Technologies du Web

Technologies côté serveur

23 octobre 2012

Pierre Senellart

Plan

HTTP (HyperText Transfer Protocol)

- Application protocol at the basis of the World Wide Web
- Latest and most widely used version: HTTP/1.1

- Client request:

```
GET /MarkUp/ HTTP/1.1
Host: www.w3.org
```

- Server response:

```
HTTP/1.1 200 OK
...
Content-Type: text/html; charset=utf-8

<!DOCTYPE html ...> ...
```

- Two main HTTP methods: GET and POST (HEAD is also used in place of GET, to retrieve meta-information only).
- Additional headers, in the request and the response
- Possible to send parameters in the request (key/value pairs).

GET

- Simplest type of request.
- Possible parameter are sent at the end of a URL, after a '?'
- Not applicable when there are too many parameters, or when their values are too long.
- Method used when a URL is directly accessed in a browser, when a link is followed, and for some forms.

Example (Google query)

URL: `http://www.google.com/search?q=hello`

Corresponding HTTP GET request:

```
GET /search?q=hello HTTP/1.1
Host: www.google.com
```

POST

- Method only used for submitting forms.

Example

```
POST /php/test.php HTTP/1.1
Host: www.w3.org
Content-Type: application/x-www-form-urlencoded
Content-Length: 100

type=search&title=The+Dictator&format=long&country=US
```

Parameter encoding

- By default, parameters are sent (with GET or POST) in the form: `name1=value1&name2=value2`, and special characters (accented characters, spaces...) are replaced by codes such as +, `%20`. This way of sending parameters is called `application/x-www-form-urlencoded`.
- For the POST method, another heavier encoding can be used (several lines per parameter), similar to the way emails are built: mostly useful for sending large quantity of information. Encoding named `multipart/form-data`.

Status codes

- The HTTP response always starts with a status code with three digits, followed by a human-readable message (e.g., 200 OK).
- The first digit indicates the class of the response:
  1. Information
  2. Success
  3. Redirection
  4. Client-side error
  5. Server-side error

Most common status codes

200	OK
301	Permanent redirection
302	Temporary redirection
304	No modification
400	Invalid request
401	Unauthorized
403	Forbidden
404	Not found
500	Server error

Virtual hosts

- Different domain names can refer to the same IP address, i.e., the same physical machine (e.g., `www.google.fr` and `www.google.com`)
- When a machine is contacted by TCP/IP, it is through its IP address
- No *a priori* way to know which precise domain name to contact
- In order to serve different content according to the domain name (virtual host): header `Host:` in the request (only header really required)

Example

```
GET /search?hl=fr&q=hello HTTP/1.1
Host: www.google.fr
```

Content type

- The browser behaves differently depending on the content type returned: display a Web page with the layout engine, display an image, load an external application, etc.
- MIME classification of content types (e.g., `image/jpeg`, `text/plain`, `text/html`, `application/xhtml+xml`, `application/pdf` etc.)
- For a HTML page, or for text, the browser must also know what character set is used (this has precedence over the information contained in the document itself)
- Also returned: the content length (can be used to display a progress bar)

Example

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Content-Length: 3046
```

Client and server identification

- Web clients and servers can identify themselves with a character string
- Useful to serve different content to different browsers, detect robots...
- ... but any client can say it's any other client!
- Historical confusion on naming: all common browsers identify themselves as Mozilla!

Example

```
User-Agent: Mozilla/5.0 (X11; U; Linux x86_64; fr;
rv:1.9.0.3) Gecko/2008092510 Ubuntu/8.04 (hardy)
Firefox/3.0.3

Server: Apache/2.0.59 (Unix) mod_ssl/2.0.59 OpenSSL/0.9.8e
PHP/5.2.3
```

Authentication

- HTTP allows for protecting access to a Web site by an identifier and a password
- Attention: (most of the time) the password goes through the network uncrypted (but for instance, just encoded in Base64, revertible encoding)
- HTTPS (variant of HTTP that includes encryption, cryptographic authentication, session tracking, etc.) can be used instead to transmit sensitive data

Example

