الجمهورية الجزائرية الديموقراطية الشعبية

République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

UNIVERSITE SAAD DAHLABDE BLIDA

FACULTE DE SCIENCES EXACTES

DEPARTEMENT D'INFORMATIQUE

Mémoire de projet de fin d'études
En vu d'obtention du diplôme
D'Ingénieur d'Etat en Informatique

Option : système d'information avancé

Présenté par

M^elle MADANI AMINA

Thème

Protection contre le piratage de logiciels fonctionnant dans une architecture client-serveur

Soutenu le 04 octobre devant le jury composé de :

M^me.BENSTITI	Présidente
M^r .BALA	Promoteur
M^r .HADJ YAHIA	Examinateur
M^me.MOKHTARI	Examinatrice

Année universitaire 2002-2003

Remerciements

Tous d'abord je remercie le bon dieu pour m'avoir guidé vers le bon chemin de la lumière et du savoir.

J'exprime mes sincères remerciements à mes parents qui m'ont beaucoup aidé durant mes études.

J'exprime particulièrement ma profonde gratitude et mes vifs remerciements à mon promoteur Mr.BALA pour son aide, sa patience, ses conseils, sa compréhensibilité et pour sa préoccupation.

J'adresse mes sincères remerciements au personnel de TELEMATIS.

Je remercie Mr.HADJ-YAHIA membre de la commission de suivi pour ses conseils.

Je tiens à remercier aussi les membres du jury pour l'honneur qu'ils m'ont accordé, en acceptant de juger mon travail.

Je remercie tous les enseignants de la faculté des sciences de BLIDA et surtout mes enseignants du département informatique.

Je remercie tout le personnel de la bibliothèque de la faculté des sciences pour leur aide durant mes cinq années d'études.

Je remercie tous ceux qui m'ont aidé de loin ou de prés, que tous ceux qui, de quelque manière que ce soit, ont œuvré à l'élaboration de ce travail, trouve ici l'expression de ma profonde gratitude.

Dédicace

Au nom de DIEU je dédie ce mémoire

à :

Mes très chers parents MENOUER et SAFIA, en signe de reconnaissance, de m'avoir transmis tout ce qui est valeureux : éducation, fierté, bonté, et sens du devoir ;
Qu'ils trouvent ici tout mon respect et mon amour.
Que dieu me les protège, et qui pour moi représentent les êtres les plus chers au monde.

Mes chers frères YOUNES et OTHMANE.

Toute ma grande famille.

Toute la section de 5ème année informatique.

Mes meilleurs amis frères et sœurs dans la foi.

Toute personne que j'aime et qui m'aime.

TABLE DES MATIERES

LISTE DES TABLEAUX

LISTE DES FIGURES

Résumé

La protection des logiciels est un créneau très important de la sécurité informatique ; C'est un souci du développeur informatique, protéger ses logiciels pour éviter un piratage de quelque ordre n'est pas chose facile. Bien évidemment, quand on parle de ce sujet, les gens pensent qu'ils ne seront jamais victimes de piratage mais c'est faux, on peut se faire pirater par hasard. En règle générale, les tentatives d'intrusions et de destructions se font à 80 % par les personnes de l'entreprise [Mie 98] qui veulent prendre leur revanche suite à un licenciement par exemple.

C'est au sein de l'entreprise qu'il faut prendre des précautions. Hormis l'équipement qui doit être placé dans des locaux protégés, il faut établir une stratégie en intégrant la protection dans le processus de développement du logiciel en intervenant sur les parties serveur et client. Il faut aussi instaurer une solution technique de protection lors du déploiement et lors de l'exécution du logiciel sur les postes clients.

Ce document nous permettra de découvrir les aspects les plus importants concernés par la protection des logiciels et en particulier ceux fonctionnant dans une architecture client-serveur.

Mots clés : Protection des logiciels, piratage, sécurité informatique, architecture client-serveur, serveur SQL, génie logiciel, cryptographie.

Abstract

Software protection is an important facet in computer security ; It is a real matter for the developer, protect its software to avoid hacking of some kind is not always easy thing. Obviously, when one speaks of this topic, people think that they will be never victims of piracy but this is false, one can be made pirate by chance. In general, the attempt of intrusions and destructions are done up to 80% by people of the company who want to take their revenge, for example following a layoff.

It is within the company that precaution should be taken. Except for the equipment which must be in protected premises, it is necessary to establish a strategy by integrating protection in the process of development of the software while intervening on the parts server and client. It is also necessary to use a protective technical solution during deployment and execution of the software on the client stations.

This document will permit us to discover the most important aspects concerned by the software protection and in particular those functioning in a client-server architecture.

Keywords : Software protection, hacking, piracy, computer security, client-server architecture, software engineering, SQL Server, cryptography.

Introduction générale

Le 2 février 1990, l'analyste du budget Marie Percham arrive à son travail, lorsqu'elle tape son code au digicode de la porte d'entrée et annonce son nom à l'interphone, la porte ne s'ouvre pas, après plusieurs tentatives, elle décide d'entrer coûte que coûte. Avec un collègue, elle parvient à ouvrir une porte de service avec une fourchette en plastic et un tournevis de poche qu'elle rangeait dans son sac à main. Elle a ainsi vaincu un système de sécurité vendu 44000 dollars soit disant inviolable [Mie 98].

Une définition très classique de la sécurité peut se présenter comme suit : situation dans laquelle quelqu'un, quelque chose n'est exposée à aucun danger ou aucun risque d'agression physique, d'accident, de vol, de détérioration. Le thème de la sécurité informatique est très large, il correspond à des concepts qui sont suffisamment formalisés de nos jours.

Aujourd'hui, l'informatique assure un rôle prédominant dans notre société. Ces dernières années toutes les entreprises désirant être compétitives, exposent commercialement des logiciels qui peuvent être copiés ou utilisés de manière illégale, en bref piratés. Ces logiciels ne devrant être utilisés que par les personnes concernées donc, il est nécessaire de mettre en place une protection efficace.

Or, il s'avère que le piratage des logiciels est devenu une sorte de sport international pour les informaticiens. Certaines de ces personnes ne le voient que comme un jeu, d'autres se livrent à des actes de vandalisme, d'autres encore font de l'espionnage industriel à des fins lucratives. De plus, il est de notoriété publique que ces informaticiens sont extrêmement compétents.

Dans le cadre de l'obtention du diplôme d'ingénieur d'état en informatique et en complément à nos connaissances théoriques, il nous a été confié un sujet de fin d'étude intitulé : «**Protection contre le piratage de logiciels fonctionnant dans une architecture client-serveur** » au sein du TELEMATIS.

Nous avons organisé notre mémoire de la manière suivante :

Un premier chapitre est consacré à une brève présentation du projet, nous y aborderons la problématique et les objectifs de notre sujet, ainsi qu'une représentation de l'organisme d'accueil.

Le deuxième chapitre concerne l'atelier du génie logiciel, il aborde le cycle de vie d'un logiciel et expose la notion de base de données et les architectures logicielles.

Le troisième chapitre présente un état de l'art des techniques et des méthodes de protection des logiciels contre le piratage tels que les moyens techniques et juridiques en détaillant les techniques de protection avec des exemples pour pouvoir choisir une meilleure technique.

Dans le quatrième chapitre, nous aborderons les principes de base de la cryptographie et les différents algorithmes utilisés actuellement.

Le cinquième chapitre est consacré à l'atelier de développement TELEMATIS, il semblait judicieux d'étudier son environnement et ses outils tel que le SQL Server.

Le sixième chapitre présente une stratégie de protection du logiciel dans l'entreprise durant ses étapes de développement.

Le dernier chapitre, présente une solution technique pour protéger des logiciels déjà développés et d'en faire un test, c'est la partie mise en oeuvre.

Nous achèverons ce document par une conclusion générale de notre projet et nous dégagerons les perspectives envisagées à cours et à moyen terme.

CHAPITRE I

Présentation du projet

Introduction

Dans ce premier chapitre, nous allons présenter premièrement notre projet en définissant d'abord la problématique de notre sujet et les objectifs à atteindre. La deuxième phase a pour finalité de définir l'organisme d'accueil.

1. Présentation du sujet

1.1. Problématique

La protection des logiciels contre le piratage constitue des préoccupations sérieuses et grandissantes pour les entreprises du monde entier. Tout développeur de logiciels veut empêcher toute utilisation non autorisée de son logiciel, parce qu'une bonne protection lui permettra de se prémunir contre la concurrence et d'augmenter significativement la rentabilité.

Notre problématique est : comment protéger contre le piratage des logiciels développés sur une architecture client-serveur et qui seront déployés et exécutés sur des postes clients?

1.2. Objectifs

Nous résumons les objectifs de notre projet par les points suivants :

- Le point de départ de tout modèle de sécurité consiste à mettre en œuvre un plan de sécurité cohérent destiné à protéger le logiciel de tout atteinte extérieure et de tout usage interne illicite. Une grande partie des données essentielles au développement des logiciels dans une entreprise peut être piratée si elle n'est pas protégée. La stratégie consiste à définir des règles et des recommandations bien précises concernant :

  - L'accès aux locaux.
  - La gestion des mots de passe.
  - Protection des documents d'analyse et conception.
  - Protection des bases de données.
  - Protection des codes sources.
  - Etc.

- Puisque le développement des logiciels dans notre contexte de travail se fait dans une architecture client-serveur, notre objectif est d'instaurer une technique de protection du logiciel lors de son déploiement et son exécution sur les postes clients en limitant :

  - Le nombre de déploiements.
  - Le nombre d'exécutions simultanées.
  - L'exécution sur certains postes clients bien identifiés.
  - Le nombre d'exécutions pour les versions d'évaluation.

2. Présentation de l'organisme d'accueil

Les nouvelles technologies de l'information et de la communication révolutionnent le monde avec l'avènement des autoroutes de l'information et l'explosion du multimédia. C'est l'interpénétration de l'informatique, des télécommunications et de l'audiovisuel

L'Algérie a pris la décision d'investir dans le marché des nouvelles technologies de l'information et de la communication. La valeur du marché annuel en Algérie est estimée à 100 Milliards de Dinars.

TELEMATIS (réseaux et télécommunication de l'entreprise) en tant qu'acteur principal et privilégié du secteur, compte bénéficier d'une importante part de ce marché.

2.1. Parcours

TELEMATIS contribue au développement du pays en matière des nouvelles technologies de l'information et de la communication depuis sa création en 1998 :

- ✓ Intégration première plate-forme logicielle Netscape/Real Audio – ENRS en 1999.
- ✓ Backbone fibre optique, 1440 postes – Ministère des Affaires Étrangères en 1999.
- ✓ Première Plate-forme Remote Access Server Cisco - Douanes Algériennes en 1999.
- ✓ Réseau WAN IP/X25 liaison internationale IP - Météo France Toulouse en 2000.
- ✓ Backbone fibre optique 700 postes, 15 centrales thermiques - SONELGAZ 2000.
- ✓ Backbone fibre optique 6Km - Entreprise Portuaire de Bejaia en 2000.
- ✓ Premier réseau national WAN IP/VoIP - Algérie Presse Service en 2000.
- ✓ Backbone fibre optique 12Km - Université d'Alger 2001.

2.2. Organisation

La société TELEMATIS est structurée de la manière suivante :

- Une Direction générale
- Une Direction d'administration et finances
- Une Direction formation et un centre d'examen Prometric
- Une Direction marketing & communication
- Une Direction Commerciale
- Une Direction Technique
- Une Direction développement logiciel
- Un bureau de transit

La ressource humaine est constituée de :

- 5 Consultants de haut niveau
- 3 Ingénieurs avant vente
  2 Ingénieurs réseau & télécoms
- 3 Ingénieurs développement logiciel
- 18 Techniciens « électriciens, câbleurs »

2.3. Services

Les principaux services de TELEMATIS sont :

- ***Système Information***

- Ingénierie des systèmes d'information
- Conception de bases de données
- Développement d'applications métiers
- Intégration des applications
- Transfert de compétences

- ***Télécommunication***

- Réseaux LAN, MAN, WAN
- Équipement de terminaison télécoms : Modem-Convertisseur RTC/LS/X25/IP/ATM
- Solutions voix/données
- Solution Vsat
- Solution IP Phone
- Solution IP TV
- VPN

- ***Ingénierie Télécommunication***

- Expertise
- Conseil & assistance
- Audit & analyse de réseaux
- Transfert de compétences
- Maquettage & validation d'architecture
- Télémaintenance
- Certification & recette

- ***Système de câblage structuré***

- Câblage cuivre UTP-FTP, Cat. 5-6-7 (100, 200, 300, 600 MHz)
- Backbone Fibre optique multi-mono mode
- Câblage téléphonique
- Câblage électrique VDI

- ***Sécurité***

- Audit des procédures d'exploitations
- Conseil en architecture de sécurité complexes
- Analyse & dépannage de réseaux
- Mise en œuvre de solutions firewalling
- Authentification, cryptage
- Anti-Virus on-line

- ***Serveurs***

- Serveur de calcul
- Serveur de données & stockage
- Solution RAID
- Clustering
- NAS

- ***Systèmes et logiciels***

- Systèmes d'exploitation : WINDOWS 2000, UNIX, LINUX
- Management : Administration de réseau, Sauvegarde centralisée, Analyse de réseau
- Passif : Gestion de pré câblage, EDITH transfert de fichiers automatisés
- Messagerie électronique
- Serveur télécopie

- ***Archivage électronique***

- Scanner, Scanner haut débit
- Scanner microfiche
- Gestion des flux & processus
- Indexation & recherche documentaire
- GED bilingue (arabe & latin)
- Moteur Excalibur

- ***Formation***

- Cours intra-entreprise
- Cours extra-entreprise
- Cursus de formation : Unix, Cisco, Microsoft, Technologie objet, État de l'art
- Plan de formation spécifique à la demande
- Formateurs certifiés
- Formation 100% pratique

CHAPITRE II

Le génie logiciel

Introduction

Compte tenu du rôle pivot que joue le génie logiciel, il est important de le clarifier et le cerner avant de parler sur la protection du logiciel. C'est pourquoi dans ce chapitre, plusieurs concepts seront survolés ; Il s'agira notamment de : Génie Logiciel (GL), Atelier de Génie Logiciel (AGL), Bases de Données (BD) et les Architectures des applications.

1. Génie Logiciel (Software Engineering)

Avant de définir le génie logiciel, il est nécessaire de définir le logiciel :

On regroupe sous le terme de logiciel les différentes formes des programmes qui permettent de faire fonctionner un ordinateur et de l'utiliser pour résoudre des problèmes [Mey 86]. En outre le terme logiciel ne désigne pas que les programmes associés à telle application ou tel produit, il comprend la documentation nécessaire à l'installation, à l'utilisation, au développement et à la maintenance [Som 88].

On appelle Génie Logiciel l'application de méthodes scientifiques au développement de théories, méthodes, techniques, langages et outils favorisant la production de logiciel de qualité [Mey 86].

Le génie logiciel est l'art de spécifier, de concevoir, de réaliser et de faire évoluer, avec des moyens et dans des délais raisonnables, des programmes, des documentations et des procédures de qualité en vue d'utiliser un ordinateur pour résoudre certains problèmes [Gau 96].

2. Atelier de génie logiciel (AGL)

On appelle Atelier de Génie Logiciel un ensemble d'outils intégrés, couvrant une partie significative du cycle de vie et qui mettent en oeuvre le Génie Logiciel [Brè 94].

2.1. Cycle de vie d'un logiciel

Le cycle de vie d'un logiciel est une modélisation conventionnelle de la succession d'étapes par lesquelles passe un logiciel, de la conception à la maintenance [Brè 94].

Royce (1970) est le premier qui a proposé un modèle du cycle de vie [Roy 70]. Depuis, de nombreuses améliorations et modifications y ont été apportées. L'esprit de ces divers modèles peut cependant être résumé par le modèle le plus classique qui est le modèle en cascade représenté dans la figure suivante :

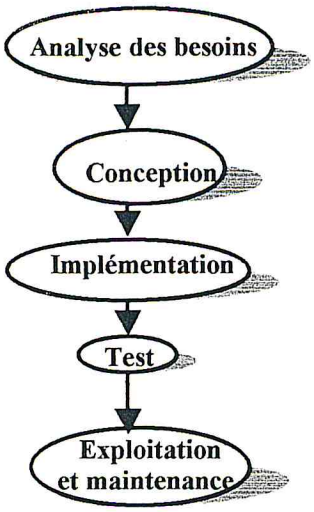

Figure II.1 : Cycle de vie d'un logiciel

Le tableau ci-dessous montre les étapes de conception des logiciels ainsi que les principaux résultats attendus de chaque étape :

Tableau II.1 : Les principales étapes d'une démarche de conception

ETAPE	RESULTAT
Analyse des besoins	-Définition du problème -Analyse grossière : coûts/bénéfices -Affiner objectifs et portée -Cahier de charges
Conception générale	-Décomposition du logiciel en module -Description des fonctions de chaque module
Conception détaillée	-Spécification des modules -Description détaillée des structures des données et des modules
Implémentation	-Programmes -Tests -Manuels de programmation, d'utilisation
Maintenance	-Correction -Perfection -Adaptation

2.1.1. Analyse des besoins

L'analyse des besoins est une étape essentielle au début du processus de développement, son but est d'éviter de développer un logiciel non adéquat. Une définition compréhensible du problème devra être produite. A partir de cette définition on pourra concevoir et réaliser le logiciel. Le résultat de cette étape est une description précise qui est appelée le document de définition des besoins (cahier de charges). Les besoins doivent être définis par les utilisateurs et par l'équipe de développement.

L'organisation possible de la définition des besoins est la suivante [Som 88] :

a. *Introduction :* Elle doit décrire brièvement les fonctions du logiciel et elle doit préciser la structure du document et décrire les notations utilisées.

b. *Matériel :* Si le logiciel doit être réalisé sur un matériel spécifique, il faut décrire ce matériel et ses interfaces.

c. *Modèle conceptuel :* C'est le modèle conceptuel sur lequel les besoins sont basés, c'est une vue des fonctions majeures du logiciel et de leurs relations. Il est décrit par des notations graphiques telle que les diagrammes de flots de données.

d. *Besoins fonctionnels :* Ce sont les services fournis à l'utilisateur. Suivant la nature de ces besoins, la notation utilisée peut être un langage naturel, un langage semi formel, un langage formel ou un mélange de ces notations.

e. *Besoins en matière de base de données :* L'organisation logique des données manipulées par le logiciel et leurs relations doivent être décrites ici.

f. *Besoins non fonctionnels :* Les contraintes sous lesquelles le logiciel doit opérer, doivent être exprimées et reliées aux besoins fonctionnels.

g. *Information destinée à la maintenance :* Cette section doit décrire les hypothèses sur lesquelles le système est fondé ainsi que les changements prévus du fait de l'évolution du matériel et l'évolution des besoins des utilisateurs

h. *Glossaire :* Il doit définir les termes techniques utilisés dans le document.

i. *Index :* Il peut être souhaitable de fournir plus d'une seule sorte d'index (index alphabétique, index des fonctions, index par chapitre…).

2.1.2. Conception

C'est la phase la plus importante du processus de développement d'un logiciel, elle consiste à enrichir la description du logiciel de détails d'implémentation afin d'aboutir à une description très proche d'un programme [Brè 94].

Elle se déroule pendant deux étapes : la conception générale et la conception détaillée. La conception générale a pour but de décomposer le logiciel en modules et de préciser les interfaces et les fonctions de chaque module. A l'issue de cette étape, on obtient une description de l'architecture du logiciel et un ensemble de spécifications de ses divers composants. La conception détaillée fournit pour chaque module une description détaillée de la manière dont les fonctions du composant sont réalisées : algorithmes, représentation des données.

- **Les méthodes de conception**

Il est facile de concevoir un logiciel lorsqu'on utilise une méthode de conception, parmi les méthodes de conception les plus utilisées on trouve :

a. *Approche cartésienne (analytique, fonctionnelle)* : Elle propose de décomposer une fonction de façon hiérarchique en un ensemble de modules jusqu'à atteindre un niveau de décomposition fin en précisant les liens entre les modules [Gau 96]. SADT (Structured Analysis and Design Technique) est un exemple de méthode cartésienne.

b. *Approche systémique* : Le logiciel est perçu comme un objet complexe actif dont il faut décrire sa structure et sa fonction, il est élaboré des points de vue modélisation de données et modélisation de traitements. Les méthodes les plus répandues sont : Merise, SSADM (Structured Systems Analysis Design Methods) et IE (Information Engineering).

c. *Approche orientée objet :* Le logiciel est vu comme une collection d'objets communiquant entre eux par des messages. A chaque objet on peut associer un ensemble d'opérations. Les méthodes les plus connues sont : OOD (Object Oriented Design), HOOD (Hierarchical Object Oriented Design).

2.1.3. Implémentation

A ce niveau, il s'agit de l'implémentation de la solution retenue au niveau de la phase de conception, d'écrire les programmes nécessaires pour l'implémentation et de faire des tests sur ces programmes.

a. *Programmation* : Cette activité consiste à passer du résultat de la conception détaillée à un ensemble de programmes ou de composants de programmes. Une des décisions la plus importante dans l'implémentation d'un logiciel est le choix du langage et outils de programmation. Le langage choisis devrait permettre d'utiliser des noms significatifs, il devrait y avoir des procédures, des fonctions et des contrôles. D'autres critères doivent être pris en compte :

- Les souhaits des clients,
- La disponibilité d'un compilateur,
- La disponibilité d'outil logiciel pour aider le développement du programme,
- La taille du projet,
- L'expérience de l'équipe de programmation,
- Le besoin de portabilité, et
- Le type de l'application.

b. *Test du logiciel* : L'objectif du test est de détecter les erreurs, la mise au point a pour but de localiser ces erreurs et de les corriger. Les systèmes logiciels sont construits en assemblant des sous systèmes qui sont eux-mêmes des assemblages de modules, qui sont eux-mêmes des assemblages de procédures (voir figure ci-dessous); Les tests se feront par étape.

