Université Nice SOPHIA ANTIPOLIS

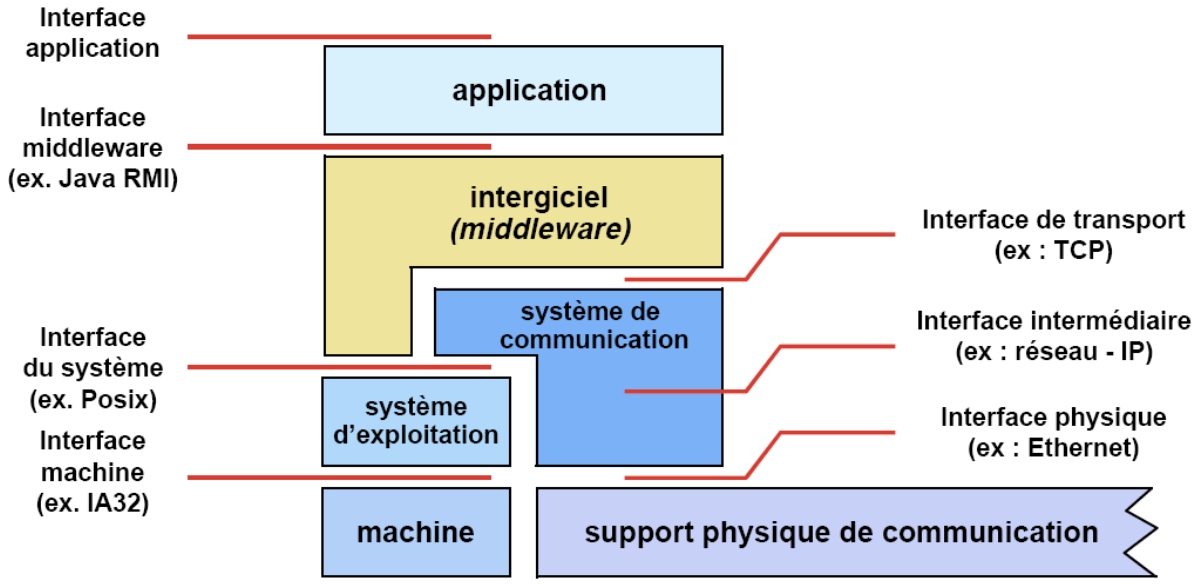
Applications Réparties

Département Sciences Informatiques
Jean-Yves Tigli – tigli@polytech.unice.fr

SI 4ème année

Applications Réparties ?

- ✓ Ensemble de processus (objets, agents, acteurs) qui:
  - Communiquent entre eux via un réseau
  - Evoluent de manière cohérente
  - Remplissent une fonction identifiable
  - Ne sont pas forcément interdépendants



Chaque interface cache les interfaces de niveau inférieur

Source: Sacha Krakowiak

Motivations et Modèles

- ✓ Une évolution logique
  - Généralisation des équipements communicants
  - Interconnexion et haut débit généralisés
  - Répond aux problématiques
    - de passage à l'échelle
    - de tolérance aux pannes
    - d'évolutivité
- ✓ Modèles de programmation:
  - Synchrone (RPC, RMI...)
  - Asynchrone (Evènements)
  - *Ressources partagées (mémoire partagée, systèmes de fichiers répartis)*
  - *Code mobile*
  - *Peer to peer*
  - ...

Programme du Cours

- ✓ Du Web aux Web Services (J.-Y. Tigli)
  - Couche Transport (exemple HTTP et Serveur Web)
  - Introduction aux Web Services
    - Généralités
    - Architecture
  - Appels de procédures distants: RPC (exemple SOAP)
  - Interface publique de service (exemple WSDL)
  - Introduction à RES
- ✓ Bus Logiciel (F. Baude)
  - Introduction au Bus Logiciel & RMI
  - RMI et Sécurité (JAAS)
  - Introduction à Corba
  - RMI Corba IIOP
- ✓ Nommage (J.-Y. Tigli, F. Baude)
  - Annuaire et JNDI
  - UDDI
- ✓ Evénements (F. Baude)
  - JMS

De Nombreux Intervenants

- ✓ Pour les cours:
  - J.-Y. Tigli
  - F. Baude
- ✓ Pour les Travaux Dirigés:
  - J.-Y. Tigli
  - Nicolas Ferry
  - Stéphane Lavirotte
  - F. Baude

Université Nice SOPHIA ANTIPOLIS

Du Web aux Web Services

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

(*) D'après les cours de Jean-Yves Tigli, Gaëtan Rey, Stéphane Lavirotte, Michel Riveill, Sacha Krakowiak, Didier Donsez, et Keith Ross
et
http://abcdrfc.free.fr/

Du Web aux Web Services

- ✓ Présentation du Cours
  - Cours 1 et 2 :
    - Couche Transport* (exemple HTTP et Serveur Web)
  - Cours 3 à 5:
    - Introduction aux Web Services
      - Généralités
      - Architecture
    - Appels de procédures distants: RPC (exemple SOAP)
    - Interface publique de service (exemple WSDL)
  - Cours 11:
    - Nommage (UDDI)
    - Evénements (Introduction UPnP)
    - Conclusion

