MASTER EN SCIENCES INFORMATIQUES

Contribution à une réalisation d'interconnexion de systèmes ouverts

De Ghellinck, Didier

Award date:
1984

Awarding institution:
Universite de Namur

Facultés Universitaires Notre-Dame de la Paix - Namur

INSTITUT D'INFORMATIQUE

CONTRIBUTION

A UNE REALISATION

D'INTERCONNEXION

DE SYSTEMES OUVERTS.

Promoteur : Ph. Van Bastelaer

Mémoire présenté par

Didier de GHELLINCK

en vue de l'obtention du titre

de licencié et maître en informatique.

Année accadémique 1983-1984

Je tiens à remercier toutes les personnes qui m'ont aidé d'une quelconque façon dans la réalisation de ce mémoire.

Je tiens à remercier tout spécialement Monsieur le professeur Ph. van Bastelaer, dont les conseils m'ont aidé dans la rédaction de ce travail.

Je tiens à exprimer toute ma reconnaissance à l'équipe de la société SOCOMA-COMPUTERPLUS, et tout spécialement à Messieurs P. Vandendaele et D. Cornil, pour l'accueil qu'ils m'ont réservé, et pour l'assistance technique de tout instant qu'ils m'ont offerts, ainsi qu'à Monsieur A. Vandamme qui m'a permis de réaliser ce texte, se débatant avec les 239.041 caractères le composant.

Je remercie Monsieur J.C. Lienart pour les explications qu'il m'a apporté concernant l'architecture O.S.I., base de ce travail.

Je souhaite encore remercier Monsieur D. Larcy pour l'aide qu'il m'a apportée dans la réalisation d'interfaces physiques pour l'APPLE II, et Messieurs J. Blanchaert et J. Demarteau pour les conseils qu'ils m'ont donnés quand à la réalisation d'interfaces.

Je remercie également Mr Rigaux de l'Institut Belge de Normalisation, qui m'a donné accès à tous les documents nécessaires à ce mémoire.

Je remercie enfin mon épouse et mes enfants qui ont supporté l'invasion d'ordinateurs et de documents, et le crépitement incessant d'imprimantes, nécessaires à la réalisation de ce travail de fin d'études.

P L A N D U T R A V A I L

1. INTRODUCTION.

1.1 PROBLEME INITIAL.

Le point de départ de ce travail était l'étude et la réalisation d'un réseau local entre micro-ordinateurs de marque APPLE-II, cette réalisation intéressant autant la société SOCOMA, que l'Institut d'Informatique de Namur. Tous deux possèdent en effet un certain nombre de micro-ordinateurs de cette marque, travaillant tous de façon autonome, ce qui impose une duplication de tous les logiciels, périphériques, etc..., et en conséquence fait croître les frais d'adjonction d'un nouvel appareil et les frais de maintenance. La solution du réseau local offre comme avantage la possibilité de partager des imprimantes, des disques, etc...

De plus, il fallait que ce réseau soit exploitable commercialement, qu'il puisse etre mis entre des mains d'utilisateurs peu soucieux des problèmes techniques et exigeant un rapport qualité/prix optimal.

1.2. CHOIX POSSIBLES.

Réaliser un réseau complet, comprenant la partie matérielle et la partie logicielle, est un travail qui dépasse le cadre d'un mémoire de fin d'études. Aussi était-il nécessaire de se limiter à un des aspects de ce genre de réalisation. Deux possibilités s'offraient à nous :

- Réaliser un interface matériel permettant d'interconnecter les différents systèmes, utilisé avec quelques primitives de bas niveau. Ce genre de travail a déja été réalisé dans le cadre d'autres travaux de fin d'études (LEMAL), mais sur un type de machines différent.

- Greffer sur un réseau existant un ensemble de primitives d'échange de données, permettant une communication synchronisée entre les systèmes.

Limiter le travail à l'un de ces deux points paraissait beaucoup plus raisonnable. Il restait à déterminer lequel serait choisi, en fonction de contraintes temporelles et financières.

1.3. EVOLUTION VERS L'OBJECTIF.

La première démarche a été la rencontre avec des personnes qui, avec leurs moyens propres, réalisent des réseaux locaux. Nous y avons trouvé des réseaux simples, fonctionnant bien et à des prix très intéressants. Seulement ce genre de réseaux est très specialisé (pas de gestion d'adressage, transfert synchrone par rapport à un poste

maître, etc..), et non conforme à l'idée d'un réseau général. Mais grâce à l'expérience de ces personnes, nous avons pu réaliser des interfaces pour l'APPLE II, bon-marché et possédant d'assez bonnes performances (64 Kbps).

De là, un nouveau projet s'amorcait : essayer de regrouper une partie de la gestion d'un réseau en un système extérieur aux machines concernées (les APPLE-II), que nous appellerons "systèmes interfaces réseau" ou N.I.U. (de l'anglais Network Interface Unit). Ces N.I.U. possèderaient un processeur, de la mémoire, et des sorties (interfaces) d'une part vers le système auquel ils appartiennent, et d'autre part vers le réseau. Les avantages de travailler avec un système exterieur sont :

- d'une part que le système est le même quel que soit le type de machine, ce qui permet d'interconnecter d'autres marques de micro-ordinateurs, éventuellement des mini-ordinateurs, en n'ayant que peu de modifications de logiciel,

- et d'autre part que les micro-ordinateurs n'auront que peu de modifications matérielles à subir, l'interface entre eux et les N.I.U. étant constitué d'interfaces les plus standards possibles (RS-422 (ZAKS)).

Des contacts ont été pris avec une société spécialisée dans la réalisation d'interfaces, la société UNINA. Leur condition pour démarrer les frais de recherche pour la réalisation d'un N.I.U. était la suivante : il nous fallait développer un protocole (c'est à dire un ensemble de conventions de dialogue) solide, permettant d'interconnecter deux systèmes. Par la suite, un des deux systèmes pourrait être remplacé par un N.I.U.

Afin de réaliser un protocole ayant des chances d'être durable, nous nous sommmes penchés sur l'étude des normes existantes, basées sur l'architecture en sept couches proposée par I.S.O. (l'Organisation de Standardisation Internationalle). A ce moment commencaient à paraître les textes des projets de normes d'un certain nombre de couches. Précisons que pour devenir une norme, un texte est d'abord proposé comme avant-projet, après premières modifications le texte devient projet international, et après vote, devient norme internationale, le processus complet pouvant prendre plusieurs années (voir 4.3.1.).

Nous en sommes donc arrivés à étudier, puis à réaliser l'ensemble des normes telles que proposées par I.S.O.

1.4. OBJECTIF FINAL.

Ayant décidé de ne faire qu'une partie de la réalisation de tout un réseau local, mais de réaliser cette partie de facon complète, le but de la réalisation dans le cadre de ce travail s'est porté sur la mise en oeuvre des protocoles de haut niveau, en accord avec les termes des textes des projets de normes internationales.

Bien que ce sera défini ultérieurement, précisons déjà que la réalisation portera sur la mise en oeuvre des quatre couches supérieures de l'architecture O.S.I (Application, Présentation, Session, Transport) sur un micro-ordinateur APPLE-II, sous système d'exploitation UCSD-P. De plus, la destination de la réalisation sera orientée vers le traitement de fichiers tel qu'il a été défini par I.S.O sous l'appellation F.T.A.M. (File Transfer, Access and Management).

Par rapport à la réalisation d'un réseau complet, le travail portera sur la synchronisation des échanges d'informations, et se devrait d'etre encadré d'une part par un gérant de base de données (niveau supérieur), et d'autre part par la réalisation d'une connexion avec des N.I.U. (niveau inférieur).

1.5. PLAN DU TRAVAIL.

Ce travail est décomposé en deux parties :

- La première partie est consacrée à une **étude générale** concernant les réseaux locaux d'ordinateurs, dans laquelle le lecteur trouvera :

  - Les cas typiques d'utilisation de réseaux locaux de micro-ordinateurs, tels que ceux qui sont vendus par la société SOCOMA.

  - Les points forts et les points faibles des réseaux locaux par rapport à des solutions plus "classiques".

  - Une vue générale de différents réseaux locaux existant sur le marché.

  - Une étude plus approfondie de deux réseaux locaux représentatifs des choix possibles d'un acquéreur.

  - La position des instituts de normalisation par rapport au marché des constructeurs.

  - Une brève description de l'architecture proposée par ces instituts de normalisation.

- La deuxième partie décrit la **réalisation** proprement dite, et comprendra :

  - Les contraintes imposées par le micro-ordinateur APPLE-II, et les choix pris par rapport à l'architecture O.S.I.

  - Pour chaque couche de l'architecture, une description des interfaces et des protocoles, ainsi que des modifications à apporter en cas d'extensions.

- Les modifications et ajoutes à effectuer en cas d'integration dans un réseau complet.

- Une liste des mots-clés utilisés, avec la référence à la définition dans le texte.

- La bibliographie utilisée.

Le lecteur trouvera de plus en annexe différentes tables d'états relatives aux protocoles des différentes couches.

PARTIE I

E T U D E

2. GENERALITES SUR LES RESEAUX.

2.1. DEFINITIONS.

Nous donnons ici les définitions des différents termes utilisés dans la suite de ce chapitre. Elles ne sont citées que pour situer le sens que nous attribuons à ces termes. Le lecteur trouvera de plus amples détails dans (TANNENBAUM), (LEMAL), (COTTON), (SPANIOL), (MACCHI).

2.1.1. Système informatique.

Un système informatique, ou système, est l'ensemble des composants nécessaires au traitement de l'information. Sous ce terme on retrouve un ou plusieurs ordinateurs, dotés de leur unité centrale et de leurs processeurs auxiliaires, les périphériques associés (lecteurs de disques, imprimantes, etc..), éventuellement les opérateurs humains, et les logiciels nécessaires à leur fonctionnement.

2.1.2. Micro-ordinateur.

Un micro-ordinateur est un système mono-utilisateur, doté de tous les périphériques et du logiciel nécessaires à son fonctionnement autonome. Cet ordinateur peut être mono ou multi-tâche. Ce terme (ou son abrege : micro) sera le seul utilisé dans la suite de ce travail, et prendra le même sens que ordinateur personnel, ordinateur domestique etc..

2.1.3. Réseau d'ordinateurs.

Un réseau d'ordinateurs est un ensemble de deux ou plusieurs systèmes autonomes ayant la possibilité de s'échanger des informations. Dans la suite de ce travail, nous utiliserons le terme réseau pour réseau d'ordinateurs.

2.1.4. Réseau hétérogène.

Un réseau est dit hétérogène si les systèmes le composant sont de conceptions ou de marques différentes.

2.1.5. Réseau étendu.

Un réseau étendu, ou W.A.N. (de l'anglais Wide Area Network), est un réseau dans lequel les différents systèmes sont fortement éloignés les uns des autres : de plusieurs kilomètres à plusieurs milliers de kilomètres. Les moyens physiques de connexion généralement utilisés sont les réseaux téléphoniques nationaux, ou les transmissions par satellite.

2.1.6 Réseau local.

Un réseau local, ou L.A.N. (de l'anglais Local Area Network), est caractérisé par une distance peu élevée entre les différents systèmes composant le réseau : elle varie de quelques mètres à une dizaine de kilomètres. Les moyens physiques de connexion sont généralement le câble coaxial, la paire torsadée et maintenant la fibre optique.

2.1.7. Réseau en boucle.

La caractéristique d'un réseau en boucle est que chaque poste n'est connecté qu'à ses deux voisins immédiats. Chaque poste reçoit des informations d'un de ses voisins, et les retransmet à son autre voisin, après éventuellement avoir effectué un traitement sur les informations reçues. Divers protocoles existent, tels la méthode du jeton, des registres à décalage, etc...

2.1.8. Réseau hiérarchique (maître-esclave).

Dans ce type de réseau, toute communication d'un système à l'autre se fait via un système commun. On retrouve des réseaux à un seul niveau hiérarchique (réseau en étoile), ou à plusieurs niveaux (réseau arborescent). Ce genre de réseau ne présente guère d'avantages dans le cadre de L.A.N, aussi n'est-il que peu utilisé.

2.1.9. Réseau à bus commun.

Dans ce type de réseau, tous les systèmes communiquent au travers d'un moyen d'interconnexion unique (généralement un câble coaxial ou une paire torsadée). Divers protocoles permettent de vérifier que les différentes communications ne se brouillent pas mutuellement.

2.2. CRITIQUE DES RESEAUX LOCAUX.

Dans la suite, nous aborderons le problème précis de ce travail, à savoir le réseau local composé de micro-ordinateurs.

La critique des réseaux locaux de micros sera faite en comparant cette architecture avec une architecture plus traditionnelle, le mini-ordinateur. Nous verrons successivement leurs points communs, puis les avantages des L.A.N. de micros, et les avantages des mini-ordinateurs.

Cette comparaison permettra de déterminer dans quels cas un L.A.N. est préférable à un mini-ordinateur, et dans quels cas ce type d'architecture peut conduire à des déboires.

2.2.1. Points communs.

2.2.1.1. Mêmes types de logiciels

De par l'accroissement de puissance des micro-processeurs dont sont dotés les micro-ordinateurs, les logiciels qui jusqu'alors étaient réservés à des minis se retrouvent maintenant sur beaucoup de micros. Pour des applications "standards" (comptabilité, tenue des stocks, gestion des salaires...), on peut considérer que les outils de développement sont identiques (systèmes d'exploitation, compilateurs...).

2.2.1.2. Multi-utilisateurs

Le mini fonctionne en temps partagé, le réseau local est basé sur la décentralisation des traitements. Les deux types d'architecture peuvent partager des périphériques et des informations.

2.2.2. Avantages des L.A.N.

Nous verrons ici les avantages que procure le réseau local.

2.2.2.1. Extensibilité.

L'ajout d'un système sur un réseau n'est l'affaire que de quelques minutes. Il suffit généralement de connecter l'interface du nouveau système sur le câble d'interconnexion des autres systèmes. Grâce à celà, le réseau peut être reconfiguré à tout moment à la dimension des besoins des utilisateurs.

2.2.2.2. Fiabilité.

De par sa conception plus simple et son système d'exploitation plus épuré, un micro-ordinateur court moins de risques de tomber en panne qu'un mini-ordinateur. De plus, lorsqu'un incident survient sur un mini-ordinateur, tous les utilisateurs en sont affectés. Sur un réseau local (en supposant qu'il n'y ait ni maître ni esclave), seul l'utilisateur du système déficient en est affecté. La fiabilité étant considérée comme le rapport du temps d'absence de panne sur le temps total, on peut en conclure que la probabilité de panne risquant d'affecter un utilisateur d'un L.A.N. est moindre que celle d'un utilisateur d'un mini.

2.2.2.3. Adéquation.

Les types de micro-ordinateurs sur le marché sont très diversifiés, et pour chaque créneau d'application, on trouve tel ou tel système spécialisé. Pour faire du traitement de texte, on admet généralement qu'il est moins important d'avoir un processeur très puissant, mais la qualité de l'écran est importante. Pour du calcul numérique par contre, la rapidité et la précision de calcul du processeur sont primordiales. Dans chaque type d'application, il est possible de choisir le système approprié, et en plus, il est possible grâce au réseau de profiter de certaines caractéristiques des autres systèmes.

2.2.2.4. Dépannage.

Lorsqu'un micro-ordinateur tombe en panne, on le remplace habituellement par un modèle équivalent, et on le répare en atelier spécialisé. Le temps d'immobilisation est donc réduit au temps de remplacement du système. Rappelons de plus que les autres utilisateurs ne sont pas affectés par la panne. De par sa taille, un mini-ordinateur doit généralement être réparé sur place, nécessitant l'arrêt complet de la machine (et du travail des utilisateurs) jusqu'à réparation de la panne.

2.2.2.5. Prix.

Un des derniers arguments en faveur des L.A.N de micro-ordinateurs, et non un des moindres, est, à configuration équivalente, la différence de prix, qui oscille entre la moitié et les deux tiers du prix d'un mini-ordinateur.

2.2.3.Avantages des mini-ordinateurs.

Il ne faut quand même pas considérer les réseaux comme une panacée universelle. Les mini-ordinateurs ont un certain nombre de propriétés qui, à l'heure actuelle, leurs assurent certains avantages. Cependant l'évolution de la technologie est telle que ces avantages sont en train de disparaître progressivement.

2.2.3.1. Base de données unique.

Sur un mini-ordinateur, les bases de données sont centralisées, ce qui simplifie considérablement tous les problèmes de gestion de ces bases de données. Dans un réseau, par contre, les données pouvant être réparties sur différents systèmes, on retrouve tous les problèmes dûs aux bases de données réparties : duplication, redondance, absence... Il faut cependant noter que certains réseaux éludent ce problème en n'autorisant qu'un seul système gérant les bases de données (voir OMNINET au 3.3.).

2.2.3.2. Puissance.

Il existe encore un grand nombre d'applications qui nécessitent et un processeur puissant, et une vaste mémoire, des processeurs spécialisés... que l'on ne trouve pas (encore) sur des micros-ordinateurs. Nous pensons à des applications écrites en langage de très haut niveau tel SIMULA ou GPSS, dont il n'existe pas encore de versions supportées par des micro-ordinateurs.

2.2.3.3. Qualité des logiciels.

Longtemps limités par leurs capacités et leur puissance, les micro-ordinateurs n'ont que très récemment pu être dotés de logiciels de haut niveau (programmation linéaire...). Apparaissent maintenant des super-micros dont les capacités sont au moins équivalentes à celles de leurs frères ainés, les mini-ordinateurs.

2.3. CRITERES DE CHOIX.

Dans la comparaison qui précède, nous avons vu un certain nombre de critères techniques, desquels nous pourrons déterminer les cas ou un mini-ordinateur est plus approprié qu'un réseau de

micro-ordinateurs, ou le contraire.

Lorsque les applications désirées sont de type classique (application comptable, gestion des salaires, tenue de stocks..), ou sont diversifiées (traitement de texte, tableurs...), et que le nombre de postes n'est pas trop élevé, un réseau de micro-ordinateurs convient parfaitement, au vu de la différence de prix.

Par contre, si le genre d'application désiré est très particulier, si du développement doit être réalisé par le client grâce à des moyens consommateurs de ressources, un mini-ordinateur est plus approprié.

Mais à côté de ces critères techniques interviennent aussi des critères commerciaux. Pour beaucoup de gens, le micro-ordinateur reste le "home-computer", celui qu'utilisent les enfants pour accomplir leur rôle de vaillant guerrier de l'espace. La "vraie" informatique, elle, se fait sur des grosses machines. C'est donc le client qui choisira le type de configuration, selon son opinion sur les micro-ordinateurs.

D'une autre part, le domaine des réseaux locaux est nouveau (la première apparition d'ETHERNET date de 1976). L'expérience dans ce domaine est plutôt restreinte, et commercialement il peut y avoir des risques à proposer un réseau local dont on n'est pas tout à fait certain.

En conclusion, nous voyons que les progrès technologiques de ces dernières années nous permettent de supposer que les réseaux locaux sont destinés à un brillant avenir. Dans le chapitre suivant, nous verrons quelques réseaux locaux commercialisés, certains dédicacés aux micro-ordinateurs, d'autres destinés à interconnecter des mini-ordinateurs. Mais dans quelques années, lorsque micro-ordinateurs et mini-ordinateurs seront devenus des synonymes, sera-t-il possible de dissocier les termes "informatique" et "réseau" ?

3. ETUDE DE QUELQUES RESEAUX LOCAUX EXISTANTS.

Dans ce chapitre, nous analyserons un certain nombre de réseaux de façon très générale, mettant l'accent sur la technologie de ces réseaux locaux, ainsi que sur leur prix. Nous verrons ainsi que ces réseaux sont très performants, mais d'un prix inabordable pour interconnecter des micro-ordinateurs. Ensuite nous étudierons deux réseaux particuliers. Le premier est ETHERNET ayant servi de base à la réalisation de nombreux autres réseaux. Le second est le réseau OMNINET, spécialement dédicacé à l'interconnexion de micro-ordinateurs, et qu'il nous a été loisible de tester.

3.1. ETUDE COMPARATIVE DE RESEAUX LOCAUX.

Dans ce paragraphe, nous ferons une comparaison des caractéristiques principales de 34 réseaux locaux (XEPHON), (ZILOG1), (CORVUS), (ROWLANDS), (COTTON), (SPANIOL). Cette comparaison se fera sous forme d'une grille, ayant en ligne les différents réseaux, en colonne les caractéristiques comparées. Les différents prix mentionnés sont de 1983. Des comparaisons entre réseaux de constructeurs ont été faites dans (MEYER) (TAN).

Cette comparaison n'a pas pour but de déterminer des critères de sélection de tel ou tel type de réseau, mais elle est donnée à titre informatif, afin de pouvoir se faire une idée du marché des réseaux locaux. Cette liste n'est bien etendu pas exhaustive.

Voici la liste des critères de comparaison que l'on retrouve en abscisse dans la première grille ci-dessous (fig 3-1) :

- Nom du réseau : C'est la référence du produit.

- Date de la première installation : Date où le premier réseau a été opérationnel.

- Nombre de réseaux installés dans le monde : Ce nombre permet de se faire une idée de la commercialisation de ce réseau.

- Architecture : On retrouve ici les architectures classiques des L.A.N (voir 2.1.). On retrouve entre autres :

  - Le réseau bus : "bus".
  - Le réseau en anneau ou boucle : "anneau".
  - Le réseau en arborescence : "arbre".
  - Le réseau arborescent à un niveau : "étoile".
  - Le réseau maillé : "maillé".
  - Les lignes commutées : "switch".

- Moyen physique : C'est le moyen physique d'interconnexion : coaxial, paire torsadée (twis.pair.), cable téléphonique, fibre optique (f.opt.), etc...

- Nombre maximum de postes : Ce nombre est celui que l'on peut mettre sur un seul segment du réseau, sans tenir compte des répéteurs permettant de multiplier ce nombre.

- Protocole : Ce protocole correspond à celui de la couche Liaison de donnees de l'architecture O.S.I. On y retrouve :

  - CSMA/CD : accès multiple avec écoute préalable, et détection de collision (voir 3.2.1.).

  - CSMA/CA : Accès multiple avec écoute préalable, en évitant les collisions (voir 3.3.).

  - Jeton : la méthode d'autorisation de remplissage de trame par possession d'un jeton.

  - Tranche (slot) vide : Le temps est réparti en tranches; chaque tranche de temps peut contenir un message signalé par un bit sur le moyen physique d'interconnexion.

  - Insertion de registre : Chaque système possède un registre à décalage. Tant qu'il ne veut rien émettre, il transfere les messages de l'entrée à la sortie. Lorsqu'il veut émettre, les messages d'entrée sont insérés dans ces registres, pendant que le système émet en sortie. A la fin de son émission, il vide les registres à la sortie.

Dans la seconde grille (fig 3-2), nous retrouvons les éléments suivants :

- Vitesse du réseau : Il s'agit de la vitesse de transmission du réseau.

- Vitesse point à point : Il s'agit ici de la vitesse sur laquelle peuvent compter les utilisateurs de deux systèmes communicants. Dans le cas où la transmission est relayée par des N.I.U, cette vitesse peut être inférieure à la vitesse du réseau.

- Distance maximale : C'est la distance extrême tolérée par le réseau entre deux systèmes communicants.

- Prix moyen du réseau : C'est le prix nécessaire pour une configuration minimale. Il faut examiner ces chiffres avec la plus grande prudence : dans certains cas, les prix ne conviennent que pour juste deux interfaces, dans d'autres cas, lorsque les réseaux sont dédicacés à certaines machines précises, les prix comprennent l'acquisition de ces machines.

- Prix d'ajout d'un poste : C'est la somme à payer pour ajouter un élément au réseau.

Nom du réseau	date prem. inst.	nombre réseaux instal.	archi-tecture	moyen physique	nombre max. de postes	protocole
ARCNET	1975	4 500	arbre	coaxial	255	jeton
CABLENET	1974		bus	coaxial	16 360	TDMA
CLUSTER/ONE	1980	850	bus	torsadé	64	CSMA/CD
DATA RING	1979	30	anneau	twis.pair.	254	slot vide
ECONET	1980	250	bus	twis.pair.	100	CSMA/CD
GRAPEVINE	1980	100	switch	téléphone	illimt.	switch.
HYPERBUS	1977	200	bus	coaxial	512	CSMA/CA
HYPERCHANNEL	1977	200	bus	coaxial	1 000	CSMA/CA
IBX S/40	1981		étoile	téléphone	8 192	switch.
INTERLAN Eth.	1982	200	bus	coaxial	100	CSMA/CD
LOCALNET	1980	175	bus	coaxial	20 000	CSMA/CD
LOOSELY COUP.	1981		bus	coaxial		TRACE
MCZ+2			bus	coaxial	255	CSMA/CD
MEGALINK	1980	250	hd bus	coaxial	255	mod.fréq.
MILLWAY II	1980	15	bus	coaxial	100	ctr.cent.
NET/ONE	1980	200	bus	coaxial	8 000	CSMA/CD
OMNINET	1980	3 000	bus	twis.pair.	63	CSMA/CA
PACX IV	1970	1 600	maillé	coax,tw.p.	12 000	bit swit.

fig 3-1 (début) grille de comparaison de L.A.N.

Nom du réseau	date prem. inst.	nombre réseaux instal.	architecture	moyen physique	nombre max. de postes	protocole
PIXNET	1975	1 650	anneau	4 cables	72	SDLC
PLANET	1982		anneau	coaxial	500	jeton
POLYNET	1981	55	anneau	twis.pair.	250	slot vide
SDS 420/E02.B	1979	575	bus	coaxial	250	CSMA/CD
SENSION	1982	15	bus	coaxial	100	CSMA/CD
SERIES/1 LCC			anneau	twinax	256	ins. rég.
SILK	1980	3	anneau	coax&f.opt	8 400	ins. rég.
SYFANET	1983		bus	coaxial	16 384	CSMA/CA
TELEVIDEO	1981	1 300	étoile	twis.pair.	16	ctr.cent.
VIDEODATA	1972	300	bus	coaxial	10 000	modula.
WANGNET	1982	9	bus	coaxial	16 384	CSMA/CD
X+NET	1979	50	maillé	twis.pair.	8 160	polling
XEROX 8000	1981		bus	coaxial	1 024	CSMA/CD
XIBUS	1980	15	anneau	twis.pair.	4 095	ins. rég.
XODIAC	1980	80	bus	coaxial	32	jeton
Z-NET	1980		bus	coaxial	255	CSMA/CD
3COM ETHERNET	1981		bus	coaxial	1 024	CSMA/CD
8100 R+LOOP	1979	12 000	étoile	twin cab.	255	polling

fig 3-1 (fin) grille de comparaison de L.A.N.

Nom du réseau	vitesse du réseau	vitesse point à point	dist. max.	prix moyen réseau	prix ajout poste
ARCNET	2.5 Mbps	123 Kbps	7 Km	$30 000	$5 000
CABLENET	14 Mbps	19.2Kbps	70 Km	$15 000	$1 000
CLUSTER/ONE	250 Kbps	250 Kbps	300 m	$8 000	$500
DATA RING	3.2 Mbps	150 Kbps	10 Km	$12 000	$350
ECONET	250 Kbps	250 Kbps	500 m	$400	$200
GRAPEVINE	500 Kbps	9600 bps	ill.	$16 000	$800
HYPERBUS	6.3 Mbps	6.3 Mbps	800 m		$1 000
HYPERCHANNEL	50 Mbps	44 Mbps	1.5 Km	$100 000	$80 000
IBX S/40	1000 Mbps	57.6 Mbps	10 Km		
INTERLAN Eth.	10 Mbps	10 Mbps	2.5 Km		
LOCALNET	19.2 Kbps	128 Kbps	50 Km	$5 000	$800
LOOSELY COUP.	50 Mbps	36 Mbps	1 Km		
MCZ+2	800 Kbps	800 Kbps	2 Km	$35 000	$4 800
MEGALINK	1 Mbps	9 600 bps	10 Km	$6 000	$2 000
MILLWAY II	1 Mbps	100 Kbps	2 Km	$40 000	
NET/ONE	10 Mbps	9.2 Mbps	2.5 Km	$40 000	$500
OMNINET	1 Mbps	1 Mbps	1.2 Km	$5 000	$700
PACX IV	78 Mbps	9 600 bps	9 Km	$88 000	$1 900

fig 3-2 (début) grille de comparaison de L.A.N.

Nom du réseau	vitesse du réseau	vitesse point à point	dist. max.	prix moyen réseau	prix ajout poste
PIXNET	56 Kbps	56 Kbps	1 Km		
PLANET	10 Mbps	1 Mbps	50 Km		
POLYNET	10 Mbps	1.3 Mbps	14 Km	$13 000	$2 000
SDS 420/E02.B	1 Mbps	1 Mbps	500 m	$35 000	$9 600
SENSION	10 Mbps	10 Mbps	1.5 Km	$10 000	$5 000
SERIES/1 LCC	2 Mbps	2 Mbps	1.5 Km	$56 000	$400
SILK	16 Mbps	100 Kbps	75 Km	$2 400	$2 000
SYFANET	3 Mbps	56 Kbps	1 Km		
TELEVIDEO	800 Kbps	800 Kbps		$12 500	$2 500
VIDEODATA	300 Mbps	10 Mbps	50 Km	$ 500	$500
WANGNET	340 Mbps	12 Mbps	4 Km	$3 200	$500
X+NET	16 Mbps	800 Kbps	4 Km	$16 000	$600
XEROX 8000	10 Mbps	10 Mbps	2.5 Km		$700
XIBUS	10 Mbps	200 Kbps	400 m	$102 000	$500
XODIAC	2 Mbps	2 Mbps	1.5 Km	$60 000	
Z-NET	800 Kbps	800 Kbps	2 Km	$2 200	$900
3COM ETHERNET	10 Mbps	10 Mbps	2.5 Km	$3 800	$1 000
8100 R+LOOP	38.4 Kbps	38.4 Kbps	2 Km		

fig 3-2 (fin) grille de comparaison de L.A.N.

Au vu de ces grilles, on constate que l'architecture de loin la plus répandue est celle du bus, avec comme protocole de controle le "Carrier Sense, Multiple Acces, Collision Detection" (C.S.M.A/C.D), avec des vitesses de transmission variant de 250 Kbps à 10 Mbps. En seconde place, on retrouve, pour la même architecture, le protocole de controle "Carrier Sense, Multiple Acces, Collision Avoidance" (C.S.M.A/C.A.), méthode qui est dérivée du CSMA/CD. Le moyen physique d'interconnexion le plus utilisé est le câble coaxial, de par sa faible sensibilité aux parasites extérieurs.

Il nous a semblé intéressant d'expliciter plus cette methode de CSMA/CD, sur bus unique, au travers d'un réseau précis, le réseau ETHERNET, qui sera le sujet du chapitre suivant.

3.2 RESEAU ETHERNET.

Le premier réseau ETHERNET a été créé par la société XEROX en 1975 (METCALFE). A partir de 1980, ce sont trois sociétés, XEROX, INTEL et DIGITAL EQUIPMENT qui en ont spécifié la version actuelle. Outre son ancienneté, l'intérêt principal de ce réseau réside dans le fait qu'il a servi de base à la conception de maints autres réseaux locaux.

Par rapport à l'architecture O.S.I., les spécifications d'ETHERNET correspondent aux deux dernières couches du modèle de référence, la couche LIAISON DE DONNEES et la couche PHYSIQUE. Afin de respecter cette architecture, nous subdiviserons l'étude du réseau ETHERNET en l'étude de chacune de ces deux couches.

Note : Les différents concepts de l'architecture en couches seront décrits plus bas (voir 5.2.). La connaissance précise de ces concepts n'est pas nécessaire à la compréhension du reste de ce chapitre. On ne verra une couche que comme un modèle fournissant des services à la couche supérieure et dialoguant avec une entité homologue d'un autre système au moyen d'un protocole. Une entité d'une couche est un élément actif de cette couche.

3.2.1. Vue générale du réseau.

Le réseau ETHERNET, décrit en détail dans (XEROX), est basé sur l'utilisation d'un câble coaxial passif comme moyen physique d'interconnexion. Chaque poste connecté au réseau utilise la méthode dite "Carrier Sense, Multiple Access, Collision Detection". Voyons en détail les principes de cette méthode.

3.2.1.1. Carrier Sense.

Cette expression signifie que tous les postes sont à l'écoute du câble. Le câble est en effet passif, donc le test de présence d'un message se fait sur la "porteuse"

(un signal sur le câble).

3.2.1.2. Multiple Acces.

L'accès au câble est multiple, ce qui revient à dire que tous les postes peuvent émettre à tout moment, sans ordre préétabli.

3.2.1.3. Collision Detection.

Chaque poste en train d'émettre est capable de détecter si le signal qu'il émet est unique sur le câble, ou si deux ou plusieurs postes sont en train d'émettre simultanément. Si c'est le cas, l'émetteur arrête sa transmission, brouille volontairement le câble pendant quelques instants (de l'ordre de 32 à 48 bits à une vitesse de 10 Mbps) afin d'être certain que tous les systèmes soient conscients de la collision, puis, après une attente de durée aléatoire (un multiple compris entre 1 et 10 du temps de transmission de 512 bits à 10 Mbps), puis réessaye la transmission.

3.2.2. Services fournis par la couche Liaison de Données.

Les spécifications générales d'ETHERNET correspondent aux deux couches inférieures de l'architecture O.S.I. L'ensemble des couches supérieures (globalisé en couche "client") peut utiliser deux services de la couche LIAISON DE DONNEES d'ETHERNET :

Note : Dans un soucis de clarté, les descriptions se feront au moyen du langage PASCAL, supposé bien connu du lecteur.

Note : Nous parlerons ici de trame, utilisant un terme propre au protocole X.25 (CCITT1) du C.C.I.T.T. (voir 4.2.2.). Une trame est un ensemble d'octets de données transmis sur un moyen physique de connexion. Le terme exact ("unité de données de protocole de liaison de données") sera défini plus bas (voir 5.1.12.). Le terme "trame", plus connu, sera utilisé ici, afin de permettre au lecteur de se référencer à un langage qui lui est plus familier.

```
- FUNCTION TRANSMIT-FRAME
  (DESTINATION-PARAM : ADRESSVALUE;
   SOURCE-PARAM      : ADRESSVALUE;
   TYPE-PARAM        : TYPEVALUE;
   DATA-PARAM        : DATAVALUE)
   : (TRANSMIT-OK,EXCESSIVE-COLLISION-ERROR);

- FUNCTION RECEIVE-FRAME
```

```
(VAR DESTINATION-PARAM : ADRESSVALUE;
 VAR SOURCE-PARAM      : ADRESSVALUE;
 VAR TYPE-PARAM        : TYPEVALUE;
 VAR DATA-PARAM        : DATAVALUE)
 : (RECEIVE-OK,FRAME-CHECK-ERROR,ALIGNMENT-ERROR);
```

La première fonction sert à émettre des trames sur le réseau, la seconde à en recevoir. Nous allons voir successivement les différents arguments de ces fonctions.

3.2.2.1. Adresses.

Une adresse est représentée par une suite de 47 bits. Les adresses peuvent être de deux types :

- Adresse physique : Une adresse physique prend une valeur unique associée à la station, et est distincte de toutes les adresses des autres stations. Dans ce cas, le bit 48 est mis d'office à "1".

- Adresse de groupe : Lorsque le bit 48 est mis à "0", une station reconnait comme son adresse propre une adresse de groupe. Les différentes adresses de groupe doivent être préalablement déclarées pour chaque système. Une adresse composée des 48 bits mis à "0" est une adresse qui sera reconnue par tous les systèmes présents sur le réseau. Il est évident que l'adresse de groupe ne peut se retrouver dans le paramètre SOURCE-PARAM.

3.2.2.2. Type de trame.

Le type de trame se compose de 16 bits non controlés par la couche LIAISON DE DONNEES. La couche client pourrait par exemple utiliser ces 16 bits pour y mettre l'adresse et les commandes définies dans l'avis X25 du C.C.I.T.T. (CCITT1).

3.2.2.3. Données.

La zone de données se compose d'un nombre entier d'octets compris entre 46 et 1500. Cette zone sera transférée ou reçue telle quelle par la couche.

3.2.2.4. Résultat de l'émission.

Si le message a été correctement émis, la fonction retournera TRANSMIT-OK. Dans le cas contraire, cela voudrait dire que le câble de transmission est saturé, et qu'apres un certain nombre d'essais, la couche a abandonné l'espoir de pouvoir transmettre correctement le message, ce qui est indiqué par EXCESSIVE-COLLISION-ERROR.

3.2.2.5. Résultat de la réception.

Si le message a été correctement reçu, la fonction retourne la valeur RECEIVE-OK. Dans le cas contraire, soit il y a eu détection d'erreur dans le calcul du total de controle (FRAME-CHECK-ERROR), soit la couche n'a pas reçu un nombre entier d'octets (ALIGNMENT-ERROR).

3.2.3. Services fournis par la couche Physique.

La couche physique fournit comme services :

- la possibilité d'émettre un bit,
- la possibilité de recevoir un bit,
- la possibilité de détecter une collision,
- la détection de l'occupation du câble, et
- une fonction d'attente de (n) milisecondes.

3.2.4. Protocoles de la couche Liaison de Donnees.

Il n'entre pas dans le cadre de cette description de spécifier en détail les protocoles utilisés par la couche Liaison de Donnees. Voyons seulement les grandes lignes de son fonctionnement.

3.2.4.1. Emission

Lorsque l'entité reçoit une demande d'émission de trame (TRANSMIT-FRAME), elle va écouter la ligne jusqu'a ce qu'elle soit libre de signaux. A ce moment commence l'émission, où l'entité envoie en série la trame, augmentée d'un C.R.C.(Cyclic Redundancy Check ou vérification de redondance cyclique). Pendant toute l'émission, elle vérifie qu'il n'y ait pas de brouillage (collision), c'est à dire qu'une autre entité ne soit pas en train d'émettre. Si elle n'a pas détecté de collision, la procédure se termine et le résultat de la fonction est TRANSMIT-OK, sinon l'entité teste le nombre d'essais de transmission, et s'il est inférieur à une valeur fixée préalablement (16 dans le modèle général), elle attend un temps aléatoire (car sachant qu'une autre entité veut émettre, il s'agit de ne pas recommencer la transmission en même temps), puis elle réitere le processus d'émission. Si le nombre de tentatives dépasse 16 elle abandonne et renvoie EXCESSIVE-COLLISION-ERROR.

3.2.4.2. Réception.

Lorsqu'une fonction RECEIVE-FRAME est émise par la couche client, l'entité se met à l'écoute du câble. Si une transmission était en cours, l'entité attend la fin de la transmission. A ce moment, elle enregistre la trame suivante sur la ligne. Si l'adresse correspond à celle de l'entité, la fonction prend comme valeur soit ALIGNMENT-ERROR si le nombre d'octets reçu n'est pas entier, FRAME-CHECK-ERROR si le calcul du C.R.C. n'est pas le même que celui qui est reçus, et RECEIVE-OK sinon. Si l'adresse ne correspond pas, l'écoute recommence.

3.2.5. Avenir du Réseau ETHERNET.

Nous voyons que le principe du réseau ETHERNET est très simple. Ses performances relativement élevées (10 Mbps, 100 postes par segments, 3 segments pour un réseau) et sa fiabilité (SHOCH) en font un système particulièrement attractif.

Pour qu'un L.A.N. ait une chance de s'imposer sur le marché, il faut que les fabricants de processeurs construisent des processeurs dédicacés. C'est la seule façon d'arriver à imposer un standard, donc de réaliser des systèmes partiellement ouverts.

Deux des grands constructeurs de processeurs (INTEL et MOTOROLA) se sont penchés sur le problème et ont commercialisé des processeurs appropriés. Dans les deux cas, il s'agit d'un groupe de deux processeurs fonctionnant en tandem, le premier pouvant être associé à la couche physique (émission et réception d'un bit, détection de collision...), le second à la couche liaison de données (calcul du C.R.C., comparaisons des adresses...).

Chez INTEL (un des trois concepteurs de la version actuelle), ces processeurs ont comme références 82 501 et 82 586 (INTEL1), (INTEL2), (INTEL3), (INTEL4), et chez MOTOROLA, leurs références sont AM7990 et AM7991. Leur prix avoisine les 10 000 FB le groupe des deux processeurs.

3.2.6. Conclusions.

Nous avons vu un modèle de L.A.N. dont la diffusion comme standard ne fait pas de doutes. I.E.E.E. l'a d'ailleurs repris dans sa norme IEEE802 sur les réseaux locaux. Dans le cadre de ce travail, nous n'avons pas réalisé de couche liaison de données, mais nous auront toujours à l'esprit que c'est ce type de primitives de la couche liaison de données qui sera présente au bas de l'architecture.

3.3. RESEAU OMNINET.

3.3.1. Introduction.

Le réseau OMNINET a été conçu et est fabriqué par la firme americaine CORVUS à San Jose, en Californie (CORVUS).

CORVUS est une société construisant, entre autres, des disques durs de type "Winchester" pour micro-ordinateurs I.B.M., Apple, T.I.,etc... Ces disques possèdent des capacités de 6,12,18,30 et 40 Mégabytes. L'idée suivie dans OMNINET est de permettre à plusieurs micro-ordinateurs de partager un ou plusieurs de ces disques durs.

Nous allons donc voir un réseau bon marché, destiné uniquement à des micro-ordinateurs, et dont le but est de partager un ou plusieurs disques durs, afin de réduire les frais.

3.3.2. Caractéristiques techniques.

Le réseau OMNINET utilise un câble composé de deux conducteurs torsadés (câble téléphonique) véhiculant les informations à une vitesse de 1 million de bits par seconde. L'interface est de type RS422, c'est à dire une transmission différentielle (un conducteur à +5v, l'autre à 0v, alternance selon la valeur du bit). Ce type de conducteur permet des transmissions jusqu'a 1 200 mètres de distance.

Le protocole d'accès est de type CSMA/CA, comparable au CSMA/CD (voir 3.2.1.), si ce n'est que quand le canal est libre, un poste désireux d'émettre attend d'abord un laps de temps aléatoire. Cette methode permet théoriquement de réduire le nombre de collisions d'un facteur 10.

Le réseau est bâti autour d'un disque dur, les autres postes ne pouvant établir de communication qu'avec ce disque dur.

Au moment de l'installation, chaque poste se voit attribuer (matériellement) une adresse, comprise entre 1 et 63, le système ne supportant que 63 utilisateurs.

3.3.3. Caractéristiques du logiciel.

Le réseau OMNINET est fourni sous forme de boîte noire. L'utilisateur n'a accès à aucune des primitives du réseau. Le logiciel de gestion du réseau porte le nom de CONSTELLATION, et est partiellement transparent à l'utilisateur.

Nous ne retrouvons que les primitives destinées à gérer des fichiers, identiques à celles utilisées pour un disque souple. Chaque système possède son répertoire sur disque, et on y retrouve bien la politique de OMNINET : partager un disque dur, sans plus.

Il est toutefois possible que plusieurs postes partagent le même répertoire, ce qui leur permet d'utiliser une base de données commune. Mais le gérant du disque ignore ce fait, et c'est à l'utilisateur à bien gérer les accès. Pour celà, le logiciel lui offre un utilitaire de gestion de sémaphores, accessibles de tous les postes, dont un des buts est de permettre la gestion des accès concurrents à la base de données.

Un autre utilitaire fourni par CONSTELLATION est la gestion d'un ensemble de fichiers créés dynamiquement et accessibles de tous les postes.

3.3.4. Extensions prévues.

Un des modules annoncé par CORVUS est le "Utility server", permettant de contrôler une imprimante et/ou un modem.

Pour les copies de sécurité, CORVUS annonce un "BANK", mémoire de masse à cartouches d'une capacité de 200 Moctets, ayant un temps d'accès moyen de 20 secondes, à un prix n'excédant guère les 100 000 FB.

Afin de permettre à OMNINET d'être connecté à d'autres réseaux, CORVUS annonce la parution d'un "Gateway", permettant les conversions de protocole vers un S.N.A. (réseau d'I.B.M.), ETHERNET (voir ci-dessus), X25 (Norme du C.C.I.T.T) et d'autres via des interfaces RS232C.

3.3.5. Appréciation.

Nous avons testé ce réseau avec six APPLE III, exécutant des tâches les plus diverses possibles simultanément, tels des chargement de fichiers, des compilations, etc... A ce niveau, les temps de réponses ont toujours été très bons.

Par contre, les accès concurrents à des fichiers doivent être très soigneusement gérés, le gérant de disque n'agissant que comme si un seul poste etait connecté.

Le temps d'installation du réseau, ainsi que celui d'ajout d'un poste, est extrèmement rapide : il suffit de pincer à travers le câble un connecteur, d'installer la carte interface dans le micro-ordinateur, et le tour est joué.

Le manque de documentation est le point noir de ce réseau, qui par ailleurs, devra séduire tous ceux qui désirent partager

un disque dur entre plusieurs utilisateurs, à frais (très) réduits.

4. NORMALISATION.

4.1 INTRODUCTION.

A partir du moment où on veut interconnecter des systèmes, il s'agit de définir des protocoles ayant pour but la compréhension mutuelle des messages émis par les systèmes coopérants.

S'il s'agit d'interconnecter des systèmes d'un même fournisseur, le fournisseur définira lui-meme les règles du dialogue qu'utiliseront les systèmes communicant. A partir du moment où plusieurs constructeurs décident de faire communiquer leurs machines, il devient nécessaire de parvenir à une définition commune de ces règles.

Pour résoudre ce problème, les instituts de normalisation ont crée des normes. Une norme est (Larousse) "un principe servant de règle". Ces normes sont votées par les principaux intéressés, et dès lors doivent être appliquées par tous.

Nous verrons dans la suite de ce chapitre les différents instituts de normalisation participant à l'élaboration des normes, puis l'état actuel de la normalisation en matière d'interconnexion de systèmes, puis les règles définies (et celles non définies) de ces normes d'interconnexion. Ensuite, nous verrons l'accueil fait par les différents constructeurs à la parution de ces normes, ainsi que leur position vis-a-vis de celles-ci, et nous en tirerons quelques conclusions par rapport à la suite de ce travail.

4.2. INSTITUTS DE NORMALISATION.

Dans ce paragraphe, nous allons passer en revue les différents instituts de normalisation ayant contribué à l'élaboration de ces normes d'interconnexion des systèmes, avec une brève description de leur fonctionnement.

4.2.1. I.S.O.

L'I.S.O., ou International Standard Organisation (Organisation Internationale de Standardisation), est une institution n'ayant pas de siège propre, mais est une juxtaposition d'instituts nationaux de normalisation. Plus de 69 pays y adhèrent, parmi lesquels : A.N.S.I, pour les Etats-Unis, D.I.N. pour l'Allemagne, B.S.I. pour la Royaume-Uni, S.I.S. pour la Suède, J.I.S.C. pour le Japon, G.O.S.T. pour l'U.R.S.S., A.F.N.O.R. pour la France, I.B.N.-B.I.N. pour la Belgique, etc...

Lors de réunions de l'I.S.O., ayant lieu dans un des pays membres, participent d'une part les représentants des bureaux de

normalisation nationaux, et d'autre part tous ceux qui sont concernés par les normes discutées. On y retrouve principalement des constructeurs, mais également des organismes tels l'E.C.M.A ou le C.C.I.T.T (voir description plus bas).

Seuls les représentants des instituts de normalisation nationaux y ont droit de vote (un par pays), les autres participants pouvant uniquement faire des suggestions. Un vote ne peut être négatif que s'il est accompagné d'une contre-proposition valable. Aussi le vote d'un texte de norme est-il précédé d'avant-projets distribués dans les pays membres longtemps à l'avance, afin que ceux-ci puissent l'étudier et préparer les contre-propositions éventuelles.

La façon dont sont organisées les décisions entre les représentants des instituts de normalisation nationaux et les constructeurs relève de l'organisation nationale. Dans certains pays, les instituts ont des commissions d'études, dans d'autres, il s'agit de simples contacts entre instituts de normalisation et constructeurs.

4.2.2. C.C.I.T.T.

Le C.C.I.T.T. ou Comité Consultatif International des Télégraphes et Téléphones est un institut international dont les membres sont les représentants nationaux des organismes des P.T.T. Son fonctionnement est semblable à celui de l'I.S.O., exception faite que les participants ayant droit de vote ne sont plus les représentants des instituts de normalisation, mais les représentants des P.T.T.

