CSI 3540

Structures, techniques et normes du Web

Assises du Web (Partie 2)

Objectifs:

- Maîtrise des concepts sur lesquels reposent les technologies du Web, incluant HTTP
- Bien comprendre les interactions entre les serveurs (httpd) et les clients (navigateurs)

Lectures:

- Web Technologies (2007) § 1
  Pages 10 à 32

Plan

1. HTTP
2. URL
3. Encodage des caractères
4. Navigateur/serveur

HTTP

HTTP

- **Hypertext Transfer Protocol**
- Protocole de la **couche application** développé pour le Web
- Fonctionne avec tous les protocoles de transport fiable, mais en pratique il repose sur **TCP**, au **port 80** (ou 8080)
- **RFC 1945** (1996) HTTP/1.0; **RFC 2068** (1997) HTTP/1.1; **RFC 2616** (1999)

HTTP

La notion d'état (session) est construite au desssus du protocole (par les Servlets, par exemple)

- C'est un protocole **sans état** (« qui ne garde pas de trace du contexte »)
- HTTP suit un modèle **requête-réponse**
  - Le client envoie une requête
  - Le serveur envoie une réponse


École d'ingénierie et de technologie de l'information – School of Information Technology and Engineering
http://www.eiti.uottawa.ca/index.html
Google
uOttawa
L'Université canadienne
Canada's university
Université d'Ottawa • University of Ottawa
École d'ingénierie et de technologie de l'information (EITI)
School of Information Technology and Engineering (SITE)
Français
English

Requête

$ telnet www.eiti.uottawa.ca 80
Trying 137.122.89.222...
Connected to web0.site.uottawa.ca.
Escape character is '^]'.
GET /index.html HTTP/1.1
HOST: www.eiti.uottawa.ca

Réponse

HTTP/1.1 200 OK
Date: Sun, 06 Jan 2008 17:44:19 GMT
Server: Apache/2.0.59 (FreeBSD) PHP/5.1.6 (...)
Last-Modified: Fri, 28 May 2004 14:11:11 GMT
ETag: "114d1b-1a1-adf3ddc0"
Accept-Ranges: bytes
Content-Length: 417
Content-Type: text/html

<html>
<head>
  <title>ÉITI</title>
</head>
<body>
École d'ingénierie et de technologie de l'information!
</body>
</html>

HTTP 0.9 et 1.0

1. Connexion du client au serveur HTTP
2. Envoi du message HTTP requête
3. Réception du message HTTP requête
4. Fermeture de la connexion


École d'ingénierie et de technologie de l'information – School of Information Technology and Engineering
http://www.eiti.uottawa.ca/index.html
Google
uOttawa
L'Université canadienne
Canada's university
Université d'Ottawa • University of Ottawa
École d'ingénierie et de technologie de l'information (EITI)
School of Information Technology and Engineering (SITE)
Français
English

Principales étapes

1. Création d'un **message requête HTTP** à l'aide de l'URL fournie par l'usager
2. Requête au serveur de nom (DNS)
3. Établir une connexion TCP
4. Envoi du **message HTTP requête**
5. Réception du **message HTTP réponse**
6. Affichage de la réponse (HTML)

Message HTTP requête

Ligne de commande → **Commande URI Version_de_protocole**
En-tête de requête

Ligne vide →

Corps (possiblement vide) de la requête

L'En-tête HOST est nécessaire pour les versions 1.1 et plus. Facilite la mise en place de serveurs virtuels sur un même serveur HTTP.
En l'absence de cet en-tête, le serveur retourne un message réponse 400 (Bad Request).

GET /index.html HTTP/1.1
HOST: www.eiti.uottawa.ca

Message HTTP réponse

Ligne de statut → **Version Code-réponse Texte-réponse**

En-tête de réponse

Ligne vide →

Corps de réponse

```
HTTP/1.1 200 OK
Date: Sun, 06 Jan 2008 17:44:19 GMT
Server: Apache/2.0.59 (FreeBSD) PHP/5.1.6 (...)
Last-Modified: Fri, 28 May 2004 14:11:11 GMT
ETag: "114d1b-1a1-adf3ddc0"
Accept-Ranges: bytes
Content-Length: 417
Content-Type: text/html

<html>
<head>
  <title>ÉITI</title>
</head>
...
</html>
```

Commandes

- **GET** : retourne la ressource spécifiée par l'URI
- **POST** : passe le corps de la requête à la ressource spécifiée par l'URI

La majorité des requêtes sont des requêtes **GET**. **GET** et **POST** forment la quasi-totalité des requêtes.

