributs du plan U (UAttributeSet)

Paramètre	Requis	Commentaire
UProtocol	O	voir la remarque et 9.3.2
MRUlocal	O	voir la remarque et 9.3.7 (PPPNegotiation)
MRUremote	O	voir la remarque et 9.3.7
Authentproto	O	voir la remarque et 9.3.7
Qualityproto	O	voir la remarque et 9.3.7
MagicNumber	O	voir la remarque et 9.3.7
Protocolcomp	O	voir la remarque et 9.3.7
Addresscomp	O	voir la remarque et 9.3.7
FCSAlternatives	O	voir la remarque et 9.3.7
SelfDescPadding	O	voir la remarque et 9.3.7
CallBack	O	voir la remarque et 9.3.7
CompoundFrame	O	voir la remarque et 9.3.7

Remarque: ces paramètres ne peuvent être utilisés que pendant la création d'un objet NCO contenant des informations sur le plan de commande. Pour plus de détails, voir en [2] le sous-paragraphe en ce qui concerne l'opération ACreateNCO.

Si des paramètres sont omis, le dispositif NAF doit utiliser les valeurs par défaut qui sont décrites dans l'Appendice I.

9.8.2 Contenu de l'ensemble statique d'attributs

nom:	U_PPP
UProtocol:	PPP
MRUlocal:	1500
MRUremote:	1500
Authentproto:	néant
Qualityproto:	néant
MagicNumber:	néant
Protocolcomp:	néant
AddressComp:	néant
FCSAlternatives:	néant
SelfDescPadding:	néant
CallBack:	néant
CompoundFrame:	néant

9.9 Informations propres au protocole sur les propriétés du NAF

Les paramètres propres au protocole PPP pour la propriété NAF sont indiqués dans le Tableau 16.

Voir également la fonction PciGetProperty dans la Partie 2 (Services de base).

Tableau 16 – Paramètre de propriété NAF à codage TLV

Paramètre	Fourni	Codage TLV			Commentaire et valeur
		Type (ID)	Lon-gueur	Valeur	
PPPNegotiation	M	14	2.27	octet	ce paramètre indique les options PPP offertes par le NAF

10 Protocole SDLC

10.1 Introduction

Le présent paragraphe traite du protocole de commande de liaison de données synchrone (SDLC).

Le plan d'utilisateur offre les services du protocole SDLC au moyen des procédures du plan U sur canal B. Pour cet accès, le dispositif NAF considère la couche 3 comme vide (NULL), comme indiqué sur la Figure 7.

Le protocole SDLC pris en compte est une liaison synchrone en mode de réponse normal, dans une configuration de point à point. La revue "IBM Synchronous Data Link Control Concepts" (GA27-3093) donne un aperçu général et des informations sur ce protocole.

Le protocole SDLC s'insère dans le modèle OSI comme indiqué sur la Figure 7 ci-dessous.

On trouvera en [2] la description générale des conventions.

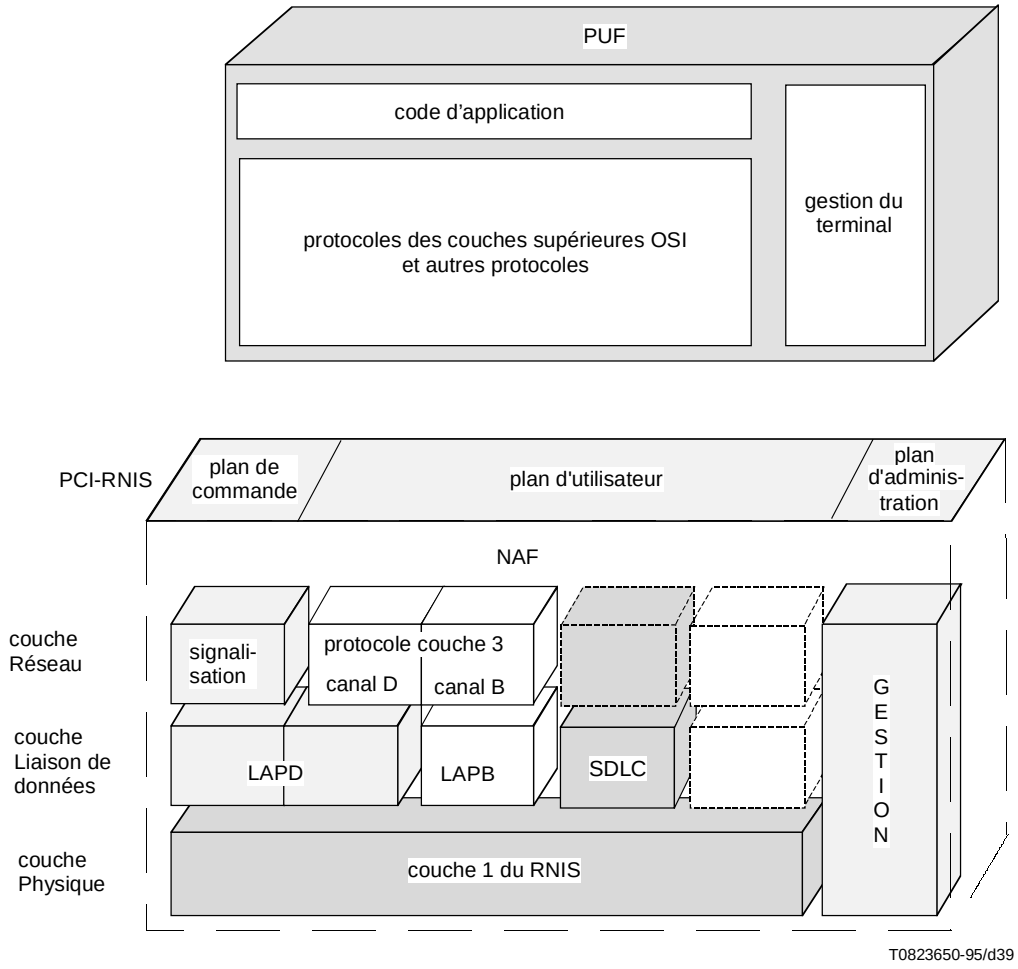


Figure 7 – Emplacement dans le modèle OSI

10.2 Messages

Les messages du plan U donnent accès aux piles protocolaires de type SDLC. Le Tableau 17 ci-dessous énumère et décrit brièvement les messages du plan U applicables, dont il donne un aperçu général.

Tableau 17 – Aperçu général des messages du plan U

Identificateur du message	Classe	Nom du message	But du message
301	1	UConnectReq	demande d'établissement d'une connexion d'utilisateur
302	1	UConnectInd	indication de l'établissement d'une connexion d'utilisateur
303	1	UConnectRsp	indication de l'acceptation de l'établissement d'une connexion d'utilisateur
304	1	UConnectCnf	confirmation de l'établissement d'une connexion d'utilisateur
305	1	UDisconnectReq	demande de suppression d'une connexion d'utilisateur
306	1	UDisconnectInd	indication de la suppression d'une connexion d'utilisateur
307	1	UDataReq	demande de transfert de données sur une connexion d'utilisateur établie
308	1	UDataInd	indication de l'arrivée de données transférées sur une connexion d'utilisateur établie
309	1	UExpeditedDataReq	demande de transfert de données exprès sur une connexion d'utilisateur établie
310	1	UExpeditedDataInd	indication de la présence de données exprès transférées sur une connexion d'utilisateur établie
317	1	UReadyToReceiveReq	message utilisé pour effectuer une commande de débit pour une connexion d'utilisateur
318	1	UReadyToReceiveInd	message utilisé pour indiquer le statut d'une commande de débit pour une connexion d'utilisateur

10.2.1 UConnectReq

Classe: 1 (classe de base).

Description: ce message permet à un dispositif PUF de lancer l'établissement d'une connexion d'utilisateur.

Paramètre:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Message associé: UConnectCnf.

10.2.2 UConnectInd

Classe: 1 (classe de base).

Description: ce message informe un dispositif PUF de l'entrée d'une demande d'établissement d'une connexion d'utilisateur. Ce message informe le PUF de la fin d'un état de repos transitoire dans la connexion d'utilisateur, dû à une réinitialisation de la couche Liaison de données.

Paramètre:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Message associé: UConnectRsp.

10.2.3 UConnectRsp

Classe: 1 (classe de base).

Description: ce message permet à un dispositif PUF d'accepter l'établissement d'une connexion d'utilisateur.

Paramètre:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Message associé: UConnectInd.

10.2.4 UConnectCnf

Classe: 1 (classe de base).

Description: ce message informe le dispositif PUF qu'une connexion d'utilisateur a été établie.

Paramètre:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Message associé: UconnectReq.

10.2.5 UDisconnectReq

Classe: 1 (classe de base).

Description: ce message permet à un dispositif PUF de supprimer une connexion d'utilisateur.

Paramètre:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Message associé: néant.

10.2.6 UDisconnectInd

Classe: 1 (classe de base).

Description: ce message informe un dispositif PUF du fait qu'une connexion d'utilisateur a été supprimée. Ce message peut informer le PUF d'un état de repos transitoire de la connexion d'utilisateur, dû à une réinitialisation de la couche Liaison de données. Dans ce cas, la valeur du paramètre SDLCCause est DiscTrans (déconnexion – état transitoire) c'est-à-dire 225 en base décimale.

Paramètre:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur
SDLCOrigin	M	identifie l'initiateur de la suppression de la connexion d'utilisateur
SDLCCause	C	cause invoquée par le protocole SDLC pour supprimer la connexion d'utilisateur

Message associé: néant.

10.2.7 UDataReq

Classe: 1 (classe de base).

Description: ce message permet à un dispositif PUF d'envoyer un paquet de données, dont la longueur est limitée à la valeur négociée lors de l'établissement de la connexion d'utilisateur. Aucun mécanisme de fragmentation n'est disponible au niveau de la couche Liaison de données pour le protocole SDLC.

Paramètre:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Remarque: les données à envoyer sont obligatoires. Elles ne sont pas envoyées sous forme de paramètres du message.

Les données obligatoires doivent être enregistrées dans le tampon de données.

Message associé: néant.

10.2.8 UDataInd

Classe: 1 (classe de base).

Description: ce message indique à un dispositif PUF la présence de données reçues. La longueur d'un paquet de données est limitée à la valeur négociée au cours de l'établissement de la connexion d'utilisateur. Aucun mécanisme de fragmentation n'est disponible au niveau de la couche Liaison de données pour le protocole SDLC.

Paramètre:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Remarque: les données reçues sont toujours fournies, mais pas sous forme de paramètre du message.

Ces données sont enregistrées dans le tampon de données qui, dans ce cas, est obligatoire.

Message associé: néant.

10.2.9 UExpeditedDataReq

Classe: 1 (classe de base).

Description: ce message permet à un PUF d'envoyer des données exprès. Ces données ne sont pas sensibles au mécanisme de commande de débit utilisé pour régler les messages de type UDataReq. Les données exprès SDLC sont transmises dans des trames d'informations non numérotées.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur
UserData	M	données exprès à transférer

Message associé: néant.

10.2.10 UExpeditedDataInd

Classe: 1 (classe de base).

Description: ce message indique à un PUF la réception de données exprès. Ces données ne sont pas sensibles aux mécanismes de commande de débit utilisés pour régler les messages de type UDataInd. Les données exprès SDLC sont reçues dans des trames d'informations non numérotées.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur
UserData	M	données exprès reçues

Message associé: néant.

10.2.11 UReadyToReceiveReq

Classe: 1 (classe de base).

Description: ce message permet au PUF d'indiquer au NAF qu'il peut accepter des données entrantes (message UDataInd). Ce message ne peut s'appliquer qu'à une connexion d'utilisateur déjà établie. Le réglage du paramètre ReadyFlag sur Vrai permet au NAF de transférer les données entrantes vers le PUF. Le réglage du paramètre ReadyFlag sur Faux inhibe ce transfert.

Ce mécanisme de commande de débit n'implique pas un contrôle de flux de bout en bout.

Paramètre:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion d'utilisateur
ReadyFlag	M	ce fanion indique si le PUF est ou non prêt à accepter des données entrantes

Remarque: pour une connexion donnée, si plus d'un message est envoyé avec le même fanion, ce message doit être ignoré par le NAF.

Message associé: UDataInd.

10.2.12 UReadyToReceiveInd

Classe: 1 (classe de base).

Description: ce message permet au NAF d'indiquer au PUF si la connexion U permet l'envoi de données (messages UDataReq). Ce message ne peut s'appliquer qu'à une connexion d'utilisateur déjà établie. Si la valeur du paramètre ReadyFlag est Faux, le dispositif NAF ne peut pas envoyer de données. Si la valeur est Vrai, le NAF indique que le transfert de données est autorisé.

Ce mécanisme de commande de débit n'implique pas un contrôle de flux de bout en bout.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur
ReadyFlag	M	ce fanion indique si le NAF est ou non prêt à recevoir des données pour transmission sur une connexion U

Message associé: UDataReq.

10.3 Paramètres contenus dans les messages

Le présent sous-paragraphe décrit les paramètres utilisés pour le protocole SDLC. Le Tableau 18 en présente le résumé.

Tableau 18 – Aperçu général des paramètres du plan U

Identificateur du paramètre	Nom du paramètre	Usage dans messages du plan U	Usage dans ensemble d'attributs U	Autre usage
38	L2ConnectionMode		X	
39	L2FrameSize		X	
40	L2WindowSize		X	
41	L2XID		X	
50	NCOType			X
55	ReadyFlag	X		
62	UProtocol		X	
63	UAttributeName			X
64	UDirection			X
65	UserData	X		
68	SDLCCause	X		
69	SDLCOrigin	X		

10.3.1 L2ConnectionMode

Description: ce paramètre n'est utilisé que s'il n'est pas défini dans le champ de valeur du paramètre L2XID. Il sert à transmettre au NAF des détails sur le mode de connexion dans la couche 2.

Type: 38.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	P	M	dte (1) – L'ETTD remplit le rôle de station secondaire pour la liaison (non négociable) dce (2) – L'ETCD remplit le rôle de station primaire pour la liaison (non négociable) auto (3) – Le rôle de station pour la liaison est négociable par échange d'identificateurs XID

10.3.2 L2FrameSize

Description: ce paramètre n'est utilisé que s'il n'est pas défini dans le champ de valeur du paramètre L2XID. Il sert à transmettre au NAF des détails sur la longueur de trame en couche 2.

Type: 39.

Champ	Type de champ	Sens	Requis	Commentaire
Value	chaîne d'octets	P	M	longueur de trame (en octets) la longueur est fixée à 2 octets le premier octet est le plus significatif

10.3.3 L2WindowSize

Description: ce paramètre n'est utilisé que s'il n'est pas défini dans le champ de valeur du paramètre L2XID. Il sert à transmettre au NAF des détails sur la longueur des fenêtres en couche 2.

Type: 40.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	P	M	longueur de fenêtre

10.3.4 L2XID

Description: ce paramètre est utilisé pour transmettre des détails sur la valeur de l'identificateur XID de la couche 2 et sur son usage. Le champ d'information XID peut comporter des valeurs ayant priorité sur certains paramètres définis par ailleurs. Les formats des champs XID des protocoles DLC sont décrits dans la publication IBM "Systems Network Architecture – Formats" (GA27-3136-11).

Type: 41.

Champ	Type de champ	Sens	Requis	Commentaire
Use	octet	P	M	non applicable au protocole SDLC
Value	chaîne d'octets	P	M	valeur XID (identificateur et signature) champ d'information XID formaté pour SDLC. Longueur maximale: 127 octets.

10.3.5 NCOType

Description: ce paramètre sert à transmettre au NAF le type d'objet de connexion réseau.

Type: 50.

Champ	Type de champ	Sens	Requis	Commentaire
Identifier	octet		M	C/U (3) – signalisation et accès d'utilisateur à la couche Réseau

Remarque: une connexion à commande SDLC ne peut être définie que par un objet NCO de type C/U. Aucun objet NCO de type U3/G ne peut être groupé avec un objet NCO définissant une connexion à commande SDLC.

10.3.6 ReadyFlag

Description: ce paramètre sert à demander et à indiquer le statut de la commande de débit concernant une connexion du plan U.

Type: 55.

Champ	Type de champ	Sens	Requis	Commentaire
Usage	Booléen	B	M	TRUE – transfert de données autorisé FALSE – transfert de données non autorisé

10.3.7 UProtocol

Description: ce paramètre sert à sélectionner le protocole du plan U.

Type: 62.

Champ	Type de champ	Sens	Requis	Commentaire
L3Protocol	octet	P	M	NULL (4)
L2Protocol	octet	P	M	SDLC (5)
L1Protocol	octet	P	O	défaut (255) – accès transparent aux canaux B

Remarque: d'autres valeurs possibles (pour d'autres protocoles) sont indiquées en [3].

10.3.8 UAttributeName

Description: ce paramètre sert au PUF à envoyer le nom d'un ensemble statique d'attributs du plan U.

Type: 63.

Champ	Type de champ	Sens	Requis	Commentaire
AttributeName	chaîne IA5	P	M	16 octets est la longueur maximale

10.3.9 UDirection

Description: ce paramètre sert à transmettre au NAF des informations concernant l'usage d'un objet NCO particulier, pour le plan U.

Type: 64.

Champ	Type de champ	Sens	Requis	Commentaire
Direction	octet	P	O	dans les deux sens (3)

10.3.10 UserData

Description: ce paramètre sert à transmettre, à destination/en provenance du PUF, des données de longueur limitée.

Type: 65.

Champ	Type de champ	Sens	Requis	Commentaire
Data	chaîne d'octets	B	M	longueur maximale: 128 octets.

10.3.11 SDLCCause

Description: ce paramètre sert à transmettre, à destination/en provenance du PUF, des informations de cause pour le protocole SDLC.

Type: 68.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	B	M	voir au 10.7.2 les valeurs des codes de retour pour le plan U

10.3.12 SDLCOrigin

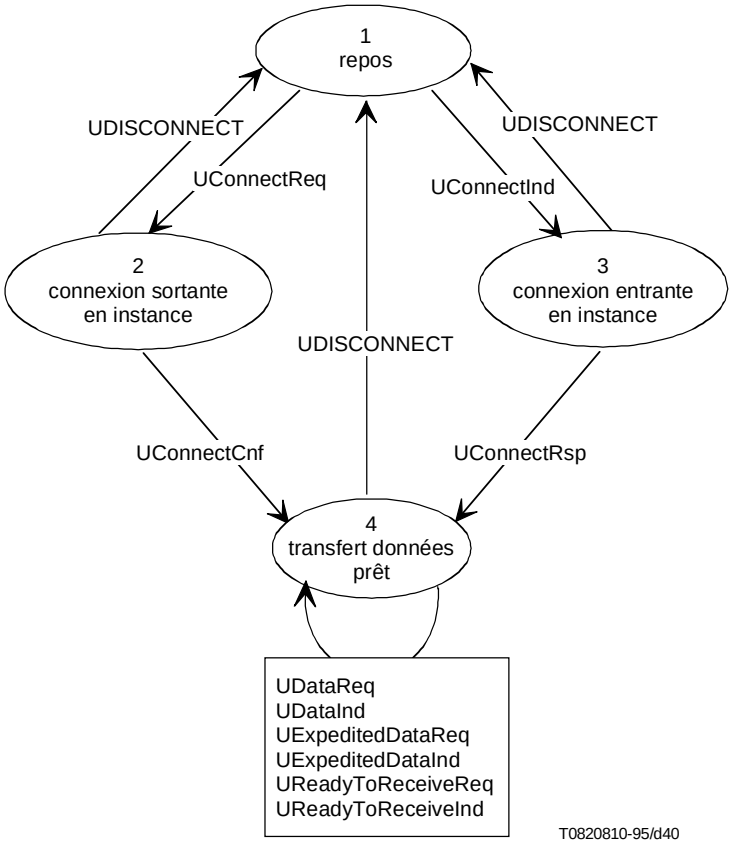
Description: ce paramètre est utilisé pour transmettre, à destination/en provenance du PUF, des informations relatives à l'origine du protocole SDLC.

Type: 69.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	B	M	indéfinie (1) fournisseur du NAF (2) utilisateur du PUF (3)

10.4 Diagramme de transition d'état

La Figure 8 montre les différents états par lesquels une connexion du plan U peut passer, sous la forme de messages du plan U présentés dans l'ordre où ils doivent être utilisés.



NOTE – L'indication UDISCONNECT implique un message UDisconnectReq ou UDisconnectInd.

Figure 8 – Aperçu général des messages dans le plan U

10.5 Fonction de coordination

La fonction de coordination ne peut pas être utilisée avec le protocole du plan U associé au protocole SDLC.

10.6 Critères de sélection

Aucun paramètre propre au protocole SDLC n'est utilisé. Les critères généraux de sélection des objets NCO sont indiqués en [2].

10.7 Traitement et codes d'erreur spécifiques

Les erreurs sont traitées comme suit.

10.7.1 Utilisation non valide de messages d'utilisateur

En cas:

– de longueur non valide du paramètre UserData dans un message UDataReq;

– d'utilisation non valide du paramètre ExpeditedData,

l'action est la suivante:

– le message UDisconnectInd est envoyé au dispositif PUF.

10.7.2 Causes

Ces valeurs sont spécifiées dans le Tableau 19 et sont retournées dans le paramètre SDLCCause.

Tableau 19 – Valeurs du paramètre SDLCCause

Code de retour		Signification	Information d'erreur spécifique
Undefined	220	situation d'erreur indéfinie	absente
DiscTrans	225	déconnexion – état transitoire indique que la couche 2 est en réinitialisation	absente
DiscPerm	226	déconnexion – état permanent indique que la station distante n'est plus accessible	absente
DiscNorm	241	déconnexion – état normal indique que la déconnexion a été demandée par la station distante	absente
ConRejectTrans	244	rejet de connexion – état transitoire indique que l'activation de la couche de Liaison de données a été refusée par la station distante	absente
ConRejectPerm	245	rejet de la connexion – état fixe indique que la station distante est inaccessible	absente

10.8 Attributs statiques

10.8.1 Paramètres de l'ensemble statique d'attributs

Tableau 20 – Paramètres d'ensemble d'attributs du plan U (UAttributeSet)

Paramètre	Requis	Commentaire
UProtocol	O	voir la remarque et 10.3.7
L2ConnectionMode	O	voir la remarque et 10.3.1
L2WindowSize	O	voir la remarque et 10.3.3
L2FrameSize	O	voir la remarque et 10.3.2
L2XID	O	voir la remarque et 10.3.4

Remarque: ces paramètres ne peuvent être utilisés que pendant la création d'un objet NCO contenant des informations sur le plan de commande. Pour plus de détails, voir en [2] le sous-paragraphe "opération ACreateNCO".

Si des paramètres sont omis, le dispositif NAF doit utiliser les valeurs par défaut qui sont décrites dans l'Appendice I.

Les paramètres L2ConnectionMode, L2WindowSize et L2FrameSize peuvent contenir des valeurs définies par l'utilisateur ou des valeurs de configuration par défaut, si ces valeurs ne sont pas fournies par le dispositif NAF. Ils peuvent également contenir des valeurs résultant d'une négociation pour l'échange d'identificateurs XID. Le paramètre L2XID contient le champ d'information d'identification XID du protocole DLC.

10.8.2 Contenu de l'ensemble statique d'attributs

nom:	U_SDLC
UProtocol:	SDLC
L2ConnectionMode:	dte
L2WindowSize:	7
L2FrameSize:	265
L2XID:	non utilisé

11 Protocole V.110

11.1 Introduction

Le présent paragraphe traite du protocole V.110, qui permet à un dispositif PUF de demander à un dispositif NAF un canal B utilisant le protocole V.110. Les deux options du protocole V.110, transfert synchrone et transfert asynchrone, sont admises.

Ce protocole utilise la couche 3 vide (NULL) comme indiqué sur la Figure 9.

L'emplacement du modèle OSI du protocole V.110 est montré à la Figure 9.

On trouvera en [2] une description générale des conventions.

Une négociation de capacités V.110 peut se produire au moyen du paramètre BearerCap. Dans ce cas, les octets 5a, 5b, 5c et 5d de ce paramètre peuvent être modifiés (voir [2]).

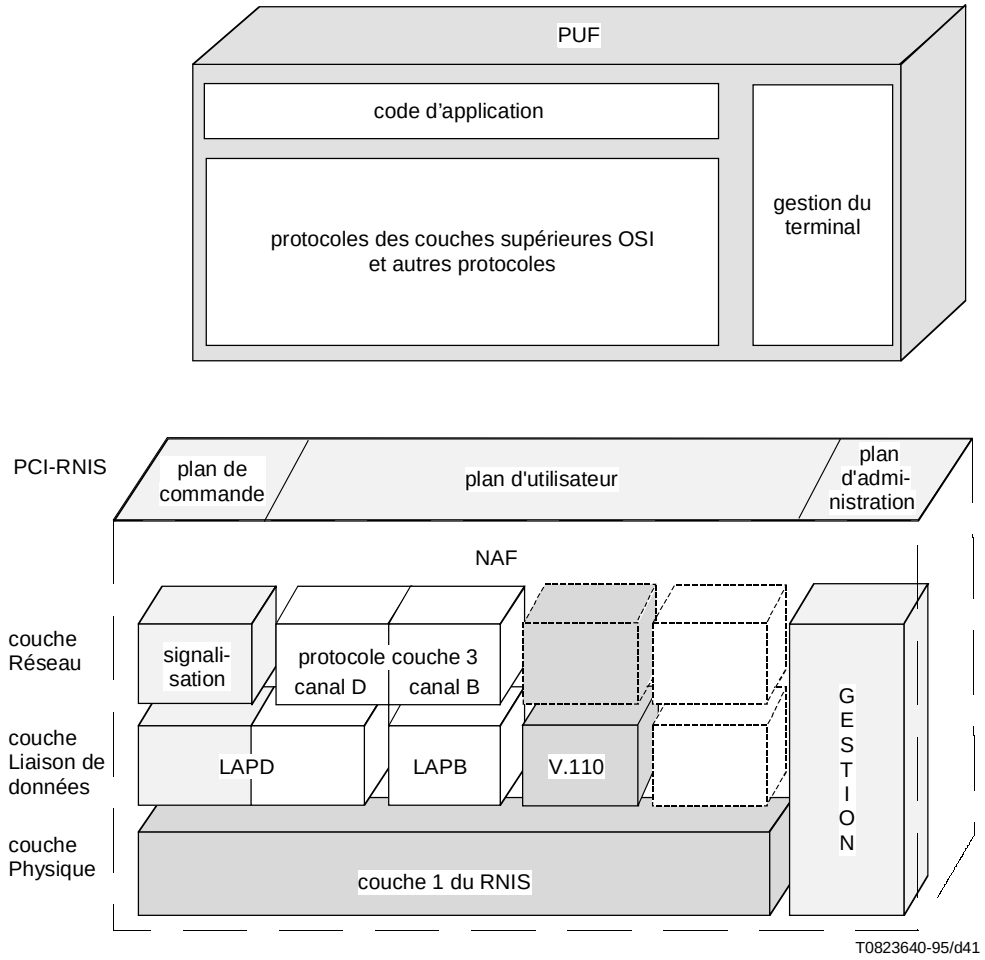


Figure 9 – Emplacement dans le modèle OSI

11.2 Messages

Les messages du plan U donnent accès aux piles protocolaires de type V.110. Le Tableau 21 ci-dessous énumère et décrit brièvement les messages du plan U applicables, dont il donne un aperçu général.

Tableau 21 – Aperçu général des messages du plan U

Identificateur du message	Classe	Nom du message	But du message
301	1	UConnectReq	demande d'établissement d'une connexion d'utilisateur
302	1	UConnectInd	indication de l'établissement d'une connexion d'utilisateur
303	1	UConnectRsp	indication de l'acceptation de l'établissement d'une connexion d'utilisateur
304	1	UConnectCnf	confirmation de l'établissement d'une connexion d'utilisateur
305	1	UDisconnectReq	demande de suppression d'une connexion d'utilisateur
306	1	UDisconnectInd	indication de la suppression d'une connexion d'utilisateur
307	1	UDataReq	demande de transfert de données sur une connexion d'utilisateur établie
308	1	UDataInd	indication de l'arrivée de données transférées sur une connexion d'utilisateur établie
317	1	UReadyToReceiveReq	message utilisé pour effectuer une commande de débit pour une connexion d'utilisateur
318	1	UReadyToReceiveInd	message utilisé pour indiquer le statut d'une commande de débit pour une connexion d'utilisateur

11.2.1 UConnectReq

Classe: 1 (classe de base).

Description: ce message permet à un dispositif PUF de lancer l'établissement d'une connexion d'utilisateur.

Paramètre:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Message associé: UConnectCnf.

11.2.2 UConnectInd

Classe: 1 (classe de base).

Description: ce message informe un dispositif PUF de l'entrée d'une demande d'établissement d'une connexion d'utilisateur.

Paramètre:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Message associé: UConnectRsp.

11.2.3 UConnectRsp

Classe: 1 (classe de base).

Description: ce message permet à un dispositif PUF d'accepter l'établissement d'une connexion d'utilisateur.

Paramètre:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Message associé: UConnectInd.

11.2.4 UConnectCnf

Classe: 1 (classe de base).

Description: ce message informe le dispositif PUF qu'une connexion d'utilisateur a été établie.

Paramètre:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Message associé: UConnectReq.

11.2.5 UDisconnectReq

Classe: 1 (classe de base).

Description: ce message permet à un dispositif PUF de supprimer une connexion d'utilisateur.

Paramètre:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Message associé: néant.

11.2.6 UDisconnectInd

Classe: 1 (classe de base).

Description: ce message informe un dispositif PUF du fait qu'une connexion d'utilisateur a été supprimée.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur
V.110Origin	M	identifie l'initiateur de la suppression de la connexion d'utilisateur
V.110Cause	C	cause invoquée par le protocole V.110 pour supprimer la connexion d'utilisateur

Message associé: néant.

11.2.7 UDataReq

Classe: 1 (classe de base).

Description: ce message permet à un dispositif PUF d'envoyer un paquet de données, dont la longueur est limitée à la valeur autorisée par l'interface PCI-RNIS, c'est-à-dire 4096 octets.

Paramètre:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Remarque: les données à envoyer sont obligatoires. Elles ne sont pas envoyées sous forme de paramètres du message. Les données obligatoires doivent être enregistrées dans le tampon de données.

Message associé: néant.

11.2.8 UDataInd

Classe: 1 (classe de base).

Description: ce message indique à un dispositif PUF la présence de données reçues. La longueur d'un paquet de données est limitée à la valeur autorisée par l'interface PCI-RNIS, c'est-à-dire 4096 octets.

Paramètre:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur

Remarque: les données reçues sont toujours fournies, mais pas sous forme de paramètre du message. Ces données sont enregistrées dans le tampon de données qui, dans ce cas, est obligatoire.

Message associé: néant.

11.2.9 UReadyToReceiveReq

Classe: 1 (classe de base).

Description: ce message permet au PUF d'indiquer au NAF qu'il peut accepter des données entrantes (message UDataInd). Ce message ne peut s'appliquer qu'à une connexion d'utilisateur déjà établie. Le réglage du paramètre ReadyFlag sur TRUE permet au NAF de transférer les données entrantes vers le PUF. Le réglage du paramètre ReadyFlag sur FALSE inhibe ce transfert.

Ce mécanisme de commande de débit n'implique pas un contrôle de flux de bout en bout.

Paramètres:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion d'utilisateur
ReadyFlag	M	ce fanion indique si le PUF est ou non prêt à accepter des données entrantes

Remarque: pour une connexion donnée, si plus d'un message est envoyé avec le même fanion, ce message doit être ignoré par le NAF.

Message associé: UDataInd.

11.2.10 UReadyToReceiveInd

Classe: 1 (classe de base).

Description: ce message permet au NAF d'indiquer au PUF si la connexion U permet l'envoi de données (messages UDataReq). Ce message ne peut s'appliquer qu'à une connexion d'utilisateur déjà établie. Si la valeur du paramètre ReadyFlag est FALSE, le dispositif NAF ne peut pas envoyer de données. Si la valeur est TRUE, le NAF indique que le transfert de données est autorisé.

Ce mécanisme de commande de débit n'implique pas un contrôle de flux de bout en bout.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion d'utilisateur
ReadyFlag	M	ce fanion indique si le NAF est ou non prêt à recevoir des données pour transmission sur une connexion U

Message associé: UDataReq.

11.3 Paramètres contenus dans les messages

Le présent sous-paragraphe décrit les paramètres utilisés pour le protocole V.110. Le Tableau 22 en présente le résumé.

Tableau 22 – Aperçu général des paramètres du plan U

Identificateur du paramètre	Nom du paramètre	Utilisé dans des messages du plan U	Utilisé dans un ensemble d'attributs du plan U	Autre usage
50	NCOType			X
55	ReadyFlag	X		
62	UProtocol		X	
63	UAttributeName			X
64	UDirection			X
68	V.110Cause	X		
69	V.110Origin	X		
75	FlowControlMechanism		X	
76	FlowControlCharacters		X	
77	MomentNumber		X	
78	V.110BChannelDisconnection		X	

11.3.1 NCOType

Description: ce paramètre sert à transmettre au NAF le type d'objet de connexion réseau.

Type: 50.

Champ	Type de champ	Sens	Requis	Commentaire
Identifier	octet		M	C/U (3) – signalisation et accès d'utilisateur à la couche Réseau

11.3.2 ReadyFlag

Description: ce paramètre sert à demander et à indiquer le statut de la commande de débit concernant une connexion du plan U.

Type: 55.

Champ	Type de champ	Sens	Requis	Commentaire
Usage	Booléen	B	M	TRUE – transfert de données autorisé FALSE – transfert de données non autorisé

11.3.3 UProtocol

Description: ce paramètre sert à sélectionner le protocole du plan U. Si sa longueur est de 3 octets, le premier octet contient le protocole de couche 3 demandé, le deuxième octet contient le protocole de couche 2 demandé et le troisième octet contient le protocole de couche 1 demandé.

Type: 62.

Champ	Type de champ	Sens	Requis	Commentaire
L3Protocol	octet	P	M	NULL (4)
L2Protocol	octet	P	M	V.110 asynchrone (6) V.110 synchrone (7)
L1Protocol	octet	P	O	défaut (255) – accès transparent aux canaux B

Remarque: d'autres valeurs possibles (pour d'autres protocoles) sont indiquées en [3].

11.3.4 UAttributeName

Description: ce paramètre sert au PUF à envoyer le nom d'un ensemble statique d'attributs du plan U.

Type: 63.

Champ	Type de champ	Sens	Requis	Commentaire
AttributeName	chaîne IA5	P	M	16 octets est la longueur maximale

11.3.5 UDirection

Description: ce paramètre sert à transmettre au NAF des informations concernant l'usage d'un objet NCO particulier, pour le plan U.

Type: 64.

Champ	Type de champ	Sens	Requis	Commentaire
Direction	octet	P	O	dans les deux sens (3)

11.3.6 V.110Cause

Description: ce paramètre sert à transmettre, à destination/en provenance du PUF, des informations de cause pour le protocole V.110.

Type: 67.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	B	M	voir au Tableau 23 les valeurs possibles

11.3.7 V.110Origin

Description: ce paramètre est utilisé pour transmettre, à destination/en provenance du PUF, des informations relatives à l'origine du protocole V.110.

Type: 68.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	B	M	indéfinie (1) fournisseur du NAF (2) utilisateur du PUF (3)

11.3.8 FlowControlMechanism

Description: ce paramètre sert à négocier le mécanisme de commande de débit pour une connexion en protocole V.110. Deux possibilités existent: la première consiste à utiliser les caractères XON/XOFF, la deuxième à utiliser les circuits 105/106 V.24. Ce paramètre sert à la négociation de bout en bout.

Type: 75.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	P	M	type de mécanisme utilisé: – 0 XON/XOFF (caractères); – 1 105/106 (circuits). valeur par défaut: 0

11.3.9 FlowControlCharacters

Description: ce paramètre sert à fixer les caractères définissant la commande de débit pour une connexion V.110. Ces caractères peuvent être différents dans chaque sens, de sorte que deux caractères doivent obligatoirement être indiqués, même s'ils ont la même valeur. Ce paramètre n'a qu'une signification locale.

Type: 76.

Champ	Type de champ	Sens	Requis	Commentaire
Value	chaîne d'octets	P	M	longueur fixe: 2. le premier octet identifie le caractère XON. Valeur par défaut: 16. le second octet identifie le caractère XOFF. Valeur par défaut: 18.

11.3.10 MomentNumber

Description: ce paramètre sert à fixer le nombre de moments pour une connexion V.110. Il n'a qu'une signification locale.

Type: 77.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	P	M	nombre de moments

11.3.11 V.110BChannelDisconnection

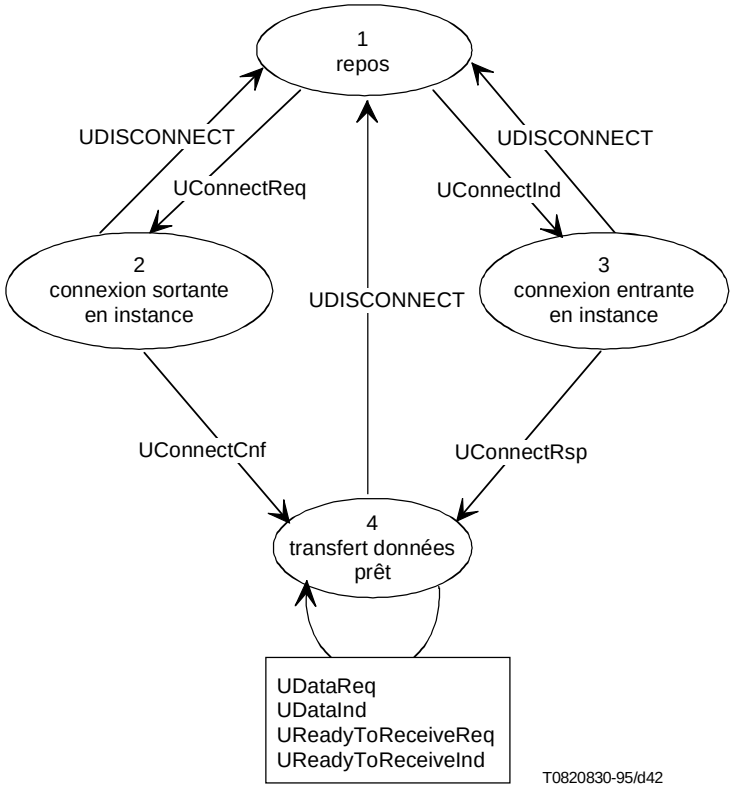
Description: une déconnexion par protocole V.110 peut impliquer la libération du canal B. Ce paramètre sert à acheminer cette information. Il n'a qu'une signification locale.

Type: 78.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	P	M	le protocole V.110 implique une déconnexion de canal B: – 0 non-déconnexion; – 1 déconnexion.

11.4 Diagramme de transition d'état

La Figure 10 montre les différents états par lesquels une connexion du plan U peut passer, sous la forme de messages du plan U présentés dans l'ordre où ils doivent être utilisés.



NOTE – L'indication UDISCONNECT implique un message UDisconnectReq ou UDisconnectInd.

Figure 10 – Aperçu général des messages dans le plan U

11.5 Fonction de coordination

La fonction de coordination ne peut pas être utilisée avec le protocole de plan U associé à l'accès V.110.

11.6 Critères de sélection

Aucun paramètre spécifique n'est à utiliser. Les critères généraux de sélection d'objets NCO sont indiqués en [2].

11.7 Traitement et codes d'erreur spécifiques

Les erreurs sont traitées comme suit.

11.7.1 Utilisation non valide de messages du plan d'utilisateur

En cas de longueur non valide du paramètre UserData contenu dans le message UDataReq, le dispositif PUF reçoit le message UDisconnectInd.

11.7.2 Causes

Les valeurs de cause peuvent être spécifiées et retournées dans le paramètre V.110Cause.

Tableau 23 – Valeur du paramètre V.110Cause

Code de retour		Signification	Information d'erreur spécifique
Undefined	220	situation d'erreur indéfinie	absente
DiscNorm	241	déconnexion – état normal	absente
ConRejectTrans	244	connexion rejetée (état transitoire)	absente
ConRejectPerm	245	connexion rejetée (état permanent)	absente

11.8 Attributs statiques

11.8.1 Paramètres de l'ensemble statique d'attributs

Tableau 24 – Paramètres d'ensemble d'attributs du plan U (UAttributeSet)

Paramètre	Requis	Commentaire
UProtocol	O	voir 11.3.3
FlowControlMechanism	O	voir 11.3.8
FlowControlCharacters	O	voir 11.3.9
MomentNumber	O	voir 11.3.10
V.110BChannelDisconnection	O	voir 11.3.11

Remarque: ces paramètres ne peuvent être utilisés que pendant la création d'un objet NCO contenant des informations sur le plan de commande. Pour plus de détails, voir en [2] le sous-paragraphe "opération ACreateNCO".

Si des paramètres sont omis, le dispositif NAF doit utiliser les valeurs par défaut qui sont décrites dans l'Appendice I.

11.8.2 Contenu de l'ensemble statique d'attributs

nom:	U_V.110
UProtocol:	V.110a
FlowControlMechanism:	0
FlowControlCharacters:	17 19
V.110BChannelDisconnection:	0

Remplacée par une version plus récente

Appendice I

Configuration

Les sous-paragraphes suivants indiquent les valeurs paramétriques à utiliser par défaut en cas d'absence d'autres valeurs dans la liste des paramètres au moment de l'opération de création d'objet NCO.