Participent également aux réunions du C.C.I.T.T. les entités de gestion des télécommunications lorsqu'il n'y a pas de monopole des communications (cas entre autre des Etats-Unis), et des fournisseurs de matériels.

A l'origine, le C.C.I.T.T. ne s'occupait que des moyens de bas niveau d'interconnexion, alors que l'I.S.O se chargeait de normaliser les procédures de plus haut niveau. Progressivement, le C.C.I.T.T. a commencé à normaliser tous les protocoles d'interconnexion, dont un exemple est le Télétex (CCITT2). Certains autre projets de normes de l'interconnexion des systèmes, émis par I.S.O., ont été intégralement repris par le C.C.I.T.T, qui les utilise comme si ces projets étaient leurs propres normes.

Cette lutte d'influence est d'autant plus marquée que chacun des deux organismes n'a pas droit de vote chez l'autre.

4.2.3. E.C.M.A.

L'E.C.M.A, ou European Computer's Manufacturers Association

(Association des Constructeurs Informatiques Européens) est un regroupement privé des principaux constructeurs européens : Bull, IBM Europe, I.C.L., Philips, Siemens, Olivetti, Nixdorf, etc...

Au niveau de l'interconnexion des systèmes, les réunions de l'E.C.M.A. ont pour but de parvenir à des accords informels entre les différents partenaires. Ces accords sont exprimés sous forme de normes, n'ayant aucune valeur stable, mais étant des propositions précises et complètes qui sont soumises tant à l'I.S.O. qu'au C.C.I.T.T.

En cas de refus de ces propositions, l'E.C.M.A. modifie toujours en conséquence ses normes, qui sont donc toujours alignées sur les normes internationales d'I.S.O. ou du C.C.I.T.T. Une norme E.C.M.A n'est donc pas une norme stable, mais plutôt un bon document de travail.

4.2.4. C.E.P.T.

Le C.E.P.T., ou Comite Européen des Postes et Téléphone, est un regroupement informel des organismes de P.T.T. européens, définissant des lignes de conduite à tenir lors de réunions du C.C.I.T.T.

4.2.5. Commissions Européennes.

Elles promouvoient certains aspects de l'I.S.O, dans le cadre de certains projets Européens, tel le projet ESPRIT.

4.3. ETAT ACTUEL DE LA NORMALISATION.

Dans ce chapitre, nous verrons d'abord les différentes étapes de conception d'une norme chez I.S.O, ensuite l'état d'avancement des travaux, puis finalement les références de base utilisées pour la réalisation.

4.3.1. Conception d'une norme.

Lorsque le besoin d'une norme se fait sentir, une demande est émise au secrétariat de l'I.S.O., qui dans un premier temps établit des groupes de travail pour l'étude du projet. Ces groupes de travail sont constitués d'experts dans le domaine, de plusieurs nationalités. L'E.C.M.A., pour ne citer que lui, se constitue typiquement comme groupe de travail.

Pendant l'étude du projet, les diverses étapes sont reprises dans des notes de service (prefixées par "N"), du sous-groupe concerné de l'I.S.O. Dans le cas de l'interconnexion des systèmes, la référence est composée de TC97, référence de ce qui

a trait à l'informatique, SC16 pour ce qui concerne l'interconnexion. Ces différentes notes de service n'ont qu'un usage interne.

Lorsque le projet semble être mûr, on édite un avant-projet référencé "D.P." (de l'anglais Draft Proposal), qui est soumis aux critiques des différents membres. A partir de ces critiques, on révise l'avant-projet aussi longtemps qu'il y a de grosses objections.

Au moment où le projet semble prendre sa forme définitive, on change l'ancienne référence "Revised D.P." en projet international de standardisation, en abrégé "D.I.S." (de l'anglais Draft International Standard).

Si le vote de la norme est positif, nous avons alors un "I.S", c'est à dire un standard international (de l'anglais International Standard), qui est soumis à la plus grande diffusion possible.

4.3.2. Etat d'avancement des travaux.

C'est en 1979 qu'ont débuté les projets de normalisation de l'interconnexion de systèmes. Depuis 1980,l'architecture de base en sept couches (voir 5.2.) a déjà été votée. Pour chacune des couches, une définition des services et des protocoles est en cours d'élaboration. Pour les couches transport et session, on en est au stade de projet international.

Au niveau de la couche application, trois grands projets ont été décrits :

- La définition d'un protocole de terminal virtuel, ou V.T.M. (de l'anglais Virtual Terminal Management).

- La Manipulation et Transfert de processus ou J.T.M. (de l'anglais Job Transfer and Manipulation), à l'état de notes de service.

- La gestion et l'accès aux fichiers, ou F.T.A.M (de l'anglais File Transfer, Acces and Management) en est à l'état d'avant-projet.

La définition de la couche présentation n'est qu'à l'état de notes de service.

Les définitions des couches inférieures ne sont pas faites, I.S.O. préférant utiliser les normes déjà définies dans le cadre des télécommunications. En fait, le protocole X.25 du C.C.I.T.T (CCITT1), (GRANT) semble être admis comme un des grands standards des couches inférieures.

Beaucoup d'annexes à l'architecture de base sont également en

cours d'élaboration, à l'état de notes de service.

On garde l'espoir que l'ensemble des normes d'interconnexion des systèmes ouverts soit voté pour 1990.

4.3.3. Références Utilisées.

- IS 7498 : Modèle de base.
- DIS 8072 : Définition du Service Transport.
- DIS 8073 : Définition du Protocole de Transport.
- DIS 8326 : Définition du Service Session.
- DIS 8327 : Définition du Protocole de Session.
- DP 8571/1 : Description générale du F.T.A.M.
- DP 8571/2 : Stockage de fichiers virtuels.
- DP 8571/3 : Définition du Service de F.T.A.M.
- DP 8571/4 : Définition du Protocole de F.T.A.M.
- N1666 : Définition du Service Présentation.
- N1667 : Définition du protocole de Présentation.
- N1740 : Définition du Service Réseau.

4.4. CONTENU DES NORMES.

Les normes I.S.O. de l'interconnexion des systèmes définissent les protocoles nécessaires au dialogue entre des systèmes ouverts. Le but de ces normes n'est pas de décrire une mise en oeuvre, mais bien de guider le concepteur dans la réalisation d'un système ouvert.

De plus y sont définis les traitements d'erreurs de protocole, en cas de parametre invalide, de collision, d'erreur de séquencement etc...

Tous les modèles sont fournis sous forme de tables d'états, reprennant chaque action à exécuter selon un état et un évènement entrant.

La création d'une norme est en fait un ensemble de concessions réalisées par chacune des parties prenantes (en général les différents constructeurs ayant déja réalisé pour leur propre compte des protocoles fonctionnels). Ainsi, on en arrive à des normes complexes,

possédant un grand nombre d'options négociées lors de l'établissement d'une connexion. Ce processus de négociation d'options rend très complexe la mise en oeuvre de ces normes.

Le modèle décrit par les normes ne fait référence à aucune mise en oeuvre particulière. Un certain nombre de problèmes techniques se posent suite aux diverses technologies utilisables, et sont donc laissés à l'appréciation du réalisateur, ce qui peut engendrer des interprétations différentes d'un réalisateur à l'autre.

En conclusion, ces normes définissent parfaitement les protocoles devant se dérouler lorsque ne surgit aucun problème dû à la technologie, mais sont muettes sur les traitements à effectuer en cas de déficiences dûes à celles-ci.

4.5. REACTION DES CONSTRUCTEURS.

Il est bien évident qu'aucun constructeur n'a intérêt pour lui-meme à suivre une quelconque normalisation. On retrouve trois catégories de position :

- Certains constructeurs ont leurs propres normes, et ont intérêt à freiner toute standardisation.

- D'autres suivent les constructeurs plus puissants qu'eux (et un en particulier), et n'ont que faire de la normalisation, préférant utiliser les normes du plus puissant.

- Il reste une catégorie de constructeurs qui suivent les normes afin de pallier à leur manque de développement dans ce domaine.

4.6. CONCLUSIONS.

Nous voyons ainsi que commercialement, l'impact de la normalisation de l'interconnexion des systèmes risque d'être assez réduit. Le modèle architectural en sept couches est sur toutes les lèvres, malheureusement il arrive à un moment où beaucoup de constructeurs se sont déja penchés sur le problème et ont établis leurs propres standards.

Dans le cadre d'un travail de fin d'études, il nous a paru quand même intéressant de vérifier la faisabilité et la cohérence des différentes normes, et en particulier leur mise en oeuvre sur des micro-ordinateurs.

Aussi est-ce à titre expérimental que l'essai de mise en oeuvre de l'architecture d'interconnexion des systèmes ouverts a été réalisé.

5.SPECIFICATIONS GENERALES.

Dans ce chapitre, nous définirons les termes utilisés dans le cadre de l'architecture en couche O.S.I., puis nous verrons brièvement les différentes fonctions de chaque couche, et ensuite les conventions de services utilisées tout au long de ce travail. La description complète de ce qui suit se trouve en (ISO1).

5.1. DEFINITIONS DES CONCEPTS ARCHITECTURAUX.

5.1.1. Système ouvert :

Des systèmes sont dits ouverts s'ils communiquent entre eux en utilisant les spécifications des normes O.S.I.

5.1.2. (n)-Couche :

Un système est composé de sous-systèmes, classés hiérarchiquement. Les sous-systèmes d'un même rang dans des systèmes différents forment une couche. Les sous-systèmes adjacents communiquent via leur frontière commune. Une (n)-couche est un sous-système de rang n. La fig. 5-1 représente cette structuration en couches.

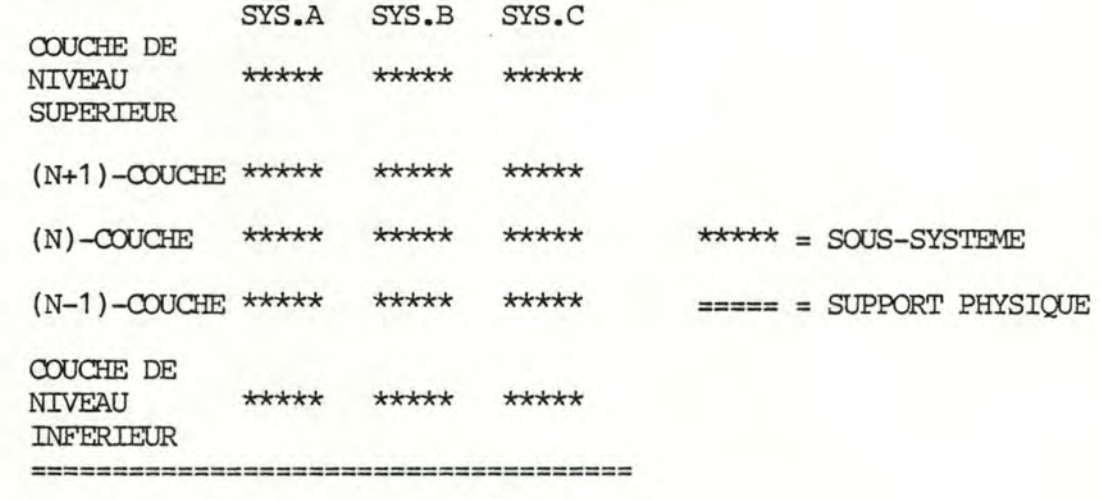


fig 5-1 Structuration en couches d'O.S.I.

5.1.3. (n)-Entité :

Une (n)-entité est un élément actif d'une (n)-couche.

5.1.4. (n)-Service :

Chaque (n)-couche fournit des (n)-services aux (n+1)-entités de la (n+1)-couche. Ces services permettent aux (n+1)-entités de coopérer. Ces (n)-services sont effectués à l'intérieur de la (n)-couche, qui s'appuie, au besoin, sur les (n-1)-services de la (n-1)-couche.

5.1.5. (n)-Connexion :

C'est une association dans une (n)-couche entre deux ou plusieurs (n)-entités.

5.1.6. (n)-Protocole :

C'est un ensemble de règles et de formats déterminant les caractéristiques de communication entre des (n)-entités.

5.1.7. (n)-Adresse de point d'accès des services ou (n)-S.A.P. :

La (n)-adresse de points d'accès des services, ou (n)-adresse, identifie le point d'accès à des (n)-services auquel une (n+1)-entité est liée. Une (n)-adresse comprend deux parties :

- La (n-1)-adresse de la (n)-entité à laquelle la (n+1)-entité est reliée par un (n)-point d'accès à des services, et

- un (n)-suffixe, permettant l'identification exclusive du point d'accès à des services dans le contexte de la (n-1)-adresse.

5.1.8. (n)-Identificateur d'extrémité de connexion :

Quand une (n+1)-entité établit une (n)-connexion avec une autre (n+1)-entité, chaque (n+1)-extrémité se voit attribuer un (n)-identificateur d'extrémité de connexion par la (n)-entité qui la sert, permettant ainsi de distinguer la (n)-connexion de toutes les autres (n)-connexions accessibles à partir du (n)-point d'accès à des services. Cet identificateur d'extrémité de connexion comporte deux parties :

- La (n)-adresse du (n)-point d'accès à des services, auquel est associée la (n)-connexion, et

- Un (n)-sufixe d'extrémité de connexion, unique dans le contexte du point d'accès à des (n)-services.

5.1.9. (n)-Informations de contrôle de protocole :

Il s'agit d'informations transférées entre deux (n)-entités, pour coordonner leurs opérations conjointes.

5.1.10. (n)-Données utilisateur :

Ce sont des données transférées entre deux (n)-entités à destination des (n+1)-entités, pour lesquelles les (n)-entités fournissent des services.

5.1.11. (n)-Unités de données.

Les unités de données sont un ensemble homogène de données.

5.1.12. (n)-Unités de données de protocole ou (n)-P.D.U. :

Ce sont des unités de données qui comprennent des (n)-données utilisateurs et des (n)-informations de contrôle de protocole.

5.1.13. (n)-Unités de données de service ou (n)-S.D.U. :

Ce sont des unités de données transférées à travers le point d'accès des services entre une (n+1)-entité et une (n)-entité en une seule opération.

5.1.14. (n)-Unités de données express de service :

Ce sont des petites (n)-unités de données de service dont le transfert est effectué de façon prioritaire par rapport aux (n)-unités de données de service.

5.2. DESCRIPTION DES COUCHES O.S.I.

Dans ce paragraphe, nous verrons d'abord les principes de décomposition en couches, puis selon ces critères la détermination des différentes couches, et ensuite une brève description du rôle de chacune de ces couches.

5.2.1. Principes de décomposition.

* P1 : Ne créer que le nombre nécessaire de couches, afin de simplifier l'intégration des différentes couches.

* P2 : Créer des frontières là où la description des interfaces est concise, et le nombre d'interactions au travers de cette frontière est réduit.

* P3 : Créer des couches là où les fonctions diffèrent manifestement au niveau technologie ou en type de traitement.

* P4 : Rassembler les fonctions similaires dans une même couche.

* P5 : Choisir des frontières là où l'experience a montré que c'était efficace.

* P6 : Créer une frontière où il est utile d'avoir un interface standardisé.

* P7 : Créer une couche là où l'on trouve un niveau différent d'abstraction dans le traitement des données.

5.2.2. Critères de détermination de l'architecture.

5.2.2.1. Couche physique.

Afin de permettre la plus grande variété possible de moyens physiques d'interconnexion, l'application des principes P3, P5 et P7 a conduit à identifier la COUCHE PHYSIQUE comme la couche la plus basse de l'architecture.

5.2.2.2. Couche liaison de données.

Les moyens physiques de communication nécessitent des techniques spécifiques pour transmettre des données entre les systèmes. Ces techniques ont été déja étudiées et standardisées (voir 3.2.5.). L'application des principes P3, P5 et P7 a conduit à identifier la COUCHE LIAISON DE DONNEES au dessus de la couche physique.

5.2.2.3. Couche réseau.

Certains systèmes agissent comme destination finale des données, d'autres comme noeuds intermédiaires. L'application des principes P3, P5 et P7 a conduit à identifier la COUCHE RESEAU au dessus de la couche liaison de données. Les protocoles orientés réseau, tels le routage, seront regroupés dans cette couche.

5.2.2.4. Couche transport.

Le contrôle de l'acheminement des données entre le système terminal source et le système terminal destination est la dernière fonction à exécuter pour fournir la totalité du service de transport des données. On trouve dès lors au dessus de la couche réseau la COUCHE TRANSPORT, qui décharge les couches supérieures de tout ce qui a trait au transport des données entre elles.

5.2.2.5 Couche session.

Afin d'organiser et de synchroniser le dialogue, et gérer les échanges de données, l'application des principes P3 et P4 a conduit à identifier la COUCHE SESSION au dessus de la couche transport.

5.2.2.6. Couche présentation.

Le reste des fonctions est lié à la représentation et la manipulation de données structurées. Par l'application des principes P3 et P4, on situe la COUCHE PRESENTATION au dessus de la couche session.

5.2.2.7. Couche application.

Les programmes d'application effectuent du traitement de l'information. L'aspect de ces processus d'application et les protocoles par lesquels ils communiquent permettent d'identifier la COUCHE APPLICATION comme la couche supérieure de l'architecture.

Obéissant aux principes P1 et P2, l'architecture résultante est formée de sept couches, représentées à la fig 5-2.

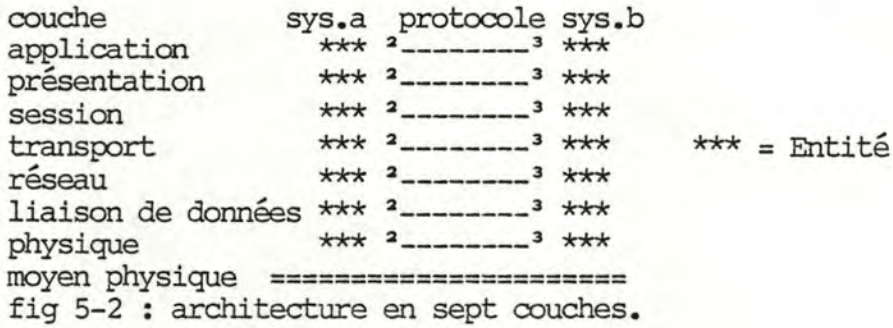


fig 5-2 : architecture en sept couches.

5.2.3. Rôle des sept couches dans l'architecture.

5.2.3.1. Couche application.

Le rôle de cette couche est de servir de fenêtre entre les entités d'application utilisant l'O.S.I. pour échanger des informations. Chaque fois que des communications ont lieu, c'est au travers de la couche application que tous les paramètres concernant un processus d'application sont portés à la connaissance de l'environnement O.S.I.

5.2.3.2. Couche présentation.

La couche présentation se charge de la représentation des informations que des entités d'application se communiquent, ou auxquelles elles se réfèrent au cours de leur dialogue. Elle ne s'occupe que de l'aspect syntaxique, et non de l'aspect sémantique. Les entités d'application peuvent utiliser n'importe quelle syntaxe, la couche présentation assurant la transformation entre cette syntaxe et la syntaxe commune de transfert.

On peut ainsi avoir simultanément trois syntaxes lors d'échanges d'informations :

- La syntaxe utilisée par l'entité application émettrice.

- La syntaxe utilisée par l'entité application réceptrice.

- Une syntaxe commune de transfert.

5.2.3.3. Couche session.

Le rôle de la couche session est de fournir aux entités présentation coopérantes les moyens nécessaires pour organiser et synchroniser leur dialogue, et pour gérer leurs échanges de données. Elle fournit à chacun de ses utilisateurs les moyens nécessaires à :

- Etablir une connexion avec un autre utilisateur du service de session, échanger des données avec celui-ci de façon synchronisée, et libérer de façon ordonnée la connexion.

- Négocier l'utilisation de jetons donnant droit à l'échange de données, à la synchronisation et à la libération de la connexion. Un jeton est un attribut d'une connexion de session qui est dynamiquement attribué à un utilisateur du service session, à un moment déterminé, pour faire usage des services décrits ci-dessus.

- De poser des points de synchronisation au cours du dialogue et, en cas d'erreur, de reprendre le dialogue à partir d'un point de synchronisation convenu.

- D'interrompre un dialogue et de le reprendre ultérieurement à un moment déterminé à l'avance.

5.2.3.4. Couche transport.

Le service transport assure un transfert transparent des données entre utilisateurs du service transport. Il libère ces utilisateurs de toute préoccupation concernant

les moyens détaillés mis en oeuvre pour réaliser ce transfert. Le service transport assure :

- Le choix de la qualité de service, afin d'optimiser l'utilisation des ressources de communication disponibles.

- L'indépendance par rapport aux ressources des couches de niveaux inférieurs.

- La signification de bout en bout : le service transport assure le transfert des données échangées entre utilisateurs du service transport jusqu'au niveau des systèmes finals.

- La transparence des informations transférrées. Elle n'impose aucune restriction au contenu, format ou codage des informations ou données, et n'a même pas besoin d'interpréter leur structure ou signification.

- Le service transport utilise un système d'adressage qui est en correspondance avec celui du service réseau qui le prend en charge.

5.2.3.5. Couche réseau.

La couche réseau fournit d'une part les moyens d'établir, de maintenir et de libérer des connexions réseau entre des systèmes contenant des entités d'application devant communiquer, et d'autre part les moyens fonctionnels et procéduraux pour échanger entre entités de transport des unités de données du service réseau sur des connexions de réseau. Le service réseau fournit le transfert transparent de données entre les utilisateurs du service, et cache à ceux-ci la façon par laquelle les ressources sont utilisées pour réaliser ce transfert.

- Indépendance vis à vis des moyens de transmission.

- Transfert de bout en bout. Toutes les fonctions de routage et de relais sont prises en charge par l'entité réseau, comprenant le cas où plusieurs moyens de transmission différents sont utilisés en parallèle ou en tandem.

5.2.3.6. Couche liaison de données.

La couche liaison de données fournit les moyens fonctionnels et procéduraux nécessaires à l'établissement, à la maintenance et à la libération des connexions de liaison de données entre entités de réseau, ainsi qu'au transfert des unités de données du service de liaison de

données. Une connexion de liaison de données est réalisée à l'aide d'une ou plusieurs connexions physiques. La couche liaison de données détecte et corrige, dans la mesure du possible, les erreurs pouvant se produire dans la couche physique.

5.2.3.7 Couche physique.

La couche physique fournit les moyens mécaniques, électriques, fonctionnels et procéduraux nécessaires à l'activation, au maintien et à la désactivation des connexions physiques destinées à la transmission de bits entre entités de liaison de données.

5.3. CONVENTIONS DE SERVICE.

Tout le modèle d'architecture est basé sur un certain nombre de conventions appelées CONVENTION DE SERVICE. Ces conventions seront utilisées tout au long de ce travail, pour définir les interfaces entre les différentes couches, permettant ainsi une compréhension plus aisée par rapport au modèle général O.S.I.

5.3.1. Définitions.

5.3.1.1. Utilisateur d'un service.

Il s'agit de la représentation abstraite de la totalité des entités d'un système déterminé qui utilisent un service (voir fig 5-3).

5.3.1.2. Fournisseur d'un service.

C'est une machine abstraite constituant un modèle du comportement de la totalité des entités fournissant le service telles qu'elles sont vues par l'utilisateur (voir fig 5-3).

5.3.1.3. Service d'une couche.

C'est le service fourni par une couche du modèle de référence (voir 5.1.4.).

5.3.1.4. Primitive.

C'est un élément abstrait d'une interaction entre un utilisateur d'un service et le fournisseur de ce service.

5.3.1.5. Requête.

C'est une primitive émise par un utilisateur de service pour demander un certain service (voir 5.3.3.5.).

5.3.1.6. Indication.

C'est une primitive émise par un fournisseur de service :

- pour indiquer un certain service(pour indiquer entre autre un refus de connexion ou une détection d'erreur), ou

- pour indiquer qu'un service a été demandé par un utilisateur du service (voir 5.3.3.5.).

5.3.1.7. Réponse.

C'est une primitive émise par un utilisateur d'un service pour achever au niveau d'un point d'accès particulier à ce service une certaine procédure préalablement lancée par une indication reçue à ce point d'accès au service (voir 5.3.3.5.).

5.3.1.8. Confirmation.

C'est une primitive émise par le fournisseur d'un service pour achever, au niveau d'un point d'accès particulier à ce service, une certaine procédure préalablement lancée par une requête reçue à ce point d'accès des services (voir 5.3.3.5.).

5.3.2. Modèle des services d'une couche.

Toutes les interactions entre deux utilisateurs du service, situées dans la (n+1)-couche, sont définies auprès de points d'accès aux services séparés. Les utilisateurs du service communiquent via le fournisseur du service se trouvant dans la (n)-couche, en utilisant tous les services des couches de niveau inférieur à la (n+1)-couche. Les informations sont échangées entre un utilisateur du service et le fournisseur du service à l'aide de primitives. Le modèle est représenté à la figure 5-3.

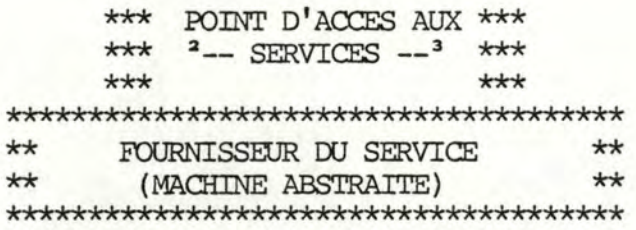


fig 5-3 Modèle d'un service.

5.3.3. Conventions d'appellation des primitives.

L'appellation de chaque primitive comprend trois éléments :

- Une appellation spécifique qui indique le type de primitive,

- Un nom générique qui indique le groupe de primitive,

- Une appellation indiquant le service.

5.3.3.1. Appellation spécifique.

L'appellation spécifique peut être :
- requête, symbolisé par 'REQ',

- indication, symbolisé par 'IND',

- réponse, symbolisé par 'RSP',

- confirmation, symbolisé par 'CNF'.

Ce symbole postfixera le nom de la primitive.

5.3.3.2. Appellation générique.

L'appellation générique sera représentée par une suite de deux ou trois lettres, formant le diminutif de l'action à effectuer.

Exemples :

CON = création de connexion

DIS = déconnexion (disconnect)

UAB = coupure par l'utilisateur (user abort)

MIN = synchronisation mineure

DT = données (data)

RA = acquiescement de resynchronisation (resynchronize ack)

Note : Les diminutifs sont ceux des termes anglais tels : DIS pour 'disconnect'.

5.3.3.3. Appellation de services.

Les appellations de services peuvent être :

- Application, symbolisé par 'A' ou 'F'
- Présentation, symbolisé par 'P'
- Session, symbolisé par 'S'
- Transport, symbolisé par 'T'
- Réseau, symbolisé par 'N'
- Liaison de données, symbolisé par 'DL'
- Physique, symbolisé par 'PH'

Ces symboles préfixeront les appellations des primitives.

5.3.3.4. Exemples.

T-CON-REQ : requête de connexion de la couche transport.

N-DT-IND : indication de données de la couche réseau.

S-RS-RSP : réponse de resynchronisation de la couche session.

P-REL-CNF : confirmation de libération (release) de la couche présentation.

5.3.3.5. Enchainement chronologique des primitives.

Posons A et B les deux points d'accès des services de deux entités communicantes. Les seuls enchainements de primitives sont :

- Service non confirmé :

requête en A ... indication en B

- Service confirmé :

requête en A ... indication en B

réponse en B ... confirmation en A

- Détection d'anomalie par le fournisseur de services :

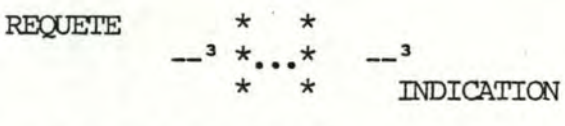
indication en A ... indication en B

- Refus d'exécution d'un service par le fournisseur de services :

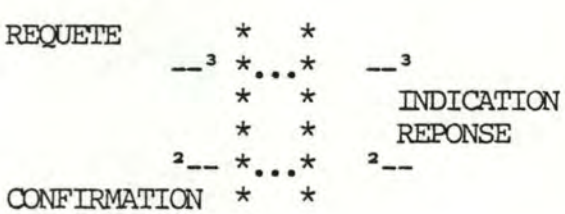
requête en A ... indication en A

La figure 5-4 donne la représentation graphique des enchainements.

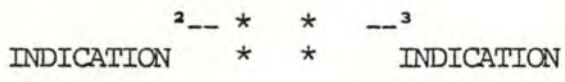
Lors d'un service confirmé ou non confirmé, les paramètres de l'indication sont ceux de la requête, les paramètres de la confirmation sont ceux de la réponses. Ces paramètres sont toujours identiques sauf dans certains cas de négociation, où cela sera précisé.



service non confirmé



service confirmé



détection d'anomalie par le fournisseur de services

refus d'exécution d'un service

fig 5-4 Diagramme d'enchainement de primitives.

PARTIE II

R E A L I S A T I O N

6. DESCRIPTION GENERALE DE LA REALISATION.

6.1. INTRODUCTION.

Dans ce chapitre, nous verrons les choix effectués selon les moyens disponibles pour mettre en oeuvre une partie des protocoles de l'O.S.I.

Nous verrons successivement une description du système utilisé, avec les contraintes portant sur ce système et restreignant la réalisation, puis les règles générales nous guidant dans la réalisation, et finalement les options choisies au niveau de chaque couche.

Ce chapitre est une justification des décisions prises. Au niveau de chaque couche, les fonctions réellement réalisées seront décrites dans les chapitres consacrés à ces dernières.

Différents facteurs nous ont obligés à ne réaliser qu'un sous-ensemble de ces normes, notamment la parution de celles-ci petit à petit. Si les premiers textes datent de décembre 1983, les derniers datent d'avril 1984. Les modifications n'étaient pas toujours très importantes, mais souvent l'interprétation de certains points douteux devait être modifiée.

6.2. DESCRIPTION DU SYSTEME UTILISE.

6.2.1. Description du matériel.

Le système utilisé est un micro-ordinateur de marque APPLE, modèle APPLE-II (APPLE1) (APPLE2) (CHAMORET1) (CHAMORET2), piloté par un processeur 8 bits, le 6502, et doté d'une mémoire vive de 64 Koctets.

La mémoire de masse se compose de deux disquettes souples de 140 Koctets chacune, auxquelles peut être adjoint une disquette virtuelle, composée de 192 Koctets de mémoire vive.

Au niveau du logiciel, l'application a été développée sur un système PASCAL U.C.S.D., dont la très large diffusion assurait

une certaine portabilité. Basé sur un interpreteur de P-code, ce logiciel comporte outre le compilateur, tout le système d'exploitation de la machine.

6.2.2. Contraintes du système.

De par le caractère mono-tâche de l'APPLE-II, il était nécessaire de faire fonctionner les différentes procédures de facon synchrone.

De plus, et c'est l'élément déterminant du choix de la réalisation, la gestion des périphériques se fait par logiciel (boucle de temps d'attente). La conséquence en est que le système d'exploitation n'accepte aucune interruption, à moins de modifier le B.I.O.S., c'est à dire le gérant des entrées-sorties (de l'anglais Basic Input Output System), opération très aléatoire et qui n'entrait pas dans le cadre de ce travail.

De par le caractère asynchrone de la transmission de données, il nous a fallu renoncer à échanger physiquement des messages d'un micro-ordinateur à l'autre. C'est pour cette raison que nous nous sommes orientés vers un fonctionnement des couches supérieures de l'architecture O.S.I.

6.2.3. Caractéristiques du Compilateur.

Le compilateur PASCAL-U.C.S.D. possède un certain nombres d'extensions par rapport à la norme PASCAL, dont certaines seront utilisées par cette réalisation.

La plus importante est la possibilité de réaliser des compilations séparées. Ces portions de programmes à compilation séparée, inexécutables seules, sont appelées "Unités" (en anglais UNIT), et sont composées de deux parties :

- Une partie "INTERFACE", dans laquelle sont définies les constantes et les variables (plus leur type), ainsi que les déclarations de procédures et de fonctions, telles qu'elles seront visibles dans les programmes utilisant l'unité.

- Une partie "IMPLEMENTATION", dans laquelle nous trouverons les constantes, les variables et les procédures locales à l'unité, ainsi que le corps des procédures déclarées dans la partie "interface".

Cette possibilité offre deux grands avantages utilisés dans la réalisation :

- Elle permet d'alléger le travail du compilateur, chose importante vu la complexité du travail et la faiblesse des ressources du compilateur (qui n'accepte, par exemple, que 255 procédures dans un programme ou une unité).

- Elle permet de plus de cacher le fonctionnement interne des procédures et fonctions déclarées dans l'interface. Grâce à celà, si les interfaces sont complètement spécifiées, et c'est le cas dans les normes O.S.I., toute extension du code des procédures n'influence pas les programmes utilisant cette unité. Ceci s'avère particulièrement intéressant dans le cadre de ce travail, ou seulement une partie du réseau est réalisée : il suffira de changer les unités de bas niveau pour bénéficier d'un réseau complet.

Une autre caractéristique du PASCAL U.C.S.D. est la possibilité d'accès direct sur disque, utilisée dans ce travail pour la gestion des mémoires-tampon.

6.3. DECISIONS DE REALISATION.

6.3.1. Règles générales.

La décomposition du travail en différentes unités correspond à la découpe en couches de l'architecture O.S.I. L'application de ce principe présente deux avantages sérieux :

- L'application des principes de décomposition en couches (voir 5.2.1.), et particulièrement les principes P2, P3, P4 et P7, semble toute indiquée pour justifier la découpe.

- Afin de pouvoir intégrer le plus facilement possible cette réalisation dans un réseau complet, il était important d'être le plus proche possible du modèle des normes O.S.I.

Dans le même souci d'extensibilité tous les accès aux différentes fonctions de chaque couche se feront sous forme d'une mise en oeuvre des conventions de services (voir 5.3.) : une couche peut activer les services de la couche de niveau inférieur par des requêtes et des réponses, et être activée par la couche inférieure via des indications et des confirmations. La facon dont ceci est effectivement mis en oeuvre varie selon chaque couche.

Dans toute l'architecture, une adresse de point d'accès aux services sera constituée de l'identificateur d'extrémité de connexion de l'entité, permettant l'identification des variables locales et de l'état de l'entité. Cet identificateur ne sera pas présent en cas de requête et d'indication de création de connexion, car à ce moment aucune valeur ne lui est encore attribuée.

Le contenu de cet identificateur de connexion sera transparent à l'utilisateur, qui l'utilisera exclusivement pour référencer la connexion jusqu'à sa cloture.

Pour faire "fonctionner" notre prototype, l'utilisateur, qui

n'a accès au réseau que par la couche Application, commencera par donner le point d'accès des services de la couche application, puis demandera l'exécution de la primitive de son choix, avec tous ses paramètres.

L'ordre de passage dans les différentes couches est le suivant :

- Si c'est une requête ou une réponse :

a.- Application appelant--³ Présentation Appelant.

b.- Présentation appelant --³ Session appelant.

c.- Session appelant --³ Transport appelant.

d.- Transport appelant --³ Réseau appelant.

e.- Réseau appelant --³ Réseau appelé.

f.- Réseau appelé --³ Transport appelé.

g.- Réseau appelé --³ Réseau appelant.

h.- Réseau appelant --³ Transport appelant.

i.- Eventuellement, itération de d. à h.

En cas d'erreur de protocole ou d'erreur dans les paramètres, ce processus est arrêté et une erreur est signalée à la couche application.

- En cas d'indication ou de confirmation :

- Application appelé --³ Transport appelé (pour voir si des unités de données sont présentes)

a.- Transport appelé --³ Session appelé.

b.- Session appelé --³ Présentation appelé.

c.- Présentation appelé --³ Application.

Ce processus est arreté si aucune unité de données de services n'est à transmettre à la couche supérieure.

6.3.2. Description de la réalisation de chaque couche.

6.3.2.1. Couche RESEAU.

Dans notre réalisation, la couche réseau est une couche

factice permettant au système de simuler un réseau. Elle recouvrera en effet toutes les couches inférieures à la couche transport. Elle se contente de tranformer une requête en indication, une réponse en confirmation.

Pour réaliser cela, elle convertit l'identificateur d'extrémité de connexion du système appelant en celui du système appelé en utilisant des tables, et signale la primitive conjointe (indication ou confirmation) à celle qui a occasionné l'activation de la couche (requête ou réponse).

6.3.2.2. Couche TRANSPORT.

La couche transport est le lieu ou se fera l'échange des données entre d'une part le réseau, et d'une autre part les entités d'application. C'est au niveau de cette couche que se résolveront les problèmes de synchronisation.

Il y a deux types d'évènements susceptibles de modifier l'état de la couche transport :

- Les requêtes et les réponses provenant de la couche SESSION, et

- Les indications et les confirmations provenant de la couche RESEAU.

Dans une mise en oeuvre réelle, ces deux types d'évènements sont asynchrones. Dans le prototype réalisé ici, nous avons tenté d'être le plus proche possible de la réalité en faisant fonctionner la couche transport de la facon suivante : toute requête à la couche réseau faite par une des entités transport est transformée en indication (les réponses en confirmations), et l'entité transport communicante modifie immédiatement son état, exécutant éventuellement les traitements appropriés, sans que l'entité session liée à ce point d'accès des services n'en soit avertie.

Par contre, les requêtes et les réponses seront exécutées immédiatement, mettant ainsi à jour l'état de l'entité.

Sachant que chaque entité transport possède des ressources en quantité limitée, la couche transport àssurera le contrôle de flux des messages par un mécanisme de crédit.

Afin de limiter le nombre de connexions de réseau, um mécanisme de multiplexage sera mis en oeuvre, permettant à plusieurs entités transport communiquant sur des connexions réseau semblables d'utiliser un même point

d'accès de services réseau.

6.3.2.3. Couche SESSION.

La couche session a un fonctionnement synchrone par rapport à l'application. Son but essentiel est de contribuer à la synchronisation des applications. Pour cela, il y a des primitives de synchronisation et de resynchronisation, que l'entité session vérifie et gère selon l'état dans laquelle elle se trouve.

On peut classer les primitives de la couche session en trois groupes :

- Les requêtes et les réponses, qui occasionnent une mise à jour immédiate de l'état de l'entité session, et éventuellement l'émission d'une unité de données vers l'entité homologue.

- Une fonction de détection d'évènement, qui testera si un évènement s'est produit au niveau de la couche transport, et si c'est le cas, mettra à jour l'état de l'entité session.

- Les indications et confirmations, qui ne seront que le désassemblage des messages préalablement reçus et détectés par la fonction ci-dessus.

Tous les arguments émis ou reçus par la couche session auront la forme de chaines d'octets, et non des types PASCAL, ce qui permettra une plus grande flexibilité de mise en oeuvre en cas d'utilisation éventuelle du modèle par des programmes d'applications écrits en d'autres langages que le PASCAL, à condition que ces systèmes puissent compiler en code natif les unités, et faire l'édition des liens entre programmes en langages différents.

6.3.2.4. Couche PRESENTATION.

Le rôle de la couche présentation est de déterminer une syntaxe de transfert commune, et de transformer cette syntaxe en une syntaxe compréhensible par chacune des entités d'application correspondantes.

Malheureusement l'état d'avancement de la normalisation de la couche présentation ne permet pas encore de tenter la réalisation de cette couche.

Son rôle dans cette réalisation sera simplement la transformation des arguments de type PASCAL provenant de

la couche application en chaines d'octets, et renvoi de la primitive vers la couche session.

6.3.2.5. Couche APPLICATION.

Parmi les trois types d'application recensés et définis par I.S.O. (J.T.M.,F.T.A.M.,V.T.M voir 4.3.2.), seul sera réalisé le F.T.A.M., c'est à dire le transfert des fichiers et la gestion de leur accès.

Il n'entrait pas dans le cadre de ce travail d'écrire un gérant de base de données, mais bien de fournir toutes les primitives nécessaires à son fonctionnement.

Dans le chapitre consacré à la couche application se trouvera une brève description des concepts du F.T.A.M., le lecteur désirant avoir la description complèète étant invité à s'en référer aux textes des normes.

Les fonctions réalisées ici seront la vérification des appels des primitives, et l'acheminement des données.

7. DESCRIPTION DE LA COUCHE RESEAU.

7.1. SPECIFICATIONS DES SERVICES.

7.1.1. Demande de connexion.

```
PROCEDURE N-CON-REQ
   (VAR N-IDT:TYPIDT;
    CALLING-NSAP,CALLED-NSAP:TYPNSAP);
```

La procédure de requête de connexion (N-CON-REQ) va engendrer une indication de connexion (N-CON-IND) au point d'accès de service "CALLED-NSAP". De plus, l'identificateur d'extrémité de connexion (N-IDT) va être défini par la couche réseau, et la couche transport devra utiliser cette variable pour référencer cette nouvelle connexion. A part l'identificateur d'extrémité de connexion, les deux autres arguments sont transmis tels quels à l'entité appelée.

```
PROCEDURE N-CON-IND
   (VAR N-IDT:TYPIDT;
    CALLING-NSAP,CALLED-NSAP:TYPNSAP);
```

Cette procédure sert à indiquer à l'entité transport à laquelle elle est liée qu 'une requête de connexion a été émise par une autre entité.

```
PROCEDURE N-CON-RSP
   (N-IDT:TYPIDT;
    RESP-NSAP:TYPNSAP);
```

La procédure de réponse de connexion (N-CON-RSP) est émise par l'entité appelée si elle accepte la connexion. L'identificateur d'extrémité, déclaré dans l'indication de connexion, est utilisé pour référencer la connexion. Le point d'accès de service en réponse (RESP-NSAP) n'est utilisé ici que par souci de respect des normes, prévoyant éventuellement un renvoi sur un point d'accès de service différent de celui appelé.

```
PROCEDURE N-CON-CNF
   (N-IDT:TYPIDT;
    RESP-NSAP:TYPNSAP);
```

Cette primitive engendre la confirmation de connexion (N-CON-CNF) au point d'accès de service de l'entité appelante.

7.1.2. Transfert de données.

```
PROCEDURE N-DT-REQ
   (N-IDT:TYPIDT;
```

```
    DATA :TYPNSDU;
    LONG :INTEGER);
```

Le transfert de données sur une connexion réseau s'effectue au moyen de la demande de données. L'identificateur d'extrémité de connexion (N-IDT), sert à identifier la connexion sur laquelle les données sont à tranférer.

Le type d'unité de données du service de réseau (DATA) est une table d'octets de longueur maximum de 512 octets. Afin de réduire la longueur des échanges, la longueur des données utiles est renseignée (LONG), afin de ne transmettre que le nombre nécessaire d'octets.

La requête d'émission de données ne peut s'effectuer que sur une connexion établie. En d'autres cas, elle n'est simplement pas exécutée. Il n'y a pas de restriction concernant la direction d'émission (la connexion est supposée simultannée).

```
PROCEDURE N-DT-IND
   (N-IDT:TYPIDT;
    VAR DATA :TYPNSDU;
    VAR LONG :INTEGER);
```

La procédure d'indication de données (N-DT-IND) sert à indiquer à une entité transport que des unités de données de service réseau sont présentes.

7.1.3. Libération de connèxion.

```
PROCEDURE N-DIS-REQ
   (N-IDT:TYPIDT;
    CAUSE:INTEGER);
```

La requête de libération de connexion (N-DISREQ) peut être à tout moment émise par une des deux entités transport pour signifier soit un refus de connexion, soit un abandon après erreur, soit simplement la fin normale de connexion. La cause (CAUSE) est un paramètre entier et comporte la cause de déconnexion provenant de la couche transport.

```
PROCEDURE N-DIS-IND
   (N-IDT:TYPIDT;
   VAR CAUSE:INTEGER);
```

L'indication de libération de connexion réseau (N-DIS-IND) peut soit indiquer qu'une requête de libération de connexion a été émise par l'entité communicante, soit que le fournisseur des services de réseau décide de clôturer la connexion suite, par exemple, à une erreur de protocole, ou à une carence de

ressources.

7.2. FONCTIONNEMENT INTERNE.

La fonction de transformation de point d'accès des services est basée sur une table de correspondance. Quatre fonctions gèrent cette table : ajout d'un élément, retrait, sélection, et mise à jour. Dans cette table, nous retrouvons les identificateurs d'extrémité de connexion appelée et appelante, et une variable booléenne indiquant si la connexion est établie.

Dans notre modèle, l'utilisation de cette variable booléenne est tout à fait inutile, car la requête de connexion est immédiatement suivie de la confirmation (ou de l'indication de déconnexion), mais dans un réseau réèl, une connexion pourrait être demandée et non encore confirmée. Si à ce moment une autre entité veut utiliser la connexion réseau, elle ne peut pas encore transférer de données, mais elle ne doit quand même pas exécuter une requ^ete de connexion.

Voyons maintenant les différentes actions selon les invocations des primitives.

7.2.1. N-CON-REQ :

Cette procédure crée dans la table un point d'entrée, recevant en retour l'identificateur d'extrémité de connexion, et positionne la variable booléenne de connexion établie à "faux". Puis elle exécute un N-CON-IND.

7.2.2. N-CON-IND :

Après avoir crée un point d'entrée (qui est le symétrique du point d'entrée de la requête), où la variable booléenne est mise à "vrai" (car à ce moment la connexion existe), il y a activation dans la couche transport de la procédure d'indication de connexion.

7.2.3. N-CON-RSP :

Cette procédure exécute simplement une confirmation de connexion à l'autre extrémité de connexion.

7.2.4 N-CON-CNF :

A la réception de la confirmation, l'entité va positionner l'indicateur d'existance de la connexion à "vrai", et activer la procédure de confirmation de connexion dans la couche transport.

7.2.5. N-DT-REQ :

Si la connexion existe, la procédure va exécuter une indication de données à l'autre extrémité de connexion.

7.2.6. N-DT-IND :

Cette procédure va activer la procédure d'indication de données dans la couche transport.

7.2.7. N-DIS-REQ :

Après avoir supprimé le point d'entrée dans la table des connexions, cette procédure exécute une indication de déconnexion.

7.2.8. N-DIS-IND :

Cette procédure retire le point d'entrée dans la table des connexions, puis active la procédure de déconnexion de la couche transport.

Dans les cas où une sélection ou une création sont impossibles (engorgement, identificateur inexistant dans le cas d'une sélection, existant dans le cas d'une création), la procédure renvoie une indication de déconnexion, avec un code d'erreur indiquant que c'est le fournisseur de services qui a réclamé la déconnexion, et si la connexion était établie, elle exécute également une requête de déconnexion.

7.3. EXTENSIONS FUTURES.

En cas d'intégration dans un réseau, c'est cette couche qui subira les plus grandes modifications. Il faudra en effet mettre en oeuvre tous les protocoles de la couche réseau, en respectant les interfaces décrits ci-dessus (ISO19).

Si l'on s'oriente vers un réseau local utilisant les spécifications de l'interface ETHERNET, les procédures seront assez simples : il faudra créer une unité de données de service réseau de requête de connexion, une unité de données de service réseau de réponse de connexion, une de requête de déconnexion et une unité de données de service réseau de transfert de données.

Toutes les procédures de la couche pourront être gardées au niveau de la gestion de la table de correspondance des identificateurs de connexion, mais la transformation de requête en indication devra être transformée en une expédition réelle de données.

A la réception d'une unité de données de service réseau, il faudra décoder le type d'unité de données de service réseau (requête ou réponse de connexion, requête de données, ou requête de déconnexion), et activer immédiatement (interruption, multi-processeurs) la procédure appropriée dans la couche.

Une autre option est la réalisation d'une couche réseau de type "connectionless-mode", c'est à dire qu'il n'est pas nécessaire de créer une connexion de réseau pour l'émission d'unités de données de protocole de réseau. Cette option est possible lors de l'utilisation de la classe 4 de protocole de transport (voir 8.2.1. et 8.3.).

8. DESCRIPTION DE LA COUCHE TRANSPORT.

8.1. SPECIFICATION DES SERVICES.

Rappelons que l'ensemble des en-têtes de procédures et de fonctions décrits ci-dessous sont les seules parties visibles par l'utilisateur des services de transport. Les règles de dénomination des noms de ces procédures et fonctions sont celles reprises dans les conventions de services (voir 5.3.).

