Université Nice SOPHIA ANTIPOLIS

Introduction au Web et Services Web

Département Sciences Informatiques
Jean-Yves Tigli – tigli@polytech.unice.fr
http://www.tigli.fr

SI 3ème année

Quelques informations sur le cours et les TDs

- ✓ Les supports et matériels pour le cours se trouve sur www.tigli.fr
- ✓ Les TDs se feront en python

- ✓ Certains TDs seront à rendre mais pas forcément notés.
- ✓ Ceci vous sera signifié dans le sujet ou en début de séance. Le rendu du TD pourra se faire en l'état, en fin de séance, ou après un certain délai qui sera mentionné.
- ✓ Vous serez évalués sur tout ou partie des TDs et avec des QCMs en début de certains cours.
- ✓

Université Nice SOPHIA ANTIPOLIS

Introduction sur le plan des trois séances :

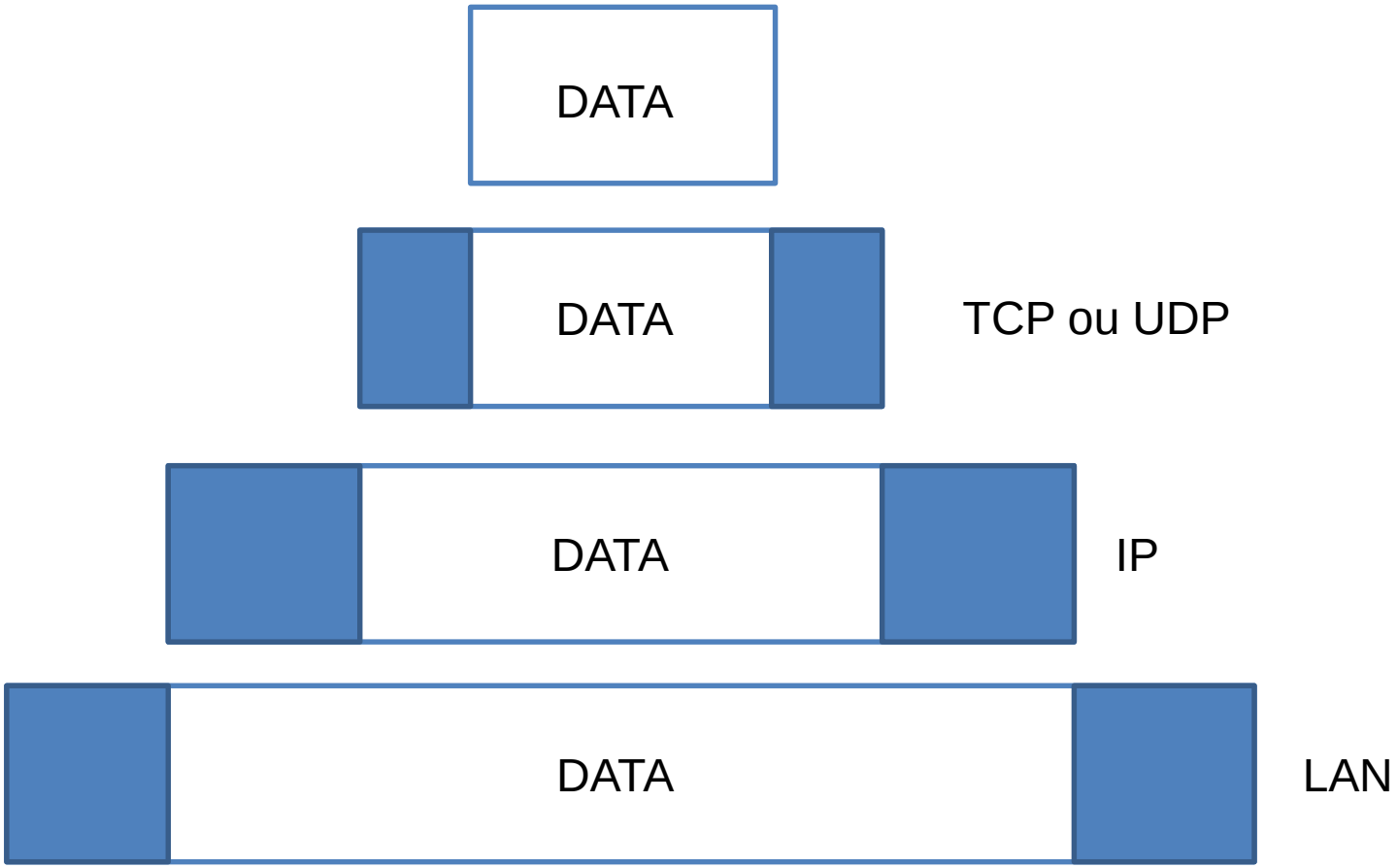
3. Des socket TCP/IP au Web statique
2. Du Web Statique au Web Dynamique
3. Du Web dynamique au Services Web