I.1 Protocole ISO/CEI 7776

Tableau I.1 – Configuration du plan U pour le protocole ISO/CEI 7776

Paramètre	Valeur par défaut suggérée	Commentaire
L2FrameSize	128	
L2WindowSize	7	
L2ConnectionMode	Auto	
L2XID		paramètre non utilisé

I.2 Protocole PPP

Tableau I.2 – Configuration du plan U pour le protocole PPP

Paramètre	Valeur par défaut suggérée	Commentaire
paramètres de type LCP		
– unité maximale de réception	1500 (octets)	permet à un homologue de signaler la longueur maximale de paquet qui est acceptée en réception
– temporisation de redémarrage	3 (secondes)	temporisation avant réponse à un paquet de requête
– décompte de terminaison	2	comptage du nombre de requêtes de terminaison envoyées sans réponse
– décompte de configuration	10	comptage du nombre de requêtes de configuration envoyées sans réponse
– décompte avant échec	10	comptage du nombre de messages de non-acquittement de configuration reçus à l'état envoyé avant l'émission d'un message de rejet de configuration, en supposant que la configuration ne soit pas en train de converger.
– nombre magique	aucune	
– compression de protocole	aucune	
– compression des champs d'adresse et de commande	aucune	

Tableau I.2 – Configuration du plan U pour le protocole PPP *(fin)*

Paramètre	Valeur par défaut suggérée	Commentaire
protocole d'authentification		
– type de protocole d'authentification à fixer	aucune	permet d'utiliser les protocoles d'authentification PPP
– ID local		longueur et nom de l'identificateur d'homologue local
– mot de passe local		longueur et nom du mot de passe local
– liste des couples "ID/Mot de passe" d'homologues distants		liste des longueurs et noms des couples distants "ID/Mot de passe"
– algorithme		type d'algorithme CHAP utilisé
contrôle qualité ligne		
– période de suivi	aucune	valeur de l'intervalle maximal d'envoi d'informations
FCSAlternatives		valeur de format de séquence FCS à utiliser
SelfDescPadding	aucune	valeur du bourrage autodescriptif à utiliser
rappel automatique		
– utilisation du rappel automatique		
– numéro à rappeler	aucune	indique si l'option de rétroappel doit être activée
trames composites	aucune	indique si l'option de trames composites (CompoundFrame) doit être activée

I.3 Protocole SDLC

Tableau I.3 – Configuration du plan U pour le protocole SDLC

Paramètre	Valeur par défaut suggérée	Commentaire
mode de connexion SDLC	cas 1	rôle de station de liaison SDLC par défaut cas 1 – station de liaison secondaire cas 2 – station de liaison primaire
mode d'initialisation SDLC	cas 1	mode d'initialisation SDLC par défaut cas 1 – envoi/réponse selon mode de mise en réponse normale (SNRM) cas 2 – envoi du module SIM/RIM
adresse SDLC	0xC1	adresse par défaut de station SDLC
module SDLC	8	module de numérotation par défaut des trames SDLC
longueur de fenêtre SDLC	7	longueur par défaut des fenêtres SDLC
longueur de trame SDLC	265	longueur maximale par défaut des trames SDLC, à l'exclusion de l'en-tête et du postambule de liaison
temporisateurs SDLC		temporisateurs du protocole SDLC
– T1	2	valeur exprimée en secondes
– T2	1	valeur exprimée en secondes
– N2	10	nombre maximal de réémissions infructueuses

I.4 Protocole V.110

Tableau I.4 – Configuration du plan U pour le protocole V.110

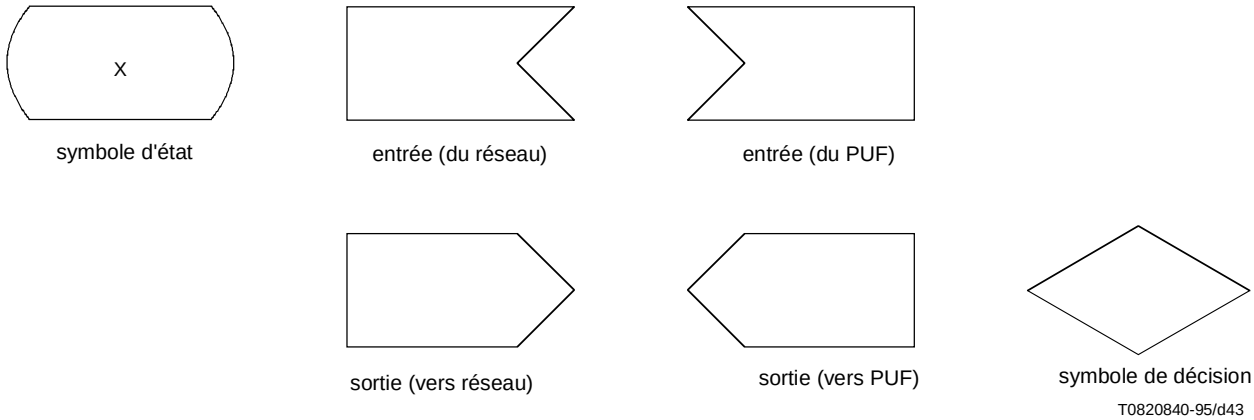
Paramètre	Valeur par défaut suggérée	Commentaire
temporisateur de perte du synchronisme	3 s	durée maximale jusqu'à resynchronisation
temporisateur de synchronisation	10 s	durée maximale jusqu'à synchronisation

Appendice II

Diagrammes SDL pour dispositifs NAF

Le mappage des messages du plan U avec les messages des protocoles est indiqué dans les tableaux suivants. Certains diagrammes SDL ou d'autres types de schémas sont présentés pour expliquer plus clairement la relation entre messages du plan U et primitives du réseau. Ces diagrammes ne couvrent pas tous les cas mais seulement certains de ceux qui peuvent se présenter.

Les symboles suivants seront utilisés dans le cadre de cette description. On trouvera dans la Recommandation Z.100 une description plus complète des symboles et de leur signification.



II.1 Protocole ISO/CEI 7776

Le Tableau II.1 montre le mappage entre les messages du plan d'utilisateur et les primitives du service ISO/CEI 7776.

Tableau II.1 – Mappage entre messages du plan U et messages du protocole ISO/CEI 7776

Interface PCI-RNIS	Protocole ISO/CEI 7776
UConnectReq	envoyer commande SABM(E)
UConnectInd	commande SABM(E) reçue
UConnectRsp	envoyer accusé UA
UConnectCnf	accusé UA reçu
UDisconnectReq	envoyer commande DISC
UDisconnectInd	commande DISC ou FRMR reçue
UDataReq	envoyer trame d'information (I)
UDataInd	trame I reçue
UReadyToReceiveReq (occupé)	envoyer signal RNR
UReadyToReceiveReq (libre)	envoyer signal RR
UReadyToReceiveInd (occupé)	signal RNR reçu
UReadyToReceiveInd (libre)	signal RR reçu
NOTE – Le dispositif NAF traite en transparence les trames de rejet (REJ) et la numérotation des trames.	

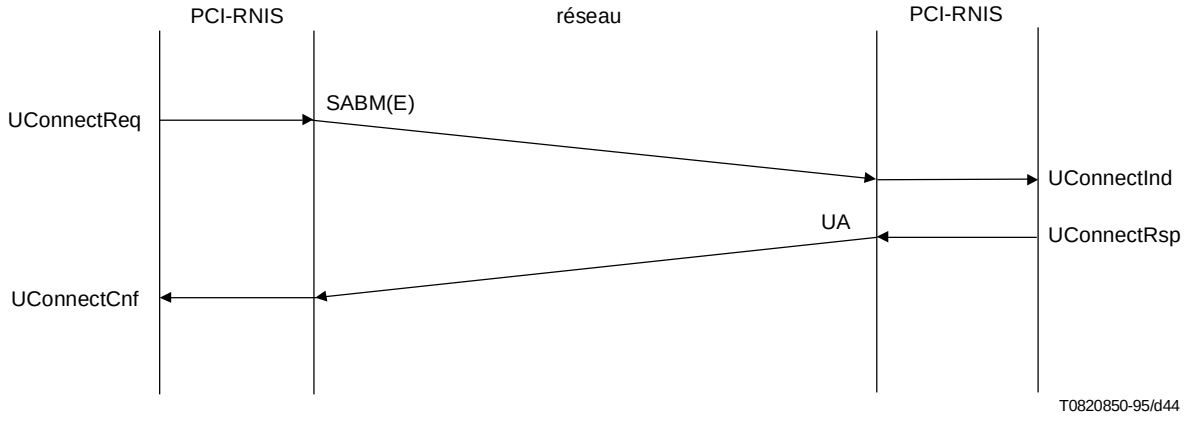


Figure II.1 – Phase de connexion

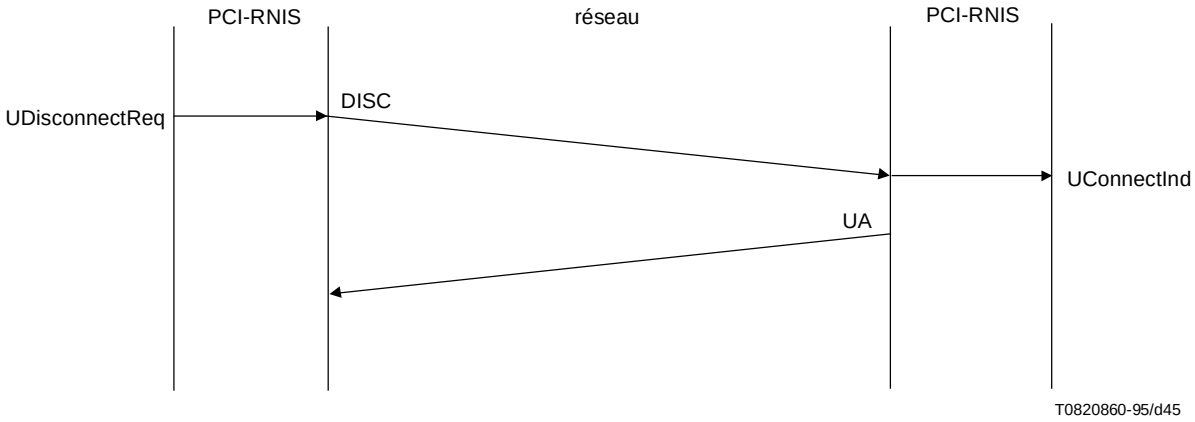


Figure II.2 – Phase de déconnexion

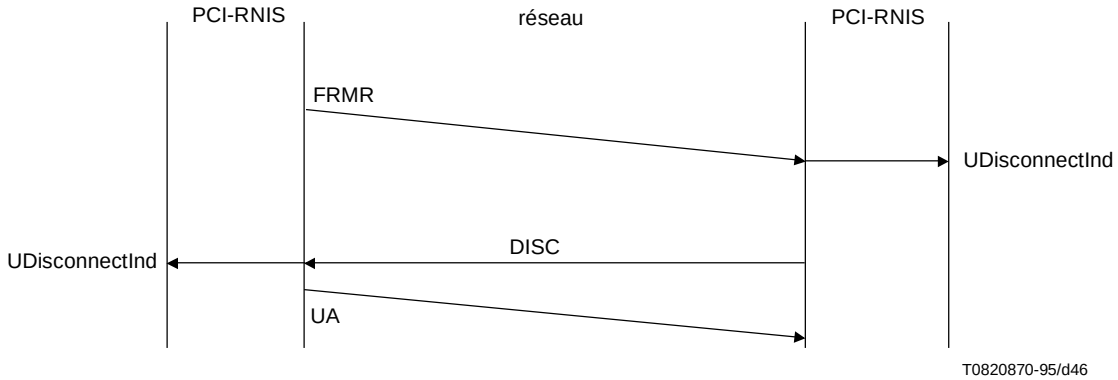


Figure II.3 – Situation d'erreur

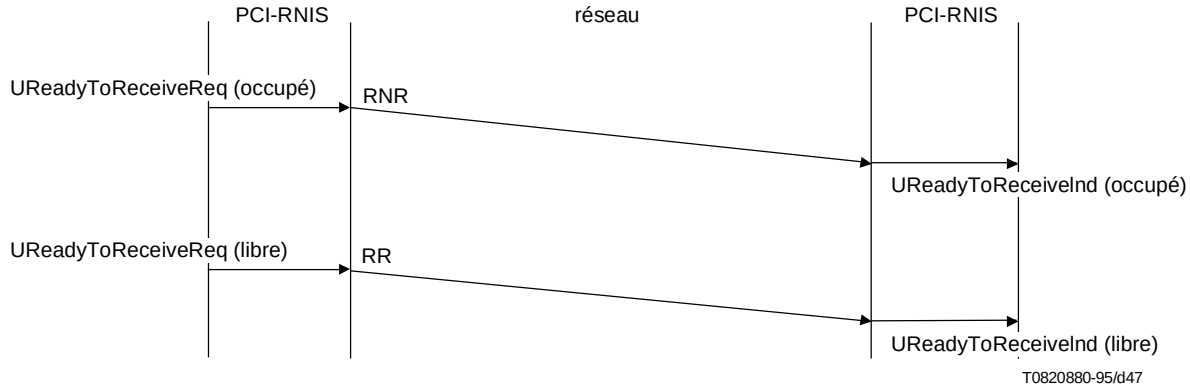


Figure II.4 – Situation de commande de débit

II.2 Protocole HDLC

Le Tableau II.2 montre le mappage entre les messages du plan U et les primitives du service HDLC.

Tableau II.2 – Mappage entre les messages du plan U et les messages du protocole HDLC

Message d'interface PCI	Primitive
UDataReq	I
UDataInd	I

II.3 Protocole HDLC avec indication d'erreur

Le Tableau II.3 montre les mappages entre les messages du plan U et les primitives de service du protocole HDLC avec indication d'erreur.

Tableau II.3 – Mappage entre les messages du plan U et les messages du protocole HDLC avec indication d'erreur

Message d'interface PCI	Primitive
UDataReq	I
UDataInd	I

II.4 Protocole PPP

Le Tableau II.4 montre le mappage entre les messages du plan U et les primitives du service PPP.

Tableau II.4 – Mappage entre les messages du plan U et ceux du protocole PPP

Message d'interface PCI	Primitive
UConnectReq	UI-CONFIGURE REQUEST
UConnectInd	UI-CONFIGURE REQUEST
UConnectRsp	UI-CONFIGURE ACK/NACK
UConnectCnf	UI-CONFIGURE ACK/NACK
UDisconnectReq	UI-TERMINATE REQUEST
UDisconnectInd	UI-CONFIGURE REJECT/ TERMINATE REQUEST
UDataReq	UI-INFO (NCP)
UDataInd	UI-INFO (NCP)
UErrorInd	UI-PROTOCOL REJECT CODE REJECT

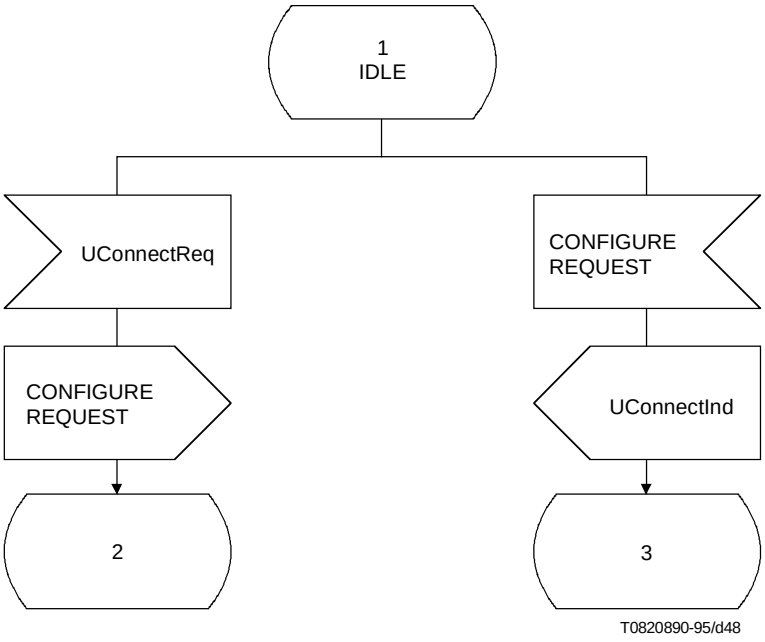


Figure II.5 – Etat de repos

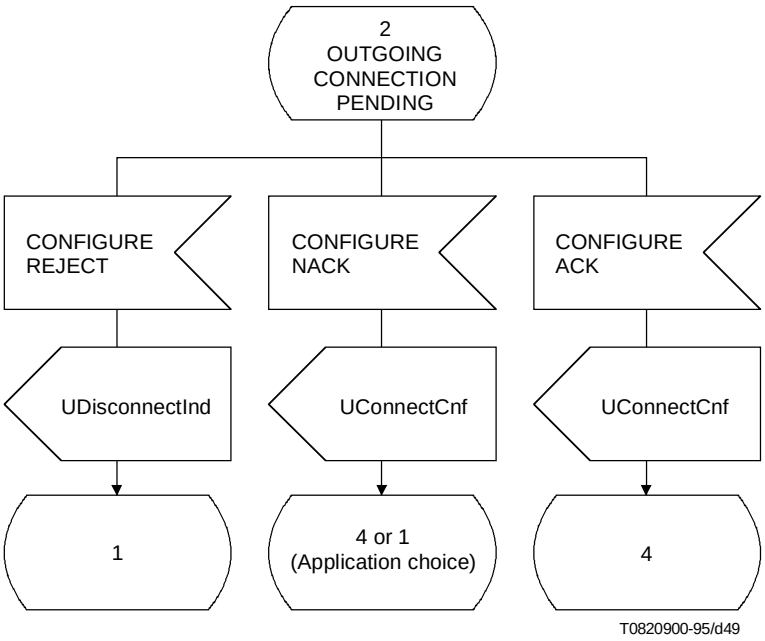


Figure II.6 – Connexion sortante en instance

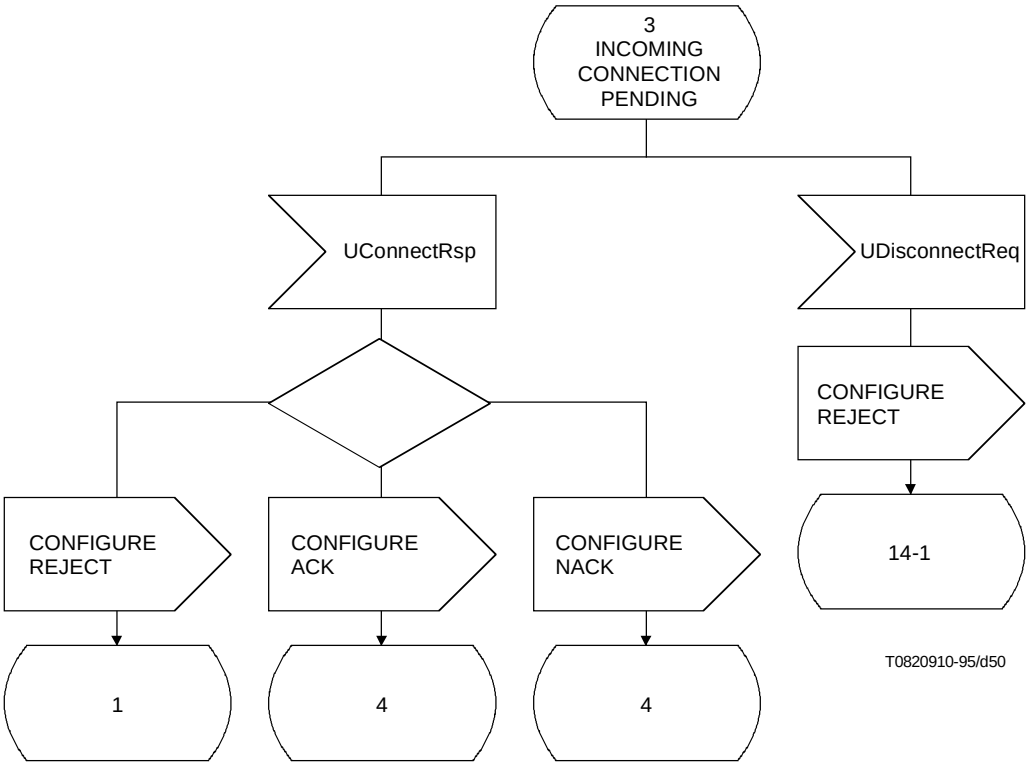


Figure II.7 – Connexion entrante en instance

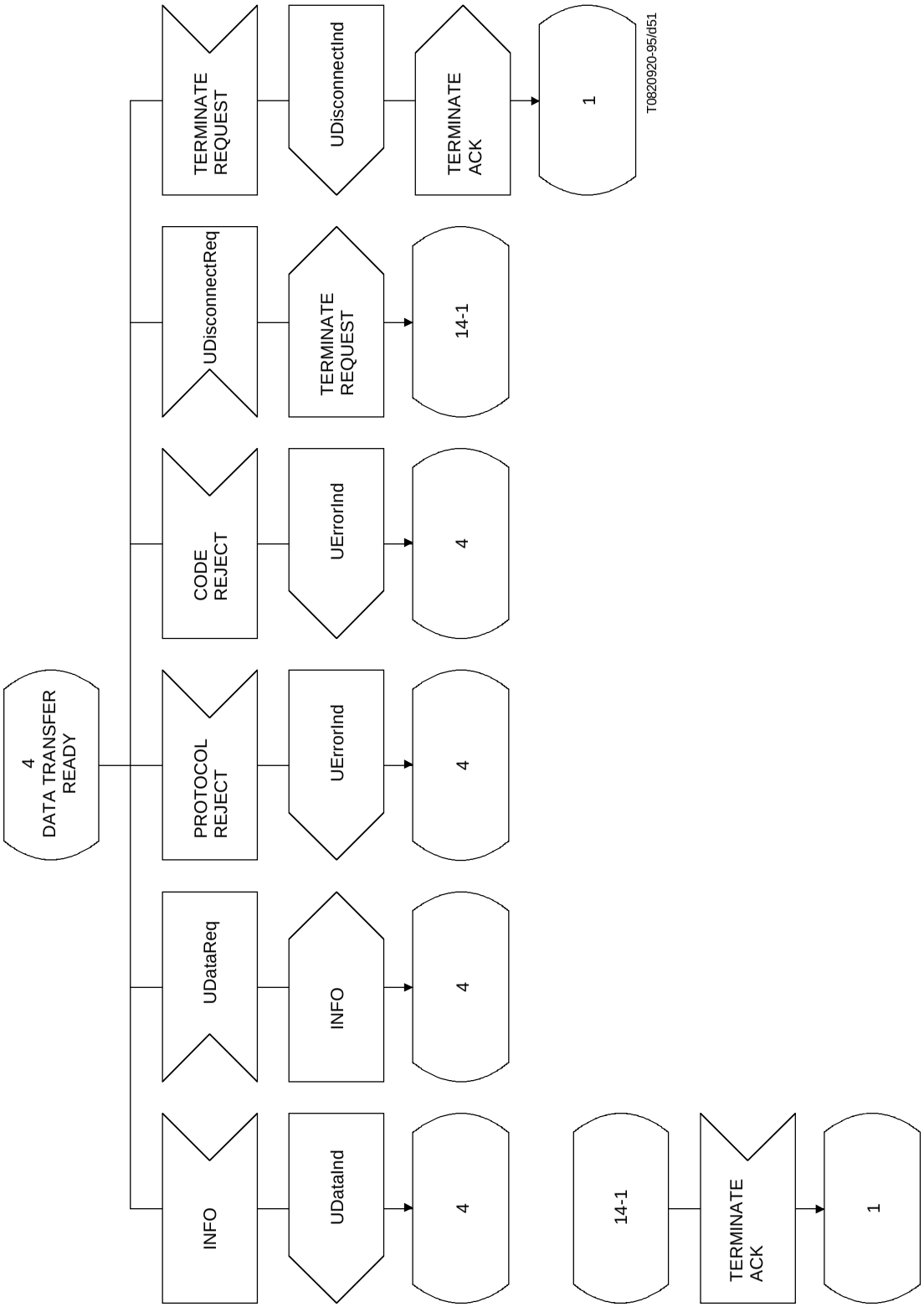


Figure II.8 – Prêt au transfert de données

II.5 Protocole SDLC

Le Tableau II.5 montre le mappage entre les messages du plan U et les primitives du service SDLC.

Tableau II.5 – Mappage entre les messages du plan U et ceux du protocole SDLC

Message de l'interface PCI	Primitive
UConnectReq	XID-P
UConnectInd	XID-P
UConnectRsp	XID-F
UConnectCnf	XID-F
UDisconnectReq	DISC-P
UDisconnectInd	RD-F ou DM-F
UDataReq	I
UDataInd	I
UExpeditedDataReq	UI
UExpeditedDataInd	UI
UReadyToReceiveReq	Portée locale
UReadyToReceiveInd	Portée locale

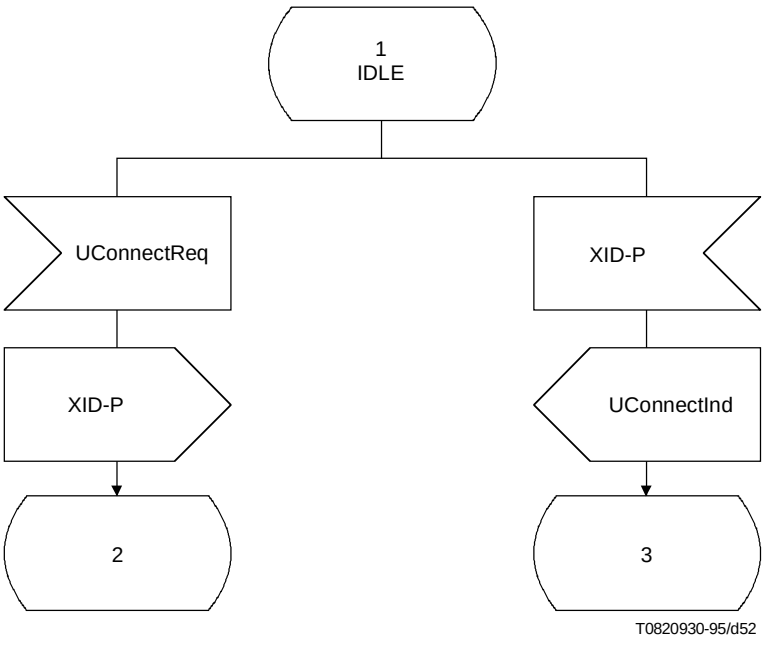


Figure II.9 – Etat de repos

Les états de type "Ix-y" sont des états intermédiaires.

Les états de type "Ox-y" sont des états facultatifs.

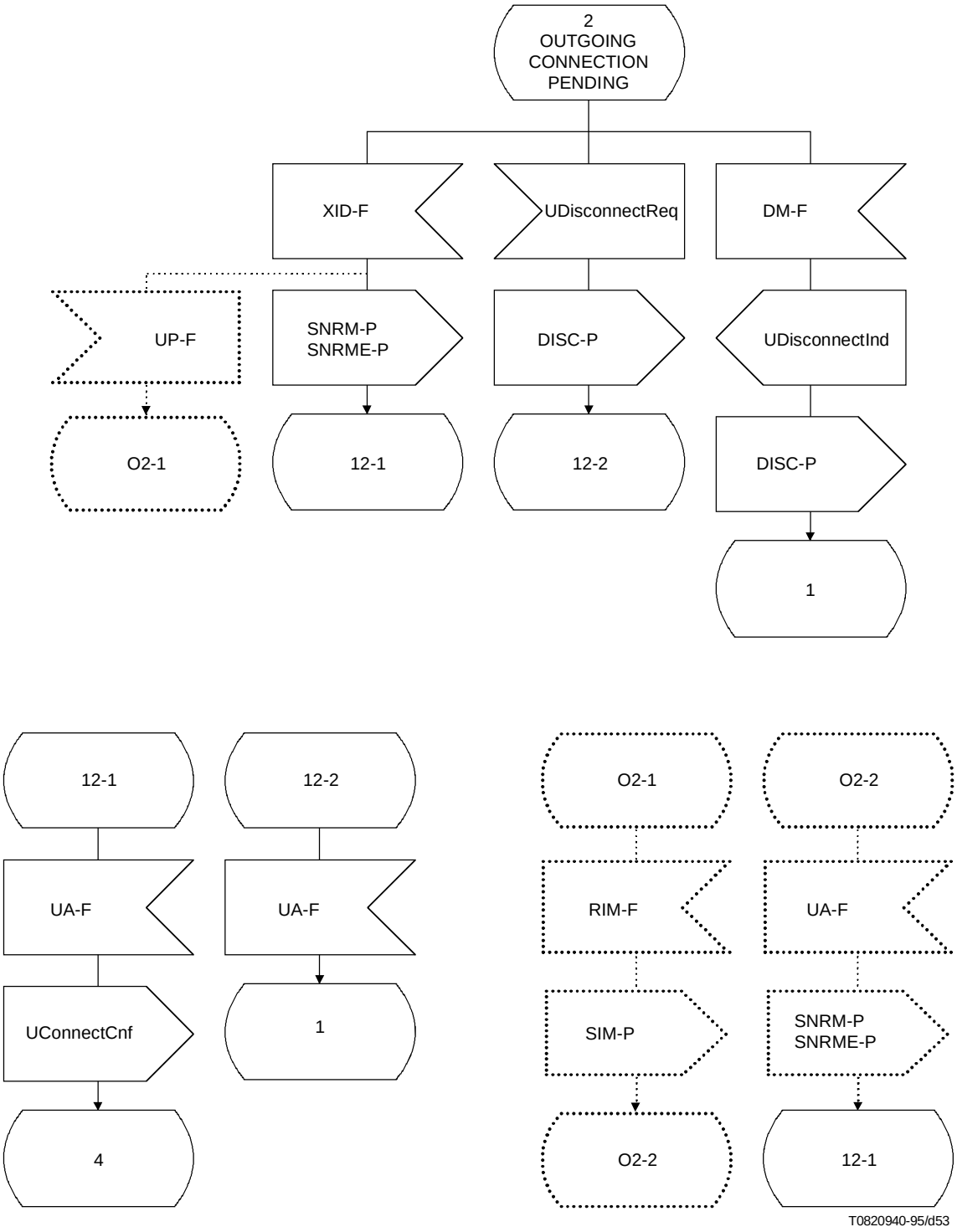


Figure II.10 – Connexion sortante en instance (liaison primaire)

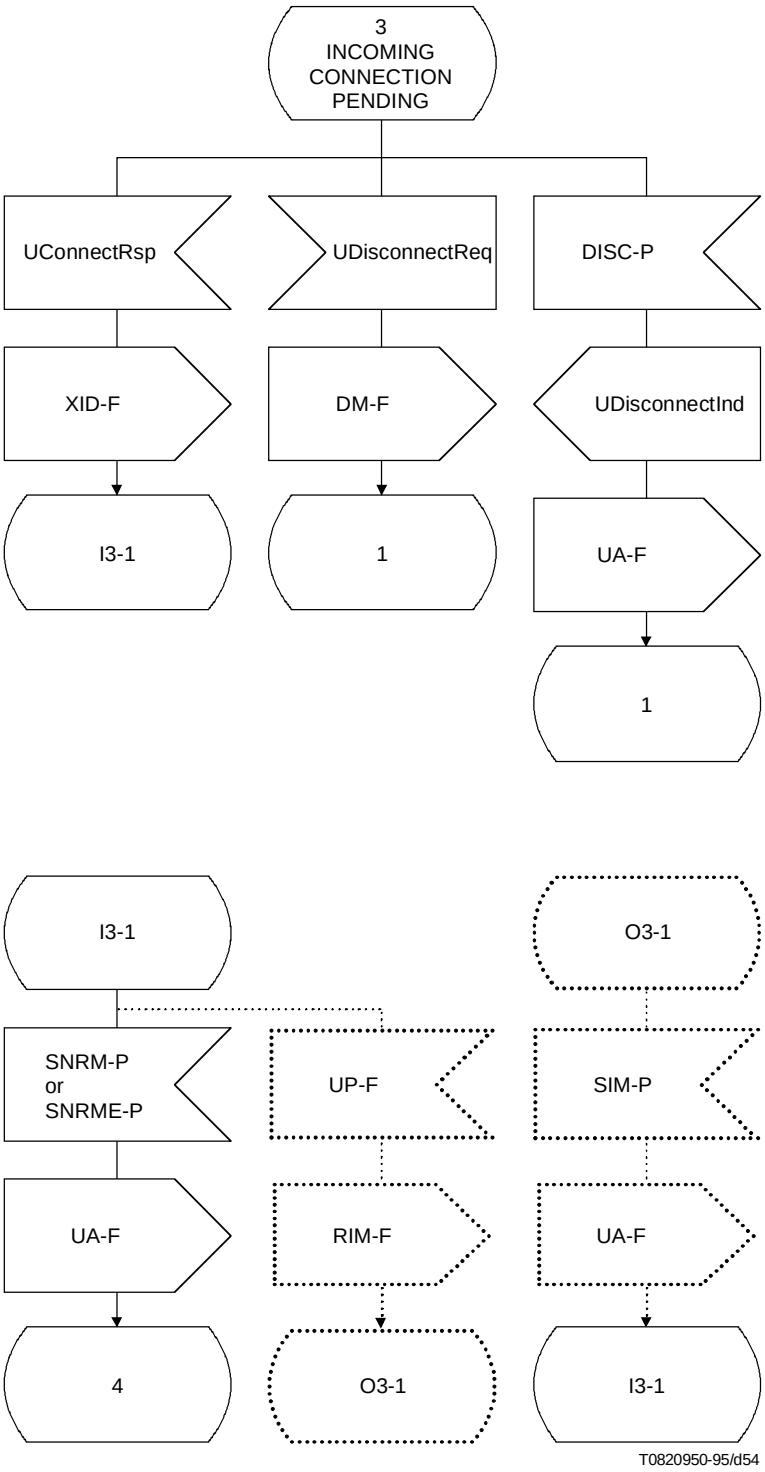


Figure II.11 – Connexion entrante en instance (liaison secondaire)

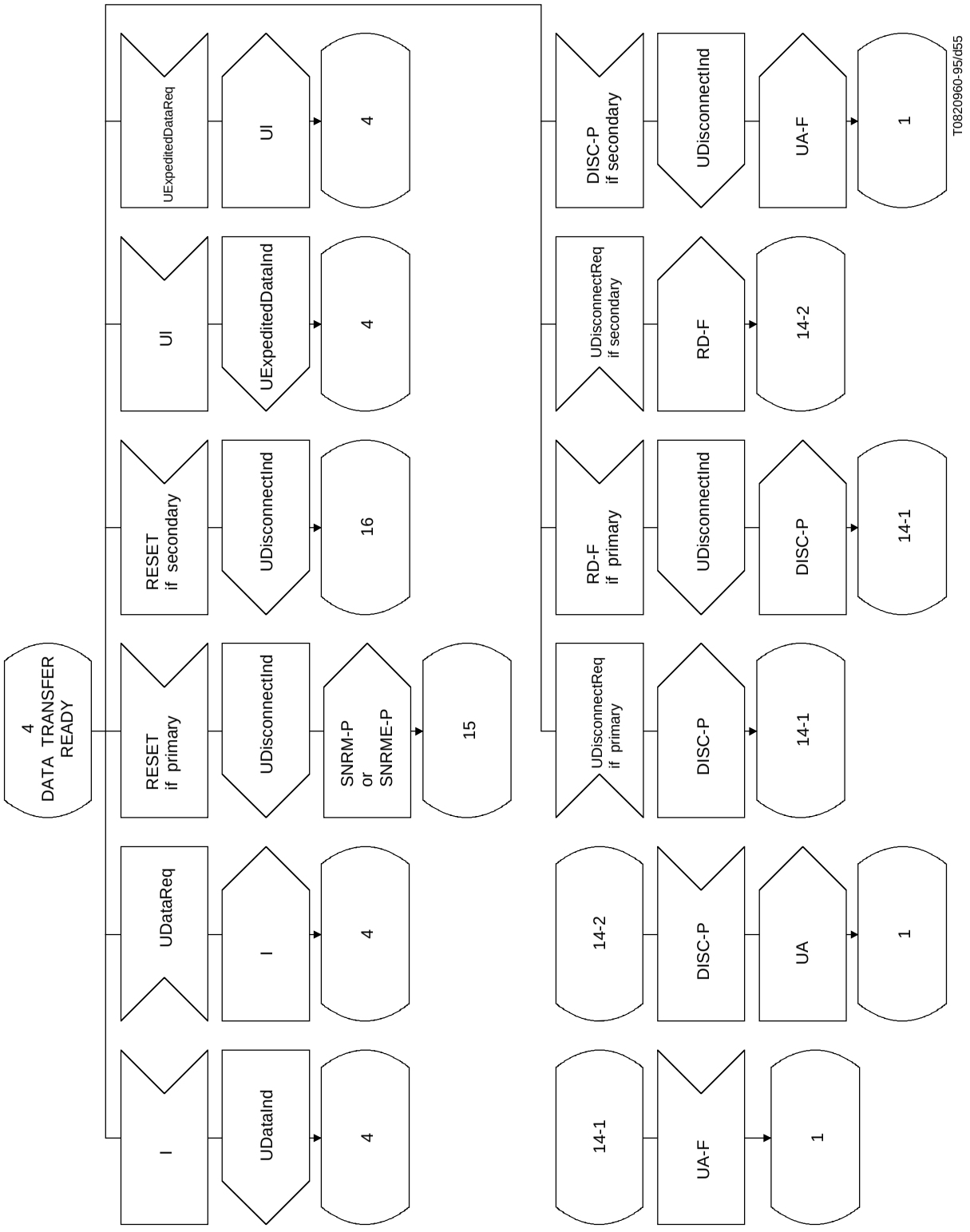


Figure II.12 – Prêt au transfert de données

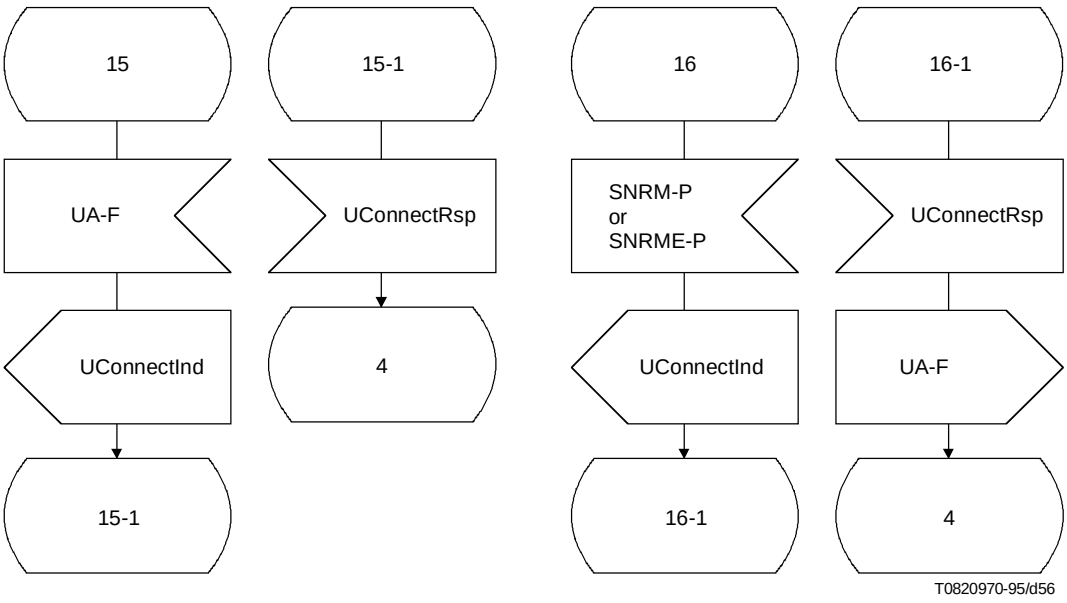


Figure II.13 – Réinitialisation

II.6 Protocole V.110

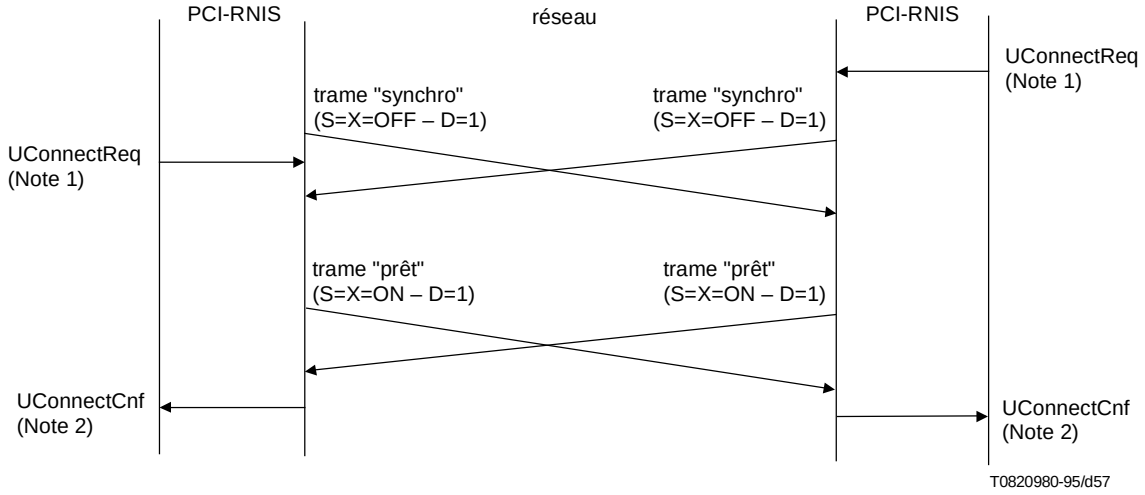
Le Tableau II.6 montre le mappage entre les messages du plan U et les éléments de service. Pour le protocole V.110, il n'y a pas de lien direct entre les messages des connexions d'utilisateur et les trames de protocole. La phase de connexion se compose d'une synchronisation et d'une négociation. Elle commence sans demande d'application, lorsque le canal RNIS est établi.

Pour faciliter la compréhension des messages de l'interface PCI-RNIS, le Tableau II.6 montre un mappage théorique entre les messages du plan U et les trames V.110.

Tableau II.6 – Mappage entre les messages du plan U et les trames V.110

Message d'interface PCI	Trame
UConnectReq	trame "synchronisation" (bit S = bit X = OFF) portée locale.
UConnectInd	trame "prêt" (bit S = bit X = ON) – portée locale: le signal de synchronisation distante a été reçu.
UConnectRsp	trame "synchronisation" (bit S = bit X = OFF) – portée locale. Note
UConnectCnf	trame "prêt" (bit S = bit X = ON) (un délai négocié peut être nécessaire avant que l'état soit "prêt au transfert")
UDisconnectReq	trame avec bit S = OFF, bit X = ON, D = 0
UDisconnectInd	trame avec bit S = OFF, bit X = ON, D = 0
UDataReq	trame de données
UDataInd	trame de données
UReadyToReceiveReq	portée locale
UReadyToReceiveInd	portée locale
NOTE – L'envoi d'un message UConnectResponse n'implique pas que la négociation soit terminée. Le transfert de données peut rester indisponible pendant un certain temps. Dans ce cas, le dispositif PUF reçoit un message d'erreur de type NAFBusy (voir la Figure II.15).	

