# Internet et les applications sur TCP/IP

1- Definitions, historique et normalisation
2- le DNS
3- la Messagerie
4- Telnet
5- FTP
6- le World Wide Web



## *1ère partie*: Définitions, historique et normalisation

- **l'Internet** (avec un I majuscule) désigne l'interconnexion à l'échelle mondiale de réseaux et d'ordinateurs utilisant le protocole IP
- Historique de l 'Internet:
  - Commencé dans les années **1970** aux Etats-Unis sous l'impulsion du DoD [Department of Defense] sous le nom **d'ARPANET**
  - Dans les années **1980**, le réseau se développe dans le monde académique américain: naissance de **NSFNET** [NSF=National Science Foundation]
  - A la **fin des années 1980**, les sociétes commerciales se connectent de plus en plus à l'Internet et de nombreux pays rejoignent ce réseau
  - En France, le réseau Renater (Réseau National de l'Enseignement et de la Recherche) est créé en **juin 1992**
  - Fin de NSFNet en **1995**: création des NAP [Network Access Points]

## Les organismes de l ’Internet (1/2)

- En 1983, création de l **’IAB** [Internet Activities Board]:
  - coordonner les activités des chercheurs de l ’ARPANET et de l ’Internet vis-à-vis du DoD et du NSF
  - travaille sous forme de rapports techniques, les **RFC** [Request for Comments]
- En 1989, l ’IAB est restructuré:
  - Changement de nom: Internet Architecture Board
  - Création de 2 comités:
    - **IETF** [Internet Engineering Task Force], traitant des problèmes à court terme; plus de 70 groupes de travail créés.
    - ⇒ L'IESG (Internet Engineering Steering Group) est l'organe de direction de l'IETF et qui est responsable de l'approbation finale des standards
    - **IRTF** [Internet Research Task Force], travaillant sur l ’évolution à long terme



## Les organismes de l ’Internet (2/2)

- En 1992, création de l **’Internet Society** (ISOC) : promotion de l’Internet. Association comparable à l ’ACM ou à l ’IEEE
- En 1995, création du W3C [World-Wide Web Consortium] :
  - hébergé par:
    - le MIT aux Etats-Unis,
    - l'INRIA en France,
    - l'université de Keio au Japon
  - Son but : coordonner les développements liés au Web
- En 1998, création de l'ICANN (Internet Corporation for Assigned Names and Numbers) qui est chargé:
  - de l'allocation des adresses IP
  - de la gestion des noms de domaine
  - de l'homologation des protocoles et de leurs paramètres
  - de la gestion des serveurs racines

## Normalisation dans le cadre de l'Internet (1/2)

- Normalisation:
  - Un document est écrit par un groupe souhaitant proposer une idée sous la forme d 'un RFC
  - Si cela présente suffisamment d 'intérêt alors il lui est donné l 'état de ***proposed standard*** (PS)
  - Pour devenir DS (***draft standard***) il faut qu'une implémentation ait été testée sur 2 sites différents pendant au moins 4 mois
  - si l 'IAB est convaincu que l 'idée est bonne et si le logiciel fonctionne bien, alors le RFC devient un ***Internet standard*** (IS)
- Les normes ont ensuite des statuts:
  - Exigé [required]: toute machine utilisant IP doit implémenter ce RFC
  - Recommandé [recommanded]
  - Facultatif [elective]
  - Usage limité [limited use]: protocole expérimental (usage déconseillé)
  - Non recommandé [not recommended]: protocole périmé

R. CHALON Internet et les applications sur TCP/IP 5

## Normalisation dans le cadre de l'Internet (2/2)

- Les RFC contiennent donc:
  - les normes réseau (exemple: IP: RFC 791)
  - les normes des applications (exemple: FTP: RFC 959)
  - des informations générales
  - ➡ numérotées par ordre croissant. Lorsqu'il y a une nouvelle version d 'un standard elle porte un nouveau numéro mais elle garde le même titre (différent de l 'OSI)
- Il existe aussi les **FYI** [For Your Information] qui sont une sélection de RFC d 'intérêt général
- Les RFC sont disponibles gratuitement sur l 'Internet (contrairement à l 'OSI où les normes sont payantes et chères!):
  - ➡ http://www.pasteur.fr/other/computer/RFC/
- Aujourd'hui plus de 2700 RFC ! (RFC 2795, 1er avril 2000)
- le document STD1 contient la liste des standards et est régulièrement ré-édité (actuellement RFC 2600, mars 2000)

R. CHALON Internet et les applications sur TCP/IP 6

## Autres définitions

- Intranet : réseau interne d'une entreprise (en général réseau étendu sur plusieurs sites, voire plusieurs continents) et utilisant les protocoles de l'Internet
- Extranet : connexion à un réseau intranet par l'intermédiaire de l'Internet d'un groupe fermé de personnes externes à l'entreprise (typiquement les clients et les fournisseurs de cette entreprise)
- Ces définitions impliquent une sécurisation des accès entre l'Internet et l'intranet de l'entreprise:
  - utilisation de pare-feux [fire-walls]: protection des attaques externes
  - utilisation de la cryptographie: protection des informations échangées sur l'Internet

R. CHALON Internet et les applications sur TCP/IP 7

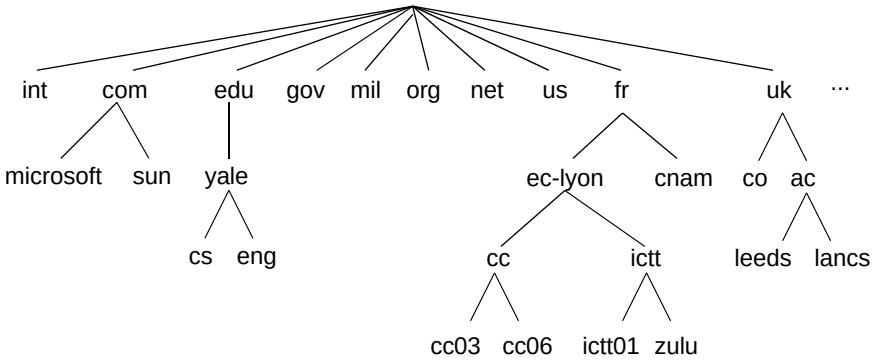
## *2ème partie*: le DNS

- Dans un réseau TCP/IP, toute machine n'est connu que par son adresse IP de la forme: **156.18.19.5**
- Pour simplifier l'usage et la mémorisation des serveurs, il serait plus simple d'utiliser un nom qu'un numéro
  - il est plus simple de retenir: **www.ec-lyon.fr** que **156.18.19.5**
- 1ère solution: stocker les correspondances nom/adresses dans un fichier sur chaque machine:
  - c'est ce qui était fait au début: fichier **/etc/hosts** sur UNIX
  - Mais solution devenue irréaliste devant l'explosion de l'Internet
- 2ème solution: base de données répartie, le DNS [Domain Name System] - Système de noms de domaine
- Normalisé par les RFC 1032, 1033, 1034 et 1035
- Autre avantage: pouvoir changer l 'adresse physique d 'une machine sans changer son nom

R. CHALON Internet et les applications sur TCP/IP 8

## Espace de noms du DNS (1/2)

- Utilisation d'un espace de noms hiérachique:
  - éviter les homonymes ==> unicité des noms à l 'échelle mondiale
  - simplifier la gestion: délégation de zones
- arbre sans nom à la racine:




R. CHALON Internet et les applications sur TCP/IP 9

## Espace de noms du DNS (2/2)

- Au premier niveau:
  - 7 domaines génériques:
    - **edu** [education], **gov** [governmental], **mil** [military] réservés aux USA
    - **com** [commercial], **net** [networks], **org** [organisation], **int** [international] utilisables dans le monde entier
  - 1 domaine par pays: utilisation du code pays à 2 lettres (ISO 3166)
    - ➡ exemples: **fr** (France), **uk** (Royaume-Uni), **jp** (Japon), etc...
- Au deuxième niveau, c'est très variable:
  - **uk** et **jp** subdivisent en **co** [commercial] et **ac** [academic],
  - **fr** ne subdivisait pas, mais aujourd'hui il exite **asso** (associations), **tm** (marques déposés), **gouv** (ministères), etc...
- Un nom complet est donné par le chemin pour remonter jusqu'à la racine en séparant chaque sous-domaine par un point:
  - ➡ exemple: **cc03.cc.ec-lyon.fr**

R. CHALON Internet et les applications sur TCP/IP 10

## Gestion des domaines (1/3)

- Un ensemble d 'organisme gère les domaines de plus haut niveau:
  - l 'InterNIC [Network Information Center] aux Etats-Unis les 7 domaines génériques et le domaine us
  - Dans chaque pays au moins un NIC local gère le domaine du pays; exemple en France: AFNIC [Association Française pour le Nommage Internet en Coopération]
  - ➡ Cette gestion est actuellement fortement remise en cause! Débat en cours entre les Etat-Unis et l 'Union Européenne (mi 1998)
- Pour créer un nouveau domaine il faut avoir l 'autorisation du domaine de niveau supérieur qui va réaliser l 'enregistrement
  - pour créer ec-lyon.fr il faut l 'autorisation de l 'AFNIC
  - pour créer ictt.ec-lyon.fr il faut l 'autorisation de l 'Ecole Centrale

