Technologies pour les applications en réseau : contribution au profil NetDevOps

RSX102

Document provisoire.
Copie et diffusion non autorisées sans accord écrit.
Documents liés aux cours : https://rsx102.seancetenante.com

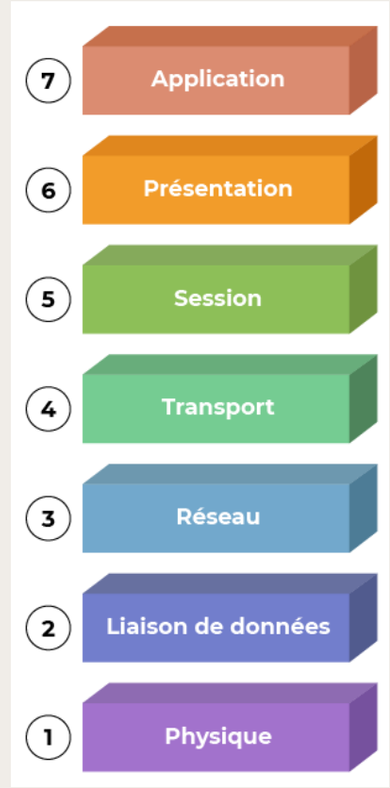
2026 *François Lacomme* <*francois.lacomme@2isa.net*>

Présentation de l'UE

Contenu et organisation

- **Contenu**
  - Cette unité d'enseignement, RSX102, **Technologies pour les applications en réseau**, (Technologies pour les applications client-serveur jusqu'en 2020), concerne donc :
  - Les couches hautes du modèle OSI (Open System Interconnection)
  - Les applications C/S (client/serveur) et distribuée dans l'architecture internet
  - Voir le plan des cours et TP dans la partie *documents* des pages rsx102.seancetenante.com

- **Rappel**
  - le modèle OSI est illustré en annexe.
  - Les couches supérieures d'OSI de l'ISO ont été décrites dans le cours UTC505. Elles sont rapidement rappelées ci-dessous.



Préambule

Rappel

- **La couche session (Session layer)**
  - Établir des sessions pour des utilisateurs travaillant sur différents postes informatiques.
  - Gestion de dialogues (gestion d'un jeton de parole : seul le processus qui possède le jeton peut réaliser une opération critique).
  - Synchronisation par gestion de points de reprise
  - La notion de session est souvent associée à l'utilisation de base de données ou de transfert de fichiers.

- **La couche présentation (Presentation layer)**
  - Syntaxe et sémantique de l'information transmise.
  - Encodage et décodage de données suivant une norme reconnue (Unicode, MIME, HTML, ASN.1/BER, *Abstract Syntax Notation-One/Basic Encoding Rule*, etc.)

Préambule

Rappel

- **La couche application (Application layer)**
  - C'est le point d'accès aux services réseaux.
    - Normes et protocoles proches de l'utilisateur de services communicants.
    - Bibliothèques et API (packages, librairies) d'accès à des services tels que :
      - Transferts et systèmes de fichiers
      - Courrier électronique
      - Messagerie instantanée
      - VoIP, ToIP ; visioconférence
      - Exécution de travaux ou de sessions à distance
      - Consultation et gestion d'annuaires

  - Dans l'architecture liée à internet, on considère plutôt :
    - La couche **application** de l'architecture TCP/IP
    - Pour le modèle lié à internet, on regroupe sous le terme de couche Application les couches session, présentation et application d'OSI (les couches supérieures d'OSI).

1 - Introduction

1.1 - Définitions

- **Client-serveur**
  - Mode de communication à travers un réseau entre plusieurs programmes ou logiciels :
    - l'un, qualifié de **client**, envoie des requêtes ;
    - l'autre, qualifié de **serveur**, attend les requêtes des clients et y répond.
  - Principales architectures
    - **Architecture à deux niveaux**, *two-tier architecture*. Cf. figure ci-contre.
    - **Architecture à 3 niveaux**, *three-tier architecture*.
      Ex. : un client web demande une ressource à un serveur web associé à un serveur d'application qui est le client d'un serveur de données.
    - ***Architecture pair à pair**, peer-to-peer…*

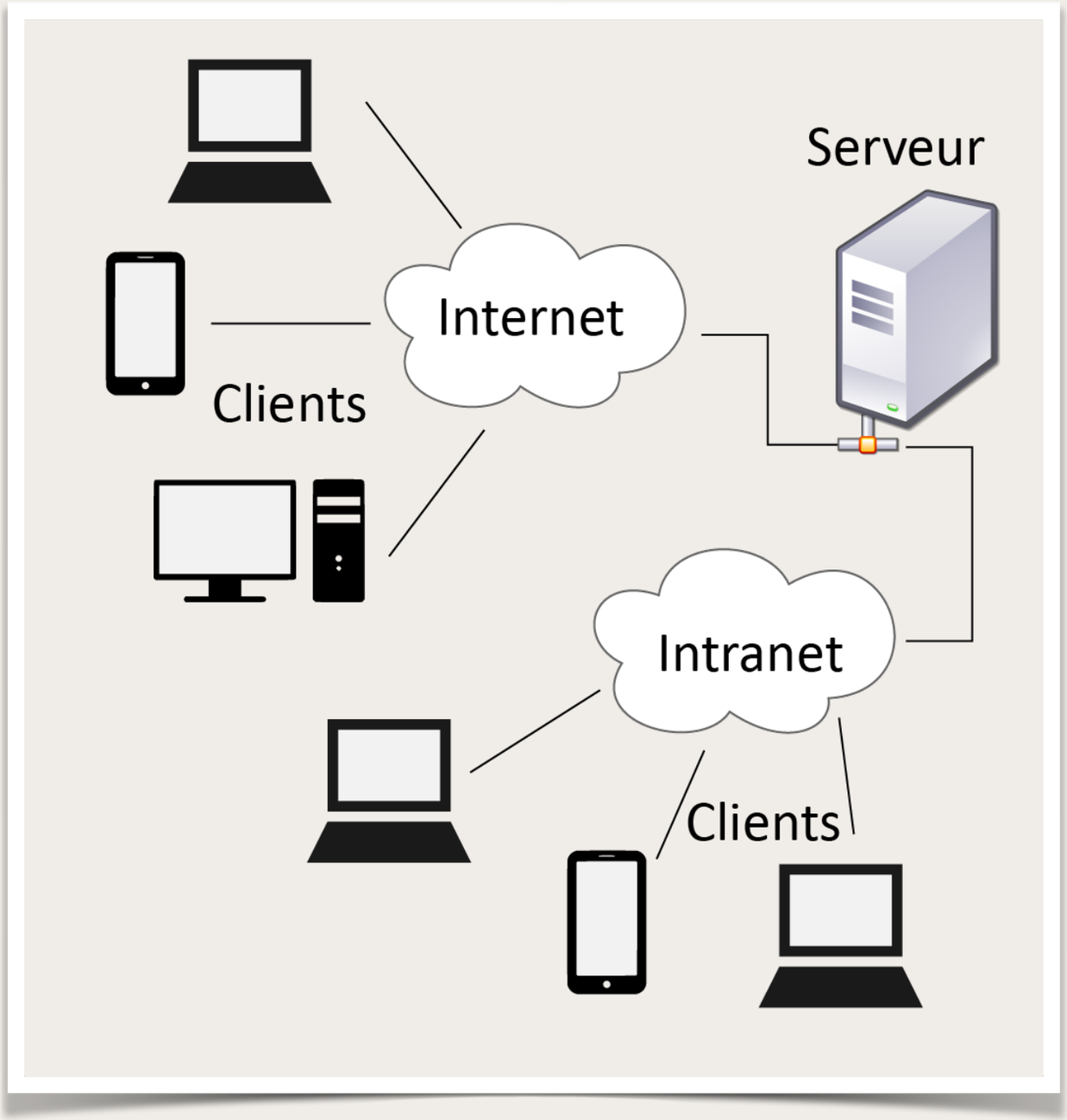


Fig 1.1 - Architecture Client-Serveur ; Internet et Intranet

1 - Introduction

1.1 - Définitions

- **P2P**
  - Une architecture **pair à pair** (***peer-to-peer*** ou P2P en anglais) est un environnement client–serveur où chaque programme connecté est susceptible de jouer **tour à tour** ou **à la fois** le rôle de client et celui de serveur.
  - Les composants d'un système P2P sont appelés **nœuds**, **pairs** ou **utilisateurs**.
  - Certains systèmes sont :
    - partiellement centralisés ; un serveur intermédiaire gère une partie des échanges
    - totalement décentralisés ; sans infrastructure particulière