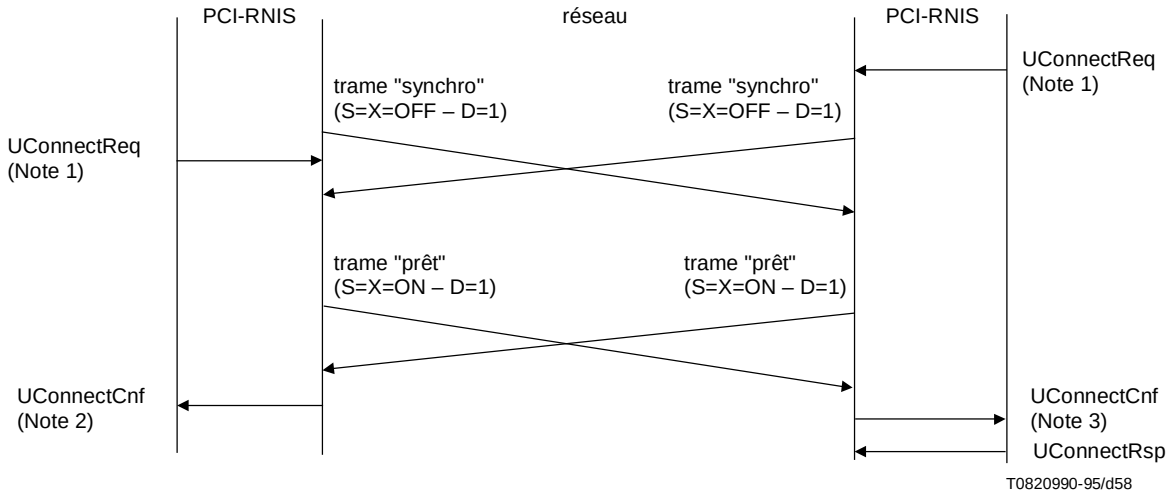
Les Figures II.14 à II.19 montrent des cas plus généraux, mais ne couvrent pas toutes les situations possibles.



NOTE 1 – Le message UConnectReq n'est pas vraiment associé à la trame "synchro"; il peut être émis avant ou après elle.

NOTE 2 – Le message UConnectCnf signifie que les deux extrémités sont en synchronisme.

Figure II.14 – Phase de connexion



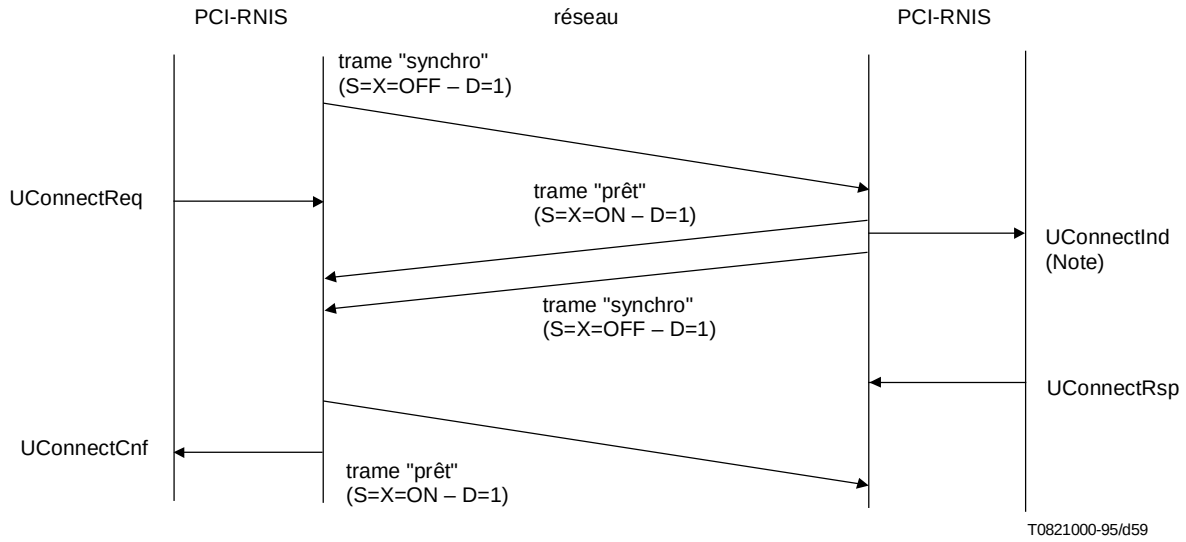
NOTE 1 – Le message UConnectReq n'est pas vraiment associé à la trame "synchro"; il peut être émis avant ou après elle.

NOTE 2 – Le message UConnectCnf signifie que les deux extrémités sont en synchronisme.

NOTE 3 – Le message UConnectInd signifie que les deux extrémités sont en synchronisme.

Figure II.15 – Phase de connexion

La Figure II.16 montre une autre situation possible. Il s'agit d'un cas théorique: le module V.110 de couche inférieure commence généralement la phase de synchronisation dès que le canal B est établi.



NOTE – Le message UConnectInd signifie que "l'autre extrémité va être prête".

Figure II.16 – Phase de connexion

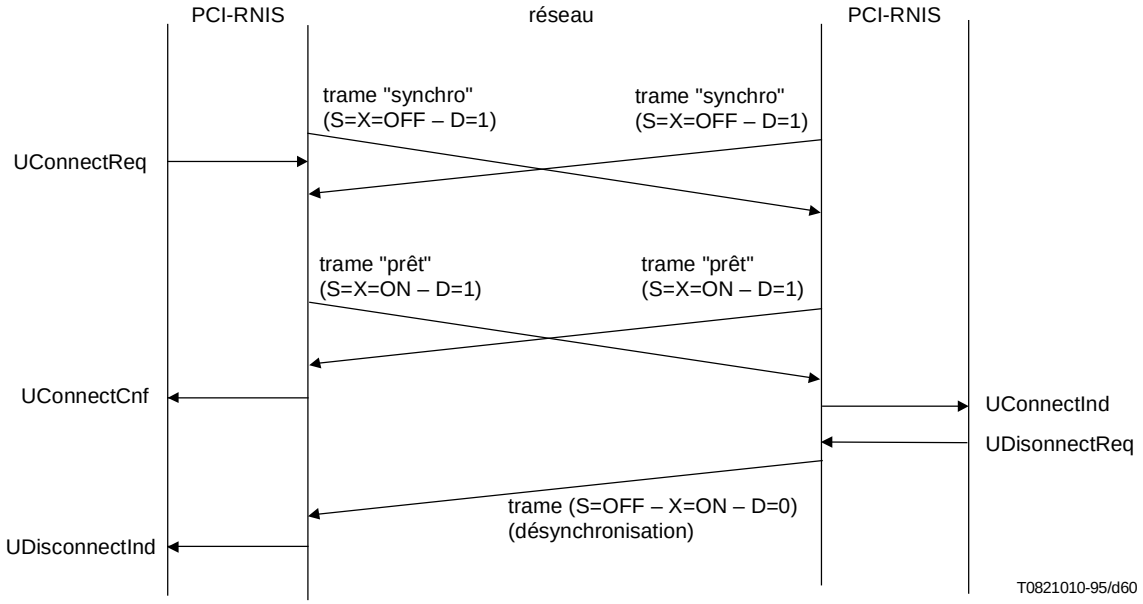


Figure II.17 – Déconnexion distante au cours de la phase de connexion

NOTE – Selon les valeurs du paramètre V.110 BChannelDisconnection, une déconnexion par utilisateur peut impliquer la déconnexion du canal B.

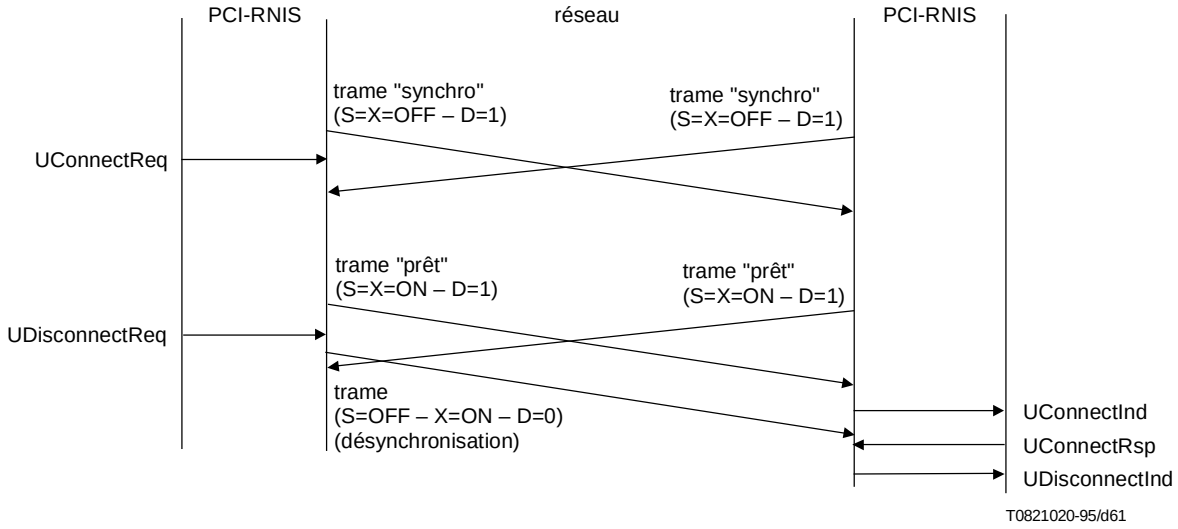


Figure II.18 – Déconnexion au cours de la phase de connexion

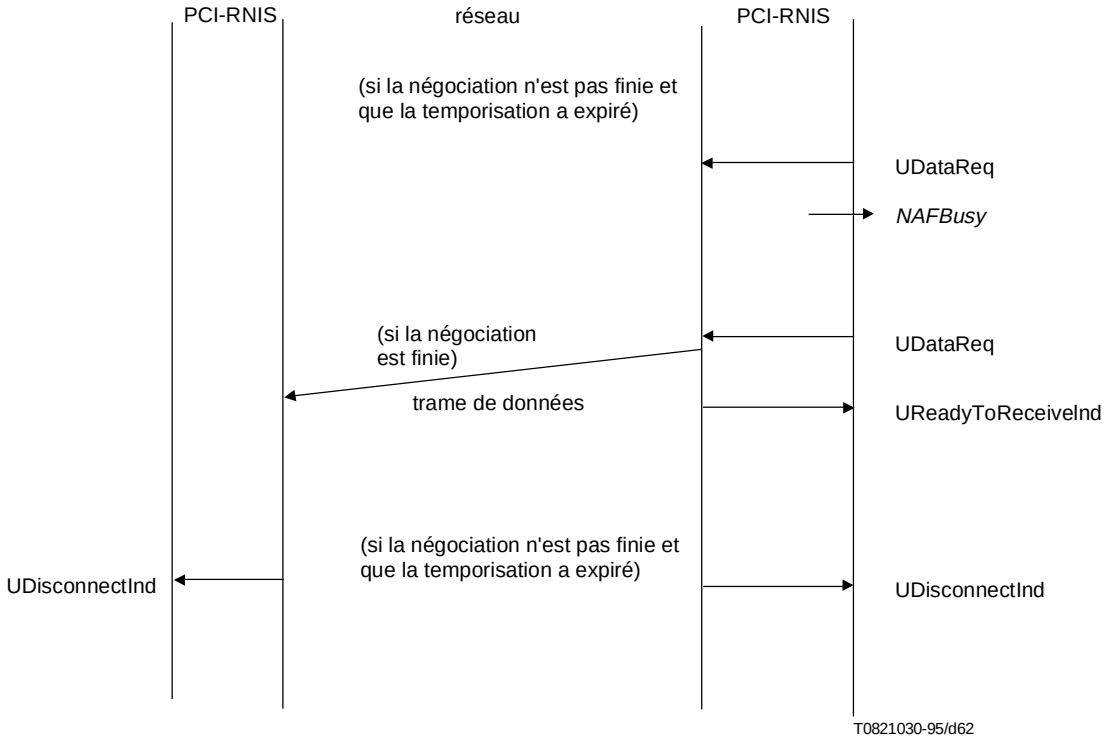


Figure II.19 – Transfert de données

Remplacée par une version plus récente

TABLE DES MATIÈRES

PARTIE 6

PARTIE 6: PROTOCOLES DE COUCHE TROIS

Résumé

La présente partie de la spécification décrit les procédures, messages et paramètres utilisés pour accéder aux protocoles du plan d'utilisateur qui assurent un service de communication dans la couche 3.

Introduction

L'utilisation de différentes interfaces de programmation par des équipements terminaux connectés au réseau numérique à intégration de services (RNIS) a fait obstacle au développement d'applications communes utilisant le RNIS et a limité le déploiement d'applications RNIS sur des équipements terminaux tels que les ordinateurs personnels.

La présente interface entre programmes d'application (API, *application programming interface*), appelée interface de programmation de communication (PCI, *programming communication interface*) par réseau numérique à intégration de services (PCI-RNIS), établie par l'UIT-T, permet aux applications d'accéder aux services des RNIS et de les gérer. L'interface PCI-RNIS fait l'objet d'une série de spécifications, dont la présente partie constitue la description de l'usage des protocoles de couche 3.

L'interface PCI-RNIS a été définie de façon à offrir aux fournisseurs d'équipement terminal une norme qui permettra de réaliser la portabilité d'applications utilisant l'interface PCI-RNIS sur une gamme d'équipements terminaux tournant sur différents systèmes d'exploitation.

L'interface PCI-RNIS a été définie en fonction des besoins des développeurs d'applications et, dans la mesure du possible, élimine la nécessité d'une connaissance approfondie du RNIS. Elle a également été conçue de façon que les futures extensions du RNIS n'aient pas d'incidence sur le fonctionnement des applications existantes.

1 Domaine d'application

La présente partie décrit les protocoles de couche 3 offerts par l'interface de programmation de communication pour réseau numérique à intégration de services (PCI-RNIS). Elle fait partie de la série des spécifications PCI-RNIS.

Cette partie décrit les éléments spécifiques (messages, paramètres, ensembles d'attributs, etc.) qui se rapportent aux protocoles du plan U dans la couche 3. La présente partie traite des protocoles suivants du plan d'utilisateur: T.90, ISO/CEI 8208 et T.70NL. La description d'autres protocoles utilisateurs de couche 3 fera l'objet de futures Recommandations UIT-T.

2 Références

[1] Recommandation UIT-T T.90 (1992), *Caractéristiques et protocoles des terminaux applicables aux services de télématique dans le RNIS.*

[2] ISO/CEI 8208:1995, *Technologies de l'information – Communication de données – Protocole X.25 de couche paquet pour terminal de données.*

[3] Recommandation UIT-T X.213 (1995), *Technologies de l'information – Définition du service de réseau pour l'interconnexion de systèmes ouverts.*

[4] Partie 1, *Architecture générale.*

[5] Partie 2, *Services de base.*

[6] Partie 3, *Architecture de gestion des protocoles du plan d'utilisateur.*

[7] Recommandation UIT-T X.31 (1995), *Prise en charge des équipements terminaux en mode paquet par un RNIS*.

3 Définitions

La présente partie définit les termes suivants:

3.1 ensemble d'attributs: ensemble de paramètres nécessaires au fonctionnement des protocoles d'utilisation et à la signalisation RNIS.

3.2 canal B: voie logique RNIS utilisée pour le transfert de données.

3.3 plan de commande: groupement logique de fonctions pour l'accès à la signalisation RNIS.

3.4 canal D: voie logique RNIS utilisée pour la signalisation et, dans certains cas, pour le transfert de données.

3.5 accès RNIS: ensemble de canaux RNIS fourni par un même dispositif d'accès réseau (NAF) pour l'accès aux services RNIS.

3.6 interface RNIS de programmation de communication (PCI-RNIS): interface logicielle orientée réseau (RNIS) qui offre des possibilités de programmer la signalisation de réseau et l'échange de données d'utilisateur.

3.7 message: unité d'information transférée de part et d'autre de l'interface PCI-RNIS, entre le dispositif d'accès réseau (NAF) et le dispositif utilisateur d'interface PCI (PUF).

3.8 dispositif d'accès réseau (NAF): unité fonctionnelle située entre l'interface PCI-RNIS et les couches associées au réseau.

3.9 objet de connexion (au) réseau (NCO): objet abstrait contenu dans le dispositif NAF, qui doit être créé par le dispositif PUF pour donner accès à la signalisation ou aux données du réseau.

3.10 couche vide: couche vide dans le modèle de référence OSI. Une telle couche ne contient aucune fonction et transmet en transparence les requêtes et les réponses aux couches suivantes.

3.11 dispositif utilisateur d'interface PCI (PUF): unité fonctionnelle faisant appel à l'interface PCI-RNIS pour accéder à un dispositif NAF. Ce terme correspond pratiquement à l'application locale qui utilise l'interface.

3.12 connexion d'utilisateur: connexion accessible par l'intermédiaire de la propriété plan d'utilisateur.

3.13 plan d'utilisateur: groupement logique de fonctions d'accès offertes aux protocoles et aux données d'utilisateur.

3.14 protocole utilisateur: protocole exploité conformément à la propriété de plan d'utilisateur.

4 Abréviations

Pour les besoins de la présente partie, les abréviations suivantes s'appliquent:

API	interface de programmation d'application (*application programming interface*)
LAPB	procédure d'accès à la liaison en mode équilibré (*link access procedure balanced*)
LAPD	procédure d'accès à la liaison sur le canal D (*link access procedure for D-channel*)
NAF	dispositif d'accès réseau (*network access facility*)
NCO	objet de connexion au réseau (*network connection object*)
NSAP	point d'accès aux services de la couche réseau (*network layer – service access point*)
PCI	interface de programmation de communication (*programming communication interface*)
PUF	dispositif utilisateur d'interface PCI (*PCI user facility*)
RNIS	réseau numérique à intégration de services
SAP	point d'accès au service (*service access point*)
X.25 PLP	protocole de couche Paquet X.25 (*X.25 packet layer protocol*)

5 Guide de lecture

5.1 Guide du lecteur

La présente partie est destinée aux développeurs de logiciels, aux réalisateurs d'applications et aux constructeurs d'équipement, qui y trouveront la description de l'usage général des protocoles d'utilisateur dans la couche 3.

5.2 Mode d'emploi de la présente partie

Les lecteurs qui:

- ont besoin d'un aperçu général rapide des protocoles du plan d'utilisateur le trouveront dans la Partie 3;
- envisagent d'implémenter une application au moyen d'un protocole d'utilisateur de couche 3 par l'interface PCI-RNIS auront avantage à lire la présente partie, dont les paragraphes 6 et 7 traitent l'usage de ce type de protocole;
- ont l'intention de construire une carte ou un équipement d'adaptation au RNIS utilisant l'interface PCI-RNIS pour un protocole de couche 3 auront avantage à lire la présente partie, dont les paragraphes 6 et 7 traitent l'usage de ce type de protocole et dont les Appendices I et II (informatifs) décrivent les valeurs de configuration par défaut ainsi que les diagrammes SDL pour dispositifs NAF.

Le Tableau 1 donne une liste qui décrit le contenu général de la présente partie.

Tableau 1 – Table des matières de la présente partie

Article, annexe, appendice	Contenu
paragraphe 1	domaine d'application de la présente partie, décrivant son objet
paragraphe 2	références
paragraphe 3	définitions de termes utilisés dans la présente partie
paragraphe 4	définitions d'abréviations utilisées dans la présente partie
paragraphe 5	aperçu général
paragraphe 6	protocole ISO/CEI 8208 et protocole T.90
paragraphe 7	protocole T.70
Appendice I	valeurs de configuration par défaut (pour information)
Appendice II	diagrammes SDL des dispositifs NAF (pour information)
Appendice III	informations sur l'utilisation du protocole X.25

La présente partie donne, pour chaque protocole supporté:

- la description des messages disponibles dans le plan U (voir le paragraphe 2);
- la description des paramètres utiles dans le plan U (voir le paragraphe 3);
- le diagramme de transition d'état (voir le paragraphe 4);
- les informations relatives à la fonction de coordination (voir le paragraphe 5);
- les critères de sélection d'éventuels objets NCO spécifiques (voir le paragraphe 6);
- le traitement et les codes d'erreur spécifiques (voir le paragraphe 7);
- la définition d'un ensemble d'attributs (voir le paragraphe 8).

L'Appendice I indique les valeurs par défaut éventuelles d'une configuration de protocole.

L'Appendice II montre, sous forme de diagrammes SDL, la plupart des situations d'un dispositif NAF.

6 Protocole ISO/CEI 8208 et protocole T.90

6.1 Introduction

Le présent paragraphe traite du protocole ISO/CEI 8208 [2] et du protocole T.90 [1]. La Figure 1 montre la localisation de ces protocoles dans le plan d'utilisateur (U).

On trouvera en [5] la description générale des conventions.

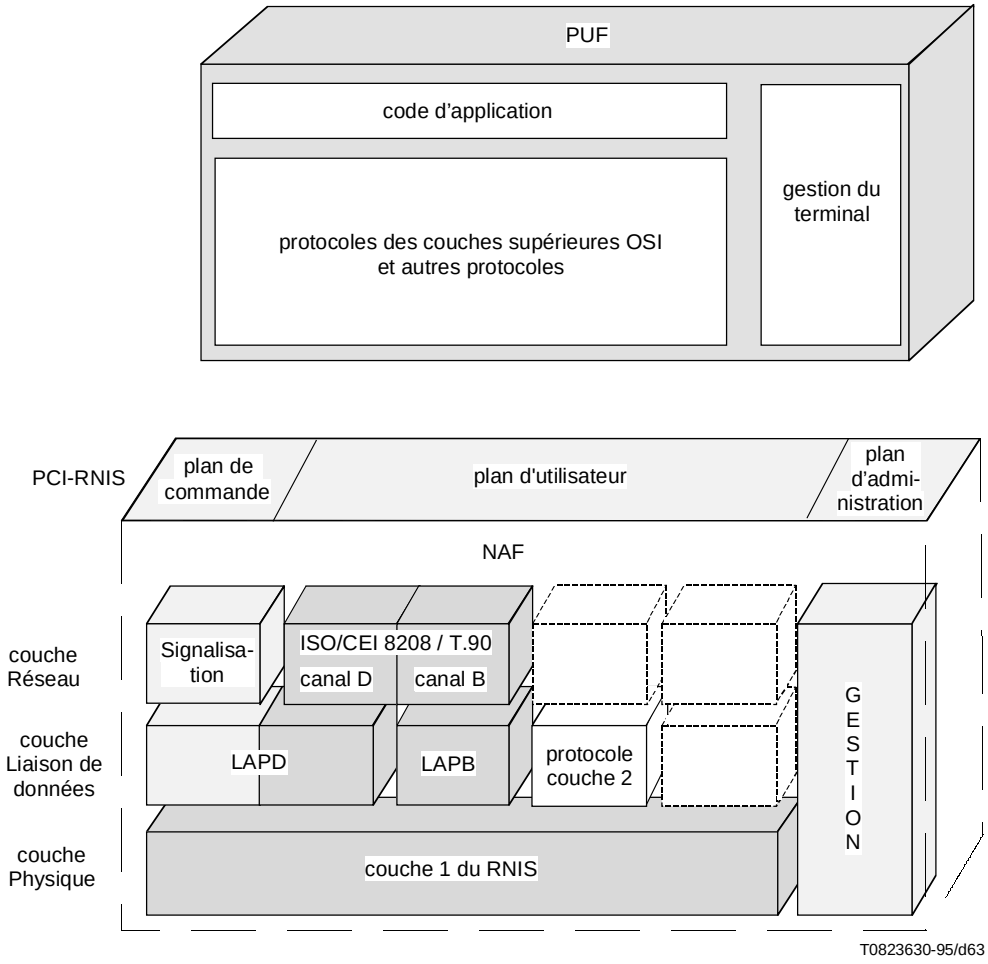


Figure 1 – Emplacement dans le modèle OSI

6.2 Description des messages

Les messages du plan d'utilisateur donnent accès, selon la Recommandation X.213, à des piles protocolaires conformes au protocole ISO/CEI 8208 ou T.90. Les paragraphes suivants énumèrent et décrivent brièvement les messages relevant du plan d'utilisateur. Le Tableau 2 donne un aperçu général de ces messages.

Tableau 2 – Aperçu général des messages dans le plan U

Identificateur du mess.	Classe	Nom du message	But du message	Utilisé pour ISO/CEI 8208	Utilisé pour T.90
301	1	UConnectReq	demande d'établissement d'une connexion dans le plan d'utilisateur	X	X
302	1	UConnectInd	indication d'une demande d'établissement de connexion dans le plan d'utilisateur	X	X
303	1	UConnectRsp	indication de l'acceptation d'établissement d'une connexion dans le plan d'utilisateur	X	X
304	1	UConnectCnf	confirmation de l'établissement d'une connexion dans le plan d'utilisateur	X	X
305	1	UDisconnectReq	demande de suppression d'une connexion du plan U	X	X
306	1	UDisconnectInd	indication de la suppression d'une connexion du plan U	X	X
307	1	UDataReq	demande de transfert de données sur une connexion d'utilisateur établie	X	X
308	1	UDataInd	indication de l'arrivée de données transférées sur une connexion d'utilisateur établie	X	X
309	1	UExpeditedDataReq	message effectuant la commande de débit pour une connexion d'utilisateur	X	
310	1	UExpeditedDataInd	message indiquant le statut de la commande de débit pour une connexion du plan U	X	
311	1	UResetReq	demande de remise à l'état initial d'une connexion d'utilisateur établie	X	X
312	1	UResetInd	indication de la remise à l'état initial d'une connexion d'utilisateur établie	X	X
313	1	UResetRsp	indication de l'acceptation de la remise à l'état initial d'une connexion d'utilisateur établie	X	X
314	1	UResetCnf	confirmation de l'acceptation de la remise à l'état initial d'une connexion d'utilisateur établie	X	X
315	1	UDataAcknowledgeReq	requête d'acquittement de réception de données sur une connexion d'utilisateur établie	X	
316	1	UDataAcknowledgeInd	indication d'acquittement de transfert de données sur une connexion d'utilisateur établie	X	
317	1	UReadyToReceiveReq	message servant à effectuer une commande de débit pour une connexion d'utilisateur	X	X
318	1	UReadyToReceiveInd	message indiquant le statut de la commande de débit sur une connexion d'utilisateur	X	X

6.2.1 UConnectReq

Classe: 1 (classe de base).

Description: ce message permet à un dispositif PUF de demander l'établissement d'une connexion d'utilisateur.

Paramètres:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion du plan U
CalledDTEAddress	O	si elle est fournie, cette valeur remplace celle du NCO.
CalledDTEAddressExt	O	si elle est fournie, cette valeur remplace celle du NCO.
CallingDTEAddress	O	si elle est fournie, cette valeur remplace celle du NCO.
CallingDTEAddressExt	O	si elle est fournie, cette valeur remplace celle du NCO.
ReceiptConfirm	O	sert à demander confirmation de réception de données pour cette connexion d'utilisateur
ExpeditedData	O	sert à demander l'utilisation de données exprès pour la connexion d'utilisateur
QOSParameters	O	qualité de service
UserData	O	longueur maximale: 16, ou 128 si le paramètre FastSelect est utilisé.
Bcug	O	utilisé pour spécifier le service de groupe fermé d'utilisateurs bilatéral (BCUG)
FastSelect	O	s'il est utilisé, ce paramètre invoque le service de sélection rapide.
PacketSize	O	valeur requise qui a priorité sur toute valeur spécifiée lors de la création de l'objet NCO
WindowSize	O	valeur requise qui a priorité sur toute valeur spécifiée lors de la création de l'objet NCO
FacilityData	O	utilisé pour fournir des services Si ce paramètre est présent, les services suivants doivent céder la priorité à des informations se trouvant ailleurs dans ce message: – BCUG; – sélection rapide; – extension de l'adresse de la ligne appelée; – extension de l'adresse de la ligne appelante.

Message associé: UConnectCnf.

Protocoles: ce message est utilisé dans les deux protocoles du plan d'utilisateur: T.90 et ISO/CEI 8208.

6.2.2 UConnectInd

Classe: 1 (classe de base).

Description: ce message informe un dispositif PUF d'une demande entrante visant à l'établissement d'une connexion d'utilisateur.

Paramètres:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion du plan U
CalledDTEAddress	O	adresse de la ligne appelée
CalledDTEAddressExt	O	extension de l'adresse de la ligne appelée
CallingDTEAddress	O	adresse de la ligne appelante
CallingDTEAddressExt	O	extension de l'adresse de la ligne appelante
ReceiptConfirm	O	paramètre utilisé pour accepter ou ne pas accepter l'emploi de la réception d'une confirmation pour des données sur cette connexion d'utilisateur
ExpeditedData	O	paramètre utilisé pour accepter ou ne pas accepter l'emploi de données exprès sur cette connexion d'utilisateur
QOSParameters	O	qualité de service
UserData	O	longueur maximale: 16, ou 128 si le paramètre FastSelect est présent.
Bcug	O	paramètre utilisé pour transmettre des informations relatives au service de groupe fermé d'utilisateurs bilatéral. S'il est présent, les informations d'adressages doivent être absentes.
FastSelect	O	type d'autorisation pour transmettre les données d'usager
PacketSize	M	utilisé pour indiquer une valeur agréée
WindowSize	M	utilisé pour indiquer une valeur agréée
FacilityData	O	utilisé pour fournir des services Les services suivants, si présents, seront présentés au moyen des paramètres spécifiques suivants: – BCUG; – FastSelect; – CalledAddressExt; – CallingAddessExt.

Message associé: UConnectRsp.

Protocoles: ce message est utilisé dans les deux protocoles du plan U: T.90 et ISO/CEI 8208.

6.2.3 UConnectRsp

Classe: 1 (classe de base).

Description: ce message permet à un dispositif PUF d'accepter l'établissement d'une connexion d'utilisateur.

Paramètres:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion du plan U
CalledDTEAddress	O	adresse de la ligne appelée
CalledDTEAddressExt	O	extension de l'adresse de la ligne appelée
CallingDTEAddress	O	adresse de la ligne appelante
CallingDTEAddressExt	O	extension de l'adresse de la ligne appelante
RespondingDTEAddress	O	adresse utilisée pour accepter la connexion d'utilisateur. Cette adresse peut être différente de l'adresse originale de la ligne appelée.
RespondingDTEAddressExt	O	extension d'adresse utilisée pour accepter la connexion d'utilisateur. Cette adresse peut être différente de l'adresse originale de l'extension de la ligne appelée.
ReceiptConfirm	O	paramètre utilisé pour accepter ou ne pas accepter l'emploi de la réception d'une confirmation pour des données sur cette connexion d'utilisateur
ExpeditedData	O	paramètre utilisé pour accepter ou ne pas accepter l'emploi de données exprès sur cette connexion d'utilisateur
QOSParameters	O	qualité de service
UserData	O	longueur maximale: 16, ou 128 si le paramètre FastSelect était présent dans le message UConnectInd.
PacketSize	O	utilisé pour indiquer une valeur agréée
WindowSize	O	utilisé pour indiquer une valeur agréée
FacilityData	O	utilisé pour fournir les différents services

Message associé: UConnectInd.

Protocoles: ce message est utilisé dans les deux protocoles du plan U: T.90 et ISO/CEI 8208.

6.2.4 UConnectCnf

Classe: 1 (classe de base).

Description: ce message informe le PUF de l'établissement d'une connexion d'utilisateur.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion du plan U
CalledDTEAddress	O	adresse de la ligne appelée
CalledDTEAddressExt	O	adresse de l'extension de la ligne appelée
CallingDTEAddress	O	adresse de la ligne appelante
CallingDTEAddressExt	O	adresse de l'extension de la ligne appelante
RespondingDTEAddress	O	adresse utilisée pour accepter la connexion d'utilisateur. Cette adresse peut être différente de l'adresse originale de la ligne appelée.
RespondingDTEAddressExt	O	extension d'adresse utilisée pour accepter la connexion d'utilisateur. Cette adresse peut être différente de l'adresse originale de la ligne appelée.
ReceiptConfirm	O	indique si la confirmation de réception de données peut être utilisée pour cette connexion d'utilisateur
ExpeditedData	O	indique si des données exprès peuvent être utilisées pour cette connexion d'utilisateur
QOSParameters	O	qualité de service
UserData	O	longueur maximale: 16, ou 128 si le paramètre FastSelect était présent dans le message UConnectReq.
PacketSize	M	valeur à utiliser pour cette connexion d'utilisateur
WindowSize	M	valeur à utiliser pour cette connexion d'utilisateur
FacilityData	O	utilisé pour fournir les différents services

Message associé: UConnectReq.

Protocoles: ce message est utilisé dans les deux protocoles du plan U: T.90 et ISO/CEI 8208.

6.2.5 UDisconnectReq

Classe: 1 (classe de base).

Description: ce message permet à un dispositif PUF de supprimer une connexion d'utilisateur.

Paramètres:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion du plan U
X213Cause	O	raison selon X.213 de supprimer la connexion d'utilisateur les deux paramètres X213Cause et X25Cause ne peuvent pas être utilisés dans le même message. Si ni le paramètre X213Cause ni le paramètre X25Cause n'est fourni, il faut utiliser le paramètre X213Cause avec la valeur de déconnexion en état normal.
RespondingDTEAddress	O	adresse utilisée pour accepter la connexion d'utilisateur. Cette adresse peut être différente de l'adresse originale de la ligne appelée.
RespondingDTEAddressExt	O	extension d'adresse utilisée pour accepter la connexion d'utilisateur. Cette adresse peut être différente de l'adresse originale de la ligne appelée.
UserData	O	autorisé seulement si le paramètre FastSelect a été spécifié au cours de l'établissement de la connexion d'utilisateur longueur maximale: 128 octets
X25Cause	O	raison de supprimer la connexion d'utilisateur les deux paramètres X213Cause et X25Cause ne peuvent pas être utilisés dans le même message
X25Diagnostic	C	information complémentaire pour la cause. Facultatif si le paramètre X25Cause a été fourni, pas autorisé dans le cas contraire.
FacilityData	O	utilisé pour fournir les différents services
NOTE – La cause X.213 et l'information X.25 s'excluent mutuellement. Si l'on utilise la cause X.25, facultativement associée au diagnostic X.25, la cause X.213 ne doit pas apparaître.		

Message associé: néant.

Protocoles: ce message est utilisé dans les deux protocoles du plan U: T.90 et ISO/CEI 8208.

6.2.6 UDisconnectInd

Classe: 1 (classe de base).

Description: ce message informe un PUF qu'une connexion d'utilisateur a été supprimée.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion du plan U
X213Origin	M	identifie l'initiateur de la suppression de la connexion d'utilisateur
X213Cause	O	raison selon X.213 de supprimer la connexion d'utilisateur
UserData	O	autorisé seulement si le paramètre FastSelect a été spécifié au cours de l'établissement de la connexion d'utilisateur longueur maximale: 128 octets
RespondingDTEAddress	O	adresse utilisée pour accepter la connexion d'utilisateur. Cette adresse peut être différente de l'adresse originale de l'extension de la ligne appelée.
RespondingDTEAddressExt	O	extension d'adresse utilisée pour accepter la connexion d'utilisateur. Cette adresse peut être différente de l'adresse originale de l'extension de la ligne appelée.
X25Cause	O	raison de supprimer la connexion d'utilisateur. Les deux paramètres X213Cause et X25Cause ne peuvent pas être utilisés dans le même message.
X25Diagnostic	C	information complémentaire pour la cause. Facultatif si le paramètre X25Cause a été fourni; sinon, pas autorisé.
FacilityData	O	utilisé pour fournir les différents services
NOTE – La cause X.213 et l'information X.25 s'excluent mutuellement. Si l'on utilise la cause X.25, facultativement associée au diagnostic X.25, la cause X.213 ne doit pas apparaître.		

Message associé: néant.

Protocoles: ce message est utilisé dans les deux protocoles du plan U: T.90 et ISO/CEI 8208.

6.2.7 UDataReq

Classe: 1 (classe de base).

Description: ce message permet à un dispositif PUF d'envoyer un paquet de données. La taille d'un paquet de données est limitée à la valeur négociée au cours de l'établissement de la connexion d'utilisateur.

Paramètres:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion du plan U
Bit_DQM	O	utilisé pour activer le bit qualificateur, la valeur du bit indicateur de données à suivre (continuation) et pour demander confirmation de réception de données.

Remarque: les données à envoyer sont obligatoires. Elles ne sont pas fournies sous forme de paramètre du message.

Les données obligatoires doivent être fournies dans le tampon de données.

Message associé: UReadyToReceiveInd.

Protocoles: ce message est utilisé dans les deux protocoles du plan U: T.90 et ISO/CEI 8208.

6.2.8 UDataInd

Classe: 1 (classe de base).

Description: ce message indique à un PUF la présence de données reçues. La taille d'un paquet de données est limitée à la valeur négociée au cours de l'établissement de la connexion d'utilisateur.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion du plan U
Bit_DQM	O	utilisé pour indiquer la valeur du bit qualificateur, la valeur du bit indicateur de données à suivre (continuation) et la nécessité de confirmer la réception de données.

Remarque: les données reçues sont toujours indiquées, mais non pas sous forme de paramètre du message.

Les données sont fournies dans le tampon de données. Ce tampon est, dans ce cas, obligatoire.

Message associé: UReadyToReceiveReq.

Protocoles: ce message est utilisé dans les deux protocoles du plan U: T.90 et ISO/CEI 8208.

6.2.9 UExpeditedDataReq

Classe: 1 (classe de base).

Description: ce message permet à un dispositif PUF d'envoyer des données exprès. Ces données ne sont pas sensibles au mécanisme de commande de débit utilisé pour régler les messages UDataReq.

Paramètres:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion du plan U
UserData	M	données exprès à transférer

Message associé: néant.

Protocole: ce message est utilisé dans le protocole ISO/CEI 8208 du plan U.

6.2.10 UExpeditedDataInd

Classe: 1 (classe de base).

Description: ce message indique à un PUF la réception de données exprès. Ces données n'ont pas été sensibles aux mécanismes de commande de débit utilisés pour régler les messages UDataInd.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion du plan U
UserData	M	données exprès reçues

Message associé: néant.

Protocole: ce message est utilisé dans le protocole ISO/CEI 8208 du plan U.

6.2.11 UResetReq

Classe: 1 (classe de base).

Description: ce message permet au PUF de réinitialiser une connexion d'utilisateur.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion du plan U
X213Cause	O	raison X.213 de réinitialiser la connexion d'utilisateur si ni le paramètre X213Cause ni le paramètre X25Cause n'est fourni, on utilisera le paramètre X213Cause avec la valeur "déconnexion – état normal".
X25Cause	O	raison de réinitialiser la connexion d'utilisateur
X25Diagnostic	C	information complémentaire. Facultatif seulement si le paramètre X25Cause a été fourni; sinon, non autorisé.
NOTE – La cause X.213 et l'information X.25 s'excluent mutuellement. Si l'on utilise la cause X.25, facultativement associée au diagnostic X.25, la cause X.213 ne doit pas apparaître.		

Message associé: UResetCnf.

Protocoles: ce message est utilisé dans les deux protocoles du plan U: T.90 et ISO/CEI 8208.

6.2.12 UResetInd

Classe: 1 (classe de base).

Description: ce message informe le PUF de la réinitialisation d'une connexion d'utilisateur.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion du plan U
X213Origin	M	identifie l'initiateur de la connexion U réinitialisée
X213Cause	O	raison selon X.213 de réinitialiser la connexion d'utilisateur
X25Cause	O	raison de réinitialiser la connexion d'utilisateur
X25Diagnostic	O	information complémentaire. Facultatif seulement si le paramètre X25Cause a été fourni; sinon, non autorisé.
NOTE – La cause X.213 et l'information X.25 s'excluent mutuellement. Si l'on utilise la cause X.25, facultativement associée au diagnostic X.25, la cause X.213 ne doit pas apparaître.		

Message associé: UResetRsp.

Protocoles: ce message est utilisé dans les deux protocoles du plan U: T.90 et ISO/CEI 8208.

6.2.13 UResetRsp

Classe: 1 (classe de base).

Description: ce message permet au PUF de répondre à une réinitialisation de connexion U, pour indiquer qu'il a traité cette réinitialisation et est prêt à continuer.

Paramètre:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion du plan U

Message associé: UResetInd.

Protocoles: ce message est utilisé dans les deux protocoles du plan U: T.90 et ISO/CEI 8208.

6.2.14 UResetCnf

Classe: 1 (classe de base).

Description: ce message termine l'opération de réinitialisation d'une connexion U. Le PUF est dès lors en mesure de transférer de nouvelles données.

Paramètre:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion du plan U

Message associé: UResetReq.

Protocoles: ce message est utilisé dans les deux protocoles du plan U: T.90 et ISO/CEI 8208.

6.2.15 UDataAcknowledgeReq

Classe: 1 (classe de base).

Description: ce message permet au PUF d'accuser réception de données. Il convient de l'utiliser lorsqu'on reçoit un message UDataInd avec le paramètre bit_DQM activé pour indiquer que la réception d'une confirmation est requise.

Paramètre:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion du plan U

Message associé: UDataInd.

Protocole: ce message est utilisé dans le protocole ISO/CEI 8208 du plan U.

6.2.16 UDataAcknowledgeInd

Classe: 1 (classe de base).

Description: ce message informe le PUF de la réception d'un accusé pour des données transférées. Il accuse réception d'un message UDataReq qui a été envoyé avec le paramètre bit_DQM demandant confirmation de la réception de données.

Paramètre:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion du plan U

Message associé: UDataReq.

Protocole: ce message est utilisé dans le protocole ISO/CEI 8208 du plan U.

6.2.17 UReadyToReceiveReq

Classe: 1 (classe de base).

Description: ce message permet au PUF d'indiquer au NAF s'il peut accepter des données entrantes (message UDataInd). Ce message ne peut s'appliquer qu'à une connexion d'utilisateur déjà établie. Le fait de mettre le paramètre ReadyFlag à la valeur Vrai permet au NAF de transférer des données entrantes vers le PUF. Le fait de mettre le paramètre ReadyFlag sur Faux empêche ce transfert.

Ce mécanisme de commande de débit n'implique pas un contrôle de flux de bout en bout.

Paramètres:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion du plan U
ReadyFlag	M	ce fanion indique si le PUF est ou non prêt à accepter des données entrantes

Remarque: pour une connexion donnée, si plus d'un seul message avec la même valeur de fanion est envoyé, ce message doit être ignoré par le NAF.

Message associé: UDataInd.

Protocoles: ce message est utilisé dans les deux protocoles du plan U: T.90 et ISO/CEI 8208.

6.2.18 UReadyToReceiveInd

Classe: 1 (classe de base).

