Institut Supérieur des Etudes Technologiques en Communications de Tunis

Projet de fin d'études

ETUDE, CONCEPTION ET DEVELOPPEMENT D'UNE APPLICATION D'INSCRIPTION UNIVERSITAIRE EN LIGNE

Réalisé par
Khaled Lâabidi
TS5 – Télécommunication

Encadré par
Mounir Eddabbabi

2000 – 2001

Institut Supérieur des Études Technologiques en Communications de Tunis

Projet de fin d ’études

Étude, Conception et Développement d ’une Application d ’Inscription Universitaire en Ligne et Services Dérivés

Projet soutenu par

Lâabidi Khaled

pour l ’obtention du diplôme Technicien Supérieur en Télécommunication

Encadré par Mr : Mounir Eddabbabi

Février 2001

Plan de l ’exposé

- Introductif
- Ce qui est réellement demandé
- Au départ…organisons les données
- Architecture de la solution : les "comment" et les "pourquoi" du projet
- Comment ça marche : une démo
- Conclusions et Perspectives

es services en ligne sur Internet :

Une mode...

➢**Croissance de la culture Internet**

Un besoin...

➢**Facilité : Pointes et cliques …c 'est fini !!**

➢**Rapidité d 'accès**

➢**Disponibilité**

Une technologie...

➢**Web**

➢**Client - Serveur**

➢**Sécurité**

➢**Outils dédiés**

NTIC

Nouvelles Technologies de l 'Information et des Communications

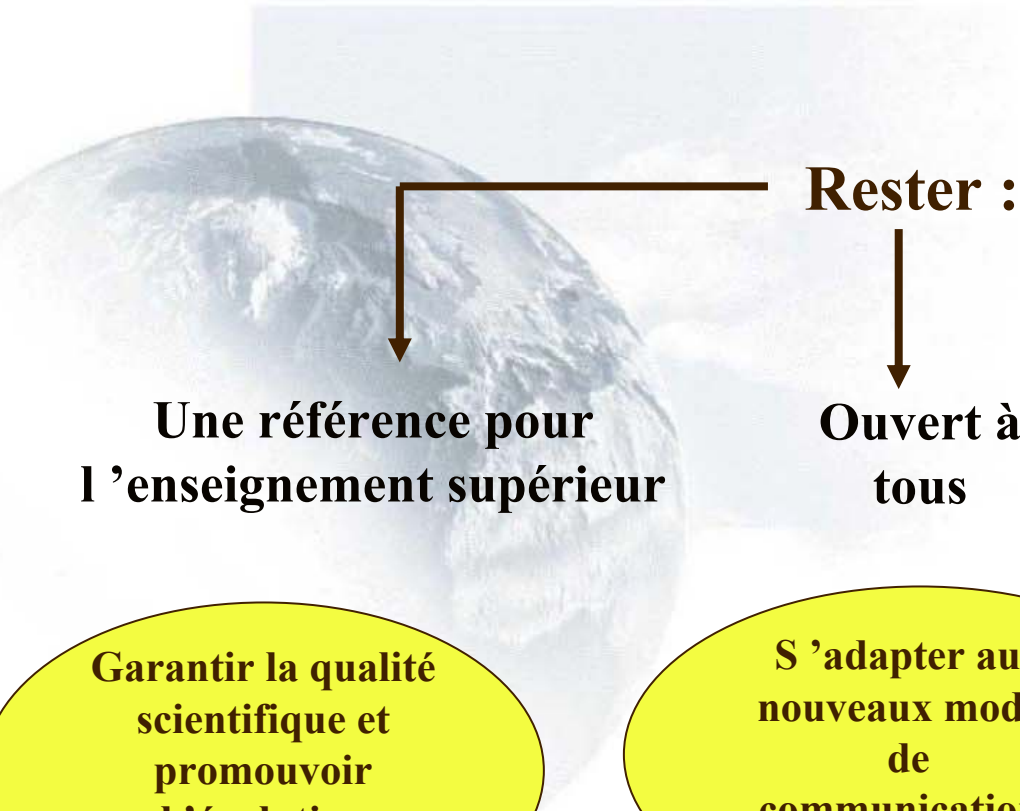
En-Ligne

Les bénéfices pour les utilisateurs :

- Elargir le champs d ’application sur Internet
- Un même environnement sans rupture pour les terminaux
- N ’importe où, des services adaptés aux besoin

En-Ligne

Les enjeux pour l 'Université :



Rester :

Une référence pour l 'enseignement supérieur

Ouvert à tous

Ouvert aux nouvelle pédagogies

Garantir la qualité scientifique et promouvoir l 'évolution pédagogique

S 'adapter aux nouveaux modes de communications sociale

Participer à l 'évolution du système éducatif

… Le Campus Virtuel

En-Ligne

Ce qui est réellement demandé 1

Fonctionnellement :

- Mettre en place le Site Web de l ’Université
- Garantir à chaque étudiant le bon déroulement de son inscription (mise à jour du dossier d ’inscription et paiement électronique)
- Réaliser des interfaces pour un accès simple et efficace aux données (Gestionnaire/scolarité)

Techniquement :

- Concevoir et implémenter la base de données du projet
- Installer et configurer le serveur Web
- Développer les modules d ’interfaçage entre le serveur et la base de données

Ce qui est réellement demandé 2

Mais en plus ...

- Sécuriser l ’application : Accès aux données et paiement
- Offrir un bon nombre de services : consultation des résultats en ligne, Bibliothèque Virtuelle, ...

Et à noter que ...

- La réalisation se fait sur 4 phases :
  - Étude et spécification
  - Conception
  - Développement et implémentation
  - Test et exploitation du projet

En-Ligne

Conceptuellement :

- Clarté, cohérence, complétude, fidélité et non redondanc des données : c 'est recommandé et exigé
- Extensibilité : Permettre l 'ajout de nouvelles tables dans la base pour suivre les nouvelles exigences et services

Implémentation :

- Utiliser un SGBD Performant : Rapidité d 'accès et sécurité des données : MySQL
- MySQL : supporte les forts pic de charge, compatible avec les standards du marché et offre un mécanisme de sécurisation d 'accès aux données

Architecture du projet 1

Perl

Modules CGI

Powered by APACHE

Powered by... FreeBSD

mySQL

client

Serveur Web de l'université

Base de données de l 'université

Secured by APACHE SSL

Passerelle de paiement électronique sécurisée
www.etijara.tn

Base de données de la poste

En-Ligne

La Plate - Forme :

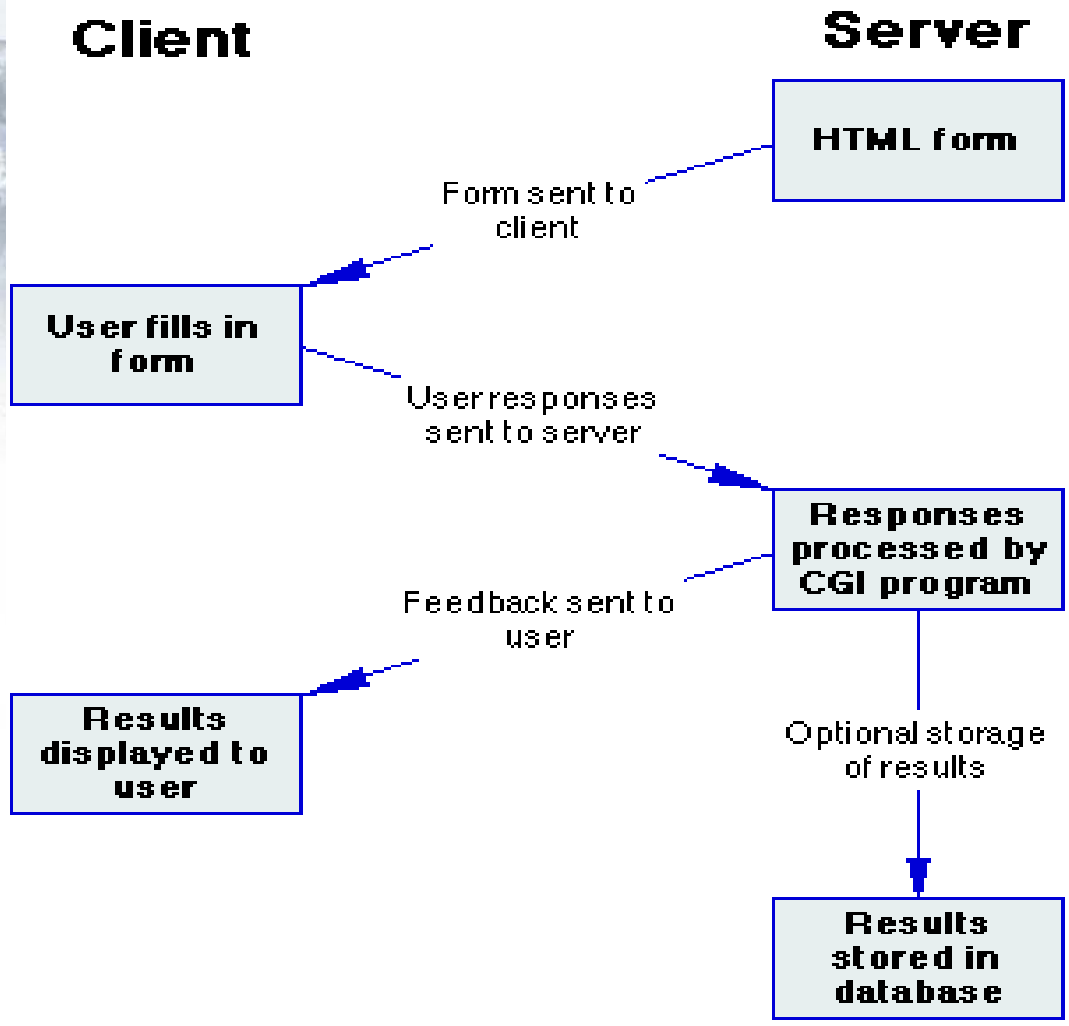
- Système : projet réalisé sous FreeBSD (Unix) mais pouvant être déployé sous NT, Solaris,...
- Serveur : Apache

  - ➢Logiciel libre
  - ➢Configuration simple et performante
  - ➢Support de protocoles de sécurité SSL

Passerelle Web - Base de données :

- La norme CGI : Common Gateway Interface
- Des programmes qui génèrent du HTML sur la sortie standard : accès aux bases de données, livres d'or, compteur d'accès…avec une multitude de langages : C, Shell, Python, Perl, …

- CGI : principe générale :



- Problème : CGI = programme exécutable sur le serveur
- Pour se protéger
  - Analyser et vérifier les données envoyées
  - Limiter la puissance des scripts et les privilèges d 'exécution
  - Eviter certaines commandes système dangereuses (eval(), exec(), …)
- Le langage Perl :
  - Logiciel libre
  - Performances : robuste, simple, portable, ...

E-Dinars et paiement électronique :

- **La monnaie électronique :**
  - **Simple**
  - **Anonyme**

- **Le Dinars électronique :**
  - **Un engagement de la Tunisie dans l ’e-business**
  - **Utilisé pour des paiements à partir de sites marchand Tunisiens**
  - **Surtout : les factures (Téléphone, STEG, …)**
  - **Et dans ce projet : utilisé pour le paiement des frais d ’inscription**

En-Ligne

Et à propos de la sécurité :

- Objectif : garantir un transfert sécurisé des données
- Exigences pratiques :

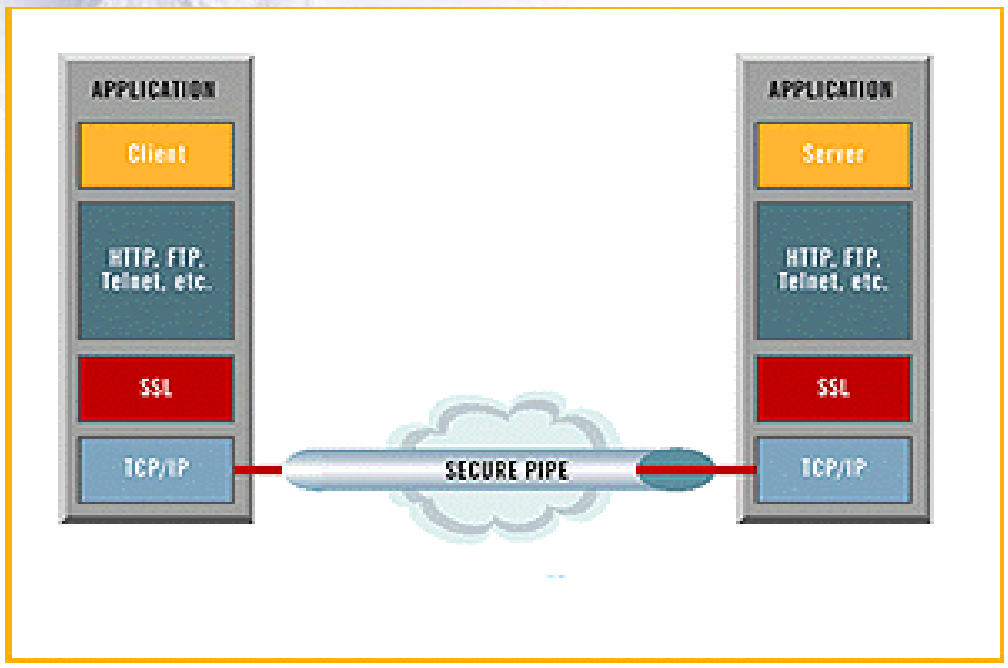
Authentification | Confidentialité | Intégrité | Non répudiation

- Technologies impliquées
  - La cryptographie
  - La certification numérique
  - Le protocole SSL

Un peu plus en détail ...

- **Le protocole SSL : Secure Socket Layer**

  SSL est un protocole de communication qui permet d'assurer l'authentification, la confidentialité et l'intégrité des données échangées. Repose sur l'algorithme RSA, un standard utilisé pour le cryptage des données et la signature de messages électroniques.



- **Les sockets sécurisées :**

  C'est un mécanisme de communication bidirectionnel en client-serveur introduit avec le système BSD (4.1, 4.3). Il fonctionne avec les protocoles de transport UDP, TCP, … et utilise des API de programmation identiques dans tous les cas. Une connexion est caractérisée par : le type du protocole, l'@ IP et le port associé au processus.

Le mot de fin…ou presque

Un projet modulaire en vue d'une utilisation de plus en plus évolutive, offrant des interfaces de gestion conviviales, simple et économique.

Cette application peut être encore enrichie d plusieurs fonctionnalités orientées services Web (Free mail, Forum, Bibliothèque Virtuelle, …) ou orientées administration (Coopération avec l'ONOU, SNT, …) renforçant l'économie immatérielle par l'utilisation du e-Dinars.

Enfin, Ce projet m' a permis de s'intégrer au sein de la société EchoNets et de côtoyer une équipe expérimentée pour approfondir le savoir et acquérir le savoir faire.

Dédicace

A tous ceux que j'aime... 

Remerciement

Après l'achèvement de ce projet de fin d'étude, je tiens à remercier l'ensemble des personnes qui m'ont aidé à le réaliser dans les conditions les plus favorables.

Un remerciement particulier à mon encadreur Mr Mounir Eddabbabi pour tout le soutien morale et technique et son suivi précieux au projet.

Résumé :

Les dernières évolutions des services en ligne déployés sur Internet ont engagé les entreprises et les organisations diverses à suivre de près ces évolutions pour en tirer le meilleur profit. En Tunisie, après l'émergence du commerce électronique d'une part et la croissance de la culture Internet d'autre part, le secteur universitaire vient d'affranchir une nouvelle étape vers le campus universitaire : l'inscription universitaire en ligne. Ce travail consiste en une étude, conception et développement de cette application tant sur le plan administrative (gestion de la scolarité) qu'économique par l'utilisation du E-Dinar comme moyen de paiement électronique.

Mots clés :

Inscription en ligne, Conception, Base de Données, Développement, CGI, E-Dinars .