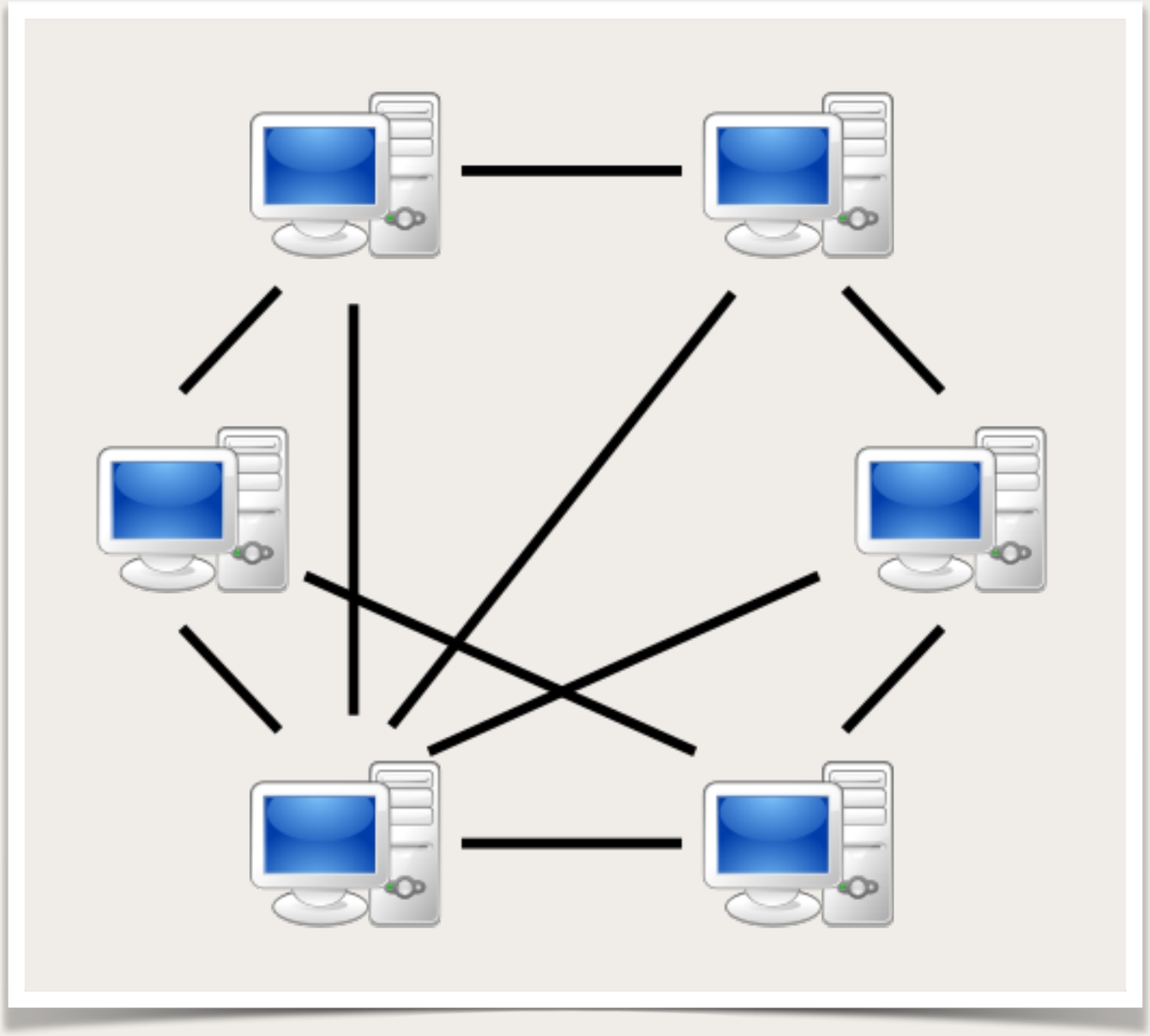


Fig 1.2 - Architecture P2P ou pair à pair

Annexe

✤ Introduction

- ★ DNS, *Domain Name System* (Système de Nom de Domaine), mis en place en 1988, recouvre deux notions :
  - ★ Une **base de données répartie**, grâce à un grand nombre de serveurs de noms qui communiquent entre eux.
  - ★ Un **protocole**, de niveau application :
    - ★ Il utilise UDP pour le transport
    - ★ Le port 53 est utilisé par convention.
  - ★ *Domain Name System* sert notamment à traduire un nom de domaine en adresse IP, ou en d'informations d'autres types.
  - ★ Voir STD 13, RFC 1034 (*Domain names - concepts and facilities)*, RFC 1035 (*Domain names - implementation and specification)*, etc.

★ Service de noms

- ★ Un service de noms permet aux utilisateurs d'accéder à une ressource en la désignant par son nom, plutôt que par son adresse.
- ★ DNS est un service de nommage standard sur internet (RFC 1033 à 1035).
- ★ Un des objectifs est donc de retrouver **l'adresse IP** d'une machine à partir de **son nom de domaine**.

Nom de domaine

- Il résulte d'un système hiérarchique :
  - Domaine de haut-niveau (TLD, ***Top Level Domain***) ou domaine de premier niveau
    - **Générique** : gTLD ; *generic TLD*
      Ex. : .com, .net, .org, .mil, .gov, .biz, .info, .name, .pro, .aero, .coop, .xxx, .museum…

      Depuis 2014, des milliers de nouveaux gTLD (nommés nTLD pour **News TLD**) peuvent être demandés, avec priorité aux propriétaires de marques déposées.
      Ex. : .paris, .bzh, .sncf, .guru…
    - **National** : ccTLD ; *country code TLD*
      Ex. : .fr, .uk, .nl, .it, .jp, .eu, .us…
  - Sous-domaine ou ***label*** : chaque sous-domaine est enregistré auprès du domaine supérieur.
    Il faut donc :
    - la validation du domaine supérieur
    - ajouter un enregistrement dans les serveurs de noms du domaine supérieur.
  - Par exemple, dans le domaine `media.nperf.com`, `media` est un sous-domaine de `nperf.com`.

Annexe

Nom de domaine

- Les TLD sont gérés par l'ICANN (Internet Corporation for Assigned Names and Numbers) ; l'ICANN délègue la gestion de chaque TLD à un organisme appelé registre de haut-niveau (*registry*).
- L' AFNIC (l'Association Française pour le Nommage Internet en Coopération) est le registry pour la France. Elle gère les ccTDL .fr, .re, .yt, .wf, .tf et .pm
- VeriSign est le *registry* qui gère les .com, .net, .org…
- Exemple 1

simon.info.arcep.fr

fr = ccTLD pour la France ; géré par l'AFNIC

arcep = Domaine de 2d niveau, enregistré et validé par l'AFNIC (propriétaire : l'Autorité de Régulation des Communications Electroniques et des Postes)

info = Service informatique enregistré pour l'ARCEP.