Description: ce message permet au NAF d'indiquer au PUF si la connexion d'utilisateur permet l'envoi de données (messages UDataReq). Ce message ne peut s'appliquer qu'à une connexion d'utilisateur déjà établie. Si la valeur du paramètre ReadyFlag est Faux, le NAF ne peut pas envoyer de données. Si la valeur est Vrai, le NAF indique que le transfert de données est autorisé.

Ce mécanisme de commande de débit n'implique pas un contrôle de flux de bout en bout.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion du plan U
ReadyFlag	M	ce fanion indique si le NAF est prêt à recevoir des données pour transmission sur une connexion d'utilisateur

Message associé: UDataReq.

Protocoles: ce message est utilisé dans les deux protocoles du plan U: T.90 et ISO/CEI 8208.

6.3 Paramètres contenus dans les messages

Le présent sous-paragraphe décrit les paramètres pour les protocoles ISO/CEI 8208 et UIT-T T.90 du plan d'utilisateur. Ces paramètres sont présentés dans l'ordre alphabétique.

La présentation des informations est décrite en [5].

Tableau 3 – Aperçu général des paramètres du plan U

Identificateur du paramètre	Nom du paramètre	Usage dans les messages ISO/CEI 8208 du plan U	Usage dans les messages T.90 du plan U	Usage dans l'ensemble UAttributeSet	Autre usage
1	Algorithm				X
2	Bilateral closed user group	X	X		
4	Bit_DQM	X	X (Note)		
5	CalledDTEAddress	X	X		
6	CalledDTEAddressExt	X	X		
9	CallingDTEAddress	X	X		
10	CallingDTEAddressExt	X	X		
29	ExpeditedData	X			
31	FacilityData	X	X (Note)		
32	FastSelect	X	X (Note)	X	
33	GroupID				X
38	L2ConnectionMode			X	
39	L2FrameSize			X	
40	L2WindowSize			X	
41	L2XID			X	
42	L3ConnectionMode			X	
43	L3IncomingCount			X	
44	L3OutgoingVCCount			X	

Tableau 3 – Aperçu général des paramètres du plan U *(fin)*

Identificateur du paramètre	Nom du paramètre	Usage dans les messages ISO/CEI 8208 du plan U	Usage dans les messages T.90 du plan U	Usage dans l'ensemble UAttributeSet	Autre usage
45	L3TwoWayCount			X	
50	NCOType				X
52	PacketSize	X	X	X	
54	QOSParameters	X	X	X	
55	ReadyFlag	X	X		
57	ReceiptConfirm	X			
58	RespondingDTEAddress	X	X		
59	RespondingDTEAddressExt	X	X		
61	TEI			X	
62	UProtocol			X	
63	UAttributeName				X
64	UDirection				X
65	UserData	X			
67	WindowSize	X	X	X	
68	X213Cause	X	X		
69	X213Origin	X	X		
70	X25Cause	X	X		
71	X25Diagnostic	X	X		
NOTE – Ce paramètre doit être utilisé conformément aux règles de la Recommandation T.90 [1].					

6.3.1 Algorithm

Description: ce paramètre sert à transmettre au NAF le nom de l'algorithme de sécurité à utiliser.

Type: 1.

Champ	Type de champ	Sens	Requis	Commentaire
Algorithm	chaîne IA5	P	M	l'algorithme de sécurité est identifié par son nom. Les noms des algorithmes disponibles peuvent être obtenus au moyen des informations relatives à la propriété. "nosecurity": pour ce paramètre, cette valeur indique que la sécurité n'est plus nécessaire pour la connexion. 16 est la longueur maximale

6.3.2 Groupe fermé d'utilisateurs bilatéral (Bcug, *bilateral closed user group*)

Description: ce paramètre sert à transmettre des informations relatives à un groupe fermé d'utilisateurs bilatéral à destination/en provenance du PUF.

Type: 2.

Champ	Type de champ	Sens	Requis	Commentaire
Bcug	chaîne d'octets	B	M	pointeur sur un groupe fermé d'utilisateurs bilatéral sélectionné pour la connexion d'utilisateur 4 est la longueur fixée.

6.3.3 Bit_DQM

Description: ce paramètre sert à transmettre à destination/en provenance du PUF:

– le besoin de recevoir des données (bit 1). Ce bit est équivalent au Bit D X.25;

– la valeur du bit qualificateur (bit 2);

– la valeur du bit de données à suivre (bit 3).

Chaque information utilise une position binaire. Le bit de plus fort poids (MSB, *most significant bit*) est le bit 8 et le bit de plus faible poids (LSB, *least significant bit*) est le bit 1. Le bit 1 représente la valeur 1, le bit 2 représente la valeur 2 et le bit 3 représente la valeur 4. La valeur de résultat applicable à ce paramètre est la somme des valeurs de chaque élément binaire (OU logique).

Type: 4.

Champ	Type de champ	Sens	Requis	Commentaire
DQM	octet	B	M	bit 1: 1 – la confirmation de la réception de données est autorisée ou requise 0 – la confirmation de la réception de données n'est pas autorisée ou n'est pas requise bit 2: 1 – activation du bit qualificateur 0 – désactivation du bit qualificateur bit 3: 1 – activation du bit de données à suivre 0 – désactivation du bit de données à suivre

Remarques: l'utilisation non valide de la valeur du bit indicateur de données à suivre (continuation) associé au bit qualificateur aura pour résultat la réinitialisation de la connexion d'utilisateur.

Pour le protocole T.90, ce paramètre doit être utilisé conformément aux règles de la Recommandation T.90 [1].

6.3.4 CalledDTEAddress

Description: ce paramètre sert à transmettre des informations d'adresse d'ETTD distant à destination/en provenance du PUF.

Type: 5.

Champ	Type de champ	Sens	Requis	Commentaire
Address	chaîne IA5	B	M	15 octets est la longueur maximale

Remarque: la conversion en codage BCD est assurée par le NAF.

Dans le transfert de messages du PUF au NAF, ce paramètre doit être fourni soit dans l'objet NCO ou dans le message approprié.

6.3.5 CalledDTEAddressExt

Description: ce paramètre sert à transmettre des informations d'extension d'adresse d'ETTD distant à destination/en provenance du PUF.

Type: 6.

Champ	Type de champ	Sens	Requis	Commentaire
AddressExt	chaîne IA5	B	M	40 octets est la longueur maximale

Remarque: la conversion en codage BCD est assurée par le NAF.

6.3.6 CallingDTEAddress

Description: ce paramètre sert à transmettre des informations d'adresse relatives à l'ETTD local à destination/en provenance du PUF.

Type: 9.

Champ	Type de champ	Sens	Requis	Commentaire
Address	chaîne IA5	B	M	15 octets est la longueur maximale

Remarque: la conversion en codage BCD est assurée par le NAF.

6.3.7 CallingDTEAddressExt

Description: ce paramètre sert à transmettre des informations d'extension d'adresse relatives à l'ETTD local à destination/en provenance du PUF.

Type: 10.

Champ	Type de champ	Sens	Requis	Commentaire
AddressExt	chaîne IA5	B	M	40 octets est la longueur maximale

Remarque: la conversion en codage BCD est assurée par le NAF.

6.3.8 ExpeditedData

Description: ce paramètre sert à transmettre des informations sur l'utilisation de données exprès à destination/en provenance du PUF.

Type: 29.

Champ	Type de champ	Sens	Requis	Commentaire
Usage	Booléen	B	M	TRUE – l'utilisation de données exprès est requise ou supportée FALSE – l'utilisation de données exprès n'est pas requise ou n'est pas supportée

6.3.9 FacilityData

Description: ce paramètre sert à transmettre des informations relatives aux services à destination/en provenance du PUF.

Type: 31.

Champ	Type de champ	Sens	Requis	Commentaire
FacilityData	chaîne d'octets	B	M	codé sous la forme des informations de services définies dans l'ISO/CEI 8208 [2] 109 octets est la longueur maximale

6.3.10 FastSelect

Description: ce paramètre sert à transmettre des informations relatives aux services de sélection rapide, à destination/en provenance du PUF.

Type: 32.

Champ	Type de champ	Sens	Requis	Commentaire
FastSelect	octet	B	M	norestriction (1) – L'ETTD appelé n'est pas requis de supprimer la connexion d'utilisateur avant son établissement complet restricted (2) – L'ETTD appelé est requis de supprimer la connexion d'utilisateur avant son établissement complet

Remarques: lorsqu'il est spécifié dans un message UConnectReq, ce paramètre permet au paramètre UserData d'avoir une longueur maximale de 128 octets. Si l'option de *restriction* est sélectionnée, elle indique que la connexion d'utilisateur ne peut pas être établie et qu'il y a lieu d'attendre un message UDisconnectInd avec un paramètre UserData d'une longueur maximale de 128 octets. Si l'option de *non-restriction* est spécifiée, la connexion d'utilisateur peut être établie et le message UConnectCnf subséquent peut avoir un paramètre UserData d'une longueur maximale de 128 octets. Après cette opération, ces deux champs des messages UDisconnectInd et UDisconnectReq peuvent également avoir des paramètres UserData d'une longueur maximale de 128 octets.

Lorsqu'il est reçu dans un message UConnectInd, ce paramètre indique que le paramètre UserData contenu dans ce message peut avoir une longueur maximale de 128 octets. Si l'option de *restriction* est sélectionnée, elle indique que la connexion d'utilisateur ne peut pas être établie et que le PUF doit répondre par le message UDisconnectReq. Le paramètre UserData contenu dans ce message peut avoir une longueur maximale de 128 octets. Si l'option de *non-restriction* est sélectionnée, le PUF peut répondre par un message UConnectRsp avec un paramètre UserData d'une longueur maximale de 128 octets. Après cette opération, ces deux champs des messages UDisconnectInd et UDisconnectReq peuvent également avoir des paramètres UserData d'une longueur maximale de 128 octets.

6.3.11 GroupID

Description: ce paramètre sert à transmettre l'identificateur de groupe à destination/en provenance du PUF.

Type: 33.

Champ	Type de champ	Sens	Requis	Commentaire
GroupID	chaîne d'octets	B	M	la valeur est unique pour une relation PUF/NAF 4 octets est la longueur fixée

6.3.12 L2ConnectionMode

Description: ce paramètre sert à transmettre au NAF des détails sur le mode de connexion en couche Liaison de données.

Type: 38.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	P	M	dte (1) – agit comme ETTD comme défini dans l'ISO/CEI 7776 dce (2) – agit comme ETCD comme défini dans l'ISO/CEI 7776 auto (3) – agit comme ETTD s'il est appelant; agit comme ETCD s'il est appelé.

6.3.13 L2FrameSize

Description: ce paramètre sert à transmettre au NAF des détails sur la longueur de trame de couche 2.

Type: 39.

Champ	Type de champ	Sens	Requis	Commentaire
Value	chaîne d'octets	P	M	longueur de trame (en octets) longueur fixée à 2 octets le premier octet contient le plus fort poids des 2 multiplets contenant la valeur

6.3.14 L2WindowSize

Description: ce paramètre sert à transmettre des détails sur la longueur de fenêtre de couche 2 au NAF.

Type: 40.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	P	M	longueur de fenêtre

6.3.15 L2XID

Description: ce paramètre sert à transmettre des détails sur la valeur d'identificateur XID de couche 2 et son utilisation.

Type: 41.

Champ	Type de champ	Sens	Requis	Commentaire
Use	octet	P	M	envoi (1) – envoi XID correspondance (2) – correspondance du XID avec le XID reçu. Si les XID ne correspondent pas, la connexion ne doit pas être établie.
Value	chaîne d'octets	P	M	valeur d'identificateur XID (identificateur et signature) longueur maximale: 64 octets

6.3.16 L3ConnectionMode

Description: ce paramètre sert à transmettre des détails sur le mode de connexion en couche Liaison de données au NAF.

Type: 42.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	P	M	dte (1) – agit comme ETTD dce (2) – agit comme ETCD auto (3) – agit comme ETTD s'il est appelant; agit comme ETCD s'il est appelé. dxe (4) – utilise le paquet de redémarrage pour déterminer s'il doit jouer le rôle d'ETTD ou d'ETCD comme dans l'option "auto" de l'ISO/CEI 8208

6.3.17 L3IncomingVCCount

Description: ce paramètre sert à transmettre le nombre de connexions qui peuvent être établies à un instant donné par des requêtes d'établissement d'appel entrant.

Type: 43.

Champ	Type de champ	Sens	Requis	Commentaire
Value	chaîne d'octets	P	M	nombre de connexions. Valeur maximale: 4095. longueur fixée à 2

6.3.18 L3OutgoingVCCount

Description: ce paramètre sert à transmettre le nombre de connexions qui peuvent être établies à un instant donné par des requêtes d'établissement d'appel sortant.

Type: 44.

Champ	Type de champ	Sens	Requis	Commentaire
Value	chaîne d'octets	P	M	nombre de connexions. Valeur maximale: 4095. longueur fixée à 2

6.3.19 L3TwoWayVCCount

Description: ce paramètre sert à transmettre le nombre de connexions qui peuvent être établies à un instant donné par des requêtes d'établissement de connexion sortante ou entrante.

Type: 45.

Champ	Type de champ	Sens	Requis	Commentaire
Value	chaîne d'octets	P	M	nombre de connexions. Valeur maximale: 4095. longueur fixée à 2

6.3.20 NCOType

Description: ce paramètre sert à transmettre le type d'objet de connexion réseau (NCO) au NAF.

Type: 50.

Champ	Type de champ	Sens	Requis	Commentaire
Identifier	octet		M	U3 (2) – accès à la couche Réseau par le plan U avec coordination sémaphore par le NAF (fonction de coordination par le NAF) C/U (3) – accès aux couches Signalisation et Réseau par le plan U U3/G (4) – accès à la couche Réseau par le plan U vers des circuits virtuels additionnels. Cet objet NCO doit être groupé avec un objet NCO de type U3 ou C/U existant.

6.3.21 PacketSize

Description: ce paramètre sert à transmettre des informations de longueur de paquet à destination/en provenance du PUF.

Type: 52.

Champ	Type de champ	Sens	Requis	Commentaire
Negotiation	Booléen	B	M	sert à indiquer si la négociation de la longueur des paquets est possible TRUE– négociation possible FALSE – négociation impossible
Invalue	octet	B	M	longueur maximale des données d'utilisateur centripètes (voir le Tableau 4) longueur maximale des données pouvant être reçues par le message UDataInd
Outvalue	octet	B	M	longueur maximale des données d'utilisateur centrifuges (voir le Tableau 4) longueur maximale des données pouvant être transmises par le message UDataReq

Remarques: ce paramètre sert à déterminer la longueur maximale des tampons de données qui peuvent être transmis par le message UDataReq et par le message UDataInd. Il est utilisé comme suit:

– dans le message UConnectReq, le PUF peut spécifier les valeurs qu'il souhaite utiliser;

– dans le message UConnectCnf, le NAF doit toujours spécifier les valeurs à utiliser pour la connexion d'utilisateur;

– dans le message UConnectInd, le NAF doit toujours indiquer les valeurs à utiliser pour la connexion d'utilisateur. Il indique également s'il est possible pour le PUF de négocier ces valeurs;

– dans le message UConnectRsp, le PUF peut spécifier des valeurs si le message UConnectInd a indiqué que la négociation était possible.

Pour le protocole T.90, ce paramètre doit être utilisé conformément aux règles de la Recommandation T.90 [1].

Tableau 4 – Valeurs précodées de longueur de paquet

Valeur précodée	Longueur de paquet (octet)	Valeur précodée	Longueur de paquet (octet)
4	16	9	512
5	32	10	1024
6	64	11	2048
7	128	12	4096
8	256		

6.3.22 QOSParameters

Description: ce paramètre sert à transmettre des informations relatives à la qualité de service à destination/en provenance du PUF.

Type: 54.

Champ		Type de champ	Sens	Requis	Commentaire
Throughput	Usage	Booléen	B	M	indique si les valeurs suivantes sont incluses
	InTarget	octet	B	C	valeurs fournies dans le Tableau 5
	InLowest	octet	B	C	valeurs fournies dans le Tableau 5
	InAvailable	octet	B	C	valeurs fournies dans le Tableau 5
	InSelected	octet	B	C	valeurs fournies dans le Tableau 5
	OutTarget	octet	B	C	valeurs fournies dans le Tableau 5
	OutLowest	octet	B	C	valeurs fournies dans le Tableau 5
	OutAvailable	octet	B	C	valeurs fournies dans le Tableau 5
	OutSelected	octet	B	C	valeurs fournies dans le Tableau 5
NCPriority	Usage	Booléen	B	M	indique si les valeurs suivantes sont incluses
	Target	octet	B	C	(Note 1)
	Lowest	octet	B	C	(Note 1)
	Available	octet	B	C	(Note 1)
	Selected	octet	B	C	(Note 1)
TransitDelay	Usage	Booléen	B	M	indique si les valeurs suivantes sont incluses
	Selected	chaîne d'octets	B	C	(Note 2)
	Target	chaîne d'octets	B	C	(Note 2)
	Maximum	chaîne d'octets	B	C	(Note 2). Présent si le champ "Target" – précédent – a été utilisé; sinon, absent.
End-to-End Transit Delay	Usage	Booléen	B	M	indique si les valeurs suivantes sont incluses
	Selected	chaîne d'octets	B	C	(Note 2)
	Target	chaîne d'octets	B	C	(Note 2)
	Maximum	chaîne d'octets	B	C	(Note 2). Présent si le champ "Target" – précédent – a été utilisé; sinon, absent.

NOTE 1 – Les champs NCPriority peuvent prendre toute valeur comprise entre 1 (priorité la plus élevée) et 10 (priorité la moins élevée). S'il n'est pas utilisé, le champ sera rempli par la valeur 0. S'il n'est pas spécifié, le champ sera rempli par la valeur 11.

NOTE 2 – Longueur fixée à 2. Le premier octet est le moins significatif. 65 535 signifie "pas utilisé". Le délai est exprimé en millisecondes.

Remarque: pour le protocole T.90, ce paramètre doit être utilisé conformément aux règles de la Recommandation T.90 [1].

Tableau 5 – Valeur de précodage du débit utile

Valeur de précodage	Classe de débit utile	Valeur de précodage	Classe de débit utile
3	75	9	4800
4	150	10	9600
5	300	11	19 200
6	600	12	48 000
7	1200	13	64 000
8	2400	0	valeur non utilisée

6.3.23 ReadyFlag

Description: ce paramètre sert à demander et à indiquer le statut de la commande de débit concernant une connexion d'utilisateur.

Type: 55.

Champ	Type de champ	Sens	Requis	Commentaire
Usage	Booléen	B	M	TRUE – transfert de données autorisé. FALSE – transfert de données non autorisé.

6.3.24 ReceiptConfirm

Description: ce paramètre sert à demander confirmation de réception de données pour une connexion du plan U.

Type: 57.

Champ	Type de champ	Sens	Requis	Commentaire
Value	Booléen	B	M	TRUE – confirmation demandée FALSE – confirmation non demandée

6.3.25 RespondingDTEAddress

Description: ce paramètre sert à transmettre des informations d'adresse concernant l'ETTD qui répond, à destination/en provenance du PUF.

Type: 58.

Champ	Type de champ	Sens	Requis	Commentaire
Address	chaîne IA5		M	16 octets est la longueur maximale

6.3.26 RespondingDTEAddressExt

Description: ce paramètre sert à transmettre, en provenance/à destination du PUF, des informations d'extension d'adresse concernant l'ETTD qui répond.

Type: 59.

Champ	Type de champ	Sens	Requis	Commentaire
AddressExt	chaîne IA5	B	M	40 octets est la longueur maximale

6.3.27 TEI

Description: ce paramètre sert à accéder à une liaison permanente avec un commutateur de paquets de données (connexion en mode paquet dans le canal D).

Type: 61.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	B	M	

6.3.28 UProtocol

Description: ce paramètre sert à sélectionner le protocole du plan U.

Type: 62.

Champ	Type de champ	Sens	Requis	Commentaire
L3Protocol	octet	P	M	valeur par défaut (255) – T.90 [1] T.90 (1) ISO/CEI 8208 (2)
L2Protocol	octet	P	O	valeur par défaut (255) – ISO/CEI 7776
L1Protocol	octet	P	O	valeur par défaut (255) – accès transparent à un canal B

Remarque: d'autres valeurs possibles sont décrites en [6].

6.3.29 UAttributeName

Description: ce paramètre sert à transmettre le nom d'un ensemble statique d'attributs du plan U, à partir du PUF.

Type: 63.

Champ	Type de champ	Sens	Requis	Commentaire
AttributeName	chaîne IA5	P	M	16 est la longueur maximale

6.3.30 UDirection

Description: ce paramètre sert à transmettre des informations concernant l'usage d'un objet NCO particulier au NAF, pour le plan U.

Type: 64.

Champ	Type de champ	Sens	Requis	Commentaire
Direction	octet	P	M	écoute (1) envoi (2) les deux (3)

6.3.31 UserData

Description: ce paramètre sert à transmettre des données qui sont limitées en longueur, à destination/en provenance du PUF.

Type: 65.

Champ	Type de champ	Sens	Requis	Commentaire
Data	chaîne d'octets	B	M	128 octets est la longueur maximale la longueur maximale autorisée varie de message en message et est différente en fonction de l'utilisation du paramètre FastSelect

6.3.32 WindowSize

Description: ce paramètre sert à transmettre des informations de longueur de fenêtre à destination/en provenance du PUF.

Type: 67.

Champ	Type de champ	Sens	Requis	Commentaire
Negotiation	Booléen	B	M	sert à indiquer si la négociation de longueur de fenêtre est possible TRUE – négociation possible FALSE – négociation impossible
Invalue	octet	B	M	longueur de fenêtre centripète
Outvalue	octet	B	M	longueur de fenêtre centrifuge

Remarques: ce paramètre sert à déterminer la longueur de fenêtre à utiliser pour une connexion d'utilisateur.

- Dans le message UConnectReq, le PUF peut spécifier les valeurs qu'il souhaite utiliser.
- Dans le message UConnectCnf, le NAF doit toujours spécifier les valeurs à utiliser pour la connexion d'utilisateur.
- Dans le message UConnectInd, le NAF doit toujours indiquer les valeurs à utiliser pour la connexion d'utilisateur. Il indique également s'il est possible pour le PUF de négocier ces valeurs.
- Dans le message UConnectRsp, le PUF peut spécifier des valeurs si le message UConnectInd a indiqué que la négociation était possible.

6.3.33 X213Cause

Description: ce paramètre sert à transmettre des informations de cause X.213 à destination/en provenance du PUF.

Type: 68.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	B	M	voir au 6.7 les valeurs des codes de retour du plan U

6.3.34 X213Origin

Description: ce paramètre sert à transmettre des informations d'origine selon X.213 à destination/en provenance du PUF.

Type: 69.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	B	M	valeur indéfinie (1) fournisseur du NAF (2) utilisateur du PUF (3)

6.3.35 X25Cause

Description: ce paramètre sert à transmettre des informations de cause selon X.25 à destination/en provenance du PUF.

Type: 70.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	B	M	voir les valeurs des codes de cause selon l'ISO/CEI 8208 [2]

6.3.36 X25Diagnostic

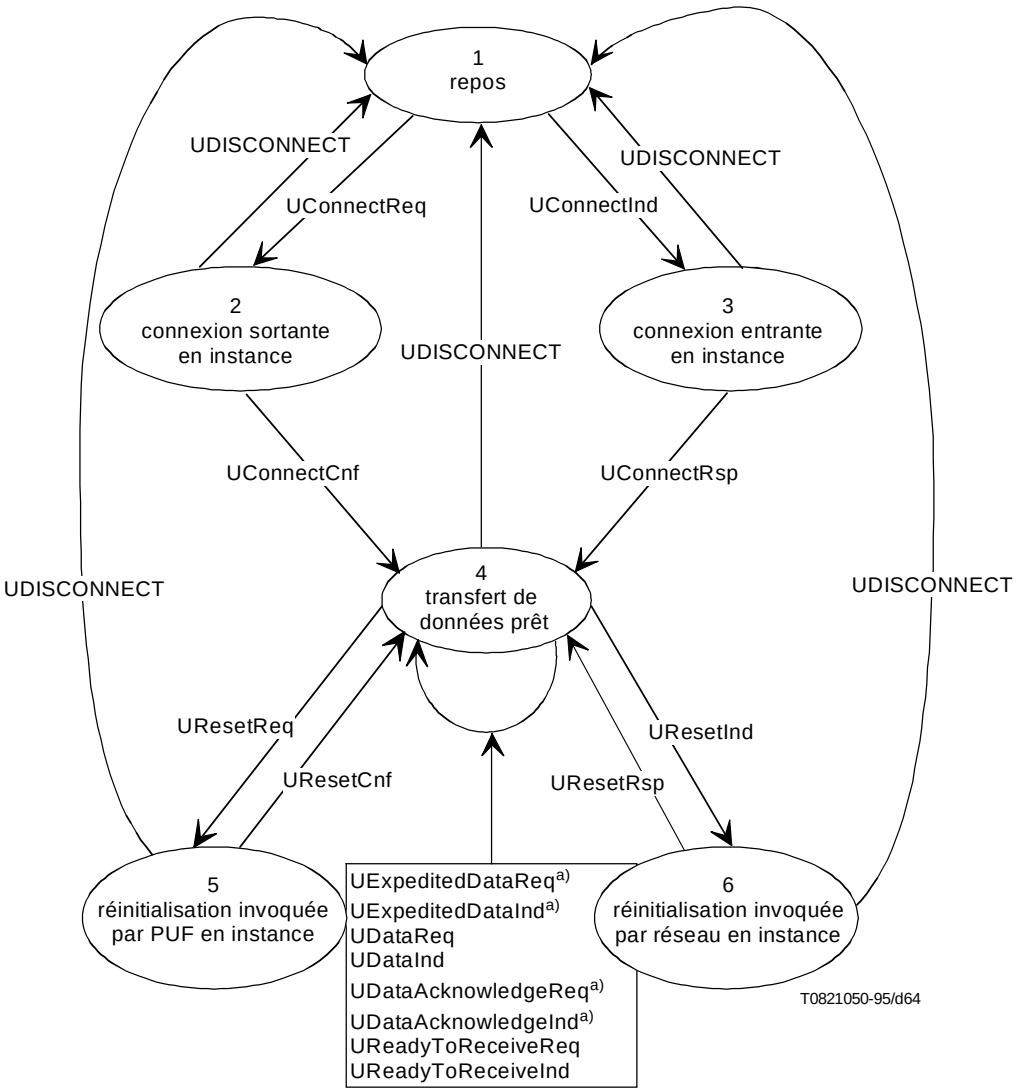
Description: ce paramètre sert à transmettre des informations de diagnostic X.25 à destination/en provenance du PUF.

Type: 71.

Champ	Type de champ	Sens	Requis	Commentaire
Value	octet	B	M	voir les valeurs de diagnostic ISO/CEI 8208

6.4 Diagramme de transition d'état

La Figure 2 montre les différents états que peut prendre une connexion d'utilisateur faisant appel aux messages du plan U et dans quel ordre ces messages doivent être utilisés.



a) Ce message n'est utilisé que pour le protocole ISO/CEI 8208.

NOTE – L'indication UDISCONNECT désigne un message de type UDisconnectReq ou UDisconnectInd.

Figure 2 – Aperçu général des messages du plan U

6.5 Fonction de coordination

La fonction de coordination peut être utilisée. Voir [6] pour les détails.

6.6 Critères de sélection

Le présent sous-paragraphe traite de paramètres spécifiques de l'ISO/CEI 8208. Les critères généraux de sélection d'objets NCO sont indiqués en [5].

6.6.1 Sélection d'objet NCO

Pour sélectionner un objet NCO, le NAF utilise les paramètres suivants:

- négociation de longueur de paquet;
- négociation de longueur de fenêtre.

6.6.1.1 Négociation de longueur de paquet

Dans le paquet INCOMING CALL (appel entrant), si la longueur de paquet n'est pas fournie, la valeur par défaut, c'est-à-dire 128 octets, est prise comme hypothèse.

La longueur de paquet d'un objet NCO est correcte si une des conditions suivantes est vérifiée:

- la longueur de paquet – indiquée dans le paramètre U3AttributeSet – est égale à la longueur de paquet indiquée dans le paquet INCOMING CALL (appel entrant) ou prise par défaut;
- s'il n'y a pas de longueur de paquet indiquée dans le paramètre U3AttributeSet.

6.6.1.2 Négociation de longueur de fenêtre

Dans le paquet INCOMING CALL (appel entrant), si la longueur de fenêtre n'est pas fournie, la valeur par défaut, c'est-à-dire 2, est prise comme hypothèse.

La longueur de fenêtre d'un objet NCO est correcte si une des conditions suivantes est vérifiée:

- la longueur de fenêtre – indiquée dans le paramètre U3AttributeSet – est égale à la longueur de fenêtre indiquée dans le paquet INCOMING CALL (appel entrant) ou prise par défaut;
- s'il n'y a pas de longueur de fenêtre indiquée dans le paramètre U3AttributeSet.

6.6.1.3 Négociation de la longueur effective des paquets et des fenêtres

Dans le message UConnectRsp, si la longueur de paquet n'est pas fournie, la longueur de paquet indiquée dans l'appel entrant – c'est-à-dire UConnectInd – est acceptée par le PUF. Les mêmes règles s'appliquent à la longueur de fenêtre.

Dans le message UConnectCnf, si la longueur de paquet/longueur de fenêtre n'est pas fournie, la longueur de paquet/longueur de fenêtre indiquée lors de l'appel sortant – c'est-à-dire UConnectReq – est acceptée pour utilisation par le PUF.

6.6.2 Action si aucun objet NCO n’est disponible

Un message de déconnexion est émis par le dispositif NAF avec la cause X.213 "Rejet de demande de connexion – raison non spécifiée – état transitoire".

6.7 Traitement et codes d'erreur spécifiques

Les erreurs sont traitées comme suit.

6.7.1 Utilisation non valide de messages du plan U

En cas:

- d'utilisation non valide de service de confirmation de réception;
- d'utilisation non valide de demande de confirmation pour un message UDataReq;
- de longueur non valide d'un paramètre UserData dans un message UDataReq;
- d'utilisation non valide de données exprès;
- d'émission non valide de messages pendant l'état de réinitialisation,

l'action est la suivante:

- le dispositif PUF reçoit le message UDisconnectInd.

En cas:

– d'utilisation non valide du paramètre Bit_DQM (association entre les bits M et Q) dans des messages subséquents de type UDataReq,

l'action est la suivante:

– le dispositif PUF reçoit le message UResetInd.

6.7.2 Autres erreurs

En cas d'erreur de contenu d'un paramètre, le dispositif PUF reçoit le message UDisconnectInd.

6.7.3 Causes

Ces valeurs peuvent être spécifiées et sont retournées dans le paramètre X213Cause.

Tableau 6 – Valeur du paramètre X213Cause

Code de retour		Signification	Information d'erreur spécifique
Undefined	220	situation d'erreur indéfinie	absente
NSAPunreachablePerm	221	rejet de connexion – point NSAP inatteignable/état fixe	absente
DiscTrans	225	déconnexion – état transitoire	absente
DiscPerm	226	déconnexion – état fixe	absente
NoReasonTrans	227	rejet de connexion – raison non spécifiée/état transitoire	absente
NoReasonPerm	228	rejet de connexion – raison non spécifiée/état fixe	absente
QOSnotavailTrans	229	rejet de connexion – QS indisponible/état transitoire	absente
QOSnotavailPerm	230	rejet de connexion – QS indisponible/état fixe	absente
NSAPunreachableTrans	231	rejet de connexion – point NSAP inatteignable/état transitoire	absente
NSAPunknown	232	rejet de connexion – adresse de point NSAP inconnue (état fixe)	absente
DiscNorm	241	déconnexion – état normal	absente
DiscAbnorm	242	déconnexion – état anormal	absente
ConRejectTrans	244	rejet de connexion – état transitoire	absente
ConRejectPerm	245	rejet de connexion – état fixe	absente
ConRejectUserData	248	rejet de connexion – informations incompatibles dans un paramètre UserData	absente

6.8 Ensemble d'attributs

6.8.1 Paramètres d'ensemble d'attributs

Tableau 7 – Paramètres d'ensemble d'attributs du plan U (UAttributeSet)

Paramètre	Requis	Commentaire
WindowSize	O	longueur de fenêtre en couche 3. Voir 6.3.32.
PacketSize	O	longueur de paquet en couche 3. Voir 6.3.21.
FastSelect	O	serv. compl. de sélection rapide. Voir 6.3.10.
QOSParameters	O	qualité de service. Voir 6.3.22.
UProtocol	O	voir remarque. Voir aussi 6.3.28.
L3ConnectionMode	O	voir remarque. Voir aussi 6.3.16.
L3TwoWayVCCount	O	voir remarque. Voir aussi 6.3.19.
L3IncomingVCCount	O	voir remarque. Voir aussi 6.3.17.
L3OutgoingVCCount	O	voir remarque. Voir aussi 6.3.18.
TEI	O	voir remarque. Voir aussi 6.3.27.
L2ConnectionMode	O	voir remarque. Voir aussi 6.3.12.
L2WindowSize	O	voir remarque. Voir aussi 6.3.14.
L2FrameSize	O	voir remarque. Voir aussi 6.3.13.
L2XID	O	voir remarque. Voir aussi 6.3.15.

Remarques: ces paramètres ne peuvent être utilisés que lors de la création d'objets NCO contenant des informations du plan de commande. Les objets NCO qui doivent être associés au moyen d'un identificateur de groupe ne peuvent pas spécifier ces paramètres. Voir le sous-paragraphe relatif à – l'opération ACreateNCO – en [5] pour les détails.

Si des paramètres sont omis, la valeur par défaut doit être utilisée. Les valeurs par défaut sont fonction du protocole du plan U. Le dispositif NAF doit fournir la valeur correcte en fonction du protocole. Les valeurs par défaut sont décrites en Appendice I.

Tableau 8 – Paramètres d'ensemble d'adresses du plan U (UAddressSet)

Paramètre	Requis	Commentaire
CalledDTEAddress	O	voir 7.3.4 pour la définition de ce paramètre
CalledDTEAddressExt	O	voir 7.3.5 pour la définition de ce paramètre
CallingDTEAddress	O	voir 7.3.6 pour la définition de ce paramètre
CallingDTEAddressExt	O	voir 7.3.7 pour la définition de ce paramètre

6.8.2 Contenu d'un ensemble d'attributs statiques

Les ensembles d'attributs décrits ci-dessous utilisent les conventions suivantes:

– le nom doit être utilisé avec le message ACreateNCOReq;

– toutes les valeurs numériques sont en notation décimale.

nom:	U_ISO8208
WindowSize:	2
PacketSize:	128 (octets)
UProtocol:	ISO/CEI 8208
L3ConnectionMode:	DXE
L3TwoWayVCCount:	arrangement local
L3IncomingVCCount:	1
L3OutgoingVCCount:	1
L2ConnectionMode:	auto
L2WindowSize:	7
L2FrameSize:	128 (octets)
L2XID:	néant

nom:	U_TELEMATIC_TERM
WindowSize:	2
PacketSize:	128 (octets)
UProtocol:	T.90
L3ConnectionMode:	DXE
L3TwoWayVCCount:	arrangement local
L3IncomingVCCount:	0
L3OutgoingVCCount:	0
L2ConnectionMode:	auto
L2WindowSize:	7
L2FrameSize:	128 (octets)

7 Protocole T.70NL

7.1 Introduction

Le présent paragraphe traite du protocole T.70. Dans la présente partie, chaque fois qu'il est fait référence à la Recommandation T.70, il s'agit implicitement du protocole de couche vide T.70NL (*null layer*).

L'emplacement du protocole T.70 dans le modèle OSI est représenté en Figure 3.

Les conventions générales de description sont indiquées en [5].

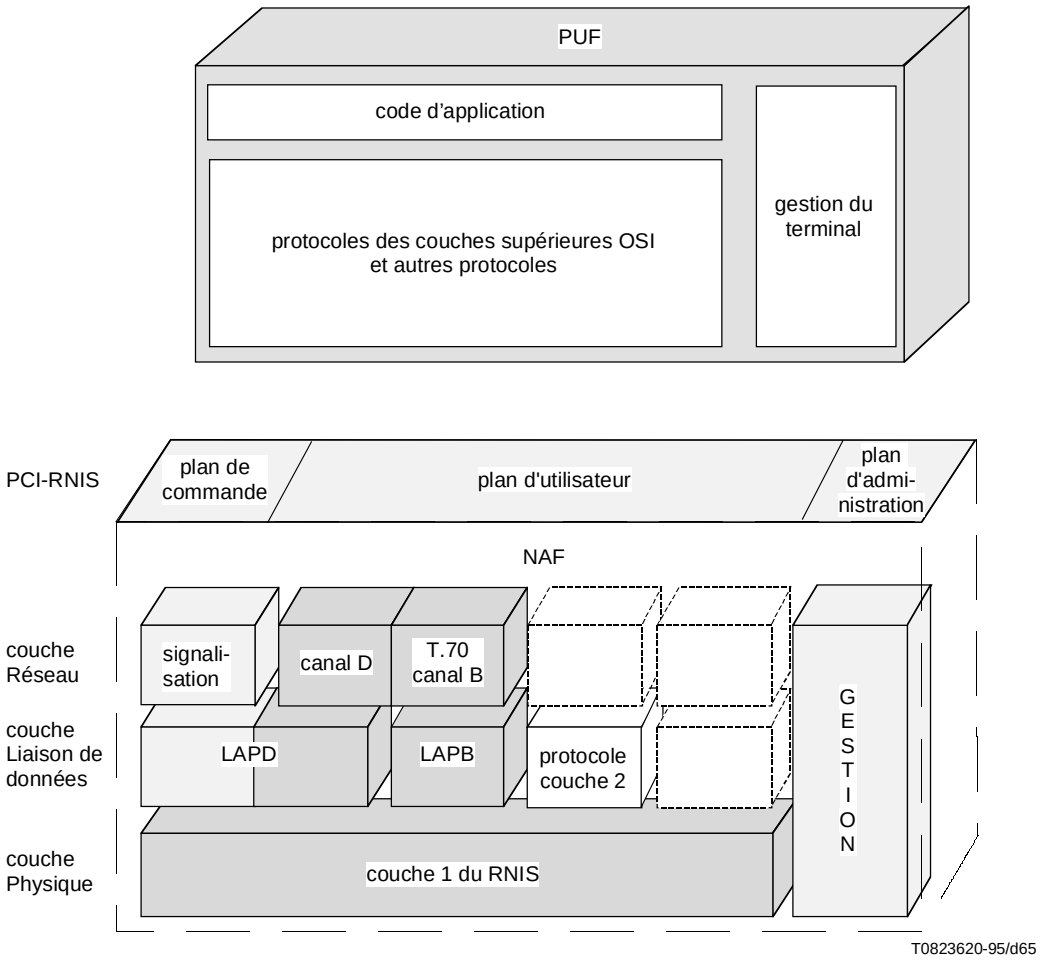


Figure 3 – Emplacement dans le modèle OSI

7.2 Messages

Les messages du plan U donnent accès aux piles du protocole T.70. On trouvera ci-dessous une liste et une brève description des messages applicables du plan U. Le Tableau 9 donne un aperçu général de ces messages.

Tableau 9 – Aperçu général des messages du plan U

Identificateur du message	Classe	Nom du message	Objet du message
307	1	UDataReq	demande de transfert de données sur une connexion d'utilisateur établie
308	1	UDataInd	indication d'arrivée de données transférées sur une connexion d'utilisateur établie

7.2.1 UDataReq

Classe: 1 (classe de base).

Description: ce message permet à un dispositif PUF d'envoyer un paquet de données. La taille d'un paquet de données est limitée à la valeur négociée au cours de l'établissement de la connexion d'utilisateur.

Paramètres:

Nom	Requis	Commentaire
NCOID	M	identifie la connexion du plan U
Bit_DQM	O	utilisé pour régler les bits M et Q du protocole T.70

Remarque: les données à envoyer sont obligatoires. Elles ne sont pas fournies sous forme de paramètre du message.

Les données obligatoires doivent être fournies dans le tampon de données.

Message associé: UReadyToReceiveInd.

7.2.2 UDataInd

Classe: 1 (classe de base).

Description: ce message indique à un PUF la présence de données reçues. La taille d'un paquet de données est limitée à la valeur négociée au cours de l'établissement de la connexion d'utilisateur.

Paramètres:

Nom	Fourni	Commentaire
NCOID	M	identifie la connexion du plan U
Bit_DQM	O	utilisé pour indiquer la réception des bits M et Q selon T.70

Remarque: les données reçues sont toujours indiquées, mais non pas sous forme de paramètre du message.

Les données sont fournies dans le tampon de données. Ce tampon est, dans ce cas, obligatoire.

Message associé: UReadyToReceiveReq.

7.3 Paramètres contenus dans les messages

Le présent sous-paragraphe décrit les paramètres du plan U pour le protocole T.70. Ils sont présentés dans l'ordre de leurs identificateurs. La présentation des informations est décrite en [4].

Tableau 10 – Aperçu général des paramètres du plan U

Identificateur du paramètre	Nom du paramètre	Usage dans le plan U	Usage dans l'ensemble UAttributeSet	Autre usage
4	Bit_DQM	X		
50	NCOType			X
52	PacketSize		X	
62	UProtocol		X	
63	UAttributeName			
64	UDirection			X

7.3.1 Bit_DQM

Description: ce paramètre sert à transmettre à destination/en provenance du PUF:

– la valeur du bit M (données à suivre) (bit 3).

Chaque information utilise une position binaire. Le bit de plus fort poids (MSB, *most significant bit*) est le bit 8 et le bit de plus faible poids (LSB, *least significant bit*) est le bit 1. Le bit 1 représente la valeur 1, le bit 2 représente la valeur 2 et le bit 3 représente la valeur 4. La valeur de résultat applicable à ce paramètre est la somme des valeurs de chaque élément binaire (OU logique).

Type: 4.

Champ	Type de champ	Sens	Requis	Commentaire
DQM	octet	B	M	bit 1: 0 – la confirmation de la réception de données n'est pas autorisée ou n'est pas requise bit 2: 0 – désactivation du bit qualificateur (Q) bit 3: 1 – activation du bit de données à suivre (M) 0 – désactivation du bit de données à suivre (M)

7.3.2 NCOType

Description: ce paramètre sert à transmettre le type d'objet de connexion réseau (NCO) au NAF.

Type: 50.