Table des matières

Introduction générale **11**

Chapitre 1 Cahier de charges du projet **13**

Chapitre 2 Système de données et problématique de la conception **14**
Introduction : 14
I - Notion de système de données : 14
II - Variété d'expressions données au système de données : 15
III - Inventaire des problèmes : 17
IV - une conception par étapes : 17
Conclusion : 18

Chapitre 3 Conception détaillée de la base de données **20**
Introduction : 20
I - Objectifs de l'étape conceptuelle : 20
II - PRINCIPES ET FONDEMENTS DE LA CONCEPTION : 21
II.1 - Analyse : 21
II.1.1 - Analyse du réel : 21
II.1.2 - Modélisation : 21
II.2 - Modèle conceptuel des données : 21
II.2.1 - Concepts de base : 21
II.2.2 – Qualités du modèle conceptuel des données : 21
III - LE MODELE RELATIONNEL : 22
III.1 - Historique et définition : 22
III.2 - Les caractéristiques du modèle relationnel : 22
III.3 - Les concepts de base du modèle : 23
III.3.1 - Domaine : 23
III.3.2 - Relation : 23
III.3.3 - Attribut : 24
III.3.4 - Clé et contraintes d'intégrité : 24
III.4 – Avantages et inconvénients du modèle relationnel : 25
III.4.1 avantages : 25
III.4.2 inconvénients : 26
IV – base de données du projet : universite 26
IV.1 – Inventaire des entités et association : 26
IV.2 - Conceptualisation : 26
IV.2.1 - Les relations de la base de données 27
IV.2.2 - LE MODELE RELATIONNEL DES DONNEES : 29
CONCLUSION : 29

Chapitre 4 Inscription en ligne : Architecture de la solution proposée **31**
INTRODUCTION : 31
I - Architecture de la solution : 31
II – composants principaux de la solution : 32
II.1 – Le client : 32
II.2 – Le serveur Web : 32
II.2.1 – Le site Web : 32
II.2.2 – Les modules de traitements : 32
II.4 – La passerelle de paiement : 32

II.5 – La base de données de la poste : 32
III – plan de navigation : 33
III.1 – Page d'accueil : 33
III.2 – Activation de la carte E-Dinar : 33
III.3 – Identification de l'étudiant : 33
III.4 – Mise à jour des renseignements : 33
III.5 – Confirmation des renseignements : 33
III.6 – Identification de la carte E-Dinars : 33
III.7 – Paiement des frais d'inscription : 34
III.8 – Confirmation de l'inscription : 34
IV – modèle des traitements : 34
IV.1 – le modèle requête-réponse : 34
IV.2 – traitement des exceptions : 34
CONCLUSION : 35

Chapitre 5 Réalisation et implémentation de la base de données 36
INTRODUCTION : 36
I - Les SGBD relationnels : 36
I.1 – Définitions : 36
I.2 - Objectifs des SGBD : 36
I.3 -Fonctionnalités des SGBD : 37
I.4 – architecture et fonctionnement : 39
I.5 – Le langage SQL : 40
I.5.1 – Intérêt de SQL : 40
I.5.2 - Base de données et client - serveur : 41
II - Choix technique pour le projet : MySQL 43
II.1 - Qu'est ce que MySQL ? 43
II.2 - Pourquoi a-t-on choisit MySQL ? 43
II.3 – Obtention et Installation : 44
II.3.1 – Compilation et installation : 44
II.3.2 - Test de l'installation : 45
III- Serveur MySQL : 45
III.1 – Connexion au serveur MySQL : 45
III.2 – Système de droit et contrôle d'accès : 45
III.2.1 – But : 45
III.2.2 – Noms et mots de passe des utilisateurs : 45
III.2.3 – Fonctionnement : 46
III.3 – Lancement du serveur MySQL : 47
III.4 – Arrêt du serveur : 47
IV – programmation MySQL : 47
IV.1 – Création de la base de données UNIVERSITE : 47
IV.2 – Création des tables de la base : 47
IV.3 – Test de la base : 48
IV.4 – Alimentation de la base de données : 49
Conclusion : 49

Chapitre 6 Serveur Web et interfaces clients 50
INTRODUCTION : 50
I - système client : 50
I.1 – Performances des applications orientées Web : 50
I.2 – Développement Web coté client : 51
I.2.1 – Les formulaires HTML : 52
I.2.2- Le langage JavaScript : 53
II – Le protocole HTTP : 55
I.1 – Requête HTTP : 55
II.2 – Réponse HTTP : 56
II.3 – Les problèmes de HTTP : 58
II.4 – HTTP-NG : la nouvelle génération : 58
III – Le serveur Web : 59

III.1 – Principes et fonctionnement : 59
III.2 – Performances d'un serveur Web : 59
III.2.1 – Exécution des scripts et autres programmes : 59
III.2.2 – Compression et cryptage des données : 60
III.2.3 – Accès aux données : 60
III – Choix technique : Apache 60
III.1 – Pourquoi APACHE ? 60
III.2 – Installation : 61
III.3 – Configuration : 62
III.4 – démarrage du serveur Apache : 62
III.4.1 – Démarrage sous UNIX : 63
III.4.1 – Démarrage sous Windows : 63
Conclusion : 63

Chapitre 7 Passerelle Web - base de données : CGI 64
INTRODUCTION : 64
I – techniques utilisées : 64
I.1 – ASP : (Microsoft) 64
I.2 - JSP : (Java) 65
I.3 – PHP : (logiciel libre) 65
I.4 - Meta-HTML : 65
I.5 - ePerl 65
II – Choix technique : CGI 66
II.1 – Définition : 66
II.2 – Spécifications : 66
II.3 – Installation : 66
II.4 - interface CGI : 66
II.4.1 – variables d'environnement : 67
II.4.2 – Entrées – Sortie : 68
II.5 – comportement spéciaux de CGI : 68
III – le langage Perl : 68
III.1 – Pour quoi Perl : 68
III.2 – Perl et CGI : 69
III.3 – accès aux bases de données : 70
IV – déboguer les scripts : 70
V – aspect sécurité : 70
V.1 – Aperçu : 70
V.2 –comment se protéger : 71
Conclusion : 71

Chapitre 8 Paiement électronique : E-Dinars 72
Introduction : 72
I - Paiement sur Internet : les enjeux 72
I.1 - Argent électronique : 72
I.1.1 – Principe : 72
I.1.2 – Caractéristiques : 72
I.1.3 – Exemples : 73
I.2 - Le E-Dinar : 73
II - Transaction et sécurité : 74
II.1 - Services de sécurité : 74
II.1.1 – Authentification : 74
II.1.2 – Confidentialité : 74
II.1.3 – Intégrité : 74
II.1.4 – Non répudiation : 74
II.2 - La cryptographie : 75
II.2.1 – Cryptographie à clé privée : 75
II.2.2 – Cryptographie à clé publique: 75
II.3 – Certification électronique: 76
II.3.1 – Autorité de certification : 76
II.3.2 – Certificat numérique : 77

II.4 – Le protocole SSL : 77
III - Technique de sécurisation du site : 78
III.1 – Récupération et installation des logiciels : 78
III.2 – Implémentation du certificat SSL serveur : 79
III.2.1 – Serveur Web et SSL : 79
III.2.2 – Génération de la demande de certificat : 79
III.2.3 – Déposer la demande de certificat : 80
III.2.4 – Envoyer les justificatifs : 80
III.2.5 – Installation du certificat : 80
Conclusion : 81

Conclusion générale 82

Bibliographie 83

Liste des figures

Figure 2.1 : rôle du système d'information dans la création de la base de données..... 15
Figure 2.2 : système d'accès aux données 16
Figure 1.3 :Les étapes de la conception de la base de données..... 18
Figure 4.1 : schéma générale de l'application..... 31
Figure 5.1 : architecture fonctionnelle des SGBD-R. 39
Figure 6.2 : Format de la ligne d'état d'une réponse HTTP 56
Figure 8.1 : paiement par argent électronique..... 73
Figure 8.2 : carte E Dinar. 73
Figure 8.3 : Hiérarchie des autorités de certifications..... 76
Figure 8.6 : Certificat du serveur secure-www.isetcom.mincom.tn..... 81

Liste des tableaux

Table 3.1 : énumération des entités de la base de données université. 29
Table 6.1 : Les méthodes HTTP..... 55
Table 6.2 : information envoyées par le client. 56
Table 6.3 : Codes résultats HTTP. 57
Table 6.4: Informations dans l'entête de la réponse HTTP..... 57
Table 6.5 : Quelques types MIME les plus utilisés. 58
Table 7.1 : variables CGI relatives à la requête. 67
Table 7.2 : variable CGI relatives la requête..... 67
Table 7.3 : variable CGI relatives la requête..... 67

Introduction générale

<<… La croissance d'Internet n'est pas une toquade ou une mode, mais la conséquence de la libération du pouvoir de créativité individuelle. S'il s'agissait d'un système d'économie mixte, il correspondrait au triomphe du marché libre sur la planification centralisée. En musique, ce serait du jazz par rapport a Bach. La démocratie par rapport à la dictature…>>

[The Economist]

Il n'est point hasardeux qu'une revue économique mondiale traite le sujet Internet avec aussi d'ouverture et optimisme si ce n'est la concrétisation des objectifs tracés depuis des années et qui ont révolutionné l'économie et la communication.

Parmi les aspects répondus qui ont marqué cette révolution, et outre le commerce électronique, on trouve le services en ligne, devenus une nécessité décrite par le besoin mais aussi par le rêve de diminuer et simplifier les distances, les frontières et l'inertie des procédures administratives, juridiques et techniques.

Ainsi, ces services en ligne assurent un accès à l'information à tout moment et en tout lieu, une information dynamique et personnalisée, des paiements en ligne sécurisés, des transactions interactives et une réactivité pour offrir encore des nouveaux services.

En Tunisie, ce secteur d'activité est en progression sans cesse et se base sur des convictions et des encouragements politiques leaders. On assiste à une émergence des banques en ligne qui permettent aux clients de gérer leurs finances, effectuer des virements, payer leurs factures et cela 7 jours sur 7, 24 heures sur 24. Mais aussi, la bourse en ligne, l'assurance en ligne, la réservation hôtelière en ligne…

Suite à des encouragements politiques et une progression technologique, c'est l'environnement universitaire qui est visé dans tous ses aspects : téléenseignements, inscription universitaire en ligne, bibliothèque virtuelle…bref le campus virtuel.

Ce projet de fin d'études est l'occasion pour étudier, concevoir, implémenter et exploiter le projet de l'inscription universitaire en ligne. Développé au sein de la société EchoNets, cette application se veut être un produit portable et implantable sur différentes plates-formes techniques et structures administratives universitaires et en particulier pour le réseau des instituts supérieures des études technologiques (ISET) ; il tient non seulement compte de l'inscription proprement dite en tant qu'une opération administrative et financière transactionnelle, mais apporte en plus la possibilité de suivie de la scolarité de l'étudiant sur tous ses aspects comme la consultation de résultats et des moyennes, la gestion de la scolarité, la gestion des affectation des bourses et des foyers…

Les objectifs de ce produit seront détaillés et les besoins étudiés en détail dans le premier chapitre consacré au cahier de charge du produit et à ses spécifications.

La conception de la base de données en premier lieu puis celle schéma général de l'application font objectif des trois chapitres suivants : système de donnée et problématique de la conception (Chapitre 2), conception détaillée de la base de données(Chapitre 3) et architecture de la solution proposée (Chapitre 4).

En suite 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 5) étude, installation et configuration du serveur Web (Chapitre 6), développement des modules d'interface entre la base de données et le serveur (Chapitre 7) et l'implémentation du module de paiement électronique par E-Dinar (Chapitre 8).

Chapitre 1 Cahier de charges du projet

Pour mettre en place le projet de l'inscription universitaire en ligne, un ensemble de règles et de contraintes fonctionnelles, techniques et méthodologiques sont exigées.

Du point de vue fonctionnel, ce projet devra permettre aux étudiants (réorientés, bacheliers ou en cours d'études) de s'inscrire dans leurs universités via Internet en accédant à une rubrique nommée " Inscription à distance " et en effectuant le paiement électronique par le E-Dinars.

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 de l'établissement. 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 données, serveur Web, …) basée sur des critères de choix qui doivent être argumentés.

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

Chapitre 2 Système de données et problématique de la conception

«… La réalité est universelle mais sa perception et son analyse sont spécifiques aux individus qui les font… »

[Guillaume Benci]

INTRODUCTION :

La fonction essentielle de tout système de données est de fournir à ceux qui l'utilisent des représentations appropriées et pertinentes de la réalités 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[1] dans notre cas), permettent de reconstruire les images de la réalité. Le projet de l'inscription universitaire en ligne, 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 conditionnées 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 suivi 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 SYSTEME DE DONNEES :

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

1 Voir le chapitre 7 : interfaçage Web et base de données – La norme **C**ommon **G**ateway **I**nterface.

données. Le système de données illustre comment on devra représenter, par des données, les types d'informations et des processus de traitements de ces informations qui sont pris en compte dans le projet.

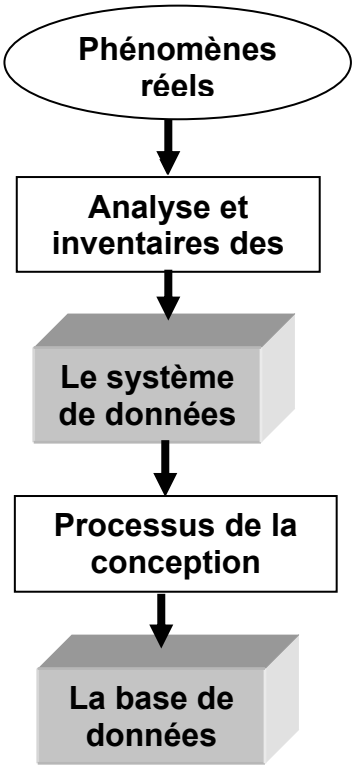


Figure 2.1 : rôle du système d'information dans la création de la base de données

II - VARIETE D'EXPRESSIONS DONNEES AU SYSTEME DE DONNEES :

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 d'inscription en ligne, ces intervenants se scindent en 4 catégories :

- La cible et le bénéficiaire principal de ce service qu'est l'étudiant (tel qu'il sera défini dans la conception détaillée de son profil).

- Le gestionnaire administratif (scolarité), financier (comptable) ou autres (responsable de la bibliothèque, du foyer, du restaurant,…) qui intervient pour la documentation, 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.

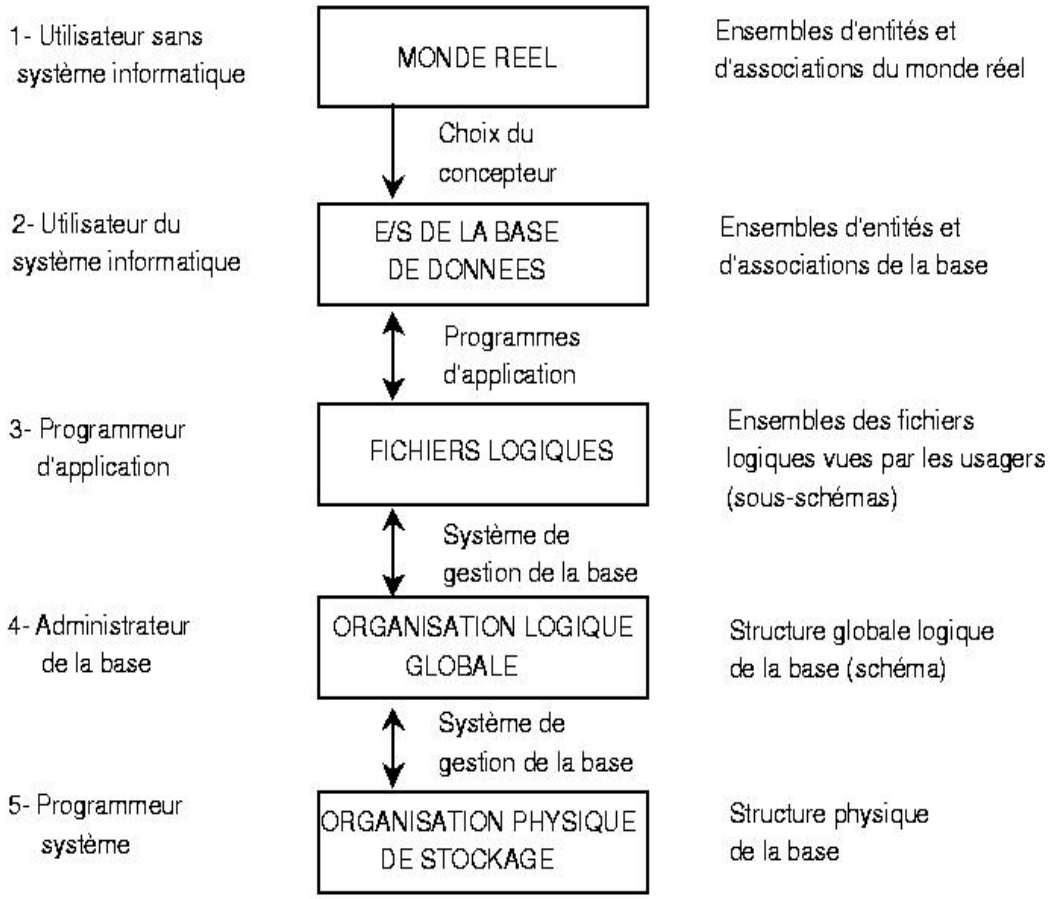


Figure 2.2 : système d'accès aux données .