8.1.1. Etablissement de connexion.

```
PROCEDURE T-CON-REQ
   (VAR T-IDT:TYPIDT;
    CALLING-TSAP,CALLED-TSAP:TYPSAP;
    EXPRESS:BOOLEAN;
    VAR ERR:INTEGER);
```

La procédure de requête de connexion (T-CON-REQ) va affecter une connexion transport entre le point d'accès de service appelé (CALLED-TSAP) et le point d'accès de service appelant (CALLING-TSAP). La variable booléenne EXPRESS signifie au fournisseur de service transport qu'un transfert de données express est ou n'est pas demandé. En retour, cette procédure définira un identificateur d'extrémité de connexion unique au niveau du point d'accès de service. Cet identificateur est indispensable car il peut y avoir plusieurs connexions transport entre les deux entités session.

Cette procédure créera si nécessaire une connexion réseau, puis fera la requête de création de connexion transport, signalée à l'autre entité. Si elle supporte la gestion des données express, elle laissera la booléenne EXPRESS à ce qu'elle était, sinon elle la met à "faux".

La variable ERR prend une valeur différente de zéro si une erreur est détectée : engorgement, erreur de protocole...

```
FUNCTION T-CON-IND
   (VAR T-IDT:TYPIDT;
    VAR CALLING-TSAP,CALLED-TSAP:TYPSAP;
    VAR EXPRESS:BOOLEAN;
    VAR ERR:INTEGER;):BOOLEAN;
```

La fonction d'indication de connexion (T-CON-IND) est activée par l'entité session appelée, et quand elle retourne une valeur "vrai", cette fonction indique qu'une requête de connexion est parvenue au point d'accès de service de l'entité appelée.

L'identificateur d'extrémité de connexion est défini, afin d'identifier la connexion transport de facon unique par l'entité

session appelée. Si le fournisseur de service possède les fonctions nécessaires à la gestion des données express, la booléenne EXPRESS prendra la valeur reçue par la connexion, sinon elle prendra la valeur "faux".

La variable ERR est toujours égale à zéro, mais est définie dans un souci d'homogénéité.

```
PROCEDURE T-CON-RSP
   (T-IDT:TYPIDT;
    RESP-TSAP:TYPSAP;
    EXPRESS:BOOLEAN;
    VAR ERR:INTEGER);
```

La procédure d'acceptation de connexion (T-CON-RSP) est émise par l'entité session appelée, pour notifier l'accord de créer une connexion. L'adresse de point d'accès de service en réponse (RESP-TSAP) est identique à l'adresse appelée, et est gardée dans un souci de respect des normes, qui étudie la possibilité de faire des reroutages au niveau de la couche transport.

La variable EXPRESS peut prendre la valeur "vrai" si elle était déjà à "vrai" lors de l'indication, c'est à dire que le service express était demandé par l'entité appelante et supporté par les fournisseurs de services de l'entité appelante et de l'entité appelée (ce qui est le cas dans notre réalisation).

La variable ERR est différente de zéro si l'identificateur d'extrémité de connexion est inconnu, si une réponse de connexion n'est pas conforme à l'état du fournisseur de service, ou si la variable EXPRESS ne prend pas une valeur autorisée.

```
FUNCTION T-CON-CNF
   (T-IDT:TYPIDT;
    VAR RESP-TSAP:TYPSAP;
    VAR EXPRESS:BOOLEAN;
    VAR ERR:INTEGER):BOOLEAN;
```

Cette fonction, exécutée par l'entité session appelante, prend la valeur "vrai" si la réponse de connexion est parvenue au fournisseur de services de la couche transport.

L'adresse en réponse de point d'accès de service est la même que celle de la primitive de réponse.

Si la variable EXPRESS a une valeur "vrai", alors le transfert de données express est autorisé sur la connexion.

La variable ERR est différente de zéro si l'identificateur d'extrémité de connexion est inconnu auprès du fournisseur de services.

8.1.2. Transfert de données normales.

```
PROCEDURE T-DT-REQ
   (T-IDT:TYPIDT;
    VAR DATA:TYPTSDU;
    LONG :INTEGER;
    VAR ERR:INTEGER);
```

Cette procédure est exécutée par une entité session désireuse d'émettre des données sujettes au contrôle de flux des données normales (voir 8.2.4.). Précisons que ce contrôle de flux est transparent à l'utilisateur des services.

L'unité de données de service transport (DATA) est une suite de longueur quelconque (inférieure à 64 Koctets) d'octets, le nombre d'octets utiles étant donné par la variable LONG. Afin de permettre une longueur non définie préalablement, le TYPTSDU est défini comme un vecteur de longueur 1, et aucun contrôle à l'exécution n'est fait sur l'indice. L'entité session émettrice de données définira un type PASCAL comme suit :

```
TYPE
 BYTE=0..255;
 TYPTSDU=PACKED ARRAY(1..1) OF BYTE;
 TYPLOCAL=(* DEFINITION LOCALE *);
 TYPDATA=RECORD
         CASE BOOLEAN
          OF TRUE : (TSDU:TYPTSDU);
             FALSE: (DATA:TYPLOCAL);
         END;
```

De la sorte, les tailles d'unités de données de service transport sont gérées dynamiquement, selon les besoins de l'utilisateur des services de transport.

La variable ERR prend une valeur différente de zéro si l'état de la connexion est tel qu'il ne permet pas la transmission de données, si l'identificateur d'extrémité de connexion est inconnu du fournisseur de services transport, ou si l'état du fournissseur de services transport est tel qu'il n'a plus les ressources nécessaires au transfert de toute l'unité de données de service (voir contrôle de flux et mécanisme de crédit en 8.2.2.).

```
FUNCTION T-DT-IND
   (T-IDT:TYPDIT;
    VAR DATA:TYPTSDU;
    VAR LONG:INTEGER;
    VAR ERR:INTEGER):BOOLEAN;
```

Cette fonction prend la valeur "vrai" si une unité de données de service transport complète est disponible. Les remarques concernant le type de données et la longueur sont identiques à celles formulées plus haut.

La variable ERR prend une valeur différente de zéro si l'identificateur d'extrémité de connexion est inconnu, ou si l'état de l'entité transport est tel qu'il ne peut recevoir de données (par exemple en attente d'une confirmation de connexion).

8.1.3. Transfert de données express.

Le transfert d'unités de données express du service transport est indépendant du contrôle de flux des unités de données normales du service transport. Il est surtout utilisé dans le cas ou le flux des données normales est congestionné.

```
PROCEDURE T-EX-REQ
   (T-IDT:TYPIDT;
    EXTSDU:TYPEXTSDU;
    LONG:INTEGER;
    VAR ERR:INTEGER);
```

Cette procédure effectue l'envoi d'une unité de données express (EXTSDU) sur la connexion transport. La longueur de cette unité de données express (LONG) est comprise entre 1 et 16 octets.

La variable ERR est différente de zéro si l'identificateur d'extrémité de connexion est inconnu, si l'état de l'entité transport est tel qu'il ne puisse recevoir cette requête, ou que l'utilisation de données express ne soit pas utilisable sur cette connexion. De plus, la valeur est non nulle si le contrôle de flux des unités de données express est congestionné (situation non persistante).

```
FUNCTION T-EX-IND
   (T-IDT:TYPIDT;
    VAR EXTSDU:TYPEXTSDU;
    VAR LONG:INTEGER;
    VAR ERR:INTEGER):BOOLEAN;
```

Cette fonction prend la valeur "vrai" si une unité de données express a été reçue par l'entité. Le contenu et la taille de l'unité de données du service sont spécifiés par les arguments EXTSDU et LONG.

La variable ERR prend une valeur différente de zéro lorsque l'identificateur d'extrémité de connexion est invalide, ou si les fonctions de gestion des données express ne sont pas fournies sur cette connexion.

8.1.4. Libération de connexion.

```
PROCEDURE T-DIS-REQ
   (T-IDT:TYPIDT;
    VAR ERR:INTEGER);
```

Cette procédure peut être activée à tout moment pour réclamer la libération de la connexion transport.

La variable ERR peut prendre une valeur différente de zéro si l'identificateur d'extrémité de connexion est invalide.

```
FUNCTION T-DIS-IND
   (T-IDT:TYPIDT;
    VAR CAUSE:INTEGER;
    VAR ERR:INTEGER):BOOLEAN;
```

Cette fonction prend la valeur "vrai" si la connexion de transport a été refusée ou liberée. Le paramètre CAUSE prend la valeur zéro si la déconnexion est due à une requête de libération de la part de l'entité transport éloignée, et prend une valeur non nulle si la déconnexion a été causée par un des deux fournisseurs de service, suite par exemple, à une erreur de protocole, à une défaillance des ressources...

La variable ERR prend une valeur différente de zéro si l'identificateur d'extrémité de connexion est invalide.

8.2. DESCRIPTION DES PROTOCOLES.

Les évènements modifiants l'état de l'entité de transport peuvent avoir deux origines : soit c'est un service demandé par une entité session liée au point d'accès de service, requête ou réponse, soit une indication ou une confirmation de l'entité réseau à laquelle est liée l'entité transport.

Quelle que soit l'origine de la primitive (session ou réseau), la couche transport doit réagir immédiatement afin d'optimiser le dialogue entre entités transport. Nous verrons en effet ci-dessous que le contrôle de flux tire sa raison d'être de l'asynchronisme existant entre les utilisateurs des services de la couche transport et les évènements provenant du réseau.

Nous verrons d'abord quelques concepts principaux, tels que les classes de protocoles réalisées, le multiplexage, le contrôle de flux, la gestion des données express, puis nous verrons comment est organisé le fournisseur de service transport.

8.2.1. Classes de protocoles.

Dans le modèle de référence, cinq classes de protocoles ont été définies pour le fournisseur de services de transport. Ces cinq classes sont fonction de la qualité de la ligne et de la

technologie des entités correspondantes.

* CLASSE 0 : C'est la classe de base. Elle suppose une connexion réseau ayant un taux acceptable d'erreurs non signalées, et un taux acceptable d'erreurs signalées. C'est typiquement la classe réservée aux terminaux.

* CLASSE 1 : C'est la classe de base avec en plus un mécanisme de reprise sur erreur. Elle suppose une connexion réseau ayant un taux acceptable d'erreurs non signalées, mais présentant un taux inacceptable d'incidents signalés. Cette classe est une version améliorée de la classe 0, donc également reservée a des terminaux (un peu plus intelligents).

* CLASSE 2 : Prévue pour une connexion ayant un taux acceptable d'erreurs (signalées et non signalées), cette classe prend en charge le multiplexage de plusieurs connexions transport sur une seule connexion réseau, ainsi que la gestion du contrôle de flux (sujet à négociation voir 8.2.2.). C'est ce genre de classe que l'on retrouve sur des systèmes autonomes.

* CLASSE 3 : C'est la classe avec reprise sur erreur et multiplexage. Destinée à fonctionner sur un système semblable à celui de la classe 2, elle est prévue pour fonctionner avec des connexions réseau présentant un taux acceptable d'erreurs non détectées, mais un taux inacceptable d'erreurs signalées.

* CLASSE 4 : C'est la classe avec détection d'erreurs. Elle présente de plus toutes les caractéristiques de la classe 3. Elle est destinée à l'utilisation de connexions réseau présentant un taux inacceptable d'erreurs (signalées et non signalées).

Dans le cadre de la réalisation d'un réseau local, et sur base de ce qui a été vu concernant le réseau ETHERNET (voir 3.2.), la classe 2 était typiquement la classe à mettre en oeuvre. En effet, le taux d'incidents sur la ligne est très faible (car la couche liaison de données d'ETHERNET ne transmet que des trames correctement reçues), et statistiquement le taux d'erreurs sur un câble coaxial est assez bas.

Pour des raisons de conformité à la norme, toute mise en oeuvre de la classe 2 entraine la mise en oeuvre obligatoire de la classe 0. Cette mise en oeuvre a été réalisée, bien qu'inutilisée dans l'architecture de ce prototype.

On pourrait certes objecter que la mise en oeuvre de la classe 3 aurait pu être plus appropriée. En effet, en cas d'erreur signalée, la classe 2 ne prévoit d'autre traitement que de libérer la connexion. Cependant, en regardant les services fournis par ETHERNET, on constate qu'en cas de détection

d'erreur sur une trame (C.R.C. incorrect ou nombre non entier d'octets), on ne peut faire aucune hypothèse sur la référence de destination, celle-ci pouvant être erronée aussi bien que le reste. Il est donc impossible de signaler une erreur à une entité précise, d'ou la classe 3, prévue pour faire des reprises sur erreurs SIGNALEES, ne nous parait pas appropriée.

La classe 4 détecte les erreurs. En fait, c'est une classe qui se justifie autant pour des L.A.N. que pour des W.A.N., seulement la réalisation d'une classe 4 pour cette couche représente à elle seule un travail considérable. L'utilisation de temporisateurs complexes (délai de création de connexion, délai d'inactivité, temporisateur de fenêtre...) fait que les interruptions sont indispensables. De plus, les ressources nécessaires sont beaucoup plus importantes, car toute unité de données est mémorisée dans le système pendant un certain temps. Dans ce travail où trois autres couches devaient être réalisées, le choix fut pris de ne pas mettre en oeuvre cette classe 4.

8.2.2. Négociations.

Comme dans toutes les couches, il est nécessaire de négocier certaines options de la couche transport. Toutes les négociations se font au moment de la création de la connexion, et les options choisies ne peuvent être modifiées durant la vie de la connexion.

Une des négociations porte sur la classe choisie. Le fournisseur de service propose en effet la classe 2, mais propose également comme classe de repli la classe 0.

Le transfert des données express est aussi sujet à négociation : l'utilisateur des services de transport peut en demander l'usage, qui peut être refusé par l'entité appelante, éventuellement par l'entité appelée, ou encore par l'utilisateur des services transport appelé. Ce n'est que si tous sont capables de supporter ou d'utiliser ce service que le transfert des données express sera réalisé.

La taille maximum des unités de données de protocoles de transport est également négociée. Chaque entité transport propose la taille maximum qu'elle peut supporter (en fonction de ses ressources), et c'est la plus petite des deux qui est choisie.

Une option encore négociée est l'absence de contrôle de flux de la classe 2, ce qui la reduit à la classe de base avec multiplexage.

8.2.3. Multiplexage et gestion d'adresses.

Le nombre de connexions de transport est égal au nombre de connexions de session (lui-même égal au nombre de connexion de présentation et d'application). Par contre, le nombre de connexions de transport est supérieur ou égal au nombre de connexions de réseau. Un multiplexage est nécessaire.

Le nombre de connexions de réseau est égal au nombre de systèmes pour lesquels il existe au moins une connexion transport.

Une adresse de point d'accès des services au transport se compose de trois parties :

- un identificateur du système.

- un point (".") servant de délimiteur.

- un identificateur local à ce système.

L'adresse entière ne peut depasser 16 caractères.

exemples : SYS2.USER5
1.TOTO

La première partie est l'adresse du point d'accès des services du réseau, la deuxième étant un identificateur local. Le tout forme l'adresse de point d'accès des services du transport.

Le fournisseur du service transport ne fait aucun contrôle de sémantique sur l'adresse de point d'accès des services. Il essaye simplement d'établir une connexion réseau entre les premières parties dès deux adresses.

Afin d'identifier une connexion transport à partir d'un point d'accès des services de réseau, chaque entité se voit attribuer un numéro entier destiné à servir de référence. Pour affecter ce numéro à une connexion, le fournisseur de service va chercher le premier entier positif non utilisé. Chaque T.P.D.U. va contenir ce numéro, permettant le démultiplexage par l'entité réceptrice.

Un exemple de la gestion des adresses de points d'accès des services est représenté à la fig 8-1.

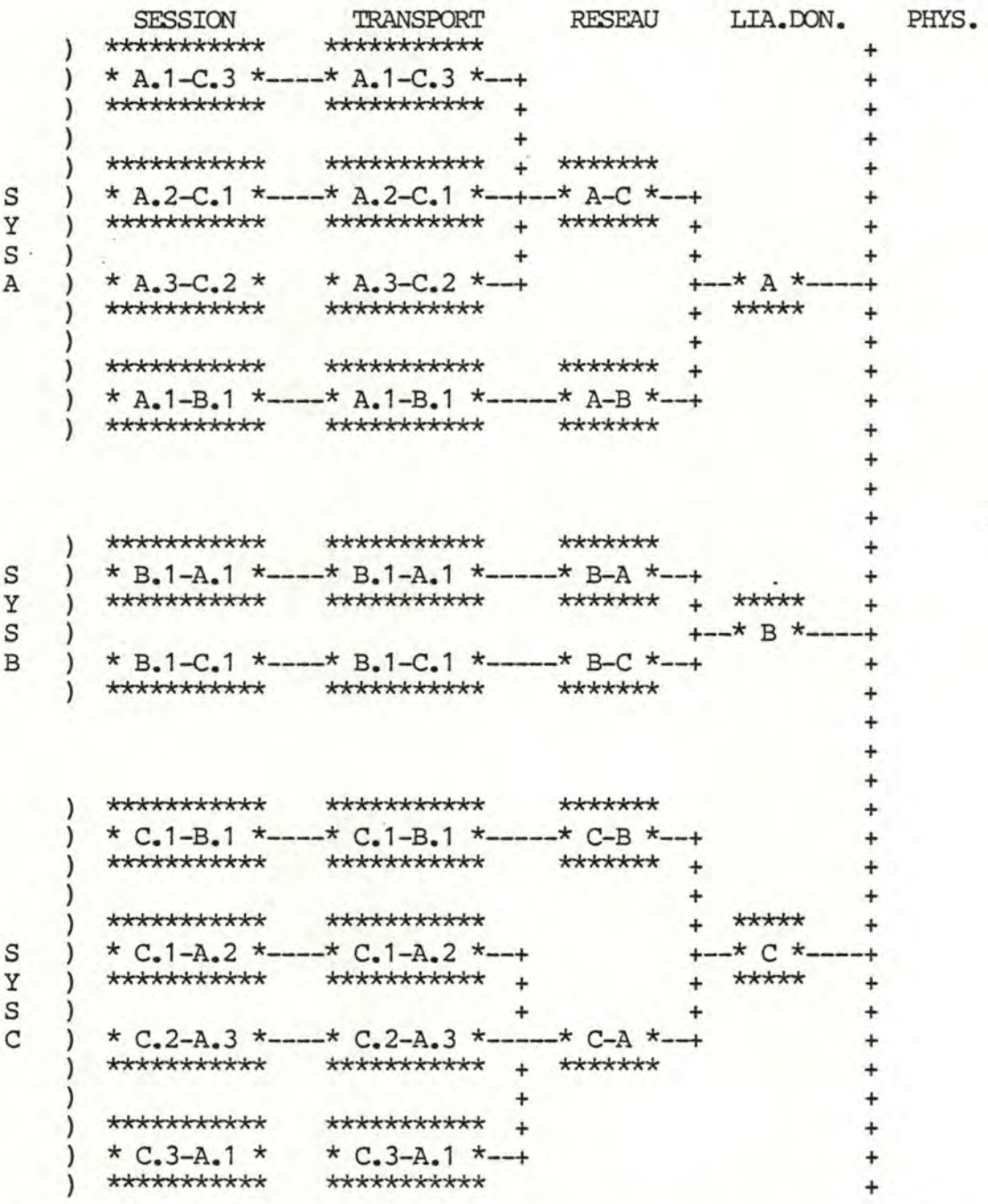


FIG 8-1 : Exemple de gestion des points d'accès des services. Chaque entité est représentée par le point d'accès des services appelant et appelé.
A.1-C.3 représente par exemple une des trois connexion du système A au système C. Pour le système A, c'est la première des trois connexions existants entre ces systèmes, pour le système C, c'est la troisième.

8.2.4. Contrôle de flux.

Les normes spécifient les règles de gestion du contrôle de flux, sans préciser la facon dont cela doit être mis en oeuvre. Nous découperons ce paragraphe en deux : les spécifications des normes d'une part, la réalisation d'une autre part.

8.2.4.1. Définitions

8.2.4.1.1. Crédit

Le crédit est le nombre d'unités de données de protocole qu'une entité est autorisée à émettre.

8.2.4.1.2. Numéro d'unité de données de protocole.

Chaque unité de données est numérotée. Le numéro d'une unité de données de protocole est égal au numéro de la dernière unité de données de protocole émise incrémenté de un. Le numéro initial est zéro.

8.2.4.1.3. Fenêtre de transmission.

La fenêtre de transmission est l'ensemble des numéros d'unités de données de protocole de transport qu'une entité a le droit d'émettre.

8.2.4.1.4. T.P.D.U. CR

C'est une unité de données de protocole de requête de création de connexion.

8.2.4.1.5. T.P.D.U. CC

C'est une unité de données de protocole de confirmation de création de connexion.

8.2.4.1.6. T.P.D.U. DT

C'est une unité de données de protocole transportant des données de l'utilisateur des services de transport.

8.2.4.1.7. T.P.D.U. AK

C'est une unité de données de protocole destinée à acquiescer des données et à modifier la fen^etre de transmission.

8.2.4.1.8. Limite inférieure de fenêtre.

C'est le plus petit numéro de T.P.D.U. DT que l'entité est autorisée à émettre.

8.2.4.1.9. Limite supérieure de fenêtre.

C'est le plus grand numéro de T.P.D.U. DT que l'entité est autorisée à émettre.

8.2.4.2. Spécifications du contrôle de flux.

Le contrôle de flux est basé sur un mécanisme de crédit. Chaque entité de transport affecte un crédit initial à l'autre entité, contenu dans la T.P.D.U. CR pour l'entité appelante, dans la T.P.D.U. CC pour l'entité appelée.

Au départ, la limite inférieure de la fenêtre d'une entité est égale à zéro, la limite supérieure au crédit reçu dans la T.P.D.U CR ou la T.P.D.U. CC.

Au fur et à mesure de la transmission, toute unité de données de protocole porte un numéro, égal à celui de la précédente unité de données plus 1, modulo 128. Une entité peut émettre des T.P.D.U. DT jusqu'à ce que le numéro de la prochaine T.P.D.U. DT soit supérieur à la limite supérieure de la fenêtre.

A tout moment, une entité peut modifier la fenêtre de l'entité correspondante en émettant une T.P.D.U. AK, contenant un numéro de T.P.D.U. et un crédit, à condition que :

- Le numéro de T.P.D.U. soit supérieur ou égal au dernier numéro de T.P.D.U. AK émis, ou supérieur à zéro si c'est la première T.P.D.U. AK émise.

- La somme des valeurs numéro de T.P.D.U. AK et crédit (modulo 128) ne soit pas inférieure à la limite supérieure de la fenêtre de l'entité correspondante (pas de réduction de crédit autorisée).

A la réception d'une T.P.D.U. AK, l'entité va mettre à jour sa fenêtre de transmission en prenant le numéro de la T.P.D.U. AK comme la limite inférieure de sa fenêtre, et la somme des valeurs numéro de T.P.D.U. AK et crédit comme sa limite supérieure de fenêtre.

Ce mécanisme est symétrique au niveau des deux entités.

8.2.4.3. Réalisation du contrôle de flux.

Au moment de la création d'une connexion transport, chaque entité se voit réserver une mémoire tampon, de taille finie.

L'entité initiatrice de la connexion va attribuer dans la T.P.D.U. CR un crédit égal à la taille du tampon divisée par la taille maximale proposée des unités de données de protocole de transport.

L'entité recevant une T.P.D.U. CR va d'abord déterminer la taille maximum des unités de données de protocole, qui est la valeur la plus petite entre la taille proposée et la taille que cette entité supporte, le résultat allant être la taille maximale des T.P.D.U. supportée par la connexion. Appliquant le même mécanisme, elle émettra dans la T.P.D.U. CC un crédit égal au quotient de la taille de son tampon et de cette taille de T.P.D.U.

A la réception d'une T.P.D.U. CC, l'entité va calculer si une augmentation de crédit est possible (cas où la taille maximum des T.P.D.U. aurait été diminuée), et si c'est le cas, émet une T.P.D.U. AK avec le nouveau crédit, obligatoirement plus grand que le crédit initial.

Lorsqu'une entité reçoit une T.P.D.U. DT, après vérification de sa validité (numéro compris dans la fenêtre de transmission et égal au dernier numéro reçu plus un), elle va calculer si le crédit peut être augmenté, ce qui est possible car l'indication de données du réseau fournit la longueur utile de la T.P.D.U., cette longueur pouvant être inférieure à la taille maximum des T.P.D.U. Si ce crédit peut être augmenté, une T.P.D.U. AK est émise avec le dernier numéro de T.P.D.U. DT reçu, et le nouveau crédit.

Lorsque l'utilisateur du service ira retirer du tampon (par la fonction T-DT-IND) une unité de données du service transport, une augmentation de crédit est également possible, ce qui comme précédemment engendrera l'émission d'une T.P.D.U. AK.

Prenons un exemple pour illustrer ce mécanisme. Supposons la taille du tampon égale à 600 octets, la taille maximum des T.P.D.U à 256 octets. A la fin de l'établissement de connexion, nous aurons :

limite inférieure = 1

mémoire tampon libre = 600

crédit réel = 600 / 256 = 2

crédit attribué = 2

limite supérieure = crédit = 2

Arrive une première T.P.D.U DT de numéro 1, d'une taille de 100 octets. La situation deviendra :

limite inférieure = 2

mémoire tampon libre = 600 - 100 = 500

crédit attribué = 1

crédit réel = 500 / 256 = 1

limite supérieure = 2

Il n'y a en effet pas de changement, une seule T.P.D.U. pouvant être reçue. Supposons qu'arrive une seconde T.P.D.U. DT portant le numéro 2 et de même taille (100 octets). Nous aurons alors :

limite inférieure = 3

mémoire tampon libre = 500 - 100 = 400

crédit attribué = 0

crédit réel = 400 / 256 = 1

limite supérieure = 2

On voit que le crédit réel est supérieur au crédit attribué. L'entité va donc émettre une T.P.D.U. AK ayant comme numéro 3 et comme crédit 1. L'entité destinatrice modifiera sa fenêtre, lui mettant les limites inférieures et supérieures à 3.

Si suite à cela, l'utilisateur des services de transport retire les deux unités de données, nous aurons :

limite inférieure = 3

mémoire tampon libre = 400 + 200 = 600

crédit attribué = 1

crédit réel = 600 / 256 = 2

limite supérieure de fenêtre = 3

De nouveau, une T.P.D.U. AK doit être émise, portant le numéro de T.P.D.U. AK 3, et signalant un nouveau crédit de 1.

8.2.5. Contrôle de flux des données express.

Le contrôle de flux des données express est beaucoup plus simple, pour deux raisons : la taille des T.P.D.U. express est très petite (max 16 octets), et on n'émet une seconde T.P.D.U. de données express que lorsque la première a été acquiescée. Voyons d'abord quelques définitions, puis nous montrerons le mécanisme gérant le contrôle de flux des données express.

8.2.5.1. T.P.D.U. ED

C'est une unité de données de protocole transport portant des données express.

8.2.5.2. T.P.D.U. EA

C'est une unité de données de protocole signifiant l'acquiescement de la dernière T.P.D.U. ED émise.

Supposons qu'une entité soit émettrice et l'autre réceptrice de données express, le mécanisme étant symétrique. L'entité réceptrice possède deux tampons de réceptions de T.P.D.U. ED., et comporte trois états :

- PREMIER-VIDE
- PREMIER-PLEIN
- SECOND-PLEIN

L'entité émettrice comporte deux états :

- PRET-EMISSION
- ATTENTE-ACQ.

Les états initiaux sont PREMIER-VIDE pour l'entité réceptrice, PRET-EMISSION pour l'entité émettrice.

Lorsqu'il y a demande d'émission (T-EX-REQ), si l'état est PRET-EMISSION, le fournisseur de services émettra une T.P.D.U. ED, et prendra l'état ATTENTE-ACQ. Sinon, il signalera un engorgement des données express.

A la réception d'une T.P.D.U. EA, le fournisseur de services signalera une erreur s'il avait l'état PRET-EMISSION, sinon il passera de l'état ATTENTE-ACQ. à l'état PRET-EMISSION.

Du côté de la réception, lorsque le fournisseur de services reçoit une T.P.D.U. ED, trois cas peuvent se présenter :

- Il est en état PREMIER-VIDE : il y a rangement du contenu de la T.P.D.U. dans le premier tampon, émission d'une T.P.D.U. EA, et passage à l'état PREMIER-PLEIN.

- Il est en état PREMIER-PLEIN : il y a rangement dans le deuxième tampon, et passage à l'état SECOND-PLEIN.

- Il est en état SECOND-PLEIN : signalisation d'une erreur de protocole.

Lorsque l'utilisateur des services veut s'enquérir de la présence de données express (par la fonction T-EX-IND), les trois cas suivant peuvent se présenter :

- Il est en état PREMIER-VIDE : la fonction retourne la valeur "faux".

- Il est en état PREMIER-PLEIN : la fonction prend la valeur "vraie", le fournisseur délivre les données contenues dans le premier tampon, et passe à l'état PREMIER-VIDE.

- Il est en état SECOND-PLEIN : la fonction prend la valeur "vrai", le fournisseur délivre les données contenues dans le premier tampon, transferre les données contenues dans le second tampon vers le premier tampon, émet une T.P.D.U EA, et passe à l'état PREMIER-PLEIN.

De la sorte, les spécifications sont bien respectées, on ne peut émettre une T.P.D.U. ED que lorsque la première a été acquiescée.

8.2.6. Traitement d'erreurs.

Lors de la mise en oeuvre de la classe 2, il n'est pas spécifié ce qui doit se passer en cas d'erreur de protocole. Dans notre réalisation, nous avons classé les erreurs en deux types : celles provenant de l'utilisateur des services, et celles provenant des indications de l'entité réseau.

Pour les premières, la variable ERR présente dans toute primitive prend une valeur non nulle, dépendante du type d'erreur. La façon dont réagit l'utilisateur de service "fautif" est une décision locale.

Pour les erreurs de protocole provenant du réseau (numéro de T.P.D.U. invalide, T.P.D.U. non attendue, etc...), il y a émission d'une T.P.D.U. ER, c'est à dire une unité de données de protocole de transport d'erreur, avec un code signalant l'erreur. Vu que la classe 2 ne fait pas de reprises sur erreur, à la réception d'une telle T.P.D.U., l'entité effectue une requête de libération de connexion.

8.2.7. Diagramme d'enchainement des primitives.

Le lecteur trouvera en annexe une table d'état déterminant

les actions à prendre en fonction des évènements entrants et de l'état de l'entité.

8.3. EXTENSIONS FUTURES.

Afin de fournir un service réseau très efficace, et sur une machine autorisant les interruptions (ou mieux le multi-tâches), une réalisation de la classe 4 constituerait une amélioration intéressante. Cette classe détecte toute anomalie de fonctionnement grâce à une batterie de temporisateurs destinés à déceler toute anomalie par l'expédition de T.P.D.U. de contrôle. Elle offre de plus certains services tels le total de contrôle au niveau d'une unité de données de protocole, qui complète le calcul du C.R.C. de la couche liaison de données d'ETHERNET.

De plus, le modèle O.S.I. prévoit l'utilisation possible dans cette classe du mode hors-connexion réseau, spécialement adaptée aux L.A.N. (ISO15) (ISO16) (ISO17) (ISO18) (STUDNITZ).

9. DESCRIPTION DE LA COUCHE SESSION.

Avant d'aborder la couche session, il faut préciser que dans ce texte, nous ne parlerons que du sous-ensemble des primitives nécessaire à l'architecture choisie, à savoir les primitives nécessaires à la réalisation de l'application de gestion des fichiers (F.T.A.M - voir 11.). Le lecteur intéressé trouvera en annexe la définition et la réalisation de TOUTES les fonctions définies dans les textes des normes relatifs à la définition du service et du protocole de la couche session.

L'ensemble de ces fonctions a été réalisé, mais par manque de capacité du système utilisé, il ne peut être mis en oeuvre simultanément.

9.1. SPECIFICATIONS DES SERVICES.

Rappelons que l'ensemble des en-têtes de procédures et de fonctions décrits ci-dessous sont les seules parties visibles par l'utilisateur des services de session. Les règles de dénomination des noms de ces procédures et fonctions sont celles reprises dans les conventions de services (voir 5.3.).

9.1.1. Concepts.

9.1.1.1. Unités fonctionnelles.

Afin de permettre une synchronisation du dialogue s'adaptant au plus grand nombre d'utilisations possible, la couche session est subdivisée en plusieurs sous-ensembles, appelés UNITES FONCTIONNELLES. On retrouve certaines de ces unités fonctionnelles dans les normes du C.C.I.T.T., certaines dans les propositions de l'E.C.M.A., etc..., ce qui pourrait faire croire que le concept de ces unités fonctionnelles a plutôt pour but de mettre d'accord toutes les parties concernées par l'interconnexion.

Le choix des unités fonctionnelles est négocié au moment de la création d'une connexion. Un utilisateur des services de session désigne les unités fonctionnelles dont il a besoin, le fournisseur de service vérifie qu'il a bien les capacités de les mettre en oeuvre, les propose à l'entité appelée, qui choisit celles qu'elle peut supporter, avant de les proposer à l'utilisateur des services de session appelé, qui parmi celles qui restent, choisit celles qui seront mises en oeuvre, à moins qu'il ne juge le restant insuffisant, auquel cas il refuse la création de la connexion (en précisant les unités qu'il peut supporter).

Voyons les différentes unités fonctionnelles proposées par l'O.S.I.

9.1.1.1.1. Noyau de base.

Le noyau de base contient les services nécessaires à la création d'une connexion session, au transfert des données, à la libération normale d'une connexion, et à sa coupure brutale. L'ensemble des primitives du noyau de base est toujours présent et n'est pas sujet à négociation.

9.1.1.1.2. Négociation de terminaison.

Cette unité fonctionnelle comprend le service de négociation de libération de connexion. L'utilisateur des services de session peut refuser la terminaison de connexion, signifiant par là qu'il a encore des données à émettre. Le jeton de négociation de libération (voir 9.1.1.2.) est disponible.

9.1.1.1.3. Transmission en semi-duplex.

Lorsque cette unité fonctionnelle a été sélectionnée, la transmission a lieu à l'alternat (l'un après l'autre) et le jeton de données (voir 9.1.1.2.) est disponible.

9.1.1.1.4. Transmission en duplex.

Cette unité ne peut être choisie qu'a l'exclusion de la précédente. Lorsqu'elle est sélectionnée, la transmission des données s'effectue de facon simultanée, et le jeton de données (voir 9.1.1.2.) n'est pas disponible.

9.1.1.1.5. Transfert de données express.

L'unité fonctionnelle de transfert de données express permet le transfert de données express sur une connexion session indépendemment de toute restriction imposée par le contrôle de flux ou l'attribution du jeton de données (voir 9.1.1.2.).

9.1.1.1.6. Transfert de données typées.

L'unité fonctionnelle de transfert de données typées permet de transférer des unités de données de service de session dites "typées", indépendemment de l'attribution du jeton des données (voir 9.1.1.2.).

9.1.1.1.7. Transfert d'informations de capacité.

Cette unité fonctionnelle permet de transférer une quantité limitée d'informations avec confirmation. Cette unité fonctionnelle ne peut être sélectionnée que si l'unité fonctionnelle de gestion d'activité a été sélectionnée (voir 9.1.1.1.12.).

9.1.1.1.8. Synchronisation mineure.

Cette unité propose à l'utilisateur la pose de points de synchronisation mineure (voir 9.1.1.3.). Le jeton de synchronisation mineure est disponible (voir 9.1.1.2.).

9.1.1.1.9. Synchronisation majeure.

Cette unité propose à l'utilisateur la pose de points de synchronisation majeure (voir 9.1.1.3.). Le jeton de synchronisation majeure (voir 9.1.1.2) est disponible.

9.1.1.1.10. Resynchronisation.

Cette unité fonctionnelle permet à l'utilisateur des services de session la pose de points de resynchronisation (voir 9.1.1.4.).

9.1.1.1.11. Signalisation d'anomalies.

L'unité fonctionnelle de signalisation d'anomalies comprend les services de signalisation d'anomalie par l'utilisateur des services et par le fournisseur des services. Cette unité fonctionnelle ne peut être choisie que si l'unité fonctionnelle de transmission semi-duplex a été choisie.

9.1.1.1.12. Gestion d'activité.

Lorsque cette unité fonctionnelle a été choisie, l'utilisateur des services de session peut gérer les activités et le service de passation de contrôle (voir 9.1.1.5.). Le jeton de synchronisation majeure (voir 9.1.1.2.) est disponible.

9.1.1.2. Concept de jeton.

Un jeton est un attribut d'une connexion de session qui est attribué dynamiquement à un seul utilisateur du service de session à la fois, pour lui permettre de faire usage de certains services. La possession d'un jeton confère le droit d'utilisation exclusive du service associé au jeton.

Quatre jetons ont été définis :

- Le jeton de données (voir 9.1.1.1.3.).

- Le jeton de synchronisation mineure (voir 9.1.1.1.8.).

- Le jeton de synchronisation majeure et d'activité (voir 9.1.1.1.9. et 9.1.1.1.12.).

- Le jeton de terminaison (voir 9.1.1.1.2.).

Un jeton est toujours soit disponible, auquel cas il est attribué à un des utilisateurs (et non attribué à l'autre), qui a le droit aux services associés au jeton, soit indisponible auquel cas deux situations se présentent :

- Soit aucun utilisateur n'a accès aux services associés au jeton.

- Soit les deux entités y ont droit.

9.1.1.3. Concepts de synchronisation.

Les utilisateurs des services de session peuvent insérer des points de contrôle au cours de leur dialogue. Chaque point de contrôle est représenté par un numéro de série (compris entre 0 et 999 998). Les numéros de série des points de contrôle sont attribués par les utilisateurs des services session, mais leur validité est contrôlée par le fournisseur de services.

On trouve deux types de synchronisation :

- La synchronisation majeure (voir 9.1.1.1.9.), et

- La synchronisation mineure (voir 9.1.1.1.8.).

La synchronisation majeure a pour but de structurer l'échange de données en une suite "d'unités de dialogue". Une unité de dialogue a comme caractéristique que tous les éléments échangés durant celle-ci sont complètement séparés de ceux qui la précèdent et de ceux qui la suivent. Un point de synchronisation majeure indique la

fin d'une unité de dialogue et le début de la suivante. Tout point de synchronisation majeure fait l'objet d'une confirmation explicite.

La synchronisation mineure sert à structurer une unité de dialogue. Chaque point de synchronisation mineure peut faire, ou ne pas faire, l'objet d'une confirmation explicite.

La fig 9-1 represente schématiquement la structuration d'une unité de dialogue.

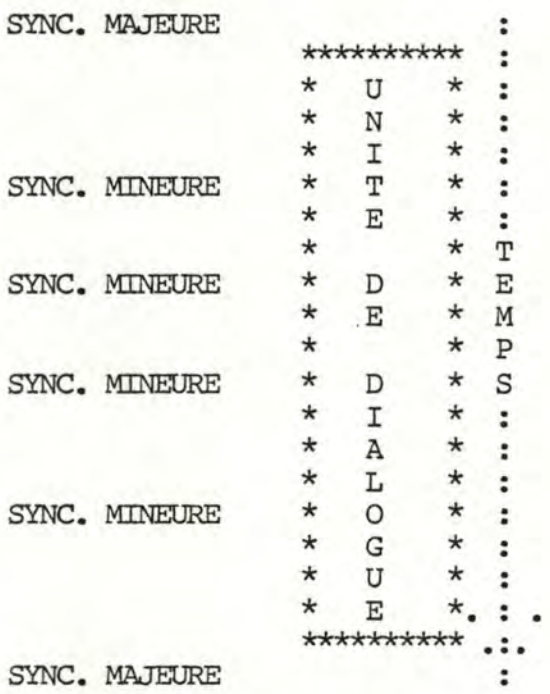


fig 9-1 exemple de structuration d'unité de dialogue.

9.1.1.4. Resynchronisation.

La resynchronisation peut être lancée par les deux utilisateurs de services de session. Elle sert à mettre la connexion dans un état défini, permettant l'attribution d'une nouvelle valeur au numéro de série du point de synchronisation, et une nouvelle attribution des jetons disponibles. Suite à une resynchronisation, le fournisseur des services élimine toutes les données reçues qu'il n'a pas encore délivré à l'utilisateur des services.

Trois options ont été définies :

- L'abandon (option ABANDON) : elle sert à mettre le numéro de série à une valeur non encore utilisée.

- Le redémarrage (option RESTART) : cette option est

utilisée pour attribuer au numéro de série une valeur ayant deja été utilisée, supérieure au numéro de série du dernier redémarrage, mais inférieure ou égale au numéro de série de la dernière synchronisation mineure émise.

- Le choix de l'utilisateur (option SET) : elle sert à attribuer une nouvelle valeur au numéro de série, valeur quelconque choisie par l'utilisateur des services.

9.1.1.5. Concept d'activité.

Le concept d'activité a été défini par conformité aux normes diffusées par le C.C.I.T.T. dans le cadre de la normalisation du télétex (CCITT2).

Une activité est un ensemble d'unités de dialogue. Une seule activité est autorisée à la fois. Une activité démarre, peut être interrompue, éventuellement reprise au cours de la même connexion session ou lors d'une connexion session ultérieure, peut être abandonnée ou cloturée normalement. La gestion des activités ne peut être utilisée que si cette unité fonctionnelle a été choisie (voir 9.1.1.1.12.).

Lorsque l'unité fonctionnelle d'activité a été sélectionnée, et qu'aucune activité n'a encore démarré, les entités peuvent s'échanger des informations de capacités (voir 9.1.1.1.7.) si cette unité fonctionnelle a été sélectionnée, afin de connaitre leurs capacités mutuelles.

Dans le cadre de cette réalisation, les unités fonctionnelles de transmission semi-duplex, de synchronisation mineure et de resynchronisation ont été mises en oeuvre, étant les seules nécessaires à la réalisation du protocole du F.T.A.M. Les autres sont données en annexe.

9.1.2. Spécification des interfaces.

9.1.2.1. Définitions communes.

```
TYPE ARRAY-24=PACKED ARRAY(1..24) OF CHAR;
     ARRAY-14=PACKED ARRAY(1..14) OF CHAR;
     ARRAY-2 =PACKED ARRAY(1..2 ) OF CHAR;
     ONE-BYTE =0..255;
     TWO-BYTES=0..MAXINT;
     SIX-BYTES=PACKED ARRAY(1..6) OF ('0'..'9');
     TYP-SSDU=VAL :ARRAY(1..1) OF 0..255;
```

Comme pour la couche transport, le type des unités de données de service sera :

```
TYPE TYPLOCAL=RECORD
              (* SELON BESOINS *);
             END;
     TYPDATA=RECORD
             CASE BOOLEAN
               OF TRUE :(SSDU:TYP-SSDU);
                  FALSE:(DATA:TYPLOCAL)
             END;
```

Sauf pour la requête et l'indication de données (ou elle est illimitée), la longueur maximale des unités de données de service de session est limitée à 512 octets.

Dans toutes les primitives, nous trouverons une variable ERR, dont la valeur est différente de zéro pour tout appel invalide de la primitive (erreur de protocole, engorgement, manque de ressources, unité fonctionnelle non mise en oeuvre, identificateur d'extrémité de connexion invalide, point d'accès aux services appelé inconnu, etc...). Certains de ces codes signalent des erreurs non persistantes (engorgement..), d'autres des erreurs persistantes (erreur de protocole, identificateur d'extrémité de connexion invalide, ...).

9.1.2.2. Création de connexion.

```
PROCEDURE S-CON-REQ
   (VAR S-IDT:TYP-IDT;
    CALLING-REFERENCE:ARRAY-24;
    COMMON-REFERENCE:ARRAY-14;
    ADDITIONAL-REFERENCE:ARRAY-2;
    CALLING-SSAP:TYP-SAP;
    CALLED-SSAP:TYP-SAP;
    SES-REQ:TWO-BYTES;
    NUM:SIX-BYTES;
    TOKEN:ONE-BYTE;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER);


FUNCTION S-CON-IND
   (VAR S-IDT:TYP-IDT;
    VAR CALLING-REFERENCE:ARRAY-24;
    VAR COMMON-REFERENCE:ARRAY-14;
    VAR ADDITIONAL-REFERENCE:ARRAY-2;
    VAR CALLING-SSAP:TYP-SAP;
    VAR CALLED-SSAP:TYP-SAP;
    VAR SES-REQ:TWO-BYTES;
    VAR NUM:SIX-BYTES;
    VAR TOKEN:ONE-BYTE;
    VAR SSDU:TYP-SSDU;
```

```
VAR ERR:INTEGER):BOOLEAN;
```

De par l'absence de multiplexage entre l'adresse de point d'accès de services de transport et l'adresse de point d'accès de services de session, ces deux identificateurs d'extrémité de connexion (S-IDT) seront identiques, et leur gestion également.

L'identification du demandeur de la connexion se fait au moyen de trois valeurs :

- La référence personnelle du demandeur (CALLING-REFERENCE).
- Une référence commune (COMMON-REFERENCE).
- Une référence supplémentaire (ADDITIONAL-REFERENCE).

Le fournisseur des services de session n'utilise pas ces références, et se contente de les transmettre à l'entité appelée.

On trouve l'origine de la présence de ces références en analysant l'avis S.62 du C.C.I.T.T. (CCITT2). La référence de l'utilisateur appelant se nomme dans l'avis S.62 "l'identification du terminal demandeur" (et comme nous le verrons plus bas la référence de l'utilisateur appelé sera "l'identification du terminal demandé"), la référence commune deviendra "la date et l'heure", et la référence additionnelle "le numéro de référence d'échange supplémentaire". Ce petit exemple nous permet d'illustrer la guerre d'influence entre ces deux instituts de normalisation (voir 4.2.).

Les adresses de point d'accès de service (CALLING-SSAP et CALLED-SSAP) servent à créer la connexion.

Les unités fonctionnelles souhaitées (SESREQ pour SESsion REQuirements) sont codées sur 16 bits, et pour chaque unité, le bit est mis à "1" lorsque l'unité est souhaitée. Voici la description du codage :

```
bit 1  = transmission semi-duplex
    2  = transmission duplex
    3  = tranfert de données express
    4  = synchronisation mineure
    5  = synchronisation majeure
    6  = resynchronisation
    7  = gestion d'activité
    8  = négociation de terminaison
    9  = transfert d'information de capacités
    10 = signalisation d'anomalies
    11 = transfert de données typées.
```

Rappelons que le modèle réalisé ne supporte que la transmission duplex ou semi-duplex, la synchronisation mineure et la resynchronisation.

Si la synchronisation mineure ou majeure, ou la resynchronisation ont été souhaitées, un numéro de série initial doit être convenu. Ce numéro de série est codé en ASCII, et comprend de 1 à 6 chiffres.

Lorsque des jetons sont disponibles (selon les choix des unités fonctionnelles), leur attribution est nécessaire. Au moment de la requête de connexion, trois valeurs sont possibles, codées sur 2 bits :

- Attribué à l'entité appelante : code 00

- Attribué à l'entité appelée : code 01

- Laissé au choix de l'entité appelée : code 10.

Le codage des jetons est effectué comme suit :

- jeton de terminaison : bits 8 & 7
  (non mis en oeuvre).

- jeton de synchronisation majeure et de gestion d'activité : bit 6 & 5
  (non mis en oeuvre).

- jeton de synchronisation mineure : bit 4 & 3.

- jeton des données : bit 2 & 1. .

Les données peuvent comporter jusqu'a 512 octets, transparents au fournisseur des services de session, et reproduits tels quels dans l'indication de connexion.

```
PROCEDURE S-CON-RSP
   (S-IDT:TYP-IDT;
    CALLED-REFERENCE:ARRAY-24;
    COMMON-REFERENCE:ARRAY-14;
    ADDITIONAL-REFERENCE:ARRAY-2;
    RESPONDING-SSAP:TYPSAP;
    RESULT:INTEGER;
    SESREQ:TWO-BYTES;
    NUM:SIX-BYTES;
    TOKEN:ONE-BYTE;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER);

FUNCTION S-CON-CNF
```

```
(S-IDT:TYP-IDT;
 VAR CALLED-REFERENCE:ARRAY-24;
 VAR COMMON-REFERENCE:ARRAY-14;
 VAR ADDITIONAL-REFERENCE:ARRAY-2;
 VAR RESPONDING-SSAP:TYPSAP;
 VAR RESULT:INTEGER;
 VAR SESREQ:TWO-BYTES;
 VAR NUM:SIX-BYTES;
 VAR TOKEN:ONE-BYTE;
 VAR SSDU:TYP-SSDU;
 VAR ERR:INTEGER):BOOLEAN;
```

L'entité appelée répond à une indication de connexion (S-CON-IND) par une réponse de connexion (S-CON-RSP). La variable RESULT aura une valeur nulle si la connexion est acceptée, un code de refus autrement.

