UNIVERSITE MOHAMED BOUDIAF – M'SILA
FACULTE DES MATHEMATIQUES ET DE L'INFORMATIQUE

DEPARTEMENT D'INFORMATIQUE

Réseaux de Communication

Programme de Deuxième Année Licence en Informatique

Notes de Cours
Dr. Lamiche Chaabane

Version 2017
Site Web : http//www.univ-msila.dz

I - Introduction aux Réseaux Informatiques

1. Définition

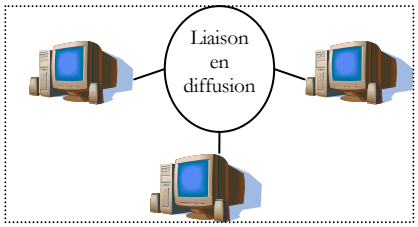
Un réseau informatique est constitué de plusieurs machines indépendantes pouvant échanger des informations via un moyen de communication.

2. Classification des réseaux :

- Technique de transmission.
- Taille du réseau (distance).

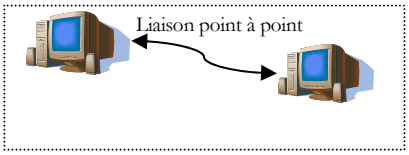
2.1. Classification des réseaux selon la technique de transmission

2.1.1. Réseaux à diffusion



Ces réseaux possèdent *un seul canal de communication* que toutes les machines partagent. Lorsque une machine envoie un message, ce dernier est reçu par toutes les machines du réseau. Le message contient l'*adresse* de la machine destinataire, A la réception, une machine teste cette adresse pour savoir si elle doit traiter ce message ou l'ignorer. Un
- diffuser vers toutes les machines du réseau ⇒ *Broadcasting*
- diffuser vers un sous ensemble de machines ⇒ *Multicasting*

2.1.2. Réseaux point à point



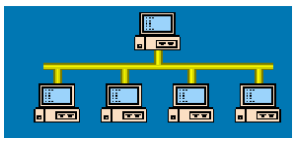
Ces réseaux sont formés d'un certain nombre de connexions entre les machines prises *deux à deux*. Pour aller de la source à la destination, un message doit passer par une ou plusieurs *machines intermédiaires*. Les réseaux de petites tailles sont des réseaux à diffusion alors que les grands réseaux sont du type point à point.

2.2. Classification des réseaux selon la distance

2.2.1. Les réseaux locaux *(LAN : Local Area Network)* : pour une entreprise

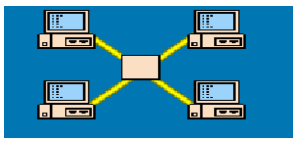
2.2.1.1. Topologies des réseaux locaux : contrairement à *la topologie logique* d'un réseaux qui décrit son mode de fonctionnement pendant la transmission de données, *la topologie physique* d'un réseau décrit la disposition des ordinateurs, des câbles et des autres composants d'un réseau.

A. Topologie en bus



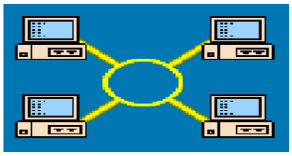
Dans la topologie en bus, tous les ordinateurs sont reliés au même câble. Chaque extrémité est reliée à une terminaison. En cas de rupture du câble en un point, toutes les communications sont interrompues.

B. Topologie en étoile



Tous les ordinateurs sont reliés à l'aide d'un câble à un concentrateur. Si l'un des câbles se rompt seul l'ordinateur relié à ce câble en est affecté, toutefois, si le concentrateur tombe en panne, l'ensemble des ordinateurs ne peut plus communiquer.

C. Topologie en anneau



Dans une topologie en anneau, les ordinateurs sont reliés à un seul câble en anneau.

2.2.2. Les réseaux métropolitains (*MAN : Metropolitan Area Network*) : à l'échelle d'une ville.

2.2.3. Les réseaux étendus (*WAN : Wide Area Network*) : à l'échelle d'un pays

3. Modèle OSI (*Open System Interconnection*) : élaboré par l'organisation internationale de normalisation ISO en 1984. Comprend 7 couches numérotées de bas en haut.

- Les couches basses (1-4) : transfert de l'information par les différents services de transport ;
- Les couches hautes (5-7) : traitement de l'information par les différents services applicatifs.

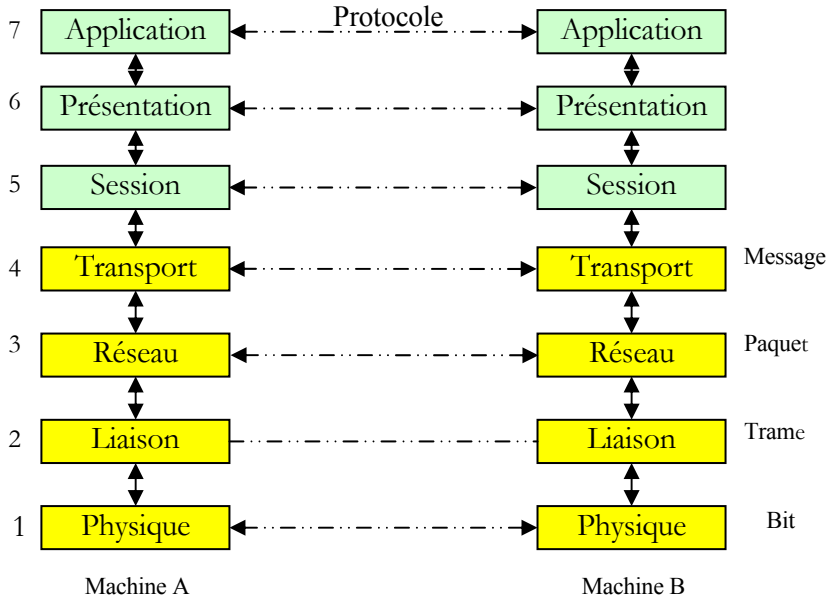


Fig. 1 : *Modèle OSI.*

⇒ Fonctionnement :

- *Un protocole* : est l'ensemble de procédures et informations échangées pour établir et gérer une communication. Les formats des informations font partie intégrante du protocole.
- Les couches de même niveau ne communiquent pas directement entre elles, mais via un mécanisme qui est transparent ; on parle *de communication virtuelle*.
- *Service :* opérations offertes par une couche inférieure à une couche supérieure.
- PDU (Protocol Data Unit): c'est l'unité de données protocolaire (trame, paquet, segment, message…).
- L'encapsulation c'est l'insertion des données de la couche supérieur dans la structure de données de la couche inférieure. La donnée à transmettre, appelée SDU (*Service Data Unit*), est donc associée aux informations de contrôle, appelées PCI (*Protocole Control Information*), par encapsulation dans une trame de données, appelée PDU (*Protocole Data Unit*).