Du Web aux Web Services

- ✓ Travaux Dirigés:
  - TD n°1 :
    - Conception d'un serveur HTTPD (liaison avec le module Internet et Réseau)
  - TD n°2 :
    - Administration HTTPD / Apache
  - TD n°3 :
    - Web Services avancés sous .Net (C#)
  - TD n°4 :
    - Interopérabilité : WS avec gSOAP (C++)
  - TD n°5 :
    - Interopérabilité : WS avec Apache Axis (Java)
  - TD n°11 :
    - UDDI sous .Net (C#)

Quelques Définitions

Le Jargon du Web

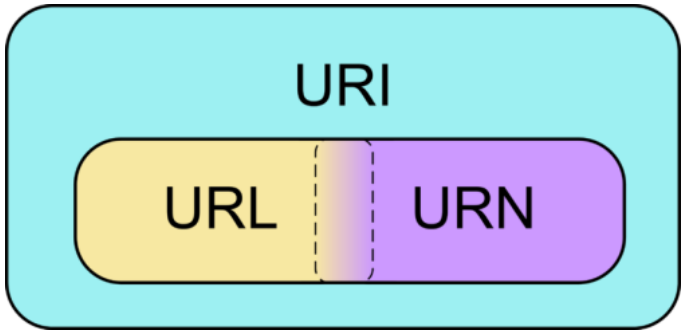
- ✓ Page Web:
  - Pointés par une URL
  - La plupart des pages WEB se composent de:
    - Une page HTML de base,
    - Différentes références à des « objets »
- ✓ L'agent utilisateur pour le Web s'appel un "browser" (butineur en français)
  - Microsoft Internet Explorer, Mozilla FireFox, Opera, Safari, Google Chrome, ...
- ✓ Un serveur pour le Web s'appelle un serveur Web :
  - Apache, Microsoft Internet Information Server (IIS), ...

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

URx: Uniform Resource ...

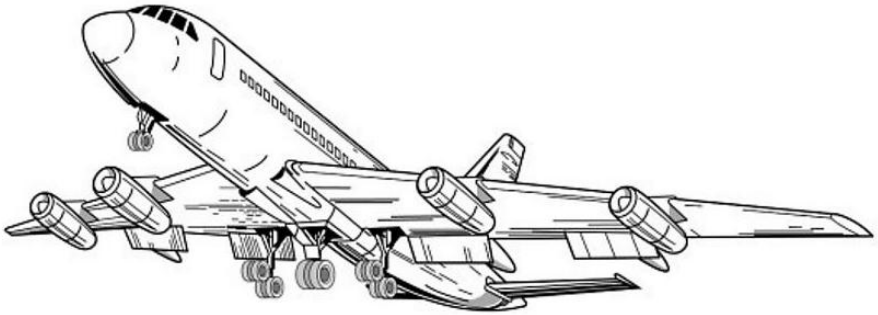
- ✓ Un URN (Uniform Resource Name)
  - Identifie une ressource par un nom dans un espace de nommage (identifie la ressource et pas sa localisation)
  - urn:NID:NSS
- ✓ Plus généralement un URI (Uniform Resource Identifier)
  - Peut être une URL ou un URN



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

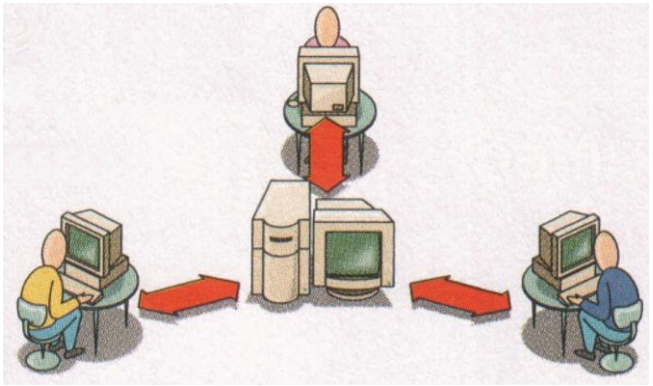
HyperText Transfer Protocol

- ✓ *HTTP : HyperText Transfert Protocole*
  - Un des protocole les plus courants sur Internet
    - Un protocole omni-présent: 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.

Un Modèle Client - Serveur

- ✓ Modèle Client / Serveur
  - client: « browser » qui demande, reçoit et affiche des documents Web.
  - server: serveur Web qui envoie des documents en réponse aux requêtes des clients.



- ✓ L'échange entre le client et le serveur se fait en mode texte.
  - Le charset généralement utilisé est l'US-ASCII sur 8 bits.
  - Il est cependant possible que cet encodage soit modifié selon le client ou le serveur.

Principe de Fonctionnement de HTTP

- ✓ HTTP : TCP 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"
  - Le serveur ne maintient pas d'information sur les requêtes passées du client.

Zoom sur un Exemple de Communication

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 (contenant l'URL) au travers le socket de communication TCP.
4. Le serveur http reçoit le message de requête, compose le message de réponse contenant les objets demandés et renvoie le message au travers le socket de communication.
5. Le client http reçoit le message de réponse contenant le fichier HTML et l'affiche.
6. Le serveur http ferme la connexion.
7. En « parsant » le fichier HTML, le client http trouve 10 références à des objets jpeg. Les étapes 1 à 6 sont répétées pour chaque référence aux 10 objets jpeg.

