Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université SaâdDahlabde Blida
USDB.
Faculté des sciences.
Département informatique.

Mémoire de fin d'études pour l'obtention du Diplôme de Master en Informatique

Option : Ingénierie du logiciel

thème :

Un Système Pour La Conception Et La Production De Documents Authentifiables

Présenté par :

ZIANE Hanane

BENHISSEN Imene

Dirigé par :

Mr Dj.BENNOUAR

président : Boucetta

examinateurs : - Touahri
- Boughraba

Promotion : 2010/2011

Remerciements

Avant tout nous remercions le Bon Dieu de nous avoir donné la force d'achever ce projet de fin d'études , représentant le couronnement d'un cursus universitaire laborieux,ElHamdouLilah.

Nous tenons à remercier particulièrement notre promoteur Mr Bennouar de nous avoir orienté avec ses précieux conseils et pour l'aide qu'il nous a apporté.

Nous remercions également nos enseignants du département d'informatique de l'université de Blida, un remerciement particulier à mademoiselle FARHI du centre de recherche de Baba Hessen.

Nous remercions toute personne ayant contribués de près ou de loin à la mise en œuvre de ce projet.

Dédicaces

A mes très chers parents qui ont toujours été là pour moi, qui m'ont donné un magnifique modèle de labeur et de persévérance. J'espère qu'ils trouveront dans ce travail toute ma reconnaissance et tout mon amour

A mes anges gardiens, mestrès chère sœurs Amel et Kamelia

A mon frère Islam gardien infatigable de mon bien être

A mon beau-frèreFarouk

A mes grands Parents ,mes tantes et mes oncles

A mes trèschères copine Yousra,Amina et Nadjat.

A mon promoteur Mr Bennouar.

A toutes les personnes ayant contribué de prés ou de loin a l'accomplissement de ce modeste travail.

Z.Hanane

Dédicaces

Je dédie ce mémoire

À mon père Omar et ma mère Assia pour leur soutien durant toute ma carrière

Pour leur bienveillance, leurs effort constants dans mes études

Et pour leurs encouragements.

À mon très chère époux Samir BELLAL qui ma soutenu toute au long de mon travaille et toute ma belle famille.

À mes chères frères Fethi, Rédha, Walide , Amine et ca femme Souad et leurs petite fille Ranya.

À mon binome Hanane et toute sa famille.

À toute mes amies et à tous ceux qui m'ont encouragé.

BENHISSEN Imene

ملخص

الهدف من هذا العمل هو تحقيق تطبيق الالكتروني يسمح للمؤسس لانتاج وثائق ورقية أو رقمية على أنه من الممكن لمصادقة من خلال شبكة الانترنت. يتم إجراء عملية المصادقة من الورق بصريا من خلال مقارنة ورقة وثيقة في نفس الوثيقة التي ترد على الموقع الإلكتروني للمؤسسة. عملية المصادقة على الوثيقة الالكترونية على أساس «watermarking»توقيع الرقمي. فمن الذي سيقدم إلى موقع المؤسسة على وثيقة رقمية أن الموقع سوف تعترف لنفسه أم لا.

كلمات البحث: التوقيع الرقمي، التوقيع بالماء، التقارير.

RESUME

L'objectif de ce travail est la réalisation d'une e-Application permettant à une institution de produire des documents sur papier ou numériques qu'il est possible d'authentifier à travers l'internet. Le processus d'authentification de documents papier se fait visuellement par comparaison du document papier et du même document reproduit par le site de l'institution. Le processus d'authentification de document numérique se base sur le tatouage et la signature numérique. Il consiste à soumettre au site de l'institution le document numérique que le site reconnaitrait comme le sien ou non.

Mots clé: Signature numérique, watermarking, reporting.

ABSTRACT

The objective of this job is the realization of an e-application allowing to an institution to produce documents about paper or numerical which it is possible to authenticate across the Internet. The process of authentication of paper documents is visually made by comparison of the paper document and the same document reproduced by the site of the institution. The process of authentication of numerical document is based on the tattoo and numerical signature. It consists in submitting to the site of the institution the numerical document which the site will identify as his or not.

Key words: Numerical Signature, watermarking, reporting.