3 semaines

J.-Y. Tigli
tigli@polytech.unice.fr

Le web et les sockets un protocole applicatif particulier

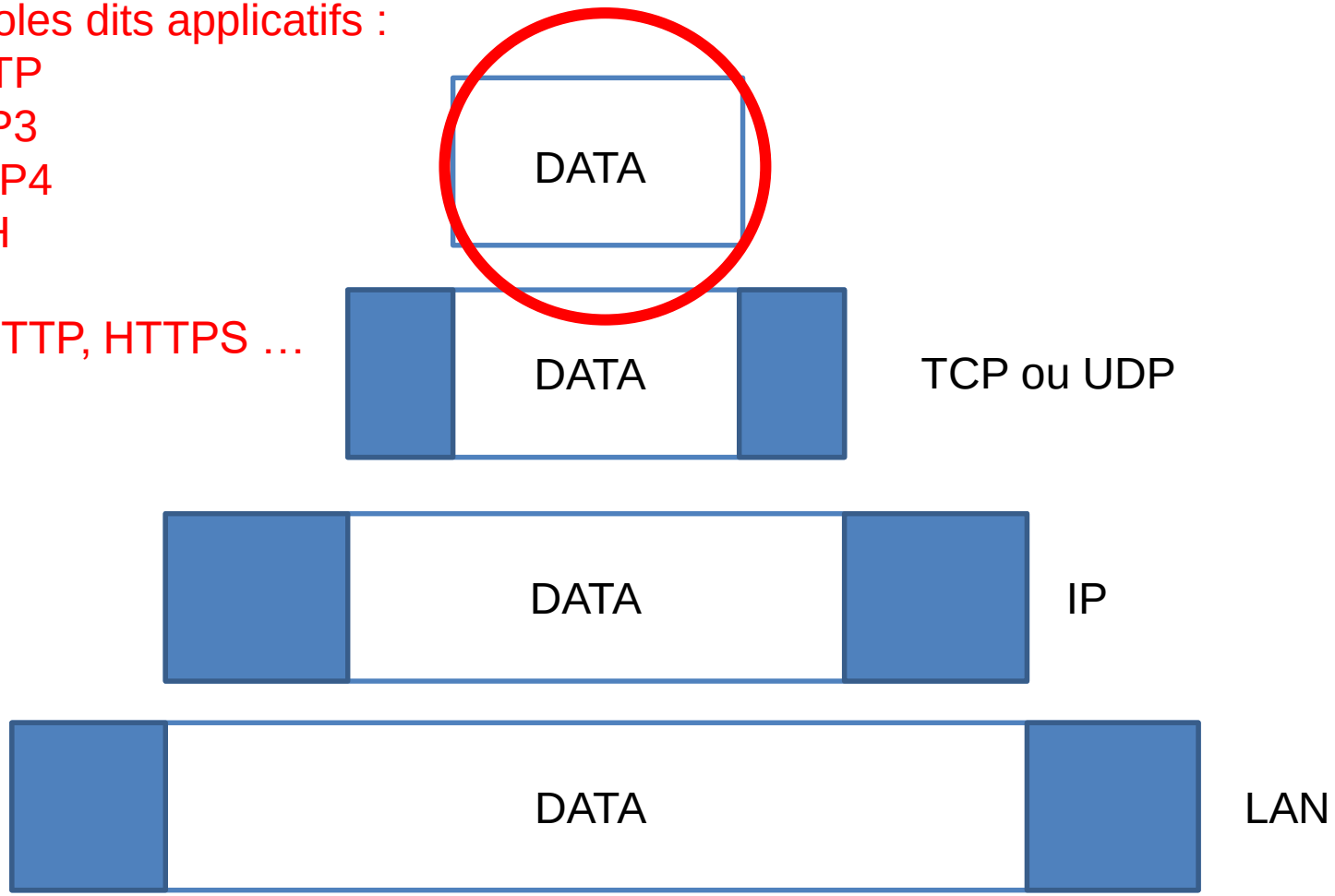
Rappel sur la pile TCP/IP



Rappel sur la pile TCP/IP

Protocoles dits applicatifs :