Persistances des Connexions

- ✓ Connexions non-persistantes
  - HTTP/1.0
  - Le serveur « parse » la requête, répond et ferme la connexion.
  - Ralentit la récupération de la page complète
  - Mais la plupart des « browsers » 1.0 utilise des connexions parallèles
- ✓ Connexions persistantes
  - Par défaut pour HTTP/1.1
  - Durant une même connexion TCP, le serveur « parse » une requête, répond puis recommence ..
  - Le client envoie des requêtes pour tous les objets référencés aussitôt qu'il reçoit la page HTML de base.
  - Accélère la récupération de la page complète

En résumé

- ✓ Le fonctionnement de HTTP est très simple pour HTTP/1.0
  - connexion
  - demande (GET) d'un document
  - renvoi du document (status=200) ou d'une erreur
  - Déconnexion
- ✓ Cependant
  - Dialogue plus complexe en cas d'authentification
  - Optimisation : une série de plusieurs requêtes sur une connexion [Connexion « KeepAlive » de HTTP/1.1 (RFC 2068)]

Dialogue HTTP

- ✓ Dialogue
  - en mode caractères ASCII (7 bits)
    - telnet www.sun.com 80
- ✓ Types de dialogue
  - 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)

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 désassociassion) 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

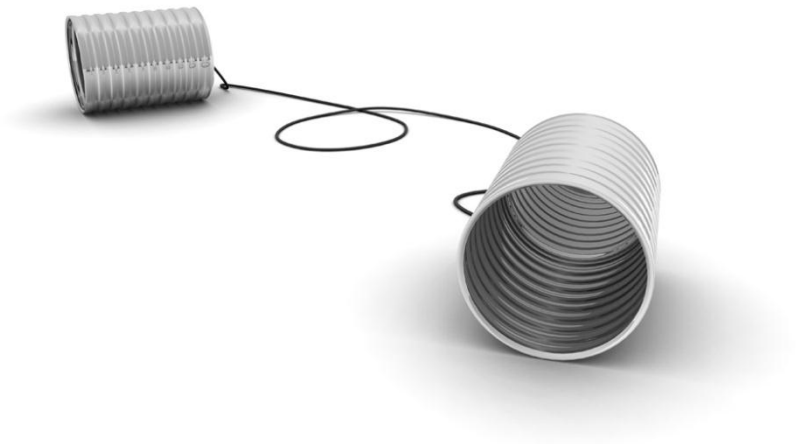
Entêtes d'Entité (3)

- ✓ 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
- ✓ ...

Internationalisation

- ✓ Langage Accepté
  - fr, de, it, en, sq (albanais), ru, (russe), ja (japonais),zh (chinois), el (grec), he (hébreu), ca (catalan) ...

- ✓ Charset (table de caractère)
  - par défaut ISO-8859-1 (Latin-1)
    - ISO-8859-2 (hongrois, albanais, ...)
    - ISO-8859- 4
    - ISO-8859-5, KOI8-R (russe, bulgare, polonais)
    - ISO-8859-7 (grec)
    - ISO-8859-8 (hébreu)
    - ISO-8859-9 (turc)
      Shift_JIS, ISO-2022-JP, EUC-JP (japonais
    - Big5 (chinois simplifié)
    - GB2312(chinois traditionnel - Taiwan)

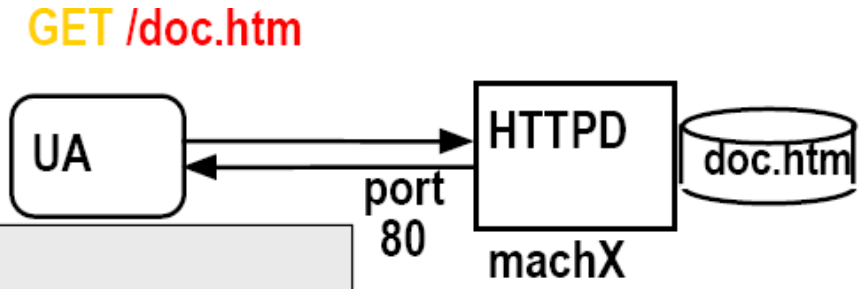


Echange de Documents

Réception et Envoi de Données

Récupération d'un Document Méthode GET

- ✓ GET /fichier



le Client envoie

GET /doc.htm HTTP/1.0 *méthode,chemin,version*
Accept: www/source *documents acceptés*
Accept: text/html
Accept: image/gif
User-Agent: Lynx/2.2 libwww/2.14
From: alice@pays.merveilles.net
* une ligne blanche *

le Serveur répond

HTTP/1.0 200 OK *ligne de status*
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 *type du document retourné*
Content-length: 2345 *sa taille*
* une ligne blanche *
<HTML><HEAD><TITLE> ...

