P rotocole
d’ **E** changes pour un
S ystème
I nterbancaire de
T élécompensation

Septembre 1989
ISBN 2-906820-11-3

Cette version annule et remplace
la version D du 15 novembre 1987

SOMMAIRE

CHAPITRE 1 INTRODUCTION

1.1 OBJECTIF ET DOMAINE D'APPLICATION

L'objectif de ces spécifications est de définir le protocole de transfert de fichiers PeSIT.

Ce protocole est notamment utilisé pour le raccordement des Centres de Traitements Bancaires (CTB) des adhérents du réseau SIT aux stations du SIT. Dans les CTB l'application bancaire transfère ou reçoit les fichiers, le protocole PeSIT est utilisé pour ce transfert.

Mais l'usage du protocole PeSIT n'est pas réservé à ce raccordement et PeSIT peut être mis en œuvre dans les environnements les plus variés.

Afin de permettre l'interfonctionnement d'implémentations diverses de PeSIT, plusieurs profils d'utilisation ont été définis qui correspondent à des domaines d'application de PeSIT clairement définis.

Quatre profils (SIT, hors SIT, Hors SIT sécurisé et ETEBAC 5) sont décrits dans ces spécifications. D'autres pourront être définis si le besoin en était exprimé bien que la multiplication des profils ne soit pas souhaitable.

1.2 PRESENTATION DU DOCUMENT

1.2.1 Plan du document

Le **premier chapitre** constitue l'***Introduction*** de ce document. Après avoir présenté son plan, un deuxième paragraphe présente les modifications apportées, sur la forme comme sur le fond, par cette version E par rapport à la version D du 15 novembre 1987.

Le **deuxième chapitre *Architecture et présentation fonctionnelle***, situe d'abord PeSIT par rapport au modèle de référence ISO/OSI, présente la notion de fichier virtuel, puis les différentes fonctions assurées par PeSIT. Il introduit ensuite les notions d'unité fonctionnelle et de profil.

Le **troisième chapitre *Description du service PeSIT*** commence par une présentation de la notion de service, énumère les phases du service PeSIT, puis les différents services qui constituent le service PeSIT. Les unités fonctionnelles sont ensuite définies comme regroupement de services, puis la notion de profil est détaillée. Les paragraphes **Description des primitives de service** et **Description des paramètres** présentent une à une les primitives de service et leurs paramètres à l'exclusion de ceux uniquement liés à l'unité fonctionnelle Sécurisation. Le paragraphe **Description des profils** détaille les caractéristiques de chacun des profils. Le paragraphe **Sécurité dans le service PeSIT** regroupe tous les aspects du service touchés par l'unité fonctionnelle Sécurisation (primitives et paramètres). Le dernier paragraphe **Séquence de primitives type** présente les automates schématisant le service PeSIT et donne des exemples d'enchaînements de primitives.

Le **quatrième chapitre *Description du protocole PeSIT*** présente d'abord les correspondances entre primitives de service et éléments de protocole. Le paragraphe **Utilisation du service système de communication** présente ensuite les différents types d'interface pour PeSIT.F, PeSIT.F', PeSIT.F'', et PeSIT.F'''. Les procédures relatives à chacune des unités de protocole (FPDU) sont ensuite détaillées. Les paragraphes suivants donnent les règles de concaténation de FPDU puis celles d'utilisation des temporisations du protocole. Le paragraphe **Structure et codage des messages PeSIT (FPDU)** présente les conventions de codage des paramètres, donne la liste de ces paramètres puis, pour chaque paramètre son format dans chacun des quatre profils. Ensuite le contenu de chacune des FPDU est détaillé pour tous les profils. Le dernier paragraphe donne enfin les tables d'état du protocole PeSIT.

L'**annexe A** définit les différents modes de compression utilisables et leurs règles de négociation, l'**annexe B** présente comment PeSIT peut être utilisé en mode Store and Forward. L'**annexe C** apporte des précisions sur l'utilisation des mécanismes de sécurité (profils ETEBAC 5 et Hors SIT sécurisé), l'**annexe D** contient la liste des diagnostics d'erreur et l'**annexe E** un récapitulatif des paramètres et des éléments de protocole.

1.2.2 Nouveautés apportées par la version E

Sur le plan de la forme la version E introduit les notions d'unités fonctionnelles et de profils dont le but est de clarifier la présentation du document. L'accroissement important du nombre de paramètres dont les formats peuvent différer d'un profil à un autre, a conduit à consacrer une page à chaque paramètre dans laquelle son format et sa signification sont détaillés pour chaque profil. Par conséquent la présentation du contenu de chaque FPDU a été modifiée : pour chaque profil les FPDU sont représentées comme un assemblage de paramètres dont le graphisme indique le caractère (obligatoire, optionnel avec ou sans valeur implicite, conditionnel). Par contre, le contenu du paramètre n'est pas rappelé dans ces descriptions des FPDU.

Sur le plan du fond les différences sont les suivantes :

Profil SIT :

Le protocole PeSIT en profil SIT décrit ici est **inchangé** par rapport à celui décrit dans la version D du 15 novembre 1987. Toutes les particularités du PeSIT ont été regroupées dans le paragraphe 3.8.1 et se retrouvent dans les descriptions des paramètres et éléments de protocole pour le profil SIT. Certaines informations qui ne figuraient pas dans la version D de PeSIT mais dans des documents internes au projet SIT ont été introduites dans le paragraphe 3.8.1 si elles étaient utiles pour la réalisation d'un produit PeSIT pour le raccordement au réseau SIT.

Profil hors-SIT :

Le protocole PeSIT en profil Hors SIT a subi quelques modifications par rapport à la version D du 15 novembre 1987 :

- le paramètre **mot de passe** (PI 5) passe de 2 à 16 octets avec possibilité de modification dynamique ;
- le paramètre **type d'accès** (PI 22) peut prendre la valeur 2 : type d'accès mixte ;
- le paramètre **utilisation d'un CRC** (PI 1) est introduit dans la FPDU.CONNECT pour l'utilisation de PeSIT sur PAD ;
- la possibilité de recevoir dans la FPDU.ACK(SELECT) des paramètres type de fichier et nom de fichier (PI 11 et PI 12) différents de ceux contenus dans la FPDU.SELECT est offerte, ce qui permet de faire des **selects génériques** ;
- la possibilité d'utiliser PeSIT pour des transferts en mode **Store and Forward** est décrite. Les nouveaux paramètres identificateur d'émetteur initial et identificateur de destinataire final (PI 61 et PI 62) sont alors introduits ;
- la longueur des paramètres **identificateur demandeur et serveur** (PI 3 et PI 4) passe de 16 à 24 octets ;
- le paramètre identificateur de transfert (PI 13) est introduit optionnel dans la FPDU.ACK(CREATE) ;
- afin de supporter le format de fichier indexé les paramètres **longueur de la clé** (PI 38) et **déplacement de la clé** (PI 39) sont introduits ;
- les paramètres **aléa** et **résidu** (PI 24 et PI 35) sont supprimés ;

- le paramètre **message libre** (PI 99) dont la longueur maximale devient 254 octets est introduit optionnel dans la FPDU.ACK(CREATE) ;
- le paramètre **identificateur de fichier** (PGI 9) est introduit dans la FPDU.ACK(SELECT) ;
- un service de **transfert de message** est introduit qui utilise de nouvelles FPDU (FPDU.MSG, FPDU.MSGDM) FPDU.MSGMM, FPDU.MSGFM et FPDU.ACK(MSG) et un nouveau paramètre **message** ;
- le **padding** dans la compression verticale est précisé ;
- le comportement du récepteur du fichier lors de l'émission d'une FPDU.TRANS.END ou FPDU.ACK(TRANS.END) est précisé.

Afin de distinguer le PeSIT Hors SIT conforme à la version D, de celui décrit dans la version E le paramètre **numéro de version** (PI 6) sera utilisé.

Profil Hors-SIT sécurisé :

Ce profil est **nouveau**. Il suppose l'utilisation d'un ensemble de paramètres propres à la sécurité qui ont été introduits dans la version E (paramètres PI 71 à 83).

Profil ETEBAC 5 :

Ce profil est **nouveau**. Il a été défini en collaboration avec les groupes de travail ETEBAC 5 transport et ETEBAC 5 sécurité du **CFONB** (Comité Français d'Organisation et de Normalisation Bancaires) pour satisfaire le besoin de transfert de fichier du standard ETEBAC 5 (protocole d'Echanges Télématiques entre les Banques et leurs Clients). Ce standard est défini dans un document produit par le CFONB, disponible auprès du **GSIT**.

CHAPITRE 2 ARCHITECTURE ET PRESENTATION FONCTIONNELLE

2.1 MODELE DE REFERENCE

Deux idées importantes apparaissent pour le protocole PeSIT :

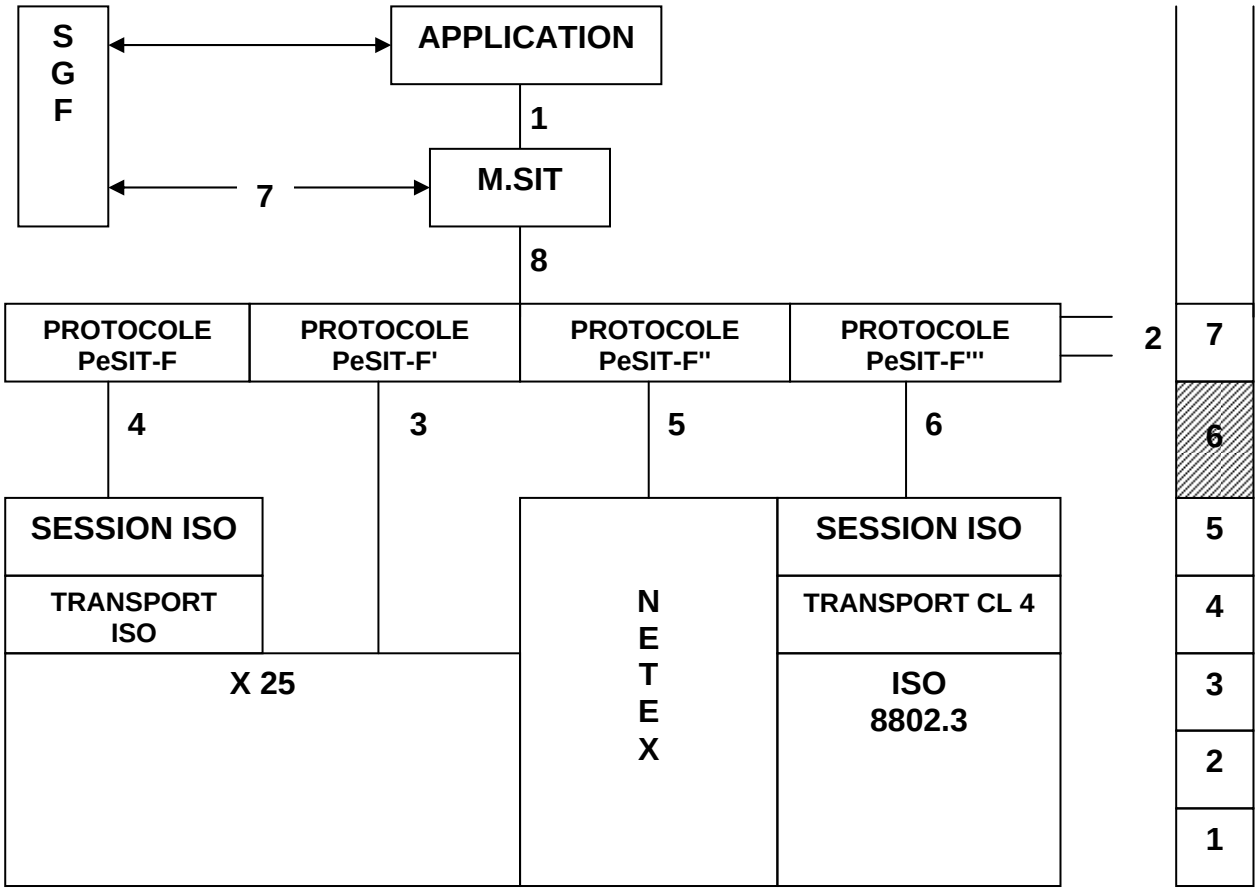
- *Interface stable par rapport aux applications.*
- *Unicité de protocole par rapport aux différentes couches de communications utilisées. L'inégale évolution des centres de traitement conduit à définir quatre versions de PeSIT, qui ne se différencient que par leur interface avec la couche inférieure utilisée.*
  - *PeSIT-F s'appuie sur la couche session normalisée ISO, pour l'utilisation sur un réseau à commutation de paquet (X-25).*
  - *PeSIT-F' s'appuie sur la couche réseau normalisée ISO (recommandation X-25 du CCITT),*
  - *PeSIT-F'' s'appuie sur la couche NETEX développée pour l'utilisation d'Hyperchannel.*
  - *PeSIT-F''' s'appuie sur la couche session normalisée ISO, pour l'utilisation sur un réseau local conforme à la norme ISO 8802-3.*

Le schéma de la page suivante décrit l'architecture logique dans laquelle s'insère PeSIT.

- Une application prépare et traite les fichiers échangés.
- Un moniteur de transfert ordonnance et gère les échanges.
- Une machine protocolaire (PeSIT) exécute le dialogue prévu.

Toutes utilisent, directement ou non, le système de gestion de fichier.

ARCHITECTURE FONCTIONNELLE PeSIT



Seule la machine protocolaire PeSIT est spécifiée dans ce document

Explication du schéma :

1 Interface de programmation entre l'application et le moniteur M.SIT. Cet interface stable est important pour assurer la pérennité des développements applicatifs.

2 Le protocole lui-même (PeSIT) qui est détaillé dans la deuxième partie de ce document.

3 Ensemble des primitives entre la couche 3 ISO et le protocole PeSIT-F' (partie 4.3.2 de ce document).

4 Ensemble des primitives entre la couche 5 ISO et le protocole PeSIT-F' (partie 4.3.1 de ce document).

5 Ensemble des primitives entre la couche NETEX et le protocole PeSIT-F'' (partie 4.3.3 de ce document).

6 Ensemble des primitives entre la couche session ISO et le protocole PeSIT-F''' (partie 4.3.4 de ce document).

7 Echanges entre le moniteur de transfert (M.SIT) et le système de gestion de fichier (primitives OPEN FILE, READ/WRITE, CLOSE FILE).

8 Primitives d'accès au service PeSIT.

2.2 MODELE DE FICHIER VIRTUEL

Il existe de nombreuses différences entre les mises en œuvre des systèmes de fichiers existants. Il est donc indispensable de disposer d'un modèle commun pour tout protocole concernant les fichiers d'un réseau hétérogène. Ce modèle est dit "fichier virtuel". Il permet d'effacer les différences de style et de spécification grâce à des fonctions de correspondance qui établissent une concordance entre les descriptions standards et les descriptions locales, et vice-versa.

Bien que l'objectif soit de traiter des fichiers réels, le modèle et le protocole ne concerneront que des fichiers virtuels. La correspondance entre fichiers réels et fichiers virtuels est considérée comme une opération du ressort de l'installation locale et c'est la raison pour laquelle elle n'est pas décrite dans le présent document.

2.3 PRESENTATION FONCTIONNELLE DU SERVICE ET DU PROTOCOLE PeSIT

2.3.1 Fonctions du service PeSIT

Les fonctions assurées par le service PeSIT sont :

- écriture de fichier à distance ;
- lecture de fichier à distance ;
- pose de point de synchronisation en cours de transfert ;
- reprise d'un transfert interrompu à partir d'un point de relance négocié ;
- resynchronisation en cours de transfert ;
- suspension de transfert ;
- sécurisation des transferts ;
- compression des données ;
- contrôle d'erreur ;
- transfert de message.

2.3.1.1 Ecriture de fichier

Cette fonction permet à un utilisateur du service PeSIT de transférer le contenu d'un fichier vers un autre utilisateur du service PeSIT. A cette fin, l'utilisateur émetteur doit avoir au préalable établi un lien logique appelé connexion avec l'autre utilisateur. L'utilisateur qui est à l'initiative de l'établissement de la connexion est appelé Demandeur, son correspondant est appelé Serveur. Dans ce cas, le transfert des données du fichier a donc lieu entre le Demandeur/émetteur et le Serveur/récepteur.

2.3.1.2 Lecture de fichier

Cette fonction permet à un utilisateur du service PeSIT de demander à un autre utilisateur du service PeSIT le transfert du contenu d'un fichier. A cette fin l'utilisateur qui désire recevoir le fichier doit avoir au préalable établi un lien logique appelé connexion avec l'autre utilisateur. Il est alors appelé Demandeur, son correspondant Serveur. Dans ce cas, le transfert des données du fichier a donc lieu entre le Serveur/émetteur et le Demandeur/récepteur.

2.3.1.3 Pose de point de synchronisation

Cette fonction permet à l'émetteur des données du fichier de poser des jalons, appelés points de synchronisation, numérotés séquentiellement, dans le cours du transfert. Le récepteur des données peut accuser réception de ces points, ce qui signifie qu'il a reçu et stocké correctement les données émises jusqu'à ce point. Ce mécanisme permet qu'en cas d'incident, un transfert soit repris à partir d'un point autre que le début du fichier.

2.3.1.4 Reprise d'un transfert

Cette fonction permet à l'utilisateur demandeur de reprendre un transfert interrompu avant son achèvement. Cette reprise est possible que le transfert soit en écriture ou en lecture, mais elle est toujours à l'initiative du demandeur. C'est le récepteur des données du fichier qui fixe le point de synchronisation à partir duquel le transfert doit reprendre.

2.3.1.5 Resynchronisation en cours de transfert

Cette fonction permet, en cas d'incident pendant le transfert des données, à un utilisateur de demander à son partenaire de reprendre le transfert à partir d'un point de synchronisation antérieur. A la différence de la reprise qui intervient après une interruption du transfert (accompagnée de fermeture et désélection du fichier) la resynchronisation a lieu en cours de transfert le fichier restant ouvert et sélectionné.

2.3.1.6 Suspension de transfert

Cette fonction permet à un utilisateur d'interrompre un transfert (donc de fermer et désélectionner le fichier concerné) pour réutiliser la connexion courante afin de procéder, sur cette connexion, à un transfert de priorité supérieure. Le transfert ainsi suspendu fera par la suite l'objet d'une procédure de reprise.

2.3.1.7 Sécurisation des transferts

Cette fonction permet aux utilisateurs du service PeSIT de mettre en œuvre des mécanismes concourant à :

- l'authentification réciproque des partenaires ;
- la confidentialité des données transmises ;
- l'intégrité des données transmises ;
- la non-répudiation réciproque.

2.3.1.8 Compression des données

Cette fonction permet aux utilisateurs du service PeSIT de mettre en œuvre des mécanismes de compression des données des fichiers transmis afin de réduire les volumes effectivement transférés.

2.3.1.9 Contrôle d'erreur

Cette fonction permet, à l'aide d'un polynôme détecteur d'erreur associé à chaque message du protocole PeSIT, de s'assurer que ceux-ci ne sont pas altérés par la transmission sur un support peu fiable.

2.3.1.10 Transfert de message

Cette fonction permet aux utilisateurs du service PeSIT de transférer une quantité d'informations de structure libre vers un autre utilisateur du service PeSIT. Les informations transférées d'utilisateur à utilisateur sont accompagnées d'un minimum d'informations de service qui permettent l'identification du transfert. Le transfert se fait dans ce cas avec un minimum d'échanges protocolaires. Les services de synchronisation, compression, reprise, resynchronisation, suspension ne sont par conséquent pas disponibles pour ce type de transfert.

Par contre, certaines fonctions de sécurité pourront être mises en œuvre dans le cadre de ce service :

- intégrité des données transmises ;
- non-répudiation réciproque.

2.3.2 Notion d'unités fonctionnelles

La mise en œuvre de toutes les fonctions décrites ci-dessus n'est pas toujours nécessaire pour satisfaire les besoins de l'utilisateur du service PeSIT. Par conséquent certaines de ces fonctions peuvent ne pas être assurées par certaines implémentations. C'est pourquoi ces fonctions sont regroupées en unités fonctionnelles.

Les unités fonctionnelles sont :

Noyau : Le noyau regroupe les services permettant l'établissement et la rupture d'un lien logique entre deux utilisateurs du service PeSIT (connexion PeSIT).

Ecriture : l'unité fonctionnelle Ecriture regroupe les services permettant la l'écriture d'un fichier à distance.

Lecture : l'unité fonctionnelle Lecture regroupe les services permettant la lecture d'un fichier à distance.

Synchronisation : l'unité fonctionnelle Synchronisation regroupe les services permettant la pose de points de synchronisation en cours de transfert.

Suspension : l'unité fonctionnelle Suspension regroupe les services permettant la suspension d'un transfert, c'est-à-dire l'interruption d'un transfert afin de réutiliser la connexion déjà établie pour un autre transfert de plus grande urgence.

Transfert de message : l'unité fonctionnelle Transfert de message regroupe les services permettant le transfert de message.

Contrôle d'erreur : l'unité fonctionnelle Contrôle d'erreur n'utilise pas de services spécifiques mais suppose la mise en œuvre par le protocole d'un mécanisme de détection d'erreur lors de l'émission et de la réception de chaque élément de protocole. L'utilisateur demande la mise en œuvre de ce service au moyen d'un paramètre présent dans la primitive de service qui sollicite l'établissement de la connexion avec l'utilisateur distant.

Sécurisation : l'unité fonctionnelle Sécurisation n'utilise pas de services spécifiques mais suppose la mise en œuvre de mécanismes de sécurité (notamment algorithmiques) qui nécessitent des échanges de paramètres entre l'utilisateur et le fournisseur du service PeSIT lors des diverses phases de service.

2.3.3 Notion de profil

L'utilisation du protocole PeSIT par une communauté d'utilisateurs ou pour une application particulière peut nécessiter le choix d'un certain nombre d'unités fonctionnelles à mettre en œuvre, de paramètres à utiliser ou de valeurs de paramètres à fixer. Ces choix caractérisent un profil d'utilisation du protocole.

A ce jour, quatre profils d'utilisation du protocole PeSIT ont été identifiés :
- le profil SIT,
- le profil hors SIT,
- le profil hors SIT sécurisé,
- le profil ETEBAC 5.

Ils se distinguent par :
- les unités fonctionnelles utilisées,
- les paramètres optionnels utilisés,
- la fixation de plages de valeurs autorisées pour certains paramètres,
- des conventions d'adressage (identification des demandeurs et serveurs) et de nommage des fichiers (types et noms de fichiers).

CHAPITRE 3 DESCRIPTION DU SERVICE PeSIT

3.1 PRESENTATION DU SERVICE

3.1.1 Objet et champ d'application

Le modèle de service décrit les interactions entre les entités utilisatrices du service (les applications) et l'entité fournisseur du service (la couche protocole). L'information est transmise entre l'utilisateur et le fournisseur du service au moyen de **primitives de service** qui peuvent transporter des paramètres.

Ce chapitre décrit d'une façon abstraite et tel qu'il est vu de l'extérieur le service fourni par la couche PeSIT à l'application utilisatrice en terme :

a) d'actions et événements spécifiés par les primitives de service,

b) de données, contenues dans les paramètres, associées à chaque action et événement spécifié par les primitives,

c) de relations entre actions et événements et d'enchaînements valides d'actions et d'événements.

Ce chapitre ne spécifie pas de forme particulière de réalisation ou de produits et n'impose aucune contrainte de réalisation pour les entités et interfaces d'un système.

3.1.2 Rôles des partenaires

Lors d'une session (*) PeSIT intervenant entre deux utilisateurs du PeSIT, le dialogue est toujours asymétrique, autrement dit, les deux utilisateurs jouent des rôles différents et complémentaires. L'initiateur de la session, appelé **DEMANDEUR** est celui qui définit le travail à exécuter au cours de la session : il est en relation plus ou moins directe avec l'usager pour le compte duquel il opère. L'autre utilisateur PeSIT, appelé **SERVEUR** exécute le travail proposé par le demandeur et lui fournit un compte rendu.

3.2 PHASES DE SERVICES

Le protocole PeSIT n'admet que le transfert d'un seul fichier à la fois lors d'une session donnée. Plusieurs fichiers peuvent être transférés simultanément au cours de plusieurs sessions en parallèle.

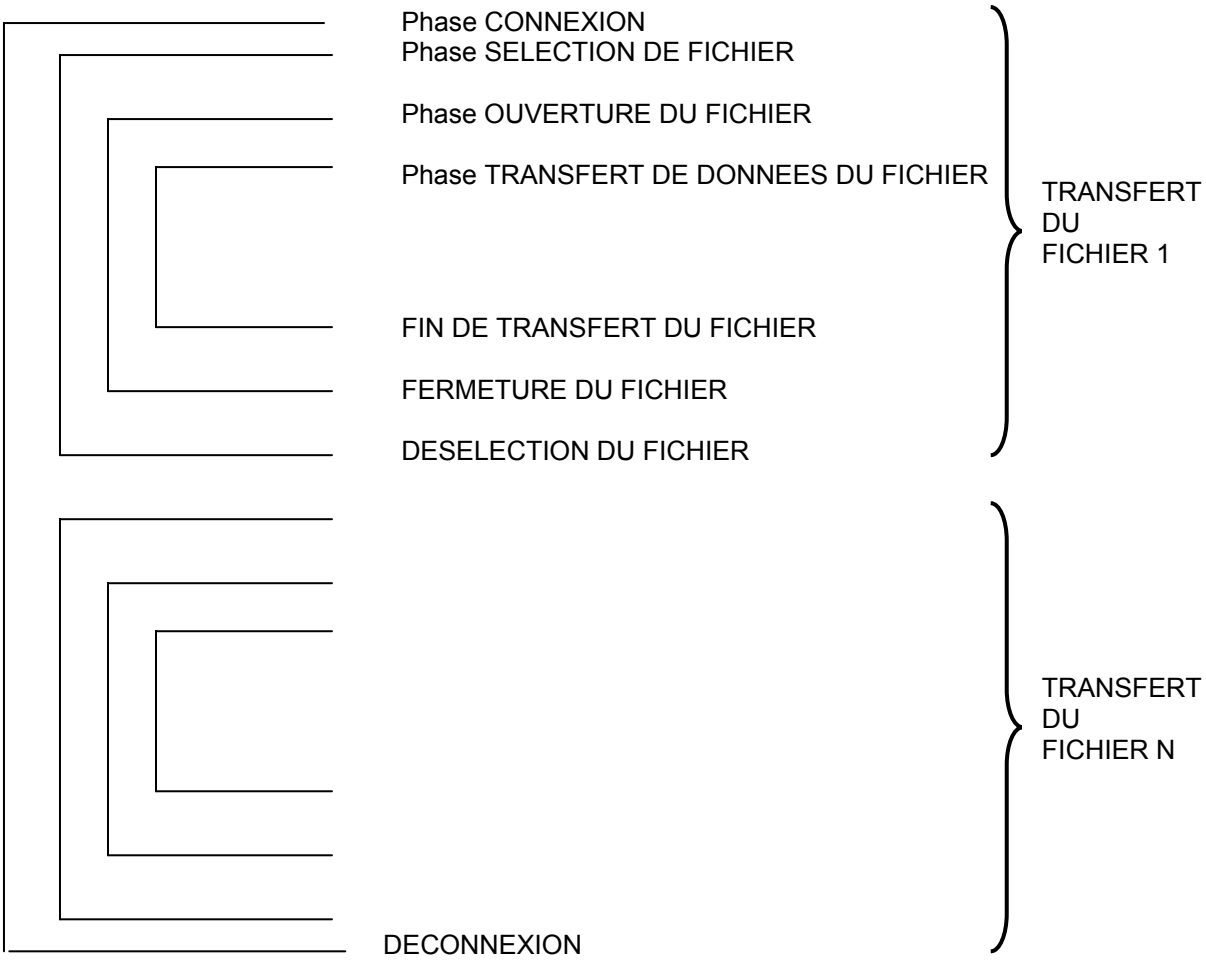
Le travail exécuté au cours d'une session PeSIT peut être structuré dynamiquement sous forme d'une suite de phases emboîtées qui doivent être ouvertes dans l'ordre hiérarchique et fermées dans l'ordre inverse. Si la session s'interrompt ou est arrêtée prématurément par un utilisateur, toutes les phases se trouvant ouvertes sont considérées comme implicitement fermées.

Ces phases sont imbriquées les unes dans les autres comme le montre le schéma suivant : une phase définit une étape dans le déroulement des opérations d'un transfert, dans laquelle un jeu spécifique de service peut être utilisé et pas d'autres. Il n’est pas possible de quitter une phase sans avoir au préalable quitté celles qu'elle contient.

---

* Une session est prise dans le sens "lien logique établi au travers du système de communication entre deux protocoles PeSIT distants".

LES PHASES DE SERVICE



Les phases sont les suivantes, dans l'ordre hiérarchique, en partant de la plus externe :

- Phase CONNEXION :

  Entre l'ouverture de la session et la fin de la session ; cette phase crée un lien logique entre deux utilisateurs homologues au travers d'une session PeSIT.

- Phase SELECTION DE FICHIER :

  Un fichier est réservé au transfert des données (entre la SELECTION/CREATION et la DESELECTION du fichier). Une phase CONNEXION contient un nombre quelconque de phases SELECTION DE FICHIERS (et éventuellement zéro).

- Phase OUVERTURE :

  Le fichier est prêt au transfert de données (entre l'ouverture et la fermeture du fichier).

- Phase TRANSFERT :

  Les données du fichier sont transmises (entre le début et la fin du transfert). Dans une phase de transfert, les utilisateurs PeSIT jouent le rôle de **L'EMETTEUR** et du **RECEPTEUR**.

3.3 LISTE DES SERVICES DU SERVICE PeSIT

Les différents services qui constituent le service PeSIT, classés par phase, sont :

Phase CONNEXION

service d'établissement de connexion

service de libération de connexion

service de coupure par l'utilisateur

service de coupure par le fournisseur.

Phase SELECTION DE FICHIER

service de création de fichier

service de sélection de fichier

service de désélection de fichier

service de transfert de message.

Phase OUVERTURE

service d'ouverture de fichier

service de fermeture de fichier.

Phase TRANSFERT

service d'écriture de fichier

service de lecture de fichier

service d'émission de données

service de pose de points de synchronisation

service de fin d'émission de données

service de fin de transfert

service d'interruption de transfert

service de resynchronisation de transfert.

3.4 UNITES FONCTIONNELLES

Les différents services sont regroupés par unités fonctionnelles. Les unités fonctionnelles et les services qu'elles nécessitent sont :

Noyau

Cette unité fonctionnelle regroupe les services :

- établissement de connexion
- libération de connexion
- coupure de l'utilisateur
- coupure par le fournisseur.

Ecriture

Cette unité fonctionnelle regroupe les services :

- création de fichier
- écriture de fichier
- ouverture de fichier
- désélection de fichier
- fermeture de fichier
- émission de données
- fin d'émission de données
- fin de transfert.

Lecture

Cette unité fonctionnelle regroupe les services :

- sélection de fichier
- lecture de fichier
- ouverture de fichier
- désélection de fichier
- fermeture de fichier
- fin d'émission de données
- fin de transfert.

Synchronisation

Cette unité fonctionnelle regroupe les services :

- pose de point de synchronisation.

Resynhronisation

Cette unité fonctionnelle regroupe les services :

- resynchronisation de transfert.

Suspension

Cette unité fonctionnelle regroupe les services :

- interruption de transfert.

Transfert de message

Cette unité fonctionnelle regroupe les services :

- transfert de message.

Sécurisation

Cette unité fonctionnelle ne nécessite pas de service particulier.

Contrôle d'erreur

Cette unité fonctionnelle ne nécessite pas de service particulier.

3.5 CARACTERISATION D'UN PROFIL

Un profil est caractérisé par :

- un ensemble d'unités fonctionnelles ;
- l'utilisation de paramètres optionnels au sein des primitives de services et par conséquent des éléments de protocole ;
- une limitation des valeurs possibles pour certains paramètres ;
- des conventions spécifiques d'adressage (identification des demandeurs et serveurs) et de nommage des fichiers (type et identifiant dans le type) ;

Le paragraphe 3.8 précise pour chaque profil quelles sont les unités fonctionnelles requises et indique quels sont les choix caractéristiques de chaque profil en ce qui concerne l'utilisation des paramètres.

Dans le chapitre 4 concernant le protocole PeSIT, le format de chaque paramètre et de chaque élément de protocole est détaillé, profil par profil.

Par contre, au niveau de la description des primitives de service PeSIT (paragraphes 3.6 et 3.7 ci-après) nous avons préféré présenter le service PeSIT tous profils confondus, mentionnant seulement quels paramètres étaient optionnels, étant entendu que des paramètres optionnels dans une description générale peuvent devenir obligatoire ou interdits dans un profil donné, ce qui est précisé pour chaque paramètre au chapitre 4.

Cependant, afin d'isoler l'unité fonctionnelle Sécurisation des transferts dont la compréhension nécessite une description globale et dont la mise en œuvre est très spécifique à certaines utilisations, nous avons volontairement omis tous les paramètres qui relèvent uniquement de cette unité fonctionnelle de la description générale des primitives de service (paragraphe 3.6) et des paramètres (paragraphe 3.7).

La description des primitives de service ainsi que des paramètres concernés par l'unité fonctionnelle Sécurisation (propre aux profils PeSIT Hors SIT sécurisé et ETEBAC 5) se trouvera donc dans le paragraphe 3.9 : Sécurité dans le service PeSIT.

3.6 DESCRIPTION DES PRIMITIVES DE SERVICE

3.6.1 Correspondance entre services et primitives

Le tableau suivant donne la correspondance entre les services et les primitives.

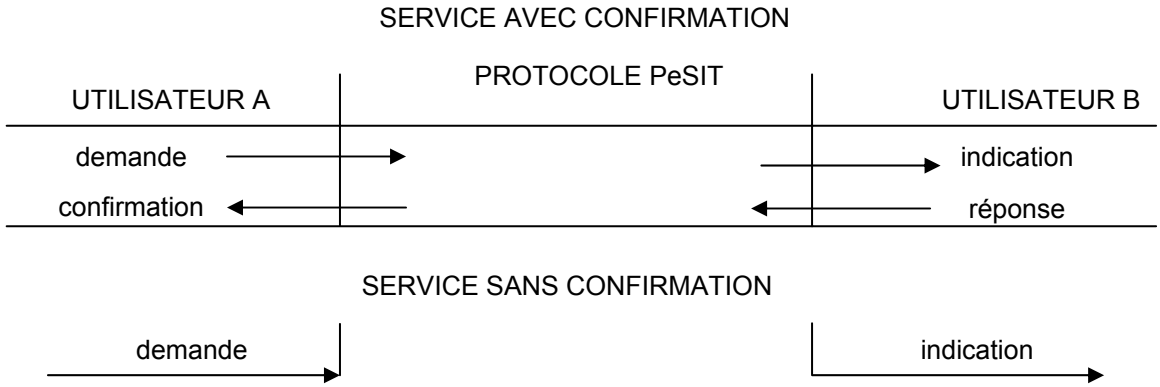
Primitive de service	Confirmation	Initiateur	Nom de service
F.CONNECT	OUI	Demandeur	Etablissement de connexion
F.RELEASE	OUI	Demandeur	Libération de connexion
F.ABORT	NON	Demandeur/Serveur	Coupure par l'utilisateur ou le fournisseur
F.CREATE	OUI	Demandeur	Création de fichier
F.SELECT	OUI	Demandeur	Sélection de fichier
F.DESELECT	OUI	Demandeur	Désélection de fichier
F.OPEN	OUI	Demandeur	Ouverture de fichier
F.CLOSE	OUI	Demandeur	Fermeture de fichier
F.WRITE	OUI	Demandeur	Ecriture de fichier
F.READ	OUI	Demandeur	Lecture de fichier
F.DATA	NON	Emetteur	Emission de données
F.DATA-END	NON	Emetteur	Fin d'émission de données
F.TRANSFER-END	OUI	Demandeur	Fin de transfert
F.CANCEL	OUI	Demandeur/Serveur	Interruption de transfert
F.CHECK	OUI	Emetteur	Pose de point de synchronisation
F.RESTART	OUI	Emetteur/Récepteur	Resynchronisation de transfert
F.MSG	OUI	Demandeur	Transfert de message

3.6.2 Conventions de description

Le concept de SERVICE est une notion abstraite qui définit les interactions entre la couche PeSIT et son utilisateur aux deux extrémités de la connexion logique. Il se définit au moyen de PRIMITIVES de service.

Il existe quatre types de primitives :
- la primitive de demande : un utilisateur sollicite l'apparition du service correspondant,
- la primitive d'indication : un utilisateur est informé de l'arrivée d'une demande de service,
- la primitive de réponse : un utilisateur répond à la demande d'un service correspondant,
- la primitive de confirmation : un utilisateur est informé de l'arrivée d'une réponse.

Chaque service élémentaire comprend une combinaison de primitives comme illustré sur les figures suivantes :



Chaque élément de service est décrit par :
- sa fonction,
- la table des paramètres associés.

Pour chaque paramètre, un "+" dans la colonne correspondante indique si le paramètre figure dans les primitives correspondantes.

La description des paramètres est regroupée dans le paragraphe 3.7.

Dans la table des paramètres, un numéro entre parenthèses, à côté de chaque paramètre, i.e. "(≠ i)" indique le sous-paragraphe du paragraphe 3.7 décrivant le paramètre.

3.6.3 Description des primitives

a) Le service F.CONNECT

*** Fonction**

Cet élément de service permet l'établissement d'un lien entre deux utilisateurs du service PeSIT. L'utilisateur qui initialise la primitive de demande F.CONNECT devient demandeur, celui qui reçoit la primitive d'indication F.CONNECT devient serveur. Le demandeur est responsable de la connexion jusqu'à la fin de l'activité. Si la connexion ne peut être établie, un code diagnostic donnant la raison du refus est retourné au demandeur.

Lors de l'établissement de la connexion certaines options (utilisation des unités fonctionnelles) sont négociées. Cette négociation concerne :
- le type d'accès "lecture", "écriture" ou "mixte" (unités fonctionnelles Ecriture ou Lecture),
- l'option points de synchronisation (unité fonctionnelle Synchronisation),
- la resynchronisation (utilisation ou non de l'unité fonctionnelle Resynchronisation),
- l'utilisation d'un polynôme détecteur d'erreur (unité fonctionnelle Contrôle d'erreur).

Le demandeur précise ses possibilités concernant ces unités fonctionnelles dans la primitive de demande F.CONNECT et le serveur lui répond par l'intersection entre ces spécifications et ses propres possibilités dans la primitive de réponse F.CONNECT.
D'autre part, la version du protocole utilisée est aussi négociée.

*** Table des paramètres**

PARAMETRES		F.CONNECT demande/indication	F.CONNECT réponse/confirmation
(≠a)	Utilisation d'un CRC	+	
(≠b)	Diagnostic		+
(≠c)	Identificateur du demandeur	+	
(≠c)	Identificateur du serveur	+	
(≠d)	Contrôle d'accès (optionnel)	+	+
(≠e)	Numéro de version	+	+
(≠f)	Option - point de synchronisation	+	+
(≠q)	Type d'accès	+	
(≠r)	Resynchronisation	+	+
(≠t)	Temporisation de surveillance (optionnel)	+	
(≠w)	Compléments de diagnostic (optionnel)		+
(≠ab)	Message libre (optionnel)	+	+

b) Le service F.RELEASE

*** Fonction**

Cet élément de service permet la libération de la connexion logique entre l'utilisateur demandeur et son homologue serveur. Seul l'utilisateur demandeur peut initialiser la primitive de demande F.RELEASE à n'importe quel moment de la connexion s'il n'y a pas de fichier sélectionné.

*** Table des paramètres**

PARAMETRES		F.RELEASE demande/indication	F. RELEASE réponse/confirmation
(≠b)	Diagnostic	+	
(≠w)	Compléments de diagnostic (optionnel)	+	
(≠ab)	Message libre (optionnel)	+	+

c) Le service F.ABORT

*** Fonction**

Cet élément de service permet de libérer brutalement et inconditionnellement la connexion logique en abandonnant toute activité en cours.

Lorsque l'élément de service F.ABORT a été initialisé ou reçu, toutes les phases du fichier se trouvant ouvertes doivent être fermées.

Aussi bien le demandeur que le serveur peuvent initialiser la primitive de demande F.ABORT à n'importe quel moment de la connexion.

F.ABORT est un service sans confirmation.

*** Table des paramètres**

PARAMETRES		F.ABORT demande/indication
(≠b)	Diagnostic	+
(≠w)	Compléments de diagnostic (optionnel)	+

d) Le service F.CREATE

*** Fonction**

Cet élément de service permet de créer un nouveau fichier à distance et de le sélectionner pour recevoir le fichier à transférer (transfert en écriture).

Seul l'utilisateur demandeur peut initialiser la primitive de demande F.CREATE et seulement à partir de la phase connexion, à condition que le type d'accès choisi lors de la connexion ait été "écriture" ou "mixte".

A la réception de la primitive d'indication F.CREATE, l'utilisateur serveur crée le fichier et le sélectionne avant de répondre par la primitive de réponse F.CREATE. En cas d'échec de la création la primitive de réponse F.CREATE peut être négative : le paramètre diagnostic indique alors la cause de cet échec.

Taille maximale de l'entité de donnée

A l'occasion de la phase de sélection du fichier se négocie la taille maximale de l'entité de donnée transmise à la couche de communication :
- le demandeur propose dans la primitive de demande F.CREATE la taille maximale qu'il souhaite utiliser,
- le serveur répond dans la primitive de réponse F.CREATE la taille maximale acceptée, qui doit être inférieure ou égale à celle proposée.

Dans la limite de cette taille maximale, il sera possible de concaténer plusieurs FPDU dans la même entité de donnée présentée à la couche de communication. La règle de concaténation des FPDU est donnée en 4.5.

Il est à noter que si l'unité fonctionnelle Segmentation n'est pas utilisée, il est nécessaire que la taille maximale d'entité de donnée retenue pour un transfert soit supérieure ou égale à la somme de la taille maximale d'article du fichier à transférer et de la longueur de l'en-tête de FPDU (six octets).

Identification du transfert

Dans la primitive de demande F.CREATE le paramètre identificateur de transfert est fixé à une valeur non nulle par le demandeur s'il s'agit d'un transfert nouveau. Le serveur peut indiquer alors dans la primitive de réponse F.CREATE une valeur (non nulle) d'identificateur de transfert différente de celle fixée par le demandeur. L'usage de l'identificateur de transfert fourni par le serveur dans la primitive de réponse F.CREATE est libre. En cas de relance d'un transfert le demandeur met dans la primitive de demande F.CREATE l'identificateur de transfert qui avait été utilisé lors de la première occurrence de ce transfert tel qu'il avait été fixé par le demandeur.

Relance d'un transfert

La relance a pour but d'éviter la répétition complète d'un transfert suspendu ou interrompu avant l'achèvement de la phase de désélection. La relance peut avoir lieu au cours de la même ou lors d'une nouvelle connexion. Dans tous les cas le fichier dont le transfert doit être relancé sera sélectionné et ouvert avec les mêmes paramètres que ceux utilisés lors du transfert initial. Le paramètre transfert relancé, dans le service F.CREATE, indique s'il s'agit d'une relance ou d'un transfert nouveau. Pour un transfert en écriture, la relance a lieu à partir un point de relance négocié au moyen du service F.WRITE.

*** Table des paramètres :**

PARAMETRES	F.CREATE demande/indication	F.CREATE réponse/confirmation
(≠b) Diagnostic		+
(≠g) Identificateur du fichier	+	+
(≠h) Identificateur du transfert	+	+
(≠j) Transfert relancé	+	
(≠k) Code-données	+	
(≠l) Priorité de transfert	+	
(≠s) Taille maximale d'une entité de données à transmettre	+	+
(≠w) Compléments de diagnostic (optionnel)		+
(≠x) Attributs du fichier	+	
(≠y) identifiant client (optionnel)	+	
(≠y) identifiant banque (optionnel)	+	
(≠z) Contrôle d'accès fichier (optionnel)	+	
(≠aa) Date et heure du serveur (optionnel)		+
(≠ab) Message libre (optionnel)	+	

e) Le service F.SELECT

*** Fonction**

Cet élément de service permet de sélectionner un fichier existant à distance pour la lecture.

Seul, l'utilisateur demandeur peut initialiser une primitive de demande F.SELECT, et seulement à partir de la phase connexion, à condition que le type d'accès choisi lors de la connexion ait été "lecture" ou "mixte".

Relance d'un transfert

De même qu'un transfert en écriture, un transfert en lecture peut faire l'objet d'une relance. La relance a pour but d'éviter la répétition complète d'un transfert suspendu ou interrompu avant l'achèvement de la phase de désélection. La relance peut avoir lieu au cours de la même ou lors d'une nouvelle connexion. Dans tous les cas, le fichier dont le transfert doit être relancé sera sélectionné et ouvert avec les mêmes paramètres que ceux utilisés lors du transfert initial. Le paramètre transfert relancé, dans le service F.SELECT, indique s'il s'agit d'une relance ou d'un transfert nouveau. Pour un transfert en lecture, la relance a lieu à partir d'un point de relance négocié au moyen du service F.READ.

Identification du fichier

Dans la primitive de demande F.SELECT, le paramètre nom de fichier peut soit être la référence nominative d'un fichier, soit être un ensemble de critères permettant au serveur de rechercher le ou les fichiers satisfaisant ces critères (numéro de version, date de création). De même, le paramètre type de fichier, dans la primitive de demande F.SELECT, peut soit désigner un type précis, soit un paramètre générique pour une classe de fichier. Dans le cas où le demandeur utilise un paramètre nom de fichier ou type de fichier générique, le serveur doit alors indiquer dans la primitive de réponse F.SELECT soit l'échec de sélection si aucun fichier ne satisfait les critères donnés par le demandeur, soit le type et le nom précis du fichier sélectionné.