III - INVENTAIRE DES PROBLEMES :

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

- 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 opérative aux demandes d'informations :
  - quelle implantation des données assurera une exploitation satisfaisante ?
  - quelle organisation des données et quel moyen d'accès retenir ?
  - comment assurer la sécurité et la confidentialité des données ?

IV - UNE CONCEPTION PAR ETAPES :

Tous les problèmes que nous venons d'évoquer justifient que l'en systématise la tache 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 ; par bonne, nous entendons complète, fidèle, cohérente et extensible.
- 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.

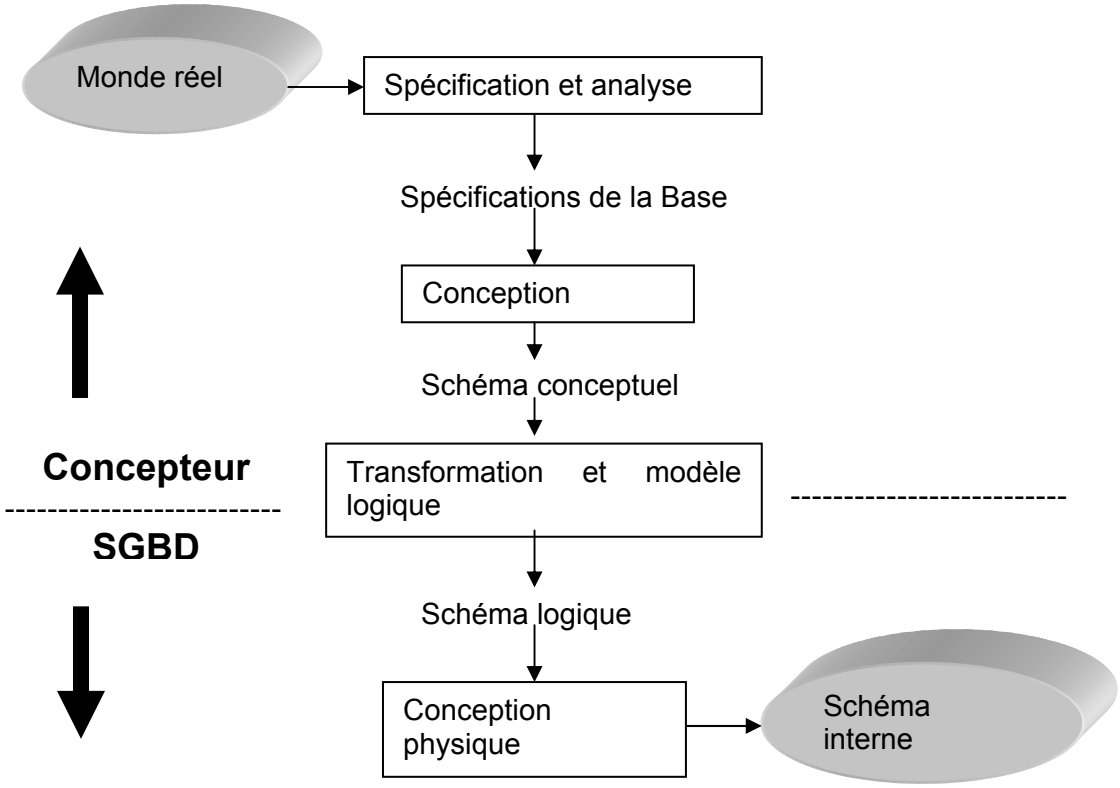


Figure 1.3 :Les étapes de la conception de la base de données.

CONCLUSION :

L'élaboration d'un système de données repose sur la description et la classification naturelle du 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 3 Conception détaillée de la base de données

« … The relationnel database model we championned 17 years ago is now the standard of our industry. SQL has become the universal lingua franca of information storage. Client-Server architecture have proved their value. Open system are increasingly triumphant… »

[L.Ellison – Oracle corporation]

INTRODUCTION :

On se propose de concevoir le schéma conceptuel de la base de données permettant la représentation la plus fidèle possible du processus de l'inscription universitaire en ligne ainsi que le suivi et la gestion de la scolarité et des autres services et procédures impliquées, de façon à en permettre une exploitation ultérieure par les modules de traitements.

I - OBJECTIFS DE L'ETAPE 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 considérations 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 FONDEMENTS 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 décrits 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 Etudiant), attribut(exemple prénom), valeur de l'attribut(exemple 05434698), association(liaison perçu entre entités), type(exemple date) et occurrence.

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 objet.

- 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.

III - LE MODELE RELATIONNEL :

III.1 - Historique et définition :

Le modèle relationnel de données a été introduit par l'américain E.F.Codd2 en 1969. Il a été développé à l'institut de recherche d'IBM à San José Californie et publié en 1970-1971. Ce modèle est aujourd'hui utilisé par la plupart des serveurs de données, bien que quelques uns commencent à proposer le modèle objet. En effet, le modèle relationnel présente plusieurs avantages, qui sont nos arguments concernant notre choix de la méthode relationnelle. Le modèle relationnel fournit une syntaxe commune aux deux mondes.

III.2 - 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 ne 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.

2 "A Relationnel Model for Large Data Banks" – Comm. De ACM juin 1979 par E.F.Codd

- 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.3 - 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, voire même comme un fichier. Chaque ligne de la table (appelée n-uplet ou tuple) peut être vue comme un fait décrivant une entité du monde. Une colonne de la table est appelée un attribut.

III.3.1 - Domaine :

Les ensemble de données de départ sont appelés des domaines. Un domaine est en théorie un ensemble quelconque de valeurs ; les domaines représentent par des valeurs atomiques – c'est à dire comme un tout indissociable – les objets dont on parle dans la base. La notion de domaine est cependant plus riche car un domaine peut être définit en extension (en donnant la liste des valeurs composantes) ou en intention (à partir de contraintes définies sur un type).

III.3.2 - 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 relation est composée de tuples (lignes ou enregistrements). Une représentation des relations sous forme de tables a été retenue dans les SGBD3 relationnels, si bien que l'on confond relation et table. De point de vue sémantique, deux notions sont à distinguer : l'intention d'une relation et l'extension d'une relation. L'intention d'une relation correspond à sa signification, c'est à dira à les prédicats et à d'autres contraintes ou lois qui déterminent la relation. Une extension de la relation est un ensemble de n-uplets qui satisfait les lois générales définissant son intention. On peut utiliser le terme relation pour désigner une extension et le terme schéma d'une relation pour désigner l'intention d'une relation.

3 Systèmes de Gestion de Base de Données

III.3.3 - Attribut :

Chaque ligne d'une relation correspond à un tuple et chaque colonne 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.3.4 - Clé et contraintes d'intégrité :

Un schéma de base de données est un ensemble de schémas de relation S = {R1, R2, ..., Rn} et un ensemble de contraintes d'intégrité CI. Une contrainte d'intégrité est une propriété du schéma, invariante 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 (un chef gagne plus que ses subordonnés). La plupart des contraintes ne sont pas supportées pas 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 bans 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.3.4.1 - Contrainte d'intégrité sur les clés :

Par définition, tous les n-uplets d'une relation sont distincts deux à deux (puisqu'il s'agit d'un ensemble). Il est également intéressant de définir des sous-ensembles du schéma qui permettent d'identifier de manière unique un nuplet (par exemple dans l'exemple de l'étudiant, l'attribut No-ss permet d'identifier un étudiant). Ces sous-ensembles du schéma s'appellent des clés. Le système doit donc garantir l'unicité des valeurs de clé (un seul n-uplet étudiant peut avoir une valeur de No-ss donnée). Il peut y avoir plusieurs clés pour une même relation (elles sont alors appelées clés candidates) et on choisit le plus souvent une clé particulière dite clé primaire qui servira d'identification privilégiée des n-uplets. Cette clé sera indiquée de manière graphique en mettant en gras les attributs la composant.

III.3.4.2 - Contrainte d'entité :

elle impose l'existence d'une clé documentée permettent l'identification de tout tuple d'un relation. Aucune composante d'une clé primaire ne peut être nulle. Une clé doit permettre une identification unique d'un tuple dans la base. Si une telle clé n'existe pas de façon naturelle il convient de la créer artificiellement, dans la plupart des cas, en donnant à chaque tuple un numéro spécifique.

III.3.4.3 - 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 du 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.4 – Avantages et inconvénients du modèle relationnel :

III.4.1 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. La notion du fichier n'existe que pour le DBA et est à la charge du SGBD.
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.
6. Gros développement des SGBD commercialisés et création du standard de cinquième génération SQL et QBE.

III.4.2 inconvénients :

1. Nécessité d'un SGBD puissant.
2. Perte d'un peu d'indépendance logique lord 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 DONNEES DU PROJET : UNIVERSITE

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 de renseignement, fiche du dossier d'inscription,…) nous ont conduit à dégager les entités suivantes :
Etudiant, bac, situation famille, réorientation, dossier, transaction, inscription, niveau d'étude, gestionnaire, agent comptable.
Cependant, ces entités peuvent encore augmenter si l'on considère un certain nombre d'autre services en ligne comme essentiellement la consultation des résultats ainsi que des informations relatives à la gestion de la session et de la navigation.

IV.2 - Conceptualisation :

La conceptualisation consiste à :

- Traduire par des schémas de relations les entités et associations qui ont été retenues lors de l'analyse.

- Transformer ces schémas par un processus de normalisation qui vise 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 schéma 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.

Etudiant et/ou Relation	Désignation symbolique	Définition & Descriptions
Etudiant *Entité sociale et juridique, qui suite à une orientation o réorientation, suit des études au sein de l'établissement en question.*	CIN	N° de l carte d'identité nationale ; sert comme clé de la table.
	NCE	N° de la carte étudiant.
	NOM	Nom
	PRENOM	Prénom
	SEXE	Sexe
	DATE_NAIS	Date de naissance au format jj-mm-aaaa
	VILLE_NAIS	Ville ou gouvernorat.
	PAYS_NAISSANCE	Payse de la naissance
	NATIOANILTE	Nationalité
	NUM_AFF_CNSS	N° d'affiliation au CNSS
	ETAT_CIVIL	Célibataire, marié, veuf, …
	ETAT_MILITAIRE	Sursis, accompli, ..
	NUM_PASSEPORT	N° du passeport pour identifier les étudiants étrangers
	BOURSE	Etudiant boursier ou non
	FOYER	Etudiant a droit au foyer ou non.
	SESSION_ID	Identificateur de session
BAC *Examen nationale aboutissant à l'obtention du diplôme permettant à l'étudiant de joindre l'établissement*	NUM_BAC	Numéro u bac
	AN_BAC	Année d'obtention
	SESSION	Principale ou contrôle
	SECTION	Math, science, …
	MENTION	Bien, assez bien, …
	PAYS	Pays d'obtention du bac
Famille *Description de l'état familiale de l'étudiant en se basant sue des informations communiquées par lui même*	NOM_PERE	Nom du père
	PRENOM_PERE	Prénom du père
	PROFF_PERE	Profession du père
	ADR_PROFF_PERE	Adresse professionnelle du père
	NOM_MERE	Nom de la mère
	PRENOM_ MERE	Prénom de la mère
	PROFF_ MERE	Profession de la mère
	ADR_PROFF_ MERE	Adresse professionnelle de la mère
	ADR_PARENT	Adresse des parents
	TELEPHONE_PARENT	Téléphone des parents

	MAIL_PARENT	e-mail des parents
	NOM_CONJOINT	Nom du conjoint
	PRENOM_CONJOINT	Prénom du conjoint
	PROFF_CONJOINT	Profession du conjoint
	ADR_PROFF_CONJOINT	Adresse professionnel du conjoint
	NBR_ENFANT	Nombre des enfants
Réorientation *Caractérisation de la réorientation*	CIN	
	ETAB_ORIGINE	Etablissement originaire de l'étudiant réorienté
	ANN_REO	Année de réorientation
	SESSION_REO	Session de la réorientation
	NIVEAU_REO	Niveau scolaire lors de la réorientation
Dossier *Description du dossier administratif de l'inscription*	CIN	
	NOM	
	PRENOM	
	DATE_DEPO	Date de déposition du dossier d'inscription
	DOSSIER_VALIDE	Dossier valide ou non
Transaction *Description de la transaction du paiement électronique*	TRANS_REF	Référence de la transaction
	DATE_TRANS	Date de la transaction
	MONTANT_CNSS	Montant CNSS
	MONTANT_MA	Montant mutuelle assurance
	MONTANT_FRAIS_INSC	Frais de l'inscription
	MONTANT_TOTAL	Montant de la transaction
	TRANS_VALIDE	Transaction acceptée ou non
Inscription *Etat finale de l'inscription*	CIN	
	NOM	
	PRENOM	
	ORDRE_INSC	Ordre de l'inscription
	INSC_VALIDE	Inscription valide ou non
	INSC_RET	Inscription retirée ou non
	DATE_RETRAIT	Date de retrait de l'inscription
Niveau *Niveau de scolarité de l'étudiant*	CIN	
	NCE	N° de la carte étudiant
	CYCLE	Cycle de formation (1er, …)
	FORMATION	Initiale, continue, …
	SPECIALITE	Spécialité de base
	DIP_PREP	Nature du diplôme préparé
	CLASSE	Classe courante de l'étudiant
	OPTION	Option
	REDOUBLANT	Redoublant : oui ou non
	NBR_REDOUB	Nombre des redoublements
Etablissement	CODE_INST	Code de l'établissement
	ADRESSE_INST	Adresse de l'établissement

Information à propos de l'institut	UNIVERSITE	Université(Tunis1, 2, …)
	NOM_INST	Nom complet de l'établissement
	SYS_PASSAGE	Semestre, année, …
Archive *Table d'archivage des étudiants qui ont quitté l'établissement*	CIN	
	NOM	
	PRENOM	
	DATE_ENTREE	Date d'entrée à l'établissement
	DATE_DEP	Date de départ de l'établissement
	DIP	Diplôme obtenu
	RAISON	Raison de départ

Table 3.1 : énumération des entités de la base de données université.

IV.2.2 - LE MODELE RELATIONNEL DES DONNEES :

Le terme modèle relationnel des données a été introduit par un groupe de travail de l'ANSI-SPARC pour qualifier l'expression conceptuelle de la base de données. Ce groupe définit le modèle relationnel des données comme « la vue qu'a l'entreprise de la nature qu'elle tente de modéliser dans la base de données »[4] .

