

## RAPPORT DE PROJET DE FIN D'ETUDES

Filière  
Ingénieurs des Travaux

# Etude, Conception et réalisation d'une plate-forme de télé-facturation téléphonique journalière

Elaboré par :

**Taghlet Moez  
Barhoumi Mourad**

Encadré par :

**Mr Bouallègue Ridha**

**Année universitaire : 2001/2002**

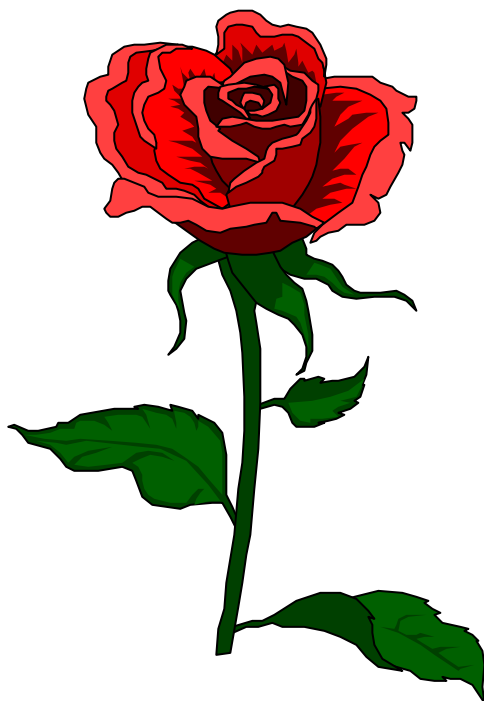
# DEDI CACES

Nous dédions ce présent travail  
A nos très chers parents,

A nos femmes et enfants,

A nos frères et sœurs,

A tous nos amis.



# REMERCIEMENTS...

Nous tenons à présenter nos sincères remerciements et notre profonde gratitude à toutes les personnes ayant contribué de près ou de loin à la réalisation de notre projet de fin d'études. Nos remerciements vont particulièrement à notre encadreur Monsieur BOUALLEGUE Ridha, notamment à nos chers collègues Keyssi Slaheddine , Nasri Imed et Farah Slaheddine pour leurs soutien moral et technique. Ce que nous ressentons de joie et de fierté en fin d'année ne peut s'exprimer que par une seule phrase : « Merci à ceux qui étaient, et sont encore près de nous, à nous guider pour un avenir réussi »

# Liste des tables

	<b>Page :</b>
Table 2.1 : Enumération des entités de la base de données télé facturation-----	27
Table 5.1 : Les méthodes http -----	48
Table 5.2 : Informations envoyées par le client -----	48
Table 5.3 : Les codes de réponses http -----	49
Table 5.4 :Les entêtes de réponses http -----	49
Table 5.5 : Quelques types MIME les plus utilisés -----	49
Table 6.1 : Variables CGI relatives à la requête -----	57
Table 6.2 : Variables CGI relatives à la connexion -----	57
Table 6.3 : Variables CGI relatives au serveur -----	57

# Liste des figures

	<b>Page :</b>
Figure 1.1 : Rôle du système d'information dans la création de la base de données ---	10
Figure 1.2 : Système d'accès aux données -----	12
Figure 1.3 : Les étapes de la conception d'une base de données -----	14
Figure 2.1 : Représentation du modèle hiérarchique -----	17
Figure 2.2 : Représentation du modèle réseau -----	18
Figure 2.3 : Représentation du modèle relationnel -----	18
Figure 2.4 : Représentation du modèle objet -----	18
Figure 3.1 : Architecture de la solution proposée -----	26
Figure 3.2 : Architecture de la solution réalisable -----	28
Figure 4.1 : Architecture fonctionnelle des SGBD-R -----	34
Figure 5.1 : Exemple d'une configuration client-serveur -----	44
Figure 5.2 : Représentation du protocole http -----	47
Figure 5.3 : Format de la ligne d'état d'une réponse -----	48
Figure 6.1 : Architecture de la solution proposée -----	55
Figure 7.1 : Procédé d'adresse pour réseau d'exploitation et de maintenance via X25	68
Figure 7.2 : Type de réseau et valeurs AFI, IDI et DSP associés -----	69
Figure 7.3 : Différentes possibilités de connexion de l'interface X25 -----	70
Figure 7.4 : Séquence de gestion et d'installation X25 -----	69

# SOMMAIRE

	Page :
Introduction générale : -----	5
Cahier des charges de projet : -----	8
<b>Chapitre 1 : Système de données et problématiques de la conception</b>	
Introduction : -----	9
I-Notion de système de données :-----	10
II- Variété d'expression au système de données :-----	11
III- Problématiques de la conception :-----	13
VI- Une conception par étapes :-----	13
Conclusion : -----	14
<b>Chapitre 2 : Conception détaillée de la base de donnée -----</b>	
Introduction : -----	15
I-Objecrifs de l'étape conceptuelle : -----	16
II- Principes et fondements de la conception : -----	16
II.1- Analyse : -----	16
II.1.1- Analyse du réel : -----	16
II.1.2- Modélisation : -----	16
II.2- Modèle conceptuel de données : -----	17
II.2.1- Concepts de base : -----	17
II.2.2- Qualités du modèle conceptuel de données : -----	17
II.2.3- Les différents modèles de base de données : -----	17
III-Le modèle relationnel : -----	19
III.1- Les caractéristiques du modèle relationnel :-----	19
III.2- Les concepts de base du modèle :-----	19
III.2.1- Domaine : -----	19
III.2.2- Relation : -----	19
III.2.3 -Attribut : -----	19
III.2.4 -Clé et contrainte d'intégrité : -----	20
III.2.4.1- Contrainte d'intégrité sur les clés : -----	20
III.2.4.2 -Contrainte d'entité : -----	20
III.2.4.3 -Contrainte référentielle : -----	20
III.3- Avantages et inconvénients du modèle relationnel: -----	21
III.3.1- Avantages : -----	21
III.3.2 -Inconvénients : -----	21
IV- Base de données du projet télé facturation : -----	21
IV.1- Inventaire des entités et associations : -----	21
IV.2- Conceptualisation : -----	21
IV.2.1- Les relations de la base des données : -----	22
IV.2.2 -Le modèle relationnel des données : -----	24
Conclusion : -----	24

<b>Chapitre 3 : Architecture de la solution proposée -----</b>	
Introduction: -----	25
I- Architecture de la solution proposée : -----	26
II- Solution réalisable sur la maquette EWSD de l'ISETCOM: -----	28
III- Structure de base de la solution proposée :-----	29
III.1- Le client :-----	29
III.2- Le serveur Web : -----	29
III.2.1- Le site Web: -----	29
III.2.2- Les modules de traitements : -----	29
IV- Plan de navigation : -----	29
IV.1- Page d'accueil : -----	29
IV.2- Activation de l'inscription : -----	30
IV.3- Mise à jour des renseignements: -----	30
IV.4- Activation de la consultation : -----	30
IV.5- Consultation de la facture : -----	30
V- Modèle de traitement : -----	30
V.1- Le modèle requête-réponse: -----	30
V.2- Traitement des exceptions: -----	31
Conclusion : -----	31
<b>Chapitre 4 : Réalisation et implémentation de la base de données -----</b>	
Introduction : -----	32
I- Les SGBD relationnelles : -----	33
I.1- Définition : -----	33
I.2- Fonctionnalités : -----	33
I.3- Architecture et fonctionnement : -----	34
I.4- Le langage SQL: -----	35
I.4.1- Intérêt de SQL: -----	35
II- Choix technique pour le projet : MYSQL -----	36
II.1- caractéristiques de MYSQL : -----	36
II.2- Pourquoi a-t-on choisit MYSQL ? : -----	36
II.3- Obtention : -----	37
II.3.1- Installation et compilation : -----	37
III- Serveur MYSQL : -----	37
III.1- Connexion au serveur MYSQL : -----	37
III.2- Système de droit et contrôle d'accès : -----	38
III.2.1- But : -----	38
III.2.2- Noms d'utilisateurs et mot de passe : -----	38
III.2.3- Fonctionnement : -----	38
III.3- Lancement du serveur MYSQL: -----	39
III.4- Arrêt du serveur : -----	39
IV- Programmation MYSQL : -----	39
IV.1- Création de la base de données PFE : -----	39
IV.2- Création des tables de la base : -----	40
IV.3- Test de la base : -----	41
IV.4- Alimentation de la base : -----	41
Conclusion : -----	42

<b>Chapitre 5 : Serveur web et interfaces clients -----</b>	
Introduction : -----	43
I- Système client : -----	44
I.1- Performances des applications orientées Web : -----	44
I.2- Développement Web coté client : -----	45
I.3- Utilisation des formulaires : -----	45
I.4- Les formulaires HTML : -----	45
II- Le protocole http : -----	47
II.1- Requête http : -----	47
II.2- Réponse http : -----	48
III- Le serveur Web : -----	50
III.1- Principes et fonctionnement : -----	50
III.2- Performances d'un serveur Web : -----	50
III.2.1- .1- Exécution des scripts et autres programmes:-----	50
III.2.2- compression et cryptage des données: -----	50
III.2.3- Accès aux données: -----	51
IV- Choix technique : APACHE -----	51
IV.1- Pourquoi APACHE ? -----	51
IV.2- Installation : -----	51
IV.3- Configuration : -----	52
IV.4- Démarrage du serveur APACHE : -----	52
V- Sécurité du serveur APACHE : -----	52
Conclusion : -----	53
<b>Chapitre 6 : Interface Web- base de données -----</b>	
Introduction : -----	54
I- choix techniques : CGI -----	56
I.1- Définition :-----	56
I.2- Spécification : -----	56
I.3- Installation : -----	56
I.4- Normalisation des CGI : -----	56
I.4.1- Variables relatives à la requête : -----	56
I.4.2- Variables relatives à la connexion : -----	57
I.4.3- Variables relatives au serveur : -----	57
I.4.4- Entrées-sorties : -----	58
II- Le langage Perl : -----	58
II.1- Pourquoi Perl ?-----	58
II.2- Perl et CGI : -----	59
II.3- Accès aux base de données : -----	59
III- Aspect sécurité : -----	59
III.1- Aperçu général : -----	59
III.2- Comment se protéger ? : -----	60
IV- Organigrammes :-----	60
Conclusion : -----	66



## **Chapitre 7 : Installation et mise en service de l'interface x 25 à la maquette EWSD**

Introduction : .....	67
I- Présentation du réseau OSI : .....	68
II- Adresses d'identification du terminal : .....	68
II.1- Adresse X 25 : .....	68
II.2- Adresse DTE : .....	69
II.3- Adresse NSAP : .....	69
III- Connexion directe (type de réseau X 25LC) : .....	70
IV- Installation et procédure de gestion : .....	71
IV.1- Câble IOP pour connexion directe (X25LC) : .....	72
IV.2- Base de données pour la création d'un lien X25 : .....	72
V- Protocole FTP : .....	73
V.1- Présentation : .....	73
V.2- Les commandes FTP : .....	73
V.2.1- Contrôle d'accès : .....	74
V.2.2- Paramètres de transfert : .....	74
V.2.3- Services FTP : .....	74
V.3- Les réponses FTP : .....	74
Conclusion : .....	75
Conclusion générale : .....	76
Bibliographie : .....	77
Glossaire : .....	78

## Introduction générale

---

De nombreuses sociétés se lancent dans la vente de services de téléphonie, toutes ont besoin d'un service de facturation. Bons nombres d'opérateurs possèdent le niveau nécessaire dans la vente et le marketing pour faire croître leur base clients. Cependant, ils ne disposent pas toujours du niveau ou des ressources pour gérer la facturation. Dès lors, toute l'attention portée à cette fonction détourne l'opérateur de son activité principale.

La facturation téléphonique exige un système fiable avec une capacité de croissance illimitée. Ordinateurs et équipements d'impression ainsi que personnel entraîné techniquement sont les éléments clés. La croissance du business nécessite l'augmentation adéquate et rapide de ces capacités. Le financement pour obtenir et maintenir ces ressources est considérable et cela pour une opération ne durant pas plus que trois à quatre jours par trimestre.

En TUNISIE avec l'apparition des systèmes de commutation électromécanique l'idée était de photographier les compteurs mécaniques afin d'éviter l'intervention humaine et en vue d'assurer la confidentialité, puis les transmettre vers le centre de facturation pour traitement. L'évolution technologique nous a permis ensuite d'installer les centres électroniques qui traitent les compteurs de taxation d'une façon logique et les sauvegarder dans les fichiers correspondants. Avec cette technologie on a eu l'occasion de sauvegarder les fichiers de taxation sur un support magnétique puis optique et de les envoyer chaque trimestre vers le centre de facturation pour le

traitement nécessaire. Aller plus loin avec le développement des systèmes d'exploitation de ces centres a dégagé l'avantage suivant :

Les centres électroniques offrent la possibilité aujourd'hui de transférer les fichiers automatiquement en appliquant le protocole FTAM et en se connectant à un réseau publique de données X 25 ou à travers le réseau numérique à intégration de service RNIS . A ce stade notre premier opérateur s'aligne avec la technologie de pointe et installe un serveur pour la collecte des fichiers de taxation dès les centres AXE et EWSD en première phase, et de cette façon on a un service de facturation évolué.

Mais de nos jours les clients professionnels attendent plus qu'une simple facture. Il est courant aujourd'hui de fournir les détails des factures sur CD ou via internet. Il est aussi essentiel de fournir des outils d'analyse ainsi qu'un service d'assistance lors de la fourniture de données électroniques. Lorsque la base de clients est d'ordre important, la mise sous enveloppe et l'affranchissement des factures deviennent également des tâches majeures. Une facture imprimée reste toujours une exigence et bien que la fourniture des détails via Internet soit une réalité.

Dans ce contexte ont a pensé à bénéficier les clients de TUNISIE TELECOM d'un service en ligne qui assure un accès à l'information à tout moment et à tout lieu , afin de pouvoir maîtriser leurs consommation téléphonique et aussi de minimiser les problèmes des retards de paiement des factures puisque les clients connaissent leurs montants dès la fin de trimestre.

Ce rapport présente les principales facettes de l'établissement d'une plate forme de télé facturation journalière téléphonique.

Les objectifs de ce produit seront détaillés et les besoins étudiés dans la partie consacrée au cahier de charge et à ses spécifications.

La conception de la base de données en premier lieu puis celle du schéma général de l'application font objectif des trois chapitres suivants : système de donnée et problématique de la conception (Chapitre 1), conception détaillée de la base de données (Chapitre 2) et architecture de la solution proposée (Chapitre 3).

Ensuite on va entamer la partie consacrée aux détails techniques du développement et de l'environnement de travail ainsi que sur les choix et décisions sur la plate-forme et le processus d'implémentation : réalisation et implémentation de la base de données (Chapitre 4) étude, installation , configuration du serveur Web et interfaces clients (Chapitre 5), développement des modules d'interface entre la base de données et le serveur (Chapitre 6), installation et configuration des interfaces X25 pour le transfert des fichiers de taxation de la maquette EWSD de l'ISCT'Com (chapitre 7).

## **Cahier des charges du projet**

Pour mettre en œuvre le projet de la plate -forme de télé facturation téléphonique en ligne, un ensemble de règles et de contraintes fonctionnelles, technique et méthodologique sont exigés.

Du point de vue fonctionnel, ce projet devra permettre aux clients de consulter leurs factures téléphoniques journalièrement.

Techniquement, l'objectif est de concevoir la base de données et développer les modules qui permettent l'accès à la base de données depuis le site Web. Chaque méthode de conception et outil de développement utilisé devra être expliqué.

La base de données ainsi que les modules seront implémentés sur une plate-forme technique ( Serveur de base de données, serveur Web, ...).

Finalement le développement de ce produit doit suivre un cycle à quatre phases : la conception, le développement, l'implémentation et les tests.

## **Chapitre 1**

### **Notion de système de données et problématiques de la conception**

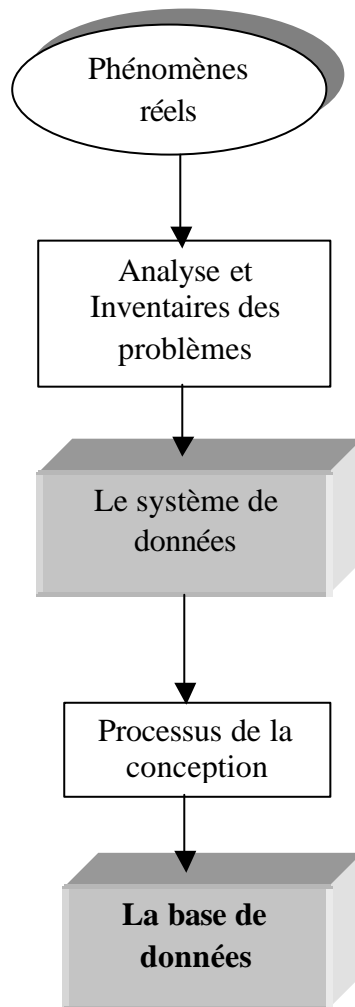
---

#### **Introduction :**

La fonction essentielle de tout système de données est de fournir à ceux qui l'utilisent des représentations appropriées et pertinentes de la réalité qu'ils ne sont pas en mesure d'observer directement. A cette fin, il convient de créer et de mémoriser des collections de données aussi représentatives que possible de la réalité en question et qui, exploitées par des programmes de traitements appropriés (des interfaces CGI dans notre cas), permettent de reconstruire les images de la réalité. Le projet de la télé facturation, fait intervenir plusieurs acteurs qui interagissent d'une ou plusieurs façons avec des institutions et organisations de natures différentes (financière, juridique, administrative ) et dont ces interactions sont conditionnés par un ou plusieurs facteurs. D'autre part, ce projet comme tout autre projet informatique, mobilise des ressources humaines, financières et techniques pour lesquelles l'adoption d'une démarche de conception non solide serait source de gaspillage fatale. L'objectif de ce chapitre est de dégager une méthode qui sera suivie lors de la conception de la base de données et qui sera déduite suite à une étude et analyse des problèmes qui seront rencontrées et des objectifs attendus.

**I- Notion de système de données :**

Lorsque le système d'information est mis au point (fonctionnel), il contient une représentation des processus réels qui intéressent l'application, sous forme de collection de données qui sont mémorisées dans la base de données. Nous admettons que les données de la base sont mémorisées sur différents support de stockage et sont gérées selon les techniques appropriées. Mais avant de créer et faire fonctionner la base de données, un important travail de conception doit être réalisé qui aboutit au système de données. Le système de données peut se définir comme la description du contenu des collections de données qui constitueront la base de données. Le système de données illustre comment on devra représenter, par des données, les types d'information et des processus de traitements de ces informations qui sont pris en compte dans ce projet.