Il est à noter que dans le cas de ces demandes génériques, chaque transfert ne concernera qu'un seul fichier identifié sans ambiguïté (pour des partenaires et une date donnés) par les paramètres type et nom de fichier présents dans la primitive de réponse F.SELECT. D'autre part, les paramètres type et nom de fichier seront de ce fait différents dans la primitive de demande et dans la primitive de réponse F.SELECT.

Identification du transfert

Dans la primitive de demande F.SELECT le paramètre identificateur de transfert doit être mis à zéro s'il s'agit d'un transfert nouveau. Le serveur indique alors dans la primitive de réponse F.SELECT la valeur (non nulle) de l'identificateur de transfert utilisée pour ce transfert. En cas de relance de transfert, le demandeur met dans la primitive de demande F.SELECT l'identificateur de transfert qui avait été utilisé lors de la première occurrence de ce transfert tel qu'il avait été fixé par le serveur.

*** Tables des paramètres :**

PARAMETRES	F.SELECT demande/indication	F.SELECT réponse/confirmation
(≠b) Diagnostic		+
(≠g) Identificateur du fichier	+	+
(≠h) Identificateur du transfert	+	+
(≠i) Attributs demandés	+	
(≠j) Transfert relancé	+	
(≠k) Code-données		+
(≠l) Priorité de transfert	+	
(≠s) Taille maximale d'une entité de données à transmettre	+	+
(≠w) Compléments de diagnostic (optionnel)		+
(≠x) Attributs du fichier		+
(≠y) Identifiant client (optionnel)	+	
(≠y) Identifiant banque (optionnel)	+	
(≠z) Contrôle d'accès fichier (optionnel)	+	
(≠aa) Date et heure du serveur (optionnel)		+
(≠ab) Message libre (optionnel)	+	

f) Le service F.OPEN

*** Fonction**

Cet élément de service permet d'ouvrir un fichier à distance.

Seul, l'utilisateur demandeur peut initialiser une primitive de demande F.OPEN et seulement après avoir sélectionné le fichier. A la réception de la primitive d'indication F.OPEN, l'utilisateur serveur ouvre le fichier précédemment sélectionné avant de répondre par la primitive de réponse F.OPEN.

C'est dans cette phase que le contexte de présentation des données est négocié. L'option possible est :

- la compression

Le demandeur indique dans la primitive de demande F.OPEN s'il désire utiliser la compression, et selon quel algorithme, pour le transfert courant. Le serveur indique dans la primitive de réponse F.OPEN s'il peut effectivement supporter la compression et quel est l'algorithme retenu. Le détail des algorithmes de compression et des règles de négociation de ces algorithmes est donné en Annexe A.

*** Tables des paramètres :**

PARAMETRES	F.OPEN demande/indication	F.OPEN réponse/confirmation
(≠b) Diagnostic		+
(≠p) Compression	+	+
(≠w) Compléments de diagnostic (optionnel)		+

g) Le service F.CLOSE

*** Fonction**

Cet élément de service permet de fermer le fichier à distance, précédemment ouvert. Lorsque cet élément de service a été initialisé et reçu, aucun autre élément de service ne peut être initialisé avant la réception de la réponse à F.CLOSE. La demande de fermeture d'un fichier ne peut pas être rejetée.

Seul l'utilisateur demandeur peut initialiser la primitive de demande F.CLOSE. A la réception de la primitive d'indication F.CLOSE, l'utilisateur serveur doit inhiber toute action en cours et fermer le fichier avant de répondre par la primitive de réponse F.CLOSE.

*** Tables des paramètres :**

PARAMETRES	F.CLOSE demande/indication	F.CLOSE réponse/confirmation
(≠b) Diagnostic	+	+
(≠w) Compléments de diagnostic (optionnel)	+	+

h) Le service F.DESELECT

*** Fonction**

Cet élément de service permet de libérer l'association créée entre l'utilisateur demandeur et le fichier sélectionné. La demande de libération de fichier ne peut pas être rejetée.

Seul l'utilisateur demandeur peut initialiser une primitive de demande F.DESELECT. A la réception de la primitive d'indication F.DESELECT, l'utilisateur serveur libère le fichier en cours avant de répondre par la primitive de réponse F.DESELECT. Après la libération, le fichier continue d'exister et est disponible pour une éventuelle sélection.

*** Tables des paramètres :**

PARAMETRES	F.DESELECT demande/indication	F.DESELECT réponse/confirmation
(≠b) Diagnostic	+	+
(≠w) Compléments de diagnostic (optionnel)	+	+

i) Le service F.READ

*** Fonction**

Cet élément de service permet d'initialiser le transfert de données en lecture du fichier à partir d'un point donné (début du fichier ou point de relance). Le service F.READ ne peut être initialisé que par l'utilisateur demandeur, à l'issue de la phase d'ouverture du fichier et si celle-ci a été précédée d'un service F.SELECT.

F.READ indique que les données du fichier seront transférées de l'utilisateur serveur vers l'utilisateur demandeur.

Détermination du point de relance

C'est dans cette phase qu'est négocié le point de relance, si le service F.SELECT indiquait que ce transfert est une relance. Le point de relance est fixé par le récepteur des données qui sait ce qu'il a correctement reçu. Il figure donc dans la primitive de demande F.READ. Il doit être soit nul (relance à partir du début du fichier), soit supérieur ou égal au dernier point de synchronisation acquitté par le demandeur récepteur. L'émetteur des données n'a donc pas à conserver le contexte concernant les points de synchronisation antérieurs au dernier acquitté. L'utilisateur serveur émetteur peut indiquer dans la primitive de réponse F.READ (paramètre diagnostic) son incapacité à reprendre le transfert à partir du point de relance souhaité par le demandeur.

*** Tables des paramètres :**

PARAMETRES	F.READ demande/indication	F.READ réponse/confirmation
(≠b) Diagnostic		+
(≠m) Point de relance	+	
(≠w) Compléments de diagnostic (optionnel)		+

j) Le service F.WRITE

*** Fonction**

Cet élément de service permet d'initialiser le transfert de données en écriture du fichier à partir d'un point donné (début du fichier ou point de relance). Le service F.WRITE ne peut être initialisé que par l'utilisateur demandeur, à l'issue de la phase d'ouverture du fichier et si celle-ci a été précédée d'un service F.CREATE.

F. WRITE indique que les données du fichier seront transférées de l'utilisateur demandeur vers l'utilisateur serveur.

Détermination du point de relance

C'est dans cette phase qu'est négocié le point de relance, si le service F.CREATE indiquait que ce transfert est une relance. Le point de relance est fixé par le récepteur des données qui sait ce qu'il a correctement reçu. Il figure donc dans la primitive de réponse F.WRITE. Il doit être soit nul (relance à partir du début du fichier), soit supérieur ou égal au dernier point de synchronisation acquitté par le serveur/récepteur. L'émetteur des données n'a donc pas à conserver le contexte concernant les points de synchronisation antérieurs au dernier acquitté.

*** Tables des paramètres :**

PARAMETRES		F.WRITE demande/indication	F.WRITE réponse/confirmation
(≠b)	Diagnostic		+
(≠m)	Point de relance		+
(≠w)	Compléments de diagnostic (optionnel)		+

k) Le service F.DATA

*** Fonction**

Cet élément de service permet de transférer un article de données du fichier de l'émetteur vers le récepteur.

Seul l'utilisateur demandeur (qui peut être le demandeur ou le serveur selon que F.READ ou F.WRITE a été initialisé) peut initialiser la primitive de demande F.DATA.

F. DATA est un service sans confirmation.

*** Tables des paramètres :**

PARAMETRES	F.DATA demande/indication
(≠ac) Article du fichier	+

l) Le service F.DATA.END

*** Fonction**

Cet élément de service signale la fin du transfert de données. Il est initialisé seulement quand toutes les données du fichier ont été transférées.

Seul l'utilisateur émetteur peut initialiser la primitive de demande F.DATA.END.

F.DATA.END est un service sans confirmation.

*** Tables des paramètres :**

PARAMETRES	F.DATA.END demande/indication
(≠b) Diagnostic	+
(≠w) Complément de diagnostic (optionnel)	+

m) Le service F.TRANSFERT.END

*** Fonction**

Cet élément de service signale la fin de la phase transfert de données du fichier.

Seul l'utilisateur demandeur peut initialiser la primitive de demande F.TRANSFER.END.

Remarque

Dans le cas **demandeur émetteur**, la demande suit directement le F.DATA.END, et la réponse, envoyée par le serveur, tient lieu d'acquittement de tous les points de synchronisation. Elle signifie notamment que toutes les données ont été reçues et écrites dans le fichier par le serveur.

Dans le cas **demandeur récepteur**, la demande est envoyée après réception du F.DATA.END et tient lieu d'acquittement de tous les points de synchronisation. Elle signifie notamment que toutes les données ont été reçues et écrites dans le fichier par le demandeur.

*** Tables des paramètres :**

PARAMETRES	F.TRANSFER.END demande/indication	F.TRANSFER.END réponse/confirmation
(≠b) Diagnostic		+
(≠u) Nombre d'octets de données (optionnel)	+	+
(≠v) Nombre d'articles (optionnel)	+	+
(≠w) Compléments de diagnostic (optionnel)		+

n) Le service F.CANCEL

*** Fonction**

Cet élément de service permet d'interrompre la transmission de données pendant la phase de transfert.

Aussi bien l'utilisateur demandeur que l'utilisateur serveur peuvent initialiser la primitive de demande F.CANCEL.

*** Tables des paramètres :**

PARAMETRES	F.CANCEL demande/indication	F.CANCEL réponse/confirmation
(≠b) Diagnostic	+	
(≠n) Code fin de transfert	+	
(≠w) Compléments de diagnostic (optionnel)	+	

o) Le service F.CHECK

*** Fonction**

Cet élément de service permet de poser des points de synchronisation dans les séquences de données transférées. Seul l'utilisateur émetteur peut initialiser la primitive de demande F.CHECK pendant la phase de transfert de données. Il n'est utilisé que si l'unité fonctionnelle Synchronisation a été retenue lors de la phase de connexion.

Lors de cette négociation se détermine le nombre maximal d'octets de données qui peuvent être transmis entre deux services F.CHECK, et les règles de confirmation de ce service.

Le service F.CHECK est un service sans confirmation si lors de la négociation de l'utilisation de l'unité fonctionnelle Synchronisation la fenêtre d'acquittement des points de synchronisation a été retenue nulle. Sinon la valeur de la fenêtre d'acquittement des points de synchronisation détermine le nombre de primitives de service F.CHECK qui peuvent être initialisées successivement sans attendre la réception d'une primitive de confirmation.

Il n'est pas nécessaire que chaque F.CHECK fasse l'objet d'une confirmation individuelle car la confirmation d'un point de synchronisation donné confirme implicitement tout point de synchronisation antérieur non confirmé.

Avant d'émettre une primitive de réponse F.CHECK le récepteur doit forcer l'écriture dans le fichier de toutes les données déjà reçues.

*** Tables des paramètres :**

PARAMETRES	F.CHECK demande/indication	F.CHECK réponse/confirmation
(≠o) Numéro du point de synchronisation	+	+

p) Le service F.RESTART

*** Fonction**

Cet élément de service permet de resynchroniser le transfert à partir d'un point de synchronisation antérieur. Il ne peut être utilisé que si les unités fonctionnelles Synchronisation et Resynchronisation ont été retenues lors de la phase de connexion. Aussi bien le demandeur que le serveur peuvent initialiser la primitive F.RESTART pendant la phase de transfert de données. Après accord entre les deux partenaires sur le point de relance, le transfert des données reprend depuis ce point sans qu'il y ait rupture de la phase transfert des données.

Détermination du point de relance

La relance peut avoir lieu soit à partir du début du fichier (point de relance 0), soit à partir d'un point de synchronisation postérieur ou égal au dernier point de synchronisation acquitté. La détermination du point de relance est l'objet d'une négociation au moyen des primitives de demande et de réponse F.RESTART. Comme pour la reprise, c'est le récepteur des données qui détermine le point de relance.

Dans le cas d'une relance à l'initiative du récepteur, celui-ci indique dans la primitive de demande F.RESTART le point de relance souhaité (supérieur ou égal au dernier qu'il a acquitté) et l'émetteur peut soit accepter la relance à partir de ce point (primitive de réponse F.RESTART avec point de relance identique à celui proposé dans la primitive de demande F.RESTART), soit demander la relance depuis le début du fichier (primitive de réponse F.RESTART avec point de relance nul).

Dans le cas d'une relance à l'initiative de l'émetteur, celui-ci indique dans la primitive de demande F.RESTART le point de relance souhaité (supérieur ou égal au dernier dont il a reçu un acquittement) et le récepteur peut soit accepter la relance à partir de ce point ou d'un point ultérieur (primitive de réponse F.RESTART avec point de relance supérieur ou égal à celui proposé dans la primitive de demande F.RESTART), soit demander la relance depuis le début du fichier (primitive de réponse F.RESTART avec point de relance nul).

*** Tables des paramètres :**

PARAMETRES	F.RESTART demande/indication	F.RESTART réponse/confirmation
(≠b) Diagnostic	+	
(≠m) point de relance	+	+
(≠w) Compléments de diagnostic (optionnel)	+	

q) Le service F.MESSAGE

*** Fonction**

Cet élément de service permet à un utilisateur du service PeSIT d'émettre une quantité d'informations de format libre vers un autre utilisateur du service PeSIT. Seul l'utilisateur peut initialiser la primitive de demande F.MESSAGE et seulement à partir de la phase de connexion.

Le service F.MESSAGE est un service avec confirmation.

*** Tables des paramètres :**

PARAMETRES	F.MESSAGE demande/indication	F.MESSAGE réponse/confirmation
(≠b) Diagnostic		+
(≠g) Identificateur du fichier	+	
(≠h) Identificateur du transfert	+	+
(≠k) Code-données	+	+
(≠w) Compléments de diagnostic (optionnel)		+
(≠i) Attributs demandés	+	
(≠x) Attributs du fichier	+	
(≠y) Identifiant client (optionnel)	+	
(≠y) Identifiant banque (optionnel)	+	
(≠ad) Message (optionnel)	+	+

3.7 DESCRIPTION DES PARAMETRES

1.1.1.11.a) Utilisation du CRC

Ce paramètre indique si un polynôme détecteur d'erreur (CRC) est ajouté à chaque FPDU pour vérifier l'intégrité des messages (voir 4.3.2). Ce paramètre est obligatoire dans le cas d'un accès à travers un PAD.

1.1.1.12.b) Diagnostic

Ce paramètre indique la gravité et la nature d'une erreur rencontrée. Il se compose des champs :

- type d'erreur qui indique la gravité de l'erreur,

- code raison qui donne le détail de la nature de l'erreur.

La liste des diagnostics, mentionnant dans quelle primitive de service ils peuvent se rencontrer, est donnée en Annexe D.

1.1.1.13.c) Identification du demandeur et du serveur

Paramètre précisant le nom de l'utilisateur demandeur ou demandé.

1.1.1.14.d) Contrôle d'accès

Clé d'accès permettant une identification réciproque du demandeur et du serveur (mot de passe). Un utilisateur peut modifier son mot de passe et indique alors son nouveau mot de passe, dans le paramètre Contrôle d'accès, à la suite du mot de passe à remplacer.

1.1.1.15.e) Numéro de version

Numéro de version logiciel. En cas de non compatibilité entre la version annoncée par le demandeur dans la primitive de demande F.CONNECT et celle supportée par le serveur celui-ci peut refuser l'établissement de la connexion. Le serveur peut aussi proposer un numéro de version inférieur à celui annoncé par le demandeur, celui-ci ayant alors le choix entre refuser la connexion ou accepter d'utiliser la version demandée par le serveur.

Le numéro de version 1 correspond au protocole décrit dans la version D des Spécifications Techniques de PeSIT, datée du 15 novembre 1987.
Le numéro de version 2 correspond au protocole décrit dans la version E des Spécifications Techniques de PeSIT, datée du 14 juillet 1989.

1.1.1.16.f) Option-point de synchronisation

Ce paramètre permet de négocier l'unité fonctionnelle Synchronisation

- intervalle entre points de synchronisation :

  * la valeur 0 de ce champ indique : pas de point de synchronisation ;

  * la valeur 65 535 de ce champ indique : intervalle indéfini ;

* toute autre valeur indique le nombre maximal d'octets du fichier (exprimé en kilo-octets, 1 kilo-octet = 1 024 octets) que l'émetteur des données peut transmettre entre deux points de synchronisation consécutifs. Cette valeur prend en compte uniquement les longueurs des champs données des FPDU.DTF (hors en-tête FPDU), en excluant les octets de longueur d'articles pour les FPDU multi-articles, mais après compression et en comptant les en-têtes de chaînes de compression s'il y a lieu.

- fenêtre d'acquittement :

* la valeur 0 de ce champ indique : pas d'acquittement des points de synchronisation ;

* une valeur non nulle désigne la fenêtre d'acquittement des points de synchronisation. La fenêtre d'acquittement des points de synchronisation est la différence maximale possible entre, d'une part :

  - le numéro du dernier point de synchronisation émis,

  - et d'autre part, le numéro du dernier point de synchronisation acquitté.

Lorsque la borne de la fenêtre est atteinte, l'émission de données est suspendue jusqu'à réception de l'acquittement d'un point de synchronisation. Dans le cas d'une fenêtre différente de un, il n'est pas nécessaire que chaque point de synchronisation soit l'objet d'un acquittement explicite, l'acquittement d'un point acquittant implicitement tous les points de synchronisation antérieurs.

Négociation de l'option synchronisation

La négociation a lieu lors de la phase de connexion. Le demandeur propose les valeurs qu'il souhaite voir utilisées, et le serveur donne le résultat de la négociation en fonction des règles suivantes :

- si le demandeur et le serveur désirent tous les deux l'utilisation de points de synchronisation, l'option est sélectionnée. Sinon elle ne l'est pas ;
- si l'option est sélectionnée, le serveur a la possibilité de choisir un intervalle entre point de synchronisation et/ou une fenêtre d'acquittement plus réduite que ceux proposés par le demandeur.

1.1.1.17.g) Identificateur du fichier

Ce paramètre échangé entre demandeur et serveur pendant la phase de sélection de fichier contribue à l'identification d'un fichier. L'identification non ambiguë d'un fichier peut requérir d'autres paramètres, différents selon les profils d'utilisation : Cf. § 3.8.

L'identificateur de fichier est composé de :

- identification du demandeur : paramètre optionnel, s'il est absent la valeur par défaut est la précédente valeur utilisée, soit pendant la phase de connexion, soit lors de la dernière phase de sélection où il figurait. S'il est présent sa valeur peut être différente de celle utilisée pendant la phase de connexion ou la précédente phase de sélection et devient celle valide jusqu'à rupture de la connexion ou utilisation d'une valeur différente lors d'une autre phase de sélection ;
- identification du serveur : paramètre optionnel, ses règles d'utilisation sont identiques à celles de l'identification du demandeur ;
- type de fichier : ce paramètre définit à quelle classe appartient un fichier : son utilisation est propre à un profil d'utilisation donné ;
- nom du fichier : ce paramètre permet d'identifier correctement un fichier pour un type donné.

Dans le cas d'une lecture les paramètres type de fichier et nom du fichier présents dans la primitive de demande F.SELECT pourront avoir des valeurs génériques désignant un ensemble de fichiers. Dans ce cas, les paramètres type de fichier et nom de fichier présents dans la primitive de réponse F.SELECT seront différents de ceux-ci et désigneront un fichier appartenant à l'ensemble préalablement défini. Si le serveur ne trouve aucun fichier dans cet ensemble il répondra par une primitive de réponse F.SELECT négative.

1.1.1.18.h) Identificateur du transfert

Valeur numérique permettant d'identifier un transfert. Dans le cas d'un transfert relancé l'identifiant de transfert utilisé pour la relance doit être identique à celui utilisé pour le transfert initial.

Dans le cas d'une écriture l'identificateur de transfert est fixé par le demandeur (valeur différente de zéro dans la primitive de demande F.CREATE).

Dans le cas d'une lecture l'identificateur de transfert est fixé par le serveur. Le demandeur doit mettre un identificateur de transfert nul (sauf s'il s'agit d'une relance) dans la primitive de demande F.SELECT. Le serveur fixe alors la valeur (non nulle) de l'identificateur de ce transfert qu'il envoie dans la primitive de réponse F.SELECT. Dans le cas d'une relance en lecture le demandeur met dans la primitive de demande F.SELECT l'identificateur de transfert que le serveur avait choisi lors de la tentative initiale de transfert, et le serveur doit répondre dans la primitive de réponse F.SELECT avec la même valeur.

1.1.1.19.i) Attributs demandés

Indique les attributs du fichier dont les valeurs doivent être communiquées en réponse au demandeur. Il s'agit d'une combinaison quelconque (éventuellement vide) des catégories suivantes : logique, physique ou historique.

1.1.1.110.j) Transfert relancé

Ce paramètre indique qu'un transfert est une nouvelle occurrence d'un transfert qui a déjà fait l'objet d'une tentative. La relance a toujours lieu à l'initiative du demandeur, mais c'est le récepteur qui fixe le point de relance à partir duquel le transfert sera repris.

1.1.1.111.k) Code-données

Ce paramètre indique le type de codage utilisé pour la transmission des données du fichier. Ces valeurs sont "binaire" (transmission transparente), "ASCII" ou "EBCDIC".

1.1.1.112.l) Priorité de transfert

Ce paramètre donne la priorité relative attribuée par le demandeur à un transfert.

1.1.1.113.m) Point de relance

C'est le numéro de point de synchronisation permettant de retrouver la position dans le fichier à partir de laquelle la transmission doit être reprise. Sa valeur est fixée par le récepteur du fichier dans la primitive de réponse F.WRITE, dans le cas d'une écriture, dans la primitive de demande F.READ, dans le cas d'une lecture. La valeur zéro désigne le début du fichier.

1.1.1.114.n) Code fin de transfert

Code précisant le type de fin de transfert de données dans une primitive F.CANCEL. Ses valeurs sont les suivantes :

- Erreur,

- Suspension,

- Annulation par le serveur,

- Annulation par le demandeur.

1.1.1.115.o) Numéro de point de synchronisation

Valeur numérique permettant d'identifier sans ambiguïté un point de synchronisation. Le numéro de point de synchronisation est incrémenté d'une unité à chaque F.CHECK successif, en partant de la valeur 1 pour le premier point de synchronisation. La valeur maximale est 999 999.

1.1.1.116.p) Compression

Ce paramètre permet de négocier, lors de la phase d'ouverture, l'utilisation de la compression pendant la transmission des données du fichier. Les types de compression possibles sont :

- compression horizontale,

- compression verticale,

- combinaison des compressions horizontale et verticale.

Le détail des mécanismes de négociation de la compression et les algorithmes sont détaillés dans l'Annexe A.

1.1.1.117.q) Type d'accès

Indique le type d'accès prévu pour le transfert "Ecriture, Lecture ou mixte"

- **Ecriture** : le cas où le demandeur est l'émetteur.
- **Lecture** : le cas où le serveur est l'émetteur.
- **Mixte** : le cas où, au cours de la même connexion, Ecriture et Lecture peuvent se succéder.

1.1.1.118.r) Resynchronisation

Ce paramètre permet de négocier pendant la phase de connexion l'utilisation de l'unité fonctionnelle Resynchronisation (utilisation du service F.RESTART).

1.1.1.119.s) Taille maximale d'une entité de données

Ce paramètre correspond au nombre maximal d'octets pouvant être transmis dans une entité de données (NSDU, SSDU…). Sa valeur se négocie pendant la phase de sélection du fichier. Le demandeur propose une taille maximale et le serveur répond une valeur maximale inférieure ou égale. C'est en fonction de la valeur retenue pour la taille maximale d'entité de données et de la taille des articles de fichier à transférer que les mécanismes de segmentation, de concaténation et de FPDU multi-articles sont mis en œuvre.

1.1.1.120.t) Temporisation de surveillance

Ce paramètre permet de fixer lors de la phase de connexion la valeur de la temporisation de surveillance du protocole qui sera utilisée dans le cours de cette connexion.

1.1.1.121.u) Nombre d'octets de données

Ce paramètre représente le nombre d'octets de données (hors octets longueurs dans le cas des FPDU multi-articles, mais incluant les en-têtes de chaîne de compression s'il y a lieu) qui ont été effectivement soit transmis, soit reçus pour un fichier pendant la phase transfert des données. Il est utilisé dans les primitives de service F.TRANSFER.END à des fins de vérification.

1.1.1.122.v) Nombre d'articles

Ce paramètre représente le nombre d'articles effectivement transmis ou reçus pour un fichier pendant la phase transfert des données. Il est utilisé dans les primitives de service F.TRANSFER.END à des fins de vérification.

1.1.1.123.w) Compléments de diagnostic

Ce paramètre peut contenir des informations complémentaires à un diagnostic de refus (explication d'une erreur de format, heure de rappel, numéro de secours…).

1.1.1.124.x) Attributs du fichier

Ce paramètre contient les paramètres caractéristiques d'un fichier. Il existe trois types d'attributs de fichier : logiques, physiques et historiques.

*** attributs logiques**

Ce sont les caractéristiques du fichier permettant d'y accéder :

- **format d'article :**

  précise le format des articles du fichier. Les valeurs autorisées sont : fixe ou variable.

- **longueur d'article :**

  précise la longueur en octets d'un article du fichier. Il s'agit de la longueur exacte dans le cas d'un fichier de format fixe, de la longueur maximale dans le cas d'un fichier de format variable.

- **organisation du fichier :**

  Ce paramètre décrit l'organisation des données à l'intérieur du fichier et par conséquent le type d'accès à ces données pour le transfert de fichier. Les valeurs possibles sont : séquentielle, relative ou indexée.

- **prise en compte de la signature :**

  Ce paramètre définit si le fichier est accompagné d'un sceau SIT.

- **sceau SIT :**

  Ce paramètre est présent dans les fichiers transmis par la station SIT vers les Centres de Traitements Bancaires.

- label du fichier :

Ce paramètre peut être utilisé pour associer un nom symbolique à un fichier.

- longueur de la clé :

Dans le cas où l'organisation du fichier annoncée est indexée, ce paramètre contient la longueur en octets de la clé d'accès au fichier.

- déplacement de la clé :

Dans le cas où l'organisation du fichier annoncée est indexée, ce paramètre contient le déplacement en octets par rapport au début de l'article, où se trouve la clé d'accès au fichier.

*** attributs physiques**

Ils définissent les caractéristiques physiques du fichier.

- unité de réservation d'espace :

Ce paramètre définit l'unité utilisée pour exprimer la valeur de réservation d'espace. Les unités possibles sont : kilo-octets ou articles.

Note :

On utilisera comme unité :

- le **kilo-octet** pour les fichiers de format d'article variable
- l'**article** ou le **kilo-octet** pour les fichiers de format d'article fixe.

- valeur maximale de réservation d'espace :

Ce paramètre définit la taille que le fichier ne peut pas dépasser.

*** attributs historiques**

Ces paramètres reflètent l'historique du fichier.

- date et heure de création :

- date et heure de dernière extraction :

Date où le dernier transfert a été terminé normalement ou interrompu.

1.1.1.125.y) Identifiant client et identifiant banque

Ce paramètre contient l'identifiant du client ou de la banque pour le compte duquel un transfert de fichier est effectué.

1.1.1.126.z) Contrôle d'accès fichier

Clé d'accès permettant l'identification du client auprès de la banque. Ce mot de passe est échangé lors de la phase de sélection et, par conséquent, est associé à un fichier. Un utilisateur peut modifier ce mot de passe et indique alors le nouveau mot de passe, dans le paramètre contrôle d'accès fichier, à la suite du mot de passe à remplacer.

1.1.1.127.aa) Date et heure du serveur

Ce paramètre contient la date et l'heure telles qu'elles sont connues du serveur au moment de la sélection du fichier.

1.1.1.128.ab) Message libre

Ce paramètre permet de faire transiter un message d'un utilisateur du service à un autre pendant le déroulement d'une des phases du transfert.

1.1.1.129.ac) Article du fichier

Ce paramètre contient les données d'un article du fichier. La correspondance entre article du fichier virtuel et enregistrement du fichier réel est du ressort des implémentations locales.

1.1.1.130.ad) Message

Ce paramètre permet de faire transiter un message d'un utilisateur du service à un autre au moyen d'un service particulier de transfert de message.

3.8 DESCRIPTION DES PROFILS

3.8.1 Profil SIT

Le profil SIT est caractérisé par :

* les unités fonctionnelles :
  Noyau
  Ecriture
  Synchronisation

* des limitations de valeurs pour certains paramètres :

  Option-point de synchronisation : l'intervalle entre points de synchronisation doit être supérieur ou égal à 4 kilo-octets, la fenêtre d'acquittement inférieure ou égale à 16.

  Taille maximale d'entité de données : doit être supérieure ou égale à 800 octets.

* un adressage spécifique :

  les identifiants demandeur et serveur sont des références d'installation constituées par :
  - 1 octet indiquant le type d'installation (valeur symbolique) :
    1 pour CTE,
    2 pour CTR,
    3 pour IE,
    4 pour IR.

  - 2 octets indiquant le numéro d'installation dans le type (valeur numérique).

  Les installations de type CTE et CTR sont abritées par un Centre de Traitement Bancaire, les installations IE ou IR par la station SIT.

  Il est à noter que les notions d'installation émettrice (IE, CTE) ou réceptrice (IR, CTR) doivent s'entendre relativement au flux des opérations bancaires (une remise aller étant par exemple, transmise d'un CTE vers un CTR, via des transferts CTE à IE au moyen de PeSIT, IE à IR sur le réseau primaire, IR à CTR via PeSIT) et non au sens de l'émission ou de la réception de fichier par PeSIT. Autrement dit, n'importe quelle installation CTE, CTR, IE, IR peut à un moment donné être vis-à-vis du protocole PeSIT demandeur/émetteur ou serveur/récepteur.

  Il y a donc quatre types de flux possibles :
  CTE vers IE
  IE vers CTE
  CTR vers IR
  IR vers CTR.

* une convention de nommage des fichiers : le paramètre "Nom de fichier" est une chaîne de 5 caractères ASCII numériques. Les paramètres retenus par la station SIT pour caractériser de manière non ambiguë un fichier SIT sont :
  le numéro de l'installation émettrice ou réceptrice (identifiant demandeur ou serveur),
  le type de fichier,
  le nom du fichier,
  la date de création du fichier.

Le type de fichier caractérise la nature du fichier transféré : remise aller, compte rendu de présentation, bordereau de fin de journée… La liste des types de fichier utilisés sur les réseaux SIT Moyens de paiement et SIT Bourse, classés par flux (de IE vers CTE, de IR vers CTR…) se trouve dans les documents "Analyse fonctionnelle de ASIT" et "Commandes et éditions CTB" SESA 70296 LP 01 210.

La date de création du fichier est une date SIT qui peut être différente de la date système courante.

* Les trois niveaux de priorité prévus par PeSIT (0, 1 ou 2) sont utilisés dans les échanges entre la station SIT et les CTB, sur chacun des flux. Le choix de la priorité pour un transfert donné est du ressort des applications utilisant le service PeSIT.

* La station SIT limite le nombre de transferts entrants à trois par installation, toutes priorités confondues.
  D'autre part, la station limite globalement pour l'ensemble des installations le nombre de transferts entrants par priorité. Ces limites sont fixées au niveau du profil de station par le GSIT.

* Les fichiers transférés entre station SIT et CTB peuvent être de format fixe ou variable et la taille maximale d'article peut être de 4 044 octets par article (ce qui implique une taille maximale d'entité de donnée de 4 050 octets). D'autre part, les articles de taille nulle ne sont pas admis.

* L'utilisation du codage ASCII ou EBCDIC pour le contenu des fichiers est une caractéristique fixée, installation par installation, lors de l'abonnement de l'adhérent au réseau SIT. La station ne gère pas le paramètre Code-données (PI 16), ni en émission ni en réception. Selon la caractéristique de l'installation concernée, la station émettra (ou s'attendra à recevoir) toujours des fichiers dans l'un ou l'autre codage.

* La compression des données n'est pas effectuée au niveau du transfert de fichier.

* Le sceau SIT est un horodatage crypté qui est associé à un fichier. Il n'est transmis que dans le sens station vers CTB. Par conséquent :
  Dans le sens CTB vers station : le paramètre "prise en compte de la signature" vaut 0 (ou est absent) et le paramètre "Sceau SIT" est absent.
  Dans le sens station vers CTB : le paramètre "prise en compte de la signature" vaut 1 et le paramètre "sceau SIT" contient le sceau. Le décryptage du sceau et le contrôle de sa validité sont du ressort de l'application.

3.8.2 Profil Hors-SIT

Le profil Hors-SIT est caractérisé par :

* les unités fonctionnelles obligatoires :

  Noyau
  Ecriture
  Synchronisation

* les unités fonctionnelles facultatives :

  Lecture
  Resynchronisation
  Suspension
  Transfert de message
  Contrôle d'erreur

* Au niveau du protocole, les options d'utilisation de FPDU multi-articles, et de segmentation de FPDU (utilisation de FPDU.DTFDA, FPDU.DTFMA, FPDU.DTFFA) sont autorisées. Il est à noter que l'utilisation de ces options ne fait pas l'objet d'une négociation dynamique au niveau du protocole, mais doit être décidée d'un commun accord au préalable.

* Adressage : les identifiants demandeur et serveur sont des chaînes de un à vingt-quatre caractères ASCII choisies par les utilisateurs du service PeSIT.

* La mise en œuvre par des moniteurs utilisant le protocole PeSIT hors SIT de fonctions de "Store and forward" (reroutage de fichier) est possible. Ce fonctionnement est détaillé dans l'annexe B.

* Nommage des fichiers : le paramètre "nom du fichier" est une chaîne de un à soixante-seize caractères ASCII. Les conventions de nommage des fichiers sont du ressort de l'utilisateur du service PeSIT.

* Le paramètre "type de fichier" doit avoir la valeur 0, sauf convention particulière entre deux moniteurs.

* La compression des données peut être mise en œuvre au niveau du transfert de fichier. Les algorithmes de compression horizontale et verticale sont donnés en annexe A.

* La phase de préconnexion :

  La connexion du protocole PeSIT peut être précédée d'une phase de préconnexion, indépendante du protocole PeSIT. Son rôle est de permettre aux moniteurs de connaître dès la connexion des couches de communications inférieures le type de protocole de transfert utilisé, ainsi que l'identifiant de l'appelant.

  Deux messages sont définis pour répondre à ce besoin :

  Message 1 : ce message est composé de 24 octets :

  - 8 premiers octets : protocole utilisé (PeSIT : 5 caractères cadrés à gauche, suivis de 3 blancs) ;

  - 8 octets suivants : identifiant (de 1 à 8 caractères cadrés à gauche, suivis de blancs) ;

- 8 derniers octets : mot de passe.

Message 2 : message d'acquittement : 4 octets :

ACK0 ou NAK0

Tous ces messages sont codés en EBCDIC. Ils ne sont pas des éléments de protocole PeSIT.

Remarques importantes :

En PeSIT.F', les messages de préconnexion sont échangés dans les premiers paquets de données, suivant l'établissement du circuit virtuel. La phase de préconnexion doit, en PeSIT.F', être considérée comme obligatoire.

En PeSIT.F comme en PeSIT.F'', il n'y a pas de phase de préconnexion. Cependant, un message de 24 octets identique au message 1 défini ci-dessus peut être utilisé :

* en PeSIT.F'' dans les O-DATA de la primitive CONNECT

* en PeSIT.F dans le champ "référence de l'utilisateur" de la primitive S-CONNECT

3.8.3 Profil Hors-SIT sécurisé

Le profil Hors-SIT sécurisé est identique au profil Hors-SIT, mais avec utilisation de l'unité fonctionnelle Sécurisation, obligatoire dans ce profil.

Il est à noter que l'unité fonctionnelle Sécurisation commune aux profils PeSIT Hors-SIT sécurisé et ETEBAC 5 utilise dans chacun de ces deux profils des algorithmes cryptographiques différents, et n'offre par conséquent pas exactement les mêmes fonctions de sécurité.

Le profil PeSIT Hors-SIT sécurisé ne suppose que l'utilisation de l'algorithme de chiffrement/scellement DES (Data Encryption Standard).

Dans ce profil l'unité fonctionnelle Sécurisation permet :
- l'authentification réciproque,
- la confidentialité des données transmises,
- l'intégrité des données transmises.

L'annexe C *: Utilisation des mécanismes de sécurité* précise comment mettre en œuvre les mécanismes de sécurité, dans le cas des profils PeSIT Hors-SIT sécurisé et ETEBAC 5.

Remarque :

L'utilisation de dispositifs de chiffrement pour des transmissions à travers des réseaux publics fait l'objet de législations différentes selon les pays que les utilisateurs du profil PeSIT Hors-SIT sécurisé seront tenus de prendre en compte.

3.8.4 Profil ETEBAC 5

Le profil ETEBAC 5 est caractérisé par :

* les unités fonctionnelles:

  Noyau
  Ecriture
  Lecture
  Synchronisation

* les unités fonctionnelles facultatives :

  Resynchronisation
  Suspension
  Sécurisation

* La description de l'utilisation du protocole PeSIT pour le transport d'ETEBAC 5 est détaillée dans le document : Echanges Télématiques entre les Banques et leurs Clients : standard ETEBAC 5.

* Dans le chapitre protocole du présent document, le format des paramètres et des éléments de protocole utilisés dans le profil ETEBAC 5 est détaillé.

* L'annexe C : *Utilisation des mécanismes de sécurité* précise comment mettre en œuvre les mécanismes de sécurité, dans le cas des profils PeSIT Hors-SIT sécurisé et ETEBAC 5.

3.9 SECURITE DANS LE SERVICE PeSIT

3.9.1 Fonctions assurées

Les fonctions de sécurité requises pour le transfert de fichier sont :

- l'authentification réciproque des partenaires,
- la confidentialité des données transmises (contenu du fichier),
- l'intégrité des données transmises (contenu du fichier),
- la non-répudiation réciproque.

Le service et le protocole PeSIT décrivent les paramètres nécessaires à la mise en œuvre des mécanismes de sécurité, ainsi que la manière de les échanger, mais la mise en œuvre proprement dite (algorithmes, gestion des clés, gestion des accréditations) n'est pas du ressort du transfert de fichier. L'Annexe C du présent document décrit cependant l'utilisation des mécanismes de sécurité dans la mesure où ils ont un impact sur le protocole.

Les paramètres de sécurité définis dans PeSIT ont pour vocation de permettre l'utilisation de différents algorithmes pour réaliser les fonctions décrites ci-dessus. Dans les deux profils utilisant l'unité fonctionnelle Sécurisation, le choix s'est porté sur RSA et DES pour le profil ETEBAC 5. Le profil PeSIT Hors-SIT sécurisé autorise un mode de fonctionnement ne nécessitant que DES. Il est à noter que, dans ce mode, la fonction de non-répudiation réciproque n'est pas possible.

Le paramétrage de la sécurité dans le protocole permet dans une large mesure l'indépendance des différentes fonctions. En profil PeSIT Hors-SIT sécurité chacune des trois fonctions disponibles peut être utilisée indépendamment des autres. En profil ETEBAC 5, la non-répudiation réciproque nécessite l'intégrité.

La mise en œuvre des fonctions de sécurité peut être négociée au niveau de chaque transfert de fichier. Cependant, il relève du choix des implémenteurs, en fonction des contraintes de sécurité, de décider si l'utilisation des fonctions de sécurité sera effectivement renégociée pour chaque transfert, ou si plusieurs transferts consécutifs au cours de la même connexion pourront utiliser les mêmes paramètres de sécurité.

3.9.2 Description des primitives

Les primitives de service PeSIT sont décrites dans le paragraphe 3.6.3 pour tout ce qui ne concerne pas l'unité fonctionnelle Sécurisation. L'objet du présent paragraphe est de compléter leur description par les aspects rattachés à la mise en œuvre de l'unité fonctionnelle Sécurisation.

Ainsi on trouvera dans ce paragraphe pour chaque primitive la fonction qu'elle remplit vis-à-vis de la sécurité et dans la table des paramètres, les paramètres relevant de la sécurité qu'elle pourra comporter en plus de ceux décrit au paragraphe 3.6.3.

a) Le service F.CREATE

*** Fonction**

Cet élément de service permet au demandeur d'indiquer au serveur quelles sont les fonctions de sécurité qui vont être mises en œuvre pour le transfert à venir. Le demandeur indique donc s'il va y avoir :

- authentification réciproque
- scellement
- chiffrement
- signature

Il est à noter qu'il ne peut y avoir signature que s'il y a scellement.

Le serveur indique par une primitive de réponse F.CREATE positive ou négative s'il accepte ou non les fonctions de sécurité requises par le demandeur.

Cet élément de service permet aussi l'échange des accréditations et des éléments d'authentification qui contribuent à l'authentification réciproque des partenaires (l'authentification réciproque complète utilise aussi le service F.OPEN).

Le détail de la gestion des accréditations est donné en Annexe C.

Dans le cas de plusieurs transferts successifs au sein d'une même connexion, l'authentification réciproque peut ne pas être répétée à chaque transfert. De même les accréditations utilisées pour le transport des clés ou des signatures pourront aussi ne pas être transférées à chaque transfert d'une même connexion (sous réserve bien sûr que les partenaires concernés, identifiés par les paramètres "identificateur du client" et "identificateur de la banque" soient inchangés). Par contre l'indication d'utilisation des fonctions chiffrement, scellement et signature ainsi que les éléments de chiffrement et de scellement seront émis préalablement à chaque transfert.

*** Table des paramètres :**

PARAMETRES	F.CREATE demande/indication	F.CREATE réponse/confirmation
(≠a) Type d'authentification (optionnel)	+	
(≠b) Eléments d'authentification (optionnel)	+	+
(≠c) Type de scellement (optionnel)	+	
(≠e) Type de chiffrement (optionnel)	+	
(≠g) Type de signature (optionnel)	+	
(≠j) Accréditation (optionnel)	+	+
(≠m) Deuxième accréditation (optionnel)		+

b) Le service F.SELECT

*** Fonction**

La fonction de cet élément de service vis-à-vis de l'unité fonctionnelle Sécurisation est identique à celle du service F.CREATE, le choix des fonctions de sécurité à mettre en œuvre pour un transfert étant de la responsabilité du Demandeur qu'il s'agisse d'une écriture ou d'une lecture de fichier.

*** Table des paramètres :**

PARAMETRES	F.SELECT demande/indication	F.SELECT réponse/confirmation
(≠a) Type d'authentification (optionnel)	+	
(≠b) Eléments d'authentification (optionnel)	+	+
(≠c) Type de scellement (optionnel)	+	
(≠e) Type de chiffrement (optionnel)	+	
(≠g) Type de signature (optionnel)	+	
(≠j) Accréditation (optionnel)	+	+
(≠m) Deuxième accréditation (optionnel)		+

c) Le service F.OPEN

*** Fonction**

La fonction de cet élément de service est de permettre le troisième échange de l'authentification réciproque dans le cas d'une écriture (les deuxième et troisième échanges dans le cas d'une lecture). Il permet de plus l'échange des éléments de scellement et de chiffrement s'il y a lieu, et des accréditations que ces échanges peuvent nécessiter.

*** Table des paramètres :**

PARAMETRES	F.OPEN demande/indication	F.OPEN réponse/confirmation
(≠b) Eléments d'authentification (optionnel)	+	
(≠d) Eléments de scellement (optionnel)	+	+
(≠f) Eléments de chiffrement (optionnel)	+	+
(≠j) Accréditation (optionnel)	+	
(≠m) Deuxième accréditation (optionnel)	+	

d) Le service F.CHECK

*** Fonction**

Cet élément de service permet la transmission de sceaux partiels si cette option a été retenue. Ces sceaux partiels sont des résultats de calculs intermédiaires du sceau portant sur l'ensemble du fichier (et sur certains paramètres d'identification du fichier).

*** Table des paramètres :**

PARAMETRES	F. CHECK demande/indication	F. CHECK réponse/confirmation
(≠h) Sceau (optionnel)	+	

e) Le service F.DATA.END

*** Fonction**

Cet élément de service permet le transfert du sceau, de la signature et de la deuxième signature du fichier s'il y a lieu.

*** Table des paramètres :**

PARAMETRES	F.DATA.END demande/indication
(≠h) Sceau (optionnel)	+
(≠j) Signature (optionnel)	+
(≠l) Deuxième signature (optionnel)	+

f) Le service F.TRANSFER.END

*** Fonction**

Cet élément de service permet le transfert de l'accusé de réception de la signature.

*** Table des paramètres :**

PARAMETRES	F.TRANSFER.END demande/indication	F.TRANSFER.END réponse/confirmation
(≠k) Accusé de réception de la signature (optionnel)	+	+

g) Le service F.MESSAGE

*** Fonction**

Cet élément de service contient un message qui peut être sécurisé.

Les fonctions qui peuvent être mises en œuvre avec ce service sont :

- scellement
- signature

Le scellement ou la signature porte sur le message contenu dans la primitive F.MESSAGE.

Les mécanismes et les paramètres permettant la mise en œuvre de ces fonctions pour ce service sont identiques à ceux utilisés pour un transfert de fichier.

*** Table des paramètres :**

PARAMETRES	F.MESSAGE demande/indication	F.MESSAGE réponse/confirmation
(≠c) Type de scellement (optionnel)	+	
(≠d) Eléments de scellement (optionnel)	+	
(≠g) Type de signature (optionnel)	+	
(≠h) Sceau (optionnel)	+	
(≠i) Signature (optionnel)	+	+
(≠j) Accréditation (optionnel)	+	+
(≠k) Accusé de réception de la signature (optionnel)	+	+

3.9.3 Description des paramètres

a) Type d'authentification

Ce paramètre indique si une procédure d'authentification va avoir lieu et quels mécanismes seront employés à cet effet.