Champ	Type de champ	Sens	Requis	Commentaire
Identifier	octet		M	U3 (2) – accès à la couche Réseau par le plan U avec coordination sémaphore par le NAF (fonction de coordination par le NAF)

7.3.3 PacketSize

Description: ce paramètre sert à transmettre des informations de longueur de paquet à destination/en provenance du PUF.

Type: 52.

Champ	Type de champ	Sens	Requis	Commentaire
Negotiation	Booléen	B	M	sert à indiquer si la négociation de la longueur des paquets est possible TRUE – négociation possible FALSE – négociation impossible
Invalue	octet	B	M	longueur maximale des données d'utilisateur centripètes (voir le Tableau 12) longueur maximale des données pouvant être reçues par le message UdataInd
Outvalue	octet	B	M	longueur maximale des données d'utilisateur centrifuges (voir le Tableau 12) longueur maximale des données pouvant être transmises par le message UDataReq

Remarque: ce paramètre sert à déterminer la longueur maximale des tampons de données qui peuvent être transmis par les messages UDataReq et UDataInd.

Tableau 11 – Valeurs précodées de longueur de paquet

Valeur précodée	Longueur de paquet (octet)	Valeur précodée	Longueur de paquet (octet)
4	16	8	256
5	32	9	512
6	64	10	1024
7	128	11	2048

7.3.4 UProtocol

Description: ce paramètre sert à sélectionner le protocole du plan U.

Type: 62.

Champ	Type de champ	Sens	Requis	Commentaire
L3Protocol	octet	P	M	T.70 (3)
L2Protocol	octet	P	O	valeur par défaut (255) – ISO/CEI 7776
L1Protocol	octet	P	O	valeur par défaut (255) – accès transparent à un canal B

Remarque: d'autres valeurs possibles sont décrites en [6].

7.3.5 UAttributeName

Description: ce paramètre sert à transmettre le nom d'un ensemble statique d'attributs du plan U à partir du PUF.

Type: 63.

Champ	Type de champ	Sens	Requis	Commentaire
AttributeName	chaîne IA5	P	M	16 octets est la longueur maximale

7.3.6 UDirection

Description: ce paramètre sert à transmettre des informations concernant l'usage d'un objet NCO particulier au NAF, pour le plan U.

Type: 64.

Champ	Type de champ	Sens	Requis	Commentaire
Direction	octet	P	O	les deux (3)

7.4 Diagramme de transition d'état

Les messages du plan U ne changent pas l'état de la connexion.

7.5 Fonction de coordination

La fonction de coordination ne peut pas être utilisée.

7.6 Critères de sélection

Aucun paramètre spécifique du protocole T.70 n'est utilisé. Les critères généraux de sélection d'objets NCO sont indiqués en [5].

7.7 Traitement et codes d'erreur spécifiques

Les erreurs sont traitées comme suit.

7.8 Traitement et codes d'erreur spécifiques

Les erreurs de protocole ne sont pas disponibles à l'interface.

7.9 Attributs statiques

7.9.1 Paramètres d'un ensemble d'attributs

Tableau 12 – Paramètres d'ensemble d'attributs du plan U (UAttributeSet)

Paramètre	Requis	Commentaire
Uprotocol	O	voir remarque. Voir aussi 7.3.4.
L3PacketSize	O	voir remarque. Voir aussi 7.3.3.

Remarque: ces paramètres ne peuvent être utilisés que lors de la création d'objets NCO contenant des informations du plan de commande. Voir le sous-paragraphe relatif à – l'opération ACreateNCO – en [5] pour les détails.

Si des paramètres sont omis, la valeur par défaut doit être utilisée par le NAF. Les valeurs par défaut sont décrites en Appendice I.

7.9.2 Contenu de l'ensemble d'attributs statiques

nom:	U_T70
UProtocol:	T.70
L3PacketSize:	128

Appendice I

Configuration

I.1 Protocole T.90

Tableau I.1 – Configuration du plan U pour le protocole T.90

Paramètre	Valeur par défaut suggérée	Commentaire
réseau de type X.25	0	permet au NAF de s'adapter à différentes applications nationales de la Rec. X.25
Rec. X.25	CCITT88	niveau reconnu de la Rec. X.25
numérotage des séquences de couche 3	8	
longueur maximale de fenêtre en couche 3	7	
valeur par défaut de longueur de fenêtre en couche 3	3	
longueur de paquet maximale en couche 3	4096	
valeur par défaut de longueur de paquet en couche 3	128	

Tableau I.1 – Configuration du plan U pour le protocole T.90 *(fin)*

Paramètre	Valeur par défaut suggérée	Commentaire
valeur par défaut du mode de connexion en couche 3	Auto	Auto – agit comme ETTD s'il est appelant; agit comme ETCD s'il est appelé. dxe – utilise le paquet de redémarrage pour déterminer s'il doit jouer le rôle d'ETTD ou d'ETCD selon l'ISO/CEI 8208 [2] dte – agit comme ETTD dce – agit comme ETCD
plus petit nombre de circuits SVC entrants (LIC)	1	
plus grand nombre de circuits SVC entrants (HIC)	1	
plus petit nombre de circuits SVC mixtes (LTC)	0	
plus grand nombre de circuits SVC mixtes (HTC)	0	
plus petit nombre de circuits SVC sortants (LOC)	0	
plus grand nombre de circuits SVC sortants (HOC)	0	
temporisateurs de couche 3		le NAF peut choisir de fournir à l'utilisateur du PUF la possibilité de configurer les temporisateurs
valeur par défaut du mode de connexion en couche 2	Auto	Auto – agit comme ETTD s'il est appelant; agit comme ETCD s'il est appelé. dte comme défini dans l'ISO/CEI 7776 dce comme défini dans l'ISO/CEI 7776
arithmétique des canaux B en couche 2	8	NOTE – Doit être 128 pour X.25 sur canal D
longueur de fenêtre en couche 2	7	
longueur de trame en couche 2	128	
type d'activation en couche 2	Case1	Case1 – envoi de la commande SABM/SABME s'il est appelant; pas d'envoi s'il est appelé. Case2 – envoi de la commande SABM/SABME s'il est appelé, pas d'envoi s'il est appelant. Passive – pas d'envoi de la commande SABM/SABME lors de l'activation Active – envoi de la commande SABM/SABME lors de l'activation
temporisateurs en couche 2		
– T1	5	valeur exprimée en secondes
– T1	1	valeur exprimée en secondes
– N2	5	nombre maximal de tentatives de retransmission

I.2 Protocole ISO/CEI 8208

Tableau I.2 – Configuration du plan U pour le protocole ISO/CEI 8208

Paramètre	Valeur par défaut suggérée	Commentaire
réseau de type X.25	0	permet au NAF de s'adapter à différentes implémentations nationales de la Rec. X.25
Rec. X.25	CCITT88	niveau reconnu de la Rec. X.25
numérotage des séquences de couche 3	8	
longueur maximale de fenêtre en couche 3	7	
valeur par défaut de longueur de fenêtre en couche 3	3	
longueur de paquet maximale en couche 3	4096	
valeur par défaut de longueur de paquet en couche 3	128	
valeur par défaut du mode de connexion en couche 3	Auto	Auto – agit comme ETTD s'il est appelant; agit comme ETCD s'il est appelé. dxe – utilise le paquet de redémarrage pour déterminer s'il doit jouer le rôle d'ETTD ou d'ETCD selon l'ISO/CEI 8208 [2] dte – agit comme ETTD dce – agit comme ETCD
plus petit nombre de circuits SVC entrants (LIC)	1	
plus grand nombre de circuits SVC entrants (HIC)	1	
plus petit nombre de circuits SVC mixtes (LTC)	0	
plus grand nombre de circuits SVC mixtes (HTC)	0	
plus petit nombre de circuits SVC sortants (LOC)	0	
plus grand nombre de circuits SVC sortants (HOC)	0	
temporisateurs de couche 3		le NAF peut choisir de fournir à l'utilisateur du PUF la possibilité de configurer les temporisateurs
valeur par défaut du mode de connexion en couche 2	Auto	Auto – agit comme ETTD s'il est appelant; agit comme ETCD s'il est appelé. dte comme défini dans l'ISO/CEI 7776 dce comme défini dans l'ISO/CEI 7776
arithmétique des canaux B en couche 2	8	NOTE – Doit être 128 pour X.25 sur canal D
longueur de fenêtre en couche 2	7	
longueur de trame en couche 2	128	
type d'activation en couche 2	Case1	Case1 – envoi de la commande SABM/SABME s'il est appelant; pas d'envoi s'il est appelé. Case2 – envoi de la commande SABM/SABME s'il est appelé; pas d'envoi s'il est appelant. Passive – pas d'envoi de la commande SABM/SABME lors de l'activation Active – envoi de la commande SABM/SABME lors de l'activation
temporisateurs en couche 2		
– T1	5	valeur exprimée en secondes
– T2	1	valeur exprimée en secondes
– N2	5	nombre maximal de tentatives de retransmission

I.3 Protocole T.70

Tableau I.3 – Configuration du plan U pour le protocole T.70

Paramètre	Valeur par défaut suggérée	Commentaire
longueur de paquet maximale en couche 3	2048	
valeur par défaut de longueur de paquet en couche 3	128	
temporisateurs en couche 2		
– T1	5	valeur exprimée en secondes
– T1	1	valeur exprimée en secondes
– N2	5	nombre maximal de tentatives de retransmission

Appendice II

Diagrammes SDL pour les dispositifs NAF

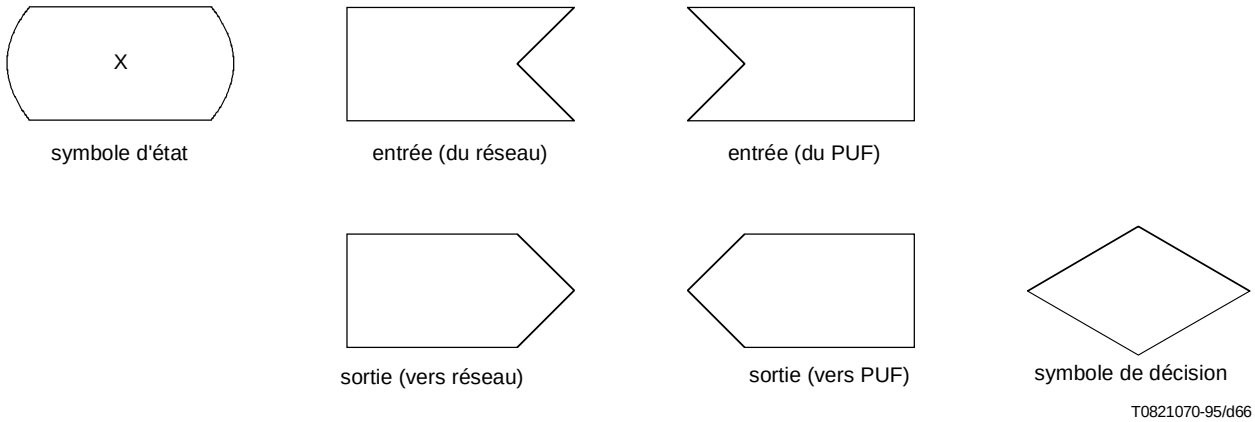
Le mappage des messages du plan U avec deux du protocole dépend de la fonction de coordination qui peut être assurée par le dispositif NAF pour une connexion particulière du plan de commande.

Si le dispositif NAF assure la fonction de coordination, le mappage des primitives de service X.213 sur les messages Q.931 [3] et sur les paquets X.25 est conforme à l'ISO/CEI 9574 et l'ISO/CEI 8878.

Si le dispositif NAF n'assure pas la fonction de coordination, le mappage des primitives de service X.213 sur les paquets X.25 est conforme à l'ISO/CEI 8878.

Certains diagrammes en langage SDL sont donnés ci-dessous afin d'expliquer plus clairement la relation entre messages du plan U et primitives du réseau. Ces diagrammes ne couvrent pas tous les cas mais ne présentent que certaines situations possibles.

Les symboles suivants sont utilisés dans cette description. La Recommandation Z.100 donne une description complète des symboles et de leur signification.



II.1 Protocole T.90

Le Tableau II.1 montre le mappage des messages du plan U sur les primitives de service X.213.

Tableau II.1 – Mappage entre messages du plan U et messages du protocole T.90

Message d'interface PCI	Primitive de service X.213
UConnectReq	demande N-CONNECT
UConnectInd	indication N-CONNECT
UConnectRsp	réponse N-CONNECT
UConnectCnf	confirmation N-CONNECT
UDisconnectReq	demande N-DISCONNECT
UDisconnectInd	indication N-DISCONNECT
UDataReq	demande N-DATA
UDataInd	indication N-DATA
UResetReq	demande N-RESET
UResetInd	indication N-RESET
UResetRsp	réponse N-RESET
UResetCnf	confirmation N-RESET
UReadyToReceiveReq	pas de primitive X.213 équivalente
UReadyToReceiveInd	pas de primitive X.213 équivalente

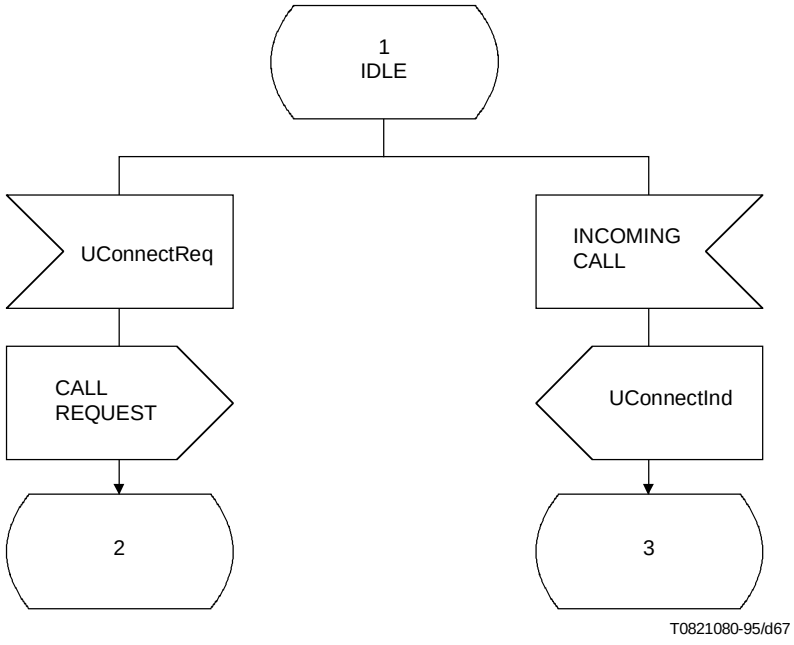


Figure II.1 – Etat de repos

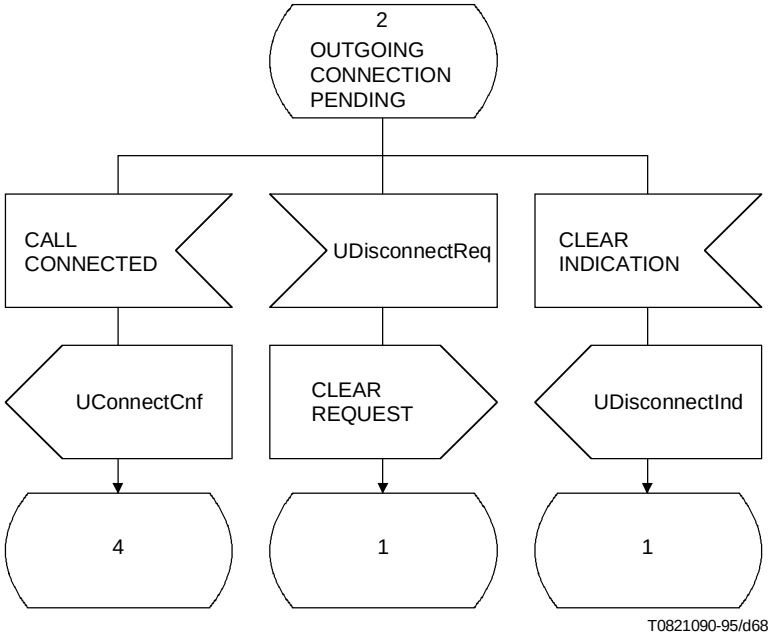


Figure II.2 – Connexion sortante en instance

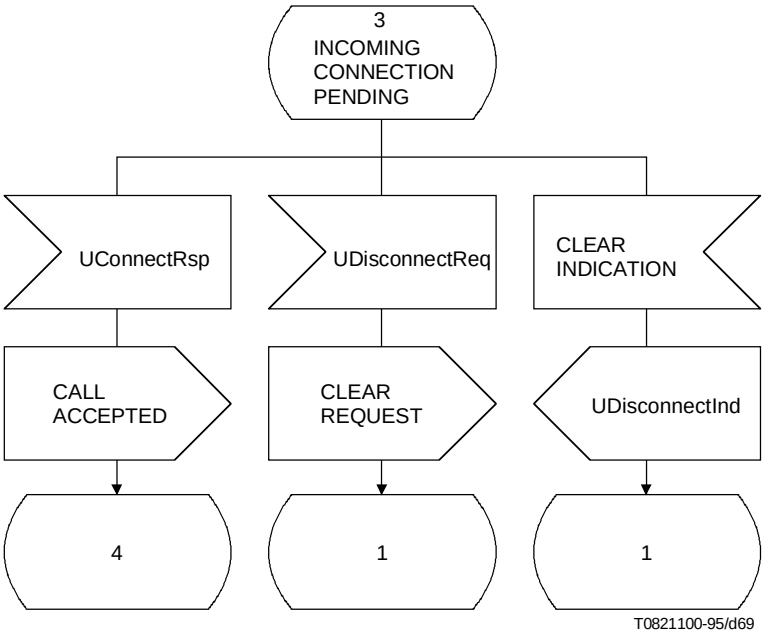


Figure II.3 – Connexion entrante en instance

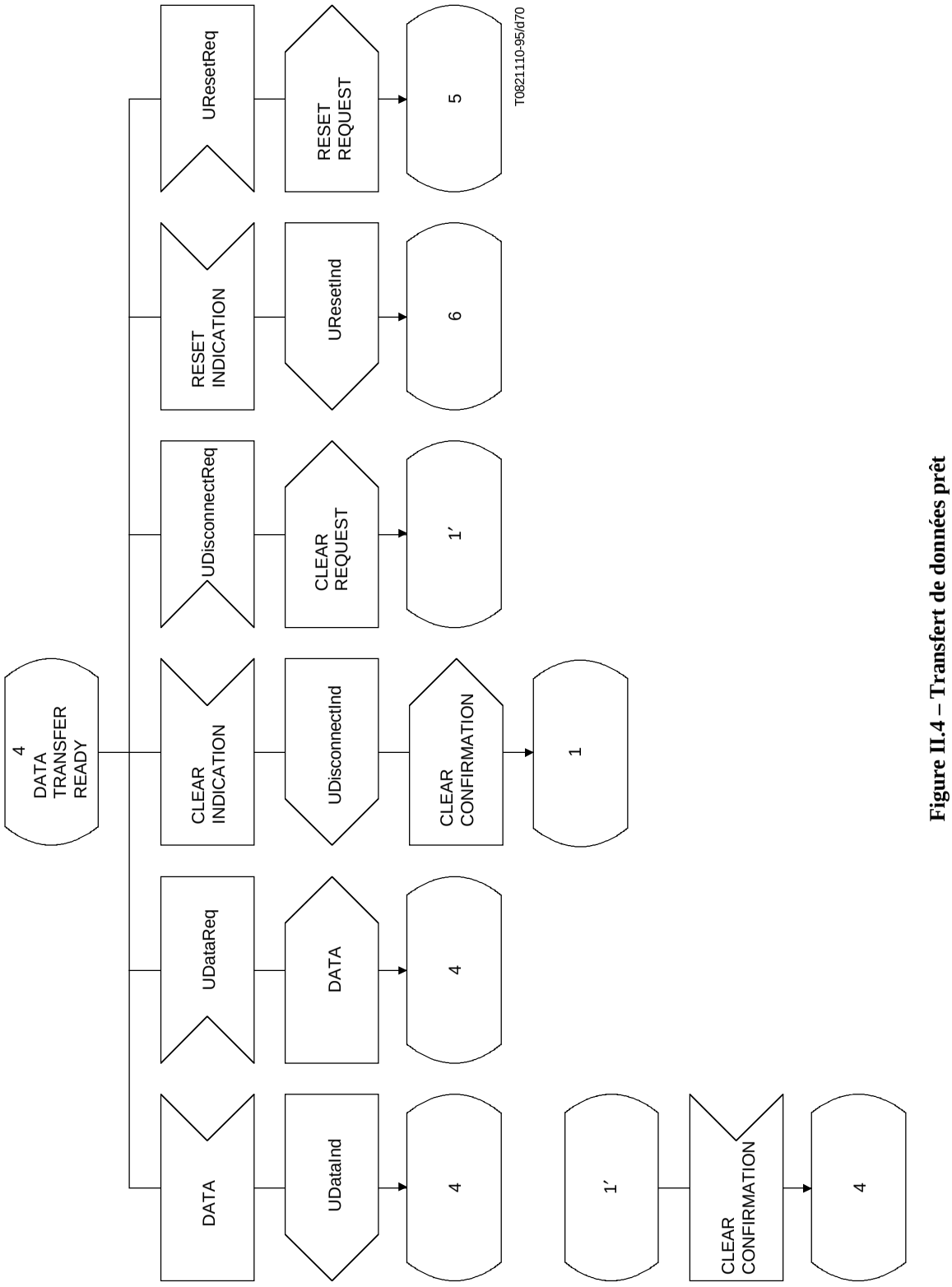


Figure II.4 – Transfert de données prêt

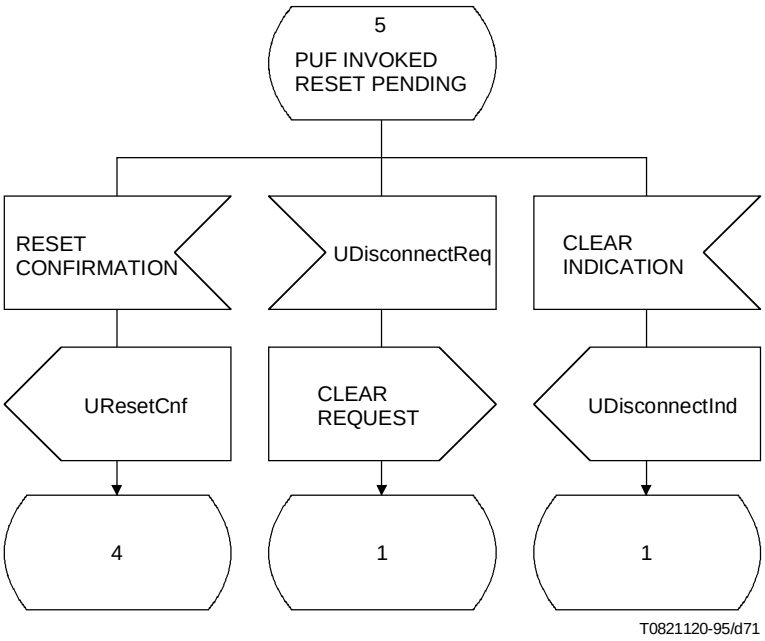


Figure II.5 – Réinitialisation invoquée par PUF en instance

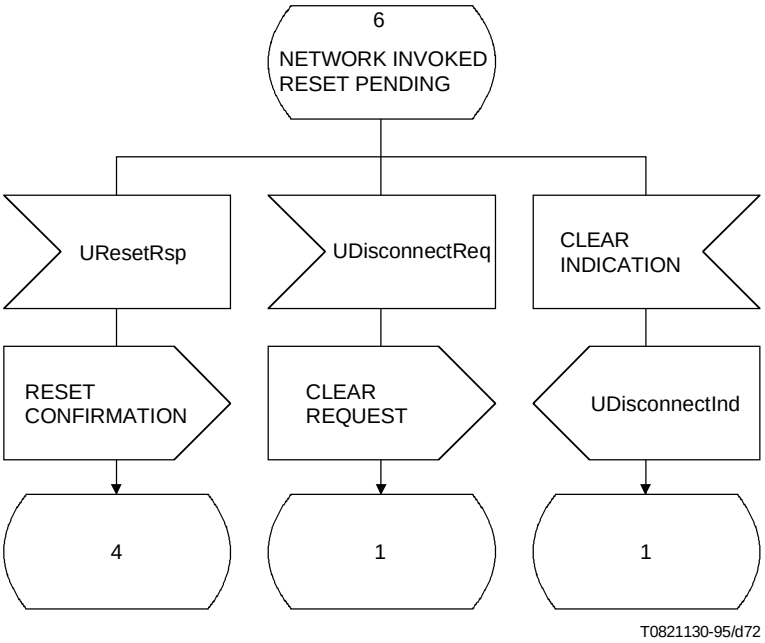


Figure II.6 – Réinitialisation invoquée par réseau en instance

II.2 Protocole ISO/CEI 8208

Le Tableau II.2 montre le mappage des messages du plan U sur les primitives de service X.213.

Tableau II.2 – Mappage entre messages du plan U et messages du protocole ISO/CEI 8208

Message d'interface PCI	Primitive de service X.213
UConnectReq	demande N-CONNECT
UConnectInd	indication N-CONNECT
UConnectRsp	réponse N-CONNECT
UConnectCnf	confirmation N-CONNECT
UDisconnectReq	demande N-DISCONNECT
UDisconnectInd	indication N-DISCONNECT
UDataReq	demande N-DATA
UDataInd	indication N-DATA
UExpeditedDataReq	demande N-EXPEDITED-DATA
UExpeditedDataInd	indication N-EXPEDITED-DATA
UResetReq	demande N-RESET
UResetInd	indication N-RESET
UResetRsp	réponse N-RESET
UResetCnf	confirmation N-RESET
UDataAcknowledgeReq	demande N-DATA-ACKNOWLEDGE
UDataAcknowledgeInd	indication N-DATA-ACKNOWLEDGE
UReadyToReceiveReq	pas de primitive X.213 équivalente
UReadyToReceiveInd	pas de primitive X.213 équivalente

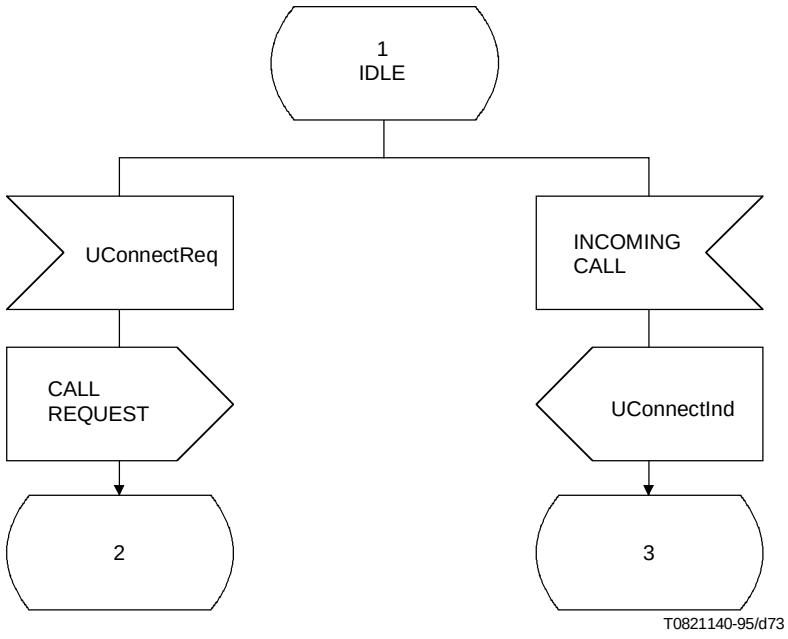


Figure II.7 – Etat de repos

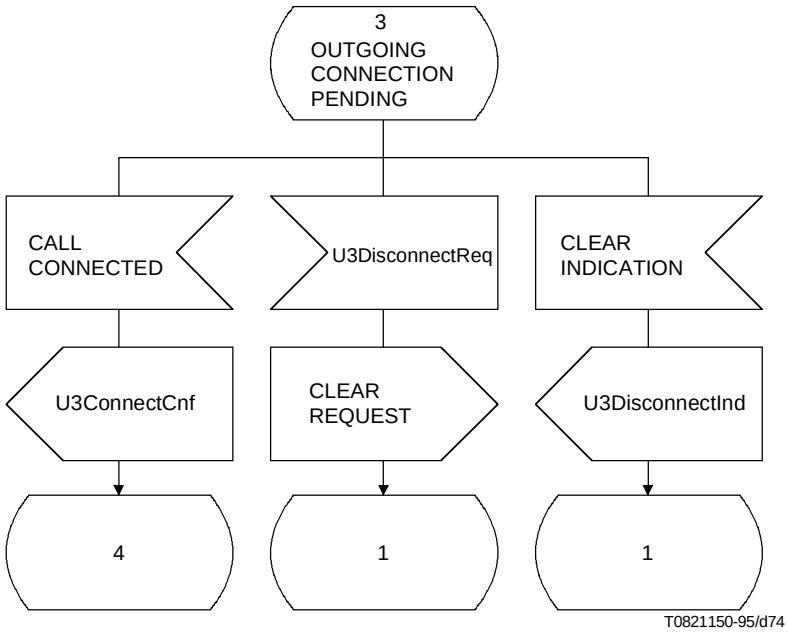


Figure II.8 – Connexion sortante en instance

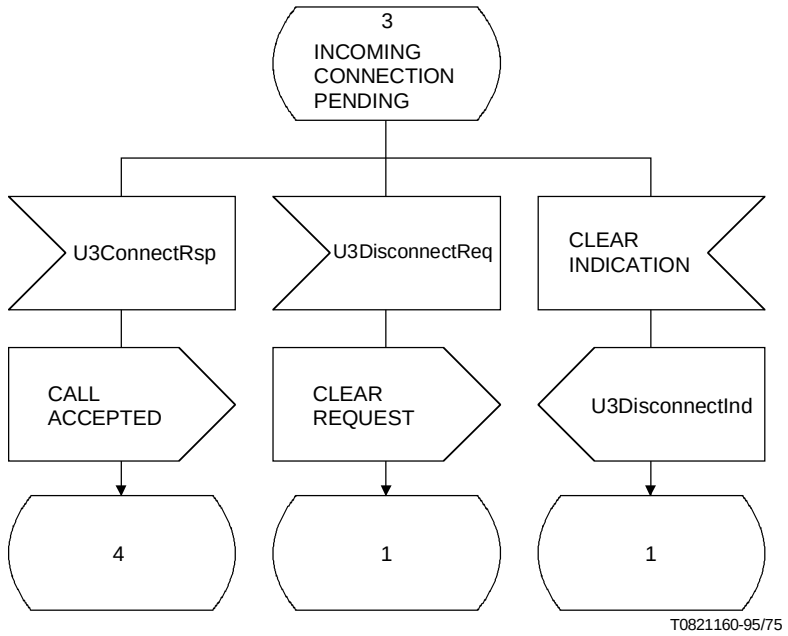


Figure II.9 – Connexion entrante en instance

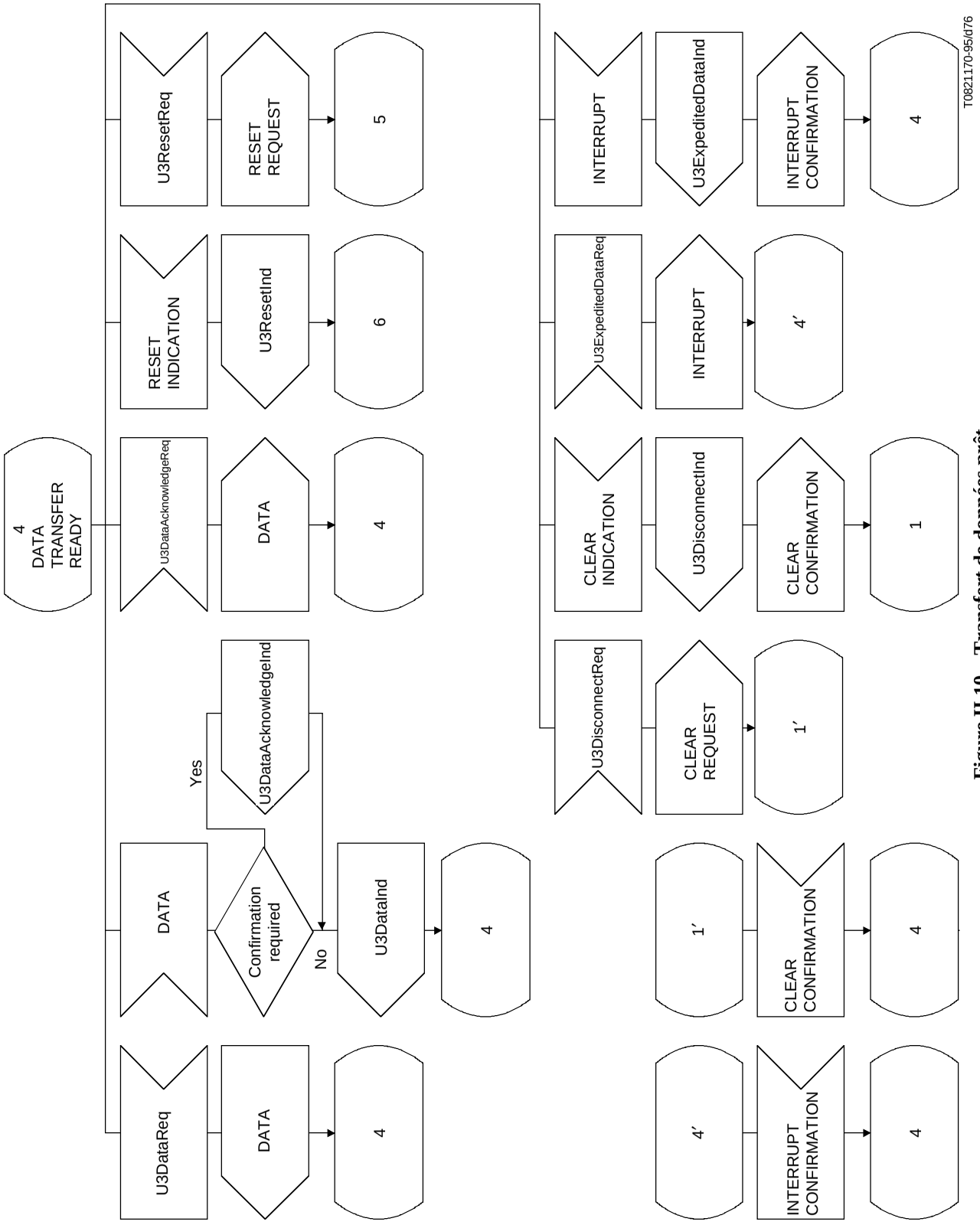


Figure II.10 – Transfert de données prêt

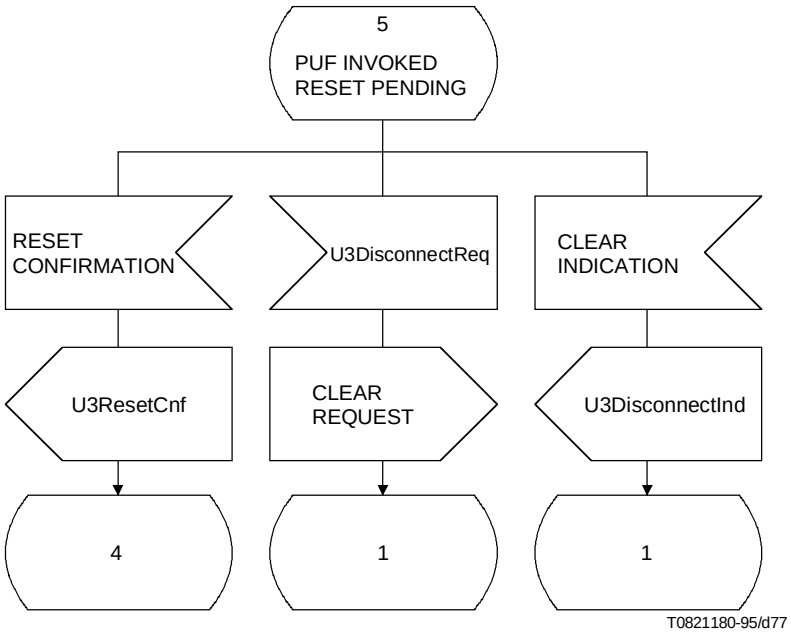


Figure II.11 – Réinitialisation invoquée par le PUF en instance

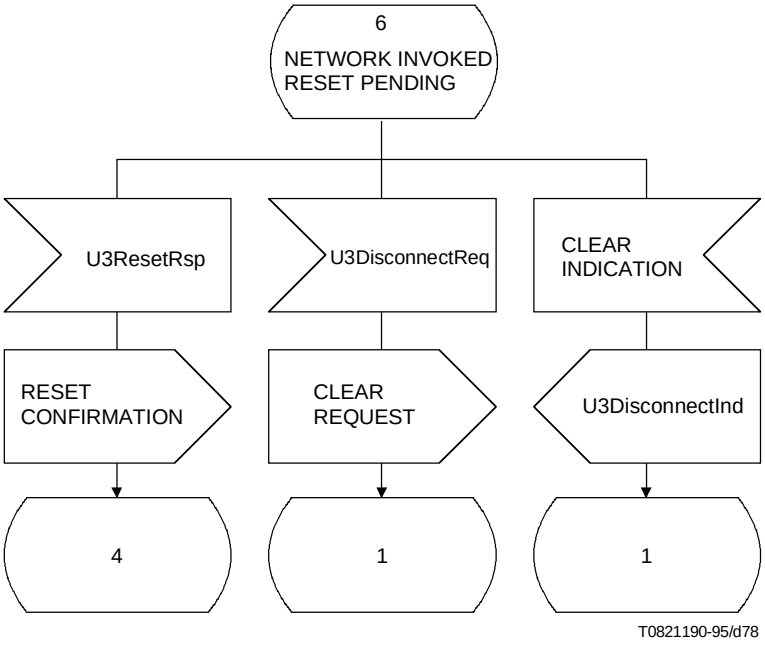


Figure II.12 – Réinitialisation invoquée par le réseau en instance

Remplacée par une version plus récente

Appendice III

Utilisation de la Recommandation X.25

III.1 Valeurs paramétriques pour l'utilisation de la Recommandation X.25

Le Tableau III.1 montre le réglage requis des paramètres pour obtenir différents types d'opérations conformes à la Recommandation X.31.

Tableau III.1 – Types d'opérations conformes à la Recommandation X.31

Type d'opération X.31	Capacité support	Sélection de canal	Numéro de canal	Numéro du demandé
cas A selon X.31, circuits commutés	64 kHz	non requis	non requis	requis
cas A selon X.31, circuits permanents	64 kHz	canal B	requis	non requis
cas B selon X.31, canal B commuté	X.25	non requis	non requis	non requis
cas B selon X.31, canal B permanent	X.25	canal B	requis	non requis
cas B selon X.31, canal D	X.25	canal D	requis	non requis

III.2 Déconnexion d'un canal RNIS avec connexions établies selon la Recommandation X.25

Dans le cas où la fonction de coordination est assurée, cette opération est décrite par l'ISO/CEI 9574.

Si la fonction de coordination n'est pas assurée, il convient de fournir les informations suivantes au dispositif PUF:

- message CDisconnectInd, avec la cause de la déconnexion du canal;
- pour chaque connexion établie selon la Recommandation X.25:
  - message U3DisconnectInd avec les paramètres X213Cause et X213Origin tels que définis par l'ISO/CEI 9574, plus le paramètre X25Cause;
- pour chaque connexion en cours d'établissement selon la Recommandation X.25:
  - message U3DisconnectInd avec les paramètres X213Cause et X213Origin tels que définis par l'ISO/CEI 9574, plus le paramètre X25Cause.

Remplacée par une version plus récente

TABLE DES MATIÈRES

PARTIE 7

PARTIE 7: MÉCANISME D'ÉCHANGE DOS

Résumé

La présente partie de la spécification définit tous les détails du système d'exploitation associé à un environnement MS-DOS™. (On trouvera dans la Partie 2 une présentation générale du mécanisme d'association.)

Introduction

La diversité des interfaces de programmation utilisées par des équipements terminaux pour RNIS (réseau numérique à intégration de services) a fait obstacle au développement d'applications RNIS et est apparue comme une contrainte limitant l'usage d'équipements terminaux modernes pour le RNIS.

La présente spécification définit pour l'UIT-T l'interface entre programmes d'application (API, *application programming interface*), appelée interface de programmation de communication (PCI, *programming communication interface*) par réseau numérique à intégration de services (PCI-RNIS), qui permet aux applications d'accéder au RNIS et de le gérer.

L'interface PCI-RNIS a été définie de façon à offrir aux fournisseurs d'équipement terminal une norme qui permettra de réaliser la portabilité d'applications utilisant l'interface PCI-RNIS sur une gamme d'équipements terminaux tournant sur différents systèmes d'exploitation.

L'interface PCI-RNIS a été définie en fonction des besoins des développeurs d'applications et, dans la mesure du possible, élimine la nécessité d'une connaissance approfondie du RNIS. Elle a également été conçue de façon que les futures extensions du RNIS n'aient pas d'incidence sur le fonctionnement des applications existantes.

Remplacée par une version plus récente

1 Domaine d'application

La présente partie spécifie, pour le système d'exploitation MS-DOS™, l'interface de programmation de communication pour réseau numérique à intégration de services (PCI-RNIS). Elle fait partie de la série de la spécification PCI-RNIS.

Elle décrit la façon dont il y a lieu qu'un dispositif PUF ou NAF, tel que décrit dans la Partie 2, dialogue ou communique par messages et paramètres afin d'établir une connexion RNIS.

D'autres parties spécifieront la méthode d'essai et les prescriptions détaillées concernant chaque application, afin d'évaluer la conformité à la présente partie.

2 Références

[1] Partie 1, *Architecture générale.*

[2] Partie 2, *Services de base.*

3 Définitions

La présente partie définit les termes suivants:

3.1 **fonction d'échange**: propriété du dispositif PUF permettant de réaliser le mécanisme d'échange.

3.2 **mécanisme d'échange**: moyen permettant au dispositif PUF d'échanger des messages avec le dispositif NAF.

3.3 **interface RNIS de programmation de communication (PCI-RNIS)**: interface logicielle orientée réseau (RNIS) qui offre des possibilités de programmer la signalisation de réseau et l'échange de données d'utilisateur.

3.4 **message**: unité d'information transférée de part et d'autre de l'interface PCI-RNIS, entre le dispositif d'accès réseau (NAF) et le dispositif utilisateur d'interface PCI (PUF).