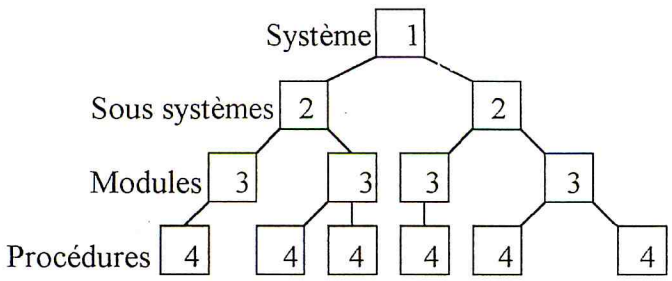

Figure II.2 : Décomposition d'un système logiciel

On distingue cinq étapes dans ce processus [Som 88] :

a. *Test unitaire :* Permet de vérifier le fonctionnement correct des procédures et fonctions constituant un module.

b. *Test de modules :* Un module est constitué d'un certain nombre de fonctions, il est nécessaire de vérifier la bonne coopération de ces fonctions.

c. *Test de sous systèmes :* Un sous système est constitué d'un certain nombre de modules qui communiquent et coopèrent, il est nécessaire de vérifier la bonne conception des interfaces entre modules.

d. *Test d'intégration :* Il est réalisé lorsque le système complet est assemblé. A cette étape les tests permettent de vérifier que le logiciel réalise bien les fonctions spécifiées lors de l'analyse et de la définition des besoins.

e. *Test d'acceptation :* Il permet de tester le système avec des données réelles.

2.1.4. Maintenance

Le terme maintenance a été appliqué au processus de modification d'un logiciel après sa livraison, ces modifications peuvent être de simples changements destinés à corriger les erreurs pendant toutes les phases du développement du logiciel [Gau 96]. La maintenance d'un logiciel peut être divisée en trois parties :

a. *La maintenance perfective* : Consiste à effectuer les changements demandés par l'utilisateur.

b. *La maintenance adaptative* : Consiste à adapter le logiciel au changement de son environnement.

c. *La maintenance corrective* : Consiste à corriger les erreurs.

2.2. Interface utilisateur

L'interface utilisateur est l'élément de référence qui permet à l'utilisateur de juger de la qualité du logiciel. Un logiciel qui offre une interface utilisateur mal conçue est susceptible d'être rejeté, quelles que soient les fonctionnalités offertes. Une interface mal conçue peut être la cause d'erreurs catastrophiques de la part de l'utilisateur. Si une information est présentée à l'utilisateur de façon confuse ou trompeuse, il peut accidentellement ne pas comprendre le sens et de ce fait, initier une suite d'actions dangereuses [Som 88]. On peut distinguer deux types d'interfaces :

a. *Interface interactive :* C'est une interface où l'utilisateur interagit directement avec l'ordinateur à l'aide d'un terminal.

b. *Interface non interactive :* C'est une interface où l'utilisateur doit préparer à l'avance les données d'entrée sous une forme lisible par la machine, les résultats associés aux données d'entrée sont ensuite envoyés à l'utilisateur.

- **Numéros de modèle**

Chaque clé SentinelPro est identifiée par un numéro de modèle composé d'un identificateur de série, d'un code de famille et d'un identificateur de ligne de réponse [Sen94].

a. *L'identificateur de série :* est une chaîne alphanumérique de 4 caractères. Actuellement, toutes les clés SentinelPro font partie de la série 4000 ; Cela signifie que les quatre premiers caractères du numéro de modèle sont compris entre 4000 et 4ZZZ.

b. *Le code de famille :* à deux lettres permet au driver d'activer la clé SentinelPro lorsqu'une interrogation lui est envoyée. Si plusieurs clés sont empilées sur le même port, elles doivent avoir des codes de familles différents pour permettre au driver de les distinguer.

c. *L'identificateur de ligne de réponse :* identifie la ligne sur laquelle la clé répond. Si le numéro de modèle se termine par -B, la clé répond à la ligne BUSY. Si le numéro de modèle se termine par -A, la clé répond à la ligne ACK.

Exemple : Le numéro de modèle 4123BH-B fait partie de la série 4000, a un code de famille BH et répond à la ligne BUSY.

- **Verrous logiciels**

Pour protéger une application, il faut intégrer une série de verrous logiciels. Ces verrous nécessitent la présence de la clé physique SentinelPro pour déverrouiller le logiciel et permettre la poursuite de l'exécution. Chaque verrou est mis en oeuvre en trois temps : interrogation de la clé SentinelPro, test de la valeur retournée et exécution du traitement approprié.

a. *Interrogation de la clé SentinelPro* : Le verrou logiciel débute par l'appel d'une routine d'interface SentinelPro fournie par Rainbow Technologies. Cet appel est une "interrogation". La routine d'interface accepte une chaîne de caractères et l'envoie à la clé physique SentinelPro. La clé SentinelPro transforme les données qui lui sont envoyées selon un algorithme interne (stocké dans la clé), et retourne la valeur ainsi cryptée à l'application.

b. *Test des valeurs retournées* : L'application évalue ensuite les données retournées. Une réponse correcte indique qu'une clé SentinelPro dotée du bon algorithme est connectée à l'ordinateur. Si une valeur incorrecte est retournée, la clé SentinelPro connectée n'est pas la bonne ou une erreur s'est produite (due par exemple à du bruit sur la ligne). Lorsque l'application évalue une valeur de réponse issue de la clé SentinelPro, elle doit toujours comparer la réponse à la valeur attendue.

- **Inconvénients de SentinelPro**

SentinelPro comme tous les dongles, reste le moyen le plus utilisé dans plusieurs entreprises pour s'assurer de la protection de ses logiciels et cela ne signifie pas que SentinelPro n'a pas d'inconvénients, par contre leur coût d'acquisition qui est très élevé est un majeur inconvénient. La défaillance d'une clé à un moment donné peut poser pas

3. Base de données et AGL

Beaucoup de grands logiciels nécessitent l'existence d'une base de données, le logiciel y trouve ses informations et y ajoute d'autres pendant l'exécution. Pour l'obtention d'un logiciel de meilleure qualité et une productivité améliorée, il faut utiliser à la fois un SGBD et un AGL.

3.1. Notion de base de données

On peut définir une base de données comme une collection de données opérationnelles, enregistrées (sur un support adressable) et utilisées par des systèmes d'application (les programmes) d'une organisation (humaine) particulière. En outre, la collection de données est structurée indépendamment d'une application particulière, elle est cohérente, de redondance minimale et accessible simultanément par plusieurs utilisateurs [Bou 99].

3.2. Système de gestion de base de données (SGBD)

Un SGBD est un ensemble d'outils logiciels permettant la création et l'utilisation de bases de données [Bou 99].

3.2.1. Les fonctions d'un SGBD

Tout SGBD doit comprendre les fonctions suivantes :

a. *La description des données :* Le SGBD doit offrir à l'utilisateur un outil pour décrire des « objets », leurs attributs, leurs liens ainsi que des contraintes éventuelles pouvant concerner ces objets, leurs attributs ou leurs liens. Cet outil est un langage appelé Langage de Description des Données (*LDD*).

b. *La manipulation des données :* Elle concerne les outils et les mécanismes qui permettent à l'utilisateur d'interagir avec la base de données. Les SGBD offrent sous diverses formes des capacités de recherche, de création, de modification et de suppression d'informations. Une de ces formes est le Langage de Manipulation de Données (*LMD*). L'action à effectuer sur la base est exprimée comme une phrase de ce langage (requête) qui est évaluée et exécutée par le SGBD. Les programmes d'application peuvent être écrits dans un langage autre que celui associé au SGBD, c'est un langage hôte (Pascal, C, C++, Java, ...).

c. *L'intégrité des données :* Le concept d'intégrité des données est relatif à la qualité de l'information enregistrée. Pour être fiable et cohérente, celle-ci doit parfois vérifier certaines propriétés, comme l'appartenance à une liste de valeurs permise pour un attribut. Ces propriétés sont appelées *contraintes d'intégrité*. Certaines sont spécifiées lors de la définition du schéma de la base, le SGBD se chargeant de les préserver pendant toute la vie de la base, alors que d'autres peuvent nécessiter un effort de programmation.

d. *La sécurité de fonctionnement :* Le SGBD doit offrir des mécanismes permettant de remettre la base de données dans un état opérationnel en cas d'incident matériel ou logiciel. Ces mécanismes sont basés sur la journalisation des opérations réalisées sur la base et leur ré-exécution automatique en cas de besoin.

e. *La confidentialité :* Au moment où la base de données devient une unité communautaire à laquelle peuvent accéder plusieurs utilisateurs, un aspect très important apparaît et nécessite une prise en charge par le SGBD, il s'agit des *droits d'accès* des différents utilisateurs à la base de données. Pour ce faire, les utilisateurs sont classés en groupes selon la nature d'activité et le travail qu'ils ont à réaliser avec les données. A chaque groupe sera attribué un nom et un mot de passe grâce à quoi le SGBD pourra identifier l'utilisateur et l'orientera alors vers la partie de la base de données à laquelle il est habilité à accéder.

Ces différentes fonctions sont illustrées par la figure suivante :

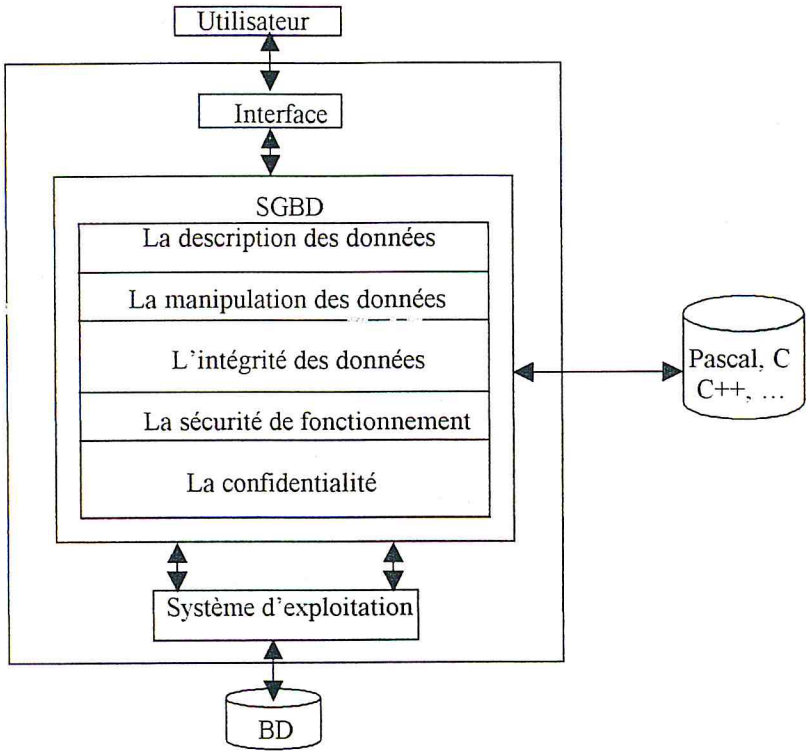

Figure II.3 : Les fonctions d'un SGBD

3.2.2. Typologie des SGBD

Les modèles actuels de représentation des données sont :

a. *Le modèle relationnel et les SGBD relationnels :* Ce modèle permet de voir une base de données comme un ensemble de tables. Les concepts utilisés par ce modèle sont issus des notions mathématiques très simples tels que : ensembles, relations, domaines, ...etc. Les SGBD relationnels offrent un langage de manipulation de données standard : SQL (Structured Query Language) qui est lui-même fondé sur l'algèbre relationnelle.

b. *Les modèles orientés objets :* Nombre de travaux ont visé à intégrer des concepts issus des langages à objet (notions de classes et sous classes...) et des concepts issus des bases de données. OQL (Object Query Language) est un langage d'interrogation de base de données objets, basé sur des requêtes proches de celles de SQL.

4. Choix d'une architecture d'application

L'architecture qu'on sélectionne détermine le développement, le déploiement et la gestion de l'application logicielle. Pour implémenter des applications client-serveur, on a le choix entre plusieurs architectures. On peut répartir les applications en trois couches logiques, qui peuvent résider physiquement sur un ou plusieurs serveurs :

a. *Présentation* : Comprend la logique de présentation des données et de l'application aux utilisateurs. Cette couche est pratiquement toujours implémentée sur un ordinateur client.

b. *Entreprise* : Comprend la logique de l'application et les règles d'entreprise. Le SGBD peut intervenir au niveau de cette couche.

c. *Données* : Comprend la définition de la base de données, la logique d'intégrité des données, les procédures stockées et toutes les autres opérations étroitement associées aux données. Le SGBD intervient principalement au niveau de cette couche.

- **Conception architecturale**

La figure suivante montre les différentes architectures d'application logicielle :

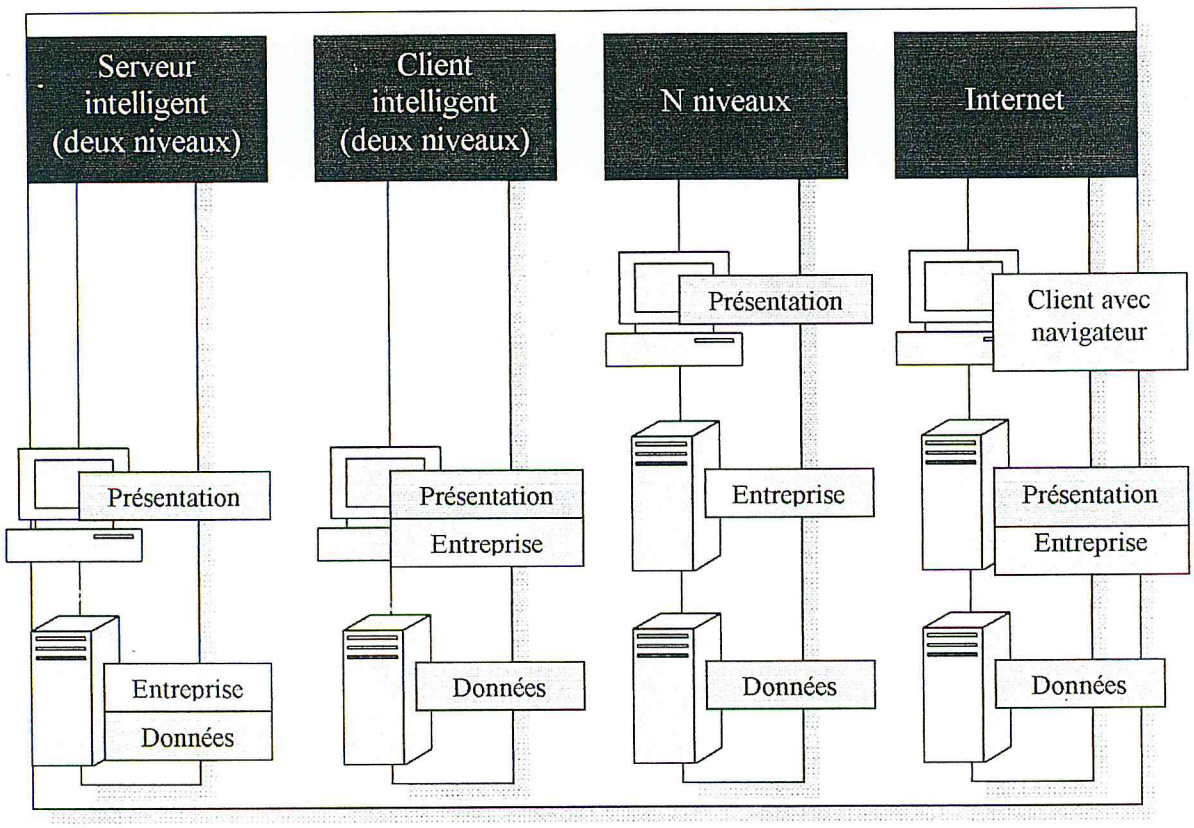

Figure II.4 : Les architectures d'application [Mic 00]

Les options standard de déploiement d'applications comprennent les éléments décrits ci-dessous :

a. *Serveur intelligent (deux niveaux)* : La plus grande partie des traitements a lieu sur le serveur, et les services de présentation sont traités sur le client. Dans de nombreux cas, l'essentiel de la logique des services d'entreprise est implémenté dans la base de données. Cette conception est utile lorsque les clients ne disposent pas de ressources suffisantes pour traiter la logique d'entreprise. Toutefois, le serveur peut devenir un goulet d'étranglement, car la base de données et les services d'entreprise sont en compétition pour les mêmes ressources matérielles. Les applications d'entreprise fondées sur une approche de base de données constituent des exemples de cette conception.

b. *Client intelligent (deux niveaux)* : La majorité des traitements a lieu sur le client, et les services de données sont traités sur le serveur. Cette conception est largement utilisée. Les applications développées pour de petites entreprises avec des produits tels que Microsoft Access constituent des exemples de cette conception.

c. *N niveaux* : Le traitement est réparti entre un serveur de base de données, un serveur d'application et des clients. Cette approche sépare la logique des services de données, et on peut aisément ajouter des serveurs d'application ou de base de données. Les applications d'entreprise multi-niveaux et les applications développées avec des moniteurs transactionnels constituent des exemples de cette conception.

d. *Internet* : Le traitement est réparti sur trois couches, les services d'entreprise et de représentation résidant sur le serveur Web et les clients utilisant de simples navigateurs.

Conclusion

Il est clair à partir des concepts présentés précédemment que le logiciel prend dans la plupart des cas une dimension très complexe, surtout lorsqu'il s'agit d'une architecture client/serveur ou à N niveaux. Dans un contexte pareil, un souci mérite d'être soulevé à ce niveau qui est la sécurité et la protection des différents objets du système : base de données, application, autorisation d'accès, ...etc.

CHAPITRE III

Les moyens utilisés pour la protection des logiciels contre le piratage

Introduction

Une entreprise vient de développer un nouveau logiciel et une personne de l'entreprise ou de l'extérieur souhaite l'exploiter commercialement, il est donc nécessaire de le protéger. En effet une bonne protection lui permettra de se prémunir contre la concurrence et d'empêcher toute utilisation non autorisée de son logiciel. L'objectif de la protection des logiciels est d'augmenter significativement la rentabilité.

Ce chapitre identifie et définit les différentes formes de piratage informatique, il présente un état de l'art des différents moyens techniques et juridiques utilisés pour la protection des logiciels contre le piratage ainsi que les avantages et les faiblesses de chacune d'elle.

1. Le piratage informatique des logiciels

▪ Définition du piratage

C'est un mot couramment utilisé afin de désigner l'action de duplication, de copie non autorisée ou la contrefaçon d'un logiciel.

Il est autorisé de créer une copie de sauvegarde d'un logiciel, utilisable seulement en cas d'altération ou de détérioration du support original. Ainsi, toute autre copie d'un logiciel ou de sa documentation est sans conteste une contrefaçon.

Celle-ci peut revêtir de multiples formes :

- La copie non autorisée.
- L'utilisation sans droit d'un logiciel d'exploitation.
- La location, la cession ou le prêt non autorisés.
- L'adaptation, l'incorporation ou la reproduction, sans droit, de données.

Le piratage informatique constitue un véritable fléau pour les auteurs et distributeurs de logiciels.

▪ Mythe ou réalité ?

Selon le CSI (Computer Security Institute), il y a eu plus de 60 % d'attaques entre 1985 et 1993 dans le monde. On a détecté 7860 intrusions qui ont réussi sur 8932, seuls 19 ont été signalées par peur d'avouer leurs défaillances en matière de sécurité [Mie 98].

L'étude conduite par IPR (International Planning and Research), dans 85 pays d'Europe, révèle que le piratage dans le monde augmente pour la deuxième année consécutive, passant de 37 % en 2000 à 40 % en 2001, avec pour conséquence des pertes s'élevant à près de 11 milliards de dollars (12 milliards d'euros) directement dues au piratage des logiciels et 50 % des logiciels installés dans le monde sont des copies illégales [Mic 01].

Donc, il faut prendre en compte que la menace n'est pas un mythe mais bien une réalité !

2. Les différentes formes de piratage

Il existe plusieurs formes de piratage [Mic 01] :

a. *Duplication de logiciels originaux à l'aide de graveurs* : Le piratage est dû essentiellement à l'apparition sur le marché de graveurs de CD vierges à bas prix. Ces appareils sont détournés de leur vocation initiale et servent à dupliquer illégalement des logiciels. Or, beaucoup de gens ignorent que les copies réservées à un usage privé, ne sont pas autorisées en matière de logiciels.

b. *Piratage sur Internet, téléchargement* : Le téléchargement via modem est strictement interdit sans l'autorisation expresse de l'auteur. Hormis les programmes des catégories "Freeware" et "Shareware" téléchargeables en toute légalité, les logiciels téléchargés sans l'autorisation expresse de l'auteur sont considérés comme illégaux.

c. *Copie du logiciel original sur plusieurs ordinateurs* : Il s'agit d'un mode de reproduction illicite, qui consiste pour l'utilisateur final d'un logiciel acquis légitimement à en faire une ou des copies sur plusieurs ordinateurs. Nombre d'entreprises privées ou publiques utilisent des logiciels sans avoir la quantité de licences correspondantes.

d. *Vente de produits "Mise à jour" en lieu et place de produits complets* : Certains logiciels, commercialisés sous forme de mise à jour, font l'objet d'un conditionnement spécifique, avec une mention "Mise à jour spéciale pour les utilisateurs d'une précédente version" ou "Offre spéciale pour les utilisateurs d'une application", en vue de leur distribution à des conditions tarifaires privilégiées. La commercialisation de ces "produits mises à jour" en lieu et place des logiciels en version complète, sans tenir compte des restrictions liées à leur commercialisation, n'est pas autorisée.

e. *La contrefaçon de produits de distribution à des fins commerciales par des revendeurs* : Le logiciel est purement et simplement recopié sur un CD Rom ou des disquettes, sans la moindre autorisation de la part du titulaire des droits du logiciel, et sans payer ces droits, dans un but de commercialisation.