- authentification ou non
- algorithme utilisé
- mode opératoire

b) Eléments d'authentification

Ce paramètre contient les éléments concourant à l'authentification des partenaires. Selon l'algorithme et le mode opératoire précisé dans le paramètre "authentification", ces éléments d'authentification contiendront, des nombres aléatoires en clair ou cryptés et/ou des clés cryptées.

c) Type de scellement

Ce paramètre indique si le fichier à transférer est scellé. Si oui ce paramètre indique quel algorithme de calcul du sceau est utilisé et selon quel mode opératoire. Le mode opératoire indique s'il y a uniquement un sceau portant sur la totalité du fichier, ou si, en plus du sceau global, des sceaux partiels seront transmis à l'occasion de l'émission des points de synchronisation (sceaux calculés sur les données émises depuis le sceau précédent). Il est aussi précisé d'autre part si les éléments de scellement utilisés sont transmis, en clair ou chiffrés, dans le paramètre "élément de scellement". Pour le calcul du sceau en plus des données du fichier, l'identifiant du fichier (PI 11, PI 12, PI 51, PI 61, PI 62) sera pris en compte : le contenu de ces PI sera traité par l'algorithme de scellement comme si concaténé en tête du premier article du fichier.

- scellement ou non
- algorithme utilisé
- mode opératoire
- transfert des éléments de scellement

d) Eléments de scellement

Ce paramètre contient des éléments (clé, vecteur d'initialisation) permettant à l'algorithme choisi de procéder au calcul du sceau du fichier. Ce paramètre peut être absent si on désire le transférer par un autre canal que le protocole. Il pourra être protégé par un chiffrement tel qu'il est indiqué dans le champ transfert des éléments de scellement du paramètre "type de scellement".

e) Type de chiffrement

Ce paramètre indique si le fichier est transmis chiffré et quel algorithme est utilisé pour ce chiffrement.

Ce paramètre indique aussi le transport ou non des éléments de chiffrement, et leur type de chiffrement, dans le paramètre "éléments de chiffrement" (ces éléments peuvent être transmis par un autre canal que le protocole de transfert de fichier).

- chiffrement ou non

- algorithme utilisé
- mode opératoire
- transfert des éléments de chiffrement

f) Eléments de chiffrement

Ce paramètre contient les éléments (clé, vecteur d'initialisation) permettant à l'algorithme choisi de procéder au chiffrement du fichier. Ce paramètre peut être absent si on désire le transférer par un autre canal que le protocole. Il pourra être protégé par un chiffrement tel qu'il est indiqué dans le champ transfert des éléments de chiffrement du paramètre "type de chiffrement".

g) Type de signature

Ce paramètre indique si le fichier transféré est signé. Le mode de signature retenu reposant sur un chiffrement du sceau, ce paramètre n'a un sens que si l'option scellement a été retenue dans le paramètre "type de scellement". L'algorithme indiqué par le champ "algorithme utilisé" est celui utilisé pour la transformation par chiffrement du sceau en signature. A ce jour, le seul algorithme utilisable est RSA. Le champ "double signature" indique la présence éventuelle d'une double signature. Dans ce cas, la deuxième signature sera un chiffrement RSA du même sceau par une autre clé secrète (dont la clé publique correspondante circulera dans le paramètre "deuxième accréditation").

- type de signature
- algorithme utilisé
- mode opératoire
- double signature

h) Sceau

Ce paramètre contient le résultat du calcul de l'algorithme de scellement. Il peut être un sceau partiel (résultat intermédiaire du calcul du sceau global portant sur les données émises depuis le sceau partiel précédent), ou le sceau portant sur l'ensemble du fichier. Pour le calcul du sceau en plus des données du fichier, l'identifiant du fichier (PI 11, PI 12, PI 51, PI 61, PI 62) sera pris en compte : le contenu de ces PI sera traité par l'algorithme de scellement comme si concaténé en tête du premier article du fichier.

i) Signature

Ce paramètre contient la signature du fichier qui est le chiffrement RSA sous la clé secrète émetteur de la concaténation d'un sceau calculé sur les PI 11, PI 12, PI 51, PI 61, PI 62 et du sceau du fichier (qui lui-même prend en compte ces mêmes paramètres). Ces deux sceaux sont calculés indépendamment mais à partir des mêmes éléments de scellement.

j) Accréditation

Ce paramètre contient l'accréditation d'une entité. Cette accréditation comporte l'identification de l'entité, le numéro de série du dispositif sur lequel est préservée l'accréditation (numéro de série de carte à mémoire), la clé publique RSA de l'identité, le numéro de bi-clé RSA du gestionnaire utilisé pour signer l'accréditation, la signature de l'accréditation (chiffrement RSA sous la clé secrète du gestionnaire d'une empreinte – résultat d'un hash-coding – des champs précédents de l'accréditation).

k) Accusé de réception de signature

Ce paramètre contient un accusé de réception sécurisé de la signature. Cet accusé de réception est le chiffrement RSA sous la clé secrète de son émetteur de la concaténation des sceaux reçus dans la signature, des dates et heures d'acquittement et d'un champ rendant compte des contrôles de sécurité effectués.

l) Deuxième signature

Ce paramètre contient la deuxième signature du fichier qui est le résultat d'un calcul identique à celui du paramètre "signature" mais en utilisant un chiffrement RSA sous une clé secrète émetteur dont la clé publique associée est contenue dans le paramètre "deuxième accréditation".

m) Deuxième accréditation

Ce paramètre contient une deuxième accréditation d'une entité lorsque au cours d'un même échange il est nécessaire de transmettre plusieurs accréditations (authentification et signature ou signataires multiples). La structure de ce paramètre est identique à celle décrite ci-dessus pour le paramètre accréditation.

3.10 SEQUENCES DE PRIMITIVES TYPE

3.10.1 Séquences normales

La logique de progression normale des services est illustrée par les diagrammes d'état suivants (fig. 1, 2, 3, 4 et 5). Les contraintes d'apparition des primitives sont illustrées par les tableaux suivants d'enchaînement des séquences du demandeur et du serveur. Ces tableaux précisent les cas valides d'apparition d'une primitive.

FIGURE 1 : DIAGRAMME D'ETATS SIMPLIFIE

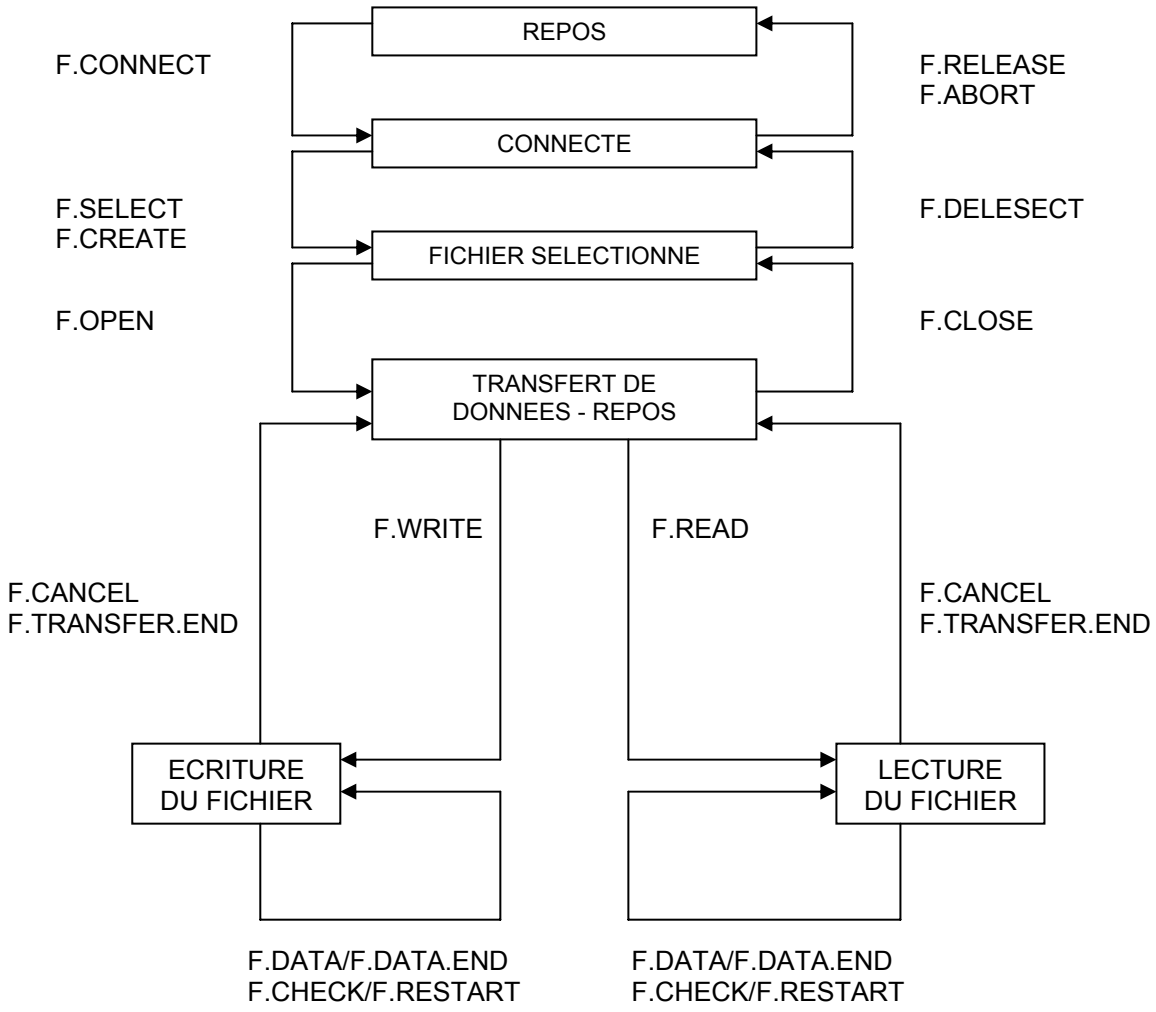
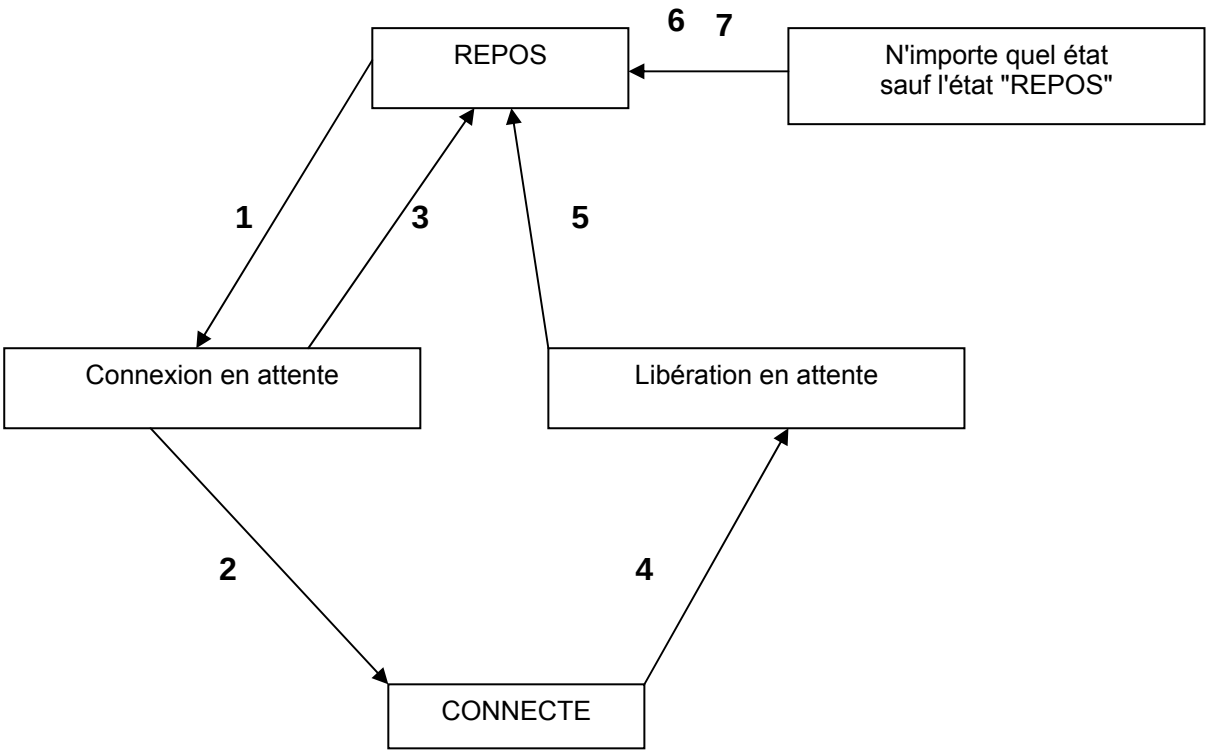


FIGURE 2 : DIAGRAMME D'ETAT
PHASE CONNEXION



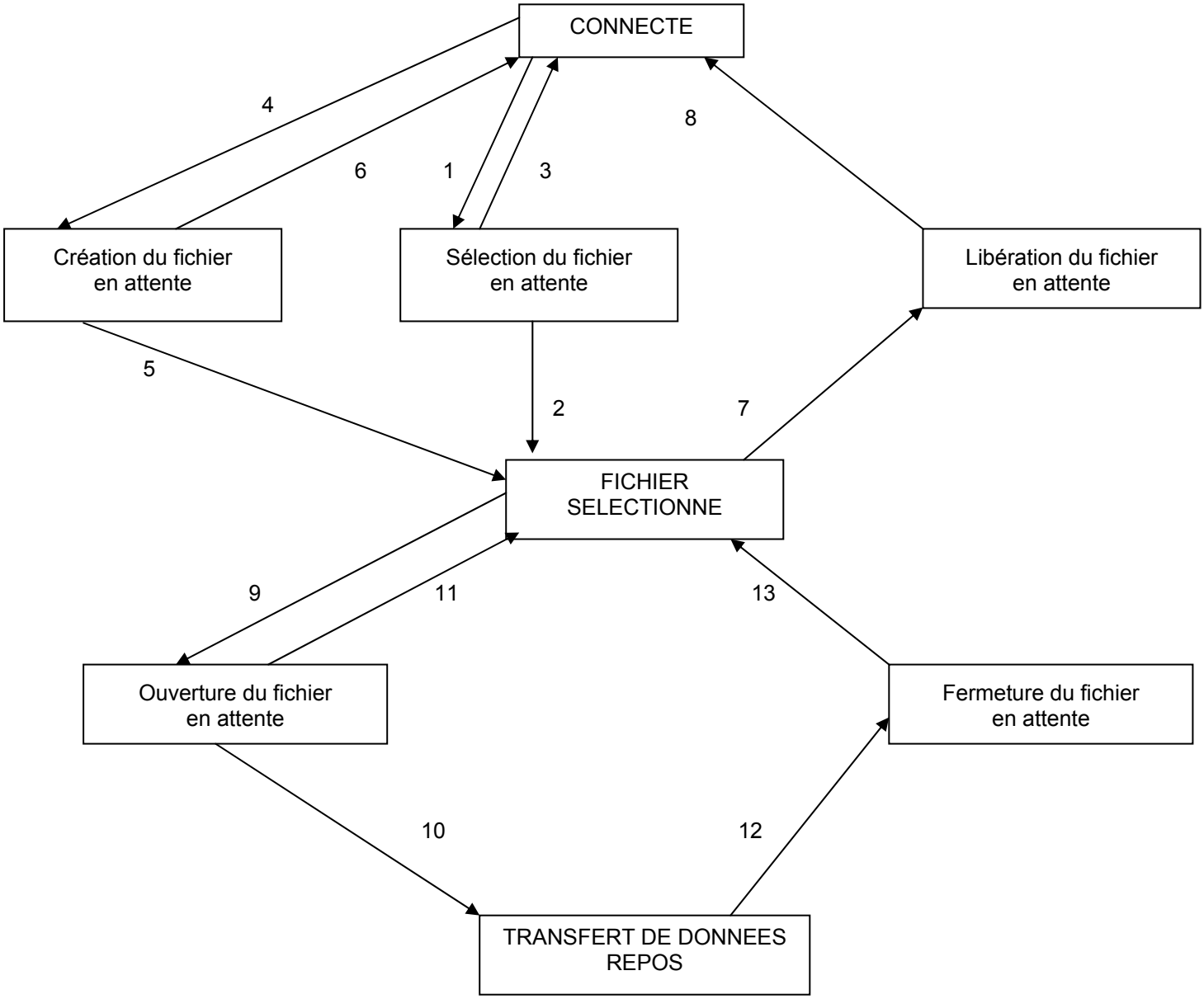
Transitions du demandeur

1 F.CONNECT, demande
2 F.CONNECT, confirmation (positive)
3 F.CONNECT, confirmation (négative)
4 F.RELEASE, demande
5 F.RELEASE, confirmation
6 F.ABORT, demande
7 F.ABORT, indication

Transitions du serveur

1 F.CONNECT, indication
2 F.CONNECT, réponse (positive)
3 F.CONNECT, réponse (négative)
4 F.RELEASE, indication
5 F.RELEASE, réponse
6 F.ABORT, demande
7 F.ABORT, indication

FIGURE 3 : DIAGRAMME D'ETAT
PHASES SELECTION ET OUVERTURE DU FICHIER



TRANSITIONS (figure 3)	DU DEMANDEUR	DU SERVEUR
1	F.SELECT, demande	F.SELECT, indication
2	F.SELECT, confirmation (+Ve)	F.SELECT, réponse (+Ve)
3	F.SELECT, confirmation (-Ve)	F.SELECT, réponse (-Ve)
4	F.CREATE, demande	F.CREATE, indication
5	F.CREATE, confirmation (+Ve)	F.CREATE, réponse (+Ve)
6	F.CREATE, confirmation (-Ve)	F.CREATE, réponse (-Ve)
7	F.DESELECT, demande	F.DESELECT, indication
8	F.DESELECT, confirmation	F.DESELECT, réponse
9	F.OPEN, demande	F.OPEN, indication
10	F.OPEN, confirmation(+Ve)	F.OPEN, réponse(+Ve)
11	F.OPEN, confirmation(-Ve)	F.OPEN, réponse(-Ve)
12	F.CLOSE, demande	F.CLOSE, indication
13	F.CLOSE, confirmation	F.CLOSE, réponse

Note : +Ve = positive
- Ve = négative

FIGURE 4 : DIAGRAMME D'ETAT DU DEMANDEUR
PHASE TRANSFERT DE DONNEES

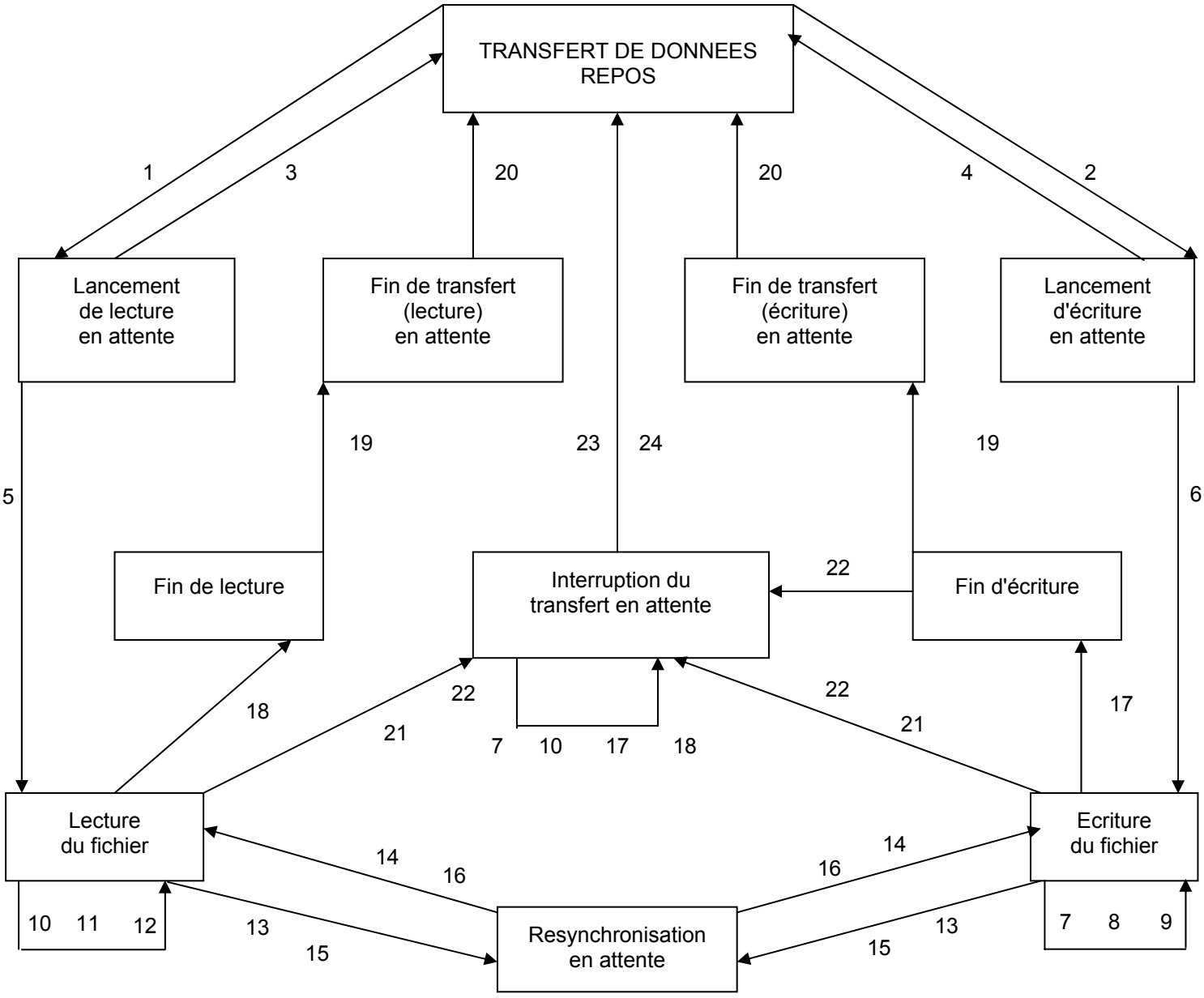
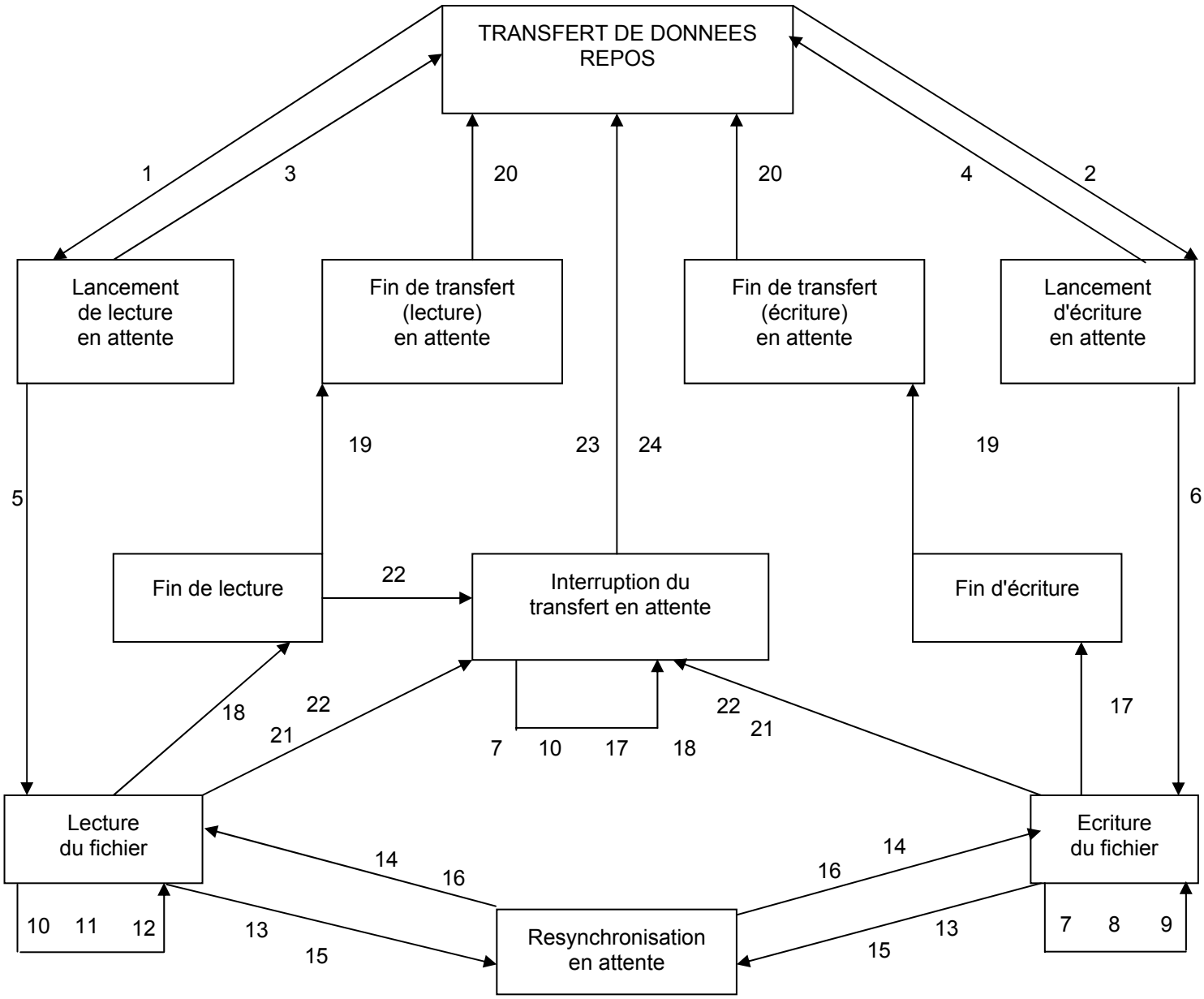


FIGURE 5 : DIAGRAMME D'ETAT DU SERVEUR
PHASE TRANSFERT DE DONNEES



	TRANSITIONS DU DEMANDEUR (figure 4)	TRANSITIONS DU SERVEUR (figure 5)
1	F.READ, D	F.READ, I
2	F.WRITE, D	F.WRITE, I
3	F.READ, C (-Ve)	F.READ, R (- Ve)
4	F.WRITE, C (-Ve)	F.WRITE, R (-Ve)
5	F.READ, C (+Ve)	F.READ, R (+Ve)
6	F.WRITE, C (+Ve)	F.WRITE, C (+Ve)
7	F.DATA, D	F.DATA, I
8	F.CHECK, D	F.CHECK, I
9	F.CHECK, C	F.CHECK, R
10	F.DATA, I	F.DATA, D
11	F.CHECK, I	F.CHECK, D
12	F.CHECK, R	F.CHECK, C
13	F.RESTART, D	F.RESTART, I
14	F.RESTART, C	F.RESTART, R
15	F.RESTART, I	F.RESTART, D
16	F.RESTART, R	F.RESTART, C
17	F.DATA.END, D	F.DATA.END, I
18	F.DATA.END, I	F.DATA.END, D
19	F.TRANSFER.END, D	F.TRANSFER.END, I
20	F.TRANSFER.END, C	F.TRANSFER.END, R
21	F.CANCEL, D	F.CANCEL, D
22	F.CANCEL, I	F.CANCEL, I
23	F.CANCEL, R	F.CANCEL, R
24	F.CANCEL, C	F.CANCEL, C

TABLEAU 1 : SEQUENCES DE PRIMITIVES DE SERVICE VALIDES DU *DEMANDEUR*

SUIVIE DE / PRECEDEE PAR	F.CONNECT, D	F.SELECT, D	F.OPEN, D	F.WRITE, D	F.DATA, D	F.DATA.END, D	F.TRANSFER.END, D	F.CLOSE, D	F.DESELECT, D	F.CANCEL, D	F.CANCEL, R	F.CREATE, D	F.CHECK, D	F.CHECK, R	F.RESTART, D	F.RESTART, R	F.READ, D	F.RELEASE, D	F.ABORT, D
F.CONNECT, D																			O
F.CONNECT, C		*										*						*	*
F.SELECT, D																			O
F.SELECT, C (+)			*						*										*
F.SELECT, C (-)		*										*						*	*
F.CREATE, D																			O
F.CREATE, C (+)			*						*										*
F.CREATE, C (-)		*										*						*	*
F.OPEN, D																			O
F.OPEN, C (+)				*				*									*		*
F.OPEN, C (-)			*						*										*
F.WRITE, D																			O
F.WRITE, C (+)					*	*				*					*				*
F.WRITE, C (-)				*				*											*
F.DATA, D					*	*				*			*		*				*
F.DATA.END, D							*								*				*
F.TRANSFER.END, D																			O
F.TRANSFER.END, C								*											*
F.READ, D																			O
F.READ, C (+)										*					*				*
F.READ, C (-)								*									*		*
F.DATA, I										*				*	*				*

Légende : D : demande - I : indication - R : réponse - C : confirmation - * : possible - O : possible (mais confirmation de la primitive précédente est en attente)

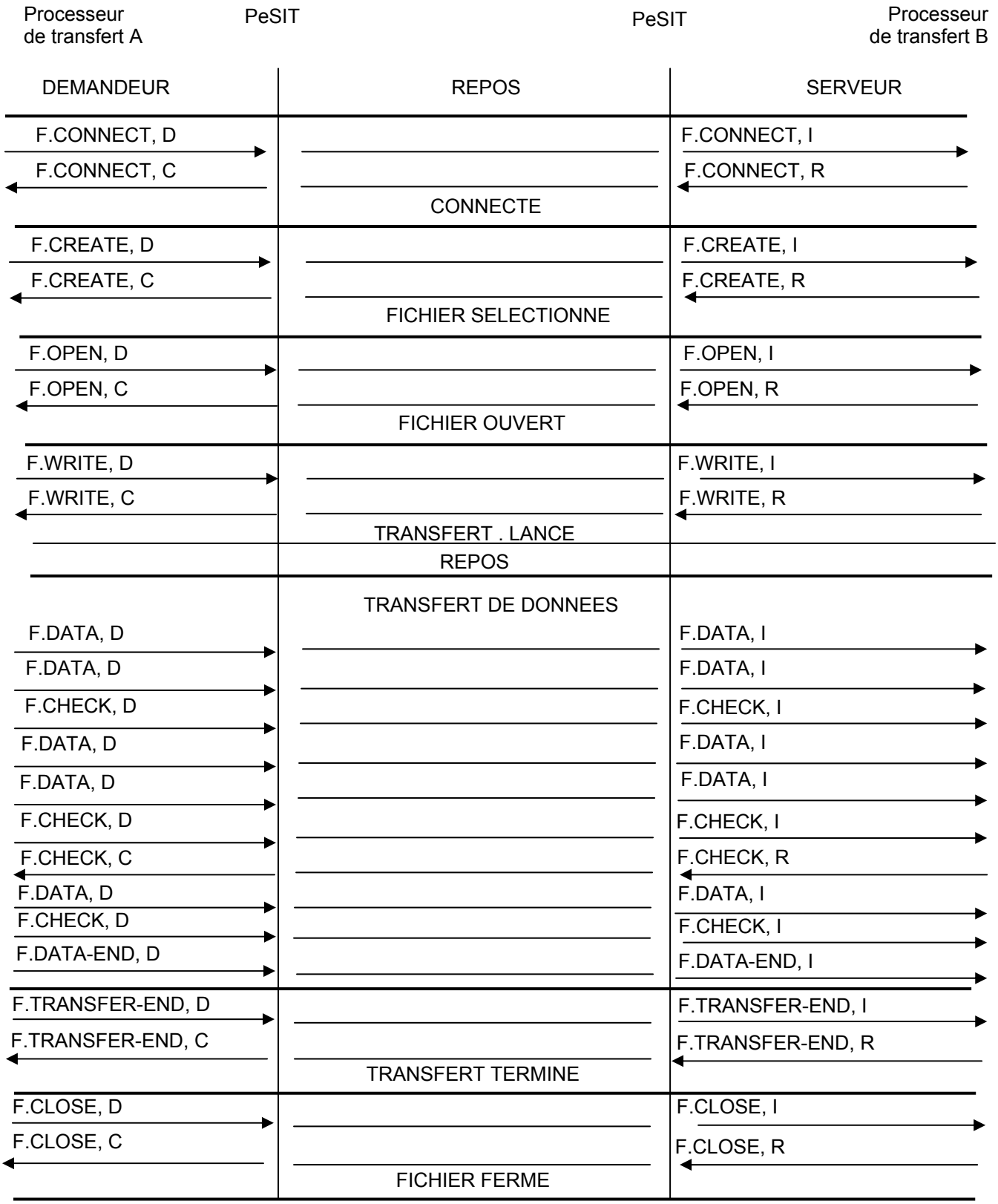
SUIVIE DE / PRECEDEE PAR	F.CONNECT D	F.SELECT D	F.OPEN D	F.WRITE D	F.DATA D	F.DATA.END D	F.TRANSFER.END D	F.CLOSE D	F.DESELECT D	F.CANCEL D	F.CANCEL R	F.CREATE D	F.CHECK D	F.CHECK R	F.RESTART D	F.RESTART R	F.READ D	F.RELEASE D	F.ABORT D
F.DATA.END, I														*	*				*
F.CANCEL, D																			O
F.CANCEL, I											*								*
F.CANCEL, R								*											*
F.CANCEL, C								*											*
F.CLOSE, D																			O
F.CLOSE, C			*						*										*
F.DESELECT, D																			O
F.DESELECT, C		*										*						*	*
F.RELEASE, D																			O
F.RELEASE, C	*																		
F.ABORT, D	*																		
F.ABORT, I	*																		
F.CHECK, D					*	*				*					*				*
F.CHECK, I										*				*	*				*
F.CHECK, R										*				*	*				*
F.CHECK, C					*	*				*					*				*
F.RESTART, D										O									O
F.RESTART, I										*						*			*
F.RESTART, R					*	*				*			*		*				*
F.RESTART, C					*	*				*			*		*				*

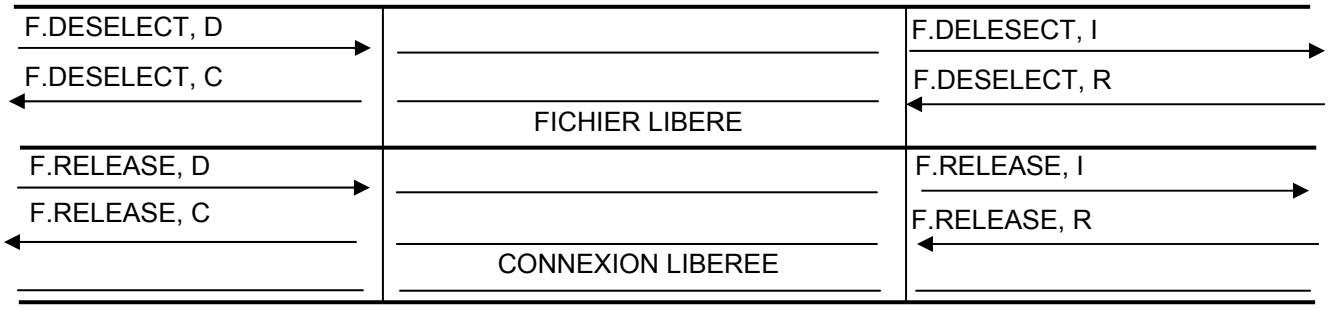
TABLEAU 2 : SEQUENCES DE PRIMITIVES DE SERVICE VALIDES DU *SERVEUR*

SUIVIE DE / PRECEDEE PAR	F.CONNECT R	F.SELECT R	F.OPEN R	F.WRITE R	F.TRANSFER.END R	F.CLOSE R	F.DESELECT R	F.CANCEL D	F.CANCEL R	F.CREATE R	F.READ R	F.DATA D	F.DATA.END D	F.CHECK D	F.CHECK R	F.RESTART D	F.RESTART R	R.RELEASE R	F.ABORT D
F.RELEASE, I																		*	*
F.RELEASE, R																			
F.ABORT, D																			
F.ABORT, I																			
F.CHECK, D								*				*	*			*			*
F.CHECK, I								*							*	*			*
F.CHECK, R								*							*	*			*
F.CHECK, C								*				*	*	*		*			*
F.RESTART, D								O											O
F.RESTART, I								*									*		*
F.RESTART, R								*				*	*	*		*			*
F.RESTART, C								*				*	*	*		*			*
F.CONNECT, I	*																		*
F.CONNECT, R																			*
F.SELECT, I		*																	*
F.SELECT, R (+)																			*
F.SELECT, R (-)																			*
F.CREATE, I										*									*
F.CREATE, R (+)																			*
F.CREATE, R (-)																			*
F.OPEN, I			*																*
F.OPEN, R (+)																			*
F.OPEN, R (-)																			*
F.WRITE, I				*															*
F.WRITE, R (+)								*									*		*
F.WRITE, R (-)																			*
F.DATA, I								*							*	*			*
F.DATA.END I															*	*			*

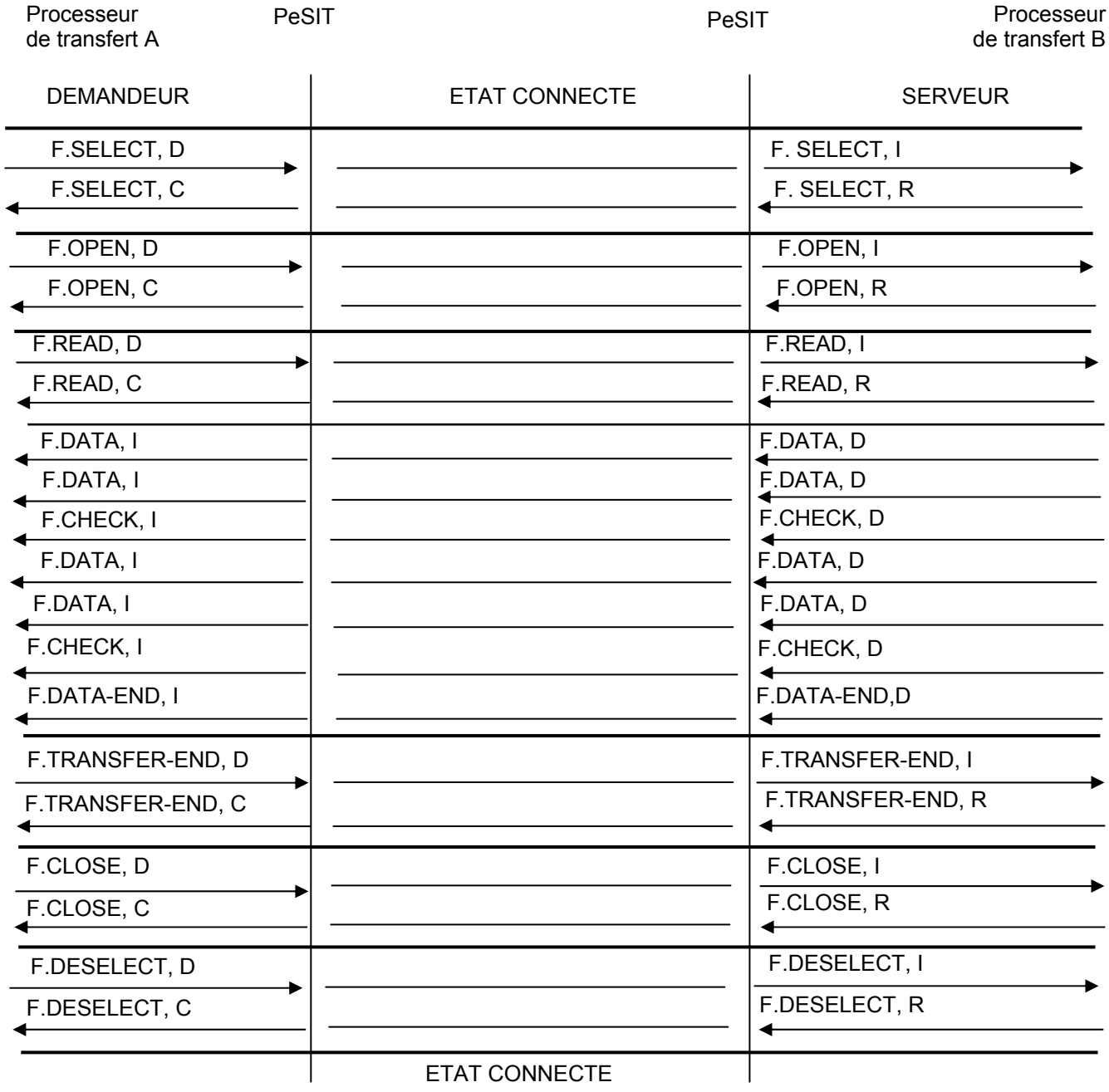
SUIVIE DE / PRECEDEE PAR	F.CONNECT R	F.SELECT R	F.OPEN R	F.WRITE R	F.TRANSFER.END R	F.CLOSE R	F.DESELECT R	F.CANCEL D	F.CANCEL R	F.CREATE R	F.READ R	F.DATA D	F.DATA.END D	F.CHECK D	F.CHECK R	F.RESTART D	F.RESTART R	R.RELEASE R	F.ABORT D
F.TRANSFERT.END, I					*														*
F.TRANSFERT.END, R																			*
F.READ, I											*								*
F.READ, R (+)								*				*	*	*		*			*
F.READ, R (-)																			*
F.DATA, D								*				*	*	*		*			*
F.DATA-END, D																*			*
F.CANCEL, D																			*
F.CANCEL, I									*										*
F.CANCEL, R																			*
F.CANCEL, C																			*
F.CLOSE, I						*													*
F.CLOSE, R																			*
F.DESELECT, I							*												*
F.DESELECT, R																			*

3.10.2 Enchaînement normal des primitives dans le cas d'écriture

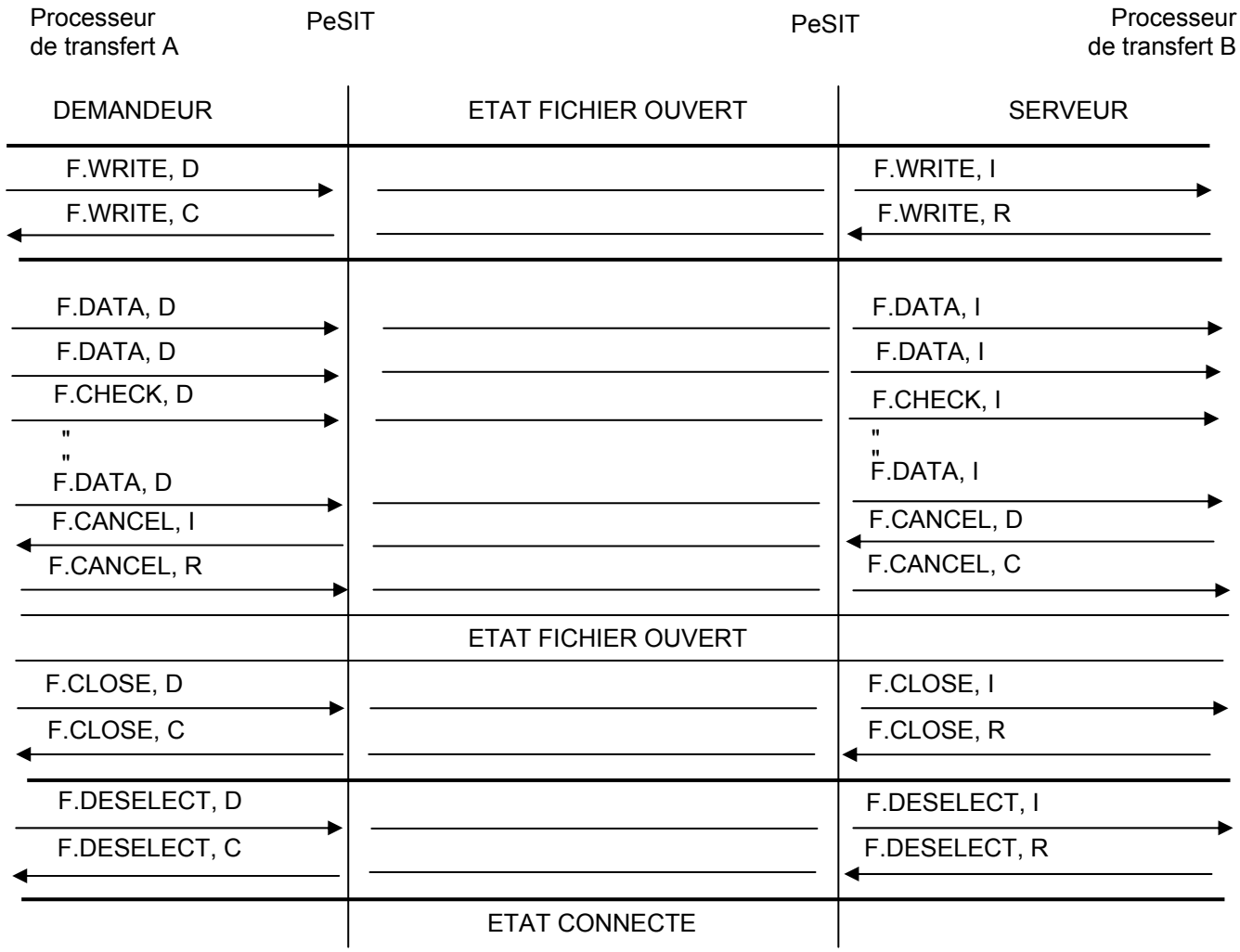



F.DESELECT, D
F.DESELECT, C
FICHIER LIBERE
F.DELESECT, I
F.DESELECT, R
F.RELEASE, D
F.RELEASE, C
CONNEXION LIBEREE
F.RELEASE, I
F.RELEASE, R

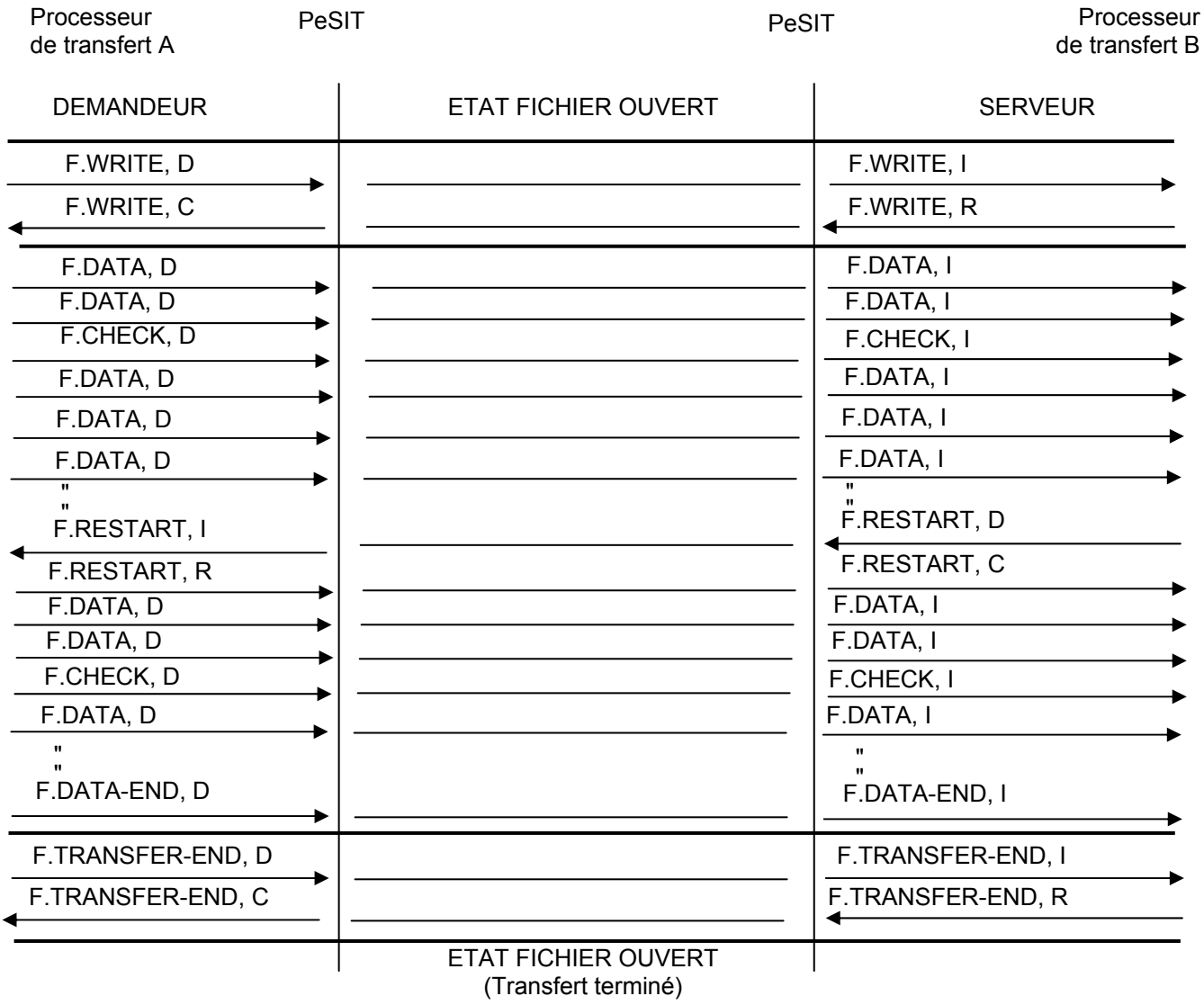
3.10.3 Enchaînement normal des primitives dans le cas de lecture



3.10.4 Enchaînement avec interruption du transfert de fichier



3.10.5 Enchaînement avec resynchronisation



CHAPITRE 4 DESCRIPTION DU PROTOCOLE PeSIT

4.1 INTRODUCTION

Le protocole PeSIT est conçu comme une machine abstraite où des messages (FPDU : File Transfer Protocol Data Unit) sont échangés entre deux entités PeSIT homologues : le demandeur et le serveur. Ces messages contiennent une en-tête protocolaire, une zone variable contenant des informations de gestion du protocole PeSIT (c'est-à-dire des paramètres) et des données du fichier. La zone variable et les données du fichier peuvent être absentes de certains messages.