- SMTP
- POP3
- IMAP4
- SSH
- …
- et HTTP, HTTPS …



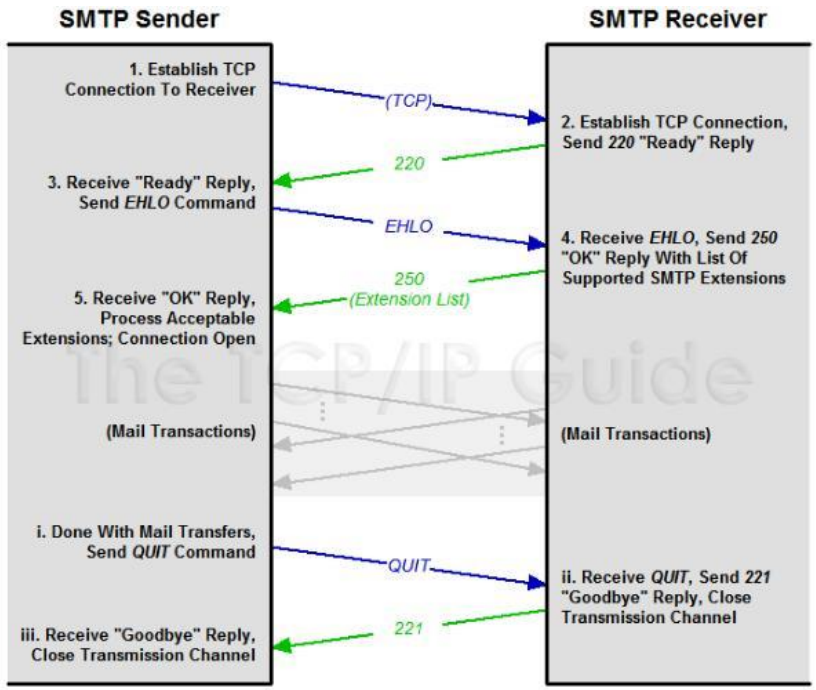
Exemple SMTP ...

```
telnet smtp.xxxx.xxxx 25
Connected to smtp.xxxx.xxxx.
220 smtp.xxxx.xxxx SMTP Ready
HELO client
250-smtp.xxxx.xxxx
250-PIPELINING
250 8BITMIME
MAIL FROM: <auteur@yyyy.yyyy>
250 Sender ok
RCPT TO: <destinataire@xxxx.xxxx>
250 Recipient ok.
DATA
354 Enter mail, end with "." on a line by itself
Subject: Test

Corps du texte
.
250 Ok
QUIT
221 Closing connection
Connection closed by foreign host.
```

En général un protocole dit applicatif ...

- ✓ Un format de messages :
- ✓ Ex. SMTP :
  - Codage ASCII
  - Format : <command> <arg>
- ✓ Des séquences définies
  - automate ou
  - diagramme de séquence



Introduction au web

Histoire et Motivations du Web

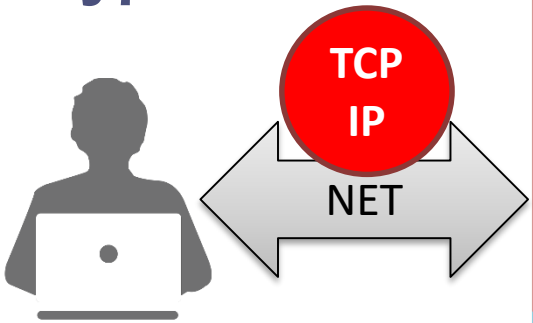
- ✓ A l'origine du web, est derrière un homme qui s'appelle Tim Berners_Lee, un informaticien du CERN En mars 1989, il rédige un projet.
- ✓ Son idée est de pouvoir échanger et partager de l'information avec ses collaborateurs.
- ✓ Le CERN étant une communauté scientifique qui réunit 80 pays, il voulait faciliter la communication entre lui et ses collaborateurs.

