Introduction à la programmation web

Philippe Rigaux

Objectifs

- Un panorama des techniques de programmation de sites web
  - L'Internet : architecture, adressage, serveurs et clients web
  - URL, HTTP, HTML, CSS !!
  - Programmation CGI (C, PHP)
- C'est une présentation d'ensemble ! Tout reste à approfondir ensuite.

Pour tout savoir !

- Le site du consortium WWW: *www.w3c.org*
  - Les documents de référence sur HTTP, HTML, CSS, XML, XHTML, DHTML, et plein d'autres gros mots.
  - Publiés sous la forme de *Request for Comments* (RFC). Certains ont acquis le statut de norme (ex HTML 4.0)
  - Ca bouge tout le temps: à consulter régulièrement

Pour en savoir **encore** plus

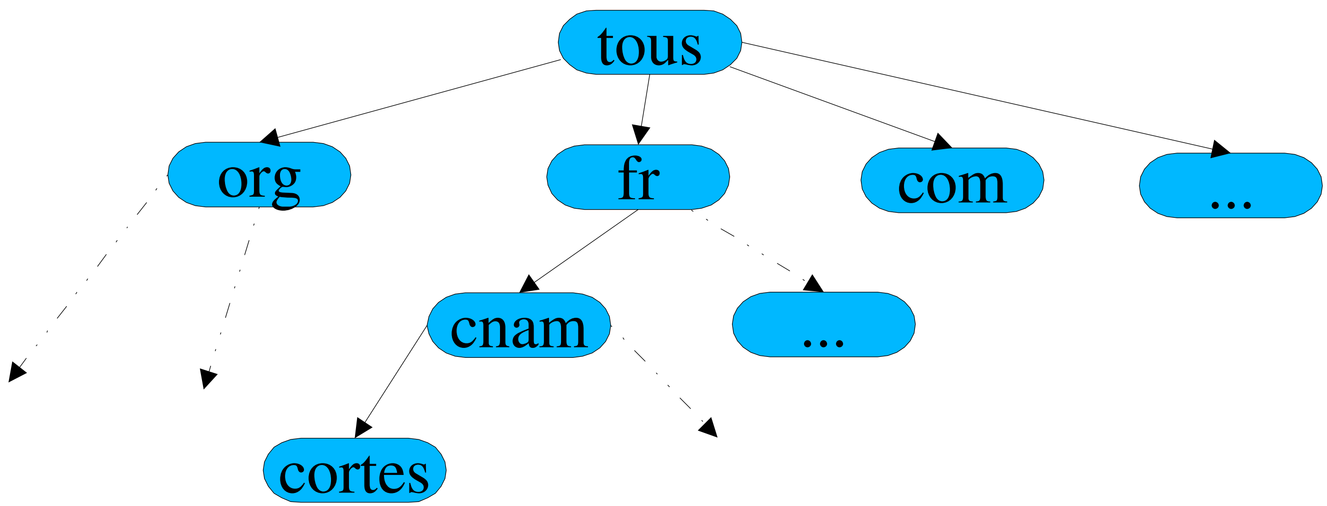
- Beaucoup (trop) de livres, la plupart **très** mauvais.
  - Les Editions O'Reilly font du travail sérieux (Apache, JavaScript, HTML, PHP/MySQL :–)))
  - Livre HTML/Javascript chez Eyrolles, de Chaléat et Charnay
  - Intranet et Internet sous Linux, chez Eyrolles, de Holtz *et al*
- Sinon on trouve tout et –surtout – n'importe quoi sur le Web.

L'Internet

- C'est un réseau !
  - Basé sur un protocole, TCP/IP
  - Permet d'échanger des données dans un environnement hétérogène
  - Chaque ordinateur est connu par une adresse IP unique (IPv6: 16 octets)
  - On peut aussi y faire référence par son nom
    - Ex: cortes.cnam.fr = 163.173.212.26

Correspondance noms/adresses: le *Domain Name Service*

- Jusqu'en 1984, tout était centralisé dans un seul fichier `hosts.txt` !
- Remplacé par le DNS: une organisation hiérarchique des noms de domaine