La description complète du protocole repose sur les éléments suivants :

- la spécification des procédures du transfert de messages d'une entité PeSIT à son homologue,
- la spécification et le codage des unités de données de protocole de PeSIT (FPDU).

Les procédures sont définies en termes :

- d'interactions entre entités PeSIT homologues, par échange de messages FPDU,
- d'interactions entre une entité PeSIT et l'utilisateur du service PeSIT du même système, par échange de primitives du service PeSIT,
- d'interactions entre une entité PeSIT et le fournisseur du service "Système de communication" par échange de primitives de service "Système de communication".

La description du protocole étant globale pour PeSIT.F, PeSIT.F', PeSIT.F" et PeSIT.F"', la différence entre eux sera signalée à chaque fois qu'il est nécessaire.

4.2 CORRESPONDANCE ENTRE SERVICE ET PROTOCOLE

La couche de protocole PeSIT communique avec l'utilisateur grâce aux primitives qui ont été définies dans le chapitre précédent (Service PeSIT). Les primitives ont pour effet ou sont le résultat de l'échange de messages FPDU entre deux entités PeSIT homologues sur une connexion "Système de communication".

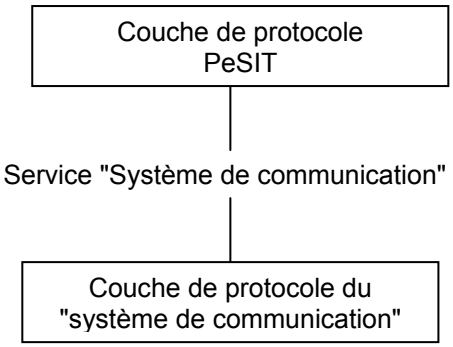
Le tableau suivant donne la liste des FPDU et leurs correspondances avec les primitives de service PeSIT. Les abréviations utilisées ont pour signification :

- DI : demande et indication,
- RC : réponse et confirmation.

SERVICE	PRIMITIVES	FPDU ASSOCIE	DEFINITION
Connexion PeSIT	F.CONNECT,DI	FPDU.CONNECT	Demande de connexion
	F.CONNECT,RC positive	FPDU.ACONNECT	Confirmation de connexion
	F.CONNECT,RC négative	FPDU.RCONNECT	Refus de connexion
Sélection et libération de fichier	F.CREATE,DI	FPDU.CREATE	Création de fichier
	F.CREATE,RC	FPDU.ACK(CREATE)	Confirmation de création
	F.SELECT,DI	FPDU.SELECT	Sélection de fichier
	F.SELECT,RC	FPDU.ACK(SELECT)	Confirmation de sélection
	F.DESELECT,DI	FPDU.DESELECT	Libération de fichier
	F.DESELECT,RC	FPDU.ACK(DESELECT)	Confirmation de libération
	F.MESSAGE,DI	FPDU.MSG	Envoi de message
		FPDU.MSGDM	Début de message
		FPDU.MSGMM	Milieu de message
		FPDU.MSGFM	Fin de message
	F.MESSAGE,RC	FPDU.ACK(MSG)	Confirmation de message
Ouverture et Fermeture de fichier	F.OPEN,DI	FPDU.ORF	Ouverture de fichier
	F.OPEN,RC	FPDU.ACK(ORF)	Confirmation d'ouverture
	F.CLOSE,DI	FPDU.CRF	Fermeture de fichier
	F.CLOSE,RC	FPDU.ACK(CRF)	Confirmation de fermeture
Début et fin de transfert	F.WRITE,DI	FPDU.WRITE	Ecriture de fichier
	F.WRITE,RC	FPDU.ACK(WRITE)	Confirmation d'écriture
	F.READ,DI	FPDU.READ	Lecture de fichier
	F.READ,RC	FPDU.ACK(READ)	Confirmation de lecture
	F.TRANSFER.END,DI	FPDU.TRANS.END	Fin de transfert
	F.TRANSFER.END,RC	FPDU.ACK(TRANS.END)	Confirmation de fin de transfert
Transfert de données	F.DATA,DI	FPDU.DTF	Emission de données
		FPDU.DTFDA	Début d'article
		FPDU.DTFMA	Milieu d'article
		FPDU.DTFFA	Fin d'article
	F.DATA-END,DI	FPDU.DTF.END	Fin d'émission de données
	F.CHECK,DI	FPDU.SYN	Point de synchronisation
	F.CHECK,RC	FPDU.ACK(SYN)	Confirmation de point de synchronisation
	F.RESTART,DI	FPDU.RESYN	Resynchronisation
	F.RESTART,RC	FPDU.ACK(RESYN)	Confirmation de resynchronisation
Interruption de transfert	F.CANCEL,DI	FPDU.IDT	Interruption de transfert
	F.CANCEL,RC	FPDU.ACK(IDT)	Confirmation d'interruption
Terminaison de connexion	F.RELEASE,DI	FPDU.RELEASE	Demande de déconnexion
	F.RELEASE,RC	FPDU.RELCONF	Confirmation de déconnexion
	F.ABORT,DI	FPDU.ABORT	Arrêt brutal de connexion

4.3 UTILISATION DU SERVICE "SYSTEME DE COMMUNICATION"

Ce paragraphe définit la manière dont sont utilisées les primitives du service "Système de communication" par le PeSIT.



Le "Système de communication" peut être de quatre types :

- couche Session ISO, utilisée sur un réseau à commutation, de paquet,
- couche réseau (X25 ou autre type de réseau),
- couche NETEX (dans le cas de l'utilisation de Hyperchannel),
- couche Session ISO utilisée sur une réseau local ISO 8802-3.

4.3.1 Utilisation du service session par PeSIT.F

Le protocole de transfert de fichier PeSIT.F s'appuie directement sur une couche Session ISO.

De manière générale, la couche Session est constituée d'une entité fonctionnelle noyau, que toute implémentation doit offrir, et de onze unités fonctionnelles qui peuvent ne pas être supportées par une entité Session et dont l'usage entre deux entités Session se négocie lors de l'établissement de la connexion.

Dans le cas du protocole de transfert de fichier PeSIT.F, l'utilisation des unités fonctionnelles suivantes est nécessaire :

- noyau,
- transmission semi-duplex (et le jeton de données associé),
- transfert de données typées.

Le tableau suivant précise quels sont les services de la Session utilisés.

SERVICE	FONCTION	UNITE FONCTIONNELLE
S-CONNECT	Demande de connexion Session	Noyau
S-ACCEPT	Acceptation de connexion Session	Noyau
S-RELEASE	Fin de Session	Noyau
S-REFUSE	Refus de connexion Session	Noyau
S-U-ABORT	Coupure par l'utilisateur	Noyau
S-P-ABORT	Coupure par le fournisseur	Noyau
S-DATA	Transfert de données normales	Noyau
S-TOKEN-GIVE	Cession de jeton (de données)	Semi-duplex
S-TYPED-DATA	Transfert de données typées	Transfert de données typées

Les messages PeSIT.F (FPDU) sont transmis soit en utilisant l'élément de service Session S-DATA, soit comme données utilisateur des autres éléments de service utilisés.

Le tableau suivant donne les éléments de service Session utilisés pour transmettre les messages PeSIT.F (FPDU) :

MESSAGES PeSIT.F	TRANSPORTE PAR	COMMENTAIRE
FPDU.CONNECT	S.CONNECT	Dans le champ utilisateur (512 octets) du service S-CONNECT
FPDU.ACONNECT	S-ACCEPT	Dans le champ utilisateur (512 octets) du service S-ACCEPT
FPDU.RCONNECT	S-REFUSE	Dans le champ utilisateur (513 octets) du service S-REFUSE
FPDU.RELEASE	S-RELEASE	Dans le champ utilisateur (512 octets) du service S-RELEASE
FPDU.RELCONF	S-RELEASE	Dans le champ utilisateur (512 octets) du service S-RELEASE
FPDU.ABORT	S-ABORT	Dans le champ utilisateur (9 octets) du service S-ABORT
FPDU.CREATE	S-DATA	Passage du jeton
FPDU.ACK(CREATE)	S-DATA	Passage du jeton
FPDU.SELECT	S-DATA	Passage du jeton
FPDU.ACK(SELECT)	S-DATA	Passage du jeton
FPDU.DESELECT	S-DATA	Passage du jeton
FPDU.ACK(DESELECT)	S-DATA	Passage du jeton
FPDU.MSG	S-DATA	Passage du jeton
FPDU.MSGDM	S-DATA	Sans passage du jeton
FPDU.MSGMM	S-DATA	Sans passage du jeton
FPDU.MSGFM	S-DATA	Passage du jeton
FPDU.ACK(MSG)	S-DATA	Passage du jeton
FPDU.ORF	S-DATA	Passage du jeton
FPDU.ACK(ORF)	S-DATA	Passage du jeton
FPDU.CRF	S-DATA	Passage du jeton
FPDU.ACK(CRF)	S-DATA	Passage du jeton
FPDU.READ	S-DATA	Passage du jeton
FPDU.ACK(READ)	S-DATA	Sans passage du jeton
FPDU.WRITE	S-DATA	Passage du jeton
FPDU.ACK(WRITE)	S-DATA	Passage du jeton
FPDU.TRANS.END	S-DATA	Passage du jeton
FPDU.ACK(TRANS.END)	S-DATA	Passage du jeton
FPDU.DTF	S-DATA	Sans passage du jeton
FPDU.DTFDA	S-DATA	Sans passage du jeton
FPDU.DTFMA	S-DATA	Sans passage du jeton
FPDU.DTFFA	S-DATA	Sans passage du jeton
FPDU.DTF.END	S-DATA	Sans passage du jeton si demandeur
FPDU.SYN	S-DATA	Sans passage du jeton
FPDU.ACK(SYN)	S-TYPED-DATA	Sans passage du jeton
FPDU.RESYN	S-DATA ou S-TYPED-DATA	Passage du jeton Si ne possède pas le jeton
FPDU.ACK(RESYN)	S-DATA	Sans passage du jeton si émetteur
FPDU.IDT	S-DATA ou S-TYPED-DATA	Passage du jeton Si ne possède pas le jeton
FPDU.ACK(IDT)	S.DATA	Sans passage du jeton si demandeur

a) Affectation d'une connexion de Session à une connexion PeSIT.F.

Une connexion de PeSIT.F est affectée à une connexion de session créée à cet effet. L'établissement d'une connexion session se fait via la primitive S-CONNECT qui transporte la FDPU.CONNECT dans la partie données de l'utilisateur. Cette primitive est toujours émise par le demandeur. C'est dans cette phase que sont négociées, d'une part les options du protocole PeSIT (utilisation de F-CHECK et F-RESTART), d'autre part les unités fonctionnelles de session à utiliser durant la session.

L'acceptation de la connexion se fait au travers de la primitive S-ACCEPT, qui est toujours émise par le serveur.

Si la connexion ne peut être établie, son refus est indiqué par la primitive S-REFUSE. La primitive S-REFUSE ne possédant pas de paramètre données de l'utilisateur, la FPDU.RCONNECT sera placée dans le champ code raison qui peut comporter, en plus de l'octet code raison, jusqu'à 512 octets de données de l'utilisateur.

b) Fin de la connexion session

La session peut prendre fin de trois façons :

- Fin normale provoquée uniquement par le demandeur (limitation de la connexion). Cette fin utilise le service S-RELEASE.

- Fin anormale provoquée par une des deux entités PeSIT.F. Cette fin utilise le service S-U-ABORT. Le champ données de l'utilisateur contient la FPDU.ABORT.

- Déconnexion spontanée par le protocole session. Cette fin utilise le service S-P-ABORT.

c) Transmission de données normales

La quasi-totalité des unités de protocole de PeSIT.F (FPDU) est transmise par l'intermédiaire du service S-DATA.

Le dialogue est de type bidirectionnel à l'alternat (semi-duplex), le droit de parole étant changé, indifféremment par le service S-TOKEN-GIVE.

d) Synchronisation / resynchronisation

Dans le cas où l'option synchronisation a été retenue, la pose des points de synchronisation se fait par l'émission de FPDU.SYN, dans le champ information de l'utilisateur de l'élément de service S-DATA. La confirmation de point de synchronisation se fait par l'émission de FPDU.ACK(SYN), dans le champ information de l'élément de service S-TYPED-DATA (émission de données typées).

Dans le cas où l'option resynchronisation a été retenue, la demande de resynchronisation se fait par l'émission de FPDU.RESYN, dans le champ information de l'utilisateur de l'élément de service S-DATA. La confirmation de resynchronisation se fait par l'émission de FPDU.ACK(RESYN), dans le champ information de l'utilisateur de l'élément de service S-DATA.

Dans le cas où l'entité PeSIT ne possède pas le jeton de données, l'émission de la FPDU.RESYN, se fait par emploi de l'élément de service S-TYPED-DATA (émission de données typées).

e) Transmission de données typées

Le service de données typées, S-TYPED-DATA, est utilisé par l'entité PeSIT.F qui ne possède pas le tour de parole pour :

- interrompre le transfert de données : émission de la FPDU.IDT (élément de service F-CANCEL),
- confirmer la pose des points de synchronisation : émission de la FPDU.ACK(SYN),
- demander la resynchronisation des échanges : émission de la FPDU.RESYN.

f) Restriction imposées par les jetons sur l'utilisation du service Session

Les règles d'emploi des données typées et du passage du jeton sont définies dans le tableau précédent. Les cas de collisions sont résolus par l'application des règles suivantes :

1. règles du paragraphe 4.8.3.

2. une entité qui possède le jeton et qui est en attente d'une FPDU.ACK(IDT) ou d'une FPDU.ACK(RESYN), doit le céder à l'autre entité avec laquelle elle dialogue, par un passage de jeton sans émission de FPDU.

Cette dernière règle concerne les cas où une entité a émis une FPDU.IDT ou une FPDU.RESYN et qu'elle reçoit par la suite une FPDU avec le jeton, qu'elle doit ignorer en application de la règle de priorité.

Les schémas donnés ci-après illustrent cette règle et son utilisation dans un certain nombre de cas représentatifs.

Conventions :

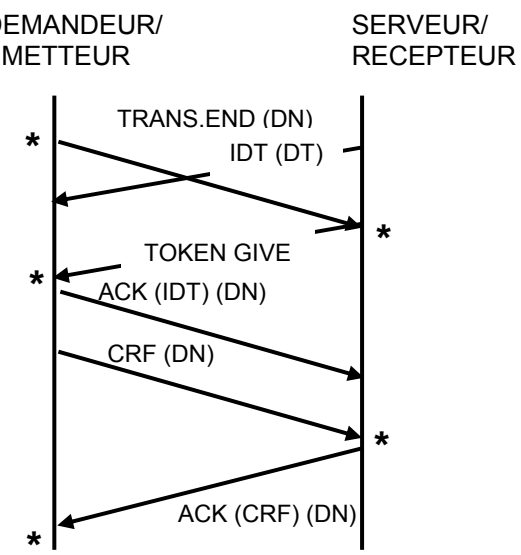
DN : données normales

DT : données typées

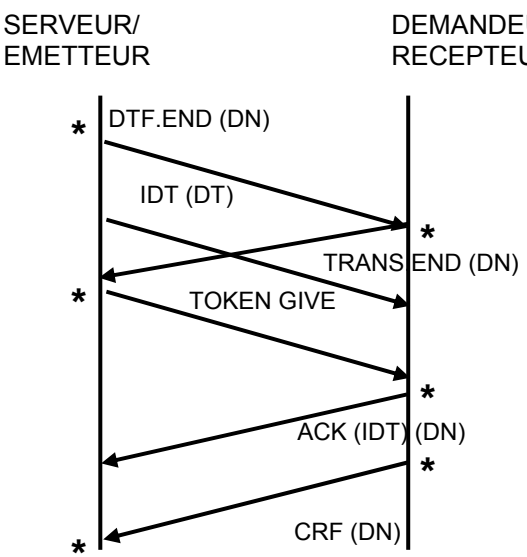
* : jeton

- **CROISEMENT TRANS.END IDT**

(ECRITURE)

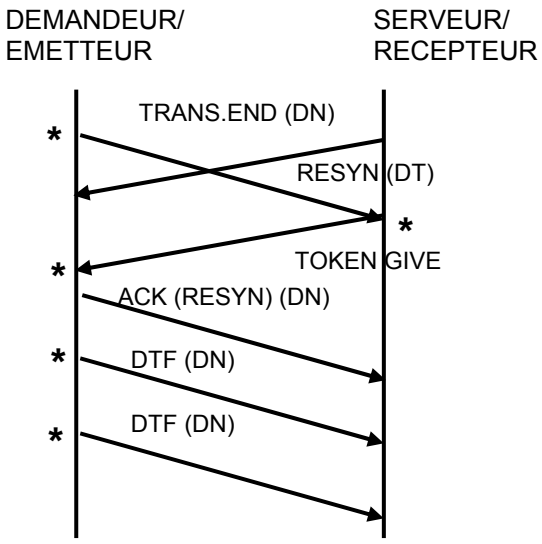


(LECTURE)

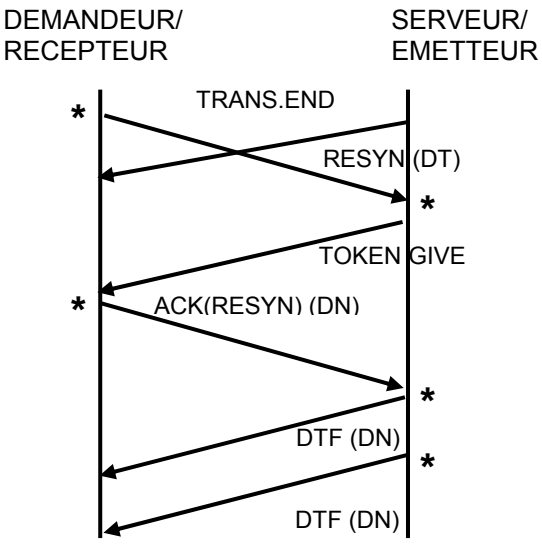


- **CROISEMENT TRANS.END RESYN**

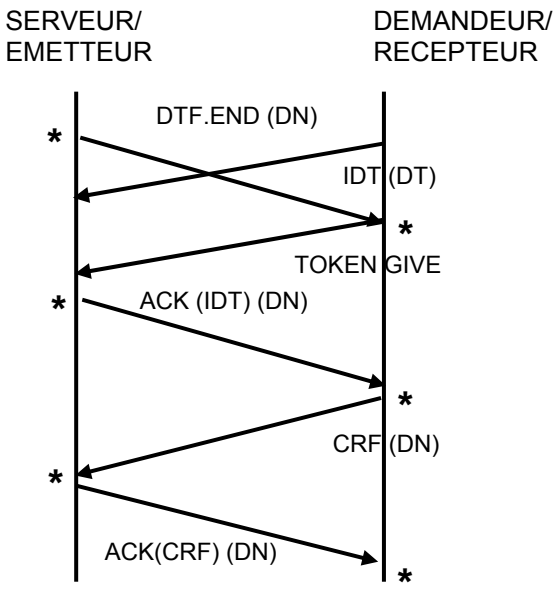
(ECRITURE)



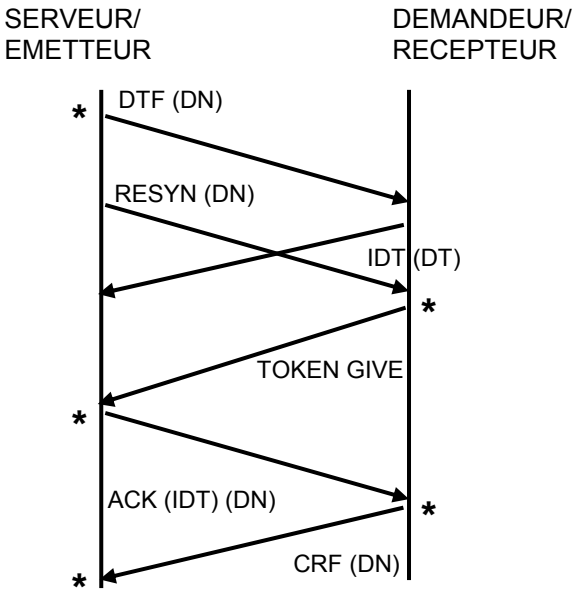
(LECTURE)



- **CROISEMENT DTF.END IDT (LECTURE)**



- **CROISEMENT RESYN IDT (LECTURE)**



4.3.2 Utilisation du service réseau par PeSIT.F'

Le protocole PeSIT.F' défini dans ce document, s'appuie sur un service réseau (couche 3).

Trois cas sont à distinguer :

- l'utilisation d'une liaison synchrone X25 en mode paquet
- l'utilisation d'une liaison téléphonique en mode X32
- l'utilisation d'une liaison asynchrone (éventuellement via le réseau téléphonique commuté) accédant à un réseau X25 à travers un PAD (Assembleur Dé-assembleur de paquets).

4.3.2.1 Utilisation d'une liaison synchrone X25

On suppose que la connexion réseau est établie préalablement et est maintenue durant la connexion PeSIT.

Toutes les unités de protocole du PeSIT.F' (FPDU) sont transmises en utilisant le service N-DATA.

Toutes les indications de services remontent de la couche réseau sont traitées de la façon suivante :

- N-RESET-IND : envoi d'une FPDU-ABORT
- N-DISC : envoi d'un F-ABORT-INDICATION
- N-EXPEDITED-DTA : (paquet d'interruption) ignoré par PeSIT.

4.3.2.2 Utilisation d'une liaison téléphonique en mode X32

L'interface offerte par une liaison téléphonique exploitée en mode X32 est identique à celle offerte par une liaison synchrone exploitée en mode X25. Le comportement de PeSIT.F' est donc, dans ce cas, identique à celui décrit ci-dessous.

4.3.2.3 Utilisation d'une liaison asynchrone (PAD)

Dans le cas d'une liaison asynchrone deux problèmes spécifiques sont à prendre en compte :

- la protection contre les erreurs de transmission sur la liaison terminale
- la non-transparence du PAD.

a) Protection contre les erreurs de transmission sur la liaison terminale

Afin de se prémunir contre les éventuels erreurs de transmission sur la liaison terminale, il est possible d'utiliser dans PeSIT.F' un mécanisme de contrôle d'erreur par calcul d'un polynôme (CRC).

Le demandeur indique dans la FPDU.CONNECT qu'il utilise un polynôme CRC. Toutes les FPDU (y compris la FPDU.CONNECT) sont alors suivies d'un CRC de 16 bits. Le CRC est calculé sur tous les octets de la FPDU en-tête et

paramètres compris. Les deux octets de CRC ne sont pas comptés dans le champ longueur de l'en-tête de la FPDU.

Il y a un CRC par FPDU, la longueur du CRC doit être prise en compte pour le calcul de la taille d'entité donnée passée à la couche inférieure (taille de NSDU). Dans le cas où le CRC est utilisé on ne fera pas de concaténation des FPDU.

L'algorithme de calcul du CRC est celui du protocole de Transport Classe 4 de l'ISO.

Le récepteur d'une FPDU doit vérifier la validité du CRC. En cas de réception de FPDU dont le CRC est incorrect, le récepteur émettra une FPDU.ABORT (diagnostic 310 : incident réseau) s'il est dans une phase autre que la phase Transfert de données, ou une FPDU.RESYNC (diagnostic 100 : erreur de transmission) pendant la phase de transfert de données.

D'autre part, l'utilisation des paramètres "nombre d'octets" (PI 27) et "nombre d'articles" (PI 28), échangés dans les FPDU.TRANS.END et FPDU.ACK(TRANS.END) permet de se prémunir contre la perte d'une ou plusieurs FPDU.DTF.

b) Transparence du PAD

Le problème de la non-transparence du PAD à certains caractères de contrôle est évité par l'utilisation du profil transparent du PAD (profil 14 de Transpac).

Le choix d'un tel profil se fait soit avant l'établissement du circuit virtuel par une commande locale de l'ETTD asynchrone au PAD, soit après l'établissement du circuit virtuel par un message de choix du profil transparent émis par l'ETTD X25 synchrone vers le PAD. Dans l'un et l'autre cas la sélection du profil PAD est effectuée avant l'envoi de la FPDU.CONNECT par le demandeur.

Le choix du profil 14 de Transpac laisse à l'ETTD asynchrone la possibilité de faire repasser le PAD en phase commande par la génération d'un signal BREAK.

4.3.3 Utilisation du service NETEX par PeSIT.F"

Le protocole de transfert de fichier PeSIT.F" s'appuie sur l'interface de type session offerte par NETEX.

Cette interface, PeSIT.F" utilise les primitives suivantes :

PRIMITIVE	FONCTION
OFFER	Acceptation d'appels entrants
CONNECT	Demande de connexion
CONFIRM	Acceptation de connexion
DISCONNECT	Coupure brutale de connexion
CLOSE	Libération de connexion
READ	Réception de données
WRITE	Emission de données

La primitive OFFER est utilisée par un PeSIT.F" acceptant d'être serveur pour notifier à NETEX qu'il accepte les éventuels appels entrants.

Toutes les primitives de l'interface NETEX autorisent l'émission de données utilisateur vers le correspondant. La longueur maximale de ces données se négocie lors de l'établissement de la connexion. Elle est supposée être toujours suffisante pour transporter les messages PeSIT.F" (FPDU).

Les message PeSIT.F" (FPDU) sont donc transmis soit en utilisant la primitive WRITE (pour l'émission) et READ (pour la réception), soit transmis comme données utilisateur des autres primitives.

Le tableau suivant donne les primitives utilisées pour transmettre les messages PeSIT.F" (FPDU).

MESSAGE PeSIT.F"	TRANSPORTE PAR	COMMENTAIRE
FPDU.CONNECT FPDU.ACONNECT FPDU.RCONNECT	CONNECT CONFIRM DISCONNECT	Sous réserve que le PeSIT.F" serveur ait émis un OFFER au préalable
FPDU.RELEASE FPDU.RELCONF FPDU.ABORT	CLOSE CLOSE DISCONNECT	
FPDU.CREATE FPDU.ACK(CREATE) FPDU.SELECT FPDU.ACK(SELECT) FPDU.DESELECT FPDU.ACK(DESELECT) FPDU.MSG FPDU.MSGDM FPDU.MSGMM FPDU.MSGFM FPDU.ACK(MSG)	READ/WRITE READ/WRITE READ/WRITE READ/WRITE READ/WRITE READ/WRITE READ/WRITE READ/WRITE READ/WRITE READ/WRITE READ/WRITE	
FPDU.ORF FPDU.ACK(ORF) FPDU.CRF FPDU.ACK(CRF)	READ/WRITE READ/WRITE READ/WRITE READ/WRITE	
FPDU.READ FPDU.ACK(READ) FPDU.WRITE FPDU.ACK(WRITE) FPDU.TRANS.END FPDU.ACK(TRANS.END)	READ/WRITE READ/WRITE READ/WRITE READ/WRITE READ/WRITE READ/WRITE	
FPDU.DTF FPDU.DTFDA FPDU.DTFMA FPDU.DTFFA FPDU.DTF.END FPDU.SYN FPDU.ACK(SYN) FPDU.RESYN FPDU.ACK(RESYN)	READ/WRITE READ/WRITE READ/WRITE READ/WRITE READ/WRITE READ/WRITE READ/WRITE READ/WRITE READ/WRITE	
FPDU.IDT FPDU.ACK(IDT)	READ/WRITE READ/WRITE	

a) Affectation d'une connexion NETEX à une connexion PeSIT.F"

Une connexion PeSIT.F" est associée à une connexion NETEX créée à cet effet. Un PeSIT.F" qui accepte d'être serveur doit effectuer un appel à la primitive OFFER afin de signaler à NETEX qu'il accepte les appels entrants. Le PeSIT.F" demandeur initialise l'établissement de la connexion par l'appel à la primitive CONNECT transportant une FPDU.CONNECT.

L'acceptation de la connexion se fait par l'émission de la FPDU.ACONNECT au moyen de la primitive CONFIRM.

Si la connexion ne peut être établie, son refus est indiqué par FPDU.RCONNECT qui est émise au moyen de la primitive DISCONNECT.

b) Fin de la connexion

La fin de la connexion peut intervenir de trois façons :

- Fin normale demandée par le demandeur qui émet une FPDU.RELEASE au moyen d'une primitive CLOSE. Le serveur répond alors par une FPDU.RELCONF au moyen d'une primitive CLOSE.

- Fin anormale provoquée par une des deux entités PeSIT.F" qui émet une FPDU.ABORT au moyen d'une primitive DISCONNECT.

- Déconnexion spontanée par la couche NETEX qui est notifiée à PeSIT.F" par une primitive DISCONNECT qui lui est remise lors d'un READ.

c) Transmission de données normales

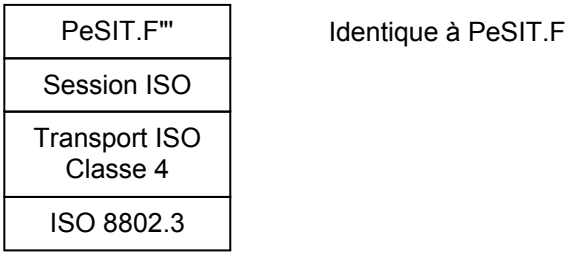
La connexion étant établie, toutes les FPDU de PeSIT.F" sont transmises et reçues par des primitives WRITE et READ. La longueur des données qui peuvent être transmises lors d'un WRITE ou READ est fixée par les deux entités NETEX lors de la connexion.

4.3.4 Utilisation du service session sur un réseau local par PeSIT.F"'

Le protocole de transfert de fichier PeSIT.F"' permet l'utilisation comme support de communication, d'un réseau local conforme à la norme ISO 8802.3 (identique à la norme IEEE 802.3).

Un tel réseau local n'offrant pas un service réseau suffisamment fiable, il est nécessaire d'utiliser au-dessus un protocole de Transport ISO, Classe 4 (classe avec détection d'erreurs et reprise sur erreur). De plus, par souci d'homogénéité il a été choisi d'utiliser au-dessus du Transport une couche Session ISO, de telle sorte que PeSIT.F"' soit identique à PeSIT.F.

L'architecture retenue pour PeSIT.F"' est donc :



Remarque : le réseau local conforme à la norme ISO 8802.3 (ou IEEE 802.3) est très voisin du réseau local Ethernet (défini par Intel Xerox et DEC). La seule différence réside dans l'en-tête de trame, le champ longueur de la norme ISO 8802.3 étant un champ type dans la définition d'Ethernet.

4.4 PROCEDURES RELATIVES AUX UNITES DE PROTOCOLE (FPDU)

Ce chapitre définit les séquences valides d'éléments de protocole de PeSIT.

Chaque message FPDU est décrit de la façon suivante :

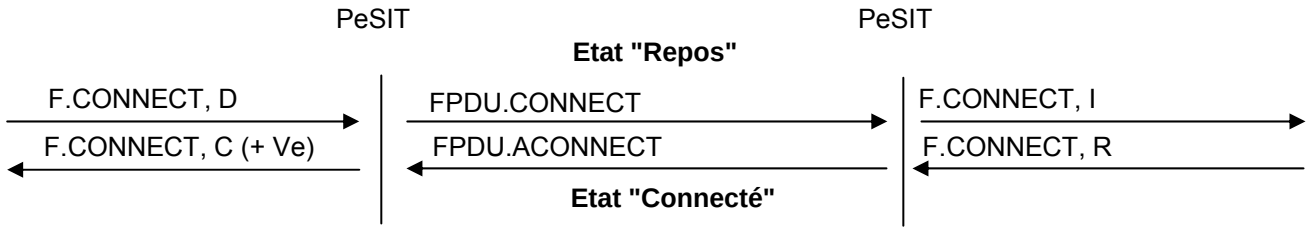
- contenu de la FPDU,
- procédure à l'envoi de la FPDU,
- procédure à la réception de la FPDU.

Un diagramme des échanges est donné pour chaque phase.

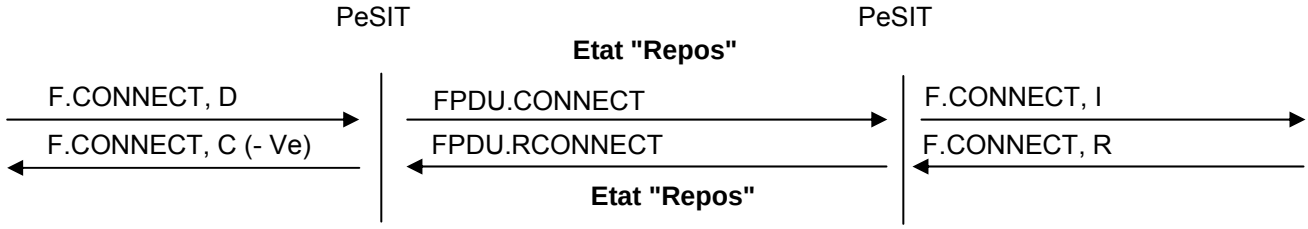
4.4.1 La FPDU.CONNECT

La FPDU.CONNECT est envoyée par le PeSIT demandeur lorsqu'il est dans l'état "Repos" pour établir une connexion PeSIT, sur une connexion "système de communication" préalablement affectée à l'unité active du même PeSIT demandeur.

- Acceptation de connexion



- Refus de connexion



a) Contenu de la FPDU.CONNECT

La FPDU.CONNECT contient tous les paramètres de la primitive de demande F.CONNECT, plus :

- ID.SRC : identification de la connexion origine attribuée par le PeSIT.F demandeur. Valeur arbitraire différente de zéro.

- ID.DST : identification de la connexion destinataire. Valeur égale à zéro dans la FPDU.CONNECT.

b) Envoi de la FPDU.CONNECT

Une primitive de demande de connexion F.CONNECT entraîne l'affectation par le PeSIT d'une connexion "système de communication" à la connexion PeSIT. Pour établir cette connexion :

- PeSIT.F envoie la FPDU.CONNECT comme données utilisateur dans la primitive de demande de connexion session S-CONNECT. Le PeSIT.F passe alors dans l'état "connexion en attente" ;

- alors que PeSIT.F' envoie la FPDU.CONNECT de manière explicite dans le flux de données normal N-DATA, après avoir établi la connexion réseau. Le PeSIT.F' passe alors dans l'état "connexion en attente".

c) Réception d'une FPDU.CONNECT

la réception d'une FPDU.CONNECT valide par le PeSIT lorsqu'il est dans l'état "Repos", entraîne la notification d'une primitive d'indication F.CONNECT à l'utilisateur de la connexion PeSIT, déterminée par le paramètre "identification du serveur" de la FPDU.CONNECT.

Le PeSIT serveur attend alors une primitive de réponse F.CONNECT de la part de l'utilisateur du service PeSIT demandé. Il passe alors dans l'état "connexion en attente".

4.4.2 La FPDU.ACONNECT

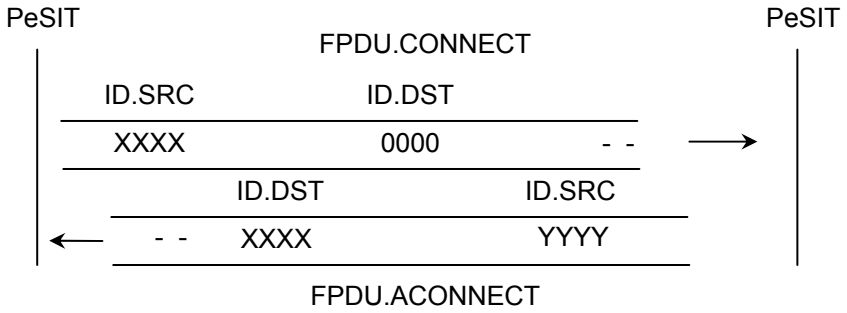
Une entité PeSIT.F recevant une FPDU.CONNECT peut accepter une proposition de connexion PeSIT en envoyant une FPDU.ACONNECT à l'entité PeSIT demandeur, sur la même connexion "système de communication".

a) Contenu de la FPDU.ACONNECT

La FPDU.ACONNECT contient tous les paramètres de la primitive de réponse F.CONNECT, plus :

- ID.SRC : identification de la connexion origine chez le serveur attribuée par le PeSIT serveur dans la FPDU.ACONNECT. Valeur arbitraire différente de zéro,

- ID.DST : identification de la connexion chez le destinataire de la FPDU. Sa valeur est égale à celle du paramètre ID.SRC reçue dans la FPDU.CONNECT.



b) Envoi de la FPDU.ACONNECT

La primitive de réponse F.CONNECT (+ Ve) entraîne l'envoi de la FPDU.ACONNECT dans le champ utilisateur de la primitive de session S-ACCEPT pour PeSIT.F ou dans le flux normal de données.

La connexion étant ainsi établie, l'entité PeSIT serveur passe dans l'état "CONNECTE" et peut recevoir n'importe quelle demande de service ou FPDU autorisée par la procédure serveur.

c) Réception de la FPDU.ACONNECT

La réception d'une FPDU.ACONNECT valide par le PeSIT demandeur lorsqu'il est dans l'état "connexion en attente", entraîne la notification d'une primitive de confirmation F.CONNECT à l'utilisateur demandeur. La connexion étant ainsi établie avec succès, l'entité PeSIT demandeur passe à l'état "CONNECTE" et peut recevoir toute demande de service ou FPDU autorisée par la procédure du demandeur.

4.4.3 La FPDU.RCONNECT

La FPDU.RCONNECT est utilisée par l'entité PeSIT appelée pour refuser une tentative d'établissement de connexion PeSIT.

a) Contenu de la FPDU.RCONNECT

La FPDU.RCONNECT contient tous les paramètres de la primitive de réponse F.CONNECT, plus :

- ID.SRC : identification de la connexion chez le PeSIT. Valeur = 0 puisqu'il y a refus de connexion,
- ID.DST : identification de la connexion destinataire de la FPDU. Valeur = celle du paramètre ID.SRC reçue dans la FPDU.CONNECT.

b) Envoi de la FPDU.RCONNECT

Une primitive de réponse F.CONNECT (refus) entraîne l'envoi de la FPDU.RCONNECT dans le champ utilisateur de la primitive de session S-REFUSE par le PeSIT.F serveur ou dans le flux normal de données N-DATA par le PeSIT.F' serveur. Aucune connexion PeSIT n'est établie.

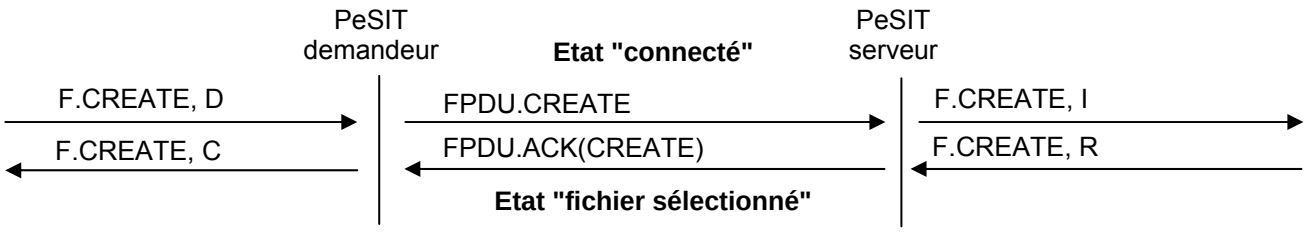
c) Réception de la FPDU.RCONNECT

La réception par le PeSIT demandeur lorsqu'il est dans l'état "connexion en attente" d'une FPDU.RCONNECT valide entraîne la notification d'une primitive de confirmation F.CONNECT (refus) à l'utilisateur et :

- le passage à l'état "repos" du PeSIT demandeur,
- la demande de déconnexion du service réseau (N-DISCONNECT) par le PeSIT.F demandeur et le passage à l'état "repos".

4.4.4 La FPDU.CREATE

La FPDU.CREATE est toujours envoyée par le PeSIT demandeur alors qu'il est dans l'état "connecté", lorsque celui-ci désire la création d'un fichier chez l'entité PeSIT serveur homologue pour un transfert en écriture. Le fichier est identifié parfaitement par le paramètre "identificateur du fichier".



a) Contenu de la FPDU.CREATE

La FPDU.CREATE contient tous les paramètres de la primitive de demande F.CREATE, plus :

- ID.DST : identification de la connexion chez le PeSIT destinataire.

b) Envoi de la FPDU.CREATE

Une primitive de demande F.CREATE entraîne l'envoi par l'entité PeSIT de la FPDU.CREATE dans le flux de données normal du "système de communication". L'entité PeSIT demandeur passe à l'état "Création de fichier en attente".

c) Réception de la FPDU.CREATE

La réception d'une FPDU.CREATE valide par le PeSIT serveur alors qu'il est dans l'état "connecté" entraîne la notification de la primitive d'indication F.CREATE à l'utilisateur serveur. L'entité PeSIT passe à l'état "création de fichier en attente".