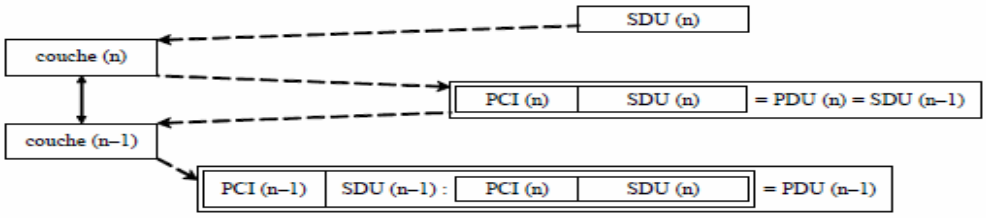


Fig. 2 : *SDU, PDU et encapsulation.*

⇒ Fonctions des couches :

1. *Couche Physique :* décrit les caractéristiques électriques et les équipements de transmission.
2. *Couche liaison de données* : le contrôle d'erreur de transmission.
3. *Couche réseau :* trouver une route pour acheminer les paquets de la source à la destination
4. *Couche transport :* fractionnement + réassemblage + réagencement ordonné des paquets.
5. *Couche session :* permet aux utilisateurs sur des machines différentes d'établir des sessions entre eux. L'établissement de point de reprise sur les transmissions longues pour leur permettre de reprendre en cas de panne.
6. *Couche présentation :* conversion de données + compression et cryptage de données.

7. *Couche application :* interfaces de communication aux utilisateurs (http, FTP, …).

4. Equipements d'interconnexion : nécessaires pour relier les éléments des réseaux

- ***Carte réseaux :*** la fonction d'une carte réseau consiste à adapter un ordinateur au support physique de transmission. Chaque carte réseau porte une adresse unique appelée adresse MAC (*Medium Access Control*) constituée de 48 bits (6 octets) et est généralement représentée sous la forme hexadécimale en séparant les octets par un double point ou un tiret. Par exemple 5E:FF:56:A2:AF:15.

- ***Les concentrateurs (Hub):*** le concentrateur est un équipement physique à plusieurs ports. Il sert à relier plusieurs ordinateurs entre eux. Son rôle c'est de prendre les données reçues sur un port et les diffuser sur l'ensemble des ports.

- ***Le commutateur (Switch) :*** consiste à connecter plusieurs ordinateurs entre eux. Il prend des décisions en fonction des adresses *MAC*, ce qui lui permet d'envoyer les données uniquement sur le port auquel le bon ordinateur est connecté.

- ***Le répéteur (repeater) :*** est un équipement qui sert à régénérer le signal entre deux nœuds pour le but d'étendre la distance du réseau. Il est à noter qu'on peut utiliser un répéteur pour relier deux supports de transmission de type différents.
- ***Le pont (bridge) :*** est un équipement qui sert à relier deux réseaux utilisant le même protocole. Quand il reçoit la trame, il est en mesure d'identifier l'émetteur et le récepteur ; comme ça il dirige la trame directement vers la machine destinataire
- ***La passerelle (gatway) :*** est un système matériel et logiciel qui sert à relier deux réseaux utilisant deux protocoles et/ou architectures différents ; comme par exemple un réseau local et internet.
- ***Le routeur :*** est un matériel de communication de réseau informatique qui a pour rôle d'assurer l'acheminement des paquets, le filtrage et le control du trafic. Le terme router signifie emprunter une route. Le routage est la fonction qui consiste à trouver le chemin optimal que va emprunter le message depuis l'émetteur vers le récepteur.

- ***Le modem (modulateur-démodulateur) :*** est un équipement qui sert à lier le réseau téléphonique au réseau informatique. le modem a pour rôle de convertir le signal numérique en signal analogique et vis versa. Le modem utilise donc les techniques de modulation et de démodulation.
- Les symboles schématiques et l'emplacement des différents équipements dans le modèle OSI est donné la figure 1.9 ci-dessous :

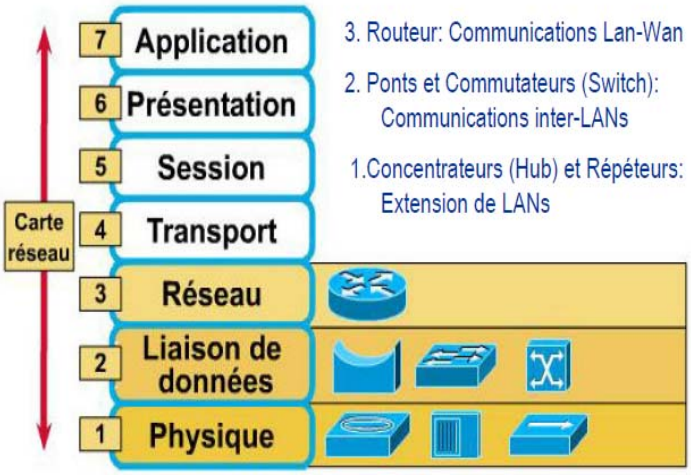


Fig. 3 : *Equipements d'interconnexion.*

II - Couche Physique

1. Fonction : la couche physique définit les caractéristiques électriques et mécaniques ainsi que les procédures et les fonctions permettant d'activer, de maintenir et de désactiver la liaison physique entre les machines d'extrémité.

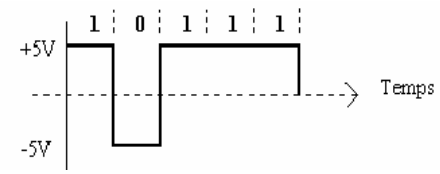
2. Supports de transmission : câbles électriques, fibre optique, espace hertzien (réseaux sans fils).

- *Câble électrique :* fils de cuivre + moins cher + sensible aux perturbations électromagnétiques
- *Câble à fibre optique :* les bits sont convertis en *faisceaux lumineux* + plus cher + insensible aux interférences + débit de données plus élevés.
- *Les ondes électromagnétiques :* les réseaux sans fils utilisent des ondes électromagnétiques qui peuvent circuler dans le vide ou dans l'air.

3. Principes de transmission

3.2. Transmission numérique : correspond au codage de l'information sous forme de succession de 0/1

3.2.1 Le code NRZ

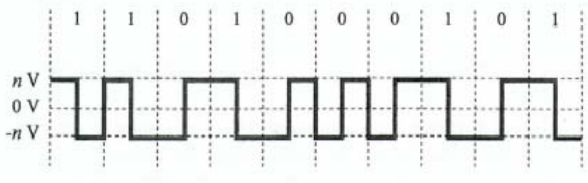


Le code NRZ (*No return to Zero*) code :

- Le bit 1 par un signal positif,
- Le bit 0 par un signal négatif.

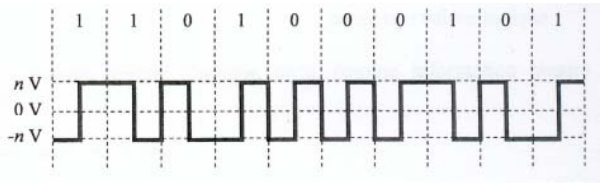
3.2.3. Le code Manchester

Le code Manchester (biphasé) est basé sur une variation du signal. Il s'agit d'observer la variation au signal entre le début et la fin du temps élémentaire.



- Le bit 1 est codé par une variation de +V à –V
- Le bit 0 est codé par une variation de -V à +V

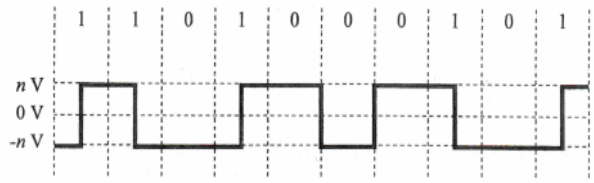
3.2.4. Le code Manchester différentiel



De la même façon, ce code (biphasé différentiel) est basé sur les transitions du signal.