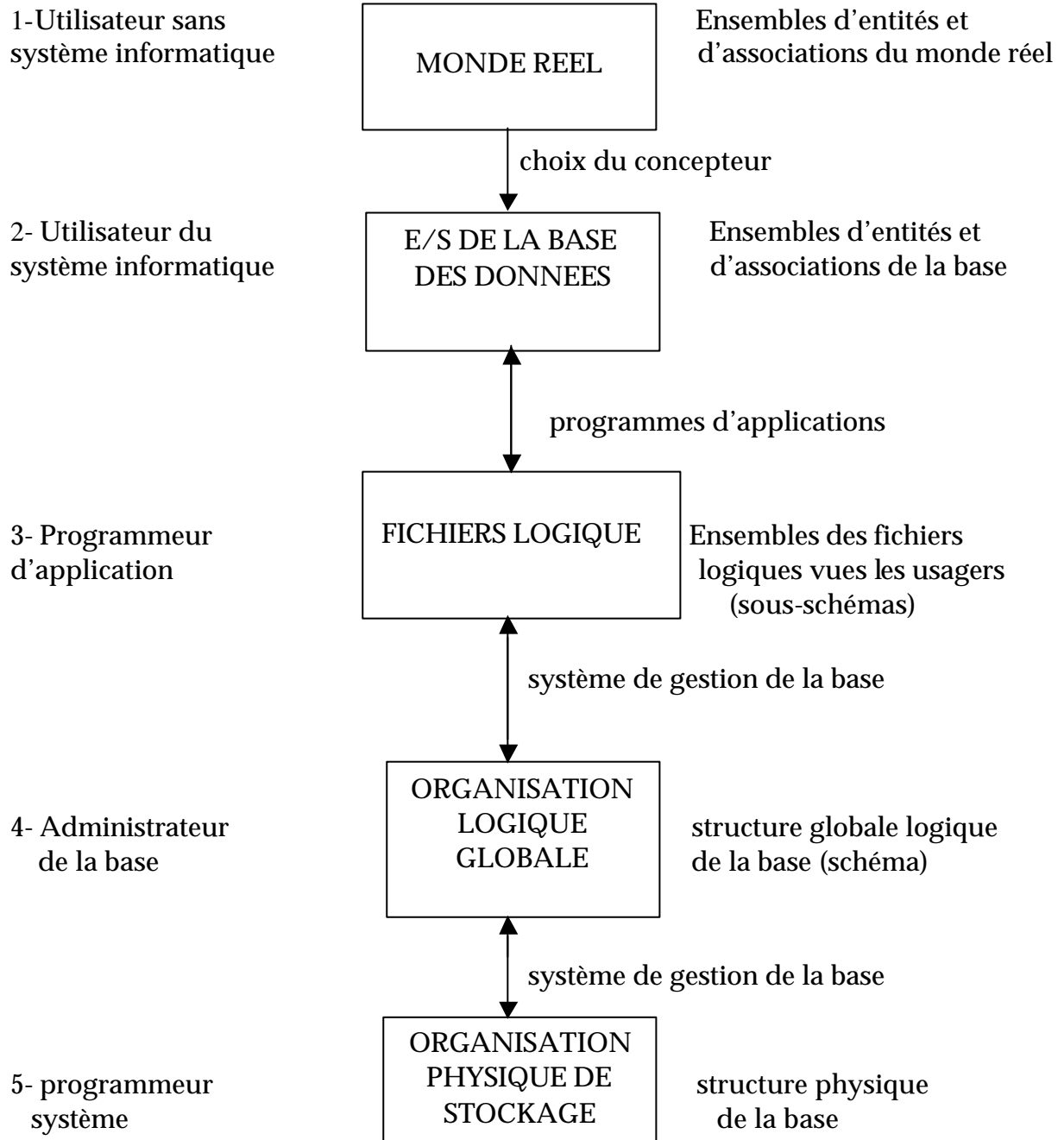
**Figure 1.1 Rôle du système d'informations dans la création d'une base de données.**

**II- Variété d'expression au système de données :**

La diversité des rôles qui incombent au système de données et des personnes qui ont à l'utiliser conduit à souhaiter que l'expression du système de données ne soit pas unique. Ainsi, dans le cas d'une application de télé facturation en ligne, ces intervenants se scindent en 4 catégories :

- **La cible et le bénéficiaire principal de ce service qui est l'abonné ( tel qu'il sera défini dans la conception détaillée de son profil).**
- **Le gestionnaire administratif (Agence commerciale de télécommunications, centre de facturation) qui intervient pour la mise à jour des données et l'exécution d'opérations diverses.**
- **L'ensemble de l'équipe des analystes et des programmeurs qui visent le développement de nouvelles applications, la modification d'applications déjà exploitées, l'optimisation des performances des programmes et scripts d'accès aux données.**
- **Le responsable de la base de données et en gros du serveur de base de données qui améliore l'organisation de l'ensemble des données compte tenu de la totalité des applications.**





**Figure 1.2 : Système d'accès aux données**

### III- Problématiques de la conception :

Si l'on tente de classer les problèmes rencontrés dans la définition d'un système de données, on fait apparaître trois grandes facteurs :

- **La représentation** : elle concerne essentiellement l'expression de la sémantique
  - **Quels** sont les entités, objets, actions , ...qui seront représentés dans la base de données ?
  - **Par** quels concepts ou combinaisons de concepts les représenter ?
  - **Par** quelles données traduire chaque concept ?
  
- **L'utilisation** : ayant à répondre à des demandes d'information :
  - **Quel** usage peut-on faire de la représentation ?
  - **Quel** modèle retenir pour structurer les données qui correspondent aux concepts ?
  - **Comment** structurer la représentation de façon à y parvenir ?
  - **Quels** sont les cheminements et relations logiques qu'il faut privilégier dans l'espace des données ?
  
- **L'exploitation** : pour répondre de manière opérationnelle aux demandes d'informations :
  - quelle implantation des données assurera une exploitation satisfaisante ?
  - quelle organisation des données et que moyen d'accès retenir ?
  - comment assurer la sécurité et la confidentialité des données ?

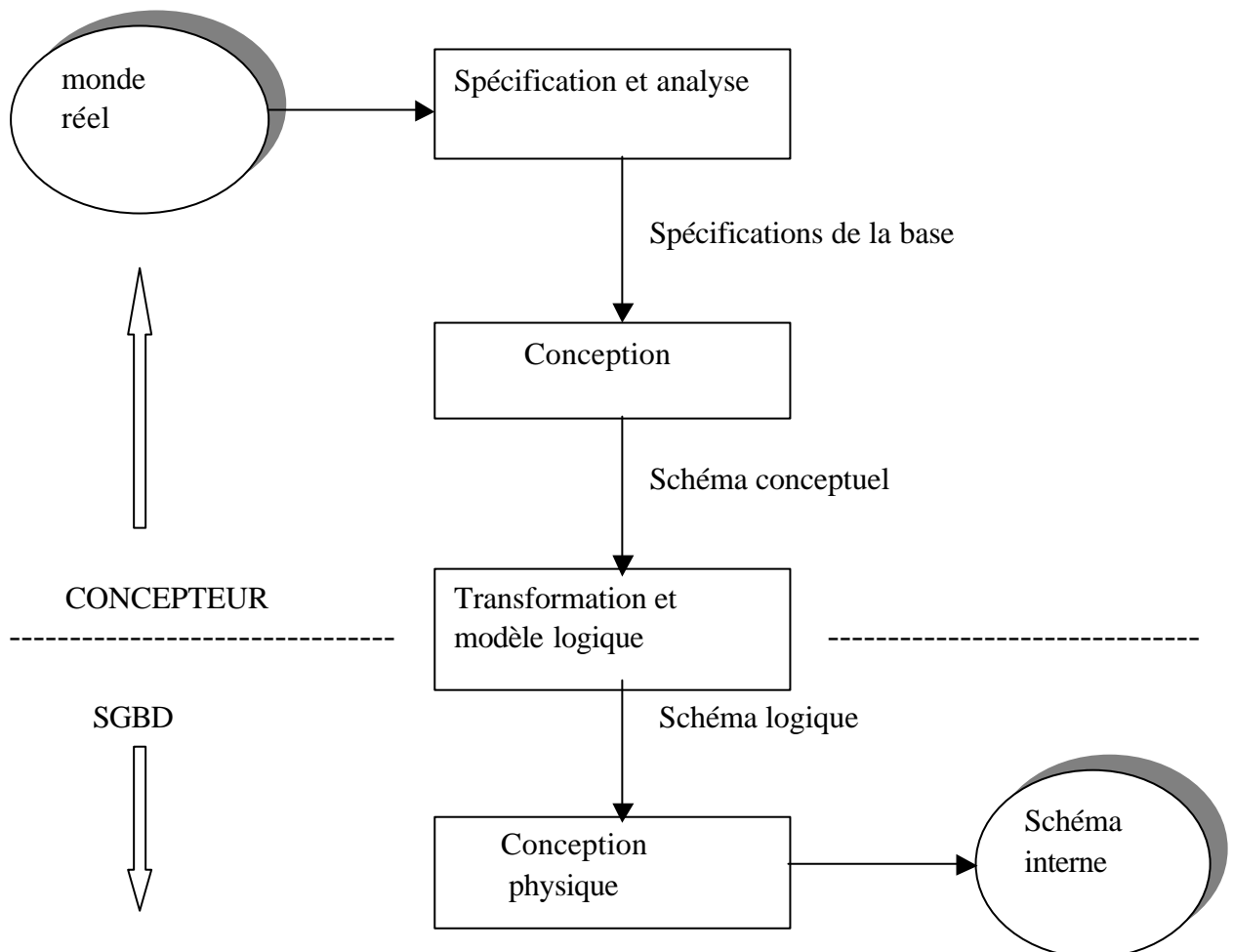
### IV- Une conception par étape :

Tous les problèmes que nous venons d'évoquer justifient que l'on systématise la tâche de conception en proposant une démarche par étapes à suivre pas à pas. Des méthodes d'analyse généralement utilisées pour concevoir les systèmes d'information, nous retenons surtout qu'il y a à traiter séparément les problèmes dits physiques et les problèmes dits fonctionnels ou logiques. Ceci nous conduit à un découpage du processus de la conception en 3 étapes successives :

-L'étape conceptuelle : elle doit permettre une bonne représentation des faits pris en compte .

-L'étape logique : elle doit assurer la définition de la solution où tous les conditions d'usage des données ont été prises en compte.

-L'étape physique : elle doit aboutir à l'expression définitive de la solution technique à partir de laquelle la réalisation de la base de données devient possible.



**Figure 1.3 : Les étapes de la conception d'une base des données**

### Conclusion :

L'élaboration d'un système de données repose sur la description et la classification naturelle réel pour aboutir à un schéma descriptif primaire, qui, par des transformations approfondies et analyse détaillée, permettront d'aboutir au schéma conceptuel, objectif finale de la conception de la base de données. Une fois le processus de la conception est défini par étapes en fixant les exigences diverses de cohérences, fidélité, extension,... l'objectif est de s'y baser pour aboutir à une conception satisfaisante de la base de données réelle qui sera décrite par le schéma du modèle conceptuel des données.

## **Chapitre 2**

### **Conception détaillée de la base de données**

---

#### **Introduction :**

On se propose de concevoir le schéma conceptuel de la base de données permettant la représentation la plus fidèle possible du processus de la télé facturation en ligne ( inscription et consultation), de façon à en permettre une exploitation ultérieure par les modules de traitements.

## **I – Objectifs de l'étape conceptuelle :**

Il s'agit de traduire, par une structure de données, la sémantique du réel que l'on veut prendre en compte dans le système de données et cela revient à :

- Exprimer de manière formelle le résultat du processus d'analyse de la partie de la réalité considérée.
- Exprimer ce résultat au moyen de types, dans une démarche structuraliste qui débouche sur une représentation claire, explicite, cohérente et surtout condensée de la variété des faits réels.
- Exprimer ce résultat en toute abstraction de considération d'ordre technique qu'il conviendra d'introduire au moment de la réalisation. Il importe, en effet, à ce stade d'assurer l'indépendance de la solution conceptuelle par rapport aux solutions ultérieures de la réalisation. Sans cette indépendance, on se priverait de choisir entre plusieurs solutions de réalisation qu'il est encore prématuré de définir.
- Exprimer, en s'efforçant de les intégrer, les multiples points de vue des utilisateurs. En effet, à la variété des faits réels correspond aussi la diversité des attentes des différents utilisateurs qui auront accès à la base de données.

## **II- Principes et fondement de la conception :**

### **II.1- Analyse :**

#### **II.1.1-Analyse du réel :**

Le monde réel est perçu comme un système abstrait. Ce système abstrait se traduit par : des classes d'entités, des propriétés sur ces classes, des liaisons entre ces classes. Le système abstrait est décrit par un schéma conceptuel.

#### **II.1.2-Modélisation :**

Principes généraux à respecter :

- Le schéma conceptuel doit être libre de toute considération non significative du système abstrait (organisation physique des données, aspects particuliers à un usager tels que des formats de messages...).
- Tous les aspects du système abstrait doivent être dans le schéma conceptuel aucun d'eux ne doit intervenir ailleurs en particulier dans des programmes d'application indépendants du schéma conceptuel.

## II.2- Modèle conceptuel des données :

### II. 2.1- Concepts de base :

Entité (exemple abonné), attribut(exemple nom), valeur de l'attribut (exemple 01837516), association(liaison perçu entre entités).

### II.2. 2- Qualités du modèle conceptuel des données :

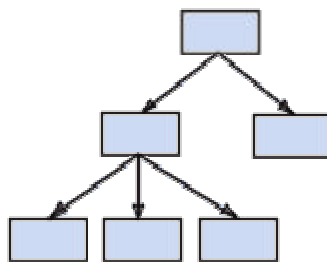
Il importe que le schéma du modèle conceptuel des données ait les qualités suivantes :

- Clarté et simplicité : pour décrire les faits, le schéma conceptuel intègre des notions dont la signification n'est pas ambiguë et reste directement parlante.
- Cohérence : il n'y a pas de contradictions ou confusion dans les représentations d'un même projet.
- Complétude : sans rechercher l'exhaustivité, qui est d'ailleurs sans limites, la représentation rend compte de l'essentiel des phénomènes à considérer.
- Fidélité : pour parvenir à une représentation sans déformations.
- Non redondance : on ne prend en compte, pour la représentation, que les éléments qui sont nécessaires.

### II.2.3- Les différents modèles de bases de données :

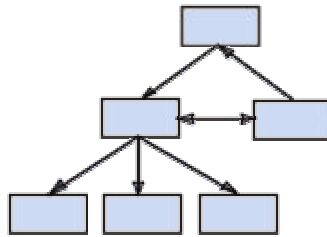
Les bases de données sont apparues la fin des années 60, à une époque où la nécessité d'un système de gestion de l'information souple se faisait ressentir. Il existe cinq modèles de SGBD, différenciés selon la représentation des données qu'elle contient :

**le modèle hiérarchique:** les données sont classées hiérarchiquement, selon une arborescence descendante. Ce modèle utilise des pointeurs entre les différents enregistrements. Il s'agit du premier modèle de SGBD.



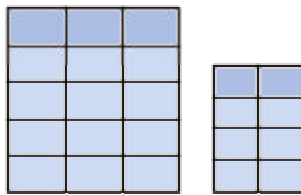
**Figure 2.1 : Représentation du modèle hiérarchique**

**le modèle réseau:** Comme le modèle hiérarchique ce modèle utilise des pointeurs vers des enregistrements. Toutefois la structure n'est plus forcément arborescente dans le sens descendant.



**Figure 2.2 : Représentation du modèle réseau**

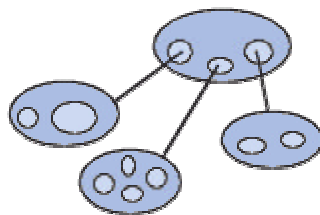
**le modèle relationnel SGBDR:** les données sont enregistrées dans des tableaux à deux dimensions (lignes et colonnes). La manipulation de ces données se fait selon la théorie mathématique des relations .



**Figure 2.3 : représentation du modèle relationnel**

**le modèle déductif:** les données sont représentées sous forme de table, mais leur manipulation se fait par calcul de prédicats.

**le modèle objet SGBDO:** les données sont stockées sous forme d'objets, c'est-à-dire de structures appelées classes présentant des données membres. Les champs sont des instances de ces classes.



**Figure 2.4 : représentation du modèle objet**

De nos jours les bases de données relationnelles sont les plus répandues. En effet, le modèle relationnel présente plusieurs avantages, qui sont nos arguments concernant notre choix de la méthode relationnelle.

### **III- Le modèle relationnel :**

#### **III.1 - Les caractéristiques du modèle relationnel :**

- Il permet un haut degré d'indépendance des programmes d'applications par rapport à la présentation interne des données, en voyant que des tables logiques et étant affranchis de la gestion des méthodes de stockage et des chemins d'accès.
- Il offre un langage de définition et de manipulation des données normalisé, basé sur des requêtes non procédurales qui permettent des transferts d'ensembles de lignes constituant de véritables sous-tables.
- Il fournit une base solide pour traiter les problèmes de cohérence des données, en supportant notamment des contraintes d'intégrité.
- C'est un modèle extensible permettant de modéliser et manipuler simplement des données tabulaires, mais pouvant être étendu pour gérer des objets complexes voire structurés.

#### **III.2- Les concepts de base du modèle :**

Le modèle relationnel représente l'information dans une collection de relations. Intuitivement, on peut voir une relation comme une table à double entrée, voir même comme un fichier. Chaque ligne de table peut être vue comme un fait décrivant une unité du mode. Une colonne de la table est appelée un attribut.

##### **III.2.1 – Domaine :**

Les ensembles de données de départ sont appelés des domaines. Un domaine est en théorie un ensemble de valeurs .

##### **III.2.2- Relation :**

Le modèle relationnel est basé sur le concept de relation. La relation est définie par son nom, sa liste de constituants et par son prédicat. Les relations modélisent les idées qui relient les objets (codés par des valeurs) des domaines entre eux. Une représentation des relations sous forme de tables a été retenue dans SGBD3 relationnels, si bien que l'on confond relation et table.

##### **III.2.3-Attribut :**

Chaque colonne d'une relation ou table correspond à un domaine qui peut apparaître plusieurs fois . Afin de pouvoir distinguer les colonnes et rendre leur ordre sans importance tout en permettant plusieurs colonnes de même domaine, il est nécessaire d'associer un nom à chaque colonne, d'où la notion d'attribut.



### III.2.4- Clé et contraintes d'intégrité :

Un schéma de base de données est un ensemble de schémas de relation  $S=\{R_1, R_2, \dots, R_n\}$  et un ensemble de contraintes d'intégrité CI. Une contrainte d'intégrité est une propriété du schéma, invariant dans le temps. On peut distinguer plusieurs catégories de contraintes. Les contraintes structurelles, définissent plus précisément la structure des associations entre les données (le modèle de données les supporte en partie) et les contraintes sur les valeurs donnent des relations entre les données. La plupart des contraintes ne sont pas supportées par le modèle de données et doivent donc être codées par les programmeurs dans des programmes d'application. Les règles d'intégrité sont des assertions qui doivent être vérifiées par les données contenues dans la base de données. Le modèle relationnel privilégie trois types de règles de contraintes d'intégrité : contrainte d'intégrité sur les clés, les contraintes d'entité et les contraintes référentielles.

#### III.2.4.1- Contrainte d'intégrité sur les clés :

Par définition, tous les valeurs d'une relation sont distincts deux à deux (puisque'il s'agit d'un ensemble). Il est également intéressant de définir des sous-ensembles du schéma qui permettent d'identifier de manière unique une valeur (par exemple dans l'exemple de l'abonné, l'attribut No-CIN permet d'identifier un abonné). Ces sous-ensembles du schéma s'appellent des clés. Le système doit donc garantir l'unicité des valeurs de clés (un seul valeur abonné peut avoir une valeur de No-CIN donnée). Cette clé sera indiquée de manière graphique en mettant en gras les attributs la composant.

#### III.2.4.2- Contrainte d'entité :

Elle impose l'existence d'une clé documentée permettant l'identification de tout valeur d'un relation. Aucune composante d'une clé primaire ne peut être nulle. Une clé doit permettre une identification unique d'une valeur dans la base.

#### III.2.4.3- Contrainte référentielle :

Elle spécifie un lien hiérarchique entre deux tables, l'une devenant dépendante de l'autre. Une contrainte d'intégrité permet en particulier de repérer les associations obligatoires entre entités.

En résumé, le modèle relationnel est très simple de point de vue modélisation de données. Il repose sur les concepts rappelés dans la figure ci-dessous. Sa simplicité fait force notamment pour le développement d'applications client-serveur.

### III.3- Avantages et inconvénients du modèle relationnel :

#### III.3.1- avantages :

1. La simplicité pour l'utilisateur qui manipule des tables.

2. L'indépendance de l'utilisateur vis à vis de la structure logique, de la structure physique et des stratégies d'accès aux données.
3. Puissance et uniformité de représentation : la théorie mathématique permet une conception rigoureuse et algorithmique du schéma.
4. Puissance d'expression de la sécurité des données : contrôles dépendant du contenu, du contenant, des contextes.
5. Existence d'interfaces non procéduraux pour usagers non informaticiens.

### **III.3.2- inconvénients :**

1. Nécessité d'un SGBD puissant.
2. Perte d'un peu d'indépendance logique lors de normalisation.
3. Difficulté qu'on a de définir une couverture minimale.

Dans ce qui suivra, on va adopter cette méthode relationnelle en commençant par un inventaire des différentes entités (relations et tables), leurs attributs et leurs clés, puis on applique cette conceptualisation pour tracer le modèle relationnel des données.

## **IV- Base de données du projet :**

### **IV.1- Inventaire des entités et association :**

L'étude et l'analyse des spécifications, du cahier de charges et d'un ensemble de documents administratifs (fiche téléphonique de l'abonné, fiche du dossier d'inscription chez l'agence commerciale des télécommunications, centre de facturation...) nous ont conduit à dégager les entités suivantes :

Abonné ou client, numéro du téléphone, type (client physique ou client moral), catégorie de l'abonnement (fixe ou prépayé), compteur téléphonique (initial ou journalier).

### **IV.2- Conceptualisation :**

La conceptualisation consiste à :

- Traduire par des schémas de relation les entités et associations qui ont été retenues lors de l'analyse .
- Transformer ces schémas par processus les normalisations qui visent successivement à :
  - Tracer le schéma de dépendance fonctionnelle entre les différentes entités.
  - Supprimer les données redondantes pour alléger l'espace de stockage des données.

- Tracer le du modèle relationnel des données.

#### IV.2.1- Les relations de la base de données :

Cette étape consiste à fixer les attributs, clés et propriétés de chaque relation.

