COUCHE 7/OSI : TRANSFERT DE FICHIERS FTAM

Le transfert de fichiers entre systèmes hétérogènes correspond à un besoin fondamental. Il en est de même pour l'accès à des fichiers ou leur manipulation sur un système à partir d'un programme ou d'une commande d'un utilisateur distant.

Ce problème est traité dans un ensemble de Standards de l'OSI regroupés dans la norme FTAM OSI/DIS 8571.

FTAM : **File Transfer , Acces and Manipulation** fournit un service complet de transfert, accès ou gestion de fichiers virtuels. Ce service est situé au niveau 7/OSI et s'appuie sur un service commun d'Application (sous-couche ACSE : Application Control Service Elements) et un service Présentation qui lui fournit les moyens d'un transfert transparent et ordonné notamment par l'utilisation d'une syntaxe de transfert permettant de traduire les syntaxes abstraites des deux applications communiquant.

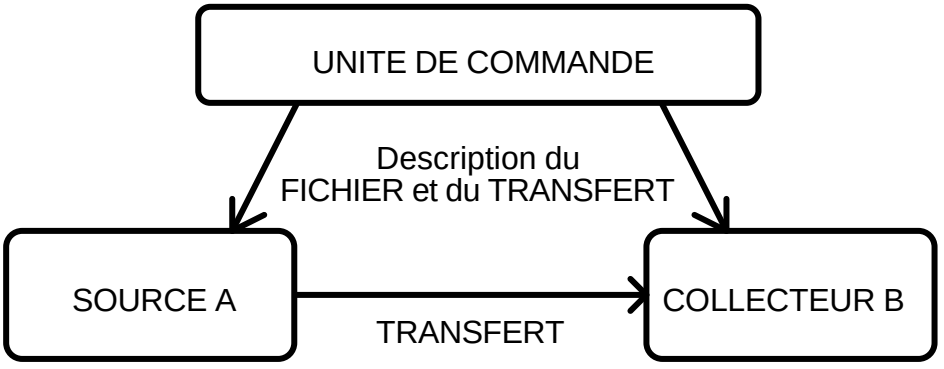
1. SERVICE FOURNI

1.1 Généralités

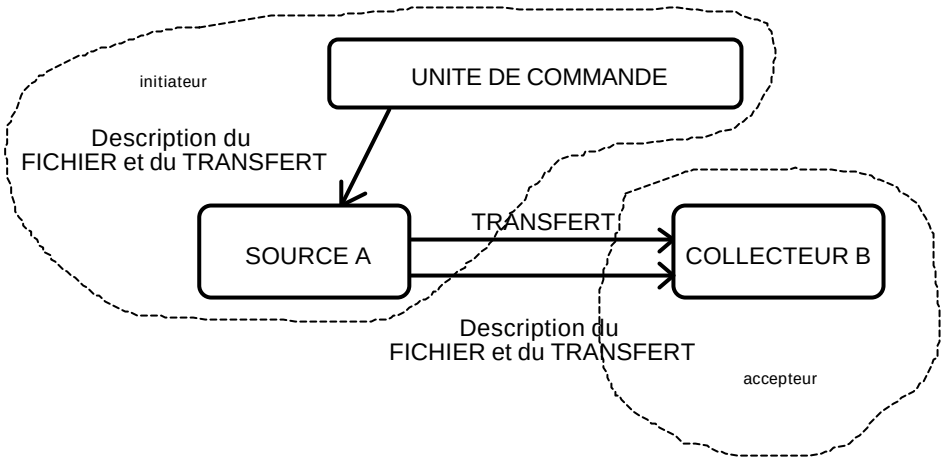
1.1.1. Commande de l'Activité Fichier virtuel

D'un point de vue logique, trois entité sont à prendre en compte :

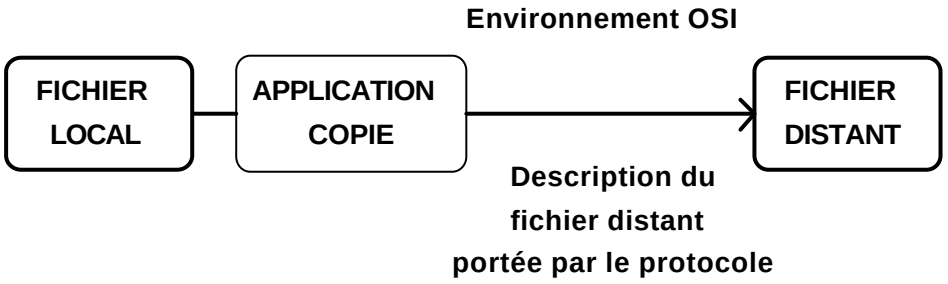
* une unité de commande qui initie le transfert
* une source de fichier virtuel
* un collecteur de fichier virtuel



Pour simplifier la coordination et la commande, on suppose que l'unité de commande véhicule ces flux via l'une des deux entités du protocole qui agit comme son agent pour réaliser le transfert. Ceci est illustré sur le schéma ci-dessous :



Le dialogue devient asymétrique :

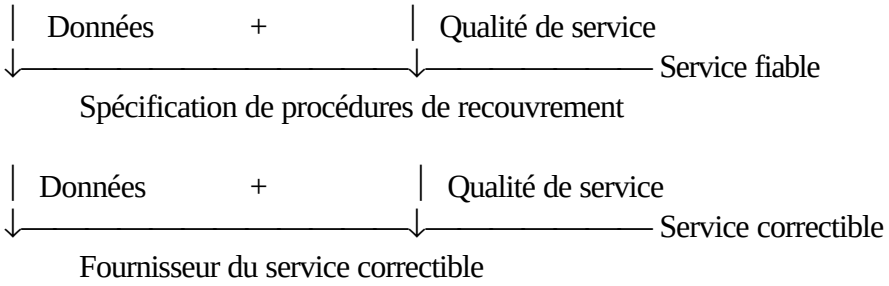


1.1.2. Service de fichier fiable et Service de fichier correctible

La communication entre initiateur et accepteur peut être décomposée en deux sous-couches :

* Service de fichier fiable avec lequel l'utilisateur n'a plus de responsabilité ou de contrainte de reprise, récupération, etc.

* Service de fichier correctible qui inclut les facilités de reprise sur défaut et d'administration. L'usager a la possibilité de choisir le mode de récupération des défauts qu'il souhaite utiliser



1.1.3 Classes de service. Unités fonctionnelles

Pour pouvoir traiter une grande variété d'applications , on dispose d'une nombre de fonctions très élevé.

Ces fonctions sont regroupées en unités fonctionnelles dont la mise en oeuvre pourra être négociée entre les deux entités communicantes.