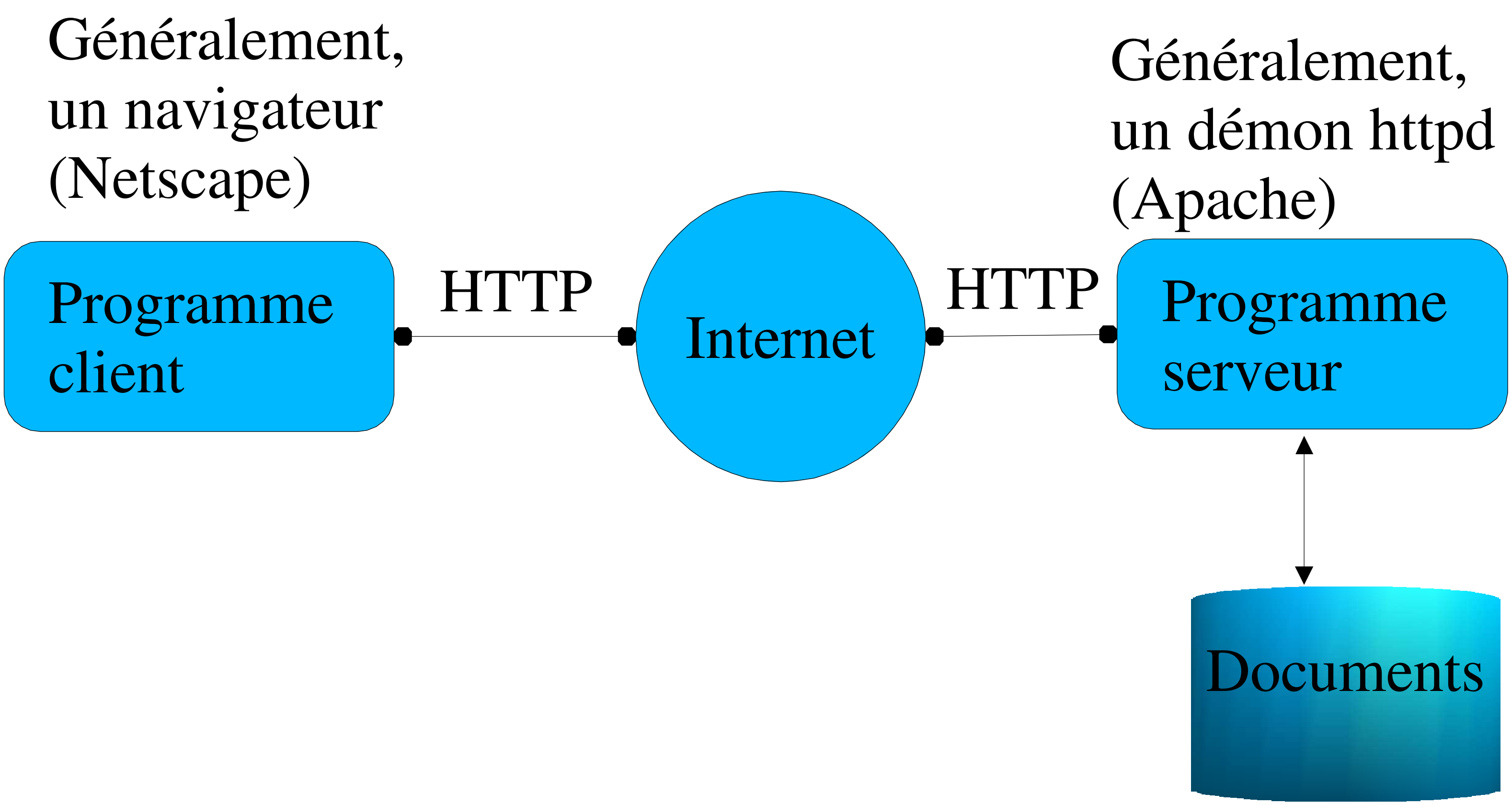
Fonctionnement du DNS

- La gestion du DNS est décentralisée: un sous–domaine est géré par un serveur de nom
  - –> pour le sous–domaine *cnam.fr*, c'est *asimov*, 163.173.128.6
- Les serveurs de nom collaborent entre eux pour convertir un nom de machine en adresse IP.