Abonné et/ou relation	Désignation symbolique	Définition & description
abonné entité sociale, existe suite à raccordement au réseau téléphonique publique	dn	Numéro d'annuaire : clé primaire de la table
	cin	N° de la carte d'identité : clé primaire de la table
	nom	Nom
	prénom	Prénom
	compteur_journalier	Compteur d'impuls téléphonique (maqu EWSD)
Catégorie Catégorie d'abonné et d'abonnement	dn	Numéro d'annuaire Clé primaire de la table
	catégorie	Ordinaire ou prépayé Physique ou moral
	login	Nom d'utilisateur
	password	Mot de passe supérieure à 6 caractères Clé primaire de la table

<p>Client_physique</p> <p>Informations communiquées Par une personne physique Pour aboutir à l'inscription</p>	nom	Nom
	prénom	Prénom
	rue	Rue
	ville	Ville
	numéro	numéro
	code_postal	code postal
	login	Nom d'utilisateur clé primaire de la table
<p>Client_moral</p> <p>Informations communiquées Par une personne qui représente Une société , entreprise...</p>	password	Mot de passe supérieure à 6 caractères clé primaire de la table
	Nom_social	Nom
	rue	Rue
	ville	Ville
	numéro	Numéro
	Code_postal	Code postal
	Login	Nom d'utilisateur clé primaire de la table
	password	Mot de passe supérieure à 6 caractères clé primaire de la table

Compteur  Valeur du compteur d'impulsions obtenu suite au transfert Du fichier de taxation à partir de la maquette EWSD .	dn	Numéro d'annuaire Clé primaire de la table
	compteur_initial	Compteur téléphonique du début du trimestre
	compteur_journalier	Compteur téléphonique journalier transférer dès la Maquette EWSD
	login	Nom d'utilisateur
	password	Mot de passe supérieure caractères

**Table 2.1 : énumération des entités de la base de données télé facturation**

#### IV.2.2- Le modèle relationnel de données :

Voici notre modèle relationnel des données :

abonne(dn , cin, nom, prénom, compteur\_journalier)

client\_physique(nom, prénom, rue, ville, numéro, code\_postal, login, password)

client\_moral(nom\_social, rue, ville, numéro, code\_postal, login, password)

compteur(dn, compteur\_initial, compteur\_journalier, login, password)

catégorie : ( dn, login, password, catégorie).

#### Conclusion :

La base de données représente le pilier de base dans ce projet puisqu'elle décrit l'application dans ses composantes (données) . La réalisation physique de la base de données ainsi que les technologies jointes seront détaillées plus loin, mais la question qui commence à passer au premier plan est « comment sera exploitée la base données ? »La réponse sera l'objectif du chapitre consacré aux technique du Data-W e b. Mais pour pouvoir accéder à une base de données via une architecture client-serveur (dans notre cas le protocole http et les interfaces Web), il faudra préciser et détailler les interactions entre les postes clients ainsi que la spécification des interfaces d'accès avec les serveurs d'applications et de base de données, permettant de dégager le schéma de base de la navigation . On commence ainsi à affranchir les premières étapes de la conception des traitements et c'est ce qui nous conduit alors dans le chapitre qui suit à entamer la conception entière de l'application et particulièrement l'interaction de la base de données avec les autres composants du projet dont essentiellement le serveur Web.

## **Chapitre 3**

### **Plate-forme de télé facturation : architecture de la solution proposée**

---

#### **Introduction :**

La solution qu'on va proposer au cours de ce projet , intègre des facteurs qui s'articulent autour de la base de données. Ce chapitre va donc être le premier pas dans la conception de l'architecture générale de l'application et la spécification des différents traitements à effectuer au moyens des formulaires HTML à présenter dans les pages W e b du site au cours de la navigation. De la même occasion, on propose le schéma qui gère les flux de données entre le différents intervenants : le client, le serveur W e b, le serveur de base de données, les passerelles de l'inscription ainsi que la consultation de télé facturation . Finalement on donnera la configuration réalisable dans la maquette de commutation EWSD de l'ISSET'Com.

# I-Architecture de la solution proposée :

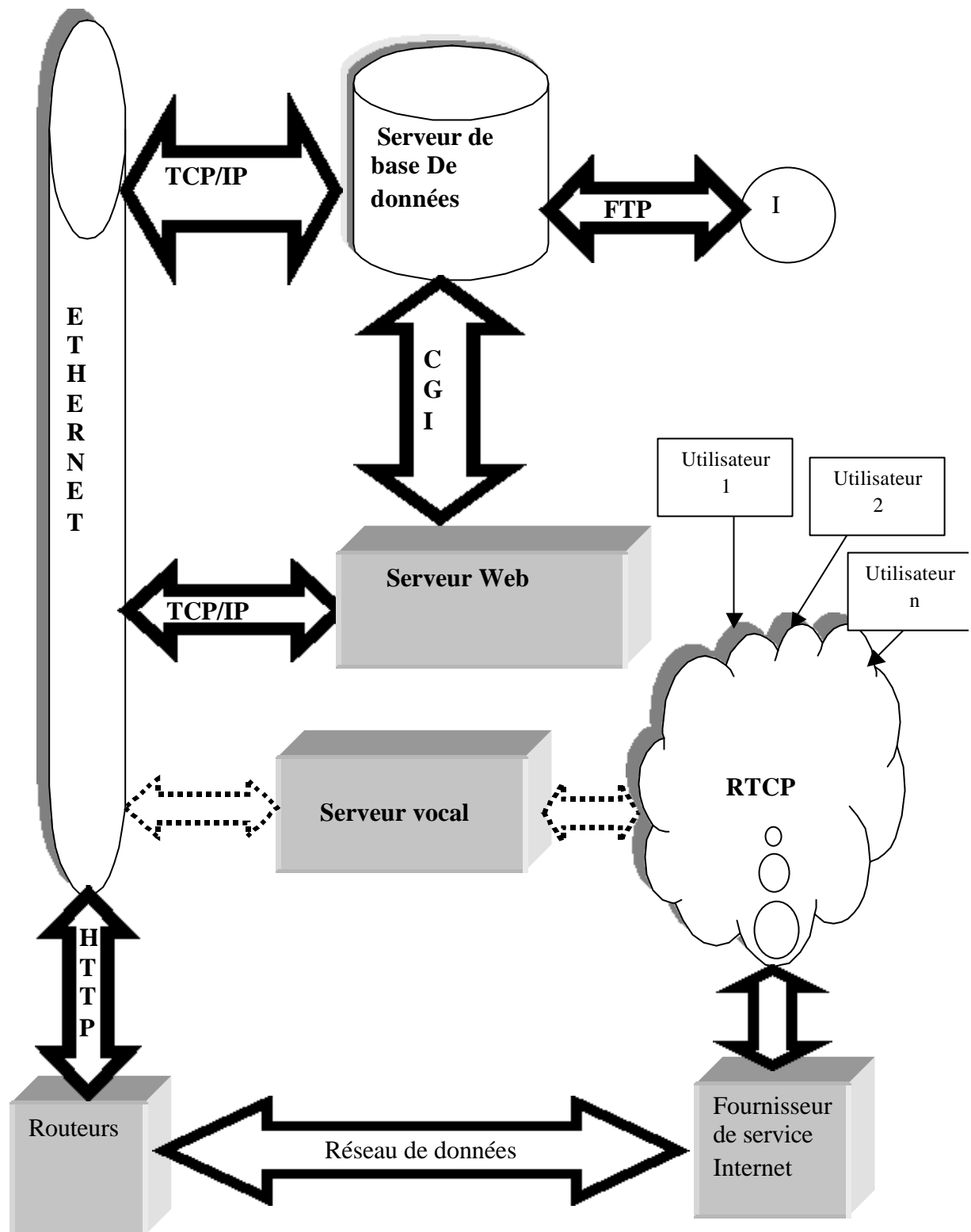
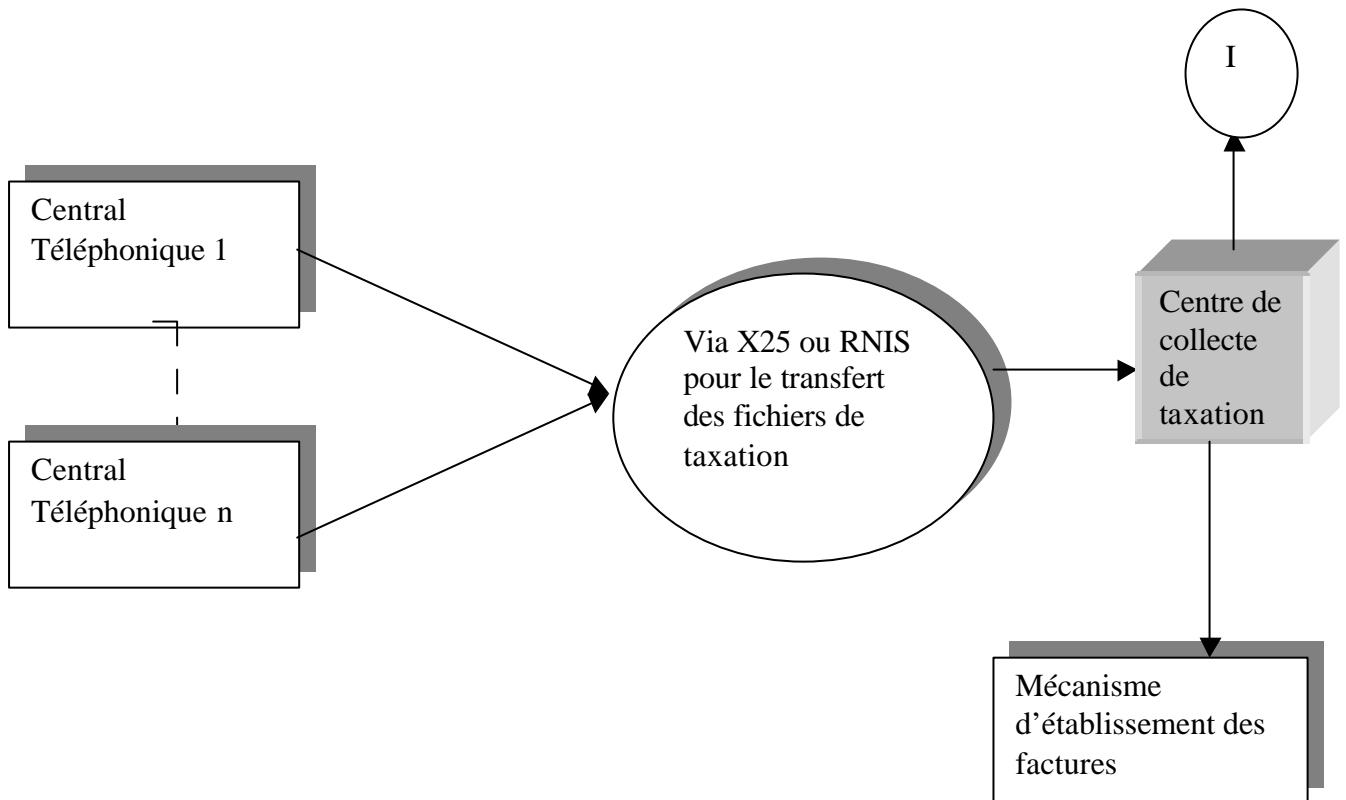


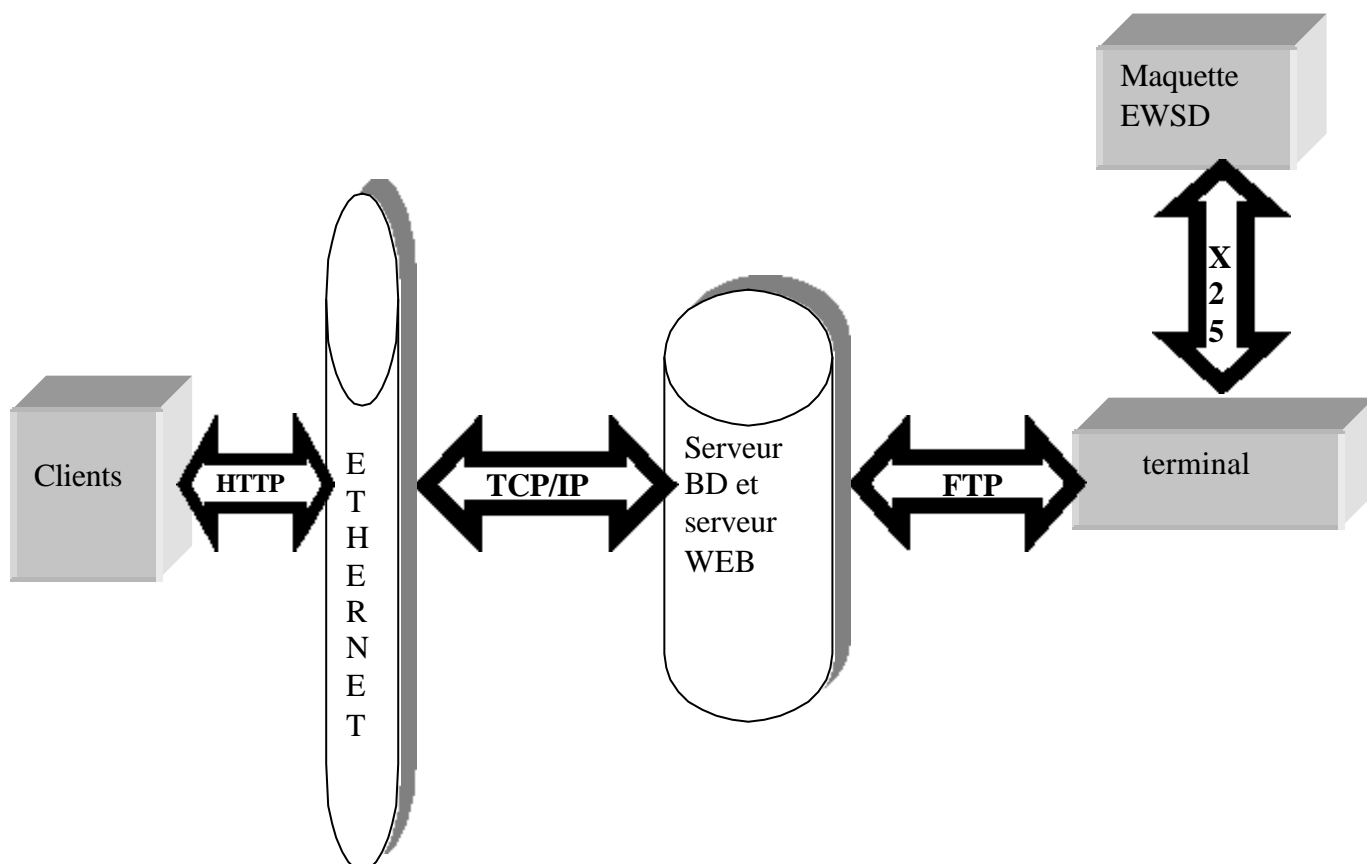
Figure 3.1 : Architecture de la solution proposée



**Figure 3.1 : Architecture de la solution proposée**



## II- Solution réalisable sur la maquette EWSD de L'ISETCOM



**Figure 3.2 : Architecture de la solution réalisable**

➤ Pour mettre en œuvre cette solution sur la maquette EWSD de L'ISETCOM, on a été obligé d'installer les modules et d'introduire la base de données nécessaires pour assurer le transfert automatique des fichiers de taxation, les détails de l'installation ainsi que l'introduction de la base de données vous sera communiqué dans l'un des chapitres qui suivent.

### **III- Structure de base de la solution proposée :**

Les composants principaux de la solution proposées sont les suivants :

#### **III.1-Le client :**

Cette partie concerne essentiellement l'accès du client au site Web choisi. En principe, on n'a pas une grande intervention à ce niveau, mais on doit prendre en compte certains paramètres de l'environnement client tel que la connexion, le type du navigateur (supporte les protocoles de sécurité comme SSL ou non,...) .

#### **III.2-Le serveur Web :**

Le serveur joue un rôle principal car il représente le point de connexion à partir duquel sera reçues ou envoyées les requêtes et les réponses. L'écriture des pages au format HTML nécessite un navigateur pour visualiser le résultat, d'autre part l'écriture des scripts CGI nécessite la présence d'un serveur http tournant sur la machine. On parle surtout de deux entités primordiales :

##### **III.2.1-Le site Web :**

Le site Web hébergé dans le serveur contient la rubrique de l'inscription et la consultation en ligne de la facture téléphonique qui sera présentée sous forme d'un lien hypertexte dans la page d'accueil.

##### **III.2.2- Les modules de traitements :**

Ce sont les scripts logés sur le serveur et qui permettent de traiter les requêtes des clients et générer toute information dynamique à partir de la base de données.

### **IV- Plan de navigation :**

Pour pouvoir développer les modules de traitements, il est nécessaire d'établir le scénario de navigation à suivre au cours de l'opération de l'inscription et de la consultation de la facture depuis la connexion au site jusqu'à la confirmation finale du bon déroulement de l'opération.

#### **IV.1- Page d'accueil :**

C'est la "Home-Page" du site concerné. Elle doit impérativement contenir le lien vers les rubriques ; inscription client\_physique, inscription client\_moral, consultation de la facture.

**IV.2- Activation de l'inscription :**

Ce formulaire inclut des informations générales sur les modalités de l'inscriptions des clients physiques ou morales . On doit aussi laisser l'occasion aux clients de saisir leurs nom d'utilisateur et leurs mot de passe qui seront utile pour la consultation de leurs factures.

**IV.3-Mise à jour des renseignements :**

Cette page est envoyée une fois que l'identification est bien déroulée et informe les clients de la réussite de l'inscription . une mise à jour dans la base de données est exécutée immédiatement.

**IV.4-Activation de la consultation :**

Ce formulaire englobe les informations de connexion précédemment saisies dans la rubrique d'inscription.

**IV.5- consultation de la facture :**

Une fois que le client ait été reconnu, la transaction s'effectue et un formulaire de facture sera généré.

**V- Modèle de traitement :****V.1- Le modèle requête-réponse :**

Tous les modules de traitements à développer se basent sur le même principe et ce du début de l'opération jusqu'à la fin :

- Envoi de formulaire depuis le serveur.
- Saisie de données de la part du client et retransmission du formulaire vers le serveur.
- Récupération des données saisies et traitement adéquat par un script serveur.
- Génération de page HTML réponse vers le client en fonction des traitements effectuées.

Les script doivent être modulaires, c'est à dire un ensemble de procédures et fonctions dont chacune effectue une tâche particulière(identification, mise à jour des données...). Le choix de la technique de ces script et du langage de développement est un détail technique à négocier plus loin dans ce projet.

**V.2- Traitement des exceptions :**

Rien n'empêche le client de communiquer des informations erronées même par inadvertance. Ces informations ne doivent pas échapper au contrôles des scripts qui doivent intégrer un mécanisme de redirection en cas d'erreur et de vérification de données à insérer dans la base. Un bon nombre de ces contrôles dits contrôles intrinsèques peuvent être transposés vers le client et en charge par des script locaux au système hôte de celui ci.

**Conclusion :**

En cette phase du projet, on vient d'entamer la première partie qui est celle de la conception. L'objectif est dès maintenant de trouver la meilleure concrétisation de cette conception par une implémentation de la base de données et le développement des modules de traitements par les systèmes et outils garantissant le meilleur rendement en matière de fonctionnalités et robustesse. Ceci va donc nous amener dans les chapitres suivants à réaliser ces objectifs en adoptant à chaque fois une démarche bien précise visant d'abord l'étude globale du système ou langage à utiliser, ensuite un bref comparatif des technologies utilisées et finalement le choix bien fondé d'un outil de travail pour l'implémentation et le développement.

## **Chapitre 4**

### **Réalisation et implémentation de la base de données**

---

#### **Introduction :**

Une approche de solutions techniques aux problèmes mentionnées lors de la conception passe par une organisation du système d'information qui doit devenir plus intégré mais aussi plus évolutif. Ceci nécessite tout d'abord de systèmes ouverts, obéissants à des standards permettant le choix d'un grand nombre de produit sur le marché. La majorité de ces produits sont articulés autour de moteur relationnel qui apporte une meilleure maîtrise de l'information et une plus grande souplesse d'évolution : c'est le standard industriel. Ce chapitre a donc pour objectif de maîtriser la technologie de base des SGBD-R, faire un choix technique et en expliciter les raisons et finalement d'implémenter la base de données du projet.

## I-Les SGBD relationnelles :

### I.1-Définitions :

Un **S**ystème de **G**estion de **B**ases de **D**onnées est un ensemble de programmes qui permettent à des utilisateurs de créer et maintenir une base de données. Les activités supportées sont la définition d'une base de données, la construction d'une base de données et la manipulation de données (principalement ajouter, supprimer, retrouver des données). Les SGBD commerciaux les plus connus sont Oracle, Sybase, Ingres, Informix et DB2.

