République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mouloud MAMMERI de Tizi-Ouzou
Faculté de Génie Électrique et Informatique
Département Informatique

Mémoire

De fin de cycle en

Vue de l'Obtention du Diplôme de MASTER en Informatique .

OPTION : Conduite de Projet Informatique.

Thème

Sécurisation des requêtes dans les bases de données Réparties.

Proposé et dirigé par :

M[r:] OUAMRANE MOHAMED.

Réalisé par :

M[elle] : SEGUEL Chahrazed.

Promotion 2013/2014

Remerciements

Je tiens tout d'abord à remercier Dieu le tout puissant qui m'a donné la force et la patience d'accomplir ce modeste travail.

Je tiens vivement à remercier mon promoteur ***Monsieur OUAMRANE MOHAMMED*** *pour avoir accepté d'encadrer ce travail et aussi pour tous les conseils prodigués pour le travail réalisé.*

Aussi je tiens à lui reconnaître le temps précieux qu'il m'a consacré.

Je voudrais également exprimer mes sincères remerciements aux membres du jury qui m'ont fait l'honneur d'accepter de juger mon travail.

Mes remerciements s'adressent à tous ceux qui m'ont aidée de près ou de loin dans mon travail.

Dédicaces

A mes chers parents qui ont toujours été là pour moi, qui m'ont donné un magnifique modèle de labeur et de persévérance, UN GRAND MERCI pour vos attentions, sacrifices et soutien tout au long de ma vie.

- A mon grand frère Madjid, sa femme Nouara ainsi que leurs fils Rayane et Samy.
- A mon frère kamel , sa femme kissa et leur petit fils Anis.
- A mes adorables fréres Abdllah et Mokrane .
- A mes sœurs de cœur: Dawia, Fazia, Thinhinane, Lynda, Nassima, Hakima, Baya et Ghania.
- A tous mes amis (es), en particulier

Hocine, Ahmed, Samir, Larbi, Ahcen, Slim, Ouhiba, Djamila, Fatima, Karima, Farida, Fifi, monia,…

- A toute ma famille.
- ET toute la promotion 2014.

Chahrazed.

Sommaire

Liste des figures

Sommaire

Chapitre II: Sécurité des bases de données réparties

Chapitre III : Elaboration d'un exemple d'une BDR

Introduction générale

Introduction générale

Dans la société actuelle, nous avons de plus en plus affaire à l'informatique. L'administration, les entreprises, les hôpitaux, les citoyens, tous les acteurs de la société utilisent aujourd'hui massivement les systèmes informatiques pour gérer leurs activités, leurs données, leurs employés, leurs clients, et même leur argent. Comme de nouveaux besoins sont apparus avec toute organisation automatisée souhaite stocker et échanger ses informations qui sont géographiquement éloignées, ce qui rend la tâche de la collecte et de traitement d'une grande quantité d'informations dispersées très délicate, de ce fait, l'amélioration des systèmes d'informations est devenue une priorité pour les gérants des entreprises.

La solution qui s'impose est de distribuer les données et les organiser dans des bases de données sur différents sites de stockage. L'ensemble de ces sites constitue un système de bases de données réparties offrant la possibilité aux utilisateurs de manipuler les différentes bases via un réseau de manière transparente, comme dans une base de données globale. Les bases de données répartie offrent beaucoup d'avantages : un gain non négligeable en puissance de calcul, en capacité de stockage et en performance, grâce à un traitement parallèle, et une grande extensibilité, car ils peuvent croître progressivement selon le besoin. Cependant, leur flexibilité et évolutivité soulèvent des problèmes supplémentaires, dont notamment la sécurité ,Étant donné qu'une information dans un base de donnée répartie peut être manipulée, échangée, dupliquée et modifiée par des entités qui évoluent dans des environnements hétérogènes et parfois peu fiables, il est impératif de trouver un moyen de contrôler leur utilisation de manière transparente et surtout peu importune pour l'utilisateur.

Dans ce mémoire nous allons, dans un premier temps, présenter d'une façon générale les différents aspects liés à la base de données réparties. Nous verrons ensuite dans le deuxième chapitre la sécurité des bases de données réparties. Nous étudierons dans le troisième chapitre un exemple de base de données réparties pour la faculté génie électrique et informatique de l'université Mouloud Mammeri Tizi-Ouzou et dans le dernier chapitre nous verrons comment mettre en place une politique de sécurité pour les requêtes qui se déroulent dans cette base de données réparties.

Chapitre I

Base de données réparties.

Introduction :

L'évolution des techniques informatiques depuis les vingt dernières années a permis d'adapter les outils informatiques à l'organisation des entreprises Vues , le grand volume de données manipulées par ces dernières, la puissance des micro-ordinateurs ,les performances des réseaux et la baisse considérable des coûts du matériel informatique ont permis l'apparition d'une nouvelle approche afin de remédier aux désagréments causés par la centralisation des données ,et ce en répartissant les ressources informatiques tout en préservant leur cohérence .

Les bases de données réparties sont un moyen très efficace pour pallier aux problèmes engendrés par l'approche centralisées, mais n'en demeure pas moins sans failles.

I. Définition d'une base de données:

Une base de données, usuellement abrégée en BD ou BDD, est un ensemble structuré et organisé d'informations. Les informations sont placées dans des fichiers, et organisées de manière à pouvoir être facilement triées, classées et modifiées par le biais d'un logiciel spécialisé appelé système de gestion de base de données (SGBD).

Une base de données se traduit physiquement par un ensemble de fichiers présent sur une mémoire de masse (bien souvent un disque). La manière dont les informations sont organisées doit permettre de retrouver très rapidement n'importe quelle information sur un lot qui en contient plusieurs millions.

I.1 Définition d'un système de gestion de base de données *:*

Le SGBD est un ensemble de services (applications logicielles) permettant de gérer les bases de données, c'est-à-dire il permet l'accès aux données de façon simple, autorise un accès aux informations à de multiples utilisateurs et manipule les données présentés dans la base de données (insertion, suppression, modification) .On peut dire que le SGBD est une interface entre les utilisateurs et la mémoire de masse, il facilite le travail des utilisateurs en leurs donnant l'impression que l'information est organisée comme ils le souhaitent.

II. Définition d'une base de données répartie(BDR) :

Une base de données répartie *BDR* est une collection de base de données localisées sur différents sites, généralement distants, mises en relations les unes avec les autres à travers un réseau d'ordinateurs, perçues pour l'utilisateur comme une base de données unique. Elle permet de rassembler des données plus ou moins hétérogènes, disséminées dans un réseau sous forme d'une base de données globale, homogène et intégrée.

II.1 Définition d'un système de gestion de base de données répartie:

Système gérant une collection de BD logiquement reliées, réparties sur différents sites en fournissant un moyen d'accès rendant la distribution transparente.

Une base de données centralisée est gérée par un seul SGBD, est stockée dans sa totalité à un emplacement physique unique et ses divers traitements sont confiés à une seule et même unité de traitement. Par opposition, une base de données distribuée est gérée par plusieurs processeurs, sites ou SGBD.

Un SGBD réparti doit rendre la répartition des bases de données transparentes aux utilisateurs. La base de données étant répartie, il faut également répartir certaines fonctionnalités du SGBD. Le schéma d'un SGBD réparti est résumé dans la figure.

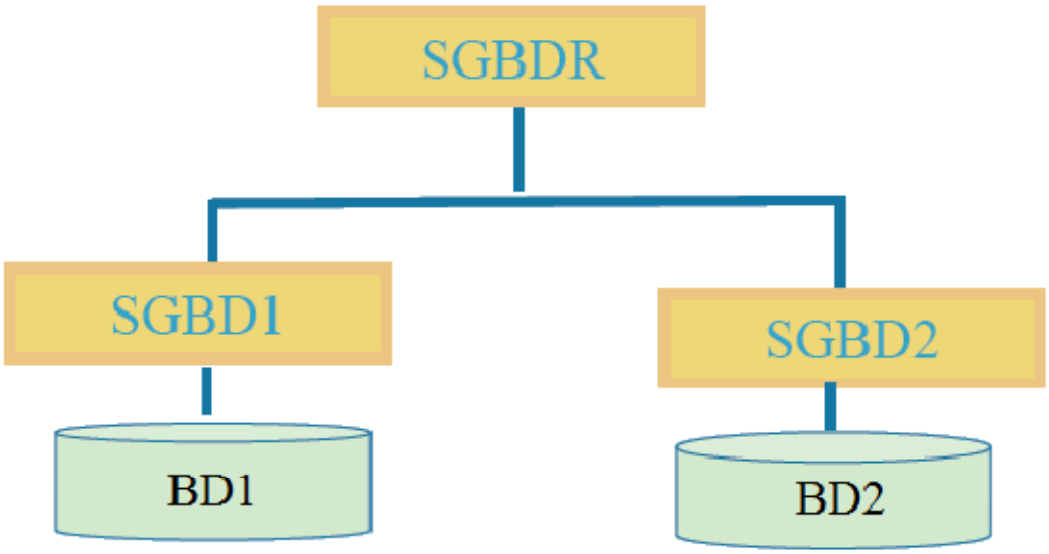


Figure I.1 : Schéma d'un SGBD

II.1.1 Mode de fonctionnement d'un SGBD répartie :

Le SGBD répartie reçoit des requêtes référençant des objets d'une base de données répartie. Il assure la décomposition des requêtes répartie en sous requêtes locales envoyées à chaque site.

La décomposition prend en compte les règles de localisation.

Dans le cas ou les bases de données sont hétérogènes, le SGBDR doit aussi assurer la traduction des requêtes exprimées dans un langage pivot (par exemple SQL) en requêtes compréhensibles par le SGBD local. [1]

II.2.2 concepts de bases :

Une base de données répartie est décrite par différents niveaux de schémas

- **Schéma local**

  Schéma décrivant les données d'une bdd locale gérée par le SGBD local. Lors de la construction de la base de données répartie, chaque base local rend visible une partie de la base aux sites clients.

- **Schéma global**

  Le schéma global permet de définir l'ensemble des types de données de la base .Il ignore les concepts d'implémentation. Dans une base de données répartie, le schéma global n'est pas forcément matérialisé. Chaque base locale implémente une partie.

- **Schéma exporté**

  Schéma décrivant les données exportées par un site vers les sites clients.

- **Schéma importé**

  Vue d'un site client, un schéma exporté par un serveur devient un schéma importé.

- **Vue intégrée**

  Schéma décrivant dans le modèle du SGBD distribué les données réparties accédé par une application.

III .Buts de la répartition des bases de données :

Les bases de données réparties ont une architecture plus adaptée à l'organisation des entreprises décentralisées.

- ✓ **Plus de fiabilité** : les bases de données réparties ont souvent des données répliquées. La panne d'un site n'est pas très importante pour l'utilisateur, qui s'adressera d'autre site.

- ✓ **Meilleures performances** : réduire le trafic sur le réseau est une possibilité d'accroître les performances. Le but de la répartition des données est de les rapprocher de l'endroit où elles sont accédées. Répartir une base de données sur plusieurs sites permet de répartir la charge sur les processeurs et sur les entrées/ sorties.

- ✓ **Faciliter l'accroissement:** l'accroissement se fait par l'ajout de machines sur le réseau.

III.1 Problèmes à surmonter :

- **Coût**

La distribution des données et des traitements entraîne des coûts supplémentaires en termes de communication, et en gestion des communications comme le hardware et software à installer afin de gérer les communications et la distribution.

La distribution est également coûteuse en matière du personnel utilisé car il faut payer les administrateurs de chaque site.

- **Distribution du contrôle**

La distribution du contrôle crée des problèmes de synchronisation et de coordination dans l'accès aux données. Dans une base de données répartie, on ne se soucie pas de la consistance et l'intégrité d'une seule base de données, mais de plusieurs copies de la base de données.

La gestion des copies doit assurer leur cohérence mutuelle, c'est-à-dire que toutes les copies de données soient identiques.

- **Sécurité**

Un des avantages évident des bases de données centralisées est sans contexte la sécurité apportée aux données, car elle peut facilement être contrôlée dans un site unique. Or, les

Bases de données réparties impliquent un réseau dont la sécurité est difficile à maintenir. La sécurité est donc un problème plus complexe dans le cas des bases de données réparties que dans le cas des bases de données centralisées.

- **Gestion distribuée des inter- blocages**

Le problème de l'inter-blocage ' *Deadlock* ' est le même que celui rencontré dans les systèmes d'exploitation. La compétition entre les utilisateurs pour accéder à une donnée peut entraîner des inter-blocages.

- **Bases de données hétérogènes**

Quand les bases de données sur différents sites ne sont pas homogènes en terme de modèle de données (*relationnel, objet, XML, . . .*), il devient nécessaire de fournir un mécanisme de translation entre les différentes bases de données, ce mécanisme de translation exige toujours une forme canonique pour faciliter la translation des données.

IV .Conception d'une base de données répartie :

La définition du schéma de répartition est la partie la plus délicate de la phase de conception d'une BDR, car il n'existe pas de méthode miracle pour trouver la solution optimale. L'administrateur doit donc prendre des décisions dont l'objectif est de minimiser le nombre de transferts entre sites, les temps de transfert, le volume de données transférées, les temps moyens de traitement des requêtes, et le nombre de copies de fragments, ... etc.[2]

IV.1 Méthodes de conception :

Deux approches fondamentales sont à l'origine de la conception des bases de données réparties : la conception descendante '*Top down design*' et la conception ascendante '*Bottom up design*'.

IV.1 .1 Conception ascendante :

Cette approche se base sur le fait que la répartition est déjà faite, mais il faut réussir à intégrer les différentes BDs existantes en une seule BD globale. En d'autre terme, les schémas conceptuels locaux existent et il faut réussir à les unifier dans un schéma conceptuel global.

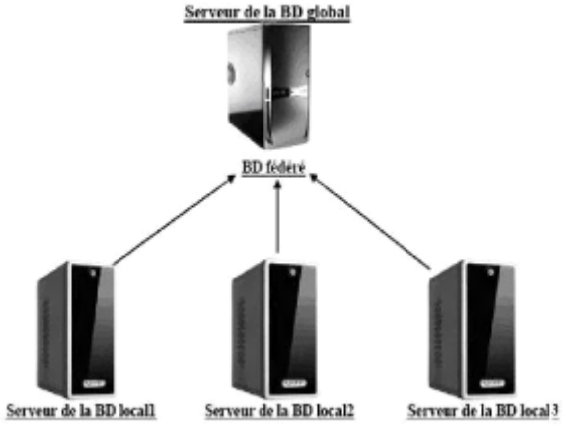


Figure I.2 : Architecture de la conception ascendante. [3]

IV.1 .2 Conception descendante

On commence par définir un schéma conceptuel global de la base de données répartie, puis on le distribue sur les différents sites en des schémas conceptuels locaux. La répartition se fait donc en deux étapes, en première étape la fragmentation et en deuxième étape l'allocation de ces fragments aux sites.

L'approche top down est intéressante quand on part du néant. Si les BDs existent déjà, la méthode bottom up est utilisée.

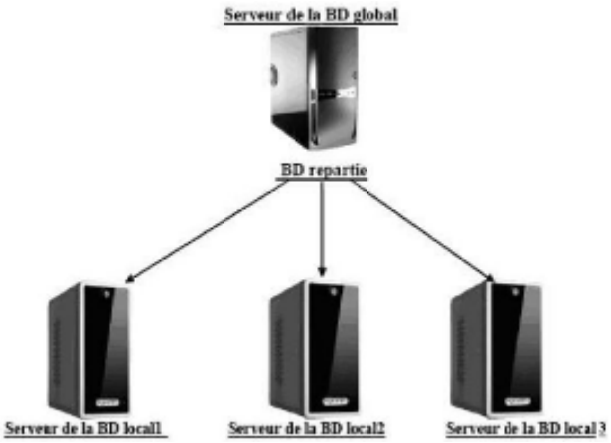


Figure I.3 : Architecture de la conception descendante. [3]

IV.2 La fragmentation :

IV.2.1 Définition :

La fragmentation est le processus de décomposition d'une base de données en un ensemble de sous bases de données. Cette décomposition doit être sans perte d'information. La fragmentation peut être coûteuse s'il existe des applications qui possèdent des besoins opposés. [4]

IV.2.2 Objectifs de la fragmentation :

Les applications ne travaillent que sur des sous-ensembles des relations. Une distribution complète des relations générerait soit beaucoup de trafic, soit une réplication des données avec tous les problèmes que cela occasionne : problèmes de mises à jour, problèmes de stockage. Il est donc préférable de mieux distribuer ces sous-ensembles.

L'utilisation de petits fragments permet de faire tourner plus de processus simultanément, ce qui entraîne une meilleure utilisation des capacités du réseau d'ordinateurs.

IV.2.3 Types de fragmentation :

IV.2.3.1 La fragmentation horizontale :

C'est un découpage d'une table en sous tables par utilisation de prédicats permettant de sélectionner les lignes appartenant à chaque fragment. L'opération de fragmentation est obtenue grâce à la sélection des tuples d'une table selon un ou des critères bien précis et la reconstitution de la relation initiale se fait grâce a l'union (U) des sous-relations. [4]

Exemple :

Table enseignants de la faculté				
Code_ensg	Nom enseignant	Prénom enseignant	Département	Grade
01	Nom-ensg 1	Prénom-ensg1	informatique	Professeur
02	Nom-ensg 2	Prénom-ensg2	informatique	M.A.C.A
03	Nom-ensg 3	Prénom-ensg3	automatique	M.C.C.A

Table enseignants de département informatique

Code_ensg	Nom enseignant	Prénom enseignant	Département	Grade
01	Nom-ensg 1	Prénom-ensg1	informatique	Professeur
02	Nom-ensg 2	Prénom-ensg2	informatique	M.A.C.A

Table1@site 1

Table enseignants de département automatique

Code_ensg	Nom enseignant	Prénom enseignant	Département	Grade
03	Nom-ensg 3	Prénom-ensg3	automatique	M.C.C.A

Table 1@site2

Figure I.4: Exemple de fragmentation horizontale.

Assemblage de fragmentation horizontale sous SQL : la reconstitution de la relation initiale se fait grâce à l'union (U) des sous-relations.la requête de la reconstruction est la suivante

```
CREATE VIEW V1
AS SELECT Table1.cle, Table1.attr1
FROM Table1@site1
UNION
SELECT Table2.cle, Table2.attr1
FROM Table2@site2
```

IV.2.3.2 La fragmentation verticale :

Elle est le découpage d'une table en sous tables par projection permettant de sélectionner les colonnes composant chaque fragment. La relation initiale doit pouvoir être recomposée par la jointure des fragments.

Table Etudiant

Code_etud	Faculté	Département	Formation	Spécialité
01	FGEI	Informatique	Licence	Informatique Général
02	FGEI	Informatique	Master I	CPI
03	FGEI	Automatique	Master II	CPI

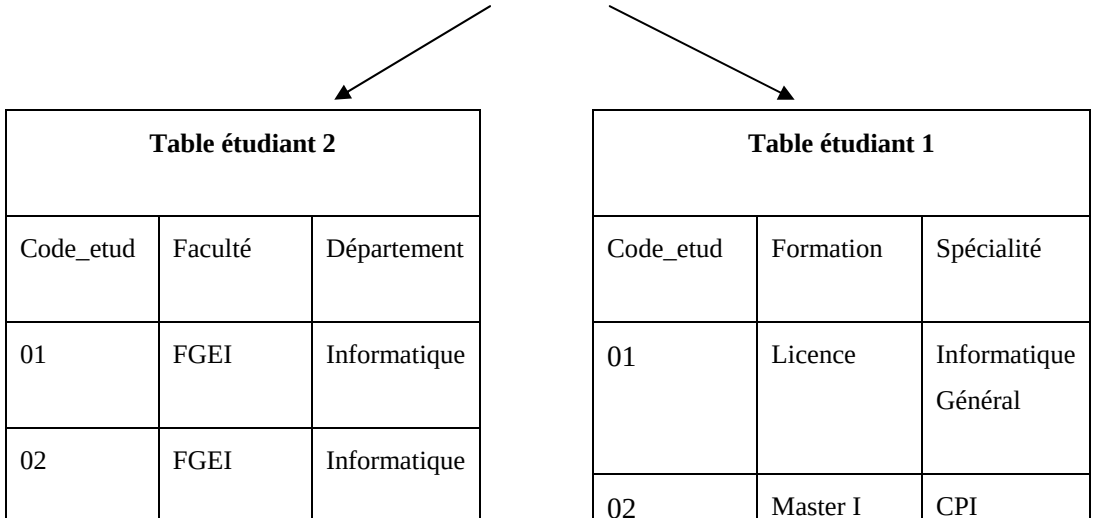


Table étudiant 2

Code_etud	Faculté	Département
01	FGEI	Informatique
02	FGEI	Informatique

Table étudiant 1

Code_etud	Formation	Spécialité
01	Licence	Informatique Général
02	Master I	CPI

Figure I.5 : Exemple de fragmentation verticale.

Assemblage : la reconstruction de la table initiale se fait par la jointure comme nous l'avons cité précédemment on exécutant des requêtes qui ressemblent à celle-ci :

```
CREATE VIEW V1
AS SELECT Table1.cle, Table1.attr1,
Table2.attr2
FROM Table1@site1, Table2@site2
WHERE Table1.cle=Table2.cle
```

IV.2.3.3 La fragmentation mixte :

Elle résulte de l'application successive d'opérations de fragmentation horizontale et verticale sur une relation globale.