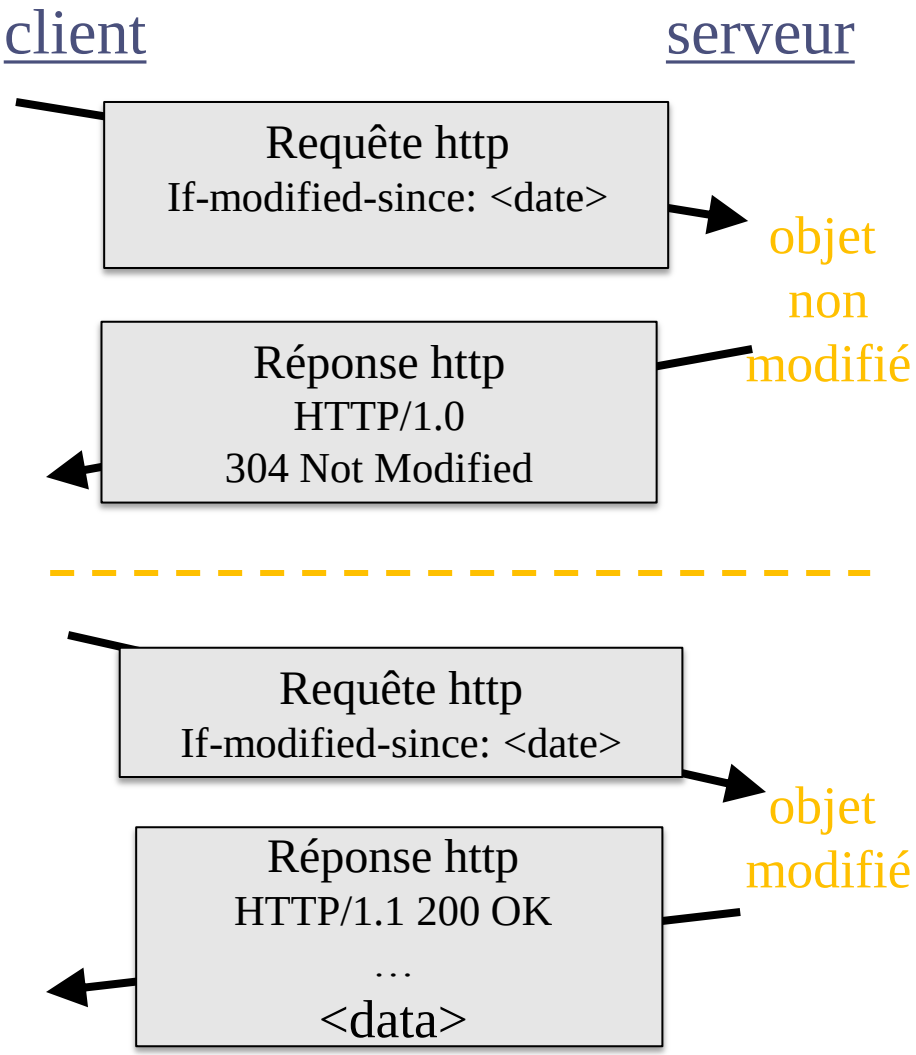
Récupération
Méthode GET conditionnelle

- ✓ **Objectif** : ne pas envoyer d'objet si le client à une version chargée à jour (en cache).
- ✓ **Client**: spécifie la date de la copie en cache dans la requête :

```
If-modified-since: <date>
```

- ✓ **Serveur**: la réponse ne contient pas de données si l'objet est à jour :

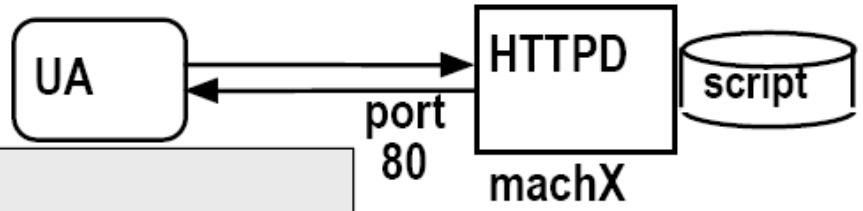
```
HTTP/1.0 304 Not Modified
```



Soumission d'un Formulaire Méthode GET

- ✓ GET/script?name1=value1&name2=value2

GET /script?name1=value1&name2=value2



le Client envoie

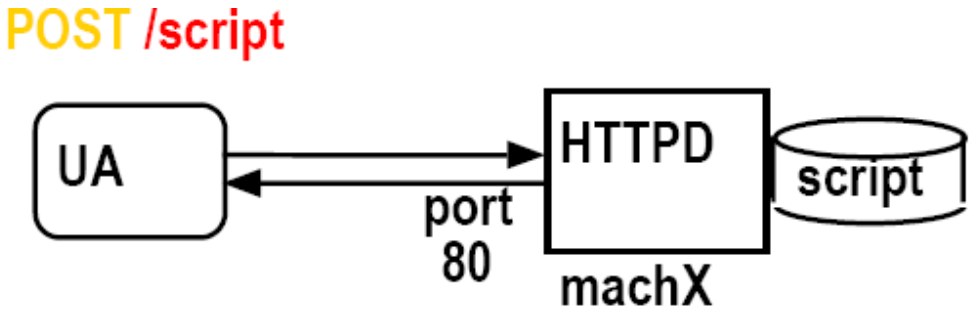
GET /script?name1=value1&name2=value2 HTTP/1.0
Accept: www/source *documents acceptés*
Accept: text/html
Accept: image/gif
User-Agent: Lynx/2.2 libwww/2.14
From: alice@pays.merveilles.net
* une ligne blanche *

le Serveur répond

HTTP/1.0 200 OK *ligne de status*
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 *type du document retourné*
Content-length: 2345 *sa taille*
* une ligne blanche *
<HTML><HEAD><TITLE> ...