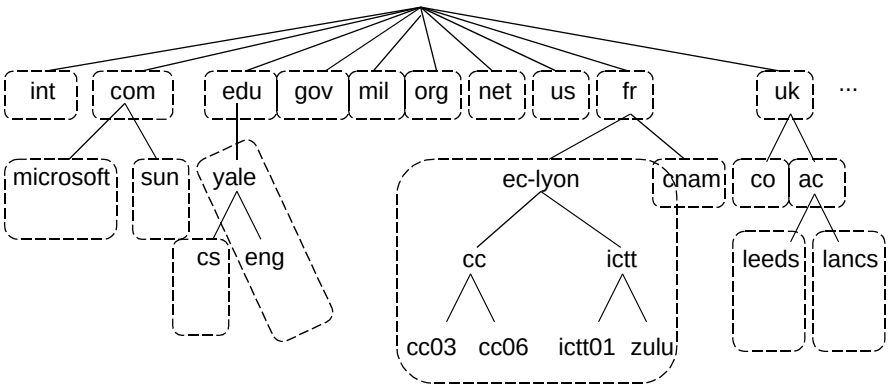


## Gestion des domaines (2/3)

- Le nommage est basé sur des organisations pas sur l 'architecture physique des réseaux:
  - ardemi.fr partage la même classe B que ec-lyon.fr (156.18.0.0)
  - urec.cnrs.fr a deux réseaux: un a l 'Université de Jussieu (Paris) et l 'autre à l 'Université de Grenoble 1
- La gestion de l 'espace de nom est répartie entre plusieurs serveurs gérant chacun un morceau de l 'arborescence:
  - chaque partie de l 'espace est appelée zone
  - les frontières d 'une zone ne dépendent que de son administrateur et des délégations de gestion qu'il a accordé

## Gestion des domaines (3/3)

- Exemple de découpage en zones:






## Serveurs de noms

- Chaque zone possède plusieurs serveurs:
  - un serveur primaire qui contient la base de donnée originale
  - un ou plusieurs serveurs secondaires qui contiennent des copies de la base et qui peuvent répondre en cas de défaillance du serveur primaire (ou du réseau d 'accès au serveur)
- Chaque serveur peut servir une ou plusieurs zones, comme serveur primaire ou secondaire
- Chaque serveur de zone primaire contient une base de donnée de description de la zone gérée:
  - description de la zone
  - liste d'enregistrements de ressources
  - pointeur vers d 'autres serveurs de noms [glue records]

## Enregistrement de ressource

- Faire correspondre à chaque nom de domaine un enregistrement de ressource:
  - l 'adresse IP d 'une machine (ou hôte)
  - un relais de messagerie
  - un serveur de nom de domaines
  - un alias d 'une machine, etc…
- Structure d 'un enregistrement:

  **`Nom_domaine durée_de_vie classe type valeur`**

  - **`Nom_domaine`**: nom du domaine (complet ou relatif)
  - **`durée_de_vie`**: stabilité de cet enregistrement; utilisé par le système de cache lors de l 'interrogation du DNS (facultatif)
  - **`classe`**: aujourd'hui seule la valeur **IN** (pour Internet) est définie
  - **`type`**: type de l 'enregistrement
  - **`valeur`**: valeur de l 'enregistrement; dépend du type



## Types d 'enregistrement (1/3)

| **Type** | **Signification** | **Valeur** |
|---|---|---|
| SOA | Début de zone [Start of Authority] | Paramètres de la zone (ensemble de valeur pour la gestion de zone) |
| A | Adresse IP d 'un hôte [Address] | Adresse sur 32 bits au format: xxx.xxx.xxx.xxx |
| MX | Relais de messagerie [Mail exchanger] | Priorité + nom du relais |
| NS | Nom de serveur [Name server] | Nom du serveur du domaine |
| CNAME | alias [canonical name] | Nom de l 'hôte |
| PTR | Pointeur [pointer] | Nom de domaine |

Il existe aussi HINFO, TXT, MINFO

## Types d ’enregistrement (2/3)

- Enregistrement SOA:
  - décrit la zone gérée par un serveur de nom
  - contient l ’adresse de messagerie électronique du responsable
  - un numéro de série unique et divers paramètres de temporisation
- Enregistrement A:
  - permet de faire correspondre un nom de machine à une adresse IP
  - exemple:
    **`cc03.cc.ec-lyon.fr.  IN  A  156.18.22.3`**
  - un même hôte peut avoir plusieurs adresses IP (cas d ’une machine ayant plusieurs cartes réseaux sur différents réseaux)
- Enregistrement MX:
  - permet de spécifier la machine qui va servir de relais de messagerie pour un domaine
  - exemple:
    **`ec-lyon.fr.  IN  MX  10  cc03.cc.ec-lyon.fr.`**



## Types d ’enregistrement (3/3)

- Enregistrement CNAME:
  - permet de créer des alias sur des hôtes
  - exemple:
    **`www.ec-lyon.fr.  IN  CNAME  cc06.cc.ec-lyon.fr.`**
  - permet de donner des noms plus « explicites » à des hôtes
  - permet de changer plus facilement la machine; exemple: migration sur trotek05.trotek.ec-lyon.fr transparente pour les utilisateurs du serveur Web
- Enregistrement NS:
  - permet à de spécifier les serveurs de noms qui gèrent un domaine afin de faire les liens entre les serveurs DNS
- Enregistrement PTR:
  - enregistrement spécial pour faire la correspondance inverse: adresse IP --> nom de domaine

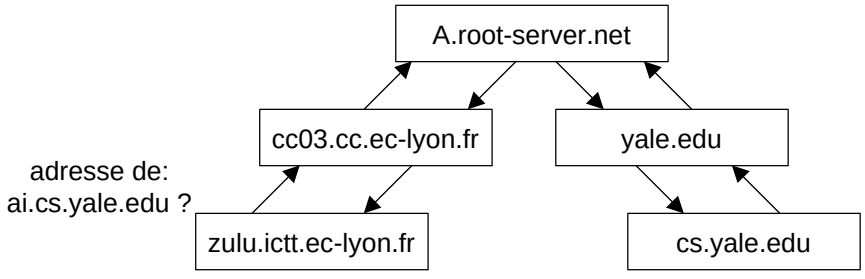
## Exemple d ’enregistrements

```
ec-lyon.fr.        IN  SOA     cc03.cc.ec-lyon.fr.
                         dnsmaster.cc03.cc.ec-lyon.fr.
                         (1997122701 21600 10800 3900000 86400)
                   IN  NS      cc03.cc.ec-lyon.fr.
                   IN  NS      trotek05.trotek.ec-lyon.fr.
                   IN  NS      calypso.urec.cnrs.fr.
                   IN  MX      10  cc03.cc.ec-lyon.fr.
   …
cc03.cc            IN  A       156.18.22.3
                   IN  MX      10 cc03.cc
   …
trotek05.trotek    IN  A       156.18.19.5
                   IN  MX      10  trotek05.trotek
                   IN  MX      20  cc03.cc
www                IN  CNAME   trotek05.trotek
   …
zulu               IN  A       156.18.28.1
                   IN  MX      10 trotek05.trotek
```



## Interrogation du DNS (1/2)

- Méthode récursive :
  - il s ’adresse au serveur de noms le plus proche:
  - si celui-ci a la réponse, il retourne directement la valeur
  - sinon il transmet la requête à un serveur de nom racine qui fait redescendre la demande et ainsi de suite
  - la réponse remonte la chaîne jusqu’au client initial




  - Remarque : cette méthode n'est pas toujours disponible...

## Interrogation du DNS (2/2)

- Méthode itérative :
  - le client s 'adresse au serveur DNS local
  - s 'il n 'a pas la réponse il lui retourne le nom et l 'adresse d'un autre serveur à contacter qui pourrait avoir la réponse
  - le client interroge ce serveur et ainsi de suite jusqu'à ce qu'il ait la réponse
- Problème de performance:
  dans tous les cas, il n 'est pas performant d 'aller interroger à chaque fois un (ou plusieurs) serveur(s) de nom à l 'autre bout du monde
- Solution:
  garder dans une mémoire cache les requêtes précédentes ce qui permet de répondre directement sans questionner de nouveau les serveurs de plus haut niveau



## Utilisation du Cache

- L 'information conservée dans les caches peut devenir obsolète suite à un changement dans l 'architecture du réseau:
  - disparition d 'une machine
  - changement d 'adresse d 'une machine, etc…
- l 'information n 'est donc conservée dans un cache que pendant la « durée_de_vie » de l 'enregistrement:
  - donné par l 'enregistrement s 'il est spécifié
  - par la valeur par défaut donnée dans l 'enregistrement SOA de la zone
- Malgré tout, comme l 'information peut être caduque:
  - le client est informé que la réponse vient d 'un cache (non-authoritative answer)
  - le serveur donne le nom et l 'adresse du serveur de nom capable de donner une réponse de « première main » et que le client peut interroger directement s 'il souhaite privilégier la fiabilité à la performance

## Protocole d ’accès au DNS

- Protocole basé sur UDP:
  - les questions et les réponses ont tous le même format:
    - faciliter pour retransmettre le message de serveur en serveur
    - la réponse est insérée dans le message qui est retourné avec la question

| 0 | 16 | 31 |
|---|---|---|
| Identification | paramètre | |
| Nombre de questions | Nombre de réponses | |
| Nombre d ’« authority » | Nombre de compléments | |
| Section des questions ... | | |
| Section des réponses ... | | |
| Section des « authority » ... | | |
| Section des informations additionnelles ... | | |

R. CHALON Internet et les applications sur TCP/IP 23

## Correspondances inverses

- Pour obtenir à partir de l ’adresse IP d ’un hôte son nom il a été imaginé de pouvoir faire des « requêtes inverses »
- Cependant, vu la complexité du problème il a été créé un pseudo-domaine IN-ADDR.ARPA avec sa propre arborescence et qui contient les correspondances:
  adresses IP --> noms d ’hôtes
- Les adresses sont spécifiées « à l ’envers »
- Utilisation des enregistrements PTR
- Exemple:

```
3.22.18.156.in-addr.arpa.   IN  PTR  cc03.cc.ec-lyon.fr.
1.28.18.156.in-addr.arpa.   IN  PTR  zulu.ictt.ec-lyon.fr.
```

- Problème:
  la cohérence entre les bases directes et inverses doit être assurée à la main!