IV.2.4 Les règles de la fragmentation :

Le problème qui se pose pour la fragmentation est comment définir un bon degré de fragmentation. Il existe trois règles pour la fragmentation :

- ***Complétude*** : pour toute donnée d'une relation globale R, il existe au moins un fragment Ri de la relation R qui possède cette donnée.
- ***Reconstruction*** : pour toute relation R décomposée en un ensemble de fragments Ri, il existe une opération de reconstruction à définir en fonction de la fragmentation. Pour les fragmentations horizontales, l'opération de reconstruction est une union. Pour les fragmentations verticales c'est la jointure.
- ***Disjonction*** : une donnée n'est présente que dans un seul fragment, sauf dans le cas de la fragmentation verticale pour la clé primaire qui doit être présente dans l'ensemble des fragments issus d'une relation.

V. Le schéma de répartition :

Pour fragmenter les requêtes, il est nécessaire de connaître les règles de localisation des données. Lors de l'exécution d'une requête, le SGBDR doit décomposer la requête globale en sous requêtes locales en utilisant le schéma de répartition.

- ***Architecture de schéma d'une base de données distribuée :***

Après la présentation des techniques de fragmentation et d'allocation, la figure suivante présente l'architecture d'une base de données distribuée :

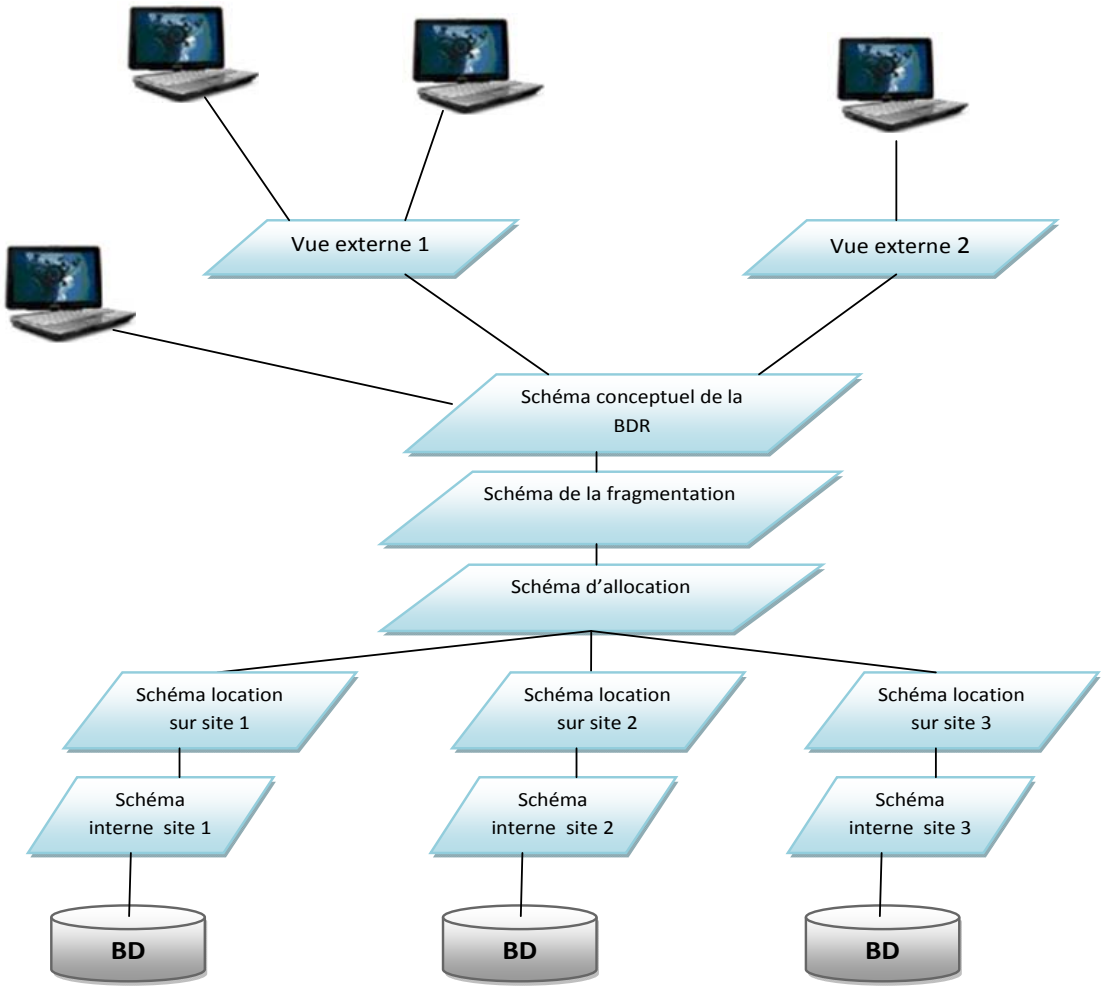


Figure I.6 : architecture de schéma d'une base de données distribuée [3].

V.1 .Techniques de répartition avancées :

V.1 .1 . Allocation avec duplication :

Cette technique consiste à dupliquer des parties de la base c'est-à-dire les fragments sont dupliqués sur un seul site, voir plusieurs sites selon les besoins. Cette approche est très intéressante car elle améliore considérablement les performances du système, étant donné que les fragments sont dupliqués un peu partout et que les accès aux données sont locaux, évitant ainsi la congestion du réseau et améliorant les temps de réponse. Le principal inconvénient de cette technique est la difficulté des mises à jour de tous les fragments dupliqués.

V.1 .2. Allocation dynamique :

Avec cette technique, l'allocation d'un fragment peut changer en cours d'utilisation de la BDR, c'est-à-dire qu'un fragment qui se trouve sur un site A à un instant T, peut être retrouvé sur un site B à un instant T+1. Cette technique est efficace mais exige le maintien du schéma d'allocation et des schémas locaux.

VI. La réplication :

La réplication consiste à copier les informations d'une base de données sur une autre. Elle peut être accompagnée d'une transformation des données sources, voir souvent d'une agrégation. Dans tous les cas, il s'agit d'une redondance d'information.

L'objectif principal de la réplication est de faciliter l'accès aux données en augmentant la disponibilité. Soit parce que les données sont copiées sur différents sites permettant de répartir les requêtes, soit parce qu'un site peut prendre la relève lorsque le serveur principal s'écroule. Une autre application tout aussi importante est l'amélioration des performances des requêtes sur les données locales, et ceci permet d'éviter les transferts de données et d'accroître la résistance aux pannes. [5]

VI. 1. Principe :

Le principe de la réplication, qui met en jeu au minimum deux SGBDs, est assez simple et se déroule en trois étapes :

- ✓ .La base maîtresse reçoit un ordre de mise à jour (INSERT, UPDATE ou DELETE).
- ✓ Les modifications faites sur les données sont détectées et stockées dans un fichier ou une file d'attente en vue de leur propagation
- ✓ Le processus de réplication prend en charge la propagation des modifications à faire sur une seconde base dite esclave. Il peut bien entendu y avoir plus d'une base esclave

VI.2. Les avantages de la réplication :

Les avantages de la réplication sont assez nombreux, selon le type on trouve :

- Allégement du trafic réseau en répartissant la charge sur divers sites. rapidité d'accès aux données.

- Amélioration des performances des requêtes.

- Résistance aux pannes par l'augmentation de la disponibilité des données.

VII. Architecture des bases de données réparties :

Autonomie :

L'autonomie se rapporte au degré avec lequel une des bases locales peut travailler indépendamment des autres. On peut distinguer trois types d'alternatives dans l'autonomie que peuvent avoir les bases locales :

L'intégration totale :

Une image unique de la base de données globale est offerte aux différents utilisateurs. D'où la BD est centralisée, le SGBD contrôle de bout en bout la requête d'un utilisateur même

si elle met en jeu différentes bases locales et donc différents SGBDs locaux. L'autonomie n'est donc pas bien importante.

La semi-autonomie :

Les SGBDs locaux peuvent opérer indépendamment mais ils participent à une collection de bases qui coopèrent afin de partager leurs données. L'isolation totale : une base locale ne connaît ni l'existence des autres bases ni la façon de se communiquer avec elles. Il ne peut donc pas y avoir du contrôle global quant à l'exécution d'une requête sur les différentes bases locales.

VII. 1.Relation entre machines :

Du point de vue organisationnel, nous distinguons deux types d'architectures :

VII. 1.1 Architecture Client / Serveur :

Dans cette architecture applicative, les programmes sont répartis en processus clients et serveurs qui communiquent à travers des requêtes et des réponses. Sur la machine cliente, les utilisateurs disposent d'une interface.

Sur les serveurs, la gestion des bases de données est effectuée (*analyse, optimisation des requêtes, et répartition*). On peut distinguer deux types de clients :

- *Client lourd* : l'utilisateur est obligé de se connecter explicitement à tous les serveurs dont il a besoin pour la requête qu'il veut formuler. [4]

Exemple :

Dans une application de gestion de la scolarité universitaire, si la BD est repartie par faculté alors la recherche d'un étudiant grâce à son matricule et sans connaître la faculté à laquelle il appartient est très délicate, car nous sommes obliger à interroger la BD de chaque faculté une par une jusqu'à trouver un résultat.

- *Client léger* : l'utilisateur ne se connecte qu'à la base de données via un unique serveur. Le SGBDR se charge alors de gérer les différentes connexions que nécessitera la requête de l'utilisateur. Donc, il offre plus de transparence.

Exemple : pour reprendre l'exemple précédent dans le cas d'un client léger, c'est le SGBDR qui se charge de trouver la faculté de l'étudiant en question, et de retourner un résultat.

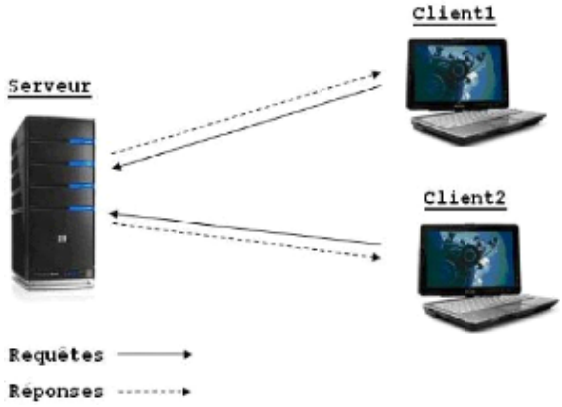


Figure I.7: Architecture Client / Serveur[4].

VII. 1 .2. Architecture Peer To Peer:

C'est un type de communication pour lequel toutes les machines ont une importance équivalente. Il n'y a pas de machine qui a une importance hiérarchique par rapport aux autres. Dite aussi, l'architecture totalement répartie.

Chacune de ces architectures possède des avantages et des inconvénients. Le Client / Serveur avec sa structure plus hiérarchique est très sensible aux problèmes de panne des serveurs, bloquant ainsi les clients. En revanche, la prise de décision des serveurs est rapide.

Pour l'architecture Peer-To-Peer, comme les machines sont strictement équivalentes, la panne d'une machine peut rarement rendre le système un peu lent. Mais cette architecture engendre énormément de communication pour toute décision.

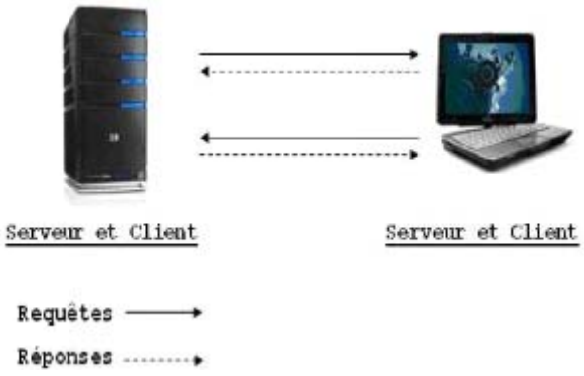


Figure I.8: Architecture Peer To Peer [4].

VII. Les requêtes réparties

VII.1. Définition :

Une requête répartie est une requête s'effectuant sur une base de données répartie. Comme une requête normale, elle se base sur les relations de la base et leurs champs, en utilisant l'algèbre relationnel. Mais elle doit tenir compte en plus de certains paramètres essentiels de fragmentation, de localisation afin d'optimiser le temps de réponse global de la requête.

VII.2.Principe de répartition :

Dans un environnement réparti, les requêtes formulées à un niveau global sont décomposées en sous-requêtes qui seront adressées aux sites locaux où elles seront exécutées. Les réponses locales sont ensuite regroupées pour élaborer la réponse globale qui sera retournée à l'utilisateur. La décomposition d'une requête globale en sous-requêtes locales s'appuie sur le schéma de fragmentation des données. La figure ci-dessous, illustre le plan d'exécution répartie d'une requête.

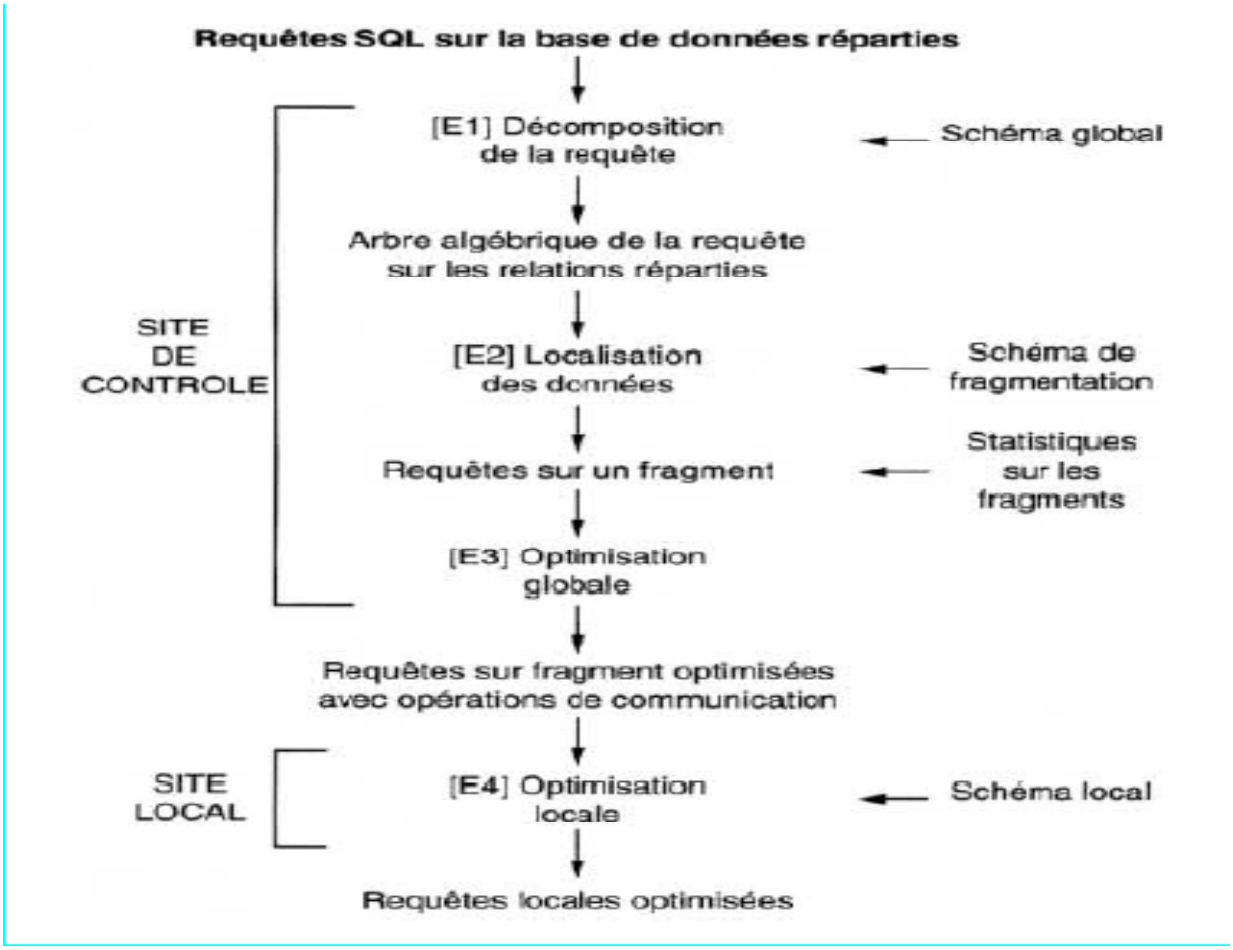


Figure I.9 : Schéma général de traitement d'une requête répartie [4].

Les phases de cette exécution répartie sont :

- L'étape [E1] a pour objectif l'élaboration d'un arbre algébrique optimisé d'évaluation de la requête. Cette étape s'appuie sur les traitements suivants :

1. normalisation de l'écriture de la requête,

2. analyse-vérification en vue d'éliminer les requêtes normalisées pour lesquelles on détecte des anomalies ou des incohérences.

3. élimination de redondances,

4. réécriture de la requête SQL en algèbre relationnelle;

- L'étape [E2] a pour objet la localisation des données distribuées ;

- L’étape [E3] concerne l’Optimisation de la stratégie d’exécution par calcul du coût des transferts et du temps de réponse. L’objectif étant de minimiser le temps de réponse du système réparti.
- [E4] Optimisation locale des sous-requêtes.

VIII .les transactions :

Une transaction est une séquence d’opérations (lectures et écritures) invoquées sur des objets partagés. Le but du contrôle de concurrence est de s’assurer que les transactions s’exécutent de manière atomique (Elle doit s’exécuter intégralement ou pas du tout).

Une transaction débute lorsque l'on se connecte à la base (en début de session donc) ou lorsque la transaction précédente se termine.

Transaction répartie ; c'est-à-dire une transaction constituée de plusieurs transactions locales. Dans ce cas, on utilise un protocole de validation à deux phases. Dans la première phase dite phase de préparation, le site coordonnateur demande aux sites participants de se préparer à la validation. Lorsqu'il reçoit les notifications positives il lance alors la phase de validation en donnant l'ordre correspondant aux sites. Dans le cas contraire il donne l'ordre d'interrompre les transactions.

Conclusion:

A travers les différents points développés dans le présent chapitre, nous avons pu constater l'intérêt particulier porté aux systèmes répartis et aux différents problèmes auxquels ce type de solution a pu remédier.

Nous avons pu également, détailler et expliquer l'intérêt des bases de données réparties et les différents avantages offerts par ce type d'approche. Ce type de système est plus difficile à mettre en place et plus compliqué, et que malgré ses nombreux avantages, néanmoins des inconvénients existent, et son inconvénient majeur est la sécurité des données transmises via le réseau de communication.

Chapitre II

Sécurité des bases de données réparties.

Introduction :

Un des avantages évident des bases de données centralisées est sans contexte la sécurité apportée aux données, car elle peut facilement être contrôlée dans un site unique. Or, les bases de données réparties impliquent un réseau dont la sécurité est difficile à maintenir. Quels sont les menaces et les attaques à défendre ? Comment peut-on sécuriser les bases de données répartie ?

I. Propriétés principales :

La sécurité des bases de données soit centralisée ou bien répartie inclus trois principales propriétés : la confidentialité, l'intégrité et la disponibilité.

- **Confidentialité** : L'information protégée ne doit pas être accessible aux utilisateurs ou un programme non autorisés. C'est crucial :

Dans des environnements critiques ou stratégiques : militaires ou commerciaux, par exemple. Pour respecter le droit des individus à décider comment et dans quel but les informations les concernant peuvent être extraites, mémorisées ou transmises à d'autres individus.

- **Intégrité** : Les données ne peuvent pas être modifiées que par les utilisateurs habilités à le faire qu'elles soient dues à :
  - Des pannes de système,
  - Des manipulations erronées,
  - Des sabotages.

- **Disponibilité** : Il s'agit de détecter ou d'empêcher des dénis de service.

Il y a déni de service lorsqu'un utilisateur ne parvient pas à accéder dans un délai raisonnable, à une information ou à une ressource pour laquelle il a une autorisation d'accès.

Par exemple, une attaque consistant à saturer un serveur de fausses requêtes empêchant les requêtes valides d'être exécutées.

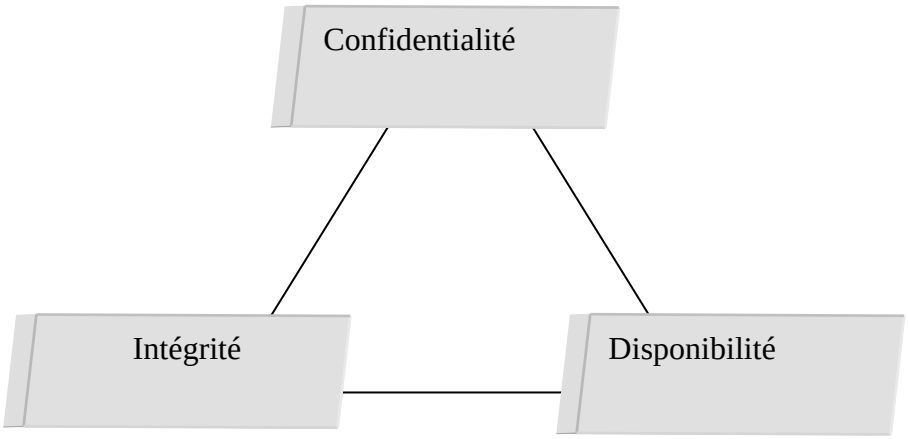


Figure II.1 : Principes de sécurité.

I. Techniques et types d'attaques [18]

II.1. Les grandes classes d'attaques :

- **Attaques visant l'authentification**
  - ✓ ***Déguisement (Mascarade)***

Pour rentrer dans un système on essaye de piéger des usagers et de se faire passer pour

Quelqu'un d'autre . **Exemple** : simulation bancaire.

