

# CLIENT/SERVEUR

## A) Introduction

L'architecture client/serveur désigne un **mode de communication** entre plusieurs composants d'un réseau. Chaque entité est considérée comme un **client** ou un **serveur**. Chaque logiciel client peut envoyer des **requêtes** à un serveur. Un serveur peut être spécialisé en serveur **d'applications**, de **fichiers**, de terminaux, ou encore de **messagerie électronique**.



Fig. 1 Principe du client/serveur : « Donne-moi une page (url) »

**En résumé, le client pose une question (ou donne un ordre)... le serveur répond à la question (ou obéit).**

## B) Dialogue client-serveur

### Un serveur

Un serveur est initialement passif, **il attend**. Il est à **l'écoute**, prêt à répondre aux **requêtes** envoyées par des clients, dès qu'une requête lui parvient, il la traite et envoie une réponse.

### Un client

Le client est d'abord actif (ou maître), il envoie des **requêtes** au serveur, il attend et reçoit les réponses du serveur.

### Le dialogue

Le client et le serveur doivent utiliser le même **protocole** de communication.

Un serveur est généralement capable de **servir plusieurs clients** simultanément.

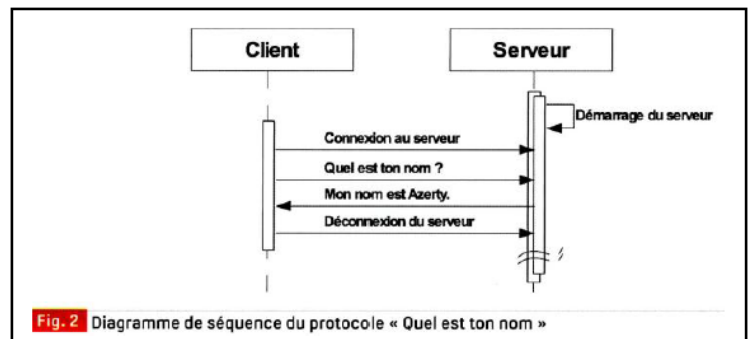


Fig. 2 Diagramme de séquence du protocole « Quel est ton nom »

\* Référence : Technologie 1STI2D Hachette

## C) HyperText Transfert Protocol, HTTP

La **consultation de pages sur un site Web** a un fonctionnement basé sur l'architecture **client/serveur**. Un internaute connecté au réseau via son ordinateur et un navigateur Web est le client, le serveur est constitué par le ou les ordinateurs contenant les applications qui délivrent les pages demandées. Dans ce cas, c'est le protocole de communication HTTP qui est utilisé.

L'**HyperText Transfert Protocol**, plus connu sous l'abréviation HTTP, littéralement le « **protocole de transfert hypertexte** », est le **protocole de communication client/serveur** développé pour le **World Wide Web**.

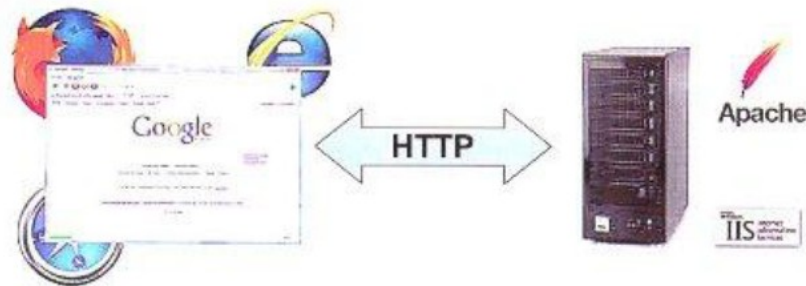


Fig. 3 Dialogue HTTP

Les **navigateurs** sont les **clients**(Firefox, Safari, Internet Explorer...). Ces clients se connectent à des serveurs HTTP tels qu'**Apache** ou IIS (Internet Information Service).

### Les méthodes

Dans le protocole HTTP, une **méthode** est une **commande** spécifiant un type de **requête**, c'est-à-dire qu'elle demande au serveur d'effectuer une action. En général, l'action concerne une ressource identifiée par l'URL qui suit le nom de la méthode.

Les méthodes les plus utilisées sont GET et POST.