TABLE DES MATIERES

CHAPITRE II: MODELISATION DU SYSTEME

CHAPITRE III: REALISATION DU SYSTEME

Liste Des Figures

Liste des tableaux

INTRODUCTION

INTRODUCTION

Le développement des technologies de l'information facilite aujourd'hui la communication et l'échange de données, indépendamment des localisations géographiques des serveurs et des acteurs, et impose des changements importants sur l'organisation de travail. A partir de cela, plusieurs réflexions ont été menées au sein des différents organismes, qui visent à diminuer les contraintes de présence physique sur les lieux de travail en offrant les outils et la technologie nécessaires pour permettre l'accomplissement des taches à distance.

Parmi les activités appartenant à ce cadre-là, on trouve les dé-livraisons des analyses médicales, des relevés de notes, et de tant d'autres papiers nécessaires dans la vie courante.

Ces activités se déroulent actuellement dans des endroits précis. Cela oblige la présence physique des personnes concernées sur les lieux de dé-livraisons, ce qui donne naissance à notre problématique.

Notre mémoire qui contient toutes les étapes du travail que nous avons accompli est constitué de quatre chapitres, au début sont exposés la problématique et les objectifs de l'étude. Puis, le chapitre état de l'art qui est présenté en deux parties, la première sur la sécurité web et la seconde sur le Watermarking. Le chapitre suivant est la modélisation du système, de la présentation de la méthode de modélisation jusqu'à l'analyse proprement dite, vient ensuite la phase de conception qui contient l'architecture technique du système, la conception détaillée des objets et la conception de la base de données. Le dernier chapitre présente en détail la réalisation: outils et technologie utilisés, les principaux algorithmes utilisés dans la programmation et nous finirons par une conclusion

1-Problématique :

Les Documents sont omniprésents dans la vie quotidienne de tout un chacun ; pour n'importe quelle procédure, on doit présenter obligatoirement un document tel qu'un acte de naissance, un certificat de scolarité, des relevés de notes, une analyse médicale, une ordonnance médicale...etc. A n'importe quel moment et à n'importe quel endroit on peut en avoir besoin. Or, il existe des situations dans lesquelles des contraintes peuvent nous empêcher de présenter au moment et à l'endroit où on se trouve le document sollicité ou tout simplement d'affirmer que le document que nous présentons est authentique.

Ainsi, nous intervenons par la mise en œuvre de ce projet pour améliorer la qualité de vie des gens et minimiser leurs efforts physique et moral en leur offrant la possibilité de se procurer leurs documents à travers Internet sans le moindre souci. Une fois le document téléchargé, et prêt à être imprimé, on se trouve confronté à un second obstacle: L'intégrité du document sans la présence du cachet humide et la signature des officiers publics ayant le droit d'instrumenter dans le lieu où le document a été rédigé, et avec les solennités requises.

Nous avons franchi le seuil du développement et de la recherche scientifique en entreprenant ce projet dont la problématique se résume en deux points essentiels et négociables tout au long de ce mémoire :

- Le non Disponibilité des documents indépendamment des localisations Géographiques.
- l'authenticité et l'intégrité des Documents produits.

Afin d'atteindre nos objectifs, nous devons répondre aux questions suivantes :

- Comment peut-on mener au mieux la création, la production, la gestion ainsi que le stockage des documents ?
- Comment peut- on garantir l'intégrité et l'authenticité des documents ?

2- Objectifs :

Le système à réaliser n'est pas destiné à une entreprise ou une institution particulière. Il devrait être capable de produire des documents authentifiables pour n'importe quelle institution, établissement ou opérateur économique et social. Ainsi le système sera indépendant du type de document à produire et authentifier. De ce fait, le système devra fournir les facilités nécessaires à la création de modèle de document. Une fois un modèle crée, le système devra permettre la définition des états de sortie de ce modèle. Souvent, pour un modèle, correspond un seul état de sortie. Cet état de sortie sera produit dans un format standard imprimable. Dans notre cas ce sera le format PDF qui sera utilisé. Pour chaque modèle de document, le système devra permettre la saisie d'information relative à ce modèle et produire un document PDF. Le document PDF devra ensuite être transformé pour produire

un document PDF facilement authentifiable que ce soit dans son état de base (fichier PDF) ou dans son état sur papier.

