Chapitre I
LE MODELE CLIENT/SERVEUR

Chapitre I

LE MODELE CLIENT/SERVEUR

I.1 Introduction

Dans l'informatique moderne, de nombreuses applications fonctionnent selon un environnement client-serveur; cette dénomination signifie que des machines clientes (faisant partie du réseau) contactent un serveur - une machine généralement très puissante en termes de capacités d'entrées-sorties - qui leur fournit des services. Nous allons voir comment cette technologie permet d'exploiter au mieux les réseaux, et permet un haut niveau de coopération entre différentes machines sans que l'utilisateur se préoccupe des détails de compatibilité

I.2. Définition du modèle client/serveur

Le modèle client-serveur s'articule autour d'un réseau auquel sont connectés deux types d'ordinateurs le serveur et le client. Le client et le serveur communiquent via des protocoles. Les applications et les données sont réparties entre le client et le serveur de manière à réduire les coûts. Le client-serveur représente un dialogue entre deux processus informatiques par l'intermédiaire d'un échange de messages. Le processus client sous-traite au processus serveur des services à réaliser. Les processus sont généralement exécutés sur des machines, des OS et des réseaux hétérogènes.

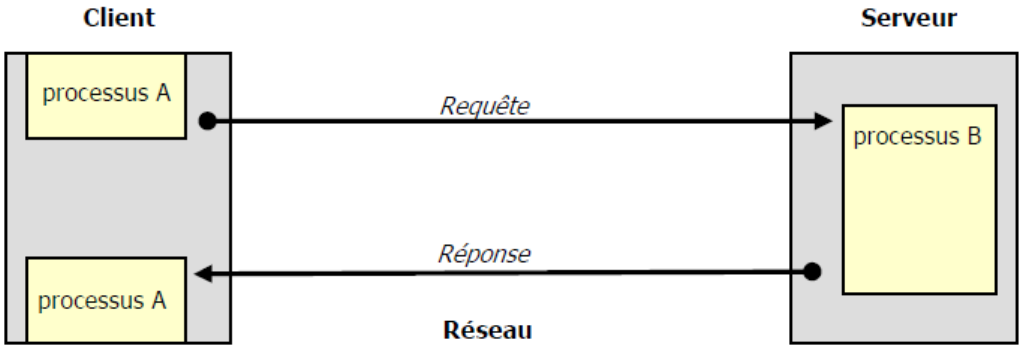

Figure I.1: Le modèle client/serveur

I.3. Caractéristiques des systèmes client serveur

Les éléments qui caractérisent une architecture client serveur sont :

- **Service**

Le modèle client serveur est une relation entre des processus qui tournent sur des machines séparées. Le serveur est un fournisseur de services. Le client est un consommateur de services.

- **Partage de ressources**

Un serveur traite plusieurs clients et contrôle leurs accès aux ressources

- **Protocole asymétrique**

Conséquence du partage de ressources, le protocole de communication est asymétrique le client déclenche le dialogue ; le serveur attend les requêtes des clients.

- **Transparence de la localisation**

L'architecture client serveur doit masquer au client la localisation du serveur (que le service soit sur la même machine ou accessible par le réseau). Transparence par rapport aux systèmes d'exploitation et aux plates-formes matérielles. Idéalement, le logiciel client serveur doit être indépendant de ces deux éléments

- **Message**

Les messages sont les moyens d'échanges entre client et serveur.

- **Encapsulation des services**

Un client demande un service. Le serveur décide de la façon de le rendre une mise à niveau du logiciel serveur doit être sans conséquence pour le client tant que l'interface message est identique.

- **Evolution**

Une architecture client serveur doit pouvoir évoluer horizontalement (évolution du nombre de clients) et verticalement (évolution du nombre et des caractéristiques des serveurs).

I.4. La répartition des tâches

Dans l'architecture client/serveur, une application est constituée de trois parties :

- L'interface utilisateur
- La logique des traitements
- La gestion des données

Le client n'exécute que l'interface utilisateur (souvent un interfaces graphique)

Ainsi que la logique des traitements (formuler la requête), laissant au serveur de bases de données la gestion complète des manipulations de données

La liaison entre le client et le serveur correspond a tout un ensemble complexe de logiciels appelé middleware qui se charge de toues les communication entre les processus.

I.5. Les différents modèles de client/serveur

En fait, les différences sont essentiellement liées aux services qui sont assurés par le serveur.

On distingue couramment:

I.5.1.Le client -serveur de donnée

Dans ce cas, le serveur assure des taches de gestion, stockage et de traitement de donnée .c'est le cas le plus connu de client- serveur est utilisé par tous les grands SGBD:

La base de données avec tous ses outils (maintenance, sauvegarde….) est installée sur un poste serveur.