CONCLUSION :

La base de données représente le pilier de base dans ce projet puisqu'elle décrit l'application dans ses composantes (données) statiques et dynamiques. 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 de données ? » La réponse à cette question se trouve dans le mot clé "en ligne" qui invoque un sujet suscitant de plus en plus de recherche et qui est le « DATA-WEB » et qui sera l'objectif du chapitre consacré aux techniques du Data-Web. Mais pour pouvoir accéder à un 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 entres 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

4 ANSI Final Report ANSI/X3 Sparc Study Group on DBMS – Washington 1977

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 4 Inscription en ligne : Architecture de la solution proposée

INTRODUCTION :

La solution du projet qu'on va proposer au cours de ce projet, intègre des acteurs 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 Web 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 Web de l'institut, le serveur de base de données, la passerelle de paiement sécurisé et le base de données de la poste.

I - ARCHITECTURE DE LA SOLUTION :

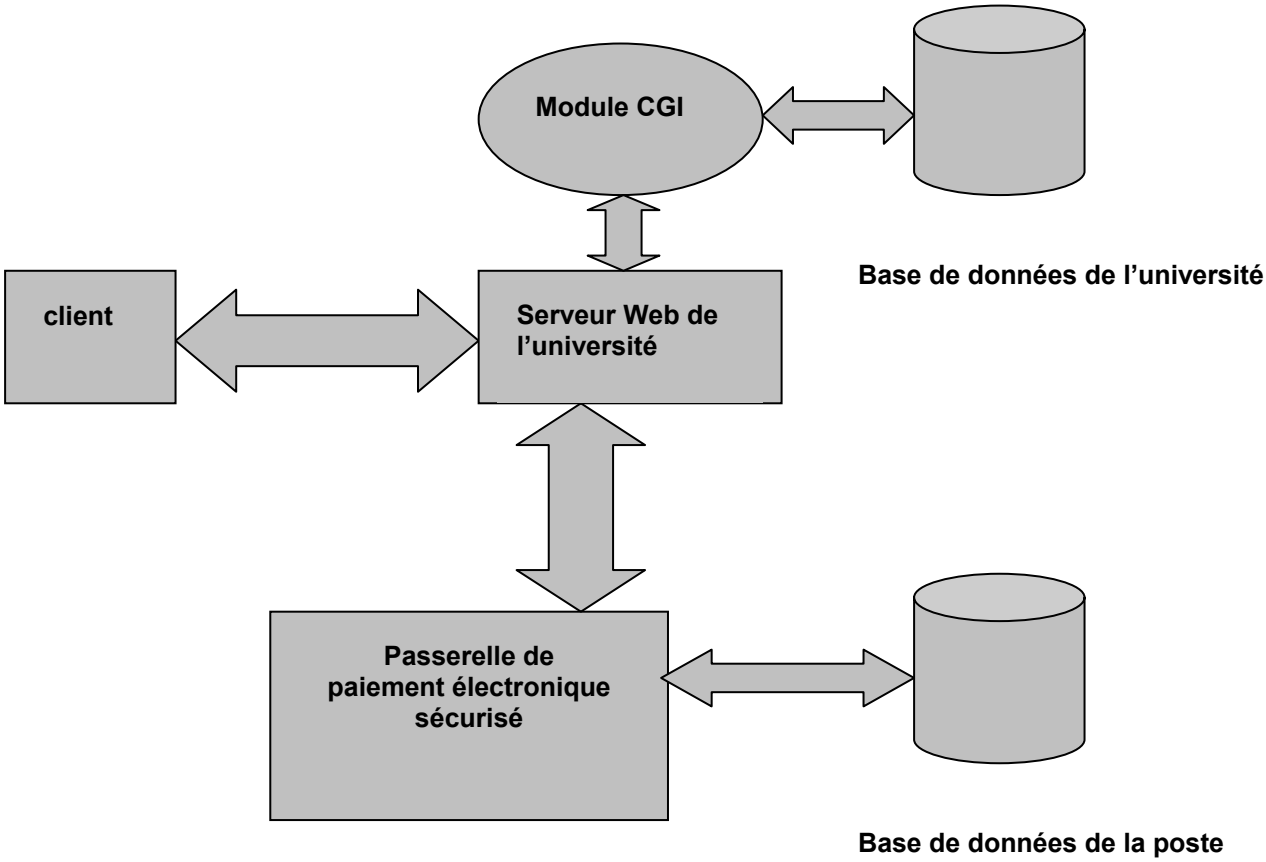


Figure 4.1 : schéma générale de l'application.

II – COMPOSANTS PRINCIPAUX DE LA SOLUTION :

II.1 – Le client :

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

II.2 – Le serveur Web :

Le serveur joue un rôle principal car il représente le point de connexion à partir du quel sera reçues ou envoyées les requêtes et les réponses. Dans ce niveau, on parle surtout de deux entités primordiales :

II.2.1 – Le site Web :

Le site Web hébergé par le serveur contient les différentes rubriques que l'établissement veut intégrer comme le moteur de recherche, la bibliothèque virtuelle, les études, …mais surtout la rubrique de l'inscription en ligne qui sera présentée sous forme d'un lien hypertexte dont l'apparition dans la page d'accueil ne sera réelle que durant les périodes effectives d'inscription(génération dynamique à partir du serveur Web).

II.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. Ces scripts permettent aussi de communiquer et recevoir des informations sécurisées entre le serveur Web et autres serveurs comme la passerelle de paiement.

II.4 – La passerelle de paiement :

Il s'agit de la passerelle de paiement sécurisé tunisienne http://www.etijara.tn. le paiement sera effectué par E-Dinars.

II.5 – La base de données de la poste :

Cette base sera utilisée pour s'assurer de l'existence réelle de la carte E-Dinars en question ainsi que pour la mise à jour du compte correspondant.

III – 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 d'inscription depuis la connexion au site jusqu'à la confirmation final du bon déroulement de l'opération.

III.1 – Page d'accueil :

C'est la "Home-Page" du site officiel de l'établissement. elle doit impérativement contenir le lien vers la rubrique "inscription en ligne".

III.2 – Activation de la carte E-Dinar :

Cette page inclut des informations générales sur les modalités de l'inscriptions et le différente options et spécialités disponibles, nombre de places, ainsi qu'un bref explicatif sur la procédure d'inscription en ligne. On doit aussi prévoir l'occasion aux étudiants d'activer leurs cartes E-Dinar avant d'effectuer le paiement, et ce en se connectant au site e-dinar.poste.tn. dans le cas contraire, un autre lien aiguille le navigateur vers la page d'identification.

III.3 – Identification de l'étudiant :

Cette page permet de saisir le N° de la carte d'identité nationale de l'étudiant ainsi que son numéro du BAC

III.4 – Mise à jour des renseignements :

Cette page est envoyée une fois que l'identification s'est ben déroulée et permet à l'étudiant de voir les données qui le concernent ainsi que la possibilité de les modifier en cas d'erreurs.

III.5 – Confirmation des renseignements :

Cette page permet de visualiser les informations saisies et/ou modifiées avec la possibilité de les modifier de nouveau ; si non, la confirmation par l'étudiant fait automatiquement pointer son navigateur vers le site de paiement électronique "https://www.etijara.tn"

III.6 – Identification de la carte E-Dinars :

Le serveur etijara va générer le formulaire d'identification avec comme informations : le bénéficiaire qui est le nom de l'établissement, le montant nécessaire à la transaction ainsi que

le numéro de la CIN de l'étudiant. Les informations à saisir seront le N° de la carte E-Dinar ainsi que le mot de passe correspondant.

III.7 – Paiement des frais d'inscription :

Une fois que la carte E-Dinars ait été reconnu, la transaction s'effectue et une page de récapitulatif sera générée.

III.8 – Confirmation de l'inscription :

Apres avoir effectuer toutes les étapes précédentes, le serveur Web envoie une page de récapitulatif affirmant le bon déroulement de l'opération pour l'étudiant en question ainsi qu'un ensemble d'autre informations relatives à l'inscription.

IV – MODELE DES TRAITEMENTS :

IV.1 – le modèle requête-réponse :

Tous les module 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 de l'étudiant et retransmission du formulaire vers le serveur
- Récupération des données saisies et traitement adéquat par un script serveur
- Génération e 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 un tâche particulière(identification, mise à jour des données, communication d'informations à la passerelle de paiement, …). Le choix de la technique de ces script et du langage de développement est un détail technique à négocier plus loin dans ce projet.

IV.2 – traitement des exceptions :

Rien n'empêche le client de communiquer des information 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 pris 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 meilleure rendement en matière de fonctionnalités et robustesse. Ceci va donc noue 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 5 Réalisation et implémentation de la base de données

INTRODUCTION :

Une approche de solutions techniques aux problèmes mentionnés 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(particularités et avantages) et finalement d'implémenter la base de données du projet.

I - LES SGBD RELATIONNELS :

I.1 – Définitions :

Un Système de Gestion de Bases de Données (SGBD) 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 (spécification des types de données à stocker), la construction d'une base de données et la manipulation des données (principalement ajouter, supprimer, retrouver des données). Les SGBD commerciaux les plus connus sont Oracle, Sybase, Ingres, Informix et DB2. Un SGBD sépare la partie description des données, des données elles mêmes. Cette description est stockée dans un dictionnaire de données et peut être consultée par les utilisateurs.

I.2 - Objectifs des SGBD :

- Indépendance physique : un remaniement de l'organisation physique des données n'entraîne pas des modifications dans les programmes d'application.

- Indépendance logique : un remaniement de l'organisation logique des données (ajout d'une nouvelle rubrique, ajout d'une nouvelle liaison…) n'entraîne pas de modifications dans les programmes d'application dont la "vision" logique n'a pas évolué.

- Manipulation des données par des langages non procéduraux : des utilisateurs non informaticiens doivent pouvoir manipuler simplement les données, c'est-à-dire les interroger et les mettre à jour sans préciser d'algorithme d'accès.

- Administration facilitée des données : un SGBD doit fournir des outils pour décrire les données, permettre leur suivi de ces structures et autoriser leur évolution. C'est la tâche des administrateurs de données : conception, création, maintenance, « arbitrage ».

- Efficacité des accès aux données : nécessité de garantir un bon débit (nombre de transactions exécutées par seconde) et un bon temps de réponse (temps d'attente moyen pour une requête). Partage des ressources (CPU, disques…) entre les utilisateurs en optimisant l'utilisation globale afin d'éviter, par exemple, qu'une requête courte d'un utilisateur attende la fin d'une requête longue d'un autre utilisateur.

- Redondance contrôlée des données : si redondance, volume de stockage plus important, opérations de mise à jour multiples, incohérences momentanées ou permanentes.

I.3 -Fonctionnalités des SGBD :

On donne ici les caractéristiques souhaitables des SGBD qui ne sont pas forcément prises en compte par les SGBD commerciaux.

- **Contrôler la redondance d'informations** : la redondance d'informations pose différents problèmes (coût en temps, coût en volume et risque d'incohérence entre les différentes copies). 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 des 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 des données (pour éviter par exemple qu'on puisse lire une information qu'on est en train de 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. On trouve des interfaces orientées utilisateur final (langages de requêtes déclaratifs comme SQL avec mise en œuvre graphique, interface de type formulaire, ...) ou bien orientées programmeurs d'applications (interface avec des langages de programmation classiques comme par exemple l'approche SQL immergé ou "embedded SQL").

- **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. On peut citer les modèles hiérarchique, réseau (première génération de modèles), relationnel (génération actuelle), sémantiques (ou orientés vers la conception tel que Entité-Association, Z, ...) ou orienté objet (la génération future ?).

- **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 classifier les contraintes d'intégrité, en contraintes structurelles et contraintes dynamiques. Les SGBD commerciaux supportent automatiquement un certain nombre de contraintes structurelles, mais ne prennent pas en compte les contraintes dynamiques (elles doivent être codées dans les programmes d'application).

- **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 pas tolérable qu'une panne puisse remettre en cause son fonctionnement de manière durable. Les SGBD fournissent des mécanismes pour assurer cette sécurité. Le premier mécanisme est celui de transaction qui permet d'assurer un comportement atomique à une séquence d'actions (elle s'effectue complètement avec succès ou elle est annulée). Une transaction est une séquence d'opérations qui fait passer la base de 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 des petites pannes (style coupure de courant).

En ce qui concerne les risques liés aux pannes disques, les SGBD s'appuie sur un mécanisme de journalisation qui permet de régénérer une base de données automatiquement à partir d'une version de sauvegarde et du journal des mouvements.

I.4 – 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 5.1 : Architecture ANSI/SPARC pour les bases de données

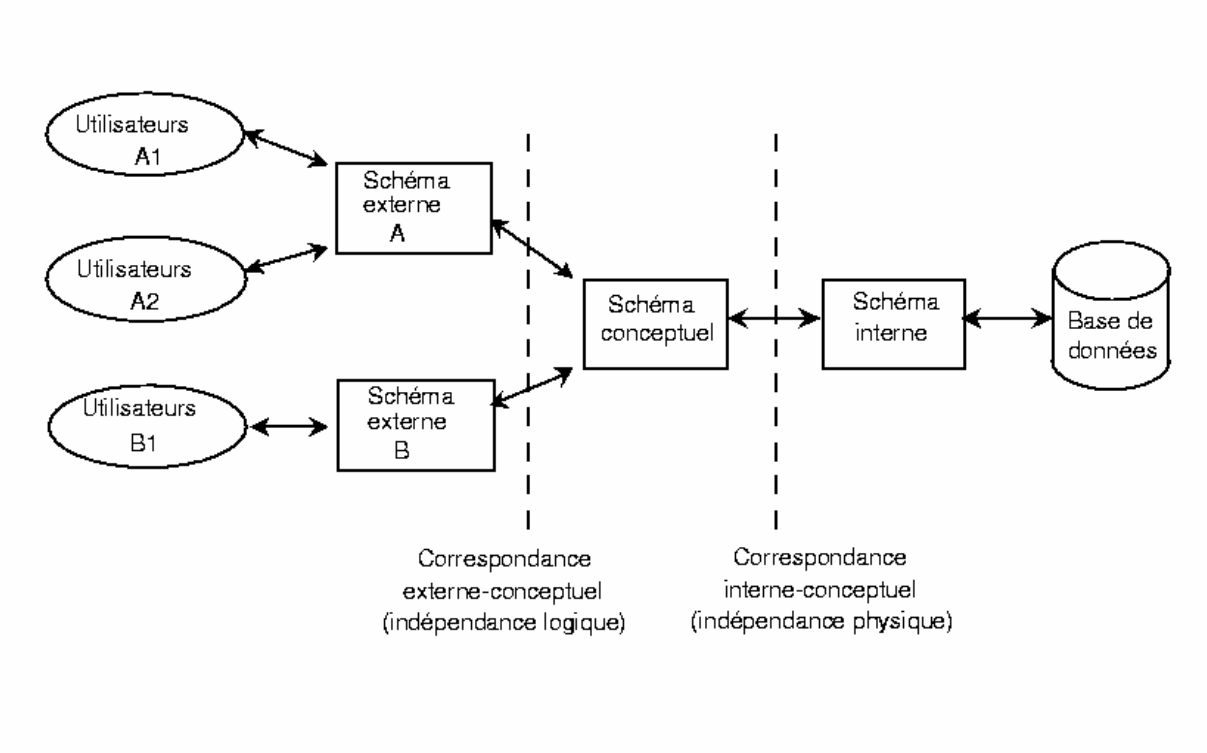


Figure 5.1 : architecture fonctionnelle des SGBD-R.

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 de données 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 à une classe d'utilisateurs.

I.5 – Le langage SQL :

I.5.1 – 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 plus de trente 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. Tandis que la station client exécute le code de l'application en gérant, en particulier, l'interface graphique, le serveur optimise la manipulation et le contrôle des données. De plus, les applications doivent être portables et gérer des données virtuelles, c'est-à-dire émanant de n'importe quel serveur. 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 et les protocoles de réseaux hétérogènes. L'accès et la manipulation des données ne sont 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 des 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.