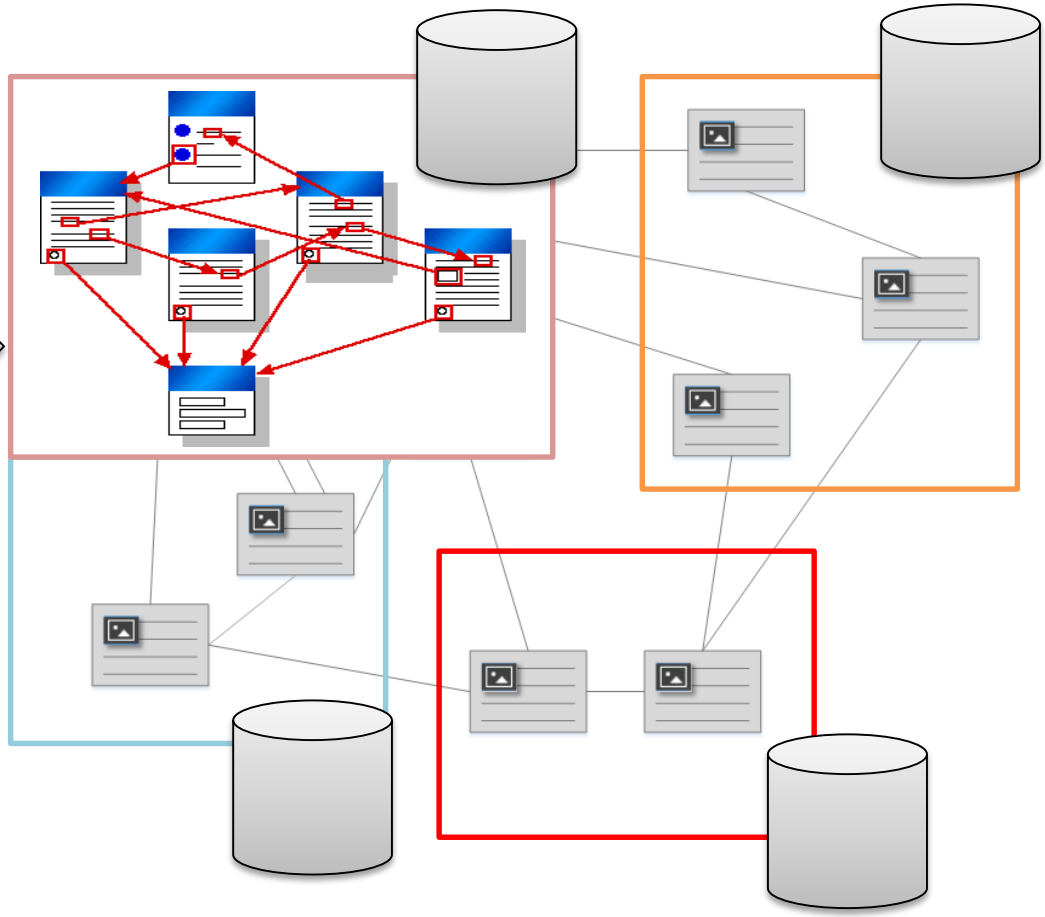
Histoire et Motivations du Web

- ✓ Système hypertexte partagé sur le réseau de l'informatique
- ✓ Une alternative à la Gestion Electronique de Documents plus rigide
- ✓ En 1990, il a mis au point le protocole HTTP (HyperText Transfer Protocol), ainsi que le langage HTML (HyperText Markup Language).
- ✓ Après l'avoir testé au CERN, le 6 août 1991, il donne officiellement vie au World Wilde Web via un message adressé au public avec l'objectif de l'accès à l'information depuis n'importe où et pour tout le monde.

Architecture Web

- ✓ Serveurs,
- ✓ Pages
- ✓ Liens hypertexte





Client / Serveur TCP/IP Particuliers

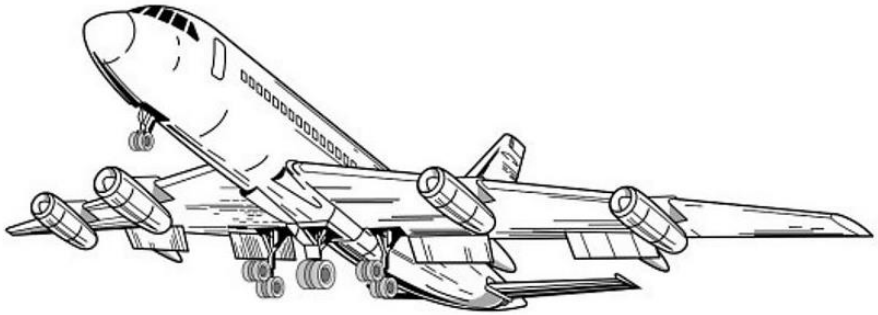
- ✓ Page Web:
  - Pointés par une URL
  - La plupart des pages WEB se composent de:
    - Une page HTML de base,
    - Différentes références à des « objets »

- ✓ L'agent utilisateur (client) pour le Web s'appel un "**browser**" (butineur en français)
  - Microsoft Internet Explorer, Mozilla FireFox, Opera, Safari, Google Chrome, ...
- ✓ Un serveur pour le Web s'appelle un serveur Web :
  - Apache, Microsoft Internet Information Server (IIS), ...