4.4.5 La FPDU.ACK(CREATE)

La FPDU.ACK(CREATE) est envoyée par le PeSIT serveur lorsqu'il est dans l'état "création de fichier en attente" pour indiquer l'acceptation ou le refus de la création du fichier pour le transfert. Le paramètre "diagnostic" contenu dans la FPDU donne la raison exacte du refus si la création ne peut se faire.

a) Contenu de la FPDU.ACK(CREATE)

Tous les paramètres de la primitive de réponse F.CREATE, plus :

- ID.DST : identification de la connexion du destinataire.

b) Envoi de la FPDU.ACK(CREATE)

La primitive de réponse F.CREATE entraîne l'envoi par le PeSIT serveur de la FPDU.ACK(CREATE) dans le flux normal de données du système de communication. Si le paramètre "diagnostic" de la primitive de réponse F.CREATE indique "succès", le PeSIT serveur passe à l'état de "fichier sélectionné". Dans le cas contraire, le PeSIT serveur passe à l'état "CONNECTE".

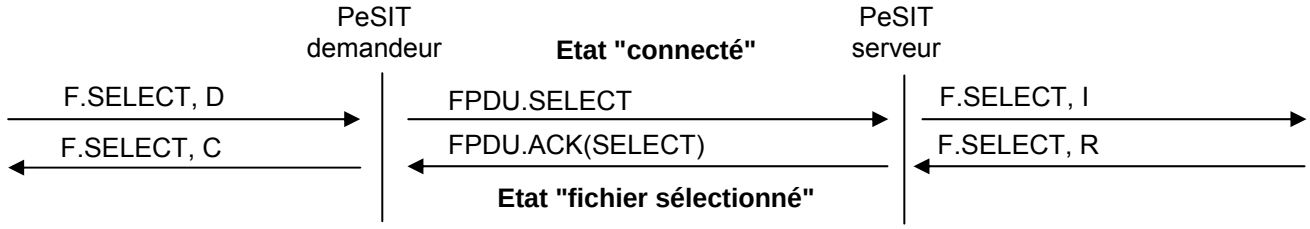
c) Réception de la FPDU.ACK(CREATE)

La réception de la FPDU.ACK(CREATE) par le PeSIT demandeur, alors qu'il est dans l'état "création de fichier en attente", entraîne la notification de la primitive de confirmation F.CREATE à l'utilisateur demandeur et le passage par l'entité PeSIT à l'état :

- "fichier sélectionné" si le paramètre "diagnostic" indique "succès",
- "CONNECTE" dans le cas de non succès.

4.4.6 La FPDU.SELECT

La FPDU.SELECT est toujours envoyée par le PeSIT demandeur lorsqu'il est dans l'état "connecté", pour sélectionner un fichier chez le PeSIT homologue pour un transfert en lecture.



a) Contenu de la FPDU.SELECT

Tous les paramètres de la primitive de demande F.SELECT, plus :

- ID.DST : identification de la connexion chez le destinataire.

b) Envoi de la FPDU.SELECT

Une primitive de demande F.SELECT entraîne l'envoi par l'entité PeSIT demandeur de la FPDU.SELECT dans le flux de données normal du "système de communication". L'entité PeSIT passe à l'état "sélection de fichier en attente".

c) Réception de la FPDU.SELECT

La réception d'une FPDU.SELECT valide par le PeSIT serveur alors qu'il est dans l'état "connecté" entraîne la notification à l'utilisateur serveur de la primitive d'indication F.SELECT. Le PeSIT passe à l'état "sélection de fichier en attente".

4.4.7 La FPDU.ACK(SELECT)

La FPDU.ACK(SELECT) est envoyée par le PeSIT lorsqu'il est dans l'état "sélection de fichier en attente" pour indiquer l'acceptation ou le refus de la sélection du fichier demandée pour le transfert. Le paramètre "diagnostic" indique la raison exacte du refus si la sélection ne peut se faire.

a) Contenu de la FPDU.ACK(SELECT)

Tous les paramètres de la primitive de réponse F.SELECT, plus :

- ID.DST : identification de la connexion du destinataire.

b) Envoi de la FPDU.ACK(SELECT)

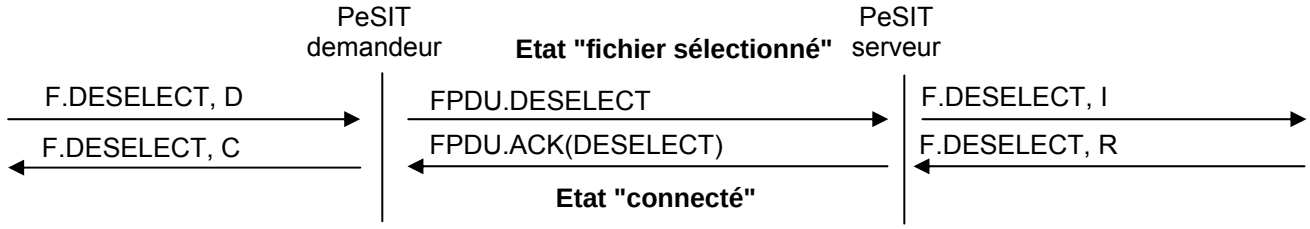
La primitive de réponse F.SELECT entraîne l'envoi par le PeSIT serveur, alors qu'il est dans l'état "sélection de fichier en attente", de la FPDU.ACK(SELECT) dans le flux normal de données du "système de communication". Si le paramètre "diagnostic" de la primitive de réponse F.SELECT indique "succès", le PeSIT serveur passe à l'état de "fichier sélectionné". Dans le cas contraire, le PeSIT serveur passe à l'état "connecté".

c) Réception de la FPDU.ACK(SELECT)

La réception de la FPDU.ACK(SELECT) valide par le PeSIT demandeur alors qu'il est dans l'état "sélection de fichier en attente", entraîne la notification d'une primitive de confirmation F.SELECT à l'utilisateur serveur. Si le paramètre "diagnostic" indique "succès", le PeSIT demandeur passe à l'état "fichier sélectionné", sinon il passe à l'état "connecté".

4.4.8 La FPDU.DESELECT

La FPDU.DESELECT est toujours envoyée par le PeSIT demandeur lorsqu'il est dans l'état "fichier sélectionné", pour demander la libération du fichier précédemment sélectionné pour le transfert.



a) Contenu de la FPDU.DESELECT

Tous les paramètres de la primitive de demande F.DESELECT, plus :

- ID.DST : identification de la connexion du destinataire.

b) Envoi de la FPDU.DESELECT

Une primitive de demande F.DESELECT entraîne l'envoi par le PeSIT demandeur lorsqu'il est dans l'état "fichier sélectionné", de la FPDU.DESELECT dans le flux de données normal du "système de communication". L'entité PeSIT passe à l'état "libération du fichier en attente".

c) Réception de la FPDU. DESELECT

La réception d'une FPDU.DESELECT valide par le PeSIT serveur alors qu'il est dans l'état "fichier sélectionné" entraîne la notification de la primitive d'indication F.DESELECT à l'utilisateur serveur. Le PeSIT passe à l'état "libération du fichier en attente".

4.4.9 La FPDU.ACK(DESELECT)

La FPDU.ACK(DESELECT) est toujours envoyée par le PeSIT serveur lorsqu'il est dans l'état "libération du fichier en attente" pour indiquer le résultat de l'exécution de la demande de libération de fichier. Le paramètre "diagnostic" de la FPDU indique la raison exacte en cas de non succès.

a) Contenu de la FPDU.ACK(DESELECT)

Tous les paramètres de la primitive de réponse F.DESELECT, plus :

- ID.DST : identificateur de la connexion chez le destinataire.

b) Envoi de la FPDU.ACK(DESELECT)

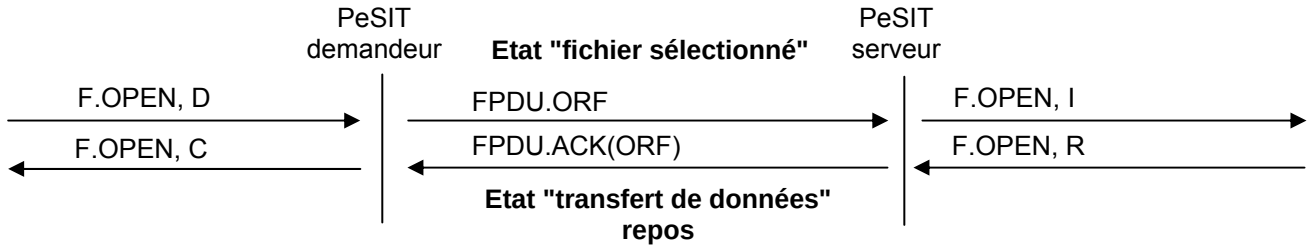
La primitive de réponse F.DESELECT entraîne l'envoi par le PeSIT serveur, de la FPDU.ACK(DESELECT) dans le flux normal de données du "système de communication". Le PeSIT passe à l'état "connecté".

c) Réception de la FPDU.ACK(DESELECT)

La réception de la FPDU.ACK(DESELECT) valide par le PeSIT demandeur alors qu'il est dans l'état "libération de fichier en attente", entraîne la notification de la primitive de confirmation F.DESELECT à l'utilisateur demandeur et le passage de l'entité PeSIT à l'état "connecté".

4.4.10 La FPDU.ORF

La FPDU.ORF est toujours envoyée par le PeSIT demandeur lorsqu'il est dans l'état "fichier sélectionné", pour demander l'ouverture du fichier à distance.



a) Contenu de la FPDU.ORF

Tous les paramètres de la primitive de demande F.OPEN, plus :

- ID.DST : identification de la connexion chez le destinataire.

b) Envoi de la FPDU.ORF

Une primitive de demande F.OPEN entraîne l'envoi par le PeSIT demandeur lorsqu'il est dans l'état "fichier sélectionné", de la FPDU.ORF dans le flux de données normal du "système de communication". L'entité PeSIT passe à l'état "ouverture du fichier en attente".

c) Réception de la FPDU.ORF

La réception d'une FPDU.ORF valide par le PeSIT serveur alors qu'il est dans l'état "fichier sélectionné" entraîne la notification de la primitive d'indication F.OPEN à l'utilisateur serveur. Le PeSIT passe à l'état "ouverture de fichier en attente".

4.4.11 La FPDU.ACK(ORF)

La FPDU.ACK(ORF) est toujours envoyée par le PeSIT serveur lorsqu'il est dans l'état "ouverture de fichier en attente" pour indiquer le résultat de la demande d'ouverture de fichier. Le paramètre "diagnostic" de la FPDU indique la raison exacte du refus en cas de non succès.

a) Contenu de la FPDU.ACK(ORF)

Tous les paramètres de la primitive de réponse F.OPEN, plus :

- ID.DST : identificateur de la connexion chez le destinataire.

b) Envoi de la FPDU.ACK(ORF)

La primitive de réponse F.OPEN entraîne l'envoi par le PeSIT, de la FPDU.ACK(ORF) dans le flux normal de données du "système de communication". Le PeSIT passe à l'état :

- "transfert de données - repos" si le diagnostic indique "succès",
- "fichier sélectionné" si le diagnostic indique le contraire.

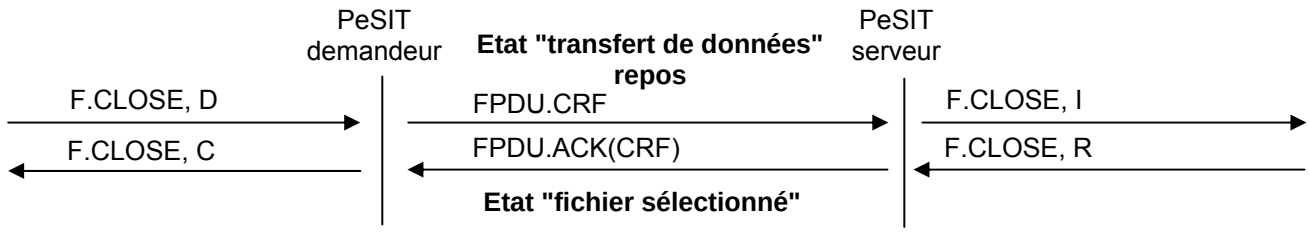
c) Réception de la FPDU.ACK(ORF)

La réception d'une FPDU.ACK(ORF) valide par le PeSIT demandeur alors qu'il est dans l'état "ouverture de fichier en attente", entraîne la notification d'une primitive de confirmation F.OPEN à l'utilisateur demandeur et le passage de l'entité PeSIT à l'état :

- "transfert de données - repos" si le diagnostic indique "succès",
- "fichier sélectionné" si le diagnostic indique le contraire.

4.4.12 La FPDU.CRF

La FPDU.CRF est toujours envoyée par le PeSIT.F/F' demandeur lorsqu'il est dans l'état "transfert de données - repos", pour demander la fermeture du fichier.



a) Contenu de la FPDU.CRF

Tous les paramètres de la primitive de demande F.CLOSE, plus :

- ID.DST : identification de la connexion chez le destinataire.

b) Envoi de la FPDU.CRF

Une primitive de demande F.CLOSE entraîne l'envoi par le PeSIT demandeur lorsqu'il est dans l'état "transfert de données - repos", de la FPDU.CRF dans le flux de données normal du "système de communication". L'entité PeSIT passe à l'état "fermeture du fichier en attente".

c) Réception de la FPDU.CRF

La réception d'une FPDU.CRF valide par le PeSIT serveur, alors qu'il est dans l'état "transfert de données - repos" entraîne la notification de la primitive d'indication F.CLOSE à l'utilisateur serveur. Le PeSIT passe à l'état "fermeture de fichier en attente".

4.4.13 La FPDU.ACK(CRF)

La FPDU.ACK(CRF) est toujours envoyée par le PeSIT serveur lorsqu'il est dans l'état "fermeture de fichier en attente" pour indiquer le résultat de la demande de fermeture de fichier. Le paramètre "diagnostic" de la FPDU indique la raison exacte du refus en cas de non succès.

a) Contenu de la FPDU.ACK(CRF)

Tous les paramètres de la primitive de réponse F.CLOSE, plus :

- ID.DST : identificateur de la connexion chez le destinataire.

b) Envoi de la FPDU.ACK(CRF)

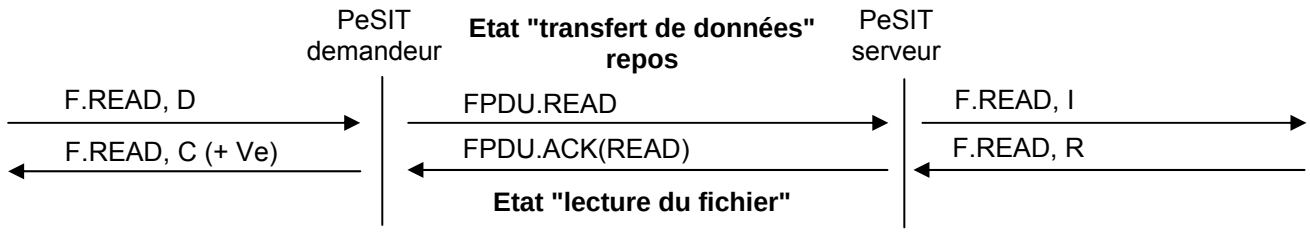
La primitive de réponse F.CLOSE entraîne l'envoi par le PeSIT serveur, de la FPDU.ACK(CRF) dans le flux normal de données du "système de communication". Le PeSIT passe à l'état "fichier sélectionné".

c) Réception de la FPDU.ACK(CRF)

La réception d'une FPDU.ACK(CRF) valide par le PeSIT demandeur alors qu'il est dans l'état "fermeture de fichier en attente", entraîne la notification d'une primitive de confirmation F.CLOSE à l'utilisateur demandeur et le passage de l'entité PeSIT à l'état "fichier sélectionné".

4.4.14 La FPDU.READ

La FPDU.READ est toujours envoyée par le PeSIT demandeur lorsqu'il est dans l'état "transfert de données - repos", pour demander le lancement du transfert de données en lecture. C'est dans cette phase que le point de relance est négocié lorsqu'il s'agit d'une relance.



a) Contenu de la FPDU.READ

Tous les paramètres de la primitive de demande F.READ, plus :

- ID.DST : identification de la connexion chez le destinataire.

b) Envoi de la FPDU.READ

Une primitive de demande F.READ entraîne l'envoi par le PeSIT demandeur, lorsqu'il est dans l'état "transfert de données - repos", de la FPDU.READ dans le flux de données normal du "système de communication". L'entité PeSIT passe à l'état "lancement de lecture en attente".

c) Réception de la FPDU.READ

La réception d'une FPDU.READ valide par le PeSIT serveur, alors qu'il est dans l'état "transfert de données - repos" entraîne la notification de la primitive d'indication F.READ à l'utilisateur serveur. L'entité PeSIT passe à l'état "lancement de lecture en attente".

4.4.15 La FPDU.ACK(READ)

La FPDU.ACK(READ) est toujours envoyée par le PeSIT serveur lorsqu'il est dans l'état "lancement de lecture en attente" pour indiquer le résultat de la demande de lancement de lecture. Le paramètre "diagnostic" de la FPDU indique la raison exacte du refus en cas de non succès.

a) Contenu de la FPDU.ACK(READ)

Tous les paramètres de la primitive de réponse F.READ, plus :

- ID.DST : identificateur de la connexion chez le destinataire.

b) Envoi de la FPDU.ACK(READ)

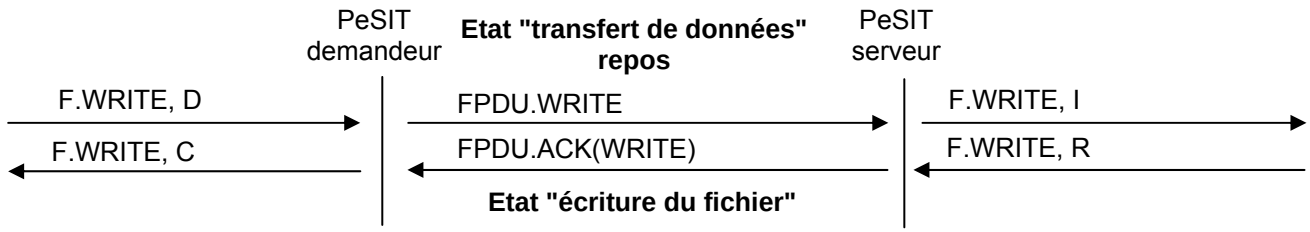
La primitive de réponse F.READ entraîne l'envoi par le PeSIT serveur, de la FPDU.ACK(READ) dans le flux normal de données du "système de communication". Le PeSIT passe à l'état "lecture du fichier ", si le code diagnostic indique "succès", sinon à l'état "transfert de données - repos".

c) Réception de la FPDU.ACK(READ)

La réception d'une FPDU.ACK(READ) valide par le PeSIT demandeur alors qu'il est dans l'état "lancement de lecture en attente", entraîne la notification d'une primitive de confirmation F.READ à l'utilisateur demandeur et le passage de l'entité PeSIT à l'état "lecture fichier", si le code diagnostic indique "succès", sinon à l'état "transfert de données - repos".

4.4.16 La FPDU.WRITE

La FPDU.WRITE est toujours envoyée par le PeSIT demandeur lorsqu'il est dans l'état "transfert de données - repos", pour demander le lancement du transfert de données en écriture. C'est dans cette phase que le point de relance est négocié lorsqu'il s'agit d'une relance.



a) Contenu de la FPDU.WRITE

Tous les paramètres de la primitive de demande F.WRITE, plus :

- ID.DST : identification de la connexion chez le destinataire.

b) Envoi de la FPDU.WRITE

Une primitive de demande F.WRITE entraîne l'envoi par le PeSIT demandeur, lorsqu'il est dans l'état "transfert de données - repos", de la FPDU.WRITE dans le flux de données normal du "système de communication". L'entité PeSIT passe à l'état "lancement d'écriture en attente".

c) Réception de la FPDU.WRITE

La réception d'une FPDU.WRITE valide par le PeSIT serveur, alors qu'il est dans l'état "transfert de données - repos" entraîne la notification de la primitive d'indication F.WRITE à l'utilisateur serveur. L'entité PeSIT passe à l'état "lancement d'écriture en attente".

4.4.17 La FPDU.ACK(WRITE)

La FPDU.ACK(WRITE) est toujours envoyée par le PeSIT serveur lorsqu'il est dans l'état "lancement d'écriture en attente" pour indiquer le résultat de la demande de lancement du transfert d'écriture. Le paramètre "diagnostic" de la FPDU donne la raison exacte du refus en cas de non succès.

a) Contenu de la FPDU.ACK(WRITE)

Tous les paramètres de la primitive de réponse F.WRITE, plus :

- ID.DST : identificateur de la connexion chez le destinataire.

b) Envoi de la FPDU.ACK(WRITE)

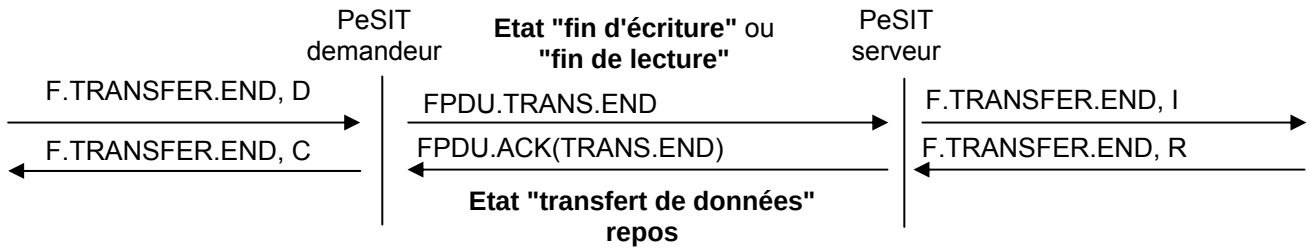
La primitive de réponse F.WRITE entraîne l'envoi par le PeSIT serveur, de la FPDU.ACK(WRITE) dans le flux normal de données du "système de communication". Le PeSIT passe à l'état "écriture du fichier ", si le code diagnostic indique "succès", sinon à l'état "transfert de données - repos".

c) Réception de la FPDU.ACK(WRITE)

La réception d'une FPDU.ACK(WRITE) valide par le PeSIT demandeur alors qu'il est dans l'état "lancement d'écriture en attente", entraîne la notification d'une primitive de confirmation F.WRITE à l'utilisateur demandeur et le passage de l'entité PeSIT à l'état "écriture du fichier", si le code diagnostic indique "succès", sinon à l'état "transfert de données - repos".

4.4.18 La FPDU.TRANS.END

La FPDU.TRANS.END est toujours envoyée par le PeSIT demandeur lorsqu'il est dans l'état "fin d'écriture" ou "fin de lecture", pour demander la fin du transfert du fichier.



a) Contenu de la FPDU.TRANS.END

Tous les paramètres de la primitive de demande F.TRANSFER.END, plus :

- ID.DST : identification de la connexion chez le destinataire.

b) Envoi de la FPDU.TRANS.END

Une primitive de demande F.TRANSFER.END entraîne l'envoi par le PeSIT demandeur, lorsqu'il est dans l'état "fin de lecture" ou "fin d'écriture", de la FPDU.TRANS.END dans le flux de données normal du "système de communication". L'entité PeSIT passe à l'état "fin de transfert de lecture en attente" ou "fin de transfert d'écriture en attente".

c) Réception de la FPDU.TRANS.END

La réception d'une FPDU.TRANS.END valide par le PeSIT serveur, alors qu'il est dans l'état "fin de lecture" ou "fin d'écriture", entraîne la notification de la primitive d'indication F.TRANSFER.END à l'utilisateur serveur. L'entité PeSIT passe à l'état "fin de transfert de lecture en attente" ou "fin de transfert d'écriture en attente".

4.4.19 La FPDU.ACK(TRANS.END)

La FPDU.ACK(TRANS.END) est toujours envoyée par le PeSIT serveur lorsqu'il est dans l'état "fin de transfert de lecture en attente" ou "fin de transfert d'écriture en attente" pour indiquer le résultat de la demande de lancement de transfert du fichier. Le paramètre "diagnostic" de la FPDU donne la raison exacte du refus en cas de non succès.

a) Contenu de la FPDU.ACK(TRANS.END)

Tous les paramètres de la primitive de réponse F.TRANSFER.END, plus :

- ID.DST : identificateur de la connexion chez le destinataire.

b) Envoi de la FPDU.ACK(TRANS.END)

La primitive de réponse F.TRANSFER.END entraîne l'envoi par le PeSIT serveur, de la FPDU.ACK(TRANS.END) dans le flux normal de données du "système de communication". Le PeSIT passe à l'état "transfert de données - repos".

c) Réception de la FPDU.ACK(TRANS.END)

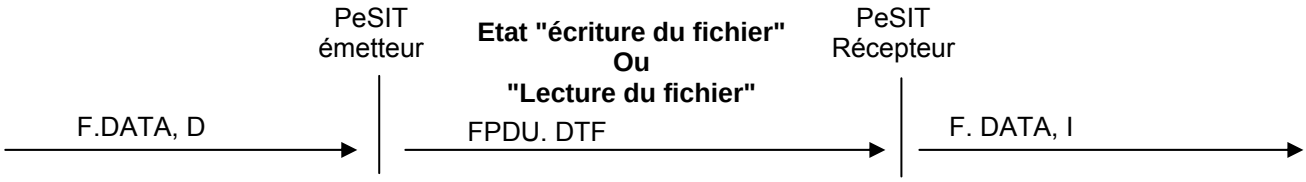
La réception d'une FPDU.ACK(TRANS.END) valide par le PeSIT demandeur alors qu'il est dans l'état "fin de transfert de lecture en attente ou "fin de transfert d'écriture en attente", entraîne la notification d'une primitive de confirmation F.TRANSFER.END à l'utilisateur demandeur et le passage de l'entité PeSIT à l'état "transfert de données - repos".

4.4.20 Les FPDU.DTF, FPDU.DTFDA, FPDU.DTFMA, FPDU.DTFFA

La FPDU.DTF, FPDU.DTFDA, FPDU.DTFMA, FPDU.DTFFA sont envoyées par le PeSIT Emetteur lorsqu'il est dans l'état "écriture de fichier" ou "lecture de fichier" pour transférer les données du fichier.

4.4.20.1 La FPDU.DTF MONO-ARTICLE

La FPDU.DTF transporte un et un seul article du fichier.



a) Contenu de la FPDU.DTF

- contenu de la primitive F.DATA (article du fichier),
- ID.DST : identification de la connexion chez le destinataire.

b) Envoi de la FPDU.DTF

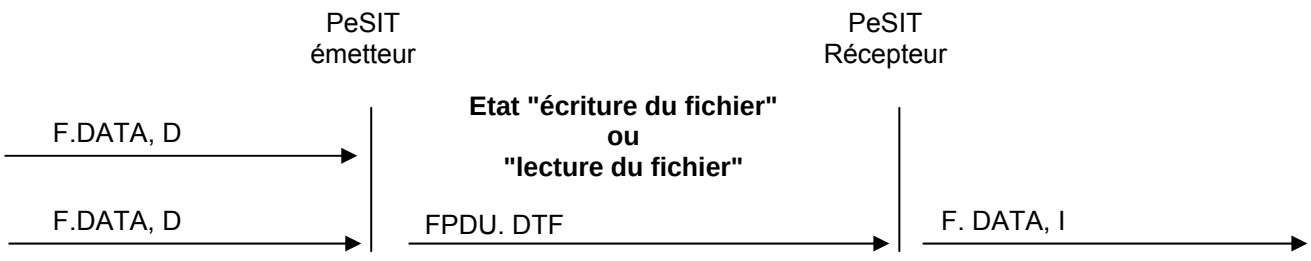
La primitive de réponse F.DATA entraîne l'envoi de la FPDU.DTF par le PeSIT émetteur, lorsqu'il est dans l'état "lecture du fichier" ou "écriture du fichier". L'entité PeSIT reste dans le même état.

c) Réception de la FPDU.DTF

La réception d'une FPDU.DTF récepteur, alors qu'il est dans l'état "lecture du fichier" ou "écriture du fichier", entraîne la notification de la primitive d'indication F.DATA à l'utilisateur récepteur. L'entité PeSIT récepteur reste dans le même état.

4.4.20.2 La FPDU.DTF MULTI-ARTICLES

La FPDU.DTF transporte plusieurs articles du fichier.



a) Contenu de la FPDU.DTF

- contenu de plusieurs primitives F.DATA (plusieurs articles du fichier),
- ID.DST : identification de la connexion chez le destinataire.

b) Envoi de la FPDU.DTF

Une primitive de demande F.DATA entraîne la concaténation de l'article dans la FPDU.DTF lorsque le PeSIT émetteur est dans l'état "lecture du fichier" ou "écriture du fichier" et peut entraîner l'envoi de la FPDU.DTF. Le nombre d'articles de la FPDU.DTF ou les critères de déclenchement de son envoi sont laissés à la liberté de l'implémenteur. L'entité PeSIT reste dans le même état.

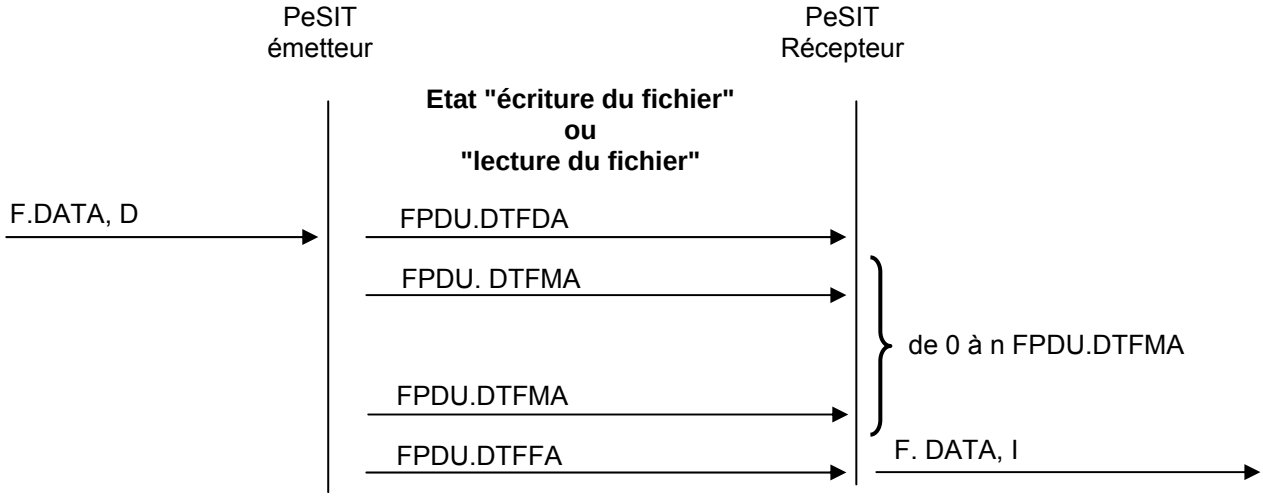
c) Réception de la FPDU.DTF

La réception d'une FPDU.DTF valide par le PeSIT récepteur, alors qu'il est dans l'état "lecture du fichier" ou "écriture du fichier", entraîne la notification de la primitive d'indication F.DATA à l'utilisateur récepteur. L'entité PeSIT récepteur reste dans le même état.

4.4.20.3 SEGMENTATION DES ARTICLES

Lorsque la taille de l'article à transporter dépasse la taille maximale supportée par une FPDU (taille de l'entité de données moins la taille de l'en-tête de la FPDU), l'article peut être segmenté par plusieurs FPDU :

- une FPDU.DTFDA (début d'article),
- un nombre positif ou nul de FPDU.DTFMA (milieu d'article),
- une FPDU.DTFFA (fin d'article).



a) Contenu de la FPDU.DTFDA, FPDU.DTFMA, FPDU.DTFFA

Les FPDU.DTFDA, FPDU.DTFMA, FPDU.DTFFA contiennent chacune :

- une fraction du contenu de la primitive F.DATA,
- ID.DST : identification de la connexion chez le destinataire.

b) Envoi des FPDU.DTFDA, FPDU.DTFMA, FPDU.DTFFA

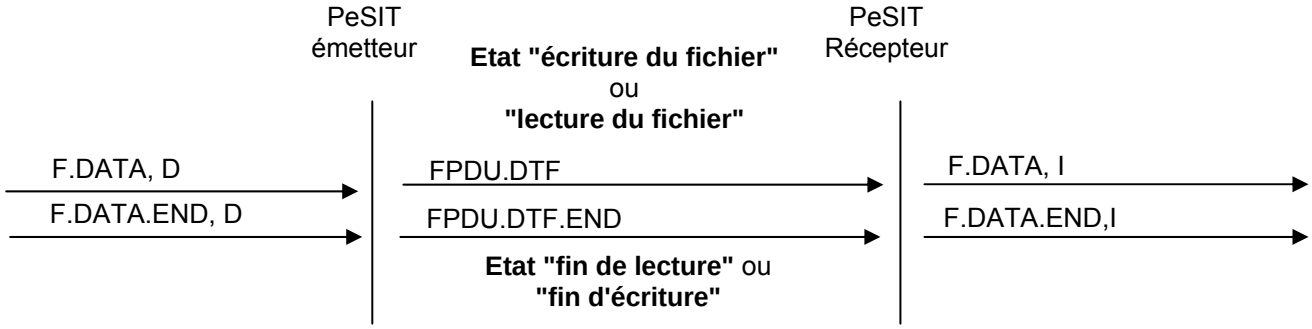
Une primitive de demande F.DATA entraîne l'envoi par le PeSIT émetteur, lorsqu'il est dans l'état "lecture du fichier" ou "écriture du fichier" d'une FPDU.DTFDA, d'un nombre de FPDU.DTFMA positif ou nul, et d'une FPDU.DTFFA. L'entité PeSIT reste dans le même état.

c) Réception de la FPDU.DTFFA

La réception d'une FPDU.DTFFA valide par le PeSIT récepteur, alors qu'il est dans l'état "lecture du fichier" ou "écriture du fichier", entraîne la notification de la primitive d'indication F.DATA à l'utilisateur récepteur. L'entité PeSIT récepteur reste dans le même état.

4.4.21 La FPDU.DTF.END

La FPDU.DTF.END est toujours envoyée par le PeSIT émetteur lorsqu'il est dans l'état "lecture de fichier" ou "écriture de fichier" pour signaler la fin du transfert de données. La paramètre "diagnostic" indique la raison exacte de la fin du transfert.



a) Contenu de la FPDU.DTF.END, plus

Tous les paramètres de la primitive de demande F.DATA.END, plus :

- ID.DST : identification de la connexion chez le destinataire.

b) Envoi de la FPDU.DTF.END

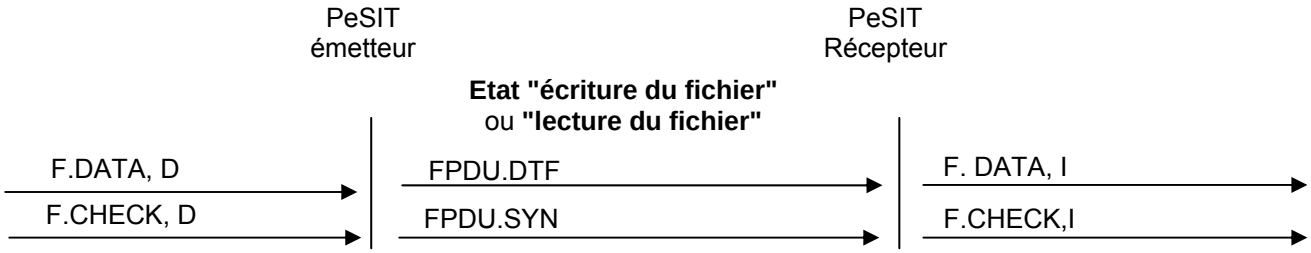
Une primitive de demande F.DATA.END entraîne l'envoi de la FPDU.DTF.END par le PeSIT émetteur, lorsqu'il est dans l'état "lecture du fichier" ou "écriture du fichier". L'entité PeSIT passe alors à l'état "fin de lecture" ou "fin d'écriture".

c) Réception de la FPDU.DTF.END

La réception d'une FPDU.DTF.END récepteur, alors qu'il est dans l'état "lecture du fichier" ou "écriture du fichier", entraîne la notification de la primitive d'indication F.DATA.END à l'utilisateur récepteur et le passage à l'état "fin de lecture" ou "fin d'écriture".

4.4.22 La FPDU.SYN

La FPDU.SYN est toujours envoyée par le PeSIT Emetteur lorsqu'il est dans l'état "lecture de fichier" ou "écriture de fichier" pour demander la pose de points de synchronisation.



a) Contenu de la FPDU.SYN

Tous les paramètres de la primitive de demande F.CHECK, plus :

- ID.DST : identification de la connexion chez le destinataire.

b) Envoi de la FPDU.SYN

Une primitive de demande F.CHECK entraîne l'envoi de la FPDU.SYN par le PeSIT émetteur. L'entité PeSIT émetteur reste dans le même état.

c) Réception de la FPDU.SYN

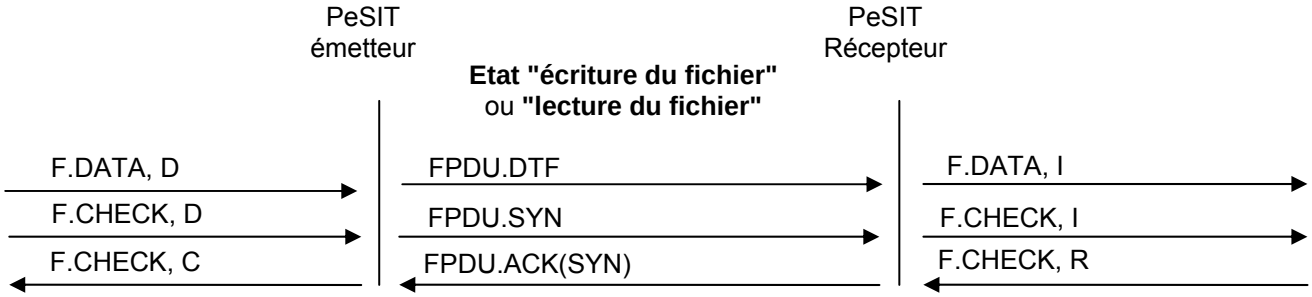
La réception d'une FPDU.SYN valide par le PeSIT récepteur, alors qu'il est dans l'état "lecture du fichier" ou "écriture du fichier", entraîne la notification de la primitive d'indication F.CKECK à l'utilisateur récepteur. L'entité PeSIT ne change pas d'état.

d) Remarque

Les FPDU.SYN doivent être émises entre les articles du fichier. Ainsi, dans le cas où les mécanismes de segmentation des articles sur plusieurs FPDU.DTFDA, FPDU.DTFMA, FPDU.DTFFA sont utilisés, les FPDU.SYN ne peuvent être émises qu'après une FPDU.DTFFA ou avant une FPDU.DTFDA .

4.4.23 La FPDU.ACK(SYN)

La FPDU.ACK(SYN) est toujours envoyée par le PeSIT récepteur lorsqu'il est dans l'état "lecture de fichier" ou "écriture de fichier" pour acquitter les points de synchronisation posés préalablement.



a) Contenu de la FPDU.ACK(SYN)

Tous les paramètres de la primitive de demande F.CHECK, plus :

- ID.DST : identification de la connexion chez le destinataire.

b) Envoi de la FPDU.ACK(SYN)

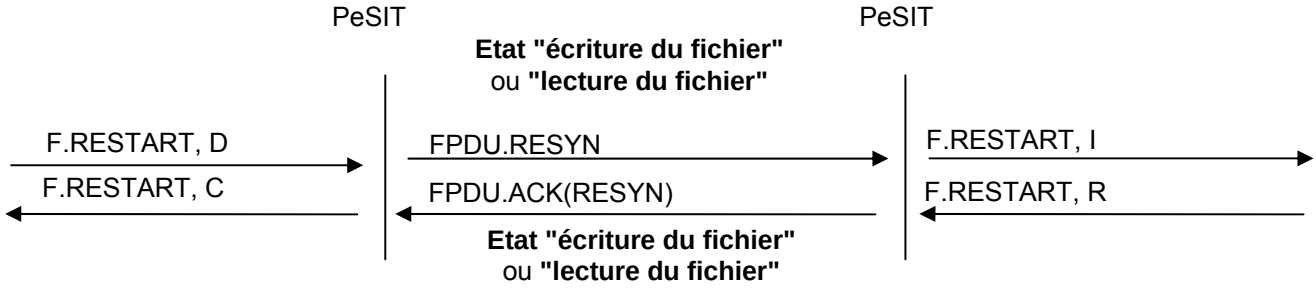
Une primitive de réponse F.CHECK entraîne l'envoi de la FPDU.ACK(SYN) par le PeSIT récepteur. L'entité PeSIT récepteur ne change pas d'état. Pour le PeSIT.F, la FPDU.ACK(SYN) est envoyée dans le flux de données typées de la couche session (S-TYPED-DATA).

c) Réception de la FPDU.ACK(SYN)

La réception d'une FPDU.ACK(SYN) valide par le PeSIT émetteur, alors qu'il est dans l'état "lecture du fichier" ou "écriture du fichier", entraîne la notification de la primitive de confirmation F.CKECK à l'utilisateur émetteur. L'entité PeSIT ne change pas d'état.

4.4.24 La FPDU.RESYN

La FPDU.RESYN est toujours envoyée par le PeSIT émetteur ou récepteur lorsqu'il celle-ci est dans l'état "lecture de fichier" ou "écriture de fichier" pour demander la reprise du transfert à partir d'un point de synchronisation.



a) Contenu de la FPDU.RESYN

Tous les paramètres de la primitive de demande F.RESTART, plus :

- ID.DST : identification de la connexion chez le destinataire.

b) Envoi de la FPDU.RESYN

La primitive de réponse F.RESTART entraîne l'envoi de la FPDU.RESYN par le PeSIT émetteur ou récepteur. L'entité PeSIT passe alors à l'état "resynchronisation en attente".

c) Réception de la FPDU.RESYN

La réception d'une FPDU.RESYN valide par le PeSIT émetteur ou récepteur, alors qu'il est dans l'état "lecture du fichier" ou "écriture du fichier", entraîne la notification de la primitive d'indication F.RESTART à l'utilisateur et le passage à l'état "resynchronisation en attente".

4.4.25 La FPDU.ACK(RESYN)

La FPDU.ACK(RESYN) est envoyée par le PeSIT émetteur ou récepteur lorsqu'il est dans l'état "synchronisation en attente" pour confirmer la resynchronisation.

a) Contenu de la FPDU.ACK(RESYN)

Tous les paramètres de la primitive de demande F.RESTART, plus :

- ID.DST : identification de la connexion chez le destinataire.

b) Envoi de la FPDU.ACK(RESYN)

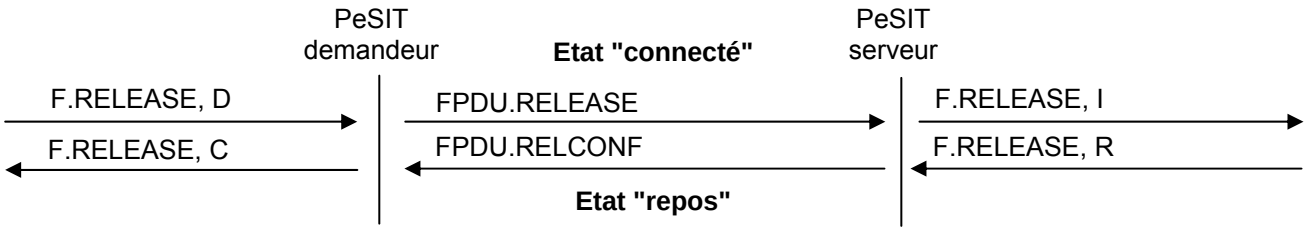
Une primitive de réponse F.RESTART entraîne l'envoi par le PeSIT de la FPDU.ACK(RESYN). Le PeSIT passe alors à l'état "lecture du fichier" ou "écriture du fichier".

c) Réception de la FPDU.ACK(SYN)

La réception d'une FPDU.ACK(RESYN) valide par le PeSIT alors qu'il est dans l'état "resynchronisation en attente", entraîne la notification de la primitive de confirmation F.RESTART à l'utilisateur et le passage à l'état "lecture du fichier" ou "écriture du fichier".

4.4.26 La FPDU.RELEASE

Comme la FPDU.CONNECT, la FPDU.RELEASE est toujours envoyée par le PeSIT demandeur lorsqu'il celle-ci est dans l'état "connecté" pour demander la libération de la connexion.



a) Contenu de la FPDU.RELEASE

Tous les paramètres de la primitive de demande F.RELEASE, plus :

- ID.DST : identification de la connexion chez le destinataire.
- ID.SRC : identification de la connexion chez l'expéditeur.

b) Envoi de la FPDU.RELEASE

Une primitive de demande F.RELEASE, entraîne l'envoi de la FPDU.RELEASE dans :

- le champ utilisateur de la primitive de demande de terminaison session S-RELEASE, par le PeSIT,
- le flux normal de données du service réseau N-DATA, par le PeSIT.

L'entité PeSIT passe alors à l'état "libération en attente".

c) Réception de la FPDU.RELEASE

La réception d'une FPDU.RELEASE valide par l'entité PeSIT, lorsqu'elle est dans l'état "connecté" entraîne la notification de la primitive d'indication F.RELEASE à l'utilisateur et le passage à l'état "libération en attente".

4.4.27 La FPDU.RELCONF

La FPDU.RELCONF est toujours envoyée par le PeSIT serveur lorsqu'il est dans l'état "libération en attente" pour confirmer la libération de la connexion.

a) Contenu de la FPDU.RELCONF

Tous les paramètres de la primitive de demande F.RELEASE, plus :

- ID.DST : identification de la connexion chez le destinataire,
- ID.SRC : identification de la connexion chez l'expéditeur,
- le flux normal de données du service réseau N-DATA, par le PeSIT.F' et l'enclenchement de la temporisation de surveillance du réseau T_r. (Voir 4.5.2)

b) Envoi de la FPDU.RELCONF

Une primitive de réponse F.RELEASE entraîne l'envoi de la FPDU.RELCONF dans :

- le champ utilisateur de la primitive de demande de réponse du service session S-RELEASE, par le PeSIT.F serveur,
- le flux normal de données du service réseau N-DATA, par le PeSIT.F' et l'enclenchement de la temporisation de surveillance du réseau T_r. (Voir 2.4.5.3 – FPDU.RCONNECT).