I.5.2 - Base de données et client - serveur :

Nous allons d'abord revoir les différents types d'application possibles pour la gestion de données distantes en commençant par les trois formes les plus simples.

- Monoposte : la base de données se trouve sur un poste et n'est pas accessible en réseau. Il faut, dans ce cas, penser à la notion de sécurité si plusieurs utilisateurs peuvent interroger la base (suppression accidentelle d'enregistrements).

- Multiposte : basée sur des terminaux, liés à un site central. C'est l'informatique multiposte traditionnelle. La gestion des données est centralisée. Les applications ont été écrites par le service informatique et seules ces applications peuvent interroger le serveur.

- Multiposte : basée sur un serveur de fichiers. C'est la première forme (la plus simple) d'architecture client-serveur. Si l'applicatif sur un PC souhaite visualiser la liste des clients habitant Paris, tous les enregistrements du fichier CLIENT transitent sur le réseau, entre le serveur et le client, la sélection (des clients habitant Paris) est faite sur le PC. Le trafic sur le réseau est énorme et les performances se dégradent lorsque le nombre de clients augmente. Les serveurs de fichiers restent très utilisés comme serveurs d'images, de documents, d'archives.

De nouveaux besoins sont apparus comme la diminution du trafic sur le réseau pour augmenter le nombre de postes sans nuire au fonctionnement, traitement des volumes de données de plus en plus grand, accès de façon transparente à des données situées sur des serveurs différents, accès aux données de façon ensembliste, même si les données sont distantes, afin de diminuer le travail du programmeur, adoption des interfaces graphiques de type Windows pour les applicatifs clients.

I.5.2.1 - base de données et architecture client - serveur :

Dans une architecture client-serveur, un applicatif est constitué de trois parties : l'interface utilisateur, la logique des traitements et la gestion des données. Le client n'exécute que l'interface utilisateur et la logique des traitements, laissant au serveur de bases de données la gestion complète des manipulations de données

- Le serveur de bases de données fournit des services aux processus clients. Les tâches qu'il doit prendre en compte sont : la gestion d'une mémoire cache, l'exécution de requêtes exprimées en SQL, exécuter des requêtes mémorisées, la gestion des transactions, la sécurité des données.

- Le client doit ouvrir une connexion pour pouvoir profiter des services du serveur. Il peut ouvrir plusieurs connexions simultanées sur plusieurs serveurs. Le client peut soumettre plusieurs requêtes simultanément au serveur et traiter les résultats de façon asynchrone.

- Communication entre le client et le serveur. Puisque l'application doit pouvoir se connecter à divers serveurs de façon transparente, le langage de communication SQL doit être compatible avec la syntaxe SQL de chaque serveur pressenti. Or, malgré les normes, les dialectes SQL sont nombreux et parfois source d'incompatibilité. La seule façon de permettre une communication plus large est d'adopter un langage SQL standardisé de communication. Une couche fonctionnelle du client traduit les requêtes du dialecte SQL client en SQL normalisé. La requête transformée est envoyée au serveur. Celui-ci traduit la requête dans le dialecte SQL-serveur et l'exécute. Le résultat de la requête suit le chemin inverse. Le langage de communication normalisé le plus fréquent est l'**ODBC** (Open DataBase Connectivity) de Microsoft. Signalons également **IDAPI** (Integrated Database Application Programming Interface) de Borland. De plus, ces programmes permettent d'interroger des bases de données autres que relationnelles.

I.5.2.2 - Les serveurs de transactions :

Une transaction correspond à une procédure SQL. Il y a un seul échange requête-réponse pour la transaction. Le succès ou l'échec concerne l'ensemble des instructions SQL. Ces serveurs sont essentiellement utilisés pour l'informatique de production (ou opérationnelle) pour

laquelle la rapidité des temps de réponse est importante sinon essentielle.

II - CHOIX TECHNIQUE POUR LE PROJET : MYSQL

II.1 - Qu'est ce que MySQL ?

MySQL[5] est un langage rapide, multi-threaded, multi-utilisateur, c'est aussi un serveur de base de données SQL robuste. MySQL est une configuration client-serveur ce qui consiste en un serveur démon mysqld, différents programmes clients et des librairies. MySQL est suffisamment rapide et flexible pour gérer des historiques et des images. 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 que ce que pouvaient offrir les distributeurs de bases de données et il est utilisé depuis 1996 dans un environnement de plus de 40 bases de données contenant 10,000 tables, dont plus de 500 contiennent plus de 7 millions d'enregistrements et C'est environ 100 giga octets de données critiques. La base sur laquelle MySQL est construite est formée d'un ensemble de routines qui ont été largement éprouvées pendant des années dans un environnement 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 choisit MySQL ?

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

- 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 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

5 La prononciation classique de MySQL est ``Maille Ess Ku Ell".

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 ODBC 2.5 et d'autres sont disponibles et on peut, par exemple, utiliser Access 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 librairies 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 bytes.
  - 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 faites 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, les sockets Unix ou les pipes nommés sous NT.

II.3 – Obtention et Installation :

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 logicielle et n'offre que des versions d'essais. Pour la plupart des distributions des systèmes UNIX et LINUX, 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 répertoire de travails particuliers.

II.3.1 – Compilation et installation :

```
récupérer l'archive mysql-3.22-32.tar.gz (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
```

II.3.2 - Test de l'installation :

```
/usr/local/bin/safe_mysqld & (lancement démon)
/usr/local/bin/mysqladmin -u root password 'motdepasse' (mot de passe admin)
/usr/local/bin/mysqladmin -p status
/usr/local/bin/mysqladmin -p shutdown
```

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 certains 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 inclus la possibilité d'accueillir un utilisateur anonyme, et de donner des droits particuliers à des fonctions spécifiques.

III.2.2 – Noms et mots de passe des utilisateurs :

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, utilisé par MySQL pour l'authentification, n'ont rien à voir avec les nom d'utilisateur de Unix (Nom de login) ou de Windows. Les noms d'utilisateurs MySQL peuvent avoir jusqu'à 16 caractères de long.

Généralement, les noms d'utilisateur Unix sont limités à 8 caractères. Les mots de passe MySQL sont cryptés avec un cryptage différents 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 alors 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 :

- **Etape 1 : 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 niveau 2, pour attendre les requêtes. L'identité lors de la connexion est basé 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 si il existe un enregistrement qui contienne l'hôte, le nom d'utilisateur et le mot de passe.

- **Etape 2 : vérification des requêtes :** Une fois que la connexion est établie, le serveur passe en niveau 2. 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, et sinon, la requête n'est pas autorisée. Pour les requêtes liées aux bases de données, telles que insert, update, etc., 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 du 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 créé automatiquement s'il n'existe pas encore.

Si le serveur est déjà actif, le système répond *'A mysqld process already exists'.*

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 UNIVERSITE et l'import des données sur le SGBD MySQL.

IV.1 – Création de la base de données UNIVERSITE :

```
mysql> CREATE DATABASE UNIVERSITE ;
```

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

```
mysql> USE UNIVERSITE
Database changed
```

La base n'a besoin d'être crée 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 host -u user -p UNIVERSITE
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 d'un des tables les plus importantes du projet : ETUDIANT ; en se referant à sa structure dans le modèle relationnelle de la base.

```
mysql> CREAT TABLE ETUDIANT (
CIN INTEGER NOT NULL UNSIGNED
NCE INTEGER UNSIGNED
NOM VARCHAR(15)
PRENOM VARCHAR(15)
SEXE ENUM("male", "femelle")
DATE_NAIS DATE
VILLE_NAIS VARCHAR(20)
PAYS_NAIS VARCHAR(20)
NATIONALITE VARCHAR(20)
NUM_AFF_CNSS INTEGER UNSIGNED
ETAT_CIVIL INT
ETAT_MILITAIRE INT
 NUM_PASSEPORT INTEGER UNSIGNED
BOURSE  INT UNSIGNED
FOYER INT UNSIGNED
SESSION_ID INTEGER UNSIGNED
) ;
```

Pour une utilisation aussi efficace de l'espace mémoire, il faut utiliser le type de colonne le plus précis possible. La représentation des valeurs monétaires est un problème commun. Avec MySQL, le meilleur choix est le type DECIMAL. Il est enregistré comme une chaîne, ce qui n'entraîne aucune perte de données. Si la précision n'est pas primordiale, le type DOUBLE peut être un bon choix.

IV.3 – Test de la base :

la commande SHOW TABLES permet de vérifier que la base de données est implémenté sur le serveur

```
mysql> SHOW TABLES ;
```

Pour vérifier que la table a été créée comme on le désire, on utilise la commande DESCRIBE

```
mysql> DESCRIBE ETUDIANT ;
```

IV.4 – Alimentation de la base de données :

Après avoir créé la table, il faut la remplir. 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 étudiant, puis de le charger directement dans la table avec une seule commande. On crée ainsi un fichier etudiant.dat contenant un enregistrement par ligne, avec des valeurs séparées par des tabulation, et dans le même ordre que l'ordre dans lequel les colonnes ont été listées dans la commande CREATE TABLE. Pour les valeurs manquantes on peut utiliser la valeur NULL.

```
mysql>  LOAD DATA LOCAL INFILE "etudiant.dat" INTO TABLE ETUDIANT;
```

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.

CONCLUSION

MySQL est un logiciel libre (sous licence GPL) 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 Web. Il n'a pas la prétention de rivaliser avec Oracle ou les autres gros logiciels commerciaux. MySQL peut être utilisé dans des scripts PHP ou Perl inclus dans des documents HTML; ce qui est le cas le plus courant.

Chapitre 6 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 impotentes et décisives dans tous projet informatique orienté communication : 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 finalement est de réaliser les différents pages et traitements du coté client, puis l'étude des performances du serveur Apache qui est utilisé dans ce projet.

I - SYSTEME 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'es exactement ce que font tous les programmes client - serveur et notmment les application très répandues 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 plates-formes existantes. De plus, les langages HTML et Java sont eux aussi indépendants et non compilés. Ils sont interprétés au fur et à mesure par le navigateur. Une application peut être donc distribuée sur n'importe quelle plate-forme d'un réseau sans avoir à être préparée par compilation de la partie cliente selon la cible. On économise ainsi une grande quantité de travail par rapport au processus de développement classique.

- **Un accès universel et aucune limite au niveau au niveau des licences** : 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 normale, et on peut même l'insérer à l'intérieur d'un autre document. Les utilisateurs n'ont plus besoin d'appeler pour dire qu'ils ont besoin de l'application. l'installation n'est plus mise et les problèmes de licence n'est plus posé non plus : l'application est automatiquement accessible à tous.

- **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. le applications de gestion et autres applications client-serveur traditionnelle permettent aussi de générer des formulaires, mais ne peuvent pas enchâsser un formulaire dans un autre document pour lui donner un pouvoir d'attraction, avec éventuellement des composants multimédias et une fonction d'auto documentation.

Les différentes pages Web du projet de l'inscription universitaire en ligne, 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 et traitement 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 en envoyées par le client. On parle alors de Web dynamique dont la technique sera étudiée dans la chapitre qui suit. Cependant, la génération à la volée de code HTML impose la construction d'un modèle (Template ou gabarit) des différentes pages du scénario de navigation. Le langage de base est HTML dans sa spécification 4.0[6] enrichie de plusieurs fonctionnalités concernant les formulaires pour plus d'interaction et de souplesse.

6 W3C – 1999 HTML 4.0

I.2.1 – Les formulaires HTML :

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

I.2.1.1 - 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 et 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'envoie du formulaire ne change rien sur le site de destination qui est le cas typique des moteurs d'indexation Web.

- **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 d'inscription en ligne en sont le meilleure exemple.

Ces deux méthodes font l'objet des standards HTML, HTTP et 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 le programmeurs des scripts.

I.2.1.2 – 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.

I.2.1.3 – TARGET :

Cet attribut n'est obligatoire mais entre en jeu après l'envoie des données et au moment de la réception de la 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.

Ainsi, le code HTML implémentant le formulaire sera :

```
<FORM  METHOD={GET | POST} ACTION={URL} TARGET={_self | _new | …} >
```

I.2.1.4 – Implémentation :

Les formulaires offrent une variété d'objets interface pour la saisie des données depuis le client, le code suivant implémente un formulaire d'identification d'un étudiant :

```
<FORM  NAME="ident" METHOD="POST" ACTION="/cgi-bin/inscription/ident">
votre CIN : <INPUT TYPE="text"  NAME="cin" VALUE=""><BR>
numéro du bac ou carte étudiant :
<INPUT TYPE="text"  NAME="nce" VALUE="">
<INPUT TYPE="submit" VALUE="continuer">
</FORM>
```

I.2.2- Le langage JavaScript :

JavaScript est un langage produit par Sun Microsystem et Netscape Communication, procédurale mais basé sur des objets natifs et relatifs au système client[7]. C'est un langage interprété qui permet des contrôles sur les formulaires HTML ainsi que des effets d'animation

et de design avancés. L'évolution de JavaScript n'est pas normalisée et elle a aboutit à deux standards : JavaScript 1.3 de Netscape et Jscript 1.5 de Microsoft. Les différences ne sont pas profondes et c'est à la charge du programmeur d'assurer la portabilité de son code sur tous les navigateurs du marché.

I.2.2.1 – Interaction JavaScript et HTLM :

Le succès de HTML et de JavaScript fait apparaître des problèmes d'un nouveau type. L'un d'entre eux concerne l'organisation des projets de développement à grande échelle où les pages Web sont complexes et l'équipe impliquée importante. Dans de tels projets, il faut trouver un mécanisme fiable pour rassembler les travaux de chacun en un ensemble cohérent. On parle d'intégration de travail par composant ou « Componentising ». JavaScript interagit avec les formulaires HTML de nombreuses manières et apporte une réelle « plus-value » :

- Création de balises HTML suite à des décisions et traitements (dynamisme client).
- Gestion des événements, programmation événementielle et Contrôle de la navigation.
- Lecture, modifications et validation des champs des formulaires.
- Stockage et réacheminement des données entrées par l'utilisateur.

I.2.2.2 – Implémentation : technique avancée :

Les possibilités de JavaScript ajoutent de la fluidité au traitement des formulaires sans change fastidieux sur le réseau. Sans JavaScript, la moindre vérification nécessiterait une requête vers le serveur Web ce qui ralentirait considérablement les choses. Le code qui suit permet d'imposer un filtrage des caractères tapés pour des champs de saisie dont la valeur doit être numérique (CIN d'un étudiant). Cette technique d'anticipation est plus performante que les contrôles au moment de l'envoie et permet un allégement des modules du serveur.

```
<script langage=" javascript ">
if(document.layers)
document.captureEvents(Event.KEYPRESS) ;
document.onkeypress=h_keypress ;
function h_keypress (e)
{
if(document.layers && ((e.which<48 || e.which>57) && (e.which !=8)))
return false ;
```

7 Norme ECMAScript : Europeean Computers Manufactors Association .

```
if(document.all && (event.keyCode<48 || event.keyCode>57))
return false ;
return true ;
}
</script>
```

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 dans le cadre des systèmes hypermédias distribués en réseau. HTTP a été crée 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 envoi 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)

I.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** (URI) : 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'URI. 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 à l'avenir.

Table 6.1 : Les méthodes HTTP.

L'URI identifie l'objet recherché. Les URI doivent respecter un ensemble de règles définies par un standard international. Le Web utilise un sous-ensemble des URI : les URL (Uniform Ressource Locator) 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. Ces informations peuvent fournir l'identification ou l'autorisation de paiement.

Champ	**Information**
User-Agent	Type du navigateur client.
If-Modified-Since	Récupération si l'information est plus récente que la date mentionnée.
Accept	Type de données (MIME) acceptées par le navigateur.
Authorization	Information d'authentification (mot de passe,…).

Table 6.2 : information envoyées par le client.