On définit ainsi 5 classes de service qui correspondent à une sélection d'un sous-ensemble d'unités fonctionnelles :

* Transfert
* Transfert et administration
* Accès
* "sans contrainte"

1.2. Fonctions associées au service fichier

1.2.1. Contrôle d'accès

Il est basé sur des listes de contrôle d'accès. Chaque entrée dans la liste fournit un ensemble d'actions permises si les conditions associées sont remplies.

1.2.2 Comptes

Un mécanisme de compte et de mesure de la charge d'utilisation est crée pour affecter les coûts de stockage à un compte particulier et les coûts d'accès à d'autres comptes (éventuellement). Des paramètres de charge permettent de calculer les coûts réels d'une opération avant de les affecter à un compte.

1.2.3 Contrôle de concurrence

L'objectif du contrôle de concurrence est d'assurer l'initiateur d'une action qu'il aura une vue consistante du fichier durant cette action en apportant les restrictions d'accès nécessaires sur les fichiers partagés.

Le niveau externe de fichier porte sur le fichier complet et ses attributs.

Le niveau interne porte sur le contenu du fichier lorsqu'il est ouvert.

Les verrous suivant peuvent être placés :

* Partagé - par tous
* Exclusif - pour un utilisateur
* non demandé - pour un utilisateur particulier
* pas d'accès - accès interdit à tous

Les opérations suivantes peuvent être contrôlées :

* lecture
* insertion
* remplacement
* effacement (erase)
* extension
* lecture d'attribut
* changement d'attribut
* suppression du fichier (delete)

Rq: la fonction effacement n'entraîne la suppression physique du fichier que s'il n'existe aucun autre lien à ce fichier.

2. SERVICES REQUIS

Le service FTAM s'appuie sur les services fournis par les sous-couches ou couches inférieures du modèle de référence.

2.1 ACSE

Il s'agit de l'élément de service de plus bas du niveau de la couche Application, généralement fourni avec la couche Présentation.

On utilise sa fonction de commande d'association pour
* établir
* relâcher

une association entre deux entités d'application.

Une seule activité fichier peut être en cours sur une association à un moment donné. Lorsque cette activité est terminée l'association peut être supprimée.

2.2 Présentation

Le service présentation fournit une syntaxe de transfert permettant de traduire la représentation locale du fichier. Il fournit aussi une syntaxe (et un contexte de transfert) pour assurer les fonctions de transfert, d'accès ou de manipulation de ces fichiers. Pour un transfert de fichier(s) une négociation de contextes de présentation devra être effectuée.

2.3 Session

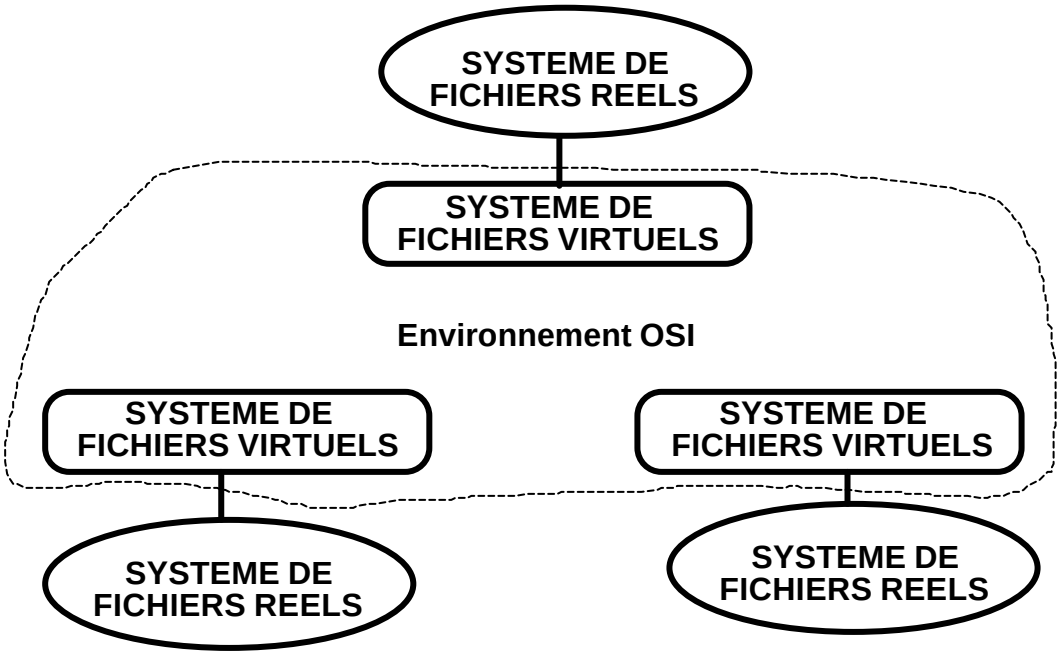
Pour pouvoir assurer un service fiable il est nécessaire de disposer des fonctions de :
* insertion de points de synchronisation
* resynchronisation
* gestion de jetons (synchro, terminaison)

Ces fonction correspondent au sous-ensemble BSS.

3. MODELE DE SYSTEME DE FICHIERS VIRTUELS

3.1. Fichiers réels et fichiers virtuels : projection.

Pour répondre à la diversité très grande des systèmes de gestion de fichiers, il est nécessaire de concevoir un système virtuel permettant de décrire toutes les propriétés (ou presque) des fichiers réels et de réaliser une "projection" locale (mapping) d'un fichier réel sur un fichier virtuel. Cette projection est illustrée par le schéma ci-dessous.



Cette projection est faite entre les actions, les accès fichiers, les fichiers, leurs attributs et les ressources dans l'environnement réel.

3.2. Caractéristiques d'un fichier virtuel

Dans un système de fichiers virtuels, un fichier est une entité qui possède :

un nom simple, non ambigu
des attributs qui expriment ses propriétés : compte, historique, etc.
des attributs décrivant sa structure logique et la dimension des données stockées
des unités de données formant le contenu du fichier

Ces caractéristiques sont intangibles et toute action faite par deux initiateurs doit produire le même effet.