Soumission d 'un Formulaire méthode POST

- ✓ POST /script



le Client envoie

```
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
    * une ligne blanche *
name1=value1&
name2=value2
```

le Serveur répond

```
HTTP/1.0 200 OK
…
Content-length: 2345
    * une ligne blanche *
<HTML><HEAD><TITLE> ...
```

Codage des « paramètres »

- ✓ Les valeurs passées (URL et contenu des entrées des formulaires) doivent être sur 7 bits et sans caractères spéciaux

- ✓ Format d'encodage : x-www-form-urlencoded
  - Espace ⇒ « + »
  - Tous les caractères spéciaux et accentués ⇒ % code ascii
    - @ %40
    - é %e9
  - Les entrées des formulaires sont encodés dans une chaîne composée de paires (nom de l'entrée)=(valeur de l'entrée) séparé par des &

- ✓ nom=Dupont+Jean&adresse=3+rue+de+la+Gait%e9%0a75014+Paris

Comportement du Client / type du document retourné

- ✓ A partir du type MIME de Content-Type
  - Visualisation native
    - la fonction de visualisation est dans le noyau (core) du client
      - `text/html, image/jpeg`
  - Visualisation par plugin
    - la fonction est présente dans une DLL, un SO ou un JAR
    - elle est liée dynamiquement pour réaliser la visualisation
      - `world/vrml, text/tex`
  - Visualisation externe
    - la fonction n'est pas présente dans le client
    - le client rapporte le document et le sauvegarde dans un fichier temporaire
      - `video/mpeg, application/postscript`

Requête Multi-parties (multipart)

- ✓ **Motivation**
  - Requête multi-document [RFC1867]
  - formulaire HTML contenant des Upload de fichiers

```
<FORM ACTION="/servlet/UploadTest" ENCTYPE="multipart/form-data"
METHOD=POST>
Your name? <INPUT TYPE=TEXT NAME=submitter> <BR>
Your first file to upload? <INPUT TYPE=FILE NAME=file1> <BR>
Your second file to upload? <INPUT TYPE=FILE NAME=file2> <BR>
<INPUT TYPE=SUBMIT>
</FORM>
```

  - Remarque : Mail multi-documents
    - (fichiers attachés, mail enrichi d'images, audio-mail ...)

Requête Multi-parties (multipart) Codage de la requête

```
Content-Type : multipart/form;boundary=End9989822
  --End9989822
Content-Disposition; form-data;name="file1";
  filename="test.htm"
  Content-Type : text/html
<HTML><BODY> Ceci est un fichier de
  test !</BODY></HTML>
--End9989822
Content-Disposition; form-data; name="file2";
  filename="test2.txt"
  Content-Type : text/plain
Ceci est un deuxième fichier de test !
  --End9989822
```

Réponse Multi-parties

- ✓ Codage
  - Content-Type : multipart/x-mixed-replace;
  - Frontière entre les parties
    - Déclaration : boundary=chaîne_aléatoire
    - Séparateur : –chaîne_aléatoire
- ✓ Comportement
  - le navigateur affiche le sous-document suivant dès qu ′il commence à le recevoir après avoir effacer la fenêtre.

Réponse Multi-parties Exemple

```
Content-Type : multipart/x-mixed-replace;
        boundary=End6557756567900183 8

--End6557756567900183 8
```

Le serveur définit une chaîne séparateur des documents

```
Content-Type : text/html

<HTML><BODY><H1>Trois … </H1><BODY></HTML>
--End6557756567900183 8
```

Le serveur attend 1 seconde avant de renvoyer la suite : le client affiche « Trois… »

```
Content-Type : text/html

<HTML><BODY><H1>Deux … </H1><BODY></HTML>
--End6557756567900183 8
```

Le serveur attend 1 seconde avant de renvoyer la suite : le client affiche « Deux… »

```
Content-Type : text/html

<HTML><BODY><H1>Un … </H1><BODY></HTML>
--End6557756567900183 8
```

Le serveur attend 1 seconde avant de renvoyer la suite : le client affiche « Un…»

```
Content-Type : text/html

<HTML><BODY><H1>Partez ! </H1><BODY></HTML>
--End6557756567900183 8
```

Le serveur clôt la connexion TCP/IP avant de renvoyer la suite :
le client affiche finalement « Partez !»

Sessions avec HTTP

Comment réaliser le suivi de sessions ?