### I.2- Fonctionnalités des SGBD :

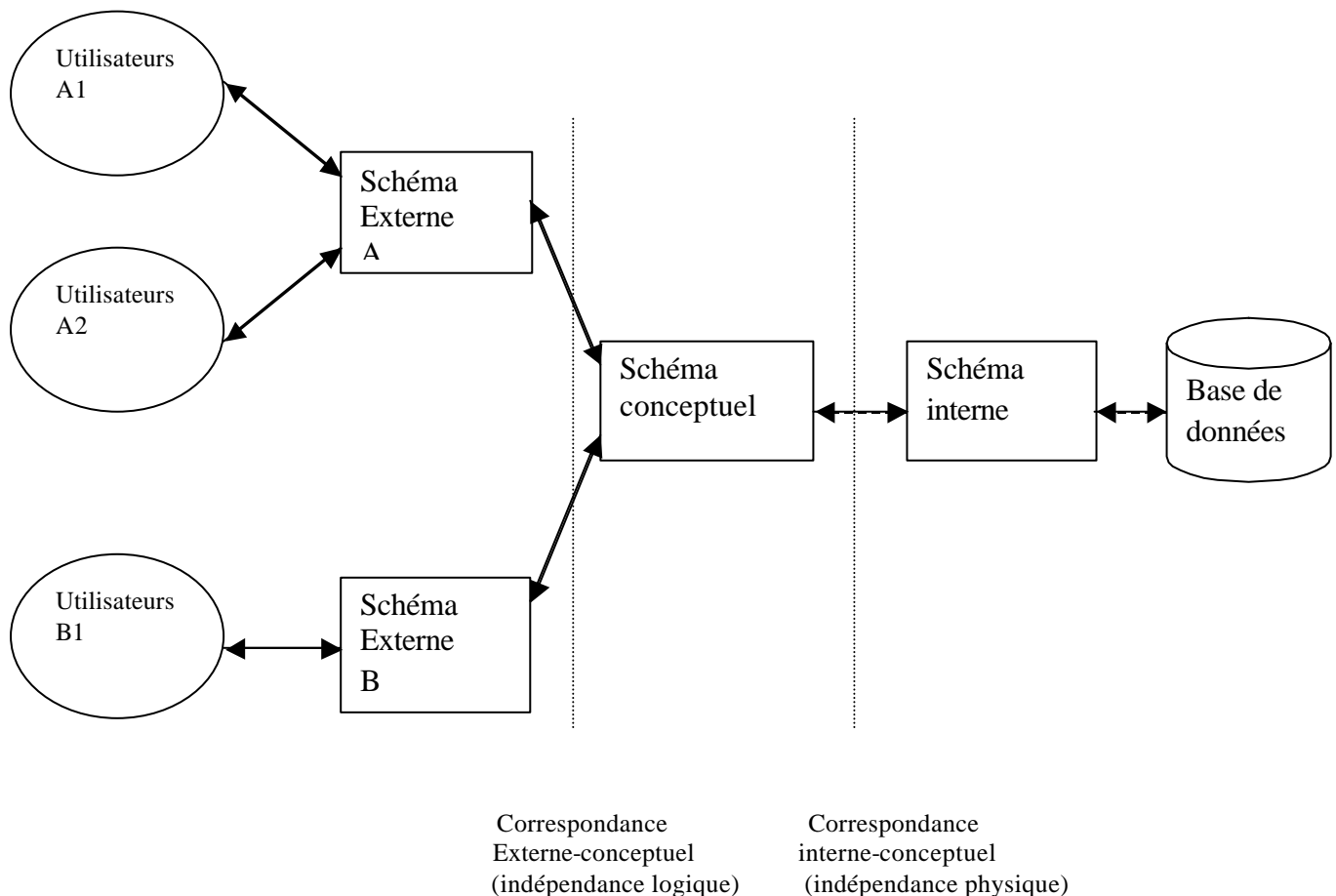
On donne ici les caractéristiques souhaitables des SGBD :

- **Contrôler la redondance d'informations :** la redondance d'informations pose différents problèmes. Un des objectifs des bases de données est de contrôler cette redondance, voire de la supprimer, en offrant une gestion unifiée des informations complétée par différentes vues pour des classes d'utilisateurs différents.
- **Partage de données :** une base de données doit permettre d'accéder la même information par plusieurs utilisateurs en même temps. Le SGBD doit inclure un mécanisme de contrôle de la concurrence basé sur des techniques de verrouillage de données (pour éviter par exemple le fait de lire une information dont on est en train de la mettre à jour).
- **Gérer les autorisations d'accès :** une base de données étant multi – utilisateurs, se pose le problème de la confidentialité des données. Des droits doivent être gérés sur les données, droits de lecture, mise à jour, création,... qui permettent d'affiner la notion de vue utilisateur.
- **Offrir des interfaces d'accès multiples :** un SGBD doit offrir plusieurs interfaces d'accès, correspondant aux différents types d'utilisateurs pouvant s'adresser à lui.
- **Représenter des relations complexes entre les données :** Un SGBD doit permettre de représenter des données inter-reliées de manière complexe. Cette facilité s'exprime à travers le modèle de données sous-jacent au SGBD. Chaque modèle de données offre ses propres concepts pour représenter les relations.
- **Vérifier les contraintes d'intégrité :** un schéma de base de données se compose d'une description des données et de leurs relations ainsi que d'un ensemble de contraintes d'intégrité. Une contrainte d'intégrité est une propriété de l'application à modéliser qui renforce la connaissance que l'on en a. On peut classer les contraintes d'intégrité, en contraintes structurelles et contraintes dynamiques. Les SGBD commerciaux se portent automatiquement un certain nombre de contraintes structurelles, mais ne prennent pas en compte les contraintes dynamiques (elles doivent être codées dans les programmes d'applications).

- **Assurer la sécurité et la reprise après panne :** une base de données est souvent vitale dans le fonctionnement d'une organisation, et il n'est tolérable qu'une panne puisse remettre en cause sans fonctionnement d'une manière durable. Les SGBD fournissent des mécanismes pour assurer cette sécurité. Le premier mécanisme est celui des transactions qui permet d'assurer un comportement atomique à une séquence d'actions (elles s'exécutent complètement avec succès ou elle est annulée). Une transaction est une séquence d'opérations qui fait passer la base des données d'un état cohérent à un nouvel état cohérent. L'exemple typique est celui du débit crédit pour la gestion d'une carte bancaire. Ce mécanisme permet de s'affranchir de petites pannes (style coupure de courant). En ce qui concerne les risques liés aux pannes disques, les SGBD s'appuient sur un mécanisme de journalisation qui permet de régénérer une base de données automatiquement à partir d'une version de sauvegarde et de journal des mouvements.

### I.3- Architecture et fonctionnement :

La plupart des SGBD suivent l'architecture standard Ansi / Sparc qui permet d'isoler les différents niveaux d'abstraction nécessaires pour un SGBD.



**Figure 4.1 : Architecture fonctionnelle des SGBD-R**

cette architecture est définie sur trois niveaux :

- Niveau interne ou physique : décrit le modèle de stockage des données et les fonctions d'accès.
- Modèle conceptuel ou logique : décrit la structure de la base globalement à tous les utilisateurs (limite la redondance). Le schéma conceptuel est produit par une analyse de l'application à modéliser et par intégration des différentes vues utilisateurs. Ce schéma décrit la structure de la base indépendamment de son implantation.
- Niveau externe : correspond aux différentes vues des utilisateurs. Chaque schéma externe donne une vue sur le schéma conceptuel une classe d'utilisateurs.

#### **I.4- Le langage SQL :**

##### **I.4.1- Intérêt de SQL :**

Tous les systèmes de gestion de données utilisent SQL pour l'accès aux données ou pour communiquer avec un serveur de données. SQL(Standard Query Language) est né à la suite des travaux mathématiques de Codd, travaux qui ont fondé les bases de données relationnelles. SQL, défini d'abord chez IBM, a subi trois tentatives de normalisation en 86, 89 et 92(SQL 2 ou SQL 92). Nous présentons trois raisons fondamentales qui justifient l'utilisation de SQL.

- D'une part, la structuration et la manipulation des données sont devenues très complexes. Pour une application de taille moyenne, la base de données contient fréquemment plusieurs tables fortement interconnectées. Il est donc hors de question de manipuler les données de façon algorithmique traditionnelle. Une requête SQL dans un langage logique simple remplace donc bien avantageusement plusieurs dizaines de lignes d'un langage de programmation comme C ou Cobol.
- D'autre part, l'architecture client-serveur est omniprésente, développer une application dans un environnement hétérogène n'est possible que parce que la communication entre l'applicatif client et le serveur est réalisée par des primitives SQL normalisées.
- Enfin, les applications à développer sont devenues de plus en plus complexes. Le profil du programmeur a fortement changé. Il doit maintenant traiter des données de plus en plus volumineuses, intégrer les techniques de manipulation des interfaces, maîtriser la logique événementielle et la programmation orientée objet, tout cela dans un contexte d'architecture client-serveur où se côtoient les systèmes d'exploitation de réseaux hétérogènes. L'accès et la manipulation des données ne sont pas que l'un des aspects de la conception et de la réalisation de programmes. On cherche donc à acquérir un environnement de développement performant qui prend en charge un grand nombre de tâches annexes. Des outils de développement sont apparus pour permettre au développeur de se concentrer sur



l'application proprement dite : générateurs d'écrans, de rapports, de requêtes, d'aide à la conception de programme, de connexion à des bases de données distantes via les réseaux. Dans tous ces outils, la simplicité et la standardisation de SQL font que SQL est utilisé chaque fois qu'une définition, une manipulation, ou un contrôle de données est nécessaire. SQL est donc un élément central entre les divers composants d'un environnement de développement dans une architecture client-serveur.

## **II –Choix technique pour le projet : MYSQL**

### **II.1- Caractéristiques de MYSQL :**

MySQL est un véritable serveur de base de données SQL, Multi-Utilisateurs et multi-threadés. Les principaux objectifs de MYSQL sont la rapidité, la robustesse et la facilité d'utilisation. MYSQL a été originellement développé suite au besoin d'un serveur SQL qui puisse gérer des grandes bases de données de manière plus rapide. Il est utilisé depuis 1996 dans un environnement de plus de 40 bases de données contenant 10000 tables dont plus de 500 contiennent plus de 7 millions d'enregistrements. C'est environ 100 giga de données critiques. La base sur laquelle MYSQL est construite est formée d'un ensemble de routines de production exigeant. Même si MYSQL est encore en développement, il propose déjà un ensemble de fonctionnalités riches et extrêmement utiles.

### **II.2- Pourquoi a-t-on choisi MySQL?**

Notre choix pour travailler avec un SGBDR comme MySQL est essentiellement dû aux caractéristiques de ce système qui répondent aux exigences en matière de services en ligne tournant sur des serveurs pouvant atteindre des pics de charge (connexions) importants :

- Complètement multi-thread et utilise les threads du noyau. Cela signifie qu'il peut utiliser plusieurs CPU ainsi qu'une garantie d'autonomie et robustesse de fonctionnement. MySQL est lui-même écrit en C++ et testé avec bon nombre de compilateurs différents.
- Support et compatibilité avec la majorité des langages de programmation comme C, C++, Eiffel, Java, Perl, PHP, Python,... et avec la plupart des systèmes d'exploitation du marché comme LINUX, WINDOWS, AIX, SOLARIS, FreeBSD,... ce qui augmente la portabilité de nos développements sans aucun souci d'incompatibilité de notre code avec la plate-forme. De plus, il est compatible avec le standard ODBC (Open-Database-Connectivity) pour Windows95 (avec source).
- Toutes les fonctions ODBC2.5 et d'autres sont disponibles et on peut, par exemple, utiliser Accès pour se connecter au serveur MySQL.

Des fonctionnalités importantes en matière de manipulation, représentation et stockage des données :

- Les fonctions SQL sont implémentées à travers des classes de bibliothèques extrêmement optimisées.
  - 16 index par tables sont autorisés. Chaque index consiste en 1 et 15 colonnes ou parties de colonnes. La longueur maximale d'un index est de 256 octets.
  - Enregistrements de longueur fixe ou variable et gestion de grandes bases de données. Pouvant atteindre plus de 50,000,000 enregistrements.
- Système flexible et sécurisé de droits et de mots de passe, et qui autorise une vérification faite sur l'hôte. Les mots de passe sont sécurisés depuis que la gestion des mots de passe est crypté entre le client et le serveur. Les clients se connectent au serveur MySQL en utilisant les connexions TCP/IP.

### II.3- Obtention :

Pour obtenir le système MySQL, il suffit de le télécharger librement depuis plusieurs serveurs Web (comme <http://www.mysql.org> ou <http://www.mysql.com>) qui proposent différentes distributions, exception faite pour MySQL pour la plate-forme Microsoft Windows qui nécessitent l'achat de licence d'exploitation du logiciel et n'offre que des versions d'essais. Pour la plupart des distributions des systèmes UNIX et LUNIX, MySQL figure parmi les bibliothèques du système et s'installe directement avec celui-ci. Son installation est principalement basée sur le dépaquetage de certains modules et programmes en mode ligne de commande et la création des répertoires de travail particuliers.

#### II.3.1- installation et Compilation :

Récupérer l'archive mysql-3.22-32.tar.gz([www.mysql.org](http://www.mysql.org))

Cd/usr/src

Tar-vzxf mysql-3.22-32.tar.gz

Cd/usr/src/mysql-3.22.32

/configure

Make

make install

scripts/mysql\_install\_db

### III-Serveur MYSQL:

#### III.1- Connexion au serveur MySQL :

mysql [-h host\_name] [-u user\_name] [-p mot de passe]

Les clients MySQL ont besoin d'un certain nombre de paramètres pour se connecter à un serveur MySQL : l'hôte qui abrite le serveur, le nom d'utilisateur et le mot de passe. Par défaut, mysql utilise les valeurs suivantes :

- le nom d'hôte par défaut est localhost, c'est à dire la machine locale.
- le nom d'utilisateur par défaut est le nom de login Unix.
- aucun mot de passe n'est envoyé si -p n'est pas précisé.

### III.2- Système de droit et contrôle d'accès :

#### III.2.1 But :

La fonction primaire du système de droits de MySQL est d'authentifier un utilisateur se connectant, et l'associer avec les droits d'utilisation des commandes **SELECT**, **INSERT**, **UPDATE** et **DELETE** sur cette base. Les fonctions secondaires incluent la possibilité d'accueillir un utilisateur anonyme, et donner des droits particuliers à des fonctions spécifiques.

#### III.2.2- Noms d'utilisateurs et mots de passe :

Il y a de grandes différences entre la gestion des noms d'utilisateur et mots de passe de MySQL, et celle de Unix ou de Windows. Les noms d'utilisateurs, utilisés par MySQL pour l'authentification, n'ont rien à voir avec les noms d'utilisateur de Unix (Nom de login) ou de Windows. Les noms d'utilisateurs MySQL peuvent avoir jusqu'à 16 caractères de longueur. Généralement, les noms d'utilisateur Unix sont limités à 8 caractères. Les mots de passe MySQL sont cryptés avec un cryptage différent de celui d'Unix.

#### III.2.3- Fonctionnement :

MySQL s'assure que tous les utilisateurs peuvent faire ce qu'ils ont le droit de faire. Lorsqu'on se connecte à un serveur MySQL, le serveur détermine l'identité grâce à l'hôte depuis lequel on se connecte, et le nom d'utilisateur qu'on spécifie. Le système alloue les droits adéquats. MySQL considère que le nom de l'hôte et le nom d'utilisateur sont suffisants pour une identification sans ambiguïté, car il y a peu de chance qu'un nom d'utilisateur soit utilisé par la même personne, depuis tous les hôtes sur Internet! MySQL permet de distinguer les utilisateurs, et de donner des

droits différents pour le même nom d'utilisateur, mais pour des hôtes différents. MySQL contrôle l'accès en deux temps :

- Etape1 : vérification de la connexion : Lorsqu'on se connecte à un serveur MySQL, le serveur accepte ou rejette la tentative en fonction de l'identité, et de la capacité à fournir le mot de passe correct. Dans le cas contraire, le serveur refuse complètement l'accès. Sinon, le serveur accepte la connexion, et passe en niveau2, pour attendre les requêtes. L'identité lors de la connexion est basée sur deux éléments : l'hôte depuis lequel on se connecte et le nom d'utilisateur MySQL. La vérification de l'identité est faite en utilisant les trois champs d'identité de la table user (Host, User et Password). Le serveur n'acceptera une connexion que s'il existe un enregistrement qui contienne l'hôte, le nom d'utilisateur et le mot de passe.

- Etape2 : vérification des requêtes : Une fois la connexion est établie, le serveur passe en niveau2. Pour chaque requête entrante, le serveur va vérifier que les droits sont suffisants pour effectuer la requête, en fonction du type d'opération. Pour les requêtes administratives telles que shutdown, reload,..., le serveur ne vérifie les droits que dans la table user, étant donné que c'est la seule qui spécifie les droits administratifs. La commande est exécutée si les droits sont disponibles, sinon la requête n'est pas autorisée. Pour les requêtes liées aux bases des données, telles que insert, update, ect..., le serveur commence par vérifier les droits globaux (droits de super utilisateur) en recherchant dans la table user. Si il trouve des droits, l'exécution de la requête est autorisé.

### III.3- Lancement de serveur MySQL :

- Lancer une session root sur la machine qui fait serveur.
- Exécuter la commande `/comm/soft/bin/safe_mysqld—log&`. L'option—log permet d'enregistrer les messages d'erreur ainsi que toutes les requêtes qui sont faites au serveur.

le fichier `log/comm2/soft/mysql/var/tecfasun1 .log` est crée automatiquement s'il n'existe pas encore.

Si le serveur est déjà actif, le système répond 'A mysql process exist'.

### III.4- Arrêt du serveur :

- Lancer une session root sur la machine qui fait serveur.
- Exécuter la commande: `/comm/soft/bin/mysqladmin shutdown`

## IV – Programmation Mysql :

Ce paragraphe décrit la procédure de création de la base de données PFE et facturation et l'import des données sur le SGBD MySQL.

### IV.1- Création de la base de données PFE :

```
mysql> create database PFE ;
```

Créer une base de données ne la sélectionne pas automatiquement. Il faut le » faire explicitement. Pour faire de PFE notre base courante, il faut utiliser la commande :

```
mysql> use PFE
Database changed
```

De même pour la base de données facturation.

```
mysql> create database facturation
```

```
mysql>use facturation  
Database changed
```

La base n'a besoin d'être créée qu'une seule fois, mais il faudra la sélectionner à chaque fois que vous commencerez une session mysql. Il suffira alors d'utiliser la même commande que ci-dessus. Alternativement, vous pouvez sélectionner une base dès la connexion, en passant le nom de la base après tous les paramètres de connexion :

```
$mysql -h -u user -p PFE/facturation  
enter password: *****
```

#### IV.2- Création des tables de la base:

Créer une base de données est facile, mais, jusqu' à présent , c'est vide. La commande 'show tables 'donne :

```
mysql>show tables ;  
Empty set (0.00 sec)
```

La partie la plus difficile est le choix de la structure de votre base de données, et des tables dont on a besoin, et quelles colonnes seront nécessaires. A titre d'exemple, on va détailler la création des tables les plus importantes du projet : abonné , client\_physique; en se référant à sa structure.

```
mysql>use facturation  
Database changed  
mysql>create table abonne (  
Dn int(8) unsigned no null default,  
Cin int(8) unsigned not null default,  
Nom varchar(50) not null,  
Prenon varchar (50) not null,  
Compteur_journalier int (16) unsigned not null,  
Primary key (dn,cin)  
);
```

```
mysql> use PFE  
Database changed  
mysql>create table client_physique (  
Nom varchar(50) not null,  
Prenon varchar (50) not null,  
Rue varchar (50) not null,  
Ville varchar (50) not null,
```

```
Numero int (4) unsigned not null,  
Code_postal int (4) unsigned not null,  
Login varchar (50) not null,  
Password varchar (50) not null,  
Primary key (login,password)  
);
```

#### IV.3- Test de la base :

La commande 'show tables' permet de vérifier que la base de données est implémenté sur le serveur et que les tables sont créées.

```
mysql> show tables ;
```

Pour vérifier que la table a été créée comme on le désire, on utilise la commande 'describe'.

```
Mysql>describe abonne ;
```

#### IV.4- Alimentation de la base de données :

Après avoir créé la table, il faut la charger. La fonction LOAD DATA et INSERT remplissent cette fonction. Etant donné qu'on commence avec une table vide, le meilleur moyen de remplir cette table est de créer un fichier texte, chaque ligne contenant les informations d'un client, puis de le charger directement dans la table avec une seule commande. On crée ainsi un fichier

Abonne.data contenant un enregistrement par ligne, avec des valeurs séparées, et dans le même ordre que l'ordre dans lequel les colonnes ont été listées dans la commande 'create table'.

```
mysql>load data local infile "abonne.data" into table abonne;
```

Pour n'ajouter qu'un seul enregistrement à la fois, la fonction INSERT est plus pratique : Dans sa forme la plus simple, on fournit les valeurs dans l'ordre des colonnes.

```
mysql>insert into 'nom de la table ' VALUES('x','y','z','') ;
```

#### Conclusion :

MySQL est un logiciel libre de base de données relationnelle fonctionnant en client-serveur. Il est maintenant très répandu et devient un standard, en particulier dans le cas

d'application W e b. il n'a pas la prétention de visualiser avec Oracle ou les autres gros logiciels commerciaux. MySQL peut être utilisé dans des scripts PERL et PHP ou inclus dans des documents HTML ; ce qui est le plus courant.