```
GET ... HTTP/1.1
Authorization: Basic dG90bzp0aXRp
```

Content negotiation

- A Web client can specify to the Web server:
  - the content type it can process (text, images, multimedia content), with preferrence indicators
  - the languages preferred by the user
- The Web server can thus propose different file formats, in different languages.
- In practice, content negociation on the language works, and is used, but content negociation on file types does not work because of bad default configuration of some browsers.

Example

```
Accept: text/html,application/xhtml+xml,application/xml;
q=0.9,*/*;q=0.8
Accept-Language: fr,fr-fr;q=0.8,en-us;q=0.5,en;q=0.3
```

Cookies

- Information, as key/value pairs, that a Web server asks a Web client to keep and retransmit with each HTTP request (for a given domain name).
- Can be used to keep information on a user as she is visiting a Web site, between visits, etc.: electronic cart, identifier, and so on.
- Practically speaking, most often only stores a session identifier, connected, on the server side, to all session information (connected or not, user name, data. . .)
- Simulates the notion of session, absent from HTTP itself

Example

```
Set-Cookie: session-token=RJYBsG//azkfZrRazQ3SPQhlo1FpkQka2;
path=/; domain=.amazon.de;
expires=Fri Oct 17 09:35:04 2008 GMT

Cookie: session-token=RJYBsG//azkfZrRazQ3SPQhlo1FpkQka2
```

Conditional downloading

- A client can ask for downloading a page only if it has been modified since some given date.
- Most often not applicable, the server giving rarely a reliable last modification date (difficult to obtain for dynamically generated content!).

Example

```
If-Modified-Since: Wed, 15 Oct 2008 19:40:06 GMT

304 Not Modified
Last-Modified: Wed, 15 Oct 2008 19:20:00 GMT
```

Originating URL

- When a Web browser follows a link or submits a form, it transmits the originating URL to the destination Web server.
- Even if it is not on the same server!

Example

