***Cours dot.NET***

Table des matières

[9.1 Généralités 2](#_Toc95487754)

[A/ Définition 2](#_Toc95487755)

[B/Notes 5](#_Toc95487756)

[9.2 Les classes .NET de la gestion des adresses IP 6](#_Toc95487757)

[A/ Définition 6](#_Toc95487758)

[B/Notes 7](#_Toc95487759)

[9.3 Les bases de la programmation internet 8](#_Toc95487760)

[A/ Définition 8](#_Toc95487761)

[B/Notes 9](#_Toc95487762)

[9.4 Découvrir les protocoles de communication de l'internet 11](#_Toc95487763)

[A/ Définition 11](#_Toc95487764)

[B/Notes 12](#_Toc95487765)

[9.5 Les classes .NET de la programmation internet 13](#_Toc95487766)

[A/ Définition 13](#_Toc95487767)

[B/Notes 14](#_Toc95487768)

[9.6 Exemples de clients / serveurs TCP 15](#_Toc95487769)

[A/ Définition 15](#_Toc95487770)

[B/Notes 16](#_Toc95487771)

[9.7 Les classes .NET spécialisées dans un protocole particulier de l'internet 17](#_Toc95487772)

[A/ Définition 17](#_Toc95487773)

[B/Notes 18](#_Toc95487774)

# 9.1 Généralités

## A/ Définition

*TCP (Transfer Control Protocol) :*

Protocoles de la couche transport du modèle [TCP/IP](https://www.commentcamarche.net/contents/539-tcp-ip). Il permet, au niveau des applications, de gérer les données en provenance (ou à destination) de la couche inférieure du modèle (c'est-à-dire le protocole [IP](https://www.commentcamarche.net/contents/530-le-protocole-ip))

*IP (Internet Protocol) :*

Définit la forme que les paquets doivent prendre et la façon dont ils doivent être gérés lors de leur émission ou de leur réception

*OSI (Open Systems Interconnection Reference Model) :*

Est un cadre de travail conceptuel utilisé pour décrire les fonctions d’un système de réseau

*Couches :*

* *Physique =* Assure la transmission de bits sur un support physique (fibre, câble de cuivre, etc…)
* *Liaison de données =* Masque les particularités physiques de la couche Physique. Détecte et corrige les erreurs de transmission
* *Réseau =* Gère le chemin que doivent suivre les informations envoyées sur le réseau (routage)
* *Transport =* Permet la communication entre deux applications alors que les couches précédentes ne permettaient que la communication entre machines
* *Session =* On va trouver dans cette couche des services permettant à une application d'ouvrir et de maintenir une session de travail sur une machine distante
* *Présentation =* Uniformiser la représentation des données sur les différentes machines
* *Application =* A ce niveau, on trouve les applications généralement proches de l'utilisateur telles que la messagerie électronique ou le transfert de fichiers

*Routage :*

Méthode d'acheminement des paquets IP à leur destination

* **Direct**  = l'acheminement d'un paquet IP directement de l'expéditeur au destinataire à l'intérieur du même réseau
* **Indirect** = l'acheminement d'un paquet IP à une destination se trouvant sur un autre réseau que celui auquel appartient l'expéditeur

*Multiplexage :*

La couche transport pourra utiliser une même connexion réseau (de machine à machine) pour transmettre des informations appartenant à plusieurs applications

*TCP (Transmission Control Protocol) :*

Assure une remise fiable d'informations entre deux clients

*UDP (User Datagram Protocol) :*

Assure une remise non fiable d'informations entre deux clients

*FTP (File Transfer Protocol) :*

Permet des transferts de fichiers

*TFTP (Trivial File Transfer Control) :*

Une variante de FTP. Il s'appuie sur le protocole UDP et est moins sophistiqué que FTP

*SMTP (Simple Mail Transfer Protocol) :*

Permet l'échange de messages entre utilisateurs du réseau

*DNS (Domain Name System) :*

Transforme un nom de machine en adresse Internet de la machine

*NFS (Network File System) :*

Protocole qui permet à une machine, de "voir" le système de fichiers d'une autre machine

*ARP (Address Resolution Protocol) :*

[Protocole](https://fr.wikipedia.org/wiki/Protocole_de_communication) utilisé pour associer l'adresse de protocole de [couche réseau](https://fr.wikipedia.org/wiki/Couche_r%C3%A9seau) (typiquement une [adresse IPv4](https://fr.wikipedia.org/wiki/Adresse_IPv4)) d'un hôte distant, à son adresse de protocole de [couche de liaison](https://fr.wikipedia.org/wiki/Couche_de_liaison) (typiquement une [adresse MAC](https://fr.wikipedia.org/wiki/Adresse_MAC))