Les différents paramètres ont la même signification que dans la requête, excepté que les unités fonctionnelles ne sont plus souhaitées, mais dans la confirmation elles sont sélectionnées.

L'attribution des jetons doit également être décidée (l'option "choix de l'entité appelée" n'est pas valide).

En cas de refus de connexion (code RESULT différent de zéro), les paramètres sont donnés à titre indicatif, par exemple la variable SESREQ contient les unités fonctionnelles supportées par l'entité appelée.

7.1.2.3. Libération de connexion

```
PROCEDURE S-REL-REQ
   (S-IDT:TYP-IDT;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER);

FUNCTION S-REL-IND
   (S-IDT:TYP-IDT;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER):BOOLEAN;
```

La procédure de requête de libération de connexion (S-REL-REQ de l'anglais RELease) peut être demandée par les deux entités si l'unité fonctionnelle de négociation de libération de connexion n'a PAS été sélectionnée, ce qui est le cas ici. Autrement, seule l'entité propriétaire du jeton de libération peut effectuer cette requête.

```
PROCEDURE S-REL-RSP
   (S-IDT:TYPDIT;
    RESULT:INTEGER;
```

```
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER);

FUNCTION S-REL-CNF
   (S-IDT:TYPIDT;
    VAR RESULT:INTEGER;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER):BOOLEAN;
```

Le paramètre RESULT prend une valeur différente de zéro si l'entité réceptrice d'une indication de libération refuse de libérer la connexion. Elle ne peut le faire que si l'unité fonctionnelle de négociation de libération a été choisie. Dans notre réalisation, cette option n'est pas mise en oeuvre, ce qui entraine que l'entité ne peut qu'accepter la libération, et le paramètre RESULT est toujours égal à zéro.

9.1.2.4. Transfert de données.

```
PROCEDURE S-DT-REQ
   (S-IDT:TYP-IDT;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER);

FUNCTION S-DT-IND
   (S-IDT:TYP-IDT;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER):BOOLEAN;
```

La taille maximale de l'unité de données de service n'est ici pas définie (elle est plus exactement limitée à 64 Koctets). La fonction d'indication ne sera vraie que si toute l'unité de données a été reçue, indépendemment de toute segmentation réalisée par le fournisseur des services.

Dans la première version de la couche session était prévu un service de mise en quarantaine, qui n'est plus défini dans la version actuelle. C'est dû au fait que l'utilisateur des services de session peut émettre une unité de données de service de longueur "illimitée", ayant l'assurance que l'indication de la réception de cette unité de données ne se fera qu'à la réception complète de l'unité.

9.1.2.5. Synchronisation mineure.

La pose de points de synchronisation mineure peut se faire à tout moment au cours de la vie d'une connexion. A chaque point de synchronisation est associé un numéro de série permettant l'identification de ce point de synchronisation.

L'entité qui pose un point de synchronisation peut demander la confirmation de ce point de synchronisation. Elle n'est cependant pas tenue d'arrêter toute transmission de données en attendant la confirmation.

```
PROCEDURE S-MIN-REQ
   (S-IDT:TYP-IDT;
    TYP-SYNC:ONE-BYTE;
    NUM:SIX-BYTES;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER);

FUNCTION S-MIN-IND
   (S-IDT:TYP-IDT;
    VAR TYP-SYNC:ONE-BYTE;
    VAR NUM:SIX-BYTES;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER):BOOLEAN;
```

Dans la pose de points de synchronisation mineure, le TYP-SYNC précise si l'entité émettrice de la requête attend une réponse de synchronisation mineure (bit 1 mis à "0") ou ne l'attend pas (bit 1 de TYP-SYNC mis à "1").

Le numéro de série (NUM) doit être supérieur d'une unité au numéro de la dernière synchronisation mineure, ou être égal au numéro contenu dans la S.P.D.U. de création de connexion ou dans celle de la dernière resynchronisation, selon la dernière primitive ayant eu lieu.

```
PROCEDURE S-MIN-RSP
   (S-IDT:TYPI-DT;
    NUM:SIX-BYTES;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER);

FUNCTION S-MIN-CNF
   (S-IDT:TYP-IDT;
    VAR NUM-SIX-BYTES;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER):BOOLEAN;
```

Lorsque la réponse à un point de synchronisation a été demandée, l'entité réceptrice de la synchronosation émet une réponse de synchronisation mineure (S-MIN-RSP), portant le numéro de synchronisation (NUM) de l'indication de synchronisation mineure. Ce numéro est indispensable, car plusieurs points de synchronisation peuvent être émis par une entité avant de recevoir la première confirmation.

9.1.2.6. Resynchronisation.

Une resynchronisation peut être émise par les deux entités, à tout moment suivant l'établissement de la connexion, et permet de modifier le numéro de série de synchronisation, purgeant ainsi les données non encore traitées, et de changer l'attribution des jetons.

Apres avoir émis une requête de resynchronisation (S-RS-REQ), l'entité ne peut qu'attendre la confirmation de la synchronisation.

L'entité qui reçoit une indication de resynchronisation arrête de délivrer des données, et doit émettre une réponse de resynchronisation.

```
PROCEDURE S-RS-REQ
   (S-IDT:TYP-IDT;
    RSYN:ONE-BYTE;
    NUM:SIX-BYTES;
    TOKEN:ONE-BYTE;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER);

FUNCTION S-RS-IND
   (S-IDT:TYPDIT;
    VAR RSYN:ONE-BYTE;
    VAR NUM:SIX-BYTES;
    VAR TOKEN:ONE-BYTE;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER):BOOLEAN;

PROCEDURE S-RS-RSP
   (S-IDT:TYP-IDT;
    NUM:SIX-BYTES;
    TOKEN:ONE-BYTE;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER);

FUNCTION S-RS-CNF
   (S-IDT:TYP-IDT;
    VAR NUM:SIX-BYTES;
    VAR TOKEN:ONE-BYTE;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER):BOOLEAN;
```

Le paramètre de type de resynchronisation (RSYN) indique le genre de confirmation requise :

0 : option "redémarrage" (RESTART)

1 : option "abandon" (ABANDON)

2 : option "choix de l'utilisateur" (SET).

La valeur du numéro de série (NUM) sera fonction du type de resynchronisation choisi.

La codification des jetons est identique à celle de la création de connexion (voir 9.1.2.2.), de même que les remarques s'y rapportant.

9.1.2.7. Attribution des jetons.

A tout moment, une entité peut offrir un jeton qu'elle possède à l'autre entité.

```
PROCEDURE S-GT-REQ
   (S-IDT:TYP-IDT;
    TOKEN:ONE-BYTE;
    VAR ERR:INTEGER);

FUNCTION S-GT-IND
   (S-IDT:TYP-IDT;
    VAR TOKEN:ONE-BYTE;
    VAR ERR:INTEGER):BOOLEAN;
```

La codification des jetons transmis est la suivante :

bit 7 : jeton de terminaison

bit 5 : jeton de synchronisation majeure
et de gestion d'activité

bit 3 : jeton de synchronisation mineure

bit 1 : jeton de données.

Si la valeur du bit est "1", l'attribution du jeton passe à l'entité recevant l'indication.

Il faut remarquer que le passage des jetons est le seul service ne permettant pas de transférer des unités de données de service. Ceci est dû au fait que ce service est presque toujours associé à une requête de transmission de données.

9.1.2.8. Coupure de connexion.

La requête de coupure de connexion (S-UAB-REQ) peut être émise par les deux entités à tout moment durant la vie de la connexion.

L'indication de coupure de connexion peut apparaître suite à une requête de l'autre entité (S-UAB-IND), mais peut également être faite par le fournisseur de services

```
(S-PAB-IND).

PROCEDURE S-UAB-REQ
   (S-IDT:TYP-IDT;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER);

FUNCTION S-UAB-IND
   (S-IDT:TYPIDT;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER):BOOLEAN;

FUNCTION S-PAB-IND
   (S-IDT:TYPIDT;
    VAR CAUSE:ONE-BYTE;
    VAR ERR:INTEGER):BOOLEAN;
```

La procédure de requête de coupure de connexion (S-UAB-REQ), ainsi que la fonction d'indication de coupure (S-UAB-IND) de l'utilisateur de service (de l'anglais User ABort) ne spécifie pas la cause de la coupure, tandis que la fonction d'indication de coupure (S-PAB-IND) du fournisseur de service (de l'anglais Provider ABort) précise la raison de la déconnexion dans le paramètre CAUSE, et est signalée aux deux entités.

9.1.2.9. Fonctions de test d'activité.

Vu l'asynchronisme règnant entre la couche transport et la couche session, deux fonctions permettant l'activation d'une entité session ont été prévues :

```
FUNCTION NEW-CON
   (VAR S-IDT:TYP-IDT;
    SSAP:TYP-SAP):BOOLEAN;

FUNCTION NET-ACT
   (S-IDT:TYP-IDT;
    VAR ERR:INTEGER):BOOLEAN;
```

La première sert à détecter l'indication d'une nouvelle connexion demandée par une autre connexion. L'adresse de point d'accès des services sert à déterminer si l'utilisateur des services de session est bien l'utilisateur appelé. Si la fonction est vraie, l'identificateur d'extrémité de connexion prend une valeur identifiant la connexion.

La seconde fonction sert à activer l'entité session liée à l'identificateur d'extrémité de connexion. Le paramètre de détection d'erreur peut être différent de zéro si l'identificateur d'extrémité de connexion est

invalide.

Ces deux primitives ne sont pas prévues dans les normes, mais ont été rajoutées dans notre réalisation.

9.2. DESCRIPTION DES PROTOCOLES.

9.2.1. Utilisation de la couche transport.

9.2.1.1. Affectation à une connexion transport.

Lors d'une requête de connexion de session, le fournisseur de services va créer une connexion transport au moyen de la requête de connexion transport (T-CON-REQ voir 8.1.1.). Ensuite l'entité va attendre une confirmation de connexion (T-CON-CNF voir 8.1.1.), ou une indication de déconnexion (T-DIS-IND voir 8.1.3.).

La fonction NEW-CON sert à détecter si une nouvelle connexion est demandée. Pour cela, l'entité va tester l'entité transport via l'indication de connexion transport (T-CON-IND voir 8.1.1.), et si cette fonction a la valeur "vrai", elle va accepter la connexion (T-CON-RSP voir 8.1.1.) si elle possède assez de ressources pour supporter la connexion, ou la refuser par une requête de déconnexion (T-DIS-IND voir 8.1.3.). La fonction NEW-CON prendra alors la valeur "vrai".

9.2.1.2. Utilisation des données normales.

Toute unité de données de protocole de session est émise via la requête de données de la session (T-DT-REQ voir 8.1.2.). Toute l'unité est expédiée en un seul appel, la segmentation éventuelle étant prise en charge par l'entité transport. Tant qu'une unité de données de protocole n'est pas complètement reçue, la fonction d'indication de données de transport (T-DT-IND voir 8.1.2.) prendra la valeur "faux".

9.2.1.3. Utilisation des données express.

Les unités de données de service express du transport sont utilisées pour transférer des petites unités de données de protocole. Son utilisation a pour but d'éviter le contrôle de flux des données normales, ce qui est particulièrement intéressant en cas de congestion. On y transferre des unités de données relatives à la coupure, et à la préparation à la réception de certaines unités de données de protocole telles la resynchronisation.

9.2.2. Spécifications du protocole.

9.2.2.1. Création de connexion.

Lorsqu'une entité désire créer une connexion, elle va d'abord créer une connexion transport (voir 9.2.1.1.), ensuite elle va émettre une S.P.D.U. CN (connect), contenant les paramètre suivants (voir aussi 9.1.1.2.) :

- L'identification de l'utilisateur appelant :
  - La référence de l'utilisateur appelant.
  - La référence commune.
  - La référence additionnelle.

- La taille maximum des T.S.D.U que cette entité est apte à émettre et à recevoir, une valeur pour chaque sens de transfert. Cette valeur peut être nulle, auquel cas il n'y a pas de segmentation des unités de données de protocole, et on suppose que le fournisseur de service de transport peut émettre des unités de données de service de taille illimitée.

- Une option de protocole indiquant que l'entité peut concaténer plusieurs unités de données de protocole session sur une unité de données de service de transport (toujours refusée dans notre réalisation).

- Un numéro de version, égal à 1 (première version du fournisseur de services).

- Le numéro de série initial proposé.

- L'attribution des jetons disponibles selon le choix des unités fonctionnelles proposées :
  - attribué à l'entité appelante,
  - attribué à l'entité appelée, ou
  - au choix de l'entité appelée.

- L'ensemble des unités fonctionnelles proposées.

- Les adresses de point d'accès des services appelante et appelée.

- Les données de l'utilisateur de service.

Après émission de cette S.P.D.U., l'entité se met en attente d'une S.P.D.U. AC (accept) ou d'une S.P.D.U. RF (refuse).

A la réception d'une S.P.D.U. CN, l'entité émet une indication de connexion et attend une réponse de connexion de l'utilisateur de services.

La réponse de connexion peut engendrer l'émission de deux types de S.P.D.U., selon que le paramètre RESULT (voir 9.1.1.2.) ait une valeur nulle (si la connexion est acceptée), ou non nulle (en cas de refus).

La S.P.D.U. AC d'acceptation de connexion comprend les paramètres suivants :

- Le groupe d'identificateur de connexion :
  - Référence de l'utilisateur appelé.
  - Référence commune.
  - Référence additionnelle.

- L'option de protocole, permettant de traiter des unités de données de protocole concatenées (voir ci-dessus), option qui peut avoir la valeur "vrai" si elle l'était déja dans la S.P.D.U. CN.

- La taille maximum des T.S.D.U., qui est pour chaque direction de transfert égale à la plus petite valeur entre celle proposée et celle supportée par l'entité appelée.

- Le numéro de version.

- Le numéro de série initial, nécessaire si l'unité fonctionnelle de synchronisation mineure, de synchronisation majeure ou de resynchronisation a été choisie, et que la gestion d'activité n'a pas été choisie.

- L'attribution initiale des jetons. Il n'y a que deux valeurs possibles dans la S.P.D.U. AC :

  - attribué à l'entité appelante, ou
  - attribué à l'entité appelée.

- La liste des unités fonctionnelles choisies pour être mises en oeuvre sur la connexion. Si le transfert en semi-duplex et le transfert en duplex avaient été tous les deux proposés dans la S.P.D.U. CN, alors une seule des deux unités fonctionnelles doit être présente.

- Les adresses de point d'accès des services des entités appelante et appelée.

- Les données de l'utilisateur.

Après l'émission d'une S.P.D.U. AC, l'entité appelée

entre dans un état de transmission et de réception de données.

A la réception d'une S.P.D.U. AC, l'entité appelante met à jour les unités fonctionnelles choisies, le numéro de série initial, l'attribution des jetons, et se met en état de transfert de données.

Si l'entité appelée refuse la connexion (paramètre RESULT different de zero), elle émet une S.P.D.U. RF, dont les paramètres sont :

- L'identification de l'utilisateur appelé :
  - référence de l'utilisateur appelé,
  - référence commune, et
  - référence additionnelle.

- Un paramètre de déconnexion de transport, spécifiant si la connexion transport pourra être réutilisée ou non. Dans notre réalisation, ce paramètre sera toujours à "faux".

- La liste des unités fonctionnelles supportées par l'entité appelée.

- Un paramètre codant la raison de refus de la connexion, et qui apparaitra dans la primitive de confirmation de connexion dans le paramètre RESULT.

Lorsqu'elle émet une S.P.D.U. RF, l'entité appelée enclenche un temporisateur, et attend soit l'expiration du temporisateur, soit une indication de déconnexion transport. Si le temporisateur expire le premier, l'entité émet une requête de déconnexion transport.

A la réception d'une S.P.D.U. RF, l'entité exécute une requête de déconnexion de transport. La connexion cesse d'exister.

9.2.2.3. Libération de connexion.

La libération de connexion ne peut être émise que par l'entité possédant tous les jetons disponibles. Pour demander la libération de la connexion, l'entité émet une S.P.D.U. FN (finish). Cette S.P.D.U. contient comme paramètres :

- Le paramètre de déconnexion de transport (voir plus haut), toujours à "faux" dans notre réalisation.

- Les données de l'utilisateur.

Lorsqu'une entité émet une S.P.D.U. FN, elle refuse toute autre S.P.D.U. que la S.P.D.U. DN (disconnect), ou la S.P.D.U. NF (not finish), cette dernière étant valide si l'unité fonctionnelle de négociation de libération a été sélectionnée (ce qui n'est pas le cas dans notre réalisation).

A la réception d'une S.P.D.U. FN, l'entité émet une indication de déconnexion et attend une réponse de déconnexion. Cette réponse de déconnexion entrainera l'émission d'une S.P.D.U. DN. Cette S.P.D.U. DN contient pour tout paramètre les données de l'utilisateur du service. Après l'émission de cette S.P.D.U., l'entité enclenche un temporisateur, et attend une indication de déconnexion de transport. Si le temporisateur expire avant l'indication de déconnexion transport, alors l'entité émet une requête de déconnexion transport.

A la réception d'une S.P.D.U. DN, l'entité cloture la connexion et exécute une requête de déconnexion transport.

9.2.2.4. Coupure de connexion.

La coupure sert à provoquer à tout moment une libération anormale de connexion. La coupure peut être provoquée par une entité via une requête de coupure, ou par le fournisseur de service au moment d'une détection d'erreur de protocole.

La S.P.D.U. AB (abort) contient l'option de déconnexion de transport, ainsi qu'une petite quantité de données utilisateur (max 16 octets) si cette S.P.D.U. a été émise suite à une requête de coupure, ou une petite quantité de données dont la définition dépend de la réalisation du système en cas de coupure faite par le fournisseur de services.

Si le transfert d'unités de données express du service de transport est disponible, la S.P.D.U. AB est émise surt ce flux express, sinon elle est émise sur le flux des données normales. L'entité enclenche alors un temporisateur, et attend une indication de déconnexion de transport. Si le temporisateur expire avant cette indication, l'entité émet une requête de déconnexion de transport.

A la réception d'une S.P.D.U. AB, l'entité considère que la connexion session cesse d'exister et émet une requête de déconnexion de transport.

Note : Dans la définition, si l'option de non déconnexion de transport est souhaitée, l'entité réceptrice de la S.P.D.U. AB peut émettre une S.P.D.U. AA (abort accept), ne contenant aucun paramètre, sur le flux des données express s'il est disponible, sur le flux des données normales autrement. Vu que dans notre réalisation cette option n'est pas prévue, nous ne nous étendrons pas davantage sur cette possibilité.

9.2.2.5. Transfert de données.

Suite à une requête d'émission de données, une S.P.D.U. DT (data) contient un paramètre de délimitation, indiquant si les octets qui suivent sont le début ou la fin d'une unité de données de service de session, ou une portion intermédiaire. Ce paramètre ne peut être présent que si le paramètre de taille de T.S.D.U. de la S.P.D.U. AC est différent de zéro.

L'indication de données se fait à la réception de la S.P.D.U. DT si la segmentation n'a pas été sélectionnée, ou à la réception du dernier segment de l'unité de données de service de session si la segmentation a été choisie (taille maximum des T.S.D.U différente de zéro pour cette direction de transfert).

Si la segmentation a été adoptée, la réception d'une S.P.D.U. de resynchronisation, de réponse à la synchronisation, ou de coupure peut avoir un effet destructif sur l'unité de données de service complète.

Le droit d'émission de données est sujet à un contrôle de la possession du jeton de données si l'unité fonctionnelle de transfert de données en semi-duplex a été sélectionnée.

9.2.2.6. Passage des jetons.

Une entité peut, lors du transfert des données, décider de céder des jetons qu'elle possède, au moyen d'une requête de session de jetons. La S.P.D.U. GT (give tokens) contient comme seul paramètre la liste des jetons transférés.

A la réception d'une S.P.D.U. GT, l'entité peut considérer qu'elle possède les jetons spécifiés dans le paramètre des jetons cédés.

L'émission de S.P.D.U. DT doit se faire avec la concaténation d'une S.P.D.U. GT. Dans ce cas, seule la

dernière partie de l'unité de données de service contiendra éventuellement le paramètre contenant la liste des jetons cédés. Cette concatenation se fait de façon transparente à l'utilisateur des services de session.

9.2.2.7. Synchronisation mineure.

La pose de points de synchronisation mineure se fait via une S.P.D.U. MIP (minor point) et la réponse à une synchronisation mineure par une S.P.D.U. MIA (minor ack). La décision d'attendre une réponse à une S.P.D.U. MIP est décidée par les utilisateurs de service de session, et est transparente au fournisseur de service.

La S.P.D.U. MIP contient les paramètres :

- Type de synchronisation, indiquant si une confirmation est attendue.

- Le numéro de série servant à identifier le point de synchronisation.

- Les données de l'utilisateur.

Lors de l'émission d'une S.P.D.U. MIP, l'entité va vérifier si le numéro de série est valide, et se mettre dans un état tel que la réception d'une S.P.D.U. MIA est valide ou non selon le paramètre de type de synchronisation.

La réception d'une S.P.D.U. MIP engendre une indication de synchronisation mineure.

Si la confirmation a été demandée dans le paramètre de type de synchronisation, une S.P.D.U. MIA doit être émise, contenant les paramètres numéro de série, identifiant le point de synchronisation, et des données uitilisateur.

La confirmation peut se faire dès la réception de l'indication de synchronisation, mais ne doit pas être spécialement faite immédiatement (d'autres S.P.D.U. DT peuvent être émises ou reçues entre-temps).

9.2.2.8. Resynchronisation.

La resynchronisation peut avoir plusieurs buts : s'approprier les jetons de manière brutale, purger des données non encore traitées... La resynchronisation s'effectue par l'envoi d'une S.P.D.U. RS (resynchronize), dont les paramètres sont :

- Un paramètre d'attribution des jetons, identique à celui de la S.P.D.U. CN (voir 9.2.2.1.).

- Un paramètre indiquant le type de resynchronisation, pouvant prendre les valeurs :

  - abandon,
  - redémarrage, ou
  - choix de l'utilisateur.

- Un paramètre numéro de série, indiquant le numéro de série à partir duquel la resynchronisation est demandée. Pour le choix du numéro de série, voir ci-dessus (voir 9.1.1.1.5.).

- Des données de l'utilisateur.

Lors d'une requête de resynchronisation, la S.P.D.U. RS est envoyée sur le flux des données normales. De plus, si le transfert des données express de transport est disponible, il y a émission d'une S.P.D.U. PR (prepare), qui a pour but de signaler à l'entité réceptrice qu'une S.P.D.U. RS va arriver.

Une S.P.D.U. PR comporte un seul paramètre, le type de préparation, qui peut être :

- Préparation à une resynchronisation.

- Préparation à une réponse de resynchronisation.

- Préparation à une réponse de synchronisation majeure (non utilisé dans notre réalisation).

Après avoir émis la S.P.D.U. RS, l'entité met au rebut toutes les S.P.D.U. entrantes sauf les S.P.D.U. PR, les S.P.D.U. RS (qui entraînent une situation de conflit), ou les S.P.D.U. AB.

A la réception d'une S.P.D.U. PR-RS (préparation de resynchronisation), l'entité réceptrice met au rebut toutes les S.P.D.U. autre que la S.P.D.U. RS (ou une S.P.D.U. AB).

La réception de la S.P.D.U. RS provoque une indication de resynchronisation, et l'entité attend une réponse de resynchronisation. Elle émet alors une S.P.D.U. RA (resynchronize ack), dont les paramètres sont :

- Un paramètre d'attribution des jetons, identique à la S.P.D.U. AC (voir 9.2.2.1.).

- Un paramètre numéro de série, qui selon le type de resynchronisation, prendra comme valeur :

- Redémarrage : le numéro de la S.P.D.U. RS.

- Choix de l'utilisateur : le numéro contenu dans la réponse.

- Abandon : le plus grand du numéro de série reçu dans la S.P.D.U. RS et du dernier numéro de synchronisation mineure.

- Les données de l'utilisateur.

Si le transfert de données express est supporté par l'entité transport, une S.P.D.U. PR-RA (préparation de réponse à la resynchronisation) sera émise sur le flux des données express, en plus de la S.P.D.U. RA. L'entité se retrouve alors dans un état de transfert normal de données.

A la réception d'une S.P.D.U. PR-RA, l'entité continue à mettre au rebut toute S.P.D.U., sauf la S.P.D.U. RA (ou une S.P.D.U. AB). A la réception de la S.P.D.U. RA, l'entité se remet dans un état de transfert normal de données, le numéro de série et l'attribution des jetons ayant étés mis à jour.

9.2.3. Réalisation.

La réalisation de la couche session peut être décomposée en quatre types de traitement :

- Les requêtes et les réponses.

- La détection d'évènements provenant de l'entité transport.

- Les indications et les confirmations.

- La détection de demandes de nouvelles connexions.

Ces quatre parties seront vues séparément, chacune regroupant des actions similaires.

Afin de gérer les ressources de chacune des entités, quatre procédures sont utilisées. La première sert à accéder aux variables locales de l'entité, la seconde à les mettre à jour, une troisième pour les créer, une quatrième pour les supprimer.

Ces variables locales comprennent tous les renseignements nécessaires au bon fonctionnement du fournisseur de service, tels l'état de l'entité, les unités fonctionnelles choisies, l'attribution des jetons, les tampons lors de la segmentation des unités de données de service, les fourchettes de numéros de série de synchronisation valables...

Afin de se simplifier la tâche, et vu que le service transport est TRES fiable (car il n'y a pas vraiment de transfert physique de données), le temporisateur utilisé ci-dessus ne parvient JAMAIS à expiration.

9.2.3.1. Requêtes et réponses.

Toute primitive de requête ou de réponse engendre

- Une vérification afin de déterminer si l'utilisateur des services a droit d'utiliser ce service (selon l'état et les unités fonctionnelles choisies).

- Une vérification de certains paramètres (numéro de série..)

- L'encodage de l'unité de données de service en une ou plusieurs unités de données de protocole.

- L'utilisation des services de transport pour effectuer l'acheminement des unités de données de protocole.

- La mise à jour de l'état et des variables locales de l'entité.

Si une des opérations ne peut être effectuée, ou si une vérification montre que quelque chose est erroné, le paramètre ERR de la primitive prend une valeur différente de zéro, et le processus est arreté.

9.2.3.2. Détection d'activité.

La fonction NET-ACT (voir plus haut) va engendrer l'exécution des fonctions T-DT-IND et T-DIS-IND de la couche transport. Si l'une ou l'autre prend la valeur "vrai", le fournisseur va exécuter les opérations suivantes :

- Déterminer le type de la S.P.D.U..

- Vérifier si sa présence est autorisée.

- Décoder et vérifier le contenu des paramètres sujets à vérification (numéro de série...).

- Mettre à jour l'état de l'entité et des variables locales.

- Si c'est le cas, mémoriser le fait qu'une fonction d'indication ou de confirmation peut être effectuée par l'utilisateur des services, auquel cas la fonction NET-ACT

prendra la valeur "vrai".

En cas de détection d'une erreur, l'entité émettra une S.P.D.U. AB et clôturera la connexion.

9.2.3.3. Indications et confirmations.

En cas d'indication ou de confirmation, le fournisseur de service vérifiera si cette primitive était bien attendue, et si c'est le cas, décodera la dernière S.P.D.U. reçue pour passer les paramètres, mettant la fonction à "vrai".

9.2.3.4. Détection d'indication de connexion.

Vu que toute indication de connexion doit être précédée d'une indication de connexion transport, le fournisseur de services va regarder si une telle indication est arrivée aun niveau de l'entité transport.

Si c'est le cas, le fournisseur de services va créer les variables locales à la connexion, l'identificateur d'extrémité de connexion, et mettre à "vrai" la fonction.

9.3. EXTENSIONS FUTURES.

Toutes les primitives de la couche session en mode connexion (ISO10) (ISO11) ont été réalisées, le reste des spécifications et de la description se trouvant en annexe. Il existe en outre un mode "hors-connexion" (ISO12), mais celui-ci n'est pas encore suffisamment défini pour être mis en oeuvre.

La seule partie non réalisée est la gestion du temporisateur de déconnexion de transport (voir plus haut). Nous imaginerions bien un jeu de trois procédures permettant de gérer les temporisateurs.

```
PROCEDURE START-TIMER
   (VAR REFERENCE-TIMER:INTEGER;
    DELAY:INTEGER);


FUNCTION WAIT
    (REFERENCE-TIMER:INTEGER):BOOLEAN;


PROCEDURE STOP-TIMER
   (REFERENCE-TIMER:INTEGER);
```

Le paramètre REFERENCE-TIMER servirait à référencer le temporisateur, et DELAY exprimerait une durée.

Rappelons que ces procédures sont déja insérées dans les programmes, ainsi que les actions découlant de l'expiration du temporisateur.

10. DESCRIPTION DE LA COUCHE PRESENTATION.

10.1. CONCEPTS.

10.1.1. Contexte de présentation.

Pour un utilisateur du service, un contexte de présentation représente les catégories d'informations q'il peut transférer ou recevoir sur une connexion de présentation à un moment donné.

10.1.2. Syntaxe de transfert.

Une syntaxe de transfert est l'ensemble des règles de représentation des données de l'utilisateur. Il y a une seule syntaxe de transfert correspondant à un contexte de présentation.

10.1.3. Jeu de contextes définis.

Un jeu de contextes définis est un ensemble de contextes de présentation qui ont été définis entre les trois parties concernées par la communication : les deux utilisateurs de services et le fournisseur de services.

10.1.4. Contexte par défaut.

Le contexte par défaut est un contexte de présentation défini par le service de présentation. Il autorise uniquement le transfert d'information sous forme de chaînes de bits. Ce contexte est d'office sélectionné lorsqu'aucun autre contexte n'est explicitement sélectionné.

10.1.5. Service de gestion de contexte.

La gestion des contextes fournit les éléments de service qui permettent :

- La définition et la redéfinition de contexte de présentation, par accord mutuel entre les utilisateurs de service. Un nom est associé à chaque contexte défini.

- La sélection via son nom d'un contexte de présentation défini.

- La suppression d'un contexte de présentation.

10.1.6. Niveaux de services.

Trois niveaux de services ont été déterminés, négociés selon les possibilités des entités de présentation communicantes. Ces trois niveaux sont les niveau 0, niveau 1 et niveau 2.

10.1.6.1. Niveau 0.

Les services de gestion de contexte ne sont pas disponibles dans ce niveau 0. Un seul contexte est utilisé durant toute la durée de la connexion. Ce contexte est défini par référence à un contexte prédéfini.

10.1.6.2. Niveau 1.

Le niveau 1 permet de choisir des contextes comme étant les contextes courants. Chacun de ces contextes est défini par référence au nom de contextes prédéfinis.

10.1.6.3. Niveau 2.

En plus des services offerts par le niveau 1, le niveau 2 permet de définir dynamiquement des contextes, soit en modifiant des contextes prédéfinis, soit sans référence à des contextes prédéfinis.

10.1.7. Contrôle du dialogue.

Toutes les primitives de contrôle du dialogue (synchronisation, resynchronisation...) sont identiques à celles de session. Une entité présentation ne fait que tranformer les primitives de présentation en primitives de session.

10.2. REALISATION.

Par manque de renseignements concernant les services et les protocoles de la couche présentation, le rôle de cette couche dans notre réalisation se bornera à tranformer la syntaxe PASCAL utilisée dans les protocoles du F.T.A.M.

10.3. EXTENSIONS FUTURES.

La définition de la couche transport est encore vague (ISO7) (ISO8). On spécifie des services de définition, de modification et de suppression de syntaxe de représentation, mais n'est pas encore défini le mécanisme de gestion de ces syntaxes.

Dans le cadre de cette réalisation, il nous semble que l'effort doit être axé sur les différents langages de programmation existants, afin de définir des règles de transformation, permettant, par exemple, un dialogue entre une application écrite en COBOL, et une application en PASCAL.

Lorsque sortiront les définitions des services et des protocoles de présentation, au moins sous forme de DIS (voir 4.), il sera temps de réaliser les services tels qu'ils sont actuellement proposés.

11. DESCRIPTION DE LA COUCHE APPLICATION.

11.1. CONCEPTS.

Dans le cadre de l'interconnexion des systèmes ouverts, l'I.S.O. a défini d'une part un certain nombre de primitives destinées à gérer le réseau (ISO3) (ISO4), et d'autre part a proposé trois applications pour lesquelles des protocoles ont été définis. De ces trois applications, nous en avons mis une en oeuvre, le F.T.A.M., relative à l'accès et à la gestion de fichiers.

Cette vue générale du modèle du F.T.A.M. est une synthèse des spécifications contenues dans les textes de normes DP8571/1, DP8571/2, DP8571/3 et DP8571/4 (ISO3) (ISO4) (ISO5) (ISO6). Nous invitons le lecteur désireux d'obtenir de plus amples informations à s'y référrer. Signalons dès à présent que seules les primitives d'accès aux commandes décrites ci-dessous ont été réalisées, et non un gérant de base de données.

11.1.1. Besoin d'un modèle de fichiers virtuels.

La façon dont sont mis en oeuvre les moyens de stockage de fichiers varie considérablement entre les systèmes. De plus, il existe beaucoup de façons d'accéder à différents fichiers, selon que ce soit un utilisateur humain, un gérant de base de données, un programme d'application accédant à des fichiers...

Ceci justifie l'utilité de créer un modèle de structuration de fichier, et de définir les protocoles d'accès aux éléments de ces fichiers (GIEN).

Dans le cadre de l'interconnexion des systèmes ouverts, nous aurons une transformation entre d'une part les actions et les éléments de données représentant le contenu des fichiers et leurs attributs, et d'une autre part les ressources des systèmes locaux.

11.1.2. Types de services fournis.

11.1.2.1. Contrôle du tranfert

Il y a trois entités concernées par le transfert de fichiers :

- Une entité qui prend l'initiative du contrôle du transfert,

- Une entité émettrice des données, et

- Une entité réceptrice des données.

De l'entité contrôlant le transfert, il y a deux flux d'informations : celui qui provient de l'entité émettrice des données, et celui provenant de l'entité réceptrice. Dans un but de simplification, on peut considérer que l'entité contrôlant le transfert ne va effectuer ses contrôles qu'à travers une seule des deux entités gérant les données. Dans beaucoup de cas, celà arrivera naturellement, le contrôleur et le gérant des données étant dans le même système.

11.1.2.2. Asymétrie du dialogue.

On trouve deux grandes asymétries dans le dialogue, qui se reflètent dans la structure des services et du protocole.

Tout d'abord, chaque activité (ensemble des opérations de transfert) est démarrée par un des utilisateurs des services de gestion des fichiers, et le système associé au gérant des fichiers joue un rôle passif, réagissant aux commandes de l'entité initiatrice.

La seconde asymétrie est liée au fait que, lorsque l'on transferre un fichier de données, une des entités est l'émettrice des données, l'autre en est la réceptrice.

11.1.2.3. Fonctions de contrôle.

On trouve deux fonctions de contrôles distinctes :

- Un service de transfert de fichier, dans lequel l'utilisateur ne se soucie ni de détection, ni de recouvrement d'erreurs. Pour cela, il fait confiance à la qualité de services fournie par les couches inférieures, et tout traitement d'erreurs est invisible à l'utilisateur.

- Un service dans lequel des primitives sont fournies pour effectuer la détection et le recouvrement d'erreurs. L'utilisateur a la possibilité de choisir parmi un certain nombre de procédures de gestion d'erreurs, selon le type d'application réalisé.

11.1.3. Vue générale des services et des protocoles.

Lors d'une session de transfert de fichiers, un certain nombres d'étapes doivent être exécutées. Ces étapes, appelées

PHASES, ont chacune un but particulier, et pour chaque phase, il n'existe qu'un certain nombre de messages valides. A tout moment une entité ne peut être que dans une seule phase, les phases ne pouvant se chevaucher.

La fig 11-1 illustre les différentes phases par lesquelles une entité doit passer au cours d'une connexion, ainsi que les opérations débutant et clôturant ces phases.

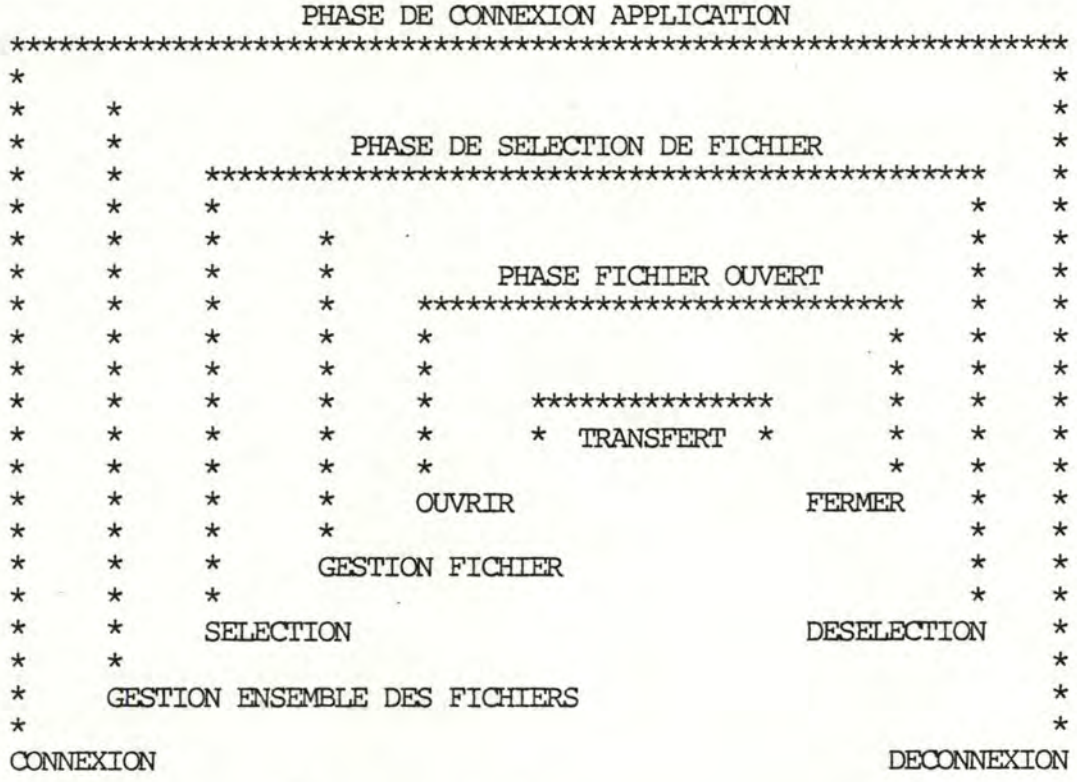


fig 11-1 phases d'une connexion.

11.1.3.1. Phase de connexion d'application.

Cette phase n'est pas spécifique au protocole de gestion de fichiers, mais est la phase établissant une connexion application.

11.1.3.2. Phase de sélection de fichiers.

Dans cette phase, l'utilisateur des services application identifie ou crée un fichier unique sur lequel porteront les opérations des phases ultérieures. La sélection portera sur le fichier jusqu'à la "désélection" (voir 11.1.3.7.) de cev fichier. De la sorte, toutes les opérations ultérieures ne devront plus référencer le fichier, celui-ci ayant été choisi pour cette connexion. Nous voyons de la sorte qu'il y aura autant de connexions

applications que de fichiers accédés simultanément.

Note : Le terme "désélection" sera utilisé, représentant mieux le sens à attribuer que les termes "clôture de sélection, fin de sélection...", bien que ce terme ne soit pas un terme français.

11.1.3.3. Gestion du fichier.

Dans cette phase, l'utilisateur a la possibilité d'effectuer certaines opérations sur les attributs du fichier (voir 11.1.6.).

11.1.3.4. Ouverture de fichier.

La phase d'ouverture du fichier établit un régime dans lequel le transfert de données peut avoir lieu.

11.1.3.5. Transfert des données.

Cette phase d'accès aux données consiste en un cerains nombre d'opérations. Chaque opération consiste en :

- Un opérateur spécifiant l'opération à accomplir, donnant le type de l'opération et identifiant l'unité de données à laquelle s'applique l'opération.

- Un transfert de données.

- Un signal de clôture d'opération.

11.1.3.6. Clôture de fichier.

Cette phase clôture le contexte opérationnel établi par l'ouverture. Elle fournit la confirmation que le transfert des données est bien clôturé.

11.1.3.7. Désélection du fichier.

La phase de désélection clôture le contexte opérationnel établi par la phase de sélection. La phase de désélection peut demander la suppression du fichier sélectionné.

11.1.3.8. Terminaison de connexion.

Cette phase libère la connexion entre l'utilisateur du service de F.T.A.M. et le gérant des données.

11.1.4. Opérations globales sur des fichiers.

11.1.4.1. Création.

Cette opération crée un nouveau fichier, et initialise les attributs (voir 11.1.6.) du fichier.

11.1.4.2. Sélection.

Cette opération crée un lien entre le requéreur et un fichier particulier. Cette opération est nécessaire pour toutes les opérations suivantes.

11.1.4.3. Lecture d'attributs.

Cette opération renvoie les valeurs des attributs demandés (voir 11.1.6.).

11.1.4.4. Modifications d'attributs.

Cette opération permet de modifier, d'ajouter ou de supprimer des attributs d'un fichier.

11.1.4.5. Ouverture d'un fichier.

Cette opération établit un régime permettant toutes les opérations sur un fichier donné (voir 11.1.5.). Elle opère sur le fichier sélectionné.

11.1.4.6. Fermeture d'un fichier.

Cette opération détruit le lien d'accès, de façon ordonnée, établi par l'opération d'ouverture d'un fichier.

11.1.4.7. Suppression d'un fichier.

Cette opération supprime un fichier, et le désélectionne. Elle clôture le régime de sélection de fichier.

11.1.4.8. Désélection d'un fichier.

Cette opération clôture le régime de sélection de fichier, en supprimant le lien d'accès entre le requéreur et le gérant des fichiers.

11.1.5. Opérations d'accès à un fichier.

11.1.5.1. Localisation.

Cette opération localise une unité de données dans le fichier. Lors de l'ouverture d'un fichier, le premier élément est localisé.

11.1.5.2. Lecture.

L'unité de données actuellement localisée est lue, et l'unité de données suivante est localisée.

11.1.5.3. Insertion.

Une nouvelle unité de données est crée, à la position appropriée du fichier (selon l'organisation du fichier).

11.1.5.4. Remplacement.

Le contenu d'une unité de données est remplacé par une nouvelle valeur. Cette opération peut être contrainte par certains attributs, telle la longueur de l'ancienne valeur.

11.1.5.5. Extension.

Les unités de données sont rajoutées à la fin du fichier.

11.1.5.6. Suppression.

L'unité de données est supprimée, et l'unité de données suivant celle supprimée est localisée.

11.1.6. Attributs d'un fichier.

Chaque fichier a un ensemble d'attributs globaux, n'ayant qu'une valeur ou qu'un jeu de valeurs. A un moment donné, tous les requéreurs d'accès à ce fichier verrons les mêmes valeurs, ou les mêmes jeux de valeurs.

a.- Nom du fichier.

Chaque fichier porte un nom. Le protocole de F.T.A.M. ne fait aucune vérification quand au nom du fichier.

b.- Types d'accès.

Cet attribut indique l'ensemble des opérations qui peuvent être effectuées sur le fichier. Il comporte deux valeurs :

- L'ordre d'accès : séquentiel ou aléatoire.

- La direction : entrée, sortie ou entrée/sortie.

c.- Contrôle d'accès.

L'attribut de contrôle d'accès est un ensemble d'attributs, chaque élément déterminant une condition sous laquelle l'accès aux éléments du fichier est autorisé. L'accès au fichier est autorisé si au moins une des conditions est satisfaite.

Jusqu'à quatre termes peuvent servir à la définition d'une condition. Chaque terme doit être satisfait pour que la condition soit remplie. Les termes des conditions sont :

- Le type d'accès autorisé, sous forme d'un vecteur booléen dont les éléments sont les opérations "lire", "insérer", "remplacer", "supprimer", "étendre", "lire les attributs", "changer les attributs" et "supprimer le fichier".

- Eventuellement une identification des utilisateurs autorisés au type d'accès.

- Eventuellement un mot de passe lié à l'identification de l'utilisateur autorisé au type d'accès.

- Eventuellement l'adresse de point d'accès de service autorisé à faire l'accès.

d.- Information comptable.

e.- Date et heure de la création.

f.- Date et heure de la dernière modification.

g.- Date et heure du dernier accès en lecture.

h.- Identification du créateur du fichier.

i.- Identification du dernier modificateur du fichier.

j.- Identification du dernier lecteur du fichier.

k.- Disponibilité du fichier.

Ce paramètre indique le délai à attendre avant l'autorisation d'ouverture du fichier.

l.- Contexte de présentation.

A ce sujet, voir les commentaires du chapitre 10.

m.- Nom d'encryptage.

n.- Type de structure d'accès.

Ce paramètre peut prendre comme valeurs "fichier non structuré", "fichier plat" ou "fichier arborescent". Ce paramètre est initialisé à la création du fichier, et n'est pas susceptible de modifications.

o.- Nom de la structure de présentation.

Ce paramètre indique le nom de la structure de présentation utilisée. Une valeur est donnée à cet attribut au moment de la création du fichier, et elle ne peut être modifiée.

p.- Taille actuelle du fichier.

Cet attribut est composé de deux entiers. Le premier donne une unité de mesure, le second le nombre d'unités de cette mesure contenu dans ce fichier.

q.- Taille future du fichier.

Cet attribut indique la taille maximum que pourra prendre le fichier. Sa représentation est identique à celle de la taille actuelle du fichier (voir ci-dessus)

r.- Qualification légale.

Ce paramètre contient les informations relatives au statut légal de ce fichier, et dépend de chaque organisation nationale.

11.2. SPECIFICATION DES INTERFACES.

Rappelons que l'ensemble des en-têtes de procédures et de fonctions décrits ci-dessous sont les seules parties visibles par l'utilisateur des services d'application. Les règles de dénomination des noms de ces procédures et fonctions sont celles reprises dans les conventions de services (voir 5.3.).

Comme pour les autres couches, nous trouverons deux paramètres communs à toutes les primitives :

- L'identificateur d'extrémité de connexion (F-IDT), servant à identifier la connexion application.

- Une variable d'erreur ERR, servant à signaler une erreur de protocole à l'utilisateur des services.

Lors de la création d'une connexion Application, l'entité qui fait la requête de création sera l'entité INITIATRICE de la connexion, l'autre étant l'entité RECEPTRICE. Toutes les primitives de REQUETE et de CONFIRMATIONS seront exécutées par l'entité initiatrice, les primitives d'INDICATION et de REPONSE seront exécutées par l'entité réceptrice. La seule exception sont les primitives concernant le transfert des données (voir 11.1.2.2., 11.2.11. et 11.2.12.), dont l'utilisation dépend de la direction du transfert.

11.2.1. Création de connexion.

```
TYPE TYP-SERVICE=(RELIABLE,USERCORRECTABLE);
     TYP-WINDOW=1..16;
     TYP-SUBSET=(FILE-TRANSFER,FILE-ACCESS,
                 LTD-FILE-MANAGEMENT,FILE-MANAGEMENT);


PROCEDURE F-CON-REQ
   (VAR F-IDT:TYPIDT;
    CALLING-SAP,CALLED-SAP:TYP-SAP;
    SERVICE:TYP-SERVICE;
    SUBSET:TYP-SUBSET;
    WINDOW:TYPWINDOW;
    IDENTITY:STRING;
    VAR ERR:INTEGER);
```

La création d'une connexion d'application est la première phase d'une activité de transfert de fichiers. L'entité qui exécute la procédure de requête de connexion (F-CON-REQ) devient l'entité initiatrice de connexion. Les différents paramètres de cette procédure sont :

- L'identificateur d'extrémité de connexion F-IDT, dont le fonctionnement est analogue à celui des autres couches.