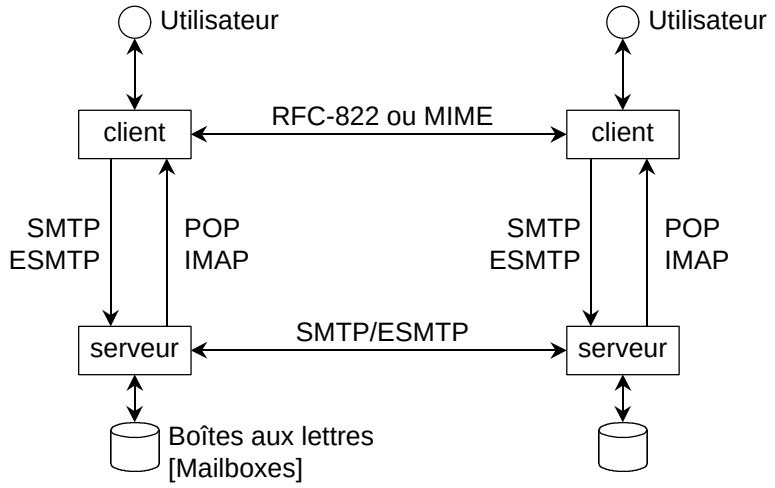
R. CHALON Internet et les applications sur TCP/IP 24

## 3ème partie: Messagerie électronique

- Messagerie électronique [Electronic Mail ou E-mail]:
  - communication interpersonnelle en temps différé
  - message écrit: persistance des messages
  - mode « store and forward »: stockage temporaire des messages avant retransmission au nœud suivant
  - avis en cas de non-remise
  - transfert de messages textuels sans accents (ASCII)
  - extensions permettant de transmettre tout type de données
- Historique:
  - 1982: RFC 821 (SMTP) et RFC 822
  - 1992: RFC 1341 (MIME)
  - 1993: RFC 1521/1522 (MIME 2ème version)
  - 1994: RFC 1651 (ESMTP)
  - 1995: RFC 1869/1870, 1891 (ESMTP 2ème version)
  - 1996: RFC 2045 à 2049 (MIME 3ème version)



## Architecture fonctionnelle



## Définitions

- SMTP [Simple Mail Transfer Protocol] et ESMTP [Enhanced SMTP]:
  - Protocoles de transfert de messages
- POP [Post Office Protocol]:
  - protocole permettant la relève de la boîte aux lettres à distance
- IMAP [Interactive Mail Access Protocol]:
  - protocole de gestion de la boîte aux lettres d'un serveur à distance
- RFC-822 et MIME [Multipurpose Internet Mail Extensions]:
  - protocoles de codage des messages électroniques
- Remarques:
  - les termes MTA (pour serveur) et UA (pour client) sont parfois utilisés en référence à l 'OSI
  - la traduction officielle en français de E-mail est « mél »; certains utilisateurs utilisent également « courriel »



## Adresses de messagerie (1/2)

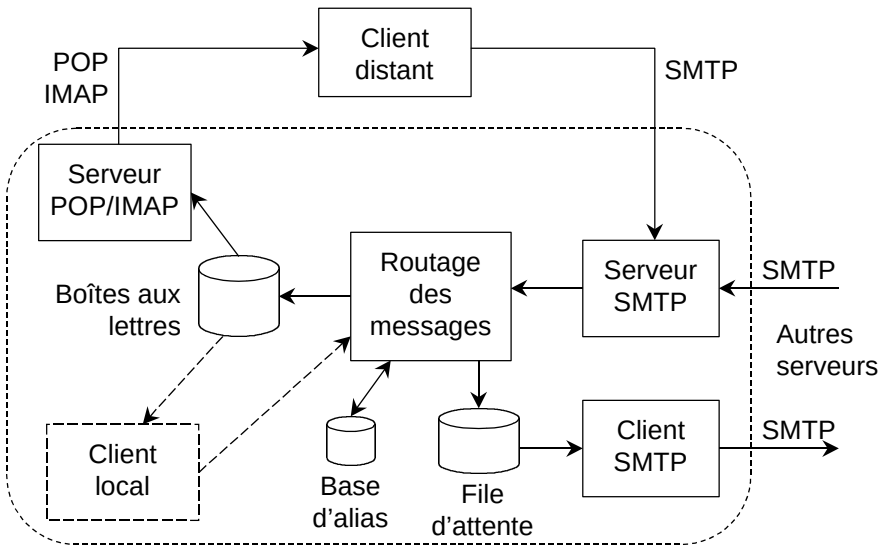
- Format des adresses:
  - **`boite@domaine`**
- Partie droite (domaine):
  - nom de domaine de messagerie
  - Doit correspondre à un (ou plusieurs) serveur(s) qui va(vont) recevoir les messages
  - la correspondance est faite par les enregistrements MX du DNS:
    - s 'il y a plusieurs serveurs, ils sont utilisés par ordre de priorité croissant
    - permet de donner des serveurs de secours en cas de panne
- Partie gauche (boite):
  - identification de la boîte aux lettres sur le serveur
  - peut correspondre à un utilisateur, à un groupe (liste de diffusion) ou à un automate

## Adresses de messagerie (2/2)

- Exemples:
  - **Rene.Chalon@ictt.ec-lyon.fr**
  - **rchalon@mrash.fr**
- Pour faciliter la mémorisation des adresses, Il est souvent utilisé des alias:
  - exemple: Rene.chalon@mrash.fr est un alias de rchalon@mrash.fr
- Pour faciliter l 'administration, pour tout domaine de messagerie existant, l 'adresse **postmaster@domaine** doit exister et correspondre à un utilisateur humain auquel on peut s 'adresser en cas de problèmes

R. CHALON Internet et les applications sur TCP/IP 29

## Structure d 'un serveur




R. CHALON Internet et les applications sur TCP/IP 30

## Protocole SMTP

- Protocole SMTP:
  - modèle client/serveur
  - construit sur TCP (le serveur écoute sur le port 25)
  - le dialogue se fait en clair (commandes en ASCII)
- Connexion directe du serveur initial (éventuellement du client!) vers le serveur du destinataire:
  - pas de relais intermédiaire (sauf pour raisons de sécurité)
- Transfert de messages ASCII non accentués seulement
- Commandes:
  - HELO: identification du client sur le serveur
  - MAIL FROM: expéditeur
  - RCPT TO: destinataire (répété si plusieurs destinataires)
  - DATA: débute l 'envoi du message (la fin est signalé par un point)
  - QUIT: fin du dialogue



## Exemple de connexion SMTP

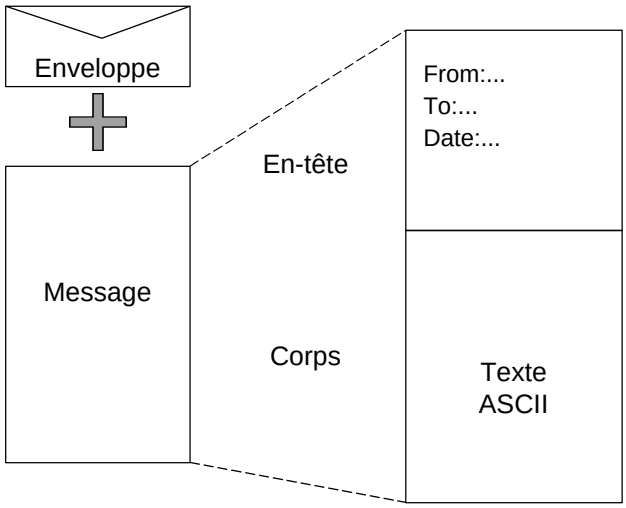
```
Client
       Serveur
       220 SMTP ready
HELO zulu.ictt.ec-lyon.fr
       250 cc03.cc.ec-lyon.fr
MAIL FROM: Rene.chalon@ictt.ec-lyon.fr
       250 ok
RCPT TO: postmaster@ec-lyon.fr
       250 ok
DATA
       354 start mail input; end with .
message …
message …
.
       250 ok
QUIT
       221 cc03.cc.ec-lyon.fr closing
```

## Protocole ESMTP

- Extension du protocole SMTP
- supporte les transferts de messages codés sur 8 bits
- Nombreuses améliorations:
  - permet de préciser la taille des messages avant l 'envoi, ce qui permet au serveur destinataire de refuser des messages trop grands
  - meilleure gestion de la sécurité
- le dialogue est initié par EHLO:
  - si le serveur répond 250, alors il supporte ESMTP
  - s 'il répond 500 (commande inconnue), alors revenir à SMTP en envoyant HELO



## Format des messages (RFC-822)



## Champs d ’en-tête RFC-822

- From: expéditeur du message
- Sender: adresse de celui qui a envoyé le message
- To: destinataires
- Cc: destinataires en copie
- Bcc: destinataires en copie muette
- Received: trace ajoutée par chaque serveur SMTP qui a relayé le message
- Return-path: adresse permettant de répondre à l émetteur (ajouté par le serveur final)
- Reply-to: adresse de la personne à qui il faut répondre (ajouté par l ’expéditeur du message)
- Message-id: identificateur unique pour le message
- Date: date et heure d ’émission du message
- Subject: objet du message
- In Reply-to: id du message auquel on répond
- References: autres id de messages liés à celui-ci