- Le reporting est notre premier réflexe pour la création et la production des documents, pour le stockage et la gestion des documents.
- La cryptographie à clé public ainsi que la sténographie dont nous avons assigné un chapitre nommé Sécurité web dans la partie état de l'art du mémoire expliquant au mieux les concepts que nous avons traité, répondrons au besoin de l'authenticité et de l'intégrité respectivement à travers la signature numérique et le Watermarking.
- les deux solutions proposées nécessiterons une interface web pour l'interaction des utilisateurs avec le Système , d'où le besoin d'une application web.

3- Méthodologie :

La méthodologie utilisée est itérative et progressive (incrémentale). Elle comporte des phases globales suivantes :

- L'analyse des besoins : Au niveau de cette phase il y'aura la compréhension du monde réel qui sera visée par l'application, la compréhension du problème et la détermination des besoins
- La conception du système
- La Réalisation
- Le Test

Pour les phases conception, réalisation et test, la méthodologie incite à commencer par les aspects les plus simples, les réaliser et les tester et passer ensuite à la réalisation d'un autre aspect.

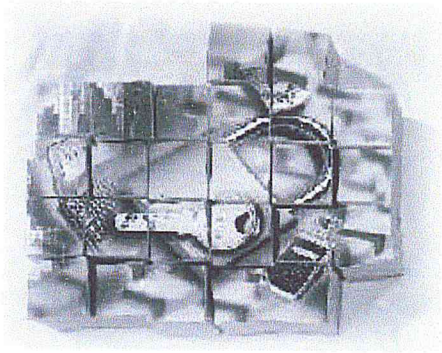
La conception, la réalisation ou le test d'un aspect pourra mettre en cause les aspects précédents. Dans ce contexte un réajustement des étapes précédentes est nécessaire. Il faut revenir en arrière pour refaire la conception/réalisation.

Ce processus pratique permettra d'avoir à chaque étape une version fonctionnelle d'une partie de logiciel.

Chapitre I

ETAT DE L'ART

I- LA SECURITE WEB



« Le seule système infaillible est celui qui est éteint et débranché, enfermé dans un coffre en titane, enterré dans un block en béton, entouré d'un nuage de gaz neuroplégique et de garde armé très bien payés. Même ainsi je ne parierais pas ma vie dessus »

Jim Conallen.

I-1 INTRODUCTION:

Depuis quelques années, la révolution des moyens de communication, particulièrement l'Internet a mené à une large liberté en circulation d'informations et une haute disponibilité de nombreuses ressources. Ceci a fait naître de nouveaux problèmes dans la sécurité informatique donc de nouveaux besoins.

La sécurité web, une branche de la sécurité informatique, c'est l'ensemble des techniques qui visent la préservation de la confidentialité et l'intégrité des données échangées via Internet. Nous présenterons dans ce chapitre les concepts de base de la cryptographie. Les méthodes de cryptage symétrique, asymétrique, l'hachage et la signature numérique.

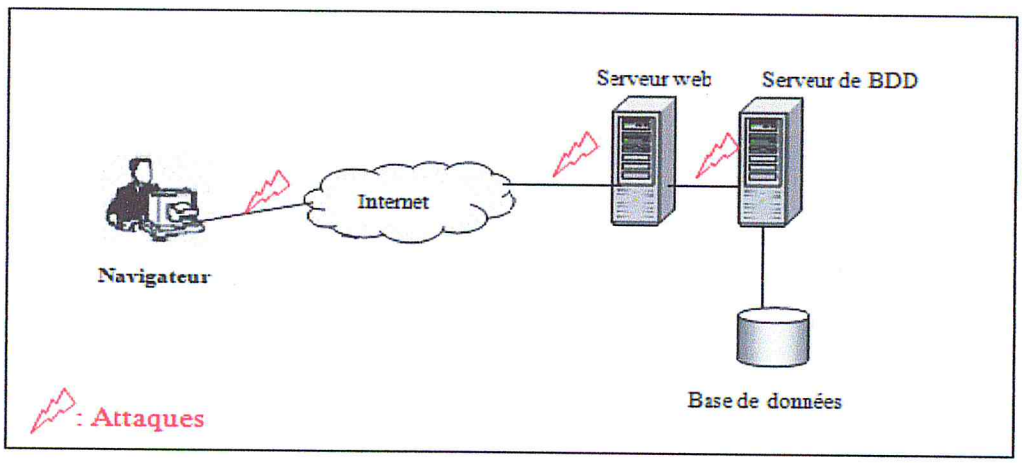


Figure 1 : Architecture d'une application web et flux menacés

I-2 CRYPTOGRAPHIE :