Commandes (suite)

- **HEAD** : semblable à **GET**, mais ne retourne que l'en-tête
- **OPTIONS** : retourne les options de communications d'une ressource ou du serveur

$ telnet www.eiti.uottawa.ca 80
Trying 137.122.89.222...
Connected to web0.site.uottawa.ca.
Escape character is '^]'.
HEAD /index.html HTTP/1.1
HOST: www.eiti.uottawa.ca

HTTP/1.1 200 OK
Date: Sun, 06 Jan 2008 18:30:45 GMT
Server: Apache/2.0.59 (FreeBSD) PHP/5.1.6
Last-Modified: Fri, 28 May 2004 14:11:11 GMT
ETag: "114d1b-1a1-adf3ddc0"
Accept-Ranges: bytes
Content-Length: 417
Content-Type: text/html

$ telnet www.site.uottawa.ca 80
Trying 137.122.89.222...
Connected to web0.site.uottawa.ca.
Escape character is '^]'.
OPTIONS / HTTP/1.1
HOST: www.site.uottawa.ca

HTTP/1.1 200 OK
Date: Sun, 06 Jan 2008 17:40:36 GMT
Server: Apache/2.0.59 (FreeBSD) PHP/5.1.6
Allow: **GET,HEAD,POST,OPTIONS,TRACE**
Content-Length: 0
Content-Type: text/html

Commandes

- **CONNECT** : utilisé afin d'établir une connexion sécurisée
- **TRACE** : retourne une copie de la requête
- **PUT** : sauvegarde le corps de la requête à l'endroit désigné par l'URI
- **DELETE** : supprime la ressource désignée par l'URI

Les 2 dernières commandes sont rarement supportées par les serveurs. Ces fonctions sont implémentées par **WebDAV**.

Réponse : Code de statut

- **Niveau 100** : Information
- **Niveau 200** : Succès
  - 200 : OK
- **Niveau 300** : Redirections
  - 307 : Temporary Redirect
- **Niveau 400** : Erreur du client
  - 400 Bad Request
- **Niveau 500** : Erreur du serveur

HTTP : Versions

- **HTTP/0.9** : Première version publique
- **HTTP/1.0** : Premier RFC; En-têtes de type MIME; HEAD et POST; Requête terminée par un double retour à la ligne;
- **HTTP/1.1** : Connexions persistantes (Connection: keep-alive, close); pipelining; en-tête **Host** devient obligatoire; nouveaux en-têtes : Accept, Accept-charset, Accept-language, etc.

$ telnet www.eiti.uottawa.ca 80
Trying 137.122.89.222...
Connected to web0.site.uottawa.ca.
Escape character is '^]'.
GET / HTTP/1.1
HOST: bio.site.uottawa.ca

HTTP/1.1 200 OK
Date: Fri, 15 Jan 2010 17:06:56 GMT
Server: Apache/2.0.59 (FreeBSD) PHP/5.1.6 with Suhosin-Patch mod_ssl/
2.0.59 OpenSSL/0.9.7e-p1
Last-Modified: Sun, 10 Jan 2010 16:16:16 GMT
ETag: "4fe041b-e99-bd7df400"
Accept-Ranges: bytes
Content-Length: 3737
Content-Type: text/html

