I. INTRODUCTION

Les pages dynamiques et l'accès aux bases de données sont des technologies indispensables au développement d'un site web. Les sites web utilisant les pages dynamiques et les bases de données règnent aujourd'hui en maitre sur internet. Il n'est plus imaginable de développer un site sans faire appel à ces technologies et aux possibilités et de personnalisation qu'elles permettent. Ce chapitre présente et explique la notion de client/serveur et le concept du Web, ainsi que les pages dynamiques.

II. PRESENTATION DE CLIENT /SERVEUR

Le mode client/serveur est un mode de fonctionnement dissymétrique dans lequel deux logiciels différents sont nécessaire pour permettre les communications : un logiciel serveur et un logiciel client, nécessaire sur toutes machines

II.1. Définition

Un environnement client/serveur désigne un mode de communication à travers un réseau informatique entre plusieurs logiciels. Un logiciel client et un logiciel serveur sont reliés par un réseau informatique. Le logiciel client peut envoyer une requête au logiciel serveur.

0II.2. Le client serveur pour web

Dans le cas qui nous intéresse ; c'est-à-dire le web, le client est un navigateur de tels logiciels existent pour tous les systèmes d'exploitation : Les sites web sont héberges sur des serveurs dédies qui sont nommés serveurs web. A l'heure actuelle ; le logiciel le plus répondu est APACHE http server ; et bien sûr il existe d'autres serveurs web.

Le navigateur émet une requête http vers un serveur web afin d'obtenir la page web désirée. Le serveur envoie les données demandées par le client ; si celui –ci est autorisé à accéder au document. Le navigateur interprète les instructions de mise en page contenus dans les données envoyées par le serveur.

II.3. Fonctionnement d'un système client/serveur

Un system client /serveur fonctionne selon le schéma suivant :

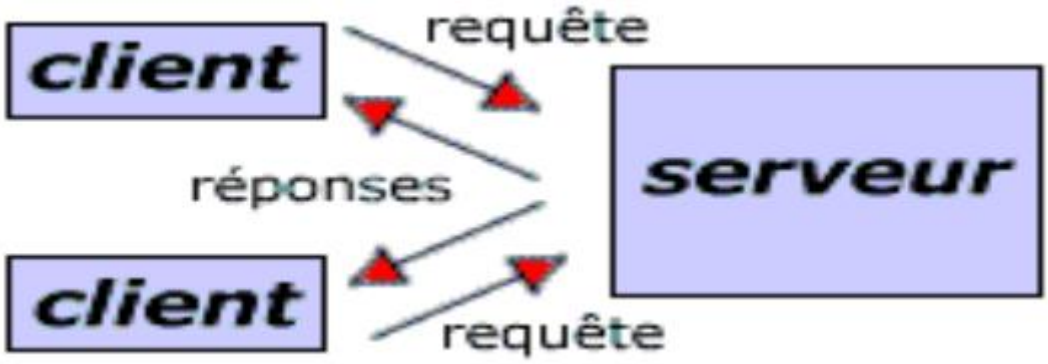

Figure I.1 : Fonctionnement de system client /serveur.

Le client émet une requête vers le serveur grâce à son adresse IP et le port ; qui désigne un service particulier du serveur. Le serveur reçoit la demande et répond à l'aide de l'adresse de la machine client et son port.

II.4. L'architecteur system client/serveur

De nombreuses applications fonctionnent selon un environnement client/serveur. Cela signifie que des machines clients (des machines faisant partie du réseau) contactent un serveur, une machine généralement très puissante en termes de capacité d'entrée-sortie, qui leur fournit des services. Ces services sont des programmes fournissant des données-t-elle que l'heure ; des fichiers, une connexion, etc.

Les services sont exploités par des programmes, appelés programmes client, s'exécutant sur les machines clients. On parle ainsi de client (client FTP, client de messagerie, etc.) lorsque l'on désigne un programme tournant sur une machine cliente, capable de traiter des informations qu'il récupère auprès du serveur (dans le cas du client FTP il s'agit de fichier ; tandis que pour le client de messagerie il s'agit de courrier électronique).

II.4.1. Présentation de l'architecteur à 2 niveaux

L'architecture à deux niveaux (aussi appelée architecture 2-tier, tier signifiant rangée en

anglais) caractérise les systèmes clients/serveurs pour lesquels le client demande une ressource et le serveur la lui fournit directement, en utilisant ses propres ressources. Cela signifie que le serveur ne fait pas appel à une autre application afin de fournir une partie du service.