Le Web

- C’est un système d’information basé sur l’Internet
  - Ce SI est constitué de « documents », terme à prendre au sens large
  - Ces documents sont fournis par des serveurs web
  - Il sont consultés par l’intermédiaire de clients web

Fonctionnement du Web



Le programme client

- Il dialogue avec le serveur
  - En lui demandant des documents
  - En lui fournissant des informations (paramètres, fichiers)
- Il communique les documents à l’utilisateur (textes, images, sons)
  - Le programme client est en charge du rendu des documents

Le programme serveur

- En général c’est un démon (Unix) ou un service (NT)
  - Il est en écoute permanente sur un port réseau –> en général le port 80
  - Il reçoit, analyse les requêtes de clients web
  - Il transmet en retour un document résultat de la requête.

Principaux composants du Web

- Le protocole HTTP
  - *Hypertext Transport Protocol*
  - Protocole d'échange des documents web
- Le langage HTML
  - *Hypertext Markup Language*
  - Langage de formatage des documents web
- Les adresses URL
  - *Uniform Resource Locator*
  - Système d'adressage des documents

HTTP: demandes du client

- Requêtes du programme client:
  - GET –> demande de lecture d'un document
  - HEAD –> demande de lecture d'un en–tête
  - POST –> envoie un document au serveur
- Exemples :
  - `HEAD /index.html HTTP/1.0`
  - `GET /ITCE/index.html HTTP/1.0`

HTTP: réponse du serveur

- Le serveur renvoie un document avec un en–tête décrivant son contenu:
  - `Date` : date de la connexion
  - `Server` : nom du serveur
  - `Content-type` : type du contenu
    - text/plain, text/html, image/gif, etc
  - `X-***` : information non normalisée
  - etc ...

Adressage avec les URL

- Syntaxe générale:

  p://ms[:port]/chemin/document?args

  - *p* est le protocole: `http`, ou autre
  - *ms* est le nom ou l'adresse IP de la machine serveur
  - *port* est le port d'écoute
  - *chemin* est l'arborescence des répertoires pour accéder au document
  - *document* est le nom du document
  - *args* est une liste de paramètres

Exemples d’URL

- *http://cortes:8080/ITCE/EXEMPLES/Ex1.html*
  - Le port est 8080
  - Le document est dans le sous–répertoire HTML du répertoire racine
    - (ex.: */usr/local/apache/htdocs*)
  - Le serveur prend le document, et le transmet
- *http://localhost/FILMS/film.php?titre=Seven*
  - Plus sophistiqué: le document est généré dynamiquement

Documents hypertextes: HTML

- Un langage de présentation de documents
  - => des balises intégrées dans le texte indiquent le formatage souhaité
  - => c’est le navigateur qui détermine la présentation finale
- Les ancres
  - Certaines parties du texte donnent accès à d’autres documents
  - Navigation *vs* lecture séquentielle

Un premier document HTML

```
<HTML><HEAD>
<TITLE>Un exemple de document HTML</TITLE>
</HEAD>
<BODY BGCOLOR="white">

<!-- Ceci est un commentaire -->

<CENTER><H1>Mon premier document HTML</H1></CENTER>

Je tape du texte

<B>n'importe comment</B>
</BODY></HTML>
```

Les balises HTML

- Les conteneurs
  - => appliquent un formatage à tout le texte compris entre <*balise*> et </*balise*>
  - Ex: `<B>Ceci est important</B>` met en caractères gras
- Les éléments vides
  - => Insèrent un élément de présentation
  - Ex.! `<HR>` trace une ligne horizontale

Les ancres

- La balise *<A HREF=url>* est un conteneur qui permet de marquer les ancres
- *url* peut être un fichier local ou n'importe quelle adresse URL
  - <A HREF= 'Ex1.html'>L'autre document</A>
  - < HREF='/FILMS/Film.php?titre=Seven>..</A>
- => on peut faire beaucoup de chemin sur le Web sans savoir ce qu'est une URL

Ex2.html: document avec des ancres

```
<HTML><HEAD>
<TITLE>Document avec des ancres</TITLE>
</HEAD>
<BODY BGCOLOR="white">
<CENTER><H1>Document avec des ancres</H1></CENTER>

Une ancre vers <A HREF='Ex1.html'>le premier
document</A>
<P>
Une ancre vers
<A HREF='/FILMS/Film.php?titre=Seven'>Seven</A>

</BODY></HTML>
```