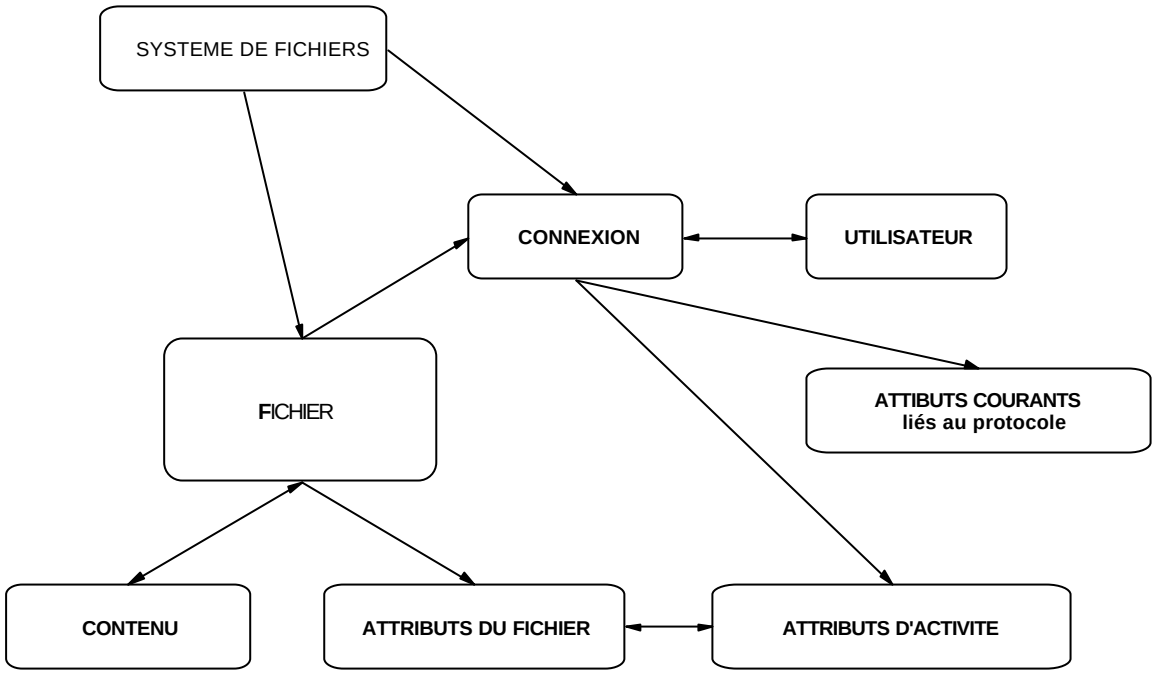
On doit ajouter

des attributs d'activité : authentification, options de transfert, coûts accumulés par exemple.

Ils sont créés et gérés par FTAM pour une activité particulière.

Enfin certains attributs portent des contraintes qui ne sont utiles qu'à un sous-ensemble des utilisateurs, par exemple un chemin d'accès complet pour un utilisateur d'une station inutile à ceux d'une autre.

3.3. Schéma d'un système de fichiers virtuels



(a) : Association 1 pour 1

3.4. Structure d'accès

Un fichier peut contenir zéro, une ou plusieurs unités de données identifiables. Ces unités de donnés sont apparentées et sont reliées par une structure **arborescente**.

A chaque noeud de l'arbre est associé **zéro ou une** unité de données. La structure des informations du point de vue de l'utilisateur peut être différente de la structure d'accès et le service utilisateur devra être projeté sur la structure d'accès.

En général les structures d'information peuvent être séquentielles, hiérarchiques, en réseau ou relationnelles.

Actuellement seule les **structures séquentielles et hiérarchiques sont supportées par FTAM.** D'autres structures pourront être ajoutées (par exemple pour traiter les bases de données relationnelles).

Les unités de données sur lesquelles portent les opérations sont nommées **FADU : File Acces Data Unit**. Ces FADU peuvent elles aussi être structurées

Dans cette structure de fichiers on distingue 4 aspects :

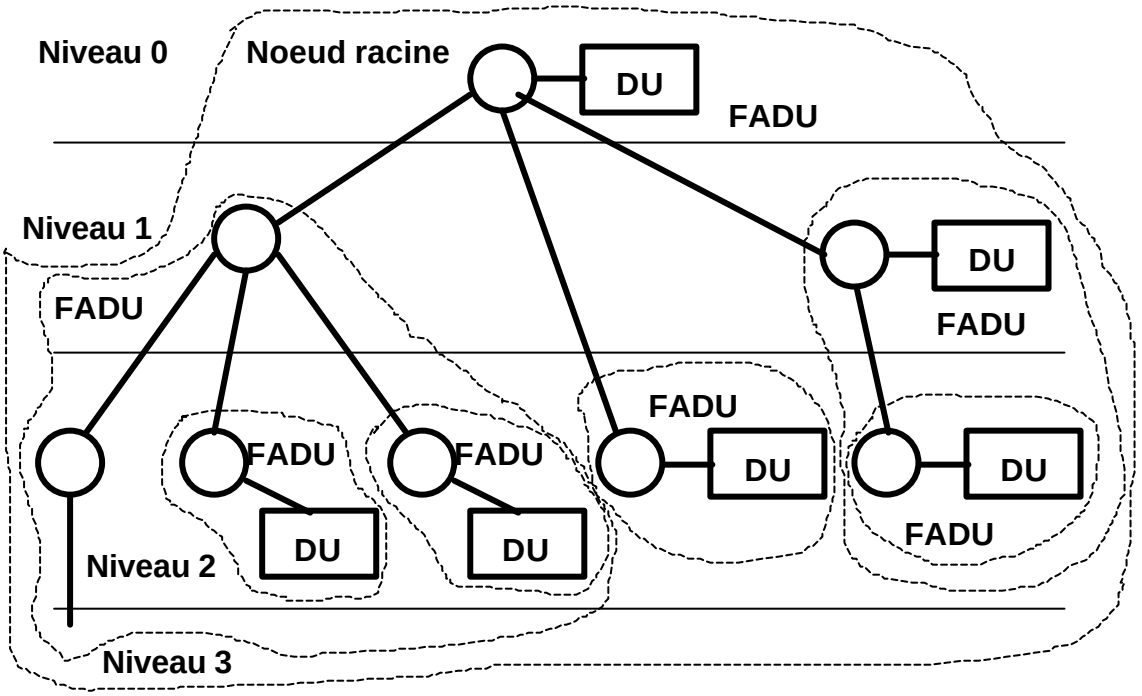
* structure d'accès

* structure de présentation décrit le syntaxes abstraites des unités de données définies dans la structure d'accès.

* structure de transfert décrit la sérialisation des unités de données pour les besoins de communication

* structure d'identification décrit le nommage des noeuds dans la structure d'accès et l'identification des fichiers à transférer.

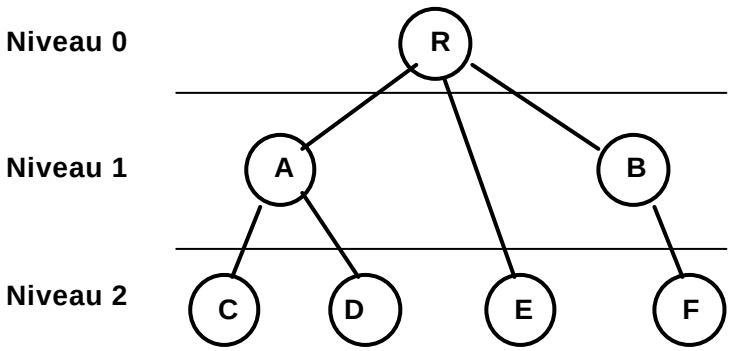
Le schéma ci-dessous donne un exemple de structure d'accès.