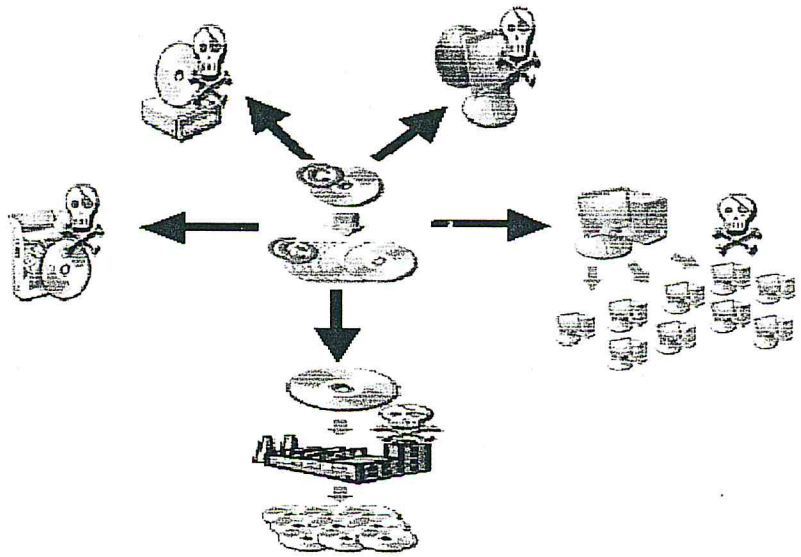

Figure III.1 : Les formes du piratage des logiciels

3. Les différents moyens de protection des logiciels

Il n'existe malheureusement aucun moyen de protection infaillible, on peut simplement retarder le moment où le logiciel sera piraté et diffusé de manière illicite.

A côté des moyens techniques de protection qui sont les clefs de protection, les mots de passe et les techniques de stéganographies, il existe aujourd'hui différents régimes juridiques de protection : la protection par droit d'auteur et la protection par brevet.

3.1. Les moyens techniques de protection des logiciels

Les principales techniques utilisées actuellement sont :

3.1.1. L'activation du logiciel par une clé

C'est certainement le moyen le plus utilisé encore aujourd'hui. Lors de l'installation ou de la première exécution du logiciel, l'utilisateur doit insérer un code d'activation ayant une structure et une longueur spécifique suivant l'algorithme de cryptage utilisé.

Pour que ce procédé soit fiable, il est nécessaire que le logiciel soit sérialisé, c'est à dire qu'à chaque exemplaire corresponde un seul code d'activation ; Sinon le code d'activation du logiciel va apparaître sur Internet et cette technique de protection ne sera plus efficace.

3.1.2. L'activation du logiciel lors de l'enregistrement

L'activation se fait dans un premier temps avec un code d'activation classique. Suite à l'acceptation de cette clé par le programme, une clé est générée par le programme en fonction de la première clé et d'autres éléments permettant d'identifier l'ordinateur, c'est les empreintes de la machines (n°de série du disque dur, n°de série d'un processeur, ...etc.). L'utilisateur contacte alors un site Internet afin de récupérer la clé finale d'activation du programme.

C'est une stratégie de protection qui se base sur le fait que chaque ordinateur est unique, en quelque sorte chaque ordinateur dispose de sa propre empreinte digitale. Il faut prévoir que le client pourra contacter l'entreprise à tout moment (via un site web, un centre d'appels) afin d'obtenir la clé d'activation de son logiciel. Il faudra également tenir compte du fait que le client doit pouvoir réinstaller son logiciel lors d'un changement d'ordinateur ou de disque dur, processeur, ...etc.

Cette technique est utilisée, entre autres, par Microsoft pour Office XP et Windows XP, et également par Borland pour ses outils de développement [Run 02].

3.1.3. La protection hardware (Dongle)

Clé physique ou « bouchon », qui doit être placé sur une interface (soit sur le port de l'imprimante soit sur l'un des ports USB[1]). La protection est obtenue au moyen d'un

[1] USB : Universal Serial Bus, nouveau type de prise permettant de relier à l'unité centrale jusqu'à 127 périphériques.

code mémorisé dans le dongle et qui est interrogé par le programme. Le logiciel recherche la clé matérielle connectée ; S'il ne la trouve pas, il cesse automatiquement de s'exécuter, cela évite qu'une même copie du logiciel ne soit installée et utilisée sur plusieurs ordinateurs.

Outre le coût non négligeable d'acquisition de ces clés, il est nécessaire de les sérialiser individuellement, on doit également prévoir une solution de dépannage et de remplacement rapide d'une clé défaillante surtout si l'application relève un caractère vital pour l'activité de l'entreprise utilisatrice.

Ce type de protection possède également un autre point faible qui est la compatibilité délicate suivant les différents pilotes des périphériques et les mises à jour des systèmes d'exploitation. En revanche, ces solutions offrent un niveau de protection convenable même s'il existe des émulateurs logiciels de ses clés de protection.

3.1.4. La stéganographie

Elle consiste à marquer chaque document (image, document texte, libraire, ...) créé par le logiciel afin de pouvoir à tout moment identifier avec quelle copie du logiciel ce document a été généré [Run 02].

Cette technique ne peut être utilisée que pour des logiciels réalisant des documents avec un format propriétaire. En effet, cette technique n'empêche pas la copie illégale mais elle permet de vérifier qu'une société diffusant des fichiers au format du logiciel possède bien une licence d'utilisation et le cas échéant permet d'agir contre elle.

3.2. Les moyens juridiques de protection des logiciels

En plus des solutions techniques de protection contre la copie, il a fallu élaborer des solutions juridiques notamment à cause du développement de la concurrence et de la facilité de duplication des logiciels. Deux solutions ont été mises en place :

3.2.1. La protection des logiciels par le droit d'auteur

La protection du logiciel par le droit d'auteur est la protection traditionnelle du logiciel depuis 1991, elle est accordée automatiquement pour une durée de 50 ans à toute œuvre originale (c'est-à-dire qui constitue une création intellectuelle propre à son auteur) et bénéficie aux programmes d'ordinateur depuis la transposition en droit français de la directive européenne[(2)] [Net 02].

La copie des logiciels est passible de sanctions pénales : un million de francs d'amende et deux ans de prison (loi du 5 février 1994 relative à la contrefaçon) ainsi que d'amende visant à réparer le préjudice civil (prix public du logiciel multiplié par le nombre de copies, plus pénalités globales pour préjudice moral) [May 97].

Même si l'absence de mention du copyright ne diminue en rien la protection au titre du droit d'auteur, il est recommandé bien évidemment de la mettre en évidence en faisant

(2) Directive 91/250/CEE du Conseil, du 14 mai 1991, concernant la protection juridique des programmes d'ordinateur, Journal officiel n° L 122 du 17/05/1991 p. 0042 – 0046.

apparaître une mention du type " © [nom] [année] " sur l'emballage, les disquettes, le cd-rom, la première page s'affichant à l'écran, ...etc.

Pour une protection par droit d'auteur il faut enregistrer le logiciel en le faisant entrer dans la catégorie des œuvres littéraires. En France, par exemple, vous pourrez enregistrer votre logiciel auprès de l'APP (Agence pour la Protection des Programmes : organisation européenne des auteurs de logiciels et concepteurs en technologies de l'information).

- **Les faiblesses de la protection des logiciels par le droit d'auteur**

La durée de protection de 50 ans peut sembler excessive compte tenu de la durée de vie économique des logiciels qui est assez courte dans un contexte technologique en constante évolution.

Le logiciel est uniquement protégé contre la copie, la traduction, l'adaptation, la modification et la distribution. Rien n'interdit dès lors à un concurrent de développer un logiciel reprenant l'idée du logiciel protégé dès lors qu'il ne copie pas directement le code source ou objet du logiciel et arrive à la même solution de façon indépendante.

Ainsi certains éditeurs utilisent des procédures de " clean room " afin de développer très rapidement des logiciels : une première équipe est chargée de l'analyse d'un logiciel existant afin de définir un cahier des charges qui sera ensuite transmis à une seconde équipe, chargée elle de développer un logiciel concurrent du premier. Du fait du cloisonnement des deux équipes il est peu probable que l'atteinte aux droits d'auteurs du premier logiciel sera reconnue.

Pour ces motifs certaines entreprises qui considèrent que le droit d'auteur est inadapté au développement industriel souhaitent savoir si elles peuvent recourir à la protection par brevet qui leur semble plus appropriée à la protection de leurs investissements.

3.2.2. La protection des logiciels par le brevet

A partir des années 60, les demandes de brevet tout comme les brevets accordés font l'objet d'une publication comportant une description détaillée de l'invention. Le brevet est un titre légal qui confère à son titulaire une exclusivité d'exploitation temporaire de son invention sur un territoire déterminé. Dès lors le titulaire du brevet peut interdire la fabrication, la vente ou l'utilisation de son invention effectuées sans son accord [Net 02].

Cependant le brevet offre des avantages par rapport au droit d'auteur dans la mesure où il permet de protéger non seulement la forme mais aussi les idées et l'algorithme du programme. Avec le brevet la durée de la protection est de 20 ans [May 97].

- **Les conditions de la brevetabilité**

Pour pouvoir être brevetable, l'invention doit satisfaire les conditions suivantes [Net 02] :

a. *Une invention nouvelle :* L'inventeur doit préalablement à sa demande de brevet effectuer une recherche d'antériorité afin de vérifier la nouveauté de son invention par rapport à l'état de l'art mais aussi que son invention n'a pas déjà été brevetée. Cette recherche lui permettra d'identifier les brevets toujours en vigueur qui pourraient

recouvrir pour partie le domaine de son invention, dans un tel cas il peut être utile de faire interpréter ces résultats par un professionnel afin d'assurer de la possibilité d'obtenir un brevet sur l'invention. Cette recherche doit être effectuée dans les bases de données des brevets français, européens ou internationaux, mais concernant les logiciels il n'existe pas encore de véritables bases de données reflétant l'état de l'art, l'inventeur devra donc étendre sa recherche à d'autres sources d'informations telles que les publications scientifiques, conférences spécialisées, ...etc.

b. *Une activité inventive :* Seules les créations logicielles innovantes pourront faire l'objet d'un brevet, celles qui ne font que reprendre un procédé déjà connu ne peuvent donc pas en bénéficier.

c. *Une invention susceptible d'applications industrielles :* Une invention est considérée comme susceptible d'applications industrielles si son objet peut être fabriqué ou utilisé dans tout genre d'industrie, y compris l'agriculture. Dès lors une invention mise en œuvre par ordinateur pourrait être susceptible d'application industrielle.

- **Les faiblesses de la protection des logiciels par le brevet**

Même si le coût d'un dépôt de brevet dans un seul pays reste raisonnable, la protection fournie sera géographiquement limitée à ce pays, l'inventeur sera donc conduit à enregistrer son logiciel dans d'autres pays afin d'assurer une plus grande protection, ce qui rend le dépôt beaucoup plus coûteux, surtout pour les petites entreprises.

3.2.3. Les conséquences d'une possible protection cumulative droit d'auteur/brevet

Le brevet et le droit d'auteur ont des champs de protection distincts, le droit d'auteur protégeant une" *forme d'expression* " tandis que le brevet protège lui une" *fonction utilitaire* ". Ces deux protections pourraient donc non pas être exclusives l'une de l'autre mais au contraire complémentaires même s'il existe des difficultés liées à la durée et au point de départ de la protection ainsi qu'au mode de rémunération de ces deux systèmes de protection.

4. Des exemples sur les techniques de protection

Après avoir abordé les différents moyens de protection des logiciels contre le piratage, il nous a paru nécessaire de présenter les moyens techniques de protection les plus répandues actuellement en les détaillant avec des exemples pour les mieux comprendre.

4.1. Exemple sur l'activation du logiciel par une clé d'installation

C'est une technique utilisée par plusieurs logiciels. Dans cet exemple nous allons présenter les étapes du lancement d'installateur du logiciel « Delphi 5» (version professionnelle) : c'est un compilateur édité par la société Borland.

- **Installation de Delphi 5**

a. *Lancement du programme d'installation* : Le programme d'installation de Delphi 5 démarre automatiquement lorsqu'on insère le CD dans le lecteur CD-Rom. Si ce n'est pas

le cas, il faut qu'on exécute le programme d'installation « INSTALL.EXE » depuis le CD. Lorsqu'on démarre le programme d'installation, la fenêtre suivante s'affiche :

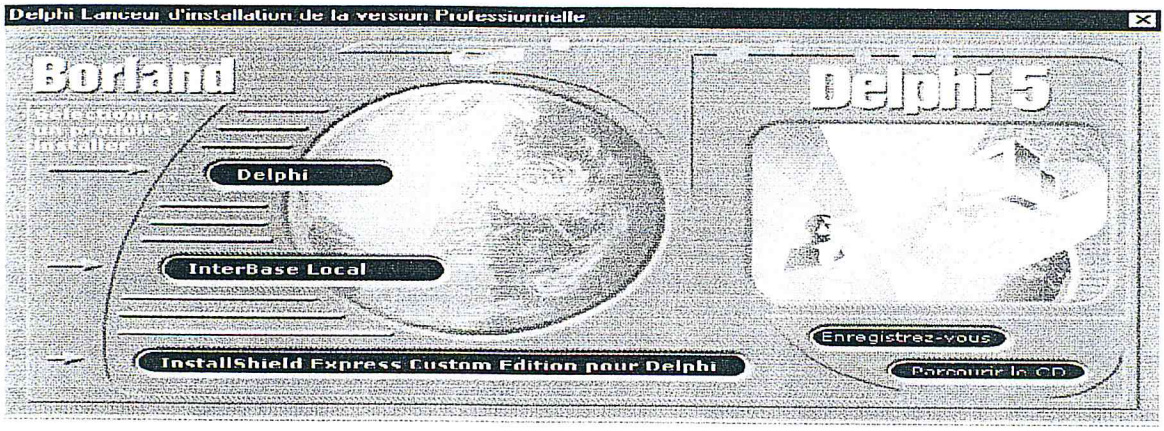

Figure III.2 : Lancement d'installation de Delphi 5 [Réf 01]

On veut installer Delphi, donc il suffit de cliquer sur le bouton Delphi.

b. *Insertion du code d'installation* : Une fenêtre de dialogue mot de passe (Figure III.3) doit apparaître pour demander à l'utilisateur d'insérer le code d'installation qui est composé dans ce cas d'un numéro de série (des chiffres) et d'une clé d'autorisation (des lettres et des chiffres).

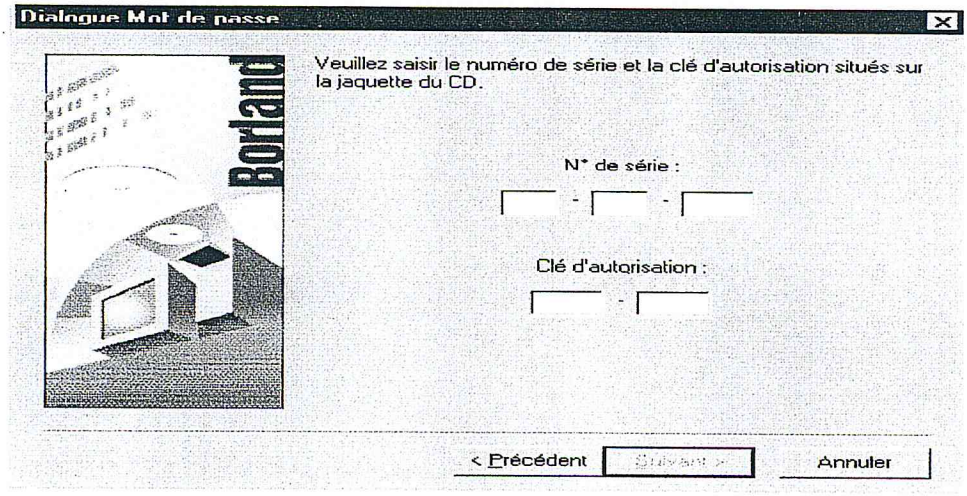

Figure III.3 : Fenêtre de saisie du Code d'installation [Réf 01]

Généralement, on trouve ce code d'installation dans la jaquette du CD ou bien dans un document texte situé dans le fichier d'installation du CD. Le code d'installation dans notre cas est le suivant :

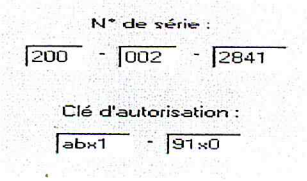

Figure III.4 : Le code d'installation du Delphi

Dans le cas où l'utilisateur aurait saisi un code erroné, un message d'avertissement (voir figure ci-dessous) doit être affiché pour lui dire qu'il n'a pas inséré un code correct et donc, il faut qu'il recommence pour pouvoir installer Delphi.

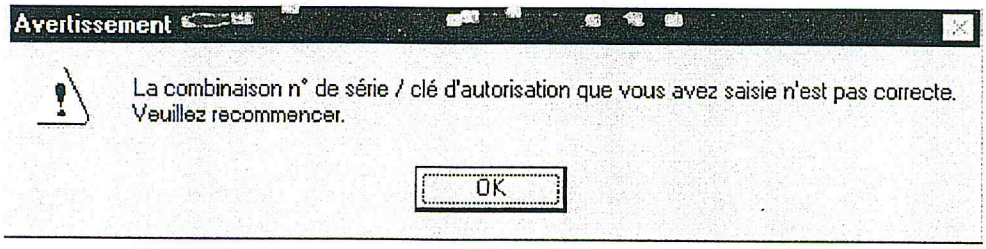

Figure III.5 : Message d'avertissement [Réf 01]

Après l'insertion du bon code, le logiciel peut être installé normalement en répondant à tous les messages affichés pendant l'installation.

- **Inconvénients**

Le code d'installation est à la portée de tout le monde parce qu'il est indiqué sur le CD-Rom, et donc cette technique n'est plus efficace pour protéger un logiciel contre le piratage, car un utilisateur peut installer le logiciel sur n'importe quel ordinateur et autant de fois qu'il veut.

4.2. Exemple de l'Activation de Windows XP via Internet

Depuis toujours, Microsoft soutient la protection de la propriété intellectuelle et lutte contre toute forme de piratage. Le nouveau système d'activation des produits Microsoft, découvert avec Windows XP, ne se cantonne plus à la simple saisie du numéro de série indiqué sur le CD-Rom. Désormais, l'installation nécessite l'enregistrement du produit directement auprès de Microsoft, via une connexion Internet ou un appel téléphonique.

L'activation de produit Microsoft est un système anti-piratage chargé de vérifier que les logiciels font bien l'objet d'une licence. Ce processus vise ainsi à réduire la forme de piratage la plus répandue : la copie illégale. L'activation est un processus rapide, simple et discret qui respecte la confidentialité.

Microsoft permettra à l'utilisateur d'installer le logiciel 30 fois avant de l'obliger à s'inscrire [Inf 01].

- **Processus d'activation du logiciel Windows XP**

Les différentes étapes d'activation du logiciel Windows XP sont décrites comme suit :

- Lors de l'installation du logiciel, l'utilisateur doit mentionner la clé du produit, cette clé se transforme ensuite en un numéro d'installation qui est une combinaison des empreintes digitales de l'ordinateur avec le code unique attribué au produit Microsoft,

- Il faut ensuite utiliser l'assistant activation du produit pour communiquer le numéro de l'installation à Microsoft soit par transfert sécurisé sur Internet soit par téléphone,

- Microsoft renvoie à l'utilisateur un ID (Identificateur) de confirmation qui lui permet d'activer le produit. Le numéro d'installation se compose de l'ID produit sous forme cryptée. L'ID de confirmation est tout simplement une clé permettant de déverrouiller l'installation Windows XP sur un PC particulier,

- Si l'utilisateur décide d'activer le produit par Internet, l'Assistant Activation détecte sa connexion Internet dès la soumission de la demande et se connecte à un serveur sécurisé pour communiquer son numéro d'installation à Microsoft. Un ID de confirmation est renvoyé à sa machine, activant ainsi automatiquement Windows XP,

- Pour activer Windows XP par téléphone, l'utilisateur doit composer tout simplement le numéro gratuit mentionné sur l'écran. Un représentant du service clientèle lui demande alors le numéro d'installation affiché sur ce même écran, l'entre dans une base de données sécurisée et il lui renvoi un ID de confirmation. Dès qu'il tape l'ID de confirmation, le processus d'activation prend fin.

Nous pouvons schématiser le processus de communication entre l'utilisateur et Microsoft par la figure suivante :

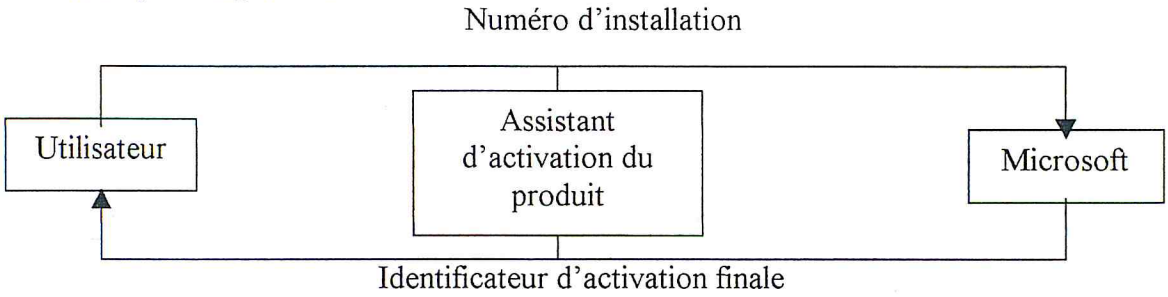

Figure III.6 : La communication pour l'activation de Windows XP

- **Pas d'obstacle aux mises à jour courantes du matériel**

Les empreintes digitales combinés avec le code du produit sont : le nom de volume et le numéro de série du disque dur, l'adresse MAC de la carte réseau, les numéros de série : du lecteur de CD-Rom, de la carte graphique, du processeur et des contrôleurs IDE et SCSI ainsi que la quantité de RAM et le modèle du processeur [BEN 01].