Au cœur : le Protocole HTTP

HyperText Transfert Protocol

HTTP 1.0 : RFC 1945
HTTP 1.1 : RFC 2616



Introduction

Les Principes et Eléments de base du Protocole

HyperText Transfer Protocol

- ✓ *HTTP : HyperText Transfert Protocole*
  - Un des protocole les plus courants sur Internet
    - Un protocole omniprésent: de IT à Embedded
  - Il est utilisé pour la navigation sur les sites Web
    - protocole de rapatriement des documents
    - protocole de soumission de formulaires

- ✓ Il en existe trois versions :
  - 0.9 (1991) : complètement obsolète
  - 1.0 (février 1997), de nos jours très rarement utilisée
  - 1.1 (octobre 2000). Les principaux changements entre les v1.0 et v1.1 sont l'ajout de 2 types de requêtes ainsi que la possibilité d'héberger plusieurs sites Web sur un même serveur dans la version 1.1.

Adressage des l'hyperlien : URL - Uniform Resource Locator

- ✓ Une URL (Uniform Resource **Locator**) a au moins deux champs (protocole, adresse de la ressource)
  - **Le protocole**: schéma de représentation
  - **L'Adresse** : localisation complète de la ressource
  - Ex:
    - mailto:Quidam.no-spam@example.com
    - news:fr.comp.infosystemes.www.auteurs
- ✓ Une URL HTTP a au moins trois champs (protocole, adresse, emplacement) :
  - **Le protocole**: *http* suivi de :
  - **L'adresse**: le nom complet de la ressource // *login* : *password* @ *nom domaine* : *port*
  - **Emplacement**: Emplacement de la ressource à l'adresse
  - Données supplémentaires optionnelles transmises

Exemple :
http://Jojo:lApIn@www.example.com:8888/chemin/index.html

Principe de Fonctionnement de HTTP

- ✓ TCP/IP transport service
  - Le client initialise une connexion TCP/IP (voir sockets) sur le serveur et le port 80.
  - Le serveur accepte la connexion du client et fournit un port de communication (utilisateur).
  - Les messages http (messages au protocole de l'application) sont échangés entre le client http et le serveur http.
  - Enfin, la connexion TCP/IP est fermée.

- ✓ HTTP est "stateless"
  - En principe, le serveur ne maintient pas d'information sur les requêtes passées du client.
  - En pratique, certaines techniques le permette
  - HTTP : Transport ou Session ?

Dialogue HTTP

- ✓ Dialogue
  - en mode caractères ASCII (7 bits)
    - telnet www.sun.com 80
- ✓ Types de Commandes
  - Récupération d'un document
    - méthode GET
  - Soumission d'un formulaire
    - méthodes GET ou POST
  - Envoi de Document et Gestion de Site
    - méthodes PUT, DELETE, LINK, UNLINK
  - Gestion de proxy/cache
    - méthode HEAD (récupération des informations sur le document)

Exemple d'Echanges

REQUETE : http://www.unice.fr/chemin/file.html

1. Le client http initialise une connexion TCP sur le serveur http *www.unice.fr*. (sur le port 80)

2. Le serveur http *www.unice.fr* en l'attente de connexions sur le port 80, accepte la demande de connexion du client

3. Le client http envoie un message de requête GET /chemin/file.html (une partie de l'URL) au travers le socket de communication TCP.

Exemple d'Echanges

REQUETE : http://www.unice.fr/chemin/file.html

1. Le serveur http reçoit le message de requête, récupère le fichier file.html dans $WebRoot/chemin/file.html et compose le message de réponse contenant les fichier demandé et renvoie le message au travers le socket de communication.