- Le bit 0 est codé par une transition en début du temps élémentaire
- Le bit 1 est codé de la même façon par une transition en milieu du temps élémentaire

3.2.5. Le code de Miller



Le code de Miller est également basé sur une codification à partir des transitions du signal

- Le bit 0 est codé par l'absence de transition pendant le temps élémentaire
- Le bit 1 est codé par une transition

Pour éviter le problème de synchronisation aux longues séquences de 0, une transition en cas de succession de 0 est réalisée en début de chaque temps élémentaire.

3.3. Quelques définitions

Bande passante : est l'intervalle de fréquences à l'intérieur duquel les signaux seront correctement transmis.

- *Valence d'une voie*

Un codage associé une valeur physique (un signal électrique) à une valeur logique (un signal binaire). La valence notée *V* est le nombre de valeurs que peut prendre l'état physique à un instant. Par exemple, dans les cas précédents, on parle des bivalences (le signal peut prendre 2 valeurs de tension +V ou –V)

- *Débit binaire*

Egalement appelé vitesse de transmission, c'est le nombre de valeurs logiques transmises par seconde. Il est noté *D* et s'exprime en bit/s.

$$D = \frac{R_m}{k} \log_2(V)$$

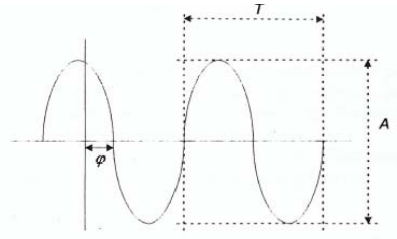
k correspond au nombre de valeurs physiques utilisées pour coder une information.

Loi de Nyquist : si une ligne de transmission possède une largeur de bande passante *W*, alors sa vitesse de modulation maximal (appelée aussi capacité du voie de transmission) est : $R_{max} = 2.W$ en Bauds, tel que *W* est exprimée en *Hz*. Si chaque signal permet de transmettre *n* bits alors la capacité du voie de transmission $C=2.n.W$ b/s. Tel que : *n* est le nombre de bit/signal avec $n = \log_2 l$ et l : nombre de valeurs possibles.

Théorème de Shanonn : si une ligne de transmission possède une largeur de bande passante *W* alors son débit binaire maximal (capacité de voie de transmission) est donnée par : $C=W.\log_2(1+S/B)$ en bauds.

- Le rapport S/B correspond aux perturbations électromagnétiques de l'environnement
- La valeur du rapport S/B en décibels (*db*) est donnée par : $N_{db}=10\log_{10}(S/B)$

3.4. Transmission analogique

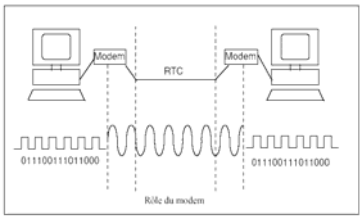


Dans ce cas, le signal est représenté par une onde sinusoïdale appelée *onde porteuse* de forme : $S(t)=A.Sin(\omega t+\varphi)$

- A est l'amplitude, *t* est le temps
- ω est la pulsion = $2\pi f$ (où f : fréquence du signal en Hz)
- φ est la phase (décalage par rapport à l'origine)

La transmission des signaux analogiques est réalisée à l'aide des appareils appelées des modems

3.4.1 Codage de l'information analogique



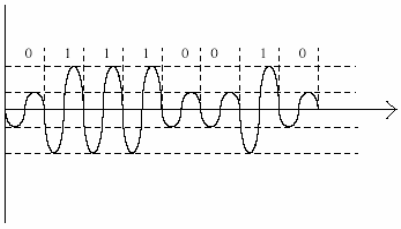
- *Modulation :* transformation du signal numérique en analogique.
- *Démodulation :* transformation du signal analogique en numérique.

La transformation par le modem est réalisée soit par :

1) Modulation d'amplitude, 2) Modulation de phase.

3.4.1.1 Modulation d'amplitude

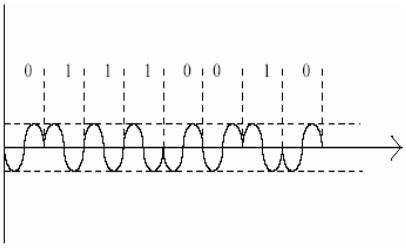
Numérique	Analogique
0	Amplitude A1
1	Amplitude A2



Cette modulation (*AM : Amplitude Modulation*) s'appuie sur l'utilisation de 2 amplitudes pour coder les 2 valeurs du signal. Dans ce cas une seule fréquence est utilisée pour transmettre l'information.

3.4.1.2 Modulation par saut de phase

Numérique	Analogique
0	Phase 1
1	Phase 2



Cette modulation (*PSK : Phase Shift Keying*) utilise la phase du signal mais surtout le déphasage afin de coder les différentes valeurs du signal.

III- Couche Liaison de Données

1. Fonctions de LDD

- Traitement des données reçues de la couche réseau en émission ou de la couche physique (en réception) ⇒ constitution des trames ;
- Contrôle d'erreurs qui peuvent survenir en cours de transmission ;
- Gestion des acquittements ⇒ pour détecter les erreurs dans les échanges de trames ;

2. Constitution des trames

La couche LDD décortique le flux venant de la couche réseau en un ensemble d'unités manipulables nommées Trames. ➔ Une trame est un ensemble de bits dont le début et la fin sont marqués pour faciliter sa reconnaissance.

- Eviter le monopole du support par le transfert d'un gros paquet en un seul bloc
- Eviter la retransmission d'un gros fichier en cas d'erreur.

Une trame a une taille fixe ⇒ la taille fixe délimite le début et la fin de la trame (exp.: ATM - 53 octets).
Une trame a une taille variable ⇒ ajouter un fanion (01111110) délimiteur entre deux trames consécutives

☞ **Pb :** le caractère 01111110 peut faire partie des bits de données ⇒ insérer à l'émission un 0 dès que l'on lit cinq 1 de suite. A la réception, A la réception, tout bit à '0' précédé de cinq '1' précédés d'un '0' est supprimé, cette technique est appelée traitement de *la transparence binaire*.

1101111111 → **01111110**11011111**0**11**01111110** → 1101111111

3. Contrôle d'erreur

3.1 Contrôle simple de parité

Ajouter un bit de parité aux *n* bits de l'information. *En cas de parité paire :* le bit de parité vaut 0 si le nombre de 1 dans le message original est paire et le bit de parité vaut 1 si ce nombre est impaire. *En cas de parité impaire faire l'inverse).*

Exemple : parité paire

Mot de données : 1001110 ⇒ mot de code (message à transmettre) : 10011100

A la réception on suppose qu'on a reçu : 11011100 ⇒ *il faut retransmettre le message.*

3.2 Contrôle de parité longitudinale et verticale (LRC[1] + VRC[2])

Soit à transmettre un bloc de *n* octets (généralement 7 octets de 7 bits) :

- Les octets sont structurés sous forme d'un tableau à 2 dimensions.
- Un bit de parité est calculé pour chaque ligne et pour chaque colonne.
- La dernière colonne constitue la parité verticale.
- La dernière ligne constitue la parité longitudinale.
- Le bloc est transmis ligne par ligne.