R. CHALON Internet et les applications sur TCP/IP 35

## Corps du message RFC-822

- Le corps est séparé de l ’en-tête par une ligne vide
- le corps ne peut contenir que du texte ASCII non-accentué sous forme de ligne de 80 caractères
- Pour envoyer des contenus d ’autre nature, il faut coder les données:
  - par exemple: uuencode, BinHex
  - ces codages transforment les données binaires en « texte » de manière à être transportées de manière transparente
  - inconvénient: le destinataire doit disposer du décodeur pour récupérer le contenu initial.
  - Le destinataire n ’a aucune information sur le contenu du fichier
- Meilleure solution: MIME

R. CHALON Internet et les applications sur TCP/IP 36

## Exemple de message

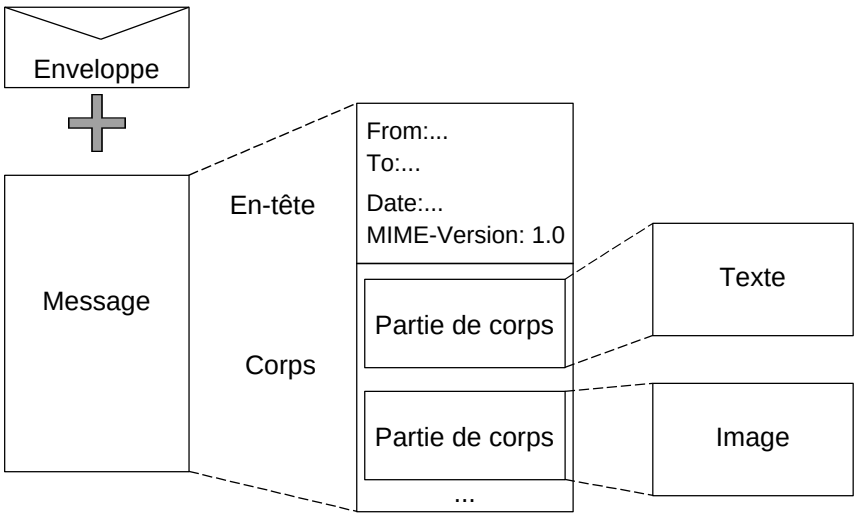
```
Received: from zulu.ictt.ec-lyon.fr (zulu.ictt.ec-lyon.fr
  [156.18.28.1]) by cc03.cc.ec-lyon.fr with ESMTP
  (8.8.8/LM.1.0) id OAA14693 for <postmaster@ec-lyon.fr>;
  Mon, 27 Apr 1998 14:22:07 +0200 (METDST)
Date: Mon, 27 Apr 1998 14:22:06 +0200 (METDST)
Reply-To: <Rene.Chalon@ictt.ec-lyon.fr>
From: Rene.Chalon@ictt.ec-lyon.fr
To: postmaster@ec-lyon.fr
Message-Id: <24782407487488@ictt.ec-lyon.fr>
Subject: Essai

Essai de courrier electronique

Bien cordialement,
Rene.
```



## MIME



## Extensions MIME

- Extensions à la RFC-822: RFC 2045 à 2047 + 2048 et 2049
- Ajout de nouveaux champs d 'en-tête:
  - MIME-Version: identification de la version (aujourd 'hui 1.0)
  - Content-Description: description en clair du contenu
  - Content-Id: identificateur unique (format idem Message-id)
  - Content-Transfer-Encoding: façon dont le contenu est codé pour être transporté
  - Content-type: spécifie le contenu du message:
    - format: `<type>/<sous-type>; <paramètres éventuels>`
- Nouvelle structuration du corps des messages:
  - le corps du message peut être d 'un seul type
  - le corps du message contient plusieurs parties --> messages multiparties



## Types de contenus

| **Type** | **Sous-type** | **Description** |
|---|---|---|
| text | plain | Texte non formaté |
| | richtext | Texte avec des commandes de formatage très simples |
| image | gif | Image au format gif |
| | jpeg | Image au format JPEG |
| audio | basic | Son non compressé |
| video | mpeg | Vidéo au format MPEG |
| application | octet-stream | Suite d 'octets non-interprétés |
| | postcript | Fichier imprimable au format PostScript |
| message | rfc822 | Message MIME RFC 822 |
| | partial | Partie de message (découpé pour la transmission |
| | external-body | Le corps contient une référence externe |

## Messages multiparties

| Type | Sous-type | Description |
|---|---|---|
| multipart | mixed | Parties indépendantes |
| | alternative | Même message en différents formats |
| | parallel | Parties à voir simultanément |
| | digest | Chaque partie est un message RFC 822 |

- Un paramètre supplémentaire précise le texte qui sera utilisé pour séparer chaque partie:
  - ➡ Content-type: multipart/mixed; boudary=azertyuiop
- Chaque partie contient:
  - un en-tête avec les champs:
    - Content-Type (si non présent, alors text/plain; charset=us-ascii)
    - Content-Transfer-Encoding
  - une ligne vide séparatrice
  - le contenu de la partie

R. CHALON Internet et les applications sur TCP/IP 41

## Codages pour le transfert

- Pour être compatible avec le système de transport SMTP (caractères ASCII seulement), il faut encoder les messages
- Pour les textes simples contenant des caractères accentués:
  - codage **Quoted-printable** (QP)
  - codage du code du caractère en hexadécimal en préfixant par =
  - exemple: é se code **=E9** (si code ISO 8859-1 utilisé)
- Pour tout type de données:
  - codage **Base64** appelé aussi **blindage ASCII**
  - on groupe 3 octets = 24 bits qu'on divise en 4 groupes de 6 bits
  - à chaque combinaison de 6 bits (64 possibles) on associe un caractère ASCII:
    - 0=« A », 1=« B », …, 25=« Z »,
    - 26=« a », …, 51=« z », 52=« 0 », …, 61=« 9 »,
    - 62=« + », 63=« / » et =, == servent au bourrage
  - des CR et LF sont ajoutés pour faire des lignes de 76 caractères

R. CHALON Internet et les applications sur TCP/IP 42

## Exemple de message MIME

```
From: Rene.Chalon@ictt.ec-lyon.fr
To: postmaster@ec-lyon.fr
MIME-Version:1.0
Message-Id: <24782407487488@ictt.ec-lyon.fr>
Content-Type: multipart/mixed; boundary=azertyuiop
Subject: Essai

--azertyuiop
Content-Type: text/plain; charset="iso-8859-1"
Content-Transfer-Encoding: quoted-printable

Essai de courrier =E9lectronique
--azertyuiop
Content-Type: application/octet-stream
Content-Transfer-Encoding: base64

Ejfh/HuGp4+qz6A
--azertyuiop--
```



## Remise finale du courrier à l'utilisateur

- SMTP (ou ESMTP) est prévu pour délivrer directement le message sur la machine de l 'utilisateur:
  - il faut que la machine soit toujours en écoute et connectée au réseau, ce qui n 'est pas le cas des PC et surtout des ordinateurs portables
- Le courrier est donc délivré sur un serveur central de l 'organisation et chaque client vient relever sa boîte aux lettres périodiquement pour voir si du courrier est arrivé:
  - soit directement sur le serveur de messagerie (protocole propriétaire)
  - soit en utilisant un protocole de relève de boîte
- Trois protocoles sont définis:
  - POP [Post Office Protocol]: RFC 1225, version 3
  - IMAP [Interactive Mail Access Protocol]: RFC 1064
  - DMSP [Distributed Mail System Protocol]: RFC 1056

## Protocole POP3

- Permet de transférer les messages depuis le serveur de messagerie sur le poste de l 'utilisateur
- Protocole POP3:
  - modèle client/serveur
  - construit sur TCP (le serveur écoute sur le port 110)
  - le dialogue se fait en clair (commandes en ASCII)
- Commandes principales:
  - USER: identification du client sur le serveur
  - PASS: mot de passe pour authentification
  - LIST: liste des messages en attente
  - TOP: lire le début d 'un message
  - RETR: récupération d 'un message
  - DELE: effacement d 'un message
  - QUIT: fin du dialogue

R. CHALON Internet et les applications sur TCP/IP 45

## Exemple de connexion POP3

```
Client                                  Serveur
                                        +OK cc03.cc.ec-lyon.fr POP3 server ready
USER chalon
                                        +OK I know you, chalon
PASS *****
                                        +OK Welcome! 1 messages (18184 bytes)
LIST
                                        +OK 1 messages (18184 bytes)
                                        1 18184
                                        .
RETR 1
                                        message …
                                        message …
                                        .
DELE 1
                                        +OK message deleted
QUIT
                                        +OK cc03.cc.ec-lyon.fr server closing down
```

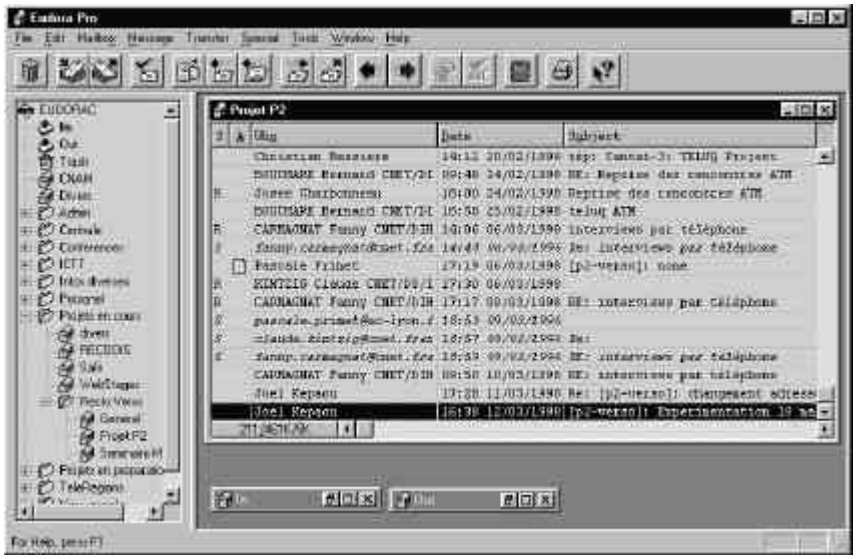
R. CHALON Internet et les applications sur TCP/IP 46