Suivi de Sessions avec HTTP (1)

- ✓ Motivations :
  - La notion de session est importante dans une application conversationnelle
    - commerce électronique
    - « j ´ajoute ce produit à mon panier (existant)»
  - Cependant HTTP est un protocole «stateless»
    - le serveur ne maintient pas d ´informations liées aux requêtes précédentes d ´un même client.
    - HTTP est donc « sessionless »
  - Comment implanter la notion de session sur plusieurs requêtes HTTP
    - documents, CGI, Servlet, ASP

Suivi de Sessions avec HTTP (2)

- ✓ Méthodes
  - Le serveur génère un identificateur de session et associe un état (et une date limite de validité) à une session
  - Le client renvoie l 'identificateur de session à chaque requête HTTP vers le serveur

- ✓ Echange et Stockage de l'identificateur de session
  - Input HIDDEN dans les formulaires
  - Réécriture des URLs EXTRA_PATH
  - Cookies (désactivable)
  - Identificateur de session SSL (Secure Socket Layer)

Suivi de Sessions avec HTTP (3)

- ✓ Une session s'étend sur plusieurs requêtes
  - documents, CGI, SSS, Servlet, ASP
  - le serveur maintient un contexte de session et y associe un identifiant de session
- ✓ 3 solutions de suivi
  - input HIDDEN
    - contient l 'identifiant de la session
  - la Ré-écriture d 'URL
    - l 'identifiant dans chaque URL (dans les documents)
  - les Cookies
    - information positionnée par le serveur sur le client la durée de vie du cookie dépasse la session
    - puis envoyée par le client à chaque requête

Suivi de Session une entrée HIDDEN par formulaire

- ✓ Chaque réponse retournée par le serveur est un formulaire qui contient un identifiant caché dans une entrée HIDDEN
- ✓ Exemple
  - page de proposition

```
<FORM METHOD="POST" ACTION="/cgi-bin/command">
<INPUT TYPE="ckeckbox" NAME="art12387">
Chaussures
...
</FORM>
```

  - réponse de /cgi -bin/command

```
<FORM METHOD="POST" ACTION="/cgi-bin/envoi">
<INPUT TYPE="hidden" NAME="TransID" VALUE="54109848932">
Nom: <INPUT TYPE="text" NAME="nom">
Adresse: <INPUT TYPE="text" NAME="adresse">
N° de Carte de Credit: <INPUT TYPE="text" NAME="numcarte">...
<INPUT TYPE="hidden" NAME="Language" VALUE="French">
</FORM>
```

Suivi de Session une entrée HIDDEN par formulaire

- ✓ Inconvénients
  - Dialogue uniquement par formulaire
    - car pas de persistance de l'identifiant côté client
  - Ambiguïté dans le cas des retours-arrière de l ′utilisateur
    - annulation d'une série d ′actions
    - ou série d'actions supplémentaires

Suivi de Session
la ré-écriture des URLs

- ✓ L ' identifiant de sessions est encodé dans les URLs des documents HTML retournés par le serveur.
  - Dans le PATH
    - http://www.mycomp.com/cgi-bin/envoi?name=toto
  - devient
    - http://www.mycomp.com/182993954/cgi-bin/envoi?name=toto
  - Dans l 'EXTRA-PATH
    - http://www.mycomp.com/cgi-bin/ envoi?name=toto
- ✓ devient
  - http://www.mycomp.com/cgi-bin/envoi/sid$182993954?name=toto

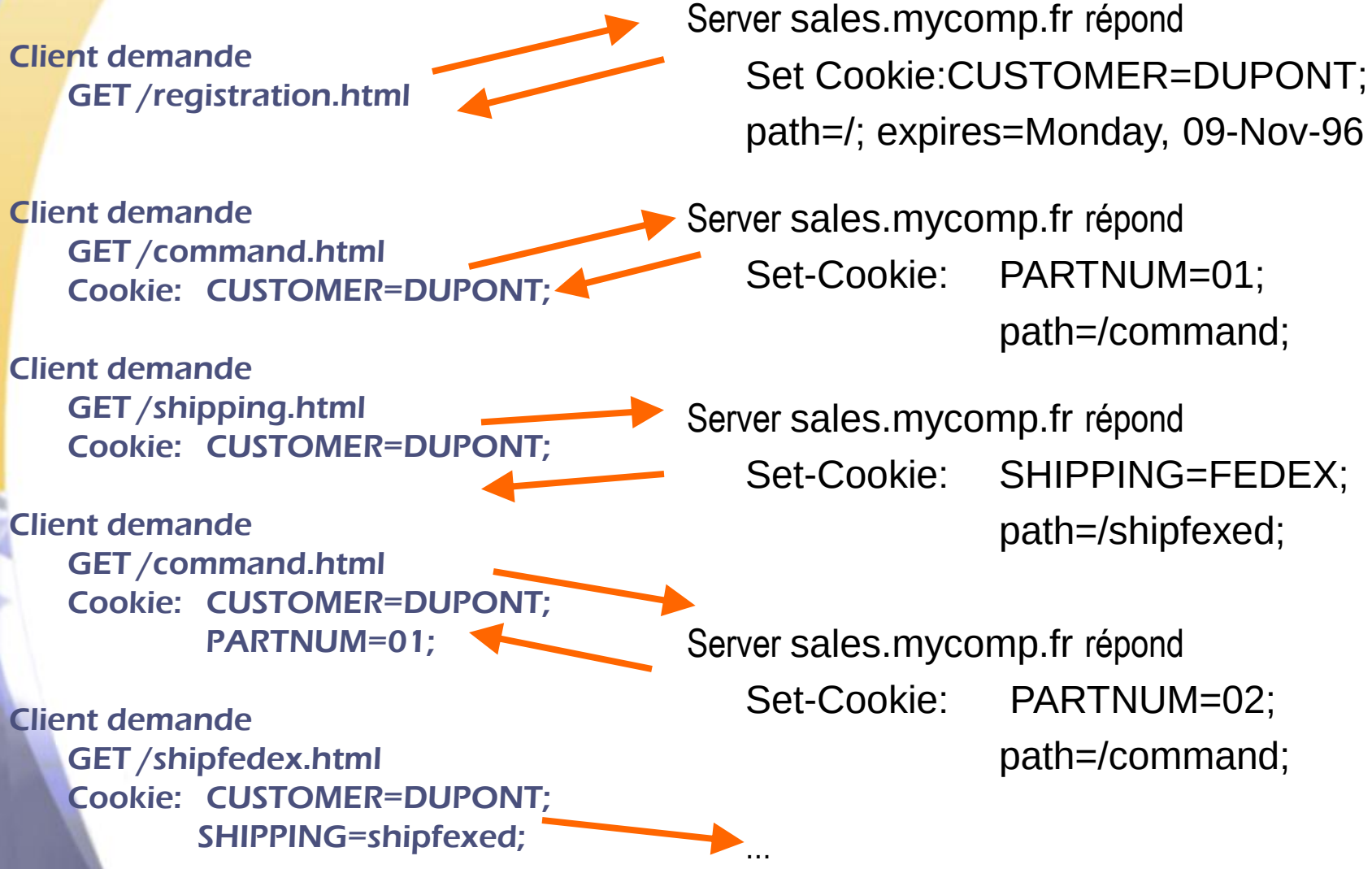
Suivi de Session
la ré-écriture des URLs