L'entité PeSIT passe alors à l'état "repos".

c) Réception de la FPDU.RELCONF

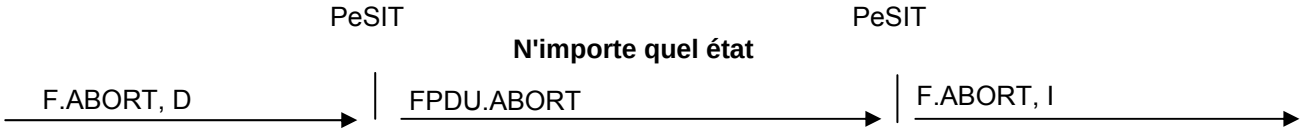
La réception d'une FPDU.RELCONF valide par l'entité PeSIT demandeur alors qu'il est dans l'état "libération en attente", entraîne la notification de la primitive d'indication F.RELEASE à l'utilisateur demandeur et :

- le passage à l'état "repos" de l'entité PeSIT,
- la demande de libération du service réseau par la primitive N-DISCONNECT par le PeSIT.F' et le passage à l'état "repos".

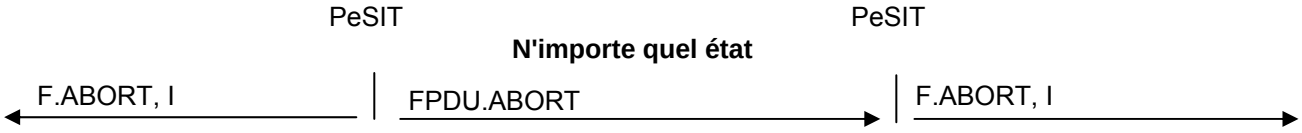
4.4.28 La FPDU.ABORT

La FPDU.ABORT est envoyée par le PeSIT demandeur ou serveur à n'importe quel moment du dialogue pour signaler la rupture brutale de la connexion. Le paramètre "diagnostic" précise la raison de la terminaison.

- Terminaison brutale à la demande de l'utilisateur (demandeur/serveur).



- Terminaison brutale à l'initiative du PeSIT ou du "système de communication".



a) Contenu de la FPDU.ABORT

Tous les paramètres de la primitive de demande F.ABORT, plus :

- ID.DST : identification de la connexion chez le destinataire,
- ID.SRC : identification de la connexion chez l'expéditeur.

b) Envoi de la FPDU.ABORT

- La primitive de réponse F.ABORT,
- ou la détection par le PeSIT
- ou la réception (cas PeSIT.F') d'une N-RESET indication venant de la couche réseau,

entraînent l'envoi d'une FPDU.ABORT dans :

- le champ utilisateur de la primitive de service session S-ABORT, par le PeSIT.F et le passage à l'état "repos",
- le flux normal de données du service réseau N-DATA, suivi de l'enclenchement du temporisateur de surveillance du réseau T_r par le PeSIT.F' (Voir 4.6).

La notification d'une primitive d'indication F.ABORT à l'utilisateur, est systématique.

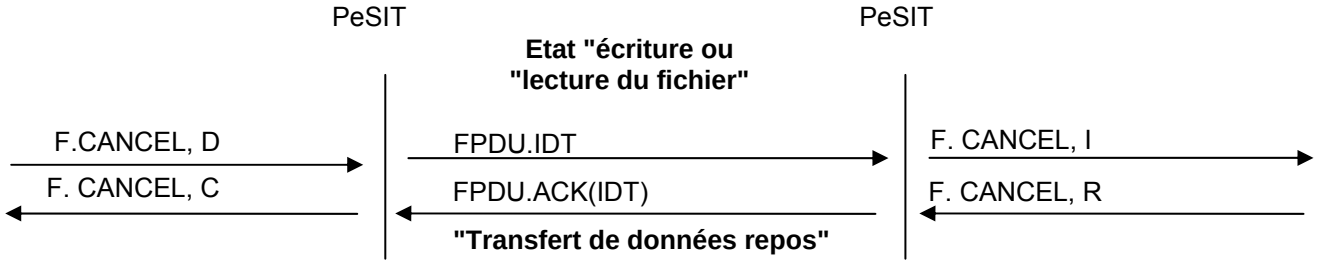
c) Réception d'une FPDU.ABORT

La réception d'une FPDU.ABORT par le PeSIT, entraîne la notification de la primitive d'indication F.ABORT à l'utilisateur et :

- le passage à l'état "repos" par le PeSIT,
- l'envoi de la demande de déconnexion du service réseau N-DISCONNECT et le passage à l'état "repos" par le PeSIT.F'.

4.4.29 La FPDU.IDT

La FPDU.IDT est envoyée par le PeSIT demandeur ou serveur lorsqu'il est dans l'état "lecture du fichier" ou "écriture du fichier" pour demander l'interruption de transfert. Le code fin de transfert indique la raison de l'interruption : "annulation", "suspension" ou "erreur".



a) Contenu de la FPDU.IDT

Tous les paramètres de la primitive F.CANCEL, plus :

- ID.DST : identification de la connexion chez le destinataire.

b) Envoi de la FPDU.IDT

Une primitive de demande F.CANCEL, entraîne l'envoi de la FPDU.IDT par le PeSIT émetteur ou récepteur pour demander l'interruption du transfert en cours. Le PeSIT passe alors à l'état "interruption de transfert en attente".

On utilise les données typées de la session (PeSIT.F).

c) Réception de la FPDU.IDT

La réception d'une FPDU.IDT par le PeSIT, alors qu'il est dans l'état "lecture de fichier" ou "écriture de fichier" entraîne la notification de la primitive d'indication F.CANCEL à l'utilisateur et le passage à l'état "interruption de transfert en attente".

4.4.30 La FPDU.ACK(IDT)

La FPDU.ACK(IDT) est envoyée par le PeSIT émetteur ou récepteur lorsqu'il est dans l'état "interruption de transfert en attente" pour confirmer la demande d'interruption de transfert.

a) Contenu de la FPDU.ACK(IDT)

Tous les paramètres de la primitive de demande F.CANCEL, plus :

- ID.DST : identification de la connexion chez le destinataire.

b) Envoi de la FPDU.ACK(IDT)

Une primitive de réponse F.CANCEL entraîne l'envoi par le PeSIT de la FPDU.ACK(IDT). Le PeSIT passe alors à l'état "transfert de données repos".

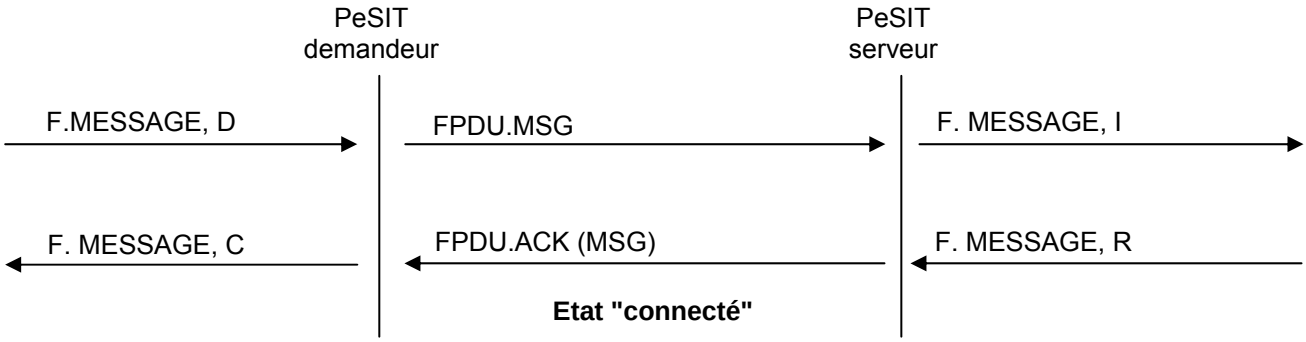
c) Réception de la FPDU.ACK(IDT)

La réception d'une FPDU.ACK(IDT) valide par le PeSIT alors qu'il est dans l'état "interruption de transfert en attente", entraîne la notification de la primitive de confirmation F.CANCEL à l'utilisateur et le passage à l'état "transfert de données repos".

4.4.31 Les FPDU.MSG, FPDU.MSGDM, FPDU.MSGMM, FPDU.MSGFM

4.4.31.1 La FPDU.MSG

La FPDU.MSG est toujours envoyée par le PeSIT demandeur alors qu'il est dans l'état "connecté" lorsque celui-ci désire l'émission d'un message à destination de l'entité PeSIT serveur homologue hors transfert de fichier.



a) Contenu de la FPDU.MSG

La FPDU.MSG contient tous les paramètres de la primitive de demande F.MESSAGE, plus :

- ID.DST : identification de la connexion chez le PeSIT destinataire.

b) Envoi de la FPDU.MSG

La primitive de demande F.MESSAGE entraîne l'envoi par l'entité PeSIT de la FPDU.MSG dans le flux de données normal du "système de communication". L'entité PeSIT demandeur passe dans l'état "libération de fichier en attente".

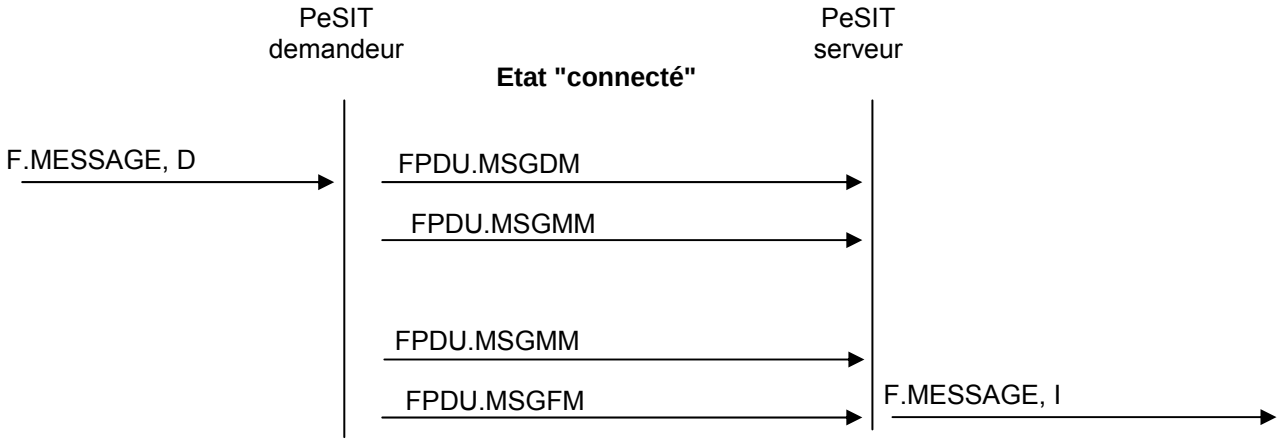
c) Réception de la FPDU.MSG

La réception d'une FPDU.MSG valide par le PeSIT serveur alors qu'il est dans l'état "connecté" entraîne la notification de la primitive d'indication F.MESSAGE à l'utilisateur serveur. Le PeSIT passe dans l'état "libération de fichier en attente".

4.4.31.2 SEGMENTATION DES MESSAGES

Lorsque la taille du message à transporter est telle que la taille de la FPDU dépasserait la taille maximale (taille de l'entité de données moins taille de l'en-tête FPDU), le message peut être segmenté et transporté par plusieurs FPDU :

- une FPDU.MSGDM (début de message)
- un nombre positif ou nul de FPDU.MSGMM (milieu d'article)
- une FPDU.MSGFM (fin de message).



a) Contenu des FPDU.MSGDM, FPDU.MSGMM, FPDU.MSGFM

Les FPDU.MSGDM, FPDU.MSGMM, FPDU.MSGFM contiennent chacune :

- une fraction du contenu du paramètre message de la primitive F.MESSAGE,
- ID.DST : identification de la connexion chez le destinataire.

b) Envoi des FPDU.MSGDM, FPDU.MSGMM, FPDU.MSGFM

Une primitive de demande F.MESSAGE entraîne l'envoi par le PeSIT demandeur, lorsqu'il est dans l'état "connecté" d'une FPDU.MSGDM, d'un nombre de FPDU.MSGMM positif ou nul, et d'une FPDU.MSGFM. L'entité PeSIT reste dans le même état.

c) Réception de la FPDU.MSGFM

La réception d'une FPDU.MSGFM valide par le PeSIT serveur alors qu'il est dans l'état "connecté" entraîne la notification de la primitive d'indication F.MESSAGE à l'utilisateur serveur. L'entité PeSIT passe à l'état "libération de fichier en attente".

4.4.32 La FPDU.ACK(MSG)

La FPDU.ACK(MSG) est toujours envoyée par le PeSIT serveur alors qu'il est dans l'état "libération du fichier en attente" pour indiquer le résultat de l'exécution de la demande de délivrance de message. Le paramètre "diagnostic" de la FPDU indique la raison exacte en cas de non succès. La FPDU.ACK(MSG) peut elle-même contenir un message à destination de l'entité PeSIT demandeur homologue.

a) Contenu de la FPDU.ACK(MSG)

La FPDU.ACK(MSG) contient tous les paramètres de la primitive de réponse F.MESSAGE, plus :

- ID.DST : identification de la connexion chez le PeSIT destinataire.

b) Envoi de la FPDU.ACK(MSG)

Une primitive de réponse F.MESSAGE entraîne l'envoi par l'entité PeSIT de la FPDU.ACK(MSG) dans le flux de données normal du "système de communication". L'entité PeSIT serveur passe dans l'état "connecté".

c) Réception de la FPDU.ACK(MSG)

La réception d'une FPDU.ACK(MSG) valide par le PeSIT demandeur alors qu'il est dans l'état "connecté", entraîne la notification de la primitive de confirmation F.MESSAGE à l'utilisateur demandeur. Le PeSIT demandeur passe à l'état "connecté".

4.5 CONCATENATION DES FPDU

Certaines FPDU peuvent être concaténées pour être envoyées dans la même entité de donnée de niveau inférieur (NSDU, SSDU,…), ceci dans la limite de la taille maximale négociée dans la phase de création ou de sélection du fichier.

Cette règle s'applique aux FPDU suivantes :

- FPDU.DTF
- FPDU.DTFDA
- FPDU.DTFMA
- FPDU.DTFFA
- FPDU.DTF.END
- FPDU.SYN.

4.6 LES TEMPORISATIONS DANS LE PROTOCOLE PeSIT

Plusieurs temporisations existent pour le traitement du protocole PeSIT.

1) Temporisation de surveillance Tp

Un des partenaires n'envoie plus rien alors qu'il est censé envoyer quelque chose. Cette anomalie peut être détectée par une fonction de temporisation locale "temporisation de surveillance de connexion : Tp". La seule possibilité dans ce cas est de mettre fin à la connexion avec un code diagnostic de dépassement de temps de surveillance.

Chez le serveur, la temporisation de surveillance Tp, sert à surveiller l'activité du PeSIT homologue :

- attente d'une FPDU parmi la liste suivante : ORF, READ, WRITE, DTF, DTFDA, DTFMA, DTFFA, DTF-END, SYN, RESYN, TRANS-END, CRF ou DESELECT.

Chez le demandeur, la temporisation de surveillance Tp sert à surveiller la réponse du PeSIT homologue aux FPDU ci-dessus, et la réponse aux FPDU.CONNECT et RELCONF.

La valeur de cette temporisation souhaitée par le demandeur pourra être transmise au serveur dans la FPDU.CONNECT (paramètre optionnel, à valeur par défaut 30 secondes).

2) Temporisation d'attente de déconnexion réseau : T_r (dans le cas de PeSIT.F')

*** Cas normal de déconnexion**

Le PeSIT.F' serveur enclenche le temporisation T_r et attend une primitive d'indication de déconnexion du serveur réseau "N-DISCONNECT,I". Si ce délai de temporisation T_r expire avant la réception de cette indication de déconnexion, le PeSIT serveur demande la déconnexion de la connexion du service réseau à l'aide d'une primitive de demande N-DISCONNECT et passe à l'état "repos". La temporisation T_r est arrêtée à la réception de la primitive d'indication de déconnexion du service réseau. Le PeSIT passe à l'état "Repos".

*** Utilisation du service F.ABORT**

Dans le cas de l'utilisation du service F.ABORT, le PeSIT émetteur de la FPDU-ABORT enclenche la temporisation T_r et attend une primitive de déconnexion du serveur réseau "N-DISCONNECT, I". Si ce délai de temporisation T_r expire avant la réception de cette indication de déconnexion. Le PeSIT émetteur de la FPDU.ABORT demande la déconnexion de la connexion du service réseau à l'aide d'une primitive de demande N-DISCONNECT et passe à l'état "repos". La temporisation T_r est arrêtée à la réception de la primitive d'indication de déconnexion du service réseau. Le PeSIT passe à l'état repos.

La valeur du délai de temporisateur de T_r (Temporisation de surveillance du réseau) est liée à la qualité de service et dépend de la réalisation locale du système.

Cette temporisation aura une valeur de l'ordre de **30 secondes**.

3) Attente d'une FPDU-CREATE, SELECT ou RELEASE (demande) : T_d

Cette temporisation sert à garder une connexion ouverte pour plusieurs transferts. Entre deux transferts, un certain temps peut s'écouler sans que la connexion soit fermée.

Cette temporisation aura une valeur de l'ordre de plusieurs minutes (> = 5 minutes).

Cette temporisation est armée chez le serveur uniquement.

4) Temporisation d'attente de connexion : Tc

Après l'établissement de la connexion réseau en PeSIT.F', le serveur arme la temporisation Tc afin de surveiller l'émission de la FPDU.CONNECT par le demandeur.

Cette temporisation aura une valeur de l'ordre de 30 secondes.

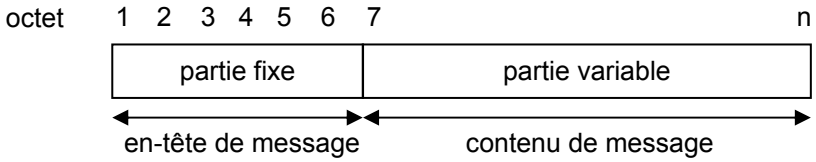
4.7 STRUCTURE ET CODAGE DES MESSAGES PeSIT (FPDU)

4.7.1 Structure de l'élément de protocole

Note : dans ce paragraphe, le terme message est pris au sens de "élément de protocole".

Chaque message PeSIT est composé de deux parties :

- un en-tête de message de 6 octets,
- un contenu de message de longueur variable :



a) L'en-tête de message

L'en-tête de message a la structure suivante :

Octets 1 et 2

Longueur totale du message (en-tête + contenu, en nombre d'octets).

Octet 3

40h : éléments de protocole de la phase connexion

00h : FPDU.DTF, FPDU.DTFDA, FPDU.DTFMA, FPDU.DTFFA

C0h : autres FPDU

Octet 4

Type de message

Octet 5

Identification chez le destinataire

Octet 6

FPDU de la phase connexion :	identification de la connexion chez l'expéditeur (ID.SRC)
FPDU.DTF mono article :	0
FPDU.DTF multi-articles :	nombre d'articles N (N> 1)
autres FPDU :	0

Dans le tableau qui suit, on prendra la valeur X comme identificateur de la connexion chez le demandeur et Y comme identificateur de la connexion chez le serveur. X et Y sont des valeurs numériques arbitraires différentes de zéro, attribuées à la connexion par le Demandeur et le Serveur.

La notification Y/X signifie que la valeur prise vaut Y si la FPDU est émise par le PeSIT Demandeur et X si elle est émise par le PeSIT Serveur.

PHASE	MESSAGE	Octet 3	Octet 4	Octet 5	Octet 6
Connexion	FPDU.CONNECT	40h	20	0	X
	FPDU.ACONNECT	40h	21	X	Y
	FPDU.RCONNECT	40h	22	X	0
	FPDU.RELEASE	40h	23	Y	X
	FPDU.RELCONF	40h	24	X	Y
	FPDU.ABORT	40h	25	Y/X (1)	X/Y
Sélection et libération de fichier	FPDU.CREATE	C0h	11	Y	0
	FPDU.ACK(CREATE)	C0h	30	X	0
	FPDU.SELECT	C0h	12	Y	0
	FPDU.ACK(SELECT)	C0h	31	X	0
	FPDU.DESELECT	C0h	13	Y	0
	FPDU.ACK(DESELECT)	C0h	32	X	0
	FPDU.MSG	C0h	16	Y	0
	FPDU.MSGDM	C0h	17	Y	0
	FPDU.MSGMM	C0h	18	Y	0
	FPDU.MSGFM	C0h	19	Y	0
	FPDU.ACK(MSG)	C0h	3B	X	0
Ouverture et fermeture de fichier	FPDU.ORF	C0h	14	Y	0
	FPDU.ACK(ORF)	C0h	33	X	0
	FPDU.CRF	C0h	15	Y	0
	FPDU.ACK(CRF)	C0h	34	X	0
Début et fin de transfert	FPDU.READ	C0h	01	Y	0
	FPDU.ACK(READ)	C0h	35	X	0
	FPDU.WRITE	C0h	02	Y	0
	FPDU.ACK(WRITE)	C0h	36	X	0
	FPDU.TRANS.END	C0h	08	Y	0
	FPDU.ACK(TRANS.END)	C0h	37	X	0
Transfert de données	FPDU.DTF	0	00	Y/X	0 (mono-article) N (multi-article)
	FPDU.DTFDA	0	41	Y/X	0
	FPDU.DTFMA	0	40	Y/X	0
	FPDU.DTFFA	0	42	Y/X	0
	FPDU.DTF.END	C0h	04	Y/X	0
	FPDU.SYN	C0h	03	Y/X	0
	FPDU.ACK(SYN)	C0h	38	Y/X	0
	FPDU.RESYN	C0h	05	Y/X	0
	FPDU.ACK(RESYN)	C0h	39	Y/X	0
Interruption du transfert	FPDU.ITF	C0h	06	Y/X	0
	FPDU.(ACK)IDT	C0h	3A	Y/X	0

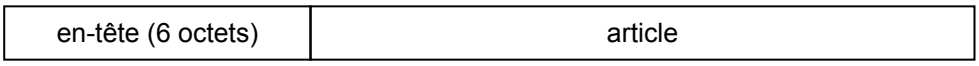
(1) Dans le cas où l'émission de la FPDU.ABORT intervient avant que l'ID.DST ne soit connu, elle est envoyée avec ID.DST = 0.

b) Le champ "contenu du message"

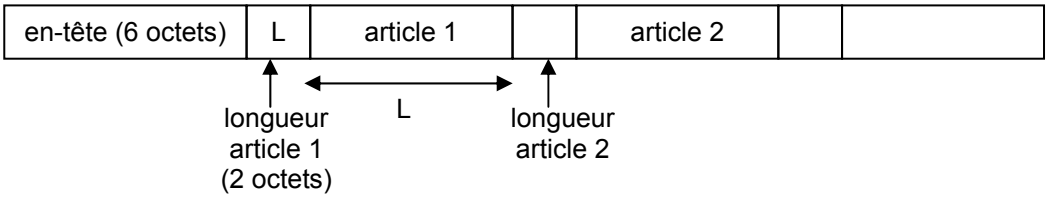
* FPDU.DTF, FPDU.DTFDA, FPDU.DTFMA, FPDU.DTFFA

Pour ces FPDU, le champ contient les données du fichier.

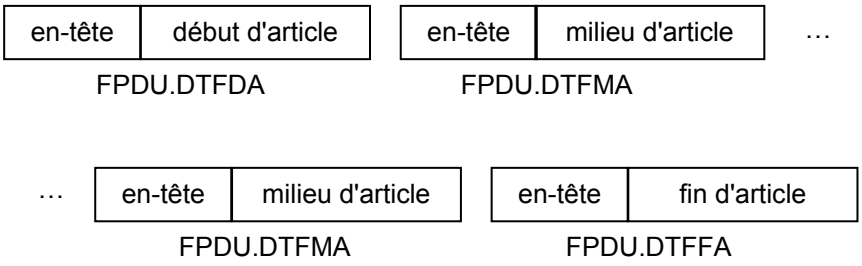
- Pour les FPDU.DTF mono-article (cas du SIT), ce champ contient un article complet :



- Pour les FPDU.DTF multi-articles, ce champ contient plusieurs articles :



- Un article d'une longueur supérieure à la taille de la SSDU-DATA est segmenté en plusieurs FPDU : une FPDU.DTFDA (début d'article), une ou plusieurs FPDU.DTFMA (milieu d'article) et une FPDU.DTFFA (fin d'article).



* AUTRES FPDU

Ce champ contient les paramètres du message, identifiés chacun par un PI (identificateur de paramètre) lesquels sont regroupés en groupes de paramètres, identifiés par un PGI (identificateur de groupe de paramètres). Les unités de PGI et de PI doivent être ordonnées par valeurs croissantes de leurs codes PGI et PI.

Note : le PGI 9 peut contenir les PI 3 et PI 4. Le code PGI 9 se trouve alors avant les codes PI 3 et PI 4 qu'il contient.

La représentation des paramètres est décrite ci-après.

4.7.2 Codage des paramètres

4.7.2.1 Conventions de codage

a) Unités de PGI

Les unités PGI contiennent, dans l'ordre suivant :
a) le champ PG qui identifie le groupe des paramètres
b) le champ LI qui indique la longueur du champ de paramètre associé,
c) le champ de paramètre consiste :

- soit en une seule valeur de paramètre (voir note),
- soit en une ou plusieurs unités de PI.

NOTE :
Une unité de PGI contenant un seul paramètre est structurellement équivalente à une unité de PI.

b) Unités de PI

Les unités PI contiennent, dans l'ordre suivant :
a) le champ PI qui identifie le paramètre
b) le champ LI qui indique la longueur du champ de paramètre associé,
c) le champ de paramètre qui contient la valeur du paramètre.

c) Champ d'identificateur PGI et PI

Les champs PGI et PI comprennent chacun un octet et contiennent respectivement un code de PGI ou de PI. Les codes de PGI et PI sont exprimés par des nombres décimaux dans la liste du § 4.7.3 et doivent être codés sous forme de nombre binaires.

d) Champ indicateur de longueur LI

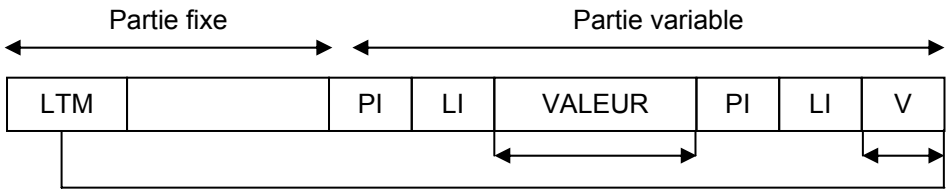
La valeur du champ LI est exprimée par un nombre binaire représentant la longueur, en octets, du champ de paramètre associé (voir Note). La valeur zéro est rejetée par le protocole.

Les champs LI indiquant des longueurs comprises entre 1 et 254 doivent contenir un octet.

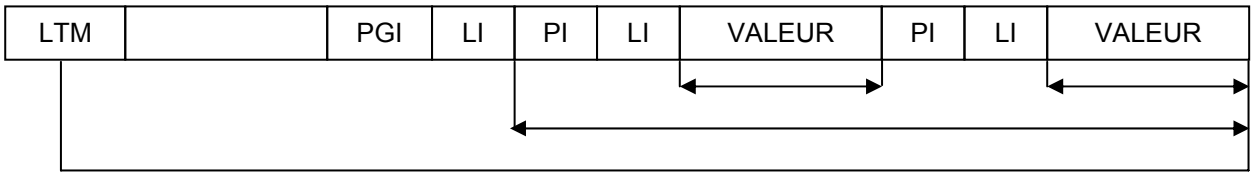
Les champs LI indiquant des longueurs comprises entre 255 et 65535 doivent contenir trois octets. Le codage du premier octet 1111 1111 et les second et troisième octets doivent contenir la longueur du champ de paramètre associé, les bits de plus haut poids étant dans le premier de ces deux octets.

Note : La valeur du champ LI ne comprend pas la longueur de ce champ lui-même.

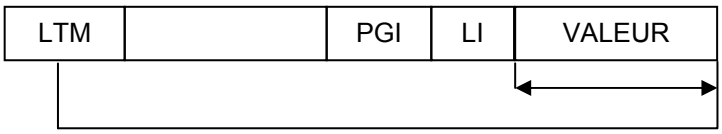
*** Quelques exemples de structure des messages**



LTM* : 14 octets, avec 2 paramètres, respectivement de 3 et 1 octet.



LTM = 17 octets, avec 1 PGI contenant 2 PI, respectivement de 2 et 3 octets.



LTM = 9, avec un PGI sans PI, de valeur de longueur 1.

* Longueur totale du message = LTM.

e) Représentation des valeurs des paramètres

Pour décrire complètement les valeurs des paramètres, il est nécessaire de définir les types de valeurs suivants qui sont utilisés pour les divers paramètres dans les tableaux de codage :

.C : Chaîne de caractères. Longueur quelconque (sauf si des limites sont définies) ; le codage des caractères utilise le code ASCII 7bits. Les blancs à droite ne sont pas significatifs et une valeur totalement à blanc est nulle.

.N : Numérique. Entier binaire sans signe.

.S : Symbolique. Comme N, c'est un entier binaire sans signe sauf que les valeurs ont des significations particulières. Longueur de 8 bits.

.M : Masque de bits. Chaîne dans laquelle chaque bit a une signification particulière. Longue de 8 à 16 bits. Tout bit dont le rôle n'est pas défini doit avoir la valeur zéro. Pour les bits représentant des options particulières, le bit a la valeur "1" si l'option est demandée, et la valeur "0" dans le cas contraire.

.D : Date et heure. La codification est celle indiquée dans les normes ISO 2014 et ISO 3307 : AAMMJJ et hhmmss, dans lesquelles AA = année, MM = mois, JJ = Jour, hh = heure, mm = minute, ss = seconde. Elle s'écrit sous forme d'une chaîne de caractères (voir ci-dessus) avec une longueur de 12 octets.

.A : Agrégat. Composé de zones de types ci-dessus (exemple : 2 zones, une numérique et l'autre symbolique).

f) Optimisation de la zone valeur

Il est recommandé d'utiliser la taille minimum des chaînes de caractères de longueur variable pour les valeurs d'expression.

Les chaînes de caractères ne doivent pas contenir d'espaces inutiles sauf si le champ est de longueur fixe.

Pour les données numériques et symboliques, tous les octets de gauche ne contenant que des bits à zéro doivent être éliminés. Néanmoins, la longueur minimale d'une chaîne est de un octet.

g) Omission des paramètres

Certains paramètres peuvent être omis. On distingue :

- Les paramètres obligatoires dont l'absence dans une FPDU est une erreur de protocole.

- Les paramètres optionnels pour lesquels une valeur implicite est définie. Cette valeur est alors soulignée dans la description du paramètre (§ 4.7.3). Si le paramètre est omis, il sera considéré comme ayant la valeur implicite.

- Les paramètres optionnels sans valeur implicite, qui peuvent être présents ou non dans la FPDU.

h) Répétition des paramètres

Sauf indication explicite dans la description des services, un paramètre donné ne peut figurer qu'une seule fois dans un message donné. En cas de répétition illégale, seule la première apparition est conservée, et une erreur est signalée.

4.7.2.2 Liste des codes PGI et PI

Les différents codes PI utilisés sont :

Code PI	Description des paramètres
1	Utilisation d'un CRC
2	Diagnostic
3	Identification Demandeur
4	Identification Serveur
5	Contrôle d'accès
6	Numéro de version
7	Option points de synchronisation
11	Type du fichier
12	Nom du fichier
13	Identificateur du transfert
14	Attributs demandés
15	Transfert relancé
16	Code-données
17	Priorité de transfert
18	Point de relance
19	Code fin de transfert
20	Numéro du point de synchronisation
21	Compression
22	Type d'accès
23	Resynchronisation
25	Taille maximale d'une entité de données
26	Temporisation de surveillance
27	Nombre d'octets de données
28	Nombre d'articles
29	Compléments de diagnostic
31	Format d'article
32	Longueur d'article
33	Organisation du fichier
34	Prise en compte de la signature
36	Sceau SIT
37	Label du fichier
38	Longueur de la clé
39	Déplacement de la clé dans l'enregistrement
41	Unité de réservation d'espace
42	Valeur maximale de réservation d'espace
51	Date et heure de création
52	Date et heure de dernière extraction
61	Identificateur Client
62	Identificateur Banque
63	Contrôle d'accès fichier
64	Date et heure du serveur

71 Type d'authentification
72 Eléments d'authentification
73 Type de scellement
74 Eléments de scellement
75 Type de chiffrement
76 Eléments de chiffrement
77 Type de signature
78 Sceau
79 Signature
80 Accréditation
81 Accusé de réception de la signature
82 Deuxième signature
83 Deuxième accréditation
91 Message
99 Message libre

Les différents PGI utilisés sont :

Dénomination	**Code PGI**	**PI**
Identificateur de fichier	9	3 4 11 12
Attributs logiques	30	31 32 33 34 36 37 38 39
Attributs physiques	40	41 42
Attributs historiques	50	51 52

4.7.3 Description des paramètres

Les pages suivantes donnent la codification détaillée des paramètres, par profil. Pour chaque paramètre, sont indiqués :

- Le nom du paramètre,
- Le type de valeur (C, N, S, M, D, A),
- La longueur maximale de la zone valeur en octets,
- La codification des valeurs admises du paramètre (la valeur implicite, si elle existe, est soulignée),
- Le caractère "optionnel" ou "optionnel avec valeur implicite" tel qu'il est défini dans le § 4.7.2.1.g.

Le paramètre peut également être :

- "conditionnel" : le paramètre est obligatoire si la condition est remplie et doit être omis si la condition n'est pas remplie
- "conditionnel avec valeur implicite " : si la condition est remplie et le paramètre omis, il est considéré comme ayant la valeur implicite.

Si aucune de ces mentions n'est indiquée, le paramètre est obligatoire.

Nota : lorsque le format d'un paramètre n'est pas précisé pour le profil PeSIT Hors-SIT sécurisé, il est identique à celui du profil Hors-SIT.

UTILISATION D'UN CRC 1

Type : S **Longueur : 1** **paramètre optionnel avec valeur implicite**

SIT

Interdit

HORS-SIT

0 = Pas d'utilisation du CRC
1 = Utilisation d'un CRC

L'utilisation du CRC est obligatoire dans le cas de l'utilisation de PeSIT.F' à travers un accès asynchrone (accès PAD).

ETEBAC5

Idem HORS-SIT

DIAGNOSTIC 2

Type : A **Longueur : 3**

SIT

Diagnostic (voir ANNEXE D)

Octet 1 : type d'erreur

Octets 2-3 : code raison

HORS-SIT

Idem SIT

ETEBAC5

Idem SIT

IDENTIFICATEUR DU DEMANDEUR 3

Paramètre optionnel sauf dans la FPDU.CONNECT

SIT

Type : A **Longueur : 3**

Octet 1 (S) : type d'application

Valeur = 1 : CTE

Valeur = 2 : CTR

Valeur = 3 : IE

Valeur = 4 : IR

Octets 2 et 3 (N) : Numéro de l'application (valeur attribuée par le Centre de Gestion)

HORS-SIT

Type : C **Longueur : 24**

Identificateur du Demandeur (cf. annexe B : mode store and forward, pour utilisation particulière)

ETEBAC5

Type : C **Longueur : 24**

Identificateur du demandeur :

Octets 1 à 3 : type identifiant (C)

valeur = ZZZ : convenu avec la banque

valeur = 005 : standard CFONB

Octets 4 à 24 : identifiant (C)

si identifiant standard

- numéro SIRET
- identifiant service et individu

IDENTIFICATEUR DU SERVEUR 4

Paramètre optionnel sauf dans la FPDU.CONNECT

SIT

Type : A Longueur : 3

Octet 1 (S) : type d'application

Valeur = 1 : CTE

Valeur = 2 : CTR

Valeur = 3 : IE

Valeur = 4 : IR

Octets 2 et 3 (N) : Numéro de l'application (valeur attribuée par le Centre de Gestion)

HORS-SIT

Type : CLongueur : 24

Identificateur du serveur (cf. annexe B : mode store and forward, pour utilisation particulière)

ETEBAC5

Type : C Longueur : 24

Identificateur du demandeur :

Octets 1 à 3 : type identifiant (C)

valeur = ZZZ : convenu avec la banque

valeur = 005 : standard CFONB

valeur = ZBF : code BdF (banque + guichet)

Octets 4 à 24 : identifiant (C)

si identifiant standard

- numéro SIRET
- identifiant service et individu

CONTRÔLE D'ACCES 5

SIT

Type : C **Longueur : 2**

Inutilisé

HORS-SIT

Type : C **Longueur : 16**

Paramètre optionnel,

Octets 1 à 8 : mot de passe partenaire

Octets 9 à 16 : nouveau mot de passe partenaire

(si octets 9 à 16 absents : mot de passe inchangé)

ETEBAC5

Idem HORS-SIT.

NUMERO DE VERSION 6

Type : C **Longueur : 2**

SIT

Numéro de version du PeSIT = 1

HORS-SIT

Numéro de version du PeSIT

valeur 1 : version D du 15 novembre 1987

valeur 2 : version E du 14 juillet 1989

ETEBAC5

Idem HORS-SIT.

OPTION POINTS DE SYNCHRONISATION 7

Type : A **Longueur : 3** paramètre optionnel avec valeur implicite

---

SIT

* Octets 1 et 2 : intervalle entre points de synchronisation en nombre de Koctets (N)

  Valeurs spéciales :
  0 = pas de points de synchronisation
  Tous bits à 1 = intervalle non défini
  L'intervalle le plus petit l'emporte sur le plus grand ou le non défini.

* Octet 3 : fenêtre d'acquittement (N) (si octets 1 et 2 différents de zéro)

  Valeurs spéciales :
  0 = pas d'acquittement des points de synchronisation.

  L'intervalle entre points de synchronisation doit être supérieur ou égal à 4 Koctets.
  La fenêtre de synchronisation doit être inférieure ou égale à 16.

HORS-SIT

* Octets 1 et 2 : intervalle entre points de synchronisation en nombre de Koctets (N)

  Valeurs spéciales :
  0 = pas de points de synchronisation
  Tous bits à 1 = intervalle non défini
  L'intervalle le plus petit l'emporte sur le plus grand ou le non défini.

* Octet 3 : fenêtre d'acquittement (N) (si octets 1 et 2 différents de zéro)
  Valeurs spéciales :
  0 = pas d'acquittement des points de synchronisation.

ETEBAC5

Idem HORS-SIT.

TYPE DU FICHIER 11

SIT

Type : N Longueur : 2

La liste de tous les types de fichiers a été spécifiée précisément dans les documents de spécification du réseau SIT.

HORS-SIT

Type : N Longueur : 2

Valeur = 0 : pas d'action spécifique au niveau du moniteur
Autre valeur : utilisée pour une action spécifique entre deux moniteurs.

Cas de la FPDU.MSG :

valeur FFFF (hexadécimal) : message aller

valeur FFFE (hexadécimal) : message retour

autres valeurs : acquittement de réception de fichier

ETEBAC5

Type : C Longueur : 8

Octet 1 : type codage (C)

valeur = 0 : convenu avec la banque

valeur = 1 : standard CFONB

Octet 2 : type de syntaxe (C)

valeur = 0 : enregistrements CFONB

valeur = 1 : messages EDIFACT

Octet 3 à 7 : nature d'application (C)
(Cf. Annexe A2 de "Echanges Télématiques Entre les Banques et leurs Clients")

Octet 8 : nature de transfert (C)
(Cf. Annexe A2 de "Echanges Télématiques Entre les Banques et leurs Clients")

NOM DU FICHIER 12

SIT

Type : C Longueur : 5

Chaîne ASCII numérique.

HORS-SIT

Type : C Longueur : 14

Octet 1 : type identifiant (C)

valeur = 0 : convenu avec la banque

valeur = 1 : standard CFONB

IDENTIFIANT :

si identifiant standard

Octet 2 : type de référence (C)

valeur = 0 : référence nominative

valeur = 1 : demande dernière version

valeur = 2 : demande versions non transmises

valeur = 3 : demande toutes versions

Format 1
(type = 0)

Octets 3 à 14 : référence du fichier (C)

Format 2
(type =1, 2, 3)

Octets 3 à 8 : date de début : AAMMJJ (D)

Octets 9 à 14 : date de fin : AAMMJJ (D)

IDENTIFICATEUR DU TRANSFERT 13

Type : N **Longueur : 3**

SIT

Valeur numérique choisie par le demandeur

HORS-SIT

Dans la FPDU.CREATE
Valeur numérique non nulle choisie par le demandeur.
Dans le cas d'un transfert relancé, elle doit être identique à celle utilisée pour la tentative de transfert précédente.

Dans la FPDU.ACK(CREATE) : paramètre optionnel
Valeur numérique non nulle choisie par le serveur.

Dans la FPDU.SELECT
Valeur = 0 pour un transfert nouveau.
Dans le cas d'une relance, la valeur est celle fixée par le Serveur lors de la précédente tentative.

Dans la FPDU.ACK(SELECT)
Valeur numérique non nulle choisie par le serveur.
(Dans le cas d'une relance, valeur identique à celle présente dans la FPDU.SELECT).

ETEBAC5

Idem HORS-SIT.

ATTRIBUTS DEMANDES 14

Type : M **Longueur : 1** paramètre optionnel avec valeur implicite

SIT

Inutilisé

HORS-SIT

Le bit est mis à 1 si les attributs correspondants sont désirés :
b1 : attributs logiques
b2 : attributs physiques
b3 : attributs historiques
b4-b8 : doivent recevoir la valeur 0 et être ignorés sinon
valeur implicite = pas d'attributs

Dans la FPDU.MSG :
Valeur = 0 pas de message attendu dans la FPDU.ACK(MSG)
Valeur = 1 message attendu dans la FPDU.ACK(MSG)

ETEBAC5

Idem HORS-SIT

TRANSFERT RELANCE 15

Type : S **Longueur : 1** paramètre optionnel avec valeur implicite

---

SIT

0 = transfert nouveau
1 : transfert relancé

HORS-SIT

Idem SIT

ETEBAC5

Idem SIT

CODE-DONNEES 16

Type : S **Longueur : 1** paramètre optionnel avec valeur implicite

---

SIT

Inutilisé

HORS-SIT

Type : S **Longueur : 1**

0 = ASCII

1 = EBCDIC

2 = binaire

> 2 : ne doit pas être utilisé

ETEBAC5

Idem HORS-SIT

PRIORITE DU TRANSFERT 17

Type : S **Longueur :**

SIT

0 = priorité 0 (urgent)

1 = priorité 1 (moyennement urgent)

2 = priorité 2 (moins urgent)

HORS-SIT

Idem SIT

ETEBAC5

Idem HORS-SIT

POINT DE RELANCE 18

Type : N **Longueur : 3**

SIT

Point de relance (0 = début de fichier)

HORS-SIT

Idem SIT

ETEBAC5

Idem SIT

CODE FIN DE TRANSFERT 19

Type : S **Longueur : 1**

SIT

4 = erreur (une relance doit suivre)

8 = suspension

12 = annulation (par le Serveur)

16 = annulation (par Demandeur)

HORS-SIT

Idem SIT

ETEBAC5

Idem SIT

NUMERO DU POINT DE SYNCHRONISATION 20

Type : N **Longueur : 3**

SIT

Numéro du point de synchronisation (0 = début de fichier)

HORS-SIT

Idem SIT

ETEBAC5

Idem SIT

COMPRESSION 21

Type : A **Longueur : 2** paramètre optionnel avec valeur implicite

SIT

Inutilisé

HORS-SIT

Octet 1 : compression des données

0 = non

1 = oui

Octet 2 : type de compression

1 = compression horizontale

2 = compression verticale

3 = compression des compressions horizontale et verticale

ETEBAC5

Idem HORS-SIT

TYPE D'ACCES 22

Type : S **Longueur : 1**

SIT

0 = écriture

HORS-SIT

0 = écriture

1 = lecture

2 = mixte (écriture ou lecture)

ETEBAC5

Idem HORS-SIT

RESYNCHRONISATION 23

Type : S **Longueur : 1** paramètre optionnel avec valeur implicite

SIT

Inutilisé

HORS-SIT

0 = F.RESTART non autorisé

1 = F.RESTART autorisé

ETEBAC5

Idem HORS-SIT

TAILLE MAXIMALE D'UNE ENTITE DE DONNEES 25

Type : N **Longueur : 2**

SIT

Longueur en octets de la taille maximale d'une entité de données. Cette longueur doit être supérieure à 800 octets. Une taille d'entité de données de 4 050 octets doit pouvoir être supportée.

HORS-SIT

Longueur en octets de la taille maximale d'une entité de données.

ETEBAC5

Idem HORS-SIT

TEMPORISATION DE SURVEILLANCE 26

Type : N **Longueur : 2** paramètre optionnel avec valeur implicite

SIT

Interdit

HORS-SIT

Inutilisé

ETEBAC5

Temporisation de surveillance du protocole (en secondes)
30 = 30 secondes

NOMBRE D'OCTETS DE DONNEES 27

Type : N **Longueur : 8** paramètre optionnel

SIT

Interdit

HORS-SIT

Nombre d'octets de données (hors octet longueur dans le cas FPDU multi-articles), mais incluant les octets d'en-tête de chaîne de compression s'il y a lieu.
Paramètre obligatoire si accès PAD.