Ainsi, pour obtenir la page Web index.html depuis le serveur , le client pourrait formuler le message suivant :

```
GET /index.html HTTP/2.0
User-Agent : Netscape Communicator 4.6
Accept */*
```

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 sur l'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 6.2 : Format de la ligne d'état d'une réponse HTTP

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

Code	**Raison**	**Explication**
200	Document Follows	Ok l'information va suivre
301	Moved Permantly	Document migré vers autre URL
302	Moved Temporarily	Document déplacé temporairement
304	Not Modified	Document non modifié
401	Unuthorized	Information protégée
402	Payement Required	Information soumise à un droit d'entrée
403	Forbidden	Accès non autorisée
404	Not Found	Information non trouvée
500	Server Error	Erreur dans le serveur

Table 6.3 : Codes résultats HTTP.

La méta-information dans la réponse indique au navigateur ce qu'il doit connaître pour interpréter et afficher l'information.

Champs	**Information**
Server	Type du serveur.
Date	Date et heure de la réponse.
Content-type	Type MIME de l'objet retourné.
Content-Length	Nombre d'octets de la réponse.
Content-Language	Langage d'expression de l'information.
Content-Encoded	Codage complémentaire comme la compression.
Last-Modified	Date et heure de la dernière mise à jour.

Table 6.4: Informations dans l'entête de la réponse HTTP.

Le Content-Type est codé à partir des types MIME sur la même base que la liste des types Accept du navigateur. Le MIME (Multi purpose Internet Mail Extension) est un standard international définissant les règles d'échange d'information pouvant contenir autres éléments que du texte. Le type MIME est capital pour les programmes et surtout pour le navigateur Web car MIME lui indique comment déchiffrer et afficher l'information. Du fait que chaque navigateur peut accepter différents formats de données et que chaque serveur dispose de plusieurs types de documents, le Content-Type permet au serveur de renseigner le client sur le contenu de la réponse.

Type MIME	Explication
Text/plain	Texte 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
x-world/x-vrml	Scène VRML

Table 6.5 : Quelques types MIME les plus utilisés.

II.3 – Les problèmes de HTTP :

La règle imposant que chaque requête doit traitée sur une connexion TCP différente représente le problème majeur de HTTP. TCP est dit « amnésique » car le serveur ne garde jamais en mémoire les requêtes précédentes, ce qui engendre deux types de problèmes : des faibles performances et des exécutions difficiles lors d'opérations demandant l'échange de plus d'une paire requête-réponse. Les problèmes de performance viennent du fait que HTTP fonctionne avec le protocole réseau TCP. Les difficultés tiennent du fait que HTTP ne permet pas au serveur de regrouper les requêtes d'une même famille en une session unique. Beaucoup de points faibles de HTTP peuvent être contournés par l'utilisation de scripts et autres programmes auxiliaires comme les procédés de paiement numériques utilisant des programmes bancaires spéciaux qui peuvent établir leurs propres connexions et les maintenir ouvertes durant toute la session. L'idée est que les programmes auxiliaires soient dédiés à une tache bien précise. On résout les problèmes de HTTP en utilisant d'autres programmes pour effectuer ce que les serveurs Web ne peuvent pas faire aussi bien.

II.4 – HTTP-NG : la nouvelle génération :

HTTP-NG est une nouvelle conception de HTTP à la base et non pas une simple extension ad-hoc du HTTP existant. Il tire partie de l'expérience acquise sur HTTP et anticipe également sur les technologies du futur. HTTP-NG abandonne le principe d'une requête par connexion qui pose tant de problèmes. Chaque connexion réseau est organisée en sessions multiples, chacune disposant de son propre canal. Un canal véhicule les informations de contrôle telles que les requêtes ou les messages d'erreur. Les données en relation avec un ou plusieurs fichiers sont transmises dans un canal propre à chaque fichier, tout le processus s'exécute sur une connexion réseau unique. HTTP-NG est particulièrement bien adapté pour les client-serveur fonctionnant avec des processus légers, qui peuvent tirer partie

d'architectures mulit-processeurs capables de traiter plusieurs requêtes à la fois. Il est également bien adapté aux nouvelles technologies réseau telles que ATM. HTTP-NG a de bonne perspectives d'avenir, cependant, les client et serveurs existants continueront d'utiliser HTTP pour de longues années ce qui fait que les fournisseurs d'information doivent continuer de supporter HTTP-NG dans un futur immédiat.

III – LE SERVEUR WEB :

III.1 – Principes et fonctionnement :

Un serveur Web est un ordinateur connecté à Internet, muni d'un logiciel système permettant son fonctionnement et sa connexion à d'autres systèmes Internet. Le matériel, son système d'exploitation et le logiciel réseau constituent la plate-forme informatique. La deuxième composante d'un serveur Web est tout simplement le logiciel serveur lui même : httpd[8]. Finalement, sans information à mettre en disposition des autres, un serveur n'est pas d'une grande utilité.

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'applications y compris le programme httpd et les scripts. Tout 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 nécessaires 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 des CGI sous UNIX.

8 Hyper Text Transfer Protocole Daemon

III.2.2 – Compression et cryptage des données :

L'utilisation des images 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 qui 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. Des entreprises comme nCipher offrent des boîtiers externes pour ces opérations (nShield et nForce) et qui donnent un résultat satisfaisant.

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.

III – CHOIX TECHNIQUE : APACHE

III.1 – Pourquoi APACHE ?

Lorsqu'on lit la presse spécialisée, on y trouve, à grands renforts d'avis de spécialistes, des analyses et comparatifs entre les serveurs HTTP des principales sociétés commerciales qui exercent dans le domaine. Rares sont les articles qui mentionnent Apache alors qu'il est le serveur HTTP le plus utilisé. Dire que la part de marché d'Apache est de l'ordre de 50 % serait un contre-sens dans la mesure où Apache est gratuit et ne vise donc pas à conquérir de quelconques parts de marché. Mais les chiffres sont là : avant même la sortie d'Apache sous Windows, il est utilisé sur 50 % des serveurs HTTP fonctionnant sur Internet. La principale différence entre Apache et les serveurs commerciaux réside en quelques différences 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 sécurisés en France, alors qu'il suffirait qu'une personne envoie les sources à ce même organisme pour qu'Apache devienne l'un des serveurs commerciaux les plus robustes et les mieux testés.

- Le couplage Apache Perl dans le monde Windows nécessite de modifier la première ligne des Script Perl pour les mettre sous la syntaxe : #!c:/perl/lib/perl.exe

Mis 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.

III.2 – Installation :

Aussi bien sous Windows que sous Unix, installer Apache ne pose pas de réelle difficulté. Sous Windows la version binaire est livrée avec un installer 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 pas non plus de problème particuliers. On va donc explorer en détail 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, mais généralement ce répertoire n'est pas gardé longtemps à cet endroit, on préfèrera utiliser un module comme CgiWrap permettant d'allouer à chaque utilisateur son espace protégé de programmes CGI.

- **conf** : est le répertoire où sont stockés les fichiers de configuration qui vont faire largement partie de la suite de notre exposé

- **htdocs** : est le répertoire par défaut où sont stockés les fichiers destinés à être publiés. Ce répertoire est souvent modifié comme le répertoire cgi-bin.
    - **manual** : contient diverses documentations en anglais.
    - **index.html** : par défaut le fichier index.html est celui qui est envoyé lorsque aucun fichier HTML n'est spécifié.

- **icons** : est le répertoire contenant toutes les icônes représentant les dossiers les fichiers textes etc. Il est donc possible de changer ces icônes pour personnaliser le serveur HTTP.

- **logs** : est le répertoire où sont stockés les fichiers mouchards :

- **access.log** : ce fichier contient les informations concernant les accès à votre serveur. Chaque ligne est par défaut de la forme :

  127.0.0.1 - - [30/May/1998:17:27:15 +0200] "GET /page1.shtml HTTP/1.0" 200 664

Les différents champs indiquent respectivement la provenance de la connexion, la date et l'heure de la connexion, la page vue et le code de lecture ainsi que la taille du fichier lu. Chaque ligne est appelée un hit et pour peu qu'une page HTML contienne dix images, le fichiers logs contiendra onze lignes relatives à m'envoi de cette page, ce qui fait souvent confondre le nombre de lecteurs et le nombre de connexions.

- **error.log** : ce fichier contient les erreurs produites par le serveur. C'est donc un fichier fort utile dans le cas de débogages de programmes CGI ou de surveillance d'une application.

- **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 !

III.3 – Configuration :

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

- **access.conf** : contient les instructions permettant de gérer les droits d'accès au serveur. Ce fichier est formé de lignes encadrées par des couples de balises <balise> </balises>.

- **srm.conf :** contient les instructions gérant les noms et les types accessibles aux utilisateurs connectés au moyen des navigateurs.

- **http.conf :** contient les directives propres au fonctionnement du serveur lui-même.

III.4 – démarrage du serveur Apache :

Une fois le serveur est près, l'étape suivante est de la démarrer réellement d'une manière qui dépend de la plate-forme et du mode d'installation.

III.4.1 – Démarrage sous UNIX :

Apres 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
```

III.4.1 – Démarrage sous Windows :

Il existe deux méthodes pour démarrer Apache sous la plate-forme Windows, l'une on lançant le programme Start->Programs->Apache Web Server->Start Apache. L'autre étant de le lancer à partir de l'invite de commande : C:\Program Files\Apache Group\APACHE>apache -k start.

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 intéresse 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 e chapitre qui suit.

Chapitre 7 Passerelle Web - base de données : CGI

« ...La devise de Perl est : "Il y a toujours plus d'une façon de le faire !"... »

[Larry Wall – créateur du langage Perl]

INTRODUCTION :

Poussés par le marché, les éditeurs de SGBD ont développé 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 dynamiques les serveurs Web et augmenter la bande passante entre le SGBD 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 SGBD. 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 comme celui de http://www.e-tijara.tn. l'objectif de cette étape serait de faire un bref comparatif entre les techniques utilisées, puis étudier la norme CGI qui sera utilisée et finalement d'implémenter les différents modules du projets.

I – TECHNIQUES UTILISEES :

Pour le développeur, le changement intervient au quotidien, avec l'apparition de technologies et d'outils de telle sorte qu'on commence à dire qu'une année Internet vaut sept ans réelles. Car qui avait entendu parlé de PHP il y a seulement deux ans ? Qui aurait parié sur le succès de Java pour autre chose que le développement d'Applets ? Qui aurait prédit que les standards du Web seront le format de représentation de données avec le nouveau venu XML qui s'impose comme solution de normalisation adoptée par tous les acteurs du marché, Microsoft et IBM en tête ?

I.1 – ASP : (Microsoft)

Cette technologie repose sur différents éléments : elle comporte une API serveur dite ISAPI, un contrôle ActiveX nommé ADO (ActiveX Data Object) qui assure la communication avec la base de données et des pages (fichiers) ASP contenant du code HTML et des scripts. Cette

architecture efficace ne présente q'un seul inconvénient : elle repose sur des architectures très propriétaires. Elle fonctionne avec le serveur IIS sous Windows NT et utilise le langage Vbscript.

I.2 - JSP : (Java)

Pour concurrencer les ASP de Microsoft, les autres éditeurs se sont réunies autour de technologie comparable qui repose sur Java à la place des contrôles AvtiveX. Avantage évident, elle fonctionne sur toute plate-forme disposant de la JVM (Java Virtual Machine). La connexion aux bases de données est assurée par le protocole JDBC (Java Data Base Connectivity). Pour développer des sites avec des outils Java, il faut passer par des outils comme Jbuilder de Inprise ou VisualAge d'IBM.

I.3 – PHP : (logiciel libre)

PHP (Personal Hypertext Preprocesseur) est à l'origine un langage de script. Son but est de rendre les sites dynamiques ave un minimum de lignes de code qui peuvent être insérées dans le code HTML. Il est repris par un groupe de développeurs pour devenir PHP3 puis PHP4, avec la récupération du concept des sessions ASP. Il s'agit d'un véritable langage de programmation du coté serveur, mélange de C, Perl et Java.

I.4 - Meta-HTML :

Meta-HTML est un système très puissant développé par Brian Fox de MIT. Il va nettement plus loin que PHP, en permettant de créer de nouvelles balises pour son code HTML. Il permet également de gérer la persistance des connections.

I.5 - ePerl

ePerl ressemble conceptuellement à PHP, à ceci près qu'il remplace son langage minable par le Perl. On bénéficie donc d'une syntaxe et sémantique bien connue, et surtout des très nombreux modules d'extension qui ont été écrits pour Perl. En le couplant avec le module mod_perl de Apache, c'est une technologie très performante

II – CHOIX TECHNIQUE : CGI

II.1 – Définition :

La Common Gateway Interface (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 (C, C++, …) ou interprété (Java, JavaScript, …).

II.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 indépendant 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.

II.3 – Installation :

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

II.4 - interface CGI :

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

II.4.1 – variables d'environnement :

II.4.1.1 – variables relatives à la requête :

Variable	Signification
CONTENT_LENGTH	Taille en octet des information jointes à la requête
CONTENT_TYPE	Type MIME des données envoyées.
QUERY_STRING	Chaîne de caractère contenant les paramètres joints à la requête en mode GET. Elle es vide on utilise POST.
REQUEST_METHOD	Contient la méthode utilisée (GET, POST, DELETE, …)

Table 7.1 : variables CGI relatives à la requête.

II.4.1.2 – variables relatives à la connexion :

Variable	Signification
HTTP_ACCEPT	Types MIMES supportés par le client.
HTTP_ACCEPT_LANGEGE	Langage utilisé par le client
HTTP_ACCEPT_ENCODING	Type d'encodage supporté par l 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 ressource 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 (sécurité)
REMOTE_PORT	Port HTTP utilisé par le client

Table 7.2 : variable CGI relatives la requête.

II.4.1.3 – variables relatives au serveur :

Variable	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 lequel le serveur a reçu la requête
SERVER_PROTOCOL	Version du protocole HTTP utilisé par le serveur
SERVER_SOFTWARE	Nom et version du logiciel serveur utilisé

Table 7.3 : variable CGI relatives la requête

II.4.2 – Entrées – Sortie :

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

- **STDOUT** : sorite 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.5 – comportement spéciaux de CGI :

La norme CGI utilise des mécanismes particuliers pour gérer les flux de données entre le client et le serveur.

- No-Parsed-Header : la réponse HTML du script CGI peut être renvoyée directement au client sans interférence de la part du serveur.
- Server-Push : un script CGI peut envoyer un document de manière répétitives au client
- No-Content : un script CGI peut indiquer qu'il n'y a rien de nouveau à afficher et cela préserve le document présent dans le navigateur au lieu de le remplacer par un autre vide.
- Cache-Control : le script peut indiquer explicitement au navigateur de stocker ou non le document qu'il envoie dans le cache pour une durée précise

III – LE LANGAGE PERL :

III.1 – Pour quoi Perl :

Perl est un langage optimisé pour extraire des informations de fichiers texte et générer des rapports basés sue 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 meilleurs 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 choisi ce langage pour ses qualités de performance dont essentiellement :

- Portabilité : Perl existe sur la plupart des plate-forme (UNIX, Windows, Mac, …).
- Economie : il est disponible gratuitement sur Internet et dans ses versions les plus récentes.
- Simplicité : quelque commandes en Perl permettent de faire ce qu'un programme de 500 lignes de 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 documentée. Le processeur XS facilite l'intégration de routines en C ou C++ dans une application Perl. Le chargement dynamique est supporté et Perl lui même peut être transformé en une librairie dynamique.

III.2 – Perl et CGI :

Perl est devenu le langage numéro un pour l'écriture de 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 JavaScript 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 "coontent-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" ;

III.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.