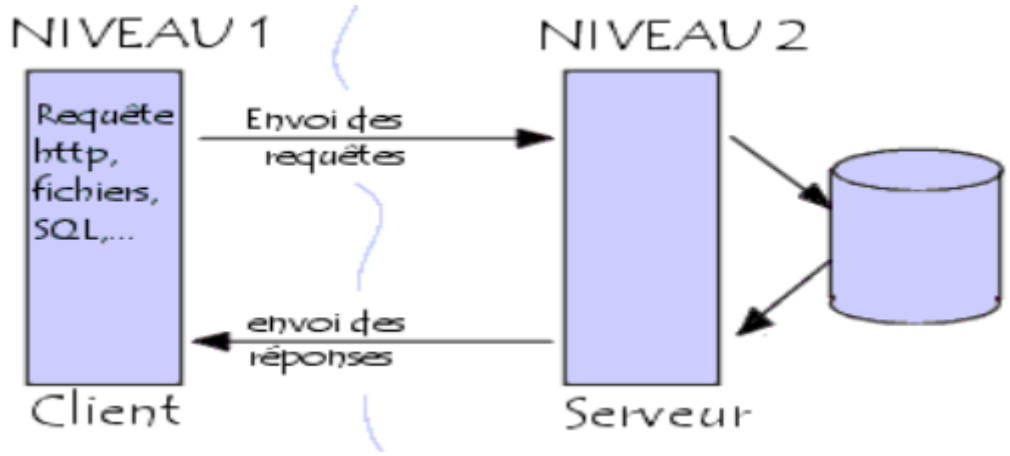

Figure I.2 l'architecteur à 2 niveaux

II.4.2. Présentation de l'architecteur à 3 niveaux

Dans l'architecture à 3 niveaux (appelée architecture 3-tier), il existe un niveau intermédiaire, c'est-à-dire que l'on a généralement une architecture partagée entre : Un client, c'est-à-dire l'ordinateur demandeur des ressources , équipée d'une interface utilisateur (généralement un navigateur web) chargée de la présentation ; Le serveur d'application (appelé également middleware), chargé de fournir la ressource mais faisant appel à un autre serveur Le serveur des données, fournissant au serveur d'application les données dont il a besoin.

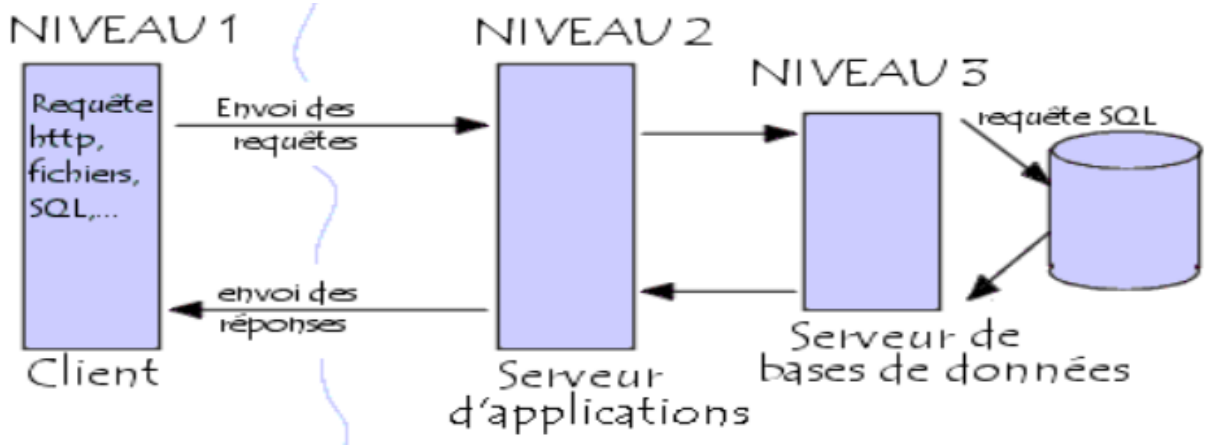

Figure I.2 l'architecteur à 3 niveaux

Etant donné l'emploi massif du terme d'architecture à 3 niveaux, celui-ci peut parfois désigner aussi les architectures suivantes.

II.4.3. Présentation de l'architecteur à N niveaux

L'architecture 3 niveaux permet de spécialiser les serveurs dans une tache précise : avantage de flexibilité, de sécurité et de performance. L'architecture peut être étendue sur un nombre de niveaux plus importants : on parle dans ce cas d'architecture à N niveaux (ou multi-tier).