## Autres protocoles de remise

- IMAP:
  - protocole plus sophistiqué
  - le courrier reste sur le serveur: gestion d ’une base de donnée de courrier
  - permet de consulter ses messages depuis différents ordinateurs: au bureau, à la maison, en voyage, etc…
  - nombreuses commandes de recherche de courrier par ses attributs
  - permet de gérer des filtres de classement et d ’effacement automatique du courrier
- DMSP:
  - le courrier peut se trouver sur plusieurs serveurs et clients
  - permet de travailler sur plusieurs machines et de les synchroniser périodiquement
  - à la différence d ’IMAP, il ne nécessite pas d ’être connecté au serveur pour travailler

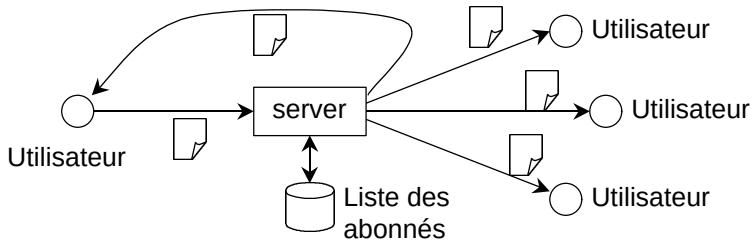


## Exemple d ’interface utilisateur: Eudora



## Listes de messagerie

- Utilisation de la messagerie, qui est a priori un moyen de communication interpersonnel, pour diffuser des messages au sein d 'un groupe
- Une adresse spécifique est déclarée sur un serveur qui correspond à la liste (exemple: tous@ec-lyon.fr)
- Tout message envoyé à cette adresse est renvoyé automatiquement par le serveur à tous les abonnés de la liste




R. CHALON Internet et les applications sur TCP/IP 49

## Caractéristiques des listes de messagerie (1/2)

- Les listes peuvent être:
  - publiques: quiconque peut s 'abonner
  - non-publiques: seuls les gestionnaires de la liste peuvent ajouter des abonnés
- Listes restreintes (ou fermées):
  - seuls les abonnés (et les modérateurs) peuvent envoyer des messages
- Listes modérées:
  - le message est lu par un (ou plusieurs) modérateurs qui juge de l 'opportunité de diffuser le message à la liste toute entière
- Mode « digest »:
  - Au lieu d 'envoyer immédiatement à tous les abonnés les messages qui arrivent à la liste, certains abonnés peuvent choisir de recevoir tous les messages des dernières 24h (ou autre période) dans un seul message journalier

R. CHALON Internet et les applications sur TCP/IP 50

## Caractéristiques des listes de messagerie (2/2)

- Listes anonymes:
  - toute référence liée à l 'expéditeur est supprimée du message
- Archivage des messages:
  - Tous les messages envoyés à la liste peuvent être archivées dans un fichier ou une base de donnée sur le serveur
  - certains systèmes proposent de pouvoir consulter cette archive sur un serveur FTP ou Web
- Autres options:
  - la réponse (champ Reply-To) peut être positionné pour répondre à la liste ou à l 'expéditeur du message
  - on peut autoriser ou interdire la consultation de la liste des membres abonnés



## Serveurs de listes de messagerie (1/2)

- De nombreux programmes (souvent gratuits) existent pour mettre en place des listes: Listserv, majordomo, TULP
- Le serveur dispose d 'une adresse spéciale permettant de lui envoyer des commandes; cette adresse est la même pour toutes les listes gérées par ce serveur !
  - Il suffit d 'envoyer un message à cette adresse avec comme contenu la liste des commandes à exécuter
  - Un message est automatiquement retourné quelques instants après contenant le résultat de l 'exécution des commandes
  - exemple: listserv@ec-lyon.fr
- Exemple des commandes de TULP:
  - **HELp**: obtenir un fichier d 'aide
  - **LISt**: obtenir la liste de toutes les listes du serveur
  - **REView <liste>**: obtenir la liste des abonnés à une liste (si l 'utilisateur est autorisé)

## Serveurs de listes de messagerie (2/2)

- **SUBscribe <liste> <prénom> <nom>**: s 'inscrire à la liste (c 'est l 'adresse de l 'expéditeur du message qui est utilisée); l 'inscription n 'est possible que pour les listes publiques
- **SIGnoff <liste>**: se désabonner à la liste
- **ADD <liste> <user@host> <prénom> <nom>**: ajoute un utilisateur a une liste (utilisé par les modérateurs pour inscrire des personnes sur une liste non-publique ou fermée)
- **DEL <liste> <user@host>**: retirer un utilisateur d 'une liste, etc...

- Exemple à l 'Ecole Centrale:
  - **listmaster@ec-lyon.fr**: contact technique (équiv. du postmaster)
  - **listserv@ec-lyon.fr**: adresse du serveur de messagerie
  - **tous@ec-lyon.fr**: liste fermée du personnel (~ 450 personnes)
  - **promo1999@ec-lyon.fr**: liste fermée des élèves de la promo 1999
  - etc… (en tout 12 listes fermées)

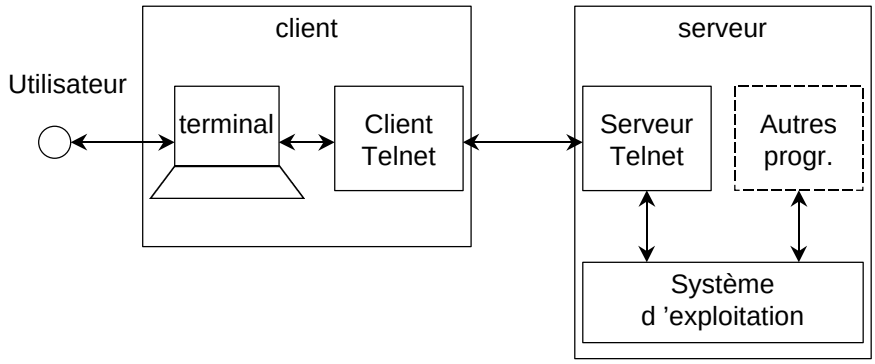


## *4ème partie*: Telnet

- Telnet fournit un service de terminal textuel simple à distance
- Il offre 3 services de bases:
  - il définit un service de terminal virtuel: NVT [Network Virtual Terminal] qui offre une interface standard d 'accès au système distant
  - il fournit un mécanisme de négociation d 'options entre le client et le serveur (par exemple, caractères sur 7 bits ou 8 bits)
  - les deux extrémités sont considérés de manière totalement symétrique
- Telnet est défini par la RFC 854 complété, pour les différentes options, par les RFC 856 à 861, 884, 885, 1091, 1096, 1097, 1041 et 1116 !!!

## Architecture fonctionnelle de Telnet

- Telnet fonctionne sur TCP
- Le serveur attend les connexions sur le port 23




R. CHALON Internet et les applications sur TCP/IP 55

## Terminal virtuel - NVT (1/2)

- NVT permet de résoudre les problèmes d'hétérogénéité entre les systèmes:
  - par exemple: fin de ligne est <CR>, <LF> ou <CR><LF> selon les systèmes d 'exploitation
- NVT convertit:
  - du côté client, le format du terminal local en format NVT
  - du côté serveur, le format NVT en format propre au serveur
- Toutes les transmissions utilisent 8 bits
- Au démarrage, NVT limite le jeu de caractère au code ASCII et réserve les codes au-delà de 128 à des codes de contrôle
- Les fins de ligne sont délimités par <CR><LF>
- Les codes de contrôle sont précédés par un caractère d 'échappement, « IAC » code ASCII 255
- Pour envoyer le caractère 255 il faut transmettre 255-255

R. CHALON Internet et les applications sur TCP/IP 56

## Terminal virtuel - NVT (2/2)

- Il existe de nombreux signaux et commandes:
  - BRK [break] (code 243): signal pour attirer l 'attention du processus
  - IP [interrupt process] (code 244): interrompre la commande en cours
  - AO [abort output] (code 245): efface toute donnée dans les tampons d 'entrée/sortie
  - AYT [Are you there?] (code 246): tester si le serveur répond
  - EC [erase character] (code 247): effacer le caractère précédent
  - EL [erase line] (code 248): effacer la ligne courante, etc…
- Pour que les commandes soient plus prioritaires que les autres caractères il est possible d 'envoyer un signal « hors bande » au serveur le forçant à ignorer les caractères ordinaires et à lire la commande en priorité:
  - basé sur le bit « urgent data » du paquet TCP



## Options Telnet

- Les options sont négociés entre le client et le serveur:
  - la négociation est symétrique: le client comme le serveur peuvent négocier des options
  - une extrémité demande si l 'autre supporte une option et l 'autre extrémité accepte ou décline
- Quelques options possibles:
  - Transmit binary: utilisation de caractères à 8-bit
  - Echo: demande de faire l 'écho des caractères reçus
  - Status: demande le statut d 'une option
  - Terminal-Type: permet d 'échanger des informations sur le type de terminal utilisé et donc d 'utiliser par exemple des codes de positionnement du curseur sur l 'écran
  - Linemode: faire de l 'édition locale et envoyer des lignes entières au lieu d 'envoyer caractère par caractère
  - etc...