- ✓ Limites
  - URL générée par un script
    - (=> programmation)
  - ou parsing des documents HTML retournés
    - mais disfonctionnement en présence de scripts JavaScript ou VBScript générant eux aussi des URL !

Suivi de Session
les Cookies [Netscape puis RFC2109]

- ✓ Chaîne décrivant l'état d'une session
  - NAME=VALUE;
  - expires=DATE;
  - path=PATH_HEAD; /<</foo<</foobar ou /foo/bar.html
  - domain=DOMAIN_TAIL; fr<<mycomp.fr << sales.mycomp.fr
- ✓ Stocké sur le client
  - Limite :
    - 300 cookies simultanées par client, 20 cookies par serveur ou domaine, 4Ko par cookie (limite la taille des VALUEs)
- ✓ Communiqué dans les entêtes de requêtes et dans les entêtes des réponses HTTP
- ✓ Accessible par les scripts JavaScript dans une page HTML

Positionnement des Cookies



L'évolution des Cookies (1)

- ✓ Les cookies menacent la vie privée (privacy) des cybernautes bien qu'ils soient très utiles
- ✓ Les navigateurs peuvent désactiver les cookies
- ✓ Un remplaçant : P3P (Platform for Privacy Preferences) en vue d'un accord juridique entre le client et le site sur
  - la définition du champs des divulgation
    - ex : nom, prénom, adresse mais pas l'âge ou le nombre d 'enfants
  - la définition de l'utilisation de ces données par le propriétaire du site
    - ex : cession des informations à des tiers
  - la définition de la procédure de modification des données ultérieurement
    - ex: je me suis marié

L'évolution des Cookies (2)

- TUID/PUID Temporary et Pairwise Unique ID
  - identifiants de session (et multi-sessions) sans information attachée
- P3P exprimé en RDF/XML, Certificats / Signatures

Authentification avec HTTP

L'authentification dans HTTP

- ✓ Indiqué dans les ACL
- ✓ Modes d'authentification
  - BASIC
    - nom d 'utilisateur et mot de passe échangé en clair (base64) !
    - base des mots de passe dans un fichier htpasswd utilitaires de gestion du fichier
  - DIGEST
    - sécurisation de BASIC
    - hachage sécurisé MD5 du (nom,password,URI, méthode,nombre aléatoire fourni par le serveur)
  - SSL
    - Secure Socket Layer (TLS : Transport Layer Security)
    - authentification avec CA du serveur (2.0) et du client (3.0)
    - confidentialité avec DES
    - puis dialogue HTTP sur la connexion SSL

L'authentification applicative

- ✓ Motivations
  - interface de login
  - identification externe
    - BD, Annuaire LDAP, ...
  - authentification plus forte
- ✓ L'application gère l'authentification de l'usager
  - formulaire d 'accueil HTML (nom, password)
    - attention le mot de passe peut-être est en clair
  - gestion des tables d 'usager
  - une session est ensuite ouverte associé à un usager authentifié (ou non : par exemple rejet au bout de 3 tentatives)

Contrôle d'Accès dans HTTP

- ✓ ACL (Access Control List)
  - spécifie les autorisations (ALLOW) ou les interdictions (DENY) d 'accès à une arborescence virtuelle du serveur
  - en fonction :
    - de l 'authentification
    - de la localisation du client sous domaine DNS, réseau ou adresse IP
- ✓ ACF (Access Control File)
  - fichier regroupant les ACL
    - global : access.conf dans Apache
    - par arborescence : .htaccess
  - combinaison des ALLOW et des DENY