Quelques balises HTML

- *<OL>   <LI> <LI>   </OL>* => listes
- *<P>* nouveau paragraphe
- *<Hn>* (0 < *n* < 8) pour les titres
- *<TABLE>* pour des tableaux
- *<FORM>* pour des formulaires
- *<IMG SRC=url>* pour insérer une image
- *<B>, <IT>, <TT>*: gras, italique, courier

Ex3.html: tableau HTML

```
<HTML><HEAD><TITLE>Tableau HTML</TITLE></HEAD>
<BODY BGCOLOR="white">

<TABLE BORDER=4 CELLSPACING=2 CELLPADDING=2>
<CAPTION ALIGN=bottom><B>Mes films</B></CAPTION>

<TR><TH>Titre<TH>Année<TH>Nom Réalisateur</TR>
<TR><TD>Alien<TD>1979<TD>Scott
<TR><TD>Vertigo<TD>1958<TD>Hitchcock
<TR><TD>Van Gogh<TD>1991<TD>Pialat
<TR><TD>Pulp Fiction<TD>1994<TD>Tarantino
</TABLE>

</BODY></HTML>
```

Couleurs, polices, et autres attributs

- Jusqu'à HTML 3.2: on utilise les attributs des balises. Exemples :
  - `<BODY BGCOLOR='white'></BODY>`
  - `<FONT COLOR='red' FACE='time'></FONT>`
- Quelques soucis:
  - Lourd et répétitif (moins vrai si on utilise un éditeur spécialisé)
  - Contrôle incomplet sur le positionnement

Feuilles de style (CSS)

- A partir de la version 4: on peut définir des feuilles de style
  - On définit globalement des intructions de mise en forme
  - Les intructions sont rassemblées dans un document indépendant
  - Ce document peut être associé à un ou plusieurs documents HTML

Exemple d’une feuille CSS

```
/* Le fond est toujours en blanc    */
BODY {background-color: white}

/* Couleur rouge pour ancres et en-têtes */

A, H1, H2, H3 {color: #ca0000}
CAPTION {font-size:large; color: #ca0000}

/* Classes pour alterner les couleurs des
   lignes dans les tableaux */

TR.A0 {background-color:white}
TR.A1 {background-color:yellow}
```

Quelques commentaires

- Des instructions **globales**:
  - `A, H1, H2, H3 {color: #ca0000}`
  - => le style sera appliqué à toutes les balises
- Des **classes** pour raffiner les styles
  - `TR.A0 {background-color:white}`
  - => on peut appliquer `<TR CLASS=A0>`
- Les classes peuvent être génériques
  - `.attention {background-color=red}`