- Les points d'accès de services appelé et appelant (CALLING-SAP et CALLED-SAP).

- Le type de service SERVICE (voir 11.1.2.3.)

- Les sous-ensembles de services désirés (SUBSET), et qui peuvent être :

  - Le service de transfert de fichier simple.

  - Le service de localisation d'éléments du fichier.

  - Le service de gestion limitée de fichiers.

  - Le service de gestion complète de fichiers.

- L'identification de l'initiateur de la connexion (IDENTITY).

- Le paramètre WINDOW sert à donner le nombre de fenêtres de transmission en cas d'utilisation du service de correction par l'utilisateur (voir 11.1.2.3.).

```
FUNCTION F-CON-IND
   (VAR F-IDT:TYP-IDT;
    VAR CALLING-SAP:TYP-SAP;
    VAR CALLED-SAP:TYP-SAP;
    VAR SERVICE:TYP-SERVICE;
    VAR SUBSET:TYP-SUBSET;
    VAR WINDOW:TYP-WINDOW;
    VAR IDENTITY:STRING;
    VAR ERR:INTEGER):BOOLEAN;
```

Cette fonction d'indication de création de connexion (F-CON-IND) prend la valeur "vrai" lorsqu'une entité reçoit une demande de connexion application. Les paramètres sont identiques à ceux de la requête.

```
PROCEDURE F-CON-RSP
    (F-IDT:TYP-IDT;
     RESP-SAP:TYP-SAP;
     SERVICE:TYP-SERVICE;
     SUBSET:TYP-SUBSET;
     WINDOW:TYP-WINDOW;
     DIAGNOSTIC:INTEGER;
```

```
    VAR ERR:INTEGER);
```

La procédure de réponse de création de connexion (F-CON-RSP) sert à l'entité appelée à accepter une connexion application. Tous les paramètres sont identiques à ceux de la requête, sauf l'adresse en réponse (RESP-SAP), qui peut être modifiée en cas de reroutage de la connexion, et le paramètre DIAGNOSTIC servant à signaler une anomalie (service non exécutable..), tout en acceptant la connexion.

```
FUNCTION F-CON-CNF
   (F-IDT:TYP-IDT;
    VAR RESP-SAP:TYP-SAP;
    VAR SERVICE:TYP-SERVICE;
    VAR SUBSET:TYPSUBSET;
    VAR WINDOW:TYPWINDOW;
    VAR DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER):BOOLEAN;
```

La fonction de confirmation de création de connexion (F-CON-CNF) est reçue par l'entité initiatrice de la connexion application, et sert à lui indiquer que la connexion est établie.

11.2.2.Terminaison de connexion.

La connexion se clôture par l'échange des primitives suivantes. C'est un service confirmé, qui ne nécéssite pas de commentaires. Seule l'entité initiatrice de la connexion a le droit d'émettre la requête de déconnexion.

```
PROCEDURE F-REL-REQ
   (F-IDT:TYP-IDT;
    VAR ERR:INTEGER);

FUNCTION F-REL-IND
   (F-IDT:TYP-IDT;
    VAR ERR:INTEGER):BOOLEAN;

PROCEDURE F-REL-RSP
   (F-IDT:TYP-IDT;
    VAR ERR:INTEGER);

FUNCTION F-REL-CNF
   (F-IDT:TYP-IDT;
    VAR ERR:INTEGER):BOOLEAN;
```

11.2.3. Coupure de connexion.

```
PROCEDURE F-AB-REQ
   (F-IDT:TYP-IDT;
    VAR ERR:INTEGER);
```

La procédure de requête de coupure de connexion (F-AB-REQ)

peut être émise par les deux entité à tout moment au cours de la connexion. Cette primitive détruit la connexion sans condition, et tout traitement en cours est abandonné.

```
TYPE TYP-ORIGINATOR=(USER-ORIGINATOR,
                     PROVIDER-ORIGINATOR);

FUNCTION F-AB-IND
   (F-IDT:TYP-IDT;
    VAR ORIGINATOR:TYP-ORIGINATOR;
    VAR ERR:INTEGER):BOOLEAN;
```

La fonction d'indication de coupure de connexion (F-AB-IND) sert à indiquer à l'utilisateur des services qu'une coupure a été faite. Le paramètre ORIGINATOR sert à indiquer si la coupure a été demandée par l'utilisateur des services ou a été faite par le fournisseur des services.

11.2.4. Sélection.

Le but de ces primitives est de sélectionner un fichier en spécifiant un nombre suffisant d'attributs pour l'identifier de façon non ambigüe.

```
PROCEDURE F-SEL-REQ
   (F-IDT:TYP-IDT;
    FILENAME:STRING;
    VAR ERR:INTEGER);
```

La requête de sélection (F-SEL-REQ) sert à demander la sélection d'un fichier, dont le nom se trouve dans FILENAME.

```
FUNCTION F-SEL-IND
   (F-IDT:TYP-IDT;
    VAR FILENAME:STRING;
    VAR ERR:INTEGER):BOOLEAN;
```

L'indication de sélection de fichier sert à signaler à l'entité réceptrice qu'un fichier doit être sélectionné.

```
PROCEDURE F-SEL-RSP
   (F-IDT:TYP-IDT;
    DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER);

FUNCTION F-SEL-CNF
    (F-IDT:TYP-IDT;
     VAR DIAGNSOTIC0:INTEGER;
     VAR ERR:INTEGER):BOOLEAN;
```

La réponse de sélection (F-SEL-RSP) émise par l'entité réceptrice, et la confirmation (F-SEL-CNF) reçue par l'entité

initiatrice de la connexion portent un paramètre DIAGNOSTIC dont la valeur différente de zéro indique que le fichier n'a pu être sélectionné.

11.2.5. Désélection.

```
PROCEDURE F-DES-REQ
   (F-IDT:TYP-IDT;
    VAR ERR:INTEGER);

FUNCTION F-DES-IND
   (F-IDT:TYP-IDT;
    VAR ERR:INTEGER):BOOLEAN;

PROCEDURE F-DES-RSP
   (F-IDT:TYP-IDT;
    DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER);

FUNCTION F-DES-CNF
   (F-IDT:TYP-IDT;
    VAR DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER):BOOLEAN;
```

La requête de désélection (F-DES-REQ) n'est émise que par l'entité initiatrice de la connexion, et ne peut l'être que si un fichier a été sélectionné.

Cette primitive, tout comme d'indication (F-DES-IND), ne comporte aucun paramètre.

La réponse de désélection (F-DES-RSP), émise par l'entité réceptrice, et la confirmation (F-DES-CNF) reçue par l'entité émettrice comportent le paramètre DIAGNOSTIC, prenant une valeur non nulle si la désélection n'a pu être effectuée.

11.2.6. Création de fichier.

```
PROCEDURE F-CRE-REQ
   (F-IDT:TYP-IDT;
    FILENAME:STRING;
    ATTRIBUT:TYP-ATTRIBUT;
    VAR ERR:INTEGER);
```

La procédure de création de fichier (F-CRE-REQ) sert à entrer dans la phase de sélection d'un fichier inexistant. Les différents paramètres sont le nom du fichier (FILENAME) ainsi que ces attributs ATTRIBUT (voir 11.1.6.). Cette requête ne peut être émise que par l'entité initiatrice de la connexion.

```
FUNCTION F-CRE-IND
   (F-IDT:TYP-IDT;
    VAR FILENAME:STRING;
```

```
    VAR ATTRIBUT:TYP-ATTRIBUT;
    VAR ERR:INTEGER):BOOLEAN;
```

La fonction d'indication de création de fichier (F-CRE-IND) sert à indiquer à l'entité réceptrice qu'un fichier doit être créé et sélectionné.

```
PROCEDURE F-CRE-RSP
   (F-IDT:TYP-IDT;
    DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER);

FUNCTION F-CRE-CNF
   (F-IDT:TYP-IDT;
    VAR DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER):BOOLEAN;
```

La procédure de réponse (F-CRE-RSP) et la fonction de confirmation (F-CRE-CNF) servent à parachever ce service de création. Le paramètre DIAGNOSTIC prend une valeur non nulle si la création n'a pu avoir lieu.

11.2.7. Suppression de fichier.

Les primitives de suppression de fichier servent à désélectionner un fichier de telle façon qu'il ne puisse être resélectionné par la suite.

```
PROCEDURE F-DEL-REQ
   (F-IDT:TYP-IDT;
    VAR ERR:INTEGER);

FUNCTION F-DEL-IND
   (F-IDT:TYP-IDT;
    VAR ERR:INTEGER):BOOLEAN;

PROCEDURE F-DEL-RSP
   (F-IDT:TYP-IDT;
    DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER);

FUNCTION F-DEL-CNF
   (F-IDT:TYP-IDT;
    VAR DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER):BOOLEAN;
```

Le paramètre DIAGNOSTIC prend une valeur différente de zéro si la suppression n'a pu avoir lieu.

11.2.8. Lecture des attributs.

```
PROCEDURE F-REA-REQ
   (F-IDT:TYP-IDT;
```

```
    ATTR-NAMES:TYP-ATTR-NAMES;
    VAR ERR:INTEGER);

FUNCTION F-REA-IND
    (F-IDT:TYP-IDT;
     VAR ATTR-NAMES:TYP-ATTR-NAMES;
     VAR ERR:INTEGER):BOOLEAN;
```

La requête de lecture des attributs (F-REA-REQ) porte comme paramètre la liste des attributs à lire (voir 11.1.6.). L'indication (F-REA-IND) comporte ce même paramètre.

```
PROCEDURE F-REA-RSP
   (F-IDT:TYP-IDT;
    ATTRIBUTE:TYP-ATTRIBUTE;
    DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER);

FUNCTION F-REA-CNF
   (F-IDT:TYP-IDT;
    VAR ATTRIBUTE:TYP-ATTRIBUTE;
    VAR DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER):BOOLEAN;
```

La réponse de lecture d'attributs (F-REA-RSP) émise par l'entité réceptrice, tout comme la confirmation (F-REA-CNF), comprend comme paramètre la liste des attributs demandés. La fonction diagnostic prend une valeur non nulle si la fonction n'a pu être exécutée correctemnent.

11.2.9. Changement d'attributs.

Ce service confirmé sert à changer les attributs d'un fichier. Les nouvelles valeurs des attributs (voir 11.1.6.) sont contenues dans le paramètre ATTRIBUTE dans la requête (F-CHA-REQ) et l'indication (F-CHA-IND). Le paramètre DIAGNOSTIC de la réponse (F-CHA-RSP) et de la confirmation (F-CHA-CNF) prend une valeur non nulle si l'opération n'a pu s'effectuer correctement.

```
PROCEDURE F-CHA-REQ
   (F-IDT:TYP-IDT;
    ATTRIBUTE:TYP-ATTRIBUTE;
    VAR ERR:INTEGER);

FUNCTION F-CHA-IND
   (F-IDT:TYP-IDT;
    VAR ATTRIBUTE:TYP-ATTRIBUTE;
    VAR ERR:INTEGER):BOOLEAN;

PROCEDURE F-CHA-RSP
   (F-IDT:TYP-IDT;
    DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER);
```

```
FUNCTION F-CHA-CNF
   (F-IDT:TYP-IDT;
    VAR DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER):BOOLEAN;
```

11.2.10. Ouverture d'un fichier.

Ce service sert à entrer dans la phase de transfert de données (fig 11-1).

```
TYPE TYP-PROCESSING=(F-READ,F-WRITE);

PROCEDURE F-OPN-REQ
   (F-IDT:TYP-IDT;
    PROCESSING:TYP-PROCESSING;
    ACTIVITY:INTEGER;
    VAR ERR:INTEGER);

FUNCTION F-OPN-IND
   (F-IDT:TYP-IDT;
    VAR PROCESSING:TYP-PROCESSING;
    VAR ACTIVITY:INTEGER;
    VAR ERR:INTEGER):BOOLEAN;
```

La fonction de requête d'ouverture d'un fichier (F-OPN-REQ) ne peut avoir lieu qu'après avoir sélectionné ce fichier. Le paramètre PROCESSING est destiné à préciser le type d'ouverture, en lecture ou en écriture.

Le paramètre ACTIVITY sert à identifier l'activité en cours sur le fichier, et sera utilisé en cas de reprise sur erreur (voir 11.2.22.). Ce paramètre N'A RIEN A VOIR avec le concept d'activité de la couche session.

```
PROCEDURE F-OPN-RSP
   (F-IDT:TYP-IDT;
    DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER);

FUNCTION F-OPN-RSP
   (F-IDT:TYP-IDT;
    VAR DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER):BOOLEAN;
```

La procédure de réponse d'ouverture (F-OPN-RSP) porte le paramètre DIAGNOSTIC, qui prend une valeur différente de zéro si l'ouverture n'a pu s'effectuer correctement. Cette valeur est transmise dans la confirmation (F-OPN-CNF).

11.2.11. Fermeture de fichier.

```
PROCEDURE F-CLO-REQ
   (F-IDT:TYP-IDT;
    VAR ERR:INTEGER);

FUNCTION F-CLO-IND
   (F-IDT:TYP-IDT;
    VAR ERR:INTEGER):BOOLEAN;

PROCEDURE F-CLO-RSP
   (F-IDT:TYP-IDT;
    DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER);

FUNCTION F-CLO-CNF
   (F-IDT:TYP-IDT;
    VAR DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER):BOOLEAN;
```

La procédure de requête de fermeture de fichier (F-CLO-REQ) et son indication à l'entité réceptrice (F-CLO-IND) servent à sortir de la phase de transfert de données (fig 11-1).

La réponse de fermeture (F-CLO-RSP) transporte le paramètre DIAGNOSTIC, qui prend une valeur non nulle en cas d'impossibilité de fermeture, et reporté dans la confirmation de fermeture de fichier (F-CLO-CNF).

11.2.12. Transfert de données.

```
PROCEDURE F-DAT-REQ
   (F-IDT:TYP-IDT;
    DATA-TYP:TYP-DATA-TYP;
    DATA-VAL:TYP-DATA-VAL;
    VAR ERR:INTEGER);

FUNCTION F-DAT-IND
   (F-IDT:TYP-IDT;
    VAR DATA-TYP:TYP-DATA-TYP;
    VAR DATA-VAL:TYP-DATA-VAL;
    VAR ERR:INTEGER):BOOLEAN;
```

La procédure de requête de transfert de données (F-DAT-REQ), et l'indication à l'entité communicante (F-DAT-IND), servent à transférer des données d'une entité à l'autre. Ces données sont définies par leur type (DATA-TYP) et leur valeur (DATA-VAL).

Ce service ne peut être effectué qu'aprés avoir émis une requête de lecture ou d'écriture (voir 11.2.15. et 11.2.16.), et en avoir reçu confirmation.

11.2.13. Fin de données.

```
PROCEDURE F-DAE-REQ
```

```
   (F-IDT:TYP-IDT;
    DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER);

FUNCTION F-DAE-IND
   (F-IDT:TYP-IDT;
    VAR DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER):BOOLEAN;
```

Ce service sert à indiquer la fin d'unités de données, et est utilisé après avoir émis une ou plusieurs requêtes de transfert de données (voir 11.2.12.). Le paramètre DIAGNOSTIC sert à signaler si le transfert a pu se clôturer normalement.

11.2.14. Localisation d'unité de données.

```
TYPE TYP-LOCATE=(FIRST,LAST,
                 CURRENT,NEXT,PREVIOUS);

PROCEDURE F-LOC-REQ
   (F-IDT:TYP-IDT;
    LOCATE:TYP-LOCATE;
    VAR ERR:INTEGER);

FUNCTION F-LOC-IND
   (F-IDT:TYP-IDT;
    VAR LOCATE:TYP-LOCATE;
    VAR ERR:INTEGER):BOOLEAN;
```

La procédure de requête de localisation d'une unité de données (F-LOC-REQ) et l'indication à l'autre entité (F-LOC-IND) servent à identifiier l'unité de données qui sera la prochaine sur laquelle le transfert de données aura lieu.

Le paramètre LOCATE peut prendre les valeurs premier (FIRST) ou dernier (LAST) par rapport au fichier, ou des valeurs relatives à la dernière unité de données accédée (CURRENT, PREVIOUS ou NEXT).

Le modèle précise en outre une façon d'accéder aux unités de données par niveau si le fichier est structuré en arbre. Nous n'avons pas développé cette méthode, mais en cas de réalisation, il faudra seulement changer le type TYP-LOCATE afin de pouvoir demander ce genre de localisation.

```
PROCEDURE F-LOC-RSP
   (F-IDT:TYP-IDT;
    DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER);

FUNCTION F-LOC-CNF
   (F-IDT:TYP-IDT;
    VAR DIAGNOSTIC:INTEGER;
```

```
    VAR ERR:INTEGER):BOOLEAN;
```

La procédure de réponse de localisation (F-LOC-RSP) et la fonction de confirmation (F-LOC-CNF) portent comme paramètre le DIAGNOSTIC prenant une valeur différente de zéro si la localisation n'a pu être réalisée.

11.2.15. Ecriture sur un fichier.

Ce service précède un transfert de données (voir 11.2.12.) et précise le sens du transfert jusqu'à la fin de transfert de données (voir 11.2.13.). Le fichier doit préalablement avoir été sélectionné et ouvert.

```
TYPE TYP-OPERATION=(REPLACE,INSERT,EXTEND);

PROCEDURE F-WRT-REQ
   (F-IDT:TYP-IDT;
    OPERATION:TYP-OPERATION;
    LOCATE:TYP-LOCATE;
    VAR ERR:INTEGER);

FUNCTION F-WRT-IND
   (F-IDT:TYP-IDT;
    VAR OPERATION:TYP-OPERATION;
    VAR LOCATE:TYP-LOCATE;
    VAR ERR:INTEGER):BOOLEAN;
```

Le type d'opération (OPERATION) représente le genre d'écriture à effectuer : remplacement (REPLACE), insertion (INSERT) ou écriture en fin de fichier (EXTEND). La localisation (LOCATE) est identique à celle de la localisation d'une unité de données (voir 11.2.14).

```
PROCEDURE F-WRT-RSP
   (F-IDT:TYP-IDT;
    DIAGNOSTIC:INTEGER;
    RECOVERY:SIX-BYTES;
    VAR ERR:INTEGER);

FUNCTION F-WRT-CNF
   (F-IDT:TYP-IDT;
    VAR DIAGNOSTIC:INTEGER;
    VAR RECOVERY:SIX-BYTES;
    VAR ERR:INTEGER):BOOLEAN;
```

La réponse et la confirmation portent un paramètre DIAGNOSTIC indiquant si l'écriture est possible ou pas.

Le paramètre RECOVERY identifie un point de contrôle où une reprise est autorisée. Sa valeur est comprise entre 1 et 999 998, et est codée en ASCII.

11.2.16. Lecture d'unité de données.

La requête de lecture (F-REA-REQ) est émise par l'entité initiatrice de connexion pour demander le transfert de données de l'entité réceptrice. C'est un des seuls cas où l'entité initiatrice prend un rôle passif et attend des indications de transfert de données (voir 11.2.12.).

```
PROCEDURE F-REA-REQ
   (F-IDT:TYP-IDT;
    LOCATE:TYP-LOCATE;
    RECOVERY:SIX-BYTES;
    VAR ERR:INTEGER);

FUNCTION F-REA-IND
   (F-IDT:TYP-IDT;
    VAR LOCATE:TYP-LOCATE;
    VAR RECOVERY:SIX-BYTES;
    VAR ERR:INTEGER):BOOLEAN;

PROCEDURE F-REA-RSP
   (F-IDT:TYP-IDT;
    DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER);

FUNCTION F-REA-CNF
   (F-IDT:TYP-IDT;
    VAR DIAGNSOTIC:INTEGER;
    VAR ERR:INTEGER):BOOLEAN;
```

Les différents paramètres sont identiques à ceux expliqués plus haut.

11.2.17. Suppression d'unité de données.

Ce groupe de primitive sert à supprimer une unité de données du fichier. Les différents paramètres sont identiques à ceux vus plus haut.

```
PROCEDURE F-ERA-REQ
   (F-IDT:TYP-IDT;
    LOCATE:TYP-LOCATE;
    VAR ERR:INTEGER);

FUNCTION F-ERA-IND
   (F-IDT:TYP-IDT;
    VAR LOCATE:TYP-LOCATE;
    VAR ERR:INTEGER):BOOLEAN;

PROCEDURE F-ERA-RSP
   (F-IDT:TYP-IDT;
    DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER);
```

```
FUNCTION F-ERA-CNF
   (F-IDT:TYP-IDT;
    VAR DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER):BOOLEAN;
```

11.2.18. Fin de transfert.

La primitive de fin -de transfert s'effectue après confirmation d'une fin de données (voir 11.2.13.)

```
PROCEDURE F-TRE-REQ
   (F-IDT:TYP-IDT;
    VAR ERR:INTEGER);

FUNCTION F-TRE-IND
   (F-IDT:TYP-IDT;
    VAR ERR:INTEGER):BOOLEAN;

PROCEDURE F-TRE-RSP
   (F-IDT:TYP-IDT;
    DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER);

FUNCTION F-TRE-CNF
   (F-IDT:TYP-IDT;
    VAR DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER):BOOLEAN;
```

Le paramètre DIAGNOSTIC a le même usage que dans les autres primitives décrites plus haut.

11.2.19. Annulation du transfert de données.

Les deux utilisateurs des services peuvent émettre une requête d'annulation du transfert de données (F-CAN-REQ), après avoir émis une réponse de lecture ou d'écriture, ou en avoir reçu confirmation, et avant l'émisssion (la réception) d'une réponse (confirmation) de fin de données. Après utilisation de l'annulation, toute indication ou confirmation non encore délivrée est supprimée. Le fichier reste cependant ouvert.

```
PROCEDURE F-CAN-REQ
   (F-IDT:TYP-IDT;
    DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER);

FUNCTION F-CAN-IND
   (F-IDT:TYP-IDT;
    VAR DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER):BOOLEAN;

PROCEDURE F-CAN-RSP
```

```
   (F-IDT:TYP-IDT;
    DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER);

FUNCTION F-CAN-CNF
   (F-IDT:TYP-IDT;
    VAR DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER):BOOLEAN;
```

Le paramètre DIAGNOSTIC indique dans la requête et l'indication la cause de l'annulation. Dans la réponse et la confirmation, il indique si une reprise sur erreur est possible ou non.

11.2.20. Insertion de points de contrôle.

L'insertion de points de contrôle est effectuée pendant le transfert de données. L'utilisation de ces primitives permet à l'entité émettrice de données de savoir qu'aucun redémarrage (voir 11.2.21.) n'aura lieu à partir des données émises avant le point de contrôle. L'émetteur peut continuer à émettre des données avant d'avoir reçu confirmation du point de contrôle. Le nombre de points de contrôle pouvant rester non confirmés est défini dans la requête de connexion (voir 11.2.1.). Tous les points de contrôle doivent être confirmés avant la requête de fin de transfert (voir 11.2.18.)

```
PROCEDURE F-CHK-REQ
   (F-IDT:TYP-IDT;
    CHECKPOINT:SIX-BYTES;
    VAR ERR:INTEGER);

FUNCTION F-CHK-IND
    (F-IDT;TYP-IDT;
     VAR CHECKPOINT:SIX-BYTES;
     VAR ERR:INTEGER):BOOLEAN;

PROCEDURE F-CHK-RSP
    (F-IDT:TYP-IDT;
     CHECKPOINT:SIX-BYTES;
     VAR ERR:INTEGER);

FUNCTION F-CHK-CNF
    (F-IDT:TYP-IDT;
     VAR CHECKPOINT:SIX-BYTES;
     VAR ERR:INTEGER):BOOLEAN;
```

Le paramètre CHECKPOINT sert à identifier le point de contrôle, et peut prendre une valeur comprise entre 1 et 999 998, codée en ASCII.

11.2.21. Redémarrage.

Le redémarrage interrompt un transfert de données, avec perte possible de données non encore délivrées, et permet de négocier un point à partir duquel le transfert doit recommencer.

Une indication de redémarrage est refusée par une requête d'annulation (voir 11.2.19).

```
PROCEDURE F-RST-REQ
   (F-IDT:TYP-IDT;
    CHECKPOINT:SIX-BYTES;
    VAR ERR:INTEGER);

FUNCTION F-RST-IND
   (F-IDT:TYP-IDT;
    VAR CHECKPOINT:SIX-BYTES;
    VAR ERR:INTEGER):BOOLEAN;

PROCEDURE F-RST-RSP
   (F-IDT:TYP-IDT;
    CHECKPOINT:SIX-BYTES;
    VAR ERR:INTEGER);

FUNCTION F-RST-CNF
   (F-IDT:TYP-IDT;
    VAR CHECKPOINT:SIX-BYTES;
    VAR ERR:INTEGER):BOOLEAN;
```

Le codage du paramètre CHECKPOINT est identique à celui des primitives vues plus haut.

11.2.22. Recouvrement de régime.

Le recouvrement de .régime sert à recréer un régime ouvert après une panne. Il permet à l'entité réceptrice de redémarrer des activités en faisant référence à un identificateur d'activité préalablement établi (voir 11.2.10.).

```
PROCEDURE F-RCV-REQ
   (F-IDT:TYP-IDT;
    ACTIVITY:INTEGER;
    VAR ERR:INTEGER);

FUNCTION F-RCV-IND
   (F-IDT:TYP-IDT;
    VAR ACTIVITY:INTEGER;
    VAR ERR:INTEGER):BOOLEAN;

PROCEDURE F-RCV-RSP
   (F-IDT:TYP-IDT;
    DIAGNOSTIC:INTEGER;
    VAR ERR:INTEGER);

FUNCTION F-RCV-CNF
   (F-IDT:TYP-IDT;
```

```
VAR DIAGNOSTIC:INTEGER;
VAR ERR:INTEGER):BOOLEAN;
```

Le paramètre DIAGNOSTIC de la réponse (F-RCV-RSP) et de la confirmation (F-RCV-CNF) signale si le recouvrement a pu être effectué ou non.

11.3. DESCRIPTION DU PROTOCOLE.

Le protocole du F.T.A.M. est assez simple, et consiste uniquement en la vérification de la validité de l'invocation des primitives.

Pour réaliser cela, on retrouve dans les variables locales deux listes d'actions autorisées :

- Une liste des primitives de requête et de réponses autorisées par l'utilisateur des services, et

- Une liste des F.P.D.U. susceptibles d'être reçues de la couche présentation.

La liste des actions et des mises à jour de ces liste est reprise en annexe.

11.4. EXTENSIONS FUTURES.

Les primitives de cette couche ont toutes été réalisées. Il est de la sorte possible de concevoir un gérant de base de données, utilisant ces primitives pour effectuer la transformation entre le modèle virtuel de fichier tel défini par O.S.I. et les mises en oeuvre proposées sur les différents systèmes existants sur le marché.

12. INTEGRATION DE LA REALISATION.

Cette réalisation comprend la partie "centrale" de l'architecture des systèmes ouverts. Dans le cas où il faudrait intéger cette réalisation dans un réseau complet, commercial, il faudrait réaliser les applications pouvant fonctionner sur le réseau, et fournir les moyens physiques et procéduraux afin d'interconnecter des systèmes différents entre eux.

12.1. REALISATION D'UN GERANT DE BASE DE DONNEES.

Toutes les primitives nécessaires à la mise en oeuvre d'un gérant de base de donnees fonctionnant sur le modele du F.T.A.M. sont fournies dans les services d'application.

Sur base des renseignements des normes, il serait intéressant de réaliser ce gérant, permettant l'accès à des fichiers structurés en arbre, et gérant les problèmes de collisions, d'interblocage, etc...

12.2. REALISATION DES MOYENS PHYSIQUES D'INTERCONNEXION.

En gardant les interfaces réseau, il suffirait de changer son protocole (réduit à sa plus simple expression dans notre réalisation), afin de permettre l'échange physique de données entre postes.

Dans le cadre d'un réseau local, la couche réseau peut n'avoir qu'un protocole très restreint, et cela d'autant plus que le contrôle de flux est effectué par la couche transport.

Il suffirait d'avoir quatre types d'unités de données de protocole réseau : demande de connexion, réponse de connexion, transfert de données et déconnexion. Un petit mécanisme d'adressage supplémentaire aurait pour effet de déterminer les points d'accès aux services de réseau.

La couche liaison de données fournirait les primitives décrites pour le réseau ETHERNET (voir 3.2.).

12.3. AMELIORATIONS DE LA REALISATION.

Une amélioration importante serait de réaliser la classe 4 dans la couche transport, qui permet de détecter des anomalies telles l'extinction non signalée d'une des entités communicantes ou la réémission d'une unité de données perdue.

La mise en oeuvre de temporisateurs se révèlerait aussi d'une grande utilité. Nous verrions assez bien une fonction de temporisateur au niveau du système d'exploitation, utilisée par toutes les couches.

Note : Les appels au temporisateur sont déja effectués dans certaines

couches. Dans la couche transport, par contre, les temporisateurs ne sont pas mis en oeuvre car ils devraient faire réagir l'entité au moment de leur expiration, alors que dans les autres couches, ce sont des fonctions booléennes testées par les entités.

Comme architecture globale, l'idéal nous semblerait d'avoir un processeur multi-tâche, où un processus serait affecté aux couches transport, réseau, liaison de données et physique, alors que les autres processus seraient affectés à une ou plusieurs applications. Le processus relatif au transport serait activé à toute réception de message provenant du réseau, et par les primitives des entités session.

13. CONCLUSIONS.

La première partie de ce travail portait sur l'étude des réseaux locaux existants sur le marché. Il nous est apparu que l'effort actuel portait sur des réseaux de mini-ordinateurs, et qu'il n'y avait encore que peu de L.A.N. bon-marché destinés aux micro-ordinateurs.

D'autre part, les différents contacts que nous avons eus nous ont apportés la conviction qu'il était possible de réaliser les interfaces physiques pour un réseau local. Nous avons réalisé des interfaces pour les APPLE-II travaillant à 64 Kbps.

Au niveau des protocoles de plus haut niveau, nous avons fait l'étude des normes de l'interconnexion des systèmes ouverts (O.S.I.) proposées par l'organisation internationale de normalisation (I.S.O.). Cette étude nous apparaissait d'autant plus judicieuse que les normes en arrivent tout doucement à leur état de maturation. Il n'est pas certain que les constructeurs leur accordent l'accueil souhaité, mais de vastes projets européens (ESPRIT) s'y attachent particulièrement.

Nous avons réalisé une mise en oeuvre de ces normes, sur un micro-ordinateur, simulant un réseau réèl. Une application-type a été faite (le F.T.A.M. ou gestion d'accès à des fichiers). L'ensemble des procédures est assez lourd, seulement le principe de négociation de sous-ensembles de services nous permet de faire fonctionner les éléments nécessaires au F.T.A.M. sans trop de dégradation des performances.

Nous n'avons pas étudié les autres applications-types (gestion des processus éloignés et terminal virtuel), estimant que ceux-ci ne présentaient guère d'intérêt dans le cadre d'un L.A.N. de micro-ordinateurs.

Ce travail s'est inscrit comme une phase nécessaire dans la conception d'un réseau local. Beaucoup de points propres à la réalisation y ont été conçus. La mise en oeuvre globale ne dépend plus de facteurs propres à l'O.S.I., mais plutôt à des contraintes :

- matérielles pour ce qui concerne son intégration dans une interconnexion physique, et

- liées à des organisation internes de fichiers, pour l'application du F.T.A.M.

Afin de disposer d'un réseau, nous estimons que ces deux points sont des sujets de réalisations intéressant, pouvant être poursuivis dans des travaux semblables à celui-ci. De la sorte, il sera possible de disposer d'un réseau sur lequel pourront se greffer des nouveaux systèmes, à condition que ceux-ci respectent la normalisation de l'O.S.I.

LISTE DES MOTS-CLES ET ABBREVIATIONS.

BIBLIOGRAPHIE.

(TAN)

"Computer Networks"
A.S. TANENBAUM
Prentice hall, 1981.

(ZAKS)

"Technique d'Interfacage aux micro-processeurs"
Rodnay ZAKS
Editions Sybex 3ème édition 1981

(XEROX)

"The ETHERNET, Data link layer and physical layer"
Digital, Intel, Xerox
Version 1.0 30 septembre 1980

(INTEL1)

"The complete VLSI L.A.N. Solution"
Intel corporation, 1982
Order number : 210783-001

(CORVUS)

"Omninet Low Cost High Performance Local Network"
Corvus Systems
publicité 1983

(ZILOG)

"Z-net Local Area Network System"
Concepts and facilities
Zilog, mai 1981

(LILEN)

"Interfaces pour micro-processeurs et micro-ordinateurs"
H.LILEN
Editions Radio, 1983

(XEPHON)

"Local Area Network"
Xephon Buyer's Guide 1983

(STUDNITZ)

"Transport Protocols : Their Performance"
Peter von STUDNITZ
Computer Networks 7 (1983), 27-35

(METCALFE)

"ETHERNET : Distribued packed switching"
R. METCALFE, D. BOGGS
Comm. of A.C.M. 19 (1976), 395-404

(SHOCH)

"Measured Performance of an Ethernet Local Network"
J.F. SHOCH, J.A. HUPP
Comm. of A.C.M. 23,12 (1980), 711-721

(ROWLANDS)

"Local Area Networks"
M.G. ROWLANDS
British Telecommunications Engineering 2 (1983) 6-10

(INTEL2)

"INA 950-1"
Intel Corporation, 1982
Order number : 210810-001

(INTEL3)

"ISBC 186/51 Communication Computer"
Intel Corporation, 1983
Order number : 230718-001

(INTEL4)

"ISBC 550 ETHERNET Communication Controller"
Intel Corporation, 1981
Order number : 143882

(GIEN)

"A File Transfer Protocol (F.T.P.)"
M. GIEN
Computer Networks 2 (1978) 312-319

(MEYER)

"Etude comparative d'architectures de réseau par rapport au modèle de référence O.S.I."
MEYER J.F.
FUNDP-Namur 1982 (Travail de fin d'Ntudes).

(LEMAL)

"Contribution à la mise en oeuvre d'un réseau local"
E. LEMAL
FUNDP-Namur 1982 (Travail de fin d'études).

(YOUNG)

"A Standard Session Protocol for O.S.I."
C.E. YOUNG
National Computer Conference 1983 619-622

(SPANIOL)

"Modelling of Local Computers"
O. SPANIOL
Computers Networks 3 (1979) 315-326

(GRANT)

"X25 Protocols and Local Area Networks"
A. GRANT, D. HUTCHISON
Computer Networks 6 (1982) 255-262

(COTTON)

"Technologies for Local Area Computer Network"
I.W. COTTON
Computer Networks 4 (1980), 197-208

(CCITT1)

"Avis X25"
C.C.I.T.T.
Fascicule VIII.2

(CCITT2)

C.C.I.T.T.
Recommendation S.62 fascicle VII.2

(ISO1)

"Modèle de référence de base de l'interconnexion des systèmes ouverts"
I.S.O./DIS 7498

(ISO2)

"A formal description technique based on an extended state transition model"
I.S.O./TC 97/SC 16 N1825

(ISO3)

"Specification of protocols for the commitment, concurrency, and recovery, common application service elements"

- "part 1 : general"
- "part 2 : the C.C.R. protocols"
- "part 3 : C.C.R. test procedures"

I.S.O./TC 97/SC 16 N1714

(ISO4)

"Draft definition of the common application service elements"
I.S.O./TC 97/SC 16 N1662

(ISO5)

"Draft notation for the definition of O.S.I. datatypes and values"
I.S.O./TC 97/SC 16 N1663

(ISO6)

"file transfert acces and management"

- "part 1 : general description"
  I.S.O./dp 8571/1
- "part 2 : the virtual filestore"
  I.S.O. dp 8571/2
- "part 3 : the file service definition"
  I.S.O. dp 8571/3
- "part 4 : the file protocol specification"

I.S.O. dp 8571/4

(ISO7)

"Specification of Abstract Syntax Notation One (A.S.N.1)
I.S.O./TC 97/SC 16 N1795

(ISO8)

"Presentation service definition"
I.S.O./TC 97/SC 16 N1666

(ISO9)

I.S.O./TC 97/SC 16 N1667

(ISO10)

"Définition du service session de base en mode connexion"
I.S.O./DIS 8326

(ISO11)

"Spécification du protocole session de bas en mode connexion"
I.S.O./DIS 8327

(ISO12)

"Addendum coverint Session symmetric synchronization for the session service"
I.S.O./TC 97/SC 16 N1687

(ISO13)

"Addendum covering Session symmetric synchronization for the session protocol"
I.S.O./TC 97/SC 16 N1688

(ISO14)

"Définition du service transport"
I.S.O./DIS 8072

(ISO15)

"Définition du protocole de transport orienté connexion"

I.S.O./DIS 8073

(ISO16)

"Addendum to the transport service definition covering connectionless-mode transmission"
I.S.O./TC 97/SC 16 N1703

(ISO17)

"Protocol for providing the connectionless-mode transport service"
I.S.O./TC 97/SC 16 N1795

(ISO18)

"Changes to enable class 4 operation over connectionless-mode network service"
I.S.O./TC 97/SC 16 N1706

(ISO19)

"Network service definition"
I.S.O./TC 97/SC 16 N1740

(APPLE1)

"Apple PASCAL, Language reference manual"
Apple Product FrA2L0027

(APPLE2)

"Apple PASCAL, Operating System manual"
Apple Product FrA2L0028

(CHAMORET1)

"Le système PASCAL UCSD 1/ Organisation générale"
T. CHAMORET
Edi-test 1983

(CHAMORET2)

"Le système PASCAL UCSD 2/Structure interne"
T. CHAMORET
Edi-test 1983

Facultés Universitaires Notre-Dame de la Paix - Namur

INSTITUT D'INFORMATIQUE

CONTRIBUTION

A UNE REALISATION

D'INTERCONNEXION

DE SYSTEMES OUVERTS.

ANNEXES

Mémoire présenté par

Didier de GHELLINCK

en vue de l'obtention du titre

de licencié et maître en informatique.

Année accadémique 1983-1984

P L A N D E S A N N E X E S

ANNEXE I

DESCRIPTION

DES TABLES D'ETATS

DE LA COUCHE TRANSPORT

I.1. VUE GENERALE.

Cette partie des annexes est consacrée à la description des protocoles de la classe 2 de la couche transport.

La couche transport a été réalisée de façon à réagir instantanément à tout évènement susceptible d'engendrer une série d'actions de sa part. Aussi trouvons nous deux types d'évènements: d'une part les requêtes et les réponses provennant de l'utilisateur des services de transport (à travers les services de la couche session), et d'autre part les indications et les confirmations provenant de la couche réseau. A chacun de ces évènements correspond un ensemble de procédures, vérifiant la validité de l'évènement, ainsi que des paramètres accompagnant cet évènement. Ensuite, si l'entité doit provoquer une indication ou une confirmation à l'utilisateur des services, elle mémorisera ce fait. Si par contre des T.P.D.U. doivent être émises en retour, l'entité provoquera immédiatement les requêtes ou les réponses appropriées à la couche réseau.

Les tables d'états décrites ci-dessous montrent les différents traitements à effectuer. En lignes, nous trouvons les différents évènements modifiant l'état de l'entité, en colonnes les différents états que peut prendre cette entité.

Nous verrons successivement la liste des états possibles de l'entité, puis la description des évènements entrants et des évènements sortants, ensuite le contenu des T.P.D.U. envoyées d'une entité à l'autre par des requêtes de transfert de données de réseau, puis suivra le tableau des différents états, et la liste des actions résultants du couple état/évènement entrant.

La réalisation de ce protocole a été faite de la façon la plus proche possible des descriptions des normes O.S.I., afin de simplifier la maintenance et l'amélioration de ces programmes. A chaque appel de primitive, le programme teste l'état de l'entité (CASE OF), et effectue le traitement selon ce dernier.

I.2. LISTE DES ETATS.

Dans cette partie, nous décrivons tous les états que peut prendre une entié transport. On retrouve ces états dans les colonnes de la table d'états ci-dessous.

CLOSED

Dans cet état, aucune connexion transport n'existe. De la part de l'utilisateur des services, seule une requête de connexion peut être émise, engendrant ainsi une création de connexion de transport (si les ressources nécessaires sont disponibles).

De la couche réseau, toute indication autre que l'indication de connexion ou l'indication de données engendrera une déconnexion de réseau.

WFNC (Wait For Network Confirmation)

L'entité parvient dans cet état quand elle a fait une requête de connexion réseau, et en attend la confirmation.

WFCC (Wait For Connection Confirm)

Après avoir émis une unité de données de protocole de demande de connexion de transport, l'entité se met dans cet état d'attente de confirmation de connexion de transport, au moyen de la T.P.D.U. CC (voir plus bas).

WFTRESP (Wait For Transport RESPonse)

Cet état est atteint après que l'entité ait fourni une indication de connexion de transport à l'utilisateur des services, et qu'elle attend une réponse de connexion (TCONRSP).

OPEN

Dans cet état, les échanges de données sont autorisés aux deux entités communicantes.

WBCL (Wait Before Closing)

Cet état est atteint lorsque un utilisateur des services a fait une requête de connexion, et sans avoir reçu la réponse, émet une requête de déconnexion. L'entité dans cet état attend d'abord une T.P.D.U. de confirmation de connexion (ou la T.P.D.U. de demande de déconnexion), avant d'émettre soit une T.P.D.U. de requête de déconnexion, soit une T.P.D.U. de confirmation de déconnexion.

CLOSING

Après qu'une T.P.D.U. de requête de déconnexion ait été émise, cet état indique que l'entité attend une T.P.D.U. de confirmation de déconnexion. Plus aucun échange de données n'est autorisé dans cet état.

I.3. LISTE DES EVENEMENTS ENTRANTS

Les évènements entrants sont découpés en deux parties : ceux qui proviennent de l'utilisateur des services, et qui sont décrits dans la partie principale de ce travail (voir 8.1.), et ceux qui proviennent de l'entité réseau.

De l'utilisateur des services :

TCONREQ : Requête de connexion (voir 8.1.).

TCONRSP : Réponse de connexion (voir 8.1.).

TDTREQ : Requête d'émission de données (voir 8.1.).

TEXREQ : Requête d'émission de données exprèss (voir 8.1.).

TDISREQ : Requête de déconnexion (voir 8.1.).

De l'entité réseau :

NCONIND

Cette primitive porte comme paramètres l'identification des points d'accès des services de l'entité appelante et de l'entité appelée, ainsi que l'identificateur d'extrémité de connexion réseau. Dans notre réalisation, il s'agira du préfixe de l'adresse de transport (séparé du reste de l'adresse par un ".").

NCONCNF

En plus de l'identificateur d'extrémité de connexion, cette primitive de confirmation de connexion spécifie l'adresse en réponse de point d'accès des services de l'entité appelée.

NDISIND

Cette primitive indique une déconnexion réseau, et porte la cause de la déconnexion. Suite à cette primitive, toutes les connexions transport liées à cette

connexion réseau cessent d'exister.

NDTIND

Cette primitive sert à signaler des données en provenance du réseau. Un démultiplexage est effectué afin d'identifier la connexion transport à partir de l'identificateur d'extremité de connexion réseau et d'un paramètre "référence destination" codé en 16 bits. Les paramètres de cette primitive sont la longueur et les valeurs des données émises sur la connexion.

Suite à cela, on procède à une identification de la T.P.D.U., et les T.P.D.U. suivantes sont reconnues (voir I.5.) :

- T.P.D.U. CR : demande de connexion.
- T.P.D.U. CC : confirmation de connexion.
- T.P.D.U. DR : demande de déconnexion.
- T.P.D.U. DC : confirmation de déconnexion.
- T.P.D.U. AK : acquièscement.
- T.P.D.U. EA : acquièscement de données express.
- T.P.D.U. ED : données express.
- T.P.D.U. DT : données normales.
- T.P.D.U. ER : erreur.

I.4. LISTE DES EVENEMENTS SORTANTS.

Nous retrouvons ici toutes les requêtes et les réponses à destination de la couche réseau, et les indications ou les confirmations destinées à l'utilisateur des services de transport.

De l'utilisateur des services :

TCONIND : Indication de connexion (voir 8.1.).

TCONCNF : Confirmation de connexion (voir 8.1.).

TDTIND : Indication de données (voir 8.1.).

TEXIND : Indication de données express (voir 81.).

TDISIND : Indication de déconnexion (voir 8.1.).

Rappelons que ces primitives sont des fonctions testées par l'utilisateur des services de transport. Par conséquent, lorsqu'il est signalé dans les traitements qu'une primitive d'indication ou de confirmation est exécutée, cela veut dire que l'entité signalera que la fonction prend la valeur "vrai" lors de son exécution par l'utilisateur des services.

La primitive d'indication de déconnexion a un effet destructif sur l'indication de données et l'indication de données express.

De l'entité réseau :

NCONREQ

Cette primitive effectue la requête de connexion réseau. Ses paramètres sont, outre l'identificateur d'extrémité de connexion appelé par adresse (c'est le fournisseur de services du réseau qui attribue une valeur à ce paramètre), les adresses de point d'accès de services appelant et appelé.

NCONRSP

Suite à une indication de connexion, cette primitive indique que l'entité transport est apte à accepter une nouvelle connexion. Ses paramètres sont l'identificateur d'extrémité de connexion, et l'adresse en retour de point d'accès aux services de l'entité appelée.

NDISREQ

Cette primitive réalise une déconnexion réseau. L'identificateur d'extrémité de connexion réseau, présent à l'appel de la primitive, cesse d'exister à la fin de l'exécution de cette primitive.

NDTREQ

Cette primitive effectue l'émission de données. Elle sera signalée implicitement lors de l'émission d'une T.P.D.U. particulière. Les T.P.D.U. sont les mêmes que celles des évènements entrants.

Au niveau de la réalisation, chaque T.P.D.U. est assemblée dans une procédure propre à la T.P.D.U., et cette procédure se clôture par la requête de données.

I.5. CONTENU DES T.P.D.U.

T.P.D.U. CR

* Un crédit initial, destiné au contrôle de flux (voir 8.2.).

* Une référence source, à utiliser par l'entité appelée si le démultiplexage est choisi.

* Une référence destination, égale à zéro.

* La classe proposée. Dans notre réalisation, il s'agira de la classe 2. Il est de plus precisé que le fournisseur de service prend en charge le contrôle de flux.

* Les identificateurs des point d'accès aux services réseau.

* La taille maximum des T.P.D.U. supportée sur la connexion réseau.

* L'option "utilisation du transfert de données express", toujours présente dans notre réalisation.

* La classe de protocole de repli. Dans notre réalisation, il s'agira de la classe 0.

T.P.D.U. CC

* Le crédit destiné au contrôle de flux (voir 8.2.).

* Une référence source, à utiliser par l'entité appelante si le démultiplexage est choisi.

* La référence destination, égale à la référence source de la T.P.D.U. CR.

* La classe choisie, égale à la classe 2 dans notre réalisation. Rappelons que la classe 0 est également supportée.

* Les identificateurs de points d'accès aux services réseau.

* La taille maximum choisie des T.P.D.U.

* L'option "utilisation du transfert de données express", si elle est supportée par le fournisseur des services, et a été demandée.

T.P.D.U. DR

* La référence destination.

* La référence source.

* La cause de déconnexion.

T.P.D.U. DC

* La référence destination.

* La référence source.

T.P.D.U. DT

* La référence destination (en classe 2).

* Un indicateur de fin d'unité de données de services de transport, utilisé dans le cas où on a du procéder à une segmentation.

* Le numéro de la T.P.D.U.

* Les données de l'utilisateur des services.

T.P.D.U. ED

* La référence destination (en classe 2).

* Le numéro de la T.P.D.U. ED. Dans la classe 2, ce numéro prend une valeur quelconque, mais est quand même présent pour des raisons de conformité.

* Les données de l'utilisateur des services, comprenant de 1 à 16 octets.

T.P.D.U. AK

* La référence destination (en classe 2).

* La nouvelle valeur du crédit.