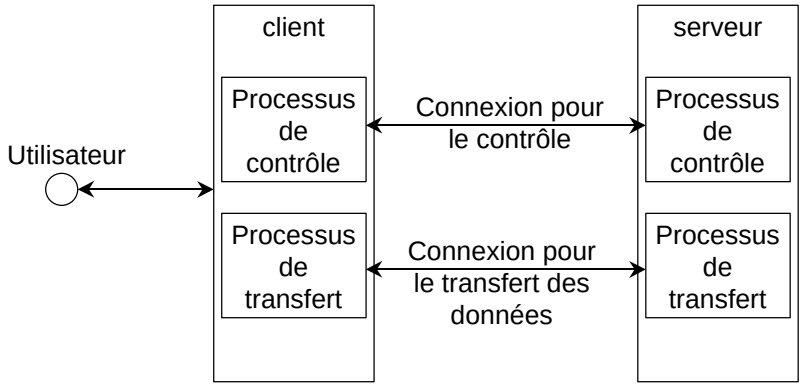
## *5ème partie*: FTP

- FTP [File Transfer Protocol] définit un protocole de transfert fiable de fichiers entre un client et un serveur (RFC 959)
- Les transferts sont à l 'initiative du client mais il peuvent tout aussi bien être:
  - des récupérations [dowloading]: récupérer un fichier sur le serveur
  - des chargements [uploading]: déposer un fichier sur le serveur
- La connexion est authentifiée par un nom d 'utilisateur et un mot de passe
- Les droits d 'accès aux fichiers dépendent de l'utilisateur et du système d 'exploitation utilisé sur le serveur
- On distingue 2 types de transferts:
  - les transferts de fichiers textes: le client et/ou le serveur convertissent les représentations des textes
  - les transferts de fichiers binaires: transfert des données brutes



## Architecture fonctionnelle de FTP

- Protocole basé sur TCP:
  - utilisation du port 21 pour les commandes
  - utilisation du port 20 pour le transfert de données



## Protocole FTP

- Le client établit une connexion sur le serveur sur le port 21 en s 'authentifiant: c 'est la **connexion de contrôle**
- Toutes les commandes sont envoyés sur cette connexion
- A chaque transfert de données une nouvelle connexion est établie entre le client et le serveur et refermée après le transfert:
  - la connexion est ouverte par le serveur vers le client qui a communiqué au préalable par la connexion de contrôle le numéro du port à utiliser
  - le client peut également établir cette connexion (mode passif) mais tous les serveurs n 'acceptent pas ce mode
- Avantages de 2 connexions:
  - plusieurs transferts peuvent être lancés en parallèle
  - le canal de contrôle reste disponible pendant les transferts (par exemple pour les interrompre)
  - on peut transférer des fichiers binaires sans risque de les mélanger avec des commandes

R. CHALON Internet et les applications sur TCP/IP 61

## Usage de FTP (1/2)

- Les clients peuvent être simples en mode texte ou présenter une interface graphique sophistiquée
- FTP ne définit pas de modèle générique de système de fichiers (comme FTAM) mais se contente de montrer et donner accès au système de fichiers du serveur:
  - en général les serveurs tournent sur UNIX mais ils peuvent très bien être sur tout autre types de machines (Windows NT, Novell, etc..) ce qui pose des problèmes de cohérences pour les utilisateurs
- Nombreuses commandes disponibles: exemple de la commande « ftp » sous Unix:
  - **open <adresse>**: ouvrir une session FTP sur un serveur
  - **dir**: afficher le contenu du rép courant du serveur (aussi **ls**)
  - **cd <répertoire>**: changer de répertoire sur le serveur
  - **lcd <répertoire>**: idem sur la machine cliente (équivalent à **!cd**)

R. CHALON Internet et les applications sur TCP/IP 62

## Usage de FTP (2/2)

- **mkdir <répertoire>**: créer un répertoire sur le serveur
- **rmdir <répertoire>**: effacer un répertoire sur le serveur
- **delete <fichier>**: effacer un fichier sur le serveur
- **rename <fichier>**: renommer un fichier sur le serveur
- **!<commande>**: exécuter une commande sur la machine cliente**:**
- **get <fichier>**: récupérer un fichier sur le serveur (aussi **recv**)
- **put <fichier>**: déposer un fichier sur le serveur (aussi **send**)
- **mget, mput, mdelete, mdir**: versions de get, put, delete et dir agissant sur plusieurs fichiers simultanément
- **binary**: passer en mode de transfert binaire
- **ascii**: passer en mode de transfert ASCII (mode par défaut)
- **status**: affiche des informations sur la connexion courante
- **help <commande>**: message d'aide sur la commande
- **close**: fermer la session
- **quit**: quitter le programme FTP, etc...



## Serveurs FTP anonymes

- Pour faciliter la distribution de logiciels comme les graticiels [freeware] ou les partagiciels [shareware] ainsi que des documents (par exemple, les RFC), il est possible de prévoir un accès anonyme sur un serveur:
  - login: **anonymous**
  - mot de passe: <adresse mél de l 'utilisateur>
- L 'accès est strictement limité à une partie restreinte du serveur et il n 'est possible que de récupérer des fichiers
- Cependant certains serveurs permettent de déposer des fichiers dans un répertoire généralement appelé « **incoming** »

## Exemple de session FTP (1/2)

```
# ftp ftp.ec-lyon.fr
connected to ftp.ec-lyon.fr
220 ftp.ec-lyon.fr FTP server ready
User: anonymous
331 guest login OK, send E-mail as password
Password: rene.chalon@ictt.ec-lyon.fr
230 guest login OK, access restrictions apply
ftp> dir
200 PORT command successful
150 ASCII data connection for /bin/ls (156.18.28.1,2363)
drw-rw-rw-  1  root  bin  25456  Mar 12 15:34  incoming
-r--r--r--  1  root  bin   1525  Jeu 14 17:03  readme.txt
226 ASCII Transfer complete.
923 bytes received in 0.17 seconds (5.43 Kbytes/sec)
ftp> get readme.txt
200 PORT command successful
150 ASCII data connection for readme.txt (156.18.28.1,2364)
226 ASCII transfer complete
1525 bytes received in 1 seconds (1.5 Kbytes/s)       .../...
```



## Exemple de session FTP (2/2)

```
ftp> cd incoming
250 CWD command successful.
ftp> !dir
   Le volume dans le lecteur C est DISQUEDUR
   Répertoire de C:\
COMMAND  COM        94 822  24/08/95   9:50 COMMAND.COM
   etc, etc…
         103 fichier(s)       10 233 371 octets
         35 répertoire(s)     57 802 752 octets libres
ftp> binary
200 Type set to I.
ftp> put command.com
200 PORT command successful
150 Binary data connection for command.com (156.18.28.1,2365)
226 Transfer complete.
94822 bytes sent in 0.33 seconds (303.3 Kbytes/sec)
ftp> close
221 goodbye
ftp> quit
```

## Autres protocoles de transfert de fichiers

- TFTP [Trivial File Transfer Protocol] (RFC 783):
  - fonctionne sur UDP: il gère donc lui-même les erreurs de transmission
  - basé sur l 'envoi de blocks numérotés de 512 octets qui doivent être acquittés avant la transmission du bloc suivant
  - protocole très simple lorsqu'on a pas besoin de la sophistication de FTP ou que le système d 'exploitation ne peut le supporter
  - Un usage typique: chargement de l 'image d 'un système d 'exploitation au démarrage d 'une machine sans disque dur (« bootstraping » réseau)
- NFS [Network File System]: partage de fichiers sur un réseau
  - ➡ ***Cf. cours sur les réseaux locaux***

R. CHALON Internet et les applications sur TCP/IP 67

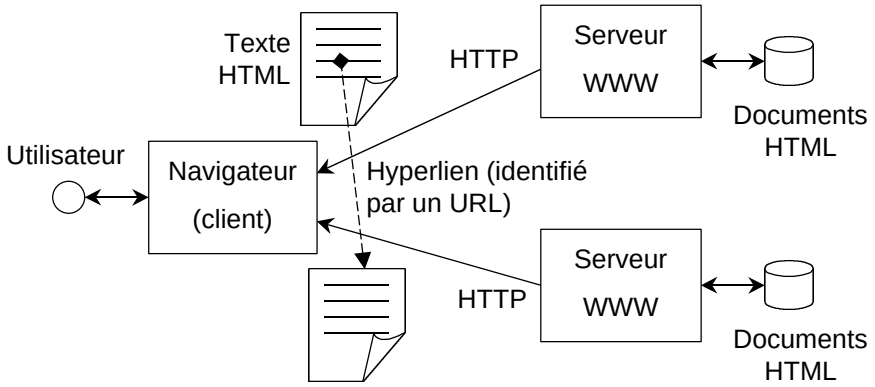
## *5ème partie*: le World Wide Web

- Le World Wide Web (WWW, W3 ou Web — en français, la Toile) désigne l 'ensemble des documents multimédia accessibles par l 'Internet et reliés entre eux par des **hyperliens**
- C 'est aujourd'hui l 'application la plus représentative de l 'Internet au point que les utilisateurs confondent les deux
- Historique:
  - le WWW est né au CERN en 1989:
    - ➡ le but était de faciliter l 'accès aux publications de recherche de la communauté de la physique des particules en les plaçant sur l 'Internet et en pouvant facilement trouver les documents associés
  - Le W3C [World Wide Web Consortium] est créé en 1994 pour favoriser le développement du Web; il est administré par le MIT et par l 'INRIA