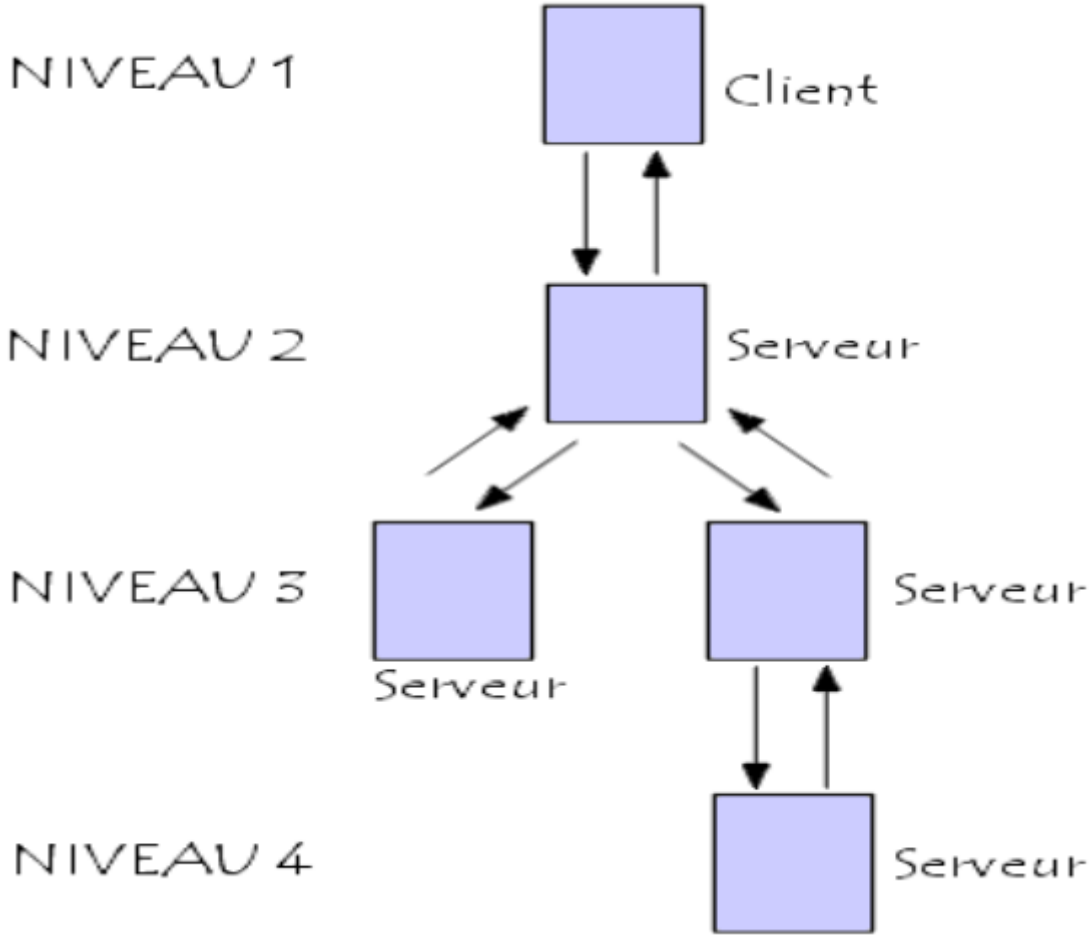

Figure 1.3 l'architecteur à N niveaux

II.5. Avantages de l'architecture client-serveur

✓ ***Des ressources centralisées :*** Toutes les données sont centralisées sur un seul serveur, ce qui simplifie les contrôles de sécurité, l'administration, la mise à jour des données et des logiciels.

Les technologies supportant l'architecture client-serveur sont plus matures que les autres. La complexité du traitement et la puissance de calculs sont à la charge d'où des serveurs, les utilisateurs utilisant simplement un client léger sur un ordinateur terminal qui peut être simplifié au maximum.

✓ ***Recherche d'information :*** les serveurs étant centralisés, cette architecture est particulièrement adaptée et véloce pour retrouver et comparer de vaste quantité d'informations (moteur de recherche sur le Web), ce qui semble être rédhibitoire pour le P2P beaucoup plus lent, à l'image de Free net.

II.6. Inconvénients de l'architecture client-serveur

Si trop des 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 peuvent être élevés. En aucun cas les clients ne peuvent communiquer entre eux, entrainant une asymétrie de l'information au profit des serveurs.

III. PRESENTATION DU WEB

Le World Wide est Web est rapidement devenu le service le plus utilisé sur l'Internet. Il a conçu le Hypertext Markup Language (HTML) à partir d'un autre format utilisé pour les documents appelé le SGML(**Standard Generalized Markup Language**). Le WWW fonctionne en utilisant le concept d'hypertexte. À l'intérieur d'une page, il y a des mots clés ou des images qui ont des liens qui, lorsque vous cliquez dessus, vous amènent à une autre page Web.