* Le numéro de séquence en réponse, indiquant le numéro de la prochaine T.P.D.U. attendue. Ce numéro sert à indiquer la limite inférieure de la nouvelle fenêtre.

T.P.D.U. EA

* La référence destination.

* Le numéro de la T.P.D.U. ED dont cette T.P.D.U. fait l'accusé de réception. En classe 2, ce numéro peut prendre n'importe quelle valeur, mais est présent pour des raisons de conformité.

T.P.D.U. ER

* La référence destination.

* La cause du rejet.

I.6. TABLE D'ETATS.

EVENEMENT ENTRANT	ETATS							
	WFNC	WFCC	WCCL	OPEN	CLOSI NG	WFTRSP	CLOSED	
TCONREQ								1
TCONRSP						2		
TDTREQ				3				
TEXREQ				4				
TDISREQ	5	6		7		8		
NCONCNF	9							
NCONIND							10	
NDISIND	11	11	12	11	12	11		
NDTIND								
CR							13	
DR		14	15	16	15		17	
DC					18		19	
CC		20	21				22	
AK				23	24		19	
EA				25	24		19	
ED				26	24		19	

:	DT	:	:	:	:27:	24:	:19:	
:		:	:	:	:	:	:	:
:	ER	:	:28:	29:	30:	30:	:19:	
:		:	:	:	:	:	:	:

I.7. LISTE DES ACTIONS.

A chaque numéro dans la table correspond :

- Soit une action, représentée par un ensemble d'opérations, suivie du nouvel état pris par l'entité, representé par une flèche "==)".

- Soit une condition, suivie d'un double point ":" et d'une action.

- Soit un ensemble de conditions, séparées par un ";".

Si l'intersection d'une ligne et d'une colonne ne contient aucun numéro, ou si pour un numéro aucune condition n'est valide, il y a erreur de protocole, et deux traitements peuvent être effectués :

- Si l'erreur survient suite à une primitive de l'utilisateur des services de transport (TDTREQ, TEXREQ, TCONREQ, TCONRSP, TDISREQ), la variable ERR de la primitive prendra une valeur différente de zéro.
- Si la connexion transport est identifiée, une T.P.D.U. ER est émise vers l'entité ayant causé l'erreur.

L'état CLOSED est un état artificiel. Une connexion ne peut avoir cet état que dans une seule situation : Une déconnexion a été demandée, et l'utilisateur des services de transport n'en n'a pas encore pris connaissance (par une fonction TDISIND).

Dans tous les autres cas, cet état représente l'absence de connexion, c'est à dire qu'il n'existe pas d'identificateur d'extrémité de connexion. C'est l'état par défaut d'une connexion n'existant pas.

1: Si plus de ressources disponibles :
TDISIND,
==) CLOSED;
Si aucune connexion réseau n'existe
entre les entités transport :
NCONREQ,
==) WFCC;
Si une connexion réseau existe entre
les entités transport :
==) WFCC;
Si une connexion réseau a été demandée
et n'est pas encore confirmée :
==) WFNC;

2 : N-DT-REQ(CC),
==) OPEN;

3 : voir 8.2.4.,
==) OPEN

4 : voir 8.2.5.,
==) OPEN;

5 : Si pas d'autre connexions transport
sur cette connexion réseau :
NDISREQ,
==) CLOSED;
S'il y a d'autres connexions transport
sur cette connextion réseau :
==) CLOSED;

6 : ==) WBCL;

7 : N-DT-REQ(DR),
==) CLOSING;

8 : N-DT-REQ(DR),
==) CLOSED;

9 : N-DT-REQ(CR),
==) WFCC;

10: Création d'une connexion réseau.

11: TDISIND,
==) CLOSED;

12: ==) CLOSED;

13: Si les ressources sont suffisantes

pour une nouvelle connexion :
TCONIND,
==) WFTRESP;
Sinon :
N-DT-REQ(DR),
==) CLOSED;

14: TDISIND,
==) CLOSED;
Si pas d'autre connexions transport
sur la connexion réseau :
NDISREQ;

15: ==) CLOSED;
Si pas d'autre connexions transport
sur la connexion réseau :
NDISREQ;

16: N-DT-REQ(DC),
TDISIND,
==) CLOSED;

17: N-DT-REQ(DC),
==) CLOSED;

18: ==) CLOSED;
Si pas d'autre connexions transport
sur la connexion réseau :
NDISREQ;

19: ==) CLOSED;

20: Si les paramétres de la T.P.D.U. sont acceptables :
TCONCNF,
==) OPEN;
Si les paramétres ne sont pas acceptables :
TDISIND,
N-DT-REQ(DR),
==) CLOSING;

21: N-DT-REQ(DR),
==) CLOSING;

22: N-DT-REQ(DR),
==) CLOSED;

23: Voir 8.2.4.,
==) OPEN;

24: ==) CLOSING;

25: Voir 8.2.5.,
==) OPEN;

26: Voir 8.2.5.,
==) OPEN;

27: Voir 8.2.4.,
==) OPEN;

28: TDISIND,
==) CLOSED;
Si pas d'autre connexion transport
sur la connexion réseau :
NDISREQ;

29: ==) CLOSED;
Si pas d'autre connexion transport
sur la connexion réseau :
NDISREQ;

30: TDISIND,
N-DT-REQ(DR),
==) CLOSING;

A N N E X E II

D E S C R I P T I O N

D E S A U T R E S S E R V I C E S

D E S E S S I O N

Dans cette annexe, nous décrirons les différents services de la couche session qui n'ont pas été définis dans la partie principale de ce travail. La réalisation de ces primitives a été effectuée, mais leur intégration dans l'ensemble du travail n'a pas été faite, pour des raisons de performance de l' APPLE-II.

Il étatit en effet totalement inutile de charger le système avec un ensemble de procédures qui ne sont pas utilisées par les couches supérieures (présentation et application F.T.A.M.).

Mais afin que le lecteur puisse se faire une idée de l'ensemble des procédures définies par I.S.O. pour la couche session, nous les décrirons ici. Les concepts définis dans la partie principale du travail sont toujours valables ici.

La présentation des services et des protocoles est identique à celle faite dans la partie principale de ce travail.

II.1. SPECIFICATION DES SERVICES.

II.1.1. Négociation de terminaison.

Lors de la réponse à une requête de déconnexion (S-REL-RSP voir 9.1.2.3.), le paramètre RESULT peut prendre une valeur différente de zéro, auquel cas la déconnexion n'est pas faite. Pour refuser cette libération de connexion, l'entité doit posséder le jeton de négociation de libération de connexion (voir 9.1.1.2.).

II.1.2. Demande de jetons.

```
PROCEDURE S-PT-REQ
   (S-IDT:TYP-IDT;
    TOKEN:ONE-BYTE;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER);

FUNCTION S-PT-IND
   (S-IDT:TYP-IDT;
    VAR TOKEN:ONE-BYTE;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER):BOOLEAN;
```

Ces primitives servent à demander des jetons que l'entité émettrice ne possède pas. Le codage des jetons est identique à celui de l'attribution des jetons (voir 9.1.2.7.).

II.1.3. Transfert de données express.

```
PROCEDURE S-EX-REQ
   (S-IDT:TYP-IDT;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER);

FUNCTION S-EX-IND
   (S-IDT:TYP-IDT;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER):BOOLEAN;
```

Ces procédures servent à émettre et à recevoir des unités de données express de service. La taille maximum de ces unités de données est limitée à 16 octets.

II.1.4. Transfert de données typées.

```
PROCEDURE S-TD-REQ
   (S-IDT:TYP-IDT;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER);

FUNCTION S-TD-IND
   (S-IDT:TYP-IDT;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER):BOOLEAN;
```

Les données typées peuvent se transférer indépendemment de la possession du jeton des données. Le reste du fonctionnement de ces primitives est identique à celui du transfert des données normales (voir 9.1.2.4.), y compris la taille illimitée des unités de données et la segmentation éventuelle.

II.1.5. Transfert de données de capacités.

L'unité fonctionnelle de transfert des données de capacités ne peut être sélectionnée que si l'unité fonctionnelle de gestion d'activité a été sélectionnée. Son existence est liée à la recherche de compatibilité avec les normes du C.C.I.T.T., dans lesquelles le transfert de données de capacités sert aux entités communicantes à décrire leurs capacités (taille de tampon, vitesse de transmission...).

```
PROCEDURE S-CD-REQ
   (S-IDT:TYP-IDT;
    VAR SSDU-TYP-SSDU;
    VAR ERR:INTEGER);

FUNCTION S-CD-IND
   (S-IDT:TYP-IDT;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER):BOOLEAN;

PROCEDURE S-CD-RSP
   (S-IDT:TYP-IDT;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER);

FUNCTION S-CD-CNF
   (S-IDT:TYP-IDT;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER):BOOLEAN;
```

L'émission de la requête de transfert de données de capacités ne peut avoir lieu que lorsque l'entité

possède le jeton de synchronisation majeure (voir 9.1.1.2.) et qu'aucune activité n'est en cours (voir 9.1.1.5. et plus bas).

Après avoir émis une requête de transfert de données de capacités, l'entité attend une confirmation de données de capacités.

Une entité recevant une indication de données de capacités est tenue d'émettre la réponse de données de capacités avant l'usage de toute autre primitive.

Le nombre maximum d'octets de l'unité de données de service est limité à 512.

II.1.6. Synchronisation majeure.

```
PROCEDURE S-MAJ-REQ
   (S-IDT:TYP-IDT;
    NUM:SIX-BYTES;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER);

FUNCTION S-MAJ-IND
   (S-IDT:TYP-IDT;
    VAR NUM:SIX-BYTES;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER):BOOLEAN;

PROCEDURE S-MAJ-RSP
   (S-IDT:TYP-IDT;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER);

FUNCTION S-MAJ-CNF
   (S-IDT:TYP-IDT;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER):BOOLEAN;
```

La synchronisation majeure est un service confirmé, permettant à une entité de poser des points de synchronisation délimitant des unités de dialogue (voir 9.1.1.3.).

Après avoir émis une requête de synchronisation majeure (S-MAJ-REQ), l'entité ne peut plus utiliser d'autres services (sauf la coupure, la resynchronisation ou l'interruption d'activité) jusqu'à la réception de la confirmation de la synchronisation (S-MAJ-CNF).

A la réception d'une indication de synchronisation majeure (S-MAJ-IND), l'entité ne peut requérir

l'utilisation de synchronisation, de clôture ou d'interruption d'activité avant d'avoir émis une réponse de synchronisation majeure (S-MAJ-RSP).

II.1.7. Service de rapport d'erreurs.

Le service de rapport d'erreurs peut être utilisé par les utilisateurs des services afin de signaler une anomalie, ou par le fournisseur des services lors de la détection d'une erreur de protocole.

Ce service n'est autorisé que si l'unité fonctionnelle de rapport d'erreurs (voir 9.1.1.1.11.) a été sélectionnée. De plus, cette unité fonctionnelle ne peut être utilisée que si l'unité fonctionnelle de transfert de données en semi-duplex a été choisie.

```
PROCEDURE S-ED-REQ
   (S-DIT:TYP-IDT;
    REASON:ONE-BYTE;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER);

FUNCTION S-ED-IND
   (S-IDT:TYP-IDT;
    VAR REASON:ONE-BYTE;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER):BOOLEAN;

FUNCTION S-ER-IND
   (S-IDT:TYP-IDT;
    VAR REASON:ONE-BYTE;
    VAR ERR:INTEGER):BOOLEAN;
```

La procédure S-ED-REQ ne peut être utilisée par une entité que si elle ne possède pas le jeton de données. La fonction S-ED-IND signale un rapport d'erreur émis par l'entité communicante.

La fonction S-ER-IND est signalée aux deux utilisateurs de services, indiquant une erreur détectée par le fournisseur des services (et ne comportant donc aucune donnée de l'utilisateurs).

II.1.8. Gestion d'activité.

II.1.8.1. Démarrage d'activité.

```
PROCEDURE S-AS-REQ
   (S-IDT:TYP-IDT;
```

```
    ACTIVITY-NAME:SIX-BYTES;
    VAR NSDU:TYP-NSDU;
    VAR ERR:INTEGER);

FUNCTION S-AS-IND
   (S-IDT:TYP-IDT;
    VAR ACTIVITY-NAME:SIX-BYTES;
    VAR NSDU-TYP-NSDU;
    VAR ERR:INTEGER):BOOLEAN;
```

Ce service permet à l'entité possédant le jeton de synchronisation majeure et de gestion d'activité (voir 9.1.1.2.) de démarrer une activité. Ce service ne peut être demandé que si l'unité fonctionnelle de gestion d'activité a été sélectionnée et qu'aucune activité n'est en cours.

II.1.8.2. Redémarrage d'activité.

```
PROCEDURE S-AR-REQ
   (S-IDT:TYP-IDT;
    CALLING-REFERENCE:ARRAY-24;
    CALLED-REFERENCE:ARRAY-24;
    COMMON-REFERENCE:ARRAY-14;
    ADDITIONAL-REFERENCE:ARRAY-2;
    OLD-ACTIVITY-NAME:SIX-BYTES;
    NEW-ACTIVITY-NAME:SIX-BYTES;
    NUM:SIX-BYTES;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER);

FUNCTION S-AR-IND
   (S-IDT:TYP-IDT;
    VAR CALLING-REFERENCE:ARRAY-24;
    VAR CALLED-REFERENCE:ARRAY-24;
    VAR COMMON-REFERENCE:ARRAY-14;
    VAR ADDITIONAL-REFERENCE:ARRAY-2;
    VAR OLD-ACTIVITY-NAME:SIX-BYTES;
    VAR NEW-ACTIVITY-NAME:SIX-BYTES;
    VAR NUM:SIX-BYTES;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER):BOOLEAN;
```

Ce service permet à une entité de reprendre une activité préalablement interrompue (voir II.1.8.3.). Dans le cas où une activité avait démarré avec une autre connexion session, les références des utilisateurs sont reprises, ainsi que le nom de l'ancienne activité, afin de faire le lien avec cette dernière.

II.1.8.3. Interruption d'activité.

```
PROCEDURE S-AI-REQ
   (S-IDT:TYP-IDT;
    REASON:ONE-BYTE;
    VAR ERR:INTEGER);

FUNCTION S-AI-IND
   (S-IDT:TYP-IDT;
    VAR REASON:ONE-BYTE;
    VAR ERR:INTEGER):BOOLEAN;

PROCEDURE S-AI-RSP
   (S-DIT:TYP-IDT;
    VAR ERR:INTEGER);

FUNCTION S-AI-CNF
   (S-IDT:TYP-IDT;
    VAR ERR:INTEGER):BOOLEAN;
```

Ce service permet d'interrompre une activité (éventuellement redémarrée plus tard - voir II.1.8.2.). L'entité désirant utiliser ce service doit posséder le jeton de gestion d'activité (voir 9.1.1.2.).

Après avoir émis une requête d'interruption d'activité (S-AI-REQ), l'entité ne peut plus requérir de services (sauf la coupure de connexion) avant d'avoir reçu la confirmation de l'interruption (S-AI-CNF).

A la réception d'une indication d'interruption d'activité (S-AI-IND), l'entité ne peut utiliser aucun service (excepté la coupure de connexion) avant d'émettre une réponse d'interruption (S-AI-RSP).

II.1.8.4. Abandon d'activité.

```
PROCEDURE S-AD-REQ
   (S-IDT:TYP-IDT;
    REASON:ONE-BYTE;
    VAR ERR:INTEGER);

FUNCTION S-AD-IND
   (S-IDT:TYP-IDT;
    VAR REASON:ONE-BYTE;
    VAR ERR:INTEGER):BOOLEAN;

PROCEDURE S-AD-RSP
   (S-DIT:TYP-IDT;
    VAR ERR:INTEGER);
```

```
FUNCTION S-AD-CNF
   (S-IDT:TYP-IDT;
    VAR ERR:INTEGER):BOOLEAN;
```

Ce service s'utilise de la même façon que le service d'interruption d'activité (voir II.1.8.3.). Une activité abandonnée ne peut cependant être redémarrée.

II.1.8.5. Terminaison d'activité.

```
PROCEDURE S-AE-REQ
   (S-IDT:TYP-IDT;
    NUM:SIX-BYTES;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER);

FUNCTION S-AE-IND
   (S-IDT:TYP-IDT;
    VAR NUM:SIX-BYTES;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER):BOOLEAN;

PROCEDURE S-AE-RSP
   (S-IDT:TYP-IDT;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER);

FUNCTION S-AE-CNF
   (S-IDT:TYP-IDT;
    VAR SSDU:TYP-SSDU;
    VAR ERR:INTEGER):BOOLEAN;
```

Une entité peut signaler la fin d'une activité au moyen de la requête de terminaison d'activité (S-AE-REQ), si elle possède le jeton de gestion d'activité (voir 9.1.1.2.). A partir de ce moment, l'entité ne peut plus utiliser de services autres que la coupure de connexion avant de recevoir la confirmation de terminaison d'activité (S-AE-CNF).

L'entité recevant une indication de terminaison d'activité (S-AE-IND) ne peut utiliser d'autres services que la coupure de connexion avant d'émettre une réponse de fin d'activité (S-AE-RSP).

II.1.8.6. Passage de contrôle.

Ce service ne peut être utilisé que lorsqu'aucune activité n'est en cours, et sert à prendre le contrôle de la connexion, en reclamant tous les jetons disponibles.

```
PROCEDURE S-CG-REQ
   (S-IDT:TYP-IDT;
    VAR ERR:INTEGER);

FUNCTION S-CG-IND
   (S-IDT:TYP-IDT;
    VAR ERR:INTEGER):BOOLEAN;
```

La requête de passage de contrôle (S-CG-REQ) émise, l'utilisateur des services peut considérer qu'il possède tous les jetons disponibles.

A la réception de l'indication de passage de contrôle (S-CG-IND), l'utilisateur des services ne possède plus aucun des jetons disponibles. Il ne lui est pas loisible de refuser ce passage de contrôle.

II.2. DESCRIPTION DES PROTOCOLES.

II.2.1. Négociation de terminaison.

Lorsque la primitive de réponse de déconnexion porte le paramètre RESULTAT différent de zéro, une S.P.D.U. NF (Not Finish) est émise, contenant juste des données de l'utilisateur. La connexion continue d'exister.

II.2.2. Demande de jetons.

La S.P.D.U. PT (Please Tokens) de demande de jetons contient comme paramètres :

- La liste des jetons demandés, et

- Les données de l'utilisateur.

La S.P.D.U. est envoyée sur le flux normal des données, et n'engendre pas de modification de l'état des entités. Le fait qu'un jeton demandé ne soit pas en possession de l'entité réceptrice de la S.P.D.U. n'engendre pas d'erreur de protocole.

II.2.3. Transfert de données express.

La S.P.D.U. EX (EXpedited data) de transfert de données express est envoyée sur le flux des données express via une requête de transfert de données express de la couche transport (T-EX-REQ), et contient les données de l'utilisateur.

II.2.4. Transfert de données typées.

Toutes les procédures de transfert de données typées sont identiques à celles de transfert de données normales (voir 9.2.2.5.), sauf que c'est une S.P.D.U. TD (Typed Data) qui est émise, et que la possesion du jeton des données n'est pas nécessaire en semi-duplex pour l'utilisation de ce service.

II.2.5. Transfert de données de capacités.

Le transfert de données de capacités est un service confirmé. Suite à une requête de données de capacités, l'entité émet une S.P.D.U. CD (Capability Data), contenant les données de l'utilisateur, sur le flux des données normales. A partir de ce moment, l'utilisateur des services ne peut plus utiliser de

services avant d'avoir reçu la confirmation de données de capacités.

A l'indication d'une S.P.D.U. de données de capacités, l'entité est tenue d'émettre une réponse de données de capacités, dont les données seront émises sous forme d'une S.P.D.U. CDA (Capability Data Ack) sur le flux des données normales. A ce moment, l'entité se remet dans un état de transfert de données.

A la confirmation de données de capacités, représentée par la S.P.D.U. CDA, l'entité se remet dans un état de transfert de données.

II.2.6. Synchronisation majeure.

La S.P.D.U. MAP (MAjor Point) de requête de synchronisation majeure comporte comme éléments :

- Un paramètre de type de synchronisation indiquant qu'il s'agit de la fin d'une unité de dialogue, et non la fin d'une activité (et cela même si l'unité fonctionnelle de gestion d'activité n'a pas été sélectionnée).

- Le numéro de série, qui indique le numéro de la synchronisation.

- Les données de l'utilisateur.

Dès que la S.P.D.U. MAP a été envoyée, l'entité se met en attente d'une S.P.D.U. PR-MAA (PRepare MAjor Ack - si le transfert de données express est autorisé) et/ou une S.P.D.U. MAA (MAjor Ack).

La réception d'une S.P.D.U. MAP engendre une indication de synchronisation majeure, et l'entité attend une réponse de synchronisation majeure de l'utilisateur des services.

Si le transfert des données express est mis en oeuvre, la réponse de synchronisation majeure entraînera d'abord l'émission d'une S.P.D.U. PR-MAA (PRepare MAjor Ack) sur le flux des données express, suivie de l'émission d'une S.P.D.U. MAA contenant le numéro de série de la synchronisation majeure et les données utilisateur. Si le transfert des données express n'est pas disponible, seule la S.P.D.U. MAA est émise.

II.2.7. Signalisation d'anomalies.

Deux types de services distincts sont gérés par les

procédures de signalisation d'anomalies :

- Les indications émises aux deux utilisateurs des services, et provenant du fournisseur de services.

- Les requêtes d'une entité signalées par des indication à l'entité communicante.

Lorsqu'une entité détecte une erreur, elle émet une S.P.D.U. ER (Exception Report) vers l'entité communicante, et une indication d'erreur à l'utilisateur des services de cette entité.

L'entité recevant une S.P.D.U. ER la signale à l'utilisateur des services en émettant une indication d'erreur.

Les deux entités entrent dans un état d'où elles ne peuvent sortir qu'à la requête ou la réception d'une S.P.D.U. de coupure ou d'interruption d'activité, de resynchronisation, de coupure de connexion ou de cession du jeton des données (rappelons que ce service n'est autorisé qu'en transmission semi-duplex).

La requête de signalisation d'erreur sert à mettre les entités dans un état d'erreur duquel elles ne pourront sortir que par une requête identique à celles ci-dessus, émise par l'entité ayant reçu la S.P.D.U. ED (Exception Data), contenant un code d'erreur et des données utilisateur.

II.2.8. Gestion d'activité.

Nous verrons ici la gestion d'activité. Rappelons que l'utilisation des primitives de gestion d'activité n'est autorisée que si cette unité fonctionnelle a été sélectionnée.

De plus, toute requête de ces primitives ne peut être exécutée par une entité que si elle possède le jeton de synchronisation majeure et de gestion d'activité (voir 9.1.1.2.).

II.2.8.1. Démarrage d'activité.

La requête de démarrage d'activité engendre l'émission d'une S.P.D.U. AS (Activity Start), contenant l'identificateur d'activité, et les données utilisateurs.

Cette S.P.D.U. ne peut être émise que si aucune activité n'est en cours. Dès émission ou réception de la S.P.D.U. AS, l'activité est en

cours, et la transmission de données peut avoir lieu.

II.2.8.2. Redémarrage d'activité.

La requête de redémarrage d'activité est destinée à signaler la reprise d'une activité préalablement interrompue. Cette requête engendre l'émission d'une S.P.D.U. AR (Activity Resume), contenant :

- Un paramètre destiné à faire le lien avec l'ancienne activité, et contenant :
  - La référence de l'utilisateur appelé.
  - La référence de l'utilisateur appelant.
  - Une référence commune.
  - Une référence additionnelle.
  - L'identificateur de l'ancienne activité.
  - Un numéro de série initial.
- L'identificateur de la nouvelle activité.
- Des données de l'utilisateur.

Dès l'émission et à la réception de la S.P.D.U. AR, l'activité est considérée comme ayant redémarré, et la transmission de données est autorisée.

II.2.8.3. Interruption d'activité.

La requête d'interruption d'activité engendre l'émission d'une S.P.D.U. AI (Activity Interrupt). Cette S.P.D.U. AI contient un paramètre de cause, mais pas de données de l'utilisateur.

Si l'utilisation des données express sur la connexion transport est autorisée, une S.P.D.U. PR-RS (PRepare ReSynchronize - préparation à la resynchronisation) est émise simultanément sur ce flux.

Après avoir émis la S.P.D.U. AI, l'entité entre dans un état où toutes les S.P.D.U. sont écartées, sauf les S.P.D.U. PR-RS (cas de collision), les S.P.D.U. PR-RA (PRepare

Resynchronize Ack - préparation à l'acceptation d'interruption), la S.P.D.U. AIA (acquiescement d'interruption d'activité - voir plus bas) et une S.P.D.U. AB de coupure de connexion.

La réception d'une S.P.D.U. AI engendre une indication d'interruption d'activité. Si la connexion transport supporte l'utilisation du transfert de données express, cette S.P.D.U. doit être précédée d'une S.P.D.U. PR-RS. L'entité se met alors en attente d'une réponse d'interruption d'activité.

La réponse d'interruption d'activité engendre l'émission d'une S.P.D.U. AIA (Activity Interrupt Ack), précédée de l'émission d'une S.P.D.U. PR-RA (préparation à l'acquiescement de resynchronisation) si l'utilisation des données express est autorisée sur la connexion transport. La S.P.D.U. AIA ne comporte aucun paramètre. Dès l'émission de la S.P.D.U. AIA, l'activité est suspendue, et ne pourra reprendre qu'à l'émission ou la réception d'une S.P.D.U. AR (activity resume, ou redémarrage d'activité).

La réception d'une S.P.D.U. AIA engendre une confirmation d'interruption d'activité, et l'activité en cours est suspendue.

II.2.8.4. Abandon d'activité.

Le fonctionnement de ce groupe de primitives est identique à celui de l'interruption d'activité (voir II.2.8.3.), sauf que ce sont des S.P.D.U. AD (Activity Discard) et ADA (Activity Discard Ack) qui sont émises, et que le redémarrage d'une activité n'est pas autorisé.

II.2.8.5. Terminaison d'activité.

Le fonctionnement de la terminaison d'activité est identique à celui de la pose de points de synchronisation majeure (voir II.2.6.), si ce n'est que l'activité se clôture, et que le paramètre de type de synchronisation n'est pas présent dans la S.P.D.U.. Les S.P.D.U. émises sont les S.P.D.U. AE (Activity End) et AEA (Activity End Ack), dont le codage est identique aux S.P.D.U. de synchronisation majeure.

II.2.8.6. Passage de contrôle.

Suite à une requête de passage de contrôle, l'entité émet une S.P.D.U. GTC (Give Tokens Confirm), ne contenant aucun paramètre. Suite à celà, l'entité n'accepte plus aucune autre S.P.D.U. que la S.P.D.U. GTA (Give Tokens Ack).

La réception d'une S.P.D.U. GTC engendre d'une part une indication de passage de contrôle, et d'une autre part l'émission d'une S.P.D.U. GTA, ne contenant aucun paramètre.

ANNEXE III

DESCRIPTION

DES TABLES D'ETATS

DE LA COUCHE SESSION

III.1. VUE GENERALE.

Cette annexe est consacrée à la description des protocoles de la couche session.

Comme dans les descriptions des protocoles de la couche transport, les protocoles de la couche session sont définis sous forme de tables, dont les lignes représentent les évènements entrants et les colonnes les différents états que peut prendre une entité session.

Les tables d'états ci-dessous reprennent l'ensemble des protocoles de la couche session. Suite aux négociations, certaines unités fonctionnelles peuvent ne pas avoir été choisies (ce qui est le cas dans notre réalisation), mais les fonctions de vérification de validité servent à détecter si tel service est mis en oeuvre ou pas.

Les évènements entrants et sortants sont décrits dans la partie principale de ce travail (voir 8.1. et 9.1.). Les noms des différents états sont les mêmes que ceux utilisés dans les textes des normes I.S.O., afin de simplifier toute modification suite à des ajoutes des instituts de normalisation.

Au niveau du programme, pour chaque évènement entrant, on procède aux tests de validité de la primitive selon les unités fonctionnelles choisies, puis selon les états, on procède à l'activation des procédures appropriées.

III.2. VARIABLES LOCALES

Unités fonctionnelles :

- FD : Transmission en duplex.
- HD : Transmission en semi-duplex.
- EXCEP : Traitement des anomalies.
- TD : Utilisation des données typées.
- NR : Négociation de clôture de connexion.
- SY : Utilisation de la synchronisation mineure.
- MA : Utilisation de la synchronisation majeure.
- RESYN : Utilisation de la resynchronisation.
- EX : Transfert de données express.
- ACT : Gestion d'activité.
- CD : Utilisation des données de capacités.

La fonction booléenne **FU(T)**, où T est une des unités fonctionnelles décrites ci-dessus, prend la valeur "vrai" si l'unité fonctionnelle a été sélectionnée lors de la création de connexion.

Jetons.

Quatre jetons existent :

- MI : Jeton de synchronisation mineure.
- MA : Jeton de synchronisation majeure et d'activité.
- TR : Jeton de négociation de terminaison.
- DK : Jeton de données.

Les fonctions suivantes sont utilisées lors de la vérification du droit d'utilisation de certains services.

- **AV(T)**, où T est un des jetons, prend la valeur "vrai" si le jeton est disponible. Les valeurs de AV(T) sont :
  - AV(MI) := FU(SY)
  - AV(DK) := FU(HD)

- AV(TR) := FU(NR)

- AV(MA) := FU(MA) ou FU(ACT)

- **OWNED(T)** est une fonction qui prend la valeur "vrai" si le jeton (T) est disponible et possédé par l'entité. Elle prend la valeur "faux" si le jeton est disponible et non possédé par l'entité, et n'est pas définie si le jeton n'est pas disponible.

- **I(T)** := non AV(T) ou OWNED(T)

- **II(T)** := AV(T) et OWNED(T)

- **A(T)** := non AV(T) ou non OWNED(T)

- **AA(T)** := AV(T) et non OWNED(T)

Quelques variables locales sont utilisées :

- **TEXP** est "vrai" si la connexion transport autorise le transfert de données express.

- **VACT** est "vrai" si une activité est en cours.

- **VNEXTACT** est "vrai" si un point de synchronisation majeure a été reçu lorsqu'une activité est en cours, à "faux" si, lors d'une activité en cours, une S.P.D.U. AE est émise ou reçue.

- **VRSP** est une variable utilisée pour résoudre les problèmes de collisions de resynchronisation, et indique le type de resynchronisation en cours, prenant les valeurs :

  - NO : pas de resynchronisation en cours.

  - A : resynchronisation ABANDON.

  - R : resynchronisation RESTART.

  - S : resynchronisation SET.

  - DSC : abandon d'activité.

  - INT : interruption d'activité.

- **VRSPNB** indique le numéro de série en cas de collision de resynchronisation RESTART.

- **SPMWINNER** est une fonction booléenne qui prend la valeur "vrai" si l'entité est gagnante en cas de collision de resynchronisation, de terminaison ou d'interruption d'activité. La fonction d'évaluation est la suivante :

- On évalue d'abord les prochaines valeurs de VRSP et VRSPNB (voir ci-dessus).

- On compare ces nouvelles valeurs aux anciennes, sachant que :

DSC prévaut sur INT

INT prévaut sur A

A prévaut sur S

S prévaut sur R

R prévaut sur NO

- En cas d'égalité et si VRSP vaut R, on compare les deux VRSPNB, et la plus petite valeur prévaut sur la plus grande.

- Si le résultat de la comparaison indique que la nouvelle resynchronisation prévaut sur l'ancienne, la fonction SPMWINNER prend la valeur "vrai", et on tient compte de cette nouvelle resynchronisation, remettant à jour les valeurs de VRSP et VRSPNB. Sinon, on oublie cette nouvelle resynchronisation.

- En cas d'égalité, et si la dernière resynchronisation a été émise par l'initiateur de la connexion, alors la fonction SPMWINNER prend la valeur "faux".

- **VTCA** est une variable booléenne prenant la valeur "faux" si l'entité est l'initiatrice de la connexion.

- **VTRR** est une variable booléenne prenant la valeur "vrai" si la connexion transport peut être réutilisée lors d'une connexion session ultérieure.

- **VCOLL** est une variable booléenne prenant la valeur "vrai" si une collision de S.P.D.U. FN (finish) a été détectée.

- **V(A)** est le plus petit numéro de série pour lequel une confirmation de synchronisation est attendue. Il n'y a pas de confirmation de synchronisation attendue lorsque V(A) = V(M) (voir plus bas).

- **V(M)** est le prochain numéro de série à utiliser.

- **V(R)** est le plus petit numéro de série pour lequel une resynchronisation avec l'option RESTART est autorisée.

- **VSC** est une variable booléenne prenant la valeur "vrai" si l'utilisateur des services a le droit d'émettre une

réponse de synchronisation mineure quand V(A) est inférieur à V(M).

III.3. LISTE DES ETATS.

* STA01 :

Cet état est factice, et correspond au cas où il n'existe pas de connexion réseau. Le seul cas où une connexion prend cet état est quand la connexion est clôturée, mais que l'utilisateur des services n'en est pas encore averti.

* STA01A :

Après avoir émis une S.P.D.U. AB (ABort), l'entité attend une S.P.D.U. AA (Abort Accept). Cet état n'est atteint que si la connexion transport est réutilisée pour une autre connexion session.

* STA01B :

Après avoir émis une requête de connexion transport (T-CON-REQ), l'entité attend une confirmation de connexion transport (T-CON-CNF).

* STA01C :

Dans cet état, la connexion session n'existe plus, mais la connexion transport n'est pas clôturée. Ce mécanisme permet d'éviter les négociations entre deux entités transport.

* STA02A :

Après émission d'une S.P.D.U. CN (CoNnect), l'entité attend soit une S.P.D.U. AC (ACcept), soit une S.P.D.U. RF (ReFuse).

* STA03 :

Suite à l'émission d'une S.P.D.U. FN (FiNish), l'entité attend une S.P.D.U. DN (DiscoNnect) ou une S.P.D.U. NF (Not Finish).

* STA04A :

Après avoir émis une S.P.D.U. MAP (MAjor Point), l'entité attend une S.P.D.U. MAA (MAjor Ack) si TEXP a la valeur "faux", une S.P.D.U. PR-MAA (PRepare MAjor Ack) si TEXP a la valeur "vrai".

* STA04B :

Après avoir émis une S.P.D.U. AE (Activity End), l'entité attend une S.P.D.U. AEA (Activity End Ack) si TEXP a la valeur "faux", une S.P.D.U. PR-MAA (PRepare MAjor Ack) si TEXP a la valeur "vrai".

* STA05A :

Suite à l'émission d'une S.P.D.U. RS (ReSynchronize), l'entité attend une S.P.D.U. RA (Resynchronize Ack) si TEXP a la la valeur "faux", une S.P.D.U. PR-RA (PRepare Resynchronize Ack) sinon.

* STA05B :

Suite à l'émission d'une S.P.D.U. AI (Activity Interrupt), l'entité attend une S.P.D.U. AIA (Activity Interrupt Ack) si TEXP a la valeur "faux", une S.P.D.U. PR-RA sinon.

* STA05C :

Suite à l'émission d'une S.P.D.U. AD (Activity Discard), l'entité attend une S.P.D.U. ADA (Activity Discard Ack) si TEXP a la valeur "faux", une S.P.D.U. PR-RA sinon.

* STA06 :

Cet état est atteint lorsque l'entité attendait une S.P.D.U. RA (Resynchronize Ack) ou une S.P.D.U. PR-RA (PRepare Resynchronize Ack), et qu'elle reçoit une S.P.D.U. PR-RS (PRepare ReSynchronize), ce qui est un cas de collision de resynchronisation. C'est la S.P.D.U. RS qui suit la S.P.D.U. PR-RS qui permettra de déterminer quelle est l'entité gagnante de la collision (fonction SPMWINNER).

* STA08 :

Après avoir signalé une indication de création de connexion (S-CON-IND), l'entité attend de la part de l'utilisateur des services une réponse de création de connexion (S-CON-RSP).

* STA09 :

Après avoir signalé une indication de déconnexion (S-REL-IND), l'entité attend une réponse de déconnexion (S-REL-RSP) de la part de l'utilisateur des services.

* **STA10A :**

Après avoir reçu la S.P.D.U. PR-MAA (PRepare MAjor Ack), l'entité attend une S.P.D.U. MAA (MAjor Ack).

* **STA10B :**

Après avoir reçu la S.P.D.U. PR-MAA, l'entité attend une S.P.D.U. AEA (Activity End Ack).

* **STA11A :**

Suite à l'indication de resynchronisation (S-RS-IND), l'entité attend une réponse de resynchronisation (S-RS-RSP) de l'utilisateur des services.

* **STA11B :**

Suite à l'indication d'interruption d'activité (S-AI-IND), l'entité attend une réponse d'interruption d'activité (S-AI-RSP) de l'utilisateur des services).

* **STA11C :**

Suite à l'indication d'abandon d'activité (S-AD-IND), l'entité attend une réponse d'abandon d'activité (S-AD-RSP) de l'utilisateur des services.

* **STA15A :**

Après avoir reçu une S.P.D.U. PR-MAA (PRepare MAjor Ack), l'entité attend soit une S.P.D.U. MAA (MAjor Ack), soit une S.P.D.U. AEA (Activity End Ack).

* **STA15B :**

Après avoir reçu une S.P.D.U. PR-RS (PRepare ReSynchronize), l'entité attend soit une S.P.D.U. RS (ReSynchronize), soit une S.P.D.U. AD (Activity Discard), soit une S.P.D.U. AI (Activity Interrupt).

* **STA15C :**

Après avoir reçu une S.P.D.U. PR-RA (prepapre Resynchronize Ack), l'entité attend soit une S.P.D.U. RA (Resynchronize Ack), soit une S.P.D.U. AIA (Activity Interrupt Ack), soit une S.P.D.U. ADA (Activity Discard Ack).

* STA16 :

L'entité atteint cet état après avoir émis soit une S.P.D.U. DN (DisconNect), soit une S.P.D.U. AB (ABort), et dans les deux S.P.D.U. le paramètre "réutilisation de la connexion transport" est mis à "faux". Dans ce cas, l'entité attend une indication de déconnexion transport (T-DIS-IND).

* STA18 :

Cet état est atteint après avoir émis une S.P.D.U. GTC (Give Tokens Confirm), et l'entité attend une S.P.D.U. GTA (Give Tokens Ack).

* STA19 :

Aprés avoir émis une S.P.D.U. ED (Exception Data), l'entité attend un recouvrement d'un des deux utilisateurs des services.

* STA20 :

Aprés avoir reçu une S.P.D.U. ED (Exception Data) ou une S.P.D.U. ER (Exception Report), l'entité attend un recouvrement d'une des deux utilisateurs des services.

lm9
* STA21 :

Après avoir émis une S.P.D.U. CD (Capability Data), l'entité attend une S.P.D.U. CDA (Capability Data Ack).

* STA22 :

Après avoir signalé une indication de données de capacités (S-CD-IND), l'entité attend une réponse de données de capacités (S-CD-RSP) de l'utilisateur des services.

* STA713 :

C'est l'état de transfert normal des données, duquel sont permises toutes les requêtes.

III.4. RELATIONS AVEC LA COUCHE TRANSPORT.

Nous feront la distinction entre les primitives de requête et de réponse d'une part, d'indication et de confirmation d'une autre part.

les primitives de requête de création de connexion (T-CON-REQ), de réponse à la création de connexion (T-CON-RSP), et de requête de libération de connexion (T-DIS-REQ) sont mentionnées telles quelles dans les listes d'actions ci-dessous. Les différents paramètres décrits dans la partie principale de ce travail (voir 8.1.) se déduisent assez facilement du contexte. La création et la libération du jeu de variables locales nécessaires à la gestion de l'entité sont implicites.

Les requêtes de transfert de données (T-DT-REQ) et de transfert de données express (T-EX-REQ) comportent comme argument le nom de la S.P.D.U. qu'ils transmettent. Ne sont pas décrits ci-dessous les segmentations et réassemblages éventuels, afin de ne pas alourdir la description.

L'indication de création de connexion (T-CON-IND), et la confirmation de création de connexion (T-CON-CNF), sont des évènements créant une nouvelle entité, et sont à ce titre repris dans les lignes des tables d'états. Les actions du protocole en découlant sont décrites dans les tables, mais non la façon dont sont créées les variables locales nécessaires à la gestion de l'entité.

De même, l'indication de libération de connexion (T-DIS-IND) détruira ces variables locales.

Les fonctions d'indication de données (T-DT-IND) et d'indication de données express (T-EX-IND) ne sont pas mentionnées dans ces tables. A la place, les noms des différentes S.P.D.U. sont donnés. En fait, on suppose une analyse des unités de données de service de transport, permettant de transformer la T.S.D.U. en une S.P.D.U. On ne décrit pas le réassemblage ou l'éclatement nécessaire pour y arriver.

Le service de transfert de données express de transport étant sujet à négociation (et pouvant donc ne pas être mis en oeuvre), on ne fait pas la distinction entre les S.P.D.U. provenant de ce transfert express ou du transfert normal des données. On suppose simplement que les données express sont analysées avant les données normales.

Certains cas de transfert de données express sont sujets à des précautions particulières. On pourrait par exemple recevoir une S.P.D.U. de données express après avoir émis une S.P.D.U. de demande de création de connexion (S.P.D.U. CN), et ne pas encore avoir reçu la réponse (S.P.D.U. AC). Afin de résoudre ces problèmes, trois primitives ont été définies. La première

consiste à sauver la S.P.D.U. dans une file, afin de la traîter ultérieurement. La seconde consiste à supprimer la file (en cas de coupure de connexion, de resynchronisation...). La troisième est plutôt un "état" dans lequel les S.P.D.U. entrantes sont extraites de la file, plutôt que des T.S.D.U. provenant de la couche transport.

III.5. TABLES D'ETATS.