Sur les clients, un logiciel d'accès est installé permettant d'accéder à la base de données du serveur

Tous les traitements sur les données sont effectués sur le serveur qui renvoie les informations demandées par le client.

I.5.2.Le client -serveur de présentation

Dans ce cas la présentation des pages affichées par le client est intégralement prise en charge par le serveur. Cette organisation présente l'inconvénient de générer un fort trafic réseaux.

I.5.3.Le client –serveur de traitement

Dans ce cas, le serveur effectue des traitements a la demande du client .Il peut S'agir de traitement particulier sur des données, de vérification de formulaire de saisie, de traitements d'alarmes

Ces traitements peuvent être réalisés par des programmes installés sur des serveurs mais également intégrés dans des bases de données, dans ce cas, la partie donnée et traitement sont intégrés.

I.6 La notion de protocole et port

I.6.1.Notion de port

Lors d'une communication en réseau, les différents ordinateurs s'échangent des informations qui sont généralement destinées à plusieurs applications (le client mail et le navigateur internet par exemple).

Seulement ces informations transitent par la même passerelle. Il faut donc savoir pour quelle application telle information est destinée. On attribue donc des ports pour chaque application. Un port est comme une porte en schématisant. Les informations sont multiplexées (comme dans les voitures récentes) et passent par la passerelle. A leur arrivée (vers le serveur) ou à leur réception (vers votre machine) elles sont dé multiplexées et chaque information distincte passe par le port qui lui est associé. Les informations sont ensuite traitées par l'application correspondante.

Un port est codé sur 16 bits, il y a donc 65536 ports.

L'adresse IP plus le port (exemple : **127.0.0.1:80**) est appelée socket.

Les ports ce sont vus attribuer une assignation par défaut pour aider à la configuration des réseaux.

Voici les principaux ports et le protocole les utilisant :

Port Service ou Application

21	FTP
23	Telnet
25	MTP
53	DNS
80	HTTP
110	POP3
119	NNTP

Les ports 0 à 1023 sont les ports reconnus ou réservés et sont assignés par l'IANA (Internet Assigned Numbers Authority).

Les ports 1024 à 49151 sont appelés ports enregistrés et les ports 49152 à 65535 sont les ports dynamiques (ou privés).

I.6.2Notion de protocoles

Un protocole est une série d'étapes à suivre pour permettre une communication harmonieuse Entre plusieurs ordinateurs.

Internet est un ensemble de protocoles regroupés sous le terme "TCP-IP" (**T**ransmission **C**ontrol **P**rotocol/**I**nternet **P**rotocol). Voici une liste non exhaustive des différents protocoles qui peuvent êtres utilisés :

HTTP : (Hyper Texte Transfert Protocol) : c'est celui que l'on utilise pour

Consulter les pages web.

- FTP : (File Transfert Protocol) : C'est un protocole utilisé pour transférer des fichiers.
- SMTP : (Simple Mail Transfert Protocol) : c'est le protocole utilisé pour envoyer des mails.
- POP : C'est le protocole utilisé pour recevoir des mails
- Telnet : utilisé surtout pour commander des applications côté serveur en lignes
- IP (internet Protocol) : L'adresse IP vous attribue une adresse lors de votre connexion à un serveur.

Les protocoles sont classés en deux catégories

-Les protocoles où les machines s'envoient des accusés de réception (pour permettre une gestion des erreurs). Ce sont les protocoles "orientés connexion"

-Les autres protocoles qui n'avertissent pas la machine qui va recevoir les données sont les protocoles "non orientés connexion"