- **Attaques visant l'intégrité des données**
- ✓ ***Modification de messages, de données***

Une personne non autorisée, un usager ou même un agent autorisé s'attribuent des avantages illicites en modifiant un fichier, un message (le plus souvent cette modification est réalisée par programme et entre dans la catégorie suivante).

Exemple : modification des données sur un serveur Web.

- **Attaques visant l'intégrité du flux de données**
- ✓ **Répétition ("replay")**

Espionnage d'une interface, d'une voie de communication (téléphonique, réseau local) pour capter des opérations (même cryptées elles peuvent être utilisables)

- ✓ **Répétition de l'opération pour obtenir une fraude :**

Exemple : Plusieurs fois la même opération de crédit d'un compte bancaire.

- **Attaques visant l'intégrité des programmes :**
- ✓ **Modification des programmes :**

***Les modifications à caractère frauduleux* :**

Pour s'attribuer par programme des avantages.

Exemple : virement des centimes sur un compte.

***Les modifications à caractère de sabotage* :**

Pour détruire avec plus ou moins de motivations des systèmes ou des données.

Deux types de modifications :

a) Infections informatiques à caractère unique

Bombe logique ou cheval

Dans un programme normal on introduit un comportement illicite mis en action par une condition de déclenchement ou trappe (la condition, le moment ou l'on bascule d'un comportement normal à anormal).

Exemple : licenciement de l'auteur du programme.

b) Infections auto reproductrices

Il s'agit d'une infection informatique simple (du type précédent) qui contient de plus Une partie de recopie d'elle même afin d'en assurer la propagation.

Virus : à action brutale.

Ver : à action lente (détruisant progressivement les ressources d'un système).

- **Attaques visant la confidentialité :**

Les attaques ayant pour but le vol d'information via un réseau par **espionnage des Transmissions de données** (espion de ligne, accès aux données dans des routeurs et des Serveurs Internet)

- **Canaux cachés**
- **Analyse de trafic**

On observe le trafic de messages échangés pour en déduire des informations sur les décisions de quelqu'un.

Exemples :

Bourse : augmentation des transactions sur une place financière.

Militaire : le début de concentration entraîne un accroissement de trafic important.

- ✓ **Inférence**

On obtient des informations confidentielles à partir d'un faisceau de questions autorisées (et d'un raisonnement visant à faire ressortir l'information).

II.2. Qui attaque ?

Dans cette partie on présente les différents attaquants :

Pirate externe :

Il est capable de s'infiltrer sur le serveur BD et de lire ses fichiers, il peut aussi casser une clé de chiffrement avec un texte connu.

Pirate utilisateur :

Ce type de pirate est reconnu par le SGBD et à accès à une partie des données suivant le mode de chiffrement, il a accès à certaines clés.

Pirate administrateur (DBA) :

Employé peu scrupuleux ou pirate s'étant octroyé ces droits,a accès à des données inaccessibles aux autres pirates (journal) et aussi peut espionner le SGBD pendant l'exécution.

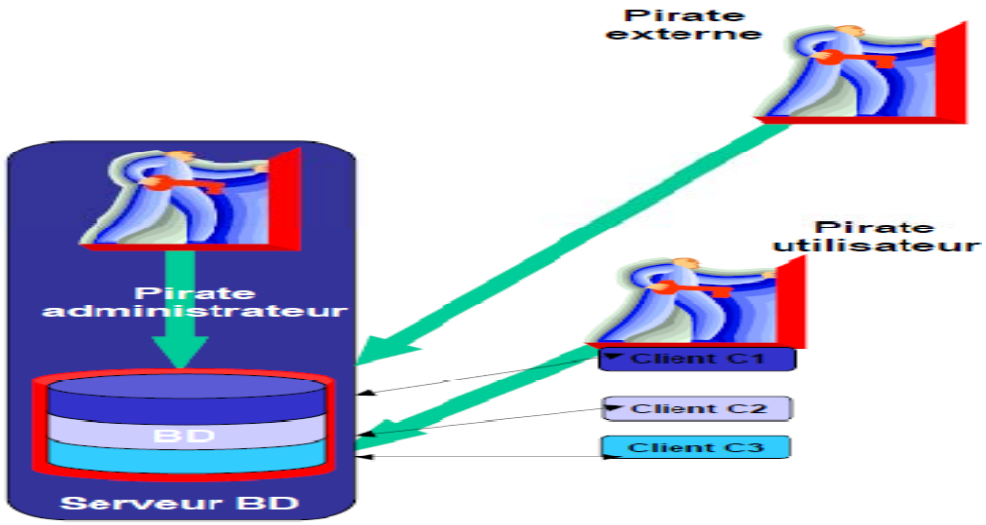


Figure II.2 : Différents attaquants. [7]

III. Les menaces :

III.1. Injection SQL

L'injection SQL directe est une technique où un pirate modifie une requête SQL existante pour afficher des données cachées, ou pour écraser des valeurs importantes, ou encore exécuter des commandes dangereuses pour la base. Cela se fait lorsque l'application prend les données envoyées par l'internaute, et l'utilise directement pour construire une requête SQL. Les exemples ci-dessous sont basés sur une histoire vraie, malheureusement. [8]

Exemple :

On suppose qu'une application récupère le nom *username* et le mot de passe *password* saisi, dans un formulaire, par un utilisateur et se connecte à la BD en exécutant la requête affectée à la variable `$requete` composée par concaténation :

```
$requete = "SELECT * FROM admin
```

WHERE username = '" & *username* & "'

AND password = '" & *password* & ";"

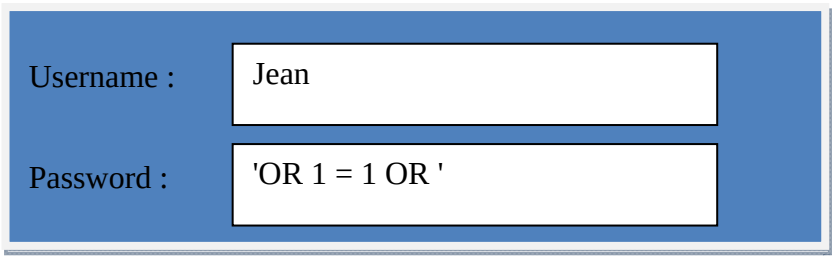


Figure II.3 : Accès avec identificateur et mot de passe.

Un attaquant désirant se connecter comme administrateur pourra saisir dans le formulaire :

La requête générée sera :

SELECT * FROM admin

WHERE username = 'Jean'

AND password = '' OR true OR '';

L'attaquant se retrouvera connecté comme 1er utilisateur de la BD qui a toutes les chances d'être l'administrateur.

III.2. Comptes utilisateurs par défaut :

Les plus grandes bases de données sont livrées avec des comptes utilisateurs par défaut, souvent oubliés au moment de leur déploiement. La tendance est désormais à leur limitation, mais ils demeurent encore très courants sur des bases en production. Ils permettent à un attaquant d'obtenir un accès mineur sur le système afin d'y exploiter ensuite des failles locales.[7]

III.2.1 Le type d'utilisateurs :

Il faut identifier les utilisateurs ayant besoin d'un accès à la base de données, ils peuvent être de différents types :

L'administrateur est une personne physique ayant tous les droits sur le SGBDR, mais pas forcément sur le contenu des bases de données : il peut réaliser des opérations de gestion des droits d'accès et des ressources systèmes mais on pourra choisir d'exclure ou non les droits d'accès en lecture et/ou écriture au contenu des bases de données. Bien que parfaitement logique d'un point de vu métier, pour la protection de données sensibles par exemple, retirer à un administrateur les droits de lecture et d'écriture sur le contenu d'une base de données n'a pas de sens d'un point de vu technique puisqu'il possède les capacités techniques de s'octroyer ses droits là. De plus, les opérations de sauvegarde, de restauration et de maintenance après incident peuvent l'amener à devoir accéder au contenu d'une base de données.

Bref, normalement, c'est l'utilisateur qui a tous les droits sur le SGBDR et les bases de données hébergées. C'est normalement une personne de confiance, compétente et prudente.

Une application peut être une application web, un outil de synchronisation entre sources d'informations ou tout programme accédant pour lui-même à la base de données. Ce type d'utilisateur logique n'a rien à voir avec l'utilisateur réel dénotant une personne physique ayant des besoins particuliers. Même si une application est utilisée par des personnes physiques, on pourra choisir de déléguer à l'application la gestion des droits d'accès à l'information en fonction des habilitations qu'elle décide de lui attribuer. Ainsi, une application peut être vue comme un utilisateur de base de données auquel on attribue des droits qu'elle pourra restreindre de façon transparente pour l'utilisateur final de l'application ainsi que pour le SGBD.

L'utilisateur est une personne physique se connectant directement à la base de données (commande *mysql* sous Linux) ou via une interface graphique (script *phpMyAdmin* sur un Intranet) ou utilisant une application qui va se connecter à la base de données sous l'identité de l'utilisateur (client lourd *MySQL Query Browser*).

III.3 Trop de privilèges :

Qu’il s’agisse de l'administrateur aux droits absolus, d’utilisateurs dont les droits ont étés copiés sur ceux d’autres collaborateurs pour se faciliter la vie ou encore d’applications à qui l’on donne un accès complet par simplicité, un accès trop lâche à la base de données est toujours à proscrire. Or, la facilité est souvent de mise dans ce domaine. [13]

On distingue :

Les privilèges objets qui concernent des opérations précises sur des tables, des vues, des procédures stockées…, dont le nom est spécifié ;

Les privilèges systèmes qui concernent des opérations sur tous les objets d'un certaine catégorie.

Ces privilèges varient sensiblement d'un SGBD à l'autre.

III.4. Des programmes vulnérables :

Plusieurs programmes peuvent présenter des vulnérabilités exploitées par un attaquant pour prendre le contrôle de la base ou de son serveur. Cela va du logiciel de gestion de la base de données (tous les éditeurs corrigent encore ou ont corrigé des vulnérabilités importantes), au système d'exploitation sur lequel elle s'appuie ou encore aux "procédures stockées" (des portions de code stockées dans la base de données au lieu d'être définies dans les applications). Certaines de ces dernières, notamment livrées par défaut, ont été à l'origine d'incidents de sécurité.

On ajoute aussi :

Le vol de données

Induit la perte de confidentialité des données stockées. La divulgation de données financières hautement confidentielles peut avoir un impact néfaste sur l'activité d'une entreprise : risque juridique, atteinte à l'image de marque, perte de confiance des partenaires industriels...

L'altération de données

Induit une perte d'intégrité, c'est-à-dire que les données ne sont plus dignes de confiance. En fonction de la rapidité de détection et de la qualité des sauvegardes, les conséquences peuvent en être réduites. Mais une application fonctionnant sur des données falsifiées peut voir son comportement fortement influencé : par exemple, un site de commerce électronique pourrait débiter le compte d'un autre client que celui réalisant la commande

La destruction de données remet sérieusement en cause la continuité de l'activité de l'entreprise concernée. Privée de ses données clients, sans sauvegarde, c'est le dépôt de bilan garanti !

IV. Politique de sécurité :

C'est l'ensemble de règles fixant les actions autorisées et interdites dans le domaine de la sécurité, et qui régissent la façon dont l'information sensible et les autres ressources sont gérées, protégées et distribuées à l'intérieur d'un système.

IV .1 Authentification :

La première étape afin de protéger les ressources d'un réseau est de pouvoir vérifier l'identité des utilisateurs. Cette vérification s'appelle authentification. L'authentification est la procédure mise en œuvre notamment pour vérifier l'identité d'une entité et s'assurer que l'identité fournie correspond à l'identité de cette entité préalablement enregistrée. *Dans la vie courante, l'authentification est réalisée par la carte d'identité nationale* .On authentifie un utilisateur en lui demandant de fournir quelque chose que seule cette personne a (par exemple *un jeton*), une information qu'elle seule connaît (*un mot de passe*) ou encore quelque chose qui est propre à cet utilisateur, comme une *empreinte digitale*. Plus l'utilisateur doit fournir des renseignements de ce type, plus faibles sont les risques qu'une autre personne parvienne à se faire passer pour cet utilisateur légitime.

NB : Chaque politique de sécurité d'un réseau devrait exiger la mise en place d'un ou plusieurs mécanismes d'authentification. [12]

Exemples :

IV .1.1 Authentification par un code d'identification et un mot de passe :

Sur la plupart des réseaux, le mécanisme d'identification et d'authentification utilise une paire code d'identification/mot de passe. Les systèmes à mot de passe peuvent être efficaces s'ils sont bien gérés, mais c'est rarement le cas. Les mécanismes d'authentification qui reposent uniquement sur les mots de passe ont souvent failli à leur tâche, et ce, pour un certain nombre de raisons. Premièrement, les utilisateurs ont tendance à créer des mots de passe qui sont faciles à mémoriser et donc, faciles à deviner, et deuxièmement,

Ils les changent rarement (alors qu'il faut le faire régulièrement). Par ailleurs, les mots de passe consistant en caractères aléatoires sont difficiles à deviner, mais ils sont également difficiles à mémoriser par les utilisateurs, ceux-ci doivent donc les écrire quelque part, la plupart du temps dans un endroit Facilement accessible dans le lieu de travail. L'utilisation de mots de passe multiples ne fait qu'aggraver le problème Le choix d'un mot de passe approprié (*un mot de passe à la fois facile à mémoriser pour l'utilisateur mais difficile à deviner pour toute autre personne*) a toujours été un problème. Les mots de passe composés de syllabes prononçables ont plus de chance d'être mémorisés que les mots de passe consistant seulement en caractères aléatoires. Il existe des logiciels de vérification des mots de passe; ils peuvent être utiles pour déterminer si un nouveau mot de passe est jugé trop facile à deviner et donc, inacceptable [A1]

IV .1.2 L'authentification à 2 niveaux de Paypal aisément contournée :

La méthode trouvée par Joshua Rogers nécessite de connaître les identifiants X(eBay) et Y(PayPal) de la personne, des informations que les programmes malveillants savent facilement récupérer sur des ordinateurs compromis. Le problème réside dans une page eBay qui permet aux utilisateurs de relier leur compte eBay avec PayPal. En le faisant, cela crée un cookie qui laisse croire à l'application PayPal que la personne est connectée, même si aucun code à six chiffres n'a été saisi, écrit Joshua Rogers sur son blog. C'est la fonction « =_integrated-registration » qui pose problème en ne vérifiant pas si la victime a activé le système d'authentification à deux facteurs, explique-t-il. De cette façon, un attaquant pourrait à plusieurs reprises accéder au compte PayPal d'une personne en le reliant à son compte eBay. [A2]

IV .2 Cryptographie :

Depuis sa création, le réseau Internet a tellement évolué qu'il est devenu un outil essentiel de communication mais qui présente tout de même des risques. Les transactions faites à travers le réseau peuvent être interceptées, d'autant plus que les lois juridiques ont du mal à se mettre en place sur Internet, il faut donc garantir la sécurité de ces informations, c'est la cryptographie qui s'en charge. La cryptographie était utilisée depuis bien longtemps ; à l'époque romaine, lorsque Jules César envoyait des messages à ses généraux, il ne faisait pas confiance à ses messagers. Aussi remplaçait il chaque A dans ses messages par un D, chaque B par un E, et ainsi de suite à travers l'alphabet. Seul quelqu'un qui connaissait la règle « décalé de 3 » pouvait déchiffrer ses messages. Aujourd'hui, la cryptographie est utilisée dans

divers applications réseaux telles que la messagerie électronique, les réseaux privés virtuels...La cryptographie désigne l'ensemble des méthodes ou techniques (*chiffrement, signature numérique et certificat*) permettant de garantir intégrité, authenticité, confidentialité des informations sensibles, et on appelle la personne qui la pratique un *cryptographe*.[10]

- **La cryptanalyse :**

C'est l'étude des procédés cryptographiques dans le but de trouver des faiblesses, et en particulier de pouvoir décrypter des textes chiffrés. Le *décryptement* est l'action qui consiste à trouver le message en clair sans connaître la clef de déchiffrement, et on appelle la personne qui pratique cet art un *Cryptanalyse*. On appelle la science qui englobe la cryptographie et la cryptanalyse, *la cryptologie*

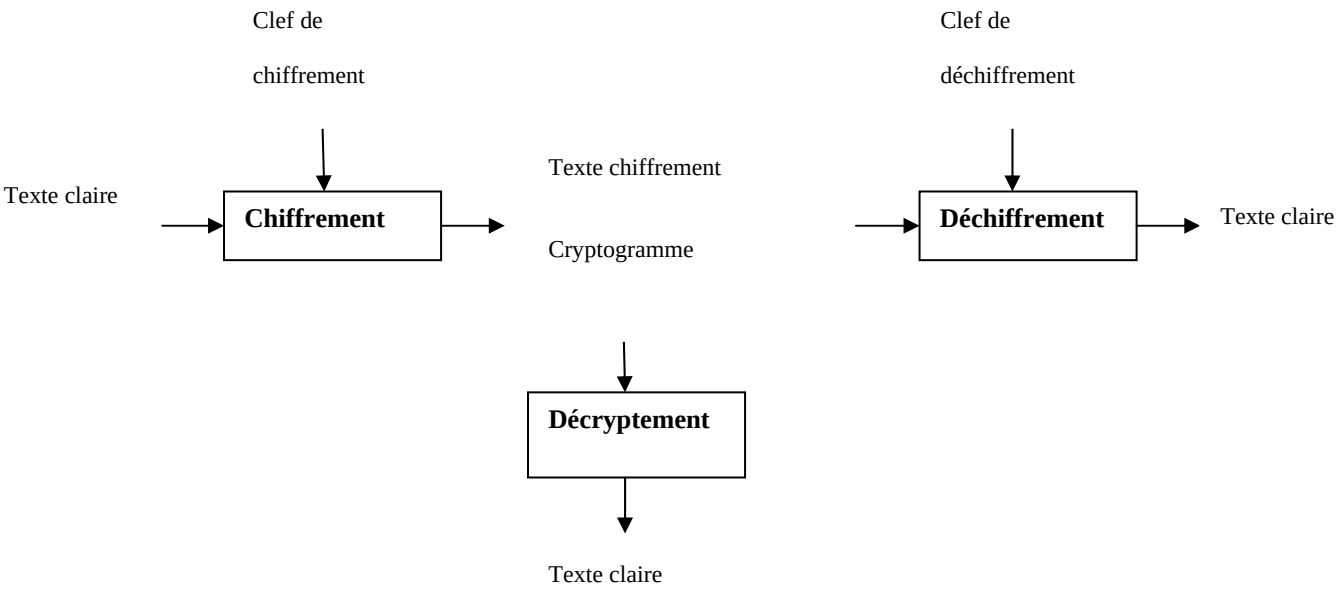


Figure II.4 : Schéma représentant la cryptologie.[10]

La cryptographie permet d'assurer :

IV .1 .1 La confidentialité :

La confidentialité est le premier problème posé à la cryptographie, il se résout par la notion de chiffrement.

IV .1 .1.2 Chiffrement :

Pour crypter un message ou un texte qu'on appellera *texte en clair*, on lui applique une série d'opérations simples telles que la substitution et la permutation suivant des règles bien définies qui ne sont connues que par l'émetteur et le récepteur du message dans le but de le rendre inintelligible pour les tiers non autorisés (cryptogramme ou texte chiffré) et on appelle ce procédé *chiffrement*. Inversement, le déchiffrement est l'action qui permet de reconstruire le texte en clair à partir du texte chiffré. Dans le monde de l'informatique moderne,

Les transformations en question sont des algorithmes construits à base de fonctions mathématiques qui dépendent d'un paramètre qu'on appelle clé de chiffrement/déchiffrement. [11]

- **Clé :** Ensemble des données d'entrée de l'algorithme qui transforme le texte clair en texte chiffrée et inversement.

Il existe deux grandes familles d'algorithmes cryptographiques à base de clefs :

a. Les algorithmes à clef privée : (*Chiffrement symétrique*)

Le chiffrement à clé privée exige que toutes les parties qui sont autorisées à lire l'information aient la même clé que celle qui est utilisée pour le chiffrement des données.

Clef de chiffrement = clé de déchiffrement

Comme exemple d'algorithme à clé privée, on peut citer : *Kerberos, Data Encryption Standard, International Data Encryption Algorithmes...*

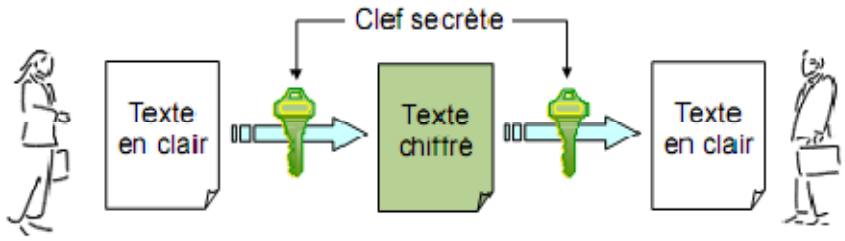


Figure II.5 : Schéma représentant le chiffrement symétrique.[10]

b. Les algorithmes à clé publique : (Chiffrement asymétrique)

Clef de chiffrement ≠ clé de déchiffrement

Le chiffrement asymétrique se base sur deux clés (*une privée et une autre publique*) pour chiffrer et déchiffrer les messages. Ces clefs sont distinctes et générées en même temps et elles dépondent étroitement l'une de l'autre, c.à.d. lorsqu'on chiffre avec l'une des clés, on doit forcément déchiffrer avec l'autre. Ainsi en utilisant la clef publique, tout le monde peut chiffrer un message que seul le propriétaire de la clef privée pourra déchiffrer, et inversement, si on utilise la clef privée pour le chiffrement, tout le monde (*ceux qui possèdent la clé publique*) peut déchiffrer.