FADU : File Acces Data Unit

DU : Data Unit

Cette structure du système de fichiers virtuels induit un préordre pour traiter : localiser, transférer ou manipuler un ensemble de fichiers.



Ici le préordre sera R , A , C, D , E , B , F

3.5. Structures de présentation

Actuellement sept structures de présentation sont supportées par FTAM :

1) non structuré une seule unité de données, sans nom
exemple : fichier flot (stream)

2) séquentiel plat
suite d'unités de données non nommées (sur le modèle des fichiers d'entrées-sorties séquentiels Fortran)
exemple : fichier séquentiel

3) ordonné plat
suite d'unités de données nommées. La manipulation de données avec des noms dupliqués est possible (liens)
exemple : fichier séquentiel indexé

4) ordonné plat à noms uniques.
suite d'unités de données avec des noms spécifiques.
exemple : peut coller à un fichier relatif ?

5) ordonné hiérarchique.
Hiérarchie générale permise, index multiples; l'insertion est basée sur la position dans ces index ; deux noeuds peuvent avoir le même nom , ils sont distingués par l'ordre de parcours de l'arbre.

6) hiérarchique général.
Hiérarchie générale mais avec contrôle complet sur le placement des nouveaux noeuds quand la structure est modifiée.

7) hiérarchique général à noms uniques

Les fichiers hiérarchiques correspondent aux "répertoires".

3.6. Types de documents

Le contenu d'un fichier peut être de différents types.

Il est possible de définir séparément le modèle de fichier, les contraintes et la syntaxe abstraite dans laquelle sont codées les données. Cependant, il est souvent plus commode de regrouper ces caractéristiques dans quelques ensembles couramment utilisés : ceci est réalisé par la définition de types de documents.

Un type de document est caractérisé par une dizaine de paramètres :

* identification du type de document
* portée souhaitée
* sémantique à appliquer pour interpréter le document
* structure d'accès par un ensemble de contraintes appliquées sur le modèle hiérarchique général
* syntaxe abstraite de l'information structurante et des unités de données dans les structures
* syntaxe de transfert
* résultat de la concaténation de deux instances d'un même type de document
* moyen de simplifier la structure d'accès pour faire voir une instance d'un document comme un type de document plus simple.

Le concept de type de document est récursif et permet par raffinements successifs de travailler sur des classes plus fines de documents.

exemples : texte non structuré , texte séquentiel (structuré en champs. .) , binaire non structuré , etc.

4. SERVICE DE TRANSFERT DE FICHIERS

Le service de transfert de fichiers et le protocole qui le supporte, permet de créer étape par étape un environnement de travail dans lequel pourront prendre place les activités à exécuter.

Le dialogue doit permettre

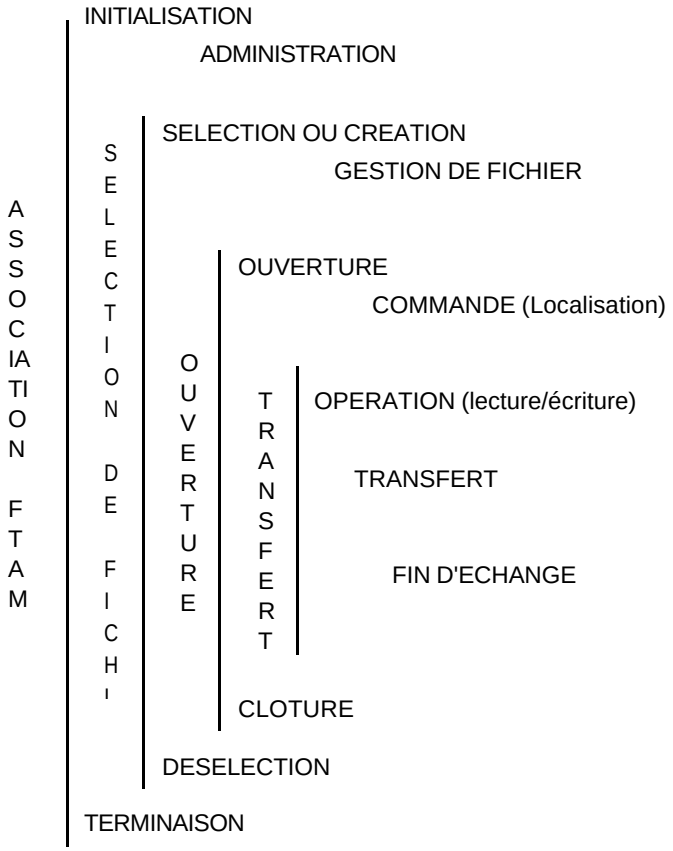
* à l'initiateur et au répondeur, d'établir leurs identités mutuelles
* d'identifier le fichier à traiter
* D'établir les attributs décrivants fichier et données
* de réaliser la gestion du fichier
* de localiser la position des données à traiter dans la structure des FADU
* de transférer, remplacer, effacer une ou plusieurs FADU

4.1 Phases et Régimes

Les périodes durant lesquelles un **état du service** est valide pour les deux applications qui correspondent est appelé un **régime**. En avançant dans les étapes on établit les régimes successifs correspondants.

Les périodes durant lesquels il y a des échanges protocolaires sont des **phases** (établissement, transfert, etc.)

Les activités progressent durant les phases.



* Le régime **association FTAM** commence après la **phase d'initialisation** et est clos par la **phase de terminaison**. Ces phases utilisent les primitives

F-INITIALIZE
F-TERMINATE
F-ABORT

Ce régime comporte aussi la **phase d'administration** du système de fichiers.