*UDP (User Datagram Protocol) :*

Permet un échange non fiable de données entre deux points, c'est à dire que le bon acheminement d'un paquet à sa destination n'est pas garanti

*TELNET :*

Protocole qui permet à un utilisateur d'une machine A du réseau de se connecter sur une machine B (appelée souvent machine hôte). TELNET émule sur la machine A un terminal dit universel. L'utilisateur se comporte donc comme s'il disposait d'un terminal connecté à la machine B. Telnet s'appuie sur le protocole TCP

*RPC (Remote Procedure Call) :*

Protocole de communication entre applications distantes, indépendant de la couche transport. Ce protocole est important : il décharge le programmeur de la connaissance des détails de la couche transport et rend les applications portables

## B/Notes

L'information passe à la couche du dessous et ainsi de suite jusqu'à arriver sur le support physique. Là, l'information est physiquement transférée à la machine destinatrice où elle retraversera les mêmes couches, en sens inverse cette fois-ci, jusqu'à arriver à l'application destinatrice des informations envoyées

L'adresse Internet d'un nœud est une adresse logique : elle est indépendante du matériel et du réseau utilisé. C'est une adresse sur 4 octets identifiant à la fois un réseau local et un nœud de ce réseau

Exemple : l'adresse de la machine Lagaffe de la faculté des Sciences d'Angers est notée 193.49.144.1 et celle de la machine Liny 193.49.144.9. On en déduira que l'adresse Internet du réseau local est 193.49.144.0. On pourra avoir jusqu'à 254 nœuds sur ce réseau.

Une machine d'un réseau A peut communiquer avec une machine d'un réseau B sans se préoccuper du type de réseau sur lequel elle se trouve : il suffit qu'elle connaisse son adresse IP

Une adresse IP est une suite de 4 octets notée souvent I1.I2.I3.I4, qui contient en fait deux adresses :

• l'adresse du réseau

• l'adresse d'un nœud de ce réseau

**Classe A**

L'adresse IP : I1.I2.I3.I4 a la forme R1.N1.N2.N3 (R adresse réseau ; N adresse machine)

**Classe B**

L'adresse IP : I1.I2.I3.I4 a la forme R1.R2.N1.N2 (R adresse réseau ; N adresse machine)

**Classe C**

L'adresse IP : I1.I2.I3.I4 a la forme R1.R2.R3.N1 (R adresse réseau ; N adresse machine)

# 9.2 Les classes .NET de la gestion des adresses IP

## A/ Définition

*IPAddress Any :*

L'adresse IP "0.0.0.0". Lorsqu'un service est associé à cette adresse, cela signifie qu'il accepte des clients sur toutes les adresses IP de la machine sur laquelle il opère

*IPAddress LoopBack :*

L'adresse IP "127.0.0.1". Appelée "adresse de boucle". Lorsqu'un service est associé à cette adresse, cela signifie qu'il n'accepte que les clients qui sont sur la même machine que lui

*IPAdress None :*

L'adresse IP "255.255.255.255". Lorsqu'un service est associé à cette adresse, cela signifie qu'il n'accepte aucun client

*Trim() :*

Supprime tous les caractères correspondant à un espace blanc au début et à la fin de la chaîne actuelle

*IPHostEntry Classe :*

Fournit une classe conteneur pour les informations sur l'adresse de l'hôte Internet

* AddressList = Obtient ou définit une liste d'adresses IP qui sont associées à un hôte
* Aliases = Obtient ou définit une liste d'alias qui sont associés à un hôte
* HostName = Obtient ou définit le nom DNS de l'hôte

*Alias d’IP :*

Est une configuration spéciale du réseau de votre serveur dédié OVHcloud, qui vous permet d’associer plusieurs adresses IP sur une seule interface réseau

## B/Notes

La [IPHostEntry](https://docs.microsoft.com/fr-fr/dotnet/api/system.net.iphostentry?view=net-6.0) classe associe un nom d’hôte DNS (Domain Name System) à un tableau d’alias et un tableau d’adresses IP correspondantes.