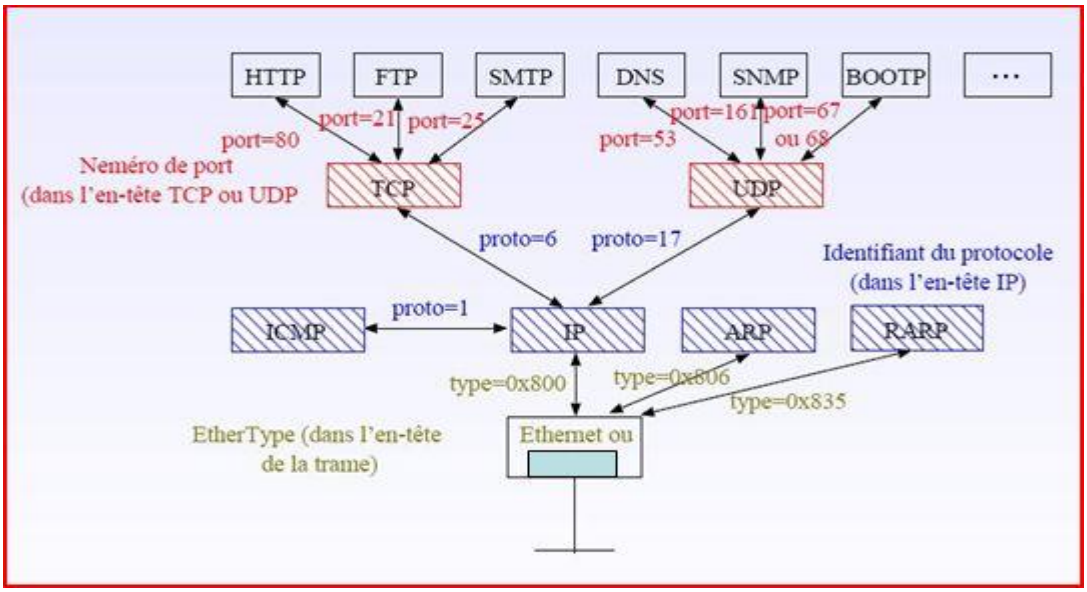

Figure I.2: protocole et port

I.7. Les Sockets

Port : Entrée réseau de la machine sur laquelle un serveur « écoute » en attendant des connexions / requêtes

Socket c'est un Tuyau entre deux programmes

Exemple

Client sur machine 1 appelle serveur sur machine 2 / port 53.

La connexion s'établit, le canal de communication est ouvert

Il devient possible de communiquer suivant un protocole application (par exemple DNS). [3]

I.7.1. API (application program interface) socket

Mécanisme d'interface de programmation Permet aux programmes d'échanger des données.

Les applications client/serveur ne voient les couches de communication qu'à travers API socket

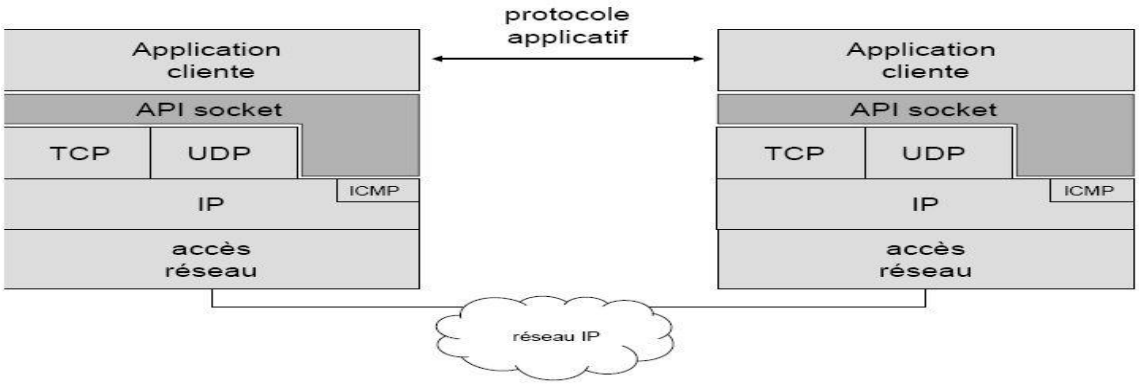

Figure I.3: Sockets

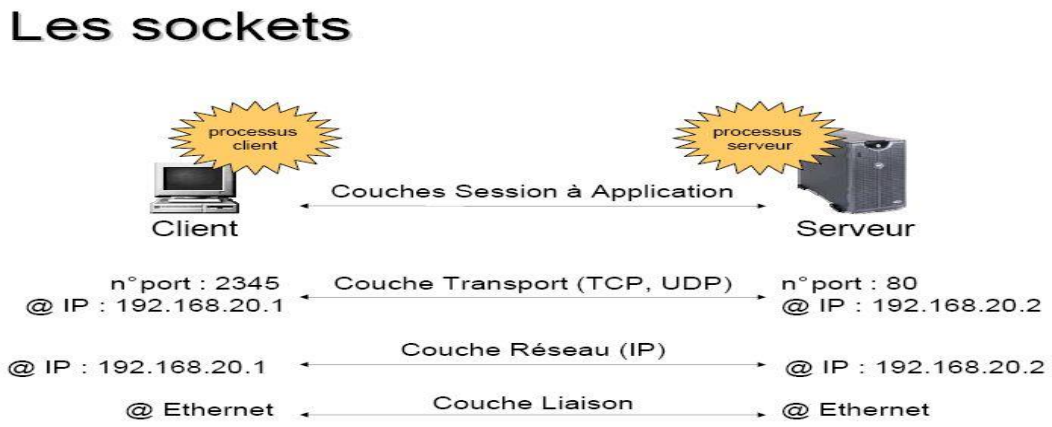

Figure I.4: Modèle OSI et sockets

I.8. Les middlewares

On appelle middleware (ou logiciel médiateur en français), littéralement " élément du milieu", l'ensemble des couches réseau et services logiciel qui permettent le dialogue entre les différent composent d'une application répartie. Ce dialogue se base sur un protocole applicatif commun, défini par l'API du middleware.