GET	<b>C'est la méthode la plus courante pour <u>demandeur une ressource</u>. Une requête GET est sans effet sur la ressource.</b>
POST	Cette méthode doit être utilisée pour ajouter une nouvelle ressource, comme un message sur un forum, un article dans un site ou encore un login et un mot de passe.

Faire les exercices 1, 2, 3

### D) File Transfert Protocol, FTP

Le protocole de transfert de fichiers, ou FTP est un protocole de communication destiné à l'échange de fichiers sur un réseau TCP/IP. Il permet, depuis un ordinateur de copier des fichiers vers un autre ordinateur du réseau, d'alimenter un site Web, ou encore de supprimer ou de modifier des fichiers sur cet ordinateur.

FTP obéit à un modèle client-serveur. C'est-à-dire qu'une des deux parties, le client, envoie des requêtes et le serveur répond.



Client FTP Filezilla

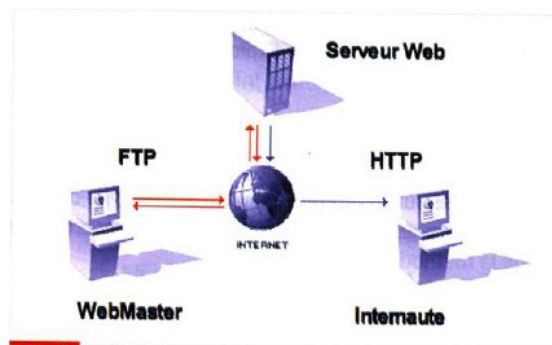


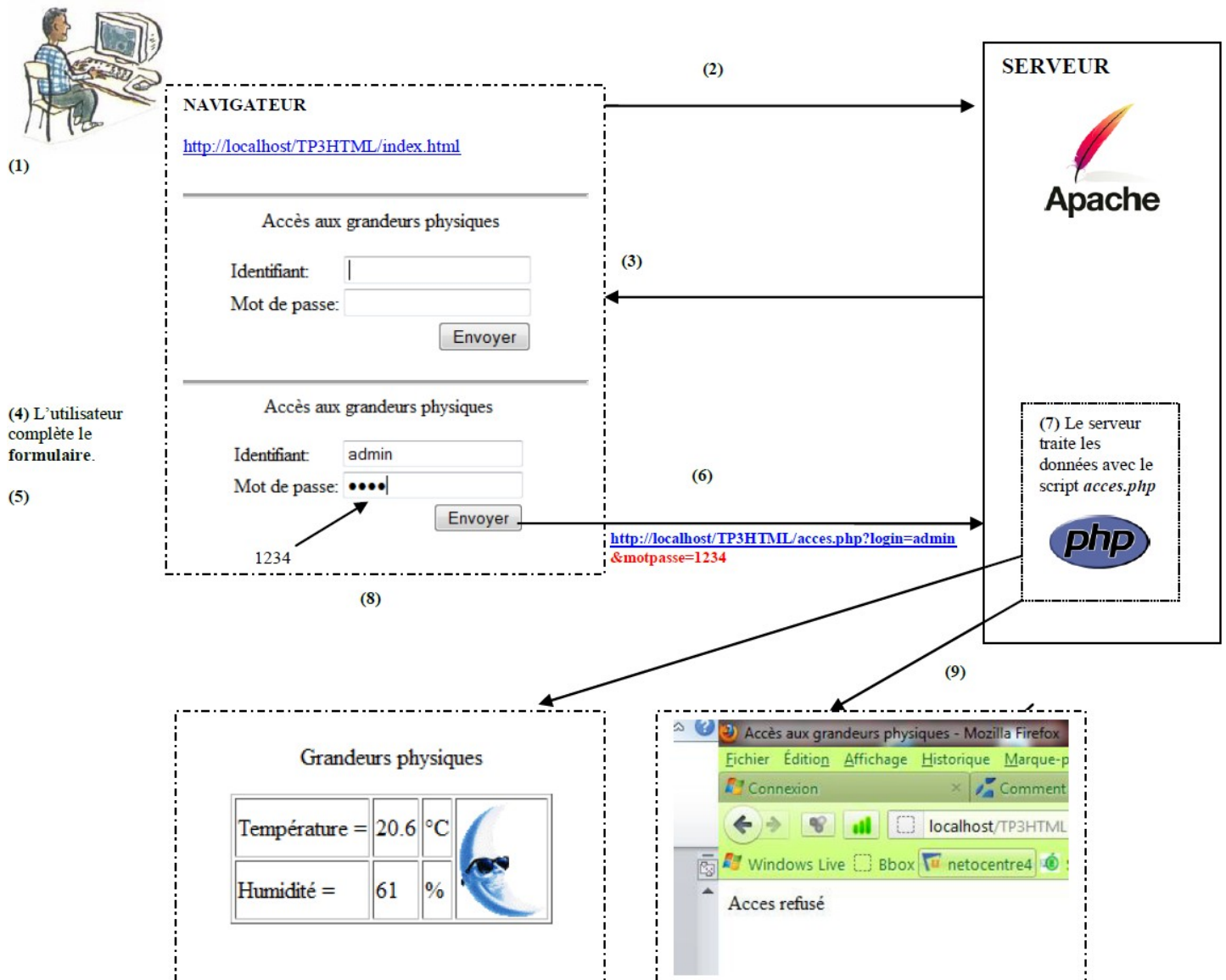
Fig. 5 Rôle de FTP dans la publication Web