« Le chiffrement est l'action de transformation d'un texte "*lisible*" en un texte "*illisible*", via une clé de chiffrement. Seule une personne disposant de la clé de déchiffrement (qui peut être la même que celle de chiffrement) sera en mesure de déchiffrer le texte. » [1]

La cryptographie est à la base de la sécurité informatique, sa connaissance est nécessaire pour comprendre les technologies de sécurité utilisées pour sécuriser les réseaux.

I-2-1 Définition :

La cryptographie est la science d'écriture et de lecture des messages codés [2]. Elle permet de transmettre des données de manière confidentielle.

Afin de protéger un message, on lui applique une transformation qui le rend incompréhensible, c'est ce qu'on appelle chiffrement ou cryptage, qui à partir d'un texte en claire donne un texte chiffré ou cryptogramme. Inversement, le déchiffrement ou décryptage est l'action qui permet de reconstituer le texte en claire à partir du texte chiffré [2].

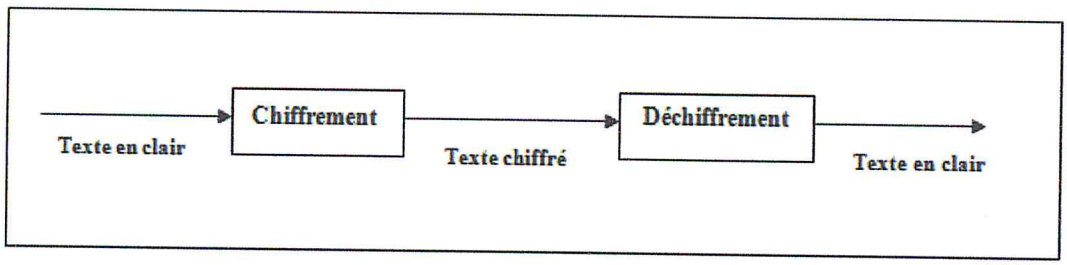


Figure 2 : Principe générale de la cryptographie.

Dans la cryptographie moderne, les transformations en questions sont des fonctions mathématiques, appelées algorithmes cryptographiques qui dépendent d'un paramètre appelé clé.

D'autres termes se réfèrent à ce domaine tel que la cryptanalyse qui est l'étude des procédés cryptographique dans le but de pouvoir décrypter des textes chiffrés et la cryptologie qui englobe les deux domaines : cryptographie et cryptanalyse [2].

I-2-2 Historique

La cryptologie a connu une évolution vertigineuse avec le développement des systèmes informatiques : passant d'une ère artisanale et confidentielle (combines et ruses déjà en 2000 av J-C) à des systèmes de très hautes technologies, nécessitant une importante puissance de calcul, essentiellement régis par l'arithmétique et des algorithmes complexes la classant ainsi parmi les sciences fondamentales. La cryptologie regroupe deux domaines : la **cryptographie** et la **cryptanalyse**.

La cryptographie du grec ***kruptos*** (caché) et ***graphein*** (écrire), qui est l'art d'écrire les messages, code des données dans le but de les rendre inexploitables par toute personne pouvant les intercepter hormis le destinataire légitime.

En parallèle au développement de la cryptographie, la cryptanalyse qui convoite à décrypter le message intercepté a progressé de façon faramineuse : les cryptanalystes s'activent au déchiffrement des systèmes les plus complexes.

Dans cette partie nous allons décrire les besoins cryptographiques, et les méthodes utiliser pour les assurées. Cette partie sera organisée comme suit :

I-2-3 Besoins cryptographiques

Transmettre des données de manière confidentielle, tel était le but de la cryptographie traditionnelle. Aujourd'hui la confidentialité ne suffit plus. Des services de sécurité plus élaborés sont offerts. En plus de la confidentialité, l'authentification, l'intégrité et la non répudiation sont les garanties de la cryptographie moderne **[3]**.

- **Authentification**

  Elle consiste simplement à vérifier l'identité de l'utilisateur ou de l'entité qui veut accéder à un système informatique.

- **Confidentialité**

  La confidentialité est le maintien du secret des informations… (Le petit Robert[1]).