Chaque ajout ou suppression de ce type de matériel provoque automatiquement la mise à jour d'un fichier système qui crée une nouvelle empreinte, tout en conservant intacte l'empreinte de la configuration d'origine. Lorsque le système constate trois différences (sur les dix éléments) entre la configuration courante et celle d'origine, une réactivation est nécessaire pour que Microsoft enregistre les changements. Seule la disparition du fichier système contenant les empreintes à l'occasion par exemple d'une réinstallation de Windows XP nécessite donc obligatoirement une réactivation.

- **Inconvénients**

Comme nous l'avons expliqué au préalable, l'activation du produit est obligatoire totalement anonyme, rapide et simple. Le problème c'est que ce genre de protection crée en général des inconvénients majeurs aux utilisateurs légitimes, car il rend nécessaire

l'utilisation d'un service Internet ou d'une communication téléphonique pour obtenir une clé qui permettra de déverrouiller l'accès au logiciel.

4.3. Exemple de la clé matérielle SentinelPro

Rainbow Technologies : ensemble des fabricants de dispositifs matériels de protection des logiciels; Ils ont élaboré plusieurs types des dongles : SentinelPro, Sentinel Superpro, NetSentinel, ...etc. Dans cet exemple, il s'agit de la clé matérielle : « SentinelPro ».

- **Présentation de SentinelPro**

Le système de protection des logiciels SentinelPro utilise une petite clé physique (voir Figure III.7) pour déverrouiller les applications logicielles. Cette clé se fixe sur le port parallèle d'un micro-ordinateur IBM PC[(1)]/XT[(2)]/AT[(3)], PS/2[(4)]. Une fois installée, SentinelPro constitue un module transparent du logiciel utilisé [Sen 94].

Pour protéger le programme contre le piratage avec la clé SentinelPro, on introduit une série de verrous logiciels dans l'application. Chaque verrou constitue un appel au driver SentinelPro qui ne peut aboutir qu'en présence de la clé physique. Si la clé est absente, un code d'erreur est renvoyé au programme pour mettre fin à l'exécution de la copie non autorisée. Le logiciel peut être piraté, mais la copie piratée ne fonctionnera pas.

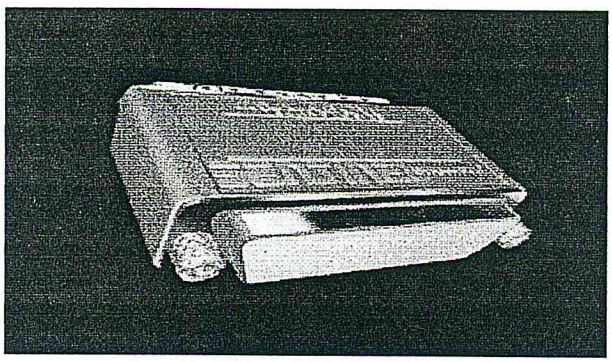

Figure III.7 : SentinelPro [Sen 94]

- **Installation de la clé SentinelPro**

Pour installer une clé SentinelPro, il faut la connecter au port d'imprimante parallèle d'un IBM PC/XT/AT, d'un PS/2. Des vis permettent de fixer la clé sur le port. L'installation se fait par un programme INSTALL qui simplifie l'installation du logiciel SentinelPro sur le disque dur. Les disquettes d'installation contiennent les interfaces logicielles de tous les langages de programmation supportés, on n'installe généralement qu'une seule interface.

---

(1) IBM PC : Le premier ordinateur personnel mis sur le marché par IBM (International Business Machines) et livré avec le système d'exploitation MS DOS de Microsoft.
(2) XT : Extended Technology, ont suivi directement les IBM PC.
(3) AT : Advanced Technology est un successeur de XT.
(4) PS/2 : Personal System/2 : utilisation d'un processeur étendu sur 16 bits et l'affichage à la norme graphique VGA.

a. *Lancement de l'installateur* : Trois versions du programme d'installation [Sen 94] sont fournies :

- *WINSTALL* pour Windows 3.x et Windows NT sur plates-formes Intel,
- *DINSTALL* pour DOS et Windows NT sur plates-formes DEC Alpha et MIPS, et
- *OINSTALL* pour systèmes OS/2.

Il faut choisir le programme d'installation correspondant à la plate-forme et au système d'exploitation utilisés. Les instructions ci-dessous partent du principe qu'on procède à l'installation à partir d'une disquette :

Pour installer le logiciel SentinelPro [sen 94] :

- Introduire la disquette dans une unité de disquette,
- Lancer l'un des programmes d'installation (WINSTALL, DINSTALL ou OINSTALL),
- Répondre aux invites qui s'affichent. Le programme demande d'indiquer une destination pour les fichiers installés et de choisir les interfaces à installer,
- Une fois l'installation terminée, un message pour l'installation du driver système sera affiché.

b. *Installation du driver* : Le driver Sentinel établit une liaison entre l'application et la clé physique. Le mode d'installation du driver est en fonction du système d'exploitation employé.

Pour installer un driver système SentinelPro [Sen 94] :

- Introduire la disquette Sentinel System Drivers dans une unité de disquette,
- Lancer le programme d'installation se trouvant dans le sous-répertoire de la plate-forme utilisée. Par exemple, Windows NT se trouve dans le répertoire WIN_NT,
- Répondre aux invites qui s'affichent,
- Une fois sorti du programme d'installation, réamorcer le système (ou redémarrer Windows) pour activer le nouveau driver.

- **Installation de plusieurs clés SentinelPro**

Jusqu'à cinq clés SentinelPro peuvent être connectées simultanément au même port parallèle pour permettre la protection de plusieurs logiciels [Sen 94]. Des clés SentinelPro empilées doivent obligatoirement avoir des codes de famille différents. Le code de famille à deux lettres figure sur l'étiquette de la clé.

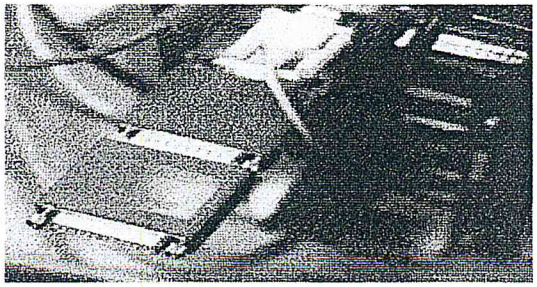

Figure III.8 : Plusieurs clés SentinelPro empilées [Sen 94]

CHAPITRE IV

La cryptographie

mal de problème pour l'activité des logiciels de l'entreprise donc il faut la remplacer le plutôt possible.

5. Comparaison entre les techniques de protection

Voici un tableau qui synthétise les méthodes de protections présentées :

Tableau III.1 : Avantages et inconvénients des techniques de protection

Les techniques	Les avantages	Les inconvénients
Activation du logiciel par une clé	-Activation simple et rapide	-La clé est à la portée de tout le monde
Activation du logiciel lors de l'enregistrement	-Activation obligatoire, anonyme, rapide et simple -Installation sur une machine particulière (code unique attribué au produit)	-La réinstallation du logiciel en cas de changement d'un élément de la machine -La nécessité d'utilisation des moyens de communication afin de récupérer la clé d'activation
Clé matérielle	-Offre un niveau de protection convenable	-Coût supplémentaire au développement -La défaillance d'une clé peut poser des problèmes pour l'activité du logiciel
Stéganographie	-Permet de vérifier qu'un utilisateur des fichiers au format du logiciel possède une licence d'utilisation ou non	-Utilisée pour des logiciels réalisant des documents -N'empêche pas la copie illégale

Conclusion

Les techniques et outils utilisés pour protéger le logiciel contre le piratage ne sont pas très nombreux. Chaque technique de protection a ses avantages et ses inconvénients, donc il n'y a pas une technique qui garantie la protection du logiciel à 100 %, le risque existe toujours.

Les logiciels activés par une clé d'installation utilisent souvent des algorithmes de cryptage et même dans le cas de l'activation du logiciel par Internet, nous avons vu que le numéro d'installation généré par WindowsXP est composé de l'ID produit sous forme crypté. Le chapitre suivant nous permet d'acquérir des connaissances approfondies sur les méthodes cryptographiques.

Introduction

La cryptographie est une science utile et nécessaire dans l'industrie de la sécurité informatique ; Les principes de base de la cryptographie et les différents types de cryptage vont être abordés dans ce présent chapitre.

1. Définitions, buts et fonctionnement de la cryptographie

▪ Vocabulaire de base

a. *Cryptographie, cryptologie :* La cryptographie vient du grec, « kruptos » qui veut dire caché et de « graphein » qui signifie écrire [Viv 97]. La cryptologie est une véritable science qui étudie le chiffrement et le déchiffrement d'informations, tandis que la cryptographie est l'utilisation de systèmes mis au point par des cryptologues [Mar 95].

b. *Cryptage, décryptage* : Le fait de coder un message pour le rendre secret s'appelle le cryptage (ou chiffrement). La méthode inverse consistant à retrouver le message original est appelée décryptage. Le cryptage se fait généralement à l'aide d'une clef de chiffrement, le décryptage nécessite quant à lui une clef de décryptage [Pil 01].

c. *Cryptanalyse* : L'ensemble des techniques permettant le décryptage s'appelle la cryptanalyse. Le but de la cryptanalyse consiste à déterminer le contenu d'un message chiffré sans en connaître la clef.

▪ La cryptographie répond à différents besoins

- La confidentialité : consiste à rendre l'information inintelligible à tous ceux qui pourraient intercepter le message.
- Le contrôle d'accès : permet de limiter l'accès aux données, aux serveurs que pour les personnes autorisées (mot de passe Unix, par exemple).
- L'intégrité des données : consiste à vérifier que cette donnée n'a pas été altérée frauduleusement.
- L'identification : permet d'assurer de l'authentification des partenaires et de l'origine des messages.

▪ Comment fonctionne la cryptographie?

Un algorithme cryptographique est une fonction mathématique utilisée dans le processus de chiffrement et de déchiffrement. Il fonctionne en combinaison avec une clé (un mot, un nombre ou une phrase) pour chiffrer le texte clair. Le même texte clair se chiffre en un texte chiffré différent si l'on utilise des clés différentes. La sécurité des données chiffrées est entièrement dépendante de deux choses : la force de l'algorithme cryptographique et le secret de la clé.

Figure IV.1 : Principe de chiffrement et de déchiffrement

2. Cryptage par transposition

La méthode de chiffrement par transposition consiste à réarranger géométriquement les données pour les rendre visuellement inexploitables. La technique assyrienne utilise le cryptage par transposition.

- **La technique assyrienne**

Cette technique de cryptage a été utilisée pour dissimuler des messages écrits sur des bandes de papyrus comme l'indique la figure suivante :

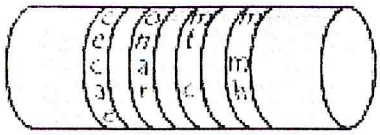

Figure IV.2 : Message écrit sur bande de papyrus [Pil 01]

La technique consistait à :

- Enrouler une bande de papyrus sur un cylindre, et
- Ecrire le texte horizontalement sur la bande ainsi enroulée.

Prenant comme exemple le message "comment ça marche", le message une fois déroulé n'est plus compréhensible ("cecae onar mt c m mh ").

Il suffit au destinataire d'avoir un cylindre de même rayon pour pouvoir déchiffrer le message. En réalité un casseur peut déchiffrer le message en essayant des cylindres de diamètre successifs différents, donc la méthode peut être cassée statistiquement (il suffit de prendre les caractères un à un, éloignés d'une certaine distance).

3. Système à clef secrète

Appelée aussi cryptographie conventionnelle ou système à clé symétrique, une seule clé est utilisée à la fois pour le chiffrement et le déchiffrement. La figure suivante est une illustration du processus du chiffrement conventionnel.

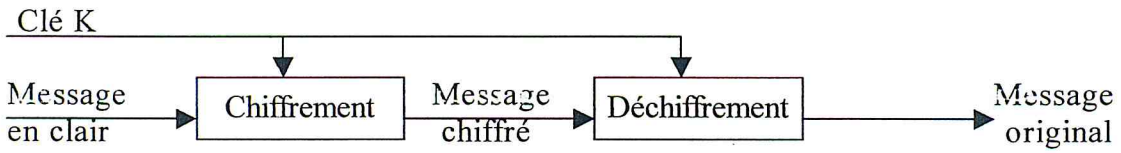

Figure IV.3 : Chiffrement et déchiffrement avec une clé secrète

Voici quelques systèmes qui utilisent le cryptage avec une clef secrète :

3.1. Le chiffrement de César

Ce type de chiffrement est un des plus anciens, Il s'agit donc de décaler l'ensemble des valeurs des caractères du message d'un certain nombre de positions, c'est-à-dire en quelque sorte de substituer chaque lettre par une autre.

Par exemple, en décalant le message "PIRATAGE" de 3 positions, nous obtenons "SLUDWDJH". Lorsque l'ajout de la valeur donne une lettre dépassant la lettre Z, il suffit de continuer en partant de A.

Nous commençons avec : **ABCDEFGHIJKLMNOPQRSTUVWXYZ**
En décalant le tout de 3, nous obtenons : **DEFGHIJKLMNOPQRSTUVWXYZABC**

Nous appelons clé le caractère correspondant à la valeur que nous ajoutons au message pour effectuer le cryptage. Dans ce cas la clé est la 3ème lettre de l'alphabet (C).

Ce système de cryptage est certes simple à mettre en oeuvre, mais il a pour inconvénient d'être totalement symétrique, cela signifie qu'il suffit de faire une soustraction pour connaître le message initial. Evidemment, c'est de la cryptographie excessivement faible au regard des normes actuelles, mais bon, cela marchait pour César, et cela illustre aussi comment fonctionne la cryptographie conventionnelle.

3.2. Le chiffrement de Vigenère

Il consiste à coder un texte avec un mot en ajoutant à chacune de ses lettres la lettre d'un autre mot appelé clé. La clé est ajoutée indéfiniment en vis-à-vis avec le texte à chiffrer. On associe dans un premier temps à chaque lettre un chiffre correspondant.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

Par exemple le texte "rendezvousamedi" avec la clé "bonjour" sera codé de la manière suivante :

Texte original :

r	e	n	d	e	z	v	o	u	s	a	m	e	d	i
18	5	14	4	5	26	22	15	21	19	1	13	5	4	9

Clé :

b	o	n	j	o	u	r
2	15	14	10	15	21	18

Texte crypté :

r+b	e+o	n+n	d+j	e+o	z+u	v+r	o+b	u+o	s+n	a+j	m+o	e+u	d+r	i+b
18+2	5+15	14+14	4+10	5+15	26+21	22+18	15+2	21+15	19+14	1+10	13+15	5+21	4+18	9+2

Pour déchiffrer ce message il suffit d'avoir la clé secrète et faire le déchiffrement inverse, à l'aide d'une soustraction.

Bien que ce chiffrement soit beaucoup plus sûr que le chiffrement de César, il peut encore être facilement cassé. En effet, lorsque les messages sont beaucoup plus longs que la clef, il est possible de repérer la longueur de la clef et d'utiliser pour chaque séquence de la longueur de la clef la méthode consistant à calculer la fréquence d'apparition des lettres, permettant de déterminer un à un les caractères de la clé. Pour éviter ce problème, il faut utiliser une clef dont la taille est proche de celle du texte afin de rendre impossible une étude statistique du texte crypté. Mais la longueur de la clé de cryptage implique une probabilité d'erreur (une seule erreur rend le texte indéchiffrable).

3.3. DES (Data Encryption Standard)

C'est à ce jour l'algorithme de cryptage le plus répandu. Il est inventé par IBM [Mar 95]. C'est un système de chiffrement par blocs de 64 bits, dont le dernier octet (8 bits) sert de test de la parité. Il consiste à faire des combinaisons, des substitutions et des permutations entre le texte à chiffrer et la clé, en faisant en sorte que les opérations puissent se faire dans les deux sens (pour le décryptage). La clé est de 56 bits, ce qui représente tout de même 2^{56} possibilités, c'est-à-dire $72*10^{15}$ clés possibles.

Les grandes lignes de l'algorithme sont les suivantes :

- Fractionnement du texte en blocs de 64 bits,
- Permutation des blocs,
- Découpage des blocs en deux parties : gauche et droite,
- Etapes de permutation et de substitution répétées 16 fois (appelées rondes), et
- Recollement des parties gauches et droites puis permutation initiale inverse.

Même si une clé de 56 bits donne un nombre énorme de possibilités, des processeurs permettent de calculer plus de 10^6 clés par seconde, ainsi, utilisés parallèlement sur un très grand nombre de machines, il peut être possible à un grand organisme (un Etat par exemple) de trouver la bonne clé.

4. Système à clé publique ou asymétrique

Ce concept fut inventé par Whitfield Diffie et Martin Hellman en1975 [Nai 98]. La cryptographie à clé publique repose sur un schéma asymétrique qui utilise une paire de clés pour le chiffrement : une clé publique qui chiffre les données, et une clé privée aussi appelée clé secrète, qui sera utilisée pour le déchiffrement.

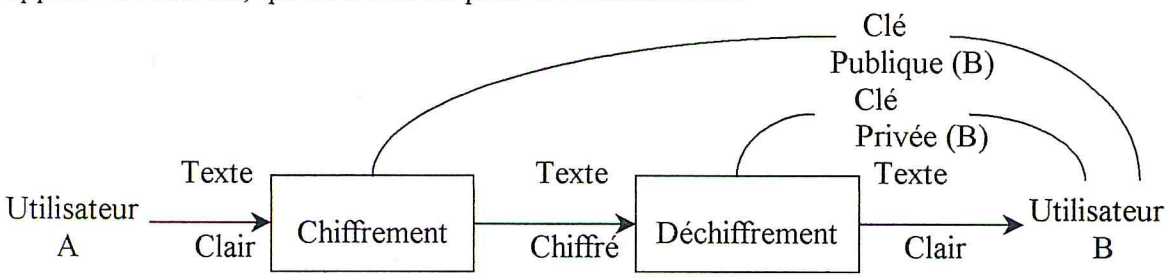

Figure IV.4 : Système à clé publique

RSA et PGP sont deux algorithmes qui utilisent le cryptage asymétrique :

4.1. Le système RSA

Il est proposé par Rivest, Shamir et Adelman (RSA) en 1977 [Riv 77]. Le cryptage RSA est un algorithme à clé publique qui a la propriété que n'importe laquelle des deux clés de la paire peut être utilisée pour coder le message, tant qu'on utilise l'autre clé pour le décoder et cela sans rendre les deux clés publiques.

- **Caractéristiques**

Les caractéristiques de RSA sont décrites ci-dessous :

- Le RSA est un cryptage par bloc dans lequel les messages (blocs) sont des entiers compris entre 0 et n-1 pour un certain n.
- Deux clés e et d (utilisées respectivement pour le cryptage et le décryptage). Le message clair M sera codé de la manière suivante : M^e mod n.
- Afin de pouvoir déchiffrer le message, on choisit d de telle manière que : $(M^e)^d$ mod n = M.
- Le cryptage et le décryptage sont mutuellement commutatifs, soit : $M = (M^e)^d$ mod n = $(M^d)^e$ mod n.

4.2. PGP (Pretty Good Privacy)

C'est un algorithme conçu par Philip Zimmermann en 1991 [Zim 98]. PGP combine à la fois les meilleures fonctionnalités de la cryptographie conventionnelle et de la cryptographie à clé publique [Kob 98].

Quand un utilisateur chiffre du texte clair avec PGP, PGP compresse d'abord le texte clair. La plupart des techniques de cryptanalyse exploitent les redondances trouvées dans le texte clair pour craquer le texte chiffré. La compression réduit ces redondances dans le texte clair, ce qui augmente grandement la résistance à la cryptanalyse. PGP crée ensuite une clé de session (une clé secrète). Le résultat est le texte chiffré. Une fois que les données sont chiffrées, la clé de session est elle-même chiffrée par la clé publique du destinataire. Enfin, elle est transmise avec le texte chiffré au destinataire.

Le déchiffrement fonctionne de la manière inverse. La copie de PGP du destinataire utilise la clé privée de celui-ci pour retrouver la clé de session temporaire, que PGP utilise ensuite pour déchiffrer le texte chiffré de manière conventionnelle.

5. Comparaison entre les systèmes cryptographiques

Nous décrivons les avantages et les inconvénients des systèmes cryptographiques dans le tableau suivant :

Tableau IV.1 : Les avantages et les inconvénients des systèmes cryptographiques

Systèmes	Avantages	Inconvénients
Système à clé secrète	-Simple -Rapide	-Une même clé doit être connue par l'émetteur et le récepteur -La difficulté de la distribution sécurisée de la clé
Système à clé privée	-Aucun échange de secret n'est nécessaire (la clef de chiffrement n'a plus besoin d'être transmise secrètement)	-Lent

Conclusion

Il faut que la cryptographie avance comme le boulet et la cuirasse, c'est à dire qu'il faut toujours chercher de nouveaux algorithmes plus puissants même si ceux utilisés aujourd'hui sont fiables.

Aujourd'hui, le cryptage des messages est devenu un problème légal, voire un problème de société, de nature à la fois technique, éthique et politique. Car la généralisation des outils informatiques (logiciels, courriers électroniques, réseaux informatiques, ...etc.) rend nécessaire l'utilisation de clés sûres.

CHAPITRE V

Atelier de développement des logiciels de TELEMATIS

Introduction

Tout développeur d'application se retrouve face au choix des outils et de la méthode de développement. Le but de ce chapitre est de nous faire découvrir l'atelier de développement de TELEMATIS. Nous aborderons le système de gestion de bases de données SQL Server (les composants serveur et client et le processus de communication), ensuite nous verrons les outils par lequel un utilisateur sur un poste client peut accéder aux objets disponibles sur SQL Server (les langages de développement et les API) et nous présentons le processus de développement des logiciels de TELEMATIS en définissant enfin les procédures stockées utilisées pour le développement des applications.

1. L'architecture Client-Serveur

Pour de très nombreuses applications, travailler seul sur une seule et même machine n'est pas satisfaisant. C'est notamment le cas pour la mise à jour et l'interrogation de bases de données dans lesquelles plusieurs personnes doivent simultanément pouvoir accéder aux données. C'est pour cela que les logiciels de TELEMATIS sont développés dans une architecture client-serveur. Le serveur est intelligent car il contient les données et la logique d'entreprise. Le client est léger, il contient que l'interface.