R. CHALON Internet et les applications sur TCP/IP 68

## Architecture fonctionnelle du Web

- Les principaux éléments du Web sont:
  - un protocole client/serveur: HTTP
  - un langage d 'écriture des pages: HTML
  - un moyen de référencer et de lier les pages : les URL



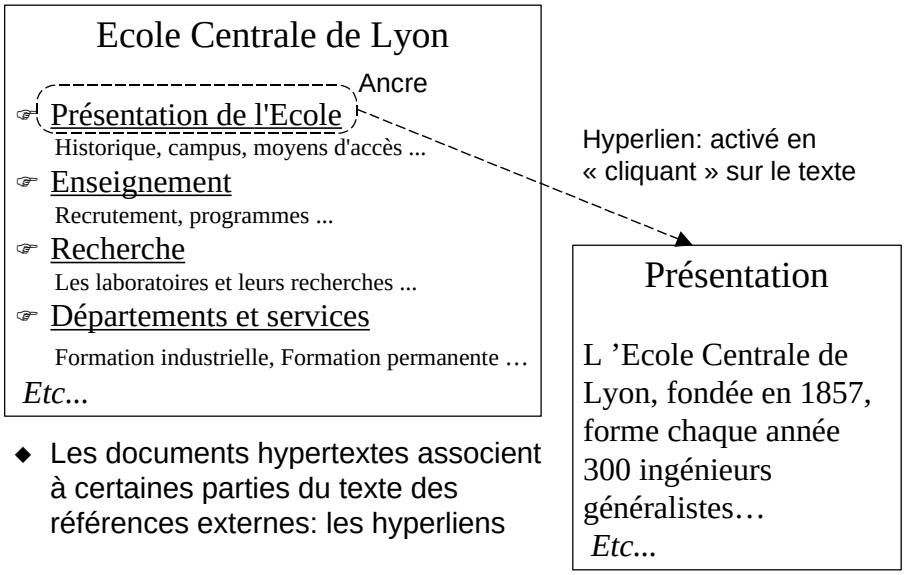

R. CHALON Internet et les applications sur TCP/IP 69

## Définitions

- HTTP [HyperText Transfer Protocol]: protocole de transfert des pages Web
- HTML [HyperText Markup Language]: langage d 'écriture des pages Web
- URL [Uniform Resource Locator]: adresse unique faisant référence à une page Web et permettant de la décharger
- Navigateur: client Web permettant de récupérer les pages et de les afficher; il permet également de suivre les hyperliens. Il est parfois appeler butineur ou fureteur (en anglais, browser)
- Hypertexte: désigne une technique dans laquelle certains éléments d 'un texte « pointent » sur d 'autres éléments du même texte ou sur d 'autres documents. C 'est le principe fondateur du WWW.

R. CHALON Internet et les applications sur TCP/IP 70

## Hypertexte et hyperliens




- Les documents hypertextes associent à certaines parties du texte des références externes: les hyperliens



## Les URL

- A chaque hyperlien est associé un URL qui forme une référence unique à tout document sur le Web
- Il est formé de 4 parties:
  - le protocole à utiliser (pour le Web: http) ①
  - le numéro du port à utiliser (par défaut pour http: 80) ②
  - le nom du serveur où se trouve le document à charger ③
  - un chemin d 'accès au document sur le serveur ④
  - un référence (label) à l 'intérieur du document ⑤
- Exemple:
  - **`http://www.ec-lyon.fr:80/presentation/index.html#signature`**


- Références relatives:
  - référence locale au document: **`"#signature"`**
  - référence à un document sur le même serveur: **`"../intro.html"`**

## Navigateur

- Le navigateur est une partie essentielle du Web car il en constitue la face visible pour l 'utilisateur
- Il est responsable de l 'affichage des documents et de fournir le moyen à l 'utilisateur d 'activer les hyperliens (en général en utilisant la souris et en cliquant sur ces liens)
- Les hyperliens sont présentés sous une forme différente (souligné, en couleur) par rapport au texte ordinaire pour les différencier
- Les navigateurs interprètent l 'URL associé à l 'hyperlien pour trouver le serveur qui contient le document
- Le navigateur s 'adresse directement à chaque serveur pour récupérer le document
- Navigateurs les plus utilisés: Internet Explorer de Microsoft et Netscape Navigator



## Serveur Web

- Chaque serveur Web écoute le port 80
- Liste des étapes pour récupérer un document (dont l 'URL est http://*nom_serveur*/*nom_document*) sur un serveur:
  - à partir de *nom_serveur* (contenu dans l 'URL), le client demande au DNS l 'adresse IP du serveur
  - le client établit une connexion TCP sur le port 80 du serveur
  - il envoie la commande: GET *nom_document* HTTP/1.0
  - le serveur envoie un en-tête et le document
  - la connexion est fermée
  - le client affiche le document
- Si le client doit demander un autre document au même serveur, il doit ouvrir une nouvelle connexion
  - peu efficace, les évolutions du protocole cherchent à améliorer cela (version 1.1)

## Protocole HTTP

- Protocole HTTP:
  - HTTP v 1.0, RFC 1945 (1996)
  - HTTP v 1.1, RFC 2068 (1997)
- HTTP fonctionne sur TCP (mais UDP pourrait aussi être utilisé)
- Protocole très simple, dialogue en ASCII:
  - le client ouvre une connexion et envoi une (ou plusieurs) commande(s) suivi d 'une ligne vide
  - le serveur répond:
    - un code (OK ou erreur) est renvoyé
    - le (ou les documents) sont envoyés en utilisant MIME
- Les transferts se font en utilisant MIME:
  - un en-tête MIME est envoyé pour spécifier la nature du document et certaines informations (date, taille)
  - le document lui-même est envoyé

R. CHALON Internet et les applications sur TCP/IP 75

## Commandes HTTP

- GET: récupérer un document
  - GET *nom_document version*
  - il est possible de demander le document que s 'il a été modifié depuis une certaine date (lié aux caches)
- HEAD: récupérer l 'en-tête d 'un document sans le contenu
  - HEAD *nom_document version*
  - utile pour récupérer la date de dernière modification ou pour faire des tests de connectivité
- POST: permet d 'envoyer des informations au serveur
  - POST *nom_document version*
  - lié aux formulaires
- PUT, DELETE: permettent d 'ajouter des documents ou d 'en supprimer (en général, désactivé)
- LINK, UNLINK: permettent d 'établir et enlever des liens

R. CHALON Internet et les applications sur TCP/IP 76

## Exemple de connexion HTTP

```
Client
       Serveur
GET /index.html HTTP/1.0
                                        <-- ligne blanche indispensable
       HTTP/1.0 200 document follows
       MIME-Version: 1.0
       Server: CERN/3.0
       Content-Type: text/html
       Content-Length: 1540
                                        <-- ligne blanche indispensable
       <HTML> <HEAD>
       <TITLE>Ecole Centrale de Lyon: Sommaire</TITLE>
       <HEAD>
       <BODY>
       etc…
       </BODY> </HTML>
```



## HTML

- Un document est la plus petite unité échangée
- Un document fourni par un serveur Web est écrit en format HTML et il est interprété par le client
- Le texte HTML contient:
  - le texte proprement dit,
  - des informations de mise en page: centrer, gras, italique, etc…, contenues dans des **balises**
  - les informations pour les hyperliens, les ancres
- HTML est dérivé de SGML [Standard Generalized Markup Language] (ISO 8879): langage de balisage généralisé
- Plusieurs versions:
  - v 1.0: la plus simple; n 'est plus utilisée
  - v 2.0: RFC 1866 (1995)
  - v 3.2: version « actuelle » (1996); assez riche mais beaucoup d 'extensions propriétaires de Microsoft et Netscape
  - v 4.0: nouvelle version (24 avril 1998)

## Structure d 'un fichier HTML

- Un fichier HTML contient le texte du document complété par des **balises** [tags] qui permettent de formater le texte:
  - exemple: <B> texte en gras </B> apparaît comme: **texte en gras**
- les espaces supplémentaires, les tabulations et les retours à la ligne sont ignorés: le navigateur reformate le texte entre les marges de la fenêtre d 'affichage:
  - pour aller à la ligne suivante il faut utiliser une balise: <BR>, <P> …
- Les balises ne précisent pas l 'apparence exacte du texte mais des indications utilisées par le navigateur pour formater le texte
  - exemple: la balise <H1> signifie « titre de niveau 1 » et sera en général rendu par le navigateur dans une police plus grande que le texte normal et en caractères gras
- Cependant la tendance actuelle est de rajouter de plus en plus de balises qui précisent les détails de l 'apparence: police (style et taille), couleurs, etc...