  La confidentialité peut être vue comme « la protection des données contre une divulgation non autorisée »,

- **Intégrité**

  permet de certifier que les données, les traitements ou les services n'ont pas été modifiés, altérés ou détruits tant de façon intentionnelle, elle est garantie grâce aux signatures numériques.

- **Non répudiation**

  Garantir la non répudiation c'est garantir qu'un correspondant ne puisse nier qu'un message lui a été envoyé. elle est assurée grâce aux signatures numériques

Pour garantir ces besoins, on utilise des méthodes basées sur des algorithmes cryptographiques que nous présenterons dans le paragraphe qui suit.

[1] ***Le Petit Robert*** est un dictionnaire de langue française, publié par les dictionnaires Le Robert.

I-2-4 Les méthodes de cryptographie :

On distingue deux types de chiffrement [1]

I-2-4-1 La cryptographie à clé secrète :

Le chiffrement symétrique, dit aussi a clé secrète est la forme la plus ancienne de cryptage. Elle consiste utiliser une valeur courte (la clé) pour rendre un message inintelligible aux tierces parties. Elle est dite symétrique car la clé de chiffrement et celle de déchiffrement.

a) Principe:

Le principe consiste a utiliser la même clé et le même algorithme de chiffrement pour chiffrer et déchiffrer un message. Donc les communicants doivent s'entendre a l' avance sur l'algorithme de chiffrement à employer et également sur la clé secrète à utiliser avec l'algorithme[1].

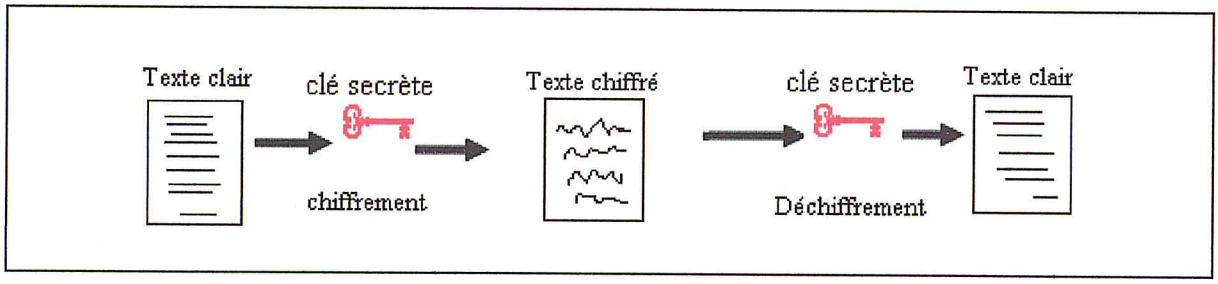


Figure 3: Chiffrement à clé secrète [4]

Il existe généralement deux types d'algorithmes symétrique, par :

b) Chiffrement symétrique par substitution :

Le code de César est le plus vieil algorithme de chiffrement symétrique par substitution connu. Il consiste à remplacer chaque lettre du message d'origine par une lettre de l 'alphabet situé n positions plus loin (par une simple translation). N constitue la clé secrète.

Exemple :

- La clé est 3 :
- Texte clair : SECURITE
- Texte chiffré: VHFXULWH

c) Chiffrement symétrique par transposition :

Le principe de codage par transposition est de modifier selon une loi prédéfinie l'ordre des caractères.

La méthode de pliage constitue un exemple simple de chiffrement par transposition. Elle consiste à écrire le message d'origine dans une matrice (écriture en ligne) comportant autant de colonnes que la clé secrète. La clé secrète est constituée de numéros de colonnes. Le cryptogramme est obtenu en lisant cette matrice en colonnes selon l'ordre défini par la clé.

Exemple :

Le message d'origine: COMMERCE ELECTRONIQUE

La clé de codage : 4312

Le message crypté : MECNE MCEOU CEETI ORLRQ

1	2	3	4
C	O	M	M
E	R	C	E
E	L	E	C
T	R	O	N
I	Q	U	E

d) Les avantages et inconvénient de la cryptographie à clé secrète :

Le chiffrement symétrique est intéressant car il est simple à mettre en œuvre et requiert un faible temps de calcul. Mais il présente un inconvénient majeur : la difficulté de protéger le secret d'une clé. En effet, au moment ou les deux parties échangent leur clé secrète, ils ne peuvent pas s'assurer que celle ci n'est pas interceptée par un tiers. Cette solution s'avère donc, seule, insuffisante.