1	0	0	1	1	1	0	0	↓ LRC
0	0	1	1	1	0	0	1	
1	1	1	0	1	1	1	0	
0	1	0	0	1	0	1	1	VRC →

1	0	0	1	**0**	1	0	0
0	0	1	1	1	0	0	1
1	1	1	0	1	1	1	0
0	1	0	0	1	0	1	1

1 erreur + deux parités

[1] Longitudinal Redundancy Check
[2] Vertical Redundancy Check

3. Code à redondance cyclique (CRC : Cyclic Redundancy Check)

Le mot de donnée D= $d_1d_2\ldots d_K$ est représenté sous forme d'un polynôme $D(x) = d_1x^{k-1} + d_2x^{k-2} + \ldots + d_k$.

A l'émission,

- Multiplier D(x) par x^r, ce qui revient à ajouter r zéros à la fin du message original D(x).
- Effectuer la division suivante modulo 2 : $x^r.D(x)/G(x) = Q(x) + R(x)$
- Ignorer le quotient Q(x) et ajouter les r bits de R(x) à la fin du message original D(x) pour construire le polynôme T(x) qui représente le polynôme cyclique : c'est le message prêt à être envoyé.

A la réception,

- On divise le polynôme T(x) par G(x).
- Si le reste de la division = 0, il n'y a pas d'erreur.
- Si le reste de la division ≠ 0, il y a une erreur. Donc on doit retransmettre le message.

Exemple : soit le message 10111 à émettre et soit $G(x) = x^3+1$ le polynôme générateur. La chaîne binaire générer par G(x) est : 1001

A l'émission : Le message est : 10111 ⇒ le polynôme associé est :

$$D(x) = 1.x^4+0.x^3+1.x^2+1.x^1+1.x^0 \text{ alors}$$
$$D(x) = x^4+x^2+x+1$$

G(x) est de degré 3 : on multiplier D(x) par x^3 on trouve :

$$x^3.D(x) = x^7+x^5+x^4+ x^3$$

On divise ce résultat par le polynôme générateur G(x) :

$x^7+x^5+x^4+ x^3$	x^3+1
x^7+x^4	x^4+x^2+1
$x^5+ x^3$	
$x^5+ x^2$	
x^3+x^2	
x^3+1	
x^2+1	

Le reste de la division est : $R(x) = x^2+1$ ce qui donne la chaîne 101 donc :
Le mot de code par à transmettre par l'émetteur est : 10111**101**

A la réception : Supposons ici qu'une erreur est apparue sur le 5ième bit ; la chaîne binaire reçue est :
10110101 alors $T(x) = x^7+x^5+x^4+ x^2+1$
Pour détecter une éventuelle erreur, T(x) est divisé par G(x) :

$x^7+x^5+x^4+x^2+1$	x^3+1
x^7+x^4	x^4+x^2
x^5+x^2+1	
$x^5+ x^2$	
1	

Le reste de division n'est pas nul, une erreur est donc apparue ⇒ demander à l'émetteur de retransmettre le message originale.

IV- Couche Réseau

1. Fonctionnement

- Un message provenant de la couche transport peut être découpé en paquets, chacun étant routé par un chemin différant des autres. Chaque paquet contient l'*adresse complète* de destination.

2. Protocole IP *(Internet Protocol)*

Un paquets IP (*datagramme IP*) est constitué de deux parties : un *champ en-tête* et un *champ données*. L'en-tête comporte une partie fixe de 20 octets et une partie optionnelle de longueur variable.

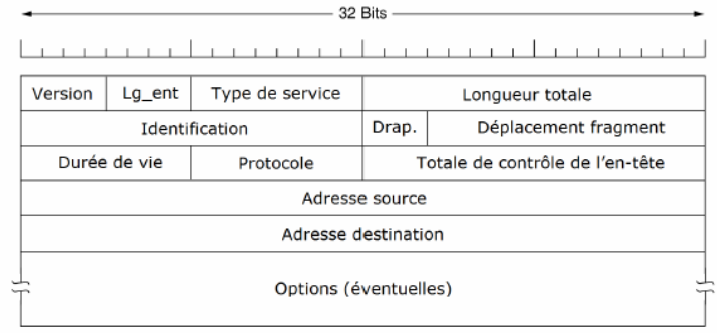


Fig. 4 :*Format de l'en-tête du paquet IP.*

- Les champs de l'en-tête sont comme suit :

Version **:** 4 bits contient la version du protocole IP (*IPv4* ou *IPv6*).

Lg_ent **:** 4 bits indique sa longueur en mots de 32 bits.
Type de service **:** le type d'acheminement souhaité : *délai*, *bande passante* et *fiabilité*.

Longueur totale : 16 bits indique la longueur en octets du paquet IP (en-tête + données utiles).

Identification : 16 bits contient un numéro d'identification pour le paquet.

Drapeau : 3 bits (dont un est inutilisé). Il comprend deux indicateurs: DF qui signifie "*ne pas fragmenter*" et MF qui signifie "*fragment non unique*".

Déplacement fragment : un numéro de séquence qui localise le fragment dans le paquet original.

Durée de vie : compteur pour limiter la durée de vie des paquets (255 sauts).

Protocole : indique à la couche réseau à quel protocole de transport (TCP/UDP).
Total de contrôle de l'en-tête : la détection d'erreur uniquement sur l'en-tête.

Adresse source, Adresse de destination: adresses des machines source et destination.

Options : pour offrir une extension aux évolutions du protocole IP.

2.1 Adresses IP

- Utilisée pour identifier une machine sur un réseau. Constituée de 4 octets séparés par un point.
- *Adresse IP* : *id_Réseau* + *id_ordinateur*. ⇒ 5 classes possibles

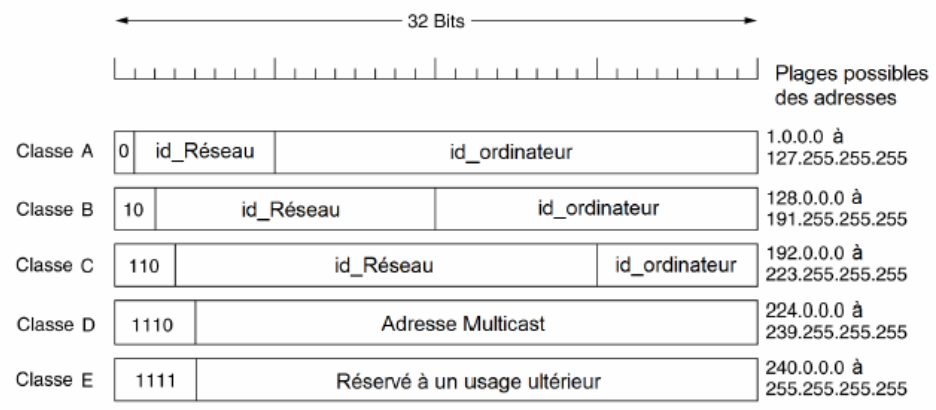


Fig. 5 :*Classes des adresses IP.*

2.2 Notions de masque

Adresse sur 32 bits pour séparer *id_réseau* de *id_ ordinateur*.