III.1. Historique d'Internet

L'Internet est un système de communication qui permet aux ordinateurs autour du monde de

communiquer et de s'échanger de l'information entre eux. Internet est né en 1969 sous l'impulsion du département américain de la défense (DOD). Le réseau, qui s'appelait alors ARPANET, devait assurer les échanges d'informations électroniques entre les centres névralgiques américains dans le contexte de la guerre froide. Le cahier de charge établi par le DOD imposait que le réseau puisse poursuivre ses activités en cas d'attaque nucléaire soviétique. Si l'un ou plusieurs des sites et lignes de connexion venait à être détruit, les messages parviendraient à leur destinataire par des itinéraires alternatifs. Un grand nombre de centres de recherche, militaires, publics et privés prirent part à ce projet. Il était normal que leurs réseaux internes furent les premiers reliés à Internet. C'est pourquoi, dès sa création, Internet sera un méta-réseau, un réseau de réseaux qui va peu à peu relier la communauté scientifique et universitaire mondiale Internet arrive en Europe en 1982. L'année 1984 est une année charnière : Internet perd son caractère militaire. Son financement n'est plus assuré par le DARPA mais par un organisme scientifique civil créé deux ans plus tard : La National Science Foundation (NSF). Le réseau est scindé en deux parties : MILnet, réseau strictement militaire et NSFnet, le backbone ou épine dorsale d'Internet. Sa facilité d'utilisation contribue grandement à d'populariser les autoroutes de l'information World Wide Web apparaît l'année suivante. Depuis la chute du mur de Berlin en 1989, Internet s'est largement ouvert au grand public et à l'exploitation commerciale.

III.2. Le WEB

Dans les années 90, un nouveau service de l'Internet est apparu : le World Wide Web, la toile d'araignée mondiale, encore désignée par l'acronyme WWW ou le diminutif Web. C'est ce service qui assure un certain succès à l'Internet. L'idée est de lire des hyperdocuments à l'aide d'un navigateur. Un hyperdocument est un document électronique contenant des images, du son, du texte, parfois des petits morceaux de programme, mais surtout des liens vers d'autres hyperdocuments : des liens hypertextes. Ces liens apparaissent dans un style qui les distinguent, et une simple action de la souris sur un lien suffit à ouvrir le document lié. Les documents peuvent se trouver sur n'importe quelle machine (serveur) de l'Internet à des endroits parfois très éloignés et c'est ce qui donne l'impression à l'utilisateur de naviguer sur le réseau. Le navigateur est l'outil qui permet de lire les hyper documents. On l'appelle aussi browser et les deux plus connus aujourd'hui sont MicroSoft Internet Explorer (MSIE) et Netscape. Au début conçu pour ne lire que les hyper documents, le navigateur intègre aujourd'hui tous les services de l'Internet (e-mail, ftp,...) Le navigateur désigne par une adresse URL (Uniform Resource Locator), les adresses complètes de l'Internet. C'est une adresse qui

contient à la fois le nom d'une machine mais aussi le nom du service demandé, le nom d'un document,...

Un autre standard incontournable de l'Internet est HTML (HyperText Markup Language). C'est le langage qui permet d'écrire des hyperdocuments de façon descriptive à l'aide de marqueurs.

III.3. Standards du web

Le web repose trois standards : les URL, http ET HTML , hypertext

III.3.1. URL

Une URL (Uniform Resource Locator) est une simple ligne de texte qui permet de retrouver une ressource (texte, image, musique, vidéo, programme...) sur internet. Il s'agit d'une chaîne de caractères ASCII imprimables qui se décompose en cinq parties :

✓ **Le nom du protocole** : c'est-à-dire en quelque sorte le langage utilisé pour communiquer sur le réseau. Le protocole le plus largement utilisé est le protocole HTTP (HyperText Transfer Protocol), le protocole permettant d'échanger des pages Web au format HTML. De nombreux autres protocoles sont toutefois utilisables (FTP, News, Mailto, Gopher, ...)

✓ **Identifiant et mot de passe** : permet de spécifier les paramètres d'accès à un serveur sécurisé. Cette option est déconseillée car le mot de passe est visible dans l'URL

✓ **Le nom du serveur** : Il s'agit d'un nom de domaine de l'ordinateur hébergeant la ressource demandée. Notez qu'il est possible d'utiliser l'adresse IP du serveur, ce qui rend par contre l'URL moins lisible.