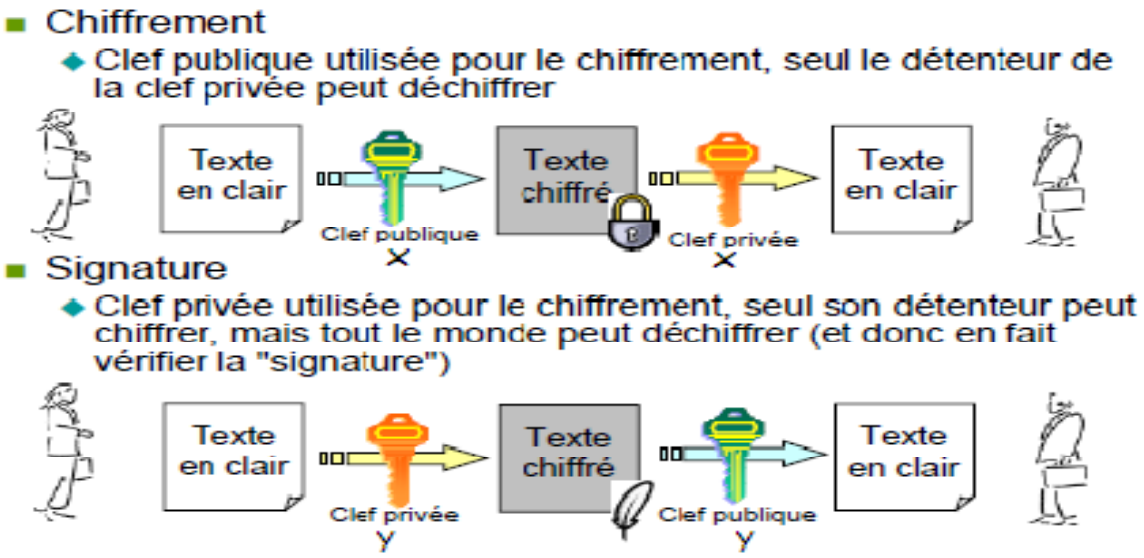


Figure II.6 : Cryptographie à clé publique. [10]

On peut identifier la provenance des données chiffrées par à la clef privée puisque une seule personne la possède, et donc lorsqu'une personne déchiffre le message avec sa clé publique, elle sait très bien d'où le message provient.

IV .1 .2 Intégrité et authenticité :

Authenticité=Authentification + Intégrité

Souvent on utilise le terme authentification afin de désigner l'authenticité, mais notez bien que l'authentification et l'intégrité sont inséparables. Lorsqu'un échange d'informations se présente au travers d'un canal de communication peu sûr, le destinataire aimerait bien s'assurer que le message s'émane de l'auteur auquel il est attribué et qu'il n'a pas été altéré pendant son voyage à travers le canal.

- **Authentification** : Consiste à s'assurer que les données s'émanent bien de l'expéditeur et non pas d'un autre utilisateur ou autre personne qui se prend pour l'expéditeur même.
- **Intégrité :** Consiste à s'assurer que les données n'ont pas été modifiées durant leur transfert. Pour répondre à ces deux critères, les signatures et les certificats numériques sont apparus.

IV .1 .3 Signature numérique :

Un des avantages majeurs de la cryptographie à clé publique est qu'elle procure une méthode permettant d'utiliser des signatures numériques. Les signatures numériques permettent à la personne qui reçoit une information de contrôler l'authenticité de son origine, et également de vérifier que l'information en question est intacte. Ainsi, les signatures numériques des systèmes à clé publique permettent l'authentification et le contrôle d'intégrité des données.

Une signature numérique procure également la non répudiation, ce qui signifie qu'elle empêche l'expéditeur de contester ultérieurement qu'il ait bien émis cette

information. Ces éléments sont au moins aussi importants que le chiffrement des données, sinon davantage.

Une signature numérique a le même objet qu'une signature manuelle. Toutefois, une signature manuelle est facile à contrefaire. Une signature numérique est supérieure à une signature manuelle en ce qu'elle est pratiquement impossible à contrefaire et, de plus, elle atteste le contenu de l'information autant que l'identité du signataire.

La norme ISO 7498-2 définit la signature numérique comme étant des données rajoutées à une unité de données ou une transformation cryptographique d'une unité de données permettant à un destinataire de prouver la source et l'intégrité de l'unité de données en question (seul l'expéditeur est apte à générer la signature).[19]

IV .3 L'audit :

Les audits fournissent un enregistrement des activités de chaque utilisateur. Les rapports d'audit associent une personne aux actions effectuées sur le système ou le réseau. Sans l'identification et l'authentification de cet utilisateur, le rapport d'audit est inutilisable car personne ne peut garantir que les événements enregistrés ont été effectivement exécutés par la personne en question. Il faut noter que le service d'intégrité doit garantir que les rapports d'audit n'ont pas été modifiés, sans quoi les informations qui figurent dans le journal d'audit deviennent toutes aussi suspectes. Electroniquement, les systèmes fournissent les journaux où sont enregistrées les actions effectuées sous l'identifiant de l'utilisateur (l'ID utilisateur). Si l'identification et l'authentification fonctionne correctement, ces événements retracent l'activité des personnes. L'analyse des informations contenues dans ces journaux permet de détecter d'éventuelles intrusions.

L'audit permet aussi de vérifier que les contrôles sont configurés correctement et conformément à la politique de l'entreprise. Comme par exemple si un utilisateur qui n'a pas le privilège d'accéder ou d'utiliser un service, on découvre grâce à l'analyse du journal qu'il arrive à avoir ce privilège, alors on remet en cause les dispositifs de sécurité qui sont en place. [14]

IV .3 .1 Le contrôle d'accès des utilisateurs :

Afin de permettre l'implantation de politiques de confidentialité et d'intégrité en leur sein, les systèmes d'exploitation disposent de mécanismes de contrôle d'accès. Typiquement, ceux-ci fonctionnent sur le modèle suivant :

Un *sujet* est une entité active inclut souvent les utilisateurs et les processus travaillant pour le compte des utilisateurs (qui peuvent être classés par groupes).

Un *objet* est une entité passive, un conteneur d'information à protéger, sur lequel un sujet peut effectuer une action (les fichiers, Données, programmes, périphériques matériels) ;

Une *permission* est une certaine action permettent aux sujets de manipuler les objets. Une permission peut être accordée ou refusée (par exemple, lecture, écriture, exécution).

Un *règlement de sécurité* constitué d'un ensemble de règles d'accès traduisant la politique de sécurité du système d'information.

Un *processeur de sécurité* qui vérifie que les requêtes adressées au système ne violent pas les règles d'accès et selon le cas autorise, modifie ou interdit la requête. [19]

Le contrôle d'accès est configuré par un ensemble de règles spécifiant un sujet, un objet et des droits d'accès. Un cas particulier de règles est celui spécifiant Une action d'un sujet vers un autre sujet (envoi de signal ou de message inter-processus).

Une fois que l'utilisateur est authentifié par l'intermédiaire, le système doit contrôler quelles données, applications et ressources l'utilisateur peut accéder dans le système. Les données ne doivent pas être protégées seulement contre les intrusions mais aussi les accès des utilisateurs ayant des limites qui doivent être respecté. Pour contrôler l'accès il faut d'abord renforcer la manière dont les utilisateurs font accès.

La sécurité typique dans les systèmes trois tiers, exige que chaque utilisateur soit limité par l'exécution des applications spécifiques sur l'intermédiaire,

Dépendant sur l'identité de l'utilisateur et le rôle lui accorder dans l'organisation. Un utilisateur qui accède à partir d'un intermédiaire ne doit pas avoir la permission d'accéder directement à la base de données. [12]

IV .3 .2 Modèles de contrôle d'accès :

Définition : une politique de contrôle d'accès est un ensemble de règles.

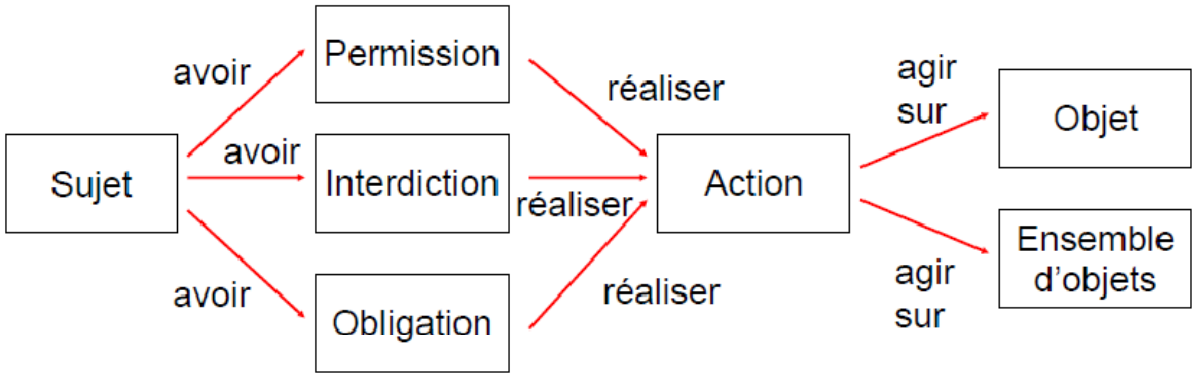


Figure II.7 : Format des règles. [12]

IV .4 Création de messages d'erreur sécurisés :

Si vous n'êtes pas prudent, un utilisateur malveillant peut déduire des informations importantes sur votre application à partir des messages d'erreur qui s'affichent. Respectez les règles ci-dessous.

- N'écrivez pas de messages d'erreurs reproduisant des informations pouvant être utilisées par des utilisateurs malveillants, telles qu'un nom d'utilisateur.
- Configurez l'application de façon à ne pas afficher d'erreurs détaillées. Si vous souhaitez afficher des messages d'erreur détaillés en vue du débogage, vérifiez préalablement que l'utilisateur est local au serveur Web. Pour plus d'informations, consultez
- Utilisez l'élément de configuration pour contrôler les personnes qui peuvent consulter les exceptions à partir du serveur.
- Créez une gestion des erreurs personnalisée pour les situations sujettes aux erreurs, telles que l'accès aux bases de données. [A3]

IV .5 Limitation du privilège de l'intermédiaire

IV .5 .1 Principe du moindre privilège :

Ce Principe stipule qu'un sujet ne doit disposer que des droits d'accès minimum pour assurer l'exécution des tâches qui lui sont assignées, pas un de plus.

Exemple : ne pas donner les droits de l'administrateur à tout utilisateur d'un système (système d'exploitation, SGBD).

Puisque le mécanisme d'authentification de l'intermédiaire est moins fort que celui de la base de données et que l'intermédiaire situé en d'hors de la zone protégée par le firewall en face des intrusions interviennes de l'Internet, la base de données doit limiter les privilèges d'un intermédiaire, et de le permis d'accéder au nom des utilisateurs spécifiques. [16]

IV .5 .2 Politique de gestion des privilèges :

Les privilèges :

Il convient pour chaque compte d'accès d'identifier les privilèges minima à accorder ainsi que le niveau de granularité adéquat.

Ici ; le terme *utilisateur* désigne une application aussi bien qu'une personne physique.

A.1. Classes d'objets et granularité :

Les SGBDR permettent généralement de spécifier assez finement les privilèges d'un utilisateur en fonction des objets manipulé :

Base de données.

Table (relation).

Colonne (attribut).

Ainsi, un utilisateur peut se voir attribuer un privilège pour toute une base de données, ou seulement pour quelques tables, ou encore sur uniquement quelques colonnes de certaines tables.

A.2. Classes de privilèges :

Les privilèges s'organisent autour de plusieurs classes :

Accès au contenu de l'information.

Gestion du schéma de la base de données.

Gestion des privilèges utilisateurs.

Gestion des paramètres systèmes.

V .La protection d’une base de données :

La base de données constitue souvent un applicatif critique pour l'entreprise. Sa sécurité, et le respect de bonnes pratiques et selon la référence [15] la protection d’une base de données suit les étapes suivantes :

V .1 Connaître son besoin

La sécurité de la base de données commence par une réflexion sur les usages et la population d'utilisateurs accédant à celle-ci, ainsi que sur la manière dont la connexion s'effectue. Est-ce directement par les utilisateurs ou par le biais d'un applicatif (interface Web, progiciel, etc.). Il est indispensable de connaitre la méthode et la nature des accès afin de définir une politique de sécurité adaptée.

"La connexion d'un SGBD avec un progiciel, qui nécessite une méthode d'interconnexion spécifique, peut avoir pour effet d'abaisser le niveau de sécurité. Les équipes sécurité et intégration, dont les missions ne sont pas forcément en accord, doivent souvent trouver un accord".

La sécurité ce n'est cependant pas uniquement la protection contre les attaques. D'autres questions se posent sur la garantie de la traçabilité, l'intégrité, l'audibilité, la confidentialité et la sauvegarde des données. La politique va par conséquent dépendre de ce que l'on souhaite garantir en fonction des besoins identifiés. Il sera ainsi incontournable de redonder les équipements d'interconnexion si la disponibilité est critique.

V .2 Une sécurité en amont :

Le déploiement d'une base de données est souvent la brique d'un projet plus global. La sécurité doit donc être pensée pour l'ensemble des éléments, surtout dans le cas d'un applicatif accédant à la base. Celle-ci peut être protégée mais si l'outil utilisé pour s'y connecter est vulnérable, il ouvrira des portes. Un SGBD ne pourra pas faire la différence entre une connexion légitime et une attaque par le biais d'un frontal Web.

V .3 Supervision

Un suivi des indicateurs de la base de données doit être assuré afin de détecter les anomalies, prévenir les interruptions de service et intervenir dans les meilleurs délais. La majorité des SGBD du marché embarquent désormais des systèmes de supervision. Charge ensuite à l'administrateur de base de données (DBA) de concevoir des filtres appropriés pour diagnostiquer toute évolution du mode de fonctionnement de la base.

V .4 Sensibiliser les DBA

L'administrateur doit être sensibilisé aux problématiques de sécurité, aux risques, à la criticité des contenus dont il a la charge et pas seulement à la performance. Un DBA peut avoir à superviser une dizaine de bases sans bénéficier de visibilité sur les données qu'elles hébergent et risquer par conséquent de ne pas avoir les bons réflexes.

V .5 Durcir le socle système

Une base de données repose sur une couche système. Cette dernière ne doit donc pas être négligée et faire l'objet d'un durcissement fort. Une base de données ne sera pas en mesure de se défendre contre une personne détenant des droits administrateur sur l'OS. Ce durcissement comprend l'application d'une politique de gestion des correctifs et du moindre privilège, la limitation des services (réseau et système) et applicatifs, la segmentation des droits ou encore une authentification via des mots de passe fort. Attention au paramétrage des SGBD lors de migration de versions.

V .6 Renforcer la couche BD

Tout comme le système, les correctifs de sécurité doivent être appliqués à la base de données. Pour des exigences de disponibilité, le patch management est cependant complexifié. Il faut veiller en outre au durcissement de l'installation par défaut.

V .7 Gestion des comptes

Les comptes par défaut doivent être verrouillés et les mots de passe remplacés pour respecter les normes de sécurité. La notion de politique du moindre privilège s'applique. C'est-à-dire qu'un utilisateur n'ayant par exemple besoin que de consulter les données ne doit en aucune façon disposer de droits en écriture. De même, la base doit être correctement segmentée pour qu'une habilitation ne concerne qu'un périmètre défini des données.

V .8 Méthodes d'accès

L'entrée sur la base de données doit être autorisée selon des méthodes précises. C'est à ce niveau que le filtrage sera défini. Si la connexion se fait depuis une application Web, alors seule celle-ci et le DBA seront autorisés à accéder. Ce filtrage est toutefois complexifié lors de l'intégration avec un PGI ou de connexion depuis une application en client lourd installée sur de nombreux postes.
Interviennent alors des aspects de gestions des profils et des utilisateurs, d'évolution des droits. Une cartographie des données et des habilitations doit être dressée pour définir les types de populations accédant à la base et les parties de celle-ci qu'ils sont autorisés à consulter.

V .9 Chiffrer les flux de données

Les informations envoyées en réponse à une requête ne doivent pas circuler en clair sur le réseau. Nul besoin de durcir l'accès et l'OS, s'il suffit d'écouter le trafic réseau. Les flux seront par conséquent chiffrés entre la base et les différents composants. [15]

Conclusion

La sécurité des accès à une base de données réparties est une préoccupation de tous les instants. Les privilèges doivent être restreints à l'indispensable et être actualisés régulièrement. Quant aux applications web, vulnérables par essence, elles doivent être développées de manière à réduire le risque d'accès frauduleux aux données.

Une collaboration étroite entre administrateur de base de données et développeur web permet de réduire considérablement les risques liés à la sécurité des bases de données. Les éléments de sécurité existant ne sont pas suffisants.

Chapitre III

Elaboration d’un exemple de BDR.

Introduction :

Pour l'étude de l'exemple on utilise la méthode UML, la méthode qui ce base sur deux parties l'analyse et la conception, et avant d'entamer la conception, il faut impérativement passer par la phase d'analyse qui permet d'identifier les différents acteurs qui intéressent avec le système ainsi que leurs besoins. Puis on passe à la phase de conception en s'appuyant sur les résultats de la phase d'analyse.et on termine par la réalisation.

Le champ d'étude se déroule au sien de la faculté génie électrique et informatique de l'université Mouloud Mammeri Tizi Ouzou.

I. Présentation de champ d'étude :

La faculté génie électrique et informatique est une unité de la commuté universitaire UMMTO (étudiants, enseignants, personnels, administratifs ……), donc il présentera un ensemble d'informations de divers domaines que ce soit pédagogiques ou administratifs. Pour cela l'organisation du doyen doit être efficace pour répondre à toute demande d'information et de facilité sa transmission.

L'organisation que nous allons proposer suit l'organisation hiérarchisée de la faculté dont elle structuré comme suit :

Faculté : doyen, les vices doyens, le secréterait général, la bibliothèque, départements.

Les départements : Automatique, Informatique, Electronique, Electrotechnique.

I.1 Organigramme de la faculté

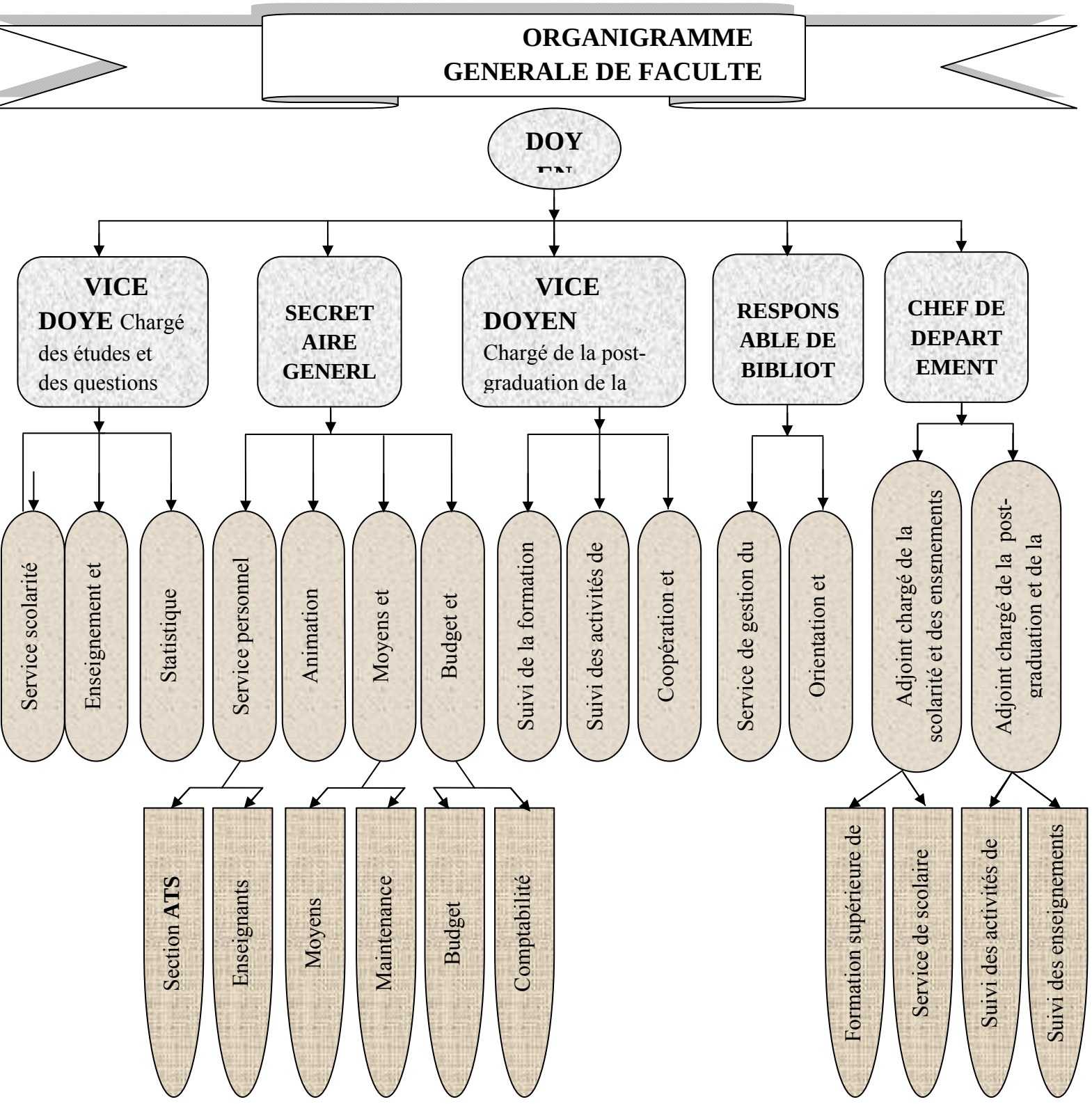


Figure III.1 : Organigramme général de la faculté génie électrique et informatique.