## Codage des caractères

- Les textes sont par défaut écrits dans le jeu de caractères ISO 8859-1 (Latin 1): extension du code ASCII aux caractères de l 'Europe de l 'Ouest
- Pour pouvoir encoder les fichiers en ASCII pur, les caractères étendus sont codés:
  - &nom; où nom désigne le caractère
  - exemples: é se code **`&egrave;`** è: **`&eacute;`** ç: **`&ccedil;`**
- Les caractères <, > et & ayant une signification spéciale, il doivent être codés pour apparaître dans un texte:
  - < est codé **`<`** (pour lower than)
    > est codé **`>`** (pour greater than)
    & est codé **`&`** (pour ampersand)
- Il est possible d 'utiliser d 'autres jeux de caractères pour les langues qui n 'utilisent pas les caractères latins

## Quelques balises HTML (1/4)

- Syntaxe des balises:
  - **<***balise* [*options*]**>** *texte* **</***balise***>**
- Format du document:
  - document complet: **<HTML>** *document* **</HTML>**
  - en-tête du document: **<HEAD>** *en-tête* **</HEAD>**
  - corps: **<BODY>** *texte du document* **</BODY>**
- Titres:
  - titre du document (il ne peut apparaître que dans l 'en-tête du document): **<TITLE> titre </TITLE>**
    ce titre n 'est pas affiché dans le document mais est utilisé par le navigateur (en général comme titre de sa fenêtre)
  - 6 niveaux de titres possibles:
    **<H1>** *titre niveau 1* **</H1>** à
    **<H6>** *titre niveau 6* **</H6>**

**Titre niveau 1**

**Titre niveau 2**

**Titre niveau 3**

Texte normal



## Quelques balises HTML (2/4)

- Paragraphes:
  - normal: **<P>** *texte* **</P>** (la balise /P est facultative)
  - avec indentation: **<BLOCKQUOTE>** *texte* **</BLOCKQUOTE>**
  - retour à la ligne: **<BR>**
- Ancres:
  - lien vers un autre document: **<A HREF=***url***>** *texte* **</A>**
  - label dans un document: **<A NAME=***label***>** *texte* **</A>**
  - Exemples:
    - **<A HREF="http://www.ec-lyon.fr/index.html">**:pointe sur le document « index.html » sur le serveur « www.ec-lyon.fr »
    - **<A HREF="#signature">**: pointe sur le label « signature » dans la même page
    - **<A NAME="signature">**: définition du label « signature »
  - le texte qui sert d 'ancre peut être de longueur et style quelconque, y compris une image

## Quelques balises HTML (3/4)

- Séparateur:
  - ligne horizontale: **<HR>**
- Styles de caractères:
  - gras: **<G>** *texte* **</G>**
  - italique: **<I>** *texte* **</I>**
  - souligné: **<U>** *texte* **</U>**
  - mais aussi **<STRONG>, <EM>** qui sont des styles « logiques » (en général affichés en gras et italique)

Exemple: ***important***
est codé:
**<G><I>important</I></G>**

- Images:
  - seules les images GIF et JPEG sont utilisables
  - images incluses: **<IMG SRC=***url* **ALT=***texte***>**
  - Exemple: **<IMG SRC="logo.gif" ALT="notre logo">**
  - Il existe de nombreux paramètres pour positionner l 'image par rapport au texte environnant: alignement, habillage, etc..



## Quelques balises HTML (4/4)

- Listes:
  - non-numérotées (une puce devant chaque ligne):
    **<UL>**
    **<LI>** *item 1*
    **<LI>** *item 2 …*
    **</UL>**

    ☞ Item 1
    ☞ Item 2
  - numérotées (numérotation automatique):
    **<OL>**
    **<LI>** *item1*
    **<LI>** *itme2 …*
    **</OL>**

    **1.** Item 1
    **2.** Item 2
  - définitions:
    **<DL>**
    **<DT>** *titre 1*
    **<DD>** *définition 1*
    **<DT>** *titre 2*
    **<DD>** *définition 2 …*
    **</DL>**

    ☞ Titre 1
    Définition 1
    ☞ Titre 2
    Définition 2

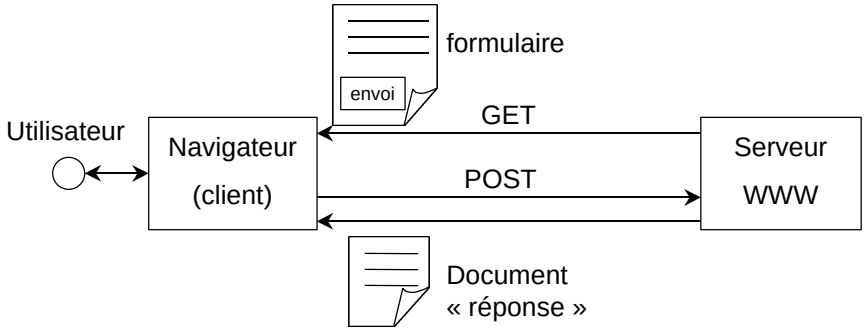
## Exemple de fichier HTML

```
<HTML> <HEAD>
<TITLE>Ecole Centrale de Lyon: Sommaire</TITLE>
<HEAD>
<BODY>
<H1><CENTER>Ecole Centrale de Lyon</CENTER></H1>
<P><DL>
<DT>* <A HREF="presentation/index.html">Pr&eacute;sentation
   de l'Ecole</A></LI>
<DD>Historique, campus, moyens d'acc&egrave;s ...
<DT>* <A HREF="enseignement.html">Enseignement</A>
<DD>Recrutement, programmes ...
<DT>* <A HREF="recherche.html">Recherche</A>
<DD>Les laboratoires et leurs recherches ...
<DT>* <A HREF="departement.html">D&eacute;partements et
   services</A>
<DD>Formation industrielle, Formation permanente …
</DL>
</BODY> </HTML>
```



## Formulaires HTML (1/2)

- Les formulaires sont des documents HTML contenant des balises spéciales qui permettent de définir des champs que l ’utilisateur peut remplir
- Un bouton permet d ’envoyer le contenu des champs au serveur (grâce à la commande POST) qui, en fonction des valeurs reçues pourra renvoyer un document personnalisé



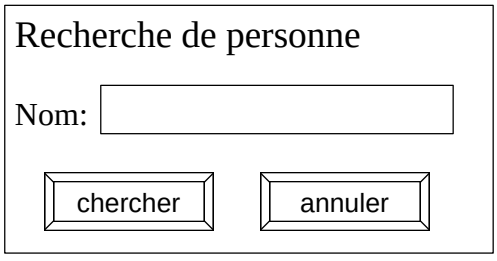
## Formulaires HTML (2/2)

- Un formulaire est encadré par les balises <FORM> … </FORM>
- Il existe de nombreux objets possibles:
  - des champs textuels,
  - des boutons radios et des cases à cocher,
  - des listes à choix uniques ou multiples,
  - des boutons pour l 'envoi des données et la remise à zéro des champs
- Chaque objet est défini par une balise qui comporte de nombreux paramètres pour ajuster son apparence et son fonctionnement
- L 'apparence exacte dépend du système d 'exploitation sur lequel tourne le client car ce sont les objets d 'interface du système d 'exploitation du client qui sont utilisés



## Exemple de formulaire (1/2)

```
<HTML> <HEAD>
<TITLE>Recherche</TITLE>
<HEAD>
<BODY>
<H1>Recherche de personne</H1>
<FORM ACTION= »http://www.ec-lyon.fr/cgi-bin/search"
  METHOD=POST>
Nom:<INPUT TYPE=TEXT NAME="mot" SIZE=20>
<P><INPUT TYPE=SUBMIT
  VALUE="chercher">
<INPUT TYPE=RESET
  VALUE="annuler">
</FORM>
</BODY>
</HTML>
```



## Exemple de formulaire (2/2)

- Lorsque l 'utilisateur clique sur le bouton « chercher », l 'action spécifiée dans la balise <FORM> est exécutée:
  - le client établit une connexion avec le serveur et envoie la commande: POST /cgi-bin/search
  - il transmet la valeur de tous les champs (dans ce cas « mot »)
  - *search* correspond à un programme externe qui est exécuté sur le serveur et qui utilise la valeur des champs pour traitement
  - le serveur renvoie un document, par exemple ici, la liste des personnes trouvées.
- La manière de passer les paramètres au programme externe est standardisé:
  - interface CGI [Common Gateway Interface]



## Liens avec les autres protocoles

- Les URL ont été généralisés pour d 'autres protocoles:
  - FTP anonyme
    - exemple: ftp://ftp.ec-lyon.fr/pub/readme.txt
  - fichiers locaux
    - exemple: file:///c|/essai/index.html
  - messagerie:
    - exemple: mailto:Rene.Chalon@ictt.ec-lyon.fr
  - telnet:
    - exemple: telnet://catalogue.bm-lyon.fr
  - autres: gopher, les news
- Les navigateurs supportent en général ces protocoles et sont donc capables d 'afficher des liens qui pointent sur des services externes au Web
  - par exemple il est possible de parcourir un serveur FTP anonyme avec Netscape

## Java (1/2)

- Java est un langage orienté objet qui a été développé par Sun Microsystems tout particulièrement pour fonctionner sur le Web
- Les programmes Java peuvent être de deux types:
  - des programmes « classiques »
  - des appliquettes [applets] qui sont chargées depuis un serveur Web et qui s ’exécutent dans l ’environnement local du navigateur
- Java est un langage semi-compilé:
  - le source est compilé dans un code intermédiaire
  - le code intermédiaire est interprété sur la machine locale
  - ➡ Java est donc indépendant des plate-formes: il suffit de disposer d ’un interpréteur de code intermédiaire
- Les appliquettes sont incorporables à l ’intérieur d ’un document HTML de la même manière que les images
  - balise: <APPLET CODE=essai.class> </APPLET>



## Java (2/2)

- Une appliquette Java peut, au cours de son fonctionnement établir des connexions avec le serveur d ’où elle a été chargée mais, pour des raisons de sécurité:
  - elle ne peut écrire sur le disque de la machine du navigateur,
  - elle ne peut établir de connexion avec d ’autres serveurs
- L ’Intérêt des appliquettes Java réside dans la possibilité d ’étendre les fonctionnalités du navigateur:
  - au démarrage, le navigateur contient juste un interpréteur Java
  - pour pouvoir afficher les documents HTML, il suffit de charger une appliquette capable d ’afficher du code HTML
  - et ainsi de suite pour tous les médias (images, son, vidéo, etc…)
  - ➡ ainsi, si un nouveau média apparaît, il suffit de programmer l ’appliquette correspondante et tous les navigateurs peuvent afficher ce nouveau média sans modifications