EVENEMENT	ETATS															
	STA01	STA01A	STA01B	STA01C	STA02A	STA03	STA04A	STA04B	STA05A	STA05B	STA05C	STA06	STA08	STA09
S-CON-REQ	1				2									
S-CON-RSP													3	
S-DT-REQ														4
S-EX-REQ														5
S-TD-REQ														6
S-MAJ-REQ														
S-MAJ-RSP														
S-AE-REQ														
S-AE-RSP														
S-MIN-REQ														
S-MIN-RSP														7
S-RS-REQ								8						9
S-RS-RSP														
S-AD-REQ									10	10				
S-AD-RSP									11	11				
S-AI-REQ														
S-AI-RSP														
S-AR-REQ														
S-AS-REQ														
S-CD-REQ														
S-CD-RSP														

+ + + + + + + + + + + + + + +

EVENEMENT	ETATS																	
	STA10A	STA10B	STA11A	STA11B	STA11C	STA15A	STA15B	STA15C	STA16	STA18	STA19	STA20	STA21	STA22	STA37			
S-CON-REQ																		
S-CON-RSP																		
S-DT-REQ	12+	13+					14+								15+			
S-EX-REQ	16+	17+					18+								19+			
S-TD-REQ	20+	21+					22+								23+			
S-MAJ-REQ							24+								25+			
S-MAJ-RSP	26+						27+											
S-AE-REQ							28+								29+			
S-AE-RSP		30+																
S-MIN-REQ							31+								32+			
S-MIN-RSP	33+	34+					35+								36+			
S-RS-REQ	37+	37+	38+			39+	40+							37+	41+			
S-RS-RSP			42+															
S-AD-REQ			43+			43+	44+					43+			43+			
S-AD-RSP					44+													
S-AI-REQ			45+			45+	46+					45+			45+			
S-AI-RSP					47+													
S-AR-REQ															48+			
S-AS-REQ															49+			
S-CD-REQ															50+			
S-CD-RSP														51+				

EVENEMENT	ETATS															
	STA01	STA01A	STA01B	STA01C	STA02A	STA03	STA04	STA04A	STA04B	STA05	STA05B	STA05C	STA06	STA08	STA09
S-CG-REQ															
S-GT-REQ							52	53							
S-PT-REQ															54
S-UER-REQ															55
S-REL-REQ															56
S-REL-RSP															57
S-UAB-REQ			58	59	59	59	59	59	59	59	59	59	59	59	59
T-CON-IND	60	///	///	///	///	///	///	///	///	///	///	///	///	///	///
T-CON-CNF	///	///	61	///	///	///	///	///	///	///	///	///	///	///	///
T-DIS-IND	///	62	63	64	63	63	63	63	63	63	63	63	63	63	63
TIMER	///	58	///	///	///	///	///	///	///	///	///	///	///	///	///

EVENEMENT	ETATS																	
	STA10A	STA10B	STA11A	STA11B	STA11C	STA15A	STA15B	STA15C	STA16	STA18	STA19	STA20	STA21	STA22	STA37
S-CG-REQ															65
S-GT-REQ	66	67											68		69
S-PT-REQ	70	71					72							73	74
S-UER-REQ	75	75					76								75
S-REL-REQ							77								78
S-REL-RSP															
S-UAB-REQ	59	59	59	59	59	59	59	59		59	59	59	59	59	59
T-CON-IND	///	///	///	///	///	///	///	///	///	///	///	///	///	///	///
T-CON-CNF	///	///	///	///	///	///	///	///	///	///	///	///	///	///	///
T-DIS-IND	63	63	63	63	63	63	63	63	63	63	63	63	63	63	63
TIMER	///	///	///	///	///	///	///	///	58	///	///	///	///	///	///

EVENEMENT	ETATS																			
	ST A0 1	ST A0 1A	ST A0 1B	ST A0 1C	ST A0 2A	ST A0 3	ST A0 4	ST A0 4A	ST A0 4B	ST A0 5A	ST A0 5B	ST A0 5C	ST A0 6	ST A0 8	ST A0 9
SPDU AC	///	79	///	58	80										
SPDU CN	///	58	///	81											
SPDU RF	///	79	///	58	82										
SPDU DT	///	79	///	58		83	84	85	86	87	88	89			
SPDU EX	///	79	///	58	90	91	92	93	94	95	96	97			
SPDU TD	///	79	///	58		98	99	100	101	102	103	104			
SPDU MAA	///	79	///	58			105	106	107	108	109	110			
SPDU MAP	///	79	///	58					111			112			
SPDU AE	///	79	///	58					113			114			
SPDU MIA	///	79	///	58		115	116	117	118	119	120	121			
SPDU MIP	///	79	///	58					122			123			
SPDU PR-MAA	///	79	///	58			124	124	107	108	109				
SPDU PR-RA	///	79	///	58					125	125	125	126			
SPDU PR-RS	///	79	///	58		127	127	127	128	108	109	126			
SPDU RA	///	79	///	58					129						
SPDU RS	///	79	///	58		130	131	131	132			133			
SPDU AD	///	79	///	58					134			135			
SPDU ADA	///	79	///	58							136				
SPDU AI	///	79	///	58					137			138			
SPDU AIA	///	79	///	58						139					

EVENEMENT	ETATS																					
	STA10A	STA10B	STA11A	STA11B	STA11C	STA15A	STA15B	STA15C	STA16	STA18	STA19	STA20	STA21	STA22	STA23	STA71						
SPDU AC									140													
SPDU CN									140													
SPDU RF									140													
SPDU DT						141	142	143	140		144					145						
SPDU EX						146		147	140	148	149					150						
SPDU TD						151	152	153	140		154					155						
SPDU MAA						156	127	128	140													
SPDU MAP							157	158	140		159					160						
SPDU AE							161		140		162					163						
SPDU MIA						164	165	166	140		167					168						
SPDU MIP							169	170	140		171					172						
SPDU PR-MAA									140													
SPDU PR-RA									140													
SPDU PR-RS	127	127				173		174	140	175	127	127				127						
SPDU RA							127	176	140													
SPDU RS	131						177		140		131	131				178						
SPDU AD	134	134					179		140		134	134				134						
SPDU ADA								180	140													
SPDU AI	137	137					181		140		137	137				137						
SPDU AIA								182	140													

EVENEMENT	ETATS																			
	STA01	STA01A	STA01B	STA01C	STA02A	STA03	STA04	STA04A	STA04B	STA05	STA05A	STA05B	STA05C	STA06	STA08	STA09
SPDU AR	///	79	///	58												
SPDU AS	///	79	///	58												
SPDU CD	///	79	///	58												
SPDU CDA	///	79	///	58												
SPDU ED	///	79	///	58		183	184	184	185	186	187	188				
SPDU ER	///	79	///	58		189	190	190	185	186	187	188				
SPDU GT	///	79	///	58			191	192	193	194	195	196				
SPDU GTA	///	79	///	58												
SPDU GTC	///	79	///	58												
SPDU PT	///	79	///	58		197	198	199	200	201	202	203				
SPDU DN	///	79	///	58		204									205	
SPDU FN	///	79	///	58		206			207	208	209	210				
SPDU NF	///	79	///	58		211										
SPDU AA	///	212	///	58												
SPDU AB	///	213	///	214	215	215	215	215	215	215	215	215	215	215	215	215

EVENEMENT	ETATS																					
	STA10A	STA10B	STA11A	STA11B	STA11C	STA15A	STA15B	STA15C	STA16	STA18	STA19	STA20	STA21	STA22	STA27	STA12	STA13	STA14	STA17	STA23	STA24	STA25	STA26
SPDU AR							216		140				221							217			
SPDU AS							218		140											219			
SPDU CD									140											220			
SPDU CDA									140				221										
SPDU ED									140		222									223			
SPDU ER									140		224									225			
SPDU GT						226	227	228	140	229	230	230								231			
SPDU GTA									140	232													
SPDU GTC									140											233			
SPDU PT						234	235	236	140	237	238		239							240			
SPDU DN									140														
SPDU FN							241		140		242									243			
SPDU NF						244			140														
SPDU AA									245														
SPDU AB	215	215	215	215	215	215	215	215	245	215	215	215	215	215	215	215	215	215	215	215	215	215	215

III.6. LISTE DES ACTIONS

```
1: T-CONREQ,
   VTCA:=faux,
   ==) STA01B;

2: si VTCA est faux :
      T-DT-REQ(CN),
      ==) STA02A;

3: si RESULT = 0 (* connexion acceptée *) :
      T-DT-REQ(AC),
      V(A):=NUM,
      V(M):=NUM,
      V(R):=0,
      VCOLL:=faux,
      VRSP:=no,
      ==) STA713;
   sinon :
      si TEXP (* données express *) :
         T-DT-REQ(RF-nr),
         start TIMER,
         ==) STA16;
      si non TEXP :
         T-DT-REQ(RF-r),
         ==) STA01C;

4: si FU(FD) et non VCOLL :
      T-DT-REQ(DT),
      ==) STA09;

5: si FU(EX) et non VCOLL :
      T-EX-REQ(EX),
      ==) STA09;

6: si FU(TD) et non VCOLL :
      T-DT-REQ(TD),
      ==) STA09;

7: si VSC et FU(SY) et (non FU(ACT) ou VACT)
   et NUM compris entre V(M) et V(A) :
      T-DT-REQ(MIA),
      V(A):=NUM+1,
      ==) STA09;

8: si FU(RESYN) :
      case RSYN of
        ABANDON : T-EX-REQ(PR-RS),
                  T-DT-REQ(RS-A),
                  VRSP:=A,
                  ==) STA05A;
        RESET : T-EX-REQ(PR-RS),
```

```
                    T-DT-REQ(RS-R),
                    VRSP:=R,
                    VRSPNB:=NUM,
                    ==) STA05A;

 9:si FU(RESYN) et non FU(ACT) :
      case RSYN of
        ABANDON : T-EX-REQ(PR-RS),
                  T-DT-REQ(RS-A),
                  VRSP:=A,
                  ==) STA05A;
        SET : T-EX-REQ(PR-RS),
              T-DT-REQ(RS-S),
              VRSP:=S,
              ==) STA05A;

        RESET : si NUM plus grand que V(R) :
                  T-EX-REQ(PR-RS),
                  T-DT-REQ(RS-R),
                  VRSP:=R,
                  VRSPNB:=num,
                  ==) STA05A;

10:si FU(ACT) et VACT et II(MA) :
      T-EX-REQ(PR-RS),
      T-DT-REQ(AD),
      VRSP:=DSC,
      ==) STA05C;

11:si FU(ACT) et VACT et II(MA) :
      T-EX-REQ(PR-RS),
      T-DT-REQ(AI),
      VRSP:=INT,
      ==) STA05B;

12:si I(DK) :
      T-DT-REQ(DT),
      ==) STA10A;

13:si I(DK) :
      T-DT-REQ(DT),
      ==) STA10B;

14:si I(DK) :
      ==) STA15B;

15:si I(DK) :
      T-DT-REQ(DT),
      ==) STA713;

16:si FU(EX) :
      T-EX-REQ(EX),
      ==) STA10A;

17:si FU(EX) :
```

```
      T-EX-REQ(EX),
      ==) STA10B;

18:si FU(EX) :
      ==) STA15B;

19:si FU(EX) :
      T-EX-REQ(EX),
      ==) STA713;

20:si FU(TD) :
      T-DT-REQ(TD),
      ==) STA10A;

21:si FU(TD) :
      T-DT-REQ(TD),
      ==) STA10B;

22:si FU(TD) :
      ==) STA15B;

23:si FU(TD) :
      T-DT-REQ(TD),
      ==) STA713;

24:si (non FU(ACT) ou VACT) et I(DK)
   et I(MI) et II(MA) :
      ==) STA15B

25:si (non FU(ACT) ou VACT) et I(DK)
   et I(MI) et II(MA) :
      T-DT-REQ(MAP),
      VNEXTACT:=vrai,
      si VSC :
         VSC:=faux,
         V(A):=V(M);
      V(M):=V(M)+1,
      ==) STA04A;

26:si NUM = V(M)-1 :
      T-EX-REQ(PR-MAA),
      T-DT-REQ(MAA),
      VACT:=VNEXTACT,
      V(R):=V(M),
      V(A):=V(M),
      ==) STA713;

27:si NUM = V(M)-1 :
     ==) STA15B;

28:si FU(ACT) et VACT et I(DK)
   et I(MI) et II(MA) :
     ==) STA15B;

29:si FU(ACT) et VACT et I(DK)
```

```
   et I(MI) et II(MA) :
      T-DT-REQ(AE),
      VNEXTACT:=faux,
      si VSC :
         VSC:=faux,
         V(A):=V(M);
      V(M):=V(M)+1,
      ==) STA04B;

30:si NUM = V(M) - 1 :
      T-EX-REQ(PR-MAA),
      T-DT-REQ(AEA),
      VACT:=VNEXTACT,
      V(R):=V(M),
      V(A):=V(M),
      ==) STA713;

31:si (non FU(ACT) ou VACT) et I(DK)
   et II(MI) :
      ==) STA15B;

32:si (non FU(ACT) ou VACT) et I(DK)
   et II(MI) :
      T-DT-REQ(MIP),
      si VSC :
         VSC:=faux,
         V(A):=V(M);
      V(M):=V(M)+1,
      ==) STA713;

33: si VSC et FU(SY) et (non FU(ACT) ou VACT)
    et NUM compris entre V(M) et V(A) :
       T-DT-REQ(MIA),
       V(A):=NUM+1,
       ==) STA10A;

34: si VSC et FU(SY) et (non FU(ACT) ou VACT)
    et NUM compris entre V(M) et V(A) :
       T-DT-REQ(MIA),
       V(A):=NUM+1,
       ==) STA10B;

35: si VSC et FU(SY) et (non FU(ACT) ou VACT)
    et NUM compris entre V(M) et V(A) :
       ==) STA15B;

36: si VSC et FU(SY) et (non FU(ACT) ou VACT)
    et NUM compris entre V(M) et V(A) :
       T-DT-REQ(MIA),
       V(A):=NUM+1,
       ==) STA713;

37:si FU(RESYN) :
      case RSYN of
        ABANDON : T-EX-REQ(PR-RS),
```

```
                    T-DT-REQ(RS-A),
                    VRSP:=A,
                    ==) STA05A;
          SET : si FU(SY) ou FU(MA) :
                    T-EX-REQ(PR-RS),
                    T-DT-REQ(RS-S),
                    VRSP:=S,
                    ==) STA05A;
          RESET : si FU(SY) ou FU(MA)
                   et NUM plus grand V(R) :
                      T-EX-REQ(PR-RS),
                      T-DT-REQ(RS-R),
                      VRSP:=R,
                      VRSNB:=NUM,
                      ==) STA05A;

38:si non SPMWINNER :
     case RSYN of
       ABANDON : T-EX-REQ(PR-RS),
                 T-DT-REQ(RS-A),
                 VRSP:=A,
                 ==) STA05A;
       SET : T-EX-REQ(PR-RS),
             T-DT-REQ(RS-S),
             VRSP:=S,
             ==) STA05A;
       RESET : si NUM plus grand V(R) :
                 T-EX-REQ(PR-RS),
                 T-DT-REQ(RS-R),
                 VRSP:=R,
                 VRSNB:=NUM,
                 ==) STA05A;

39:si FU(RESYN) et non FU(ACT) ou VNEXTACT :
     case RSYN of
       ABANDON : T-EX-REQ(PR-RS),
                 T-DT-REQ(RS-A),
                 VRSP:=A,
                 ==) STA05A;
       SET : T-EX-REQ(PR-RS),
             T-DT-REQ(RS-S),
             VRSP:=S,
             ==) STA05A;
       RESET : si NUM plus grand V(R) :
                 T-EXREQ(PR-RS),
                 T-DT-REQ(RS-R),
                 VRSP:=R,
                 VRSPNB:=NUM,
                 ==) STA05A;

40:si FU(RESYN) et VRSP=NO :
     case RSYN of
       ABANDON : T-EX-REQ(PR-RS),
                 T-DT-REQ(RS-A),
                 VRSP:=A,
```

```
                ==) STA05A;
        SET : si FU(SY) ou FU(MA) :
                T-EX-REQ(PR-RS),
                T-DT-REQ(RS-S),
                VRSP:=S,
                ==) STA05A;
        RESET : si FU(SY) ou FU(MA)
                et NUM plus grand V(R) :
                   T-EX-REQ(PR-RS),
                   T-DT-REQ(RS-R),
                   VRSP:=R,
                   VRSNB:=NUM,
                   ==) STA05A;

41:si FU(RESYN) et (non FU(ACT) ou VACT) :
      case RSYN of
        ABANDON : T-EX-REQ(PR-RS),
                  T-DT-REQ(RS-A),
                  VRSP:=A,
                  ==) STA05A;
        SET : si FU(SY) ou FU(MA) :
                T-EX-REQ(PR-RS),
                T-DT-REQ(RS-S),
                VRSP:=S,
                ==) STA05A;
        RESET : si FU(SY) ou FU(MA)
                et NUM plus grand V(R) :
                   T-EX-REQ(PR-RS),
                   T-DT-REQ(RS-R),
                   VRSP:=R,
                   VRSNB:=NUM,
                   ==) STA05A

42:si (VRSP=R et NUM=VRSPNB) ou VRSP=S ou VRSP=A :
      T-EX-REQ(PR-RA),
      T-DT-REQ(RA),
      V(A):=NUM,
      V(M):=NUM,
      si VRSP=A ou VRSP=S :
         V(R):=0;
      mettre à jour la position des jetons,
      ==) STA713;

43:si FU(ACT) et VACT et II(MA) :
      T-EX-REQ(PR-RS),
      T-DT-REQ(AD),
      VRSP:=DSC,
      ==) STA05C;

44:si FU(ACT) et VACT et II(MA) et VRSP=NO :
      T-EX-REQ(PR-RS),
      T-DT-REQ(AD),
      VRSP:=DSC,
      ==) STA05C;
```

```
44:T-EX-REQ(PR-RA),
   T-DT-REQ(ADA),
   VACT:=faux,
   VRSP:=NO,
   OWNED(DK):=vrai,
   OWNED(MI):=vrai,
   OWNED(MA):=vrai,
   OWNED(NR):=vrai,
   ==) STA713;

45:si FU(ACT) et VACT et II(MA) :
      T-EX-REQ(PR-RS),
      T-DT-REQ(AI),
      VRSP:=INT,
      ==) STA05C;

46:si FU(ACT) et VACT et II(MA) et VRSP=NO :
      T-EX-REQ(PR-RS),
      T-DT-REQ(AI),
      VRSP:=INT,
      ==) STA05C,

47:T-EX-REQ(PR-RA),
   T-DT-REQ(AIA),
   VACT:=faux,
   VRSP:=NO,
   OWNED(DK):=vrai,
   OWNED(MI):=vrai,
   OWNED(MA):=vrai,
   OWNED(NR):=vrai,
   ==) STA713;

48:si FU(ACT) et non VACT et I(DK) et I(MI) et II(MA) :
      T-DT-REQ(AR),
      VACT:=vrai,
      V(A):=NUM + 1,
      V(M):=V(A),
      V(R):=1,
      ==) STA713;

49:si FU(ACT) et non VACT et I(DK)
   et I(MI) et II(MA) :
      T-DT-REQ(AS),
      VACT:=vrai,
      V(A):=1,
      V(M):=1,
      V(R):=1,
      ==) STA713;

50:si FU(CD) et FU(ACT) et non VACT
   et I(DK) et I(MI) et II(MA) :
      T-DT-REQ(CD),
      ==) STA21;

51:T-DT-REQ(CDA),
```

```
    ==) STA713;

  52:si tous les jetons cédés sont disponibles et
possédés :
       T-DT-REQ(GT),
       mettre à jour la position des jetons,
       ==) STA04A;

  53:si tous les jetons cédés sont disponibles et
possédés :
       T-DT-REQ(GT),
       mettre à jour la position des jetons,
       ==) STA04B;

  54:si tous les jetons reçus sont disponibles :
       T-DT-REQ(PT),
       ==) STA09;

  55:si FU(EXCEP) et (non FU(ACT) ou VACT) et AA(DK) :
       T-DT-REQ(ED),
       ==) STA19;

  56:s'il n'existe aucun jeton disponible
     et (non FU(ACT) ou non VACT) :
       T-DT-REQ(FN-nr),
       VTRR:=faux,
       VCOLL:=vrai,
       ==) STA09;

  57: si RESULT = 0 (* libération acceptée *) :
        si non VTRR :
           T-DT-REQ(DN),
           start TIMER,
           ==) STA16;
        si VTRR et non VCOLL :
           T-DT-REQ(DN),
           ==) STA01C;
        si VTRR et VCOLL :
           T-DT-REQ(DN),
           VTRR:=faux,
           ==) STA03;

  58:T-DIS-REQ,
     ==) STA01;

  59:si TEXP :
       T-EX-REQ(AB-nr),
       start TIMER,
       ==) STA16;
     si non TEXP :
       T-DT-REQ(AB-r),
       start TIMER,
       ==) STA01A;

  60:T-CON-RSP,
```

```
      VTCA:=vrai,
      ==) STA01C;

   61:T-DT-REQ(CN),
      ==) STA02A;

   62:stop TIMER,
      ==) STA01;

   63:S-PAB-IND,
      ==) STA01;

   64:==) STA01;

   65:si FU(ACT) et non VACT
      et aucun des jetons disponibles n'est possédé :
         T-DT-REQ(GTC),
         OWNED(DK):=vrai,
         OWNED(MI):=vrai,
         OWNED(MA):=vrai,
         OWNED(NR):=vrai,
         ==) STA18;

   66:si tous les jetons cédés sont disponibles et
possédés :
         T-DT-REQ(GT),
         mettre à jour la position des jetons,
         ==) STA10A;

   67:si tous les jetons cédés sont disponibles et
possédés :
         T-DT-REQ(GT),
         mettre à jour la position des jetons,
         ==) STA10B;

   68:si tous les jetons cédés sont disponibles et
possédés :
         T-DT-REQ(GT),
         mettre à jour la position des jetons,
         si DK est parmi les jetons :
            ==)STA713;
         si DK n'est pas parmi les jetons :
            ==) STA20;

   69:si tous les jetons cédés sont disponibles et
possédés :
         T-DT-REQ(GT),
         mettre à jour la position des jetons,
         ==) STA713;

   70:si tous les jetons demandés sont disponibles :
         T-DT-REQ(PT),
         ==) STA10A;

   71:si tous les jetons demandés sont disponibles :
```

```
        T-DT-REQ(PT),
        ==) STA10B;

72:si tous les jetons demandés sont disponibles :
        ==) STA15B;

73:si tous les jetons demandés sont disponibles :
        T-DT-REQ(PT),
        ==) STA22;

74:si tous les jetons demandés sont disponibles :
        T-DT-REQ(PT),
        ==) STA713;

75: si FU(EXCEP) et (non FU(ACT) ou VACT) et AA(DK) :
         T-DT-REQ(ED),
         ==) STA19;

76: si FU(EXCEP) et (non FU(ACT) ou VACT) et AA(DK) :
         ==) STA15B;

77:si tous les jetons disponibles sont possédés :
        ==) STA15B;

78:si tous les jetons disponibles sont possédés :
        si non VTCA et non TEXP :
           T-DT-REQ(FN-r),
           VTRR:=vrai,
           ==) STA03;
        si VTCA ou TEXP :
           T-DT-REQ(FN-nr),
           VTRR:=faux,
           ==) STA03;

79: ==) STA01A;

80:S-CON-CNF,
   V(A):=NUM,
   V(M):=NUM,
   V(R):=0,
   VCOLL:=faux,
   VRSP:=no,
   ==) STA713,
   vider la file;

81:si VTCA :
        S-CON-IND,
        ==) STA08;
   si non VTCA :
        T-DIS-REQ,
        ==) STA01;

82:S-CONCNF,
   T-DIS-REQ,
   ==) STA01;
```

```
83:si A(DK) et non VCOLL :
     S-DT-IND,
     ==) STA03;

84:si FU(FD) :
      S-DT-IND,
      ==) STA04A;

85:si FU(FD) :
      S-DT-IND,
      ==) STA04B;

86:si A(DK) :
      ==) STA05A;

87:si A(DK) :
      ==) STA05B;

88:si A(DK) :
      ==) STA05C;

89:si A(DK) :
      ==) STA06;

90:sauver dans la file,
   ==) STA02A;

91:si FU(EX) et non VCOLL :
      S-EX-IND,
      ==) STA03;

92:si FU(EX) :
      S-EX-IND,
      ==) STA04A;

93:si FU(EX) :
      S-EX-IND,
      ==) STA04B;

94:si FU(EX) :
      ==) STA05A;

95:si FU(EX) :
      ==) STA05B;

96:si FU(EX) :
      S-EX-IND,
      ==) STA05C;

97:sauver dans la file,
   ==) STA06;

98:si FU(TD) et non VCOLL :
      S-TD-IND,
```

```
       ==) STA03;

 99:si FU(TD) :
       S-TD-IND,
       ==) STA04A;

100:si FU(TD) :
       S-TD-IND,
       ==) STA04B;

101:si FU(TD) :
       ==) STA05A;

102:si FU(TD) :
       ==) STA05B;

103:si FU(TD) :
       ==) STA05C;

104:si FU(TD) :
       ==) STA06;

105:si non TEXP et NUM=V(M)-1 :
       S-MAJ-CNF,
       VACT:=VNEXTACT,
       V(R):=V(M),
       V(A):=V(M),
       ==) STA713;

106:si non TEXP et NUM=V(M)-1 :
       S-AE-CNF,
       VACT:=VNEXTACT,
       V(R):=V(M),
       V(A):=V(M),
       ==) STA713;

107: ==) STA05A;

108: ==) STA05B;

109: ==) STA05C;

110: ==) STA06;

111:si (non FU(ACT) ou VACT) et A(DK)
    et A(MI) et AA(MA) :
       ==) STA05A;

112:si (non FU(ACT) ou VACT) et A(DK)
    et A(MI) et AA(MA) :
       ==) STA06;

113:si FU(ACT) et VACT et A(DK)
    et A(MI) et AA(MA) :
       ==) STA05A;
```

```
114:si FU(ACT) et VACT et A(DK)
    et A(MI) et AA(MA) :
       ==) STA06;

115:si (non FU(ACT) ou VACT)
    et FU(SY) et non VSC
    et NUM compris entre V(M) et V(A) :
       S-MIN-CNF,
       V(A):=NUM+1,
       ==)STA03;

116:si (non FU(ACT) ou VACT) et FU(SY) et non VSC
    et NUM compris entre V(M) et V(A) :
       S-MIN-CNF,
       V(A):=NUM+1,
       ==)STA04A;

117:si (non FU(ACT) ou VACT) et FU(SY) et non VSC
    et NUM compris entre V(M) et V(A) :
       S-MIN-CNF,
       V(A):=NUM+1,
       ==)STA04B;

118:si (non FU(ACT) ou VACT) et FU(SY) et non VSC :
       ==) STA05A;

119:si (non FU(ACT) ou VACT) et FU(SY) et non VSC :
       ==) STA05B;

120:si (non FU(ACT) ou VACT) et FU(SY) et non VSC :
       ==) STA05C;

121:si (non FU(ACT) ou VACT) et FU(SY) et non VSC :
       ==) STA06;

122:si (non FU(ACT) ou VACT) et A(DK) et AA(MI) :
       ==) STA05A;

123:si (non FU(ACT) ou VACT) et A(DK) et AA(MI) :
       ==) STA06;

124: ==) STA15A;

125: ==) STA15C;

126: sauver dans la file,
     ==) STA06;

127: ==) STA15B;

128: ==) STA06;

129:si FU(RESYN) et non TEXP et ((VRSP=R et NUM=VRSPNB)
      ou VRSP=S ou VRSP=A) :
```

```
        S-RS-CNF,
        V(A):=NUM,
        V(M):=NUM,
        si VRSP=A ou VRSP=S :
           V(R)=0;
        VRSP=NO,
        ==) STA713;

130:si FU(RESYN) et non TEXP et NON FU(ACT) :
        case RSYN of
          RS-a : V(M):=max(V(M),NUM),
                 S-RS-IND,
                 VRSP:=A,
                 ==) STA11A;
          RS-s : S-RS-IND,
                 VRSP:=S,
                 ==) STA11A;
          RS-r : si NUM plus grand V(R) :
                    S-RS-IND,
                    VRSP:=R,
                    VRSPNB:=NUM,
                    ==) STA11A;

131:si FU(RESYN) et TEXP :
        case RSYN of
          RS-a : V(M):=max(V(M),NUM),
                 S-RS-IND,
                 VRSP:=A,
                 ==) STA11A;
          RS-s : S-RS-IND,
                 VRSP:=S,
                 ==) STA11A;
          RS-r : si NUM plus grand V(R) :
                    S-RS-IND,
                    VRSP:=R,
                    VRSPNB:=NUM,
                    ==) STA11A;

132:si FU(RESYN) et non TEXP et SPMWINNER :
        case RSYN of
          RS-a : V(M):=max(V(M),NUM),
                 S-RS-IND,
                 VRSP:=A,
                 ==) STA11A;
          RS-s : S-RS-IND,
                 VRSP:=S,
                 ==) STA11A;
          RS-r : si NUM plus grand V(R) :
                    S-RS-IND,
                    VRSP:=R,
                    VRSPNB:=NUM,
                    ==) STA11A;

133:si SPMWINNER :
        case RSYN of
```

```
          RS-a : V(M):=max(V(M),NUM),
                 S-RS-IND,
                 VRSP:=A,
                 ==) STA11A;
          RS-s : S-RS-IND,
                 VRSP:=S,
                 ==) STA11A;
          RS-r : si NUM plus grand V(R) :
                    S-RS-IND,
                    VRSP:=R,
                    VRSPNB:=NUM,
                    ==) STA11A;

134:si FU(ACT) et non TEXP et AA(MA) :
       S-AD-IND,
       VRSP:=DSC,
       ==) STA11C;

135:si FU(ACT) et TEXP et AA(MA) :
       S-AD-IND,
       VRSP:=DSC,
       ==) STA11C;

136:si FU(ACT) et non TEXP :
       S-AD-CNF,
       VACT:=faux,
       VRSP:=NO,
       OWNED(DK):=vrai,
       OWNED(MI):=vrai,
       OWNED(MA):=vrai,
       OWNED(NR):=vrai,
       ==) STA713;

137:si FU(ACT) et non TEXP et AA(MA) :
       S-AI-IND,
       VRSP:=INT,
       ==) STA11B;

138:si FU(ACT) et TEXP et AA(MA) :
       S-AI-IND,
       VRSP:=INT,
       ==) STA11B;

139:si FU(ACT) et non TEXP :
       S-AI-CNF,
       VACT:=faux,
       VRSP:=NO,
       OWNED(DK):=vrai,
       OWNED(MI):=vrai,
       OWNED(MA):=vrai,
       OWNED(NR):=vrai,
       ==) STA713;

140: ==) STA16;
```

```
141:si FU(FD) :
       S-DT-IND,
       ==) STA15A;

142:si A(DK) :
       ==) STA15B;

143:si A(DK) :
       ==) STA15C;

144: ==) STA19;

145:si A(DK) :
       S-DT-IND,
       ==) STA713;

146:si FU(EX) :
       sauver dans la file,
       ==) STA15A;

147:si FU(EX) :
       sauver dans la file,
       ==) STA15C;

148:si FU(EX) :
       S-EX-IND,
       ==) STA18;

149:si FU(EX) :
       ==) STA19;

150:si FU(EX) :
       S-EX-IND,
       ==) STA713;

151:si FU(TD) :
       S-TD-IND,
       ==) STA15A

152:si FU(TD) :
       ==) STA15B;

153:si FU(TD) :
       ==) STA15C;

154:si FU(TD) :
       ==) STA19;

155:si FU(TD) :
       S-TD-IND,
       ==) STA713;

156:si NUM=V(M)-1 :
       si non FU(ACT) ou non VNEXTACT :
          S-MAJ-CNF;
```

```
        si FU(ACT) et VNEXTACT :
           S-AE-CNF;
        VACT:=VNEXTACT,
        V(R):=V(M),
        V(A):=V(M),
        ==) STA713,
        vider la file;

157:si (non FU(ACT) ou VACT) et A(DK)
    et A(MI) et AA(MA) :
        ==) STA15B;

158:si (non FU(ACT) ou VACT) et A(DK)
    et A(MI) et AA(MA) :
        ==) STA15C;

159:si (non FU(ACT) ou VACT)
    et A(DK) et A(MI) :
      et AA(MA) et NUM=V(M)
        si non VSC :
           V(A):=V(M);
        V(M):=V(M)+1,
        ==) STA19;

160:si (non FU(ACT) ou VACT) et A(DK) et A(MI)
    et AA(MA) et NUM=V(M) :
        S-MAJ-IND,
        VNEXTACT:=faux,
        si non VSC :
               V(A):=V(M);
        V(M):=V(M)+1,
        ==) STA10A;

161:si FU(ACT) et VACT et A(DK)
    et A(MI) et AA(MA) :
        ==) STA15B;

162:si FU(ACT) et VACT et A(DK) et A(MI)
    et AA(MA) et NUM=V(M) :
        si non VSC :
               V(A):=V(M);
        V(M):=V(M)+1,
        ==) STA19;

163:si (non FU(ACT) ou VACT) et A(DK) et A(MI)
    et AA(MA) et NUM=V(M) :
        S-MAJ-IND,
        VNEXTACT:=faux,
        si non VSC :
               V(A):=V(M);
        V(M):=V(M)+1,
        ==) STA10A;

164: si (non FU(ACT) ou VACT) et FU(SY) et non VSC
     et NUM compris entre V(M) et V(A) :
```

```
        S-MIN-CNF,
        V(A):=NUM+1,
        ==) STA15A;

165: si (non FU(ACT) ou VACT) et FU(SY) et non VSC :
        ==) STA15B;

166: si (non FU(ACT) ou VACT) et FU(SY) et non VSC :
        ==) STA15C;

167: si (non FU(ACT) ou VACT) et FU(SY) et non VSC
     et NUM compris entre V(M) et V(A) :
        V(A):=NUM+1,
        ==) STA19;

168: si (non FU(ACT) ou VACT) et FU(SY) et non VSC
     et NUM compris entre V(M) et V(A) :
        S-MIN-CNF,
        V(A):=NUM+1,
        ==) STA713;

169:si (non FU(ACT) ou VACT) et A(DK) et AA(MI) :
       ==) STA15B;

170:si (non FU(ACT) ou VACT) et A(DK) et AA(MI) :
       ==) STA15C;

171:si (non FU(ACT) ou VACT) et A(DK)
    et AA(MI) et NUM=V(M) :
       si non VSC :
              VSC:=vrai,
              V(A):=V(M);
       V(M):=V(M)+1,
       ==) STA19;

172:si (non FU(ACT) ou VACT) et A(DK)
    et AA(MI) et NUM=V(M) :
       si non VSC :
              VSC:=vrai,
              V(A):=V(M);
       S-MIN-IND,
       V(M):=V(M)+1,
       ==) STA713;

173:sauver dans la file,
    ==) STA15A;

174:supprimer la file,
    ==) STA15B;

175:sauver dans la file,
    ==) STA18;

176:si FU(RESYN) et TEXP et ((VRSP=R et VRSPNB=NUM)
    ou VRSP=S ou VRSP=A :
```

```
        S-RS-CNF,
        V(A):=NUM,
        V(M):=NUM,
        si VRSP=A ou VRSP=S :
           V(R)=0;
        mettre à jour la position des jetons,
        ==) STA713,
        vider la file;

177:si FU(RESYN) et (non FU(ACT) ou VACT) :
        case RSYN of
          RS-a : V(M):=max(V(M),NUM),
                 S-RS-IND,
                 VRSP:=A,
                 ==) STA11A;
          RS-s : S-RS-IND,
                 VRSP:=S,
                 ==) STA11A;
          RS-r : si NUM plus grand V(R) :
                    S-RS-IND,
                    VRSP:=R,
                    VRSPNB:=NUM,
                    ==) STA11A;

178:si FU(RESYN) et (non FU(ACT) ou VACT)
    et non TEXP :
        case RSYN of
          RS-a : V(M):=max(V(M),NUM),
                 S-RS-IND,
                 VRSP:=A,
                 ==) STA11A;
          RS-s : S-RS-IND,
                 VRSP:=S,
                 ==) STA11A;
          RS-r : si NUM plus grand V(R) :
                    S-RS-IND,
                    VRSP:=R,
                    VRSPNB:=NUM,
                    ==) STA11A;

179:si FU(ACT) et TEXP et AA(MA) :
        S-AD-IND,
        VRSP:=DSC,
        ==) STA11C;

180:si FU(ACT) et TEXP et VRSP=DSC :
        S-AD-CNF,
        OWNED(DK):=vrai,
        OWNED(MI):=vrai,
        OWNED(MA):=vrai,
        OWNED(NR):=vrai,
        VACT:=faux,
        VRSP:=NO,
        ==) STA713,
        vider la file;
```

```
181:si FU(ACT) et TEXP et AA(MA) :
       S-AI-IND,
       VRSP:=INT,
       ==) STA11B

182:si FU(ACT) et TEXP et VRSP=INT :
       S-AI-CNF,
       OWNED(DK):=vrai,
       OWNED(MI):=vrai,
       OWNED(MA):=vrai,
       OWNED(NR):=vrai,
       VACT:=faux,
       VRSP:=NO,
       ==) STA713,
       vider la file;

183:si FU(EXCEP) et non FU(ACT) et II(DK) :
       S-UER-IND,
       ==) STA20;

184:si FU(EXCEP) et FU(HD) :
       S-UER-IND,
       ==) STA20;

185:si FU(EXCEP) et FU(HD) :
       ==) STA05A;

186:si FU(EXCEP) et FU(HD) :
       ==) STA05B;

187:si FU(EXCEP) et FU(HD) :
       ==) STA05C;

188:si FU(EXCEP) et FU(HD) :
       ==) STA06;

189:si FU(EXCEP) et non FU(ACT) et II(DK) :
       S-PER-IND,
       ==) STA20;

190:si FU(EXCEP) et FU(HD) :
       S-PER-IND,
       ==) STA20;

191:si tous les jetons cédés sont disponibles et non
possédés :
    S-GT-IND,
    mettre à jour la position des jetons,
    ==) STA04A;

192:si tous les jetons cédés sont disponibles et non
possédés :
    S-GT-IND,
    mettre à jour la position des jetons,
```

```
    ==) STA04B;

193:si tous les jetons cédés sont disponibles et non
possédés :
    ==) STA05A;

194:si tous les jetons cédés sont disponibles et non
possédés :
    ==) STA05B;

195:si tous les jetons cédés sont disponibles et non
possédés :
    ==) STA05C;

196:si tous les jetons cédés sont disponibles et non
possédés :
    ==) STA06;

197:si tous les jetons demandés sont disponibles :
       S-PT-IND,
       ==) STA03;

198:si tous les jetons demandés sont disponibles :
       S-PT-IND,
       ==) STA04A;

199:si tous les jetons demandés sont disponibles :
       S-PT-IND,
       ==) STA04B;

200:si tous les jetons demandés sont disponibles :
       ==) STA05A;

201:si tous les jetons demandés sont disponibles :
       ==) STA05B;

202:si tous les jetons demandés sont disponibles :
       ==) STA05C;

203:si tous les jetons demandés sont disponibles :
       ==) STA06;

204:S-REL-CNF,
    si VTRR :
       ==) STA01C;
    si non VTRR :
       T-DIS-REQ,
       ==) STA01;

205:si VCOLL :
       S-REL-CNF,
       ==) STA09;

206:si non((non FU(ACT) ou non VACT)
    et aucun jeton n'est disponible) :
```

```
        S-REL-IND,
        VTRR:=faux,
        VCOLL:=vrai,
        ==) STA09;

207:si tous les jetons disponibles ne sont pas possédés
    et (non FU(ACT) ou non VACT) :
        ==) STA05A;

208:si tous les jetons disponibles ne sont pas possédés
    et (non FU(ACT) ou non VACT) :
        ==) STA05B;

209:si tous les jetons disponibles ne sont pas possédés
    et (non FU(ACT) ou non VACT) :
        ==) STA05C;

210:si tous les jetons disponibles ne sont pas possédés
    et (non FU(ACT) ou non VACT) :
        ==) STA06;

211:si FU(NR) :
        S-REL-CNF,
        ==) STA713;

212:stop TIMER,
    ==) STA01C;

213:stop TIMER,
    case AB of
        AB-nr : T-DIS-REQ,
                ==) STA01;
        AB-r  : ==) STA01C;

214:case AB of
      AB-nr : (* pas de réutilisation transport *)
              TDISREQ,
              ==) STA01;
      AB-r  : (* réutilisation connexion transport *)
              si non TEXP :
                 ==) STA01C;
              si TEXP :
                 T-DIS-REQ,
                 ==) STA01;

215:S-UAB-IND,
   case AB of
      AB-nr : (* pas de réutilisation transport *)
              TDISREQ,
              ==) STA01;
      AB-r  : (* réutilisation connexion transport *)
              si non TEXP  :
                    ==) STA01C;
              si TEXP :
                 T-DIS-REQ,
```

```
                ==) STA01;

216:si (FU(ACT) et non VACT) et A(DK)
    et A(MI) et AA(MA) :
       S-AR-IND,
       VACT:=vrai,
       V(A):=NUM+1,
       V(M):=NUM+1,
       V(R):=1,
       ==) STA15B;

217:si (FU(ACT) et non VACT) et A(DK)
    et A(MI) et AA(MA) :
       S-AR-IND,
       VACT:=vrai,
       V(A):=NUM+1,
       V(M):=NUM+1,
       V(R):=1,
       ==) STA713,
       vider la file;

218:si (FU(ACT) et non VACT) et A(DK)
    et A(MI) et AA(MA) :
       S-AS-IND,
       VACT:=vrai,
       V(A):=1,
       V(M):=NUM+1,
       V(R):=1,
       ==) STA15B;

219:si (FU(ACT) et non VACT) et A(DK)
    et A(MI) et AA(MA) :
       S-AS-IND,
       VACT:=vrai,
       V(A):=1,
       V(M):=NUM+1,
       V(R):=1,
       ==) STA713,
       vider la file;

220:si FU(CD) et (FU(ACT) et non VACT)
    et A(DK) et A(MI) et AA(MA) :
       S-CD-IND,
       ==) STA22;

221:S-CD-CNF,
    ==) STA713;

222:si FU(EXCEP) et (non FU(ACT ou VACT) et II(DK) :
       S-UER-IND,
       ==) STA19;

223:si FU(EXCEP) et (non FU(ACT) ou VACT) :
       S-UER-IND,
       si AA(DK) :
```

```
          ==) STA713;
       si II(DK) :
          ==) STA20;

224:si FU(EXCEP) et (non FU(ACT ou VACT) et II(DK) :
       S-PER-IND,
       ==) STA19;

225:si FU(EXCEP) et (non FU(ACT) ou VACT) :
       S-PER-IND,
       si AA(DK) :
          ==) STA713;
       si II(DK) :
          ==) STA20;

226:si tous les jetons cédés sont disponibles et non
possédés :
    S-GT-IND,
    mettre à jour la position des jetons,
    ==) STA15A;

227:si tous les jetons cédés sont disponibles et non
possédés :
    ==) STA15B;

228:si tous les jetons cédés sont disponibles et non
possédés :
    ==) STA15C;

229:si tous les jetons cédés sont disponibles et non
possédés :
    S-GT-IND,
    mettre à jour la position des jetons,
    ==) STA18;

230:si tous les jetons cédés sont disponibles et non
possédés :
    S-GT-IND,
    mettre à jour la position des jetons,
    si DK est dans les jetons cédés :
      ==) STA713;
    si DK n'est pas dans les jetons cédés :
       ==) STA20;

231:si tous les jetons cédés sont disponibles et non
possédés :
    S-GT-IND,
    mettre à jour la position des jetons,
    ==) STA713;

232:==) STA713,
    vider la file;

233:si (FU(ACT) et non VACT)
    et tous les jetons disponibles sont possédés :
```

```
        S-CG-IND,
        T-DT-REQ(GTA),
        OWNED(DK):=faux,
        OWNED(MI):=faux,
        OWNED(MA):=faux,
        OWNED(NR):=faux,
        ==) STA713;

234:si tous les jetons demandés sont disponibles :
        S-PT-IND,
        ==) STA15A

235:si tous les jetons demandés sont disponibles :
        ==) STA15B;

236:si tous les jetons demandés sont disponibles :
        ==) STA15C;

237:si tous les jetons demandés sont disponibles :
        S-PT-IND,
        ==) STA18;

238:si tous les jetons demandés sont disponibles :
        ==) STA19;

239:si tous les jetons demandés sont disponibles :
        S-PT-IND,
        ==) STA21;

240:si tous les jetons demandés sont disponibles :
        S-PT-IND,
        ==) STA713;

241:si tous les jetons disponibles ne sont pas possédés
    et (non FU(ACT) ou non VACT) :
        ==) STA15C;

242:si tous les jetons disponibles ne sont pas possédés
    et (non FU(ACT) ou non VACT) :
        ==) STA19;

243:si tous les jetons disponibles ne sont pas possédés
    et (non FU(ACT) ou non VACT) :
        S-REL-IND,
        VIRR:=faux,
        ==) STA09;

244:si FU(NR) :
        S-REL-CNF,
        ==) STA15B;

245:stop TIMER,
    T-DIS-REQ,
    ==) STA01;
```

A N N E X E IV

D E S C R I P T I O N

D E S P R O T O C O L E S

D E L A C O U C H E

A P P L I C A T I O N : F . T . A . M .

IV.1. VUE GENERALE.

Les tables d'états qui suivent donnent les différentes actions à suivre lors d'évènements provoquant l'activation de l'entité application.

Deux types d'évènements peuvent se produire :

- Des requêtes ou des réponses en provenance de l'utilisateur des services.

- Des indications et des confirmations en provenance de la couche inférieure.

On a fait la distinction entre l'entité initiatrice de la connexion et l'entité réceptrice.

L'entité réceptrice est celle qui sera utilisée avec un gérant de base de données. Rappelons que ce gérant devra se charger de la transformation entre le modèle de gestion des fichiers virtuels, décrits par I.S.O., et la réalisation locale des moyens de gestion de fichiers.

Cette entité a un rôle passif. Suite à des indications lui parvenant de la couche présentation, qu'elle transmet à l'utilisateur des services d'application, elle reçoit de ce dernier des réponses. Les seules exceptions concernent les requêtes d'abandon de transfert et de coupure de connexion, ainsi que le transfert des données lorsque celui-ci prend la direction de l'entité réceptrice vers l'entité initiatrice (cas d'ouverture en lecture).

Dans la description des protocoles par table d'etats, nous supposerons toujours que l'entité reçoit des F.P.D.U., contenues dans les P.S.D.U. ou unitées de données de service de présentation, quelle que soit le type de primitive les transportant (resynchronisation, synchronisation mineure...).

L'entité initiatrice de connexion sera celle qui dirige le transfert des données. Elle n'émet que des requêtes, puis attend la confirmation. Les seules exceptions sont les indication d'abandon de transfert de données ou de coupure de connexion, et les indications de données dans le cas où le fichier a été ouvert en lecture.

Tous les transferts de données sont précédés d'une F.P.D.U. DSTART, indiquant le début d'une séquence de données de l'utilisateur des services d'application. Cette F.P.D.U. est transparente à l'utilisateur des services.

IV.2. SERVICES UTILISES DE LA COUCHE PRESENTATION.

Seuls les éléments du service présentation décrits ci-dessous sont utilisés :

P-CON-REQ	P-CON-IND
P-CON-RSP	P-CON-CNF
P-REL-REQ	P-REL-IND
P-REL-RSP	P-REL-CNF
P-UAB-REQ	P-UAB-IND
	P-PAB-IND
P-MIN-REQ	P-MIN-IND
P-MIN-RSP	P-MIN-CNF
P-RS-REQ(abandon)	P-RS-IND(abandon)
P-RS-REQ(reset)	P-RS-IND(reset)
P-RS-REQ(set)	P-RS-IND(set)
P-RS-RSP	P-RS-CNF
P-GT-REQ(mi)	P-GT-IND(mi)
P-DT-REQ	P-DT-IND
P-DFX-REQ	P-DFX-IND
P-DFX-RSP	P-DFX-CNF
P-DLX-REQ	P-DLX-IND
P-DLX-RSP	P-DLX-CNF
P-SLX-REQ	P-SLX-IND
P-SLX-RSP	P-SLX-CNF

A part les trois derniers types de primitives (DFX: define context, DLX: delete context et SLX: select context) qui sont spécifiques à la couche présentation, toutes les autres primitives sont gérées par la couche session, et ont pour but d'harmoniser le dialogue. Toutes ces primitives peuvent porter des données de l'utilisateur des services, qui seront représentées par le nom de la F.P.D.U.

IV.3. VARIABLES LOCALES.

Un certain nombre de variables locales sont utilisées dans les tables d'états :

* **TRANSPARANCY** : Cette variable booléenne prend la valeur "vrai" quand les données des indications de transfert de données de la couche présentation (P-DT-IND) sont des données de l'utilisateur des services applications, et non des informations de protocole du F.T.A.M. Lorsqu'elle prend la valeur "faux", les F.P.D.U. sont analysées, et leur contenu peut modifier l'état de l'entité.

* **DISCARD** : Cette variable booléenne prend la valeur "vrai" dans le cas où un recouvrement d'erreur a lieu, et que les données préalablement émises ne peuvent plus être prises en compte par l'entité réceptrice. Il faut noter que la couche session prend en charge une partie de ces suppression.

* **RESET** : Cette variable booléenne indique qu'une resynchronisation a eu lieu, et que toute donnée reçue doit être écartée avant que la fin de la resynchronisation n'ait été effectuée.

* **EXPECTED-CHECK** : Il s'agit d'une variable numérique indiquant la liste des points de synchronisation dans un fichier, mise à jour lors de la création d'une connexion ou le recouvrement d'erreurs.

* **NEXT-SYNC** : Cette variable numérique indique le dernier point de synchronisation pour lequel une resynchronisation est autorisée.

* **P-DEFINE** : Cet indicateur booléen sert à indiquer qu'un contexte de présentation doit être défini.

* **P-DELETE** : Cet indicateur sert à mémoriser le besoin de supprimer un contexte de présentation.

* **P-SELECT** : Cet indicateur sert à mémoriser le besoin de sélectionner un autre contexte de présentation.

De plus, un temporisateur est utilisé, pour prévenir les risques d'interblocages. Ce temporisateur est déclenché à chaque requête, et arrêté à chaque confirmation. Les valeurs d'attente sont fonctions du type de procédure, et ne sont pas définies dans les normes.

A l'expiration d'un temporisateur, la connexion est libérée.

IV.4. DESCRIPTION DES ETATS.

Plutôt que de faire la description de tous les états possibles d'une entité, nous décrirons les conventions utilisées dans les tables ci-dessous.

Il y a trois types d'états dans lesquels peut être une entité, différents selon que ce soit l'entité initiatrice de connexion ou l'entité réceptrice.

- Pour l'entité initiatrice :

  - Soit elle attend une requête de l'utilisateur des services. Dans ce cas, l'état correspond à une des phases de la connexion (voir 11.1.3.). On retrouve notamment :

  CONNECTED (la connexion est créée).

  SELECTED (un fichier est sélectionné).

  DXFRIDLE (de l'anglais data transfer idle, le fichier est ouvert).

  READ (phase de transfert de données).

  WRITE (phase de transfert de donéées).

  - Soit l'entité attend une indication ou une confirmation de la couche transport. Dans ce cas, le nom de l'état est préfixé par "P-" et postifxé par "-PD" (de l'anglais pending, attendant).

  exemples :

  P-CONNECT-PD (attente de confirmation de connexion présentation).

  P-RELEASE-PD (attente de confirmation de déconnexion présentation).

  - Soit l'entité attend une F.P.D.U. de réponse de l'entité réceptrice, auquel cas le type de F.P.D.U. est postifixé pr "-PD" (de l'anglais pending).

- Pour l'entité réceptrice :

- Soit elle se trouve dans une des phases de connexion, comme l'entité initiatrice (voir ci-dessus), auquel cas l'entité attend une F.P.D.U. susceptible de la faire changer de phase.

- Soit elle attend une réponse de l'utilisateur des services, suite à une indication émise, et l'état est le nom de la réponse attendue, postfixé par "-PD".

exemple :

SELECT-PD (attente de réponse de sélection).

READ-PD (attente de réponse de lecture).

- Soit elle attend une confirmation de la couche présentation, auquel cas la dénomination de l'état est le nom de la primitive attendue, préfixé par "-P" et postfixé par "-PD".

exemple:

P-DLXCF-PD (attente de confirmation de suppression de contexte).

Dans notre réalisation, nous avons procédé comme suit : Deux vecteurs booléens indiquent les évènements pouvant se produire. Un des deux concerne les requêtes et les réponses (selon l'entité) pouvant être émises par l'utilisateur des services). L'autre concerne les F.P.D.U attendues. Chaque état est donc représenté par un ensemble d'évènements autorisés. Nous supposons que les indications et confirmations de la couche présentation sont toujours valides (car vérifiées dans la couche session), aussi ne les vérifions nous pas dans la couche application. Seules nous intéressent les unités de donées de services de présentations, ou unités de données de protocole d'application.

IV.5. CONVENTIONS D'ECRITURE.

Les conventions de description des actions sont identiques à celles de la couche transport (voir annexe I).

Rappelons que si aucune condition n'a pu être vérifiée pour un numéro donné, cela doit être considéré comme une erreur de protocole, et les traitements correspondent à ceux des cases vides.

IV.6. TABLES D'ETATS DE L'ENTITE INITIATRICE.