2. Le client http reçoit le message de réponse contenant le fichier HTML et l'affiche.

3. Le serveur http ferme la connexion.

4. En « parsant » le fichier HTML, le client http trouve 10 références sur d'autres fichiers (jpeg, html ..). Les étapes 1 à 6 sont répétées pour chaque référence aux fichiers ...

A vous ....

- ✓ Qu'elles sont les étapes au niveau TCP/IP et HTTP pour récupérer les pages suivantes ?
- ✓ http://localhost:8080/documents/cours.html
- ✓ http://192.168.0.1:4545/doc/exercices/sujet1.xml
- ✓ http://jacques@localhost:4540/index.html
- ✓ http://edt.polytech.unice.fr/
- ✓ https://www.google.fr/

- ✓ Autre ... ftp://raymond@server.unice.fr/doc/exercices/sujet1.xml
- ✓ Démo Web Browser en mode développeur ...

Structure

- ✓ **Les requêtes et les réponses sont bâties sur le même modèle**

```
{Ligne d'introduction}{SEP}
{En-têtes séparées par des {SEP}}
{SEP}{SEP}
{Corps}
```

- ✓ **Le seul élément capable de différencier une requête d'une réponse, c'est la** *Ligne d'introduction.*

Format de la Requête

<Méthode> <URI> HTTP/<Version>
[<Champ d 'entête>: <Valeur>]
[<tab><Suite Valeur si >1024>]
ligne blanche
[corps de la requête pour la méthode POST]

```
GET /docu2.html HTTP/1.0
Accept: www/source
Accept: text/html
Accept: image/gif
User-Agent:  Lynx/2.2  libwww/2.14
From:  alice@pays.merveilles.net
 * une ligne blanche *
```

```
POST /script HTTP/1.0
Accept: www/source
Accept: text/html
Accept: image/gif
User-Agent:  Lynx/2.2  libwww/2.14
From:  alice@pays.merveilles.net
Content-Length: 24
   * une ligne blanche *
name1=value1&
name2=value2
```

Source: Didier Donsez

Méthodes de la Requête

- ✓ GET
  - demande pour obtenir des informations et une zone de données concernant l 'URI
- ✓ HEAD
  - demande pour seulement obtenir des informations concernant l 'URI
- ✓ POST
  - envoie de données (contenu du formulaire vers le serveur, requête SOAP ...). Ces données sont situées après l'entête et un saut de ligne
- ✓ PUT
  - enregistrement du corps de la requête à l 'URI indiqué
- ✓ DELETE
  - suppression des données désignées par l 'URI

Méthodes de la Requête

- ✓ OPTIONS
  - demande des options de communication disponibles
- ✓ TRACE
  - retourne le corps de la requête intacte (débogage)
- ✓ LINK / UNLINK
  - association (et désassociations) des informations de l'entête au document sur le serveur
- ✓ Nouvelles extensions de WebDAV
  - PROPFIND, PROPPATCH, MKCOL,COPY, MOVE, LOCK, UNLOCK
- ✓ Nouvelles extensions HTTP/U HTTP/MU
  - NOTIFY, ... (UPnP)

Champs d'Entête

- ✓ Ils permettent la transmission d'informations complémentaires sur la requête, et le client lui-même.
- ✓ Ces champs agissent comme "modificateurs" de la requête, utilisant une sémantique identique à celle des paramètres passés par un appel d'une méthode de langage de programmation de haut niveau.

Format de la Réponse

HTTP/<Version> <Status> <Commentaire Status>
Content-Type: <Type MIME du contenu>
[< Champ d 'entête >: <Valeur>]
[<tab><Suite Valeur si >1024>]
Ligne blanche
Document

```
HTTP/1.0 200 OK
Date: Wed, 02Feb97 23:04:12 GMT
Server: NCSA/1.1
MIME-version: 1.0
Last-modified: Mon,15Nov96 23:33:16 GMT
Content-type: text/html
Content-length: 2345
    * une ligne blanche *
<HTML><HEAD><TITLE> ...
</BODY></HTML>
```

Source: Didier Donsez

Statuts des Réponses HTTP (RFC2068)

- ✓ 1xx Information
  - 100 : Continue (le client peut envoyer la suite de la requête), ...
- ✓ 2xx Succès de la requête client
  - 200: OK, 201: Created, 204 : No Content, ...
- ✓ 3xx Redirection de la Requête client
  - 301: Redirection, 302: Found, 304: Not Modified, 305 : Use Proxy,
- ✓ 4xx Requête client incomplète
  - 400: Bad Request, 401: Unauthorized, 403: Forbidden, 404: Not Found
- ✓ 5xx Erreur Serveur
  - 500: Server Error, 501: Not Implemented,
  - 502: Bad Gateway, 503: Out Of Resources (Service Unavailable)