## Exercices

### Exercice 1 : Dialogue HTTP

Q1) Complétez le schéma ci-dessous avec les phrases suivantes :

- **Requête** (méthode GET). Le navigateur envoie les données dans l'URL.
- L'utilisateur se connecte au serveur.
- Le serveur traite les données avec un script PHP.
- **Réponse**. Sinon Affichage (« Accès refusé ») ;
- L'utilisateur complète le formulaire.
- L'utilisateur envoie les données au serveur
- **Requête**. Le navigateur envoie l'URL.
- **Réponse**. Si (Identifiant = **admin** et Mot de passe = **1234**) alors affichage de la page « Grandeurs physiques »
- **Réponse**. Le serveur renvoie la page index.html





## Exercice 2 : Formulaire et PHP

On donne le fichier HTML du formulaire permettant d'accéder au tableau des grandeurs physiques.

```
<html>

<!-- Fichier index.html
      Login + code d'accès
-->
<head>
    <title> Accès aux grandeurs physiques </title>
</head>

<body>
    <p>Accès aux grandeurs physiques</p>
    <form action="VerifCode.php" method="GET">
        Identifiant:
        <input type="text" name="Identite" size="20"/><br>
        Mot de passe:
        <input type="password" name="Code" size="20"/><br>
        <input type="submit" name="EnvValeur" value="Envoyer"/>
    </form>
</body>

</html>
```



Accès aux grandeurs physiques

Identifiant:

Mot de passe:

Q2) Quelle est la méthode utilisée pour transmettre l'identifiant et le mot de passe au serveur ? \_\_\_\_\_

Q3) Quel est le nom du script de traitement du formulaire ? \_\_\_\_\_

Q4) Quel est le nom des informations transmises par le formulaire ? \_\_\_\_\_

On donne le fichier PHP de traitement du formulaire.

```
<html>
    <!--VerifCode.php-->
<head>
    <title> Accès aux grandeurs physiques </title>
</head>

<body>
<?php
    if (($Identite == "admin") && ($Code == "1234"))
    {
        header("Location: TblPhy.php");
    }
    else
    echo "Acces refusé";
?>
</body>

</html>
```

Q5) Comment le script de traitement « *VerifCode.php* » est-il capable de récupérer les données du formulaire ? \_\_\_\_\_

Q6) Le serveur Apache étant installé en local, écrivez l'URL transmise par le formulaire si l'identité est **admin** et le code **1234**. \_\_\_\_\_

### Exercice 3 : Formulaire et PHP

On donne le fichier HTML d'un formulaire destiné à alimenter une base de données.

```
<!-- fichier ajout.html -->
<html>
<head>
  <title> Formulaire d'ajout </title>
</head>
<body>
  <h2> Ajout d'une adresse </h2>
  <form name="ajout" methode="POST" action="ajout.php">
    <table border = "0">
      <tr>
        <td> Nom </td> <td> <input type="text" name ="nom"> </td>
      </tr>
      <tr>
        <td> Prenom </td> <td> <input type="text" name ="prenom"> </td>
      </tr>
      <tr>
        <td> Adresse </td> <td> <input type="text" name ="adresse"> </td>
      </tr>
      <tr>
        <td> Téléphone </td> <td> <input type="text" name ="telephone"> </td>
      </tr>
      <tr>
        <td> </td> <td> <input type="submit" value="Envoyer"/></td>
      </tr>
    </table>
  </form>
</body>
</html>
```