ETEBAC5

Idem HORS-SIT

NOMBRE D'ARTICLES 28

Type : N **Longueur : 4** paramètre optionnel

SIT

Interdit

HORS-SIT

Nombre d'articles
Paramètre obligatoire si accès PAD.

ETEBAC5

Idem HORS-SIT

COMPLEMENTS DE DIAGNOSTIC 29

Type : A **Longueur : 254** paramètre optionnel

SIT

Interdit

HORS-SIT

Paramètre optionnel de format libre

ETEBAC5

Format 1 (diagnostic = 310)

Octet 1 : cause X25 (N)

Octet 2 : diagnostic X25 (N)

Format 2 (diagnostic = 318)

Octet 1 : nombre PI incorrects (N)

Octet 2 : code PI (N)

Octet 3 : code erreur (S)

Valeur = 1 : PI absent

Valeur = 2 : erreur de syntaxe

Valeur = 3 : valeur non supportée

Valeur = 4 : valeur hors borne

Octet 4 à 23 : libellé

Format 3 (diagnostic = 321)

Octet 1 à 15 : numéro de secours (N)

Format 4 (diagnostic = 322)

Octet 1 à 6 : délai de réappel : HHMMSS (D)

FORMAT D'ARTICLE 31

Type : M **Longueur : 1** paramètre optionnel avec valeur implicite

---

SIT

Valeur = 0 : fixe

Valeur = 80h : variable

HORS-SIT

Idem SIT

ETEBAC5

Idem SIT

LONGUEUR D'ARTICLE 32

Type : N **Longueur : 2**

SIT

Longueur d'article (en octets)

HORS-SIT

Idem SIT

ETEBAC5

Idem SIT

ORGANISATION DU FICHIER 33

Type : S **Longueur : 1** paramètre optionnel avec valeur implicite

---

SIT

0 = séquentiel

1 = relatif

2 = indexé

HORS-SIT

Idem SIT

ETEBAC5

Idem SIT

PRISE EN COMPTE DE LA SIGNATURE 34

Type : N **Longueur : 2** paramètre optionnel avec valeur implicite

---

SIT

0 = pas de signature

1= fichier scellé par le SIT

Dans le sens CTB vers station, paramètre absent ou valeur = 0
Dans le sens station vers CTB, valeur = 1

HORS-SIT

Inutilisé

ETEBAC5

Inutilisé

SCEAU SIT 36

Type : N **Longueur : 64** paramètre conditionnel

SIT

Présent si le PI 34 vaut 1

HORS-SIT

Interdit

ETEBAC5

Interdit

LABEL DU FICHIER 37

Type : C **Longueur : 80** paramètre optionnel

SIT

Label du fichier

HORS-SIT

Label du fichier

ETEBAC5

Label du fichier

LONGUEUR DE LA CLE 38

Type : N **Longueur : 2** paramètre conditionnel

SIT

Interdit

HORS-SIT

Paramètre présent si le fichier est indexé (PI 33 = 2).

ETEBAC5

Idem HORS-SIT

DEPLACEMENT DE LA CLE 39

Type : N **Longueur : 2** paramètre conditionnel avec valeur implicite

---

SIT

Interdit

HORS-SIT

Déplacement en octets de la clé dans l'article.

Paramètre présent si le fichier est indexé (PI 33 = 2)

Valeur implicite = 0

ETEBAC5

Idem HORS-SIT

UNITE DE RESERVATION D'ESPACE 41

Type : S **Longueur : 1** paramètre optionnel avec valeur implicite

---

SIT

Unité de réservation d'espace :

0 = Koctets

1 = articles

L'unité de réservation d'espace doit être le kilo-octet si le fichier est de format variable (PI 31 = 80h)

HORS-SIT

Idem SIT

ETEBAC5

Idem SIT

VALEUR MAXIMALE DE RESERVATION D'ESPACE 42

Type : N **Longueur : 4**

SIT

Valeur maximale de réservation d'espace

HORS-SIT

Idem SIT

ETEBAC5

Idem SIT

DATE ET HEURE DE CREATION 51

Type : D **Longueur : 12**

SIT

Date et heure de création (les dates et heures sont celles du SIT)

Octets 1 à 6 : date (AAMMJJ)
Octets 7 à 12 : heure (HHMMSS)

HORS-SIT

Date et heure de création

ETEBAC5

Idem HORS-SIT

DATE ET HEURE DE DERNIERE EXTRACTION 52

Type : D **Longueur : 12** paramètre optionnel

SIT

Date et heure de la dernière extraction

Octets 1 à 6 : date (AAMMJJ)

Octets 7 à 12 : heure (HHMMSS)

HORS-SIT

Idem SIT

ETEBAC5

Inutilisé

IDENTIFICATEUR CLIENT 61

Type : C **Longueur : 24**

SIT

Interdit

HORS-SIT

Paramètre optionnel :
Identificateur de l'émetteur initial (Cf. annexe B : mode store and forward, pour utilisation particulière)

ETEBAC5

Paramètre obligatoire :

Identificateur client :

Octets 1 à 3 : type identifiant (C)
Valeur = ZZZ : convenu avec la banque
Valeur = 005 : standard CFONB

Octets 4 à 24 : identifiant (C)

si identifiant standard :

- numéro SIRET
- Identifiant service et individu

IDENTIFICATEUR BANQUE 62

Type : C **Longueur : 24**

SIT

Interdit

HORS-SIT

Paramètre optionnel :
Identificateur destinataire final (Cf. annexe B : mode store and forward, pour utilisation particulière)

ETEBAC5

Paramètre obligatoire :

Identificateur banque :

Octets 1 à 3 : type identifiant (C)
Valeur = ZZZ : convenu avec la banque
Valeur = 005 : standard CFONB
Valeur = ZBF : code BdF (banque + guichet)

Octets 4 à 24 : identifiant (C)

si identifiant standard :

- numéro SIRET
- Identifiant service et individu

CONTRÔLE D'ACCES FICHIER 63

Type : C **Longueur : 16** paramètre optionnel

SIT

Interdit

HORS-SIT

Inutilisé

ETEBAC5

Octets 1 à 8 : mot de passe client
Octets 9 à 16 : nouveau mot de passe

DATE ET HEURE DU SERVEUR 64

Type : D **Longueur : 12**

SIT

Interdit

HORS-SIT

Intilisé

ETEBAC5

Octets 1 à 6 : date (AAMMJJ)
Octets 7 à 12 : heure (HHMMSS)

TYPE D'AUTHENTIFICATION 71

Type : A **Longueur : 3** paramètre optionnel avec valeur implicite

SIT

Interdit

HORS-SIT

Interdit

HORS-SIT sécurisé

Octet 1 : présence d'authentification (S)
0 = pas d'authentification
1 = authentification

Octet 2 : algorithme utilisé (S)
0 = RSA
1 = DES

Octet 3 : mode opératoire (S)
0 = authentification simple
1 = authentification réciproque
2 = authentification réciproque DES seul

ETEBAC5

Octet 1 : présence d'authentification (S)
0 = pas d'authentification
1 = authentification

Octet 2 : algorithme utilisé (S)
0 = RSA

Octet 3 : mode opératoire (S)
0 = authentification simple
1 = authentification réciproque

ELEMENTS D'AUTHENTIFICATION 72

Type : N **Longueur : n** paramètre conditionnel

SIT

Interdit

HORS-SIT

Interdit

HORS-SIT sécurisé

Eléments d'authentification

Paramètre présent si PI 71, octet 1 = 1 et PI 71, octet 3 différent de 0.

Si PI 71, octet 3 = 1 : Cf. Profil ETEBAC 5

Si PI 71, octet 3 = 2

Dans CREATE/SELECT :	KEKNAME1 nom de la clé de chiffrement KEK1 : 8 octets KAUTH1 clé d'authentification chiffrée sous KEK1 : 8 octets ALEA1 chiffré DES sous KAUTH1 : 8 octets
Dans ACK(CREATE/ SELECT) :	KEKNAME2 nom de la clé de chiffrement KEK2 : 8 octets KAUTH2 clé d'authentification chiffrée sous KEK2 : 8 octets ALEA1* chiffré DES sous KAUTH2 : 8 octets ALEA2 chiffré DES sous KAUTH2 : 8 octets
Dans ORF :	ALEA2* chiffré DES sous KAUTH2 : 8 octets

* : Cf. Annexe C : utilisation des mécanismes de sécurité

ETEBAC5

Eléments d'authentification.
Paramètre présent si PI 71, octet 1 = 1 et PI 71, octet 3 = 1.

Dans CREATE/SELECT :	ALEA1 : 8 octets
Dans ACK(CREATE/ SELECT) :	ALEA1 chiffré RSA sous clé secrète émetteur : 64 octets ALEA2 : 8 octets
Dans ORF :	ALEA2 chiffré RSA sous clé secrète émetteur : 64 octets.

TYPE DE SCELLEMENT 73

Type : A **Longueur : 4** paramètre optionnel avec valeur implicite

SIT

Interdit

HORS-SIT

Interdit

HORS-SIT

Octet 1 : présence de scellement (S)
0 = pas de scellement
1 = scellement

Octet 2 : algorithme utilisé (S)
1 = DES

Octet 3 : mode opératoire (S)
1 = transmission de sceaux partiels, calcul article par article
2 = transmission du sceau final uniquement, calcul article par article
3 = transmission de sceaux partiels, calcul sur la totalité du fichier
4 = transmission du sceau final uniquement, calcul sur la totalité du fichier

Octet 4 : transfert des éléments de scellement (S)
1 = pas de transfert des éléments de scellement
2 = transfert des éléments de scellement, en clair
3 = transfert des éléments de scellement, chiffré RSA
4 = transfert des éléments de scellement, chiffré DES

ETEBAC5

Octet 1 : présence de scellement (S)
0 = pas de scellement
1 = scellement

Octet 2 : algorithme utilisé (S)
1 = DES

Octet 3 : mode opératoire (S)
1 = transmission de sceaux partiels, calcul article par article
2 = transmission du sceau final uniquement, calcul article par article
3 = transmission de sceaux partiels, calcul sur la totalité du fichier
4 = transmission du sceau final uniquement, calcul sur la totalité du fichier

Octet 4 : transfert des éléments de scellement (S)
0 = pas de transfert des éléments de scellement
1 = transfert des éléments de scellement, en clair
2 = transfert des éléments de scellement, chiffré RSA

ELEMENTS DE SCELLEMENT 74

Type : N **Longueur : n** paramètre conditionnel

SIT

Interdit

HORS-SIT

Interdit

HORS-SIT sécurisé

Eléments de scellement

Paramètre présent si PI 73, octet 1 = 1 et PI 73, octet 4 différent de 0.

Si PI 73, octet 4 = 1 : clé de scellement K2 : 8 octets
vecteur d'initialisation IV2

Si PI 73, octet 4 = 2 : clé de scellement K2, vecteur d'initialisation IV2 chiffrés
RSA sous clé publique destinataire : 64 octets

Si PI 73, octet 4 = 3 : KEKNAME nom de la clé de chiffrement de clé KEK : 8 octets
clé de scellement K2 chiffrée DES sous KEK : 8 octets
vecteur d'initialisation IV2 chiffré DES sous KEK

ETEBAC5

Eléments de scellement

Paramètre présent si PI 73, octet 1 = 1 et PI 73, octet 4 différent de 0.

Si PI 73, octet 4 = 1 : clé de scellement K2 : 8 octets
vecteur d'initialisation IV2

Si PI 73, octet 4 = 2 : clé de scellement K2, vecteur d'initialisation IV2 chiffrés
RSA sous clé publique destinataire : 64 octets

TYPE DE CHIFFREMENT 75

Type : A **Longueur : 4** paramètre optionnel avec valeur implicite

SIT

Interdit

HORS-SIT

Interdit

HORS-SIT sécurisé

Octet 1 : présence de chiffrement (S)
0 = pas de chiffrement
1 = chiffrement

Octet 2 : algorithme utilisé (S)
0 = RSA
1 = DES
2 = GOC
> = 3 : autres

Octet 3 : mode opératoire (S)
1 = chiffrement des données du fichier, calcul article par article
2 = chiffrement des données du fichier, calcul sur la totalité du fichier

Octet 4 : transfert des éléments de chiffrement (S)
0 = pas de transfert des éléments de chiffrement
1 = transfert des éléments de chiffrement, chiffrés RSA
2 = transfert des éléments de chiffrement, chiffrés DES

ETEBAC5

Octet 1 : présence de chiffrement (S)
0 = pas de chiffrement
1 = chiffrement

Octet 2 : algorithme utilisé (S)
1 = DES

Octet 3 : mode opératoire (S)
1 = chiffrement des données du fichier, calcul article par article
2 = chiffrement des données du fichier, calcul sur la totalité du fichier

Octet 4 : transfert des éléments de chiffrement (S)
0 = pas de transfert des éléments de scellement
1 = transfert des éléments de scellement, chiffrés RSA

ELEMENTS DE CHIFFREMENT 76

Type : N **Longueur : n** paramètre conditionnel

SIT

Interdit

HORS-SIT

Interdit

HORS-SIT sécurisé

Eléments de chiffrement

Paramètre présent si PI 75, octet 1 = 1 et PI 75, octet 4 différent de 0.

Si PI 75, octet 4 = 1 : clé de scellement K1 : vecteur d'initialisation IV1 chiffrés
RSA sous clé publique destinataire : 64 octets

Si PI 75, octet 4 = 2 : KEKNAME nom de la clé de chiffrement de clé KEK : 8 octets
clé de scellement K1 chiffrée DES sous KEK : 8 octets
vecteur d'initialisation IV1 chiffré DES sous K1 : 8 octets
vecteur d'initialisation IV1* chiffré DES sous K1 : 8 octets

ETEBAC5

Eléments de chiffrement

Paramètre présent si PI 75, octet 1 = 1 et PI 75, octet 4 différent de 0.

Si PI 75, octet 4 = 1 : clé de chiffrement K1, vecteur d'initialisation IV1 chiffrés
RSA sous clé publique destinataire : 64 octets

TYPE DE SIGNATURE 77

Type : A **Longueur : 4** paramètre optionnel avec valeur implicite

SIT

Interdit

HORS-SIT

Interdit

HORS-SIT sécurisé

Octet 1 : présence de signature (S)
0 = pas de signature
1 = signature

Octet 2 : algorithme utilisé (S)
0 = RSA

Octet 3 : mode opératoire (S)
1 = signature ETEBAC5
2 = signature du sceau

Octet 4 : double signature (S)
1 = simple signature
2 = double signature

ETEBAC5

Octet 1 : présence de signature (S)
0 = pas de signature
1 = signature

Octet 2 : algorithme utilisé (S)
1 = RSA

Octet 3 : mode opératoire (S)
1 = signature ETEBAC5

Octet 4 : double signature (S)
1 = simple signature
2 = double signature

SCEAU 78

Type : N **Longueur : 4** paramètre conditionnel

SIT

Interdit

HORS-SIT

Interdit

HORS-SIT sécurisé

Sceau.

Le sceau est le résultat de l'algorithme de scellement DES portant sur les données du fichier et sur les paramètres constituant le FID du fichier : PI 11, PI 12, PI 51, PI 61, PI 62.

Paramètre présent dans la FPDU.SYN si le PI 73 octet 1 = 1 et PI 73 octet 3 = 1, ou 3.
Paramètre présent dans la FPDU.DTF.END si le PI 73 octet 1 = 1.

ETEBAC5

Sceau.

Le sceau est le résultat de l'algorithme de scellement DES portant sur les données du fichier et sur les paramètres constituant le FID du fichier : PI 11, PI 12, PI 51, PI 61, PI 62.

Paramètre présent dans la FPDU.SYN si le PI 73 octet 1 = 1 et PI 73 octet 3 = 1, ou 3.

SIGNATURE 79

Type : N **Longueur : 4** paramètre conditionnel

SIT

Interdit

HORS-SIT

Interdit

HORS-SIT sécurisé

Signature.

La signature est le résultat du chiffrement RSA sous la clé secrète émetteur des sceaux portant sur le fichier (données du fichier et FID) et du sceau portant sur le FID seulement : paramètres PI 11, PI 12, PI 51, PI 61, PI 62.

Paramètre présent dans la FPDU.DTF.END si le PI 73 octet 1 = 1 et PI 77 octet 1 = 1.

ETEBAC5

Signature.

La signature est le résultat du chiffrement RSA sous la clé secrète émetteur des sceaux portant sur le fichier (données du fichier et FID) et du sceau portant sur le FID seulement : paramètres PI 11, PI 12, PI 51, PI 61, PI 62.

Paramètre présent dans la FPDU.DTF.END si le PI 73 octet 1 = 1 et PI 77 octet 1 = 1.

ACCREDITATION 80

Type : N **Longueur : 168** paramètre conditionnel

SIT

Interdit

HORS-SIT

Interdit

HORS-SIT sécurisé

Accréditation.

La nature de l'accréditation utilisée en profil PeSIT HORS-SIT sécurisé est laissée au choix de l'utilisateur.

ETEBAC5

Accréditation.

Octet 1 (C) : type d'accréditation
0 = authentification
1 = signature
2 = test

Octets 2 à 25 (C) : identificateur du partenaire

Octets 26 à 37 (C) : numéro de série du dispositif

Octets 38 à 103 (N) clé publique du partenaire (modulo et exposant)

Octet 104 (C) : numéro de bi-clé du gestionnaire

Octets 105 à 168 (N) : signature de l'accréditation

ACCUSE DE RECEPTION DE LA SIGNATURE 81

Type : N **Longueur : 64** paramètre conditionnel

SIT

Interdit

HORS-SIT

Interdit

HORS-SIT sécurisé

Accusé de réception de la signature.

L'utilisation et la nature de l'accusé de réception de la signature en profil PeSIT HORS-SIT sécurisé est laissée au choix de l'utilisateur.

ETEBAC5

Accusé de réception de la signature.

L'accusé de réception de la signature est le résultat du chiffrement RSA sous la clé secrète émetteur de : sceaux reçus, date et heure, acquittement.

Le champ acquittement contient deux octets :

Octet 1: contrôle des éléments de sécurité (C)
0 = accepté
1 = rejeté
2 = non effectué

Octet 2 : valeur de l'acquittement (C)
X = non significatif
0 = fichier transmis pour traitement

DEUXIEME SIGNATURE 82

Type : N **Longueur : 64** paramètre conditionnel

SIT

Interdit

HORS-SIT

Interdit

HORS-SIT sécurisé

Inutilisé.

ETEBAC5

Deuxième signature.

Paramètre présent dans la FPDU.DTF.END si le PI 73 octet 1 = 1 et PI 77 octet 1 = 1,
PI 77 octet 4 = 2.

Signature de même format que le PI 79 mais obtenue avec une autre clé secrète (Cf. Annexe C : Utilisation des mécanisme de sécurité).

DEUXIEME ACCREDITATION 83

Type : N **Longueur : 168** paramètre conditionnel

SIT

Interdit

HORS-SIT

Interdit

HORS-SIT sécurisé

Inutilisé

ETEBAC5

Deuxième accréditation.

Accréditation de même format que le PI 80 mais relative à un autre bi-clé
(Cf. Annexe C : Utilisation des mécanismes de sécurité)

MESSAGE 91

Type : N **Longueur : 4096** paramètre optionnel

SIT

Inutilisé

HORS-SIT/HORS-SIT sécurisé

Message

ETEBAC5

Inutilisé

MESSAGE LIBRE 99

Type : N **Longueur : 254** paramètre optionnel

SIT

Inutilisé

HORS-SIT

Message libre.

Aucun contrôle n'est fait par le protocole sur le codage, la structure ou la sémantique du contenu du message libre.

ETEBAC5

Message libre.

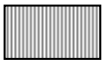
Le standard ETEBAC5 impose que le contenu du message libre soit du type Caractère (codage ASCII).

4.7.4 Structure des éléments de protocole

Les pages suivantes donnent la structure de chaque message du protocole pour chaque profil.

La représentation fait apparaître la liste des PI et PGI d'un message ainsi que le caractère optionnel ou conditionnel d'un PI :

PI optionnel



PI optionnel avec valeur implicite

PI conditionnel

EXEMPLE :

Profil : SIT

Code PGI :	9						
Code PI :	**3**	**4**	11	12	**13**	15	17	25

CONVENTION :

(E) : PI présent lors d'un transfert en écriture uniquement.

(L) : PI présent lors d'un transfert en lecture uniquement.

1) Type de message 20 = FPDU.CONNECT

SIT

3	4	6	**7**	22	**23**	99

La FPDU.CONNECT émise par la station ne contient pas de PI 5. Il est admis que la FPDU.CONNECT venant du CTB contienne un PI 5, celui-ci est alors ignoré par la station.

HORS-SIT/HORS-SIT sécurisé.

1	3	4	**5**	6	**7**	22	**23**	**99**

ETEBAC 5

1	3	4	**5**	6	**7**	22	**23**	**26**	**99**

2) Type de message 21 = FPDU.ACONNECT

SIT

6	**7**	**23**	**99**

La FPDU.ACONNECT émise par la station ne contient pas de PI 5. Il est admis que la FPDU.CONNECT venant du CTB contienne un PI 5, celui-ci est alors ignoré par la station.

HORS-SIT/HORS-SIT sécurisé/ETEBAC 5.

5	6	**7**	**23**	**99**

3) Type de message 22 = FPDU.RCONNECT

SIT/HORS-SIT/HORS-SIT sécurisé

2	**99**

ETEBAC 5

2	**29**	**99**

4) Type de message 23 = FPDU.RELEASE

SIT/HORS-SIT/HORS-SIT sécurisé

2	**99**

ETEBAC 5

2	**29**	**99**

5) Type de message 24 = FPDU.RELCONF

SIT/HORS-SIT/HORS-SIT sécurisé/ETEBAC 5

99

6) Type de message 25 = FPDU.ABORT

SIT/HORS-SIT/HORS-SIT sécurisé

2

ETEBAC 5

2	**29**

Remarque :

Le code PI et la longueur sont supprimés dans la FPDU-ABORT, car l'utilisation dans PeSIT du S-ABORT ne permet que 9 octets dans le champ de données utilisateur (Cf. § 4.3.1).

7) Type de message 11 = FPDU.CREATE

SIT

9							
3	**4**	11	12	13	**15**	17	25

30						40		50		
31	32	**33**	**34**	**36**	**37**	**41**	42	51	**52**	**99**

La FPDU.CREATE émise par la station ne contient pas de PI 16. Il est admis que la FPDU.CREATE venant du CTB contienne un PI 16, celui-ci est alors ignoré par la station.

HORS-SIT

9								
3	**4**	11	12	13	**15**	**16**	17	25

30						40	
31	32	**33**	**37**	**38**	**39**	**41**	42

50			
51	**52**	**61**	**62**	**99**

HORS-SIT sécurisé

9								
3	**4**	11	12	13	**15**	**16**	17	25

30						40		50	
31	32	**33**	**37**	**38**	**39**	**41**	42	51	**52**

61	**62**	**63**	**71**	**72**	**73**	**75**	**77**	**80**	**99**

7) Type de message 11 = FPDU.CREATE (suite)

ETEBAC 5

9								
3	**4**	11	12	13	**15**	**16**	17	25

30						40		50	
31	32	**33**	**37**	**38**	**39**	41	42	51	**52**

61	**62**	**63**	**71**	**72**	**73**	**75**	**77**	**80**	**99**

8) Type de message 30 = FPDU.ACK (CREATE)

SIT

2	**25**	**99**

HORS-SIT

2	**13**	25	**99**

HORS-SIT sécurisé

2	**13**	25	**72**	**80**	**83**	**99**

ETEBAC 5

2	**13**	25	**29**	**64**	**72**	**80**	**83**	**99**

9) Type de message 12 = FPDU.SELECT

SIT

Cette FPDU est inutilisée en profil SIT

HORS-SIT

9									
3	**4**	11	12	13	**14**	**15**	17	25	**99**

HORS-SIT sécurisé

9											
3	**4**	11	12	13	**14**	**15**	17	25	**61**	**62**	**63**

71	**72**	**73**	**75**	**77**	**80**	**99**

ETEBAC 5

9											
3	**4**	11	12	13	**14**	**15**	17	25	**61**	**62**	**63**

71	**72**	**73**	**75**	**77**	**80**	**99**

10) Type de message 31 = FPDU.ACK (SELECT)

SIT

Cette FPDU est inutilisée dans le SIT

HORS-SIT

	9						
2	**3**	4	11	12	**13**	**16**	25

30						40		50		
31	32	**33**	**37**	**38**	**39**	**41**	42	51	**52**	**99**

HORS-SIT sécurisé

	9						
2	**3**	4	11	12	13	**16**	25

30						40	
31	32	**33**	**37**	**38**	**39**	**41**	42

50					
51	**52**	**72**	**80**	**83**	**99**

ETEBAC 5

	9							
2	**3**	**4**	11	12	13	**16**	25	**29**

30						40	
31	32	**33**	**37**	**38**	**39**	**41**	42

50						
51	**52**	**64**	**72**	**80**	**83**	**99**

11) Type de message 13 = FPDU.DESELECT

SIT/HORS-SIT/ HORS-SIT sécurisé

2	**99**

ETEBAC 5

2	**29**	**99**

12) Type de message 32 = FPDU.ACK(DESELECT)

SIT/HORS-SIT/ HORS-SIT sécurisé

2	**99**

ETEBAC 5

2	**29**	**99**

13) Type de message 14 = FPDU.ORF

SIT

La FPDU.ORF émise par la station ne contient pas de paramètre. Il est admis que la FPDU.ORF venant du CTB contienne un PI 21, celui-ci est alors ignoré par la station.

HORS-SIT

21	

HORS-SIT sécurisé/ETEBAC 5

21	**72**	**74**	**76**	**80**	**83**
		(E)	**(E)**		

14) Type de message 33 = FPDU.ACK(ORF)

SIT

2

La FPDU.ACK(ORF) émise par la station ne contient pas de paramètre PI 21. Il est admis que la FPDU.ACK(ORF) venant du CTB contienne un PI 21, celui-ci est alors ignoré par la station.

HORS-SIT

2	**21**

HORS-SIT sécurisé

2	**21**	**74**	**76**
		(L)	**(L)**

ETEBAC 5

2	**21**	**29**	**74**	**76**
			(L)	**(L)**

15) Type de message 15 = FPDU.CRF

SIT/HORS-SIT/ HORS-SIT sécurisé

2

ETEBAC 5

2	**29**

16) Type de message 34 = FPDU.ACK(CRF)

SIT/HORS-SIT/ HORS-SIT sécurisé

2

ETEBAC 5

2	**29**

17) Type de message 01 = FPDU.READ

SIT/HORS-SIT/ HORS-SIT sécurisé/ETEBAC 5

18

18) Type de message 35 = FPDU.ACK(READ)

SIT/HORS-SIT/ HORS-SIT sécurisé

2

ETEBAC 5

2	**29**

19) Type de message 02 = FPDU.WRITE

SIT/HORS-SIT/ HORS-SIT sécurisé/ETEBAC 5
Ne contient aucun paramètre

20) Type de message 36 = FPDU.ACK(WRITE)

SIT/HORS-SIT/ HORS-SIT sécurisé

2	**18**

ETEBAC 5

2	18	**29**

21) Type de message 08 = FPDU.TRANS.END

SIT
Ne contient aucun paramètre

HORS-SIT

27	**28**

HORS-SIT sécurisé/ETEBAC 5

27	**28**	**81**

(L)

22) Type de message 37 = FPDU.ACK(TRANS.END)

SIT

2

HORS-SIT

2	**27**	**28**

HORS-SIT sécurisé

2	**27**	**28**	**81**

(E)

ETEBAC 5

2	**27**	**28**	**29**	**81**

(E)

23) Type de message 00 = FPDU.DTF

SIT/HORS-SIT/HORS-SIT Sécurisé/ETEBAC 5

La FPDU.DTF contient les données du fichier mais ne contient pas de paramètre.

Note : il n'existe pas de FPDU.DTF nulle.

24) Type de message 41 = FPDU.DTFDA

Type de message 40 = FPDU.DTFMA

Type de message 42 = FPDU.DTFFA

SIT

Ces FPDU sont inutilisées dans le SIT.

HORS-SIT/HORS-SIT sécurisé/ETEBAC 5

Les FPDU.DTFDA, FPDU.DTFMA, FPDU.DTFFA contiennent une partie d'un article du fichier mais ne contiennent pas de paramètres.

Nota : Il n'existe pas de FPDU.DTFDA, FPDU.DTFMA, FPDU.DTFFA nulle.

25) Type de message 05 = FPDU.DTF.END

SIT/HORS-SIT

2	

HORS-SIT sécurisé

2	**78**	**79**

ETEBAC 5

2	**29**	**78**	**79**	**82**

26) Type de message 03 = FPDU.SYN

SIT/HORS-SIT

20

HORS-SIT sécurisé/ETEBAC 5

20	**78**

27) Type de message 36 = FPDU.ACK(SYN)

SIT/HORS-SIT/HORS-SIT sécurisé/ETEBAC5

20

28) Type de message 05 = FPDU.RESYN

SIT

Cette FPDU est inutilisée dans le SIT.

HORS-SIT/HORS-SIT sécurisé

2	18

ETEBAC 5

2	18	**29**

29) Type de message 39 = FPDU.ACK(RESYN)

SIT

Cette FPDU est inutilisée dans le SIT.

HORS-SIT/HORS-SIT sécurisé/ETEBAC 5

18

30) Type de message 06 = FPDU.IDT

SIT/HORS-SIT/HORS-SIT sécurisé

2	19

ETEBAC 5

2	19	**29**

31) Type de message 3A = FPDU.ACK(IDT)

SIT, HORS-SIT, ETEBAC 5

Ne contient aucun paramètre.

32) Type de message 16 = FPDU.MSG

SIT

Cette FPDU n'est pas utilisée dans le SIT.

HORS-SIT

9						
3	**4**	11	12	13	**14**	**16**

50			
51	**61**	**62**	**91**

HORS-SIT sécurisé

9						
3	**4**	11	12	13	**14**	**16**

50										
51	**61**	**62**	**73**	**74**	**77**	**78**	**79**	**80**	**81**	**91**

ETEBAC 5

Cette FPDU n'est pas utilisée.

33) Type de message 17 = FPDU.MSGDM

SIT

Cette FPDU n'est pas utilisée dans le SIT.

HORS-SIT

9						
3	**4**	11	12	13	**14**	**16**

50			
51	**61**	**62**	**91**

HORS-SIT sécurisé

9						
3	**4**	11	12	13	**14**	**16**

50							
51	**61**	**62**	**73**	**74**	**77**	**80**	**91**

ETEBAC 5

Cette FPDU n'est pas utilisée.

34) Type de message 18 = FPDU.MSGMM

SIT

Cette FPDU n'est pas utilisée dans le SIT.

HORS-SIT/ HORS-SIT sécurisé

91	

ETEBAC 5

Cette FPDU n'est pas utilisée.

35) Type de message 19 = FPDU.MSGFM

SIT

Cette FPDU n'est pas utilisée dans le SIT.

HORS-SIT

91	

HORS-SIT sécurisé

78	**79**	**91**	

ETEBAC 5

Cette FPDU n'est pas utilisée.

36) Type de message 3B = FPDU.ACK(MSG)

SIT

Cette FPDU n'est pas utilisée dans le SIT.

HORS-SIT

2	**13**	**16**	**91**

HORS-SIT sécurisé

2	**13**	**16**	**80**	**81**	**91**

ETEBAC 5

Cette FPU n'est pas utilisée.

4.8 TABLES D'ETAT DU PROTOCOLE PeSIT

4.8.1 Eléments utilisés dans la description formelle

Ce chapitre décrit formellement le protocole sous la forme d'un automate fini avec des états, des événements et des actions.

Cette description concerne aussi bien les protocoles PeSIT-F, PeSIT-F' et PeSIT F" de manière uniforme.

Les tables d'état décrites dans ce chapitre indiquent pour chaque état d'une connexion PeSIT, les événements qui peuvent se produire dans le protocole, les actions effectuées et l'état résultant.

Avant de décrire les tables d'état, il est nécessaire de définir toutes les abréviations utilisées concernant :

- les états,
- les événements,
- et les actions.

4.8.1.1 Liste des états

Abréviation	Nom et description
CN…	**PHASE CONNEXION**
CN01	REPOS, non connecté
CN02A	Connexion en attente d'une FPDU. ACONNECT, ou RCONNECT
CN02B	Connexion en attente d'une primitive F.CONNECT, R
CN03	CONNECTE
CN04A	Libération en attente d'une FPDU.RELCONF.
CN04B	Libération en attente d'une F.RELEASE, R.
SF…	**PHASE SELECTION DU FICHIER**
SF01A	Création de fichier en attente d'une FPDU.ACK (CREATE)
SF01B	Création de fichier en attente d'une primitive F.CREATE, R
SF02A	Sélection de fichier en attente d'une FPDU.ACK (SELECT)
SF02B	Création de fichier en attente d'une primitive F.SELECT, R
SF03	FICHIER SELECTIONNE
SF04A	Libération de fichier en attente d'une FPDU.ACK (DESELECT)
SF04B	Libération de fichier en attente d'une primitive F.DESELECT, R
OF…	**PHASE OUVERTURE DE FICHIER**
OF01A	Ouverture de fichier en attente d'une FPDU.ACK (ORF)
OF01B	Ouverture de fichier en attente d'une primitive F.OPEN, R
OF02	TRANSFERT DE DONNEES - REPOS
OF03A	Fermeture de fichier en attente d'une FPDU.ACK (CRF)
OF03B	Fermeture de fichier en attente d'une primitive F.CLOSE, R

TDL…	**PHASE TRANSFERT DE DONNEES EN LECTURE**
TDL01A	Lancement de lecture en attente d'une FPDU.ACK (READ) (Demandeur)
TDL01B	Lancement de lecture en attente d'une primitive F.READ, R (Serveur)
TDL02A	Réception de données (Demandeur)
TDL02B	Emission de données (Serveur)
TDL03	Resynchronisation en attente d'une FPDU.ACK (RESYN)
TDL04	Resynchronisation en attente d'une primitive F.RESTART, R
TDL05	Interruption du transfert en attente d'une FPDU.ACK (IDT)
TDL06	Interruption du transfert en attente d'une primitive F.CANCEL, R
TDL07	Fin de lecture
TDL08A	Fin de transfert de lecture en attente d'une FPDU.ACK(TRANS.END) (Demandeur)
TDL08B	Fin de transfert de lecture en attente d'une primitive F.TRANSFER.END, R (Serveur)

Abréviation	Nom et description
TDE…	**PHASE TRANSFERT DE DONNEES EN ECRITURE**
TDE01A	Lancement d'écriture en attente d'une FPDU.ACK(WRITE) (Demandeur)
TDE01B	Lancement d'écriture en attente d'une primitive F.WRITE, R (Serveur)
TDE02A	Emission de données (Demandeur)
TDE02B	Réception de données (Serveur)
TDE03	Resynchronisation en attente d'une FPDU.ACK (RESYN)
TDE04	Resynchronisation en attente d'une primitive F.RESTART, R
TDE05	Interruption du transfert en attente d'une FPDU.ACK (IDT)
TDE06	Interruption du transfert en attente d'une primitive F.CANCEL, R
TDE07	Fin d'écriture
TDE08A	Fin de transfert d'écriture en attente d'une FPDU.ACK (TRANS.END) (Demandeur)
TDE08B	Fin de transfert d'écriture en attente d'une primitive F.TRANSFER.END, R (Serveur)

Note :

L'abréviation F.XXXX, R signifie une primitive de réponse.

Tous les états se terminant par la lettre "A" concernent exclusivement le PeSIT demandeur et ceux se terminant par la lettre "B", le serveur. Les autres états sont communs aux deux entités.

4.8.1.2 Liste des événements

Pour l'automate fini, décrivant le comportement d'une entité PeSIT, les événements entrants sont :

- soit la réception d'une primitive de service émanant de l'utilisateur local (demande ou réponse),
- soit la réception d'un FPDU de l'autre entité PeSIT,
- soit la détection d'une erreur (expiration d'un temporisateur de surveillance ou erreur de protocole).

Les principaux événements sortants sont constitués par l'émission :

- d'une primitive de service à destination de l'utilisateur local (indication ou confirmation),
- ou d'un message destiné à l'entité PeSIT homologue.

Les abréviations suivantes sont utilisés pour désigner les événements :

- XXX : FPDU.XXX
- A (XXX) : FPDU.ACK en réponse à une FPDU.XXX
- YYY(D) : primitive de demande pour le service YYY
- YYY(I) : primitive d'indication pour le service YYY
- YYY(R) : primitive de réponse pour le service YYY
- YYY(C) : primitive de confirmation pour le service YYY.

Les valeurs autorisées pour XXX et YYY sont données en 4.2.

4.8.1.3 Liste des conditions

Les transitions d'état et les actions sont parfois conditionnelles. Les abréviations relatives aux conditions utilisées sont définies comme suit :

Abréviation	Signification
+	ACK positif (OK ou avertissement)
-	ACK négatif (remédiable ou bloquant)
ab	transfert arrêté prématurément (1)
ck	option points de synchronisation négociée
dr	le demandeur est le récepteur
rel	relance demandée
cft	code fin de transfert = annulation ou suspension
cc>o	il existe des points de synchronisation en attente

Le symbole suivant est aussi utilisé pour la condition :
& ET

(1) "ab" est positionné par F.CANCEL avec "code fin de transfert" = suspension ou annulation. Il est remis à NON ab par F.DESELECT. Il a pour objet d'empêcher un rebouclage par F.READ (ou WRITE) ou F.OPEN lorsqu'un transfert a été arrêté prématurément.

4.8.1.4 Liste des actions

Les actions peuvent être conditionnelles ou inconditionnelles. Une action donnée consiste à :

- émettre un événement sortant, indiqué par son nom abrégé,
- à exécuter une action spécifique,
- ou à imposer une condition.

Lorsque plusieurs actions sont spécifiées, leur ordre est **impératif**.

a) Actions implicites

Un certain nombre d'actions ne sont pas spécifiées dans les tables d'état, elles sont implicites. Ces actions sont :

- dans le cas d'une intersection vide dans les tables d'état (l'événement correspondant est invalide), arrêt prématuré de la connexion (ABORT) avec le code diagnostic approprié,
- pour chaque FPDU reçu, un contrôle de validité systématique (vérification des paramètres utilisés). Si une erreur est détectée, la procédure de relance est mise en œuvre,
- dans le cas de la réception d'une demande d'arrêt prématurée de connexion, le passage systématique à l'état "REPOS" quel que soit l'état (sauf l'état CN01), la notification de l'ABORT à l'utilisateur local ou au PeSIT homologue et la mise à l'état initial de tous les paramètres.

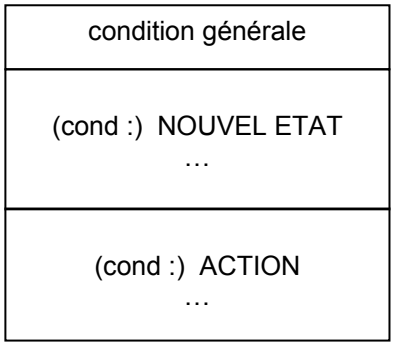
b) Actions spécifiques

REC	Réception du message suivant du système de communication
SAB	Positionnement de l'indicateur d'arrête prématuré
RESAB	Reset de l'indicateur d'arrête prématuré
SCK	Positionnement de l'indicateur de pose de point de synchronisation
RESCK	Reset de l'indicateur de point de synchronisation
SDR	Positionnement de l'indicateur demandeur = récepteur
RESDR	Reset de l'indicateur demandeur = récepteur
SREL	Positionnement de l'indicateur de relance
RESREL	Reset de l'indicateur de relance.

4.8.2 Conventions générales

Chaque table d'états constitue une partie de la table d'état générale, pour des raisons de présentation. La colonne de gauche donne l'ensemble des événements et la ligne supérieure les différents états.

Chaque intersection dans la table entre événement et état contient ce qui suit :



Où :

- la condition générale est une condition de validité pour toute l'intersection (exprimée par "COND : cond"),
- (cond :) est une condition facultative s'appliquant à la ligne qui la suit : les conditions sont toujours exprimées en minuscules,
- NOUVEL ETAT : c'est l'état qui est adopté lorsque les actions indiquées ont été exécutées. Les états sont toujours exprimés en majuscules,
- ACTION consiste à émettre un événement sortant, à imposer une condition ou à exécuter une action spécifique quelconque. Lorsque plusieurs actions sont spécifiées, leur ordre est **impératif.** Les actions sont toujours exprimées en majuscules.

Les caractères spéciaux suivants sont aussi utilisés :

- "." caractère de fin de condition
- "," caractère d'enchaînement de liste.

Une intersection dans la table est considérée comme invalide si elle ne contient aucune indication ou si la condition générale indiquée par "COND" n'est pas satisfaite.

Toutes les intersections invalides entre états et événements "primitives de service entrée" (demande ou réponse) sont considérées comme des erreurs locales et ne sont pas standardisées dans ce document.

Lors du découpage de la table d'états, toutes les lignes et toutes les colonnes ne contenant que des intersections vides ont été supprimées. Il s'ensuit que toute intersection n'apparaissant pas dans une table d'états ci-après est implicitement invalide.

4.8.3 Règles pour les collisions

Dans la phase de transfert de données, des cas de collision peuvent se produire. Les règles suivantes sont alors appliquées :

1. Ordre de priorité des FPDU

   1.- ABORT
   2.- IDT
   3.- RESYN
   4.- TRANS.END
   5.- ACK.SYN
   6.- DTF.END
   7.- DTF

2. S'il y a collision entre deux FPDU de même type, c'est le demandeur qui est prioritaire sur le serveur.

4.8.4 Les tables d'état

Il existe deux jeux de tables d'états qui varient selon le rôle dévolu aux entités PeSIT : demandeur ou serveur. Dans le cas de l'automate de transfert de données, nous avons distingué quatre tables :

. demandeur-émetteur, demandeur-récepteur, serveur-récepteur, serveur-émetteur.