✓ **Le numéro de port** : il s'agit d'un numéro associé à un service permettant au serveur de savoir quel type de ressource est demandée. Le port associé par défaut au protocole est le port numéro 80. Ainsi, lorsque le service Web du serveur est associé au numéro de port 80, le numéro de port est facultatif

✓ **Le chemin d'accès à la ressource** : Cette dernière partie permet au serveur de connaître l'emplacement auquel la ressource est située, c'est-à-dire de manière générale l'emplacement (répertoire) et le nom du fichier demandé Une URL a donc la structure suivante :

Protocole	Mot de passe (facultatif)	Nom du serveur	Port(facultatifsi 80)	Chemin
http://	user:password@	www.ccm.net	:80	/glossair/glossair.php3

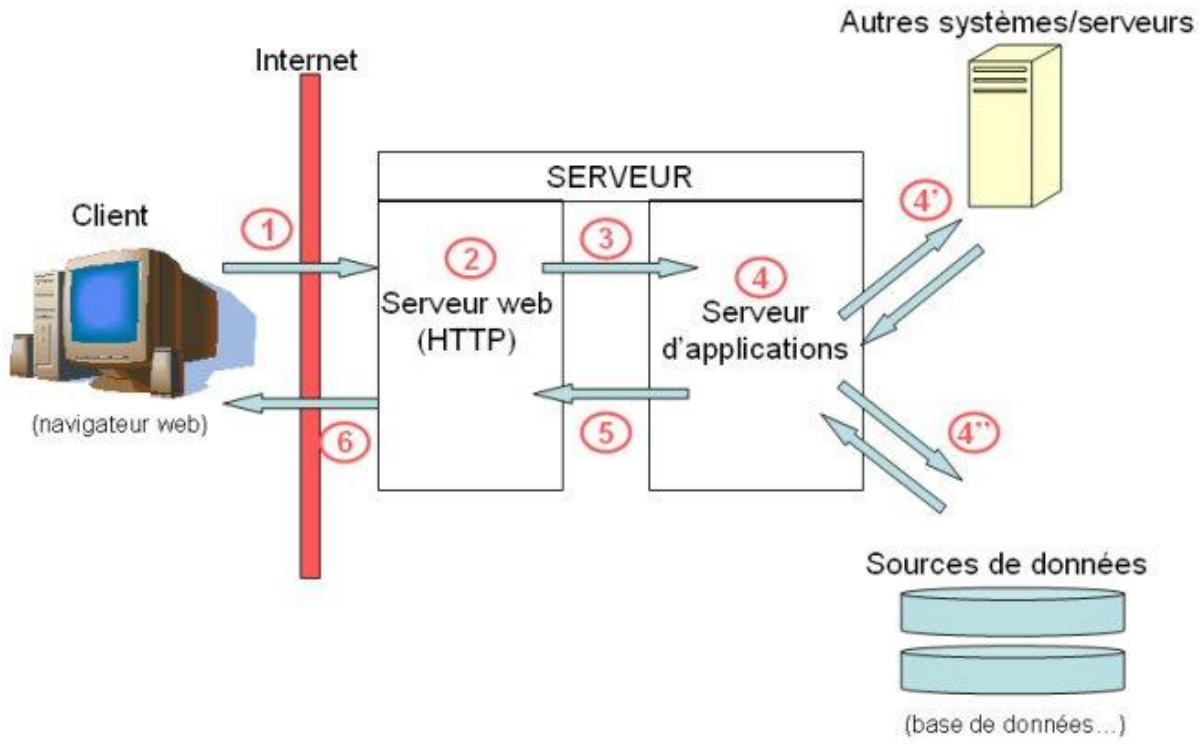

Figure 1.4 Déroulement d'une requête complète

- 1) Le client émet une requête (i.e. appelle une URL) pour demander une ressource au serveur. Exemple : **http://leserveur.com/welcome**. Il ne sait pas ici si la réponse qui va lui parvenir est statique (page HTML simple) ou dynamique (générée par une application web). Dans notre cas, il s'agit d'une application répondant à l'adresse **welcome** sur le serveur **leserveur.com**.
- 2) Côté serveur, c'est le serveur web (exemple : Apache) qui traite les requêtes HTTP entrantes. Il traite donc toutes les requêtes, qu'elles demandent une ressource statique ou dynamique. Seulement, un serveur HTTP ne sait répondre qu'aux requêtes visant des ressources statiques. Il ne peut que renvoyer des pages HTML, des images,... existantes.
- 3) Ainsi, si le serveur HTTP s'aperçoit que la requête reçue est destinée au serveur d'applications, il la lui transmet. Les deux serveurs sont reliés par un canal, nommé connecteur.
- 4) Le serveur d'applications (exemple : Tomcat !) reçoit la requête à son tour. Il est, lui, en mesure de la traiter. Il exécute donc le morceau d'application (la servlet) auquel est destinée la requête, en fonction de l'URL. Cette opération est effectuée à partir de la configuration du serveur. La servlet est donc invoquée, et le serveur lui fournit