❖ **La démarche de modélisation de l'application.**

Vu qu'UML ne propose pas de démarche de développement nous allons suivre la démarche suivante : analyse, conception et réalisation mais avant de commencer le processus d'analyse et conception nous allons présenter les objectifs et les fonctionnalités attendus de notre application.

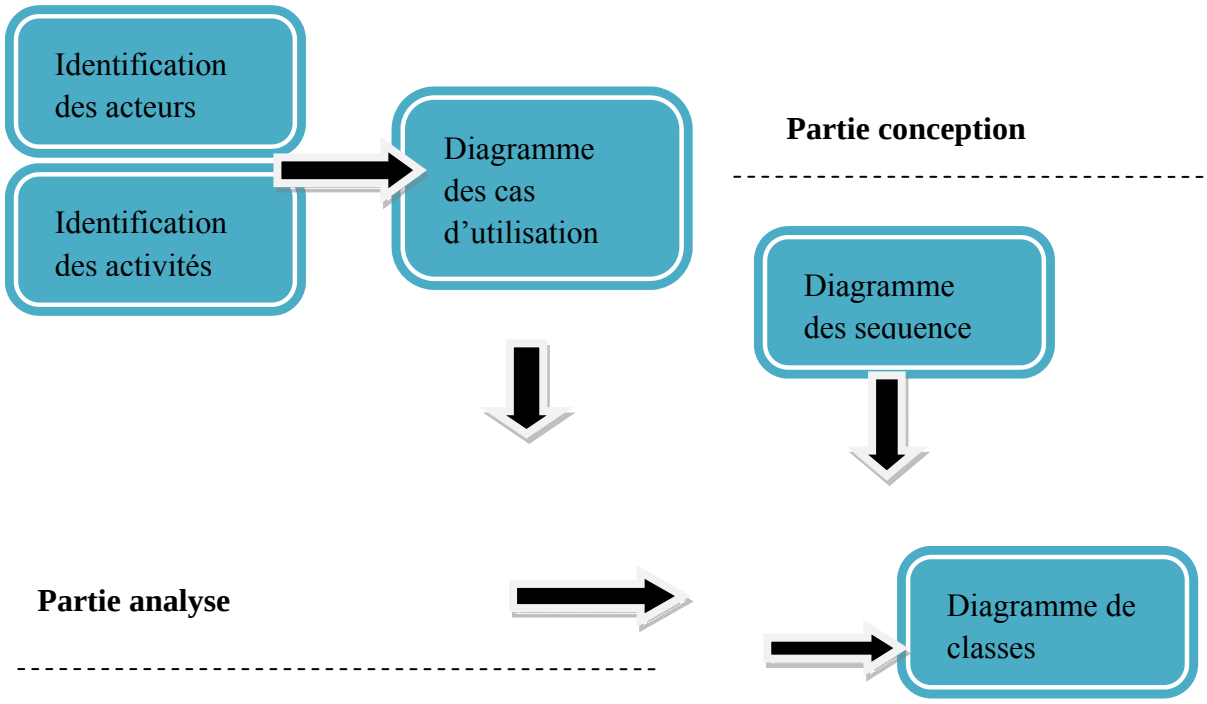


Figure III.2: la démarche de modélisation de l'application

II. Analyse :

Cette phase comprend l'identification des besoins de système, les acteurs participent, leurs interactions avec le système ainsi que les cas d'utilisation.

II.1 Spécification des besoins.

Notre projet consiste à concevoir une base de données répartie et son implémentation d'une base de données centralisée vers une base de données distribuée sur plusieurs sites, autrement dit sur les différents départements incluant dans la faculté.

Notre cas d'étude s'appuie sur un exemple de la faculté génie électrique et d'informatique, ou nous avons choisie d'intégrer les données des quatre départements d'informatique, automatique, électronique et électrotechnique a un service commun de la faculté.

La problématique réside au niveau des données partagées entre les départements et le service commun et comment faire circuler ce flux d'information d'une manière transparente en vue des acteurs de la faculté et commet gérer les données spécifiques a chacun des deux.

L'objectif de notre projet est de proposer une bonne solution à la problématique posée précédemment par une bonne conception et intégration d'une base de données répartie, qui va être utilisé par la faculté.

Donc pour la conception de la base de données de la faculté nous allons procéder d'une façon ascendante on suivant les étapes suivantes :

- Etudier les besoins et les acteurs des sites locaux et établissement des schémas conceptuels locaux.
- Etablissement du schéma d'intégration on utilisant l'approche ascendante.
- Réalisation du schéma globale après intégration.
- Traduction des schémas conceptuels obtenus en schémas relationnels logiques en appliquant les règles de passages UML –relationnel.

II.2 Identification des acteurs :

Comme nous avons cité déjà notre système comporte plusieurs sites (sites locaux et un site global) donc nous allons identifier les acteurs de chaque site comme le monter le schéma suivant :

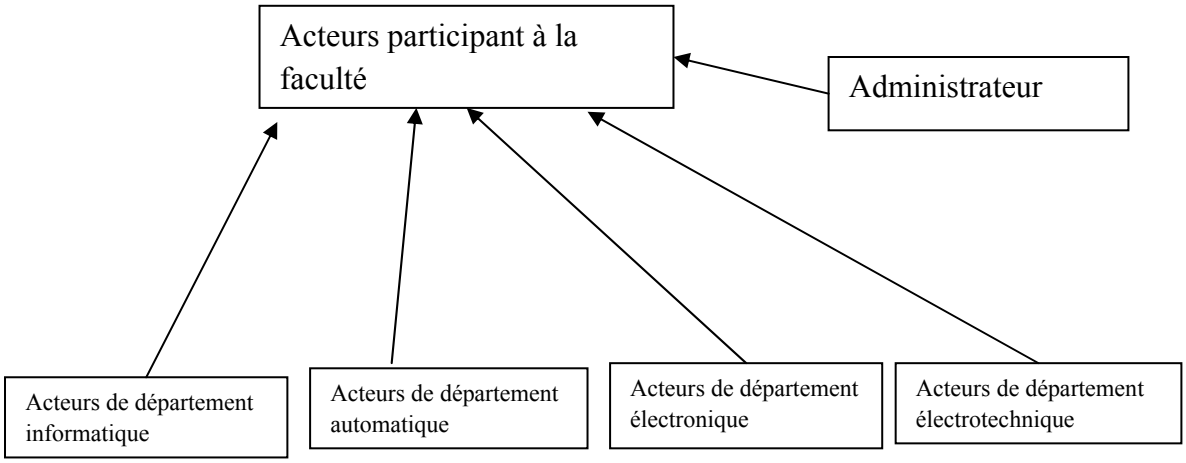


Figure III.3: **Les acteurs participant à notre système.**

II.2.1 Identification des acteurs de chaque département (informatique, automatique, électronique, électrotechnique).

- Les acteurs d'un département sont nombreux et ils sont tous concernés par la faculté dans notre cas on se limite aux acteurs suivant :

✓ Enseignant : est un acteur qui peut accéder à la faculté pour avoir les informations et les applications spécifiques au département et aussi a l'espace processus métier.

✓ Etudiant : est un acteur concerné juste par les applications et les données spécifiques au département.

✓ L'administrateur d'e-département : c'est l'administrateur de site département.

- **Administrateur :**

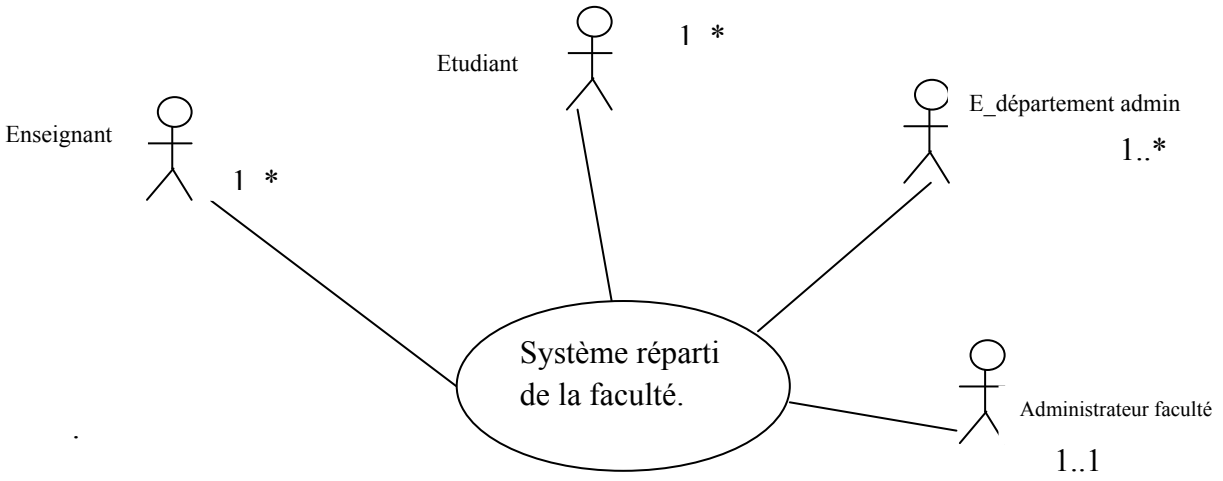
E_département admin.

E-faculté admin.

Pour l'administrateur du système réparti, l'administrateur de faculté s'occupe de la gestion de la BDD répartie.

II.3 Diagramme de contexte :

Le diagramme suivant illustre le contexte de notre travail



.

Figure III.4 : Diagramme de contexte

II.4 Identification et représentation des cas d'utilisation.

Définition :

Un cas d'utilisation désigne une fonctionnalité visible de l'extérieur du système dont on désire la fonctionnalité .un cas d'utilisation doit répondre à un besoin en définissant une partie du comportement attendu du système sans révéler sa structure interne.

Le tableau récapitulatif suivant présente les différents taches réaliser par les acteurs de système sur les différents sites soit local ou intégré ou bien l'espace ou naviguer (espace pédagogique ou administratif

Site	Acteur	Cas d'utilisation Espace pédagogique	Base locale	Base répartie
	Etudiant	T1 :S'inscrire à la faculté. T2 :S'authentifier. T3 : Accéder a son espace. T4 : Modifier son profil. T5 : Consulter sa messagerie. T6 : Envoyer un message à un enseignant T7 : Envoyer un message à l'administration. T8 : Consulter son emploi de temps. T9 : Voir les filières et les spécialités. T10 : Voir les modules d'une spécialité. T11 : Consulter le site de département. T12 : Consulter la bibliothèque de département. .	OUI	
	Enseignant	T1, T2, T3, T4, T5, T6, T7, T8, T9, T10, T11, T12. Idem qu'étudiant. T14 : Envoyer et répondre aux messages d'Etudiants. T15 : Déposer des cours. T 16: Remettre les notes des étudiants au chef de département.	OUI	
	E_departement Admin	T17 : Se connecter. T18 : Consulte ces informations. T19 : Gestion des enseignants (valider, modifier, supprimer). T20 : Gestion des étudiants (valider inscription, modifier, supprimer). T21 : Lire et répondre aux messages reçus. T22 : Gestion des formations du département (filières, spécialités, modules). T23 : Envoi PV des notes d'étudiants, au service de scolarité	OUI	**oui**
	Administrateur Faculté	T29 : Se connecter. T30 : Gestion de départements. T31: Gestion d'E-departement_admin. T33 : Répondre aux messages reçus.	oui	**oui**

II.4.1 Les diagrammes de cas d'utilisations :

Après l'identification des différents acteurs ainsi que les cas d'utilisation qui sont mis en œuvre par ces acteurs, nous terminons la phase d'analyse par l'élaboration des différents diagrammes des cas d'utilisation associés a chaque acteur définie précédemment.

II.4.1.1 Définition :

- **Cas d'utilisation:** (en anglais *use case*) permet de mettre en évidence les relations fonctionnelles entre les acteurs et le système étudié. Le format de représentation d'un cas d'utilisation est complètement libre mais UML propose un formalisme et des concepts issus de bonnes pratiques.

  Le diagramme de cas d'utilisation permet de représenter visuellement la séquence d'actions réalisées par un système. Il est représenté par une boîte rectangulaire, produisant un résultat sur un acteur, appelé acteur principal, et ceci indépendamment de son fonctionnement interne.

- **Relation entre cas d'utilisation :**

  Il existe trois types de relation standards entre cas d'utilisation :

  - **Include**: le cas d'utilisation incorpore explicitement et de manière obligatoire un autre cas d'utilisation à l'endroit spécifié,
  - **Extend:** le cas d'utilisation incorpore implicitement de manière facultative un autre cas d'utilisation à l'endroit spécifié.

  - **Généralisation:** les cas d'utilisation descendants héritent des propriétés de leurs parents.