Middleware c'est une interface de communication universelle entre processus. Il représente véritablement la clef de voûte de toute application client/serveur.

C'est logiciel qui assure les dialogues entre clients et serveurs hétérogènes, ou entre 2 applicatifs n'ayant pas les même API. Fait de l'« adaptation de protocole » des couches 5, 6, 7 du modèle OSI

L'objectif principal du middleware est d'unifier, pour les applications, l'accès et la manipulation de l'ensemble des services disponibles sur le réseau, afin de rendre l'utilisation de ces derniers presque transparente.

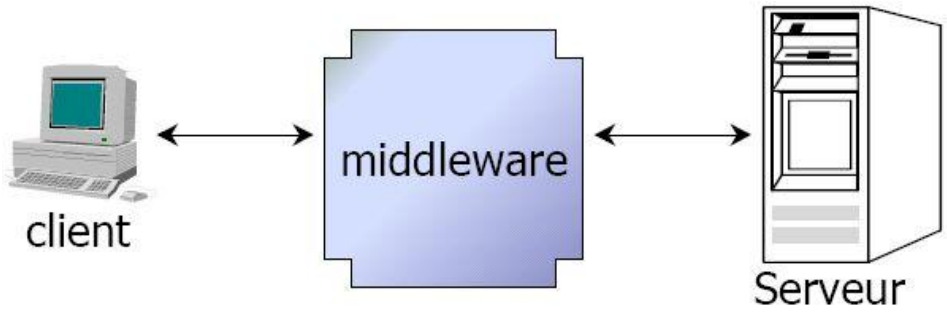

Figure I.5: Middlewares

I.8.1. Les services des middlewares

Un middleware susceptible de rendre les services suivants :

- **Conversion** :
  Services utilisé pour la communication entre machine mettant en œuvre des formats de données différentes

- **Adressage** :
  Permet d'identifier la machine serveur sur laquelle est localisé le service demandé afin d'en déduire le chemin d'accès. Dans la mesure du possible.

- **Sécurité** :
  Permet de garantir la confidentialité et la sécurité des données à l'aide de mécanismes d'authentification et de cryptage des informations.

- **Communication** :
  Permet la transmission des messages entre les deux systèmes sans altération. Ce service doit gérer la connexion au serveur, la préparation de l'exécution des requêtes, la récupération des résultats et la déconnexion de l'utilisation.

Le middleware masque la complexité des échanges inter-applications et permet ainsi d'élever le niveau des API utilisées par les programmes. Sans ce mécanisme, la programmation d'une application client/serveur serait complexe et difficilement évolutive.