```
Referer: http://www.google.fr/
```

Plan

Rôle des programmes côté serveur

- Le Web, ce n'est pas qu'un ensemble de documents HTML statiques !
- Les programmes côté serveur permettent :
  - de traiter des soumissions de formulaire ;
  - d'afficher de manière uniforme l'ensemble des pages d'un site ;
  - de proposer des applications interactives ;
  - de permettre à l'utilisateur d'ajouter ou modifier du contenu ;
  - etc.

Web servers

Server	Share	Distribution
Apache	65%	Windows, Mac OS, Linux, Unix FS
Microsoft IIS	15%	with some versions of Windows
nginx	10%	Windows, Mac OS, Linux, Unix FS
lighthttpd	2%	Windows, Mac OS, Linux, Unix FS
Unidentifiable	8%	

- Market share: according to some studies by Opera and Netcraft, precise numbers do not really mean anything.
- Many large software companies have either their own Web server or their own modified version of Apache (notably, GFE/GWS for Google).
- nginx and lighthttpd are lighter (i.e., less feature-rich, but faster in some contexts) than Apache.
- The versions of Microsoft IIS released with consumer versions of Windows are very limited.

Langages de programmation

- CGI (Common Gateway Interface) : interface normalisée permettant de faire communiquer le serveur Web avec un programme s'exécutant sur le serveur ;
- CGI permet d'utiliser n'importe quel langage de programmation (compilés comme C, C++, Java, ou interprétés comme Perl, Python, Ruby, etc.) pour écrire des langages côté serveur.
- Mais certains langage sont plus adaptés au développement Web :
  - comportent des fonctions spécialement dédiées à HTTP, HTML, etc. ;
  - s'intègrent de manière plus efficace et plus pratique avec le serveur Web (moyennant des extensions logicielles) ;
  - ont une syntaxe spécialement conçue, qui mêle code HTML envoyé tel quel et instructions de programmation interprétées.
- Quelle que soit la technologie utilisée, le code du programme n'est pas accessible par le navigateur Web, seulement le résultat de son exécution.

Langages côté serveur

PHP : un des langages les plus populaires, s'intègre très facilement avec Apache (libre)

ASP et ASP.NET : destiné à être utilisé avec IIS (Microsoft, commercial)

ColdFusion (Adobe, commercial)

JSP (Java Server Pages) : permet de mêler instructions Java et code HTML ; nécessite un serveur d'applications Java (p. ex., Tomcat) en plus d'Apache (Sun, gratuit voire libre)

Servlets Java : véritables programmes Java, plutôt pour les applications complexes côté serveur avec peu d'interaction côté client ; nécessite un serveur d'applications Java en plus d'Apache (Sun, gratuit voire libre)

Frameworks d'applications Web

- Les langages présentés ci-avant restent assez basiques et généralistes.
- N'encouragent pas forcément une organisation propre d'un site Web.
- Framework : ensemble d'un langage de programmation, d'une bibliothèque de fonctions, d'outils externes, de bonnes pratiques à suivre. . .
- Permet d'abstraire la création d'une page Web.
- Suit en général le modèle MVC (voir transparent suivant).
- Inclut parfois la génération de code JavaScript côté client pour créer directement une application Web fortement dynamique (p. ex., validation de formulaire) ; intégration Ajax également.
- Fortement recommandé pour créer des applications complexes. . . mais également complexe à maîtriser !

Modèle MVC

- Principe de génie logiciel, utilisé dans d'autres domaines
- Particulièrement adapté au cas des applications Web !
- Séparation propre entre :

  Modèle : données manipulées par l'application et fonctions de manipulation de ces données ; réutilisable pour d'autres applications Web

  Vue : présentation des données ; facilement échangeable pour changer l'apparence et la structure du site

  Contrôleur : contrôle la manière dont l'utilisateur interagit, au travers de la vue, avec les données du modèle

Frameworks côté serveur les plus populaires

ASP.NET : DotNetNuke

ColdFusion : Model-Glue, Fusebox

Java : Struts, Spring, JavaServer Faces, Google Web Toolkit

Perl : Catalyst

PHP : CakePHP, Symphony, Zend

Python : Django

Ruby : Ruby on Rails (a eu beaucoup d'influence !)

Smalltalk : Seaside

. . . et beaucoup d'autres !

CMS

- CMS (Content Management System)
- Permettent de créer des sites Web sans aucun développement.
- Fonctionnalités :
  - édition simplifiée de page (syntaxe wiki ou bbcode, ou contrôle JavaScript texte enrichi) ;
  - ajout de contenu externe (images, documents annexes, etc.) ;
  - gestion d'utilisateurs, contrôle d'accès, etc. ;
  - modules de gestion de forums, de blogs ;
  - thèmes graphiques prêt à l'emploi ;
  - contrôle de version.
- Suivant les CMS, extensions pouvant être nombreuses.
- Certains CMS spécialisés : blogs (Dotclear, Movable Type, TypePad), commerce électronique (PrestaShop, Magento), forums (phpBB, MyBB), etc.

CMS les plus utilisés

Nombre de téléchargements hebdomadaires :

WordPress	PHP	433 767
Joomla !	PHP	189 429
Drupal	PHP	62 500
Umbraco	.NET/XSLT	5 670
eZ Publish	PHP	5 612
CMS Made Simple	PHP	4 903
SilverStripe	PHP	2 500
e107	PHP	2 242
Xoops	PHP	1 209
TikiWiki	PHP	373
phpWebSite	PHP	347
Typo3	PHP	100
Alfresco	Java	57
DotNetNuke	ASP.NET	?
Jahia	Java	?
Liferay	Java	?
modx	PHP	?
OpenCMS	Java	?
Plone	Python	?
TextPattern	PHP	?

CMS et programmation Web

Même en utilisant un CMS, il est utile de connaître les technologies de base du Web (HTML, CSS, JavaScript, langage côté serveur) :

- pour créer ses propres styles CSS (quasi-indispensable) ;
- pour développer des applications complexes, propres au site ;
- pour développer ou adapter des petites extensions ;
- pour comprendre ce qui se passe en cas de problème ;
- pour s'assurer que les pages respectent certaines conditions (validité W3C, accessibilité).

CMS et Wikis

Frontière peu nette entre les deux... Les Wikis mettent l'accent sur :

- a priori, tout est éditable par un grand nombre d'utilisateurs ; pas de notion de « propriété »d'une page ou d'un document.
- insistance sur le contrôle de versions

Nombreux wikis : MediaWiki, TWiki, Dokuwiki...

Choisir un CMS

Point auxquels faire attention :

- la licence (la plupart des CMS populaires sont libres, mais pas tous)
- le langage sous-jacent, important s'il y a besoin de modifier/personnaliser le logiciel, et également important pour savoir comment le déployer
- la disponibilité de fonctionnalités spécifiques au site : blogs, forums, support vidéo, etc. Il est possible d'utiliser des composants externes pour chacune des tâches, mais plus simple de rester sur un seul système.

Inconvénients potentiels

- Plus lent qu'un site classique ; fonctionnalités de cache permettant d'accélérer, mais peut poser problème dans certains contextes.
- Failles de sécurité, code sur lequel on n'a pas de contrôle.
- Logiciels orphelins.
- Migration vers un autre système potentiellement coûteuse.
- Non adapté à tous les usages.

Repérer une technologie côté serveur

Impossible d'avoir accès au code source, donc comment savoir avec quelle technologie côté serveur un site a été réalisé ?

- URL : (`.php` pour PHP, `.jsp` pour JSP, `.asp` pour ASP, `/servlet/` pour une servlet...). Tout cela est configurable, mais bien souvent la configuration par défaut est utilisée.
- Ça peut être écrit en toutes lettres sur le site (souvent pour un CMS)
- Regarder les commentaires dans le code HTML, CSS...
- Organisation des fichiers et répertoires, librairies JavaScript ou CSS chargées, etc.
- Cookies, en particulier identifiants de session

Plan

PHP et HTML

- Script PHP : document HTML (par exemple), dans lequel est incorporé du code PHP.
- Le code PHP est à l'intérieur d'une pseudo-balise `<?php ... ?>` (ou `<? ... ?>`, ou `<?= ... ?>` qui est un raccourci pour `<? echo ... ?>`).

Exemple

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">

<html>
  ...
  <body>
  <h1><?php echo 2+2; ?></h1>
</body>
</html>
```