* Le régime **sélection de fichier** suit une **phase de sélection ou une phase de création** de fichier. Il se termine par une **phase de désélection ou une phase d'effacement** (delete). Il peut comporter une phase de gestion de fichier. Les primitives suivantes sont utilisées

F-SELECT
F-CREATE
F-READ-ATTRIB
F-CHANGE-ATTRIB
F-DESELECT
F-DELETE

* Le régime d'**ouverture de fichier** va de la **phase d'ouverture** à la **phase de clôture**. Il comporte une **phase de commande des opérations** qui permet de localiser le noeud origine dans la structure d'accès et de supprimer une FADU de cette structure (erase). Ces phases utilisent les primitives

F-OPEN
F-LOCATE
F-ERASE
F-CLOSE

* Le régime de **transfert de données** comporte des **phases d'accès aux données**. Un transfert comporte trois opérations :

- définition du type d'opération et identification des données d'accès à appliquer
- tout transfert de données utile
- fin de l'échange

Les primitives ci-dessous sont mises en oeuvre
F-READ
F-WRITE
F-DATA
F-DATA-END
F-TRANSFER-END

Ainsi pour réaliser un transfert de fichier les phases suivantes sont exécutées successivement :

* initialisation de l'association FTAM
* administration du système de fichier (si nécessaire)
* sélection du fichier
* administration du fichier (si nécessaire)
* ouverture du fichier
* accès aux données
* définition de l'opération (lecture ou écriture)
* transfert proprement dit
* fermeture du fichier
* désélection
* terminaison de l'association FTAM

4.2. Eléments de service

Le service FTAM fournit un **Transfert de données** *unidirectionnel* **avec des points de contrôle confirmés**.

Pour simplifier l'implantation des entités de protocole, les classes de service et les unités fonctionnelles ont été choisies pour donner des automates simples avec une inter-dépendance aussi faible que possible entre unités fonctionnelles.

4.2.1 Concaténation

Une **concaténation** des PDU est introduite pour éviter des échanges trop nombreux correspondant aux phases élémentaires. Les PDU deviennent ainsi de simples champs d'un message.

Cette concaténation est obligatoire pour la classe de service de transfert. Pour des classes de service plus puissantes, elle est optionnelle pour donner plus de flexibilité et supporter une plus grande variété d'applications.

4.2.2 Transparence

Pour le fournisseur du service présentation, les données utilisateur (fichier) comme les informations de contrôle du protocole d'application se présentent comme une suite de données. Si ces deux types de données se servent de la même syntaxe abstraite, des ambiguï tés peuvent se produire. Pour les éviter, FTAM utilise deux **contextes de présentation** différents pour faire transférer PCI et SDU.

4.2.3 Points de contrôle

Pour permettre des reprises du transfert et la récupération des défauts, des points de contrôle sont insérés dans le flux de données.

Source et collecteur doivent sauvegarder les informations donnant la position relative dans un fichier des différents points de contrôle.

Il est aussi nécessaire de garder des informations sur l'endroit de la structure où est insérer le point de contrôle. Ces informations sont de deux types :

information sur le contenu du fichier telles que position dans la structure d'accès ou relations entre unités de données (pointeurs, identificateur)

informations sur l'état de l'interprétation de la syntaxe abstraite ; pour les simplifier, l'insertion des points de contrôle est généralement réalisée à la limite d'unités de données complètes définies dans la syntaxe abstraite.

Ainsi, un fichier devra apparaître comme une suite d'éléments de données concaténés, chacun d'eux étant codable indépendamment des autres.

4.2.4 Diagnostics et résultats

Le protocole fournit deux niveaux d'information pour évaluer le succès ou l'échec d'une opération demandée.

Le premier donne des informations très générales
sur le résultat d'action
sur : le protocole lui-même
le système de fichiers

Par exemple, dans une opération de suppression, la transition de sélection au niveau du protocole aboutit toujours alors que la suppression dans le système de fichiers peut ne pas être effective.

Le résultat d'une telle action peut être : succès, erreur récupérable ou erreur complète.

Le second niveau, donné par le paramètre diagnostic, peut contenir des informations plus détaillées et plus spécifiques de l'utilisateur.

4.2.5 "Bordereau" et stockage rémanent

La récupération sur défaut permet de poursuivre une activité même après une rupture du service présentation ou de l'association fournie par le service ACSE. Pour cela il est nécessaire de préserver les informations concernant l'activité et l'étape atteinte. Le corps de l'information à sauvegarder est appelé un **"bordereau" (docket)**.

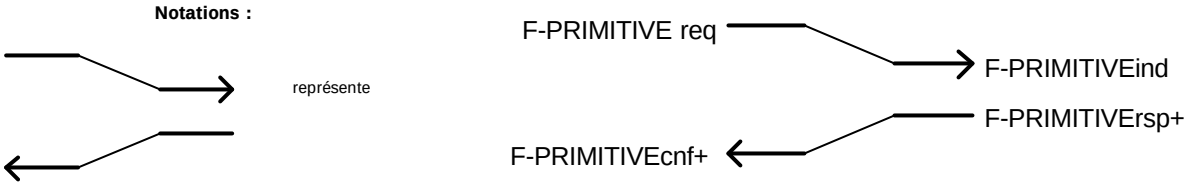
Ceci n'implique pas une concentration en un seul endroit, mais une **permanence** de cette information. Elle peut être stockée sur disque , en mémoire RAM rémanent (secourue) ou sur tout autre support.

4.2.6 Mécanismes de récupération sur erreur

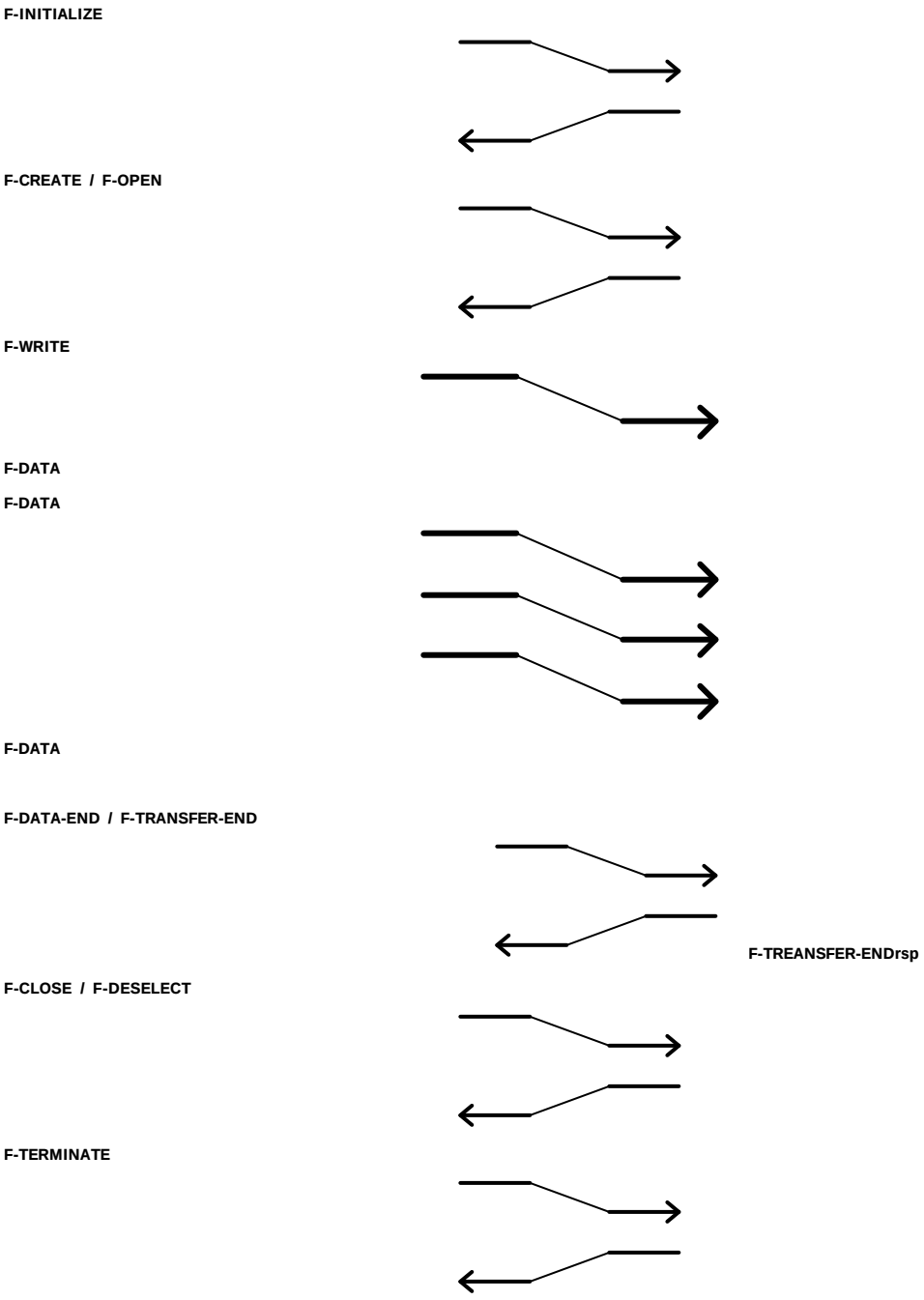
Ces mécanismes doivent fournir un **service de transfert fiable**.