notamment deux objets Java (Tomcat est un serveur d'applications Java) exploitables : un représentant la requête, l'autre représentant la réponse. La servlet peut maintenant travailler, et générer la réponse à la demande. Cela peut passer par la consultation de sources de données, comme des bases de données (4" sur le schéma). Ou bien par l'interrogation d'autres serveurs ou systèmes (4' sur le schéma), l'environnement Java web permettant de se connecter à de nombreux systèmes.

- 5) Une fois sa réponse générée, le serveur d'applications la renvoie, par le connecteur, au serveur web. Celui-ci la récupère comme s'il était lui-même allé chercher une ressource statique. Il a simplement délégué la récupération de la réponse, et celle-ci a été générée, mais ce n'est plus le problème.
- 6) La réponse est dorénavant du simple code HTML, compréhensible par un navigateur. Le serveur HTTP peut donc retourner la réponse au client.

III.3.2. Protocole http

L'HTTP (Hypertext Transfer Protocol) est le protocole de transport utilisé par les navigateurs Web (Firefox, Internet Explorer...) et les serveurs Web (Apache, IIS...) pour communiquer entre eux. C'est lui qui est utilisé par exemple pour obtenir un fichier HTML, une image, poster un formulaire Internet. Il est donc au cœur de l'Internet. Techniquement c'est un protocole texte (donc lisible en clair) s'appuyant les protocoles plus bas-niveau TCP/ IP. Avec HTTP la communication entre un navigateur et un serveur Web est finalement assez simple. En voici le déroulé : Une URL telle que http://www.monsite.com/fichier.html est donnée au Navigateur par un internaute.

Le navigateur en extrait le nom de domaine 'www.monsite.com' et à partir de cette information sait comment trouver le serveur Web distant (grâce à une opération dite de DNS look up qui résout un nom de domaine en une adresse IP) ,à partir de là une connexion (basée sur les protocoles TCP/IP) est établie entre le navigateur et le serveur Web distant. Une requête HTTP demandant la ressource '/fichier.html' est alors transmise par le navigateur, le serveur Web trouve la ressource correspondante et en renvoie le contenu dans une réponse http, le navigateur est désormais capable d'afficher le fichier HTML à l'internaute.

III.3.3. Pages web dynamique

Le principe d'un page dynamique est d'être construite à la demande (à la volée) par le serveur (côté serveur), en fonction de critères spécifiques. La présentation et le contenu affichés peuvent ainsi être personnalisés de manière interactive, en fonction des produits, des internautes, des langues, etc.

On reconnaît facilement un page dynamique grâce à l'URL qui s'affiche dans le navigateur web de l'utilisateur :

✓ **Page statique :** affiche la page accueil.htm, stockée telle quelle sur le serveur,

✓ **Page dynamique :** affiche la page accueil.php en demandant au serveur d'afficher le contenu de cette page en français.

Alors que les pages statiques font appel au html, langage de description de données, les pages dynamiques sont mises en œuvre grâce à un langage de programmation. Grâce à lui, on pourra disposer d'instructions conditionnelles, des boucles et des fonctions de traitement complexes. Le langage de programmation variera en fonction de la technologie retenue (PHP, ASP, Java, etc.).

III.3.4. Hypertexte

Le système hypertext est un système contenant des nœuds liés entre eux par des hypertextes est donc un document qui contient des hyperliens et des nœuds. Un nœud est « une unité minimale d'information », notion assez floue qui signifie simplement que l'information d'un nœud sera toujours présentée entière

Lorsque les nœuds ne sont pas uniquement textuels ; mais aussi audiovisuels ; on peut parler de système et de document hypermedias.

III.4. Les technologies dynamiques du coté serveur

Le serveur web est un ensemble ordinateur/logiciel paramétré pour pouvoir traiter certains types de pages et notamment celles qui contiennent des instructions de programmation. Il reconnaît ces pages grâce à l'URL qu'il reçoit, effectue les traitements demandés et transmet le résultat au format html au browser de l'internaute.

III.4.1. Active server pages(ASP)

Active Server Pages (ASP) est un ensemble des logiciels développés par Microsoft et utilisés dans la programmation Web.