#### Ajout d'une adresse

Nom	<input type="text"/>
Prenom	<input type="text"/>
Adresse	<input type="text"/>
Téléphone	<input type="text"/>
<input type="submit" value="Envoyer"/>	

Q7) Quelle est la méthode utilisée pour transmettre l'identifiant et le mot de passe au serveur ? \_\_\_\_\_

Q8) Quel est le nom du script de traitement du formulaire \_\_\_\_\_

Q9) Quel est le nom des informations transmises par le formulaire ? \_\_\_\_\_

On donne le script du formulaire *ajout.php* situé dans le répertoire *./www/* du serveur Apache

```
<!-- fichier ajout.php -->
<?php include("config.inc"); ?>
<html>
  <head>
    <title> Ajout d'une adresse </title>
  </head>
  <body>
    <h1> Ajout </h1>
    <?php
      $con = mysql_connect($serv,$log,$pass) or die("Erreur de connexion");
      mysql_select_db($base,$con) or die("Erreur de sélection");
      $req = "insert into $table values('$nom','$prenom','$adresse','$telephone');";
      $res = mysql_query($req,$con) or die("Erreur d'insertion");
      mysql_close($con);
    ?>
    <p> personne ajoutée :
      <ul>
        <li> <?php print $nom ?> </li>
        <li> <?php print $prenom ?> </li>
        <li> <?php print $adresse ?> </li>
        <li> <?php print $telephone ?> </li>
      </ul>
    </p>
  </body>
</html>
```

Q10) Quel est le nom des variables utilisées pour « récupérer » les informations transmises par le formulaire ? \_\_\_\_\_

On donne le fichier construit à partir de *ajout.php* et renvoyé au client par le serveur.



```
<!-- fichier ajout.php -->
<!-- fichier config.inc -->
<html>
  <head>
    <title> Ajout d'une adresse </title>
  </head>
  <body>
    <h1> Ajout </h1>
    <p> personne ajoutée :
      <ul>
        <li> Dupont </li>
        <li> Denis </li>
        <li> 17, rue des Lauriers 07000 Privas </li>
        <li> </li>
      </ul>
    </p>
  </body>
</html>
```

Clic droit

## Ajout

personne ajoutée :

- Dupont
- Denis
- 17, rue des Lauriers 07000 Privas
- 

Q11) Comparez ce fichier à celui de la page précédente. Que remarquez-vous ? Pourquoi *ajout.php* n'apparaît-il pas sur le poste client tel qu'il est sur le serveur ?

## Synthèse

### SYNTHÈSE

Le mode client/serveur n'est pas le modèle de communication parfait, il n'y en a pas ! Connaissant les avantages et les inconvénients par rapport au mode distribué (par exemple : pair à pair), vous pourrez choisir celui qui convient.

#### Avantages de la communication client/serveur

- Toutes les données sont centralisées sur un seul serveur, contrôle de sécurité simplifié. » Les technologies supportant l'architecture client/serveur sont plus matures que les autres (et plus anciennes).
- L'administration se porte au niveau serveur, les clients ayant peu d'importance dans ce modèle. » Toute la complexité/puissance peut être déportée sur le(s) serveur(s), les utilisateurs utilisant simplement un client léger.
- Recherche d'information : les serveurs étant centralisés, cette architecture est particulièrement adaptée et vélocité pour retrouver et comparer de vastes quantités d'information (moteur de recherche sur le Web).

#### Inconvénients de la communication client/serveur

- Si trop de clients veulent communiquer avec le serveur au même moment, ce dernier risque de ne pas supporter la charge (alors que les réseaux pair à pair fonctionnent mieux en ajoutant de nouveaux participants).
- Si le serveur n'est plus disponible, plus aucun des clients ne fonctionne (le réseau pair à pair continue à fonctionner, même si plusieurs participants quittent le réseau).
- Les coûts de mise en place et de maintenance sont élevés.
- En aucun cas les clients ne peuvent communiquer entre eux, entraînant une asymétrie de l'information au profit des serveurs.