Audit des Requêtes

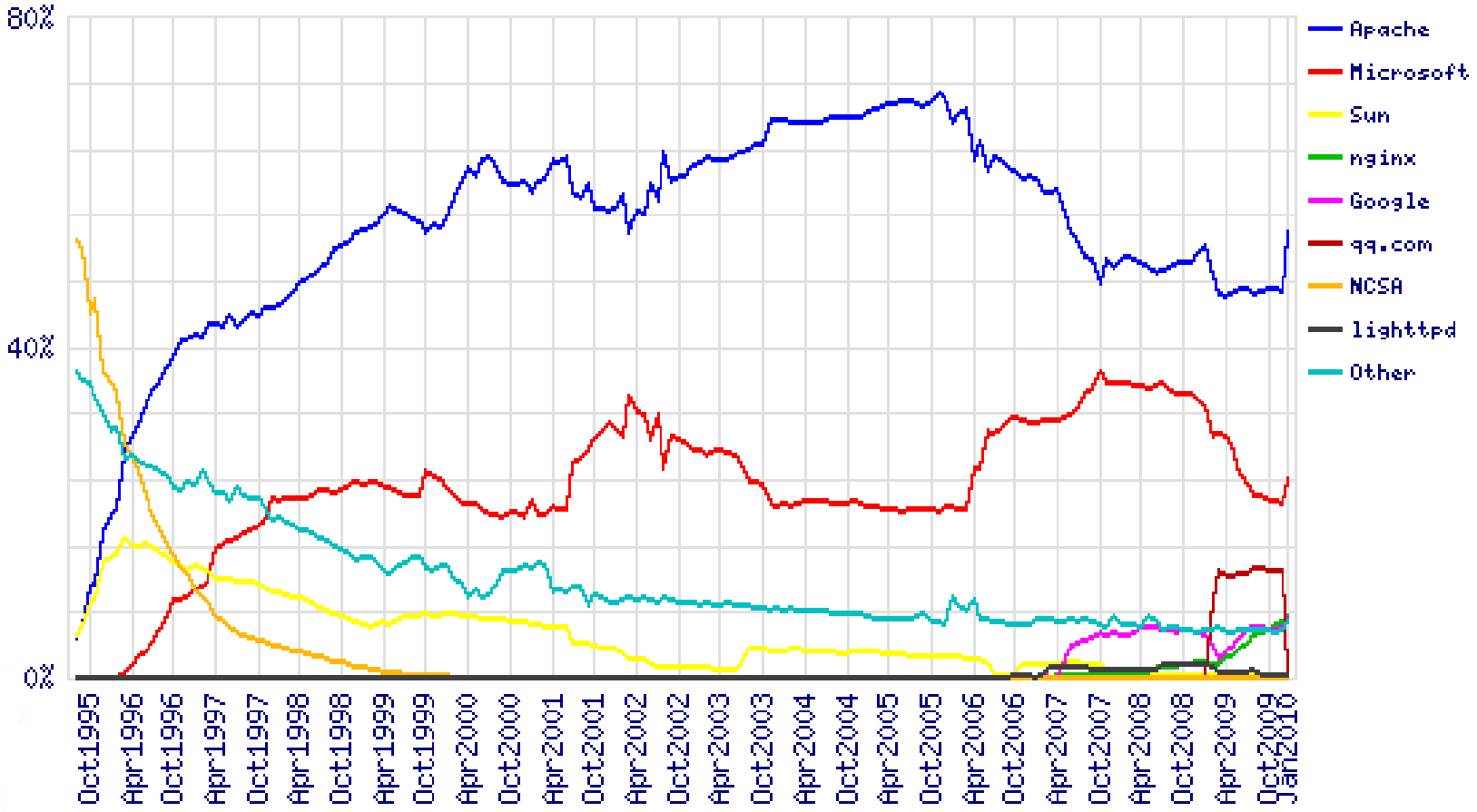
- ✓ Journaux des requêtes
  - les accès (access.log, refferee.log), et les erreurs (error.log),... sont journalisés
- ✓ Exploitation des Journaux
  - erreur dans les liens, ...
  - clientèle, analyse d'activité, ...
- ✓ Reporting (Présentation Synthétique)
  - Pour Apache
    - AccessWatch, Wusage,Analog, wwwstat
  - IIS, NS
    - intégré et visualisé par un script
  - Généraux
    - Net Analysis (Net Genesis), Enterprise Suite (Web Trends)

Infrastructure HTTP

Les Serveurs du Marché

- ✓ Offre très large
  - Apache HTTPD
  - Netscape Entreprise Server
  - Microsoft Internet Information Server
  - W3C Jigsaw
  - Sun JavaServer
  - Oracle Web Server
  - IBM Web Sphere
  - ...etc...
- ✓ Fonctionnalités supplémentaires
  - gestion des sessions, des transactions, ...
  - accès aux serveurs d 'applications, ...

Répartition part de Marché Serveurs Web



Source: http://news.netcraft.com/

Apache

- A patch of NCSA HTTPD
  - serveur le plus répandu (« toujours » plus de 50% de 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, ...
  - srm.conf
    - traitement des ressources locales lors des requêtes index, script, répertoire, AddType, AddIcon, Alias, DocumentRoot
  - access.conf
    - contrôle d 'accès global (ACF : access config file)
  - mime.types
    - table de correspondance suffixe fichier -> type MIME document
- ✓ Outil
  - GUI : Vision (focus-array.com)
  - ...

2. 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