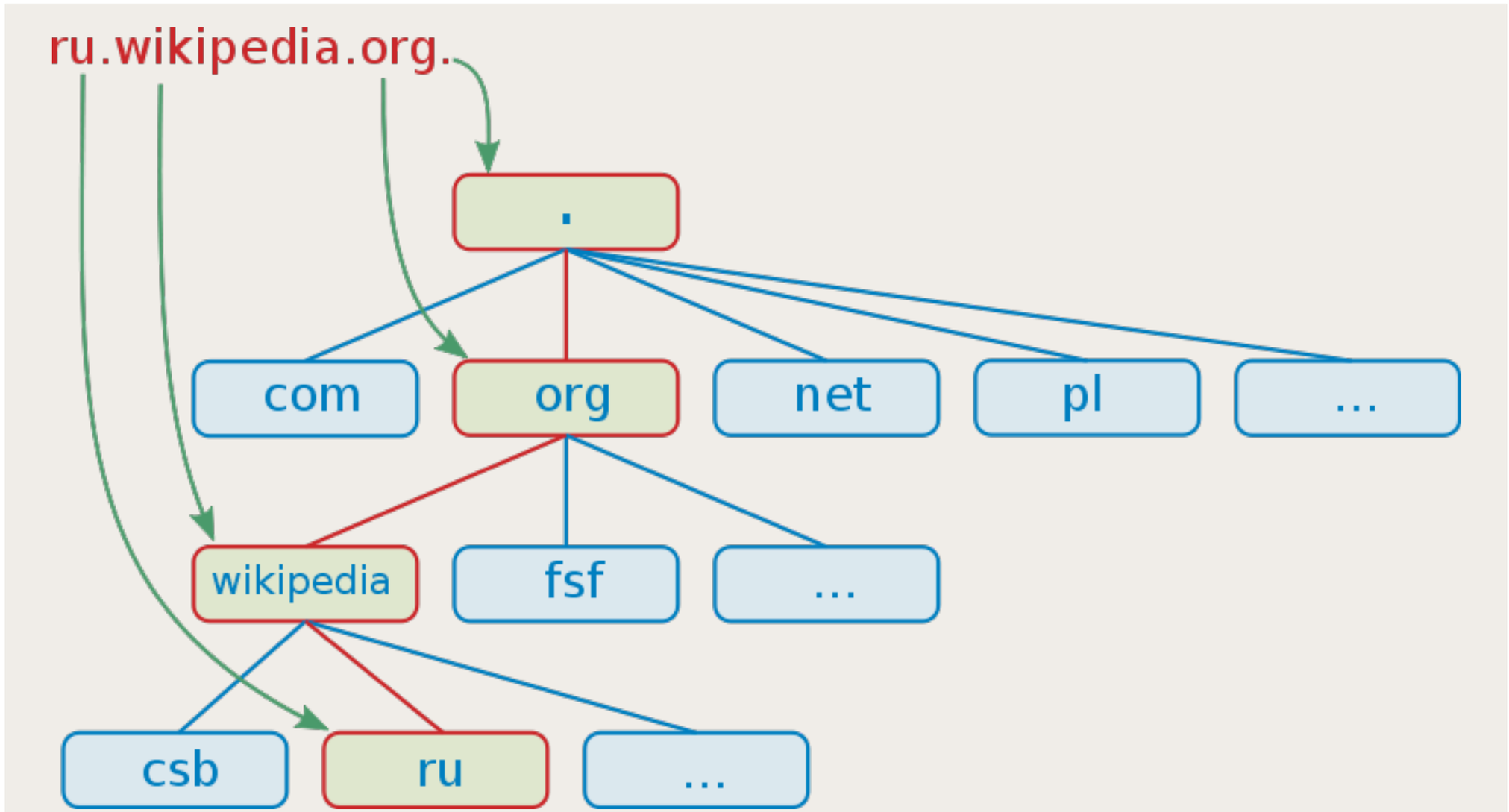
simon = Nom d'une machine (celle de Simon) au sein du service informatique de l'ARCEP.

- **Nom de domaine** (suite…)
  - Exemple 2

```
ftp.aecnam.asso.fr
fr =          ccTLD pour la France ; géré par l'AFNIC
aecnam.asso = Domaine composé d'une association enregistré et validé par l'AFNIC
ftp =         Nom d'une machine (ou d'un service) de l'association.
```

- Dans l'exemple 2 ci-dessus, **aecnam** est un élément de nom de domaine ou ***label***.
- Un élément d'un nom de domaine a au maximum 63 caractères.

- **Nom de domaine** (suite…)
  - Exemple 3 - La hiérarchie du domaine « ru.wikipedia.org. »



Racine

TLD

SLD
Second level domain

Fig 2.1 - Hiérarchie de domaine

Annexe

FQDN

✤ FQDN : Fully Qualified Domain Name

★ Un nom de domaine pleinement qualifié (FQDN) est un nom de domaine écrit de façon absolue.
Il comporte donc :

- tous les sous-domaines (ou labels),
- le domaine de premier niveau (TLD),
- et il est ponctué par un point final qui représente la racine.

★ Exemple : `ru.wikipedia.org.`

Avec un point final !

✤ Enregistrer un nom de domaine

★ On utilise un **bureau d'enregistrement** de noms de domaine, ***registrar*** en anglais, qui gère la réservation de nom de domaine, conformément aux règles imposées par les registres de haut-niveau (***registry***).

★ Voir :

- www.gandi.net, www.ovh.com/fr/domaines/, etc.
- ou l'annuaire des bureaux d'enregistrement de l'AFNIC : www.afnic.fr/fr/votre-nom-de-domaine/comment-choisir-et-creer-mon-nom-de-domaine/

Annexe

Enregistrement de ressources

- **Enregistrement de ressources**
  - Un enregistrement de ressources, ***resource record***, se compose de **cinq éléments** :

    `Nom de domaine ; Durée de vie ; Classe ; Type ; Valeur`

  - `Nom de domaine` : un **FQDN**, qui se termine donc par un point (.), symbole de la racine des domaines. Ex. : `simon.info.arcep.fr.`
  - `Durée de vie` en secondes
  - `Classe` : IN pour Internet
  - `Type` : type d'enregistrement
    - A = Address Record : la valeur est l'adresse IP v4 du nom de domaine
    - AAAA = Address Record : la valeur est l'adresse IP v6 du nom de domaine
    - MX = Relai de messagerie
    - SOA = Start of Authority : serveur principal d'une zone
    - NS = Serveur de noms
    - CNAME = nom canonique : Alias de nom de domaine
    - PTR = Pointeur
    - HINFO = description de l'hôte
    - TXT = Texte de commentaire
  - `Valeur` : en fonction du type

Enregistrement de ressources

- Exemple :

Nom de domaine	*Durée de vie*	*Classe*	*Type*	*Valeur*
fr.wikipedia.org.	575	IN	CNAME	text.wikimedia.org.
text.wikimedia.org.	1522	IN	CNAME	text.esams.wikimedia.org.
text.esams.wikimedia.org.	2193	IN	A	91.198.174.232

Annexe
dig

✤ Exemple avec l'utilisation de dig :

★ *dig* est une commande du package dnsutils sous Linux pour **interroger** des serveurs DNS. Les réponses sont des **enregistrements de ressources**.

```
iMac-F:~ francois$ dig fr.wikipedia.org

; <<>> DiG 9.6.0-APPLE-P2 <<>> fr.wikipedia.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 28507
;; flags: qr rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;fr.wikipedia.org.   IN   A

;; ANSWER SECTION:
fr.wikipedia.org.       575     IN  CNAME   text.wikimedia.org.
text.wikimedia.org.     1522    IN  CNAME   text.esams.wikimedia.org.
text.esams.wikimedia.org. 2193 IN   A    91.198.174.232

;; Query time: 42 msec
;; SERVER: 212.27.40.241#53(212.27.40.241)
;; WHEN: Sat Aug 27 20:15:20 2011
;; MSG SIZE  rcvd: 104
```

✤ Résolution de noms

- L'espace des noms DNS est divisé en zones distinctes. Cet espace forme un arbre et chaque zone contient :
  - une partie de l'arbre
  - des serveurs de noms :
    - un serveur primaire
    - des serveurs secondaires
- Un programme demandeur transmet une requête pour un nom de domaine à un solveur (*resolver*) local.
  - Si la réponse est dans le cache local, celle-ci est retournée au programme demandeur ;
  - Si la destination est locale, le solveur local donne la réponse, en retournant un enregistrement officiel ;
  - si la destination est distante, le solveur local peut répondre si l'information est disponible dans son cache ;
  - sinon, le solveur interroge le serveur primaire, qui résout la requête directement ou par requêtes **itératives** ou **récursives** jusqu'à l'obtention de l'enregistrement officiel.

Résolveurs DNS

- BIND 9 est un logiciel client-serveur (C-S) répandu pour fournir un service de noms. Il utilise un démon (*deamon*) : `named`, pour sa partie serveur.
- Voir aussi Knot Resolver et Unbound.