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
 <head>
  <title>Marcel Turcotte</title>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
  <link rel="stylesheet" type="text/css" href="v1/css/default.css" />
  <link rel="shortcut icon" href="v1/images/favicon.ico"/>
  <link rel="icon" href="v1/images/favicon.ico"/>
 </head>
 <body>...
</html>
```

$ telnet www.google.com 80
Trying 64.233.167.99...
Connected to www.l.google.com.
Escape character is '^]'.
GET /index.html HTTP/1.1
HOST: www.google.ca

HTTP/1.1 200 OK
Cache-Control: private
Content-Type: text/html; charset=ISO-8859-1
Set-Cookie:
PREF=ID=31becbf95e47da87:TM=1181064920:LM=1181064920:S=rVDZHAHe
YrR3WM4p; expires=Sun, 17-Jan-2038 19:14:07 GMT; path=/;
domain=.google.ca
Server: GWS/2.1
Transfer-Encoding: chunked
Date: Tue, 05 Jun 2007 17:35:20 GMT

<html>...</html>

URI

URI

- La ligne commande (première ligne de la requête) comporte 3 parties, **Commande**, **URI**, **Version du protocole**
- Signifie «Uniform Resource Identifier»
- **RFC 3986** 2005
- Syntaxe en deux parties :
  - schéma : information spécifique au schéma
  - http://www.site.uottawa.ca

Interprétation classique

- Deux types d'URIs :
  - **URL** (Uniform Resource Locator)
    - Définit **l'emplacement** d'une ressource
  - **URN** (Uniform Resource Name)
    - Définit un **nom** de ressources (indépendant de l'emplacement)
    - urn:ISBN:0-395-36341-1

Interprétation nouvelle

- Les URI définissent des espaces de noms
- “Les schémas d'identificateurs Web sont en général des schémas URI ; un schéma URI donné peut définir des sous-espaces”

URL

- L'URL identifie une ressource par **une représentation de son mécanisme d'accès**
- http et ftp utilisent l'emplacement
- Autres URLs : mail, telnet, ...
- Nous y reviendrons

URL : http

http://www.example.org:56789/a/b/c.txt?t=win&s=chess#para5

hôte (FQDN) — port — chemin — requête — fragment

authorité — URI-requête

- Le navigateur utilise la portion hôte + port (autorité) afin d'établir une connexion TCP
- L'URI-requête est utilisée afin de créer la ligne de commande
  (/ sera utilisé si URI-requête est vide)
- Le fragment n'est pas transmis au serveur (information utilisée par le navigateur)

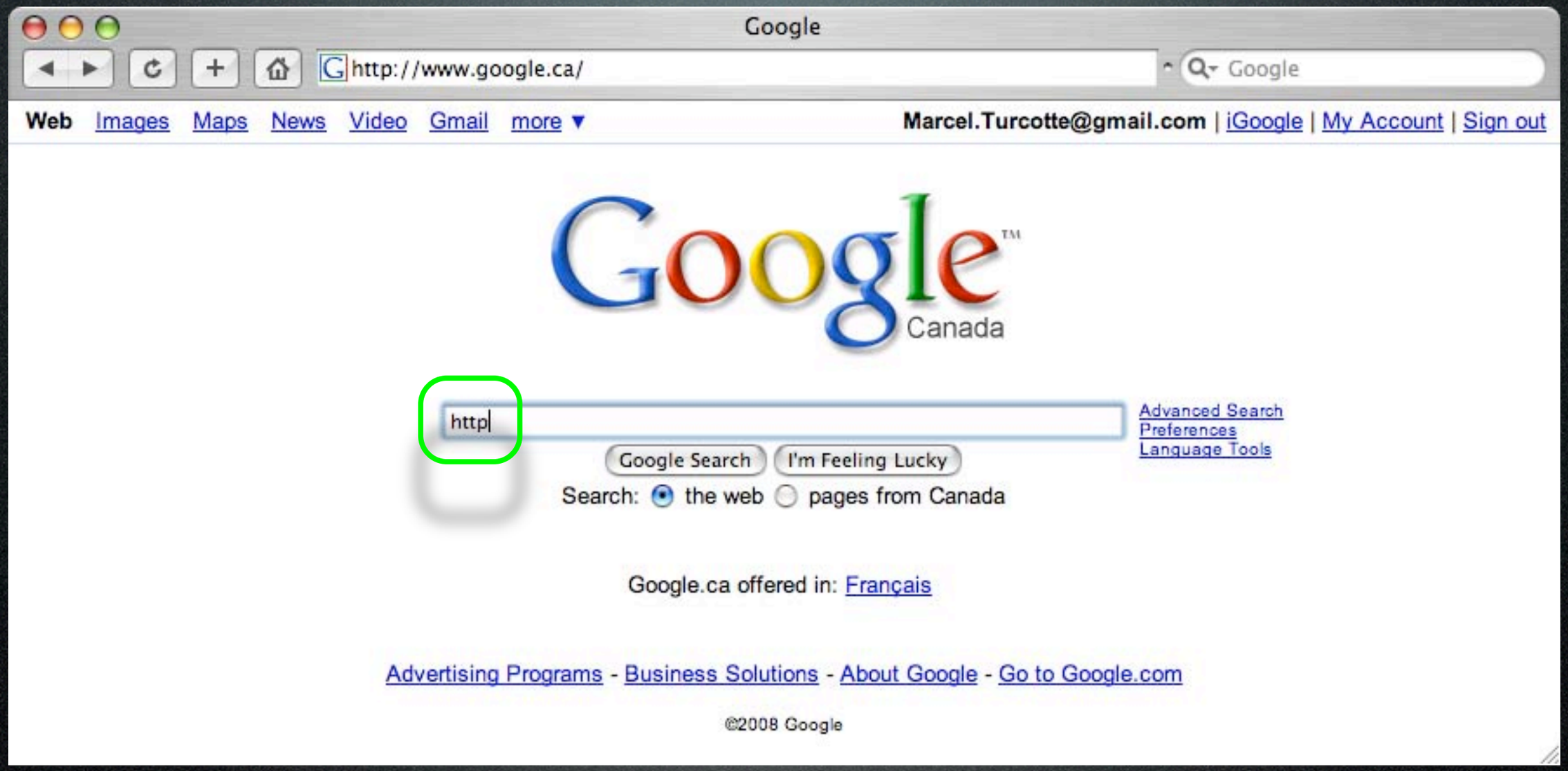
URL : http