- bit de l'adresse IP définissant le réseau → bit correspondant du masque à 1 ;
- bit de l'adresse IP définissant l'hôte → bit correspondant du masque à 0.

⇒ *id_Réseau* = (Adresse IP) **ET logique** (Masque)

⇒ *id_ordinateu*r = (Adresse IP) **ET logique** (le complément à 1 du masque)

⇒ Pour obtenir l'adresse de broadcast (adresse de diffusion), tous les bits *de id_ordinateur* sont mises à 1. ou bien : (Adresse IP du réseau) **OU logique** (le complément à 1 du masque).

⇒ le masque s'écrit sous la forme /n, collée à l'adresse IP (le nombre de bits à 1 du masque en partant de la gauche). Par exemple, si l'adresse IP 192.168.1.72 est associée au masque de réseau 255.255.255.0, il sera abrégée en : 192.168.1.72 / **24** (les 24 premiers bits définissent l'identifiant réseau, et donc les 8 restants l'identifiant d'hôte). Cette écriture est appelée notation CIDR (*Classless Inter-Domain Routing)*

Exemple : Soit l'adresse IP 192.168.1.72/24, calculer: *id_réseau*, *id_ordinateur* et l'adresse de broadcast ?

Adresse IP	1100 0000 . 1010 1000 . 0000 0001 .	0100 1000	192.168.1.72
Masque	1111 1111 . 1111 1111 . 1111 1111 .	0000 0000	255.255.255.0
Identifiant réseau	1100 0000 . 1010 1000 . 0000 0001 .	0000 0000	192.168.1.0
Identifiant hôte	0000 0000 . 0000 0000 . 0000 0000 .	0100 1000	0.0.0.72
Adresse broadcast	1100 0000 . 1010 1000 . 0000 0001 .	1111 1111	192.168.1.255

Le complément à 1 du masque s'obtient on inversant les bits comme suit :
Complément à 1 du masque : 0000 0000 . 0000 0000 . 0000 0000 . 1111 1111 ⇒ 0.0.0.255

L'adresse de broadcast (diffusion) s'obtient alors :

192.168.1.72 **OU logique** 0.0.0.255 ⇒ 192.168.1.255

- Il faut noter que chaque classe possède un masque par défaut :

Classe	Masque en binaire	Masque en décimal	CIDR
Classe A	11111111.00000000.00000000.00000000	255.0.0.0	/8
Classe B	11111111.11111111.00000000.00000000	255.255.0.0	/16
Classe C	11111111.11111111.11111111.00000000	255.255.255.0	/24
Classe D	/	/	/
Classe E	/	/	/

2.3 Les sous-réseaux

- Pour mieux exploiter les adresses IP ⇒ partitionner le réseau en sous-réseaux à usage interne.
  ⇒ *On utilise une adresse IP (Classe A, B ou C) pour créer d'autres sous-réseaux.*

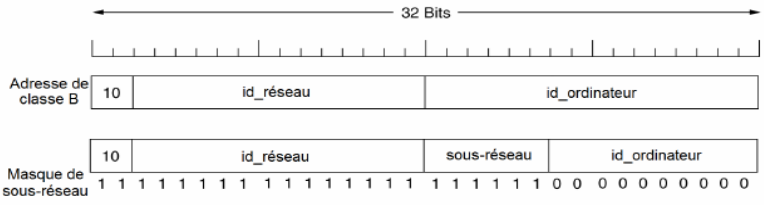


Fig. 6 :*Subdivision en sous-réseaux d'une adresse de classe B.*

Exemple

On considère le réseau de classe C, *id_réseau* : 192.168.1.0 avec le masque par défaut 255.255.255.0.
On veut découper ce réseau en 4 sous-réseaux.

1) Calcule du nombre de sous-réseau

Si l'on utilise 1 bit -> $2^1 = 2$ sous-réseaux,
Si l'on utilise 2 bits -> $2^2 = 4$ sous-réseaux. ⇒ 2 bits pour découper ce réseau en 4 sous réseaux.

2) Calcul du masque de sous-réseau

Le masque de chaque sous-réseau est obtenu en rajoutant 2 bits à 1 au masque initial.
Le masque de réseau par défaut est 255.255.255.0 : Soit 11111111.11111111.11111111.00000000
En ajoutant 2 bits on obtient 11111111.11111111.11111111.**11**000000
En fin, on a le masque de sous-réseau : 255.255.255.**192** ⇒ **/26**

3) Calcul du id_réseau de chaque sous-réseau

- Le *id-réseau* de chaque sous-réseau sera constitué de 26 bits
- Les 24 premiers bits seront ceux de l'écriture en binaire de 192.168.1.
- Les 2 bits suivants seront constitué du numéro du sous-réseau 00, 01, 10, 11 (On garde 00 et 11 relativement à la règle du document RFC1878 de 1995)

4) Calcul des adresses des sous-réseaux (id-réseau)

S/R1 : 192.168.1.00xxxxxx → 192.168.1.0
S/R1 : 192.168.1.01xxxxxx → 192.168.1.64
S/R1 : 192.168.1.10xxxxxx → 192.168.1.128
S/R1 : 192.168.1.11xxxxxx → 192.168.1.192

5) Calcul des id_ordinateur (Id-hôte) des sous-réseaux

• Adresse IP de S/R1 est : **192.168.1.0** 192.168.1.00 000000 : Non utilisable 192.168.1.00 000001 = 192.168.1.1 192.168.1.00 000010 = 192.168.1.2 _ . . . . | 62 machines . 192.168.1.00 111110 = 192.168.1.62 192.168.1.00 111111 : Non utilisable	• Adresse IP S/R2 est : **192.168.1.64** 192.168.1.01000000 : Non utilisable 192.168.1.01000001 = 192.168.1.65 192.168.1.01000010 = 192.168.1.66 _ . . . . | 62 machines . 192.168.1.01111110 = 192.168.1.126 192.168.1.01111111 : Non utilisable
Adresses IP de S/R3 est : **192.168.1.128** 192.168.1.10 000000 : Non utilisable 192.168.1.10 000001 = 192.168.1.129 192.168.1.10 000010 = 192.168.1.130 _ . . . . | 62 machines . 192.168.1.10 111110 = 192.168.1.190 192.168.1.10 111111 : Non utilisable	Adresses IP de S/R4 est: **192.168.1.192** 192.168.1.11000000 : Non utilisable 192.168.1.11000001 = 192.168.1.193 192.168.1.11000010 = 192.168.1.194 _ . . . . | 62 machines . 192.168.1.11111110 = 192.168.1.254 192.168.1.11111111 : Non utilisable

⇒ **Nombre de machines pour chaque sous réseau = $2^{\text{NB bits id\_ordinateur}}$ - 2 = 2^6-2 = 62 machines**

6) Calcul des adresses de diffusion

Pour obtenir l'adresse de diffusion dans chaque sous-réseau; on met à 1 tous les bits de *id_ordinateur*.