Lorsque le système de transfert est détruit, le système reconstruit, **à partir des "bordereaux"** une connexion supportant le transfert et la replace dans l'état précédant le défaut.

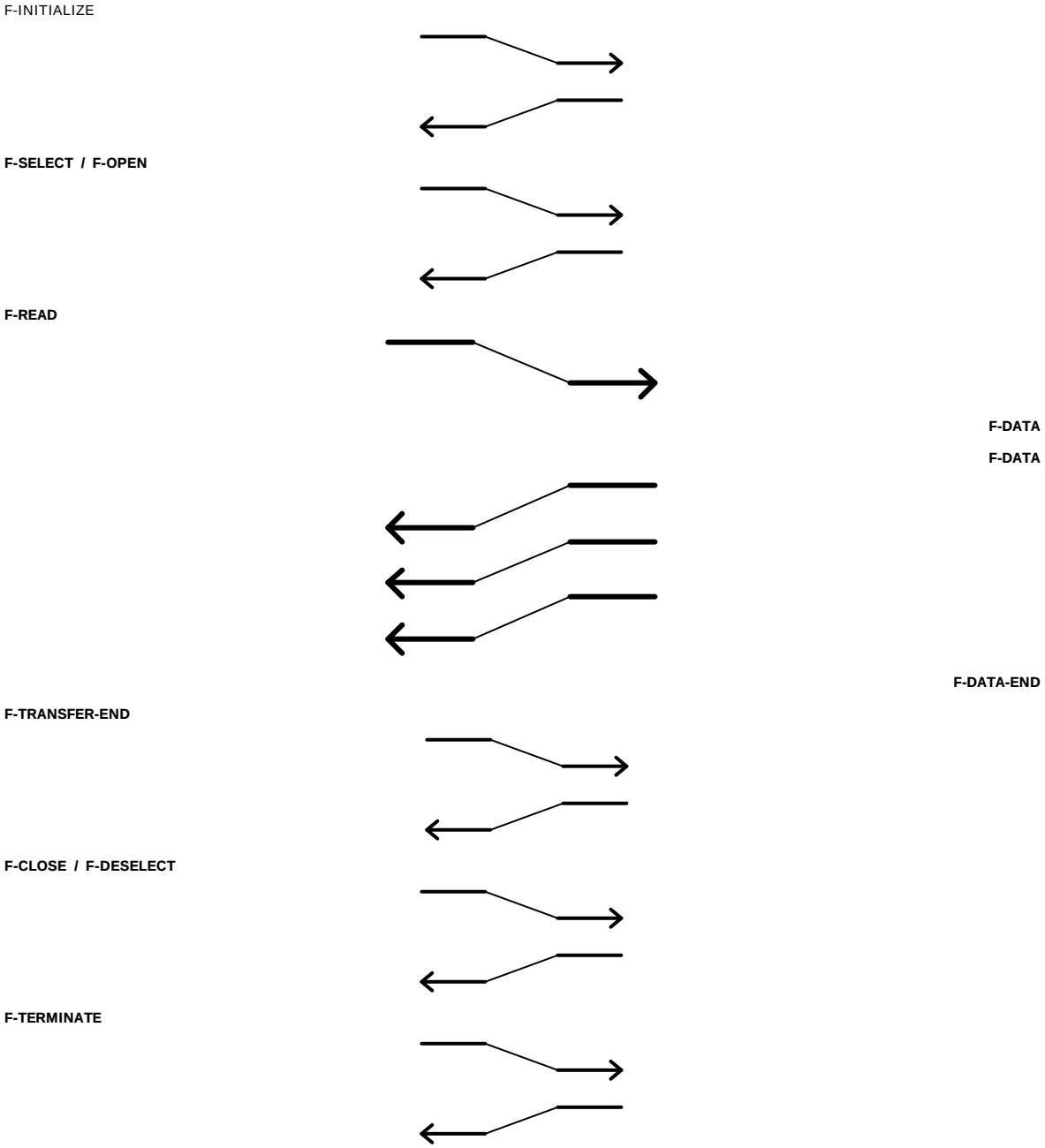
5. EXEMPLES



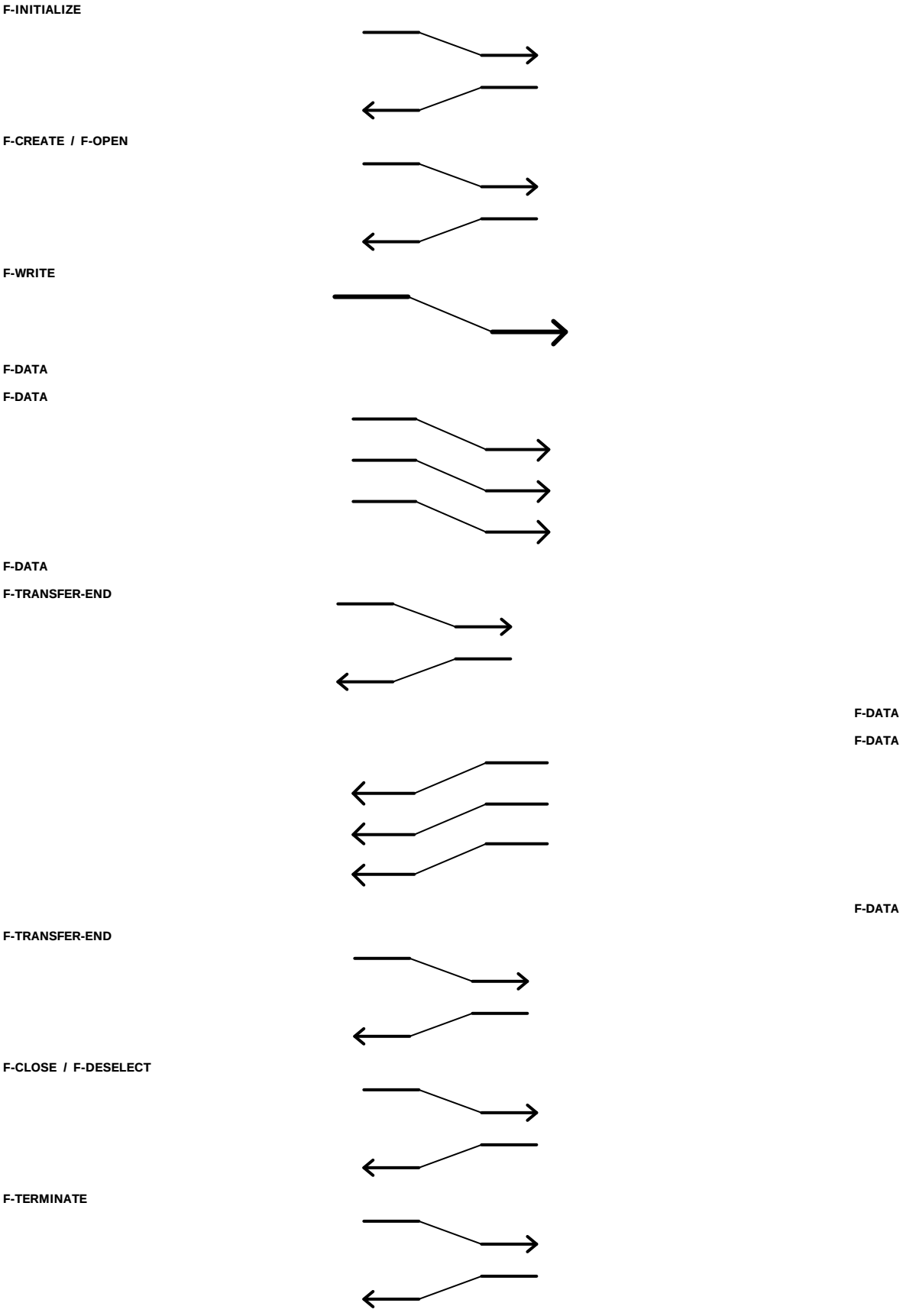
5.1 Transmission d'un fichier complet à un système distant



5.2 Accès à une base de données

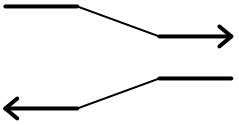


5.3 Serveur de fichiers sur un réseau local

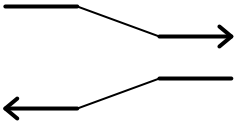


5.4 Administration de fichier

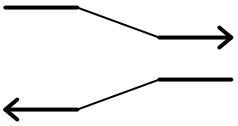
F-INITIALIZE



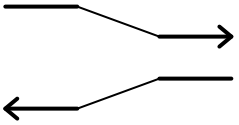
F-CREATE / F-OPEN



F-CLOSE/F-DESELECT



F-TERMINATE



ANNEXE 1 : OBJETS DEFINIS en syntaxe ASN.1

L syntaxe abstraite ASN.1 est décrite dans le standard ISO 8824. Elle correspond à la syntaxe de transfert CCITT X409 Contexte application [1] iso-ftam[1]

syntaxe abstraite [2] ftam-pci [1]
texte-simple [2]
texte-structure [3]
binaire-simple [4]
binaire-structure [5]
structure-hiérarchique [6]

syntaxe de transfert[3] ftam-pci [1]
texte-simple [2]
texte-structure [3]
binaire-simple [4]
binaire-structure [5]
structure-hiérarchique [6]

modèle de fichier [4] hiérarchique [1]

ensemble de contraintes [5]
non-structuré [1]
plat-sequentiel [2]
plat-ordonné [3]
plat-ordonné-nom-unique [4]
hiérarchique-ordonné [5]