Introduction au langage

- Un script PHP est une suite d'instructions terminées par des points-virgules.

Exemple (Écrire une phrase avec l'instruction echo)

```
echo 'Ceci apparaîtra dans la page générée.';
```

- Ces instructions contiennent des éléments variables ou constants, peuvent être conditionnées ou encore itérées plusieurs fois.

Notion de variable

- Une variable est le nom d'un récipient destiné à contenir une valeur. Dans un ordinateur, ce récipient est en fait une zone mémoire.
- L'affectation `$var=valeur` est une instruction qui permet de changer la valeur d'une variable `$var` :
  - La valeur de la variable à gauche de `=` est remplacée par la valeur à droite de `=`.
- L'affectation est une instruction dite destructive : l'ancienne valeur de la variable est détruite, écrasée, effacée par la nouvelle valeur !
- Une valeur peut être :
  - un nombre entier ou réel ;
  - une chaîne de caractères définie entre deux guillemets simple (`'`) ou entre deux guillemets double (`"`).

Affectation

- La valeur affectée à une variable peut également être le résultat d'une expression.
- Attention, à droite d'une affectation `=`, les variables utilisées dans l'expression désignent les valeurs qu'elles contiennent.

Exemple

`$Compteur= $Compteur + 1;` a pour effet de mettre le résultat de la somme de la valeur de `$Compteur` avec la valeur 1 dans le récipient `$Compteur`.

Instruction conditionnelle

- Une instruction conditionnelle est de la forme :
  Si *la condition C est vraie* alors *faire liste d'instructions T*
  sinon *faire liste d'instructions E*
- où une condition est une expression et *T* (resp. *E*) est un bloc d'instructions.
- En PHP, on utilise la syntaxe suivante :

```
if (C) { T } else { E }
```

Exemple

```
if ($sexe=='f') {
  echo 'Madame';
} else {
  echo 'Monsieur';
}
```

- La condition est généralement un test de comparaison, ou plusieurs tests reliés par des opérateurs logiques.
- Le `else` et son bloc associé sont optionnels.
- Si un bloc ne contient qu'une seule instruction, on pourra omettre les accolades.