Commandes et outils

- **dig** est une commande du package dnsutils sous Linux pour interroger des serveurs DNS.
  - `dig fr.wiktionary.org`
  - `dig mx gmail.com`
- **host** affiche plus simplement les redirections DNS.
  - `host fr.wiktionary.org`
- **nslookup** sera utilisable sous Windows au lieu de dig.
- **whois** permet d'effectuer des recherches sur les bases de données de bureau d'enregistrement.
- RDAP, *Registration Data Access Protocol* est une alternative à Whois ; voir **rdap.org**

DoH ; DNS over HTTPS

- DNS via HTTPS, *DNS over HTTPS* (**DoH**) est un protocole pour la résolution DNS à distance via le protocole HTTPS.
- Les requêtes DNS habituelles sont réalisées en clair (non chiffrés) vers le port 53 du serveur DNS par défaut, ce qui pose des problèmes de sécurité et de confidentialité.
- **DoH** permet de sécuriser les requêtes en faisant passer le trafic DNS sur le protocole sécurisé HTTPS, vers un serveur tel que Cloudflare (https://mozilla.cloudflare-dns.com/dns-query) ou NextDNS (https://trr.dns.nextdns.io/).
- Mozilla Firefox est le premier navigateur web à proposer ce protocole DoH
  - Voir : https://support.mozilla.org/fr/kb/dns-via-https-firefox

À voir :

- RFC 9499: DNS Terminology - Stéphane Bortzmeyer
- Guide pratique du titulaire d'un nom de domaine en .fr - AFNIC

Exercices

Vos réponses avec le document Exercices-RSX102-DNS.docx

Interrogation de serveurs DNS :

- a/ Quelle est l'adresse IP de *amio-millau.fr* ?
- b/ Quel est l'enregistrement de type MX lié à *amio-millau.fr* ?

Enregistrement de nom de domaine :

- c/ Comment procéder pour enregistrer un nom de domaine :
  - en .fr ;
  - en .com ?
- d/ Combien coûte l'enregistrement de nom de domaine :
  - en .fr ;
  - en .com ;
  - en .security ?
- e/ Quel est le rôle de l'AFNIC ?
- f/ Qui est le propriétaire du domaine *aliceandbob.io* ?
  - Et comment avez-vous fait pour le savoir ?

DoH :

- g et h/ Activez DNS-over-HTTPS (DoH) dans Firefox et dans Windows 11.
  - Quels pages web vous a aidé.

Introduction ; les standards de base du web

- La toile, WWW, *World Wide Web*.
- Créé au CERN (Conseil Européen pour la Recherche Nucléaire) par Tim Berners-Lee en 1989, il repose alors sur les standards URL, HTTP, HTML, puis CGI.
- Depuis, d'autres standards importants doivent être considérés.
- www.home.cern/science/computing/birth-web
- En 1993, le CERN a mis le logiciel World Wide Web dans le domaine public. Plus tard, le CERN a publié une version sous licence libre, pour optimiser sa diffusion. Ces actions ont permis au web de prospérer.

Fig 2.2 - Tim Berners-Lee en 2010

Fig 2.2bis - Premier logo du web, par Robert Cailliau

- **Introduction ; les standards de base du web**

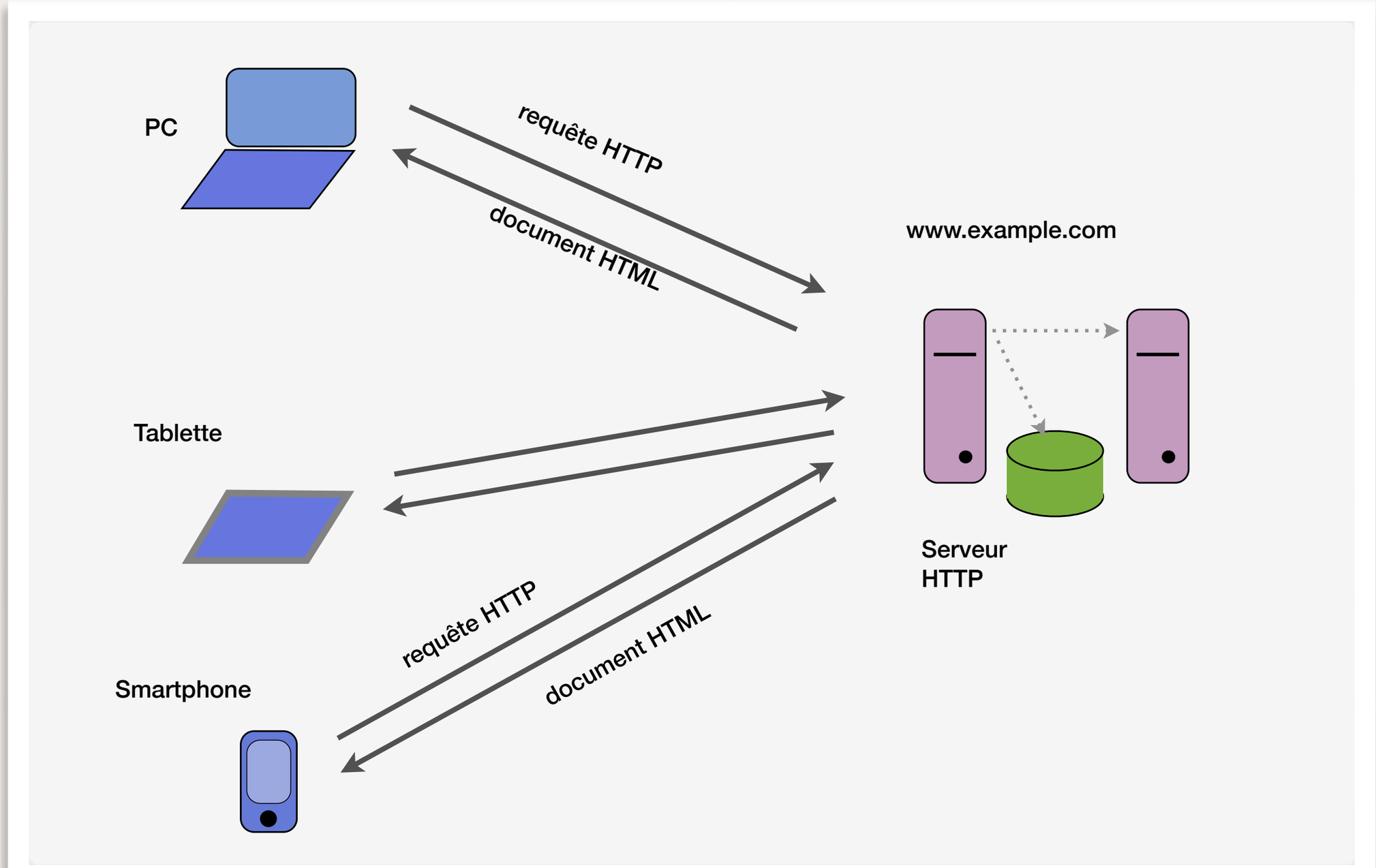


Fig 2.3 - Fonctionnement de base du web

- **URL ; URI**

  - URL , *Uniform Resource Locator*.

  - Format de nommage de ressources ; adresse réticulaire ; (usuellement) adresse web.

- On décompose en général une URL en quatre parties :

  - Le nom du protocole (http ; https ; ftp ; file ; mailto ; news...), suivi par « : »
  - Le nom du serveur : le nom de domaine du serveur, précédé par « // »
  - (en option) Le numéro de port TCP ; pour HTTP, le n° de port par défaut est 80.
  - Le chemin d'accès à la ressource ; si la ressource nécessite des paramètres, elle est suivi par un point d'interrogation (?) et des paramètres.