2. Serveur de bases de données SQL Server

SQL Server est un système de gestion de bases de données relationnelle. Il est utilisé pour traiter des transactions, stocker et analyser des données, et développer des nouvelles applications. Il est responsable des taches suivantes :

- Gérer les relations entre les données de la base de données,
- Vérifier que les données sont stockées correctement et que les règles qui définissent les relations entre les données ne sont pas transgressées,
- Récupérer toutes les données à un état de cohérence antérieur en cas de panne du système,
- Répondre aux requêtes des applications clientes.

2.1. Composants client

Les composants client de l'architecture de communication sont décrits ci-dessous :

a. *Application cliente* : Une application cliente exécute les instructions Transact-SQL et reçoit les ensembles de résultats.

b. *API de base de données* : Les interfaces API de base de données (OLEDB, ODBC) utilisent un fournisseur, un pilote ou une bibliothèque de liaison dynamique (DLL, Dynamic Link Library) pour transmettre les instructions Transact-SQL et recevoir les ensembles de résultats. Il s'agit d'une interface utilisée par une application pour envoyer des requêtes à SQL Server et pour traiter les résultats renvoyés par SQL Server.

c. *Bibliothèque réseau cliente* : Une bibliothèque réseau cliente gère les connexions réseau et le routage sur le client. Il s'agit d'un composant logiciel de communication qui assemble les requêtes de base de données et les résultats pour faire la transmission par le protocole réseau approprié.

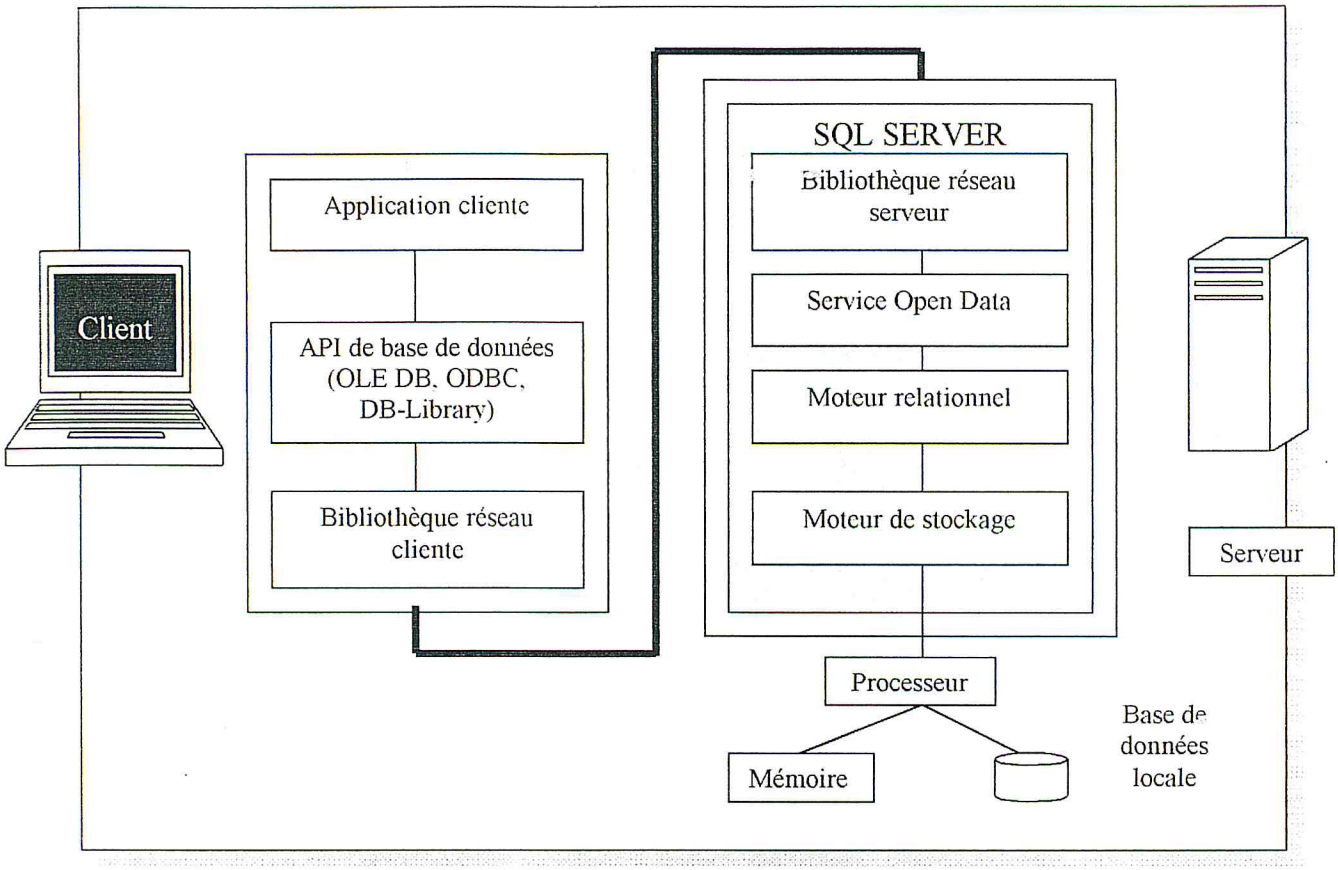

Figure V.1 : Les composants de communication de SQL Server [Mic 00]

2.2. Composants serveur

Les composants serveur de l'architecture de communication sont décrits ci-dessous :

a. *Bibliothèque réseau serveur* : La bibliothèque réseau cliente doit correspondre à l'une des bibliothèques réseau serveur pour que la communication puisse s'effectuer correctement. SQL Server prend en charge les protocoles réseau tels que TCP/IP (Transmission Control Protocol/Internet Protocol), les canaux nommés (Named Pipes), NWLink, IPX/SPX (Internetwork Packet eXchange/Sequenced Packed eXchange), VIA Server Net II SAN, VIA GigaNet San, Banyan VINES et AppleTalk.

b. *Services Open Data* : Les services Open Data mettent le client en relation avec SQL Server de manière transparente, en fournissant une interface réseau qui prend en charge les processus de protocole réseau et les routines de serveur. Il s'agit d'un composant de SQL Server qui gère les connexions réseau, en transmettant les requêtes des clients à SQL Server pour traitement et en renvoyant tous les résultats et réponses aux clients.

c. *Moteur relationnel* : Le moteur relationnel analyse les instructions Transact-SQL, optimise et exécute les requêtes, traite le langage de définition de données (DDL, Data Definition Language).

d. *Moteur de stockage* : Le moteur de stockage gère les fichiers de base de données et l'utilisation de l'espace dans ces fichiers, construit et lit les données à partir des pages physiques, gère les tampons de données, les entrées/sorties (E/S) physiques, contrôle les

conflits d'accès, réalise les applications d'ouverture de session et de récupération et implémente des fonctions utilitaires telles que le vérificateur de cohérence de base de données (DBCC, DataBase Consistency Checker), la sauvegarde et la restauration.

2.3. Processus de communication client-serveur

En général, les clients et les serveurs communiquent par le biais d'un réseau. La séquence ci dessous illustre le processus de communication client-serveur classique avec une API de base de données au moyen d'une requête :

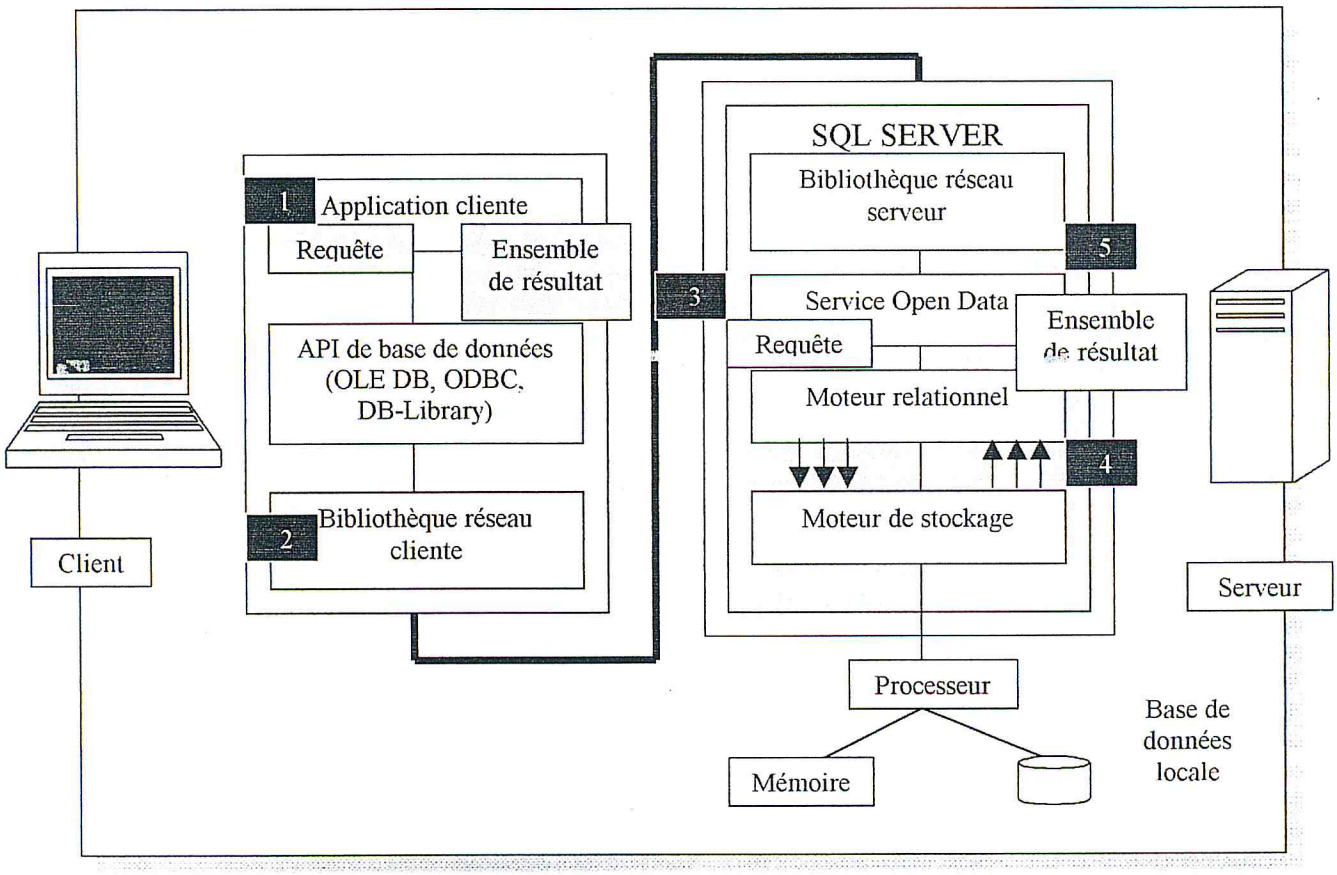

Figure V.2 : Processus de communication client-serveur [Mic 00]

1. Une application cliente soumet une requête. Le client appelle l'API de base de données et transmet la requête. L'API de base de données utilise un fournisseur, un pilote ou un DDL pour encapsuler la requête dans un ou plusieurs paquets TDS (Tabular Data Stream) et les transmettre à la bibliothèque réseau cliente.
2. La bibliothèque réseau cliente assemble les paquets TDS en paquets de protocole réseau. La bibliothèque réseau cliente appelle une API IPC[(1)] (InterProcess Communication) Windows pour envoyer les paquets des protocoles réseau à une bibliothèque réseau serveur en utilisant la pile de protocoles réseau du système d'exploitation. La bibliothèque réseau serveur appropriée extrait les paquets TDS de la pile de protocoles réseau et les transmet aux services Open Data.
3. Les services Open Data extraient la requête des paquets TDS et la transmettent au moteur relationnel. Le moteur relationnel compile ensuite la requête dans un plan

(1) IPC : La communication inter-processus est un échange de données entre deux taches ou deux processus.

d'exécution optimisé. Il exécute le plan d'exécution. Le moteur relationnel communique avec le moteur de stockage à l'aide de l'interface OLE DB.

4. Le moteur de stockage transfère les données d'une base de données à des tampons de données, puis transmet les ensembles de lignes contenant des données au moteur relationnel. Le moteur relationnel combine les ensembles de lignes dans l'ensemble de résultats final et le transmet aux services Open Data.
5. Les services Open Data prépare l'ensemble de résultats et le renvoient à l'application cliente en utilisant une bibliothèque réseau serveur, la pile de protocole réseau, la bibliothèque réseau cliente et l'API de base de données.

3. Applications clientes

Pour accéder aux objets disponibles sur SQL Server (bases de données, procédures stockées, fonctions, ...etc.), les utilisateurs utilisent des applications clientes développées avec :

3.1. Langages de développement

- **Transact-SQL :** C'est un principal langage de programmation et de requêtes utilisé par SQL Server.
- **Visual Basic 6.0 :** Le langage Visual Basic 6.0 sera utilisé pour développer l'interface utilisateur.

3.2. API et objets pour la connectivité à la base de données

API (Application Programming Interface), c'est une interface normalisée par l'intermédiaire de laquelle le programmeur a directement accès aux fonctions d'un système d'exploitation ou d'une interface utilisateur. API est un format de communication qui facilite la connexion entre les clients et les bases de données qui existent dans le monde Windows et les différents SGBD [Mic 00].

- **Interface de programmation OLEDB et ODBC**

Les commandes d'une application cliente sont envoyées à SQL Server via ces interfaces :

a. *ODBC (Open DataBase Connectivity)* : Est une interface de niveau appel (CLI [2]: Call-Level Interface). Il s'agit d'un format défini par Microsoft permettant la communication entre des clients et des bases de données [Isr 01].

b. *OLEDB (Object Linking and Embedding Data Base)* : Est une API de type COM[3] qui permet un accès universel à diverses sources de données. SQL Server inclut un fournisseur OLEDB natif et grâce à ce fournisseur, SQL Server prend également en charge les objets ou les composants utilisant OLEDB, tels que ActiveX[4], ADO [Mic 00].

---

[2] Une CLI est une API composée des fonctions qu'une application peut appeler pour obtenir un ensemble de services.
[3] COM : Component Object Model : utilisée pour créer une instance d'un objet sur un serveur distant.
[4] Active X : Standard élaboré par Microsoft pour permettre à différents logiciels de communiquer entre eux quel que soit le matériel ou le logiciel utilisés.

- **Objets des données ActiveX**

L'objet de données le plus récent est utilisé pour l'accès aux données OLTP[5] (OnLine Transaction Processing) est : ADO (ActiveX Data Objects).

c. *ADO* : Cette API de base de données définit la manière d'écrire une application pour qu'elle se connecte à une base de données à l'aide de OLE DB et comment transmettre des commandes Transact-SQL à une base de données [Isr 01].

4. Plate forme

- Les serveurs fonctionnent sous Windows 2000 Server.
- Les postes clients fonctionnent sous Windows 98/NT/2000/XP.

5. Processus de développement

Nous résumons le processus de développement des logiciels par le tableau suivant :

Tableau V.1 : Les étapes de développement des produits logiciels

Analyse	- Comprendre les procédures - Recenser les besoins, les données ainsi que les règles - Perspectives du système - Diagnostic - Rapport d'analyse
Conception	- Conception des modèles de données en utilisant un outil d'aide à la conception basé sur la méthode Merise - Vérification et validation des modèles - Génération de la base de données sur SQL Server - Rapport de conception
Développement des applications	- Développement des applications clientes : VB 6.0, Transact-SQL, OLEDB, ODBC et ADO - Développement des procédures stockées sur SQL Server avec Transact-SQL
Test	- Des séances de test et démonstration seront organisées pour les utilisateurs
Empaquetage des applications	- Regroupement des objets de l'application et préparation du programme d'installation pour réaliser le déploiement de l'application

5.1. Les procédures stockées

Une procédure stockée est prévue pour effectuer un traitement sur les données. Elle est une extension du langage SQL Standard et permet de traiter certains problèmes. Elle peut être appelée par un client, depuis une autre procédure stockée ou être exécutée automatiquement par le serveur [Isr 01].

[5] OLTP : un système transactionnel vise la rapidité de mise à jour à l'intérieur de transactions.

- **Les avantages des procédures stockées**

Les avantages découlant de l'utilisation des procédures stockées sont :

a. *Partage de la logique d'application :* Elles peuvent partager une logique d'application avec d'autres applications, contribuant ainsi à garantir la cohérence des accès aux données et de leurs modifications. Elles peuvent encapsuler des règles d'entreprise. Les règles et stratégies d'entreprise encapsulées dans les procédures stockées peuvent être modifiées dans un endroit unique. Tous les clients peuvent utiliser les mêmes procédures stockées afin de garantir la cohérence des accès aux données et de leurs modifications.

b. *Détails des schémas de base de données cachés aux utilisateurs :* Elles évitent aux utilisateurs d'avoir un accès aux détails des tables de la base de données. Si un jeu de procédures stockées prend en charge la totalité des fonctions d'entreprises que les utilisateurs doivent traiter, ceux-ci n'ont pas besoin d'accéder directement aux tables.

c. *Implémentation de mécanisme de sécurité :* Lorsque tous les accès à la base de données sont exécutés par le biais des procédures stockées, il n'est plus nécessaire de permettre à n'importe quel utilisateur d'avoir des droits de modification/suppression sur n'importe quelle table de base de données. Ceci simplifie considérablement les questions de sécurité puisque tous les utilisateurs auraient les droits d'exécution à des classes spécifiques de procédures stockées, selon les droits de sécurité.

d. *Diminution du trafic réseau :* Elles contribuent à la diminution du trafic sur le réseau. Au lieu d'envoyer des centaines d'instructions Transact-SQL sur le réseau, les utilisateurs peuvent effectuer une opération complexe à l'aide d'une seule instruction, ce qui a pour effet de diminuer le nombre de requêtes échangées entre le client et le serveur.

e. *Vitesse :* Un SGBD compile la procédure stockée ce qui accélère sensiblement son exécution à l'opposé d'un énoncé SQL standard, qui est interprété et analysé durant l'exécution.

Conclusion

La connaissance des outils de développement d'une entreprise et l'analyse du processus de développement du logiciel à protéger est très important pour l'élaboration d'une stratégie de sécurité et pour l'instauration d'une technique de protection.

Notre prochain chapitre consiste à établir une méthodologie de protection du logiciel dans l'entreprise durant son processus de développement.

CHAPITRE VI

La protection durant le développement des logiciels

Introduction

Le point de départ de tout modèle de sécurité consiste à mettre en œuvre des normes et des stratégies destinées à protéger le système de tout atteinte extérieure et de tout usage interne illicite. La sécurité des ressources informatiques, des applications et des données associées fait partie intégrante de la stratégie de sécurité d'une entreprise. Sécuriser un système suppose de mettre en oeuvre un ensemble de procédures, de méthodes et de technologies destinées à protéger l'infrastructure informatique, les logiciels et les données associées dans l'ensemble de l'organisation concernée.

La méthodologie de sécurité décrite dans ce chapitre intéressera les responsables de la sécurité informatique, elle a pour but d'aider précieusement les développeurs à protéger leur logiciel dans l'entreprise durant les étapes de développement.

1. La sécurité des locaux

C'est un aspect non négligeable pour la protection des logiciels. C'est une sécurité physique qui constitue l'une des premières barrières à mettre en place. La sécurité de n'importe quel système informatique commence par celles des locaux accueillant les ordinateurs, pour sécuriser ces locaux où est effectué le développement des logiciels il faut :

- Installer le local qui porte le matériel informatique (salle de développement) au centre du bâtiment, dans une pièce ne présentant ni fenêtre ni mur sur l'extérieur.
- Prévoir l'installation d'une salle de surveillance à proximité immédiate de la salle de développement ; Ainsi tous les points d'accès et tous les postes de contrôle environnementaux relatifs au système seront surveillés 24h/24 et 7j/7.
- Il faut restreindre l'accès des locaux de développement en appliquant un contrôle d'identification des personnes par différents systèmes : de badges magnétiques, de clavier sur lequel la personne tape son code personnel (code à chiffres ou lettres ou les deux combinés), d'identification physique (reconnaissance d'empreintes digitales, de la voix ou de la signature), ...etc.
- Ne pas prêter les clés (clé du local et clé du bureau) ; En cas de perte ou de vol de celle-ci, la personne concernée doit le signaler immédiatement au responsable du service informatique (ou bien au responsable de la sécurité informatique s'il existe).

La protection humaine ou «gardiennage » est encore la plus efficace. En effet, seul l'homme peut faire face à des situations les plus inattendues. L'inconvénient de ce mode de protection reste son coût élevé.

Ce n'est pas suffisant de sécuriser les locaux, la sécurité ne se met pas en œuvre en une seule fois, elle fait partie intégrante du cycle de vie du logiciel. Nous décrivons ci-dessous des recommandations et des méthodes utiles pendant les étapes de développement du logiciel (analyse, conception et développement des applications).

2. La protection pendant l'analyse et la conception

Durant les phases : analyse des besoins et conception, l'équipe de développement prépare des documents qui contiennent la définition des besoins (cahier de charge), la définition de la structure modulaire du logiciel et la définition des algorithmes à utiliser (rapport de conception) ; Ces documents sont sous forme de papiers ou sous forme de fichiers stockés sur des supports (disque dur, disquette, CD-Rom). Ces documents doivent être protégés contre la perte, le vol et l'utilisation par des personnes non autorisées par les deux manières suivantes :

- Il faut en premier lieu empêcher l'indésirable de pénétrer dans le système informatique en effectuant un processus d'identification.
- Il faut ensuite contrôler l'accès aux documents sous forme de papiers.

2.1. Protection des documents enregistrés sur machine

Pour garantir que seules les personnes autorisées peuvent accéder aux machines, il convient en premier lieu d'identifier les utilisateurs ; Cela demande la mise en place d'un processus d'identification au cours duquel les utilisateurs doivent prouver leur identité [Ass 82]. L'identification est basée sur un système de mots de passe (c'est le meilleur compromis entre la sûreté et la souplesse).