hiérarchique-général [6]
hiérarchique-général-nom-unique [7]

type de documents [6] texte-non-structuré [1]
texte-séquentiel [2]
binaire-non-structuré [3]
binaire-séquentiel [4]
hiérarchique-simple [5]

ANNEXE 2 : MODELE DE SYSTEME DE FICHIERS

A.2.1 Concepts de base

Le système de fichiers comporte un nombre quelconque de fichiers

Les propriétés de chaque fichier sont décrites par l'ensemble des attributs du fichier. Ces attributs sont globaux et accessibles à tous les utilisateurs à un instant donné.

Tout fichier peut être vide ou avoir un contenu et une structure. Des attributs identifient les aspects structuraux du contenu.

Il existe un ensemble d'attributs d'activité FTAM associé à chaque régime. Ces attributs sont de deux types.

Pour le premier, il y a correspondance un pour un avec les attributs du fichier. Il indique la valeur active des attributs tels qu'ils sont perçus par l'initiateur.

Le second fournit la valeur courante de l'information d'état concernant les échanges en cours. Ces attributs sont dérivés des paramètres du protocole.

Un nombre arbitraire d'utilisateurs (supérieur ou égal à zéro) peut avoir initialisé des régimes FTAM à un instant donné, mais les échanges entre initiateur et répondeur ne correspondent, à un instant donné, qu'à un seul fichier dans le système de fichier du répondeur, fichier lié à un régime FTAM particulier.