- **URL ; URI**

```
https://www.w3.org/People/Berners-Lee/FAQ.html
```

Protocole | Nom de domaine | Accès à la ressource

```
http://altavista.digital.com:8080/find.pl?q=url&lang=fr
```

Protocole | Nom de domaine | Port | Accès à la ressource

- Voir : https://rsx102.seancetenante.com/anatomie-url.php

URL ; URI

- URI, *Uniform Resource Identifier*, est un identifiant de ressource physique ou abstraite.
- URI inclut les URL et les URN (Uniform Resource Name). Ce dernier permet d'identifier une ressource dans un espace de noms.

```
urn:ietf:rfc:2396
```

Identifiant de type URN pour le RFC 2396

```
urn:isbn:0-395-36341-1
```

Identifiant de type ISBN (International Standard Book Numbers) :
Référence d'un livre publié.

```
http://fr.wikipedia.org/wiki/Uniform_Resource_Locator

http://codex.wordpress.org/Using_Permalinks
```

Identifiants de type URL

- **HTTP**
  - HTTP, *HyperText Transfer Protocol*
  - C'est le protocole de transfert de documents web (les documents HTML, les images JPEG, PNG, GIF..., les documents CSS, JS, etc.) demandés usuellement par un navigateur (client web) à un serveur HTTP.
  - Lorsque par exemple l'utilisateur clique sur un lien dans un document affiché par son navigateur, les opérations suivantes adviennent :

```
<a href="http://www.w3.org/Graphics/PNG/Overview.html">Portable Network Graphics</a>
```

  - Le navigateur détermine l'URL de la ressource demandée ;
  - Il demande au résolveur DNS l'adresse IP de www.w3.org ;

- **HTTP**
  - DNS répond avec un enregistrement de ressource :

```
www.w3.org.  300    IN     A      128.30.52.37
```

  - Le navigateur se connecte en TCP avec le port 80 de 128.30.52.37 ;
  - Il envoie une requête HTTP ; celle-ci comporte une commande GET indiquant la ressource demandée :

```
GET /Graphics/PNG/Overview.html HTTP/1.1
...
```

- **HTTP**
  - Le serveur retourne le document demandé ;
  - Le navigateur interprète le document, en affiche le texte et refait autant de requête HTTP que d'images ou autres ressources incluses dans le document.

  - Nota :
    - Avec HTTP version 0.9, la connexion TCP était libérée après chaque réponse à une requête.
    - Avec HTTP version 1.1, les autres documents liés à une ressource sont demandés et transférés pendant une seule connexion TCP.
    - Suivant le type MIME du document :
    - le navigateur l'affiche directement
    - ou un module d'extension (plug-in) le traite
    - ou le navigateur appelle une application auxiliaire (helper)

- HTTPS est une variante de HTTP sécurisée par l'usage des protocoles SSL ou TLS

- **HTTP**

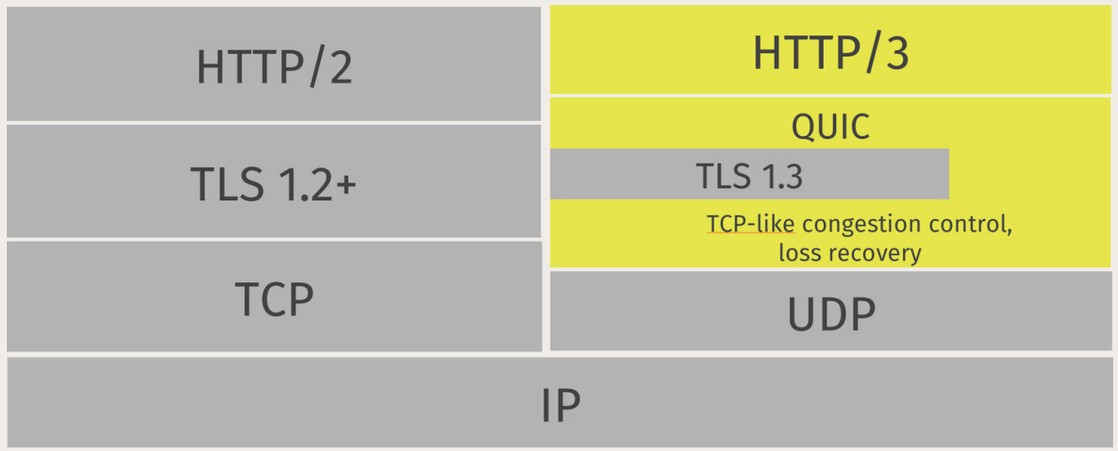
  - Principales commandes HTTP :
    - GET Demande en lecture d'une ressource
    - HEAD Demande d'information sur la ressource
    - POST Soumission ou création de données ;
      L'URI indiquée est celle de la ressource qui traite les données envoyées
    - PUT Mise à jour de ressource
    - DELETE Suppression de ressource

  - Voir en annexe : HTTP : Requête et réponse.

- **HTTP/2**
  - HTTP/2 est finalisé par IETF depuis mai 2015 avec le RFC 7540.
  - Les deux axes majeurs de HTTP/2 sont la rapidité et la sécurité de la navigation.
  - Les avancées de HTTP/2 :
    - La compression des headers des requêtes et des réponses. Cette optimisation permet de réduire la bande passante lorsque les headers (tels que des cookies) sont similaires.
    - Le multiplexage des requêtes au serveur, pour économiser les multiples connexions entre le client et le serveur.
      - De nombreux flux logiques sont envoyés sur la même connexion TCP physique.
    - Le push des ressources du serveur au navigateur. Désormais, le serveur pourra envoyer l'ensemble des ressources référencées dans une même page (CSS, JS…), avant même que le navigateur n'ait analysé celle-ci.
    - HTTP/2 a corrigé le problème de blocage de tête de ligne HTTP (*head-of-line blocking*), dans lequel les clients devaient attendre la fin de la première requête en ligne avant que la suivante ne puisse être envoyée.
  - Voir : https://http2-explained.haxx.se/fr

HTTP/3

- HTTP/3, alias HTTP-over-QUIC, est normalisé en juin 2022 (RCF 9114).
- C'est le premier et principal protocole à être transporté sur QUIC.
- Le client envoie sa requête HTTP sur un flux QUIC bidirectionnel initié par le client.
- Les requêtes HTTP effectuées via HTTP/3 utilisent un ensemble spécifique de flux.
  - Les flux QUIC sont légèrement différents des flux HTTP/2.
  - Dans QUIC, les flux sont fournis par le transport lui-même (dans HTTP/2, les flux étaient créés dans la couche HTTP).
  - Les flux étant indépendants les uns des autres, le protocole de compression d'en-tête utilisé pour HTTP/2 ne pourrait pas être utilisé sans provoquer une situation de blocage de tête de bloc.



- **HTTP/3**
  - HTTP/3 utilise des URLs en https:// uniquement.
  - Le client envoie une requête HTTP sur un flux QUIC bidirectionnel initié par le client.
    - Le serveur renvoie sa réponse HTTP sur ce flux bidirectionnel : une trame HEADERS, une série de trames DATA et éventuellement une dernière trame HEADERS.
  - HTTP/3 envoi donc un ensemble de trames à l'autre extrémité.
  - Les types de trames les plus importants sont :
    - HEADERS, qui envoie des en-têtes HTTP compressés ;
    - DATA, envoie le contenu des données binaires ;
    - GOAWAY, veuillez arrêter cette connexion.
  - Un *push server* est la réponse à une requête que le client n'a jamais envoyée !
    - Les push serveur ne sont autorisés que si le côté client les a acceptés.
  - Voir :
    - https://http3-explained.haxx.se/fr
    - Ch. 3, § 3.3 - QUIC
    - https://www.bortzmeyer.org/9114.html

- **HTTP**

  - Quelques logiciels serveur populaires :
    - Nginx (30 % des sites web)
    - Apache HTTP Server (25 % des sites web)
    - Cloudflare server
    - LiteSpeed Web Server
    - Caddy
    - Apache Tomcat (Un serveur l'application)
  - Voir aussi :
    - MS IIS (Internet Information Services) ; Node.js ; Freenginx

  - Voir sur Geekflare : 6 serveurs Web Open Source pour les petits et grands sites
  - Guide Apache HTTP Server - Stéphane ROBERT

- **HTTP**

  - Quelques logiciels client :
    - NCSA Mosaic (1993 à 1997)
    - Netscape Navigator (1995 à 2008)
    - Internet Explorer (1995) très déprécié et remplacé par :
    - Microsoft Edge (2015 - project **Spartan**) un nouveau navigateur pour Windows 10 et 11 sur PC, tablette et smarphone
    - Mozilla Firefox (2004)
    - Opera (1994)
    - Safari (2003)
    - Chrome (2008)
    - Brave (2016)
  - Autres utilitaires :
    - **Wget** : outil de ligne de commande qui permet de télécharger des fichiers dans votre répertoire actif
    - **cURL** : cf. Page suivante.