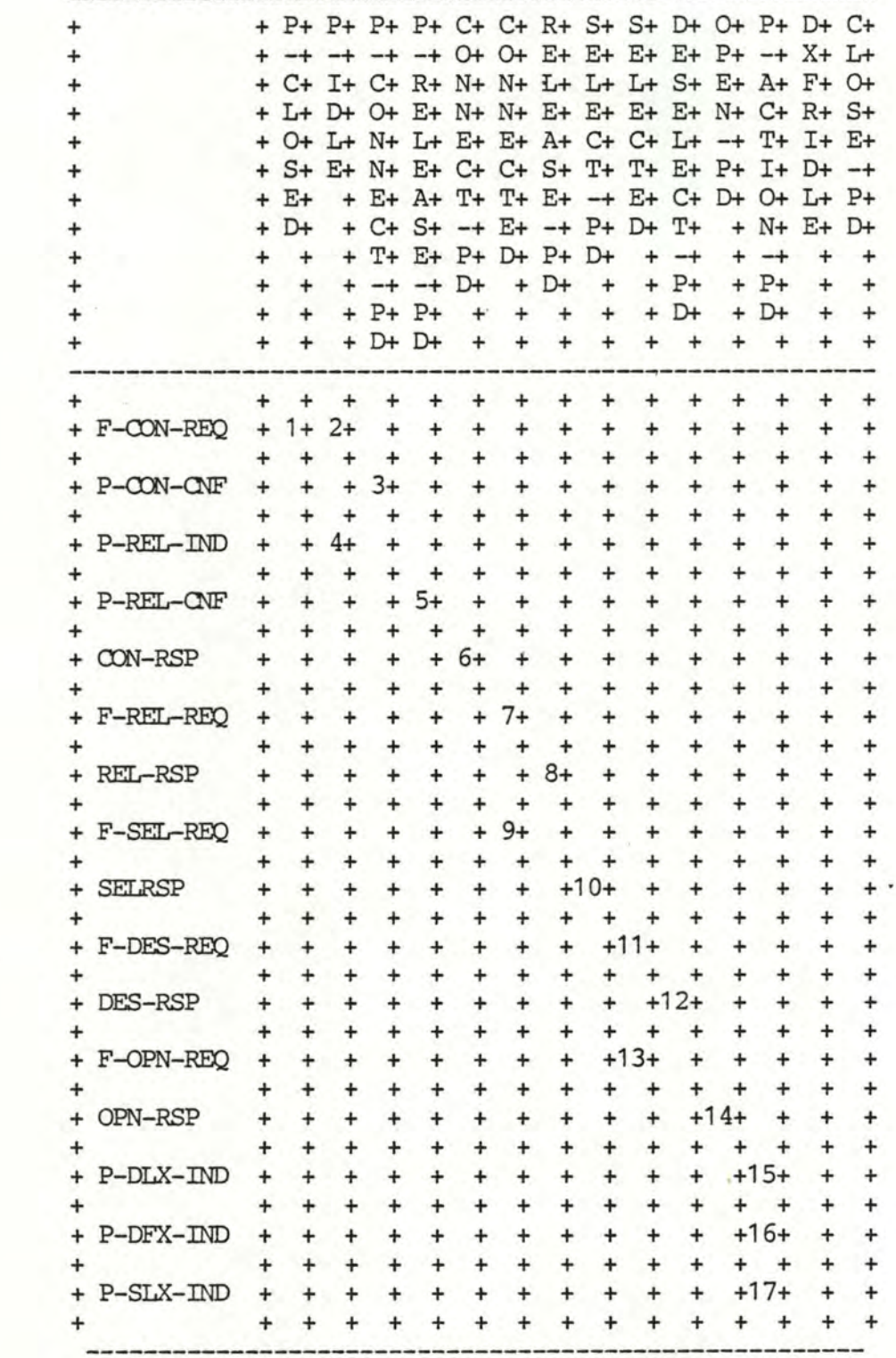
Cette partie est consacrée à la description des actions selon le couple état/évènement entrant de l'entité initiatrice de la connexion.

Chaque action est représentée par un numéro à l'intersection de l'état et de l'évènement entrant, et sa description est faite en regard du numéro dans la liste suivant les tables.

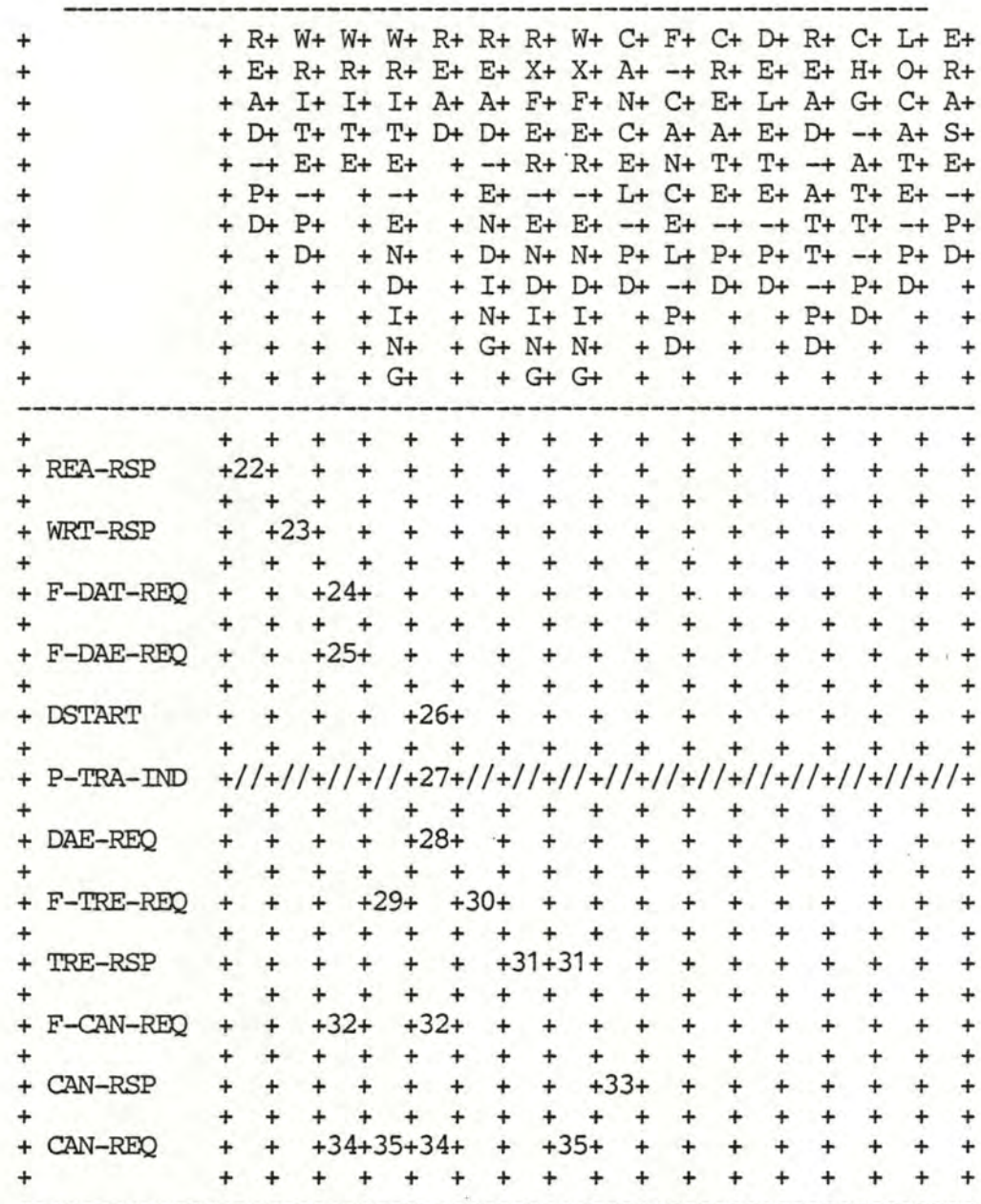
Toute intersection composée de "///" représente une combinaison état/évènement impossible.

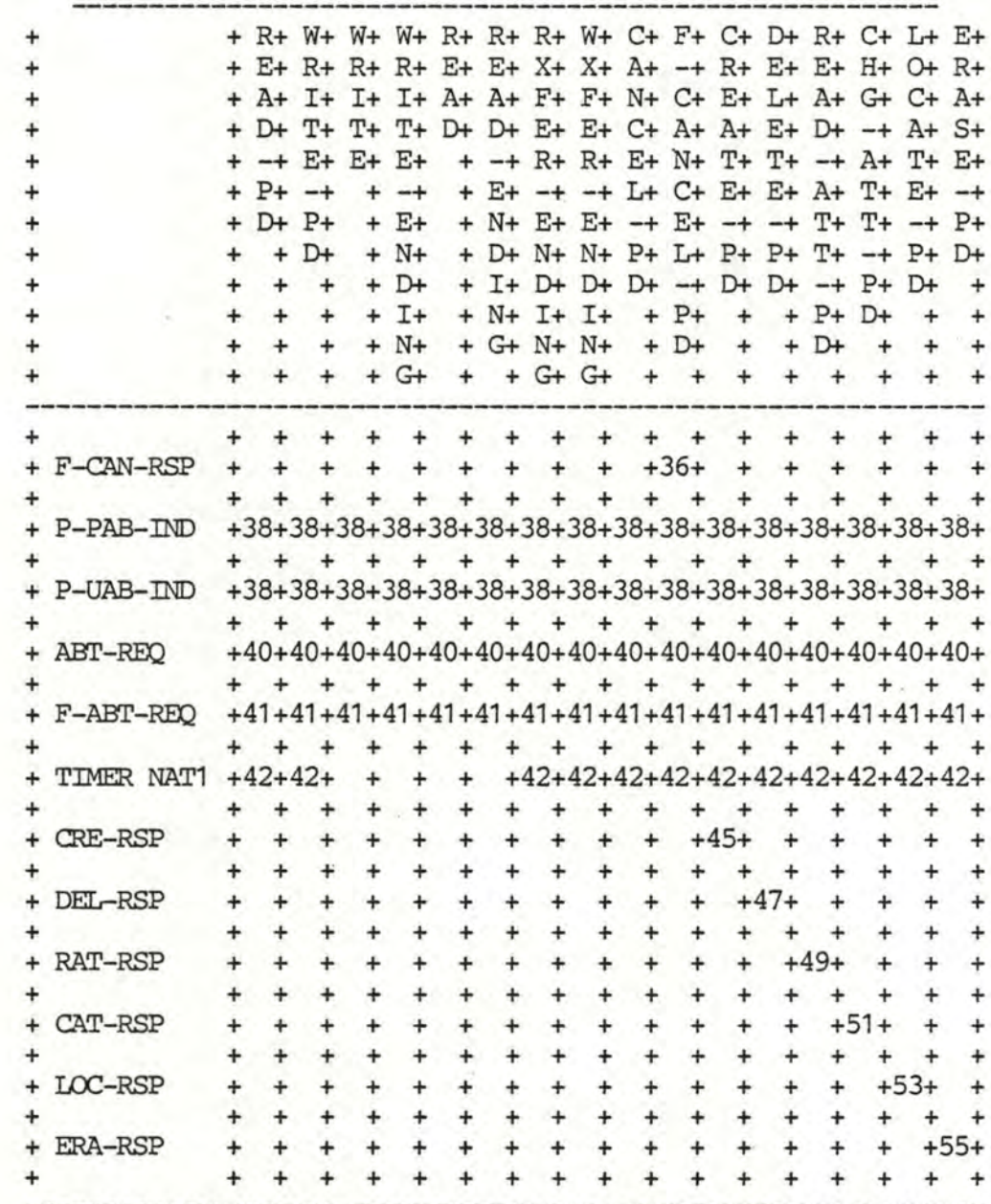
Toute intersection vide représente une erreur de protocole. Si l'évènement entrant provient de l'utilisateur des services, le code d'erreur ERR dans les paramètres de la primitive prend une valeur non nulle. Si l'Nvènement causant l'erreur provient du réseau (via la couche présentation), la connexion est coupée (P-PAB-IND).

A l'expiration d'un temporisateur, la connexion est également coupée.


P-CON-REQ
F-CON-REQ
P-CON-CNF
P-REL-IND
P-REL-CNF
CON-RSP
F-REL-REQ
REL-RSP
F-SEL-REQ
SELRSP
F-DES-REQ
DES-RSP
F-OPN-REQ
OPN-RSP
P-DLX-IND
P-DFX-IND
P-SLX-IND

	P+	P+	P+	P+	C+	C+	R+	S+	S+	D+	O+	P+	D+	C+
	-+	-+	-+	-+	O+	O+	E+	E+	E+	E+	P+	-+	X+	L+
	C+	I+	C+	R+	N+	N+	L+	L+	L+	S+	E+	A+	F+	O+
	L+	D+	O+	E+	N+	N+	E+	E+	E+	E+	N+	C+	R+	S+
	O+	L+	N+	L+	E+	E+	A+	C+	C+	L+	-+	T+	I+	E+
	S+	E+	N+	E+	C+	C+	S+	T+	T+	E+	P+	I+	D+	-+
	E+	+	E+	A+	T+	T+	E+	-+	E+	C+	D+	O+	L+	P+
	D+	+	C+	S+	-+	E+	-+	P+	D+	T+	+	N+	E+	D+
	+	+	T+	E+	P+	D+	P+	D+	+	-+	+	-+	+	+
	+	+	-+	-+	D+	+	D+	+	+	P+	+	P+	+	+
	+	+	P+	P+	+	+	+	+	+	D+	+	D+	+	+
	+	+	D+	D+	+	+	+	+	+	+	+	+	+	+
F-REA-REQ	+	+	+	+	+	+	+	+	+	+	+	18+	+	+
F-WRT-REQ	+	+	+	+	+	+	+	+	+	+	+	19+	+	+
F-CLO-REQ	+	+	+	+	+	+	+	+	+	+	+	20+	+	+
CLO-RSP	+	+	+	+	+	+	+	+	+	+	+	+	21+	+
P-PAB-IND	+	37+	38+	38+	38+	38+	38+	38+	38+	38+	38+	38+	38+	38+
P-UAB-IND	+	37+	38+	38+	38+	38+	38+	38+	38+	38+	38+	38+	38+	38+
ABT-REQ	+	39+	40+	40+	40+	40+	40+	40+	40+	40+	40+	40+	40+	40+
F-ABT-REQ	+	39+	41+	41+	41+	41+	41+	41+	41+	41+	41+	41+	41+	41+
TIMER NAT1	+	+	42+	42+	+	42+	+	+	+	42+	42+	+	+	42+
TIMER AT1	+	43+	+	+	+	+	+	+	+	+	+	+	+	+
F-CRE-REQ	+	+	+	+	+	44+	+	+	+	+	+	+	+	+
F-DEL-REQ	+	+	+	+	+	+	+	46+	+	+	+	+	+	+
F-RAT-REQ	+	+	+	+	+	+	+	48+	+	+	+	+	+	+
F-CAT-REQ	+	+	+	+	+	+	+	50+	+	+	+	+	+	+
F-LOC-REQ	+	+	+	+	+	+	+	+	+	+	+	52+	+	+
F-ERA-REQ	+	+	+	+	+	+	+	+	+	+	+	54+	+	+


R+ W+ W+ W+ R+ R+ R+ W+ C+ F+ C+ D+ R+ C+ L+ E+
E+ R+ R+ R+ E+ E+ X+ X+ A+ -+ R+ E+ E+ H+ O+ R+
A+ I+ I+ I+ A+ A+ F+ F+ N+ C+ E+ L+ A+ G+ C+ A+
D+ T+ T+ T+ D+ D+ E+ E+ C+ A+ A+ E+ D+ -+ A+ S+
-+ E+ E+ E+ + -+ R+ R+ E+ N+ T+ T+ -+ A+ T+ E+
P+ -+ + -+ + E+ -+ -+ L+ C+ E+ E+ A+ T+ E+ -+
D+ P+ + E+ + N+ E+ E+ -+ E+ -+ -+ T+ T+ -+ P+
+ D+ + N+ + D+ N+ N+ P+ L+ P+ P+ T+ -+ P+ D+
+ + + D+ + I+ D+ D+ D+ -+ D+ D+ -+ P+ D+ +
+ + + I+ + N+ I+ I+ + P+ + + P+ D+ + +
+ + + N+ + G+ N+ N+ + D+ + + D+ + + +
+ + + G+ + + G+ G+ + + + + + + + +
REA-RSP
WRT-RSP
F-DAT-REQ
F-DAE-REQ
DSTART
P-TRA-IND
DAE-REQ
F-TRE-REQ
TRE-RSP
F-CAN-REQ
CAN-RSP
CAN-REQ
22
23
24
25
26
27
28
29
30
31
32
33
34
35


R+ W+ W+ W+ R+ R+ R+ W+ C+ F+ C+ D+ R+ C+ L+ E+
E+ R+ R+ R+ E+ E+ X+ X+ A+ -+ R+ E+ E+ H+ O+ R+
A+ I+ I+ I+ A+ A+ F+ F+ N+ C+ E+ L+ A+ G+ C+ A+
D+ T+ T+ T+ D+ D+ E+ E+ C+ A+ A+ E+ D+ -+ A+ S+
-+ E+ E+ E+ + -+ R+ R+ E+ N+ T+ T+ -+ A+ T+ E+
P+ -+ + -+ + E+ -+ -+ L+ C+ E+ E+ A+ T+ E+ -+
D+ P+ + E+ + N+ E+ E+ -+ E+ -+ -+ T+ T+ -+ P+
+ D+ + N+ + D+ N+ N+ P+ L+ P+ P+ T+ -+ P+ D+
+ + + D+ + I+ D+ D+ D+ -+ D+ D+ -+ P+ D+ +
+ + + I+ + N+ I+ I+ + P+ + + P+ D+ + +
+ + + N+ + G+ N+ N+ + D+ + + D+ + + +
+ + + G+ + + G+ G+ + + + + + + + + +
F-CAN-RSP
P-PAB-IND
P-UAB-IND
ABT-REQ
F-ABT-REQ
TIMER NAT1
CRE-RSP
DEL-RSP
RAT-RSP
CAT-RSP
LOC-RSP
ERA-RSP
36
38
40
41
42
45
47
49
51
53
55

IV.7. LISTE D'ACTIONS DE L'ENTITE INITIATRICE.

```
1:si la requête est acceptable :
     P-CON-REQ(CON-REQ),
     start timer CT1
     TRANSPARANCY:=faux,
     DISCARD:=faux,
     RESET:=faux,
     EXPECTED-CHECK:=0,
     NEXT-SYNC:=0,
     P-DEFINE:=faux,
     P-SELECT:=faux,
     P-DELETE:=faux,
     ==) P-CONNECT-PD;
  si la requête n'est pas acceptable :
     F-ABT-IND,
     ==) P-CLOSED;

2:si la requête est acceptable :
     P-DT-REQ(CON-REQ),
     start timer CT1
     TRANSPARANCY:=faux,
     DISCARD:=faux,
     RESET:=faux,
     EXPECTED-CHECK:=0,
     NEXT-SYNC:=0,
     P-DEFINE:=faux,
     P-SELECT:=faux,
     P-DELETE:=faux,
     ==) CONNECT-PD;
  si la requête n'est pas acceptable
  et la connexion présentation est gardée :
     F-CON-CNF,
     ==) P-IDLE;
  si la requete n'est pas acceptable
  et la connexion transport n'est pas gardée :
     F-CON-CNF,
     P-REL-REQ,
     ==) P-RELEASE-PD;

3:si la fpdu est contenue dans la confirmation
présentation:
     stop timer CT1,
     F-CON-CNF,
     ==) CONNECTED;
  si la fpdu n'est pas contenue dans laconfirmation
présentation,
  mais que la confirmation présentation est positive :
     ==) CONNECT-PD;
  si la confirmation est non positive :
     ==) P-IDLE;

4:si la libération de connexion présentation est
```

```
acceptée :
      ==) P-CLOSED;
   sinon :
      ==) P-IDLE;

 5:si la connexion présentation est libérée :
      ==) P-CLOSED;
   sinon :
      ==) P-IDLE;

 6:si le DIAGNOSTIC = 0
      stop timer CT1,
      F-CON-CNF,
      ==) CONNECTED;
   sinon :
      F-CON-CNF,
      si la connexion présentation est libérée :
         ==) P-IDLE;
      sinon :
         P-REL-REQ,
         ==) P-CLOSED;

 7:P-DT-REQ(REL-REQ),
   start timer RLT1,
   ==) RELEASE-PD;

 8:F-REL-CNF,
   stop timer RLT1,
   si la connexion présentation est libérée :
      ==) P-IDLE;
   sinon :
      P-REL-REQ,
      ==) P-CLOSED;

 9:P-DT-REQ(SEL-REQ),
   start timer ST1,
   ==) SELECT-PD;

10:F-SEL-CNF,
   stop timer ST1,
   si le DIAGNOSTIC = 0 :
      ==) SELECTED;
   sinon :
      ==) CONNECTED;

11:P-DT-REQ(DES-REQ),
   start timer DST1,
   ==) DESELECT-PD;

12:F-DES-CNF,
   stop timer DST1,
   ==) CONNECTED;

13:P-DT-REQ(OPN-REQ),
   start timer OT1,
```

```
   ==) OPN-PD;

14:si le DIAGNOSTIC non = 0 :
      F-OPN-CNF,
      stop timer OT1,
      ==) SELECTED;
  si le DIAGNOSTIC = 0 :
      si le paramètre DELETE-CONTEXT est présent :
         P-DELETE:=vrai;
      si le paramètre DEFINE-CONTEXT est présent :
         P-DEFINE:=vrai;
      si le paramètre SELECT-CONTEXT est présent :
         P-SELECT:=vrai;
      si non P-DEFINE et non P-SELECT et non P-DELETE :
         F-OPN-CNF,
         stop timer OT1,
         ==) DXFRIDLE;
      sinon :
         mémoriser la F.P.D.U.,
         ==) P-ACTION-PD;

15:si P-DELETE :
      P-DELETE:=faux,
      P-DLX-RSP,
      si non P-DEFINE et non P-SELECT :
         F-OPN-CNF,
         stop timer OT1,
         DXFRIDLE;
      sinon :
         ==) P-ACTION-PD;

16:si P-DEFINE :
      P-DEFINE:=faux,
      P-DFX-RSP,
      si non P-DELETE et non P-SELECT :
         F-OPN-CNF,
         stop timer OT1,
         DXFRIDLE;
      sinon :
         ==) P-ACTION-PD;

17:si P-SELECT :
      P-SELECT:=faux,
      P-SLX-RSP,
      F-OPN-CNF,
      stop timer OT1,
      DXFRIDLE;

18:start timer RT1,
   si EXPECTED-CHECK=NEXTPOINT :
      P-DT-REQ(REA-REQ);
   sinon :
      P-RS-REQ(reset,REA-REQ);
   =) READ-PD;
```

```
19:start timer WT1,
   si EXPECTED-CHECK=NEXTPOINT :
      P-DT-REQ(WRT-REQ);
   sinon :
      P-RS-REQ(reset,WRT-REQ);
   =) WRITE-PD;

20:P-DT-REQ(CLO-REQ),
   start timer CLT1,
   ==) CLOSE-PD;

21:F-CLO-CNF,
   stop timer CLT1,
   =) SELECTED;

22:F-REA-CNF,
   stop timer RT1,
   si la F.P.D.U. a été extraite d'une primitive de resynchro
:
      EXPECTED-CHECK:=NEXT-SYNC;
   si DIAGNOSTIC = 0 :
      ==) READ;
   sinon :
      ==) DXFRIDLE;

23:F-WRT-CNF,
   stop timer WT1,
   si la F.P.D.U. a été extraite d'une primitive de resynchro
:
      EXPECTED-CHECK:=NEXT-SYNC;
   si DIAGNOSTIC = 0 :
      ==) WRITE;
   sinon :
      ==) DXFRIDLE;

24:si non TRANSPARANCY :
      TRANSPARANCY:=vrai,
      P-DT-REQ(DSTART);
   P-DT-REQ(DAT-REQ),
   ==) WRITE;

25:si TRANSPARANCY :
      TRANSPARANCY:=faux;
   P-DT-REQ(DAE-REQ),
   ==) WRITE-ENDING;

26:TRANSPARANCY:=vrai,
   ==) READ;

27:si non DISCARD :
      F-DAT-IND;
   ==) READ;

28:F-DAE-IND,
```

```
   ==) READ-ENDING;

29:P-DTREQ(TRE-REQ),
   start timer TET1,
   ==) WXFER-ENDING;

30:P-DTREQ(TRE-REQ),
   start timer TET1,
   ==) RXFER-ENDING;

31:F-TRE-CNF,
   stop timer TET1,
   ==) DXFRIDLE;

32:DISCARD:=vrai;
   P-RS-REQ(abandon,CAN-REQ),
   start timer CAN1,
   ==) CANCEL-PD;

33:DISCARD:=faux,
   F-CAN-CNF,
   stop timer CAN1,
   ==) DXFRIDLE;

34:TRANSPARANCY:=faux,
   F-CAN-IND,
   ==) F-CANCEL-PD;

35:F-CAN-IND,
   ==) F-CANCEL-PD;

36:P-RS-RSP(CAN-RSP),
   ==) DXFRIDLE;

37:==) P-CLOSED;

38:F-ABT-IND,
   ==) P-CLOSED;

39:==) P-IDLE;

40:F-ABT-IND,
   P-REL-REQ,
   ==) P-RELEASE-PD;

41:DISCARD:=vrai;
   TRANSPARANCY:=faux,
   P-RS-REQ(abandon,ABT-REQ),
   start timer AT1,
   ==) P-IDLE;

42:F-ABT-IND,
   DISCARD:=vrai,
   TRANSPARANCY:=faux,
   P-RS-REQ(abandon,ABT-REQ),
```

```
   start timer AT1,
   ==) P-IDLE;

43:P-UAB-REQ,
   ==) P-CLOSED;

44:P-DT-REQ(CREREQ),
   start timer CRT1,
   ==) CREATE-PD;

45:F-CRE-CNF,
   stop timer CRT1,
   si DIAGNOSTIC = 0 :
      ==) SELECTED;
   sinon
      ==) CONNECTED;

46:P-DT-REQ(DEL-REQ),
   start-timer DLT1,
   ==) DELETE-PD;

47:F-DEL-CNF,
   stop timer DLT1,
   ==) CONNECTED;

48:P-DT-REQ(RAT-REQ),
   start timer RAT1,
   ==) READ-ATT-PD;

49:F-RAT-CNF,
   stop timer RAT1,
   ==) SELECTED;

50:P-DT-REQ(CAT-REQ),
   start timer CAT1,
   ==) CHG-ATT-PD;

51:F-CAT-CNF,
   stop timer CAT1,
   ==) SELECTED;

52:P-DT-REQ(LOCREQ),
   start timer LT1,
   ==) LOCATE-PD;

53:F-LOC-CNF,
   stop timer LT1,
   ==) DXFRIDLE;

54:P-DT-REQ(ERA-REQ),
   start timer ET1,
   ==) ERASE-PD;

55:F-ERA-CNF,
   stop timer ET1,
```

==) DXFRIDLE

IV.8. TABLES D'ETATS DE L'ENTITE RECEPTRICE.

Cette partie est consacrée à la description des actions selon le couple état/évènement entrant de l'entité réceptrice de la connexion.

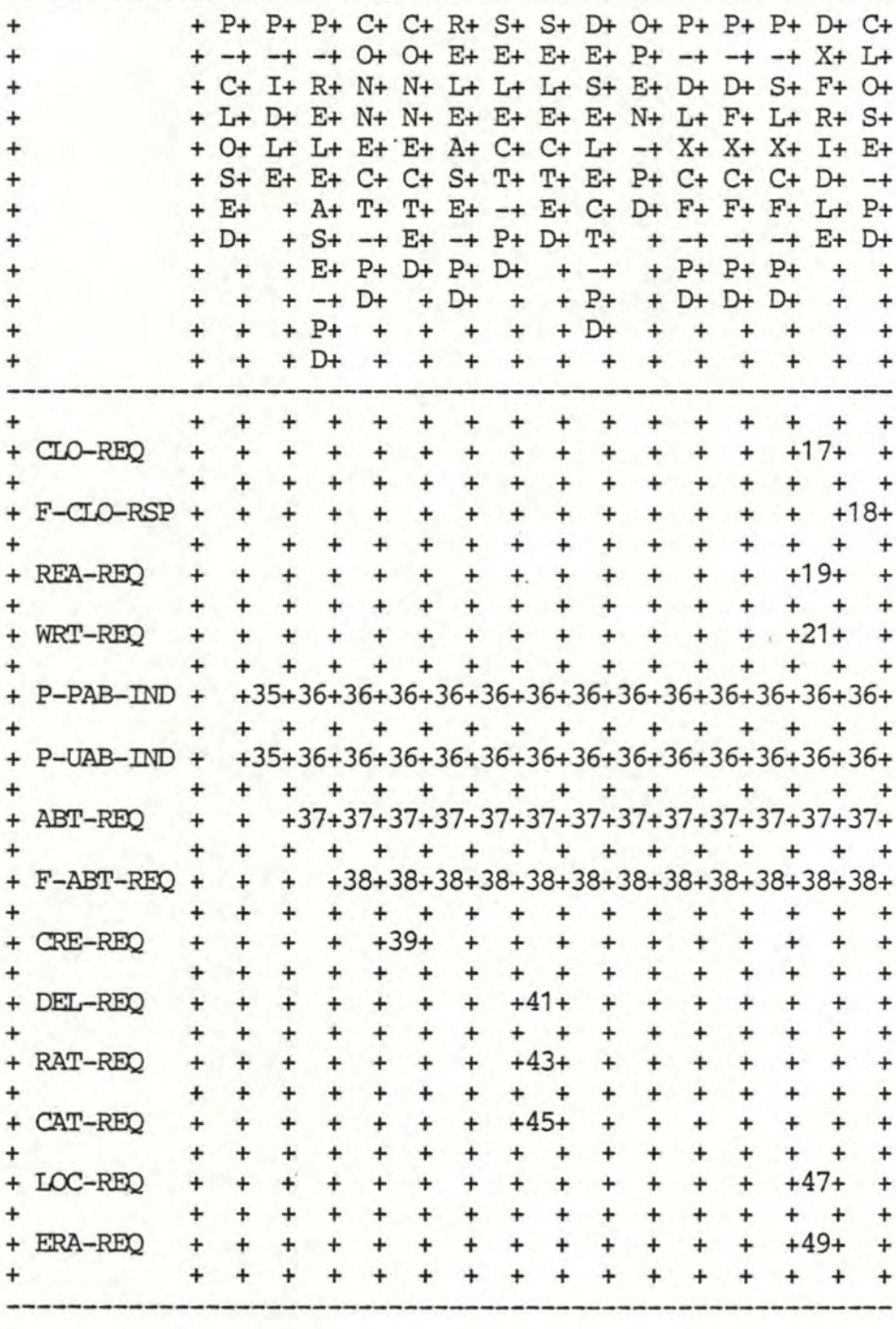
Chaque action est représentée par un numéro à l'intersection de l'état et de l'évènement entrant, et sa description est faite en regard du numéro dans la liste suivant les tables.

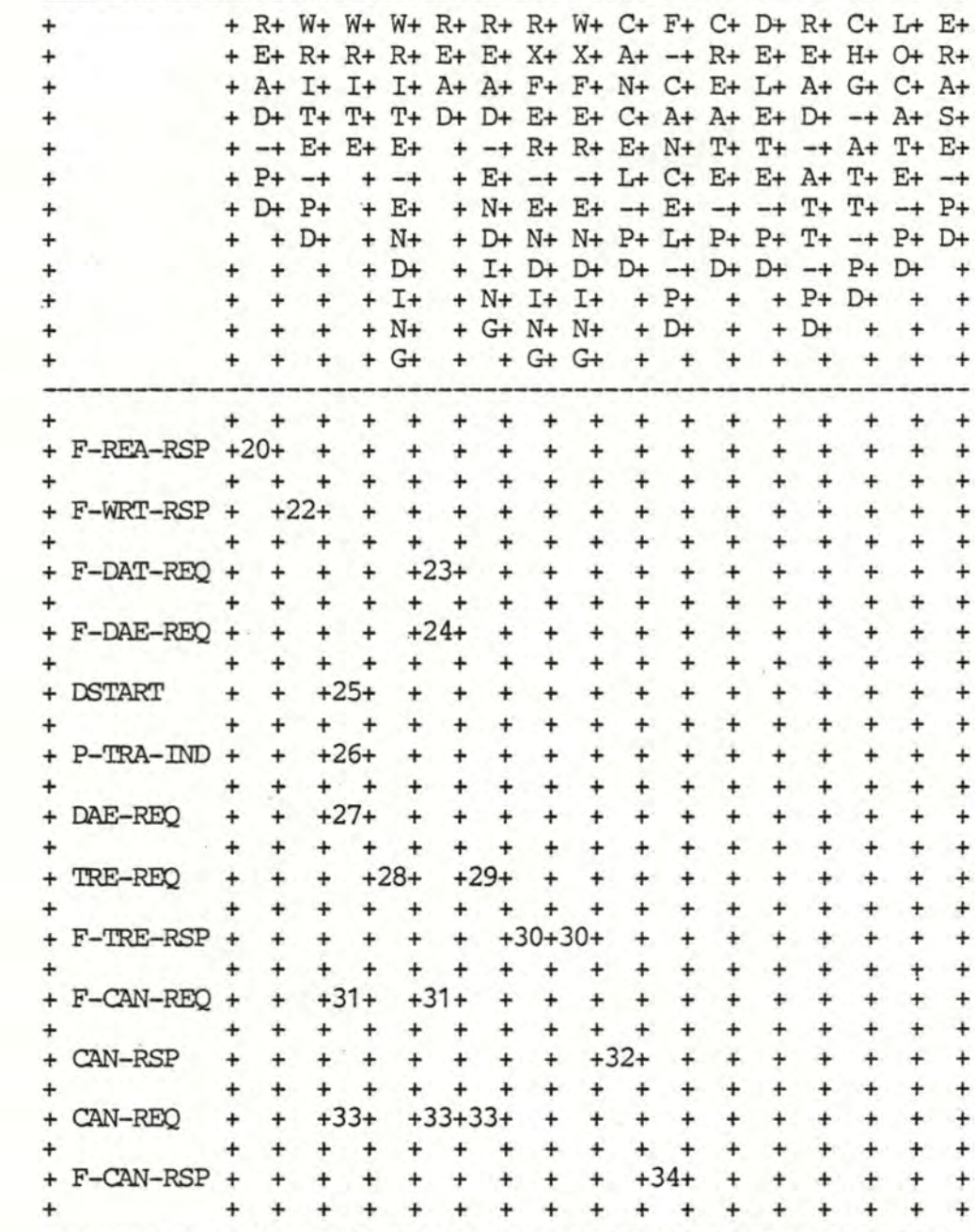
Toute intersection composée de "///" représente une combinaison état/évènement impossible.

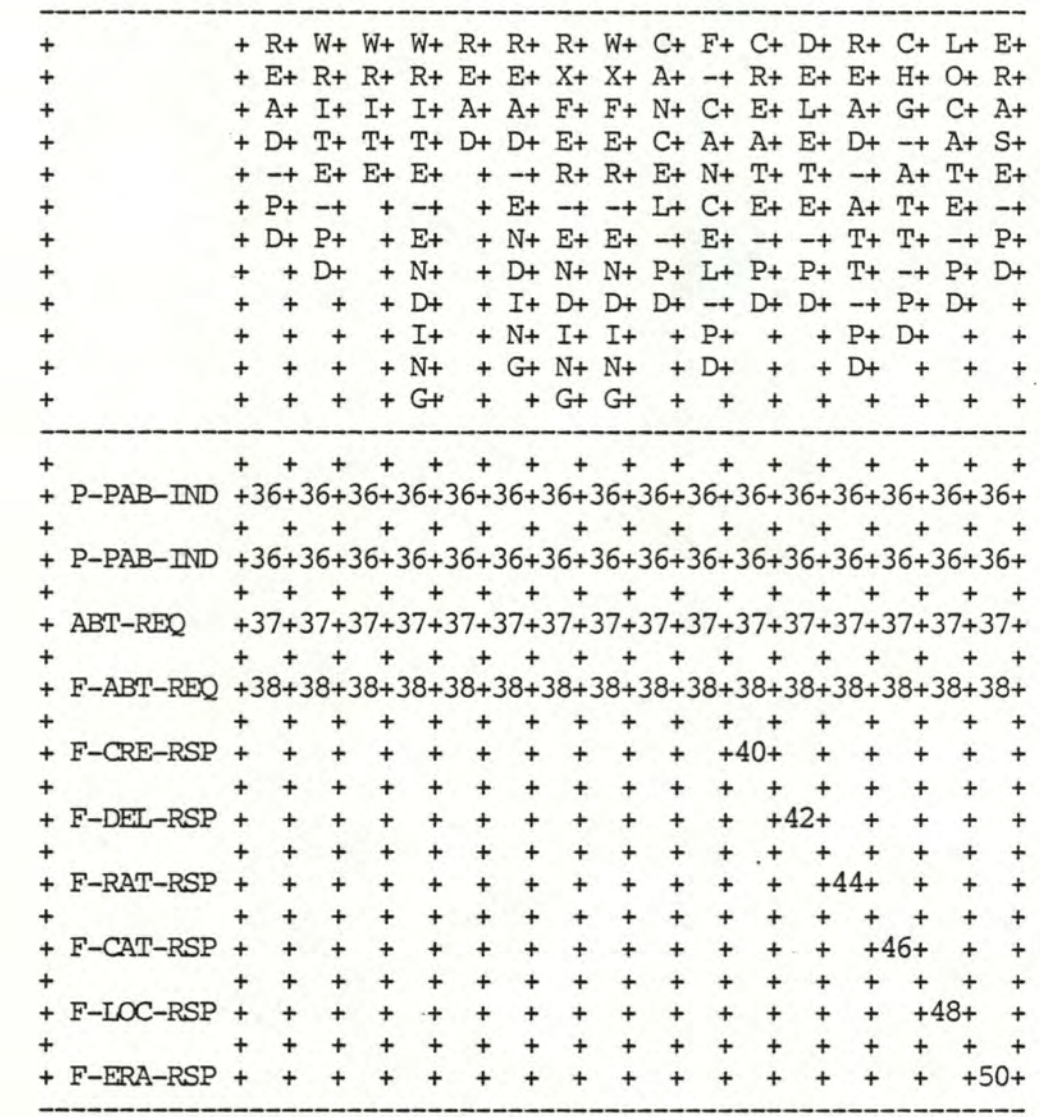
Toute intersection vide représente une erreur de protocole. Si l'évènement entrant provient de l'utilisateur des services, le code d'erreur ERR dans les paramètres de la primitive prend une valeur non nulle. Si l'évènement causant l'erreur provient du réseau (via la couche présentation), la connexion est coupée (P-PAB-IND).

A l'expiration d'un temporisateur, la connexion est également coupée.

	P-CON-IND	CON-REQ	P-REL-IND	P-REL-CNF	F-CON-RSP	REL-REQ	F-REL-RSP	SEL-REQ	F-SEL-REQ	DES-REQ	F-DES-RSP	OPN-REQ	F-OPN-RSP	P-DLX-CNF	P-DFX-CNF	P-SLX-CNF
P-P-CONNECTION-REQUEST	1															
P-CONNECTION-REQUEST		2														
P-P-RELEASE-IND			3													
P-P-RELEASE-CNF				4												
F-CONNECTION-RSP					5											
RELEASE-REQ						6										
F-RELEASE-RSP							7									
SELECT-REQ						8										
F-SELECT-REQ								9								
DESTROY-REQ								10								
F-DESTROY-RSP									11							
OPEN-REQ								12								
F-OPEN-RSP										13						
P-DLX-CNF												14				
P-DFX-CNF													15			
P-SLX-CNF														16		


P-CON-REQ
P-CON-IND
P-CON-RSP
P-CON-CNF
P-DTA-REQ
P-DTA-IND
P-EXP-REQ
P-EXP-IND
P-TYP-REQ
P-TYP-IND
P-CAP-REQ
P-CAP-IND
P-SYN-REQ
P-SYN-IND
P-SYN-RSP
P-SYN-CNF
P-ACT-REQ
P-ACT-IND
P-ACT-RSP
P-ACT-CNF
P-RES-REQ
P-RES-IND
P-RES-RSP
P-RES-CNF
P-PAB-IND
P-UAB-IND
ABT-REQ
F-ABT-REQ
CRE-REQ
DEL-REQ
RAT-REQ
CAT-REQ
LOC-REQ
ERA-REQ
CLO-REQ
F-CLO-RSP
REA-REQ
WRT-REQ


R W W W R R R W C F C D R C L E
E R R R E E X X A - R E E H O R
A I I I A A F F N C E L A G C A
D T T T D D E E C A A E D - A S
- E E E + - R R E N T T - A T E
P - + - + E - - L C E E A T E -
D P + E + N E E - E - - T T - P
+ D + N + D N N P L P P T - P D
+ + + D + I D D D - D D - P D +
+ + + I + N I I + P + + P D + +
+ + + N + G N N + D + + D + + +
+ + + G + + G G + + + + + + + +
F-REA-RSP 20
F-WRT-RSP 22
F-DAT-REQ 23
F-DAE-REQ 24
DSTART 25
P-TRA-IND 26
DAE-REQ 27
TRE-REQ 28 29
F-TRE-RSP 30 30
F-CAN-REQ 31 31
CAN-RSP 32
CAN-REQ 33 33 33
F-CAN-RSP 34


P-PAB-IND
P-PAB-IND
ABT-REQ
F-ABT-REQ
F-CRE-RSP
F-DEL-RSP
F-RAT-RSP
F-CAT-RSP
F-LOC-RSP
F-ERA-RSP

IV.9. LISTE DES ACTIONS DE L'ENTITE RECEPTRICE.

```
1:si l'indication de connexion n'est pas acceptable :
     P-UAB-REQ,
     ==) P-CLOSED;
  sinon :
     si l'indication ne contient pas la F.P.D.U. CON-REQ :
        P-CON-RSP,
        ==) P-IDLE;
     sinon :
        si la F.P.D.U. CON-REQ est acceptable :
           TRANSPARANCY:=faux,
           DISCARD:=faux,
           RESET:=faux,
           EXPECTED-CHECK:=0,
           NEXT-SYNC:=0,
           P-DEFINE:=faux,
           P-DELETE:=faux,
           P-SELECT:=faux,
           F-CON-IND,
           ==) CONNECT-PD;
        sinon :
           P-UAB-REQ,
           ==) P-CLOSED;

2:si la F.P.D.U. CON-REQ est acceptable :
     TRANSPARANCY:=faux,
     DISCARD:=faux,
     RESET:=faux,
     EXPECTED-CHECK:=0,
     NEXT-SYNC:=0,
     P-DEFINE:=faux,
     P-DELETE:=faux,
     P-SELECT:=faux,
     F-CON-IND,
     ==) CONNECT-PD;
  sinon :
     P-UAB-REQ,
     ==) P-CLOSED;

3:P-REL-RSP,
  si l'indication de déconnexion présentation est acceptable
:
     ==) P-CLOSED;
  sinon :
     ==) P-IDLE;

4:si DIAGNOSTIC = 0 :
     ==) P-CLOSED;
  sinon :
     ==) P-IDLE;

5:si DIAGNOSTIC = 0 :
```

```
      si la réponse de connexion présentation
   n'est pas encore émise :
         P-CON-RSP(CON-REQ),
         ==) CONNECTED;
      sinon :
         P-DT-REQ(CON-REQ),
         ==) CONNECTED;
   sinon :
      si la réponse de connexion présentation n'est pas
        encore émise :
         P-CON-RSP(CON-REQ),
         ==) P-CLOSED;
      sinon :
         P-DT-REQ(CON-REQ),
         ==) P-IDLE;

 6:F-REL-IND,
   ==) RELEASE-PD;

 7:P-DT-REQ(REL-RSP),
   ==) P-IDLE;

 8:F-SEL-IND,
   ==) SELECT-PD

 9:P-DT-REQ(SEL-RSP),
   si DIAGNOSTIC = 0 :
      ==) SELECTED;
   sinon :
      ==) CONNECTED;

10:F-DES-IND,
   ==) DESELECT-PD;

11:P-DT-REQ(DES-RSP),
   ==) CONNECTED;

12:F-OPN-IND,
   EXPECTED-FILE := 1,
   ==) OPEN-PD;

13:si un contexte de présentation doit être défini :
     P-DEFINE:=vrai;
   si un contexte de présentation doit être choisi :
     P-SELECT:=vrai;
   si un contexte de présentation doit être détruit :
     P-DELETE:=vrai;
   si non P-DEFINE et non P-SELECT et non P-DELETE :
      P-DT-REQ(OPN-RSP),
      ==) DXFRIDLE;
   sinon :
      P-DT-REQ(OPN-RSP),
      si P-DELETE :
         P-DLX-REQ,
         ==) P-DLXCF-PD;
```

```
        sinon :
           si P-DEFINE :
              P-DFX-REQ,
              ==) P-DFXCF-PD;
           sinon :
              si P-SELECT :
                 P-SLX-REQ,
                 ==) P-SLXCF-PD;

14:si P-DELETE :
      P-DELETE:=faux,
      si non P-DELETE et non P-SELECT :
         ==) DXFRIDLE;
      sinon :
         si P-DEFINE :
            ==) P-DFXCF-PD;
         sinon :
            si P-SELECT :
               ==) P-SLXCF-PD;

15:si P-DEFINE :
      P-DEFINE:=faux,
      si non P-SELECT :
         ==) DXFRIDLE;
      sinon :
         ==) P-SLXCF-PD;

16:si P-SELECT :
      P-SELECT:=faux,
      ==) DXFRIDLE;

17:F-CLO-IND,
   ==) CLOSE-PD;

18:P-DT-REQ(CLO-RSP),
   ==) SELECTED;

19:si le paramètre "point de sync." est présent :
      RESET:=vrai;
   F-REA-IND,
   ==) READ-PD;

20:si DIAGNOSTIC = 0 :
      si RESET :
         P-RS-REQ(reset,REA-RSP);
      sinon :
         P-DT-REQ(REA-RSP);
      ==) READ;
   sinon :
      ==) DXFRIDLE;

21:si le paramètre "point de sync." est présent :
      RESET:=vrai;
   F-WRT-IND,
   ==) WRITE-PD;
```

```
22:si DIAGNOSTIC = 0 :
      si RESET :
         P-RS-REQ(reset,WRT-RSP);
      sinon :
         P-DT-REQ(WRT-RSP);
      ==) WRITE;
   sinon :
      ==) DXFRIDLE;

23:si non TRANSPARANCY :
      TRANSPARANCY:=vrai,
      P-DT-REQ(dstart);
   P-DT-REQ(DAT-REQ),
   ==) READ;

24:si TRANSPARANCY :
      TRANSPARANCY:=faux;
   P-DT-REQ(DAE-REQ),
   ==) READ-ENDING;

25:TRANSPARANCY:=vrai,
   ==) WRITE;

26:si non DISCARD :
      F-DAT-IND;
   ==) WRITE;

27:F-DAE-IND,
   ==) WRITE-ENDING;

28:F-TRE-IND,
   ==) RXFER-ENDING;

29:F-TRE-IND,
   ==) WXFER-ENDING;

30:P-DT-REQ(TRE-RSP),
   P-TG-REQ(mi),
   ==) DXFRIDLE;

31:DISCARD:=vrai;
   P-RS-REQ(abandon,CAN-REQ),
   start timer CAN1,
   ==) CANCEL-PD;

32:DISCARD:=faux,
   F-CAN-CNF,
   stop timer CAN1,
   ==)DXFRIDLE;

33:F-CAN-IND,
   ==) F-CANCEL-PENDING;

34:P-DT-REQ(CAN-RSP),
```

```
   ==) DXFRIDLE;

35:==) P-CLOSED;

36:F-ABT-IND,
   ==) P-CLOSED;

37:F-ABT-IND,
   P-REL-REQ,
   ==) P-RELEASE-PD;

38:DISCARD:=vrai,
   TRANSPARANCY:=faux,
   P-RS-REQ(abandon,ABT-REQ),
   ==) P-RELEASE-PD;

39:F-CRE-IND,
   ==) CREATE-PD;

40:P-DT-REQ(CRE-RSP),
   si DIAGNOSTIC = 0 :
      ==) SELECTED;
   sinon :
      ==) CONNECTED;

41:F-DEL-IND,
   ==) DELETE-PD;

42:P-DT-REQ(DEL-RSP),
   ==) SELECTED;

43:F-RAT-IND,
   ==) READ-ATT-PD;

44:P-DT-REQ(RAT-RSP),
   ==) SELECTED;

45:F-CAT-IND,
   ==) CHG-ATT-PD;

46:P-DT-REQ(CAT-RSP),
   ==) SELECTED;

47:F-LOC-IND,
   ==) LOCATE-PD;

48:P-DT-REQ(LOC-RSP),
   ==) DXFRIDLE;

49:F-ERA-IND,
   ==) ERASE-PD;

50:P-DT-REQ(ERA-RSP),
   ==) DXFRIDLE;
```