C'est une suite des logiciels destinées à créer des sites web dynamiques. Elle nécessite pour fonctionner une plate-forme Windows avec IIS installé, ou encore une plate-forme Linux ou Unix avec une version modifiée d'Apache. ASP est une structure composée d'objets accessibles par deux langages principaux : le VBScript et le JScript. Il est possible d'utiliser d'autres langages comme le PerlScript, le REXX, ou encore le Python en ajoutant le moteur d'interprétation du langage adéquat à IIS.

À l'inverse de certains langages de programmation (C, C++), cette technologie n'utilise pas de langages compilés, mais des langages interprétés.

III.4.2. Java Server Pages(JSP)

Le Java Server Pages ou JSP est une technique basée sur Java qui permet aux développeurs de créer dynamiquement du code HTML, XML ou tout autre type de page web. Cette technique permet au code Java et à certaines actions prédéfinies d'être ajoutés dans un contenu statique. Il s'agit en réalité d'un langage de script puissant exécuté du côté du serveur (au même titre que les scripts CGI,PHP,ASP ..) et non du côté client.

III.4.3. Comun Gateway Interface(CGI)

Un script CGI (interface de passerelle commune) est un programme exécuté par le serveur web (on dit généralement « côté serveur »), permettant d'envoyer au navigateur de l'internaute un code HTML créé automatiquement par le serveur. Un des principaux intérêts de l'utilisation de CGI est la possibilité de fournir des pages dynamiques, c'est-à-dire des pages personnalisées selon un choix ou une saisie de l'utilisateur. L'application la plus fréquente de cette technique repose sur l'utilisation de formulaires HTML permettant à l'utilisateur de choisir ou de saisir des données, puis de cliquer sur un bouton de soumission du formulaire, envoyant alors les données du formulaire en paramètre du script CGI.

III.4.4. PHP

PHP est un langage de programmation informatique essentiellement utilisé pour produire à la volée des pages web dynamiques, PHP s'est imposé comme le langage de référence sur le web en raison de sa simplicité, de sa gratuité et de son origine de logiciel libre. Il très puissant, rapide

et principalement exécuté par le compilateur PHP. Un script PHP est multiplateforme, très bon support des bases de données (Oracle, Microsoft, MySQL).

III.4.5. MYSQL

MySQL (My Structured Query Language) est un Système de Gestion des Bases des données (SGBD) Open Source très rapide, robuste et multiutilisateur. Le serveur MySQL supporte le langage de requêtes SQL, langage standard de choix des SGBD modernes. Il est facilement accessible en réseaux et supporte des connexions sécurisées grâce au protocole SSL. La portabilité du serveur MySQL lui permet de s'exécuter sur toutes les plateformes et d'être intégré à plusieurs serveurs web.

IV. Les éléments et les balises

Les éléments

HTML est un *markup language* (langage de marquage ?). Il permet d'enrichir un texte avec des informations structurelles, sémantiques et de présentation. Le principe de HTML, commun à ce type de langages (SGML pour *Standart Generalized Mark-up Language*), consiste à utiliser des *éléments* délimités par des *balises*.

En fait, **HTML** est un langage de type **SGML**, correspondant à un ensemble particulier d'éléments pour décrire des pages destinées au World Wide Web.

Les **éléments** permettent d'associer à différents blocs (texte, images...) les informations structurelles, sémantiques et de présentation désirée. Les principaux éléments HTML permettent de définir des liens hypertextes, des titres, des paragraphes, des listes, des tableaux, des images, et cætera...

Les balises

Les éléments sont délimités par des **balises**. On place une balise de début avant le contenu de l'élément et une balise de fin après. Mais dans certains cas, la balise de fin n'est pas nécessaire.

Une balise est formée en encadrant le nom de l'élément avec les symboles < et > (soit `<element>` où *element* est le nom de l'élément). Pour une balise de fin on ajoute le caractère `/` avant le nom de l'élément (soit `</element>`. Dans l'exemple suivant on définit un titre ou en-tête principal (élément `H1`). Seul le texte balisé constitue l'élément H1. Les navigateurs courants utilisent pour ce genre d'éléments une typographie plus forte.

```
<H1>L'estuaire de Seine</H1>
Lieu magique, rencontre du fleuve et de la mer, un environnement unique,
l'estuaire de Seine est aussi un lieu de communication important...
```

Ce code produit l'affichage suivant dans un navigateur graphique :

L'estuaire de Seine

Lieu magique, rencontre du fleuve et de la mer, un environnement unique, l'estuaire de Seine est aussi un lieu de communication important...