```
$ telnet www.example.org 56789
/a/b/c.txt?t=win&s=chess
HOST: www.example.org
```

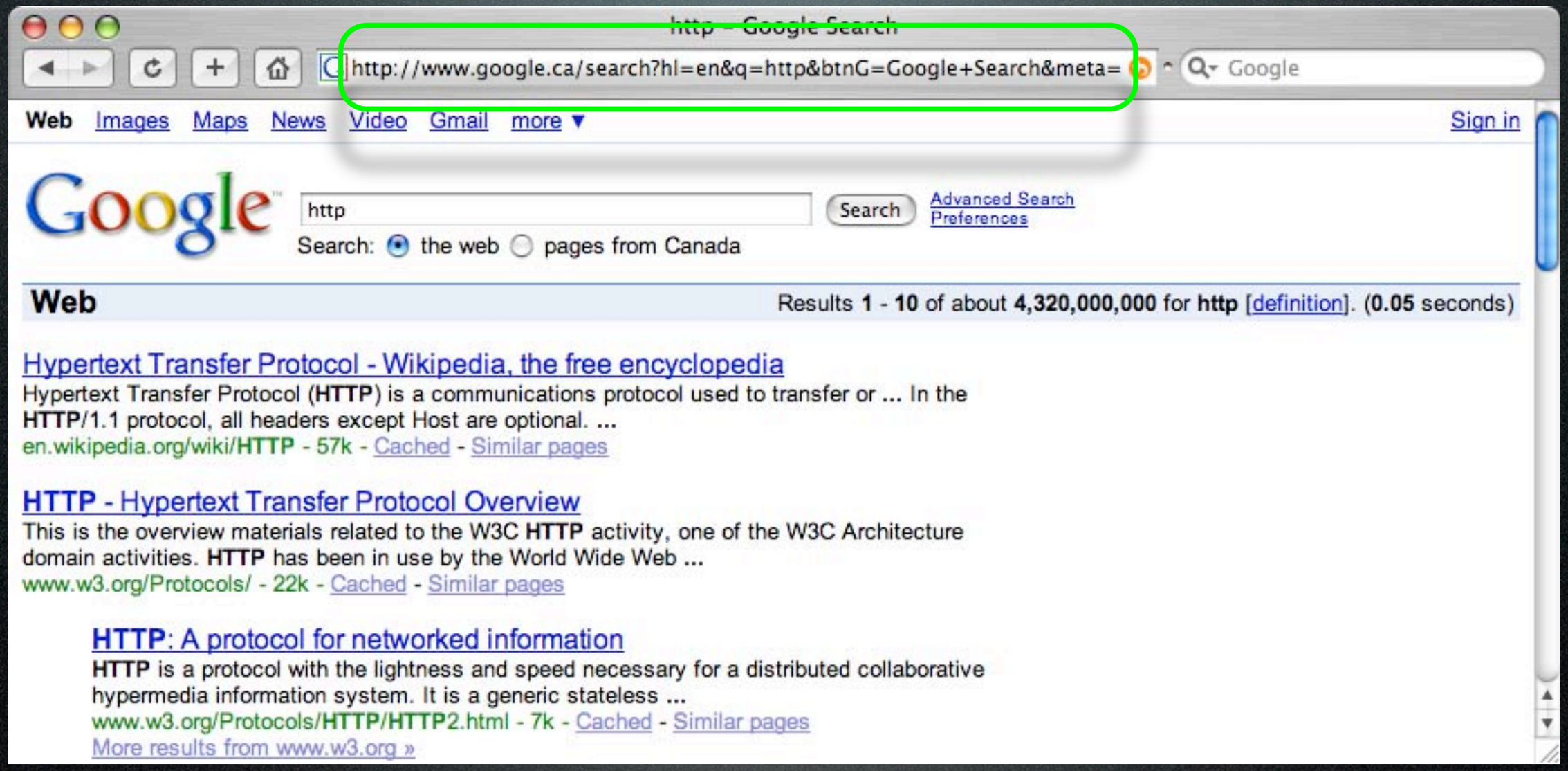
Le fragment, suffixe #para5, n'est pas transmis au au serveur, le navigateur s'en sert afin d'afficher la page de sorte que l'étiquette para5 du document HTML soit visible.

Get, Post et URI-requête

- Lorsqu'on complète un masque de saisi (formulaire), le navigateur génère soit un message **Get** ou **Post**, le choix est dicté par le concepteur du formulaire
- Bien souvent, il s'agit d'un message **Get**
- Dans ce cas, les informations sont encodées à même l'URI-requête



GET /search?hl=en&q=http&btnG=Google+Search&meta= HTTP/1.1
HOST: www.google.ca



GET /search?hl=en&q=http&btnG=Google+Search&meta= HTTP/1.1
HOST: www.google.ca

Example de Post

Exemple de Post