## **Chapitre 5**

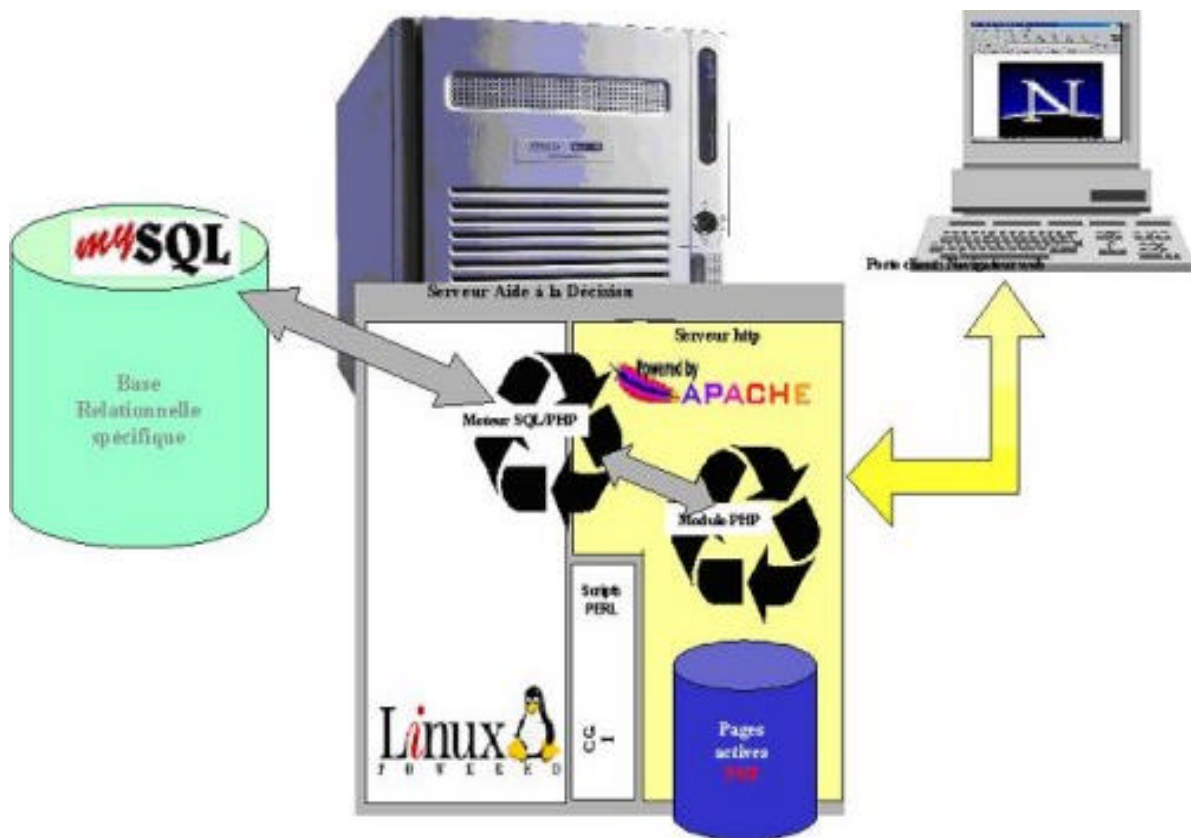
### **Serveur Web et interfaces clients**

---

#### **Introduction :**

La migration des systèmes informatiques vers le client-serveur est aujourd'hui une tendance fondamentale de l'industrie car elle met en jeu trois notions importantes et décisives dans tous projets informatiques orienté communications : la base de données, les réseaux de communication et les outils de développement. L'implémentation Web de l'architecture client-serveur est très simple mais cette simplicité est imposée par les besoins et les exigences en terme de rapidité et de facilité d'accès car c'est un environnement ouvert au grand public. Il est donc primordial d'étudier l'architecture client-serveur Web en détaillant les trois composantes principales : le client et les outils de développement qui sont jointes, le protocole http et le serveur Web. L'objectif final est de réaliser, les différents pages et traitement du côté client, puis l'étude des performances du serveur APACHE qui est utilisé dans ce projet .





**Figure 5.1 : Exemple d'une configuration client-serveur**

## **I – Système client :**

### **I.1 – Performances des applications orientées Web :**

La technologie Web n'est pas la première à permettre de récupérer des informations de la part des utilisateurs via un réseau. C'est exactement ce que font tous les programmes client –serveur et notamment les applications très étendues comme oracle ou lotus Notes. Mais en ajoutant un composant Web à une application client-serveur habituelle, on bénéficie d'avantages supplémentaires :

- Une compatibilité instantanée entre plates-formes : les navigateurs Web ont été conçus pour fonctionner sur pratiquement toutes les plat formes existantes. De plus, les langages HTML et java sont indépendants et non compilés. Ils sont interprétés au fur et à mesure par le navigateur. On économise ainsi une grande quantité de travail par rapport au processus de développement classique.
- Un accès universel : un formulaire Web peut être consulté depuis n'importe quel nœud d'un réseau via un simple lien . Dès qu'on crée un formulaire sur le serveur,

toute personne possédant un navigateur et accédant au réseau peut l'ouvrir. Le formulaire peut ressembler à un document Web normal, et on peut même l'insérer à l'intérieur d'un autre document.

➤ Une interface utilisateur universelle : les formulaires Web permettent de proposer des applications client-serveur via une seule interface utilisateur, celle du navigateur, qui peut être commune à des dizaines d'applications.

Les différentes pages Web du projet de l'inscription de télé facturation téléphonique sont basées sur les formulaires HTML qui seront traités à double reprise premièrement au niveau client pour des contrôles intrinsèques via un langage de scripting client et deuxièmement au niveau du serveur par des modules spécifiques au projet. Le paragraphe suivant étudie les techniques de développement coté client.

## I.2 – Développement Web coté client :

La nature du projet impose que la plupart des pages Web visualisées soient générées dynamiquement depuis le serveur et ce en fonction de la situation encours qui dépend des données à envoyées par le client. On parle alors de Web dynamique dont la technique sera étudiée dans le chapitre qui suit. Ce pendant, la génération à la volée de code HTML impose la construction d'un modèle des différentes pages du scénario de navigation. Le langage de base est HTML dans sa spécification 4.06 enrichie de plusieurs fonctionnalités concernant les formulaires pour plus d'interaction et souplesse.

### I.3- Utilisation de formulaires :

Afin d'exploiter les bases de données, il faut fournir une interface à l'utilisateur lui permettant de visualiser des données en fonction de certains critères. Pour cela il existe un outil: les formulaires.

Un formulaire est une interface présentant des composants permettant d'afficher, de saisir ou sélectionner des données.

De nombreux outils permettent la création de formulaires, c'est notamment le cas du HTML. De nombreux environnements pour créer des formulaires existent aussi pour chaque SGBD (Access, Windev, ...).

Notre choix dans ce projet est d'utiliser les formulaires HTML.

### I.4- Les formulaires HTML :

Les formulaires forment un type particulier de document HTML qui en définit les règles. En effet, ce sont des documents Web ordinaires qui disposent d'emplacements permettant à l'utilisateur d'entrer les informations. Les navigateurs Web doivent savoir comment interpréter les formulaires qui peuvent utiliser toute commande de formatage liée à HTML, telles que les listes ou les tableaux. Le code du formulaire doit contenir essentiellement trois éléments : la méthode d'envoi des données (METHOD), l'adresse

du serveur qui va les recevoir (ACTION) et la fenêtre cible de la réponse du serveur (TARGET).

### ➔ METHOD :

Cet attribut est utilisé pour envoyer les données en spécifiant quelle méthode d'envoi utiliser, On distingue deux méthodes :

- **GET** : elle place les données à la fin de l'URL dans la requête. Les données de l'URL sont séparées par le caractère « ? ». Puisque GET concatène les données et l'URL, le navigateur lui même peut mémoriser la requête comme un signet qui pourra être réutiliser. Cependant, les URL ont une longueur maximale admissible et GET ne convient pas aux requêtes contenant trop de données. En pratique, cette méthode n'est utilisée que dans le cas où l'envoi du formulaire ne change rien sur le site de destination qui est le cas typique des moteurs d'indexation W e b.
- **Post** : elle envoie les données comme un message séparé de l'URL mais qui reste lié à la requête sous forme d'objet HTTP. Cela signifie que le client peut envoyer de l'information au serveur de la même façon que le serveur envoie les données au client. La méthode POST permet d'installer des documents sur le serveur même si cette utilisation est relativement rare. Les formulaires de l'inscription et la consultation de facture sont les meilleurs exemple.

Ces deux méthodes font l'objet des standards HTML, HTTPet CGI. Ils dépendent du type du système utilisé. Les détails sur la manière d'écrire un script pour traiter des données en provenance d'un formulaire sont radicalement différentes entre les systèmes de type Macintosh, Windows NT ou UNIX. Le standard CGI spécifie les différences relatives à chaque type de système d'exploitation. Ces différences ne doivent pas être apparentes pour l'utilisateur, elles ne concernent que les programmeurs des scripts.

### ➔ ACTION :

Cet attribut spécifie l'URL où seront dirigées les données. L'URL indiquée peut être codé sous forme absolue ou relative et peut même spécifier certains paramètres à envoyer au serveur.

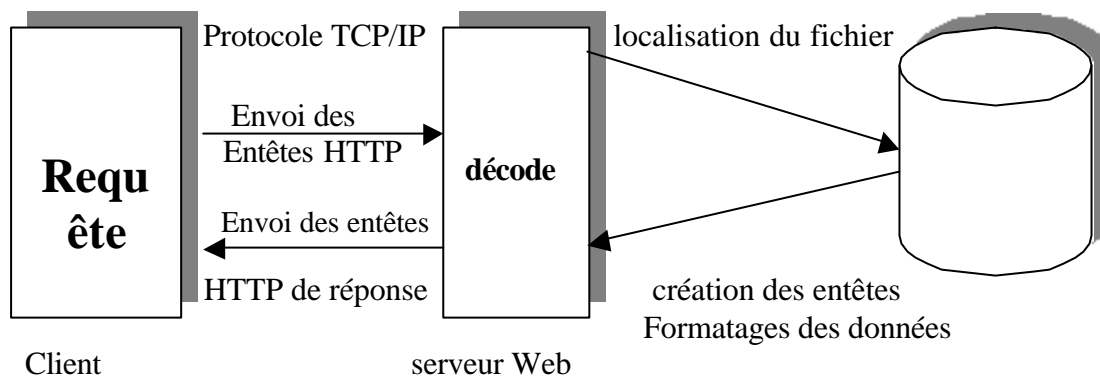
### ➔ TARGET :

Cet attribut n'est pas obligatoire mais entre en jeu après l'envoi des données et au moment de la réception de réponse pour indiquer l'endroit où afficher le document reçu. Il peut prendre l'une des valeurs suivantes :

- **\_SELF** :fenêtre ou frame en cours.
- **\_NEW** : nouvelle fenêtre.
- **\_PARENT** : fenêtre ou frame parent.
- **Target\_Name** : nom explicite du cadre cible.

## II-Le protocole http :

Le serveur et le navigateur Web communiquent entre eux par l'intermédiaire du protocole http qui est simple et conçu pour être utilisé sans le cadre des systèmes hypermédias distribués en réseau. http a été créée spécialement pour le Web et définit un mode simple de conversation selon le modèle requête-réponse. Un programme client établit une connexion avec un programme serveur et lui envoie la requête à laquelle réagit le serveur en émettant une réponse contenant l'information attendue par le client. http n'intervient pas dans la manière dont la connexion réseau est produite et gérée ni dans la manière de transmettre l'information (ceci est à la charge des protocoles de bas niveau comme TCP et IP).



**Figure 5.2 : Représentation du protocole http**

### II.1- Requête http:

Une requête http se compose des éléments suivants :

- La méthode : qui doit faire partie de l'ensemble des actions légales.
- L'identificateur de ressources universelles (URL) : qui est le nom de l'information recherchée.
- La version du protocole.
- Autres informations permettant de modifier ou compléter la requête.

L'idée fondamentale véhiculée par la notion de requête est que la méthode doit être appliquée à l'objet référencé par l'URL. La méthode doit faire partie des choix standards listés dans la table suivante :

Méthode	Action
GET	Récupère l'information.
HEAD	Retourne l'information concernant l'objet.
POST	Demande le stockage d'une information.

PUT	Envoie une nouvelle copie d'un objet existant sur le serveur .
DELETE	Détruit l'objet d'une manière irréversible.
Autre	D'autres méthodes peuvent être définies l'avenir

**Table 5.1 : les méthodes http**

L'URL identifie l'objet recherché. Ils doivent respecter un ensemble des règles définies par un standard international. Le Web utilise un sous-ensemble des URL : les URL qui contient trois éléments : l'identificateur du protocole, nom logique du serveur et le chemin complet du document. La requête peut inclure des informations complémentaires sur la manière de remplir la requête.

CHAMP	INFORMATION
User-Agent	Type du navigateur client
If-Modified-Since	Récupération si l'information est plus récent que la mentionnée.
Accepte	Type de données(MIME) acceptées par le navigateur
Autorisation	Information d'authentification(mot de passe ,...).

**Table 5.2 : information envoyées par le client**

## II.2- Réponse HTTP:

Une réponse http contient les éléments suivants :

- Une ligne d'état : indique le succès ou l'échec de la requête.
- Une description de la réponse : il s'agit d'une information appelée « méta-information ».
- L'information attendue.

L'idée de base est que le serveur répond à la requête en fournissant une description de l'information retournée et qui est suivie de l'information proprement dite. Le format de la ligne d'état est le suivant :

Version http	Code Result	Reason
--------------	-------------	--------

**Figure 5.3 : Format de la ligne d'état d'une réponse http**

Le code résultat est un nombre indiquant le succès ou l'échec de la requête. La raison et une phrase expliquant la signification du code.

Code	message	Explication
200	OK	La requête à été accompli correctement
301	moved permantly	Document migré vers autre URL
401	Unauthorised	Information protégée
404	not found	Information non trouvée
500	server error	Erreur dans le serveur

**Table 5.3 : les codes de réponse HTTP**

Nom de l'entête	description
server	Type de serveur
Date	Date et heure de la réponse
Content_type	Type de contenu du corps de la réponse exp : (text/html)
Content_length	Nombre d'octets de réponse
Content_laguage	Langage d'expression de l'information
Content_encoded	Type de codage du corps de la réponse
Last_modified	Date et heure de la dernière mise à jours

**Table 5.4 : Les entêtes de réponse HTTP**

Le content\_type est codé à partir de type MIME sur la même base que la liste des types Accept du navigateur . le MIME est un standard international définissant les règles d'échange d'informations pouvant contenir autre éléments que de texte. Le type MIME est capitale pour les programmes est surtout pour le navigateur Web, car MIME indique comment déchiffrer et afficher l'information. Du faite que chaque navigateur peut accepter différent formats de données et que chaque serveur dispose de plusieurs type de documents , le content\_type permet au serveur de renseigner le client sur le contenu de la réponse.

Type MIME	explication
Text/plain	Text ASCII pur et sans formatage
Text/html	Document html
Application/octet_stream	Application exécutable
Image/gif	Image au format GIF
Video/mpeg	Clip vidéo au format MPEG

**Table 5.5 : quelques type MIME les plus utilisés**

### **III- Le serveur Web :**

#### **III-1-Principes et fonctionnement :**

Un serveur web est un logiciel permettant à des clients d'accéder à des pages web, c'est-à-dire en réalité des fichiers au format HTML à partir d'un navigateur installé sur leur ordinateur distant.

Un serveur web est donc un "simple" logiciel capable d'interpréter les requêtes HTTPD arrivant sur le port associé au protocole HTTP (par défaut le port 80), et de fournir une réponse avec ce même protocole.

Les principaux serveurs web sur le marché sont entre autres:

- Apache
- Microsoft IIS
- Microsoft PWS
- Xitami

#### **III.2- Performances d'un serveur Web :**

Un système serveur dispose typiquement d'un processeur et d'une mémoire en quantité limitée. L'ensemble doit être partagé entre le système d'exploitation et tous les programmes d'application y compris le programme httpd et les scripts. Tous temps de calcul et toute parcelle de mémoire utilisée par le système d'exploitation ne profite pas au travail effectif consistant à traiter des requêtes Web .

##### **III.2.1- Exécution des scripts et autres programmes :**

Des ressources de calcul supplémentaires sont nécessaire lorsque le serveur autorise l'exécution des scripts qui partagent le processeur et la mémoire avec le programme serveur httpd et le système d'exploitation et prennent généralement plus de temps à s'exécuter qu'une simple requête GET. Les performances des scripts varient sensiblement suivant les systèmes : les DLL sous Windows donnent plus de performances que sous des CGI sous Unix.

##### **III.2.2- Compression et cryptage des données :**

L'utilisation des images à haute résolution peut nécessiter des techniques de compression permettant de réduire le temps de transmission sur le réseau. Les données sont compressées par application d'un algorithme effectuant un nouveau codage supprimant les redondances ce qui donne une version plus compactée des données mais consomme pas mal de temps machine. Le cryptage et décryptage des données sont essentiels pour assurer la confidentialité des données transmises sur le



réseau mais peuvent demander des temps de traitement important dont l'impact sur le serveur Web est significatif .

### **III.2.3- Accès aux données :**

Le serveur, outre ses accès aux processeurs et à la mémoire, doit lire des données sur un dispositif de stockage permanent. L'efficacité du disque dépend essentiellement de deux facteurs : la vitesse du dispositif et l'efficacité du logiciel dans sa gestion du périphérique. Le temps d'accès aux données doit être minime car les navigateurs n'ont pas été conçu pour patienter plusieurs minutes.

## **IV- Choix technique : APACHE**

### **IV.1-Pourquoi APACHE ?**

Apache est le serveur le plus répandu sur Internet, il est très puissant. Il s'agit d'une application fonctionnant à la base sur les systèmes d'exploitation de type UNIX, mais il a désormais été porté sur de nombreux systèmes, dont Microsoft Windows. Dire que la plus part de marché d'apache est de l'ordre important serait un contre-sens dans la mesure où Apache est gratuit et ne vise donc pas à conquérir les autres type de serveur commerciaux . La principale différence entre Apache et les serveurs se résume en quelques points notables :

- Apache se configure par modifications de fichiers textes. Ce type de configuration reste le plus puissant et on peut donc dire qu'Apache est le plus configurable des serveurs HTTP.
- Apache supporte les SSL mais ceux-ci ne sont pas reconnus par le SCSSI, l'organisme de validation des systèmes de sécurisés en France.
- Le couplage Apache Perl dans le monde Windows nécessite de modifier la première ligne des scripts Perl pour les mettre sous la syntaxe :  
# !c:/perl/lib/perl.exe

Mais à part ces quelques limitations, Apache reste la solution la plus compatible entre les mondes Windows et Unix et cela constitue un intérêt non négligeable.

### **IV.2- Installation :**

Aussi bien sous Windows que sous Unix, installer Apache version 1.3.14 ne pose pas de réelle difficulté. Sous Windows la version binaire est livrée avec un installateur identique pour la version NT et la version 95. Les sources sont également disponibles. Sous Unix, on préfère compiler les sources mais cela ne pose non plus de problème.



particuliers. On va donc explorer en détails la partie configuration du serveur. Cette configuration se fait à l'aide de quelques fichiers . mais avant de les examiner, voici l'arborescence du serveur :

- cgi-bin : est le répertoire par défaut contenant les CGI .
- conf : est le répertoire où sont stockés les fichiers de configuration .
- htdocs : est le répertoire par défaut où sont stockés les fichiers destinés à être publiés .
- icons : est le répertoire par défaut contenant toutes les icônes représentant les dossiers et les fichiers textes etc.
- logs : est le répertoire où sont stockés les fichiers mouchards.
- modules : contient un certain nombre de modules optionnels qui peuvent être adjoints au serveur.
- src : quel plaisir de savoir qu'on dispose des sources du serveur dans ce répertoire !

#### **IV.3-Configuration :**

Bien qu'il existe des logiciels pour configurer le serveur Apache , on donne ici les méthodes de configuration du serveur par modification textes contenus dans le répertoire conf. Les fichiers de configuration du serveur sont placés par défaut dans le répertoire /etc/httpd/conf. On dénombre trois fichiers importants dans la configuration .