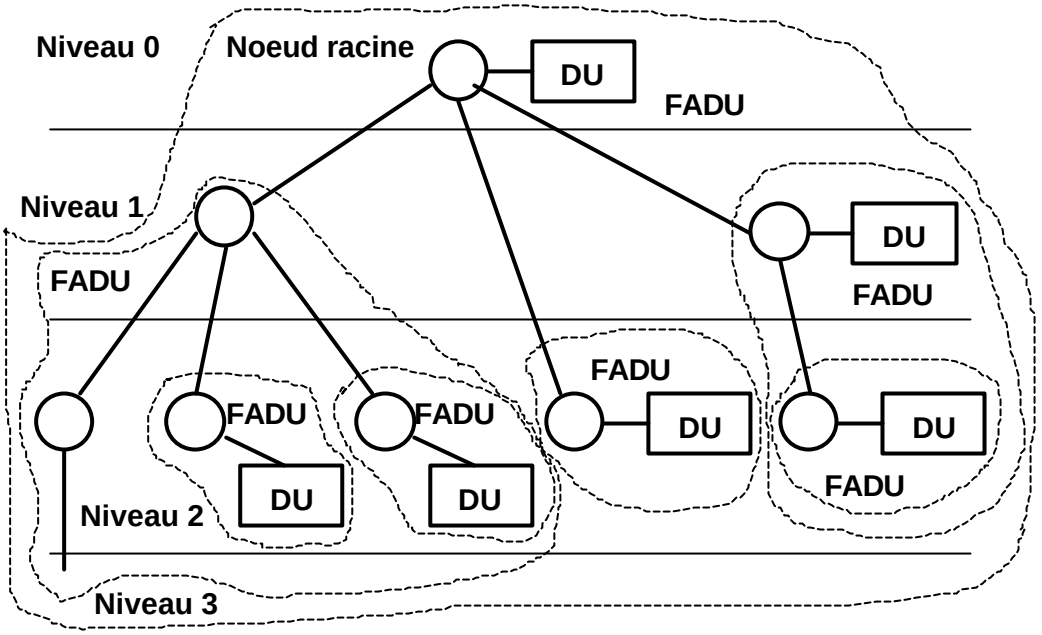
NB : Voir schéma § 3.3 page 6

A.2.2 Structure d'accès

Le modèle de système de fichiers FTAM a une **structure hiérarchique**. La structure d'accès aux fichiers est un **arbre ordonné**.

A chaque noeud de cet arbre est attaché **zéro ou une** unité de données.

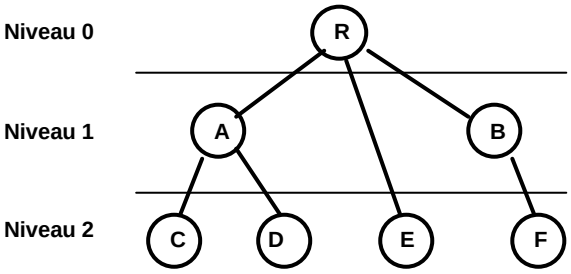
Chaque noeud dans la structure donne accès à un sous-arbre. Ce sous-arbre est appelé **unité de données d'accès au fichier** ou FADU (File Acces Data Unit). Une FADU comporte les noeuds du sous-arbre et les unités de données qui leur sont attachées. Le noeud racine donne accès au fichier entier.



Optionnellement un noeud peut recevoir un nom

Le nombre de niveaux, le nombre d'arcs originaire d'un noeud, et la longueur d'un arc peuvent être quelconque.

Un **préordre** est établit pour balayer l'arbre. Il suit une séquence transversale. L'exemple ci-dessous illustre ce préordre :



Cet arbre sera balayé dans l'ordre
R, A, C, D, E, B, F

* On ajoute le noeud racine d'un sous-arbre à la fin de la séquence transversale établie jusqu'ici.

* Chaque enfant d'un sous-arbre est traité dans son ordre d'apparition.

A.2.3 Description formelle

En syntaxe ASN.1 une FADU est définie comme ci-dessous :

```
FADU DEFINITIONS ::=
  BEGIN
      FADU ::= Subtree
     Subtree ::= SEQUENCE  E
     node Node-Descriptor-Data-Element,
     data DU OPTIONAL,
     children    Children OPTIONAL     ]

      Children ::= SEQUENCE E
      enter-subtree Enter-Subtree-Data-Element,
      SEQUENCE OF    Subree,
      exit-subtree  Exit-Subtree-Data-Element   ]

      DU  ::= SEQUENCE OF File-Contents-Data-Elements

      Node-Descriptor-Data-Element ::=
        [APPLICATION 20] IMPLICIT SEQUENCE E
      name Node-Name OPTIONAL,
      arc-length  [1] IMPLICIT INTEGER DEFAULT 1,
      data-exists   [2] IMPLICIT BOOLEAN DEFAULT TRUE ]

      Node-Name ::= CHOICE E
      ftam-coded  [0] IMPLICIT GraphicString,
      user-coded  EXTERNAL ]

      Enter-Subtree-Data-Element ::=
        [APPLICATION 21] IMPLICIT NULL

      Exit-Subtree-Data-Element  ::=
        [APPLICATION 22] IMPLICIT NULL

      File-Contents-Data-Elements ::= EXTERNAL

      Data-Element  ::= CHOICE E
      Node-Descriptor-Data-Element,
      Enter-Subtree-Data-Element,
      Exit-Subtree-Data-Element,
      File-Contents-Data-Element  ]

  END
```

A.2.4 Structure de transfert

La structure de transfert FTAM est dérivée de sa syntaxe abstraite. Un fichier est transféré comme une **suite d'éléments de données (Data-Elements définis ci-dessus)**.

Les informations structurantes : Descripteur de noeud, sous-arbre d'entrée, sous-arbre de sortie sont transmises dans la PCI.

Le contenu du fichier est transmis dans les données utilisateur. PCI et Données utilisateur utilisent des **contexte de transfert différents**.

Sept contextes d'accès sont définis :

HA Tous les éléments de la FADU sont transférés

HD La structure d'accès (PCI) seule est transférée. Il n'y a pas de contenu de fichier

FA Fichier plat. On transfert le descripteur de noeud et le contenu du fichier.

FL Le descripteur de noeud et le contenu de tous les fichiers d'un niveau qui contiennent des données son transférés.

FS Le descripteur de noeud et tous les contenus de fichiers appartenant à ce noeud racine sont transférés.

UA Non structuré. Seuls sont transférés les éléments de données de type contenu de la FADU adressée.

US Les éléments de données de type contenu appartenant au noeud racine de la FADU sont transférés.

A.2.5 Actions sur un fichier

Sur un fichier complet

créer
sélectionner (créer une relation)
changer les attributs
lire les attributs
ouvrir un fichier
clore un fichier
désélectionner
effacer(delete : désélectionne et efface)

Pour l'accès à un fichier

localiser première FADU
dernière FADU
FADU courante
FADU suivante
FADU précédente
début
fin
Nom de noeud
séquence de noms de noeuds
numéro de noeud (dans le préordre racine = 0)
numéro de niveau (seulement pour accès FL)

lire localiser et lire la FADU
insérer
remplacer
étendre
gommer (erase) (sauf racine)