```
use DBI ;
use CGI ;
$req=new CGI ;
$cin=req->param('cin') ; #récupérer la valeur de CIN
#connexion à  la base de données
$connexion = DBI->connect("dbi : MySQL : université" , "admin" , "admin") ;
#préparation de la requête
$SqlReq=$connection->prepare("SELECT * FROM ETUDIANT WHERE CIN= ?") ;
#exécution de la requête
$result = $SqlReq->execute($cin) ;
#    opérations de formatage en HTML des données extraites
#fermeture de la sélection
$SqlReq->finish ;
#déconnexion de la base de données
$connection->disconnect ;
```

Ce script permet l'identification d'un étudiant à partir du numéro sa carte d'identité nationale envoyé dans le formulaire d'identification et de lui retourner des informations extraites de la base de données UNIVERSITE.

IV – DEBOGUER LES SCRIPTS :

En cas d'erreurs des scripts , la première chose à vérifier est que l'exécution manuelle du programme ne produit pas d'erreurs. on se palace dans le répertoire cgi-bin/ et on lance le script par ./<nom du script>.pl, si l'erreur ne venait pas de là, on vérifie bien que le programme soit exécutable et lisible par tous en changeant ses doits d'accès par chmod 755 <nom du script>.pl. Si malgré ces points il se produit toujours des problèmes, on vérifie que le content-type est bien suivie d'une ligne blanche content-type : text/html \n\n car si non le navigateur ne peut pas lire la réponse.

V – ASPECT SECURITE :

V.1 – Aperçu :

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 (login nobody) il est possible d'avoir accès à des fichiers sensibles comme le célèbre /etc/passwd contenant les mots de passe cryptés des utilisateurs.

V.2 –comment se protéger :

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

- Les données entrées de l'utilisateurs 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 le serveur.

- Les script doivent être limités au strict nécessaire en matière de puissance. Dans le cadre des serveurs mulit-utilisateurs, les script doivent s'exécuter avec le minimum de privilèges système et jamais avec les privilèges du super user root.

- Les scripts générés dynamiquement doivent être proscris. L'utilisation de commandes comme exec, system(), eval() et popen() augmente les possibilités offertes à un utilisateur malin d'envoyer des commandes pouvant faire des ravages dans le système. En règles générales il vaut mieux éviter de faire des appels au Shell UNIX dans un script CGI. L'interpréteur Perl offre des alternatives plus sûres comme sa fonction exec().

CONCLUSION :

La mise en place des différents modules CGI du projet nous ont permis de maîtriser bon nombre de techniques et outils de développement Internet. Mais il reste de réaliser un module très important celui du paiement électronique qui permet de communiquer avec la passerelle de paiement "e-tijara" : c'est l'objectif du chapitre suivant.

Chapitre 8 Paiement électronique : E-Dinars

INTRODUCTION :

Le paiement sur Internet n'est plus un sujet nouveau dans une époque où les chiffres d'affaire des transaction sur le Web se valent par des Milliards de $. En Tunisie, la mise en place du système E-Dinars reflète le dynamisme et l'interaction entre l'économie d'une part et le secteur des technologies de communications d'autres part : cette fusion qui a donné naissance au commerce électronique. En dépit de la diversité des moyens de paiement numérique, il existe partout un certain nombre de points qui sont communs et qui se rapportent u problème de la sécurité des transactions. Ce projet n'est en fait qu'une utilisation du E-Dinars comme moyen de paiement des frais d'inscription, mais il est aussi l'occasion de voir même globalement les systèmes de sécurité et finalement implémenter un protocole de sécurité sur le serveur du site.

I - PAIEMENT SUR INTERNET : LES ENJEUX

I.1 - Argent électronique :

I.1.1 – Principe :

L'argent électronique se présente comme des valeurs validées par une banque et utilisées sous forme de pièces portant un N° de série qui sert d'identifiant unique. Ce N) ainsi que d'autres informations seront véhiculées sur le réseau de façon sécurisé à chaque opération de paiement. L'identification de la pièce se fait par consultation de la base de données de la banque. Généralement les transaction effectuées par ces carte ou pièces correspondent à des flux monétaires entre des institutions financières.

I.1.2 – Caractéristiques :

- Simplicité : bien adapté aux transactions en ligne et régulation de la circulation de l'argent par des logiciels appropriés.
- Anonymat : il n'y a pas de relation nominative entre les pièces et leur détenteur et donc personne ne sait l'identité du consommateur.

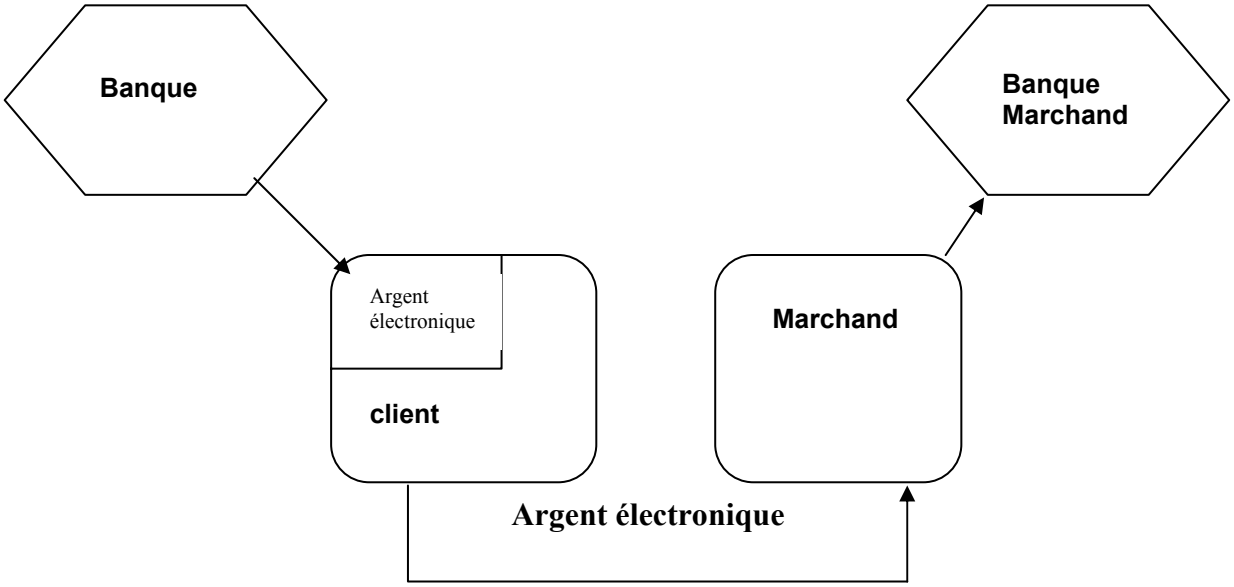


Figure 8.1 : paiement par argent électronique.

Cependant, certain trouvent l'argent électronique source de menace à grande échelle vu qu'elle devient vulnérable aux tentatives de fraudes bien menées de la part des experts. De plus, la législation n'est pas encore solidement définie sur ce point.

I.1.3 – Exemples :

DigiCash (USA), NetCash(USA), Mondex(UK), …

I.2 - Le E-Dinar :

Ce moyen de paiement est le fruit de l'engagement de l'économie tunisienne dans le monde du e-commerce. C'est un moyen pour des paiement sur des site marchands tunisiens présentant des services et des biens en ligne. Il est notamment utilisé pour différents types de paiement comme le téléphone, le frais d'inscription, …

Figure 8.2 : carte E Dinar.

Il s'agit d'une monnaie électronique simple, entièrement sécurisée, permettant à son porteur d'effectuer des micro-paiements sur Internet où et quand il le désire, jour et nuit, 7 jours sur 7.

II - TRANSACTION ET SECURITE :

La nécessité de la sécurité n'est en fait qu'un besoin à satisfaire pour gagner la confiance des visiteurs de notre site surtout s'il s'agit de transactions monétaires. Pour ce là certaines exigences en matière de sécurité doivent être satisfaites. On parle alors de services de sécurité.

II.1 - Services de sécurité :

Ces services[9] sont ce que l'on souhaite d'un système de communication sécurisé.

II.1.1 – Authentification :

comment on est sure que le message que l'en reçoit a réellement été envoyé par la personne qui prétend le faire ? pour cela, deux solutions existent : l'identification du correspondant dès sa connexion au réseau par des procédures informatiques ou bien l'intervention d'organisme tierce qui gère des signatures et certificats électroniques.

II.1.2 – Confidentialité :

Il s'agit de s'assurer que seules les personnes autorisée à lire les données peuvent le faire. Pour remédier à ce problème, on utilise la cryptographie qui consiste à chiffrer le message envoyé par des algorithmes adéquats

II.1.3 – Intégrité : .

L'intégrité des données transmises sur le réseau est importantes surtout en cas de données confidentielle pour garantir la légitimité de l'information.

II.1.4 – Non répudiation :

C'est le non désaveu et s'applique à tout opération dont on souhaite prouver qu'elle a bien été ordonné par une personne réelle.

[9] Norme ISO 7489

II.2 - La cryptographie :

La cryptographie, science qui a pour but de protéger le caractère confidentiel d'une information donnée, existe depuis des milliers d'années. Les méthodes cryptographiques modernes permettent le chiffrement, le déchiffrement et la signature numérique. Le chiffrement garantit la confidentialité. Autrement dit, il protège l'information contre toute divulgation non autorisée ou toute visualisation par le brouillage mathématique du texte original. Il existe principalement deux méthodes cryptographiques. Dans le cas de la cryptographie à clé secrète, la même clé (ou une copie de cette clé) est utilisée pour chiffrer et déchiffrer les données.

II.2.1 – Cryptographie à clé privée :

La cryptographie à clé secrète peut être utilisée pour chiffrer des données, pour les stocker ensuite sur un support électronique (disquette ou disque dur) ou les transmettre à un proche associé. Toutefois, cette méthode est fort limitée en soi, car elle ne convient pas à la diffusion générale sur des réseaux publics entre utilisateurs qui ne se connaissent pas. Dans le cas de la cryptographie à clé secrète, les deux parties doivent au préalable se communiquer la clé unique qui sera utilisée aux fins du chiffrement et du déchiffrement. Si l'on a recours au chiffrement en raison de l'insécurité de la voie de communication (p. ex., un réseau informatique), il est évident qu'il ne faut pas transmettre la clé secrète par la même voie, car n'importe qui pourrait la copier et déchiffrer toutes les données. On reconnaît généralement que les principaux problèmes que rencontre la cryptographie à clé secrète sur les réseaux ouverts ont trait à la distribution des clés et à la variabilité dimensionnelle (la variabilité dimensionnelle recouvre non seulement la notion d'accroissement du nombre d'utilisateurs, mais aussi la notion selon laquelle les réseaux ouverts comprennent des entités de taille différente, allant des particuliers aux multinationales, ainsi que des transactions dont le volume et la valeur varient).

II.2.2 – Cryptographie à clé publique:

La cryptographie à clé publique offre cependant une solution à ces deux problèmes puisqu'elle prévoit l'utilisation d'une paire de clés différentes, quoique connexes. Chaque utilisateur détient une clé privée et une clé publique. La clé privée demeure confidentielle et n'est connue que de l'utilisateur; l'autre clé peut être rendue publique et être transmise à chaque correspondant par l'entremise du réseau ou mieux encore, publiée dans un annuaire sûr, qui

est presque l'équivalent électronique d'un annuaire de téléphone. Pour utiliser ce système, l'émetteur chiffrerait un message à l'aide de la clé publique du destinataire, qui pourrait le déchiffrer uniquement à l'aide de sa clé privée. La cryptographie à clé publique permet donc la transmission de données en toute sécurité sur des réseaux ouverts, comme Internet, sans qu'il soit nécessaire d'échanger une clé secrète au préalable. Les parties qui ne se connaissent pas peuvent ainsi échanger et authentifier des informations et mener des affaires en toute sécurité.

II.3 – Certification électronique:

II.3.1 – Autorité de certification :

L'autorité de certification est l'organisme qui délivre les certificats numériques. Avant de fournir un certificat, l'autorité de certification doit s'assurer de l'identité du propriétaire de la clé privé. Ceci est réalisé en vérifiant les informations fournies lors de la demande du certificat. Une autorité de certification peut certifier une autre. En effet, les autorités de certification sont parfois organisées selon un modèle hiérarchique. Quand un utilisateur veut examiner un certificat pour s'assurer de l'identité de la personne avec qui il communique , il doit avant tout vérifier si l'autorité de certification qui a délivré le certificat est digne de confiance ou non. Ainsi on doit monter dans l'hiérarchie jusqu'à atteindre une autorité dans laquelle on a confiance. Ceci pose un autre problème : qui certifie la racine de la hiérarchie ? La réponse est simple , c'est la racine elle même qui s'auto-certifie. On doit donc donner confiance absolu à la racine des autorités de certification

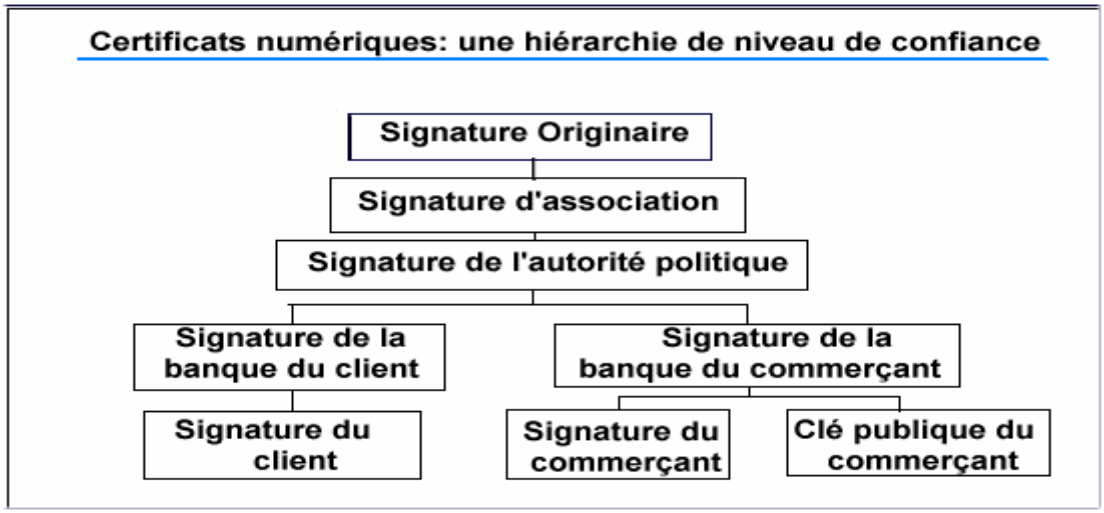


Figure 8.3 : Hiérarchie des autorités de certifications

Dans les navigateurs, il existe quelques autorités de certification pré-configurées. Quelques compagnies, comme Verisign et AT&T Certificate Services sont connues comme des autorités de certification. Elles fournissent les services suivants : accepter les demandes de certificats, vérifier les demandes de certificats et générer des certificats.

II.3.2 – Certificat numérique :

Un certificat est une "carte d'identité numérique" possédant une durée de vie limitée et attestant qu'une clé publique appartient à une entité particulière. Plus précisément, le rôle du certificat est d'associer de façon non équivoque une clé publique avec une entité. Un certificat doit être signé par une entité qui assure la validité du certificat. Cette entité possède elle-même un certificat signé par une autorité supérieure. Les autorités de certification (CA) sont organisées en structure d'arbre. Au sommet de la hiérarchie se trouve le *root* qui signe lui-même son certificat. Un certificat X.509 est principalement composé des champs suivants : une suite d'informations non chiffrées contenant entre autres le nom et la clé publique de l'entité possédant le certificat, le nom de l'algorithme de signature choisi, l'émetteur du certificat (l'autorité qui a signe le certificat), la clé publique du sujet, l'algorithme au travers duquel elle peut être utilisée et la signature du CA (Certificate Authority)

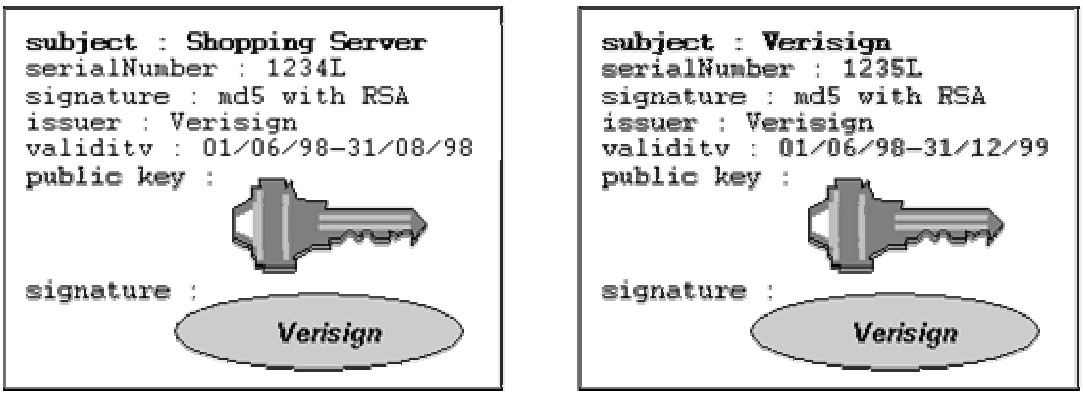


Figure 8.4 : Certificat signé par le CA Verisign, et certificat de Verisign.

II.4 – Le protocole SSL :

Le protocole SSL (Secure Socker Layer) développé par Netscape, est un projet de standard de l'IETF destiné à sécuriser les voies de communication entre les consommateurs Web et les serveurs. Le protocole SSL met en œuvre une communication sécurisée au niveau de la

couche Transport. Il utilise : un algorithme a clé publique (RSA) pour l'authentification et l'échange de clé secrète, un algorithme a clé secrète (DES, RC4) pour le chiffrement des données et un algorithme de hachage non réversible (SHA, MD5) pour le calcul des résumes de messages. SSL est implémenté au-dessus de la couche TCP/IP et en dessous des protocoles applicatifs tel que HTTP ou IMAP , il permet à un serveur SSL d'identifier un consommateur et vice versa, il permet aussi d'établir une connexion sécurisée entre deux machines pour un transfert de données cryptées.

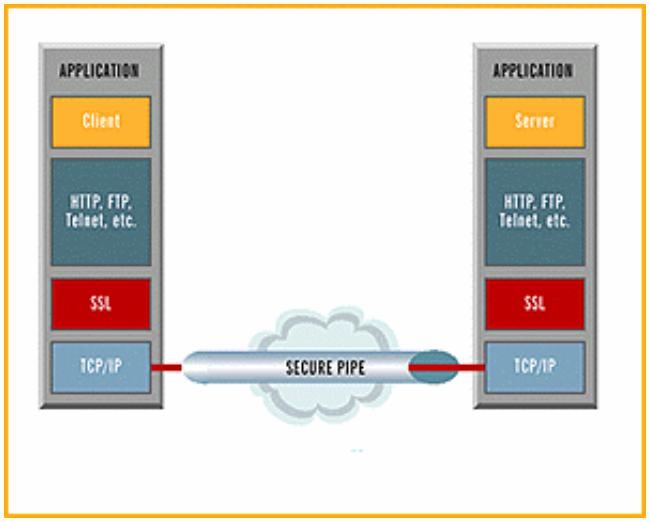


Figure 8.5 : Architecture du protocole SSL

III - TECHNIQUE DE SECURISATION DU SITE :

Cette partie résume l'ensemble des opérations qui ont été effectuées sur le serveur WWW du projet pour mettre en œuvre SSL dans notre serveur Apache et pour créer une autorité de certification. Nous n'avons utilisé que des logiciels libres et leurs documentations. Ces logiciels constituent les briques de bases de la solution.

III.1 – Récupération et installation des logiciels :

L'installation du serveur Web sécurisé et de l'autorité de certification nécessite les logiciels Apache, OpenSSL et le module mod_ssl d'Apache. Pour chaque élément, le premier lien pointe vers le répertoire de distribution de cet élément et le deuxième pointe vers la version que nous avons utilisée.

III.2 – Implémentation du certificat SSL serveur :

Pour activer la sécurité SSL (pour faire du commerce électronique par exemple) il est indispensable d'avoir un certificat SSL serveur. Ce document détaille l'obtention de ce certificat. Vous trouverez ici une fiche récapitulative à imprimer pour vous accompagner dans votre demande.

III.2.1 – Serveur Web et SSL :

Avant de chercher à obtenir un certificat pour un serveur, diverses vérifications sont nécessaires. Tout d'abord on vérifie que votre serveur supporte SSL De plus, avec le 128-bit disponible sans contrainte majeure, on vérifie que le serveur supporte le 128-bit et si non, une mise à jour doit être faite avant d'aller plus loin. Apache et ses dérivés OpenSSL supportent nativement le 128-bit, IIS est facilement "upgradable"

III.2.2 – Génération de la demande de certificat :

Après avoir installé le serveur on doit utiliser une fonction de ce serveur pour générer une demande de certificat (CSR, Certificate Signing Request). Il est recommandé de générer une clef publique de 1024-bit si possible (c'est le cas des serveurs 128-bit). Lors de la demande de certificat on va générer une clé privée. Dès que cette clé est générée, on en une copie de sauvegarde pour la protéger très sérieusement. Vérifiez exactement comment sauvegarder cette clef privée avec le fournisseur de votre logiciel serveur. Lors de la génération de CSR, un certain nombre de champs vous seront proposés pour y saisir les informations. Il est vivement recommandé d'avoir le ou les documents spécifiés sous la main pour bien remplir les champs. Toute erreur dans la saisie d'un champ entraîne forcement du retard!

- Common name / domain name / nom du serveur : ici il faut indiquer le nom de votre serveur SSL, par exemple www.universiteOnLine.tn. Pas d'adresse IP.
- Country / Pays : TN, FR, US, etc.
- State / Département :
- City / Ville
- Organisation / Organisation :
- Organisational unit / Division / Branche : s'il y a lieu, préciser la division de votre société qui possède ce serveur (texte libre)

L'utilitaire "openssl" utilisée pour générer la clé et le CSR est fournie avec le logiciel libre Open SSL et est généralement placée dans /usr/local/ssl/bin. Elle est utilisée pour générer une clé privée de 1024 bits, l'encrypter avec le triple-DES cipher et finalement créer le CSR

III.2.2.1 – Génération de la clé privée :