- access.conf : contient les instructions permettant de gérer les droits d'accès au serveur .
- srm.conf : contient les instructions gérant les noms et les types accessibles aux utilisateurs connectés aux moyens des navigateurs.
- http.conf : contient des directives propres au fonctionnement du serveur lui-même.

#### **IV.4- Démarrage du serveur Apache :**

Une fois le serveur est prêt, l'étape suivante est de le démarrer réellement d'une manière qui dépend de plate-forme et du mode d'installation. Puisque notre plate-forme est sous UNIX la procédure de démarrage du serveur Apache est la suivante : Après le lancement de la commande Install, le démarrage du serveur se fait par la commande suivante :

```
/usr/local/web/apache/bin/apachectl start  
/usr/local/web/apache/bin/apachectl start: httpd started
```

## V- Sécurité du serveur APACHE :

Le principe est de sécuriser les protocoles Internet (du type HTTP) pour lesquels les données passent "en clair" en intercalant entre eux et la couche TCP une couche remplissant cette fonction avec les propriétés suivantes :

- **Transparence**, après l'établissement de la connexion, pour les 2 extrémités
- **Authentification** du serveur ou du client par certificat.
- **Handshake** incluant la négociation des algorithmes de chiffrement les mieux adaptés au client et au serveur

SSL utilise pour le chiffrement des messages une clé de session symétrique :

- transmise par mécanisme cryptographique à clé publique.
- éventuellement réutilisable d'une session à une autre pour diminuer le temps d'établissement de la connexion.

La couche SSL peut notamment s'intercaler entre TCP et HTTP : HTTPS implémenter dans le module apache mod\_ssl .

Le protocole HTTPS utilise le port 443 par défaut, mais le changement est possible par une simple configuration.

## Conclusion :

L'importance de cette étape est capitale car elle a permis d'une part l'étude du mode de communication en client-serveur Web et dégager les éléments qui nous intéressent pour ce projet dans la partie programmation de scripts CGI et d'autre part de préparer l'ensemble de la plate-forme matérielle et logicielle sur laquelle sera implémenté le code source de l'application dont il sera question dans le chapitre qui suit.

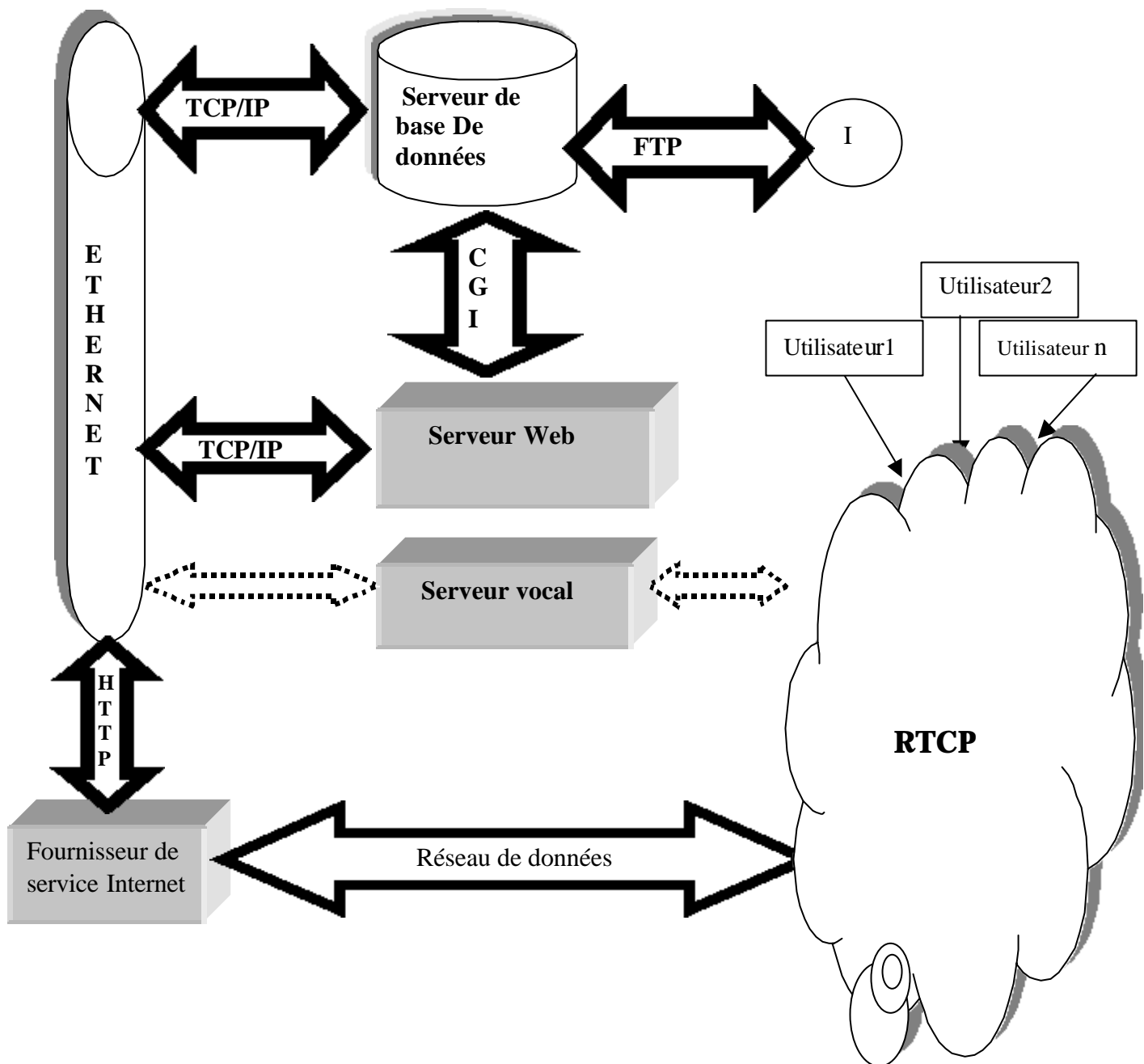
## **Chapitre 6**

### **Interface Web – base de données et organigrammes**

---

#### Introduction :

Poussés par le marché, les éditeurs de SGBDR ont développés des passerelles de plus en plus sophistiquées entre le Web et les bases de données. Les nouvelles architectures pour le data-Web, c'est à dire le couplage du Web et les bases de données, nécessitent d'étendre HTML pour saisir les données et requêtes, rendre dynamique les serveurs Web et augmenter la bande passante entre le SGBDR et l'application. Ces architectures s'articulent sur trois niveaux désormais classiques : le premier gère l'interface utilisateur du navigateur, le second héberge le serveur Web et le complète par le serveur d'application, et le troisième assure la gestion des données au sein du SGBDR. De même, ce projet intègre la notion du data-Web et se base sur des passerelles qui permettent l'accès aux données et la communication avec d'autres serveurs.



**Figure 6.1 : Architecture de la solution proposée**

## **I- Choix techniques : CGI**

### **I.1- Définition :**

CGI est une norme définissant l'interfaçage d'applications externes avec des serveurs d'information. Le programme étant initialement un fichier de commande Shell ou Perl. Rapidement, il est devenu un programme quelconque écrit dans un langage compilé ou interprété. En simplifiant, un script CGI est tout simplement un programme pouvant être exécuté par un serveur http.

### **I.2- Spécifications :**

Les spécifications CGI indiquent quelles informations doivent transiter entre le serveur et le programme CGI et de quelle façon elles doivent le faire. Les trois types de données sont :

- Des données d'administration concernant la requête passée au programme.
- Des données de formulaires en provenance du navigateur.
- Un nouveau document HTML résultat du programme CGI passé au serveur Web pour être renvoyé au client.

Comme les programmes CGI sont indépendants les uns des autres, ils peuvent être écrits dans n'importe quel langage et le contenu de tel programme n'est jamais révélé au navigateur.

### **I.3-Installation :**

Durant la configuration du serveur Apache, on doit spécifier le chemin du répertoire contenant les scripts CGI ainsi que les extensions qui leur sont associées (cgi.pl ...). Celui-ci est généralement appelé cgi-bin/. Une fois le script est codé, on doit le rendre exécutable et autoriser son exécution.

### **1.4- Normalisation des CGI :**

La norme CGI spécifie une interface de communication basée sur des variables dites variables d'environnements qui retournent des informations diverses et un mode d'entrée-sortie basé sur des flots standard de lecture et écriture.

### 1.4.1- variables relatives à la requête :

Variables	Signification
<b>CONTENT-LENGTH</b>	Taille en octet des informations jointes à la requête
<b>CONTENT-TYPE</b>	Type de mime des données envoyées.
<b>QUERY-STRING</b>	Chaîne de caractères contenant les paramètres joints à la requête En mode GET .Elle est vide en utilisant POST
<b>REQUEST-METHOD</b>	<b>Contient la méthode utilisée</b>

**Table 6.1 : variables CGI relatives à la requête**

### 1.4.2- Variables relatives à la connexion :

Variable	Signification
HTTP-accept	Types MIMES supportés par le client
HTTP-accept-langage	Langage utilisé par le client
HTTP-accept-encoding	Type d'encodage supporté par le client
HTTP-accept-charset	Table de caractères supportés par le client
HTTP-accept-cookie	Liste des cookies associées par le client
HTTP-user-agent	Information sur le client
HTTP- reffer	URL de la ressources ayant envoyé la requête
REMOTE-ADR	Adresse IP du client
REMOTE - HOST	Adresse DNS du client
REMOTE-USER	Identifiant de l'ordinateur client
AUTH-TYPE	Protocole utilisé pour valider l'identité
REMOTE-PORT	Port http utilisé par le client

**Table 6.2 : variables CGI relatives à la connexion**

### 1.4.3- Variables relatives au serveur :

Variables	Signification
DOCUMENT-ROOT	Nom du répertoire contenant la racine du serveur
GATEWAY-INTERFACE	Version CGI supportée par le serveur
HTTP-HOST	Adresse IP du serveur
SERVER-ADMIN	Adresse e-mail de l'administrateur du serveur
SCRIPT-NAME	URL du chemin d'accès au script CGI

SERVER-ROOT	Port sur le quel le serveur a reçu la requête
SERVER-PROTOCOL	Version du protocole http utilisée par le serveur
SERVER-SOFTWARE	Nom et version du logiciel serveur utilisé

**Table 6.3 : variables CGI relatives au serveur**

#### **1.4.4- Entrées- sorties :**

La norme CGI définit les règles d'entrée-sortie entre le programme et le serveur Web. CGI reconnaît trois entrée- sortie standards :

- **STDOUT** : sortie standard. Elle est utilisée par l'application pour fournir au serveur Web les informations constituant la page HTML.
- **STDIN** : Entrée standard. Elle est utilisée pour recevoir les données en provenance du client.
- **STDERR** : C'est la sortie qui reçoit les messages d'erreurs.

## **II- Le langage Perl :**

### **II.1- Pourquoi Perl ?**

Perl est optimisé pour extraire des informations de fichiers texte et générer des rapports basés sur ces informations. C'est aussi un bon langage pour de nombreuses tâches d'administration système. Il est écrit dans le but d'être pratique. Perl combine les meilleures fonctionnalités de C, sed, awk, sh et utilise des techniques sophistiquées de recherche de motif pour pouvoir traiter très rapidement de grandes quantités de données. Nous avons choisis ce langage version 5.6.0 pour ses qualités et performances dont nous citons essentiellement :

- Portabilité : Perl existe sur la plus part des plate-forme .
- Economie : il est disponible gratuitement sur Internet et dans ses versions les plus récentes.
- Simplicité : quelques commandes en Perl est l'équivalent d'un programme de 500 lignes en code C .
- Robustesse : pas d'allocation mémoire à manipuler, chaînes, piles, nom de variables illimité,...
- Extensibilité : Perl peut être intégré simplement dans des applications C ou C++ et peut indifféremment appeler ou être appelé par des routines à travers une interface documenté.

## II.2- Perl et CGI :

**Perl est devenu le langage numéro un pour l'écriture des scripts CGI. Il permet de générer du code HTML suite à des traitements et calculs sophistiqués. Un script CGI en Perl est référencé depuis le client soit dans l'attribut ACTION d'un formulaire, soit dans l'attribut HREF d'un lien ou par java script via l'objet location. La structure**

d'un CGI en Perl dépend de la nature de l'application mais il y a des points qui doivent être respectés :

- Indiquer le chemin de l'interpréteur Perl : # !/usr/bin/perl
- Générer un entête qui indique au client que le document envoyé par le serveur est un document HTML : print "content-type : text/html\n\n" ;
- Utiliser des modules et des bibliothèques spécifiques aux traitement des formulaires comme le module CGI : use CGI ; ou la bibliothèque cgi-lib : require "cgi-lib.pl" ;

## II.3- Accès aux bases de données :

Un des aspects les plus intéressants de Perl est de permettre l'intégration de requêtes SQL dans un script CGI. Depuis sa version 5, on accède de la même manière à une base de données quelque soit le système choisi. Et ce en utilisant les modules DBD et DBI et en spécifiant les caractéristiques de la connexion.

### # initialisation de la base de données

```
use CGI ; # utilisation de module d'interface script-web
use DBI ; # chargement d'un module de pilote de base de données
$pfe = new CGI ; # création et initialisation d'une variable CGI
$dn = $pfe->param('dn') ; # récupérer la valeur de dn ( exemple)
$dbh = DBI connect('DBI: MYSQL : PFE ','nom d'utilisateur','mot de passe','hôte') #
connexion à la base de données nommée PFE.
$sql = " insert into client_moral vlues (...)" # requête sql
$dbh-> disconnect ; # déconnexion de la base de données.
```

## III- Aspect sécurité :

### III.1- Aperçu général :

Proposer des formulaires et des scripts CGI est le plus grand risque en matière de sécurité pour un service Web. Etant donné qu'un script CGI est exécutable, son utilisation correspond à laisser n'importe qui exécuter un programme sur notre ordinateur. Même s'il est exécuté avec des droits limités il est possible d'avoir accès à



des fichiers sensibles comme le célèbre `/etc/passwd` contenant les mots de passe cryptés des utilisateurs .

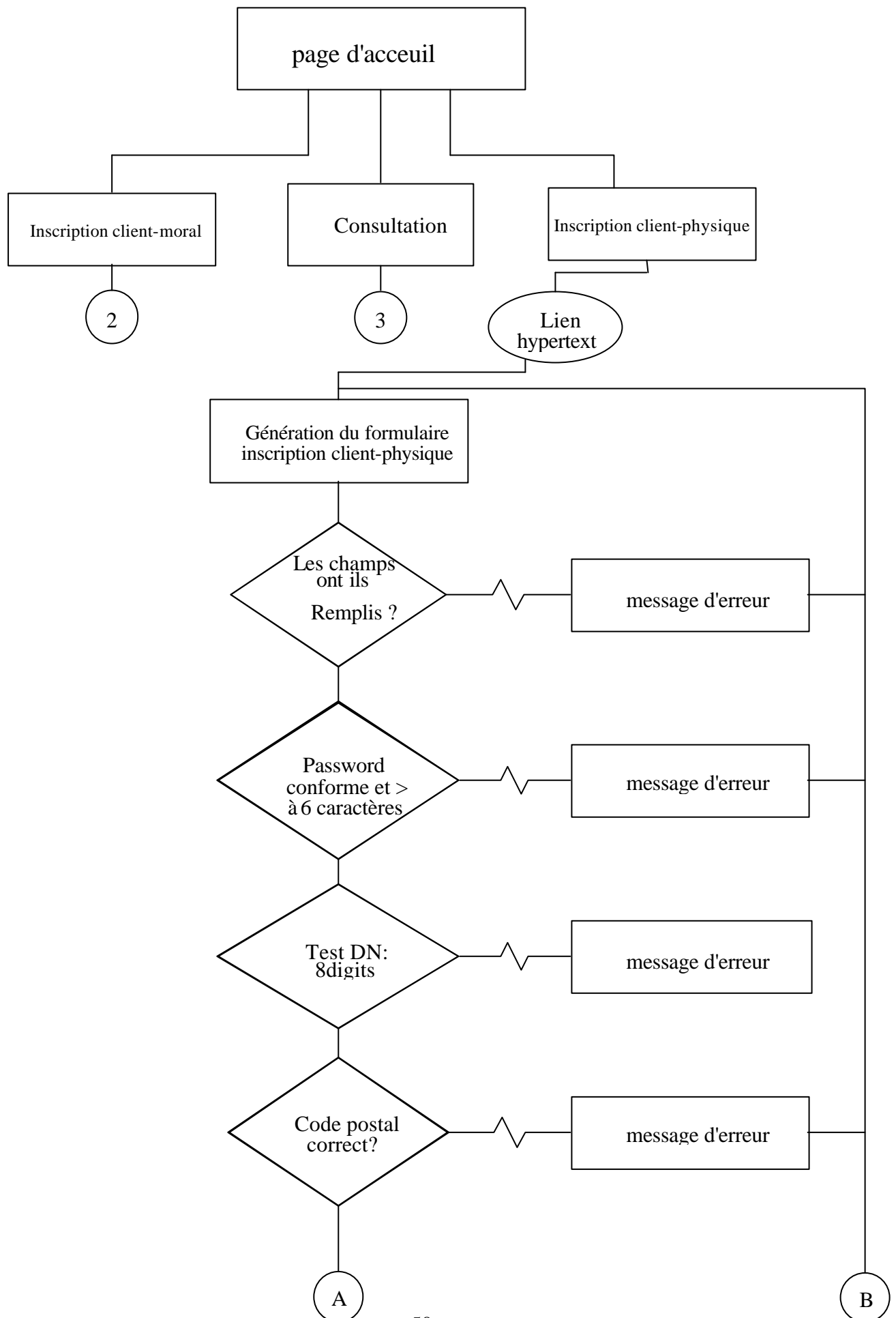
### **III.2- Comment se protéger?**

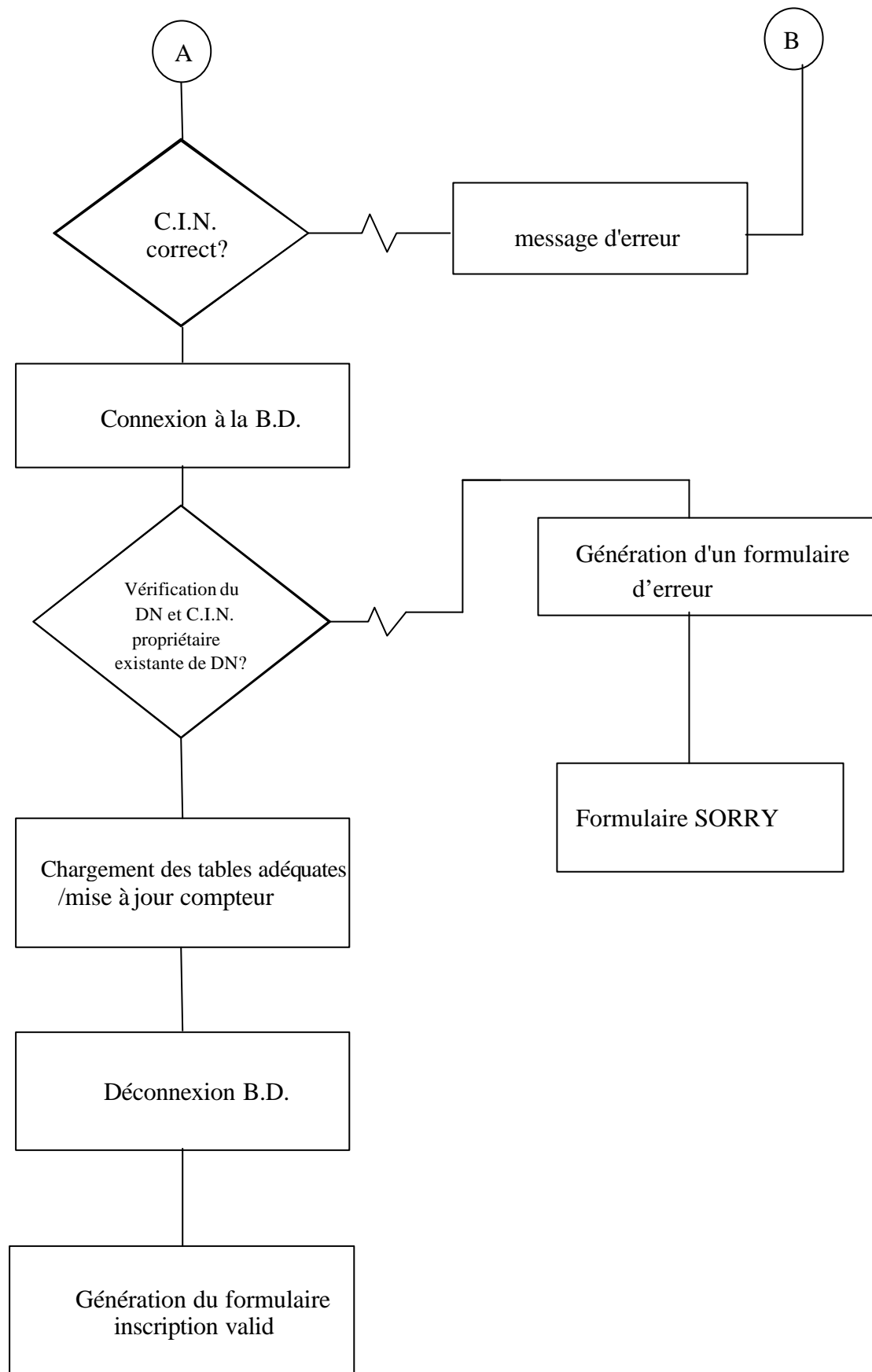
Il existe trois principes généraux en matière de prévention guidant l'usage des scripts :