2.1.1. Choix des mots de passe

Le mot de passe étant un rempart de sécurité essentiel, et pour cela il faut éviter :

- D'utiliser les mots de passe vides.
- De choisir le nom, le prénom, la date de naissance, le nom d'un animal ou de n'importe quoi se rapportant de prés ou de loin à sa propre personne.
- D'employer le thème pour le mot de passe. Exemple : N'employez pas le piratage comme mot de passe si la page est au sujet de piratage.
- De choisir une date ou un événement connu.
- De choisir tout mot ou nom propre se trouvant dans un dictionnaire quelconque.

Par contre il est très favorable de :

- Choisir un mot de passe qui doit comporter plus de huit caractères (selon de récents rapports concernant les problèmes de sécurité, la plupart des programmes de piratage se basent sur un mot de passe de huit caractères), être une combinaison de minuscules, majuscules, chiffres, et caractères de ponctuation.
- Trouver une combinaison d'une phrase qui ait un sens mnémotechnique. Exemple : "j'ai 2 amours mon pays et Alger" pourrait donner "G2ampeA".

2.1.2. Protection des mots de passe

Après avoir judicieusement choisi le mot de passe, il convient maintenant de le protéger pour ne pas le laisser tomber dans de mauvaises mains. Pour cela il faut respecter ces quelques règles simples :

- Les mots de passe doivent être changés fréquemment.
- Ne pas inscrire sur la machine ou tout autre endroit visible les mots de passe.
- Ne pas donner les mots de passe pour quelques raisons que ce soit.
- Il faut prêter attention aux indiscrétions des autres utilisateurs.
- Les mots de passe doivent être suffisamment complexes pour ne pas être trouvés par des individus.

2.1.3. Protection des informations stockées sur les supports magnétiques

Les employés peuvent causer des dommages très importants en quittant leur bureau avec des informations stockées sur des disquettes 3 ½ pouces ou des CD-ROM. Les entreprises doivent en outre contrôler les supports magnétiques pour réduire les pertes de logiciels (applications et systèmes d'exploitation) et n'autoriser que les accès administratifs aux unités de disquette et de CD-ROM.

2.2. Protection des documents sous forme de papiers

Pour protéger les données enregistrées sur papier pendant les phases analyse et conception, il faut prendre en compte les points suivants :

- **Tous les utilisateurs n'ont pas forcément le droit de tout voir**

Ce n'est pas parce qu'un utilisateur peut accéder à un local donné qu'il dispose de tous les droits et qu'il peut tout lire ou tout modifier. Il faut donc se préoccuper de la protection des documents d'analyse et de conception, contenant des informations importantes pour le développement des logiciels. Pour cela il faut :

- Protéger ces documents en utilisant un moyen sûr : porteur accrédité, pli recommandé, container fermant à clef ou les ranger dans des bureaux fermant à clé.
- L'accès à cette documentation doit être réglementé et placé sous la responsabilité d'une personne précise.
- Ne pas mettre à la corbeille à papier n'importe quel document.
- N'effectuer le transfert de ces documents qu'avec bordereau d'accompagnement récapitulant le nombre et la nature des documents transmis.

- **Toutes les données n'ont pas la même valeur**

Il peut être nécessaire de gérer et de protéger les informations de différentes manières. Les entreprises doivent déterminer la valeur des différents types d'informations présents dans les documents d'analyse et de conception afin de pouvoir planifier les niveaux de protection appropriés.

3. Protection pendant le développement des applications

Après la protection des documents d'analyse et de conception, il s'agit maintenant de protéger le logiciel durant la phase de développement. A ce niveau, l'implémentation consiste à développer les applications selon l'architecture adoptée. Comme le développement à TELEMATIS se fait dans une architecture client-serveur, le système se réalise en intervenant sur la partie client (interface, requêtes SQL, appel de procédures

stockées, ...etc.) et sur la partie serveur (création des bases de données, procédures stockées, fonctions, ...etc.).

3.1. Protection des bases de données

Les bases de données sont traitées au niveau du serveur, elles contiennent souvent des informations sensibles. Donc, il faut sécuriser les bases de données et ces éléments (les tables, les vues, ...etc.) comme suit :

- **Protection contre les accès non autorisés**

L'accès à une base de données doit être accordé seulement aux utilisateurs autorisés. Le serveur de base de données inclut une gestion de la sécurité pour empêcher les accès non autorisés. Généralement, le serveur demande un nom d'utilisateur et un mot de passe lors de la procédure de connexion avant d'autoriser l'accès à une base de données. Donc, les utilisateurs sont autorisés d'accéder aux données, quand ils sont identifiés comme étant des utilisateurs autorisés par le système.

- **Cryptage des procédures stockées**

Lors de la création d'une procédure stockée, pour être sûr qu'aucun autre utilisateur ne pourra visualiser cette procédure, on utilise la clause WITH ENCRYPTION. La définition de la procédure stockée est alors dans un format illisible. Une fois cryptée, la définition de la procédure stockée ne peut être décryptée et visualisée par personne, que ce soit son propriétaire ou l'administrateur système.

3.2. Protection des codes sources

Pour protéger le code source, il faut respecter ces règles :

- Eviter de lui attribuer un nom qui exprime son contenu, sinon une simple opération de recherche permet de trouver le code source recherché.
- Eviter d'enregistrer le code source dans le fichier du langage de programmation (qui est proposé par défaut) ou sur le bureau, et il faut l'enregistrer dans des sous répertoires difficiles à trouver.
- Utiliser une méthode de cryptage.

Conclusion

Cette présentation succincte montre clairement qu'une grande partie des données tellement essentielle au développement des logiciels dans une entreprise doit être bien protégée contre l'utilisation non habilitée.

Après l'intégration de la protection dans le processus de développement du logiciel, nous allons voir dans le chapitre suivant la mise en oeuvre d'une technique de protection du logiciel lors de son utilisation sur les postes clients.

CHAPITRE VII

La mise en oeuvre

Introduction

Comme nous l'avons vu dans les chapitres précédents, le développement des logiciels dans notre contexte de travail se fait dans une architecture client-serveur, donc il est très intéressant de centraliser la protection en implémentant les procédures de protection du logiciel sur le serveur, c'est une manière de simplifier et de centraliser les procédures de protection.

Pour réaliser cette protection, nous avons besoin de deux choses : des données et des traitements. Les données sont les paramètres de protection qui vont être enregistrées dans des tables, les traitements sont effectués par des procédures stockées et toutes ces tables et ces procédures stockées sont localisées dans une base de donnée.

Ce chapitre présente les différentes étapes de la mise en œuvre d'une technique de protection du logiciel pour restreindre le déploiement et l'exécution du logiciel que sur un certain nombre de postes clients autorisés.

1. Les paramètres de protection

Dans notre projet, nous utilisons plusieurs paramètres de protection qui sont :

- Module : spécifie le nom de l'application à protéger.
- NDA : c'est un paramètre qui représente le nombre de déploiements autorisés pour le logiciel sur les postes clients.
- NPD : représente le nombre de postes clients où le logiciel est déployé.
- CI : c'est un code d'installation généré par le logiciel lors de son déploiement sur un poste client ; C'est une combinaison de plusieurs empreintes digitales de la machine, il se base sur le fait que chaque ordinateur est unique.
- NCS : c'est le nombre de connexions simultanées autorisées pour les clients exécutant le logiciel.
- NPC : c'est le nombre de postes connectés (en cours d'exécution du logiciel).
- NEA : c'est le nombre d'exécutions autorisées.
- NER : c'est le nombre d'exécutions réalisées.

Remarque : Il est supposé qu'un logiciel en cours de déploiement ou en cours d'exécution soit connecté avec le serveur pour l'accès aux ressources.

Nous divisons ces paramètres de protection en deux classes de paramètres : la première classe est celle des paramètres de déploiement qui sont : NDA, NPD. La deuxième classe des paramètres d'exécution contient : NEA, NER, NCS, NPC. Les deux paramètres : Module et CI sont utilisés dans les deux cas de déploiement et d'exécution (voir Figure VII.1).

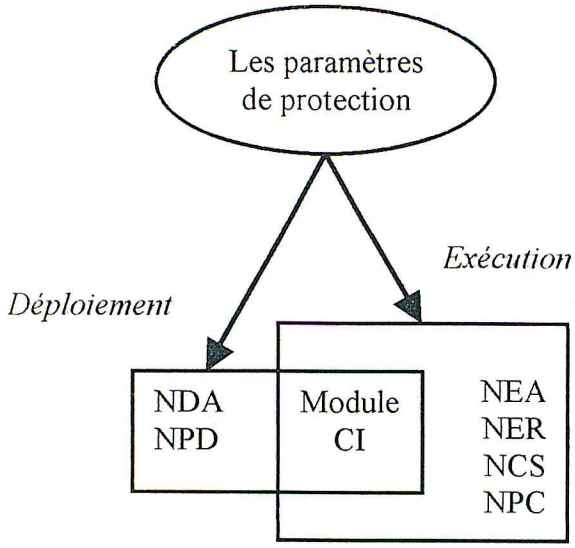

Figure VII.1 : Représentation des paramètres de protection

2. Création de la base de données de protection

La base de données qui contiendra les paramètres permettant de protéger le logiciel est constituée de quatre tables créés sur le serveur SQL, dont le système fera accès à chaque tentative de déploiement ou d'exécution pour vérifier l'autorisation ou non de l'opération en utilisant des procédures stockées.

Pour cette base de données que nous l'appelons BDP (Base de Données de Protection), nous proposons l'architecture suivante :

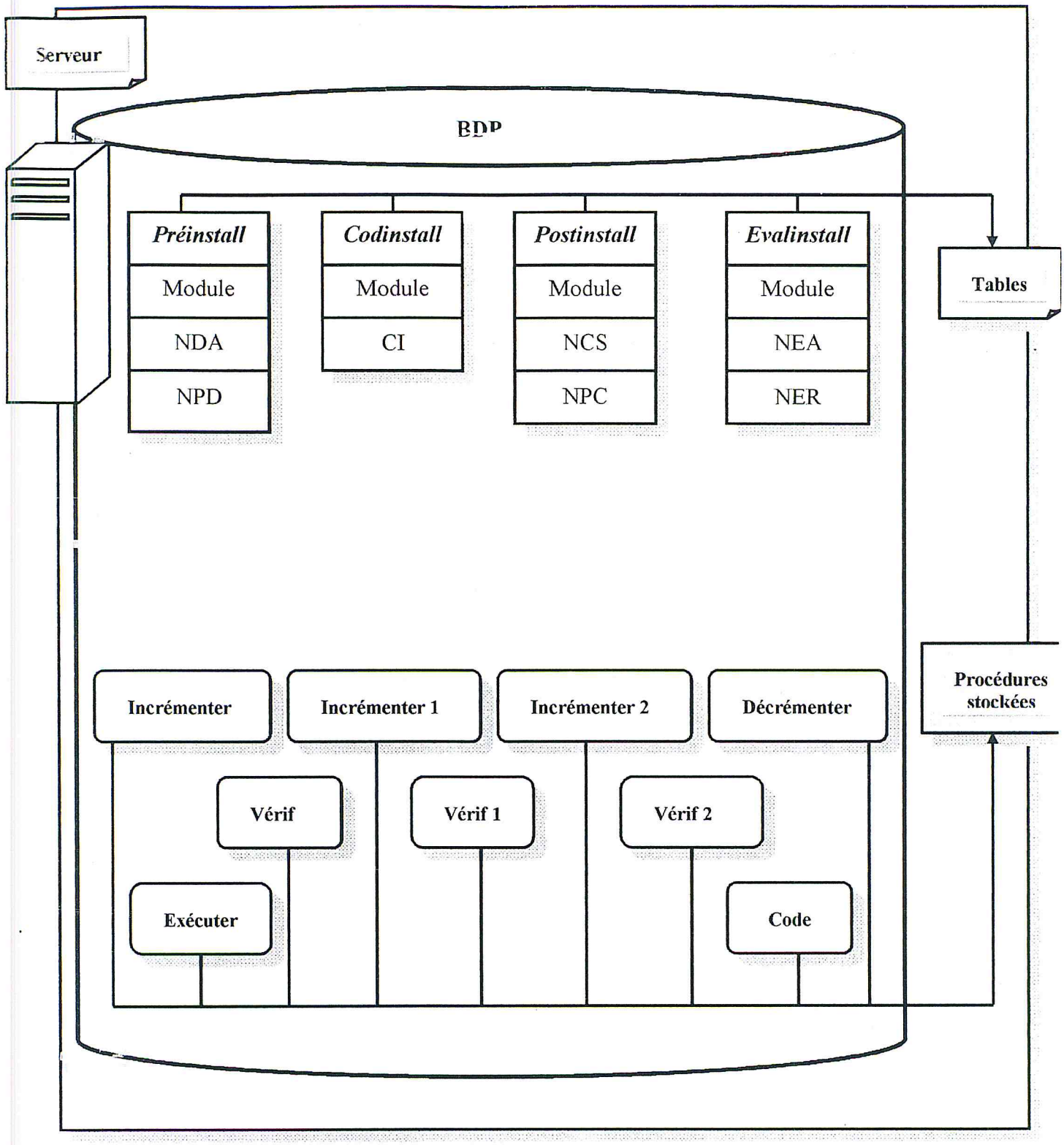

Figure VII.2 : Architecture de la base de données BDP

2.1. Création des tables

Dans notre travail, la création des tables est la partie la plus importante et la plus fondamentale pour la mise en place des données dans la base de données de protection (BDP). Donc, nous devrons créer les tables où les paramètres de protection seront stockés, ces tables sont les suivantes :

a. *Table Préinstall* : Cette première table contient les paramètres utilisés lors de déploiement du logiciel, ce dernier ne peut être installé que si les paramètres de protection sont vérifiés. Ces paramètres sont : Module, NDA et NPD comme l'indique la table dans la figure suivante :

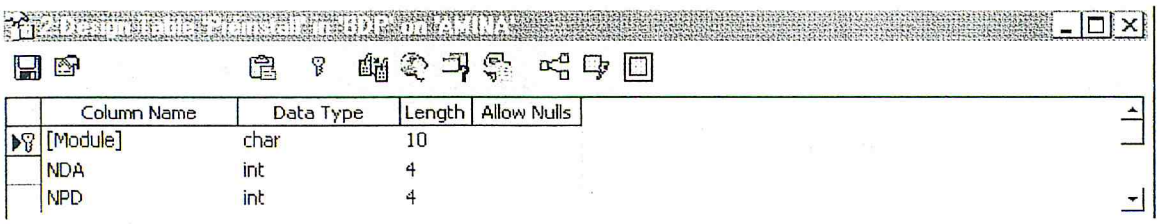

Figure VII.3 : Conception de la table *Préinstall*

b. *Table Codinstall* : Elle contient le paramètre Module et tous les CI des postes clients autorisés pour le déploiement, sa structure est la suivante :

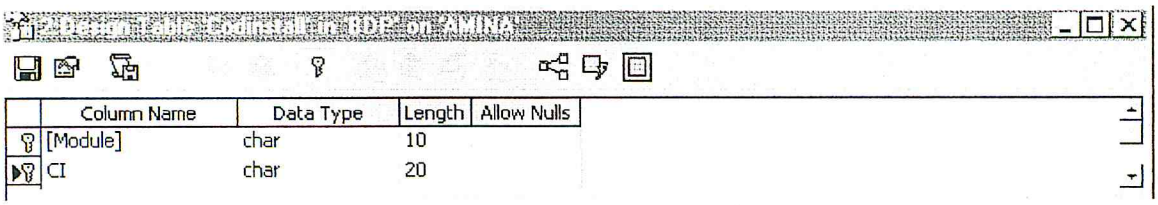

Figure VII.4 : Structure de la table *Codinstall*

c. *Table Postinstall* : Elle contient les paramètres suivants : Module, NCS, NPC, ces paramètres sont vérifiés lors de l'exécution du logiciel. *Postinstall* est structurée comme suit :

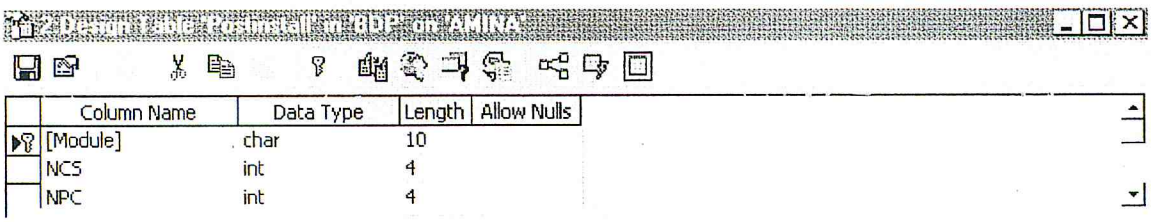

Figure VII.5 : Structure de la table *Postinstall*

d. *Table Evalinstall* : Elle contient les paramètres suivants : Module, NEA, NER, ces paramètres sont vérifiés lors de l'exécution du logiciel en version d'évaluation. *Evalinstall* est comme suit :

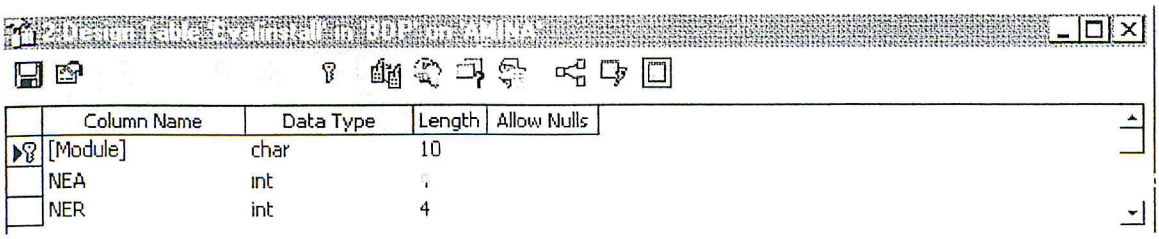

Figure VII.6 : La table *Evalinstall*

Remarque : La clé dans les tables *Préinstall, Postinstall et Evalinstall* est le paramètre Module parce que, c'est par ce paramètre qui porte le nom du logiciel que nous pouvons distinguer entre les paramètres de protection de chaque logiciel dans le cas où nous aurions plusieurs logiciels à protéger. Dans la table *Codinstall* la clé est la concaténation des deux paramètres Module et CI.

2.2. Création des procédures stockées

Un ensemble de procédures stockées est créé pour faire des traitements sur les paramètres des tables de BDP pour protéger le logiciel lors de son utilisation. Un certain nombre de ces procédures est utilisé lors de déploiement, d'autres lors de l'exécution du logiciel ; Ces procédures sont les suivantes :

- **Les procédures stockées de déploiement**

Lors de déploiement du logiciel sur un poste client, les procédures stockées suivantes sont utilisées :

a. *Vérif* : C'est la première procédure stockée qui doit être appelée lors de déploiement du logiciel, elle vérifie la possibilité d'installer le logiciel ou non sur un poste client, en comparant le nombre de déploiements autorisés avec le nombre de postes clients déjà déployés (NDA et NPD), elle retourne un paramètre P qui vaut 0 quand NPD=NDA et donc le déploiement du logiciel est impossible, sinon le déploiement est possible.

b. *Code* : Après la vérification de la possibilité de déploiement, le logiciel génère un code d'installation (CI), c'est un code unique pour chaque poste client, l'utilisateur doit insérer une clé d'activation pour pouvoir continuer le déploiement du logiciel. Le rôle de cette procédure est d'insérer ce code d'installation à l'enregistrement CI de la table *Codinstall* après l'insertion de la clé d'activation.

c. *Incrémenter* : Après que le déploiement du logiciel est terminé sur un poste client, cette procédure doit incrémenter le nombre de postes déployés (NPD).

- **Les procédures stockées d'exécution**

Toutes les procédures stockées qui doivent être appelées lors de l'exécution du logiciel sur un poste client sont décrites ci-dessous :

a. *Exécuter* : C'est la première procédure stockée qui doit être exécutée à chaque tentative d'exécution du logiciel, elle vérifie que le code unique de la machine où le logiciel est en cours d'exécution existe dans les différents CI enregistrés dans la table

Codinstall, elle retourne un paramètre Q qui vaut 0 quand les codes sont différents et donc l'exécution du logiciel est impossible, sinon Q vaut 1 et dans ce cas on peut commencer l'exécution.

b. *Incrémenter1* : Lors de l'exécution du logiciel, cette procédure doit incrémenter le nombre de postes connectés au serveur (NPC).

c. *Vérif1* : Cette procédure vérifie à chaque exécution du logiciel le nombre de connexions simultanées autorisées avec le nombre de postes en cours de connexion (NCS et NPC) dans la table *Postinstall*, elle retourne un paramètre L qui vaut 0 quand le NCS=NPC et donc l'exécution du logiciel est non autorisée, sinon l'exécution est possible.

d. *Décrémenter* : Cette procédure doit décrémenter le nombre de postes connectés au serveur (NPC) lors de fermeture de l'exécution du logiciel sur un poste client.

e. *Vérif2* : C'est une procédure stockée qui vérifie la possibilité d'exécuter le logiciel ou non sur un poste client, en comparant le nombre d'exécutions autorisées avec le nombre d'exécutions déjà faites (NEA et NER), elle retourne un paramètre M qui vaut 0 quand NEA=NER et donc l'exécution du logiciel est impossible, sinon l'exécution est possible.

f. *Incrémenter2* : Cette procédure doit incrémenter le nombre d'exécutions réalisées (NER) lors de l'exécution du logiciel.

Les procédures stockées *Vérif1*, *Incrémenter1* et *Décrémenter* sont utilisées pour contrôler le nombre de connexions simultanées du logiciel en version finale et les procédures *Vérif2* et *Incrémenter2* sont utilisées pour contrôler le nombre d'exécutions du logiciel en version d'évaluation.

Remarque : A toutes ces procédures stockées décrites ci-dessus est ajoutéela clause : [WHERE Module ='nom du logiciel'], donc la procédure ne peut être exécutée que si elle vérifie le nom de l'application logicielle à protéger.