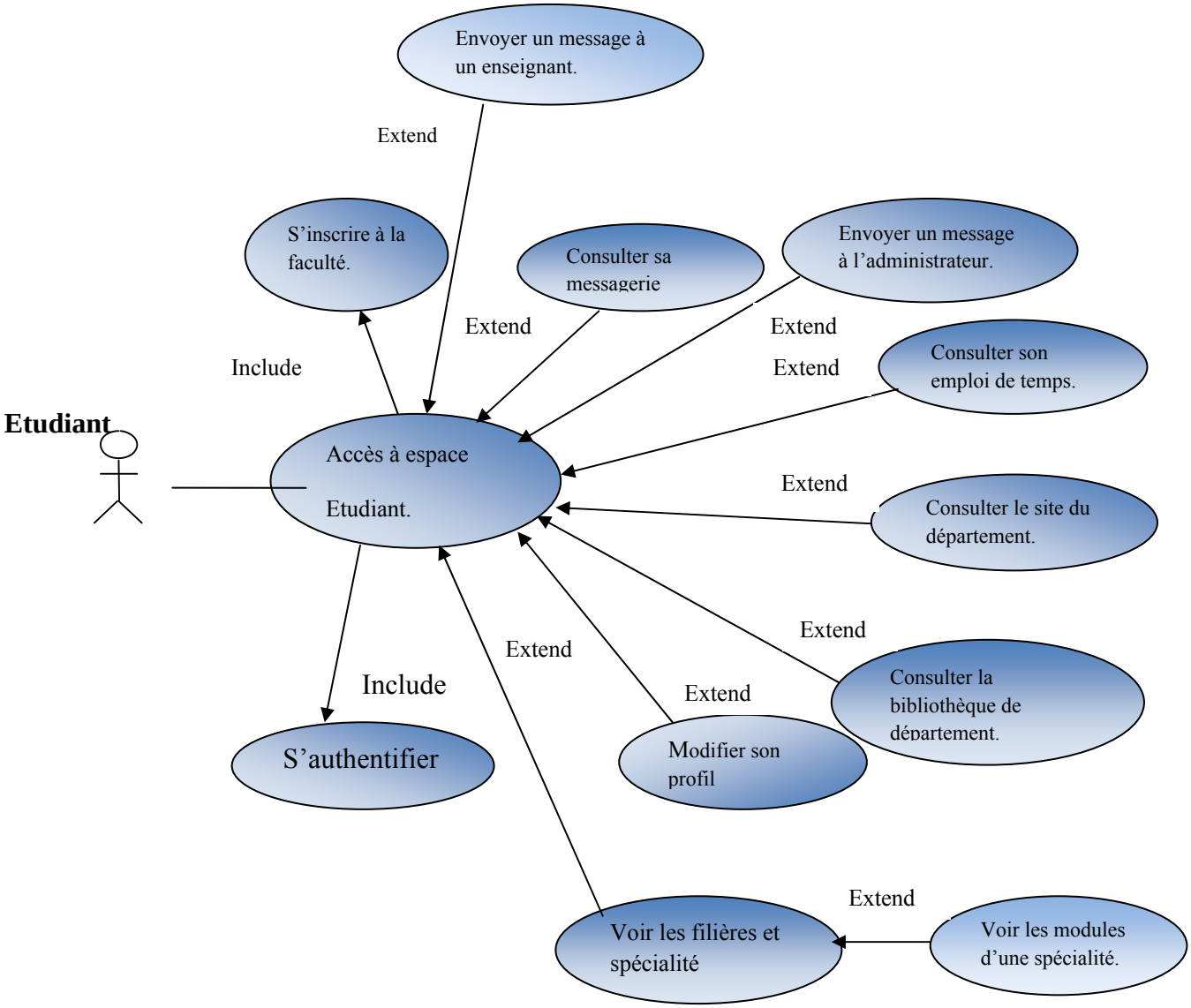


Figure III.5 : **Diagramme de cas d'utilisation pour « l'étudiant ».**

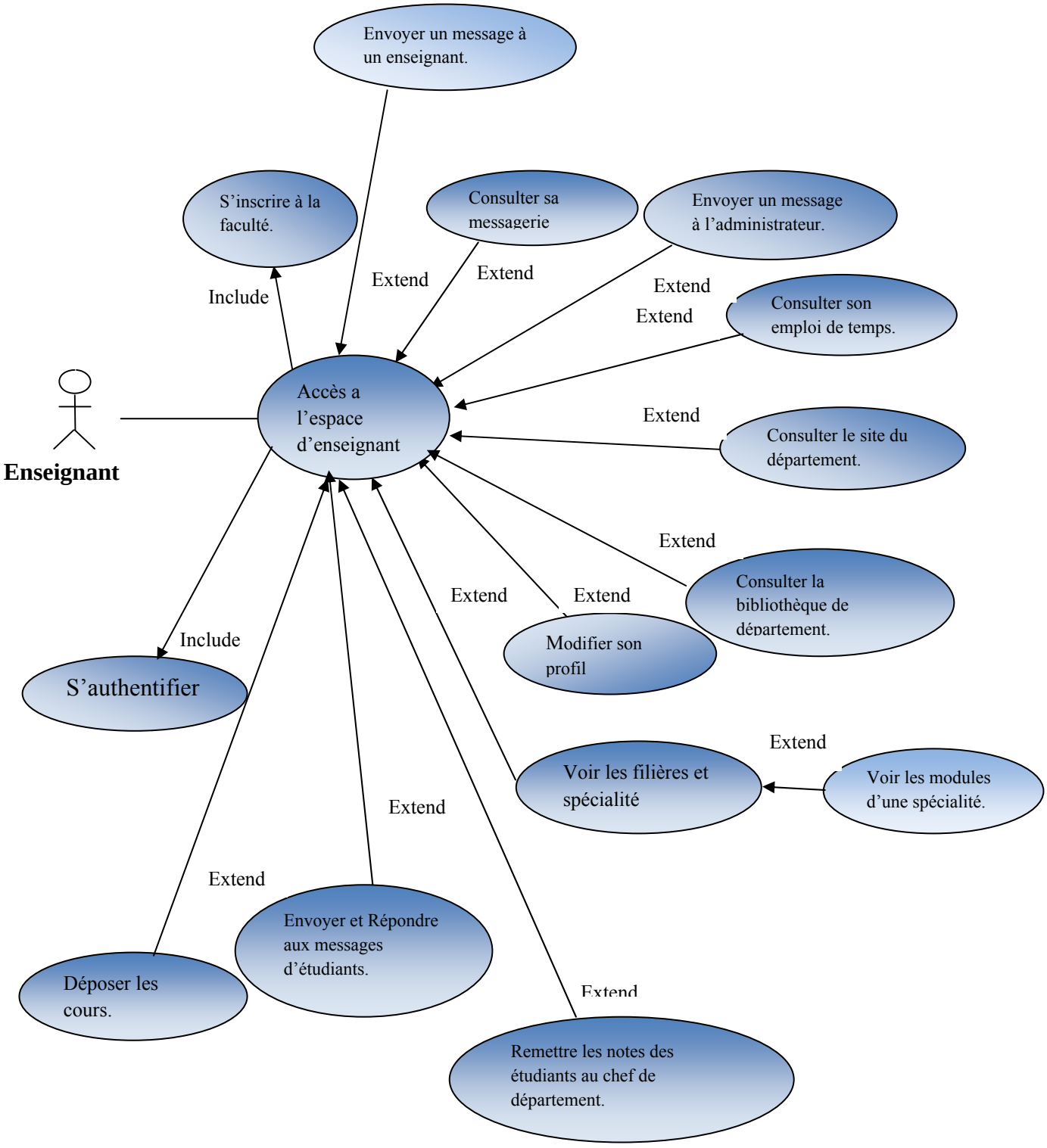


Figure III.6 : **Diagramme de cas d'utilisation pour « l'enseignant ».**

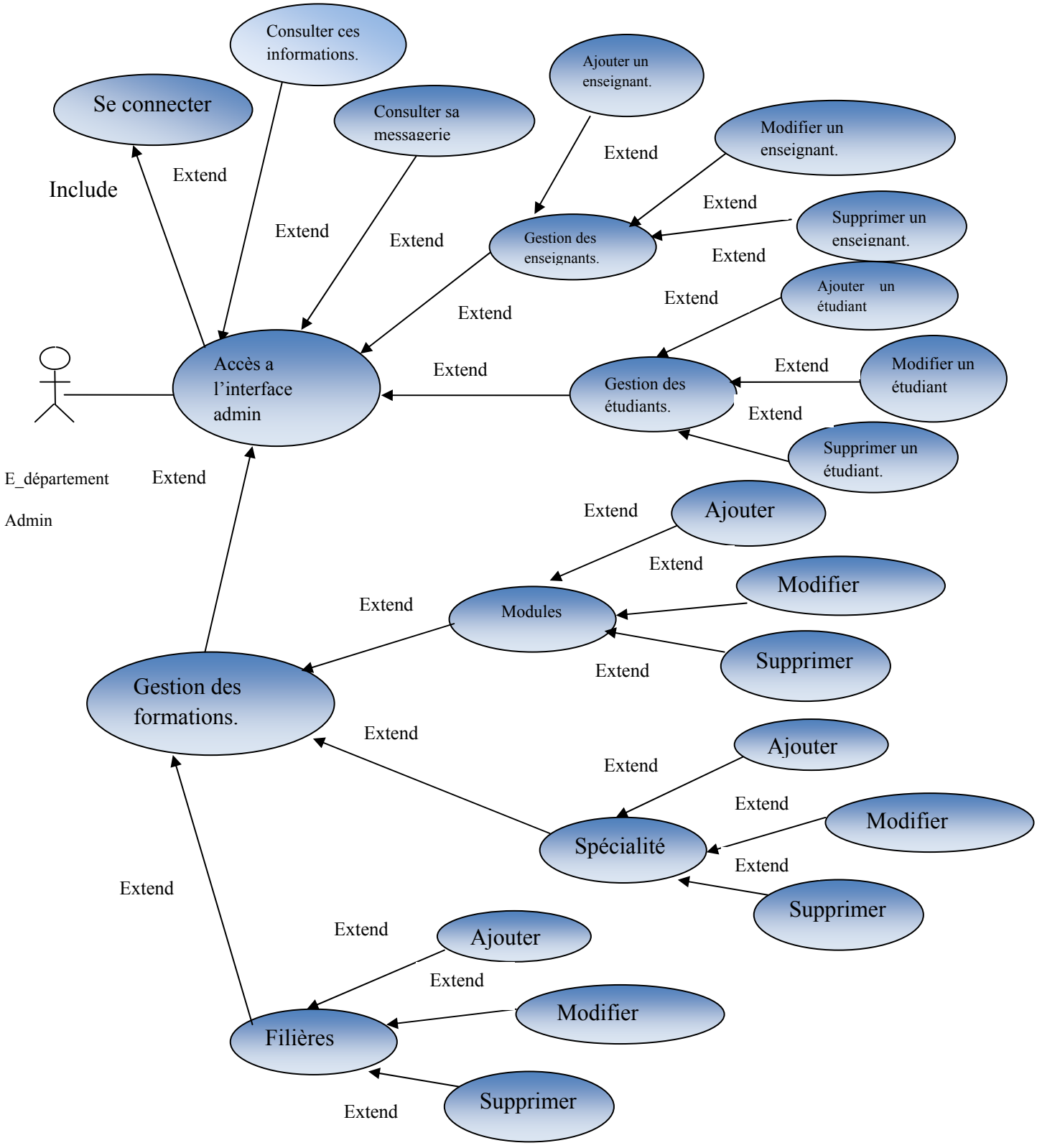


Figure III.7: **Diagramme de cas d'utilisation pour « E_département Admin ».**

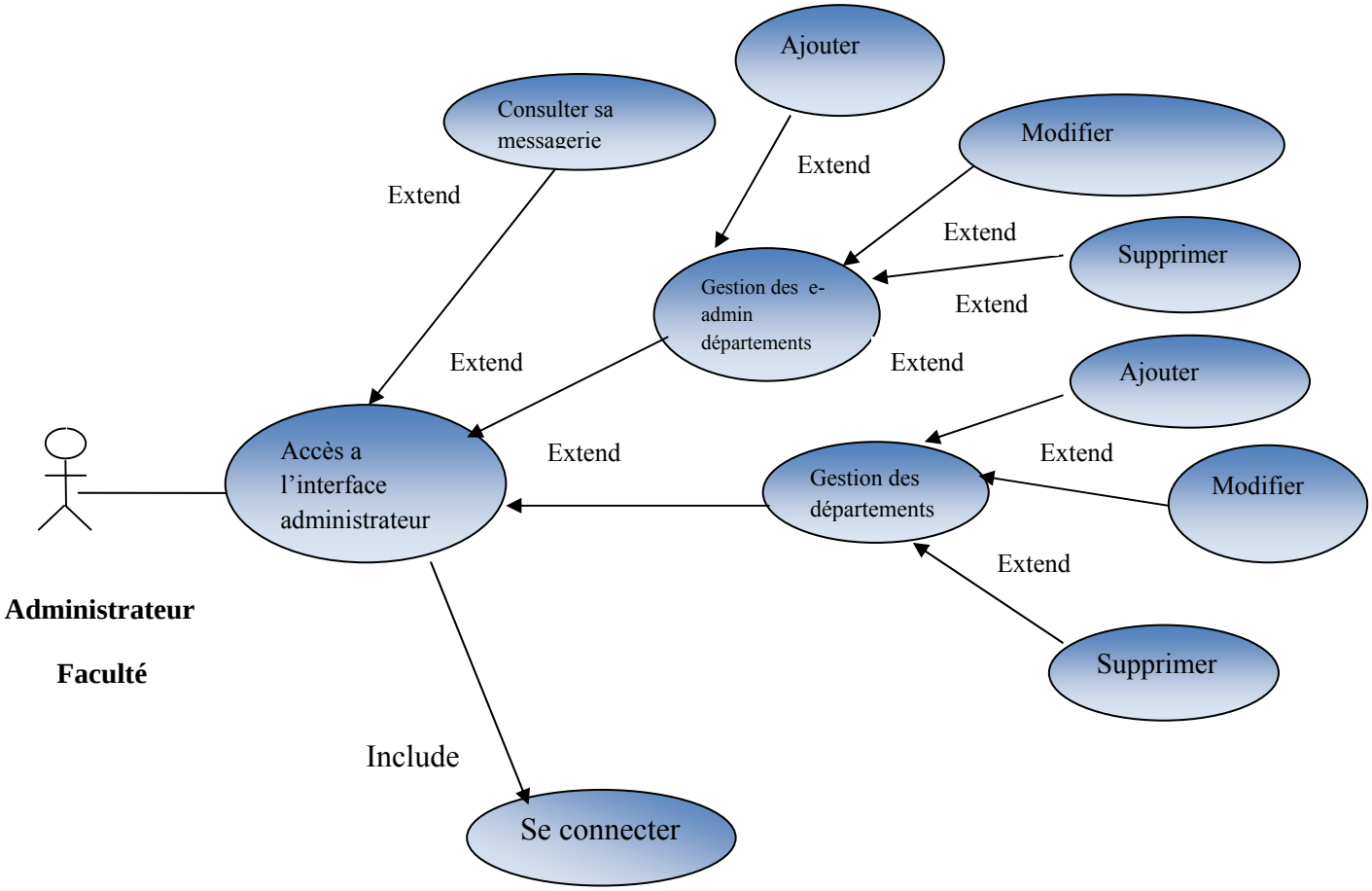


Figure III.9 : **Diagramme de cas d'utilisation pour « Administrateur Faculté ».**

III .Conception :

Après avoir analysé les différents acteurs participant, ainsi leur cas d'utilisation ;

Nous allons passer dans ce qui concerne la phase de conception qui se base essentiellement sur les différents diagrammes de séquences et diagrammes de classe que nous allons présenter dans ce qui suit :

- **Les diagrammes de séquence :** il présente la vue dynamique du système. L'objectif du diagramme de séquence est de présenter les interactions des objets en indiquant la chronologie des échanges. Cette représentation se réalise par cas d'utilisation.
- **Les diagrammes de classes :** représentent la vue statique des objets qui donnent naissance à la BDD, leur intérêt majeur est de modéliser les entités d'un système.

Autrement dit, ils expriment les relations existantes entre les pages client et serveur.

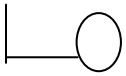
Dans ce qui suit on représentera quelques diagrammes de séquences et de classes, correspondant aux cas d'utilisation déjà décrits.

III.1 Elaboration avec les diagrammes de séquence :

Les diagrammes de séquence permettent de représenté les interactions entre objet selon un point de vue temporel.

- **Les objets interface :**

Un objet interface représente l'interface entre l'acteur et le système

Icône : 

- **Les objets entités :**

Les objets entités sont des objets d'écrit dans plus d'un cas d'utilisation.

Icône : 

- **Les objets de contrôle :**

Il représente les activités des processus du système, il dirige les activités des objets interfaces et entités.

Icône : 

III.2 Présentation de quelques diagrammes de séquence correspondant aux cas d'utilisation déjà décrits :

Nous avons choisie de présenter quelques exemples des diagrammes de Séquences correspondant aux cas d'utilisation présenté précédemment.

Comme nous l'avons cité précédemment notre démarche procède d'une manière ascendante donc nous allons schématisés quelques diagrammes de séquence de quelques taches déroulent au niveau des sites locaux.

Site locale	Acteurs	Diagrammes de séquences de cas d'utilisation
Site département « informatiques »	Etudiant	- S'inscrire a la faculté. - S'authentifier.
	Enseignant	- envoyer un message à un étudiant ou administration
	E-admin_département	- Valider une inscription (enseignant, étudiant).

III.2.1 Cas d'utilisation « S'inscrire a la faculté ».

Il contient les objets suivants :

Ce cas d'utilisation contient les objets suivant :

- **Les objets de type interface :**
  - Page principale.
  - Formulaire d'inscription.
  - Page de confirmation.

- **Les objets de type entité :**
  - Inscription
- **Les objets de type contrôle:**
  - Enregistrer dans la base de données.

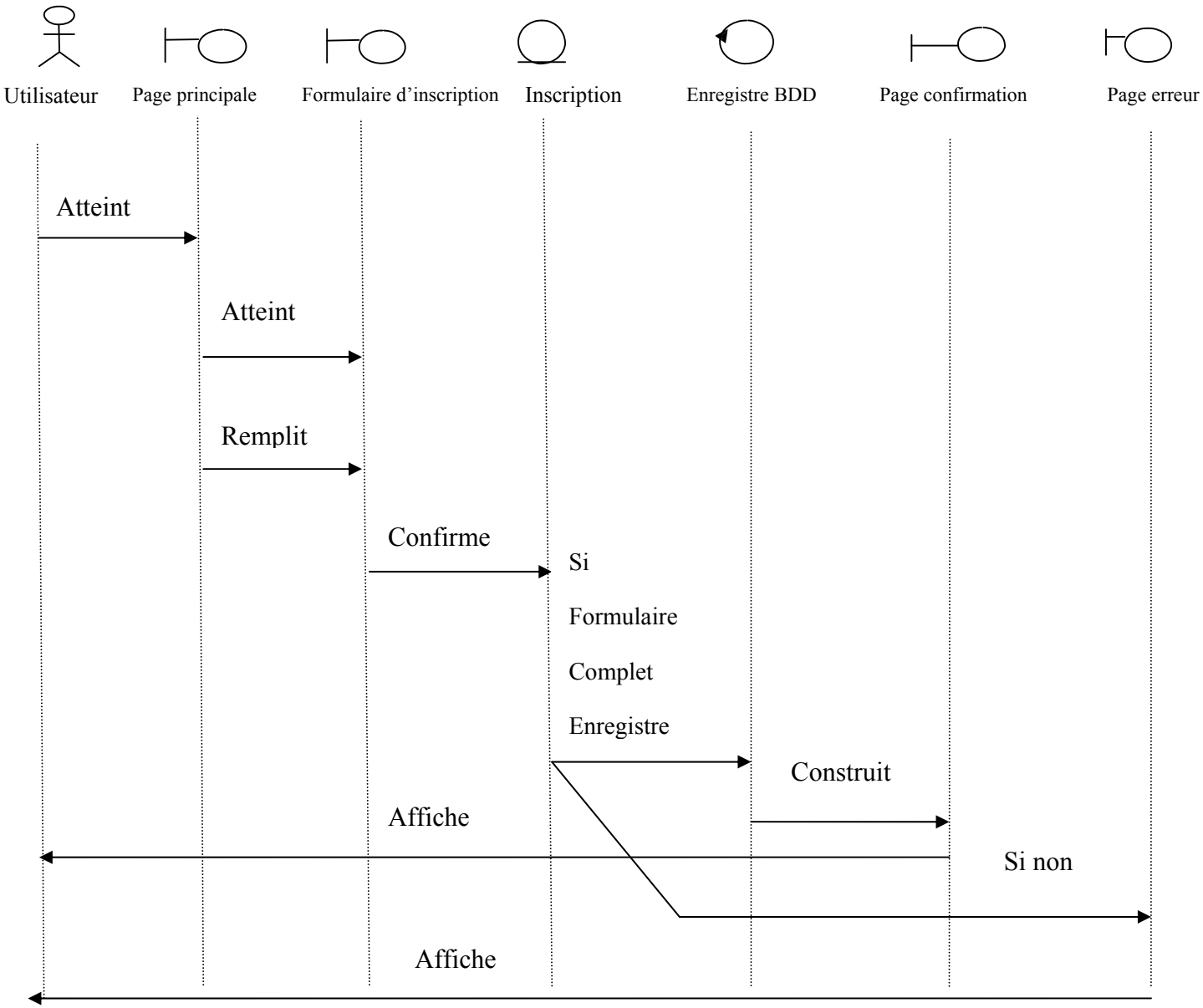


Figure III.10 : Diagramme de séquence détaillé pour le cas <<s'inscrire à la faculté >>

1. L'utilisateur atteint la page principale (page d'accueil) de la faculté en saisit l'URL.
2. L'utilisateur clique sur le lien inscription et le système lui affiche le formulaire d'inscription.
3. L'utilisateur remplit le formulaire d'inscription.

4. L'utilisateur confirme son inscription avec le bouton valider.
5. Le système vérifie si le formulaire est complet, il enregistre l'inscription dans la base de données.
6. Le système construit la page de confirmation.
7. Le système affiche la page de confirmation
8. Si non il redirige vers la page d'erreur.
9. Le système affiche un message d'erreur (page d'erreur).

III.2.2 Cas d'utilisation « s'authentifier » :

Il contient les objets suivants :

- ***Objet interface :***

  - Page principale
  - Page d'authentification
  - Espace personnel

- ***Objet contrôle :***
  - La recherche dans la base de données

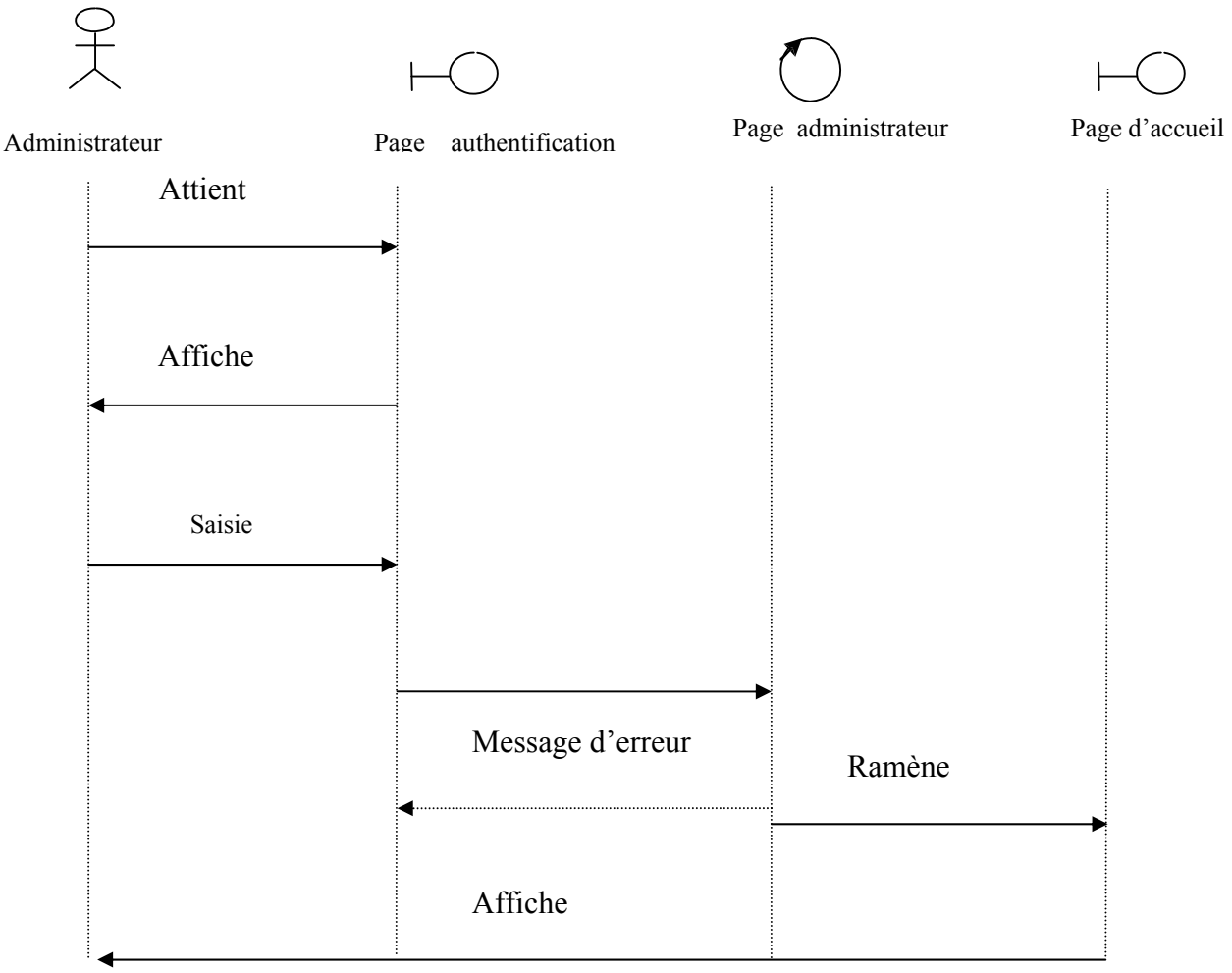


Figure III.11: Diagramme de séquence de cas d'utilisation :<<s'authentifier>>

1. L'administrateur atteint la page d'authentification.

2. Le système l'affiche la page d'authentification

3. L'administrateur saisit ses cordonnées.

4. Le système vérifie les informations saisies.

5. Le système construit l'interface d'administrateur si les informations sont correctes si non il affiche un message d'erreur.

III.2.3 Cas d'utilisation « Envoyer un message à un étudiant ou enseignant ou administration » :

Il contient les objets suivants :

- ***Objet interface :***

- Page messagerie
- Page nouveau message
- Page de confirmation

- ***Objet contrôle :***
  - Enregistrer dans la base de données
- ***Objet entité*** :
  - Message

Remarque : dance ce diagramme l'utilisateur c'est étudiant et enseignant.

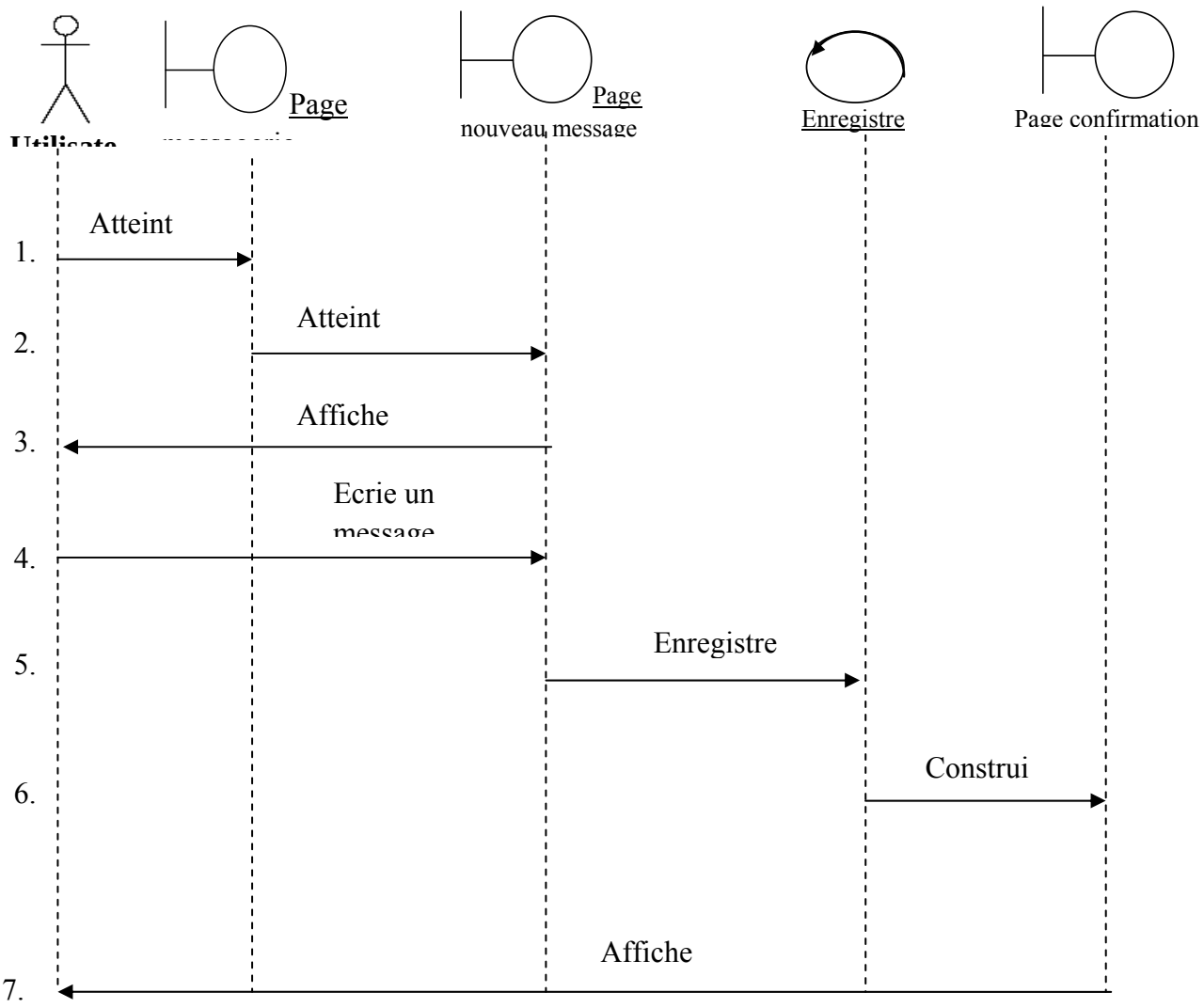


Figure III.12 : Diagramme de séquence de cas d'utilisation « envoyer un message à un étudiant ou enseignant ou administration

1. L'utilisateur atteint la page de messagerie après l'étape d'authentification

2. L'utilisateur clique sur le lien nouveau message et le système lui affiche la page de nouveau message

3. L'utilisateur saisit le message et l'adresse de destinataire et clique sur le bouton envoyer

4. Le système enregistre le message dans la base de données

5. Le système construit la page de confirmation

6. Le système affiche la page de confirmation de l'envoi de message.

III.3 Diagramme de classe :

Définition *:* un diagramme de classe est la vue statique des objets qui donnent naissance à la base de données, leur intérêt majeur est de modéliser les entités d'un système.