- L'adresse de diffusion de S/R1 est : 192.168.1.00**111111**, soit 192.168.1.63
- L'adresse de diffusion de S/R2 est : 192.168.1.01**111111**, soit 192.168.1.127
- L'adresse de diffusion de S/R3 est : 192.168.1.10**111111**, soit 192.168.1.191
- L'adresse de diffusion de S/R4 est : 192.168.1.11**111111**, soit 192.168.1.255

Ou la règle : $\text{Adresse}_{\text{broadcast}}$=(Adresse IP du S/réseau) **OU logique** (le complément à 1 du masque)

V- Couche Transport

1. Introduction

La couche transport ⇒ gère le *fractionnement* + *réassemblage* en paquets du flux de données + *réordonner* les paquets d'un même message ⇒ deux protocole *TCP* et *UDP*.

2. Adressage au niveau transport

- L'adresse IP ⇒ identifier une machine sur le réseau
- Le *port de connexion* ⇒ identifier une application sur la machine
- L'ensemble (adresse + port) noté *adresse:port* ⇒ est appelé *socket*
- Le couple de sockets *adresse_src:port_src / adresse_dst:port_dst* ⇒ communication s'établie.
- *Exemples* : 20-21 : FTP, 25 : SMTP, 80 : http

Ainsi un ordinateur émettant une requête http vers un serveur web établit donc une connexion entre un port local déterminé aléatoirement par le SE et le port 80 du serveur web, soit donc par exemple le couple de sockets : 212.195.198.187 :**2256** / 78.109.84.60 :**80**

3. Le Protocole TCP (*Transmission Control Protocol*)

La structure d'un segment TCP est illustrée dans la figure 6 ci-dessous :

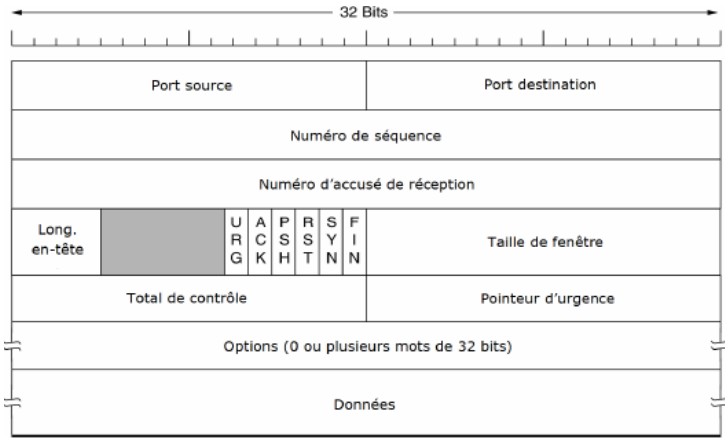


Fig. 6 : *Structure d'un segment TCP.*

La taille maximale d'un segment TCP (avec son en-tête) est de 65535 octets. Chaque segment débute par un en-tête d'une longueur fixe de 20 octets. Cet en-tête fixe peut être suivi par des options.

- *Port Source et Port Destination (16 bits)* ⇒ identifient les extrémités locales de la connexion.
- *Numéro de séquence (32 bits)* ⇒ indique le numéro de séquence du premier octet du segment.
- *Numéro d'accusé de réception (32 bits)* ⇒ indique le prochain octet attendu.
- *Longueur de l'en-tête TCP (4 bits)*⇒ indique combien de mots de 32 bits contient l'en-tête.
- un champ de 6 bits ⇒ n'est pas utilisé. On trouve ensuite un champ composé de 6 drapeaux d'un bit chacun :
- *URG* est positionné à 1 ⇒ la couche transport *envoie immédiatement* les données sans les stocker.
- *ACK* est positionné à 1 ⇒ pour indiquer la validité du numéro d'accusé de réception.
- *PSH* ⇒ la couche transport d'envoyer immédiatement les données et de ne pas les stocker.
- *RST* ⇒ demande de réinitialisation de connexion. *SYN* ⇒ demande d'une connexion
- *FIN* ⇒ utilisé pour libérer une connexion.
- *Taille de fenêtre (16 bits)* ⇒ nombre d'octets à transmettre.
- *Total de contrôle(16 bits)* ⇒ permet de détecter les erreurs de transmission.
- *Pointeur d'urgence(16 bits)*⇒ indique où l'on peut trouver les données urgentes .

3.1. Gestion d'une connexion TCP

Les connexions TCP sont établies à l'aide de la méthode *Three Ways Handshake* comme le montre la figure 7 ci-dessous :

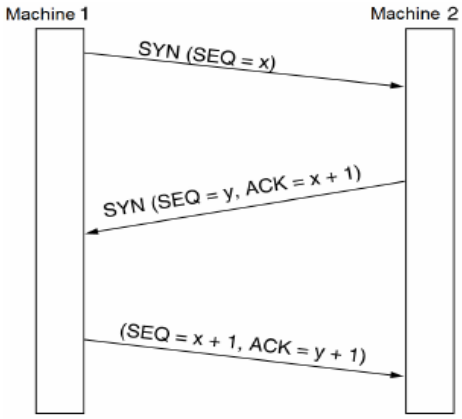


Fig. 7 : *Établissement d'une connexion TCP.*

- Une extrémité exécute la primitive CONNECT qui envoie un segment TCP avec le bit **SYN** à 1 et le bit *ACK* à 0 (CONNECTION REQUEST) et attend une réponse. Le champ *numéro de séquence* de ce segment contient un nombre aléatoire x.
- Quand ce segment arrive à destination, cette dernière peut soit accepter, soit rejeter la connexion. Si elle l'accepte, elle renvoie un segment TCP avec les deux bits **SYN** et **ACK** à 1 (CONNECTION ACCEPTED). Le champ *numéro de séquence* de ce segment contient un nombre aléatoire y, et le champ *accusée de réception* contient le nombre x + 1.
- Enfin, lorsque le segment CONNECTION ACCEPTED est reçu, un acquittement est envoyé, contenant dans le champ *accusée de réception* le nombre y+1. Il faut noter que ce dernier segment a le bit *SYN* à 0, en plus il peut transporter des données.

☞ Un principe similaire est mis en jeu à la fermeture de la connexion (Fig. 8) :

- La partie désireuse de finaliser la connexion envoie un segment avec les bits FIN à 1 et ACK à 0, ainsi qu'un numéro de séquence (X) (fermeture avec la primitive *close*) ;
- L'autre partie acquitte ce segment reçu en renvoyant un segment avec le bit ACK à 1 et un numéro d'acquittement (X+1); la connexion est maintenant fermée dans le sens *demandeur_finalisation*→ *acquitteur_finalisation*, il ne reste qu'une « demi »-connexion ; seuls des acquittements peuvent être envoyés par le *demandeur_finalisation* ;
- Même chose pour l'autre partie lorsqu'elle désire fermer sa « demi »-connexion.