3. La connectivité à BDP

La réalisation de notre projet est basée sur SQL Server pour la partie serveur, un langage de programmation pour la partie cliente et un middleware (intermédiaire) pour établir la connexion entre le serveur et le client.

3.1. Choix du langage de programmation

Une des décisions les plus importantes dans le développement est le choix du langage de programmation. Notre choix est porté sur le langage Visual Basic 6.0 parce que d'une part nous voulons travailler dans un environnement homogène puisqu'il est utilisé pour le développement des logiciels dans notre contexte de travail, et d'autre part parce qu'il permet de travailler avec des données orientées client-serveur comme SQL Server en facilitant la connexion entre VB et SQL Server par l'intermédiaire du modèle ADO.

3.2. ADO

Quatre objets majeurs sont à notre disposition dans le modèle ADO, ils sont représentés par le schéma suivant :

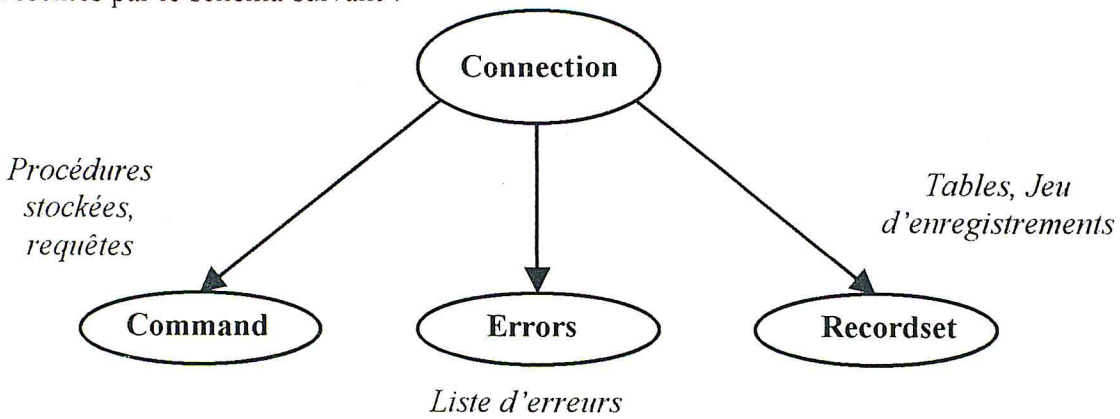

Figure VII.7 : Modèle objets des ADO

L'objet *Connection* garantit l'établissement d'une connexion vers le serveur de base de données, le programme utilisateur peut envoyer des requêtes par l'objet *Command* sous forme de chaîne (SELECT * FROM Table) ou exécuter des traitements sous forme de procédures stockées. L'objet *Recordset* détermine les propriétés des enregistrements. Et en fin la collection *Errors* quant à elle nous renseigne sur les erreurs qui peuvent survenir durant la connexion établie.

ADO s'appuie sur l'API OLEDB pour réaliser les différentes opérations qui exigent l'accès à une base de données ; Le couple ADO-OLEDB offre un moyen simple, souple, puissant et performant de connexion à des bases de données relationnelles ou non [Isr 01] (voir Figure VII.7).

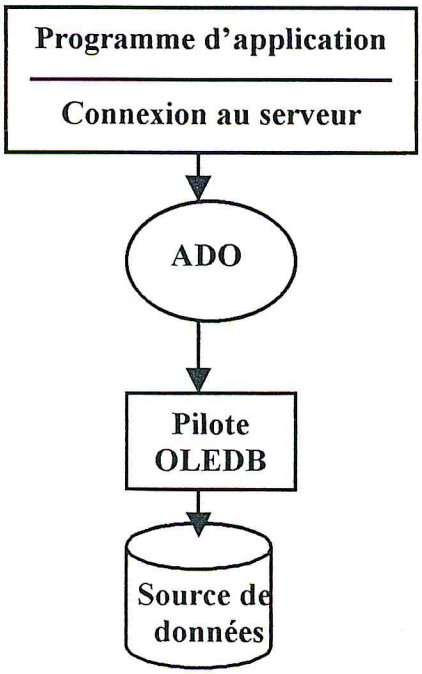

Figure VII.8 : Schéma d'accès aux sources de données

4. Déploiement et exécution du logiciel

Le schéma ci-dessous représente les étapes de déploiement d'un logiciel protégé sur un poste client, en supposant que le logiciel vérifie toutes les conditions de déploiement :

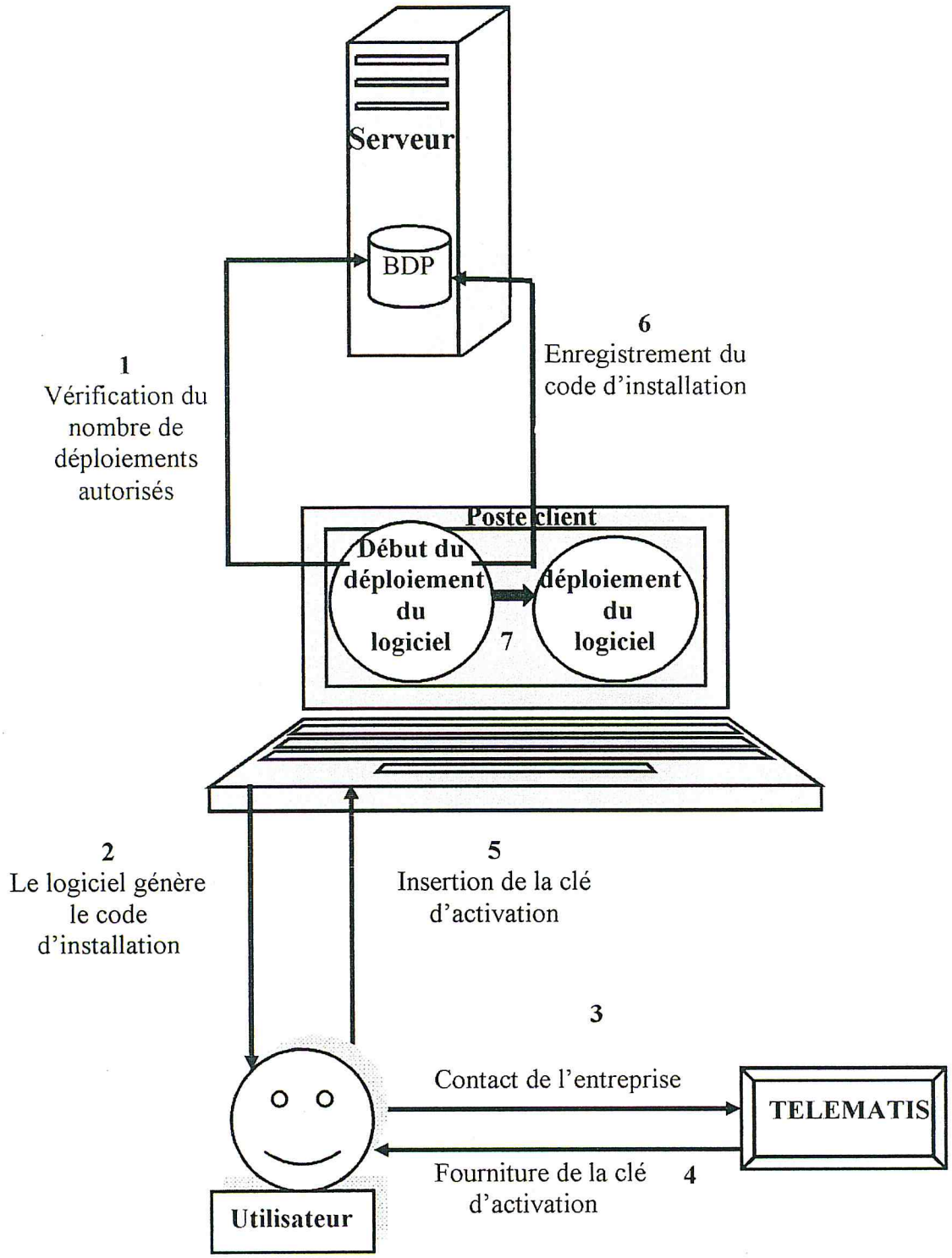

Figure VII.9 : Les étapes de déploiement d'un logiciel protégé contre le piratage

Lors de déploiement du logiciel, la procédure *Vérif* vérifie dans BDP que ce déploiement ne dépasse pas le nombre de déploiements autorisés (1), un code d'installation est alors généré par le logiciel (2), l'utilisateur contacte ensuite l'entreprise via un site Internet, téléphone ou fax afin de transmettre le code d'installation généré par le logiciel et de récupérer la clé d'activation du logiciel (3,4), il insère cette clé (5), le code généré par le logiciel est alors enregistré dans la table *Codinstall* de BDP à travers la procédure stockée *Code* (6) et le logiciel peut être convenablement installé (7).

Le schéma ci-dessous représente l'exécution d'un logiciel protégé contre le piratage sur un poste client, en supposant que le logiciel vérifie toutes les conditions et les procédures d'exécution :

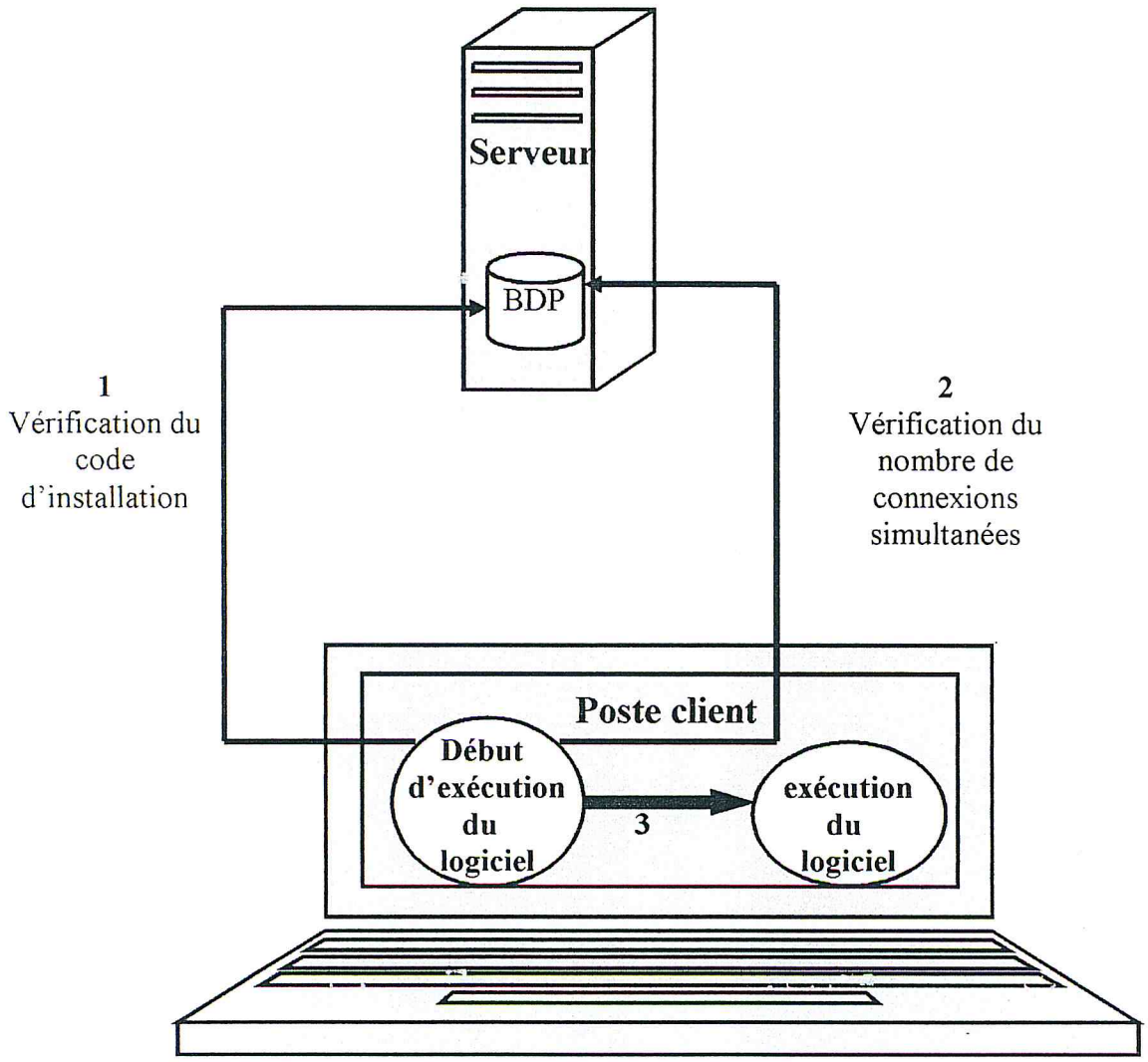

Figure VII.10 : Les étapes d'exécution d'un logiciel protégé contre le piratage

Lors de l'exécution du logiciel, la procédure stockée *Exécuter* vérifie que le code correspondant à la machine où le logiciel est en cours d'exécution existe parmi les codes d'installation enregistrés dans *Codinstall* de BDP (1), ensuite la procédure *Vérif1* vérifie le nombre de postes connectés simultanément (2). Après ces vérifications, le logiciel peut être normalement exécuté (3).

Pour les versions d'évaluation, si le logiciel vérifie le nombre d'exécutions autorisées, il doit être exécuté normalement.

5. Organigrammes et algorithmes

La représentation schématique des étapes de déploiement et d'exécution d'un logiciel protégé contre le piratage sur les postes clients est indiquée par les organigrammes et les algorithmes ci-dessous :

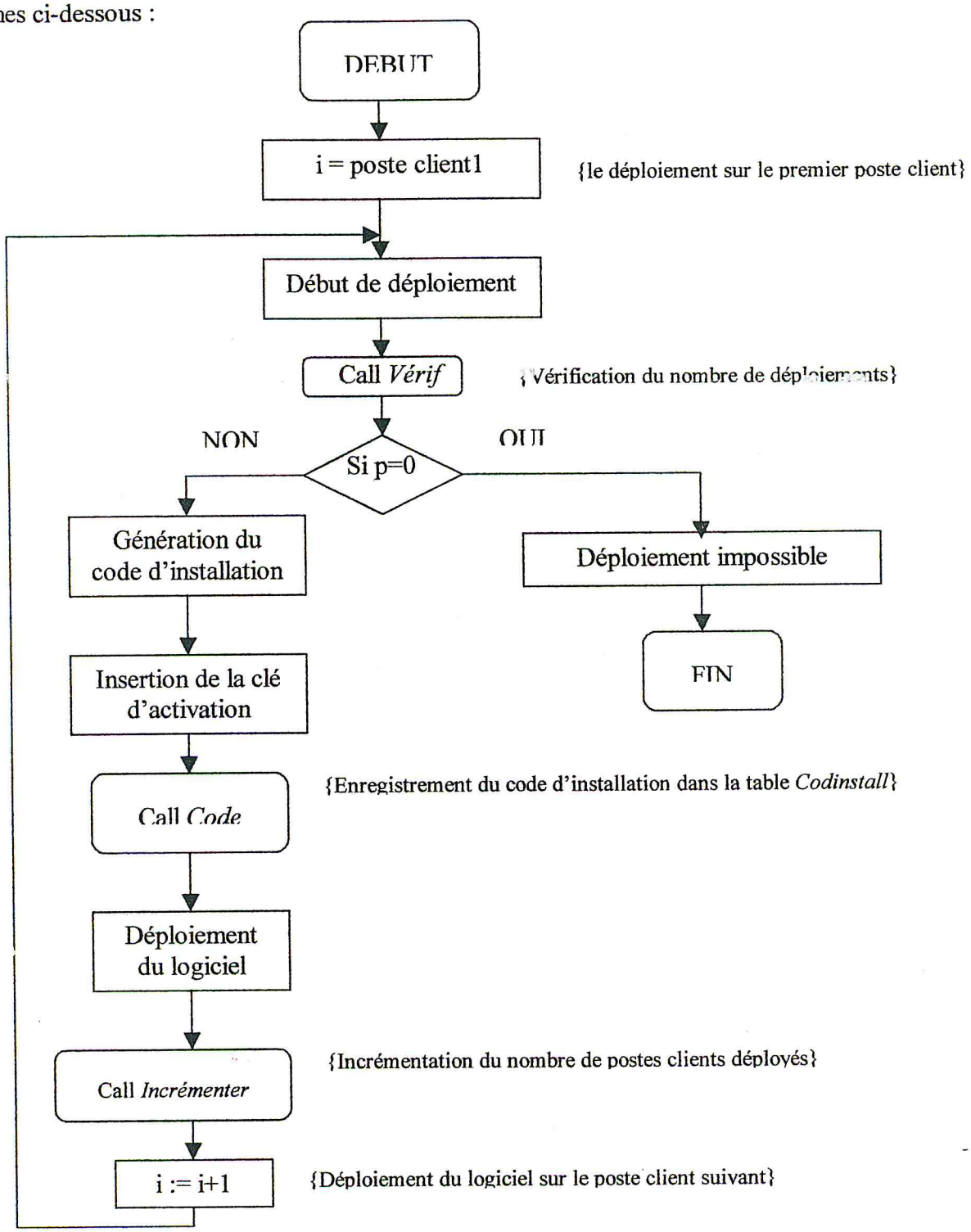

Figure VII.11 : Organigramme de déploiement

Nous pouvons déduire l'algorithme de déploiement suivant à partir de l'organigramme de déploiement précédent :

Algorithme de déploiement

```
Pour i : = client1 à client n
  Faire
      Etablir la connexion entre le client et le serveur
      Appel procédure stockée Vérif {Vérification du nombre de déploiements}

      Si P=0 alors Déploiement impossible
          sinon
            Début

              Génération du code d'installation unique de la machine
              Insertion de la clé d'activation du logiciel
              Appel procédure stockée Code {Enregistrement du CI dans La table Codinstall}
              Déploiement du logiciel
              Appel procédure stockée Incrémenter {Incrémentation du nombre de postes
               déployés}
            Fin
      Fin Si
     i: = i+1 {Déploiement sur le poste suivant}
  Fait
```

L'organigramme qui présente les étapes d'exécution est le suivant :

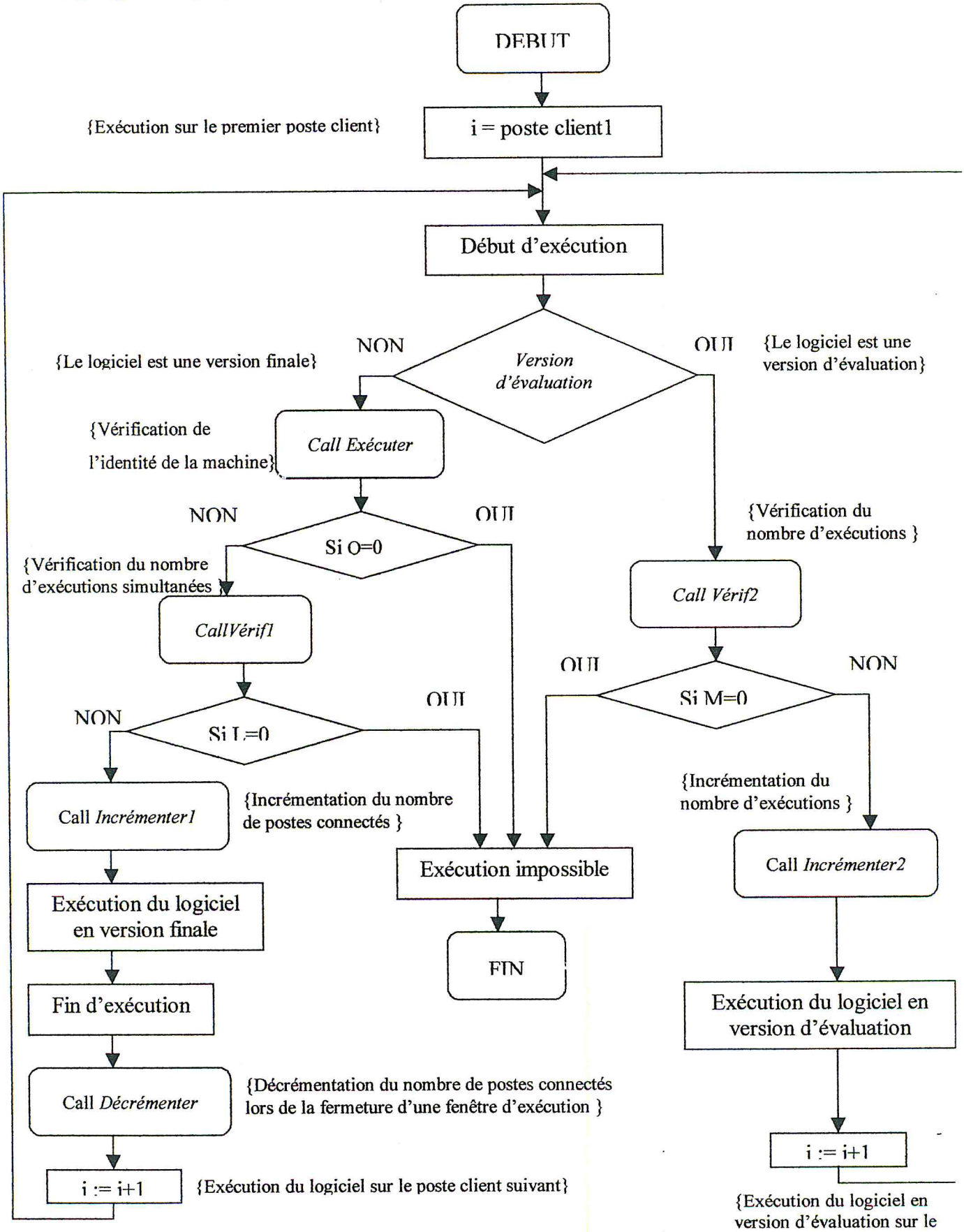

Figure VII.12 : Organigramme d'exécution

De cet organigramme d'exécution, nous déduisons l'algorithme d'exécution ci-dessous :

Algorithme d'exécution

```
Pour i : = client1 à client n
Faire
    Etablir la connexion entre le client et le serveur
    Si le logiciel est la version finale
        Alors Call Exécuter {Vérification de l'identité de la machine}
        Début
            Si Q=0 alors exécution impossible
                Sinon
                Début
                    Call Vérif1 {Contrôle le nombre de connexions simultanées}
                    Si L=0 alors exécution impossible
                        Sinon
                            Début
                                Exécution possible
                                Call Incrémenter1{Incrémentation du nombre de postes connectés }
                                Fin d'exécution
                                Call Décrémenter {Décrémentation du nombre de postes connectés }
                            Fin
                    Fin Si
                Fin
            Fin Si
        Fin
        Sinon {Le logiciel est une version d'évaluation}
        Début
            Call Vérif2 (Contrôle le nombre d'exécutions)
            Si M=0 alors exécution impossible
                Sinon
                Début
                    Exécution possible
                    Call Incrémenter2{Incrémentation du nombre d'exécutions réalisées}
                Fin
            Fin Si
        Fin
    Fin Si
    i :=i+1{Exécution sur le poste client suivant}
Fait
```

6. Choix de la méthode de cryptage

Le code généré par le logiciel lors de son déploiement sur les postes clients est une empreinte de la machine cryptée, ainsi que la clé d'activation insérée par l'utilisateur est un cryptage de l'empreinte de la machine.