Autrement dit, ils expriment les relations existantes entre les pages client et serveur.

Nous allons présenter les schémas conceptuels locaux des bases locales, par la suite on les vues créées pour compléter le schéma conceptuel global de la base de données fédérée comme le montre la figure suivante.

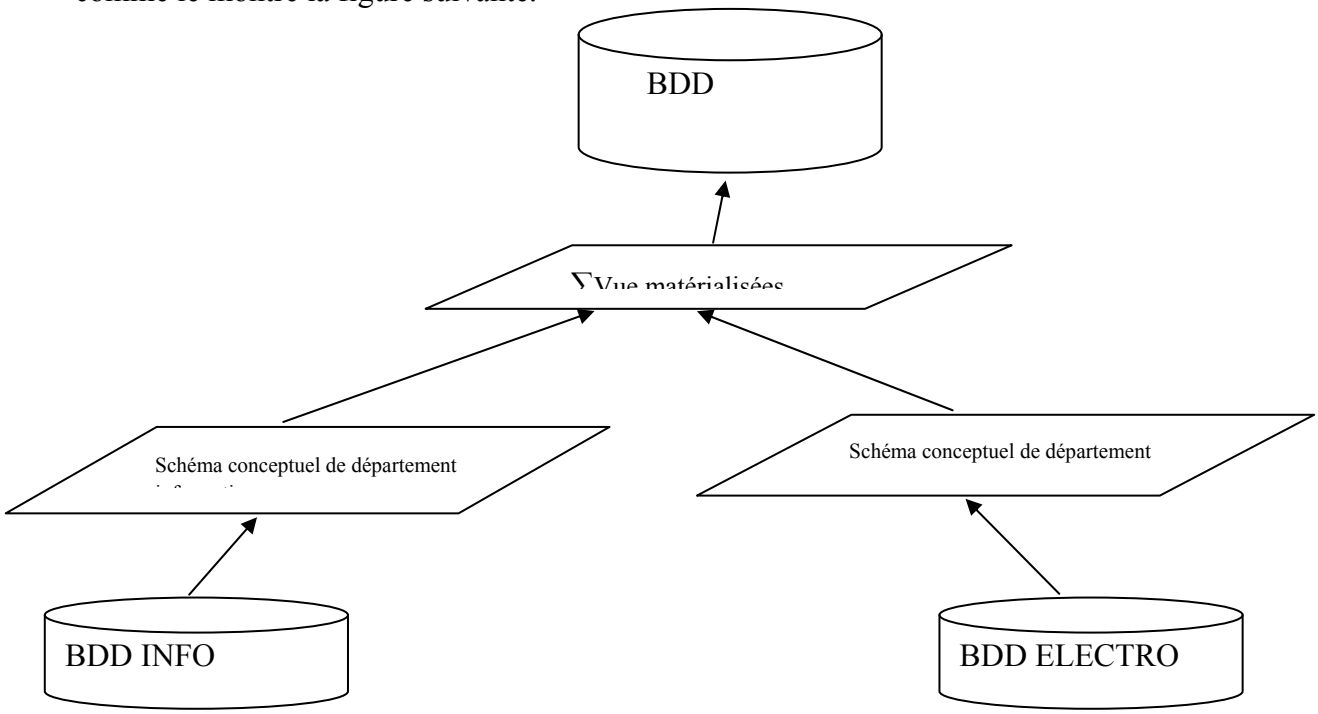


Figure III.13: ***Plan de représentation des schémas conceptuels.***

III.2.1 diagramme du schéma logique du site département

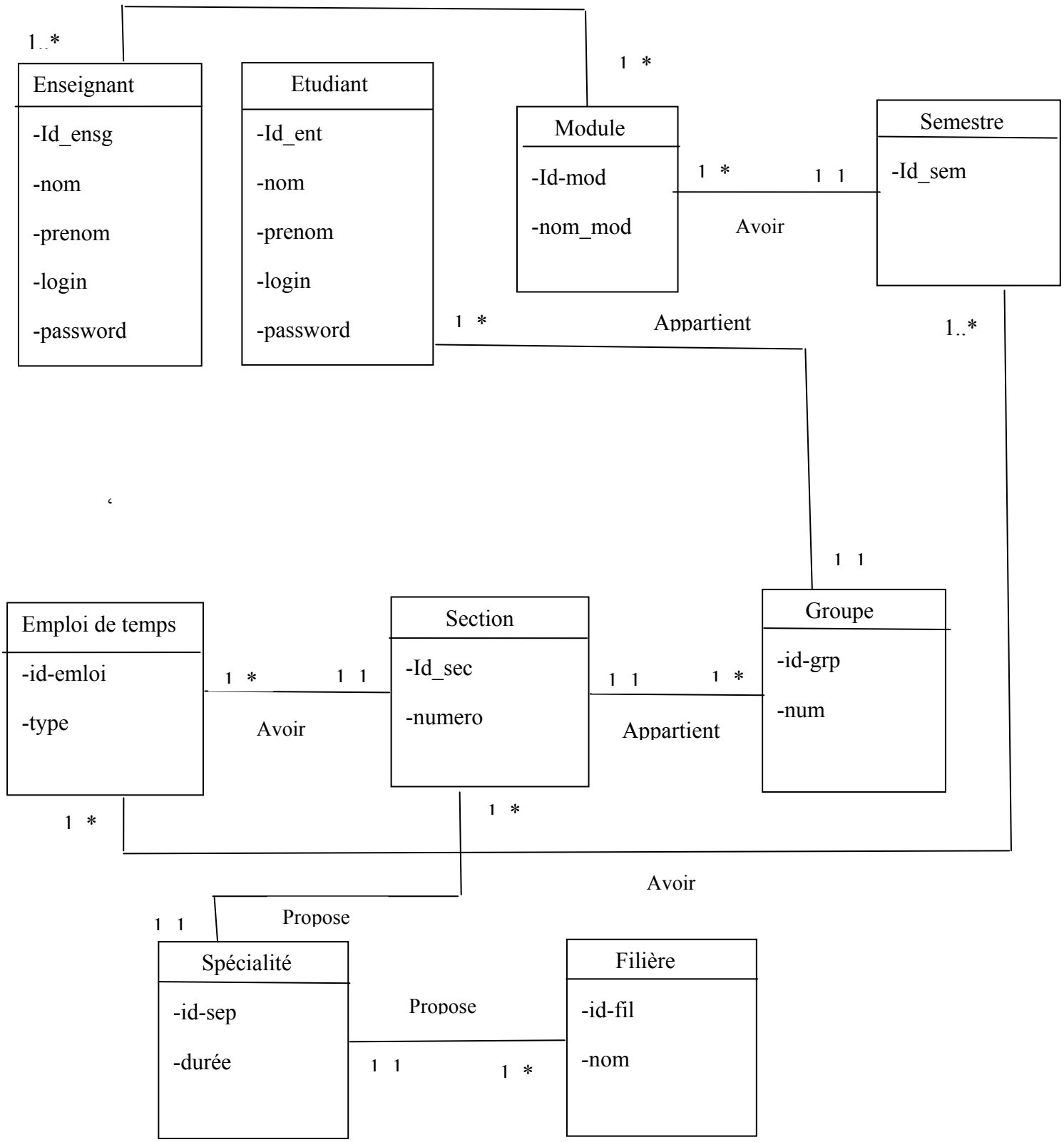


Figure III.14 : diagramme du schéma logique du site département.

III.2.2 Diagramme du schéma logique de la base faculté :

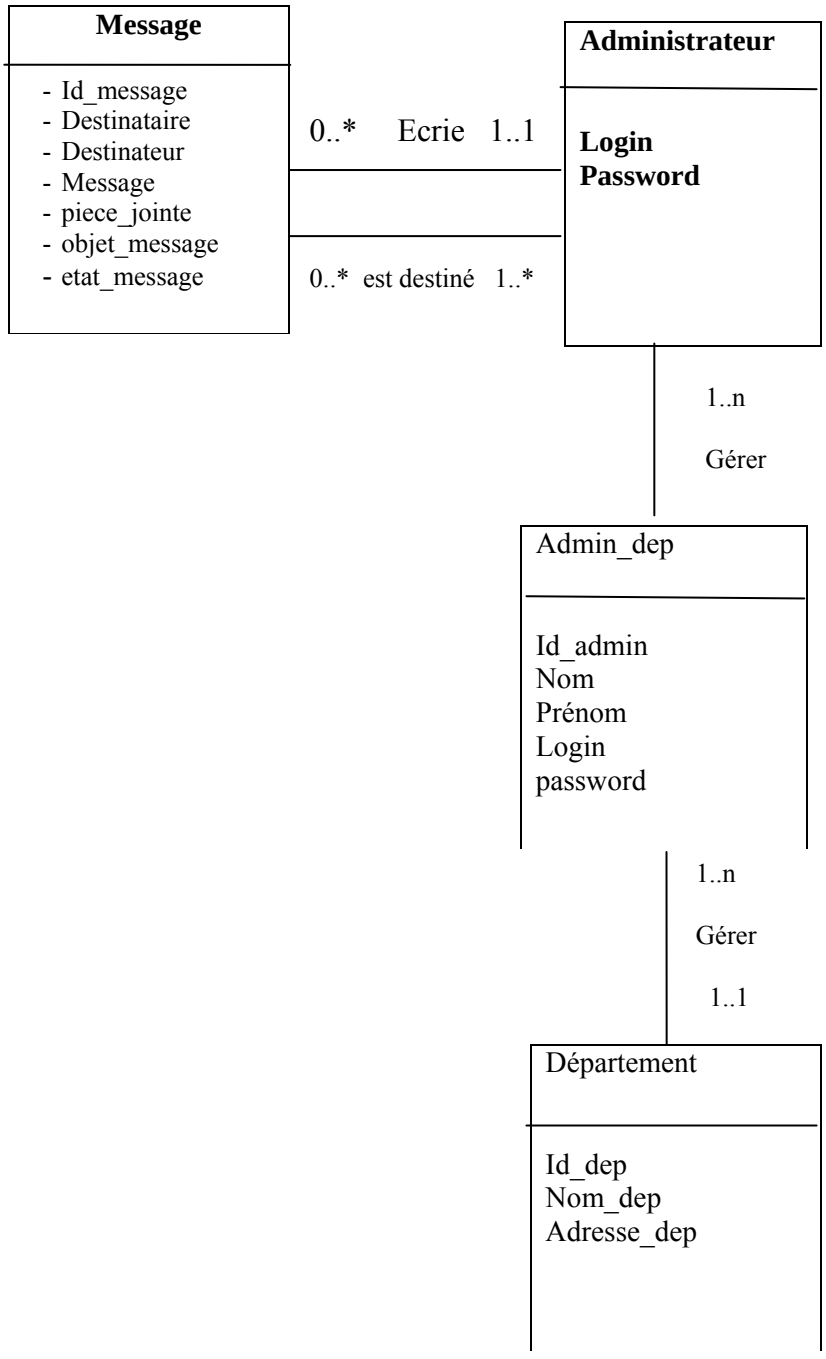


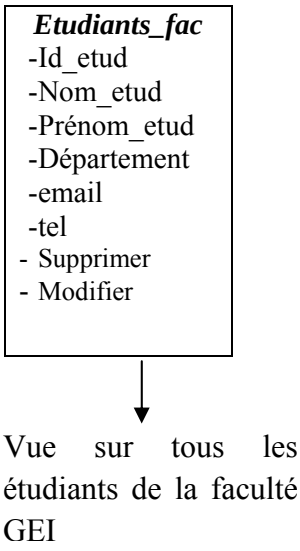
Figure III.15 : Diagramme du schéma logique du la faculté

III.3.4 : les Vues matérialisées créer :

La fédération des données ce fait en utilisant des vues matérialisées, c'est-à-dire que les données spécifiques à chaque site restant au niveau de la BDD de ce site, par contre les données communes entre les sites sont représentées par des vues matérialisées.

Dans notre cas les données communes entre le département et la faculté et la scolarité qui sont stockées ou niveau de la base de données département « informatique ou automatique », mais qui va être utilisée, d'où le doyen a besoin de voir l les étudiants ainsi leurs classement, donc on doit récupérer ces données de la base département sous forme de vue, qui sera manipulé par le doyen dans son processus métier.

Le table suivante présente un exemple de vue matérialisée créés :



Figue III.16 : exemple de vue matérialisé pour les étudiants de la faculté.

III.3.4 Exemple de modèles logiques de site de département :

Filière (id_filiere, désignation)

Spécialité (id_spec, désignation, $id_filiere)

Section (id_section, section, annee_cour,$id_spec)

Semestre (id_semestre, semestre, $id_section)

Groupe (id_group, groupe, $id_section)

Module (id_module, nom_module, coefficient, $id_semestre,$id_ensiegnant)

Enseignant (id_ensg, nom, prénom, date_naiss, adresse, département, faculté, mail, Tel, , login, pwd)

Etudiant (id_etud, nom, prénom, date_naiss, adresse, mail, Tel, login, pwd, $id_group)

Employ_temps (id_emploi, libelle,$id-section,$id-semestre)

Ensg_enseigne_module (id_module, id_ensg).

IV. Réalisation:

Après avoir présenté les différentes étapes de conceptions, nous allons passer dans cette partie à la traduction de ces dernières en code source.

Nous commençons d'abord par la présentation de l'environnement de développement et les outils qui serviront à la réalisation de notre exemple d'application, puis nous présentons l'application développé à travers quelques interfaces prise en captures d'écrans.

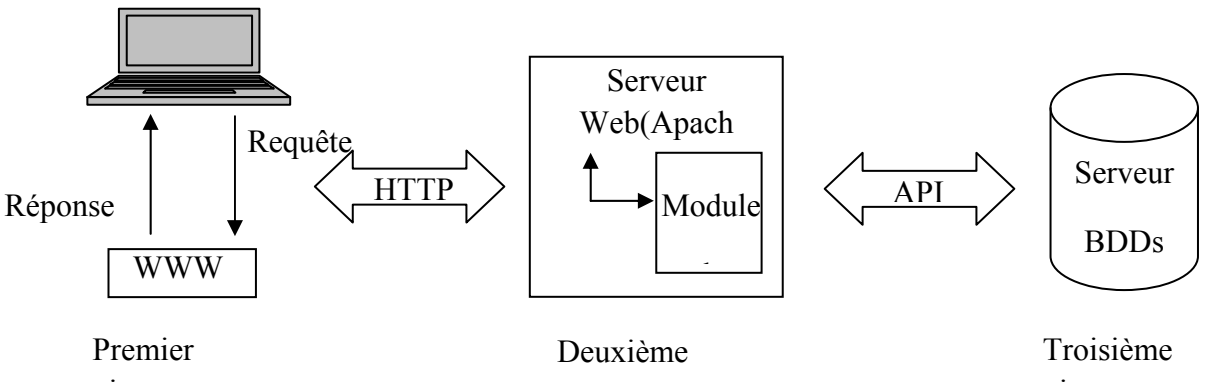


Figure III.17 : ***L'architecture du déploiement de l'application***

Premier niveau : il comprend le navigateur qui interprète et affiche les interfaces utilisateur relatives aux différents services de l'application.

Deuxième niveau : le deuxième niveau est constitué d'un serveur Web Apache doté de module PHP (l'interpréteur PHP est installé comme module Apache).

Troisième niveau : le troisième niveau est composé d'un serveur de base de données (MYSQL) performant souple et disposant d'un jeu de commande SQL large.

IV.1 Environnement de développement et d'implémentation:

- ✓ **Les langages de programmation *:***
  - ➢ ***HTML*** (**H**yper **T**ext**M**arkup**L**anguage) .
  - ➢ ***CSS (en anglais "Cascading Style Sheets", abrégé CSS).***
  - ➢ ***Langage de requête SQL.***
  - ➢ ***JavaScript.***
  - ➢ ***PHP (HypertextPreprocessor, Pré processeur Hypertexte PHP)*** .
- ✓ **Serveur web Apache.**
- ✓ ***Serveur MySQL.***
- ✓ ***PHPMyAdmin.***
- ✓ ***Les logiciels :***
  - ➢ ***WAMP (Windows, Apache, MySQL, PHP).***
  - ➢ ***Macromedia Dreamweaver 8.***

IV.2 présentation de quelques interfaces de notre application :

1. La page d'accueil :

Figure III.18 : Interface de la page d'accueil.

2. la page de département :

Figure III.19: Interface de la page du département informatique.

3. la page de l'administrateur de département:

Figure III.20 : Interface de la page d'administrateur de département.

Conclusion :

Ce chapitre est consacré à la réalisation d'un exemple de BDDR, j'ai présenté le processus de conception de mon application en deux niveaux, le niveau applicatif et le niveau de données. En premier lieu, j'ai commencé l'analyse et la conception par le niveau applicatif qui concerne les fonctionnalités et les traitements de l'application, ensuite j'ai passé au niveau de données qui ma a permis d'avoir le modèle logique de la base de données de chaque site et aussi j' ai présenté la procédure d'intégration de données d'une base à l'autre qui nous a permis de compléter le schéma globale.

En fin la traduction de cette conception en un ensemble de scripts, en ce qui concerne partie applicative.

Chapitre VI

Mise en œuvre d'une politique de sécurité dans les BDDRs.

Introduction

Pour atteindre ces objectifs de sécurité, il est nécessaire de mettre en œuvre une politique de sécurité, applicable à l'ensemble des entités à l'intérieur d'un domaine géographique ou fonctionnel. Cette politique désigne l'ensemble des lois et des consignes aux fins de protéger les ressources et les informations contre tout préjudice à leur confidentialité, leur intégrité et leur disponibilité, lequel serait dû à un usage inapproprié (incorrect, abusif ou frauduleux).

La politique exhibe, dans sa rédaction sous forme de règles, des sujets et des objets et précise les activités et opérations autorisées et interdites. Pour ce qui concerne la sécurité logique, il est essentiel de connaître la part de la politique de sécurité, traitée informatiquement et dévolue intrinsèquement au réseau et au système. Le reste de la politique sera pris en charge par des mesures non techniques, organisationnelles ou juridiques.

I. Schéma de la politique de sécurité :

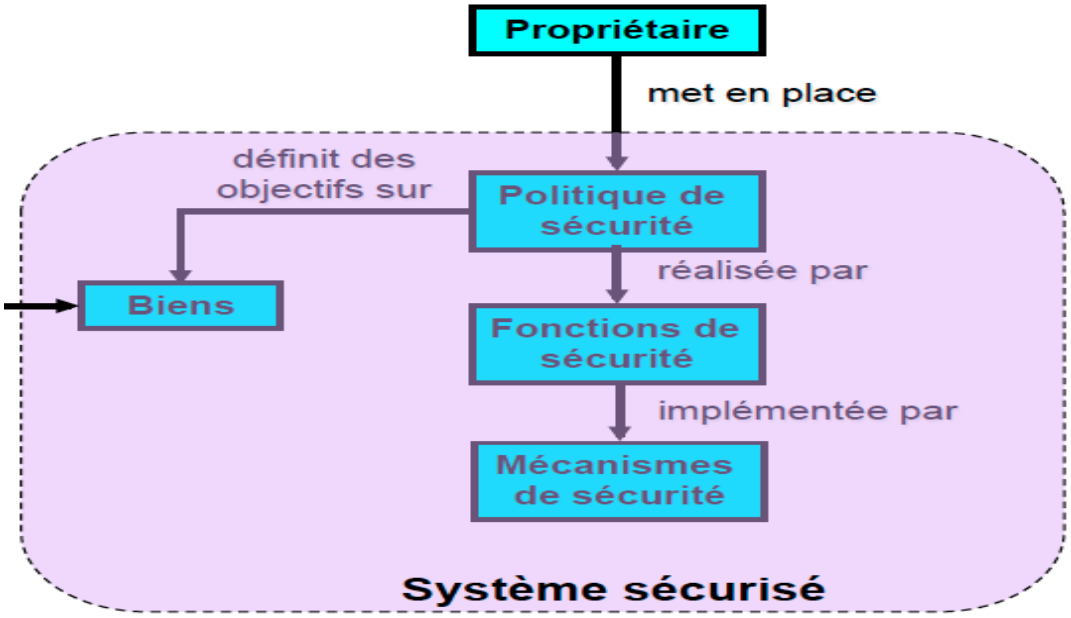


Figure VI.1 : Schéma de la politique de sécurité. [ ]

a) Politique de sécurité :

Ensemble des lois, règles et pratiques qui régissent la façon de gérer, protéger et diffuser les biens, en particulier les informations sensibles, au sein de l'organisation.