3.5 **dispositif d'accès réseau (NAF)**: unité fonctionnelle située entre l'interface PCI-RNIS et les couches associées au réseau.

3.6 **dispositif utilisateur d'interface PCI (PUF)**: unité fonctionnelle faisant appel à l'interface PCI-RNIS pour accéder à un dispositif NAF. Ce terme correspond pratiquement à l'application locale qui utilise l'interface.

4 Abréviations

La présente partie utilise les abréviations suivantes:

API	interface de programmation d'application (*application programming interface*)
DOS	appellation générique des systèmes d'exploitation compatibles avec le système d'exploitation MS-DOS™
MS-DOS	marque commerciale de la société Microsoft Corporation INC
NAF	dispositif d'accès réseau (*network access facility*)
PCI	interface de programmation de communication (*programming communication interface*)
PciMPB	bloc de paramètres pour messages d'interface PCI (*PCI message parameter block*)
PUF	dispositif utilisateur d'interface PCI (*PCI user facility*)
RNIS	réseau numérique à intégration de services

5 Guide de lecture

5.1 Guide du lecteur

La présente partie est destinée aux développeurs de logiciels, aux réalisateurs d'applications et aux constructeurs d'équipement, qui y trouveront la description des mécanismes d'échange et des exemples de codage conformes à [2] pour l'environnement DOS.

5.2 Mode d'emploi de la présente partie

Les lecteurs qui:

- ont besoin d'un aperçu général rapide sur le mécanisme d'échange en général le trouveront dans la Partie 2: "Services de base" [2]. D'autres systèmes d'exploitation sont décrits. Le lecteur est invité à consulter la Partie 1: "Architecture générale" [1], où il trouvera des informations sur la possibilité d'utiliser d'autres systèmes d'exploitation. Le 6.1 contient une description générale du mécanisme d'échange pour le DOS;
- envisagent d'implémenter une application DOS au moyen de la présente interface PCI-RNIS sont invités à consulter les paragraphes 5 et 6. Les paragraphes 3 et 4 donnent des renseignements utiles sur les termes et abréviations utilisés. Des exemples de codage sont présentés en Appendice I. Le lecteur trouvera dans la Partie 2: "Services de base" [2] des informations sur les paramètres ainsi que des listes et des descriptions des codes de retour;
- ont l'intention de construire une carte ou un équipement d'adaptation au RNIS utilisant l'interface PCI-RNIS sont également invités à lire les paragraphes 5 et 6. Les paragraphes 3 et 4 donnent d'utiles informations sur la définition des termes et abréviations utilisés. On trouvera au 6.4 de plus amples renseignements sur les dispositifs NAF. Les exemples de codage présentés dans l'Appendice I montrent comment un dispositif PUF peut accéder à un dispositif NAF. Le lecteur trouvera dans la Partie 2: "Services de base" [2] des informations sur les paramètres ainsi que des listes et des descriptions des codes de retour.

Le Tableau 1 donne une liste qui décrit le contenu général de la présente partie.

Tableau 1 – Table des matières de la présente partie

Paragraphe, appendice	Contenu
paragraphe 1	domaine d'application de la présente partie, décrivant son objet
paragraphe 2	références
paragraphe 3	définitions de termes utilisés dans la présente partie
paragraphe 4	définitions d'abréviations utilisées dans la présente partie
paragraphe 5	aperçu général et directives de lecture
paragraphe 6	prescriptions relatives au système d'exploitation et règles d'implémentation pour le DOS
Appendice I	codage d'échantillons en langage C pour illustrer l'implémentation spécifique du système d'exploitation DOS au mécanisme d'échange de messages

6 Implémentation spécifique du système d'exploitation DOS

La présente partie décrit l'implémentation spécifique d'un système d'exploitation pour le cas du DOS. Pour la description suivante, la version de base du logiciel MS-DOS est le numéro 3.1.

En DOS, une implémentation de dispositif NAF doit offrir la propriété de fonctions d'échange qui est décrite de façon générique en [2].

Dans la présente partie, le mappage et l'implémentation de ces fonctions sont décrits fonction par fonction. Pour chacune, on donne un exemple de codage en langage C.

6.1 Introduction

A l'exception de la fonction **PciGetHandles** (obtention de pointeurs d'interface PCI), l'implémentation de la méthode d'échange en DOS est fondée sur un mécanisme d'accès direct. Le point d'accès est une adresse de fonction distante fournie par le dispositif NAF. Cette adresse de fonction est mappée avec le type générique PCI_HANDLE.

Pour s'assurer que l'adresse de fonction fournie par le dispositif NAF est correcte, le dispositif PUF peut, avant d'appeler le NAF, vérifier une signature située en face de l'adresse de fonction.

Pour effectuer cette vérification, le dispositif PUF doit examiner la zone de mémoire située exactement en face de l'adresse de fonction. C'est là que se trouve la signature, qui contient la constante suivante, sous la forme d'une chaîne de huit caractères: "PCI-RNIS". Si cette signature est présente, le dispositif PUF admet que l'adresse de fonction fournie par le dispositif NAF est correcte.

Un seul point d'accès doit être offert par le dispositif NAF. Un paramètre de message doit indiquer la fonction à invoquer. Ce paramètre est appelé **code de fonction**.

Les paramètres sont transmis du PUF au NAF au moyen de la mémoire en pile (à liste refoulée). Le dispositif PUF doit offrir une pile minimale de 128 octets par appel. Lorsque le dispositif NAF reçoit la commande issue de l'unité centrale, le premier paramètre de la pile est le code de fonction, suivi des paramètres correspondant à cette fonction.

Le code de fonction est transmis sous la forme d'une valeur d'entier sur 2 octets.

Le dispositif NAF doit placer le code de retour (point de sortie de procédure) dans le registre AX. La procédure assurée par le dispositif NAF ne comporte pas le vidage de la pile au point de sortie. La convention d'appel suivante est utilisée en langage C: le dispositif PUF appelant décale les paramètres de droite à gauche et reconstitue la pile en sortie.

L'alignement de la structure générique des blocs PCI-MPB s'effectue par **octets**.

6.2 Mappage des types génériques et des constantes

En DOS, les mappages suivants doivent être utilisés pour les types génériques décrits dans la Partie 2: Services de base [2]:

Type générique	Mappage en DOS
PCI_INTEGER	entier sur 2 octets (un mot)
PCI_BYTEARRAY	pointeur d'adresse distante (adresse segment – décalage)
PCI_EXID	identificateur unique fourni par NAF (entier sur 2 octets)
PCI_HANDLE	adresse de fonction distante (adresse segment – décalage)
PCI_PROCEDURE	adresse de fonction distante (adresse segment – décalage)

Comme d'habitude en DOS, toutes les valeurs sont présentées dans l'ordre "petit-boutiste" (premier octet inférieur au second).

Le **code de fonction** utilisé pour invoquer les fonctions d'échange doit être attribué comme suit aux noms de fonction:

Fonction	Valeur du code de fonction
PciGetProperty	1
PciRegister	2
PciDeregister	3
PciPutMessage	4
PciGetMessage	5
PciSetSignal	6

En langage C, ces définitions se présenteront comme suit:

```
/*
 * Mappages des types génériques
 */
typedef short int           PCI_INTEGER;
typedef char far *          PCI_BYTEARRAY;
typedef short int           PCI_EXID;
typedef short int           (far *      PCI_HANDLE) ();
typedef void                (far *      PCI_PROCEDURE) ();
```

```
/*
 * Constantes de codes de fonction
 */
#define  PCIGETPROPERTY                1
#define  PCIREGISTER                   2
#define  PCIDEREGISTER                 3
#define  PCIPUTMESSAGE                 4
#define  PCIGETMESSAGE                 5
#define  PCISETSIGNAL                  6

/*
 * Signature
 */
#define  PCISIGNATURE          'ISDN-PCI'      /* constante à plusieurs caractères */
```

6.3 Description des fonctions

Le dispositif PUF est chargé d'offrir une mémoire de pile minimale au cours d'un appel de fonction. La capacité minimale d'une mémoire de pile est de 128 octets.

Dans les exemples de codage, on ne décrit l'accès qu'à un seul dispositif NAF, pour plus de clarté. L'accès d'un dispositif PUF à plusieurs dispositifs NAF n'est toutefois pas exclu.

6.3.1 PciGetHandles

En DOS, l'implémentation de la fonction **PciGetHandles** doit faire appel à un module de gestion en mode caractère nommé "PCIDD$", permettant de récupérer les pointeurs d'interface PCI disponibles. Cet appel de fonction fait exception au principe – accès direct – en DOS.

Le nombre maximal de pointeurs PCI pouvant être récupérés est théoriquement de 4096. En pratique, le gestionnaire implémenté aura probablement une limite bien inférieure, déterminée par la nature même de ce module.

Le dispositif PUF doit effectuer les opérations suivantes, dans cet ordre:

- ouvrir le module de gestion en mode caractère "PCIDD$";
- préparer un tampon en mémoire, assez grand pour contenir le nombre maximal de pointeurs PCI à récupérer;
- émettre un appel système de lecture par commande IOCTL: récupération des données de commande dans le gestionnaire de caractères;
  - le registre BX doit contenir le pointeur DOS sur le gestionnaire;
  - le registre CX doit contenir la longueur de la mémoire tampon préparée ci-dessus;
  - l'adresse DS:DX doit délimiter le tampon;
- vérifier le succès de l'opération (contrôle de l'indicateur de report);
- en cas d'erreur, émettre par exemple un appel de fonction DOS de type "obtention du code d'erreur étendu" afin de recevoir un code d'erreur plus détaillé;
- en sortie normale, le registre AX contient le nombre d'octets fournis par le gestionnaire et le tampon contient la séquence des pointeurs PCI disponibles. Le nombre de pointeurs PCI disponibles se calcule en divisant la valeur AX par le nombre 4 qui correspond à la longueur d'un pointeur de fonction d'adressage distant;
- fermer le module gestionnaire.

Exemple de codage en C:

```
...
#include  <dos.h>                    /* déclarations pour appel de commande IOCTL */
#include  <fcntl.h>                  /* déclarations pour mode d'ouverture */

...
#define  SUCCESS          0          /* pas d'erreur */
#define  MAXHANDLES       64         /* nombre maximal de pointeurs à lire */
```

```
...
PCI_HANDLE PCIHandlesArray[MAXHANDLES]     /* tampon pour recevoir pointeurs PCI */

...
PCI_INTEGER MaxHandles;                    /* nombre maximal de pointeurs à lire */
PCI_HANDLE far * PCIHandles;               /* pointeur d'adresse distante du tampon de pointeurs PCI */
PCI_INTEGER far * ActualHandles;           /* pointeur d'adresse distante pour le nombre de pointeurs PCI
                                              reçus */

{
int fildes;                                /* descripteur de fichier */
int error;
union _REGS regs;
struct _SREGS segregs;
struct _DOSERROR errorinfo;
/* ouvrir le gestionnaire */
if (_dos_open ("PCIDD$", _O_RDWR, &fildes) != SUCCESS)
        {
        /* gestionnaire inaccessible; effectuer traitement d'erreur */
        error = ...
        }

else
        {
        /* préparer la commande IOCTL de lecture dans le gestionnaire */
        _segread (&segregs);
        segregs.ds = FP_SEG (PCIHandles);          /* préparer adresse segment */
        regs.x.dx = FP_OFF (PCIHandles);           /* et décalage */
        regs.x.cx = MaxHandles * sizeof(PCI_HANDLE);
        regs.x.bx = fildes;                        /* fixer pointeur de fichier DOS */
        regs.x.ax = 0x4402;                        /* préparer commande IOCTL de lecture dans gestionnaire
                                                      caractères */

        /* émettre commande de lecture dans gestionnaire */
        _intdosx (&regs, &regs, &segregs);

        /* fermer le gestionnaire */
        _dos_close (fildes);

        /* vérifier présence d'erreur */
        if (regs.x.cflag & 1)                      /* vérifier indicateur de report de processeur */
                {
                /* une erreur s'est produite; effectuer traitement d'erreur */
                _dosexterr (&errorinfo);
                error = doserror.exterror;
                ...
                }

        else
                {
                /* Opération réussie. Poser nombre de pointeurs reçus */
                *ActualHandles = regs.x.ax / sizeof(PCI_HANDLE);
                error = SUCCESS;
                }
...
```

6.3.2 PciGetProperty

Cette fonction a pour objet d'obtenir du NAF l'information "Propriété-NAF". Pour émettre cet appel de fonction, le dispositif PUF doit posséder le pointeur-PCI du NAF auquel il souhaite accéder. Avant d'accéder au dispositif NAF, le dispositif PUF peut vérifier si le pointeur-PCI qu'il utilise est valide. A cette fin, il contrôle la signature du point d'accès désigné par le pointeur-PCI.

Les opérations suivantes doivent être effectuées dans l'ordre indiqué par le dispositif PUF:

- examiner la zone de mémoire éventuellement désignée par le pointeur-PCI afin de déterminer si l'adresse du NAF est chargée et vérifier dans ce cas la signature représentée par la constante en caractères "PCI-RNIS";
- appeler l'adresse avec le **code de la fonction** PciGetProperty et avec les paramètres fournis par le dispositif PUF;
- contrôler le code de retour (au point de sortie).

Exemple de codage en langage C:

```
...
#include  <memory.h>                          /* comparer en mémoire les déclarations de fonction */

...
#define  SUCCESS                  0           /* pas d'erreur */
#define  PCIGETPROPERTY           1
#define  PCISIGNATURE             "ISDN-PCI"
#define  SIGNATURESIZE            8

...
PCI_HANDLE PCIHandle;
PCI_INTEGER MaximumSize;
PCI_BYTEARRAY Property;
PCI_INTEGER far * ActualSize;
{
PCI_INTEGER error;
char far * signature;

signature = (char far *) PCIHandle – SIGNATURESIZE;
if (_fmemcmp (signature,PCISIGNATURE,SIGNATURESIZE) == SUCCESS)
        {
        /* signature correcte. appel du point d'entrée */
        error = (*PCIHandle) (PCIGETPROPERTY, MaximumSize, Property, ActualSize);
        ...
        }

else

        {
        /* signature erronée. traitement d'erreur */
        error = ...

...
```

6.3.3 PciRegister

Cette fonction a pour objet d'établir une association entre un dispositif PUF et un dispositif NAF. Pour émettre l'appel de cette fonction, le dispositif PUF doit toujours être en possession du pointeur-PCI sur le dispositif NAF auquel il souhaite accéder. Avant d'effectuer cet accès, le dispositif PUF peut vérifier si le pointeur-PCI qu'il utilise est valide. Pour cela, il contrôle la signature du point d'accès désigné par le pointeur-PCI.

Pour l'appel de cette fonction, le dispositif PUF doit préparer 2 structures et les envoyer dans la pile de fonctions. La première structure est l'information PCIRegisterInfo qui est déclarée conformément à la Partie 2 [2]. La deuxième est l'information PCIOpSysInfo, qui dépend du système d'exploitation et à laquelle le DOS donne la présentation suivante.

Nom de l'élément	Type	Validité	Explication
MaxNCOCount	entier sur 2 octets	pendant l'appel de fonction	cet élément doit indiquer le nombre maximal d'objets NCO que le PUF a l'intention de créer au cours de l'association.
MaxPacketSize	entier sur 2 octets	pendant l'appel de fonction	cet élément doit indiquer la longueur maximale d'un paquet de données, que le NAF doit accepter pour une connexion du plan U.
MaxPacketCount	entier sur 2 octets	pendant l'appel de fonction	cet élément doit indiquer le nombre maximal de paquets de longueur maximale, que le NAF doit mettre en tampon pour chaque connexion U.
AddBufferSize	entier sur 4 octets	pendant l'appel de fonction	si le PUF souhaite fournir de l'espace tampon au NAF, il doit régler la valeur de cet élément sur la taille souhaitée du tampon. Sinon la valeur est mise à zéro.
AddBufferSpace	adresse distante (segment: décalage)	pendant l'appel de fonction	si l'élément structurel AddBufferSize a une valeur non nulle, cet élément doit pointer (à distance) sur l'espace tampon additionnel qui a été concédé.
BufferNeeded	entier sur 4 octets	au retour de fonction	si le NAF ne dispose pas d'assez d'espace tampon pour garantir les caractéristiques requises pour la connexion, le NAF sort en retour vers cet élément pour indiquer la capacité tampon additionnelle dont il a besoin.

L'information fournie par cette structure aide le dispositif NAF à optimiser ses ressources internes. Les informations fournies par le dispositif PUF doivent donc être évaluées avec précision, cela étant d'autant plus vrai qu'il s'agit d'un environnement dans lequel un dispositif NAF dessert plusieurs dispositifs PUF simultanément.

Si un dispositif NAF ne dispose pas de ressources suffisantes en terme de mémoire pour garantir les caractéristiques requises, la fonction PciRegister va échouer et renvoyer un code d'erreur de type BuffersTooSmall. Dans ce cas, la capacité mémoire manquante peut être extraite au moyen de l'élément BufferNeeded de la structure ci-dessus.

Dès que la fonction PciRegister a retourné un code de succès, l'identificateur d'échange (EXID) devient disponible et doit être utilisé comme paramètre lors d'ultérieurs appels de fonction dans le cadre du mécanisme d'échange.

Les opérations suivantes doivent être effectuées par le dispositif PUF, dans cet ordre:

- examiner la zone mémoire désignée par le pointeur-PCI pour déterminer si l'adresse du NAF est chargée. Contrôler les caractères "PCI-RNIS" de la signature;
- affecter et préparer les deux structures PCIRegisterInfo et PCIOpSysInfo. En option, la structure PCIOpSysInfo peut contenir un pointeur désignant un espace tampon additionnel, qui doit être affecté au NAF;
- appeler la fonction d'échange au moyen du **code de fonction** PciRegister et des paramètres fournis par le PUF;
- contrôler le code de retour. Si ce code indique une erreur de type OutOfBuffers, l'appel peut être répété avec réglage correct de l'espace tampon à attribuer au NAF;
- conserver EXID retourné, en vue d'un appel ultérieur.

Exemple de codage en langage C:

```
...
#include  <memory.h>                    /* comparer en mémoire les déclarations de fonction */
#include  <malloc.h>                    /* fonctions d'attribution des zones mémoire */

...
#define  SUCCESS                0       /* pas d'erreur */
#define  PCIREGISTER            2
#define  PCISIGNATURE           'ISDN-PCI'
#define  SIGNATURESIZE          8
#define  E_OUT_OF_BUFFERS       148     /* code d'erreur de type BuffersTooSmall */
```

```
...
struct pci_register {                                   /* structure contenant les informations d'association */
        PCI_INTEGER     PUFVersion;                     /* sur option: indiquer version du PUF */
        PCI_INTEGER     PUFType;                        /* sur option: indiquer type du PUF */
        PCI_INTEGER     MaxMsgSize;                     /* retour: longueur maximale d'un message */
};

struct pci_opsys {                                      /* structure contenant les informations d'association */
        short int       MaxNCOCount;                    /* sur option: indiquer nombre maximal d'objets NCO */
        short int       MaxPacketSize;                  /* sur option: indiquer longueur maxi et nombre */
        short int       MaxPacketCount;                 /* maxi des paquets en tampon */
        long int        AddBufferSize;                  /* sur option: donner au NAF les longueurs et pointeurs */
        void far *      AddBufferSpace;                 /* pour le tampon additionnel */
        long int        BufferNeeded;                   /* retour: capacité de tampon addit. requise */
};

...
/*
 * avant d'effectuer d'ultérieurs appels à la fonction PCIRegister, affecter et préparer les structures
 * requises par l'appel de cette fonction
 */

struct pci_register PCIRegisterInfo {
        2,                      /* Régler à 2 le numéro de version du PUF, ce qui correspond à la version de
                                   la Recommandation en vigueur */
        0,                      /* Régler le type de PUF à 0 comme indiqué en [2] */
        0                       /* Initialiser la valeur (prévue) de retour pour MaxMsgSize */
};
struct pci_opsys PCIOpSysInfo {
        2,                      /* Fixer le nombre maximal d'objets NCO que le PUF prévoit de créer */
        1024,                   /* Fixer la longueur maximale des paquets que le NAF doit accepter */
        8,                      /* Fixer le nombre maximal de paquets que le NAF doit mémoriser pour chaque NCO */
        0,                      /* Fixer la capacité mémoire que le PUF veut allouer au NAF */
        (void far *) NULL,      /* Fixer le pointeur sur l'espace tampon (affecté) */
        0                       /* Initialiser la valeur (prévue) de retour pour BufferNeeded */
};

...
PCI_HANDLE PCIHandle;
struct pci_register far * RegisterStruct;;
PCI_EXID far * ExchangeID
{
PCI_INTEGER error;
char far * signature;
void far * buffer;

signature = (char far *) PCIHandle – SIGNATURESIZE;
if (_fmemcmp (signature,PCISIGNATURE,SIGNATURESIZE) != SUCCESS)
        {
        /* signature erronée. traitement d'erreur */
        error = ...
        }

else

        {
        /* signature correcte. appeler le point d'entrée */
        error = (*PCIHandle) (PCIREGISTER, &PCIRegisterInfo, &PCIOpSysInfo, ExchangeID);
        if (error == E_OUT_OF_BUFFERS)
                {
                /* Le NAF a besoin de plus d'espace tampon; essayer d'en affecter */
                buffer = _fmalloc ((size_t) PCIOpSysInfo.BufferNeeded);
                if (buffer)
```

```
                    {
                    /* il reste du tampon, donc un autre essai se justifie; ajuster la structure PCIOpSysInfo */
                    PCIOpSysInfo.AddBufferSize = PCIOpSysInfo.BufferNeeded;
                    PCIOpSysInfo.AddBufferSpace = buffer;
                    PCIOpSysInfo.BufferNeeded = 0;
                    /* refaire un appel à la fonction PciRegister ... */
                    }

            }
        error=(*PCIHandle)(PCIREGISTER,&PCIRegisterInfo,&PCIOpSysInfo,ExchangeID);
        if (error)
            {
            /* Traitement d'erreur */
            ...
...
```

6.3.4 PciDeregister

Cette fonction a pour objet de dissocier un PUF d'un NAF.

Les opérations suivantes doivent être effectuées par le dispositif PUF, dans cet ordre:

- appeler l'adresse par le **code de la fonction** PciDeregister et avec l'identificateur EXID se rapportant à l'association en cours;
- contrôler le code de retour en sortie de procédure.

Exemple de codage en langage C:

```
...
#define  PCIDEREGISTER              3

...
PCI_HANDLE PCIHandle;
PCI_EXID ExchangeID;
{
PCI_INTEGER error;

/* appeler le point d'entrée */
error = (*PCIHandle) (PCIDEREGISTER, ExchangeID);
...
```

6.3.5 PciPutMessage

Cette fonction a pour objet de transmettre un message de PUF à NAF. Les paramètres doivent être fournis dans l'ordre indiqué par la description générique de la fonction PciPutMessage.

Les opérations suivantes doivent être effectuées par le dispositif PUF, dans cet ordre:

- appeler l'adresse par le **code de la fonction** PciPutMessage et avec l'identificateur EXID se rapportant à l'association en cours, ainsi qu'avec la configuration correcte en terme de bloc de paramètres de messages PCI et des tampons associés;
- contrôler le code de retour en sortie de procédure.

Exemple de codage en langage C:

```
...
#define  PCIPUTMESSAGE              4
...
struct pci_mpb {
                PCI_INTEGER     MessageID;
                PCI_INTEGER     MessageMaximumSize;
```

```
                PCI_INTEGER     MessageActualUsedSize;
                PCI_INTEGER     DataMaximumSize;
                PCI_INTEGER     DataActualUsedSize;

};
...
PCI_HANDLE PCIHandle;
PCI_EXID ExchangeID;
struct pci_mpb far * PCIMpb;
PCI_BYTEARRAY Message;
PCI_BYTEARRAY Data;
{
PCI_INTEGER error;

/* appel du point d'entrée */
error = (*PCIHandle) (PCIPUTMESSAGE, ExchangeID, PCIMpb, Message, Data);
...
}
```

6.3.6 PciGetMessage

Cette fonction a pour objet de fournir au PUF un message venant du NAF. Les paramètres doivent être fournis dans l'ordre indiqué par la description générique de la fonction PciGetMessage.

Les opérations suivantes doivent être effectuées par le dispositif PUF, dans cet ordre:

- appeler l'adresse par le **code de la fonction** PciGetMessage et avec l'identificateur EXID se rapportant à l'association en cours, ainsi qu'avec la configuration correcte en terme de bloc de paramètres de messages PCI et des tampons associés;
- contrôler le code de retour en sortie de procédure.

Exemple de codage en langage C:

```
...
#define  PCIGETMESSAGE              5
...

struct pci_mpb {
                PCI_INTEGER     MessageID;
                PCI_INTEGER     MessageMaximumSize;
                PCI_INTEGER     MessageActualUsedSize;
                PCI_INTEGER     DataMaximumSize;
                PCI_INTEGER     DataActualUsedSize;

};
...
PCI_HANDLE PCIHandle;
PCI_EXID ExchangeID;
struct pci_mpb far * PCIMpb;
PCI_BYTEARRAY Message;
PCI_BYTEARRAY Data;
{
PCI_INTEGER error;

/* appel du point d'entrée */
error = (*PCIHandle) (PCIGETMESSAGE, ExchangeID, PCIMpb, Message, Data);
...
}
```

6.3.7 PciSetSignal

Cette fonction a pour objet de fournir au NAF l'adresse d'une fonction implantée dans le PUF, qui doit être rappelé automatiquement si un message devient disponible pour ce dispositif PUF.

Les opérations suivantes doivent être effectuées par le dispositif PUF, dans cet ordre:

- appeler l'adresse par le **code de la fonction** PciSetSignal et avec l'identificateur EXID se rapportant à l'association en cours, ainsi qu'avec l'adresse correcte de la fonction de configuration de la routine de rappel automatique;
- contrôler le code de retour en sortie de procédure.

Exemple de codage en langage C:

```
#define  PCISETSIGNAL                    6
/* Fonction de rappel appelée au moyen d'une interruption */
void far CallBackFunc ()
{
...
return;
}

/*
 * Code pour configurer le processus de notification
 */

...
PCI_HANDLE PCIHandle;
PCI_EXID ExchangeID;
{
PCI_INTEGER error;

/* appel du point d'entrée */
error = (*PCIHandle) (PCISETSIGNAL, ExchangeID, &CallBackFunc);
...
}
```

Le NAF rappelle le dispositif PUF en appliquant les conventions suivantes:

- Le NAF fournit une pile minimale de 128 octets.
- Les valeurs des segments de données et supplémentaires (DS et ES) sont indéfinies.
- Les interruptions sont désactivées.

Lorsqu'il a repris la main, le dispositif PUF:

- peut activer ou ne pas activer les interruptions;
- est autorisé à appeler le NAF au moyen de la fonction PciGetMessage ou PciPutMessage;
- ne doit pas invoquer d'autre fonction d'échange en dehors des deux fonctions précédentes;
- ne doit pas émettre d'appels propres au système DOS;
- ne doit pas laisser des interruptions désactivées pendant un laps de temps prolongé et doit sortir de la fonction de rappel automatique dès que possible.

Le dispositif NAF appelé par la fonction PCIGetMessage ou PciPutMessage peut activer des interruptions. Il ne doit cependant pas relancer la routine de rappel autoamtique avant que celle-ci ait fait l'objet d'une sortie normale.

A la fin de la routine de rappel, le dispositif PUF doit retourner au NAF. Seule la paire de registres SS:SP doit être préservée par le PUF.

6.4 Disponibilité de pointeurs PCI_HANDLE désignant des dispositifs NAF

Pour devenir accessible au moyen de l'appel de la fonction PciGetHandle, un dispositif NAF doit effectuer une action déclarative. L'action inverse – c'est-à-dire l'extraction à partir de la liste des dispositifs NAF disponibles – est également décrite. Ces actions sont propres au système d'exploitation.

6.4.1 Action déclarative

En DOS, le dispositif NAF utilise le gestionnaire de caractères (PCIDD$) pour se déclarer, en émettant un ordre d'écriture IOCTL qui transfère au NAF une structure contenant le code d'action (déclarative) ainsi que le pointeur correspondant.

Le nombre maximal de dispositifs NAF que le gestionnaire PCIDD$ peut enregistrer pour association est de 32.

Les opérations suivantes doivent être effectuées, dans cet ordre:

- ouvrir le gestionnaire PCIDD$;
- préparer la structure suivante:
  - un seul mot: code de commande, 0x4544 (caractères 'DE', correspondant à "DEclaration");
  - un mot double: adresse du point d'entrée du NAF;
- émettre un ordre IOCTL d'écriture pour un appel système:
  - le registre CX contient la longueur de la structure de déclaration (6);
  - les registres DS:DX désignent la structure;
- vérifier le succès de l'opération (contrôle de l'indicateur de report);
- en cas d'erreur, émettre un appel à la fonction GetExtendedError pour obtenir un code d'erreur plus détaillé;
- fermer le gestionnaire en mode caractère.

Cette commande se terminera normalement, même si le dispositif NAF est déjà déclaré car, dans ce cas, rien ne se passe.

Cette commande donne une erreur dans les cas suivants, qui n'ont aucune suite:

- erreurs DOS typiques (pointeur non valide, numéro de fonction non valide, etc.);
- longueur incorrecte du tampon transféré (registre CX): code étendu d'erreur 24 (longueur incorrecte de la structure de requête);
- code de commande non valide (code étendu d'erreur 31, panne générale);
- 32 NAF déjà déclarés **et** le NAF à déclarer n'est pas encore déclaré (code étendu d'erreur 29, erreur en écriture).

6.4.2 Action extractive

Le dispositif NAF utilise le gestionnaire PCIDD$ pour effectuer une extraction interne en envoyant une commande IOCTL d'écriture qui transfère une structure contenant le code d'action (Extract) et le pointeur du NAF.

Les opérations suivantes doivent être effectuées, dans cet ordre:

- ouvrir le gestionnaire PCIDD$;
- préparer la structure suivante:
  - un seul mot: code de commande, 0x5845 (caractères 'EX', correspondant à "EXtraction");
  - un mot double: adresse du point d'entrée du NAF;
- émettre un ordre IOCTL d'écriture pour un appel système:
  - le registre CX contient la longueur de la structure de déclaration (6);
  - les registres DS:DX désignent la structure;
- vérifier le succès de l'opération (contrôle de l'indicateur de report);
- en cas d'erreur, émettre un appel à la fonction GetExtendedError pour obtenir un code d'erreur plus détaillé;
- fermer le gestionnaire en mode caractère.

Cette commande se terminera normalement, même si le dispositif NAF est déjà déclaré car, dans ce cas, rien ne se passe.

Cette commande donne une erreur dans les cas suivants, qui n'ont aucune suite:

- erreurs DOS typiques (pointeur non valide, numéro de fonction non valide, etc.);
- longueur incorrecte du tampon transféré (registre CX): code étendu d'erreur 24 (longueur incorrecte de la structure de requête);
- code de commande non valide (code étendu d'erreur 31, panne générale).

Appendice I

Echantillons codés d'application du système d'exploitation DOS

Ces échantillons représentent une façon d'implémenter, du point de vue du dispositif PUF, le mécanisme d'échange par appels de fonction.

```
/*****************************************************************************************
Ce code de bibliothèque peut être associé à un dispositif PUF pour permettre un accès uniforme à de multiples
dispositifs NAF. L'accès aux différents NAF avec un identificateur ExID unique, est réalisé au moyen d'une table locale,
d'une capacité de MAX_EXID identificateurs.
*****************************************************************************************/
/*
 * Inclure fichiers
 */
#include  <dos.h>
#include  <fcntl.h>
#include  <memory.h>
#include  <malloc.h>
#include  <stdio.h>

/*
 * Définitions de type générales
 */
typedef void                ( *   PFRV) ();       /* Pointeur sur fonction retournant un type vide */
typedef short int           ( far *  FPFRS) ();   /* Pointeur d'adresse distante sur fonction retournant
                                                     un type abrégé */
typedef void                ( far *  FPFRV) ();   /* Pointeur d'adresse distante sur fonction retournant
                                                     un type vide */
typedef int                 ( far *  FPFRI) ();   /* Pointeur d'adresse distante sur fonction retournant
                                                     un type interruption */

/*
 * Mappage des définitions de type générique
 */

typedef short int           PCI_INTEGER;
typedef char far  *         PCI_BYTEARRAY;
typedef short int           PCI_EXID;
typedef FPFRI               PCI_HANDLE;
typedef FPFRV               PCI_PROCEDURE;

/*
 * Définition des codes de fonctions
 */
```

```
#define  PCIGETPROPERTY          (short) (1)
#define  PCIREGISTER             (short) (2)
#define  PCIDEREGISTER           (short) (3)
#define  PCIPUTMESSAGE           (short) (4)
#define  PCIGETMESSAGE           (short) (5)
#define   PCISETSIGNAL           (short) (6)

/*
 * Définition des erreurs
 */
#define  E_DEVICE_DRIVER_NOT_FOUND      128
#define  E_DEVICE_DRIVER_CONTROL        128
#define  E_NAF_NOT_FOUND                130
#define  E_NAF_INVALID_ADDRESS          130
#define  E_TOO_MANY_ASSOCIATIONS        133
#define  E_INVALID_EXCHANGE_ID          136

/*
 * Autres définitions
 */

#define  SUCCESS                                0
#define  MAX_EXID                               32 /* permet 32 associations PUF_NAF */

/*
 * Structures
 */
struct pci_mpb {
        PCI_INTEGER MessageID;
        PCI_INTEGER MessageMaximumSize;
        PCI_INTEGER MessageActualUsedSize;
        PCI_INTEGER DataMaximumSize;
        PCI_INTEGER DataActualUsedSize;
};
struct pci_register {                           /* structure contenant les informations d'association */
        PCI_INTEGER PUFVersion;                 /* sur option: indiquer version du PUF */
        PCI_INTEGER PUFType;                    /* sur option: indiquer type du PUF */
        PCI_INTEGER MaxMsgSize;                 /* retour: longueur maximale d'un message */
};
struct pci_opsys {                              /* structure contenant les informations d'association */
        short int       MaxNCOCount;            /* sur option: indiquer nombre maximal d'objets NCO */
        short int       MaxPacketSize;          /* sur option: indiquer longueur maxi et nombre */
        short int       MaxPacketCount;         /* maxi des paquets en tampon */
        long int        AddBufferSize;          /* sur option: donner au NAF les longueurs et pointeurs */
        void far *      AddBufferSpace;         /* en tampon additionnel */
        long int        BufferNeeded;           /* retour: capacité de tampon addit. requise */
};
struct loc_exid_map {                           /* structure utilisée localement pour les ExID */
        PCI_HANDLE      pci_handle;
        PCI_EXID        exchange_id;
};

/*
 * Constantes des fonctions
 */
const char PCIsign[8]="ISDN-PCI";

/*
 * Variables locales
 */
static struct loc_exid_map _exid_map[MAX_EXID];   /* table contenant MAX_EXID identificateurs ExID */
static short int_exid_cnt = MAX_EXID;             /* nombre d'emplacements libres dans la table des ExID */
```

```
/*****************************************************************************
PciGetHandles:  cette fonction demande au gestionnaire "PCIDD$" une liste des pointeurs-PCI disponibles (points d'entrée de NAF)
        Elle retourne les pointeurs PCI disponibles dans le tampon
        PciHandles attribué. L'élément MaxHandles indique le nombre
        maximal de pointeurs PCI demandés. Cette fonction échoue si la
        valeur de l'élément MaxHandles est inférieure au nombre de
        pointeurs disponibles dans le gestionnaire.
***/
short int PciGetHandles (       short int MaxHandles,
                                FPFRI * PCIHandles,
                                short int * ActualHandles )
{
short int fildes;               /* descripteur de fichier */
union _REGS regs;
struct _SREGS segregs;

        /* ouvrir le gestionnaire */
if (_dos_open ("PCIDD$", _O_RDWR, &fildes) != SUCCESS)
        return E_DEVICE_DRIVER_NOT_FOUND;       /* gestionnaire inaccessible; retour erreur */

        /* préparer la commande IOCTL de lecture dans le gestionnaire */
_segread (&segregs);
segregs.ds = FP_SEG (PCIHandles);               /* préparer adresse segment */
regs.x.dx = FP_OFF (PCIHandles);                /* et décalage */
regs.x.cx = MaxHandles * sizeof(PCI_HANDLE);
regs.x.bx = fildes;                             /* fixer pointeur de fichier DOS */
regs.x.ax = 0x4402;                             /* Commande IOCTL de lecture dans gestionnaire caractères */

        /* émettre commande de lecture dans gestionnaire */
_intdosx (&regs, &regs, &segregs);

        /* fermer le gestionnaire */
_dos_close (fildes);

        /* vérifier présence d'erreur */
if (regs.x.cflag & 1)                           /* vérifier indicateur de report de processeur */
        return E_DEVICE_DRIVER_CONTROL;         /* une erreur s'est produite; retour sur erreur */

        /* Opération réussie. Calculer nombre de pointeurs-PCI reçus */
*ActualHandles = regs.x.ax / sizeof(PCI_HANDLE);

return SUCCESS;
}       /* Fin de la fonction PciGetHandles() */

/*****************************************************************************
PciGetProperty:  cette fonction demande au NAF ses propriétés, qui se présentent sous la forme d'une liste d'articles à codage TLV (type-longueur-valeur).
        Cette fonction retourne en sortie ces propriétés dans le tampon de propriétés concédé, dont la capacité maximale est indiquée par le paramètre MaximumSize. Cette fonction échoue si le paramètre MaximumSize a une valeur inférieure à celle de la propriété que le NAF peut signaler.
***/

short int PciGetProperty ( FPFRI PCIHandle,
        short int MaximumSize,
        char * Property,
        short int * ActualSize )
{
register short int error;
unsigned char far * signature;
```

```
/* Vérifier si le NAF est disponible */
if ( PCIHandle == NULL)
        return E_NAF_INVALID_ADDRESS;        /* NAF inaccessible, car adresse non valide */

/* calculer l'adresse de signature et la contrôler */
signature = (unsigned char far *) PCIHandle – sizeof(PCIsign);
if ( _fmemcmp (PCIsign, signature, sizeof(PCIsign) != SUCCESS))
        return E_NAF_NOT_FOUND;              /* NAF inaccessible car signature non valide */

/* Appeler le NAF pour obtenir les informations de propriété */
error = (*PCIHandle) (  PCIGETPROPERTY,
        MaximumSize,
        (char far *) Property,
        (short int far *) ActualSize );
return error;
}       /* Fin de la fonction PciGetProperty() */

/*****************************************************************************
PciRegister:    cette fonction tente d'associer un PUF appelant avec un NAF sélectionné.
        Elle fournit l'ExID, qui devra être utilisé lors des appels subséquents.
        Deux structures doivent être fournies par le PUF appelant:
        – la structure PCIRegisterInfo, et
        – la structure PCIOpSysInfo.
***/

short int PciRegister ( FPFRI PCIHandle,
        struct pci_register * PCIRegisterInfo,
        struct pci_opsys * PCIOpSysInfo,
        short int * ExID )
{
register short int error;
register short int exchange_id;
unsigned char far * signature;
struct loc_exid_map *exid_map;                  /* pointeur dynamique sur la table de mappage locale
                                                   des identificateurs ExID */

/* Vérifier si le NAF est disponible */
if ( PCIHandle == NULL)
        return E_NAF_INVALID_ADDRESS;        /* NAF inaccessible, car adresse non valide */

/* calculer l'adresse de signature et la contrôler */
signature = (unsigned char far *) PCIHandle – sizeof(PCIsign);
if ( _fmemcmp (PCIsign, signature, sizeof(PCIsign) != SUCCESS))
        return E_NAF_NOT_FOUND;              /* NAF inaccessible car signature non valide */

/* vérifier s'il y a encore de la place dans notre table de mappage locale d'identificateurs ExID */
if ( ! _exid_cnt)
        return E_TOO_MANY_ASSOCIATIONS;      /* Indiquer que tous les articles de la table ont été lus */

/* Appeler le NAF pour l'informer d'une nouvelle association avec un PUF */
error = (*PCIHandle) (  PCIREGISTER,
        (struct pci_register far *) PCIRegisterInfo,
        (struct pci_opsys far *) PCIOpSysInfo,
        (short int far * ) ExID );
if ( ! error)
        {
        /* L'association a été établie; l'enregistrer dans la table locale */
        exchange_id = 0;
        exid_map = &_exid_map[0];            /* configurer le pointeur sur la table locale des ExID */
        while (exid_map->pci_handle)
```

```
                {
                exid_map++;
                exchange_id += 1;
                }
        exid_map->exchange_id = *ExID;
        exid_map->pci_handle = PCIHandle;
        *ExID = exchange_id;                    /* calculer et préparer l'identificateur d'échange ExID */
        _exid_cnt -= 1;
        }

return error;
}       /* Fin de la fonction PciRegister() */

/*****************************************************************************
PciDeregister:      termine une association existante avec un NAF.
                    L'identificateur d'échange d'une association existante doit être fourni.
***/

short int PciDeregister (  PCI_EXID ExID )
{
register short int error;
struct loc_exid_map *exid_map;                  /* pointeur dynamique sur table locale de mappage des ExID */

/* Vérifier si l'ExID est valide et configurer le pointeur sur la table locale de mappage des ExID */
exid_map = &_exid_map[ExID];
if (ExID < 0 || ExID >= MAX_EXID || ! exid_map->pci_handle)
        return E_INVALID_EXCHANGE_ID;

/* Appeler le NAF pour l'informer de la fin de l'association */
error = (*exid_map->pci_handle) (   PCIDEREGISTER,
        exid_map->exchange_id );

/* supprimer l'association de la table locale */
exid_map->pci_handle = NULL;
_exid_cnt += 1;

return error;
}       /* Fin de la fonction PciDeregister() */

/*****************************************************************************
PciPutMessage:      transfère au NAF un message et les données associées.
***/

short int PciPutMessage (  short int ExID,
        struct pci_mpb * PCIMPB,
        char * Message,
        char * Data )
{
register short int error;
struct loc_exid_map *exid_map;                  /* pointeur dynamique sur table locale de mappage des ExID */

/* Vérifier si l'ExID est valide et configurer le pointeur sur  la table locale de mappage des ExID */
exid_map = &_exid_map[ExID];
if (ExID < 0 || ExID >= MAX_EXID || ! exid_map->pci_handle)
        return E_INVALID_EXCHANGE_ID;
```

```
/* Appeler le NAF et envoyer le message */
error = (*exid_map->pci_handle) (   PCIPUTMESSAGE,
            exid_map->exchange_id,
            (struct pci_mpb far *) PCIMPB,
            (char far *) Message,
            (char far *) Data );
return error;
}           /* Fin de la fonction PciPutMessage() */

/*****************************************************************************
PciGetMessage:     reçoit du NAF un message et les données associées.
***/
short int PciGetMessage (  short int ExID,
            struct pci_mpb * PCIMPB,
            char * Message,
            char * Data )
{
register short int error;
struct loc_exid_map *exid_map;                          /* pointeur dynamique sur table locale de mappage
                                                           des ExID */

/* Vérifier si l'ExID est valide et configurer le pointeur sur la table locale de mappage des ExID */
exid_map = &_exid_map[ExID];
if (ExID < 0 || ExID >= MAX_EXID || ! exid_map->pci_handle)
            return E_INVALID_EXCHANGE_ID;

/* Appeler le NAF et recevoir le message */
error = (*exid_map->pci_handle) (   PCIGETMESSAGE,
            exid_map->exchange_id,
            (struct pci_mpb far *) PCIMPB,
            (char far *) Message,
            (char far *) Data );

return error;
}           /* Fin de la fonction PciGetMessage() */

/*****************************************************************************
PciSetSignal:          communique au NAF l'adresse d'une procédure de signalisation.
            La procédure de signalisation recevra ensuite notification des événements de communication
            (c'est-à-dire un message pouvant être récupéré).
***/

short int PciSetSignal (   short int ExID,
            short int Signal,
            PFRV SignalProcedure )
{
register short int error;
struct loc_exid_map *exid_map;                      /* pointeur dynamique sur table locale de mappage des ExID */

/* Vérifier si l'ExID est valide et configurer le pointeur sur la table locale de mappage des ExID */
exid_map = &_exid_map[ExID];
if (ExID < 0 || ExID >= MAX_EXID || ! exid_map->pci_handle)
            return E_INVALID_EXCHANGE_ID;

/* Appeler le NAF pour préparer la fonction de signalisation */
error = (*exid_map->pci_handle) (   PCISETSIGNAL,
            exid_map->exchange_id,
            (FPFRV) SignalProcedure );

return error;
}           /* Fin de la fonction PciSetSignal() */
```

Remplacée par une version plus récente

TABLE DES MATIÈRES

PARTIE 8

PARTIE 8: MÉCANISME D'ÉCHANGE MS-WINDOWS

Résumé

La présente partie de la spécification définit tous les détails du système d'exploitation associé à un environnement MS-WINDOWS™. (On trouvera dans la Partie 2 une présentation générale du mécanisme d'association.)