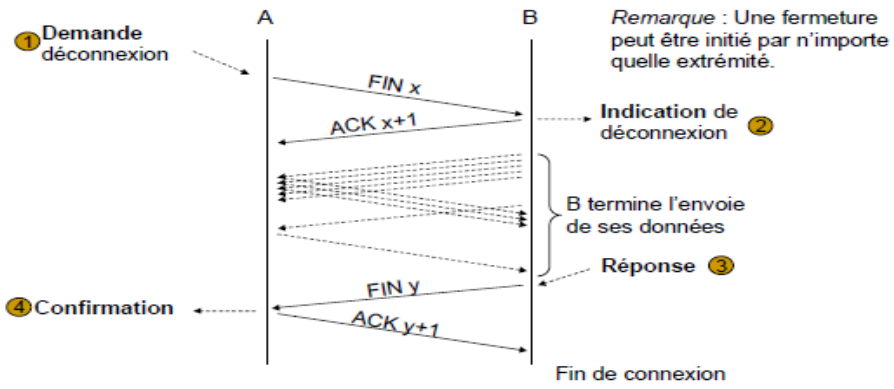


Fig. 8 : *Fermeture d'une connexion TCP.*

3.2 Contrôle de flux TCP

- Dans TCP, la gestion de la *fenêtre d'anticipation* de fait de manière *dynamique* (appelée aussi *fenêtre glissante*). Ainsi, la taille de la fenêtre peut être *ajustée* par le récepteur selon ses capacités.
- Supposons, par exemple, que le récepteur possède un tampon mémoire de 4 Ko comme indiqué à la figure 9 :

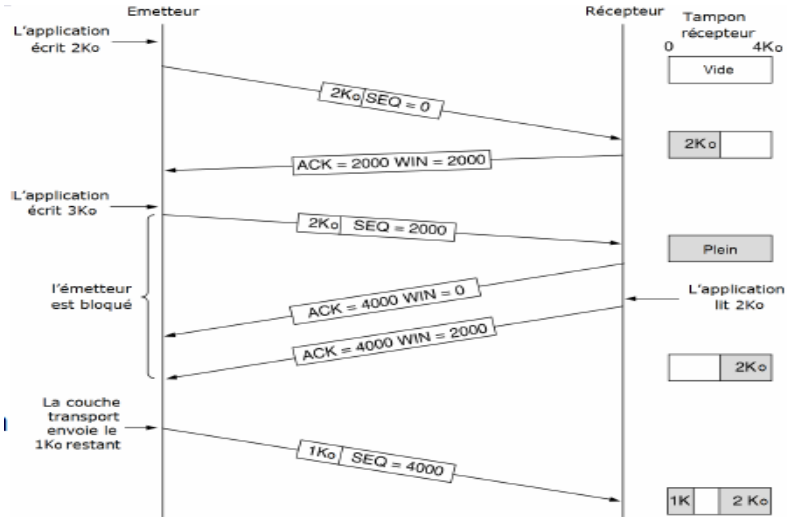


Fig. 9 : *Gestion de la fenêtre d'anticipation.*

- Si l'émetteur transmet un segment de 2000 octets, le récepteur acquitte ce segment. Toutefois, le récepteur n'a maintenant que 2000 octets de libre dans son tampon mémoire (jusqu'à ce que l'application y retire des données), il annonce donc une fenêtre de 2000 octets.
- L'émetteur envoie alors de nouveau 2000 octets, qui sont acquittés, mais la taille de la fenêtre devient *nulle*. L'émetteur doit donc *s'arrêter* le temps que l'application située sur l'ordinateur récepteur ôte des données du tampon mémoire, ce qui permet alors à la couche transport d'*annoncer* une fenêtre plus grande.
- Il est à noter que l'émetteur n'est pas obligé de *transmettre immédiatement* les données qui proviennent des applications. Par exemple, dans la figure précédente, quand les deux premiers Kilo-octets de données arrivent, la couche transport aurait pu tout à fait attendre le bloc suivant de 2 Ko pour transmettre un unique segment de 4 Ko. On utilise souvent cette caractéristique de TCP pour améliorer les *performances* du réseau.

VI - Couche Application

1. Introduction

Les couches en dessous de la couche application assurent l'acheminement des données, mais ils n'effectuent pas de travail réel pour les utilisateur. La couche application est l'interface entre l'utilisateur et le réseau via un ensemble de protocoles. Citons en particulier :

- **SMTP** (Simple Mail Transfer Protocol) : transfère du courrier électronique.
- **HTTP** (HyperText Transfer Protocol) : transfère de pages Web.
- **FTP** (File Transfer Protocol) : transfère de fichiers.
- **…**

2. Protocoles de la couche application

2.1. Système de noms de domaine (DNS)

Au niveau application, on n'utilise pas d'adresses en binaire pour les machines, boîtes aux lettres et autres ressources. On fait usage, le plus souvent, de chaînes de caractères du type *monnom@gmail.com*. Cependant, le réseau ne comprend que les adresses binaires, d'où la nécessité d'un mécanisme de traduction des chaînes de caractères en adresses réseau. Cette traduction dans Internet est assurée par le système **DNS** (*Domain Name System*).

Description du DNS

- Au coeur du DNS il y a un schéma de nommage hiérarchique et une base de données répartie qui implémente ce schéma de nommage.
- Il est principalement utilisé pour faire correspondre aux *adresses IP* les noms de machines et les destinations de courrier électronique (Fig. 10).
- Le **DNS** est utilisé comme suit. Pour trouver une adresse IP correspondant à un *nom*, une application appelle une primitive nommée *solveur*, et lui passe le nom en paramètre. Le *solveur* envoie alors un segment UDP à un *serveur* DNS qui cherche le nom dans sa table et renvoie l'adresse IP correspondante. Connaissant l'adresse IP, l'application peut ainsi établir avec la destination une connexion TCP ou bien lui envoyer un segment UDP.

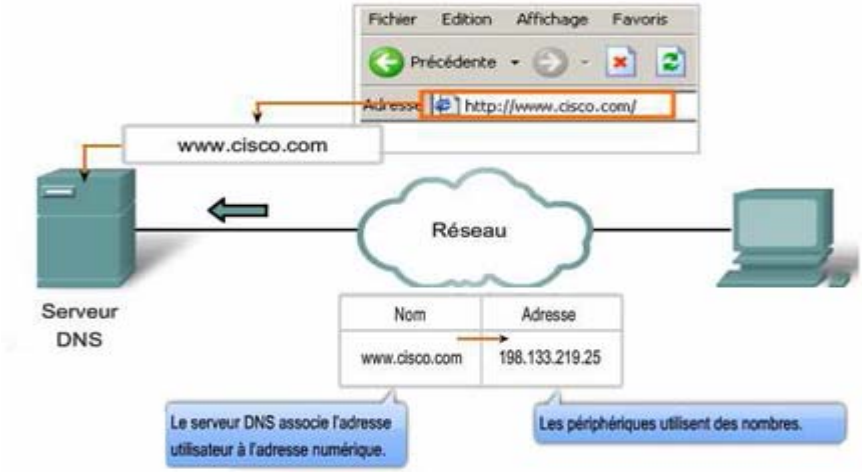


Fig. 10 : *Résolution d'adresses DNS.*

2.2. Protocole HTTP

Le protocole HTTP (*Hypertext Transfer Protocol*), l'un des protocoles de la suite TCP/IP, a été à développé à l'origine pour publier et extraire des pages HTML.

Lorsqu'une adresse Web (ou URL) est tapée dans un navigateur Web, ce dernier établit une connexion au service Web s'exécutant sur le serveur à l'aide du protocole HTTP. Les URL (*Uniform Resource Locator*) sont les noms que la plupart des utilisateurs associent aux adresses Web. Par exemple : *http://www.cisco.com/index.html* est une adresse URL qui se réfère à une ressource spécifique ; une page Web nommée index.html située sur le serveur cisco.com

Pour accéder au contenu Web, les clients Web (*navigateurs*) établissent des connexions au serveur et demandent les ressources voulues. Le serveur retourne les ressources et, à la réception de ces ressources, le navigateur interprète les données et les présente à l'utilisateur (Fig. 11).