- **cURL,** ***client URL request library*** **ou** ***see URL***
  ou ***curl URL request library***

  - Utilitaire en ligne de commande qui repose sur une bibliothèque *libcurl*
  - Permet de lire, créer ou modifier une ressource à l'aide d'une URL.
  - Usage : voir `curl -h`
  - Exemple :
    - `curl -I https://amio-millau.fr` # requête HTTPS méthode HEAD
    - ```
      curl 'https://rsx102.seancetenante.com/documents/fichier10M.txt'
      -H 'User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11;
      rv:78.0) Gecko/20100101 Firefox/78.0' -H 'Accept: text/plain'
      ```

  - Voir :
    - https://ec.haxx.se
    - https://www.it-connect.fr/curl-loutil-testeur-des-protocoles-divers/
      ```

HTML et CSS

- HTML, *HyperText Markup Language*
- C'est le langage de description de page web, constitué de balises et interprété par le navigateur.
  - C'est une simplification de SGML (*Standard Generalized Markup Language*), norme ISO de description de documents.
  - HTML décrit la présentation du document, qui reste statique. L'interactivité se limite aux liens hypertextes (les hyperliens) qui permettent d'afficher une autre page.
  - HTML a évolué en différentes versions jusqu'à une version 4.0 qui perdure depuis 1999. Certains préfèrent une variante basée sur XML, avec une version XHTML 1.0.
  - HTML 5.0, normalisé en octobre 2014, apporte de grandes et bonnes nouveautés.
- HTML est lié à la structure d'une page web.
- CSS, *Cascading Style Sheets* (Feuille de styles en cascade)
  - Ce langage offre des outils avancés permettant la mise en page et la présentation du contenu de pages web.
  - CSS3 reste en cours de spécification.

- **JavaScript**

  - JavaScript est un langage de script, reposant sur de la programmation événementielle.
    - Le code et les fonctions sont incluses dans le code d'une page HTML et exécutées, sur le client web, soit à différents niveaux du cycle de vie de la page, soit en réponse à certaines actions de l'utilisateur.

  - Par exemple, JavaScript :
    - aide à contrôler les données saisies dans des formulaires HTML
    - ou bien permet d'interagir avec le document HTML via l'interface DOM (*Document Object Model*).

  - Le '*Document Object Model*' décrit la structure et le contenu des pages web.

- **Ajax**
  - Ajax, *Asynchronous JavaScript and XML*, est une combinaison de technologies comme :
    - **JavaScript** et **DOM** (Document Object Model), qui sont utilisés pour modifier l'information présentée dans le navigateur par programmation.
    - l'objet XMLHttpRequest, alias XHR, est utilisé pour dialoguer de manière asynchrone avec le serveur Web.
    - CSS (Cascading Style Sheets ; feuilles de style en cascade)
    - XML, utilisé pour l'échange de données. En alternative, les applications Ajax peuvent utiliser les fichiers texte ou JSON (JavaScript Object Notation).
  - Ajax permet de développer des sites web dynamiques (et des applications). Il s'inscrit dans la mode du *Web 2.0*.
  - Avec cette nouvelle architecture, trois concepts sont utilisés :
    - **Des événements légers coté serveur** : des composants d'une application web peuvent effectuer des petites requêtes sur le serveur, pour obtenir des informations qui ne vont modifier qu'une partie du DOM ; le rafraichissement complet de la page n'est pas nécessaire.
    - **Asynchronisme** : les requêtes envoyées ne bloquent pas le navigateur, se sorte que l'utilisateur peut utiliser d'autres parties de l'application. L'interface graphique peut l'informer de la requête en cours.
    - **Généralisation** : tout événement de l'utilisateur (clic souris, survol du pointeur, action sur des touches du clavier) peut déclencher une requête asynchrone.

- **CGI ; Scripts coté serveur**
  - CGI , *Common Gateway Interface* (littéralement : Interface de passerelle commune...)
  - Avec les formulaires, qui existent depuis HTML 2.0, il est devenu nécessaire de construire coté serveur des pages web dynamiques, générées en fonction de données saisies par l'utilisateur.

- **CGI ; Scripts coté serveur**

  - Voici un exemple :
    - ① L'utilisateur, après avoir rempli les champs d'un formulaire d'une page web, clique sur un bouton 'Submit' ;
    - ② Le navigateur envoie une requête HTTP :
      - avec l'URL généralement indiquée comme valeur de l'attribut 'action' de la balise <form> ;
      - et en associant les données saisies soit en fin d'URL (méthode GET) soit dans le corps de la requête HTTP (méthode POST) ;
    - ③ Le serveur HTTP :
      - reçoit la requête ;
      - lance un programme ou un script de type CGI, *Common Gateway Interface*...
      - qui récupère les données soit via des variables d'environnement (méthode GET), soit dans un flux d'entrée (méthode POST) ;
    - ④ Ce programme procède aux traitements en fonction des données transmises et retourne une page web relayée par le serveur HTTP ;
    - ⑤ Le navigateur affiche la page web résultante.

- **CGI ; Scripts coté serveur**

  - Nota :
  - En ③ , on préfère actuellement les variantes suivantes :
    - FastCGI : cela permet de lancer le programme qu'une fois. Il reste ensuite en cache pour une utilisation ultérieure
    - Les interprèteurs sont maintenant intégrés au sein du serveur HTTP sous forme de modules.
  - **Scripts coté serveurs**
    Les CGI sont des programmes compilés ou (très souvent) interprétés. Les langages serveur suivants sont populaires :
    - Perl
    - Python
    - JSP (Java Server Pages)
    - ASP (Active Server Pages)
    - PHP (PHP HyperText Preprocessor)
    - Servlets (avec Java)

- **Introduction**
  - XML est un langage de description adapté à la représentation structurée de données.
  - C'est un langage **extensible**, utilisant des balises (***markup***).
  - Un document XML est un fichier texte fortement structuré, destiné à représenter tout type de données.
  - Un inconvénient de HTML est qu'il ne permet pas de séparer la forme du contenu d'un document. XML peut servir à décrire un **contenu** ; la forme, le cas échéant, dépend de feuilles de styles associées.
  - XML, dérivé de SGML (*Standard Generalized Markup Language*), est un standard d'échange de données normalisé par W3C (*World Wide Web Contorsium*) : la version 1.0 en 1998 et la version 1.1 en 2004.
  - XML est utilisé pour :
    - stocker ou échanger des données ;
    - définir des langages, comme XHTML, SVG, etc.