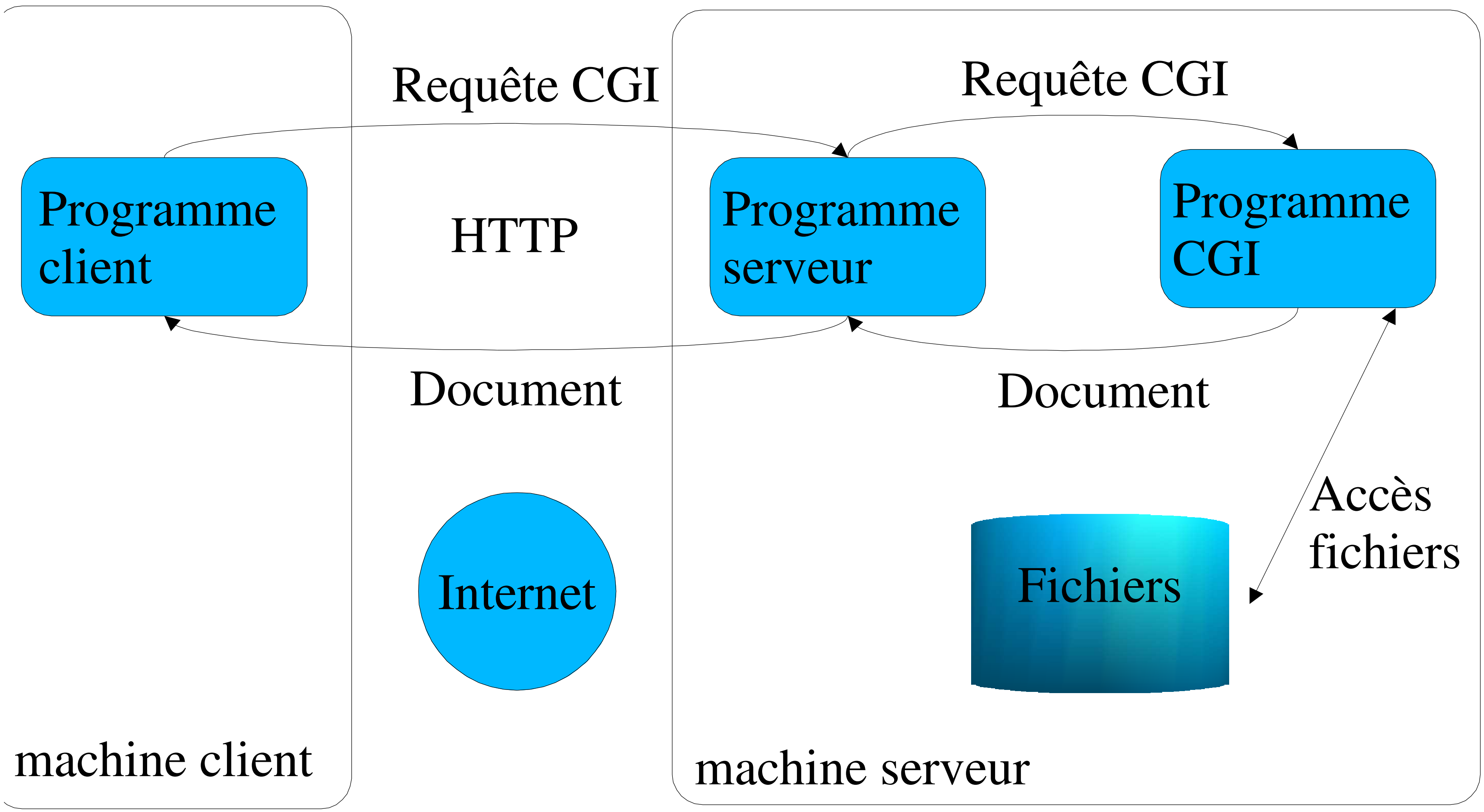
Application d’une feuille CSS

```
<HTML><HEAD><TITLE>Tableau HTML</TITLE>
<LINK REL=stylesheet HREF="ExCSS.css" TYPE="text/css">
</HEAD><BODY>

<TABLE BORDER=4 CELLSPACING=2 CELLPADDING=2>

<TR><TH>Titre<TH>Année<TH>Nom Réalisateur</TR>
<TR CLASS=A0><TD>Alien<TD>1979<TD>Scott
<TR CLASS=A1><TD>Vertigo<TD>1958<TD>Hitchcock
<TR CLASS=A0><TD>Van Gogh<TD>1991<TD>Pialat
<TR CLASS=A1><TD>Pulp Fiction<TD>1994<TD>Tarantino
</TABLE>
</BODY></HTML>
```

Programmation CGI

- Common Gateway Interface
  - Principe : on déclenche une action (un programme) sur la machine serveur
  - On peut transmettre des informations (typiquement saisies dans un formulaire)
  - Le programme CGI renvoie un document
- => le moyen le plus ancien de créer **dynamiquement** des pages HTML

Architecture du CGI



Le programme CGI

- Lecture des paramètres
  - –> transmis par le programme serveur
- Traitement de la demande
  - –> tout est permis ! Accès BD, génération d’images, ...
- Production du document
  - –> il est renvoyé au programme serveur, qui le transmet au navigateur

Transmission au serveur

- En principe à partir d'un formulaire HTML (mais pas obligatoire !)
  - Les arguments sont fournis dans une chaîne au format *nom1=val1&nom2=val2&...*
  - Deux méthodes principales:
    - GET : la chaîne est transmise dans l'URL
    - POST : la chaîne est transmise séparément
  - => POST est préférée, car elle évite les très longues URL

Actions du serveur

- Le programme serveur transmet la requête au programme CGI sous forme de variables d'environnement
  - `REQUEST_METHOD` : GET ou POST
  - `QUERY_STRING` : la chaîne des paramètres (méthode GET)
  - `CONTENT_LENGTH` : longueur de la chaîne sur l'entrée standard (méthode POST)