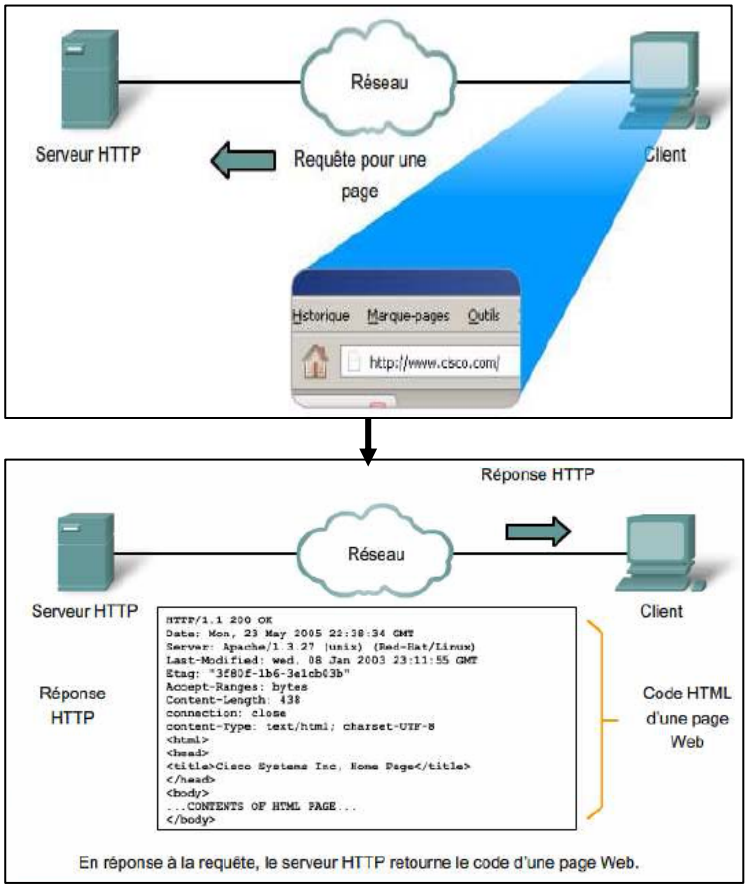


Fig. 11 : *Fonctionnement du protocole HTTP.*

Le protocole HTTP constitue un protocole de requête/réponse. Lorsqu'un client (généralement un navigateur Web) envoie une requête à un serveur, le protocole HTTP définit les types de messages que le client utilise pour demander la page Web, ainsi que les types de messages que le serveur utilise pour répondre. Les trois types de messages courants sont GET, POST et PUT :

- **GET** est une requête cliente pour obtenir des données.
- **POST** et **PUT** sont utilisées pour envoyer des messages qui téléchargent des données vers le serveur Web. Par exemple, lorsque l'utilisateur entre des données dans un formulaire incorporé à une page Web, la requête POST comprend les données dans le message envoyé au serveur.
- PUT permet de télécharger des ressources ou du contenu vers le serveur Web.

2.3. Protocole FTP

Le protocole FTP (*File Transfer Protocol*) est un autre protocole de couche application couramment utilisé. Il a été développé pour permettre le transfert de fichiers entre un client et un serveur. Un client FTP est une application s'exécutant sur un ordinateur et utilisée pour extraire des fichiers d'un serveur exécutant FTP. Pour transférer les fichiers correctement, le protocole FTP nécessite que deux connexions soient établies entre le client et le serveur : une connexion pour les commandes et les réponses et une autre pour le transfert même des fichiers (Fig. 12).

- Le client établit la première connexion au serveur sur le port TCP 21. Cette connexion est utilisée pour le trafic de contrôle et se compose de commandes clientes et de réponses serveur.
- Le client établit la seconde connexion au serveur via le port TCP 20. Cette connexion est destinée au transfert même des fichiers et est établie à chaque transfert de fichiers.
- Notons que le client peut télécharger un fichier à partir du serveur ou en direction du serveur.

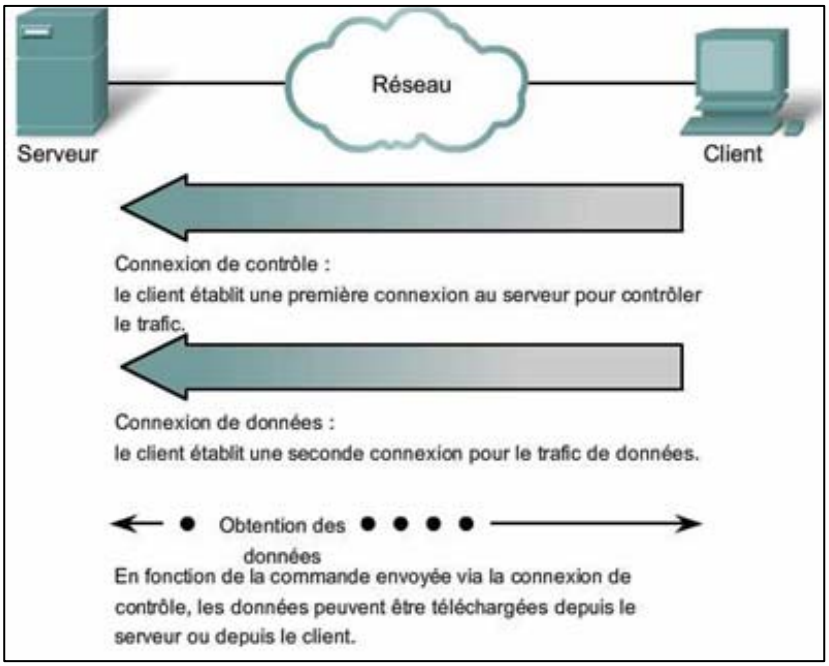


Fig. 12 : *Processus FTP.*

Références

1. Andrew Tanenbaum et David J.Wetherall, Réseaux, *Pearson Education*, 5ième édition, 2011.
2. Guy Pujolle. Les réseaux. *Eyrolles*, 2010
3. Danièle Dromard, Dominique Seret, Architecture des Réseaux, *Pearson Education*, France, 2009.
4. Bertrand Petit, Architecture des Réseaux : Cours et Exercices Corrigés, *Ellipses Édition*, 2002.
5. Claude Servin, Réseaux et Télécoms : Cours avec 129 exercices corrigés, 2ième édition, *Dunod*, 2006.
6. Paolo Zanella, Yves Ligier, Architecture et Technologie des Ordinateurs, *Dunod*, 1989.
7. Dominique Seret, Ahmed Mehaoua, Neilze Dorta, Réseaux et Télécommunications : Support de cours, *Université René Descartes - Paris 5*, 2006.
8. Alonso Stéphane, Cours Réseaux, *TS IRIS-LEGT Louis Modest-Leroy-Évreux*, 2004
9. Péan Bruno, Support de cours Réseaux, *EISTI, EISTI-Cergy*, 2001.
10. Laurent Signac, Réseaux : TCP/IP, Internet, Sécurité, https://deptinfo-ensip.univ-poitiers.fr, 2013.
11. N. Guellati, support de Cours Réseaux de communication, *Université de Setif*, 2010.