- **Exemples de balises :**

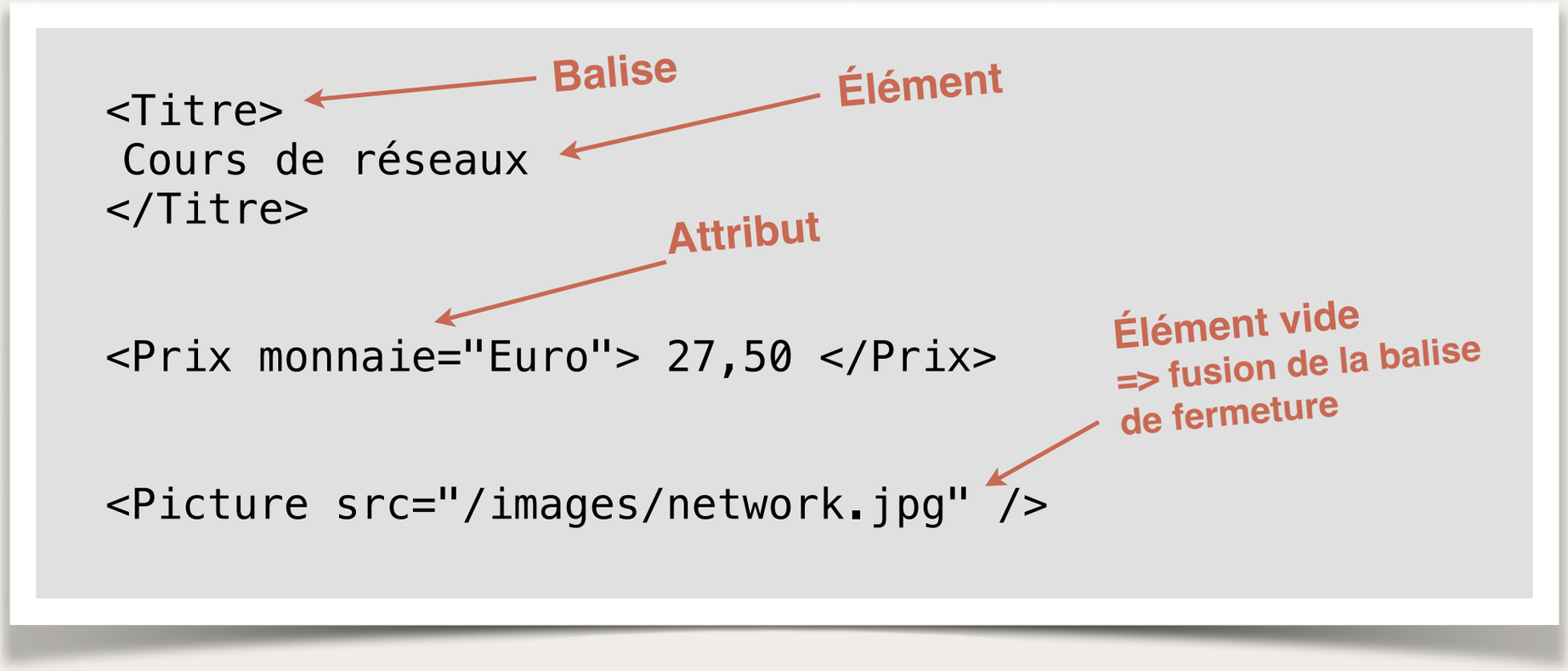


Fig 2.4 - XML : Balise, élement et attribut

- **Utilisation de XML**
  - Une application liée à XML utilise jusqu'à trois types de documents :
    1. Le document XML
    2. Le document DTD (*Document Type Definition*)
    3. Une feuille de style XSL (*eXtensible Stylesheet Language*)

- ① **Le document XML**
  - Il contient les données du document, structurées à l'aide des marqueurs ou balises.
  - Il doit être bien formé :
    - Il a toujours une et une seule racine, alias *nœud du document* ;
    - Une balise doit toujours être refermée. Sans entrecroisements.
      Ex. `<auteur>Victor <b>Hugo</b></auteur>`
    - Une balise ne peut pas commencer par `-.` et ne peut pas contenir d'espace ni `! »#$%&'()*+,/;<=>?@[\]^`{|}~`
  - Voici un exemple de document XML :

2 - Applications client-serveur dans internet

2.3 - XML ; Extensible Markup Language

```
<?xml version="1.0" encoding="iso-8859-1" standalone="no" ?>
<!DOCTYPE biblio SYSTEM "sample.dtd">
<!-- Ci-dessus le prologue : déclaration et DTD utilisé -->
<!-- Élément racine -->
<biblio>
  <!-- Premier enfant -->
  <livre>
    <!-- Élément enfant titre -->
    <titre>Les Misérables</titre>
    <auteur>Victor Hugo</auteur>
    <nb_tomes>3</nb_tomes>
  </livre>
  <livre>
    <titre>Les Confessions</titre>
    <auteur>Jean-Jacques Rousseau</auteur>
    <auteur>Jacques Perrin</auteur>
  </livre>
  <livre lang="en">
    <titre>David Copperfield</titre>
    <auteur>Charles Dickens</auteur>
    <nb_tomes>3</nb_tomes>
  </livre>
</biblio>
```

2 - Applications client-serveur dans internet

2.3 - XML ; Extensible Markup Language

- Il débute avec un prologue, composé d'une déclaration XML et, en option, d'une déclaration de type de document

```
<?xml version="1.0" encoding="iso-8859-1" standalone="no" ?>
<!DOCTYPE biblio SYSTEM « sample.dtd">
```

- La déclaration de type de document précise l'élément racine du document XML (**biblio** dans l'exemple) et indique un lien (sample.dtd) vers un document **DTD,** *Document Type Definition*

2 - Applications client-serveur dans internet

2.3 - XML ; Extensible Markup Language

- La ligne suivante est la balise d'ouverture de la racine du document XML ; la racine (biblio) définit une structure (une liste de livres dans cet exemple) :

```
<biblio>
  <livre> premier ouvrage </livre>
  <livre> deuxième ouvrage </livre>
  <!-- etc. -->
</biblio>
```

- La racine peut admettre des attributs et contient un ensemble d'éléments fils (livre) ou petit fils (titre, auteur, etc.)

- ② **Le document DTD, *Document Type Definition***
  - Ce 2e type de document est celui indiqué le cas échéant avec la 2e ligne du prologue du document XML principal.
  - Il spécifie les règles pour les éléments et les attributs présents dans le document XML.
    - La DTD va permettre de vérifier la conformité du document XML.
    - Le document XML sera dit valide s'il respecte la DTD.
    - Le DTD est optionnel. L'attribut standalone prendra dans la déclaration XML la valeur "yes" s'il n'y a pas à rechercher de DTD externe, "no" sinon.

- ③ **Une feuille de style XSL, *eXtensible Stylesheet Language***
  - Ce 3e type de document, optionnel, est indiqué dans le document XML principal par une ligne du type

```
<?xml-stylesheet type="text/xml" href="biblio.xsl"?>
```

  - Le document XSL :
    - dicte le formatage des éléments lors de leur affichage ;
    - il est composé au moins de 2 parties distinctes et complémentaires
    - XSLT, un langage de transformation de documents (ex. génération d'une page HTML à partir d'un fichier XML et de la feuille de style)
    - XSL-FO, un ensemble d'instructions de formatage XML dédié à la présentation (*Formatting Objects*) de documents en vue d'une impression. Il comporte un modèle de format et des propriétés de style basées sur CSS2.

- **Exploitation de document XML**
  - Pour l'exploitation d'un document XML, deux API sont couramment utilisées par les applications :
  - DOM (*Document Object Model*) est une API pour des objets de type document définis par une structure XML ; cela permet de naviguer dans le document (décrit par un arbre) et d'accéder à ses parties.
    - La totalité du document XML est chargé en mémoire, sous forme d'objet.
  - SAX : *Simple API for XML* est une interface plus simple, pour un même usage.
    - Le document est analysé avec une approche en flux.

- **À voir :**
  - https://developer.mozilla.org/fr/docs/Web/XML/XML_introduction
  - https://www.w3schools.com/xml/

2 - Applications client-serveur dans internet

2.4 - JSON ; JavaScript Object Notation

- **Comme XML, JSON est un langage de description adapté à la représentation structurée de données.**

```
{
  "menu": "File",
  "commands": [
     {
        "value": "New",
        "action":"CreateDoc"
     },
     {
        "value": "Open",
        "action":"OpenDoc"
     },
     {
        "value": "Close",
        "action":"CloseDoc"
     }
  ]
}
```

menubar.json

```
<?xml version="1.0" ?>
<menubar>
 <menu name="File">
  <command value="New" action="CreateDoc" />
  <command value="Open" action="OpenDoc" />
  <command value="Close" action="CloseDoc" />
 </menu>
</menubar>
```

menubar.xml

Fig 2.5 - Comparaison d'un document Json et XML

2 - Applications client-serveur dans internet

2.4 - JSON ; JavaScript Object Notation

- **JSON, *JavaScript Object Notation***
  - est un format de données d'abord dédié aux échanges entre le navigateur et le serveur
  - est très facile à utiliser en **JavaScript** ; il fait partie du langage
  - est un format texte complètement indépendant de tout langage
  - il se base sur deux structures :
    - Un tableau associatif de JavaScript (www.xul.fr/ecmascript/associatif.php)
    - Un tableau, donc une liste de valeurs ordonnées
  - il date de 2002 et est devenu populaire lorsqu'Ajax à commencé à être largement utilisé
  - il est même devenu un type de donnée dans PostgreSQL
- **Le format JSON reconnaît les même types de données que JavaScript**
  - **Number, String, Boolean** et **null** sont des primitives que l'on peut assigner à une clé qui doit être une chaîne
  - **Array** est une liste de clé-valeurs placées entre crochets
  - **Object** est une liste de clé-valeurs placées entre accolades
- **Pour en savoir plus : http://json.org**