```
<form action="upload1.php" method="post" enctype="multipart/form-data">
 <input type="hidden" name="MAX_FILE_SIZE" value="5000">
 <input type="file" name="fichier" size="20" maxlength="80">
 <input type="submit" value="Envoyer">
</form>
```

Exemple de Post

POST /~ferment/http/prog/upload1.php HTTP/1.1
Host: www.u-picardie.fr
Content-Type: multipart/form-data; boundary=--9055292283946
Content-Length: 334

----9055292283946
Content-Disposition: form-data; name="MAX_FILE_SIZE"

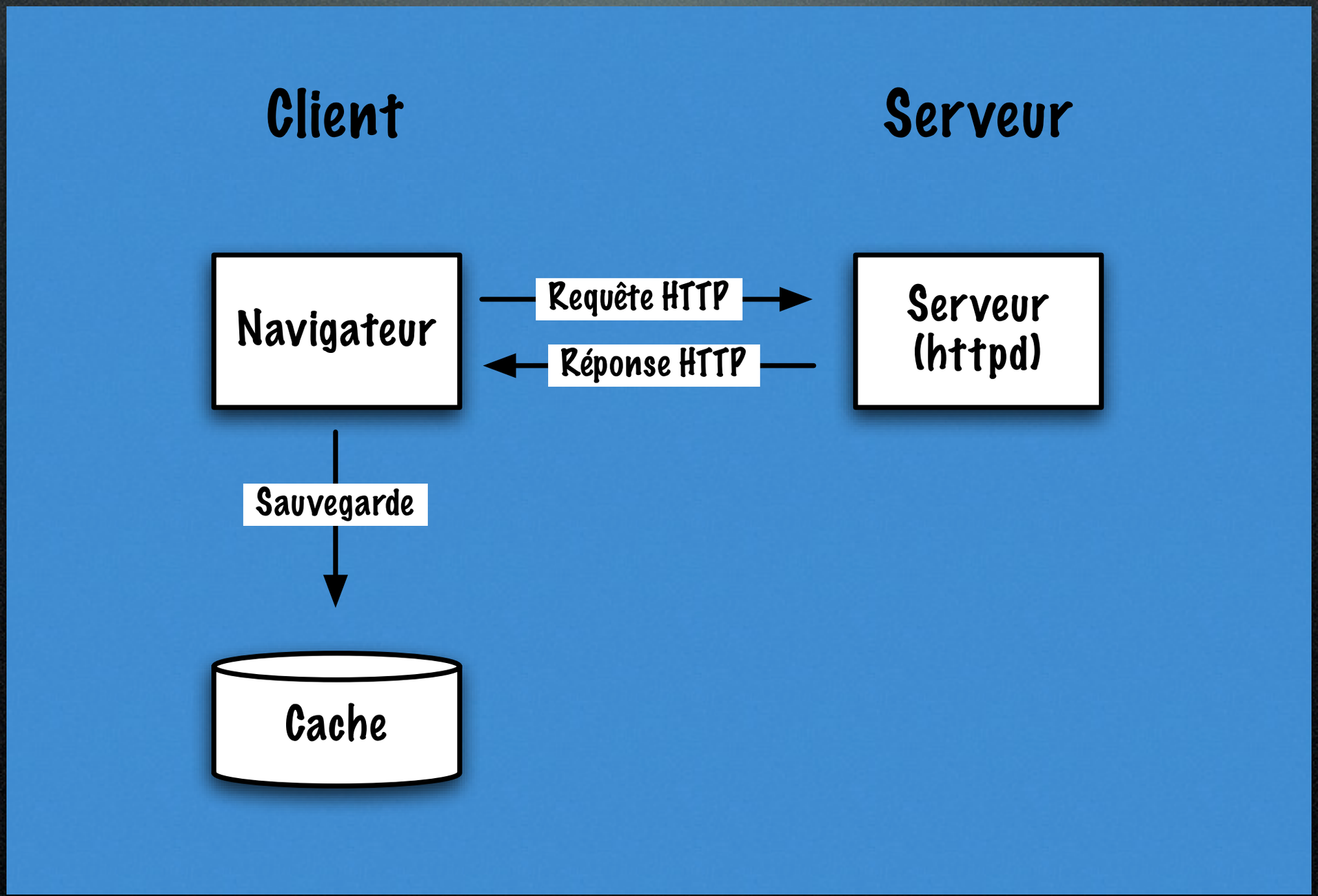
5000
----9055292283946
Content-Disposition: form-data; name="fichier"; filename="texte.txt"
Content-Type: text/plain

Coucou
Ceci est le texte envoye
fini
----9055292283946--

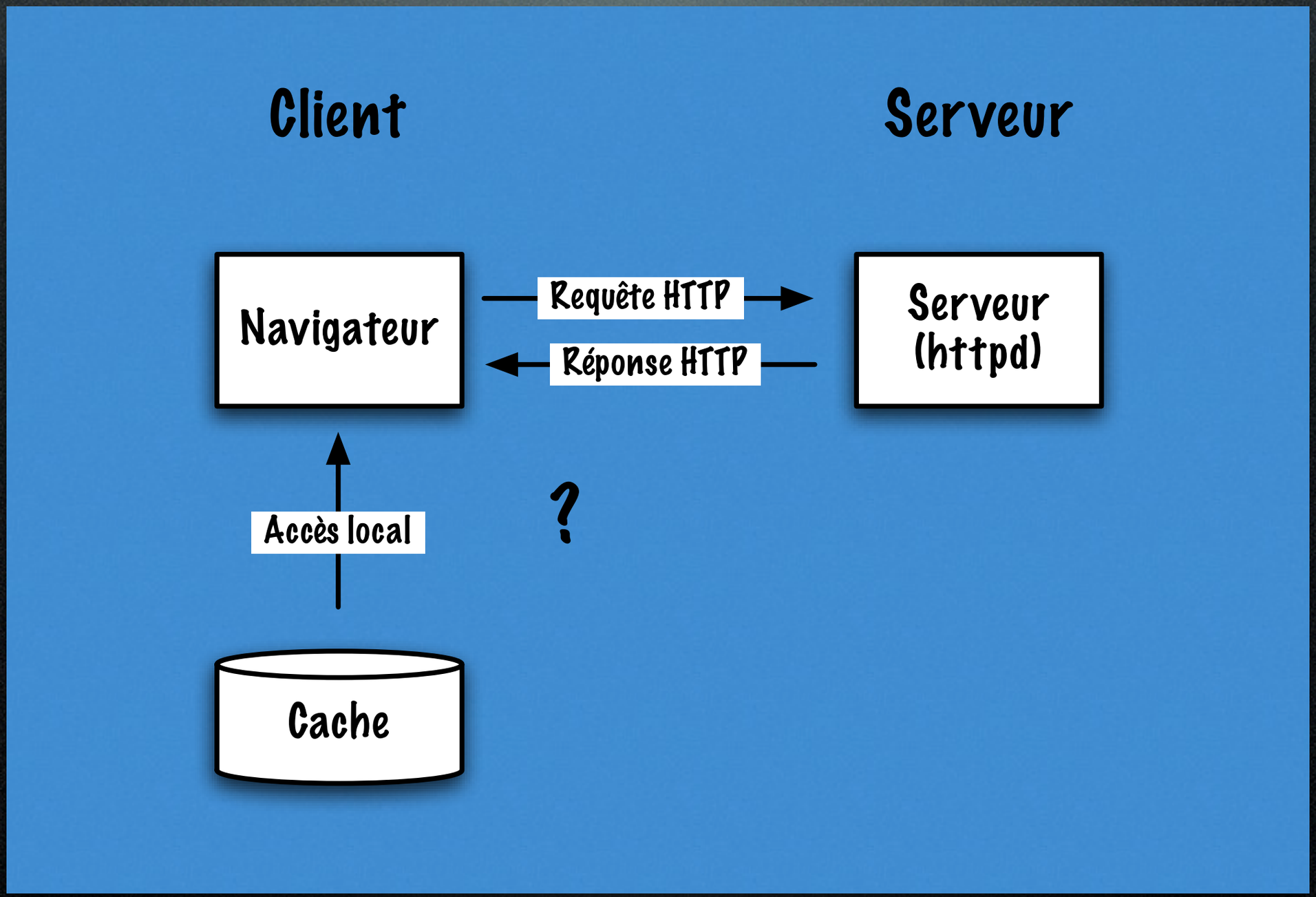
Cache côté client

- Une **cache** est un **dépôt local** de **copies d'informations** obtenues des serveurs Web
- Existe depuis les tous débuts du Web
- Évite plusieurs requêtes coûteuses
  - Toutes les pages de l'université ont une bannière commune

Cache côté client



Cache côté client



Cache côté client

- Avantages :
  - Bien plus rapide qu'une requête HTTP
  - Réduit le trafic sur le réseau
  - Réduit la charge du serveur
- Désavantages :
  - Informations locales périmées

Cache côté client

- Validation des ressources locales :
  - Envoie d'une requête **HEAD**.
    Consulte les en-têtes **Last-Modified** ou **ETag** ou **Expires**
- Sinon, le navigateur utilise une heuristique
  - Expires = 0.01 * (Date – Last-Modified) + Date

Cache-Control

- L'en-tête du message réponse HTTP peut contenir une directive afin de prévenir des problèmes liés à l'utilisation d'une cache

- **Cache-Control : no-cache** ou **no-store**
- **Cache-Control : max-age=delta-second**
- **Cache-Control : public** ou **private**

Tutoriel

- Caching Tutorial for Web Authors and Webmasters [http://www.mnot.net/cache_docs] 9 février 2008

Jeux et encodage de caractères

Jeu de caractères

- **character set**
- Un **jeu de caractères** établit une correspondance entre des **valeurs entières**, code caractère ou code-point et un **répertoire de caractères** (symboles)
  - Examples : US-ASCII, UNICODE (ISO 10646)

Le code caractère est un index pour une table de caractères.

Encodage de caractères

L'encodage est nécessaire pour l'envoie des données sur un réseau ou la sauvegarde des informations sur un support informatique.

- **character encoding**
- Un **encodage de caractères** établit une correspondance entre **une chaîne de bits** et une **séquence de code caractères** (ces derniers sont alors interprétés à l'aide d'un jeu de caractères)
- Exemples : **UTF-8** et **UTF-16**

UTF-8

- UTF-8 est un encodage utilisant un seul octet pour les caractères US-ASCII et jusqu'à 4 octets pour les autres caractères UNICODE