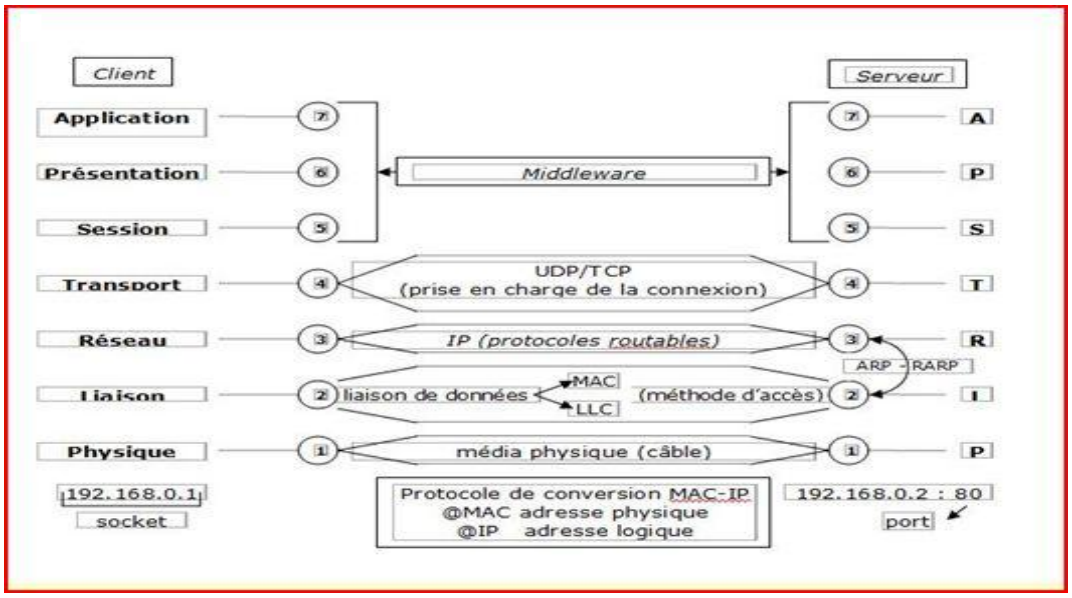

Figure I.6: Modèle OSI et middleware

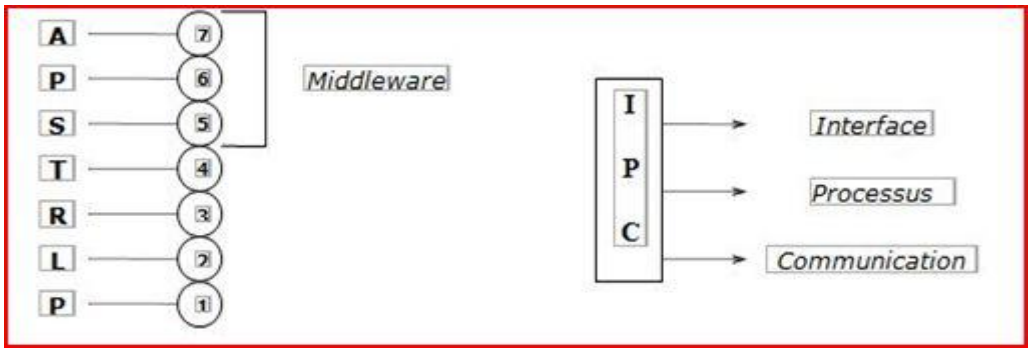

Figure I.7: IPC

I.9. RCP

Appel de procédure distance : RPC = Remote Procédure Call

Technique permettant d'appeler une procédure distante comme une procédure locale en rendant transparente les messages échangés

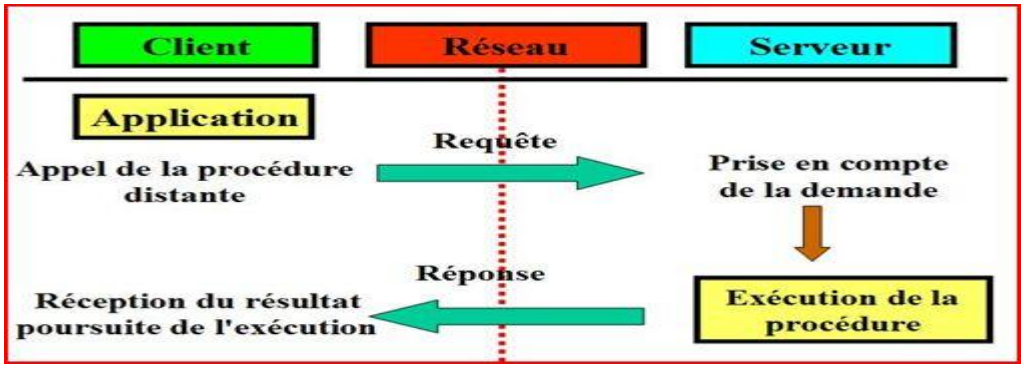

Figure I.8: Appel de procédure à distance

Dans les RPC, la communication client/serveur : peut se faire par datagramme (par paquets), ou par connexion (flux de données dans un canal).

Elle peut être :

Synchrone : le serveur attend la requête du client ; et pendant que le serveur fait le traitement, le client attend,

Asynchrone : pendant qu'un des acteurs traite les informations, l'autre acteur, au lieu d'attendre, continue de « vivre sa vie ». Il est interrompu (par une interruption système) quand l'autre acteur lui envoie de nouveau de l'information, afin qu'il aille traiter ce flot entrant. [3]

I.10.Conclusion

Le modèle client /serveur est la base de tous les services réseaux informatique, c'est pour cela nous sommes intéressé par l'étude de se modèle. Le but de ce chapitre c'est décrire les différentes notions de base de ce modèle comme le middleware, les protocoles, les sockets et l'appel de procédure à distance. Afin de mieux configurer et administrer les différents services qui font l'objectif de notre projet. La configuration de la carte réseaux est une étape fondamentale et importante pour faire cette configuration sur la distribution l'linux Ubuntu, qui fait l'objet du deuxième chapitre de notre mémoire.