Exemple

```
if ($sortie=='q') { echo 'Merci et au revoir!'; }

if ($a > $b) { echo "$a est plus grand que $b"; }
else {
  if ($a==$b) { echo "$a est égal a $b"; }
  else { echo "$a est plus petit que $b"; }
}
```

Boucles

- Boucle : permet de répéter une opération

```
for (I;C;P) { B }
```

- où *I* est une affectation permettant d'initialiser la variable itérée,
- *C* est la condition d'arrêt de la boucle (valeur limite que doit atteindre la variable itérée),
- *P* est le pas de la boucle : c'est une affectation qui permet de modifier à chaque itération la valeur de la variable itérée,
- *B* est le bloc d'instructions itéré.

Exemple

```
for( $i=0; $i<10; $i=$i+1 ) { echo 'Au secours!'; }

$fact=1;
for( $i=13; $i>0; $i=$i-1 ) { $fact = $fact*$i; }
```

$_GET et $_POST

- Les paramètres HTTP peuvent être récupérées en PHP grâce au tableaux associatifs `$_GET` et `$_POST`.
- Les valeurs de ce tableau peuvent être des variables simples ou des tableaux indicés : ces derniers sont les paramètres à choix multiples dont on a suffixé le nom de `[]` dans le code HTML.

Exemple

```
echo "<p>Votre login est: " . $_POST["login"] . "</p>";
echo "<p>Vous avez coché les genres: ";
for($i=1;$i<=count($_POST['genre']);$i=$i+1) {
  echo $_POST['genre'][$i] . " ";
}
echo "</p>";
```

Validation

- Un script PHP avec du code PHP incorporé au sein de code HTML n'est pas un document XHTML valide !
- Ce qu'il faut valider, c'est une (des) page(s) HTML produites par le script.
- Possibilité d'indiquer une URL au validateur du W3C. Malheureusement pas utilisable quand l'URL est privée. Possibilité dans ce cas de sauvegarder le fichier HTML produit, et de l'envoyer au validateur du W3C comme fichier local.
- Il n'y a pas de « validateur » PHP, mais les erreurs de syntaxe causeront des erreurs à l'exécution du script.
- Rien ne change pour les CSS, on peut les valider à part.

Plan

SGBD

- SGBD : Système de Gestion de Bases de Données
- Fournit des méthodes efficaces pour gérer des données qui répondent à une structure (un schéma) précis.
- Rechercher des données : requêtes
- Ajouter, supprimer, modifier des données : mises à jour
- Traite de manière rapide de grandes quantités de données.
- Gérer des transactions (ne pas donner à deux clients la même place dans un train !)

Modèle relationnel

- Modèle le plus répandu et le plus classique.
- Les données sont organisées en des tables, chacune des colonnes représentant un attribut des données.

Exemple

Prénom	Nom	Date de naissance
Jean	Dupont	1967-08-07
Pascale	Dupuis	1981-09-12
Alfred	Lambert	NULL

- Chaque attribut (colonne) est typé.
- SQL (Structured Query Language) : langage standard de requête et de mise à jour des données (petites variantes suivant les SGBD).

SGBD relationnels les plus connus

Oracle	commercial, le plus utilisé en entreprise
IBM DB2	commercial
Microsoft SQL Server	commercial
Microsoft Access	commercial, pauvre en fonctionnalités
MySQL	libre, le plus utilisé sur le Web
PostGreSQL	libre, plus abouti et plus complexe que MySQL

Types de données

(Types MySQL)

- **INT :** entier (`42`)
- **REAL :** nombre en virgule flottante (`3.14159`)
- **VARCHAR(N) :** chaîne de caractères ayant au plus N caractères ; les valeurs sont délimitées par des apostrophes (`'Ceci est une chaîne'`).
- **TEXT :** longue chaîne de caractères, sans limite de taille
- **DATE :** date (`2005-11-08`)
- **TIME :** temps (`09:30:00`)

NULL

- NULL : valeur spéciale
- Dénote l'absence de valeur.
- Différent de `0`, de `''` ...
- Une comparaison normale (`=`, `<>`) avec `NULL` renvoie toujours FAUX.
- `IS NULL`, `IS NOT NULL` peuvent être utilisées pour tester une valeur.
- Chacune des colonnes doit être déclarée comme acceptant ou non la valeur `NULL`.