2.4 - JSON ; JavaScript Object Notation

- **Exemple d'échange entre client et serveur web**

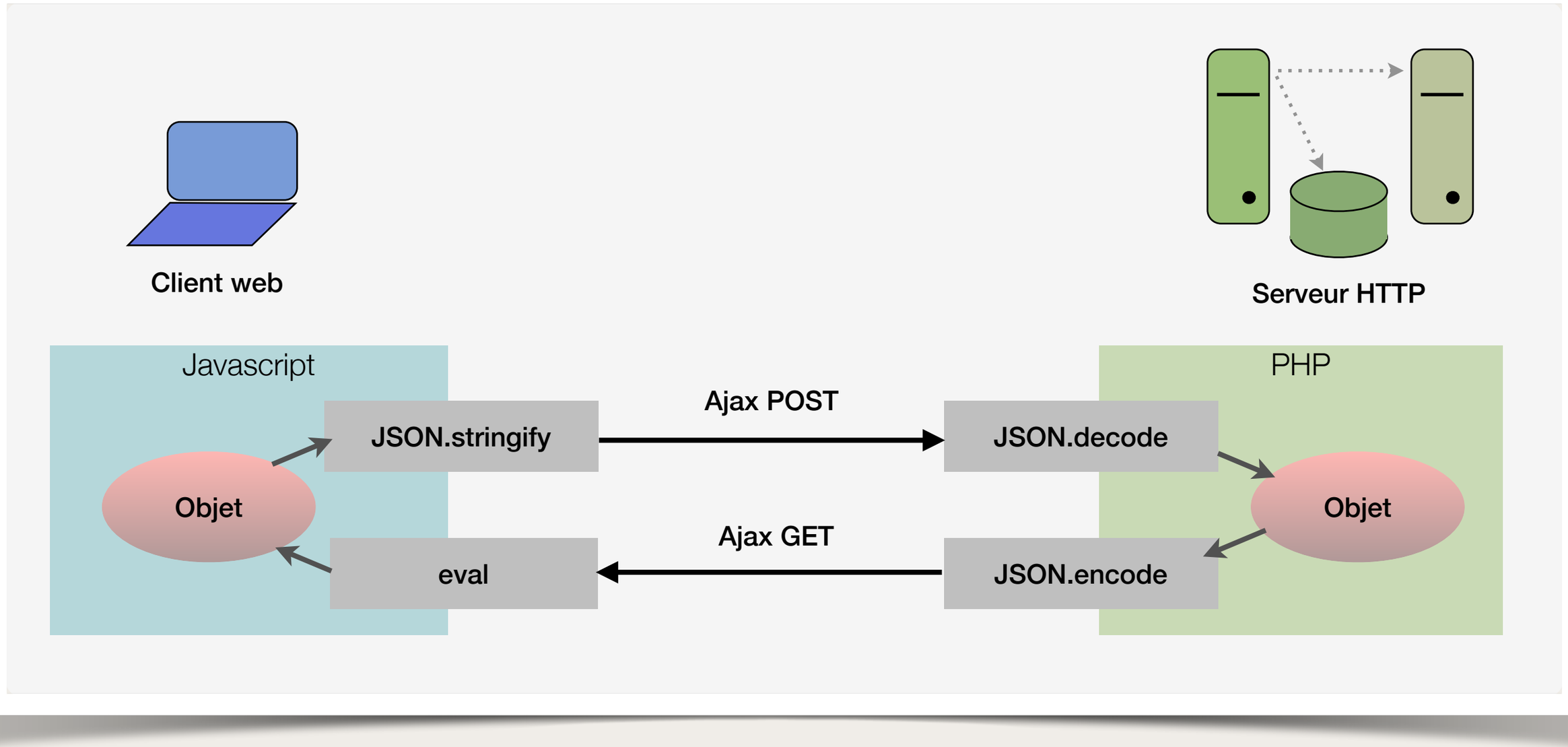


Fig 2.6 - Échange via Json entre client et serveur web

2 - Applications client-serveur dans internet

2.5 - Architectures web 3 tiers

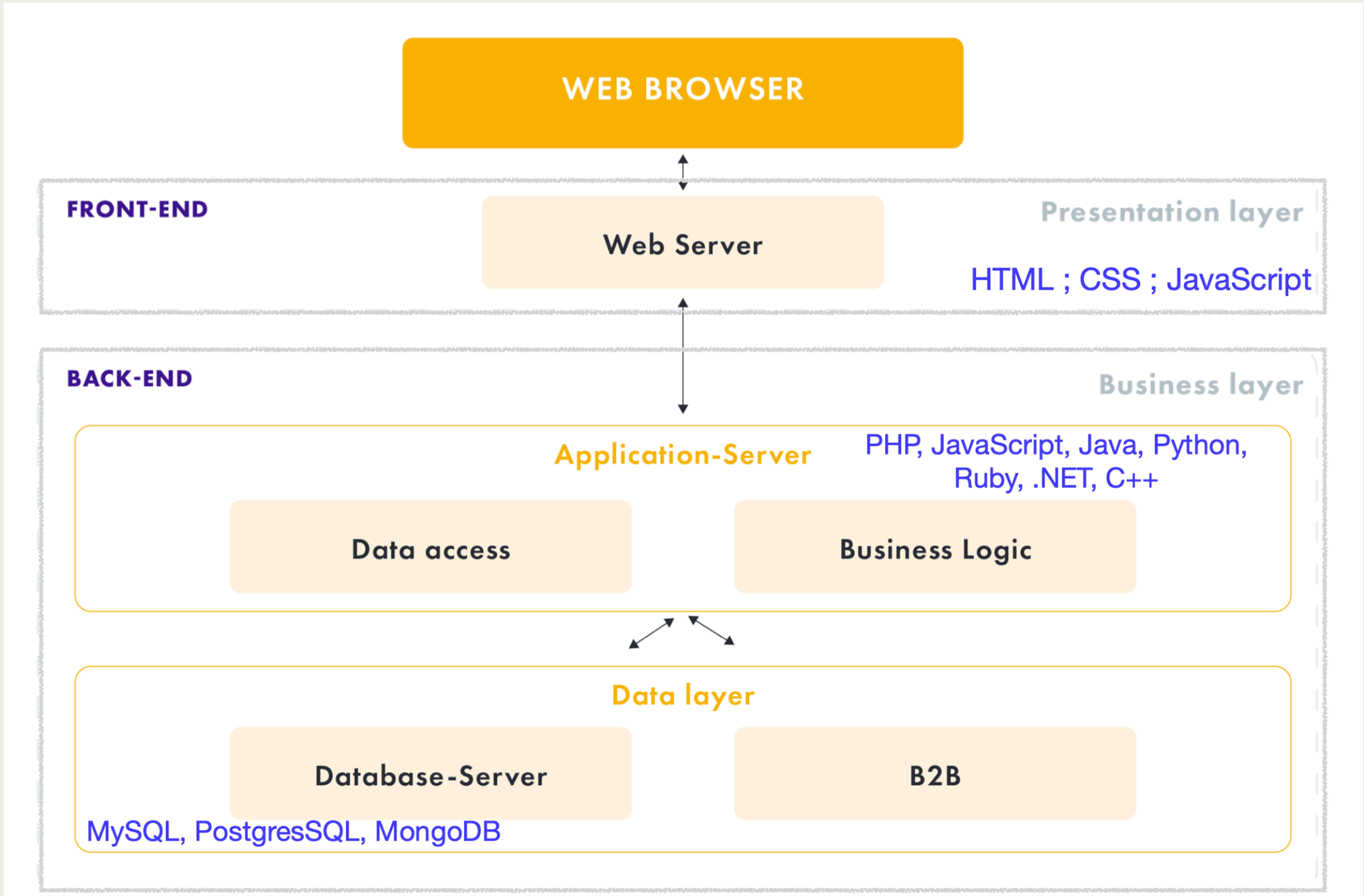
- **Front-End, Back-End, Data Base**
  - 

Fig 2.7 - Architectures web 3 tiers