Introduction

La diversité des interfaces de programmation utilisées par des équipements terminaux pour RNIS (réseau numérique à intégration de services) a fait obstacle au développement d'applications RNIS et est apparue comme une contrainte limitant l'usage d'équipements terminaux modernes pour le RNIS.

La présente spécification définit pour l'UIT-T l'interface entre programmes d'application (API, *application programming interface*), appelée interface de programmation de communication (PCI, *programming communication interface*) par réseau numérique à intégration de services (PCI-RNIS), qui permet aux applications d'accéder au RNIS et de le gérer.

L'interface PCI-RNIS a été définie de façon à offrir aux fournisseurs d'équipement terminal une norme qui permettra de réaliser la portabilité d'applications utilisant l'interface PCI-RNIS sur une gamme d'équipements terminaux tournant sur différents systèmes d'exploitation.

L'interface PCI-RNIS a été définie en fonction des besoins des développeurs d'applications et, dans la mesure du possible, élimine la nécessité d'une connaissance approfondie du RNIS. Elle a également été conçue de façon que les futures extensions du RNIS n'aient pas d'incidence sur le fonctionnement des applications existantes.

Remplacée par une version plus récente

1 Domaine d'application

La présente partie spécifie, pour le système d'exploitation MS-WINDOWS™, l'interface de programmation de communication pour réseau numérique à intégration de services (PCI-RNIS). Elle fait partie de la spécification PCI-RNIS.

Elle décrit la façon dont il y a lieu qu'un dispositif PUF ou NAF, tel que décrit dans la Partie 2: "Services de base", dialogue ou communique par messages et paramètres afin d'établir une connexion RNIS.

D'autres spécifications spécifieront la méthode d'essai et les prescriptions détaillées concernant chaque application, afin d'évaluer la conformité à la présente partie.

2 Références

[1] Partie 1, *Architecture générale.*

[2] Partie 2, *Services de base.*

3 Définitions

La présente partie définit les termes suivants:

3.1 **fonction d'échange**: propriété du dispositif PUF permettant de réaliser le mécanisme d'échange.

3.2 **mécanisme d'échange**: moyen permettant au dispositif PUF d'échanger des messages avec le dispositif NAF.

3.3 **interface RNIS de programmation de communication (PCI-RNIS)**: interface logicielle orientée réseau (RNIS) qui offre des possibilités de programmer la signalisation de réseau et l'échange de données d'utilisateur.

3.4 **message**: unité d'information transférée de part et d'autre de l'interface PCI-RNIS, entre le dispositif d'accès réseau (NAF) et le dispositif utilisateur d'interface PCI (PUF).

3.5 **dispositif d'accès réseau (NAF)**: unité fonctionnelle située entre l'interface PCI-RNIS et les couches associées au réseau.

3.6 **dispositif utilisateur d'interface PCI (PUF)**: unité fonctionnelle faisant appel à l'interface PCI-RNIS pour accéder à un dispositif NAF. Ce terme correspond pratiquement à l'application locale qui utilise l'interface.

4 Abréviations

La présente partie utilise les abréviations suivantes:

API	interface de programmation d'application (*application programming interface*)
NAF	dispositif d'accès réseau (*network access facility*)
PCI	interface de programmation de communication (*programming communication interface*)
PciMPB	bloc de paramètres pour messages d'interface PCI (*PCI message parameter block*)
PUF	dispositif utilisateur d'interface PCI (*PCI user facility*)
RNIS	réseau numérique à intégration de services
WINDOWS	appellation générique des systèmes d'exploitation compatibles avec la version 3.0 du système d'exploitation MS-WINDOWS™ (*Microsoft Corporation, Inc.*).

5 Guide de lecture

5.1 Guide du lecteur

La présente partie est destinée aux développeurs de logiciels, aux réalisateurs d'applications et aux constructeurs d'équipement, qui y trouveront la description des mécanismes d'échange et des exemples de codage conformes à [2] pour l'environnement WINDOWS™.

5.2 Mode d'emploi de la présente partie

Les lecteurs qui:

- ont besoin d'un aperçu général rapide sur le mécanisme d'échange en général le trouveront dans la Partie 2: "Services de base" [2]. D'autres systèmes d'exploitation sont décrits. Le lecteur est invité à consulter la Partie 1: "Architecture générale" [1], où il trouvera des informations sur la possibilité d'utiliser d'autres systèmes d'exploitation. Le 6.1 contient une description générale du mécanisme d'échange pour WINDOWS™;
- envisagent de réaliser une implémentation WINDOWS au moyen de la présente interface PCI-RNIS sont invités à consulter les paragraphes 5 et 6. Les paragraphes 3 et 4 donnent des renseignements utiles sur les termes et abréviations utilisés. Des exemples de codage sont présentés en Appendice I. Le lecteur trouvera dans la Partie 2: "Services de base" [2] des informations sur les paramètres ainsi que des listes et des descriptions des codes de retour en sortie de fonction;
- ont l'intention de construire une carte ou un équipement d'adaptation au RNIS utilisant l'interface PCI-RNIS sont également invités à lire les paragraphes 5 et 6. Les paragraphes 3 et 4 donnent d'utiles informations sur la définition des termes et abréviations utilisés. On trouvera au 6.4 de plus amples renseignements sur les dispositifs NAF. Les exemples de codage présentés dans l'Appendice I montrent comment un dispositif PUF peut accéder à un dispositif NAF. Le lecteur trouvera dans la Partie 2: "Services de base" [2] des informations sur les paramètres ainsi que des listes et des descriptions des codes de retour en sortie de fonction.

Le Tableau 1 donne une liste qui décrit le contenu général de la présente partie.

Tableau 1 – Table des matières de la présente partie

Paragraphe, appendice	Contenu
paragraphe 1	domaine d'application de la présente partie, décrivant son objet.
paragraphe 2	références
paragraphe 3	définitions de termes utilisés dans la présente partie
paragraphe 4	définitions d'abréviations utilisées dans la présente partie
paragraphe 5	des directives de lecture.
paragraphe 6	prescriptions relatives au système d'exploitation et les règles d'implémentation pour WINDOWS™
Appendice I	codage d'échantillons en langage C pour illustrer l'application spécifique du système d'exploitation WINDOWS™ au mécanisme d'échange de messages

6 Implémentation spécifique du système d'exploitation WINDOWS

6.1 Introduction

Sauf pour l'appel à la fonction PciGetHandles, le mécanisme des bibliothèques de liens dynamiques (DLL, *dynamic link library*) est le principal moyen utilisé pour la méthode d'échange PCI-RNIS sous Windows. Chaque dispositif NAF doit faire l'objet d'un lien DLL et doit exporter un point d'entrée pour chaque fonction PCI-RNIS utilisant le même nom (PciGetProperty, PciRegister, PciGetMessage, PciPutMessage, PciSetSignal, PciDeregister).

NOTE – Le nom de fonction exporté par le NAF doit être celui qui est décrit dans la Partie 2 [2] mais avec des paramètres différents.

La fonction PciGetHandles doit pouvoir accéder au fichier PCI.INI.

Les fonctions PciRegister et PciGetProperty contrôlent si la bibliothèque DLL, accessible par son nom, est disponible.

Pour accéder à un dispositif NAF, le dispositif PUF n'a besoin de connaître que le nom de la bibliothèque DLL. L'adresse d'accès à cette bibliothèque DLL peut être fournie de manière transparente au dispositif PUF, au moyen des fonctions du mécanisme d'échange PCI indiquées dans l'Appendice I.

La fonction PciRegister charge dynamiquement le dispositif NAF. Elle a besoin de conserver en mémoire le pointeur du NAF avec son lien DLL, de sorte que ce pointeur devra faire partie de l'identificateur d'échange (ExID). Le dispositif NAF a également besoin de garder trace du dispositif PUF, de sorte qu'il attribue un identificateur à ce PUF au moment de l'enregistrement de son association avec ce dispositif. Cet identificateur fourni par le dispositif NAF constitue l'autre partie de l'identificateur ExID.

Sous Windows, les conventions communes d'appel pour fournir un paramètre à une bibliothèque DLL sont les conventions d'appel en langage PASCAL. Ces conventions sont également utilisées par la méthode d'échange PCI-RNIS dans ce cas.

Les paramètres des pointeurs sont distants. La structure des blocs PCI-MPB est toujours transmise au moyen d'un pointeur. La structure des ExID est également toujours transmise au moyen d'un pointeur.

L'alignement des structures est en **octets**.

6.2 Implémentation du type de base

Sous Windows, les valeurs suivantes doivent être utilisées:

PCI_HANDLE	nom de la bibliothèque DLL
PCI_EXID	contenu de structure: pointeur fourni par Windows lors du chargement du lien DLL (hInstance) identificateur unique fourni par le NAF pour identifier le PUF
PCI_PROCEDURE	adresse de fonction exportée (FARPROC) fournie par le PUF
PCI_INTEGER	2 octets
PCI_BYTEARRAY	pointeur distant

6.3 Structures et prototypes de fonction en langage C

```
/* Types de base */
typedef SHORT                   PCI_INTEGER;
typedef LPSTR                   PCI_BYTEARRAY;
typedef LPSTR                   PCI_HANDLE;
typedef struct                  {
                                HINSTANCE       DLLInstance;
                                PCI_INTEGER     Exchange_Id;
                                } PCI_EXID;

typedef void (far pascal *PCI_PROCEDURE)(void);

/*
 * Structures
 */
struct pci_mpb {
                PCI_INTEGER     MessageID;
                PCI_INTEGER     MessageMaximumSize;
                PCI_INTEGER     MessageActualUsedSize;
                PCI_INTEGER     DataMaximumSize;
                PCI_INTEGER     DataActualUsedSize;
};

struct pci_register {                   /* structure contenant des informations d'enregistrement
                                           d'association */
        PCI_INTEGER PUFVersion;         /* sur option: indiquer version du PUF */
        PCI_INTEGER PUFType;            /* sur option: indiquer type du PUF */
        PCI_INTEGER MaxMsgSize;         /* retour: longueur maximale d'un message */
};

struct pci_opsys {                      /* structure contenant des informations propres au système d'exploitation */
        int             DummyParameter; /* Pas de prescription propre à WINDOWS */
};

/* Prototypes des fonctions d'échange */
```

```
PCI_INTEGER far PASCAL PciGetHandles (   PCI_INTEGER MaxHandles,
                                         PCI_BYTEARRAY PCIHandles,
                                         PCI_INTEGER far * ActualHandles);

PCI_INTEGER far PASCAL PciGetProperty (  PCI_HANDLE PCIHandle,
                                         PCI_INTEGER MaximumSize,
                                         PCI_BYTEARRAY Property,
                                         PCI_INTEGER far * ActualSize);

PCI_INTEGER far PASCAL PciRegister (     PCI_HANDLE PCIHandle,
                                         struct pci_register * PCIRegisterInfo,
                                         struct pci_opsys * PCIOpSysInfo,
                                         PCI_EXID far *ExID);

PCI_INTEGER far PASCAL PciDeregister (   PCI_EXID far *ExID);

PCI_INTEGER far PASCAL PciPutMessage (   PCI_EXID far *ExID,
                                         struct pci_mpb far *PCIMPB,
                                         PCI_BYTEARRAY Message,
                                         PCI_BYTEARRAY Data);

PCI_INTEGER far PASCAL PciGetMessage (   PCI_EXID far *ExID,
                                         struct pci_mpb far *PCIMPB,
                                         PCI_BYTEARRAY Message,
                                         PCI_BYTEARRAY Data);

PCI_INTEGER far PASCAL PciSetSignal (    PCI_EXID far *ExID,
                                         PCI_INTEGER Signal,
                                         PCI_PROCEDURE SignalProcedure);
```

6.4 Description des fonctions

Ce sous-paragraphe décrit l'implémentation, sous Windows, des fonctions de la méthode d'échange PCI-RNIS. Au cours d'un appel de PUF à NAF, la capacité de la mémoire de pile doit être d'au moins 1024 octets.

6.4.1 PciGetHandles

Sous Windows, la fonction PciGetHandles utilise un fichier PCI.INI, qui se trouve dans le répertoire Windows, afin d'obtenir les pointeurs-PCI disponibles.

Dans ce fichier PCI.INI, la section [Drivers] contient toutes les entrées des dispositifs NAF installés. Chaque entrée a le format suivant:

pciDriver<numéro>=DLLName (numéro=1..32)

Les opérations suivantes doivent obtenir tous les noms des gestionnaires de NAF installés:

– en itération de 1 à 32:

- construction du nom clé 'pciDriver' associé à la valeur actuelle de la boucle;
- envoi d'un message GetPrivateProfileString avec les éléments suivants:

  sectionKey = 'DRIVERS';

  le nom clé 'pciDriver' construit ci-dessus;

  aucune valeur par défaut;

  longueur maximale = 128;

  nom de fichier = 'PCI.INI'.

6.4.2 PciGetProperty

Cette fonction a pour objet de fournir au PUF la propriété du NAF. Implicitement, cette fonction vérifie si le dispositif NAF est disponible lors du chargement de la bibliothèque DLL au moyen de la fonction LoadLibrary.

Les opérations suivantes doivent être effectuées, dans cet ordre:

- charger la bibliothèque DLL;
- lire l'adresse de la fonction PciGetProperty exportée par le NAF;
- aller à cette adresse avec les paramètres fournis par le PUF;
- libérer la bibliothèque chargée.

6.4.3 PciRegister

Cette fonction a pour objet de fournir une association entre un PUF et un NAF. Le NAF est chargé et la partie DLLInstance de l'identificateur d'échange est fournie. La disponibilité du NAF choisi est vérifiée lors du chargement de la bibliothèque, qui est identifiée par son nom. Les paramètres de l'opération d'enregistrement d'association sont groupés selon la structure suivante:

- PUFType (PCI_INTEGER)
- PUFVersion (PCI_INTEGER)
- MaxMsgSize (PCI_INTEGER) indiquant que le NAF donnera la longueur maximale pour un message.

Les opérations suivantes doivent être effectuées, dans cet ordre:

- charger le lien DLL;
- fournir la partie DLLInstance de l'identificateur ExID indiquant l'instance du lien DLL;
- lire l'adresse de la fonction PciRegister exportée par le NAF;
- aller à cette adresse pour informer le NAF de l'existence d'un nouveau PUF. L'adresse de la structure paramétrique d'association et l'adresse de la structure ExID sont transmises au NAF sous forme de paramètres;
- à la sortie en retour du NAF, la partie Exchange_Id de l'identificateur ExID et le paramètre de longueur maximale de message de la structure paramétrique d'association ont été fournis par le NAF;
- retour au PUF avec le code de retour en sortie du NAF.

6.4.4 PciDeregister

Cette fonction a pour objet de désassocier un PUF et un NAF. Le numéro d'appel de bibliothèque DLL doit être décrémenté par Windows mais la bibliothèque DLL n'est pas retirée de la mémoire chaque fois qu'un PUF supprime son association avec un NAF.

Les opérations suivantes doivent être effectuées, dans cet ordre:

- lire l'adresse de la fonction PciDeregister exportée par le NAF;
- aller à cette adresse pour informer le NAF de la fin de l'association. L'identificateur PCI_EXID est transmis au NAF par adressage;
- libérer le lien DLL.

6.4.5 PciPutMessage

Cette fonction a pour objet de fournir un message, et les données associées éventuelles, entre un PUF et un NAF. Les paramètres sont présentés dans le même ordre que dans la description du PciGetMessage.

Les opérations suivantes doivent être effectuées, dans cet ordre:

- lire l'adresse de la fonction PciPutMessage exportée par le NAF;
- aller à cette adresse pour transmettre le paramètre au NAF (y compris l'adresse de l'identificateur PCI_EXID).

6.4.6 PciGetMessage

Cette fonction a pour objet de fournir un message, et les données associées éventuelles, entre un PUF et un NAF. Les paramètres sont présentés dans le même ordre que dans la description du PciGetMessage. Les tampons fournis par le PUF sont directement utilisés par le NAF.

Les opérations suivantes doivent être effectuées, dans cet ordre:

- lire l'adresse de la fonction PciGetMessage exportée par le NAF;
- aller à cette adresse pour transmettre le paramètre au NAF (y compris l'adresse de l'identificateur PCI_EXID).

6.4.7 PciSetSignal

Cette fonction permet à un dispositif PUF de fournir un mécanisme d'information direct qui sera utilisé par le NAF en cas d'événement entrant. Deux mécanismes s'excluant mutuellement sont offerts sous Windows:

- un mécanisme pour la procédure de signalisation;
- un mécanisme pour la messagerie d'utilisateur.

Une fois qu'un mécanisme a été choisi par le PUF, l'autre est désactivé par le NAF pour ce PUF particulier. Ces deux mécanismes doivent être supportés par un NAF.

Le premier mécanisme n'utilise pas le paramètre de signalisation. Ce paramètre doit être mis à 0.

Le second mécanisme utilise le paramètre de signalisation pour identifier la valeur associée avec le message WM_USER WINDOWS. Dans ce cas, le paramètre de signalisation ne doit pas être égal à 0.

6.4.7.1 Procédure pour le mécanisme de signalisation

L'adresse de la routine, fournie par le PUF dans le paramètre SignalProcedure, est utilisée directement par le NAF. Elle doit être rendue accessible au NAF avant d'être fournie par le PUF. La routine est appelée sans aucun paramètre.

Dans ce cas, le paramètre de signalisation n'est pas utilisé mais le paramètre doit être transmis au NAF avec la valeur 0.

La mémoire de pile utilisée au cours de l'appel à la procédure de signalisation n'est pas celle du PUF. La procédure de signalisation doit être compilée sans supposer que le registre SS est égal au registre DS, c'est-à-dire sous forme de lien DLL.

Le dispositif NAF est autorisé à appeler le PUF pour réémettre un appel de signalisation. Afin d'éviter la nécessité d'une pile importante, le dispositif NAF doit attendre le retour de la procédure de signalisation du PUF avant de réémettre le prochain appel de signalisation.

Le rappel automatique du PUF au NAF au cours de l'exécution de la procédure de signalisation n'est pas autorisé. La capacité de la mémoire de pile n'est pas garantie lorsque le dispositif NAF appelle le PUF. Par conséquent, la capacité de pile nécessaire pour le traitement du PUF doit être aussi petite que possible.

6.4.7.2 Procédure pour le mécanisme de messagerie d'utilisateur

Le paramètre Signal contient une valeur de PUF à ajouter à la constante de messagerie WM_USER WINDOWS. Ce message est envoyé à une fenêtre du PUF. Le pointeur-PCI pour cette fenêtre est fourni par le PUF dans le mot non significatif du paramètre SignalProcedure de la fonction PciSetSignal. Il doit s'agir d'un numéro de fenêtre (HWND) valide.

Lorsque le dispositif NAF envoie au PUF le message WM_USER + Signal, il utilise un appel de type PostMessage selon l'interface d'application (API) de WINDOWS. Le PUF trouvera comme troisième paramètre (appelé wParam) le type du message reçu. Dans le quatrième paramètre (lParam), le PUF trouvera, comme mot significatif, la longueur de message associée à ce message et, en mot moins significatif, la longueur des données associées. L'appel de fonction se présentera donc comme suit:

```
PostMessage(    LOWORD( SignalProcedure),
                WM_USER+Signal,
                MessageID,
                (DWORD) (MessageSize << 16) | (DataSize));
```

Comme c'est l'appel PostMessage de l'API Windows qui est utilisé, le PUF est autorisé à rappeler le dispositif NAF pendant le traitement du message.

Ce mécanisme est simple à implémenter mais une importante contrainte doit être soulignée:

- sous Windows, un appel de type PostMessage peut échouer en raison d'un manque d'espace disponible dans la file d'attente des messages. Le PUF est chargé de traiter assez rapidement les messages pour faire en sorte qu'aucun message de NAF ne soit perdu. Le PUF ne peut pas compter sur la réémission d'un message défectueux par le NAF.

6.4.7.3 Mécanisme de désactivation

Pour désactiver un quelconque mécanisme de signalisation, les paramètres Signal et SignalProcedure de la fonction PciSetSignal doivent avoir la valeur NULL. Une fois désactivé, le mécanisme précédent ne doit plus être utilisé par le NAF pour appeler le PUF.

6.5 Disponibilité de pointeurs-PCI désignant des dispositifs NAF

Pour être accessible au moyen de l'appel de la fonction PciGetHandles, un dispositif NAF doit lancer une action déclarative. L'action inverse – c'est-à-dire l'extraction à partir de la liste des dispositifs NAF disponibles – est également décrite. Ces actions sont propres au système d'exploitation.

6.5.1 Action déclarative

Tout d'abord, le dispositif NAF peut lire la liste des pointeurs-PCI disponibles pour vérifier s'ils ne sont pas déjà déclarés. Le mécanisme utilisé par le dispositif NAF est le même que pour obtenir, à partir d'un PUF, les dispositifs NAF disponibles: l'utilisation de pointeurs PciGetHandles (voir 6.4.1).

Si les pointeurs ne sont pas encore déclarés, le dispositif NAF inclut ses propres pointeurs PCI_HANDLE dans la liste.

```
PCI_BYTEARRAY    ownDLLName = "xxx";
PCI_BYTE         driverName[128];
WORD             index;
char             keyName[20];

/* Vérifier si le NAF n'est pas déjà installé */
for (index = 1; index <=  32; index++)
        {
        sprintf( keyName, "pciDriver%d", index);
        if (GetPrivateProfileString( "DRIVERS",        /* Nom de la section correspondante */
                                     keyName,          /* "pciDriver"+1..n */
                                     NULL,             /* Aucune valeur par défaut nécessaire */
                                     driverName,
                                     sizeof( driverName),
                                     "PCI.INI") > 0)
                {
                if (strcmpi(driverName, ownDLLName) == 0) return; /* NAF installé, OK retour */
                }
        }

/* Chercher une position libre dans le gestionnaire d'interface PCI */
for (index = 1, index <= 32; index++)
        {
        sprintf( keyName, "pciDriver%d", index);
        if (GetPrivateProfileString( "DRIVERS",        /* Nom de la section correspondante */
                                     keyName,          /* "pciDriver"+1..n */
                                     NULL,             /* Aucune valeur par défaut nécessaire */
                                     driverName,
                                     sizeof( driverName),
                                     "PCI.INI") == 0)
                {
                /* Entrée inexistante, ajouter nom du gestionnaire du NAF particulier */
                WritePrivateProfileString( "DRIVERS", keyName, ownDLLName, "PCI.INI");
                return;
                }
        }
```

Le nombre maximal de dispositifs NAF qui peuvent être associés est de 32.

6.5.2 Action extractive

Tout d'abord, le dispositif NAF parcourt la liste des pointeurs-PCI disponibles pour vérifier s'il est déclaré. Si c'est le cas, le dispositif NAF supprime son propre pointeur-PCI de la liste gestionnaire contenue dans le fichier "PCI.INI".

```
PCI_BYTEARRAY    ownDLLName = "xxx";
PCI_BYTE         driverName[128];
WORD             index;
char             keyName[20];
```

```
for (index = 1, index <= 32; index++)
        {
        sprintf( keyName, "pciDriver%d", index);
        if (GetPrivateProfileString( "DRIVERS",        /* Nom de la section correspondante */
                                     keyName,          /* "pciDriver"+1..n */
                                     NULL,             /* Aucune valeur par défaut nécessaire */
                                     driverName,
                                     sizeof( driverName),
                                     "PCI.INI") > 0)
                {
                /* Vérifier la présence de son propre nom */
                if (strcmpi(driverName, ownDLLName) == 0)
                        {
                        /* Supprimer ce nom du gestionnaire */
                        WritePrivateProfileString( "DRIVERS", keyName, "", "PCI.INI");
                        }
                }
        }
```

Appendice I

Echantillons codés d'implémentation du système d'exploitation WINDOWS

Ces échantillons présentent une manière d'implémenter, du point de vue du dispositif PUF, l'appel de fonction dans le mécanisme d'échange.

Le module codé ci-après montre un exemple d'implémentation des fonctions d'échange par PUF dans un environnement Windows. Cet échantillon est écrit en langage "C".

```
/* Cette norme inclut: */
#include <windows.h>

/* Types de base */
typedef short           PCI_INTEGER;
typedef LPSTR           PCI_BYTEARRAY;
typedef LPSTR           PCI_HANDLE;
typedef struct {
        HINSTANCE               hDLLInstance;
        PCI_INTEGER             Exchange_Id;
        } PCI_EXID;
typedef void (far pascal *PCI_PROCEDURE)();


/* Structures d'interface PCI */
struct pci_mpb {
        PCI_INTEGER   MessageID;
        PCI_INTEGER   MessageMaximumSize;
        PCI_INTEGER   MessageActualUsedSize;
        PCI_INTEGER   DataMaximumSize;
        PCI_INTEGER   DataActualUsedSize;
        };
typedef struct pci_mpb PCI_MPB;

struct pci_register {                           /* structure contenant des informations d'enregistrement
                                                   d'association */
        PCI_INTEGER PUFVersion;                 /* sur option: indiquer version du PUF */
        PCI_INTEGER PUFType;                    /* sur option: indiquer type du PUF */
        PCI_INTEGER MaxMsgSize;                 /* retour: longueur maximale d'un message */
};
```

```
struct pci_opsys {                                  /* structure contenant des informations d'enregistrement
                                                       d'association */
        int     DummyParameter;                     /* Pas de prescription propre à WINDOWS */
};

/*
 * L'interface PCI définit:
 */
#define PCI_HANDLE_LENGTH                       128     /* la longueur de chaque pointeur du tampon est
                                                           indiquée par la fonction PciGetHandles */
#define PCI_E_SUCCESS                           0
#define PCI_E_QUERY_ENTITY_NOT_AVAILABLE        128
#define PCI_E_INVALID_PCI_HANDLE                130
#define PCI_E_NAF_NOT_AVAILABLE                 255

/*
////////////////////////////////////////////////////////////////////////////////
/// PciGetHandles()
*/
PCI_INTEGER far PASCAL PciGetHandles (      PCI_INTEGER MaxHandles,
                                            PCI_HANDLE PCIHandles,
                                            PCI_INTEGER far * ActualHandles)
        {
        int                     nafNumber;
        int                     nafFound;
        int                     size;
        char                    keyName[20];
        PCI_BYTEARRAY           buffer;

        buffer = PCIHandles;
        for (nafNumber = 1, nafFound = 0; nafNumber <= MaxHandles; nafNumber++)
                {
                wsprintf( keyName, "pciDriver%d", nafNumber);
                size = GetPrivateProfileString(     "DRIVERS",      /* Nom de la section correspondante*/
                                                    keyName,        /* 'pciDriver'+1..n */
                                                    NULL,           /* Aucune chaîne par défaut n'est nécessaire */
                                                    buffer,         /* Adresse où ranger le résultat */
                                                    128,            /* Longueur maximale du résultat */
                                                    "PCI.INI");     /* Nom du fichier INI */
                if (size > 0)
                        {
                        nafFound++;       /* Un autre dispositif NAF a été trouvé */
                        buffer += 128;  /* Prochain emplacement pour un pointeur-PCI (longueur fixe de
                                           128 octets) */
                        }
                }
        *ActualHandles = nafFound;
        }


/*
////////////////////////////////////////////////////////////////////////////////
/// PciGetProperty()
*/
PCI_INTEGER far PASCAL PciGetProperty (     PCI_HANDLE PCIHandle,
                                            PCI_INTEGER MaximumSize,
                                            PCI_BYTEARRAY Property,
                                            PCI_INTEGER far * ActualSize)
        {
        PCI_INTEGER iReturnCode;
        HINSTANCE hDLLInstance;
        FARPROC lpfnGetProperty;
```

```
        /* charger le lien du NAF */
        hDLLInstance = LoadLibrary(PCIHandle);
        if (hDLLInstance < HINSTANCE_ERROR)
                return PCI_E_INVALID_PCI_HANDLE;      /* erreur dans LoadLibrary */

        /* lire le point d'entrée du lien DLL vers la fonction "PciGetProperty" */
        lpfnGetProperty = GetProcAddress(hDLLInstance, "PciGetProperty");
        if (lpfnGetProperty == NULL)
                {
                FreeLibrary(hDLLInstance);
                return PCI_E_NAF_NOT_AVAILABLE;      /* erreur dans GetProcAddress */
                }

        /* appeler le point d'entrée du lien DLL vers la fonction "PciGetProperty" */
        iReturnCode = lpfnGetProperty(PCIHandle, MaximumSize, Property, ActualSize);

        /* libérer le lien DLL de toute façon */
        FreeLibrary(hDLLInstance);

        /* sortir par le code de retour du lien DLL */
        return iReturnCode;
        }


/*
////////////////////////////////////////////////////////////////////////////////////
/// PciRegister()
///         Le paramètre PCIOpSysInfo n'est conservé que pour assurer la compatibilité
*/
PCI_INTEGER far PASCAL PciRegister (   PCI_HANDLE PCIHandle,
                                       struct pci_register * PCIRegisterInfo,
                                       struct pci_opsys * PCIOpSysInfo,
                                       PCI_EXID far *ExID)
        {
        PCI_INTEGER iReturnCode;
        FARPROC lpfnRegister;
        HINSTANCE hDLLInstance;

        /* charger le lien du NAF */
        hDLLInstance = LoadLibrary(PCIHandle);
        if (hDLLInstance < HINSTANCE_ERROR)
                return PCI_E_INVALID_PCI_HANDLE;      /* erreur dans LoadLibrary */

        /* mettre l'instance de lien DLL dans l'ExID */
        ExID->hDLLInstance = hDLLInstance;

        /* lire le point d'entrée du lien DLL vers la fonction "PciRegister" */
        lpfnRegister = GetProcAddress(hDLLInstance, "PciRegister");
        if (lpfnRegister == NULL)
                {   /* erreur dans GetProcAddress */
                FreeLibrary(hDLLInstance);
                return PCI_E_NAF_NOT_AVAILABLE;
                }

        /* appeler le point d'entrée du lien DLL vers la fonction "PciRegister" */
        iReturnCode = lpfnRegister(PCIRegisterInfo, ExID);

        if (iReturnCode != 0)
                {   /* erreur dans PciRegister : libérer le lien DLL */
                FreeLibrary(hDLLInstance);
                }
```

```
        /* sortir par le code de retour du lien DLL */
        return iReturnCode;
        }


/*
//////////////////////////////////////////////////////////////////////////////////////
/// PciDeRegister()
*/
PCI_INTEGER far PASCAL PciDeregister(PCI_EXID far *ExID)
        {
        PCI_INTEGER iReturnCode;
        FARPROC lpfnDeregister;

        /* lire le  point d'entrée du lien DLL vers la fonction "PciDeregister" */
        lpfnDeregister = GetProcAddress(ExID->hDLLInstance, "PciDeregister");
        if (lpfnDeregister == NULL)                     /* erreur dans GetProcAddress */
                return PCI_E_NAF_NOT_AVAILABLE;

        /* appeler le point d'entrée du lien DLL vers la fonction "PciDeRegister" */

        iReturnCode = lpfnDeregister(ExID);

        /* libérer le lien DLL de toute façon */
        FreeLibrary(ExID->hDLLInstance);

        /* sortir par le code de retour du lien DLL */
        return iReturnCode;
        }


/*
//////////////////////////////////////////////////////////////////////////////////////
/// PciPutMessage()
*/
PCI_INTEGER far PASCAL PciPutMessage(       PCI_EXID far *ExID,
                                            PCI_MPB far *PCIMPB,
                                            PCI_BYTEARRAY Message,
                                            PCI_BYTEARRAY Data)
        {
        FARPROC lpfnPutMessage;

        /* lire le point d'entrée du lien DLL vers la fonction "PciPutMessage" */
        lpfnPutMessage = GetProcAddress(ExID->hDLLInstance, "PciPutMessage");
        if (lpfnPutMessage == NULL)   /* erreur dans GetProcAddress */
                return PCI_E_NAF_NOT_AVAILABLE;

        /* appeler le point d'entrée du lien DLL vers la fonction "PciPutMessage" */
        /* et sortir par le code de retour du lien DLL */
        return lpfnPutMessage(ExID, PCIMPB, Message, Data);
        }


/*
//////////////////////////////////////////////////////////////////////////////////////
/// PciGetMessage()
*/
PCI_INTEGER far PASCAL PciGetMessage(       PCI_EXID far *ExID,
                                            PCI_MPB far *PCIMPB,
                                            PCI_BYTEARRAY Message,
                                            PCI_BYTEARRAY Data)
```

```
        {
        FARPROC lpfnGetMessage;
        /* lire le point d'entrée du lien DLL vers la fonction "PciGetMessage" */

        lpfnGetMessage = GetProcAddress(ExID->hDLLInstance, "PciGetMessage");
        if (lpfnGetMessage == NULL)   /* erreur dans GetProcAddress */
                return PCI_E_NAF_NOT_AVAILABLE;

        /* appeler le point d'entrée du lien DLL vers la fonction "PciGetMessage" */
        /* et sortir par le code de retour du lien DLL */
        return lpfnGetMessage(ExID, PCIMPB, Message, Data);
        }


/*
////////////////////////////////////////////////////////////////////////////////
/// PciSetSignal()
*/
PCI_INTEGER far PASCAL PciSetSignal(        PCI_EXID far *ExID,
                                            PCI_INTEGER Signal,
                                            PCI_PROCEDURE SignalProcedure)
        {
        FARPROC lpfnSetSignal;

        /* lire le point d'entrée du lien DLL vers la fonction "PciSetSignal" */
        lpfnSetSignal = GetProcAddress(ExID->hDLLInstance, "PciSetSignal");
        if (lpfnSetSignal == NULL)   /* erreur dans GetProcAddress */
                return PCI_E_NAF_NOT_AVAILABLE;

        /* appeler le point d'entrée du lien DLL vers la fonction "PciSetSignal" */
        /* et sortir par le code de retour du lien DLL */
        return lpfnSetSignal(ExID, Signal, SignalProcedure);
        }
```

Remplacée par une version plus récente

TABLE DES MATIÈRES

PARTIE 9

Page

PARTIE 9: MÉCANISME D'ÉCHANGE UNIX

Résumé

La présente partie de la spécification définit tous les détails du système d'exploitation associé à un environnement UNIX™. (On trouvera dans la Partie 2 une présentation générale du mécanisme d'association.)

Introduction

La diversité des interfaces de programmation utilisées par des équipements terminaux pour le RNIS (réseau numérique à intégration de services) a fait obstacle au développement d'applications RNIS et est apparue comme une contrainte limitant l'usage d'équipements terminaux modernes pour le RNIS.

La présente spécification définit pour l'UIT-T l'interface entre programmes d'application (API, *application programming interface*), appelée interface de programmation de communication (PCI, *programming communication interface*) par réseau numérique à intégration de services (PCI-RNIS), qui permet aux applications d'accéder au RNIS et de le gérer.

L'interface PCI-RNIS a été définie de façon à offrir aux fournisseurs d'équipement terminal une norme qui permettra de réaliser la portabilité d'applications utilisant l'interface PCI-RNIS sur une gamme d'équipements terminaux tournant sur différents systèmes d'exploitation.

L'interface PCI-RNIS a été définie en fonction des besoins des développeurs d'applications et, dans la mesure du possible, élimine la nécessité d'une connaissance approfondie du RNIS. Elle a également été conçue de façon que les futures extensions du RNIS n'aient pas d'incidence sur le fonctionnement des applications existantes.

Remplacée par une version plus récente

1 Domaine d'application

La présente partie spécifie, pour le système d'exploitation UNIX, l'interface de programmation de communication pour réseau numérique à intégration de services (PCI-RNIS). Elle fait partie de la spécification sur le PCI-RNIS.

Elle décrit la façon dont il y a lieu qu'un dispositif PUF ou NAF, tel que décrit dans la Partie 2: "Services de base", dialogue ou communique par messages et paramètres afin d'établir une connexion RNIS.

D'autres spécifications spécifieront la méthode d'essai et les prescriptions détaillées concernant chaque application, afin d'évaluer la conformité à la présente partie.

2 Références

[1] Partie 1: *Architecture générale.*

[2] Partie 2: *Services de base.*

3 Définitions

La présente partie définit les termes suivants:

3.1 **fonction d'échange**: propriété du dispositif PUF permettant de réaliser le mécanisme d'échange.

3.2 **mécanisme d'échange**: moyen permettant au dispositif PUF d'échanger des messages avec le dispositif NAF.

3.3 **interface RNIS de programmation de communication (PCI-RNIS)**: interface logicielle orientée réseau (RNIS) qui offre des possibilités de programmer la signalisation de réseau et l'échange de données d'utilisateur.

3.4 **message**: unité d'information transférée de part et d'autre de l'interface PCI-RNIS, entre le dispositif d'accès réseau (NAF) et le dispositif utilisateur d'interface PCI (PUF).

3.5 **dispositif d'accès réseau (NAF)**: unité fonctionnelle située entre l'interface PCI-RNIS et les couches associées au réseau.

3.6 **dispositif utilisateur d'interface PCI (PUF)**: unité fonctionnelle faisant appel à l'interface PCI-RNIS pour accéder à un dispositif NAF. Ce terme correspond pratiquement à l'application locale qui utilise l'interface.

4 Abréviations

La présente partie utilise les abréviations suivantes:

API	interface de programmation d'application (*application programming interface*)
NAF	dispositif d'accès réseau (*network access facility*)
PCI	interface de programmation de communication (*programming communication interface*)
PciMPB	bloc de paramètres pour messages d'interface PCI (*Pci message parameter block*)
PUF	dispositif utilisateur d'interface PCI (*Pci user facility*)
RNIS	réseau numérique à intégration de services
UNIX	appellation générique des systèmes d'exploitation compatibles avec le système d'exploitation UNIX

5 Guide de lecture

5.1 Guide du lecteur

La présente partie est destinée aux développeurs de logiciels, aux réalisateurs d'applications et aux constructeurs d'équipement, qui y trouveront la description des mécanismes d'échange et des exemples de codage conformes à [2] pour l'environnement UNIX. La compréhension du concept de flux de communication (STREAMS) aidera à entrer dans la description détaillée du mécanisme d'échange ci-après.

5.2 Mode d'emploi de la présente partie

Les lecteurs qui:

- ont besoin d'un aperçu général rapide sur le mécanisme d'échange en général le trouveront dans la Partie 2: "Services de base" [2]. D'autres systèmes d'exploitation sont décrits. Le lecteur est invité à consulter la Partie 1: "Architecture générale" [1], où il trouvera des informations sur la possibilité d'utiliser d'autres systèmes d'exploitation. Le 6.1 contient une description générale du mécanisme d'échange pour UNIX;

- envisagent d'implémenter une application UNIX au moyen de la présente interface PCI-RNIS sont invités à consulter les paragraphes 5 et 6. Les paragraphes 3 et 4 donnent des renseignements utiles sur les termes et abréviations utilisés. Des exemples de codage sont présentés en Appendice I. Le lecteur trouvera dans la Partie 2: "Services de base" [2] des informations sur les paramètres ainsi que des listes et des descriptions des codes de retour en sortie de fonction;

- ont l'intention de construire une carte ou un équipement d'adaptation au RNIS utilisant l'interface PCI-RNIS sont également invités à lire les paragraphes 5 et 6. Les paragraphes 3 et 4 donnent d'utiles informations sur la définition des termes et abréviations utilisés. On trouvera au 6.5 et au 6.6 de plus amples renseignements sur les dispositifs NAF. Les exemples de codage présentés dans l'Appendice I montrent comment un dispositif PUF peut accéder à un dispositif NAF. Le lecteur trouvera dans la Partie 2: "Services de base" [2] des informations sur les paramètres ainsi que des listes et des descriptions des codes de retour en sortie de fonction.

Le Tableau 1 donne une liste qui décrit le contenu général de la présente partie.