- Spécifie les autorisations, interdictions et obligations des sujets (agents, notion qui inclut à la fois les utilisateurs et les applications) qui peuvent accéder au système informatique.
- Inclut les aspects organisationnels, physiques et techniques.

b) Système sécurisé :

Système démarrant dans un état autorisé par la politique de sécurité, et ne pouvant pas entrer dans un état non autorisé.

c) Fonction de sécurité :

Mesure technique, susceptible de satisfaire un objectif de sécurité.

Exemple : classes de fonctions de sécurité dans les *Critères* Communs (CC)

FIA : Identification et Authentification

FTA : Accès à la cible d'évaluation

FAU : Audit de sécurité

FPR : Intimité ("privacy")

FCO : Sécurité des Communications

FDP : Protection des Données utilisateurs

… et d'autres (11 classes en tout)

d) Mécanisme de sécurité :

Logique, algorithme ou protocole qui implémente par matériel ou logiciel une fonction particulière dédiée à la sécurité ou contribuant à la sécurité.

Assure que le système ne rentre pas dans un état non autorisé.

Exemple : mécanismes d'authentification

- ✓ Mots de passe à usage unique.

- ✓ Biométrie.
- ✓ Protocole de type défi-réponse.

II. Sécuriser les flux de données

Il existe deux types de flux de données :

- **Les données entrantes** : dans un premier temps, nous allons devoir nous occuper de sécuriser les données qui proviennent du membre et donc de formulaires avant de pouvoir les entrer dans la BDD.
- **Les données sortantes** : puis, nous verrons comment sécuriser l'affichage des données issues de la BDD.

II.1. Protéger les données entrant dans les BDDs

L'attaque par injection SQL est très fréquente car elle est rapide à mettre en place, peut occasionner des dégâts irréversibles dans votre base de données ou, si elle est utilisée de manière plus subtile, elle permet de récupérer en toute discrétion les mots de passe et identifiants. Le pirate détourne votre requête en injectant du code dans les champs du formulaire : d'où le terme d'injection SQL.

1) Détournement de clause WHERE:

Exemple:

```
<?php
$requete = "SELECT * FROM admin
WHERE  pseudo = '".$_GET['pseudo']."'
AND password = '".$_GET['password']."' ";
?>
```

Après le détournement on aura :

```
<?php
$requete = "SELECT * FROM admin
WHERE  pseudo = '".$_GET['pseudo']."'
AND password = '' OR 1 = '1' ";
 ?>
```

2) Détournement de la clause DELETE

Exemple:

```
<?php
$requete = "DELETE FROM admin
```

```
    WHERE id = '".$_GET['id']."' ";
    ?>
```

Après le détournement on aura :

```
  <?php
    $requete = "DELETE FROM membre
    WHERE id = '1' OR id > '0' ";
    ?>
```

II.2 Protéger les données issues de la BDD ou de formulaires

Il existe un autre grand type d'injection connue des pirates, c'est l'injection HTML ou plus communément appelée *Cross-Site Scripting* (XSS).

Si les flux de données ne sont pas protégés à l'affichage, les pirates peuvent entrer des balises nocives et notamment du JavaScript. En bidouillant un peu, on peut facilement injecter de nouvelles balises et donc réécrire la page XHTML à notre guise. C'est pour ça qu'en français, on traduit l'acronyme XSS par injection HTML.

II.3. Les listes (contrôle des données) :

- **Liste blanche:**

Une liste blanche correspond à l'ensemble des données que l'internaute a le droit de rentrer. Instaurer une liste blanche permet un contrôle total sur les données transmises.

- **Liste noire :**

Contrairement à la liste blanche, la liste noire filtre les données interdites. On autorise donc tout, sauf certaines valeurs que l'on aura précisé. La liste noire permet généralement de lutter contre le *spam* et les robots. En effet, le but du *spam* est de promouvoir un produit ou un service en envoyant le plus de messages possibles et de manière à ce qu'ils soient visibles par le plus grand nombre. Vos forums ont peut-être déjà été la cible du *spam* et une bonne solution est de mettre en place une liste noire.

- **Liste grise :**

Du noir et du blanc sa donne gris ; la meilleure technique pour filtrer les données est d'utiliser en complément la liste blanche et la liste noire.

- une liste blanche qui autorise seulement des valeurs données.
- une liste noire qui filtre les données incorrectes.

III. Quelques mécanismes de sécurité :

III.1 Délégation :

La délégation est un mécanisme de sécurité qui permet à un sujet de déléguer ses permissions et droits à un autre sujet. Quand une entité est approuvée pour la délégation, elle peut représenter l'autre entité et faire appel à des services pour son compte. La délégation est utile si on veut optimiser le nombre d'identités stockées, ou éviter d'avoir un recours systématique à l'autorité de certification.

Il existe deux types de délégation :

– Délégation au niveau de l'**authentification** : Elle est définie si un mécanisme d'authentification fournit une identité différente de l'identité valide de l'utilisateur, à condition que le propriétaire de l'identité effective a déjà autorisé l'autre utilisateur à utiliser sa propre identité.

– Délégation au niveau du **contrôle d'accès** : Elle est réalisée quand un utilisateur délègue certaines de ses permissions à un autre pour accéder à une ressource.

La délégation veut dire "*attribuer à quelqu'un le pouvoir d'agir en tant que représentant*". Cela peut faire changer la matrice de contrôle d'accès. La délégation est réalisée grâce à un certificat SPKI (*Simple Public Key Infrastructure*) 1, représenté par 5 champs : C = (I, S, D, A, V)

– **Émetteur** (*I* pour *Issuer*) : l'autorité qui a créé et signé le certificat. Représentée par sa clé publique ou son *hash*.

– **Sujet** (*S pour Subject*) : Partie pour qui le certificat est délivré

– **Autorité** (*A* pour *Authority*) : Contenu sémantique spécifique à l'application représentant l'autorité.

– **Délégué ?** (*D* pour *Delegated* ?) : Est-ce que cette autorité dans le certificat peut être déléguée à quelqu'un d'autre ?

– **Validité** (*V* pour *Validity*) : Quand est-ce que le certificat est valide ? (période, URL d'un service de vérification en ligne...)

L'émetteur délègue le droit *A* au sujet *S*. Si *S* est une clé publique et si *D* est vrai, alors *S* peut déléguer ce droit à quelqu'un d'autre. La validité de la délégation est limitée par *V*. Le système d'exploitation des nœuds représente la seule source d'autorité primaire du système. La personne ou système qui installe le système d'exploitation pour la première fois a la possibilité de créer des délégations initiales de cette autorité (équivalent à établir un compte administrateur avec un mot de passe).

III.2 CAS: Community Authorization Service:

CAS (*Community Authorization Service*) est un service qui évalue les règles de la politique concernant la décision d'autoriser les actions dans un système, en se basant sur des informations sur le demandeur, la cible et la requête. Ce service autorise une politique flexible et expressive à être créée en prenant en considération plusieurs utilisateurs d'une même organisation. Dans le but de se connecter et d'utiliser une ressource, l'utilisateur doit réaliser les actions suivantes :

– Il s'authentifie à et reçoit les assertions exprimant la politique de la sécurite en termes d'utilisation des ressources.

– Il présente l'assertion à la ressource de la sécurité avec la requête d'utilisation

– En évaluant si la requête sera autorisée, la ressource vérifie la politique locale et la politique de sécurité exprimée dans l'assertion CAS.

III.3 PolicyMaker :

PolicyMaker est un moteur de gestion de confiance qui adresse le problème d'autorisation directement, plutôt que de le traiter indirectement via l'authentification et le contrôle d'accès. Dans le PolicyMaker, les créances et les politiques sont des assertions représentées comme des paires d'autorité source et de programme décrivant la nature de l'autorité à accorder et des parties à qui elle est accordée.

Les assertions peuvent être :

– Des assertions de politiques : *source = POLICY*. L'ensemble des assertions de politiques fournies au PolicyMaker représente la "racine de confiance" qui définit la décision sur la requête.

– Des assertions de créances : *source = clef publique de l'autorité émettrice*. Ces assertions doivent être signées par leurs émetteurs et ces signatures sont vérifiées avant l'utilisation des créances.

Avec PolicyMaker, c'est l'application qui est responsable de toutes les vérifications cryptographiques des signatures sur les créances et les requêtes. Le module de gestion de la confiance reçoit comme entrée le triplet (r,C,P), disant que l'ensemble des créances *C* contient une preuve que la requête *r* se conforme avec la politique P.

III.4 Authentification forte:

a) Authentification avec mysql « Session » :

Le principe de la session PHP permet de sauvegarder des données inter-pages. Grâce à ceci, nous comparerons les données recueillies avec la liste des User enregistrés dans MySql et leur mot de passe pour vérifier que la session existe et est valable et L'utilisateur est correctement authentifié et le couple Login/Mot de passe existe.

Exemple :

```
<?php
session_start();
$_SESSION['Login']=$_POST['Login'];
$_SESSION['Password']=$_POST['Password'];
HEADER('Location:Liste.php');
?>
```

b) Crypter les mots de passes dans une base de données :

Par sécurité, il est une bonne pratique qui consiste à crypter les mots de passe stockés dans une table de votre base de données. Ainsi, dans le cas ou une personne malveillante arriverait à consulter une table, elle ne pourrait pas voir le mot de passe, mais une suite de caractères dépourvue de sens. On utilisant des fonction en PHP comme MD5(*Message Digest 5*) cette fonction est un standard qui produit des empreintes de 128 bits.

IV. Solution proposé pour sécuriser une base de données répartie :

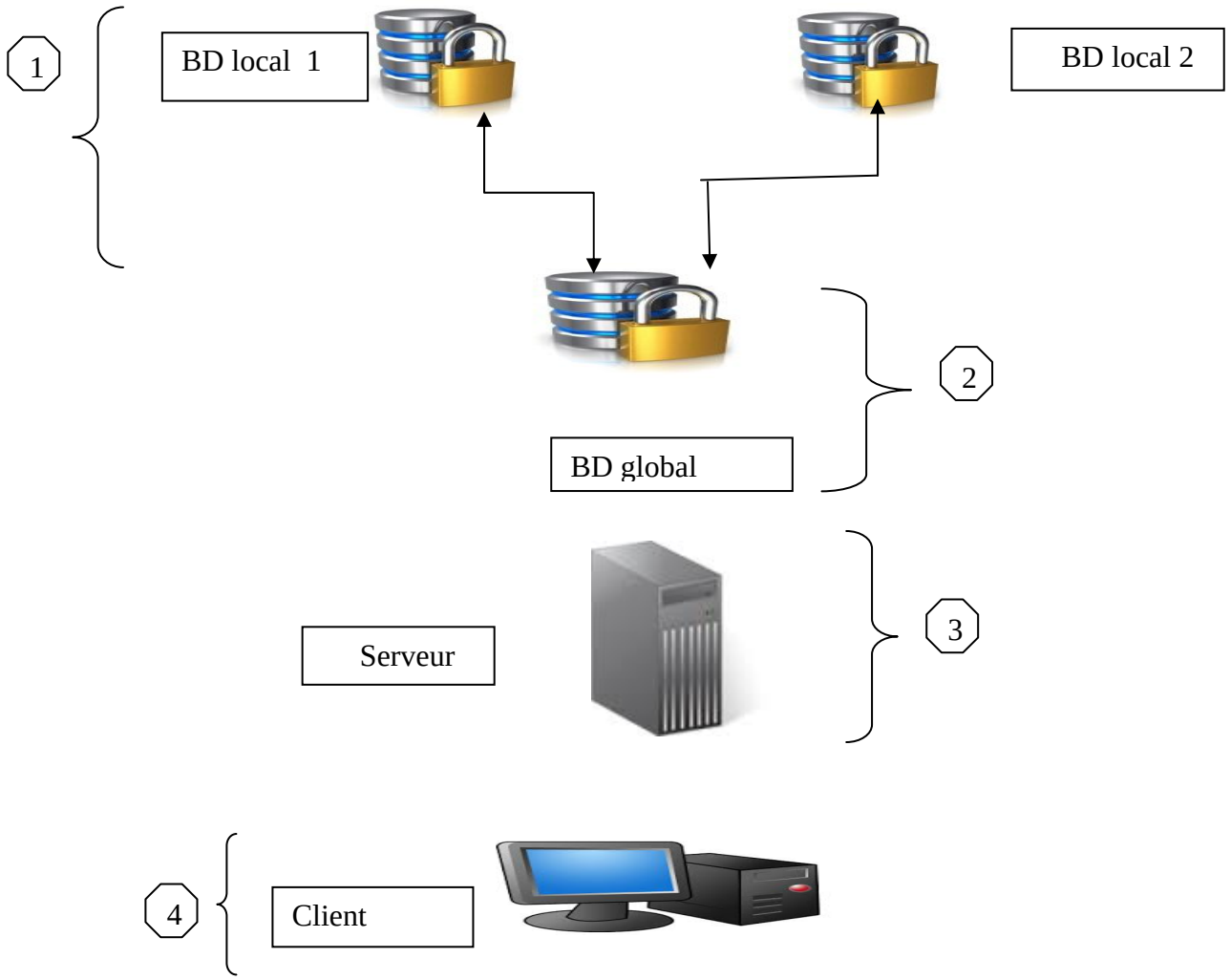


Figure VI.1: Solution proposé pour sécurisé une BDDR.

1 : Sécuriser au niveau du service de base de données locales.

2 : Sécuriser au niveau du service de la base de données global.

3 : Sécuriser au niveau du serveur.

4 : Sécuriser au niveau du client.

VI.1 Sécuriser au niveau du service de bases de données locales :

L'accès à chaque base de données local doit être sécurisé avec mot de passe. Ce mot de passe est saisi lors de sa création.

```
CREATE DATABASE 'informatique';
```

Mysql _connect ('localhost','informatique','root ','mot de passe') ;

VI.2 Sécuriser au niveau du service de bases de données global :

L'accès à la base de données doit être sécurisé avec mot de passe. Ce mot de passe est saisi lors de sa création.

```
CREATE DATABASE 'faculte' ;
```

Mysql _connect ('localhost','informatique','root ','mot de passe') ;

VI.3 Sécuriser au niveau du serveur :

Le Serveur doit être sécurisé de plusieurs façons et le choix pour notre application a été une configuration au niveau de Windows :

Lors d'installions le service Apache sous Windows fonctionne avec un compte « Système local ». Ce compte possède des droits élevés sur la machine mais aucun privilège sur le réseau. Afin d'éviter qu'un attaquant ne prenne possession de la machine hébergeant le service Apache, il peut être utile de faire fonctionner ce service avec un compte qui n'a aucun privilège sur cette machine.

VI.4 Sécuriser au niveau du client :

L'accès de client à l'interface du l'application doit être muni d'une sécurisation mot de passe et login.et limitation des privilèges(les droit d'accès a l'application) ;

```
GRANT Liste_de_permissions ON Liste_d_objets TO Liste_d_utilisateurs
[WITH GRANT OPTION];
```

Exemple :

Ne pas permettre a l'etudiant 'alex' de modifier ou bien inserer.

```
GRANT INSERT,UPDATE  ON etudiant TO  'alex' ;
```

Dans la solution proposé on s'intéresse au niveau des bases de données réparties de vérifier l'accès et sécuriser les requêtes manipuler entre la base de données globale et les bases de données locales on évite l'injection SQL

Conclusion :

La politique de sécurité permet de transcrire le travail de modélisation effectué pour comprendre les risques et leurs impacts, en des mesures concrètes de sécurité. Sa spécification est un des garants du bon dimensionnement des mesures de sécurité et d'une gestion efficace. Elle donne de la cohérence à la gestion et permet d'adopter vis à vis des risques et menaces, une attitude préventive et proactive et pas seulement réactive.

La bonne réalisation d'une politique de sécurité permet au mieux, de maîtriser les risques informatiques, tout en réduisant leur probabilité d'apparition. Toutefois, il ne faut pas perdre de vue que même un bon gestionnaire de la sécurité, tout en anticipant et prévenant certains accidents volontaires ou non, n'est pas devin. L'on évoque couramment l'intégrité des données, moins souvent celle des hommes. Nul service de sécurité, aussi perfectionné soit-il, ne tient si l'intégrité des administrateurs, responsables réseau, hommes systèmes ou utilisateur se trouve mise en cause. Ne perdons pas de vue que le maillon faible de la sécurité est l'homme.

Conclusion générale

Conclusion générale

Dans ce mémoire nous avons présenté une démarche pour mettre en œuvre la conception et l'implémentation d'une base de données distribuée pour l'espace numérique du travail de la faculté génie électrique et informatique on basant sur le mécanisme de sécurité. Et nous avons aussi proposé une organisation générale et fonctionnelle pour ce dernier.

Le travail réalisé nous a ramené à étudier plusieurs domaines dans le cadre de la sécurité informatique et notamment en base de données réparties.

Mon mémoire est articulé autours de deux parties ; la première est théorique, elle contient deux chapitres, le premier traite les bases de données réparties en général, le deuxième parle d'un domaine spécifique dans le cadre des bases de données réparties qui est la sécurité.

La seconde partie de mon mémoire est la partie pratique, dans laquelle nous avons implémenté une application d'une base de données répartie, une conception a été nécessaire pour mieux contourner toutes les fonctionnalités de chaque cas d'utilisation, aussi une proposition d'un schéma de politique de sécurité.

Au cours de l'élaboration de mon mémoire, j'ai acquis plusieurs connaissances qui s'avèrent bénéfiques dans le cadre de ma formation et qui sont un complément essentiel pour la consolidation de plusieurs données théoriques acquises tout au long de ma formation académique.

Ce travail peut être encore amélioré en envisageant les perspectives suivantes :

- Compléter le processus de fédération des bases de données par l'extension de cette dernière vers le rectorat et l'inclusion de toutes les facultés d'UMMTO.
- Mettre en place des mécanismes de sécurité plus , par exemple un crypto-système pour les transferts de données à distance.

Références Bibliographiques

Bibliographie

[1] : Bases de données réparties, Présentation Générale
Matthieu Exbrayat Licence Pro Informatique 2008-2009.

[2]: Systèmes de Gestion de Bases de Données Réparties & Mécanismes de Répartition avec Oracle (partie I : les BDDs réparties) Rim Moussa 2005-2006.

[3] : Mémoire Master II, Université A/Mira de Bejaia « Conception et réalisation d'une base de données repartie sous oracle : cas de l'hébergement des résidences universitaires »
Réalisé par Hakim MADI, année 2009.

[4]: Manipulations multibases et distribuées Partie 1*WitoldLitwin2012*.

[5]: Bases de données réparties, Architecteur Mise en œuvre, Duplication et Réplication

Réalisé par Michel Tuffery.

[6] : Sécurité des bases de données, Jacques le Maitre, année 2010

[7] : Nicolas Anciaux, « sécurité et bases de données », source de transparents : Luc bouganim, philipe pucheral, Fridiric, Cuppens.

[8] : http://en.wikipedia.org/wiki/SQL_injection.

[9] : Mémoire de fin d'études Master II « Etude de sécurité en base de données avec une application pour le contrôle d'accès »Réalisé par *EL HADJ MIMOUNE Khadidja.*

[10] **« Introduction à la cryptographie »,** support de cour du cabinet Hervé Schauer

Consultant , http://www.hsc.fr.

[11] : Solange Ghernaouti-Hélie « Sécurité Internet Stratégie et technologie » DUNOD 2000.

[12] : Mathieu_Blanc, « Sécurité des systèmes d'exploitation répartis : architecture décentralisée de métapolitique pour l'administration du contrôle d'accès obligatoire.», Thèse A L'UNIVERSITÉ D'ORLÉANS, Soutenue le : 19 décembre 2006 version 161 Mar 2010.

[13] : http://cyberzoide.developpez.com/securite/privileges-base-de-donnees/

[14] : *Eric Maiwald* **« Sécurité des réseaux »,** Edition CampusPress, 2001.

[15] : http://www.journaldunet.com/solutions/securite/analyses/07/0917-9-etapes-securiser-sgbd.shtml, 23 mai 2011.

[16] :http://cyberzoide.developpez.com/securite/privileges-base-de-donnees/

Bibliographie

[17]: Security for Grid services. In *Proceedings. 12th IEEE International Symposium on High Performance Distributed*
Computing, 2003.

[18] : Mémoire de fin d'études Master II « Sécurité des systèmes d'informations »Réalisé par DAHMANE MOURAD 2013/2014.

[19]: http://www.code.pediapress.com

Les articles :

[A1]Sécurité : les entreprises tentées par les solutions d'authentification multi-facteurs ? par Perrine Tiberghien, mai 2012. *MISC* (Multi-system & Internet Security Cookbook) Magazine sécurité informatique édité par Diamond Editions.
Site web: Http: //www.miscmag.com

[A2] : Sécurité : L'authentification à 2 niveaux de Paypal aisément contournée, le 05 aout 2014 ; société Paypal, Maryse Gros avec IDG (International Data Group)News Service ;édité par Mryse Gros

Site web: http://www.lemondeinformatique.fr

[A3] : massage d'erreurs attaque indirecte aux bases données ; juin 2008, PHP sécurité,édité par Antoine Cappelle .

Site web: http://www.webmaster-freelance.com