SQL

- Langage standardisé pour interagir avec un SGBD.
- Exemple de requête :

```
SELECT a.name, COUNT(*)
FROM actors a, casting m
WHERE a.id=m.actor_id
GROUP BY a.name
```

- Exemple de mise à jour :

```
UPDATE availabilities
SET nb_available=nb_available-1
WHERE train_id=1234
```

Bases de données pour le Web

- SGBD très utiles pour beaucoup d'applications Web (annuaires, catalogues, réservations, forums de discussion, blogs, etc.)
- Possibilité d'accéder aux bases de données plus ou moins simplifiée suivant les langages :
  - En PHP, bibliothèques de fonctions propres à chaque SGBD (p. ex., `mysql` ou `mysqli` pour MySQL).
  - En JSP, bibliothèque JDBC avec les scriptlets et balises `<sql :*>` avec la JSTL, pour accéder à n'importe quel SGBD.

Exemple : utilisation de MySQL avec PHP

```
if(!mysql_pconnect("server","login","password"))
  { echo "<p>Desolé, connexion impossible</p>"; exit; }
if(!mysql_select_db('database'))
  { echo "<p>Desolé, accès à la base impossible</p>"; exit; }

$resultat= mysql_query("SELECT * FROM Films");
if($resultat) {
   while($film=mysql_fetch_assoc($resultat)){
     echo "<p>".$film["Titre"]." est paru en ".
       $film["Annee"]." </p>";
  }
} else {
   echo "<p>Erreur dans l'éxécution de la requête.</p>";
   echo "<p>Message de MySQL: ".mysql_error()."</p>";
}
```

Plan

Outils

- Outils HTTP : Netcat, Wget, Curl
- Firebug (ou similaire) pour analyser du trafic HTTP
- N'importe quel éditeur de textes pour la plupart des langages côté serveur
- Un serveur Web (p. ex., Apache)
- L'interpréteur du langage côté serveur (p. ex., PHP et mod_php pour Apache)
- Pour JSP, Servlets : un serveur d'applications Java (p. ex., Tomcat)
- Un SGBD (p. ex., MySQL) ; sous Windows, EasyPHP regroupe Apache+PHP+MySQL
- La plupart du temps, un moyen de transférer les programmes vers le serveur Web distant (SSH, FTP, application Web. . .)

Références

HTTP
- http://www.faqs.org/rfcs/rfc2616.html
- http://www.ietf.org/rfc/rfc2109.txt
- http://www.ietf.org/rfc/rfc2965.txt

PHP
- http://www.php.net/
- *Pratique de MySQL et PHP*, Philippe Rigaux

MySQL : http://dev.mysql.com/doc/

Licence de droits d'usage

Contexte public } avec modifications

Par le téléchargement ou la consultation de ce document, l'utilisateur accepte la licence d'utilisation qui y est attachée, telle que détaillée dans les dispositions suivantes, et s'engage à la respecter intégralement.

La licence confère à l'utilisateur un droit d'usage sur le document consulté ou téléchargé, totalement ou en partie, dans les conditions définies ci-après et à l'exclusion expresse de toute utilisation commerciale.
Le droit d'usage défini par la licence autorise un usage à destination de tout public qui comprend :
– le droit de reproduire tout ou partie du document sur support informatique ou papier,
– le droit de diffuser tout ou partie du document au public sur support papier ou informatique, y compris par la mise à la disposition du public sur un réseau numérique,
– le droit de modifier la forme ou la présentation du document,
– le droit d'intégrer tout ou partie du document dans un document composite et de le diffuser dans ce nouveau document, à condition que :
  – L'auteur soit informé.

Les mentions relatives à la source du document et/ou à son auteur doivent être conservées dans leur intégralité.

Le droit d'usage défini par la licence est personnel et non exclusif.
Tout autre usage que ceux prévus par la licence est soumis à autorisation préalable et expresse de l'auteur : `sitepedago@telecom-paristech.fr`