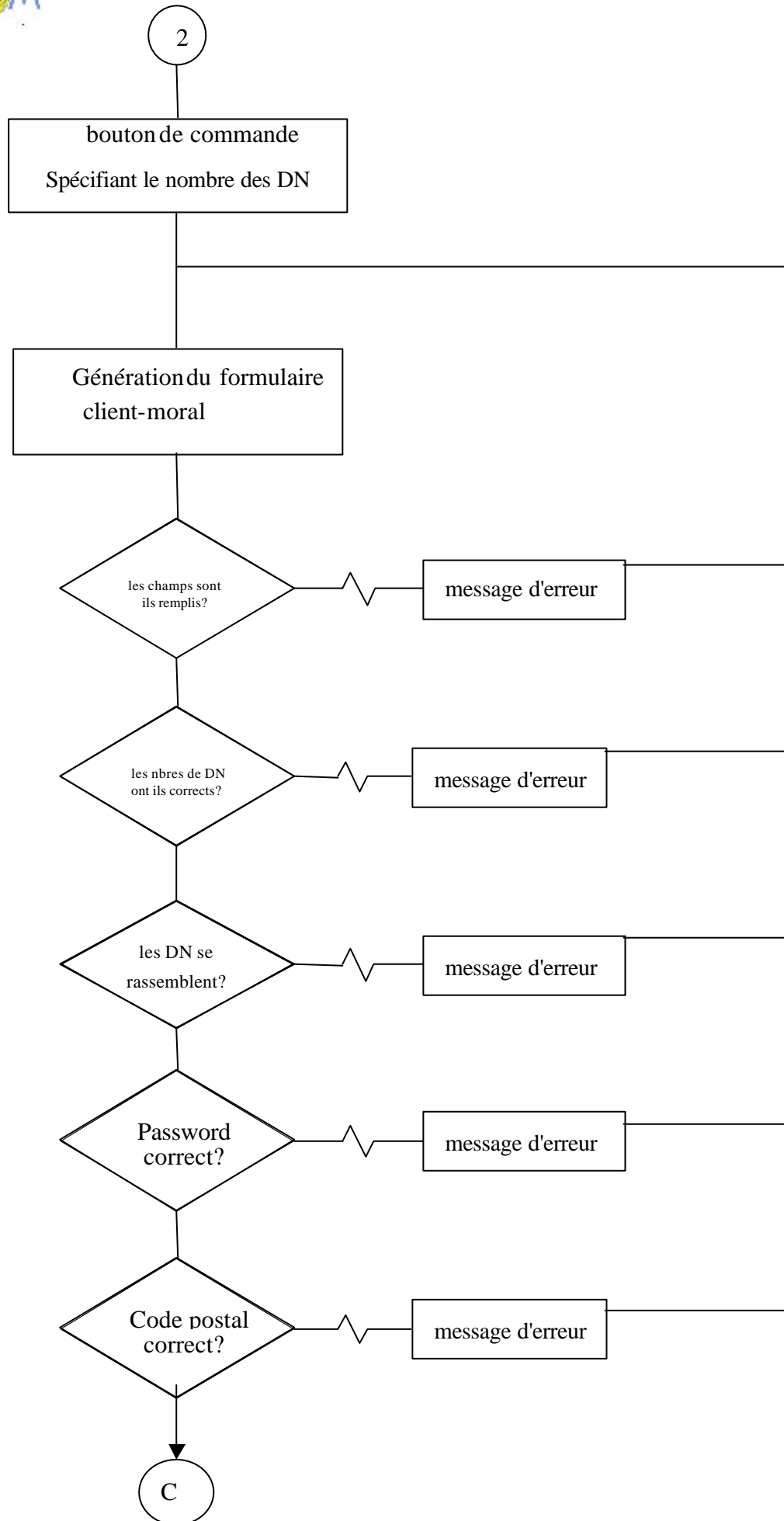
- Les données entrées de l'utilisateur doivent être analysées et vérifiées. Un script doit être soigneusement écrit de manière à vérifier les données reçus, à s'assurer qu'elle exécutent des actions légales et non pas des instructions qui peuvent planter la serveur .
- Les scripts doivent être limités au strict nécessaire en matière de puissance. Dans le cadre des serveurs Multi-Utilisateurs, les scripts doivent s'exécuter avec le minimum de privilèges système et jamais avec les privilèges du super user.

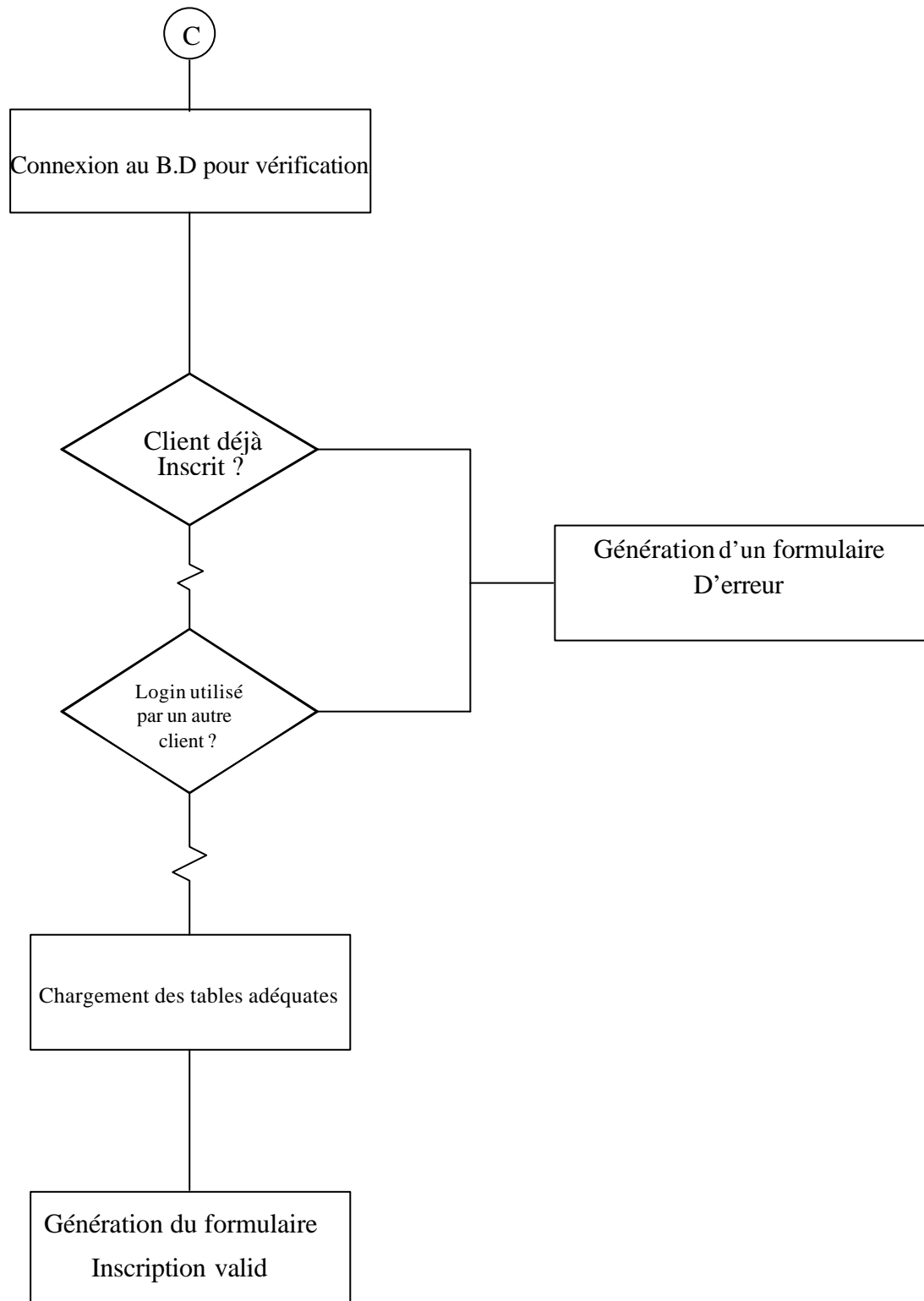
### **IV- Organigrammes :**

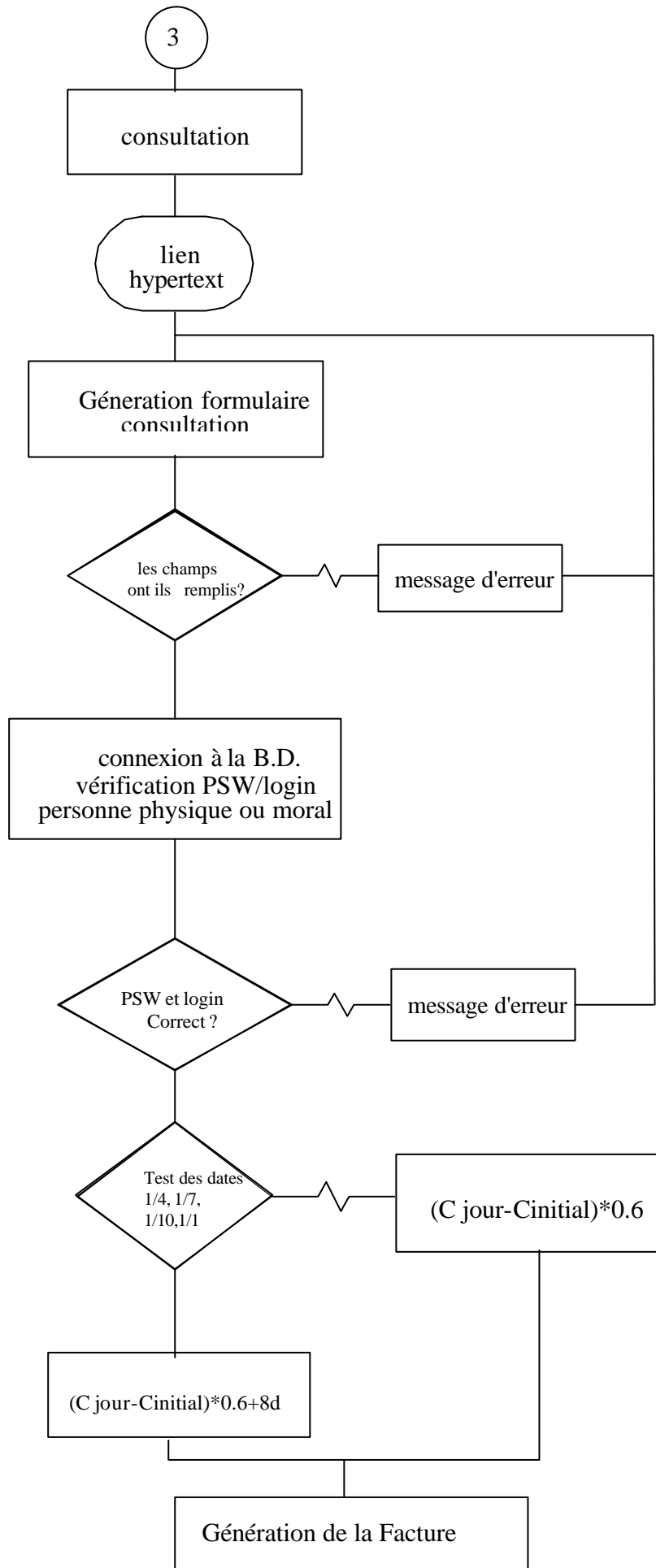
Ce langage nous a permis de développer les programmes dont les organigrammes sont les suivants :











## **Conclusion :**

La mise en place des différents modules CGI du projet nous ont permis de maîtriser un bon nombre de techniques et outils de développement Internet.

Les notions présentées jusqu'ici nous permettent de passer à la partie applicative ( une démonstration réelle ) de notre projet de fin d'études, ce qui fera l'objet du chapitre suivant.

## **Chapitre 7**

### **Installation et mise en service de l'interface X 25 à la maquette EWSD de l'ISSET'Com**

---

#### **INTRODUCTION :**

La mise au point de la maquette EWSD de l'ISSET'Com a pour but de démontrer un cas réel du réseau tunisien , pour cela on a appliqué la procédure nécessaire Afin d'assurer le transfert automatique des fichiers de taxation vers notre serveur de base de données .

Pour connecter un terminal d'exploitation et de maintenance à l'EWSD via une liaison de données X.25, deux possibilités sont offertes. Soit à travers des connexions directes (point à point), soit via un réseau publique de communication de données X.25. Dans notre cas , on se réfère aux connexions directes comme type de réseau X25LC. Les processeurs distants comme un terminal sont identifiés au moyen des adresses OSI et des informations DTE (équipement de terminal de données) administrées dans la base de données du processeur de coordination du centre de commutation EWSD.



## I.1- Présentation du réseau OSI :

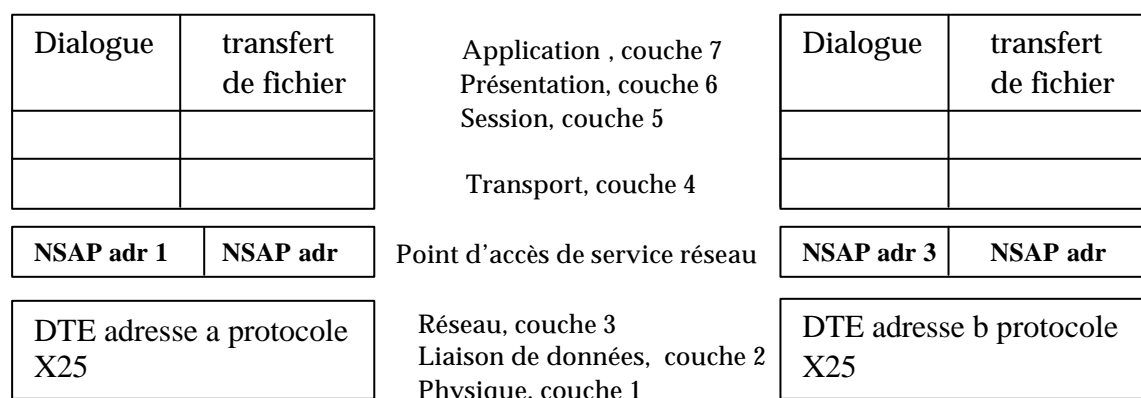
Un réseau OSI permet l'établissement d'un circuit virtuel entre deux DTE (processeurs) selon le modèle de référence de couche ISO 7. A l'aide de l'adresse DTE et de l'adresse NSAP (Point d'accès de service de réseau), tous les processeurs ainsi que les applications appropriées sont identifiés de façon univoque. Un lien de communication entre des applications locales et distantes peut être fourni, soit avec des PVC (Circuits virtuels permanents), soit avec des SVC (Circuits virtuels commutés).

## II- Adresses d'identification du terminal :

Trois adresses sont nécessaires pour identifier un terminal raccordé au processeur de coordination de EWSD via une connexion de données X25 :

### II.1-Adresse X.25 :

Le protocole X.25 fournit deux champs, le champ d'adresse (AF) et le champ d'extension d'adresse (AEF) pour la transmission d'adresses DTE et NSAP. Le champ d'adresse (AF) est utilisé pour l'identification du terminal de réseau dans le réseau X.25 et contient l'adresse DTE évaluée en couche 3. Le champ d'extension d'adresse (AEF) contient l'adresse NSAP, avec laquelle il est possible d'identifier une application particulière. Pour l'exploitation et maintenance d'EWSD via X.25, il faut s'adresser à OSI via AEF. Définition avec commande **CR X25DTE**, paramètre **FACIL=AEFIN&AEFOUT**. La fig. suivante indique le procédé d'adresse tel qu'il sera applicable pour un réseau d'exploitation et de maintenance via X.25 standard.



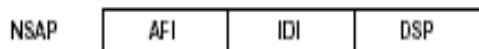
**Figure 7.1 : Procédé d'adresse pour un réseau d'exploitation et de maintenance via X25**

## II.2- Adresse DTE :

Une adresse **DTE** est assignée à une terminaison X.25 particulière et définit le port "physique" dans un réseau X.25. Selon les recommandations CCITT X.121, 14 chiffres décimaux **DTE** peuvent être utilisés comme adresse pour les réseaux X.25. Pour les connexions directes (**X25LC**) 15 chiffres décimaux constituent un maximum pour l'adresse **DTE**. Les adresses distantes **DTE** dans les types de réseaux **X25** et **ISDN** sont créées avec **CR X25DTE** tandis que toutes les adresses locales **DTE** et distantes **DTE** dans le type de réseau **X25LC** doivent être créées avec la commande MML **CR X25ROUTE**. Les valeurs de l'adresse **DTE** peuvent être choisies librement, du moment qu'elles n'empiètent pas sur des entrées de base de données déjà existantes. Cependant, si un X.25 DCN (PSPDN) public est utilisé, les adresses **DTE** devront être commandées depuis la gestion PSPDN.

## II.3- Adresse NSAP :

Au moyen de l'adresse NSAP, fournie dans le champ d'extension d'adresse du protocole X.25, une application spécifique est sélectionnée. Le point d'accès du service réseau (NSAP) définit le seuil entre la couche réseau (couche 3) et la couche transport (couche 4) dans le modèle de référence OSI. L'adresse NSAP identifie un certain point d'accès du service réseau avec son application associée et se compose de 3 champs.



L'adresse DTE est assignée de façon univoque à la partie IDI d'une adresse OSI (NSAP) dans un réseau X.25 par là même, toutes les adresses avec une certaine valeur IDI sont assignées au même DTE et doivent être attribuées au même et unique processeur. Cependant, le NSAP pour connexions directes ne contient pas de valeur IDI (IDI=0). Le type de connexion X.25 au CP détermine la valeur AFI à utiliser pour les adresses NSAP. Certaines applications (dialogue d'impression, transfert de fichier) peuvent être sélectionnées avec la partie DSP. Le tableau suivant indique les différences entre les types de réseau **X25**, **X25LC** et **ISDN** avec les formats et valeurs possibles AFI = Authority Format Identifier

Type de réseau	AFI	IDI chiffres décimaux	DSP chiffres décimaux
<b>X25</b> (DCN X.25)	36	max. 14	max. 24
<b>X25LC</b> (connexion directe)	48	1 (=0)	max. 38
<b>ISDN</b>	44	max. 15	max. 23

**Figure 7.2 : types de réseau et valeurs AFI, IDI et DSP associées**

IDI = Initial Domain Identifier, DSP = Domain specific

### III- Connexion directe (Type de réseau X25LC) :

Cette configuration représente un réseau non ouvert avec une seule connexion point à point et également définie comme un réseau local (X25LC).

Le schéma ci-dessous présente les différents cas de figures adaptés pour une connexion directe au système EWSD .

Dans notre projet on a choisit la première solution , c'est la plus simple à réaliser, et vue l'indisponibilité du matériels pour les autres .