La [IPHostEntry](https://docs.microsoft.com/fr-fr/dotnet/api/system.net.iphostentry?view=net-6.0) classe est utilisée comme classe d’assistance avec la [Dns](https://docs.microsoft.com/fr-fr/dotnet/api/system.net.dns?view=net-6.0) classe

# 9.3 Les bases de la programmation internet

## A/ Définition

*full-duplex* :

L’information peut transiter dans les deux sens

*Communication dyssymétrique :*

La machine *A* initie une connexion pour demander un service à la machine *B*. Il précise qu'il veut ouvrir une connexion avec le service *SB1* de la machine B. Celle-ci accepte ou refuse. Si elle accepte, la machine *A* peut envoyer ses demandes au service *SB1*

*Machine cliente :*

Machine qui demande des services à une application serveur

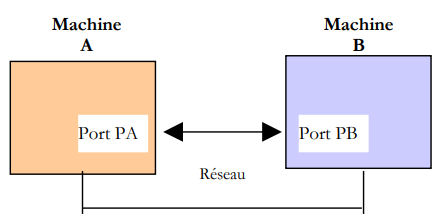
*Machine serveur :*

Machine qui offre les services à la machine client

*Port de service :*

Port de la machine serveur qui permet à cette dernière de rendre plusieurs services ne même temps (le port effectue les demandes quand la machine en reçoit plusieurs)

## B/Notes



Lorsqu’une application *AppA* d'une machine A veut communiquer avec une application *AppB* d'une machine B de l'Internet, elle doit connaître plusieurs choses :

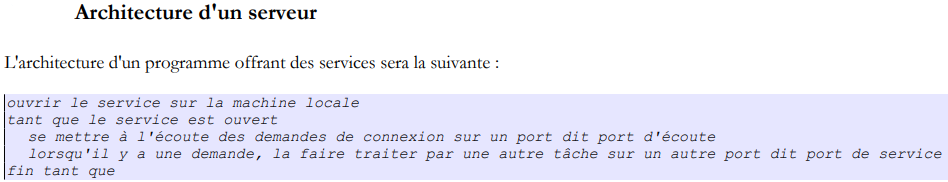
* l'adresse IP ou le nom de la machine B
* Le numéro du port avec lequel travaille l'application *AppB*
* Les protocoles de communication compris par la machine B
* Le protocole de dialogue accepté par l'application *AppB*

Caractéristiques du protocole de transport TCP:

* Le processus qui souhaite émettre établit tout d'abord une connexion avec le processus destinataire des informations qu'il va émettre.
* Tous les paquets émis par le processus source suivent ce chemin virtuel et arrivent dans l'ordre où ils ont été émis
* L'information émise a un aspect continu. Le processus émetteur envoie des informations à son rythme
* Chaque segment envoyé par le protocole TCP est numéroté. Le protocole TCP destinataire vérifie qu'il reçoit bien les segments en séquence. Pour chaque segment correctement reçu, il envoie un accusé de réception à l'expéditeur.
* Lorsque ce dernier le reçoit, il l'indique au processus émetteur. Celui-ci peut donc savoir qu'un segment est arrivé à bon port.
* Si au bout d'un certain temps, le protocole TCP ayant émis un segment ne reçoit pas d'accusé de réception, il retransmet le segment en question, garantissant ainsi la qualité du service d'acheminement de l'information.
* Le circuit virtuel établi entre les deux processus qui communiquent est ***full-duplex*** : cela signifie que l'information peut transiter dans les deux sens. Ainsi le processus destination peut envoyer des accusés de réception alors même que le processus source continue d'envoyer des informations. Cela permet par exemple au protocole TCP source d'envoyer plusieurs segments sans attendre d'accusé de réception.

Une image contenant texte

Description générée automatiquement



Le programme serveur traite différemment la demande de connexion initiale d'un client de ses demandes ultérieures visant à obtenir un service. Le programme n'assure pas le service lui-même. S'il le faisait, pendant la durée du service il ne serait plus à l'écoute des demandes de connexion et des clients ne seraient alors pas servis.

Il procède donc autrement : dès qu'une demande de connexion est reçue sur le port d'écoute puis acceptée, le serveur crée une tâche chargée de rendre le service demandé par le client. Ce service est rendu sur un autre port de la machine serveur appelé ***port de service***. On peut ainsi servir plusieurs clients en même temps

Une image contenant texte

Description générée automatiquement

# 9.4 Découvrir les protocoles de communication de l'internet

## A/ Définition

*Mode Raw :*

Connexion Tcp brute

*POP (Post Office Protocol) :*

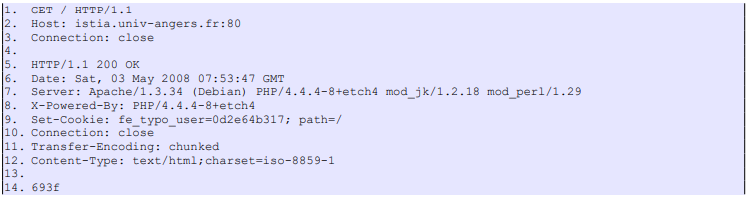
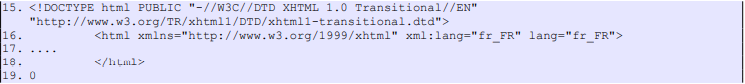
P[rotocole](https://fr.wikipedia.org/wiki/Protocole_r%C3%A9seau) qui permet de récupérer les [courriers électroniques](https://fr.wikipedia.org/wiki/Courrier_%C3%A9lectronique) situés sur un [serveur de messagerie électronique](https://fr.wikipedia.org/wiki/Serveur_de_messagerie_%C3%A9lectronique)