Entêtes HTTP

Entêtes HTTP

- ✓ 4 types de champs d'entête
  - Général
    - Commun au serveur, au client ou à HTTP
  - Requête du client
    - formats de documents et paramètres pour le serveur
  - Réponse du serveur
    - informations concernant le serveur
  - Entité
    - informations concernant les données échangées

Entêtes Généraux

- ✓ Cache-Control
  - contrôle du caching.
- ✓ Connection = listes d'option
  - close pour terminer une connexion.
- ✓ Date
  - date actuelle (format RFC1123 mais aussi RFC850).
- ✓ MIME-Version
  - version MIME utilisé.
- ✓ Pragma
  - instruction pour le proxy.
- ✓ Transfer-Encoding
  - type de la transformation appliquée au corps du message.
- ✓ Via
  - utilisé par les proxys pour indiquer les machines et protocoles intermédiaires.
- ✓ ....

Entêtes de Requêtes Client (1)

- ✓ Accept
  - type MIME visualisable par l 'agent
- ✓ Accept-Encoding
  - méthodes de codage acceptées
  - compress, x-gzip, x-zip
- ✓ Accept-Charset
  - jeu de caractères préféré du client
- ✓ Accept-Language
  - liste de langues
  - fr, en, ...
- ✓ Authorization
  - type d 'autorisation
  - BASIC nom:mot de passe (en base64) (donc en transmis en clair!)
  - NB : Préalablement le serveur a répondu un WWW-Authenticate
- ✓ Cookie
  - cookie retourné

Entêtes de Requêtes Client (2)

- ✓ From
  - adresse email de l 'utilisateur
  - rarement envoyé pour conserver l'anonymat de l 'utilisateur
- ✓ Host
  - spécifie la machine et le port du serveur
  - un serveur peut héberger plusieurs serveurs
- ✓ If-Modified-Since
  - condition de retrait
  - la page n'est transférée que si elle a été modifiée depuis la date précisée. Utilisé par les caches
  - indique si le document demandé peut être caché ou pas.
- ✓ If-Unmodified-Since
  - condition de retrait
- ✓ ...

Entêtes de Requêtes Client (3)

- ✓ Max-Forwards
  - nombre max de proxy
- ✓ Proxy-Authorization
  - identification
- ✓ Range
  - zone du document à renvoyer
  - bytes=x-y (x=0 correspond au premier octet, y peut être omis pour spécifier jusqu 'à la fin)
- ✓ Referer
  - URL d 'origine
  - page contenant l'ancre à partir de laquelle le visualisateur a trouvé l'URL.
- ✓ User-Agent
  - modèle du visualisateur

Entêtes de Réponses Serveur

- ✓ Accept-Range
  - accepte ou refus d ′une requête par intervalle
- ✓ Age
  - ancienneté du document en secondes
- ✓ Proxy-Authenticate
  - système d ′authentification du proxy
- ✓ Public
  - liste de méthodes non standards gérées par le serveur
- ✓ Retry-After
  - date ou nombre de secondes pour un ressai en cas de code 503 (service unavailable)
- ✓ Server
  - modèle de HTTPD
  - utilisé par Satan !!!!
- ✓ Set-Cookie
  - crée ou modifie un cookie sur le client
- ✓ WWW-Authenticate
  - système d ′authentification pour l ′URI

Entêtes d'Entité (1)

- ✓ Allow
  - méthodes autorisées pour l 'URI
- ✓ Content-Base
  - URI de base
  - pour la résolution des URL
- ✓ Last-Modified
  - date de dernière modification du doc.
  - Utilisé par les caches
- ✓ Content-Length
  - taille du document en octet
  - utilisé par le client pour gauger la progression des chargements
- ✓ Content-Encoding
  - type encodage du document renvoyé
  - compress, x-gzip, x-zip
- ✓ Content-Language
  - le langage du document retourné
  - fr, en ...
- ✓ ...