```
#cd /usr/local/ssl/private/
#openssl genrsa –des3 –rand /kernel:/bin/bash:/usr/bin/perl:/home/httpd:/home/me/img.gif
> http://www.universitOnLine.tn.key
```

III.2.2.2 – Génération du CSR :

```
# cd /usr/local/ssl/certs
# openssl req  -new  -key  ../private/www.unievrsiteOnLine.tn.key > www.universiteOnLine.tn.csr/
# openssl req -x509 –key ../private/www.universiteOnLine.tn.key -in www.universiteOnLine.tn.csr >
www.universiteOnLine.tn.crt/
```

III.2.3 – Déposer la demande de certificat :

Elle s'effectue depuis le site du CA par mécanisme de remplissage de formulaire par les données comme la clé publique. Le CSR devra être copié dans un formulaire

```
-----BEGIN CERTIFICATE REQUEST-----
MIIBPTCB6AIBADCBhDELMAkGA1UEBhMCWkExFTATBgNVBAgTDFdlc3Rlcm4gQ2FwZTESMBAG
A1UEBxMJQ2FwZSBUb3duMRQwEgYDVQQKEwtPcHBvcnR1bml0aTEYMBYGA1UECxMPT25saW
5lIFNlcnZpY2VzMRowGAYDVQQDExF3d3cuZm9yd2FyZC5jby56YTBaMA0GCSqGSIb3DQEBAQUA
A0kAMEYCQQDT5oxxeBWu5WLHD/G4BJ+PobiC9d7S6pDvAjuyC+dPAnL0d91tXdm2j190D1kgDoS
p5ZyGSgwJh2V7diuuPlHDAgEDoAAwDQYJKoZIhvcNAQEEBQADQQBf8ZHIu4H8ik2vZQngXh8v+iG
nAXD1AvUjuDPCWzFu pReiq7UR8Z0wiJBeaqiuvTDnTFMz6oCq6htdH7/tvKhh -----END
CERTIFICATE REQUEST-----
```

III.2.4 – Envoyer les justificatifs :

En plus du contrat à valider en ligne il sera demandé de fournir au plus vite dès que la soit demande déposée : une preuve de droit d'utilisation du nom et une preuve de profession du nom de domaine

III.2.5 – Installation du certificat :

Une fois le certificat fabriqué, le contact technique du dossier recevra une notification par email. Depuis une page Web sécurisée on fera un copier/coller du certificat vers un fichier sur le serveur qui sera utilisé pour installer le certificat. Le fichier www.universiteOnLine.tn.crt n'est autre que notre certificat local utilisé temporairement. Dès la réception du certificat, on doit faire une mise à jour de la configuration de notre serveur ApacheSSL et on installe le certificat dans /usr/local/ssl/certs/www.universiteOnLine.tn.crt

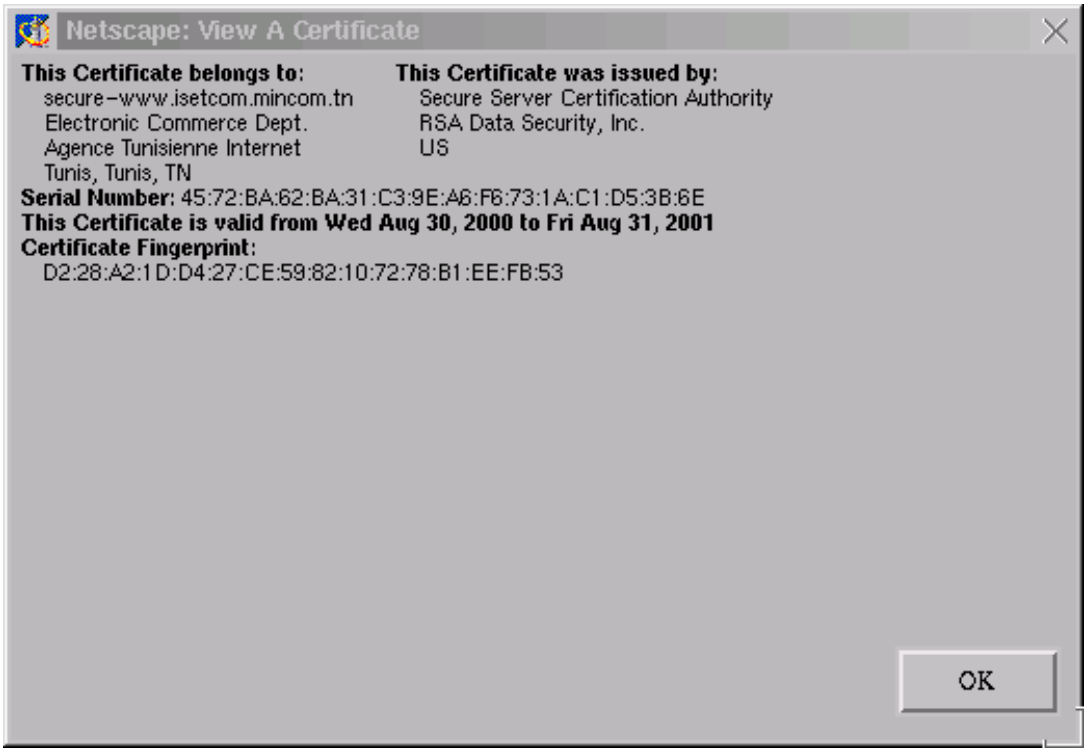


Figure 8.6 : Certificat du serveur secure-www.isetcom.mincom.tn

CONCLUSION :

Avec l'installation du système de sécurisation sur notre serveur Web, on peut normalement commencer l'exploitation du produit. L'utilisation des logiciels libres comme Open SSL ou ApacheSSL nous a permis de profiter du meilleure des technologies de sécurisation des serveurs Web. La charge est maintenant du coté de l'administrateur du site qui doit suivre le serveur en permanence pour détecter toute anomalie de fonctionnement et prendre les décisions nécessaires. Il est aussi amené à suivre de près

Conclusion générale

Ce projet est composé sous forme de modules intégrables offrant divers services et interfaces de gestion et de maintenance relatives aux divers directions appropriées. En effet, vu son importance dans le déploiement de l'économie immatérielle qui à son tour facilite la tâche soit aux responsables et dirigeants soit aux membres de l'administration en leur fournissant une interface ergonomique, conviviale et plus simple à utiliser basée sur le Web et dont le nouvelles technologies règnent à son égard. Une structure chevaleresque au niveau de la conception des divers besoins en passant de la phase d'écoute et d'échantillonnage, via un langage ambiguë, vers une modélisation matérielle reflétant les besoins sous une forme conceptuelle concise, efficace, cohérente et non ambiguë : c'est l'art de l'ingénierie.

Cependant, sur le plan des fonctionnalités, ce projet reste ouvert à des évolutions sous forme de modules intégrables de manière indépendante et dont on cite premièrement celui permettant aux étudiants, une fois l'inscription achevée, de créer leurs comptes personnels sur le serveur Web et bénéficier de services multiples (Free Mail, Personnel Web Pages, Bibliothèque Virtuelle, Forum de Discussion, Chat, …) et deuxièmement le module qui permet d'accepter autres moyens de paiement que le E-Dinars (Inscription depuis l'étranger).

Parmi les autres modules non achevés et qui vont ete inclus au sein de ce dernier, nous citions celle de la coopération avec le ministère de l'enseignement supérieur afin que l'étudiant bénéficie des services de l'Office des Œuvres Universitaires, des restaurants et foyers universitaires. Le cas se présente même avec le ministère de transport où l'étudiant pourra régler ses obligations à distances.

Comme dernier mot, j'espère que j'étais à la hauteur de la confiance qui est mise en ma personne et surtout le fait d'avoir côtoyer des personne expérimentés qui m'a permis de joindre le savoir au savoir faire et enrichir d'autant mes connaissance soit au niveau professionnel que technique.

Bibliographie

- Stéphane Philippe **Introduction aux Bases de Données** Editest 1986.
- Ryan Bernard **L'Intranet En Entreprise** Sybex 1997.
- George Gardarin & Paul Valduriez **Bases De Données Relationnelles : Analyse Et Conception** Eyrolles 1988
- Guillaume Benci & Colette Roland **Bases De Données : Conception Canonique Et Réalisation Extensible** SCM 1979.
- Nancy Yeager & Robert E. McGrath **Technologie Des Serveurs Web** Thomson Publishing 1997
- Gerorge Gardarin & Olivier Gardarin **Le Client Serveur** Eyrolles 1996.
- Marc Grégory **Le Guide Du Programmeur JavaScript** Eyrolles 1998.
- Randal L. Schwartz & Tom Christiansen **Introduction à Perl** O'Reilly 1998
- David Till **Perl 5 En 21 Jours** SAMS 1996.
- Paul Gaborit **Documentation Perl** Ecole des Mines d'Albi 2000

Magazines

- Langages & Systèmes **Quels Outils Pour Le Web ?** IDG Main - Juin 2000
- Langages & Systèmes **eXtreme programming** IDG Octobre - Novembre 2000

Sites Web

- http://www.iup.univ-avigran.fr/~Cours/
- http://www.univ-st-etienne.fr/monnet/formation/
- http://www.essi.fr/~buffa/cours/internet99/PROJETS/
- http://directory.google.fr/
- http://dev.nexen..net/docs/mysql/

Etude, conception et réalisation d'une application d'inscription universitaire en ligne

Réalisé par : Khaled Lâabidi
TS - Télécommunication
Promotion Février 2001

Résumé :

Les dernières évolutions des services en ligne déployés sur Internet ont engagé les entreprises et les organisations diverses à suivre de près ces évolutions pour en tirer le meilleur profit. En Tunisie, après l'émergence du commerce électronique d'une part et la croissance de la culture Internet d'autre part, le secteur universitaire vient d'affranchir une nouvelle étape vers le campus universitaire : l'inscription universitaire en ligne. Ce travail consiste en une étude, conception et développement de cette application tant sur le plan administrative (gestion de la scolarité) qu'économique par l'utilisation du E-Dinar comme moyen de paiement électronique.

Mots clés :

Inscription en ligne, Conception, Base de Données, Développement, CGI, E-Dinars .

2000 - 2001