*Code :* [*https://fr.wikipedia.org/wiki/Liste\_des\_codes\_HTTP*](https://fr.wikipedia.org/wiki/Liste_des_codes_HTTP)

* 200 : succès de la requête
* 301 : déplacement permanente
* 302 : déplacement temporaire
* 303 : réponse requête ailleurs
* 304 : document non modifié depuis la dernière requête
* 307 : Redirection temporaire
* 308 : Redirection permanente
* 310 : Trop de redirections
* 400 : syntaxe de la requête est erronée
* 401 : utilisateur non authentifié
* 403 : accès refusé
* [404](https://fr.wikipedia.org/wiki/Erreur_404) : ressource non trouvée
* 500 : erreur interne du serveur
* 502 : Mauvaise passerelle
* 503 : Service indisponible
* 504 : le serveur n'a pas répondu

## B/Notes

HTTP

SMTP

Une image contenant texte

Description générée automatiquement

POP

Une image contenant texte

Description générée automatiquement Une image contenant texte

Description générée automatiquement

# 9.5 Les classes .NET de la programmation internet

## A/ Définition

*Socket :*

Classe qui opère le plus près du réseau. Elle permet de gérer finement la connexion réseau. Le terme *socket* désigne une prise de courant. Le terme a été étendu pour désigner une prise de réseau logicielle

*NetworkStream :*

Classe qui représente le flux réseau entre le client et le serveur. Elle est dérivée de la classe *Stream*

*StreamReader :*

Lire des lignes dans le flux réseau

*StreamWriter :*

Écrire ces lignes dans le flux réseau

## B/Notes

*Tcp Client*

Une image contenant texte

Description générée automatiquement

*Tcp Listener*

Une image contenant texte

Description générée automatiquement

*IPAddress.Any* désigne toutes les adresses IP d'une machine

# 9.6 Exemples de clients / serveurs TCP

## A/ Définition

## B/Notes

Le client et le serveur sont lié car ils sont sur la même machine (« IPAdress.Any »)

On peut donc utiliser les adresses IP suivantes :

* Adresse IP sur notre Wifi (obtenue avec *ipconfig*)
* Les adresses IP locales (127.0.0.1 à 127.0.0.255)

# 9.7 Les classes .NET spécialisées dans un protocole particulier de l'internet

## A/ Définition

*URI (Uniform Resource Identifier) :*

Une chaîne qui fait référence à une ressource ([URL](https://developer.mozilla.org/fr/docs/Glossary/URL))

*[WebClient].ResponseHeaders :*

Représente une collection valuée dont les clés sont les noms des entêtes HTTP et les valeurs, les chaînes de caractères associées à ces entêtes

*WebException :*

Exceptions qui sont levées lors d'un échange client

*WebRequest :*

Représente la totalité de la demande du client Web

*WebResponse :*

Représente la totalité de la réponse du serveur Web

*GetResponse() :*

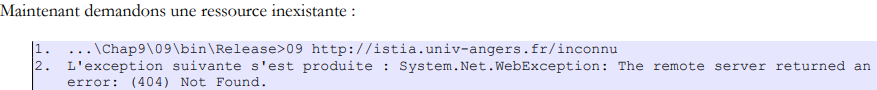
Demande le document identifié par son Uri et rend un objet WebRequestResponse qu'on transforme ici en objet HttpWebResponse

## B/Notes

*Client Web*

Une image contenant texte

Description générée automatiquement



Une WebException c’est lancé car le fichier demandé n’est pas trouvé (Erreur 404)

Une image contenant texte

Description générée automatiquement

WebRequest & WebResponse sont plus précis que WebClient car ce dernier gère plusieurs types de fichier en même temps (http:, https:, ftp:, file: ) alors que WebRequest & WebResponse peuvent êtres plus précis car on peut les adaptés au types de fichiers comme avec l’exemple que nous utilisons dans ce cours : HttpWebRequest, HttpWebResponse pour un client http

Proxy : Port 3128

*WebRequestResponse*

Une image contenant texte

Description générée automatiquement