Pour la méthode de cryptage, nous avons choisi le système à clé secrète, parce qu'il est simple et il est mille fois plus rapide que le système à clé privée [Nai 98].

7. Test d'application de comptabilité

Pour tester notre système de protection, nous avons utilisé l'application de comptabilité.

7.1. Installation du logiciel

L'installation de ce logiciel n'est autorisée que pour deux postes clients seulement, donc NDA=2 et NPD=0.

- ***Installation sur le premier poste client***

Lors de l'installation du logiciel de comptabilité sur le premier poste client, la fenêtre suivante est affichée :

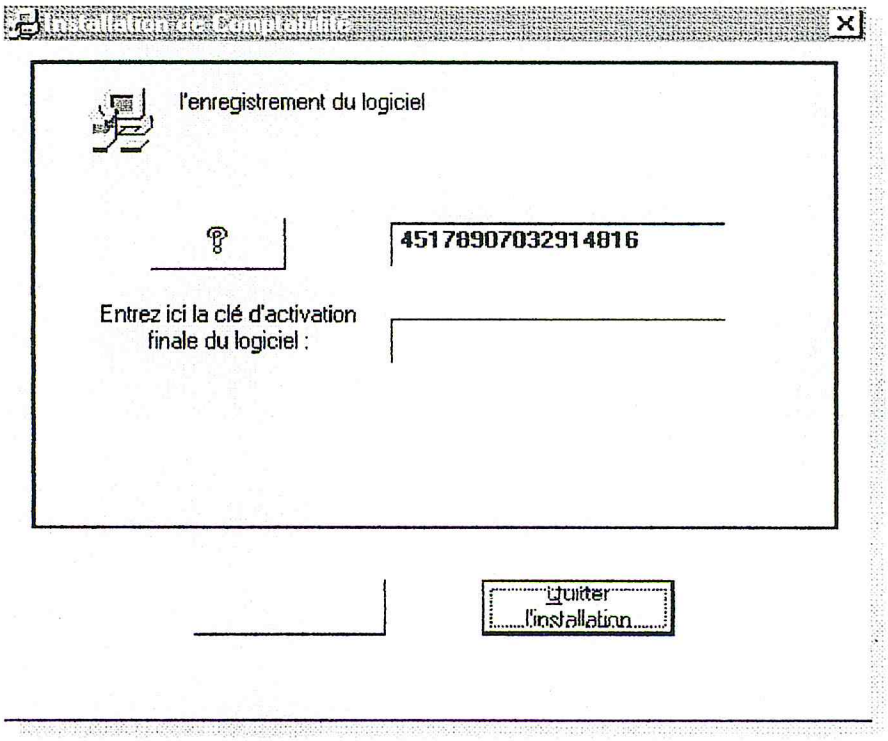

Figure VII.13 : La fenêtre d'authentification du premier poste client

L'utilisateur constate qu'un code est généré, s'il clique sur le bouton : [?], le message suivant est affiché :

Figure VII.14 : Fenêtre d'information sur le code d'installation

Donc, il faut que l'utilisateur contacte TELEMATIS afin de transmettre le code généré par le logiciel et récupérer la clé d'activation du logiciel.

Le bouton : est désactivé jusqu'au moment où la bonne clé est insérée. Dés l'insertion de la bonne clé, le bouton est alors activé comme l'indique la figure ci-dessous et l'utilisateur peut continuer le déploiement du logiciel normalement.

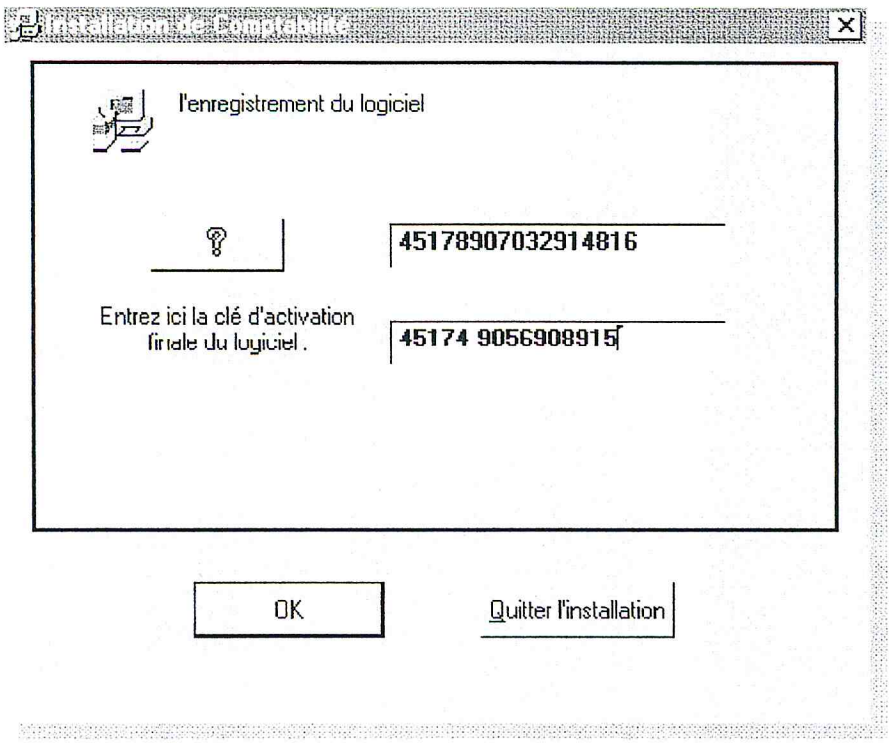

Figure VII.15 : L'insertion de la clé d'activation du premier poste client

- ***Installation sur le deuxième poste client***

Lors de l'installation du logiciel sur le deuxième poste client la même fenêtre précédente est affiché, la différence est dans le code généré par le logiciel (voir figure suivante).

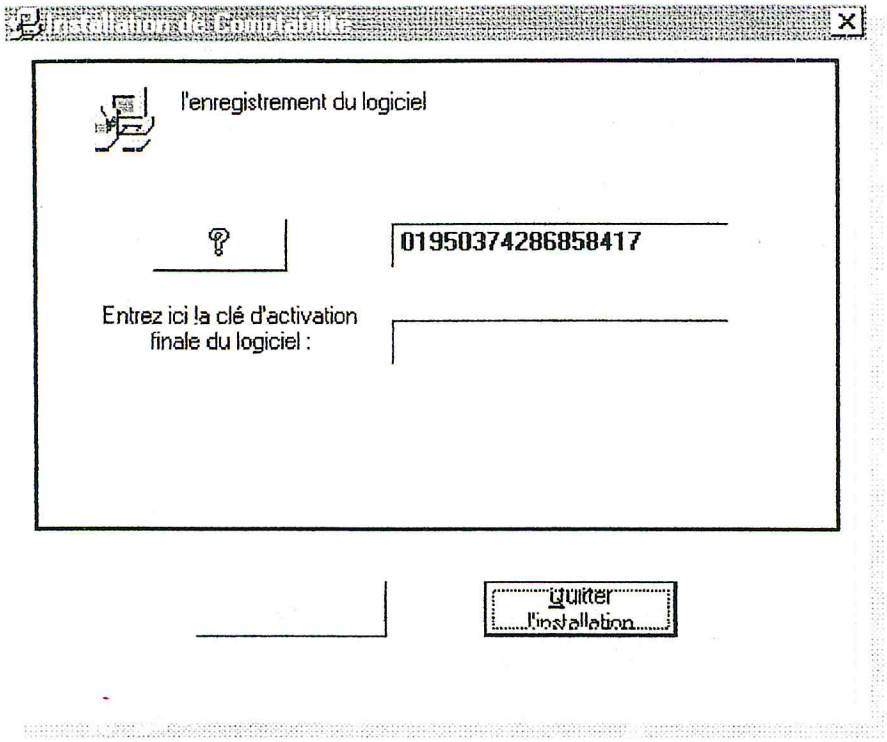

Figure VII.16 : La fenêtre d'authentification du deuxième poste client

Si l'utilisateur essaye d'entrer la clé d'activation insérée dans le premier poste client, il ne pourra pas continuer l'installation (voir Figure VII.17), donc chaque poste client a son propre code généré et propre clé d'activation.

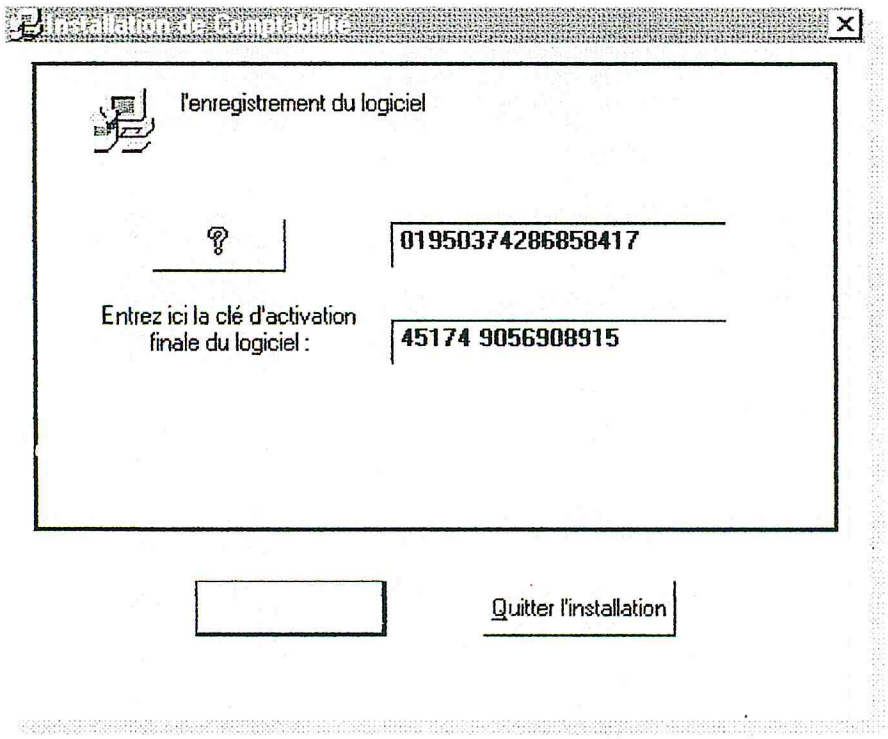

Figure VII.17 : L'insertion de la clé d'activation du premier poste client

L'utilisateur contacte TELEMATIS et insère la clé d'activation, le bouton est activé et il peut continuer l'installation du logiciel (voir Figure ci-dessous).

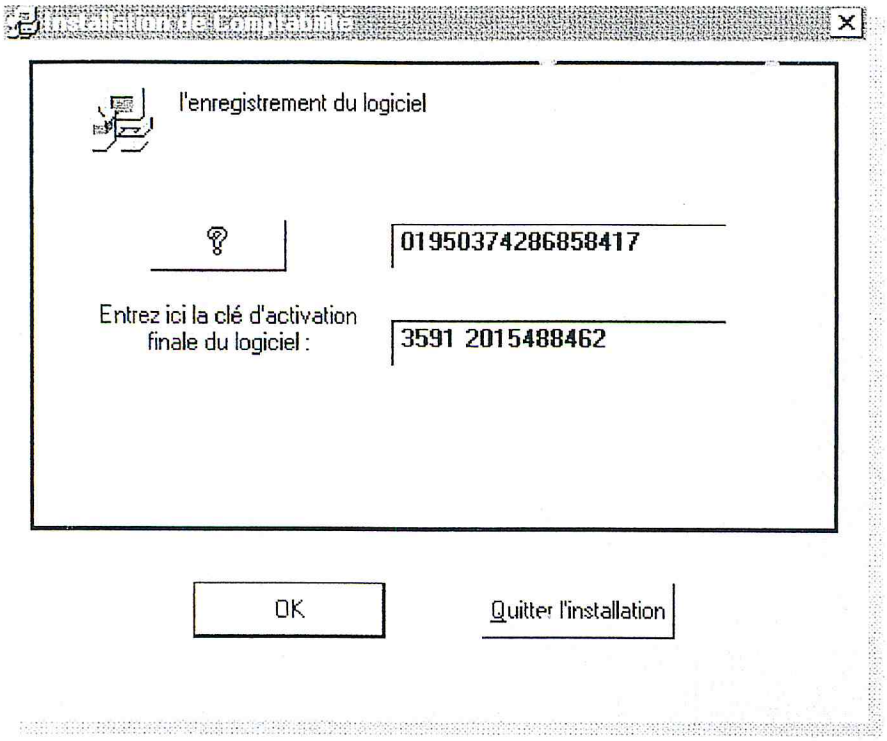

Figure VII.18 : L'insertion de la clé d'activation du deuxième poste client

- ***Installation sur un troisième poste client***

L'utilisateur n'a le droit de déployer le logiciel que sur deux postes clients, s'il essaye quand même de le déployer sur un troisième poste client, le message suivant est affiché :

Figure VII.19 : Autorisation d'installation

Il ne peut que cliquer sur le bouton : OK pour sortir de l'installation.

7.2. Exécution du logiciel

Lors de l'exécution du logiciel sur les postes clients où il est installé, si les deux conditions d'exécution décrites précédemment sont vérifiées (l'identité de la machine et le nombre d'exécutions simultanées), la fenêtre d'exécution du logiciel est affichée.

Sinon, dans le cas où une condition ne serait pas vérifiée la fenêtre suivante est affichée :

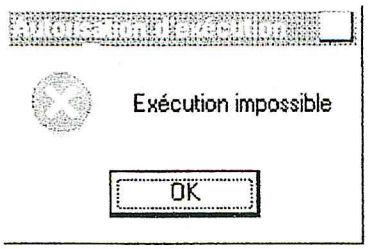

Figure VII.20 : Autorisation d'exécution

Conclusion

Dans ce dernier chapitre, Nous avons présenté la méthode de développement de notre système de protection en utilisant l'architecture client-serveur comme un environnement de développement.

Nous avons commencé par la création de la base de données de protection sur le serveur, elle contient les paramètres de protection enregistrés dans les quatres tables (*Préinstall*, *Codinstall*, *Postinstall et Evalinstall*) et les procédures stockées qui doivent être appelées et exécutées à chaque tentative de déploiement ou d'exécution du logiciel pour effectuer un traitement sur les paramètres de protection. Lors de l'installation du logiciel, ce traitement permettra de vérifier le nombre de déploiements autorisés ; il permettra aussi de vérifier l'identité de la machine et le nombre d'exécutions simultanées lors de l'exécution du logiciel, ainsi que le nombre d'exécutions pour les logiciels en version d'évaluation. Nous avons testé notre système de protection en utilisant un logiciel de comptabilité.

A la fin, nous signalons qu'il est fortement recommandé que le déploiement d'une application sur une architecture client-serveur se fasse par les développeurs informatique.

Conclusion générale

La protection des logiciels est souvent considérée comme une tâche technique, généralement réservée à des spécialistes Ce projet nous a permis de comprendre que la sécurité de manière générale et la protection du logiciel en particulier ne se limite pas à des solutions techniques car celles ci ne sont pas infaillibles. La protection du logiciel est une stratégie qui englobe les techniques ainsi qu'un processus permettant à tout intervenant qu'il soit développeur, chef de projet, concepteur, etc. de prendre les précautions et les bonnes habitudes qui minimisent les failles mettant le logiciel en danger. Le terme stratégie implique que ces précautions et bonnes habitudes soient rédigées pour constituer un document référence au sein de l'entreprise.

L'objectif primordial de ce mémoire consiste à bien cerner le domaine de la protection des logiciels, nous avons œuvré pour offrir à TELEMATIS une protection efficace de ses logiciels contre le piratage, nous avons établi une stratégie qui consiste à protéger le logiciel durant ses phases de développement avec des recommandations précises, nous avons élaboré un système qui peut protéger le logiciel non seulement lors de son déploiement mais pendant son exécution aussi. Ainsi nous espérons avoir atteint l'objectif essentiel de cette étude.

Ce projet nous a permis de nous mettre en contact direct avec la vie professionnelle, il nous a été bénéfique dans la mesure où il a contribué à compléter notre formation sur le plan pratique.

Ce projet nous a de plus permis d'approfondir nos connaissances sur beaucoup de concepts tels que : l'architecture client-serveur, le serveur de bases de données SQL Server, les procédures stockées et sur le langage de programmation VB 6.0.

Ce travail nous a révélé un certain nombre de perspectives qui sont résumées comme suit :

- La base de données des paramètres (BDP) implantée sur le serveur nécessite elle même une protection.
- En dehors des recommandations présentées pour la protection des codes sources, ces derniers méritent aussi d'être protégés par une technique comme le cryptage.
- Des tests très sérieux doivent être effectués sur des applications en production pour déceler toutes les failles qui persistent.

Bibliographies - webographie

[Ass 83] : Association française de normalisation ;
Sécurité informatique, protection des données ;
Afnor, Paris; 1983.

[Ben 01] : Benchmark Group : JDN et Solutions ;
Protection des produits Microsoft de la gamme XP ;
http://solutions.journaldunet.com/010713-activation-winxp.shtml ;Juillet2001.

[Bou 99] : Nacer Boudjlida ;
Base de données et systèmes d'information, Le modèle relationnel : langages, systèmes et méthodes ;
Dunod, Paris ; 1999.

[Brè 94] : John Brès ;
Ateliers de génie logiciel : réalités et tendances ;
Masson ; 1994.

[Fra 02] : Frantz Gérard ;
Visual Basic 6 : le guide du programmeur ;
OEM édition ; 2002.

[Gau 96] : Marie Claude Gaudel - Bruno Marre - Françoise Schlienger - Gilles Bernot ;
Précis de génie logiciel ;
Masson ; 1996.

[Inf 01] : http://www.Infoworld.com/articles/hn;
Microsoft et XP : protection de la propriété par la force brute ;
CyberRelu Edition ; Vol 2 n°=17 ; mai 2001.

[Isr 01] : Marc Israel ;
SQL Server 2000 ;
Eyrolles ; octobre 2001.

[Kob 98] : Neal Koblitz ;
A Course in Number Theory and Cryptography ;
Springer-Verlag ; ISBN: 0-387-94293-9 ;1998.

[Mar 95] : Xavier Marsault ;
Compression et cryptage des données multimédias ;
Hermes ; Paris ; 1995.

[May 97] : Maylis Pradeau - Julie Sallenave - Florent Michel ;
Informatique et liberté ;
http://www.mygale.org/07/hackview/ ; 1997.

[Mey 86] : Bertrand Meyer ;
Génie logiciel, techniques de l'ingénieur ;
H2050 ; 1986.

[Mic 00] : Microsoft training ;
Programmation d'une base de données Microsoft SQL Server 2000 ;
Manuel de travail du stagiaire, volume 1,2,3 ; 2000.

[Mic 01] : *L'impact économique du piratage, Les différentes formes de piratage* ;
http://www.Microsoft.com/france/logiciel original/piratage ; 2001.

[Mie 98] : Mielcarek François - Maudoux Xavier - WillyBiro Eric ;
La sécurité informatique ;
http://www.Telecom.epita.fr ; Mai 1998.

[Nai 98] : http://www.Nai.com ;
Une introduction à la cryptographie ;
Network Associates. Inc ; USA ; 1998.

[Net 02] : CR Netzer – Joly ;
La protection des logiciels par le brevet et par le droit d'auteur ;
http://www.NJuris.net/dossiers.aspx ; le 02/05/2002.

[Pil 01] : Jean François Pillou ;
La cryptographie ;
http://www.commentcamarche.net/cryptologie ; 2001.

[Ref 01] : CD-Rom du compilateur Delphi 5,Version professionnelle ;
Edition Borland.

[Rén 01] : Rény Lentzner ;
Visual Basic 6 et les bases de données ;
OEM édition, Paris ;2001.

[Riv 77] : R.Rivest - A.Shamir - L.Adleman ;
A method for obtening digital signatures and public key-cryptosystems ;
Laboratory for Computer Sciences ; 1977.

[Roy 70] : Royce, W.W ;
Managing the development of large software systems, concepts and technique ;
Proc.Wescon ; 1970.

[Run 02] : Hugo Runjolet ;
Les moyens techniques de protection des logiciels ;
http://www.NJuris.net/dossiers.aspx ; le 02/05/2002.

[Sen 94] : Rainbow Technologies ;
Guide de développeur SentinelPro ;
1994.

[Som 88] : Ian Sommerville ;
Le génie logiciel et ses applications ;
InterEdition, Paris ; 1988.

[Viv 97] : Thomas Vivest.
La cryptographie ;
http://www.dmi.ens.fr/equipes/grecc/Crypto/intro/intro.html ; 1997.

[Zim 98] : Philip R. Zimmermann ;
Cryptographie pour Internet ;
Paru dans Scientific American ; Octobre 1998.