Préférences du client

- Le client peut spécifier ses préférences à l'aide d'un en-tête Accept-charset du message HTTP requête
- **Accept-charset: ISO-8859-1, utf-8; q=0.8, * ;q=0.5**
- En principe, le document peut être "transcodé" dynamiquement par le serveur, si le serveur ne peut utiliser un encodage acceptable alors il devrait signaler l'erreur à l'aide du code 406

Encodage du document retourné par le serveur

- Le message réponse du serveur peut contenir un en-tête Content-Type dont l'un des arguments du type/sous-type indique le jeu/encodage utilisé
  - Content-Type: text/html; charset=UTF-8
  - Le format des documents prévoit une façon de spécifier le jeu/encodage (attribut de balise XML, balise META, BOM, etc.)

Jeux et encodages de caractères

- Malheureusement, les attributs et paramètres des en-têtes utilisent l'étiquette charset pour parler de l'encodage!
  [HTML 4.01 Specification, § 5.2]
- Registre des jeux de caractères http://www.iana.org/assignments/character-sets

Préférences

RFC 2616 HTTP/1.1 § 14

- Le protocole HTTP fournit un mécanisme permettant aux agents-usager (AU) de stipuler leurs préférences
- L'AU peut quantifier ses préférences à l'aide d'un index de qualité
- Accept: audio/*; q=0.2, audio/basic

Mais encore

- Accept-Encoding: compress;q=0.5, gzip;q=1.0
- Accept-Language: fr, en-gb;q=0.8, en;q=0.7

Principales en-têtes du message HTTP requête

- **Host**: nom de l'hôte (obligatoire)
- **User-Agent**: type du navigateur
- **Accept**: types MIME acceptés par le client
- **Connection**: keep-alive ou close

Principales en-têtes du message HTTP requête

- **Content-Type:** type MIME du corps de la requête (POST), typiquement application/x-www-form-urlencoded
- **Content-Length**: longeur du corps en octets
- **Referer**: L'URL du document ayant servi à construire cette requête HTTP

Principales en-têtes du message réponse HTTP

- **Connection, Content-Type, Content-Length**
- **Date:** date à laquelle le message a été généré (obligatoire)
- **Location:** URI pour une redirection
- **Last-Modified:** date de la dernière modification

Principales en-têtes du message réponse HTTP

- **Expires:** date et heure à laquelle le corps du message sera considéré périmé
- **ETag:** un identificateur unique pour cette resource **(changera si la resource change)**

Fureteurs/serveurs

Client Web (Agent usager)

- Interface textuelle (lynx, W3 sous Emacs)
- Robots (googlebot, etc.)
- wget
- Téléphones portables
- Etc.

Client : interface vocale

- "Voice Browser" Activity
  www.w3.org/Voice
- Google Voice Local Search
  1-800-GOOG-411 (1-800-466-4411)

Clients : interface graphique