Table	Phase de protocole	Rôle
1	connexion	demandeur
2	connexion	serveur
3	sélection de fichier	demandeur
4	sélection de fichier	serveur
5	ouverture de fichier	demandeur
6	ouverture de fichier	serveur
7	transfert de données	demandeur-émetteur (écriture)
8	transfert de données	serveur-récepteur (écriture)
9	transfert de données	demandeur-récepteur (lecture)
10	transfert de données	serveur-émetteur (lecture)

TABLE 1 : Phase connexion (demandeur)

	CN01	**CN02A**	**CN03**	**CN04A**
F.CONNECT(D)	CN02A CONNECT			
A.CONNECT		CN03 F.CONNECT (C) ck : SCK		
R.CONNECT		CN01 F.CONNECT (C)		
F.RELEASE(D)			CN04A RELEASE	
REL.CONF				CN01 F.RELEASE (C) ck : RESCK
F.ABORT(D)		CN01 ABORT	CN01 ABORT	CN01 ABORT
ABORT		CN01 F.ABORT (I)	CN01 F.ABORT (I)	CN01 F.ABORT (I)

TABLE 2 : Phase connexion (serveur)

	CN01	**CN02B**	**CN03**	**CN04B**
CONNECT	CN02B F.CONNECT(I)			
F.CONNECT(R)		+ : CN03 - : CN01 + : A.CONNECT ck : SCK - : R.CONNECT		
RELEASE			CN04B F.RELEASE(I)	
F.RELEASE(R)				CN01 REL.CONF ck : RESCK
ABORT		CN01 F.ABORT (I)	CN01 F.ABORT (I)	CN01 F.ABORT (I)
F.ABORT(D)		CN01 ABORT	CN01 ABORT	CN01 ABORT

TABLE 3 : Phase sélection de fichier (demandeur)

	CN03	**SF01A**	**SF02A**	**SF03**	**SF04A**
F.SELECT	SF02A SELECT				
A(SELECT)			+ : SF03 - : CN03 F.SELECT(C) ; SDR rel : SREL		
F.CREATE(D)	SF01A CREATE				
A(CREATE)		+ : SF03 - : CN03 F.SELECT (C) rel : SREL			
F.DESELECT (D)				SF04A DESELECT	
A.(DESELECT)					CN03 F.DESELECT(C) ab : RESAB dr : RESDR rel : RESREL

TABLE 4 : Phase sélection de fichier (serveur)

	CN03	**SF01B**	**SF02B**	**SF03**	**SF04B**
SELECT	SF02B F.SELECT(I)				
F.SELECT(R)			+ : SF03 - : CN03 A(SELECT) rel : SREL		
CREATE	SF01B F.CREATE(I)				
F.CREATE		+ : SF03 - : CN03 A(CREATE) rel : SREL			
DESELECT				SF04B F.DESELECT(I)	
F.DESELECT(R)					CN03 A(DESELECT) ab : RESAB

TABLE 5 : Phase sélection de fichier (demandeur)

	SF03	**OF01A**	**OF02**	**OF03A**
F.OPEN(D)	COND : NON ab OF01A ORF			
A(ORF)		+ : OF02 - : SF03 F.OPEN(C)		
F.CLOSE(D)			OF03A CRF	
A(CRF)				SF03 F.CLOSE(C)

TABLE 6 : Phase ouvertrue de fichier (serveur)

	SF03	**OF01B**	**OF02**	**OF03B**
ORF	COND : NON ab OF01B F.OPEN(I)			
F.OPEN(R)		+ : OF02 - : SF03 A(ORF)		
CRF			OF03B F.CLOSE(I)	
F.CLOSE(R)				SF03 A(CRF)

TABLE 7 : Transfert en écriture : Demandeur - Emetteur

	OF2A	TDE01A	TDE02A	TDE03	TDE04	TDE05	TDE06	TDE07	TDE08A
F.WRITE(D)	COND-NONab TDE01A WRITE								
A(WRITE)		= : TDE02A - : OF02 F.WRITE(C)							
F.CANCEL(D)			TDE05 IDT	TDE05 IDT	TDE05 IDT		TDE05 IDT	TDE05 IDT	TDE05 IDT
A(IDT)						OF02 F.CANCEL(C) SAB			
IDT			TDE06 F.CANCEL(I)	TDE06 F.CANCEL(I)	TDE06 F.CANCEL(I)	PAS DE TRAITEMENT	PAS DE TRAITEMENT	TDE06 F.CANCEL(I)	TDE06 F.CANCEL(I)
F.CANCEL(R)							OF02 A(IDT) SAB		
F.DATA(D)			TDE07 DTF.END cft:SAB		PAS DE TRAITEMENT		PAS DE TRAITEMENT		
F.TRANSFER.END(D)					PAS DE TRAITEMENT			TDE08A TRANS.END	
A(TRANS.END)				PAS DE TRAITEMENT		PAS DE TRAITEMENT			OF02 F.TRANSFER.END(D)
F.DATA(D)			TDE02A DTF		PAS DE TRAITEMENT		PAS DE TRAITEMENT		
F.CHECK(D)			COND:ck&cc <256) TDE02A SYN		PAS DE TRAITEMENT		PAS DE TRAITEMENT		
A(SYN)			COND:cc>0 TDE02A F.CHECK(C)	PAS DE TRAITEMENT		PAS DE TRAITEMENT		TDE07 F.CHECK(C)	PAS DE TRAITEMENT
F.RESTART(D)			COND:ck TDE03 RESYN		TDE03 RESYN		PAS DE TRAITEMENT		
A(RESYN)				TDE02A F.RESTART(C)		PAS DE TRAITEMENT			
RESYN			COND:ck TDE04 F.RESTART(I)	PAS DE TRAITEMENT		PAS DE TRAITEMENT		COND:ck TDE04 F.RESTART(I)	COND:ck TDE04 F.RESTART(I)
F.RESTART(R)					TDE02A A(RESYN)		PAS DE TRAITEMENT		

TABLE 8 : Transfert en écriture : Serveur - Récepteur

	OF02	TDE01B	TDE02B	TDE03	TDE04	TDE05	TDE06	TDE07	TDE08B
WRITE	COND-NONab TDE01B F.WRITE(I)								
F.WRITE(R)		+ : TDE02B - : OF02 A(WRITE) + : REC							
F.CANCEL(D)			TDE05 IDT	TDE05 IDT	TDE05 IDT		PAS DE TRAITEMENT	TDE05 IDT	TDE05 IDT
A(IDT)						OF02 F.CANCEL(C) SAB			
IDT			TDE06 F.CANCEL(I)	TDE06 F.CANCEL(I)	TDE06 F.CANCEL(I)	TDE06 F.CANCEL(I)		TDE06 F.CANCEL(I)	TDE06 F.CANCEL(I)
F.CANCEL(R)							OF02 A(IDT) SAB		
DTF.END			TDE07 F.DATA.END(I) cft:SAB	PAS DE TRAITEMENT		PAS DE TRAITEMENT			
TRANS.END				PAS DE TRAITEMENT		PAS DE TRAITEMENT		TDE08B F.TRANSFER.END(I)	
F.TRANSFER.END(R)							PAS DE TRAITEMENT		OF02 A.TRANS.END(D)
F.CHECK(R)			TDE02B A(SYN) REC		PAS DE TRAITEMENT		PAS DE TRAITEMENT	TDE07 A(SYN)	
DTF			TDE02B F.DATA(I) REC	PAS DE TRAITEMENT		PAS DE TRAITEMENT			
SYN			COND:(ck&cc <256) F.CHECK(I) REC	PAS DE TRAITEMENT		PAS DE TRAITEMENT			
F.RESTART(D)			COND:ck TDE03 RESYN		PAS DE TRAITEMENT		PAS DE TRAITEMENT	COND:ck TDE03 RESYN	COND:ck TDE03 RESYN
A(RESYN)				TDE02B F.RESTART(C) REC		PAS DE TRAITEMENT			
RESYN			COND:ck TDE04 F.RESTART(I)	COND:ck TDE04 F.RESTART(I)		PAS DE TRAITEMENT			
F.RESTART(R)					TDE02B A(RESYN) REC		PAS DE TRAITEMENT		

TABLE 9 : Transfert en lecture : Demandeur - Récepteur

	OF02	TDL01A	TDL02A	TDL03	TDL04	TDL05	TDL06	TDL07	TDL08A
F.READ(D)	COND-NONab TDL01A READ								
A(READ)		+ : TDL02A - : OF02 F.READ(C) + : REC							
F.CANCEL(D)			TDL05 IDT	TDL05 IDT	TDL05 IDT		TDL05 IDT	TDL05 IDT	TDL05 IDT
A(IDT)						OF02 F.CANCEL(C) SAB			
IDT			TDL06 F.CANCEL(I)	TDL06 F.CANCEL(I)	TDL06 F.CANCEL(I)	PAS DE TRAITEMENT		TDL06 F.CANCEL(I)	TDL06 F.CANCEL(I)
F.CANCEL(R)							OF02 A(IDT) SAB		
DTF.END			TDL07 F.DATA.END(I) cft:SAB	PAS DE TRAITEMENT		PAS DE TRAITEMENT			
F.TRANSFER.END(D)					PAS DE TRAITEMENT		PAS DE TRAITEMENT	TDL08A TRANS.END	
A(TRANS.END)				PAS DE TRAITEMENT		PAS DE TRAITEMENT			OF02 F.TRANSFER.END(C)
DTF			TDL02A F.DATA(I)	PAS DE TRAITEMENT		PAS DE TRAITEMENT			
F.CHECK(R)			COND:(ck&cc <256) A(SYN)		PAS DE TRAITEMENT		PAS DE TRAITEMENT	PAS DE TRAITEMENT	
SYN			COND:cc>0 TDL02A F.CHECK(I)	PAS DE TRAITEMENT		PAS DE TRAITEMENT			
F.RESTART(D)			COND:ck TDL03 RESYN		COND:ck TDL03 RESYN		PAS DE TRAITEMENT	COND:ck TDL03 RESYN	COND:ck TDL03 RESYN
A(RESYN)				TDL02A F.RESTART(C)		PAS DE TRAITEMENT			
RESYN			COND:ck TDL04 F.RESTART(I)	PAS DE TRAITEMENT		PAS DE TRAITEMENT			
F.RESTART(R)					TDL02A A(RESYN)				

TABLE 10 : Transfert en lecture : Serveur - Emetteur

	OF02	TDL01B	TDL02B	TDL03	TDL04	TDL05	TDL06	TDL07	TDL08B
READ	COND-NONab TDL01B F.READ(I)								
F.READ(R)		+ : TDL02B - : OF02 A.READ							
F.CANCEL(D)			TDL05 IDT	TDL05 IDT	TDL05 IDT		PAS DE TRAITEMENT	TDL05 IDT	TDL05 IDT
A(IDT)						OF02 F.CANCEL(C) SAB			
IDT			TDL06 F.CANCEL(I)	TDL06 F.CANCEL(I)	TDL06 F.CANCEL(I)	TDL06 F.CANCEL(I)		TDL06 F.CANCEL(I)	TDL06 F.CANCEL(I)
F.CANCEL(R)							OF02 A(IDT) SAB		
F.DATA.END(D)			TDL07 DTF.END cft:SAB		PAS DE TRAITEMENT		PAS DE TRAITEMENT		
TRANS.END						PAS DE TRAITEMENT		TDL08B F.TRANS.END(I)	
F.TRANSFER.END(R)							PAS DE TRAITEMENT		OF02 A(TRANS.END)
F.DATA(D)			TDL02B DTF REC		PAS DE TRAITEMENT		PAS DE TRAITEMENT		
A(SYN)			COND:(ck&cc<256) F.CHECK(C) REC	PAS DE TRAITEMENT		PAS DE TRAITEMENT		PAS DE TRAITEMENT	
F.CHECK(D)			TDL02B SYN REC		PAS DE TRAITEMENT		PAS DE TRAITEMENT		
F.RESTART(D)			COND:ck TDL03 RESYN		PAS DE TRAITEMENT		PAS DE TRAITEMENT		
A(RESYN)				TDL02B F.RESTART(C) REC		PAS DE TRAITEMENT			
RESYN			COND:ck TDL04 F.RESTART(I)	COND:ck TDL04 F.RESTART(I)		PAS DE TRAITEMENT		COND:ck TDL04 F.RESTART(I)	
F.RESTART(R)					TDL02B A(RESYN) REC		PAS DE TRAITEMENT		

ANNEXES

2 : COMPRESSION

1. NEGOCIATION DU TYPE DE COMPRESSION

D'après le protocole PeSIT, dans la FPDU.ORF, il est possible de proposer un type de compression. Il s'agit du PI 21, contenant deux octets :

- Le premier octet a deux valeurs possibles :
  - 0 : pas de compression,
  - 1 : compression proposée.

- Le deuxième octet aura la signification suivante :
  - 1 : compression horizontale,
  - 2 : compression verticale,
  - 3 : combinaison des compressions horizontale et verticale.

La FPDU.ACK(ORF) contient le même PI 21. Les valeurs acceptées dans cette réponse sont :

- Octet 1 = 0 la compression est refusée,

- Octet 1 = 1 la compression est acceptée. Dans ce cas et si le type de compression demandé dans le deuxième octet du PI 21 de la FPDU.ORF était 3, le deuxième octet aura la signification suivante :
  - 1 : compression horizontale,
  - 2 : compression verticale,
  - 3 : combinaison des compressions horizontale et verticale.

2. DEFINITION DES TYPES DE COMPRESSION

*** Compression horizontale**

Il s'agit de la compression de caractères consécutifs identiques. Si cette compression a été acceptée à l'ouverture du fichier, chaque article du fichier est décomposé en chaînes, chaque chaîne étant précédée d'un octet d'en-tête de chaîne. Cet octet a le format suivant :

bit 7.	bit 6.	bit 5.	bit 4.	bit 3.	bit 2.	bit 1.	bit 0

- bit 7 :
  - 0 : la chaîne n'est pas compressée,
  - 1 : la chaîne est compressée,

- bit 6 :
  - 1 : compression horizontale,
  - 2 : compression verticale (non utilisée ici).

Remarque :

Les combinaisons permises quand la compression horizontale a été négociée sont uniquement 00 et 10 pour les bits 7 et 6 (01 et 11 sont interdits).

- bits 0 à 5 :
  Longueur de la chaîne (de 1 à 63 octets), l'octet d'en-tête étant exclu.

L'octet suivant l'en-tête est le caractère constitutif de la chaîne, s'il y a compression.

Exemple :

La chaîne 01 02 02 02 02 02 02 03 devient 01 01 86 02 01 03.

*** Compression verticale**

Il s'agit de la compression par compression d'articles consécutifs.

Le premier article n'est jamais compressé, mais il doit contenir l'en-tête de chaîne défini ci-après.

A partir du deuxième article, on compare par rapport à l'article précédent pour repérer des chaînes de caractères identiques.

Remarque :

Afin de simplifier le mécanisme de reprise de transfert, le premier article suivant chaque point de synchronisation ne sera jamais compressé verticalement.

Chaque article est décomposé en chaînes précédées d'un octet d'en-tête de chaîne.

Le format de l'en-tête est le suivant :

bit 7.	bit 6.	bit 5.	bit 4.	bit 3.	bit 2.	bit 1.	bit 0

- bit 7 :
- 0 : la chaîne n'est pas compressée,
- 1 : la chaîne est compressée,

- bit 6 :
- 1 : compression horizontale,
- 2 : compression verticale.

Remarque :

Les combinaisons permises quand la compression verticale a été négociée sont uniquement 00 et 11 pour les bits 7 et 6 (les combinaisons 01 et 10 sont interdites).

- bits 0 à 5 : Longueur de la chaîne (de 1 à 63 octets), l'octet d'en-tête étant exclu.

Exemple :

- 1er article : 01 02 03 02 03 05 06 07
- 2ème article : 05 06 03 02 03 05 08 09

Ces deux articles deviennent :

- 1er article : 08 01 02 03 02 05 06 07
- 2ème article : 02 05 06 C4 02 08 09

Remarque : dans le cas où les articles consécutifs seraient de longueur différente un padding avec des caractères 40 hexadécimal sera effectué avant compression.

*** Combinaison des compressions horizontale et verticale**

Il est possible d'utiliser pour le même transfert la compression horizontale et la compression verticale. Dans ce cas, chaque article de fichier sera décomposé en sous-chaînes et chaque sous-chaîne pourra être compressée soit horizontalement soit verticalement. Le choix de la compression pour une sous-chaîne donnée est laissé à la liberté de l'émetteur.

Dans le cas où la compression verticale est retenue pour une sous-chaîne et où l'article précédent par rapport auquel la compression verticale se réfère était lui-même compressé horizontalement, la compression verticale se fera par rapport à l'article précédent avant compression horizontale.

Exemple :

- 1er article : 01 01 01 01 02 03
- 2ème article : 01 01 01 01 02 04

Ces deux articles deviennent :

- 1er article : 84 01 02 02 03
- 2ème article : C5 01 04

3. INTERVALLE ENTRE POINTS DE SYNCHRONISATION ET COMPRESSION

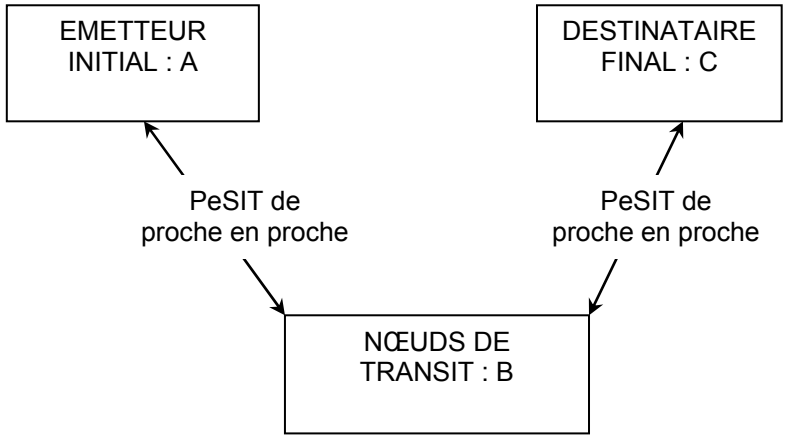
Dans le cas où la compression est utilisée, l'intervalle entre points de synchronisation doit être calculé sur les données après compression, telles qu'elles sont effectivement transmises.

Les octets d'en-tête de chaîne de compression font partie de ces chaînes et doivent être comptés pour le calcul de l'intervalle entre points de synchronisation.

3 : MODE STORE AND FORWARD

1. INTRODUCTION

On appelle mode ***store and forward*** la possibilité de faire du routage de fichier de machine à machine. Dans ce mode un fichier devant être émis d'une machine A vers une machine C, qui ne sont pas directement connectés, transitera par une machine B qui, en le recevant de A, saura le ré-émettre vers C.



STORE AND FORWARD

PeSIT étant un protocole local - c'est-à-dire dont les paramètres n'ont de signification que pour une connexion donnée entre deux entités reliées par un circuit virtuel (PeSIT.F'), une connexion session ISO (PeSIT.F), ou une connexion NETEX (PeSIT.F") - autoriser un mode de transfert ***store and forward*** nécessite de définir comment l'adressage PeSIT permet le routage de fichier et quels sont les paramètres qui doivent être véhiculés au cours des transferts successifs d'un même fichier au travers des différents nœuds intermédiaires d'un réseau.

Nous ne définirons ici que le cas de l'écriture de fichier en considérant les points suivants :

* **adressage** : pour un transfert donné, il faut identifier les partenaires qui effectuent le transfert (Demandeur et Serveur pour la connexion PeSIT) et les partenaires pour le compte desquels le transfert est effectué (émetteur initial et destinataire final du fichier).

* **identification des transferts** : il faut définir une identification des transferts qui soit non ambiguë pour l'ensemble du domaine couvert par le ***store and forward.***

* **acquittement des transferts** : il faut donner à l'émetteur initial du fichier l'indication que son fichier est effectivement parvenu à son destinataire final (acquittement de bout en bout).

2. CONSEQUENCES SUR LE PROTOCOLE

2.1 Dans la FPDU.CONNECT

La phase de Connexion met en relation deux partenaires reliés par une connexion directe : les paramètres qui figurent dans les FPDU.CONNECT et FPDU.ACONNECT sont purement locaux.

PI 3 et PI 4 : identifient les partenaires entre lesquels la connexion est établie : chaîne de 1 à 24 octets.

2.2 Dans la FPDU.CREATE

A partir de la phase de Sélection on peut distinguer les partenaires pour le compte desquels le transfert est effectué de ceux qui effectuent le transfert. Les PI 61 et 62 (identificateur Client et identificateur Banque de ETEBAC 5) seront donc utilisés pour identifier l'émetteur initial et le destinataire final. C'est à partir d'eux que les moniteurs relais effectueront si nécessaire le routage vers la machine suivante. Les PI 3 et PI 4 (optionnels) de la FPDU.CREATE seront utilisés pour définir un adressage plus précis.

PI 3 et PI 4 : 24 octets :
octets 1 à 8 : nom d'application
octets 9 à 17 : nom d'utilisateur
octets 17 à 24 : libre.

PI 61 et PI 62 (optionnels) : définissent les partenaires pour le compte desquels le transfert s'effectue : émetteur et destinataire final. 24 octets.

Le PI 13 (identificateur de transfert) contribuera à l'identification non ambiguë du transfert : il sera propagé identique au cours des transferts successifs pour un même fichier. Il s'agit ici du PI 13 de la FPDU.CREATE, le PI 13 optionnel de la FPDU.ACK(CREATE) que le serveur peut renvoyer est purement indicatif et sera spécifique à chaque transfert.

L'identification d'un fichier sera donc réalisée à l'aide des paramètres :

PI 11 (type de fichier)
PI 12 (nom de fichier)
PI 13 (identificateur de transfert)
PI 61 (identificateur d'émetteur initial)
PI 62 (identificateur de destinataire final).

3. FPDU.MSG

La FPDU.MSG sera utilisée pour transporter l'acquittement du transfert de bout en bout. Le contenu de cette FPDU.MSG sera :

PGI 9 :
PI 3 : Identificateur du Demandeur (optionnel) : identique à PI 4 qui était dans la FPDU.CREATE utilisée pour le transfert du fichier acquitté par la FPDU.MSG*
PI 4 : Identificateur du Serveur (optionnel) : identique à PI 3 qui était dans la FPDU.CREATE utilisée pour le transfert du fichier acquitté par la FPDU.MSG*
PI 11 : Type de fichier : PI 11 du fichier dont la FPDU.MSG transporte l'acquittement
PI 12 : Nom de fichier : PI 12 du fichier dont la FPDU.MSG transporte l'acquittement

PI 13 : Identificateur de transfert : PI 13 utilisé de bout en bout pour le transfert du fichier dont la FPDU.MSG transporte l'acquittement
PI 14 : Attributs demandés : valeur indifférente
PI 16 : Code données (optionnel) : éventuellement pertinent pour le contenu du PI 91, s'il y en a un

PI 50 : Attributs historiques :

PI 51 : PI 51 du fichier dont la FPDU.MSG transporte l'acquittement

PI 61 : Identification émetteur initial : identique au PI 62 du fichier dont la FPDU.MSG transporte l'acquittement
PI 62 : Identification destinataire final : identique au PI 61 du fichier dont la FPDU.MSG transporte l'acquittement
PI 91 : Message (optionnel)

* si ce PI contribue à l'identification du fichier.

4 : UTILISATION DES MECANISMES DE SECURITE

1 INTRODUCTION

L'utilisation de mécanismes de sécurité au niveau des transferts de fichier est prévue dans deux profils de PeSIT :

- le profil ETEBAC 5

- le profil PeSIT Hors-SIT sécurisé

Le profil ETEBAC 5 est le résultat des groupes de travail ETEBAC 5 (groupes "transport" et "sécurité ») du CFONB. La description complète de l'utilisation de la sécurité dans le standard ETEBAC 5 se trouve dans le document **"Echanges Télématiques entre les Banques et leurs Clients. STANDARD ETEBAC 5"** du CFONB. Il est à noter que l'utilisation du profil ETEBAC 5 suppose l'utilisation d'accréditations dont la production et la distribution aux partenaires ETEBAC 5 se fait sous l'autorité du CFONB.

Le profil PeSIT HORS-SIT sécurisé résulte du désir de mettre en œuvre des fonctions de sécurité en dehors du cadre des échanges ETEBAC 5. Ce profil utilise les paramètres de sécurité définis pour ETEBAC 5. Cependant, il est conçu pour permettre de réaliser les fonctions : authentification réciproque, intégrité, confidentialité en utilisant uniquement l'algorithme DES (Data Encryption Standard), alors que le profil ETEBAC 5 suppose l'emploi obligatoire de l'algorithme RSA (Rivest Shamir Adelman).

Il est à noter que l'utilisation de dispositifs et d'algorithmes cryptographiques pour les transmissions de données à travers les réseaux publics doit se faire en accord avec la législation en vigueur à ce sujet.

2. MISE EN ŒUVRE DES ALGORITHMES

2.1 Chiffrement DES

La mise en œuvre du chiffrement DES sera celle décrite dans le document ISO DP 10126. Une valeur initiale de chaînage de 8 octets sera utilisée ainsi que le padding pour obtenir des champs à chiffrer de longueur multiple de 8 octets.

2.2 Scellement DES

La mise en œuvre du scellement DES sera celle décrite dans le document ISO DIS 8731. Le DES sera utilisé en mode CBC, avec une valeur initiale de chaînage nulle, et un padding à 0 binaire.

2.3 Ordre des opérations

Le scellement des données s'effectue sur les données en clair.

Le chiffrement s'effectue sur les données en clair (le sceau ne fait pas partie des données).

Le scellement et le chiffrement peuvent se faire article par article, ou sur la totalité du fichier. Dans le mode article par article, le padding se fait à la fin de chaque article, dans le mode calcul sur la totalité du fichier, le padding ne se fait qu'à la fin du fichier. Le mode choisi est indiqué dans les octets "mode opératoire" des paramètres "type de chiffrement" et "type de scellement". Le chiffrement et le scellement ne portent, pour ce qui concerne les données du fichier, que sur les contenus des articles ce qui signifie que dans le cas d'une FPDU multi-articles, les octets de longueur d'article ne sont pas pris en compte pour le scellement ni chiffrés.

Lorsque le calcul du sceau est effectué article par article, le sceau de l'article n sert de valeur d'initialisation pour le calcul du sceau de l'article n + 1. Dans le cas du transfert des sceaux partiels, ceux-ci seront transmis à l'occasion de l'envoi des points de synchronisation et porteront donc sur un ou plusieurs articles. Le sceau final - ou sceau de l'ensemble du fichier - sera par nature le sceau du dernier article du fichier.

La compression se fait après scellement et chiffrement. Le résultat du chiffrement étant une suite quasi aléatoire d'octets, la compression après chiffrement est vraisemblablement inutile. Il faut donc entendre que chiffrement et compression sont en pratique exclusifs l'un de l'autre.

2.4 Sécurité et relance

Dans le cas d'une relance de transfert, les mêmes éléments de chiffrement et de scellement sont utilisés pour les tentatives successives. Ces éléments peuvent ne pas être retransférés, lors de la relance. S'ils sont effectivement retransférés, ils doivent être identiques à ceux précédemment transmis. En particulier les vecteurs d'initialisation seront ceux initiaux et seront donc non significatifs pour la relance. Cette relance ayant lieu nécessairement à partir d'un point de synchronisation, donc d'une frontière d'article, l'émetteur comme le récepteur reprendront le calcul du sceau ou du chiffrement de la valeur courante correspondant à ce point de synchronisation.

2.5 Algorithme RSA

Les clés et modulo RSA, ainsi que tous les champs après chiffrement RSA, sont transportés comme des valeurs numériques (N), l'octet de plus fort poids en premier, l'octet de plus faible poids en dernier.

Dans les accréditations, les couples modulo et clé publique sont présent dans l'ordre : modulo (sur 64 octets, des zéros binaires remplissant les octets de plus fort poids si nécessaire) puis clé publique sur 2 octets.

Il n'y a pas de redondance avant chiffrement RSA. Toutes les données à chiffrer par RSA sont cadrées à droite et paddées par des zéros binaires.

Exemple 1 : format du PI 76 "éléments de chiffrement"

Avant chiffrement RSA ce champ est long de 16 octets :
Octets 1 à 8 : Clé de chiffrement (poids fort octet 1, poids faible octet 8)
Octets 9 à 16 : vecteur d'initialisation (poids fort octet 9, poids faible octet 16)

Pour le chiffrement RSA, ce champ est considéré comme une valeur numérique de 64 octets, dont les 48 octets de plus fort poids sont nuls, l'octet de plus fort poids éventuellement non nul étant l'octet de plus fort poids de la clé de chiffrement.

Exemple 2 : format du PI 79 "signature"

Avant chiffrement RSA ce champ est long de 16 octets :
Octets 1 à 8 : sceau FID (poids fort octet 1, poids faible octet 8)
Octets 9 à 16 : sceau FID et données (poids fort octet 9, poids faible octet 16)

Pour le chiffrement RSA, ce champ est considéré comme une valeur numérique de 64 octets, dont les 48 octets de plus fort poids sont nuls, l'octet de plus fort poids éventuellement non nul étant l'octet de plus fort poids du sceau FID.

Exemple 3 : format du PI 81 "accusé de réception de la signature"

Avant chiffrement RSA ce champ est long de 30 octets :
Octets 1 à 8 : sceau FID (poids fort octet 1, poids faible octet 8)
Octets 9 à 16 : sceau FID et données (poids fort octet 9, poids faible octet 16)
Octets 17 à 28 : date et heure (AAMMJJHHMMSS)
Octets 29 à 30 : ACK/NAK

Pour le chiffrement RSA, ce champ est considéré comme une valeur numérique de 64 octets, dont les 34 octets de plus fort poids sont nuls, l'octet de plus fort poids éventuellement non nul étant l'octet de plus fort poids du sceau FID.

2.6 Transformation *

Une transformation est parfois appliquée dans des modes d'utilisation impliquant uniquement DES et par conséquent en dehors du standard ETEBAC 5. C'est le cas, soit pour les aléas afin de contribuer à l'authentification réciproque, soit pour les vecteurs d'initialisation de chiffrement afin de permettre la vérification immédiate que les correspondants utilisent les mêmes clés de chiffrement de clés.

Cette transformation est notée*. La convention est :

A* signifie "A XOR FFFFFFFF00000000" dans laquelle :

A est une chaîne de 8 octets
XOR est l'opération ou exclusif
FFFFFFFF00000000 est le nombre de 8 octets dont les 4 octets de fort poids ont tous leurs bits à 1 et dont les 4 octets de faible poids ont tous leurs bits à 0.

3. UTILISATION DES ACCREDITATIONS

L'échange des accréditations a pour but de transmettre à son correspondant sa clé publique certifiée par l'Autorité.

Il est nécessaire de distinguer les bi-clés (clé publique, clé secrète) donc les accréditations contenant les clés publiques certifiées, utilisées pour des différentes fonctions de sécurité.

Les différents cas d'utilisations des bi-clés RSA sont donc :

1. échanges des nombres aléatoires (ALEAn) pour l'authentification réciproque : utilisation de la clé secrète émetteur
2. échanges des éléments de scellement chiffrés en RSA : utilisation de la clé publique récepteur
3. échanges des éléments de chiffrement chiffrés en RSA : utilisation de la clé publique récepteur
4. échange de la signature : utilisation de la clé secrète émetteur
5. échange de la deuxième signature : utilisation de la clé secrète émetteur
6. échange de l'accusé de réception de la signature : utilisation de la clé secrète récepteur

L'émission d'une donnée chiffrée sous la clé publique récepteur nécessite d'avoir au préalable reçu son accréditation.

L'émission d'une donnée chiffrée sous la clé secrète émetteur nécessite de transmettre aussi sa propre accréditation.

Les contraintes de sécurité imposent d'utiliser les bi-clés, donc des accréditations différentes pour les opérations d'accréditations et les opérations de transport de clés et de signature.

Cas de l'écriture de fichier (Demandeur/émetteur vers Serveur/récepteur)

Considérons un demandeur et un serveur quels que soient leurs statuts vis-à-vis du standard ETEBAC 5 (Client, Atelier, Banque, Prestataire).

Le Demandeur possèdera un bi-clé Pd1, Sd1 correspondant à l'accréditation **A <<Pd1>>** utilisé pour le chiffrement des ALEAn.

Le Demandeur possèdera un bi-clé Pd2, Sd2 correspondant à l'accréditation **A <<Pd2>>** utilisé pour la signature unique (ou première signature s'il y a double signature).

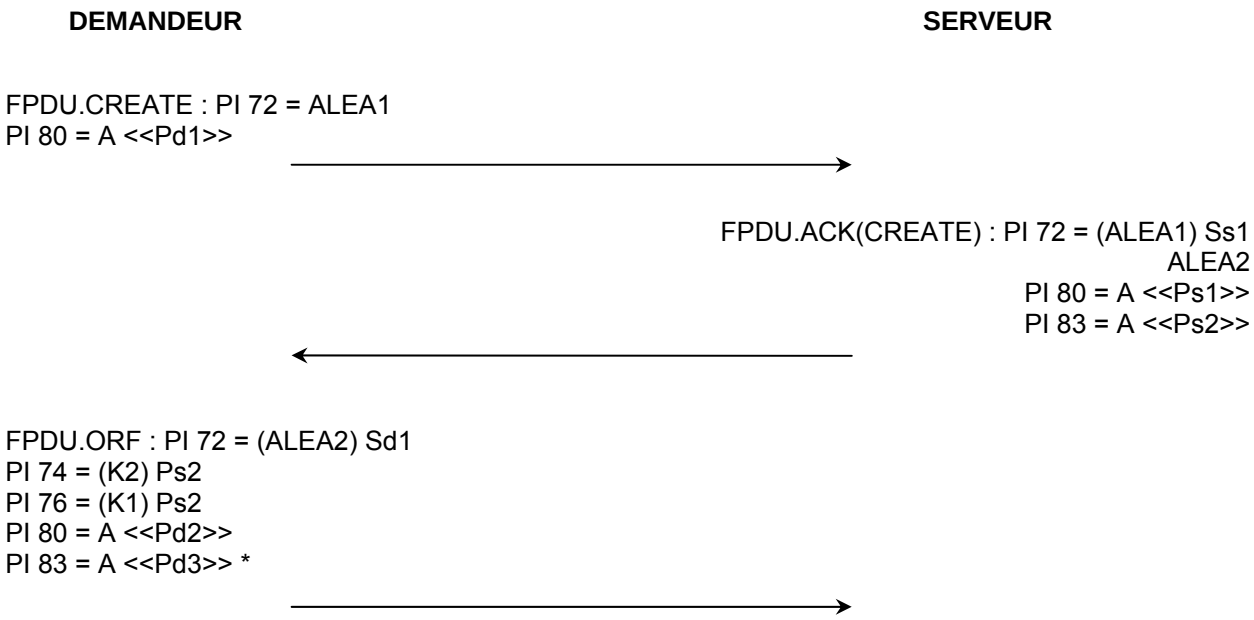
Le Demandeur possèdera un bi-clé Pd3, Sd3 correspondant à l'accréditation **A <<Pd3>>** utilisé pour la deuxième signature s'il y a double signature.

Le Serveur possèdera un bi-clé Ps1, Ss1 correspondant à l'accréditation **A <<Ps1>>** utilisé pour le chiffrement des ALEAn.

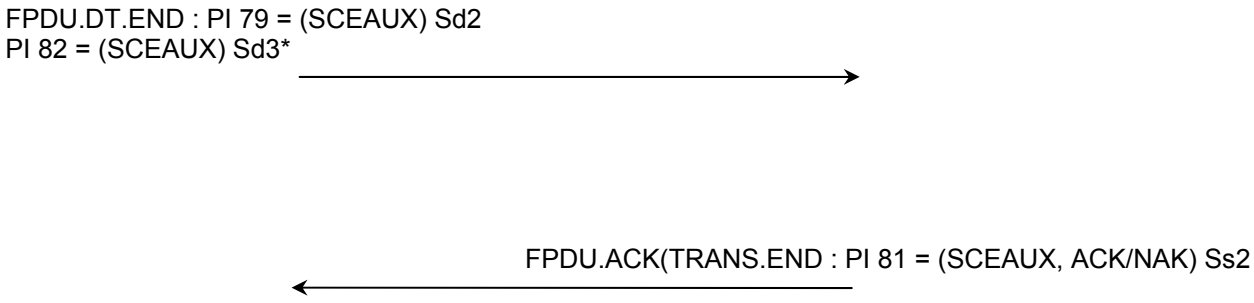
Le Serveur possèdera un bi-clé Ps2, Ss2 correspondant à l'accréditation **A <<Ps2>>** utilisé par le Demandeur pour le transport des éléments de chiffrement des éléments de scellement et par le Serveur pour le transport de l'accusé de réception de la signature.

Le schéma suivant récapitule les échanges en ne faisant apparaître que les paramètres concernés par les clés RSA. Il fait l'hypothèse d'un transfert avec authentification réciproque, non-répudiation réciproque et confidentialité.

ECRITURE DE FICHIER



TRANSFERT DU FICHIER



* Si double signature

Cas de la lecture de fichier (Demandeur/récepteur vers Serveur/émetteur)

Considérons un demandeur et un serveur quels que soient leurs statuts vis-à-vis du standard ETEBAC5 (Client, Atelier, Banque, Prestataire).

Le Demandeur possèdera un bi-clé Pd1, Sd1 correspondant à l'accréditation **A <<Pd1>>** utilisé pour le chiffrement des ALEAn.

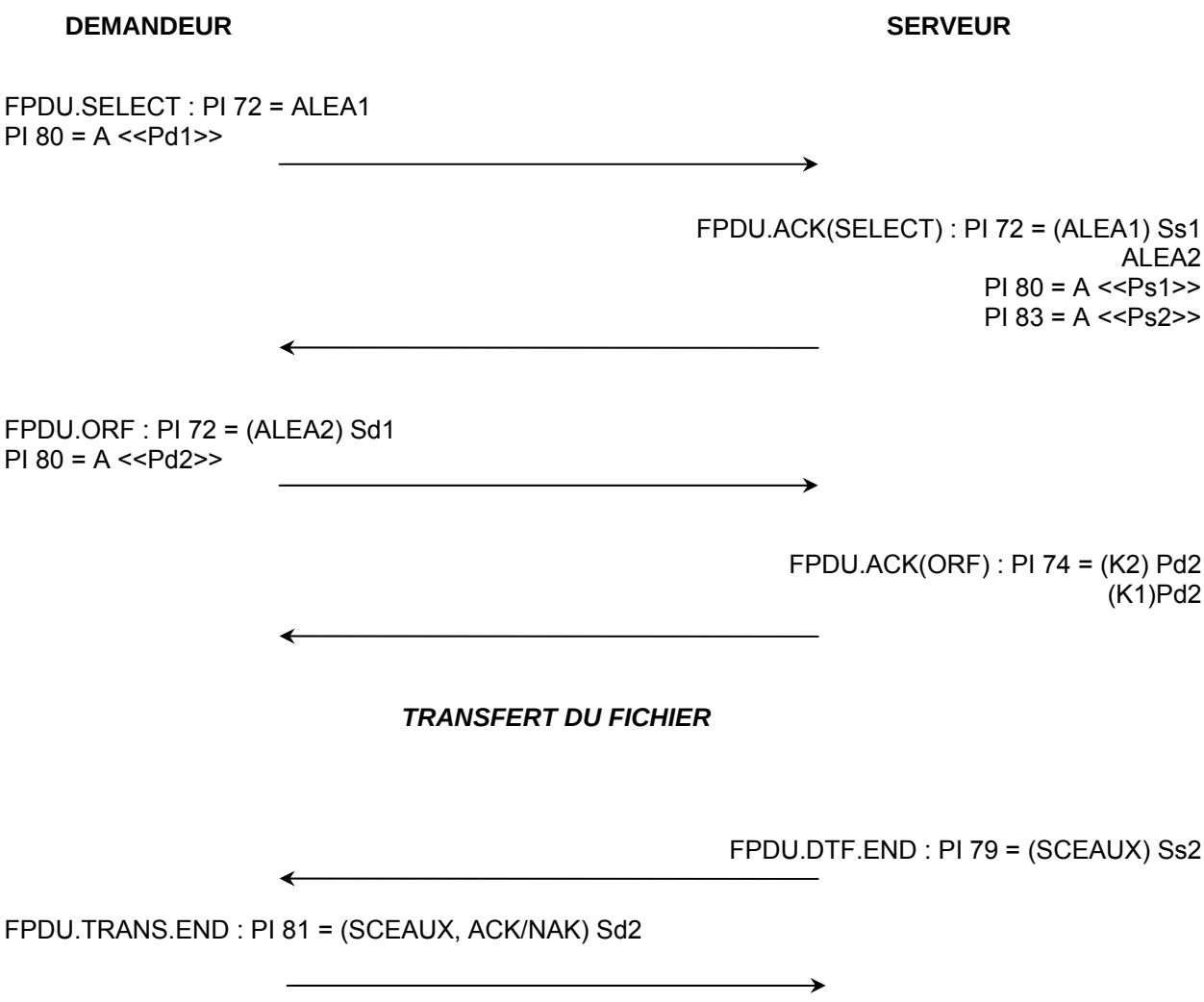
Le Demandeur possèdera un bi-clé Pd2, Sd2 correspondant à l'accréditation **A <<Pd2>>** utilisé par le Serveur pour le transport des éléments de chiffrement des éléments de scellement et par le Demandeur pour le transport de l'accusé de réception de la signature.

Le Serveur possèdera un bi-clé Ps1, Ps1 correspondant à l'accréditation **A <<Ps1>>** utilisé pour le chiffrement des ALEAn.

Le Serveur possèdera un bi-clé Ps2, Ps2 correspondant à l'accréditation **A <<Ps2>>** utilisé pour la signature unique (pas de double signature en lecture).

Le schéma suivant récapitule les échanges en ne faisant apparaître que les paramètres concernés par les clés RSA. Il fait l'hypothèse d'un transfert avec authentification réciproque, non-répudiation réciproque et confidentialité.

LECTURE DE FICHIER



TRANSFERT DU FICHIER

5 : DIAGNOSTICS D'ERREUR

Code diagnostic Type d'erreur	Code diagnostic Code raison	Raison	Elément service concerné
0	000	"Succès" : pas d'erreur	Tous
3	300	Congestion du système de communication local	F.CONNECT
3	301	Identification demandé inconnu	
3	302	Demandé non attaché à un SSAP	
3	303	Congestion système de communication distant (trop de connexions)	
3	304	Identification demandeur non autorisé (sécurité)	
3	305	Echec d'une négociation : - SELECT	
3	306	- RESYN	
3	307	- SYNC	
3	308	Numéro de version non supporté	
3	309	Trop de connexions déjà en cours pour ce CT	
3	321	Appeler le numéro de secours	
3	322	Rappeler ultérieurement	
3	399	Autres	
3	312	Fermeture du service demandée par l'utilisateur	F.RELEASE
3	313	Connexion rompue en fin d'intervalle d'inactivité TD	
3	314	Connexion inutilisée rompue pour accueillir une nouvelle connexion	
3	316	Connexion rompue à cause d'une commande de l'administration	
3	399	Autres	
3	304	Identification demandeur non autorisé (sécurisé)	F.ABORT
3	309	Trop de connexions déjà en cours pour ce CT	
3	310	Incident réseau	
3	311	Erreur de protocole PeSIT distant	
3	312	Fermeture du service demandée par l'utilisateur	
3	313	Connexion rompue en fin d'intervalle d'inactivité TD	
3	314	Connexion inutilisée rompue pour accueillir une nouvelle connexion	
3	315	Echec de négociation	
3	316	Connexion rompue à cause d'une commande de l'administration	
3	317	Echéance de temporisation	
3	318	PI obligatoire absent ou contenu illicite d'un PI	
3	319	Nombre d'octets ou d'articles incorrects	
3	320	Nombre excessif de resynchronisations pour un transfert	
3	399	Autres	

Code diagnostic		Raison	Elément service concerné
Type d'erreur	Code raison		
2	200	Caractéristiques du fichier insuffisantes	F.CREATE
2	201	Ressources système provisoirement insuffisantes	F.SELECT
2	202	Ressources utilisateur provisoirement insuffisantes	
2	203	Transfert non prioritaire	
2	204	Fichier existe déjà	
2	205	Fichier inexistant	
2	206	Réception du fichier causera un dépassement du quota disque	
2	207	Fichier occupé	
2	208	Fichier non vieux (antérieur à J-2 au sens SIT)	
2	209	Message de ce type non accepté sur l'installation référencée	
2	226	Refus de transfert	
2	299	Autres	
3	304	Identification demandeur non autorisée	
3	321	Appeler le numéro de secours	
3	322	Rappeler ultérieurement	
3	399	Autres	
2	210	Echec de négociation de contexte de présentation	F.OPEN
2	211	Ouverture de fichier impossible	
2	299	Autres	
2	212	Impossibilité fermeture normale fichier	F.CLOSE
2	299	Autres	
2	213	Erreur d'entrée/sortie bloquante	F.READ
2	214	Echec de négociation sur point de relance	
2	299	Autres	
2	213	Erreur d'entrée/sortie bloquante	F.WRITE
2	299	Autres	
2	213	Erreur d'entrée/sortie bloquante	F.DATA.END
2	214	Echec de négociation sur point de relance	F.CANCEL
2	215	Erreur propre au système	
2	216	Arrêt prématuré volontaire	
2	217	Trop de points de synchronisation sans acquittement	
2	218	Resynchronisation impossible	
2	219	Espace fichier épuisé	
2	220	Article de longueur supérieure à celle attendue	
2	221	Echéance du délai de fin de transmission	
2	222	Trop de données sans point de synchronisation	
2	299	Autres	

Code diagnostic		Raison	Elément service concerné
Type d'erreur	**Code raison**		
2	223	Fin de transfert anormal	F.TRANSFERT.END
2	224	La taille du fichier transmis est plus importante que celle annoncée dans le F.CREATE	F.DESELECT
2	225	Congestion de l'application station : le fichier a bien été reçu mais SCRS n'a pu le donner à l'application station	
2	299	Autres	
1	100	Erreur de transmission	F.RESTART
2	299	Autres	

6 : RECAPITULATIF DES FPDU ET PARAMETRES

LISTE DES FPDU

Code	Type de FPDU
00	FPDU.DTF
01	FPDU.READ
02	FPDU.WRITE
03	FPDU.SYN
04	FPDU.DTF.END
05	FPDU.RESYN
06	FPDU.IDT
08	FPDU.TRANS.END

11	FPDU.CREATE
12	FPDU.SELECT
13	FPDU.DESELECT
14	FPDU.ORF
15	FPDU.CRF
16	FPDU.MSG
17	FPDU.MSGDM
18	FPDU.MSGMM
19	FPDU.MSGFM

20	FPDU.CONNECT
21	FPDU.ACONNECT
22	FPDU.RCONNECT
23	FPDU.RELEASE
24	FPDU.RELCONF
25	FPDU.ABORT

30	FPDU.ACK(CREATE)
31	FPDU.ACK(SELECT)
32	FPDU.ACK(DESELECT)
33	FPDU.ACK(ORF)
34	FPDU.ACK(CRF)
35	FPDU.ACK(READ)
36	FPDU.ACK(WRITE)
37	FPDU.ACK(TRANS.END)
38	FPDU.ACK(SYN)
39	FPDU.ACK(RESYN)
3A	FPDU.ACK(IDT)
3B	FPDU.ACK(MSG)

40	FPDU.DTFMA
41	FPDU.DTFDA
42	FPDU.DTFFA

LISTE DES PARAMETRES

Code PI	Nom du paramètre
1	Utilisation du CRC
2	Diagnostic
3	Identification du Demandeur
4	Identification du Serveur
5	Contrôle d'accès
6	Numéro de version
7	Option-point de synchronisation
9 (PGI)	Identificateur du fichier
11	Type de fichier
12	Nom de fichier
13	Identificateur du transfert
14	Attributs demandés
15	Transfert relancé
16	Code-données
17	Priorité de transfert
18	Point de relance
19	Code fin de transfert
20	Numéro de point de synchronisation
21	Compression
22	Type d'accès
23	Resynchronisation
25	Taille maximale d'une entité de données
26	Temporisation de surveillance
27	Nombre d'octets de données
28	Nombre d'articles
29	Compléments de diagnostic
30 (PGI)	Attributs logiques
31	Format d'article
32	Longueur d'article
33	Organisation du fichier
34	Prise en compte de la signature
36	Sceau SIT
37	Label du fichier
38	Longueur de la clé
39	Déplacement de la clé
40 (PGI)	Attributs physiques
41	Unité de réservation d'espace
42	Valeur maximale de réservation d'espace
50 (PGI)	Attributs historiques
51	Date et heure de création
52	Date et heure de dernière extraction
61	Identifiant Client (émetteur initial)
62	Identifiant Banque (destinataire final)
63	Contrôle d'accès fichier
64	Date et heure du Serveur
71	Type d'authentification
72	Eléments d'authentification
73	Type de scellement
74	Elément de scellement
75	Type de chiffrement
76	Eléments de chiffrement
77	Type de signature
78	Sceau
79	Signature
80	Accréditation
81	Accusé de réception de la signature
82	Deuxième signature
83	Deuxième accréditation
91	Message
99	Message libre