Entêtes d'Entité (2)

- ✓ Content-MD5
  - résumé MD5 de l 'entité
- ✓ Content-Range
  - position du corps partiel dans l 'entité
  - bytes x-y/taille
- ✓ Content-Transfert-Encoding :
  - transformation appliqué du corps de l 'entité
  - 7bit, binary, base64, quoted-printable
- ✓ Content-Type
  - type MIME du document renvoyé
  - utilisé par le client pour sélectionner le visualisateur (plugin)
- ✓ Etag
  - transformation appliqué du corps de l'entité
  - 7bit, binary, base64, quoted-printable

Entêtes d'Entité (3)

- ✓ Expires
  - date de péremption de l 'entité
- ✓ Last-Modified
  - date de la dernière modification de l 'entité
- ✓ Location
  - URI de l'entité
  - quand l 'URI est à plusieurs endroits
- ✓ URI
  - nouvelle position de l'entité
- ✓ ...

Les Serveurs du Marché











- Apache Web Server
- Apache Tomcat
- Microsoft IIS Windows Web Server
- Nginx web server
- lighttpd web server
- The Jigsaw web server software from W3C
- Klone web server
- Abyss web server
- Oracle Web Tier
- X5 (formerly Xitami) web server
- Zeus Technology Ltd. - Zeus web server

Apache

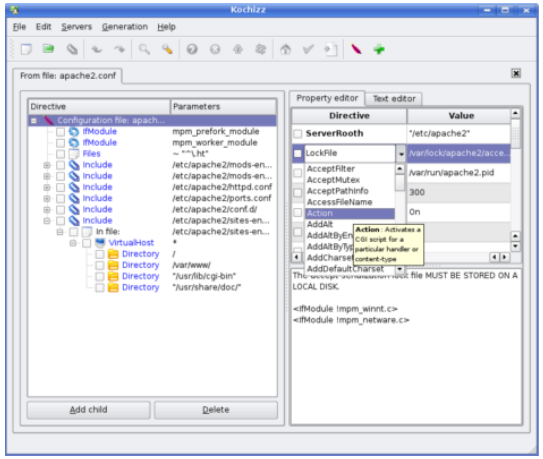
- A patch of NCSA HTTPD
  - serveur le plus répandu (« toujours » la ‚plus grosse part de marché)
  - gratuit, issu du serveur NCSA HTTPD
  - très nombreuses plates-formes Unix et Windows NT
  - extensible par des modules tiers
- Nombreux Modules Tiers
  - possibilité d'étendre Apache avec des modules externe http://www.zyzzzyva.com/server/module_registry)
    - mod_auth_cookies_file, mod_auth_cookies_msql, mod_cgi_sugid, mod_perl, mod_perl_fast, mod_auth_kerb, mod_auth_dbi, mod_rewrite, mod_jserv(servlet), mod_java (CGI écrit en Java), php3
  - nombreux sous-projets autour de Java (Jakarta) et XML (Xerces, Xalan, XSP, Cocoon, ...)

Configuration Apache

- ✓ Fichiers de configuration
  - httpd.conf
    - comportement de base port TCP/IP, journaux, keepalive, UID, virtualhost, proxy, ...
  - Les autres fichiers sont rajouté à l'aide de l'instruction *Include*
    - Exemples :
    - Include /usr/local/apache2/conf/ssl.conf
      Include /usr/local/apache2/conf/vhosts/*.conf

Outil GUI : Kochizz, éditeur de configuration Apache.

…



Quelques Manipulations ....

✓ **1. Utilisation de Telnet pour contacter un serveur Web :**

```
telnet www.unice.fr 80
```

Ouvre une connexion sur le port 80 (port par défaut) de www.unice.fr

Tout ce qui est tapé est maintenant transmis au serveur sur le port 80

2. Envoi d'une requête GET

```
GET /index.html HTTP/1.0
```

En tapant ceci, vous envoyez cette requête GET, minimale mais complète au serveur http (suivi de 2 « retour chariot »).

3. Récupération de la réponse du serveur Web

Rappel python serveur TCP/IP

server.py

```
#!/usr/bin/env python
# coding: utf-8

import socket

socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
socket.bind(('', 15555))

while True:
        socket.listen(5)
        client, address = socket.accept()
        print "{} connected".format( address )

        response = client.recv(255)
        if response != "":
                print response

print "Close"
client.close()
stock.close()
```

Rappel python pour client TCP/IP

client.py

```
#!/usr/bin/env python
# coding: utf-8

import socket

hote = "localhost"
port = 15555

socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
socket.connect((hote, port))
print "Connection on {}".format(port)

socket.send(u"Hey my name is Olivier!")

print "Close"
socket.close()
```