- 1993, Mosaic (NCSA - National Center for Supercomputer Applications)
- Netscape (Netscape Communications Corportation) (Gecko)
- Mozilla (Gecko), Firefox (Gecko)
- Internet Explorer (IE)Microsoft (Trident)
- Safari (WebKit), Cameo (Gecko)
- Opera (Presto)
- Chrome (WebKit)

CSI 3540 :: Structures, techniques et normes du Web :: Marcel Turcotte

http://www.site.uottawa.ca/~turcotte/teaching/csi-3540/

Google

Getting Started Latest Headlines Gmail - Inbox

bio :: start | CSI 3540 :: Structures, techniq... | World Wide Web Consortium

CSI 3540. Structures, techniques et normes du Web [Accueil | Plan de cours | Horaire | Ressources]

CSI3540. Structures, techniques et normes du Web (3,1.5,1.5) 3 cr.

Infrastructure de base du Web. Serveurs et navigateurs. Exemples de protocoles. Internet et virus. Architecture de moteur de recherche. Contenu et présentation Web. Pages Web, leur structure et leur interprétation. HTML, XML et leurs dérivés. Interfaces Web vers les logiciels et bases de données. Témoins et droit à la vie privée. Web sémantique et ontologies. Services Web.

Préalables : CSI2510, CSI 2532.

- Notes de cours
- Devoirs
- Examens

Done

- Génération des messages HTTP requêtes
- Interprétation de la réponse (HTML)
- Exécution de scripts (Javascript)
- Gestion d'événements (cliques et mouvements de souris, saisie de caractères)

- Sécuriser la communication
- Exécution de «plugins» ou d'applications externes
  - about:plugins, about:blank, about:config, about:cache

Serveurs Web

- 1993, NCSA développe le premier serveur HTTP
- 1995, Conception d'une version code libre
  - Source du nom **Apache**?
  - «a patchy server»
- Microsoft's Internet Information Server (IIS)

Serveurs Web

- Apache
  - Linux, Windows, Mac OS X, OS/2 (Unix en général)
    - 53.84%
- IIS
  - Windows
  - 24.08%

GlassFish

- Environnement de référence
- Successeur de Tomcat

- Environnement intégré : HTTP, conteneur de servlets Java, outils et librairies pour la création de services Web, banque de données, etc.

Sun Java System Application Server Platform Edition – Server Running

http://localhost:8080/

Google

Google ▾ Sun ▾ uOttawa ▾ Bioinformatics ▾ ML ▾ References ▾ Timeout ▾

bio :: start | glassfish: Quick Start Gu... | Java(TM) System Applica... | Java System Application ...

sun.com

Sun Java System Application Server Platform Edition 9.0

Your server is up and running!

To replace this page, overwrite *<install_dir>*/domains/*<domain_name>*/docroot/index.html, where *<install_dir>* is the Application Server installation directory, and *<domain_name>* is the domain name (for example, domain1).

You are invited to register your product now. Registration is optional, but as a registered user, you will get:

- News about product updates
- Access to value-added contents
- Notification of promotional programs
- Entry in Java EE platform-related gift give-aways

Also check out the Glassfish project, the open source community for the Java EE application server.

More Information: For more information about the Application Server, samples, documentation, and additional resources, see *<install_dir>*/docs/about.html, where *<install_dir>* is the Application Server installation directory.

Sun Java(TM) System Application Server Platform Edition 9.0 Admin Console

http://localhost:4848/asadmin/admingui/TopFrameset

Google

Google ▾ Sun ▾ uOttawa ▾ Bioinformatics ▾ ML ▾ References ▾ Timeout ▾

bio :: start | glassfish: Quick Start Gu... | Java(TM) System Applica... | Java System Application ...

HOME | VERSION | UPGRADE | REGISTRATION | LOGOUT | HELP

User: admin **Server:** localhost **Domain:** domain1

Sun Java™ System Application Server Admin Console

Java™

Sun™ Microsystems, Inc.

Common Tasks

- Application Server
- **Applications**
  - Enterprise Applications
  - **Web Applications**
  - EJB Modules
  - Connector Modules
  - Lifecycle Modules
  - App Client Modules
- Web Services
- Custom MBeans
- Resources
- Configuration

Application Server > Applications > Web Applications

Web Applications

A Web application module consists of a collection of Web resources such as JavaServer Pages (JSPs), servlets, and HTML pages that are packaged in a WAR (Web Application Archive) file or directory.

Deployed Web Applications (0)

Deploy... | Undeploy | Enable | Disable

Application Name	Enabled	Context Root	Actions
No applications found. Click "Deploy..." above to deploy a new application.			

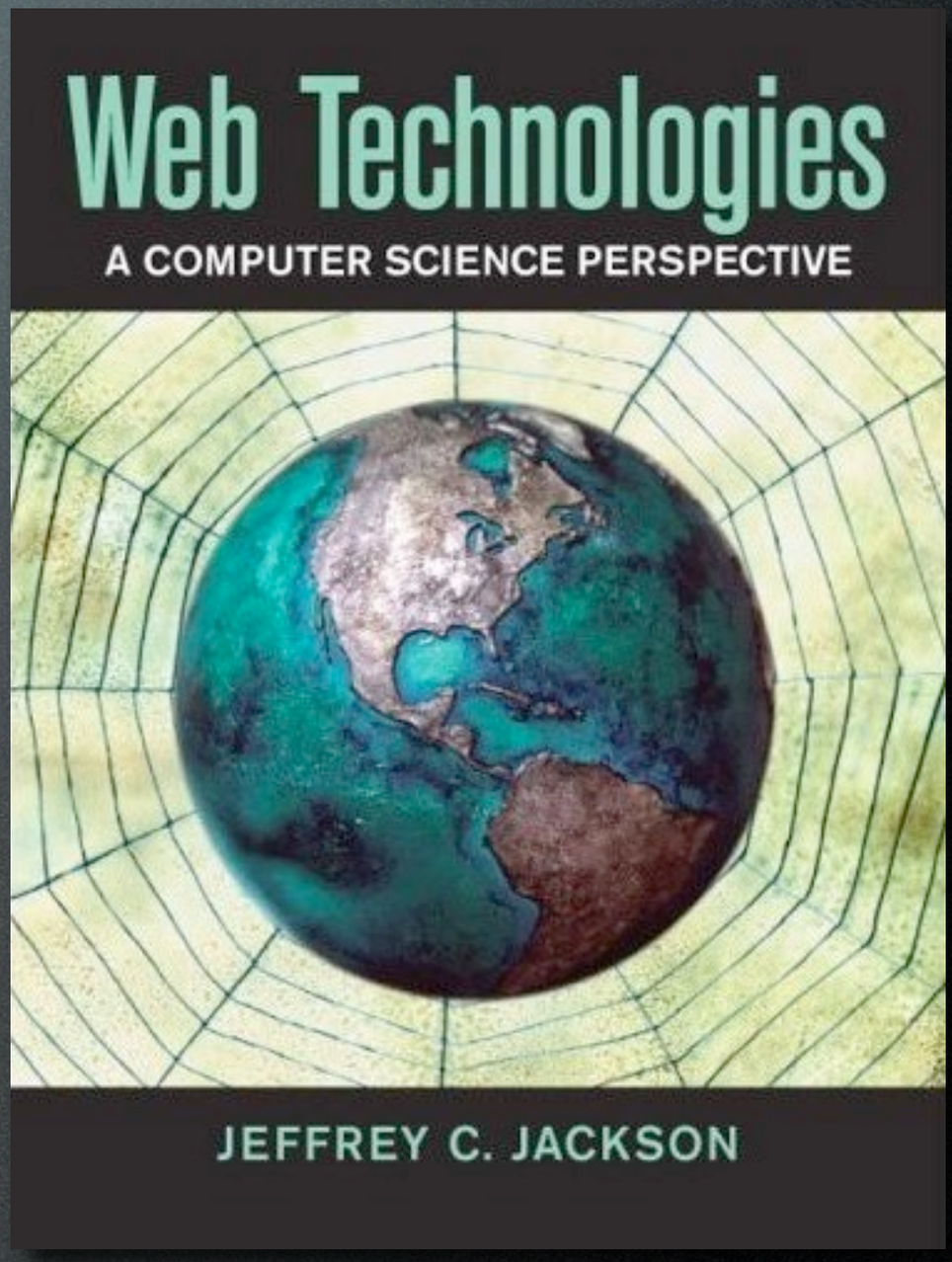
- Écoute sur le port HTTP, reçoit les requêtes HTTP à l'aide de TCP
- Sélection du serveur virtuel
- Sélection de la ressource spécifiée par l'URI requête
  - Exécution de programmes externe et retourne le message HTTP de retour
  - Retourne le message HTTP réponse (et le contenu d'un fichier)
- Création des en-têtes HTTP (MIME)
- Sauvegarde d'information (logs)

Web

- Ensemble de ressources hypertextes
  - Utilise le protocole de communication **HTTP**
    - Les informations sont représentées à l'aide de **HTML** (**XHTML**)
  - Graphe orienté dont les noeuds sont des documents et les arêtes sont des liens

Ressources

- J. C. Jackson (2007) Web Technologies : A Computer Science Perspective. Pearson Prentice-Hall.



Resources

- Liste des traductions françaises disponibles des documents W3C [http://www.w3.org/2005/11/Translations/Lists/ListLang-fr.html] 2007
- Character Model for the World Wide Web 1.0: Fundamentals [http://www.w3.org/TR/charmod] 14-Jan-2008