Enfin, un certain nombre de balises n'ont pas besoin de marqueur de fin. La fin de l'élément sera déterminée automatiquement par le logiciel de navigation. C'est le cas en particulier de l'élément `P` qui indique un nouveau paragraphe. Le code précédent serait d'ailleurs plus correct en ajoutant une balise `<P>` après le titre :

```
<H1>L'estuaire de Seine</H1>
<P>Lieu magique, rencontre du fleuve et de la mer, un environnement unique,
l'estuaire de Seine est aussi un lieu de communication important...
```

Les attributs

Les attributs permettent d'apporter certaines précisions à des éléments. Il peut s'agir par exemple d'un nom de fichier (pour placer une image) ou d'une référence à une adresse HTML (lorsqu'on crée un lien). On place les attributs dans la balise de début de l'élément concerné. La syntaxe est simple : "nom de l'attribut"="valeur de l'attribut".

Dans l'exemple suivant, on place une image (élément `IMG`). Deux attributs permettent l'un de déterminer le nom du fichier contenant l'image et l'autre de placer une brève description de cette image.

```
<IMG src="bateau01.jpg" alt="Un bateau">
```

Remarque 1 : l'élément `IMG` n'a pas besoin de balise de fin.

Remarque 2 : l'attribut `alt` de l'élément `IMG` est indispensable pour construire des pages accessibles. Il a d'ailleurs été rendu obligatoire par la norme HTML 4.0.

Jeu de caractères

Pour différentes raisons le jeu de caractère utilisé pour stocker les documents HTML est le code ASCII standard sur 7 bits. Ce code permet de décrire le jeu de 128 caractères de base comprenant les lettres majuscules et minuscules, les chiffres et les signes de ponctuation. Les caractères accentués sont codés dans des jeux de caractères étendus à 8 bits.

La principale raison qui aujourd'hui oblige à continuer d'utiliser le code ASCII 7 bit et non un code étendu est que tous les ordinateurs n'utilisent pas les mêmes codes étendus (caractères 128 à 255). Les machines sous Windows et la plupart des machines Unix utilisent maintenant le code dit ISO-LATIN1, mais les PC sous DOS, les MACintosh et différents autres types d'ordinateurs utilisent d'autres codes.

Pour écrire des documents susceptibles d'être consultés sans problèmes quelque soit le type de système du client, il est nécessaire de coder les caractères accentués en utilisant un codage un peu particulier. On indique entre des caractères `&` et `;` la lettre et l'accent. Ainsi le caractère `é` doit-il être représenté par la chaine `&eacute;`.

Mais les caractères accentués ne sont pas les seuls à être codés de cette façon. Le caractère & lui-même doit être codé de façon particulière sinon comment différencier les cas où on l'utilise pour lui-même de ceux où il débute un caractère spécial ? De même les < et > pourraient être confondus avec le début où la fin d'une balise.

Voici quelques exemples parmi les plus fréquement utilisés.

```
&eacute; é      &agrave; à      &lt;   <
&egrave; è      &acirc;  â      &gt;   >
&ecirc;  ê      &ugrave; ù      &quot; "
&euml;   ë      &ccedil; ç      &amp;  &
&Eacute; É      &Ccedil; Ç        espace insécable
```

Structure d'un document

Le squelette de base que l'on retrouvera dans tout document HTML est le suivant :

```
<HTML>
<HEAD>

    en-tête du document

</HEAD>
<BODY>

    corps du document

</BODY>
</HTML>
```

Dans l'en-tête, on définira obligatoirement un titre à l'aide de l'élément `TITLE`.

Voir également les attributs de la balise `BODY` pour la définition des couleurs du document et d'une éventuelle image de fond.

Pour être plus rigoureux, la première ligne d'un document HTML doit mentionner un *DOCTYPE*, c'est-à-dire le *langage HTML* que l'on va utiliser dans la suite. Les variations possibles sont :

- respecter ou pas les contraintes d'un langage XML : si oui, il s'agit de `XHTML`, sinon de `HTML 4.01` ;
- se restreindre aux éléments purement sémantiques ou s'autoriser les éléments de mise en forme : dans le premier cas on fait du `Strict`, sinon du `Transitional`.

Une autre précision important est l'encodage utilisé pour les caractères : il pourrait s'agir de `latin1`, d'`utf8`, etc.

Finalement, si l'on fait le choix d'un HTML avec contraintes XML, éléments de mise en forme et codage des caractères en UTF8, le squelette du document devient :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
                      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

    en-tête du document avec un title

</head>
<body>

    corps du document

</body>
</html>
```

.