Tableau 1 – Table des matières de la présente partie

Paragraphe, appendice	Contenu
paragraphe 1	domaine d'application de la présente partie, décrivant son objet
paragraphe 2	références
paragraphe 3	définitions de termes utilisés dans la présente partie
paragraphe 4	définitions d'abréviations utilisées dans la présente partie
paragraphe 5	aperçu général et directives de lecture
paragraphe 6	prescriptions relatives au système d'exploitation et les règles d'implémentation pour UNIX
Appendice I	codage d'échantillons en langage C pour illustrer l'application spécifique du système d'exploitation UNIX au mécanisme d'échange de messages

6 Implémentation spécifique du système d'exploitation UNIX

6.1 Introduction

Les fonctions d'échange décrites dans la Partie 2: "Services de base" [2] doivent être mappées avec les fonctions appropriées du mécanisme central de flux de communication STREAMS du système UNIX.

L'interface compatible au niveau binaire avec un dispositif NAF fonctionnant dans l'environnement UNIX doit être implémentée au moyen du mécanisme central de flux STREAMS.

Les descriptions sont écrites en langage C parce que celui-ci est le moyen d'expression naturel dans l'environnement UNIX.

6.2 Implémentation des types de base

Le tableau suivant montre le mappage entre les types de base de la méthode d'échange et les types en langage C.

Type de base	Mappage et usage
PCI_INTEGER	ce type peut être implémenté sous la forme d'un entier signé de 2 ou de 4 octets, selon la définition donnée pour une constante système UNIX sous-jacente pour le type 'int'.
PCI_BYTEARRAY	type implémenté sous la forme d'un pointeur sur le type 'char'
PCI_EXID	type implémenté sous la forme d'un type 'int'. Comme la méthode d'échange utilise le noyau (service de base) STREAMS, l'identificateur ExID a la même valeur et le même type que le descripteur de fichier UNIX fourni par le mécanisme central STREAMS.
PCI_HANDLE	type implémenté sous la forme d'un pointeur sur le type 'char', c'est-à-dire d'une chaîne de caractères UNIX qui doit contenir le nom du dispositif STREAMS dans lequel le NAF est utilisé.
PCI_PROCEDURE	type implémenté sous la forme d'une adresse de fonction retournant le type "vide", comme défini par l'appel à la fonction système UNIX signal ().

6.3 Conventions de transfert de paramètres

Les conventions en langage C suivantes s'appliquent au transfert de paramètres:

- les valeurs d'appel de fonction sont transmises soit comme telles (par exemple PCI_INTEGER, PCI_EXID) ou au moyen de pointeurs (par exemple PCI_BYTEARRAY, PCI_HANDLE);
- les valeurs de retour sont transmises au moyen d'un pointeur permettant d'insérer la valeur (transmission par référence).

Les erreurs se produisant dans le gestionnaire du NAF sont retournées sous forme d'entiers positifs (de type PCI_INTEGER) dont les valeurs sont définies en [2]. Dans le cadre de la présente partie, une valeur 0 signifie "pas d'erreur" (succès).

Les erreurs se produisant dans les fonctions d'échange PCI seront normalement retournées sous forme d'entiers négatifs (de type PCI_INTEGER), dont les valeurs ne sont pas définies car elles dépendent de l'implémentation du dispositif NAF.

6.4 Définition des types, des constantes et des prototypes fonctionnels

Si un alignement sur le système cible UNIX est nécessaire, on utilisera la séquence du type **int**.

```
/*
 * Types de base
 */
typedef int             PCI_INTEGER;
typedef char *          PCI_BYTEARRAY;
typedef int             PCI_EXID;
typedef char *          PCI_HANDLE;
typedef void (*         PCI_PROCEDURE) ();

/*
 * Structures
 */
struct pci_mpb {
        PCI_INTEGER     MessageID;
        PCI_INTEGER     MessageMaximumSize;
        PCI_INTEGER     MessageActualUsedSize;
        PCI_INTEGER     DataMaximumSize;
        PCI_INTEGER     DataActualUsedSize;
};
```

```
/*
 * Prototypes  des fonctions d'échange
 */
PCI_INTEGER PciGetHandles (     PCI_ INTEGER        MaxHandles,
                                PCI_BYTEARRAY       PCIHandles,
                                PCI_INTEGER *       ActualHandles);

PCI_INTEGER PciGetProperty (    PCI_HANDLE          PCIHandle,
                                PCI_INTEGER         MaximumSize,
                                PCI_BYTEARRAY       Property,
                                PCI_INTEGER *       ActualSize);

PCI_INTEGER PciRegister (       PCI_HANDLE          PCIHandle,
                                PCI_INTEGER         PUFVersion,
                                PCI_INTEGER         PUFType,
                                PCI_EXID *          ExID,
                                PCI_INTEGER *       MaxMsgSize);

PCI_INTEGER PciDeregister (     PCI_EXID            ExID);

PCI_INTEGER PciPutMessage (     PCI_EXID            ExID,
                                struct pci_mpb *    PCIMPB,
                                PCI_BYTEARRAY       Message,
                                PCI_BYTEARRAY       Data);

PCI_INTEGER PciGetMessage (     PCI_EXID            ExID,
                                struct pci_mpb *    PCIMPB,
                                PCI_BYTEARRAY       Message,
                                PCI_BYTEARRAY       Data);

PCI_INTEGER PciSetSignal (      PCI_EXID            ExID,
                                PCI_INTEGER         Signal,
                                PCI_PROCEDURE       SignalProcedure);
```

6.5 Adaptation au mécanisme central STREAMS

6.5.1 Généralités

Un dispositif NAF implémenté dans le noyau central UNIX doit présenter son interface PCI-RNIS au moyen du mécanisme central de communication STREAMS. Pour chaque dispositif NAF implémenté, un seul accès de flux STREAMS doit être ménagé, quel que soit le nombre d'accès RNIS offert par ce dispositif NAF. Un tel accès de flux STREAMS peut, en principe s'il est implémenté par le NAF, servir à plusieurs dispositifs PUF. Par ailleurs, en raison de l'architecture UNIX, un seul dispositif PUF peut accéder à plusieurs flux STREAMS et donc à plusieurs dispositifs NAF simultanément. Les dispositifs NAF doivent être définis comme étant des flux de type CLONE.

Le mécanisme central des flux STREAMS du système UNIX offre deux files d'attente: une file d'écriture et une file de lecture. Les informations envoyées par les fonctions d'échange vers le gestionnaire de flux (informations centrifuges) sont placées dans la file d'écriture par un composant de flux STREAMS appelé "tête de flux". Le lancement de cette action par la tête de flux s'effectue par l'envoi de l'appel de la fonction système "STREAM putmsg()".

Le gestionnaire de flux peut accéder aux informations de la file d'écriture, les traiter et insérer les données résultantes dans la file de lecture, dont le contenu (informations centripètes) est mis à la disposition des fonctions d'échange par l'appel de la fonction système "STREAMS getmsg()".

6.5.2 Communication entre fonctions d'échange PUF et gestionnaire de flux NAF

La communication entre une fonction d'échange et le gestionnaire de flux NAF s'effectue par appel de cette fonction d'échange au moyen de la commande getmsg() ou putmsg() dans le cas des fonctions PciGetMessage() et PciPutMessage() ainsi qu'au moyen d'une commande centrale de type ioctl() dans le cas de toutes les autres fonctions d'échange.

Les informations acheminées au moyen de ce flux sont appelées "message STREAMS". Il convient de ne pas confondre les messages STREAMS avec les messages définis dans le cadre du PCI-RNIS.

Le mécanisme central STREAMS subdivise chaque message d'interface PCI en deux parties: une partie commande et une partie données. Pour les messages échangés au moyen des fonctions PciGetMessage() et PciPutMessage(), la partie commande d'un message STREAMS contient le message d'interface PCI tandis que la partie données contient les données de ce message PCI. Le gestionnaire de flux NAF est informé des longueurs des parties composant le message PCI au moyen des commandes UNIX normales getmsg() et putmsg().

Pour tous les autres messages, la commande individuelle est envoyée au gestionnaire de flux NAF dans le champ de commande "ioc_cmd" de la structure "struct iocblk". La partie données qui est associée à cette commande est transmise au gestionnaire de flux NAF dans les blocs de données contenus dans le message M_IOCTL.

Définitions des termes:

mp	est du type mblkt_t * (voir /usr/include/sys/stream.h)
struct iocblk	type défini dans /usr/include/sys/stream.h

Le gestionnaire de flux STREAMS du dispositif NAF peut obtenir les informations nécessaires à son fonctionnement au moyen des mécanismes suivants:

1) messages d'interface PCI échangés au moyen de la fonction PciPutMessage()

Information	**Disponibilité**
Longueur de la partie commande	mp->b_wptr - mp->b_rptr
Contenu de la partie commande	mp->b_rptr
Présence d'une partie données	mp->b_cont != NULL
Longueur de la partie données	msdgsize(mp)
Contenu de la partie données	mp->b_cont->b_rptr

2) messages d'interface PCI échangés au moyen du mécanisme ioctl()

Information	**Disponibilité**
Fonction requise	((struct iocblk *)mp->b_rptr)->ioc_cmd
Longueur de la partie commande	((struct iocblk *)mp->b_rptr)->ioc_count
Contenu de la partie commande	mp->b_cont->b_rptr
Espace pour les données retournées	mp->b_cont->b_rptr ((struct iocblk *)mp->b_rptr)->ioc_rval

La fonction requise est définie comme suit:

#define PCI_PROPERTY	(('Z' << 8) | 1)
#define PCI_REGISTER	(('Z' << 8) | 2)
#define PCI_DEREGISTER	(('Z' << 8) | 3)
#define PCI_SETSIGNAL	(('Z' << 8) | 4)

6.5.3 Considérations particulières

Plusieurs aspects d'implémentation d'un dispositif NAF doivent être pris en considération par le dispositif PUF qui implémente les fonctions d'échange:

- le PUF donne au NAF l'autorisation d'insérer les messages PCI entrants dans la file de lecture, qui sert donc de tampon. La commande de débit est réalisée par le mécanisme normalisé UNIX de marques de crue et d'étiage, qui permet au gestionnaire de flux NAF de gérer la commande de débit en transparence à son propre niveau;
- la longueur d'un élément de file d'informations de communication est limitée. Un gestionnaire de flux NAF doit être en mesure, sur demande du dispositif PUF, de fournir des parties données d'une longueur de 4096 octets dans un message de flux; mais il doit également garantir que cette valeur maximale d'acheminement ne sera pas dépassée. Des blocs de données d'une longueur supérieure à 4096 octets peuvent toutefois être gérés si le flux est mis en mode "non-rejet de message" [voir l'élément streamio(7)]. Si un message contenant un bloc de données d'une longueur supérieure à 4096 octets arrive en tête de flux, un appel à la fonction PciGetMessage retournera les 4096 premiers octets de ce bloc de données et les appels suivants à la fonction PciGetMessage retourneront les blocs suivants. Chacun des appels suivants à la fonction PciGetMessage retournera un message dont la partie commande aura une longueur nulle;
- seul le signal SIGPOLL du système UNIX doit être émis par le dispositif NAF implémenté.

6.6 Description des fonctions

Le présent sous-paragraphe décrit l'implémentation, au moyen du mécanisme UNIX de flux STREAMS, des fonctions de la méthode d'échange PCI-RNIS. La description de chaque fonction sera divisée en trois parties comme suit:

1) corps de fonction: description du corps de la fonction, y compris la description générale du comportement de la fonction

2) STREAMS putmsg(): création d'une structure pour les appels à la fonction putmsg()

3) STREAMS getmsg(): contenu de la structure après retour de la fonction getmsg()

Les prototypes des fonctions putmsg () et getmsg () sont les suivants:

```
int putmsg (fd, ctlptr, dataptr, flags)
        int fd;                     /* Descripteur de fichier          */
        struct strbuf *ctlptr;      /* Partie commande du message      */
        struct strbuf *dataptr;     /* Partie données du message       */
        int flags;                  /* Priorité du message.            */
int getmsg (fd, ctlptr, dataptr, flags)
        int fd;                     /* Descripteur de fichier          */
        struct strbuf *ctlptr;      /* Partie commande du message      */
        struct strbuf *dataptr;     /* Partie données du message       */
        int *flags;                 /* Priorité du message.            */
with
struct strbuf    {
        int maxlen                  /* Longueur maximale du tampon     */
        int len                     /* Longueur des données            */
        char *buf                   /* Pointeur sur tampon             */
        }
```

En variante, pour les fonctions d'échange de l'interface PCI-RNIS qui font appel au mécanisme ioctl(), la description de chaque fonction est subdivisée en deux parties comme suit:

1) corps de fonction: description du corps de fonction, y compris description générale du comportement de cette fonction

2) ioctl(): création d'une structure pour l'appel à la commande ioctl()

Le prototype de la commande ioctl () est:

```
int ioctl (fd, command, arg)
        int fd;                 /* Descripteur de fichier                                  */
        int command;            /* commande ioctl  telle que définie dans la structure streamio(7)  */
        char *arg;              /* argument propre à la commande                           */
```

Chaque fois que la commande est I_STR , il y a lieu que l'argument désigne une structure de type strioctl, où strioctl est défini comme suit:

```
struct strioctl   {
        int  ic_cmd;            /* Commande définie par l'utilisateur      */
        int ic_timeout;         /* Temporisation de la commande            */
        int ic_len;             /* Longueur de la partie données à suivre  */
        char  *ic_dp;           /* Arguments propres à la commande         */
        }
```

6.6.1 PciGetHandles

Corps de fonction:

```
PCI_INTEGER PciGetHandles (     PCI_INTEGER MaxHandles,
                                PCI_BYTEARRAY PCIHandles,
                                PCI_INTEGER *ActualHandles)
{
...
}
```

Le paramètre MaxHandles indique le nombre maximal de pointeurs-PCI que le paramètre PCIHandles peut recevoir. Au retour, le paramètre ActualHandles, qui est un pointeur sur une valeur d'entier, contient le nombre de pointeurs-PCI copié dans le paramètre PCIHandles.

Cette fonction doit:

- examiner le répertoire **/etc/pcidd** afin de lire le nombre et le nom des pointeurs-PCI disponibles;
- mettre à jour les paramètres PCIHandles et ActualHandles;
- retourner en sortie le code d'erreur approprié.

6.6.2 PciGetProperty

Corps de fonction:

```
PCI_INTEGER PciGetProperty      (   PCI_HANDLE          PCIHandle,
                                    PCI_INTEGER         MaximumSize,
                                    PCI_BYTEARRAY       NAFProperty,
                                    PCI_INTEGER         *ActualSize)
{
        struct strioctl     strioctl;
        extern int          errno;
        int                 filedes;
}
```

Le paramètre PCIHandle pointe sur le nom du chemin du gestionnaire STREAMS; le paramètre MaximumSize indique la capacité du tampon contenant les propriétés. Le paramètre NAFProperty pointe sur ce tampon et le paramètre ActualSize pointe sur une valeur d'entier indiquant en retour la longueur réelle des informations de propriété contenues dans le dispositif NAF.

Cette fonction doit:

- ouvrir le gestionnaire STREAMS au moyen du paramètre PCIHandle;
- émettre l'appel à la commande ioctl ();
- récupérer la valeur du paramètre ActualSize et le code d'erreur;
- fermer le gestionnaire STREAMS;
- retourner le code d'erreur approprié.

Commande STREAMS ioctl():

Le composant ic_cmd doit être mis à la valeur de la fonction PCI_PROPERTY, le composant ic_len doit être mis à la valeur du paramètre MaximumSize et le composant ic_dp doit être mis à la valeur de pointage sur le tampon indiqué par le paramètre NAFProperty.

Au retour de l'appel à la commande ioctl(), la valeur retournée doit être comparée à 0, qui indique le succès. Toute autre valeur de retour indique une situation d'erreur qui doit être indiquée dans la variable errno. Le composant ic_len de la structure strioctl contient le nombre d'octets retournés à la fin de la commande ioctl appelée. Le composant ic_dp pointe sur la propriété retournée.

NOTE – La longueur indiquée en retour est toujours celle de la propriété implantée dans le dispositif NAF.

```
strioctl.ic_cmd         = PCI_PROPERTY;
strioctl.ic_timeout     = 0;
strioctl.ic_len         = MaximumSize;
strioctl.ic_dp          = (char *) NAFProperty;

if (ioctl (filedes, I_STR, &strioctl) == 0) {
        *ActualSize = strioctl.ic_len;
        return 0;
}
else {
        *ActualSize = 0;
        return errno;
}
```

6.6.3 PciRegister

Corps de fonction:

```
PCI_INTEGER PciRegister   (PCI_HANDLE      PCIHandle,
                           PCI_INTEGER     PUFVersion,
                           PCI_INTEGER     PUFType,
                           PCI_EXID        *ExID,
                           PCI_INTEGER     *MaxMsgSize)
{
        struct strioctl          strioctl;
        struct pci_register_t    pci_register;
        extern int               errno;
}
```

Le paramètre PCIHandle pointe sur le nom du chemin du gestionnaire de flux STREAMS. Les paramètres PUFVersion et PUFType sont réglés comme indiqué en [2]. L'identificateur ExID pointe sur un entier recevant la valeur de l'identificateur d'échange retourné, qui doit être égale à celle du descripteur de fichier UNIX retourné par l'appel système open(). Le paramètre MaxMsgSize est un entier recevant la longueur du message du dispositif NAF, comme décrit en [2].

Cette fonction doit:

- ouvrir le gestionnaire STREAMS au moyen du paramètre PCIHandle;
- émettre l'appel à la commande ioctl();
- récupérer les valeurs de retour de la structure pci_control;
- laisser le gestionnaire STREAMS ouvert et affecter à l'ExID la valeur du descripteur de fichier indiquée par la commande open();
- retourner le code d'erreur approprié.

Commande STREAMS ioctl():

Le composant ic_cmd doit être mis à la valeur de la fonction PCI_REGISTER, le composant ic_len doit être mis à la longueur de la structure pci_register et le composant ic_dp doit être mis à la valeur de pointage sur la structure pci_register qui est créée avec les valeurs des paramètres PUFVersion et PUFType. Au retour de l'appel à la commande ioctl(), la valeur retournée doit être comparée à –1, qui indique une situation d'erreur spécifique, qui doit être indiquée dans la variable externe errno. Toute autre valeur de retour indique le succès et la valeur de retour de l'appel à la commande ioctl() doit indiquer la longueur maximale de message PCI que le dispositif NAF peut reconnaître.

```
struct pci_register_t {
        int             puf_version;
        int             puf_type;
} pci_register;

pci_register.puf_version    = PUFVersion;
pci_register.puf_type       = PUFType;

strioctl.ic_cmd             = PCI_REGISTER;
strioctl.ic_timeout         = 0;
strioctl.ic_len             = sizeof (pci_register);
strioctl.ic_dp              = (char *) &pci_register;

if ((*ExID = open (PCI_HANDLE, O_RDWR)) == –1) {
        *ExID = 0;
        return <cant_open_device : errno provides more information>;
}

if ((*MaxMsgSize = ioctl (*ExID, I_STR, &strioctl)) < 0) {
        *MaxMsgSize = 0;
        return errno;
}
else {
        return 0;
}
```

6.6.4 PciDeregister

Corps de fonction:

```
PCI_INTEGER PciDeregister       (PCI_EXID     *ExID)
{
        struct strioctl         strioctl;
        extern int              errno;
}
```

L'identificateur ExID désigne le gestionnaire STREAMS ouvert. Il est identique au descripteur de fichier retourné par l'appel système open(). Cette fonction doit:

- émettre l'appel à la commande ioctl();
- récupérer le code de retour sur erreur;
- fermer le gestionnaire STREAMS;
- retourner le code d'erreur approprié.

Commande STREAMS ioctl():

Le composant ic_cmd doit être mis à la valeur de la fonction PCI_DEREGISTER, le composant ic_len doit être mis à zéro et le composant ic_dp doit être mis à la valeur NULL. Au retour de l'appel à la commande ioctl(), la valeur retournée doit être comparée à –1, qui indique une situation d'erreur spécifique, qui doit être indiquée dans la variable externe errno. Toute autre valeur de retour indique le succès.

```
strioctl.ic_cmd         = PCI_DEREGISTER;
strioctl.ic_timeout     = 0;
strioctl.ic_len         = 0;
strioctl.ic_dp          = (char *) NULL;

if (ioctl (*ExID, I_STR, &strioctl) == –1) {
        return errno;
}
else {
        close (*ExID);
        return 0;
}
```

6.6.5 PciPutMessage

Corps de fonction:

```
PCI_INTEGER PciPutMessage(      PCI_EXID                ExID,
                                struct  pci_mpb         *PCIMPB,
                                PCI_BYTEARRAY           Message,
                                PCI_BYTEARRAY           Data)
{
        struct strbuf ctlbuf;   /* pointeur sur la partie commande du message de flux */
        struct strbuf databuf;  /* pointeur sur la partie données du message de flux */
}
```

L'identificateur ExID désigne le gestionnaire de flux STREAMS. La structure PCI-MPB est un pointeur sur le bloc de paramètres du message d'interface PCI. Les structures "Message" et "Data" sont les parties du message PCI qui doivent être envoyées au gestionnaire de flux NAF. Ces deux structures peuvent être facultatives, auquel cas elles sont spécifiées sous la forme NULL. Pour plus d'efficacité (voir la fonction STREAMS putmsg ci-dessous), il est recommandé que le pointeur PCI-MPB soit écrit immédiatement avant la partie "Message", ce qui permet d'éviter de faire une copie en mémoire.

Cette fonction doit:

- préparer les structures de mémoire tampon "ctlbuf" et "databuf";
- émettre l'appel à la commande putmsg();
- récupérer le code de retour sur erreur;
- retourner le code d'erreur approprié.

Commande STREAMS putmsg():

```
/* L'idée générale est de transférer dans la structure ctlbuf un tampon contenant le pointeur PCIMPB suivi du contenu de
la partie "Message" et de transférer dans la structure databuf le contenu de la partie "Data" */
if (Message && ((char *)Message != (char *)PCIMPB + sizeof(pci_mpb))) {
        /*      Il y a une partie "Message" de valeur non égale à NULL et les structures PCIMPB et "Message" ne sont
                pas contiguës en mémoire,
                Il faut construire un tampon où le pointeur PCIMPB est suivi du contenu de la partie "Message" */
        char  *buffer; /* pointeur sur un tampon de capacité suffisante pour recevoir le pointeur PCIMPB et le contenu
                de la partie "Message" */
        ...
        /* Ici, un processus d'affectation de zones mémoire peut avoir lieu */
        ...
        memcpy (buffer, PCIMPB, sizeof(pci_mpb));
        memcpy ((buffer + sizeof(pci_mpb), Message, PCIMPB->MessageActualUsedSize);
        ctlbuf->buf             = buffer;
        ctlbuf->len             = PCIMPB->MessageActualUsedSize + sizeof(pci_mpb);
}
else {
        /* ou bien il n'y a pas de partie "Message" ou bien le pointeur PCIMPB et la partie "Message" sont contigus en
        mémoire */
        ctlbuf->buf             = PCIMPB;
        ctlbuf->len             = Message ? PCIMPB->MessageActualUsedSize + sizeof(pci_mpb) : sizeof(pci_mpb);
}

databuf->buf    = Data;
databuf->len    = Data ? PCIMPB->DataActualUsedSize : 0;

if (putmsg (ExID, &ctlbuf, &databuf, flags) != 0) {
        /* Situation d'erreur, la variable "errno" sera définie       */
        ....
}
else {
        /* Réussite de l'opération     */
        ....
}
```

6.6.6 PciGetMessage

Corps de fonction:

```
PCI_INTEGER PciGetMessage(      PCI_EXID                ExID,
                                struct pci_mpb          *PCIMPB,
                                PCI_BYTEARRAY           Message,
                                PCI_BYTEARRAY           Data)
{
        struct strbuf           ctlbuf;         /* pointeur sur la partie commande du message de flux */
        struct strbuf           databuf;        /* pointeur sur la partie données du message de flux */
}
```

L'identificateur ExID désigne le gestionnaire de flux STREAMS. La structure PCI-MPB est un pointeur sur le bloc de paramètres du message d'interface PCI. Les structures "Message" et "Data" sont les parties du message PCI qui doivent être envoyées au gestionnaire de flux NAF. Ces deux structures peuvent être facultatives, auquel cas elles sont spécifiées sous la forme NULL. Pour plus d'efficacité (voir la fonction STREAMS getmsg ci-dessous), il est recommandé que le pointeur PCI-MPB soit écrit immédiatement avant la partie "Message", ce qui permet d'éviter de faire une copie en mémoire.

Cette fonction doit:

- préparer les structures de mémoire tampon **ctlbuf** et **databuf**;
- émettre l'appel à la commande getmsg();
- récupérer les valeurs de retour à partir des structures **ctlbuf** et **databuf**;
- retourner le code d'erreur approprié.

Commande STREAMS getmsg():

```
/* L'idée générale est de transférer dans la structure ctlbuf un tampon de capacité suffisante pour recevoir le pointeur
PCIMPB suivi du contenu de la partie "Message" et de transférer dans la structure databuf le contenu de la partie "Data".
Le code d'erreur du NAF est disponible dans la variable errno. */

if (Message && ((char *)Message != (char *)PCIMPB + sizeof(pci_mpb))) {
        /* Il y a une partie "Message" de valeur non égale à NULL et le pointeur PCIMPB et la partie "Message" ne
        sont pas contigus en mémoire; il faut réserver un tampon où le pointeur PCIMPB peut être suivi du contenu de
        la partie "Message" */
        char  *buffer; /* pointeur sur un tampon de capacité suffisante pour recevoir le pointeur PCIMPB et le contenu
        de la partie "Message" */
        /* Ici, un processus d'affectation de zones mémoire peut avoir lieu */
        ctlbuf->buf             = buffer;
}
else {
        /* ou bien il n'y a pas de partie "Message", ou bien le pointeur PCIMPB et la partie "Message" sont contigus en
        mémoire */
        ctlbuf->buf             = PCIMPB;
}
ctlbuf->maxlen          = Message ? PCIMPB->MessageMaximumSize + sizeof(pci_mpb):sizeof(pci_mpb);
databuf->buf            = Data;
databuf->maxlen         = Data ? PCIMPB->DataMaximumSize : 0;
if (getmsg (ExID, &ctlbuf, &databuf, flags) != 0) {
        /* Situation d'erreur, la variable "errno" sera définie */
        PCIMPB->c_error = errno;
        ....
}
else {      /* Réussite de l'opération     */
        if (ctlbuf->len != –1 && ctlbuf->len >= sizeof(pci_mpb)) {
                /* Un message, éventuellement de longueur 0, est présent */
                PCIMPB->MessageActualUsedSize       = ctlbuf->len - sizeof(pci_mpb);
                if (Message && ((char *)Message != (char *)PCIMPB + sizeof(pci_mpb)))
                        {
                        /* Il y a une partie "Message" de valeur non égale à NULL et le pointeur PCIMPB et la partie
                        "Message" ne sont pas contigus en mémoire; un tampon, où le pointeur PCIMPB peut être suivi
                        du contenu de la partie "Message", a été utilisé */
                        memcpy (PCIMPB, buffer, sizeof(pci_mpb));
                        memcpy (Message,(buffer + sizeof(pci_mpb)), (ctlbuf->len - sizeof(pci_mpb)));
                }
                else {
                        /* le pointeur PCIMPB et la partie "Message" sont contigus en mémoire, sans utilisation de
                        tampon additionnel */
                        Message = PCIMPB + sizeof(pci_mpb);
                }
        }
        else {
        /* Absence de partie "Message" ou message trop petit: erreur (au moins le pointeur PCIMPB devrait être
        présent) */
        ..............
        }
if (databuf->len != –1) {
        /* Un bloc de données, éventuellement de longueur = 0, est présent */
        PCIMPB->DataActualUsedSize = databuf->len;
}
else {
        /* Absence de données */
        PCIMPB->DataActualUsedSize = 0;
}
}
```

6.6.7 PciSetSignal

Corps de fonction:

```
PCI_INTEGER  PciSetSignal(      PCI_EXID            *ExID,
                                PCI_INTEGER         Signal,
                                PCI_PROCEDURE       SignalProcedure)
{
        extern int      errno;
}
```

L'identificateur ExID désigne le gestionnaire de flux STREAMS et le paramètre Signal désigne le numéro de signal UNIX. Le paramètre SignalProcedure contient l'adresse, dans le dispositif PUF, du gestionnaire de signaux (fonction 'C'). Seul le signal SIGPOLL du système UNIX doit être émis par le dispositif NAF implémenté. Par conséquent, toute valeur non nulle du paramètre Signal doit déclencher l'émission de signaux SIGPOLL par le système UNIX, tandis qu'une valeur nulle arrêtera cette émission.

Le paramètre SignalProcedure défini par le dispositif PUF doit réémettre le signal par l'intermédiaire de l'appel système à la commande signal() – voir ci-dessous. Ce mécanisme est obligatoire, sinon le signal suivant issu du NAF déconnecterait le dispositif PUF.

Plusieurs signaux peuvent être envoyés par un NAF à un PUF avant que celui-ci n'ait accès au NAF. Un accès au NAF par un PUF en cours de traitement de la procédure de signalisation n'est pas recommandé.

Cette fonction doit:

- émettre l'appel à la commande ioctl();
- récupérer le code d'erreur;
- associer la signalisation UNIX par SIGPOLL avec la tête de flux au moyen de l'appel système: ioctl (..., I_SETSIG, S_MSG);
- associer la signalisation UNIX par SIGPOLL avec le système d'exploitation au moyen de l'appel système: signal (SIGPOLL, SignalProcedure);
- retourner le code d'erreur approprié.

Commande STREAMS ioctl():

La fonction doit contrôler le paramètre Signal et doit, si sa valeur est nulle, mettre à zéro la variable Signal_options afin d'interrompre la signalisation. Par ailleurs, la fonction de signalisation doit être dissociée par l'appel à la commande signal() appropriée.

Si le paramètre Signal a une valeur non nulle, la variable Signal_options doit être réglée de façon à permettre les signaux SIGPOLL ainsi que toutes autres options nécessitées par l'implémentation [voir la commande sigaction()]. Par ailleurs, la fonction de signalisation doit être associée au moyen de l'appel système à la commande signal().

```
if (Signal == 0) {
        Signal_options = 0;
        if (ioctl (ExID, I_SETSIG, &Signal_options) == –1)
                return errno;
        signal (SIGPOLL, SIG_DFL);
        return 0;
}
else {
        Signal_options = <SETSIG options>
        if (ioctl (ExID, I_SETSIG, &Signal_options) == –1)
                return errno;
        signal (SIGPOLL, SignalProcedure);
        return 0;
}
```

6.7 Disponibilité de pointeurs PCI désignant des dispositifs NAF

Pour être accessible au moyen de l'appel de la fonction PciGetHandles, un dispositif NAF doit lancer une action déclarative. L'action inverse – c'est-à-dire l'extraction à partir de la liste des dispositifs NAF disponibles – est également décrite. Ces actions sont propres au système d'exploitation.

6.7.1 Action déclarative

Au cours de la routine d'installation du gestionnaire de flux STREAMS, le répertoire **/etc/pcidd** est mis à jour par un fichier fictif qui est le nom du nouveau dispositif NAF. La routine d'installation peut contrôler la disponibilité du dispositif NAF avant de créer le nouveau fichier fictif.

6.7.2 Action extractive

Au cours de la routine de désinstallation du gestionnaire de flux STREAMS, le répertoire **/etc/pcidd** est mis à jour par suppression du fichier fictif contenant le nom du dispositif NAF.

Appendice I

Echantillons codés d'application du système d'exploitation UNIX

Ces échantillons présentent une manière d'implémenter, du point de vue du dispositif PUF, l'appel de fonction dans le mécanisme d'échange.

L'appel à la fonction **PciGetHandles** n'est pas présenté.

```
/* Inclure fichiers et définitions de base */
#include  <stddef.h>
#include  <fcntl.h>
#include  <signal.h>
#include  <stropts.h>
#include  <errno.h>
#include  <stdlib.h>

#define ERROR        (–1)    /* Valeur d'erreur */
#define Success      (0)     /* Valeur de succès */

/* Types de base */
typedef int                  PCI_INTEGER;
typedef char *               PCI_BYTEARRAY;
typedef int                  PCI_EXID;
typedef char *               PCI_HANDLE;
typedef void                 (* PCI_PROCEDURE)();

/* Structures */
struct pci_mpb {
        PCI_INTEGER     MessageID;
        PCI_INTEGER     MessageMaximumSize;
        PCI_INTEGER     MessageActualUsedSize;
        PCI_INTEGER     DataMaximumSize;
        PCI_INTEGER     DataActualUsedSize;
};

struct pci_register {                       /* structure contenant les informations d'association */
        PCI_INTEGER PUFVersion;             /* facultatif: indiquer version du PUF */
        PCI_INTEGER PUFType;                /* facultatif: indiquer type du PUF */
        PCI_INTEGER MaxMsgSize;             /* retour: longueur maximale d'un message */
};

struct pci_opsys {                          /* structure contenant les informations d'association */
        int     DummyParameter;             /* Pas d'exigence spécifique pour WINDOWS */
};
```

```
/* Définition des fonctions */
#define PCI_PROPERTY                    (('Z' << 8) | 1)
#define PCI_REGISTER                    (('Z' << 8) | 2)
#define PCI_DEREGISTER                  (('Z' << 8) | 3)
#define PCI_SETSIGNAL                   (('Z' << 8) | 4)

/*****************************************************************
 *          Fonction PciGetProperty
 */
PCI_INTEGER PciGetProperty (PCIHandle, MaximumSize, NAFProperty, ActualSize)
        PCI_HANDLE PCIHandle;           /* char *       */
        PCI_INTEGER MaximumSize;        /* int          */
        PCI_BYTEARRAY NAFProperty       /* char *       */
        PCI_INTEGER * ActualSize;       /* int *        */
{
register int filedes;           /* descripteur de fichier */
struct strioctl     strioct;    /* pointeur sur la partie commande du message de flux */

*ActualSize = ERROR;            /* préréglé avec valeur d'erreur */

if ((filedes = open (PCIHandle, O_RDWR)) < Success)
        return ERROR;

strioct.ic_cmd          = PCI_PROPERTY;
strioct.ic_timeout      = 0;
strioct.ic_len          = MaximumSize;
strioct.ic_dp           = (char *) NAFProperty;

if (ioctl (filedes, I_STR, &strioct) == 0) {
        *ActualSize = strioct.ic_len;
        close (filedes);
        return 0;
        }
else
        {
        *ActualSize = 0;
        close (filedes);
        return errno;
        }
}

/*****************************************************************
 *          Fonction PciRegister
 */
PCI_INTEGER PciRegister (PCIHandle, pci_register, pcidummy, ExID)
        PCI_HANDLE      PCIHandle;      /* char *   */
        struct pci_register pciregister;
        struct pci_opsys    pcidummy;
        PCI_EXID *      ExID;           /* int *    */
{
struct strioctl     strioctl;

struct pci_register_t {
        int             puf_version;
        int             puf_type;
} pci_reg;

pci_reg.puf_version     = pciregister.PUFVersion;
pci_reg.puf_type        = pciregister.PUFType;
```

```
strioctl.ic_cmd                 = PCI_REGISTER;
strioctl.ic_timeout             = 0;
strioctl.ic_len                 = sizeof (pci_reg);
strioctl.ic_dp                  = (char *) &pci_reg;

if ((*ExID = open (PCIHandle, O_RDWR)) == –1)
        {
        *ExID = 0;
        return errno;
        }

if ((pciregister.MaxMsgSize = ioctl (*ExID, I_STR, &strioctl)) < 0)
        {
        pciregister.MaxMsgSize = 0;
        return errno;
        }
else
        {
        return 0;
        }
}


/*****************************************************************
 *      Fonction PciDeregister
 */
PCI_INTEGER PciDeregister (ExID)
        PCI_EXID ExID;  /* int          */
{
struct strioctl  strioctl;

strioctl.ic_cmd                 = PCI_DEREGISTER;
strioctl.ic_timeout             = 0;
strioctl.ic_len                 = 0;
strioctl.ic_dp                  = (char *) NULL;

if (ioctl (ExID, I_STR, &strioctl) == –1)
        {
        return errno;
        }
else
        {
        close (ExID);
        return 0;
        }
}

/*****************************************************************
 *      Fonction PciPutMessage
 */
PCI_INTEGER PciPutMessage (ExID, PCIMPB, Message, Data)
        PCI_EXID ExID;                  /* int      */
        struct pci_mpb * PCIMPB;
        PCI_BYTEARRAY Message;      /* char *   */
        PCI_BYTEARRAY Data          /* char *   */
{
struct strbuf ctlbuf;
struct strbuf databuf;
char  *buffer = NULL; /* pointeur sur un tampon, de capacité suffisante pour recevoir le pointeur PCIMPB et le contenu
                         de la partie "Message" */
int nErr;
```

```
if (Message && ((char *)Message != (char *)PCIMPB + sizeof(struct pci_mpb)))
        {
        /* Il y a une partie "Message" de valeur non égale à NULL et le pointeur PCIMPB ainsi que la partie
        "Message" ne sont pas contigus en mémoire. Il faut construire un tampon où le pointeur PCIMPB est suivi du
        contenu de la partie "Message" */
        /* Ici, un processus d'affectation de zones mémoire peut avoir lieu */
        buffer = (char *) (malloc( sizeof(struct pci_mpb) + PCIMPB->MessageActualUsedSize));

        memcpy (buffer, PCIMPB, sizeof(struct pci_mpb));
        memcpy (buffer + sizeof(struct pci_mpb), Message, PCIMPB->MessageActualUsedSize);
        ctlbuf.buf = buffer;
        ctlbuf.len = PCIMPB->MessageActualUsedSize + sizeof(struct pci_mpb);
}
else
        {
        /* ou bien il n'y a pas de partie "Message" ou bien le pointeur PCIMPB et la partie "Message" sont contigus en
        mémoire */
        ctlbuf.buf = (char *)PCIMPB;
        ctlbuf.len = Message ? PCIMPB->MessageActualUsedSize + sizeof(struct pci_mpb) : sizeof(struct pci_mpb);
        }

databuf.buf     = Data;
databuf.len     = Data ? PCIMPB->DataActualUsedSize : 0;

if (putmsg (ExID, &ctlbuf, &databuf, 0) != 0)
        nErr = errno;           /* errno contient le code d'erreur */
        }
else
        {
        nErr = 0;
        }
if (buffer != NULL) free(buffer);
return nErr;
}

/*****************************************************************
 *      Fonction PciGetMessage
 */
PCI_INTEGER PciGetMessage (ExID, PCIMPB, Message, Data)
        PCI_EXID ExID;                  /* int          */
        struct pci_mpb * PCIMPB;
        PCI_BYTEARRAY Message;      /* char *       */
        PCI_BYTEARRAY Data;         /* char *       */
{
struct strbuf ctlbuf;
int flags;
struct strbuf databuf;
char  *buffer = NULL; /* pointeur sur un tampon de capacité suffisante pour recevoir le pointeur PCIMPB et le contenu
                        de la partie"Message" */
int nErr = 0;

if (Message && ((char *)Message != (char *)PCIMPB + sizeof(struct pci_mpb)))
        {
        /* Il y a une partie "Message" de valeur non égale à NULL et  PCIMPB et la partie "Message" ne sont pas
        contigus en mémoire,
        il faut réserver un tampon où le pointeur PCIMPB peut être suivi du contenu de la partie "Message" */
        /* Ici, un processus d'affectation de zones mémoire peut avoir lieu */
        buffer = (char *) (malloc( sizeof(struct pci_mpb) + PCIMPB->MessageMaximumSize ));
        ctlbuf.buf = buffer;
        }
```

```
else {
        /* ou bien il n'y a pas de partie "Message" ou bien le pointeur PCIMPB et la partie "Message" sont contigus en
        mémoire */
        ctlbuf.buf  =  (char *)PCIMPB;
        }
ctlbuf.maxlen    = Message ? PCIMPB->MessageMaximumSize + sizeof(struct pci_mpb):sizeof(struct pci_mpb);
databuf.buf      = Data;
databuf.maxlen   = Data ? PCIMPB->DataMaximumSize : 0;

if (getmsg (ExID, &ctlbuf, &databuf, &flags) != 0)
        {
        /* Situation d'erreur, la variable "errno" sera définie       */
        nErr = errno;
}else {
        /* Réussite de l'opération     */
        if (ctlbuf.len != –1 && ctlbuf.len >= sizeof(struct pci_mpb)) {
                /* Un message, éventuellement de longueur 0, est présent */
                PCIMPB->MessageActualUsedSize         = ctlbuf.len - sizeof(struct pci_mpb);
                if (Message && ((char *)Message != (char *)PCIMPB + sizeof(struct pci_mpb)))
                        {
                        /* Il y a une partie "Message" de valeur non égale à NULL et PCIMPB et la partie "Message" ne
                        sont pas contigus en mémoire,
                        un tampon où le pointeur PCIMPB peut être suivi du contenu de la partie "Message", a été
                utilisé */
                        memcpy ( PCIMPB, buffer, sizeof(struct pci_mpb));
                        memcpy ( Message,(buffer + sizeof(struct pci_mpb)), (ctlbuf.len - sizeof(struct pci_mpb)));
                        }
                else
                        {
                        /* PCIMPB et Message sont contigus en mémoire, aucun tampon supplémentaire n'a été utilisé */
                        Message = (char *) (PCIMPB + sizeof(struct pci_mpb));
                        }
        }else   {
                /* Absence de partie "Message" ou message trop petit: erreur (au moins le pointeur PCIMPB devrait
                être présent) */
                PCIMPB->MessageID = 0;
                PCIMPB->MessageActualUsedSize = 0;
                }
if (databuf.len != –1)
        {
        /* Un bloc de données, éventuellement de longueur = 0, est présent */
        PCIMPB->DataActualUsedSize     = databuf.len;
        }
else
        {
        /* Absence de données */
        PCIMPB->DataActualUsedSize = 0;
        }
}
if (buffer != NULL) free( buffer);

return nErr;
}
```

```
/*****************************************************************
 *          Fonction PciSetSignal
 */
PCI_INTEGER PciSetSignal (ExID, Signal, SignalProcedure)

PCI_EXID ExID;                  /* int              */
PCI_INTEGER Signal;             /* int              */
PCI_PROCEDURE SignalProcedure;  /* void (*) ()      */
{
int Signal_options;
if (Signal == 0)
        {
        Signal_options = 0;
        if (ioctl (ExID, I_SETSIG, &Signal_options) == –1)
                return errno;
        signal (SIGPOLL, SIG_DFL);
        return 0;
        }
else
        {
        Signal_options = S_MSG;
        if (ioctl (ExID, I_SETSIG, &Signal_options) == –1)
                return errno;
        signal (SIGPOLL, SignalProcedure);
        return 0;
        }
}
```