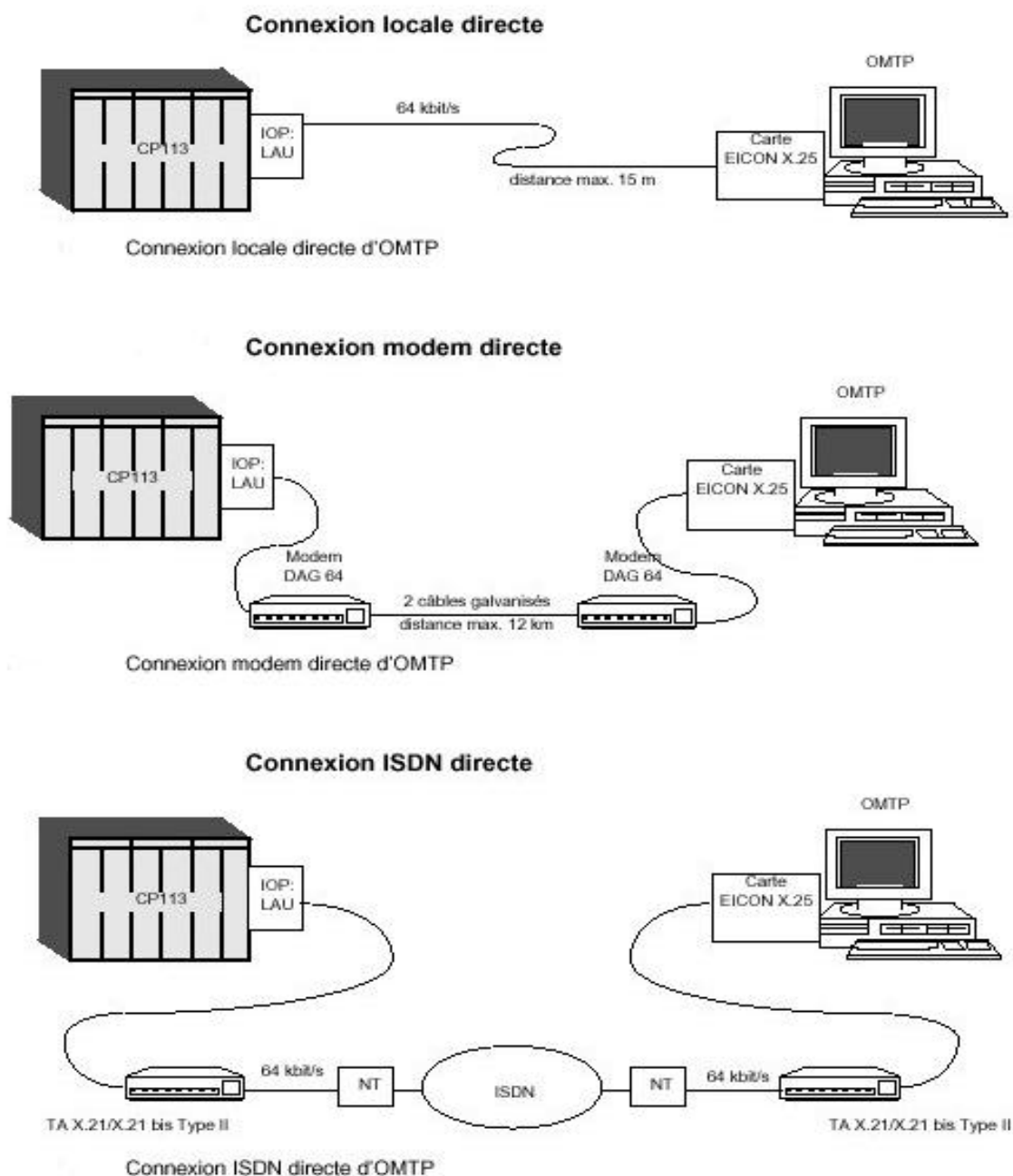
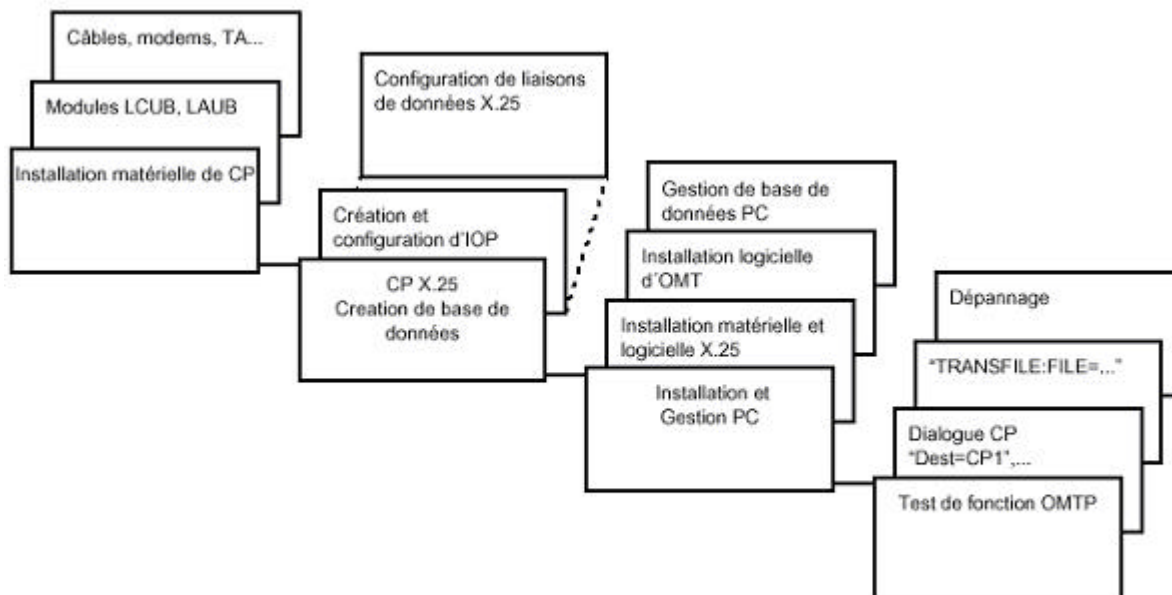


Figure 7.3 : Différentes possibilités de connexion de l'interface X 25

#### IV- Installation et procédures de gestion :

Pour installer un réseau d'exploitation et de maintenance X.25, il est recommandé de suivre les étapes de base suivantes. Si la base de données du processeur de coordination d'EWSD est créée en premier, toutes les données nécessaires pour l'installation logicielle du terminal seront déterminées. Le DTE du terminal et les adresses NSAP ainsi que le nom du processeur tel qu'il est entré dans la table du processeur de coordination sont demandés lors de l'installation logicielle du terminal d'exploitation et de maintenance.

1. Installer IOP:LAU (cartes LCUB et LAUB) dans la baie CP (B:IOC) incluant le câblage.
2. Créer les entrées de la base de données de processeur de coordination pour X.25 (IOP, liaisons, applications, adresses, etc.).
3. Installer le matériel requis (carte EICON X.25) et les ensembles logiciels (pilote X.25, BCTcommun, openft), sur le terminal approprié.



**Figure 7.4 : Séquence de gestion et d'installation X 25**

#### IV.1- Câble IOP pour connexion directe (X25LC) :

Le système EWSD offre plusieurs possibilités de réaliser une liaison de données point à point :

- Câble pour connexion directe locale IOP:LAU <--> terminal qui doit être équipé d'une carte réseau X25 EICON.
- Connexion directe via modem (DAG 64) nécessitant un câble type :  
IOP:LAU <--> modem DAG 64 ( interface X.21)
- Connexion directe via ISDN nécessitant un câble type :  
IOP:LAU <--> Adaptateur terminal X.21/X.21bis (interface V.36):

#### IV.2- Base de données pour la création d'un lien X25 :

La procédure complète de la création d'une liaison de données X25 dans un centre de commutation EWSD type CP113 est comme suit :

##### ➤ Définition des processeurs d'entrées sorties (IOPLAU) :

La position du LAUB est 273 dans les F:IOP-0 et -1  
la position du LCUB est 285 dans les F:IOP-0 et -1  
<CR IOP: IOC=0,BIOC=15,IOP=IOPLAU-0,BIOCR=15,IOPR=IOPLAU-1,LOADTP=100;

##### ➤ Configuration de IOP:LAU :

```
<CONF IOP: IOP=IOPLAU-0,OST=MBL;
<CONF IOP: IOP=IOPLAU-1,OST=MBL;
<CONF LAU: LAU=0,OST=MBL;
<CONF LAU: LAU=1,OST=MBL;
```

##### ➤ Création de la liaison de données X 25 :

```
<CR X25LINK:X25LINK=2,LAU=0,CHAN=0,PRTCL=X25,NET=ISDN,
#L1DTT=DTE,L1IF=X21,L1MCA=1,L2DTT=DTE,L2LAM=RESPON,
#L1BAUD=B64000,
#L2N1=135,L2N2=7,L2K=7,L2T1=3000,L2T2=300,L2T3=20000,
#L3DTT=DTE,L3R20=1,L3R22=1,L3R23=1,L3T20=180,L3T21=200,
#L3T22=180,L3T23=180,L3T24=60,L3T25=150,
#L3TCB=1&&32;
<CONF X25LINK:X25LINK=2,OST=MBL;
<CONF IOP:IOP=IOPLAU-0,OST=ACT;
<CONF LAU:LAU=0,OST=ACT;
```

```
<CONFIOP:IOP=IOPLAU-1,OST=ACT;
<CONFLAU:LAU=1,OST=ACT;
<CROSIADR:ADRNAM=OSILDG,NSADR=48-0-101,NET=X25LC;
<CROSIADR:ADRNAM=OSILFT,NSADR=48-0-109,NET=X25LC;
<CROSIADR:ADRNAM=OSIRDG,NSADR=48-0-1010,LOCADR=OSILDG,
NET=X25LC;
<CROSIADR:ADRNAM=OSIRFT,NSADR=48-0-1090,LOCADR=OSILFT,
NET=X25LC;
<CRAPPL:APPLID=DIALG,PRONAM="KHER2",ADRNAM=OSILDG;
<CRAPPL:APPLID=NEABD,PRONAM="KHER2",ADRNAM=OSILFT;
<CRAPPL:APPLID=DIALG,PRONAM="BCT",ADRNAM=OSIRDG,AUT=1;
<CRAPPL:APPLID=NEABD,PRONAM="BCT",ADRNAM=OSIRFT;
<CRX25DTE:DTENAM=DTELBCT,ADRNAM=OSILDG&OSILFT,NET=X25LC;
<CRX25DTE:FACIL=AEFIN&AEFOUT,DTENAM=DTERBCT,ADRNAM=OSIRDG&
OSIRFT,NET=X25LC;
<CRX25ROUTE:ROUTNAM=RTLBCT,DTENAM=DTELBCT,DTEADR=111,
X25LINK=2,FACIL=FLOWNEG,WNDSZ=7-7,PCKSZ=128-128;
<CRX25ROUTE:ROUTNAM=RTRBCT,DTENAM=DTERBCT,DTEADR=11000,
X25LINK=2;
<CRPRO:PRONAM="BCT",PROTYP=OSS;
```

### ➤ **Activation de IOP:LAU et de la liaison X 25 :**

```
<CONF IOP:IOP=IOPLAU-0,OST=ACT;
<CONF IOP:IOP=IOPLAU-1,OST=ACT;
<CONF LAU:LAU=0,OST=ACT;
<CONF LAU:LAU=1,OST=ACT;
<CONF X25LINK:X25LINK=2,OST=MBL;
<CONF X25LINK:X25LINK=2,OST=ACT;
```

## **V- Protocole FTP :**

### **V.1- Présentation :**

FTP est la protocole qui définit les transferts de données sur un réseau. On l'utilise en général à partir de logiciels aux noms évocateurs, qui s'appellent directement FTP, ou bien qui contiennent FTP dans leurs noms ( exemple openft ) . Les objectifs de ce protocole sont de permettre un partage de fichiers ou programmes sur des machines distantes, de permettre des modifications à distance sur des fichiers, et de transférer des données via un réseau. Bien qu'utilisable par un utilisateur directement depuis un terminal, FTP est pratiquement utilisé par l'intermédiaire de programmes. Ce protocole a été mis en place dès 1971.

## **V.2- Les commandes FTP :**

Il existe plusieurs types de commandes qui peuvent être envoyées sur la connexion de contrôle. En voici quelques échantillons :

### **V.2.1- Contrôle d'accès :**

USER NAME (user): L'argument est une chaîne Telnet identifiant l'utilisateur. C'est cette identification qui est requise par le serveur pour pouvoir accéder à son système de fichiers.

PASSWORD (PASS) : L'argument est une chaîne Telnet spécifiant le mot de passe de l'utilisateur. Cette commande doit être précédée par immédiatement par USER NAME.

REINITIALIZE (REIN) : Cette commande achève la commande USER, annulant toutes les informations d'entrées-sorties, excepté les transferts en cours qui sont autorisés à s'achever.

### **V.2.2- Paramètres de transfert :**

DATA PORT (PORT) : L'argument est un numéro de port machine pour pouvoir établir la connexion pour les données.

FILE STRUCTURE (STRU) : L'argument est un simple caractère Telnet spécifiant la structure du fichier ( exemple : F (file), R (record structure)).

TRANSFERT MODE (MODE) : L'argument est un simple caractère Telnet spécifiant le mode de transfert des données exemple : B (block), C (compressed).

### **V.2.3- Services FTP :**

RESTART (REST) : L'argument représente le marqueur auquel le transfert doit être repris.

RENAME FROM (RNFR) : Pour renommer un fichier.

ABORT (ABOR) : Indique au serveur qu'il doit abandonner la commande FTP précédente.

## **V.3- Les réponses FTP :**

Les réponses sont là pour s'assurer de la synchronisation entre le client et le serveur. Toute commande génère au moins une réponse, et dans le cas où il y en a plusieurs, elles doivent pouvoir se distinguer sans ambiguïté. Les réponses sont composées

d'un numéro à trois chiffres, utile pour les programmes , et d'un texte, pour l'utilisateur humain.

Voici la signification de quelques codes renvoyés par le serveur :

1yz : réponse préliminaire positive

2yz : réponse positive de réalisation

3yz : réponse intermédiaire positive

### **Conclusion :**

Dans cette partie on a eu l'occasion d'approfondir nos connaissances au système de commutation électronique EWSD, et d'amélioré notre expérience professionnelle pour mettre la maquette au point afin de démontrer un cas réel.

Cette étape étant crucial dans le développement de notre projet de fin d'études, car c'est dans cette phase que toutes nos connaissances ont été conjuguées afin d'aboutir à une réalisation cohérente •



## **Conclusion générale**

---

Dans ce projet, il nous a été confié à l'étude , la conception et la réalisation d'une plate forme de télé facturation téléphonique sur Web, permettant à un abonné non prépayé de consulter sa facture d'une façon journalière.

Nous avons procéder de manière à concevoir une base de données, à développer les modules de traitement qui permettent l'accès à la base dès le serveur web, ainsi d'assurer le transfert automatique des fichiers de taxation depuis la maquette EWSD vers notre serveur de base de données, il nous a été possible d'établir une configuration répondant aux exigences du cahier des charges.




Le travail que nous avons effectué tout au long de l'élaboration de ce projet de fin d'étude, nous à été bénéfique à renforcé nos connaissances dans ce domaine, nous pouvons dire que ce projet à été un complément de formation fondamental pour regrouper plusieurs disciplines et consolider plusieurs données théoriques.

En perspectives nous pourrons rendre notre plate forme de télé facturation téléphonique via Internet extensible vers la voie vocale sur RTCP, par développement du serveur adéquat, ou aller plus loin vers la voix sur IP, ce que pourrait faire l'objet d'un travail d'étude pour d'autres étudiants.









## Bibliographie

---

### Les ouvrages

-  George Gardarin & Paul Valduriez **Bases de données relationnelles : analyse et conception** Eyrolles 1988
-  Paul Gaborit **Documentation Perl** Ecole des Mines d'Albi 2000
-  George Gardarin & Olivier Gardarin **Le Client Serveur** Eyrolles 1996

### Sites Web

-  <http://www.mysql.com> (manuel de documentations et cours)
-  <http://www.mysql.org> (téléchargement du MySQL)
-  [http://www.linux\\_france.org/prj/inetdoc](http://www.linux_france.org/prj/inetdoc) (documentation linux)
-  <http://www.perl.com> (documentation Perl)
-  <http://www.commentcamarche.net> (présentation des protocoles http, ftp, TCP/IP)
-  <http://www.alianwebserver.com/informatique/linux/apache.htm> (installation et configuration d'apache)
-  <http://www.trucsweb.com/securite/> (sécurité des serveurs web)
-  <http://www.mandrake.org> (obtention des logiciels)

## Glossaire

---

Ce glossaire explique les mots clés rencontrés dans le présent manuel. Ils sont classés par ordre alphabétique.

**ANSI** : (American National Standards Institute) Organisation américaine chargée de définir des normes.

**APACHE** : C'est un serveur Web (logiciel).

**Browser**: Un programme qui permet de lire des documents Web. Netscape est un browser. Le terme français est navigateur.

**CGI** : (Common Gateway Interface) ce sont les programmes qui sont lancés sur le serveur http après envoi par un lecteur d'un formulaire.

**Client** : C'est un programme qui est utilisé pour contacter un serveur. On parle alors de modèle client-serveur. L'avantage du modèle est que le client sait faire un certain nombre de tâches et ne soumet au serveur que les informations nécessaires. D'autre part un serveur peut fournir des clients sur PC Macintosh ou machine Unix.

**Connexion** : Installation permettant de relier un ordinateur au réseau Internet.

**Ethernet** : protocole de communication de bas niveau (câbles, cartes et logiciel) permettant à des ordinateurs de communiquer sur un réseau local. Ethernet de base permet de communiquer à 10 Mb/s, Ethernet base 100 permet de communiquer à 100 Mb/s.

**EWSD** : c'est un système de commutation électronique fournit par SIMENS .

**FTP** : (File Transfer Protocol) protocole d'échange de fichiers entre sites informatiques. En général les sites ouverts au public sont dits anonymous ftp car le nom de login est anonymous.

**HOST** : ordinateur sur lequel on se connecte (hôte).

**HTML** : (Hyper Text Markup Language) Les pages Web sont écrites dans un format assez simple, appelé HTML. On peut voir le contenu d'une page HTML dans un des menus d'un lecteur de Web en demandant à voir le code source.

**HTTP** : Un serveur http est un serveur chargé d'envoyer les pages Web (en HTML) à un ordinateur, lorsque on lis une page Web (hyper text transfer protocol).

**HTTDP** : Hypertext transfert protocole daemon : processus qui tourne en arrière plan et qui assure le service http.

**HTTPS** : c'est du http sécurisé via SSL ( secure sockets layer) .

**Hypertexte** : Ce sont des textes marqués dans un document qui permettent de naviguer vers d'autres documents. On pourrait parler de lien. Les hypertextes sont d'une couleur particulière pour être facilement identifiables et ils changent de couleur une fois utilisée.

**Login** : c'est votre nom de connexion sur un ordinateur. Vous donnez ce nom par un message d'invite que vous remplissez soit au moment où vous vous connectez sur un autre ordinateur, soit en commençant à travailler sur votre ordinateur.

**Microsoft IIS** : (internet information server), c'est un type de serveur Web.

**Microsoft PWS** : (personnel Web server), c'est un serveur Web personnel.

**MIME** : (Multi Purpose Internet Mail Extensions) Système d'encodage permettant d'expédier fichiers attachés aux courriers électroniques.

**Naviguer** : action de se promener de site Web en site Web au moyen d'un logiciel de visualisation de Web. Le terme fureter est préféré par les Québécois.

**Page d'accueil** : première page d'un site Web

**Password** : Mot de passe accompagnant votre nom d'utilisateur (login) et permettant d'assurer la confidentialité de votre compte. Il est personnel et confidentiel (comme un numéro de carte bleue). Généralement les logiciels affichent votre mot de passe avec des étoiles.

**Perl** : ( Pratical extraction and report language) c'est un langage de scripts, on peut l'utiliser sans compilation.

**Port** : Numéro précédé du symbole : dans certaines adresses. Cet artifice permet de mettre plusieurs serveurs sur une même adresse avec une différenciation par le numéro de port.

**Protocole** : le mot protocole désigne en général les messages échangés entre deux machines. L'intérêt d'un protocole est de définir des méthodes d'échange d'information, indépendantes des matériels. Ainsi, une fois le protocole défini, chaque terminal ou client ou serveur implémente ce protocole sans se soucier des autres ordinateurs.

**Serveur** : un ordinateur qui fournit des services à des clients. Il fournit ces services à des ordinateurs par des messages ce qui permet d'avoir plusieurs type de clients.

**SGBD** : c'est l'abréviation du système de gestion de base de données.

**SGBD-O** : système de gestion de base de données objet.

**SGBD-R** : système de gestion de base de données relationnel.

**SQL** : (Standard query language) langage standard des requêtes.

**TCP/IP** : TCP/IP est le nom de la partie cachée de l'Internet. Il existe plusieurs protocoles réseau (Netware, LanManager...). TCP/IP est le plus propice aux interconnexions de réseaux. Il existe beaucoup de réseaux TCP/IP qui ne sont pas reliés à l'Internet.

**Telnet** : logiciel permettant de se connecter sur un serveur pour y exécuter des commandes.    TN3270 : Logiciel permettant de se connecter sur un host IBM.  
**UNIX** : système d'exploitation multitâche multi-utilisateurs à temps partagé.  
**URL** : (Uniformed Resources Locator) Ce sont des textes de couleur différente (par défaut ils sont en bleu) dans un document qui permettent lorsqu'on clique dessus avec la souris d'aller vers l'endroit référencé.  
**Web** : en français, toile d'araignée ; symbolise le réseau maillé de serveurs d'informations formant une toile d'araignée. Ces serveurs vont des pages personnelles aux interfaces de base de données. Par extension on parle de Web pour un serveur de page HTML.  
**WWW** : (World Wide Web) Système mondial d'interconnexion des informations par le protocole Web.