Ce problème a incité à la réflexion à une autre méthode cryptographique plus sûre et moins complexe en matière de gestion. La cryptographie à clé publique est apparue.

I-2-4-2 La cryptographie à clé publique

Le chiffrement asymétrique, dit aussi a clé publique découle de découvertes théoriques relativement récentes dans le domaine mathématique. Il repose sur l'existence de fonctions mathématiques difficiles à inverser **[4].**

a) Principe:

Chaque communicant utilisent deux clés, l'une est connue par tous (clé publique), l'autre n'est connue que par lui-même (clé privée). Le message crypté avec l'une ne peut être décrypté qu'avec l'autre

Les deux clés sont reliées mathématiquement entre elles de telle sorte qu'il est impossible de retrouver la clé privée en connaissant la clé publique.

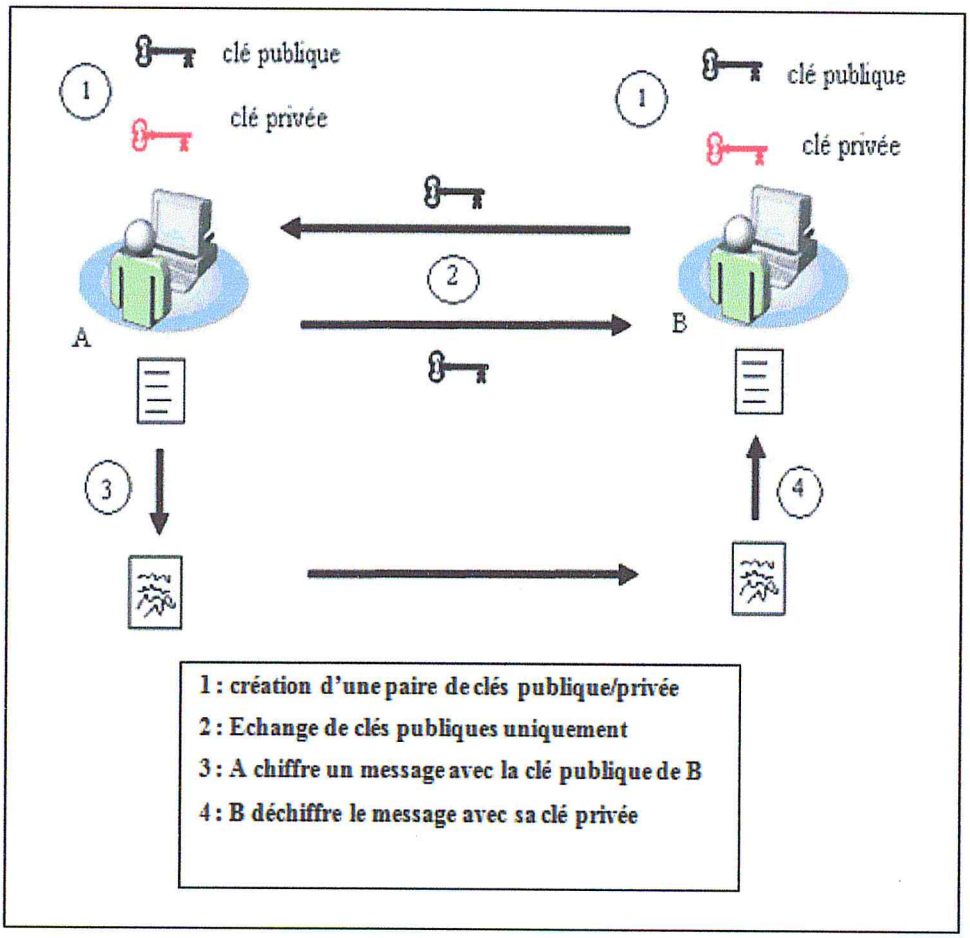


Figure 4 : Chiffrement à clé publique **[1]**

Parmi les algorithme de cryptage asymétrique il y a l'algorithme DSA et RSA ,nous allons utiliser le RSA , vue qu'il est très utiliser en public, simple et fiable , nous allons expliquer le principe ainsi les points fort et faible du RSA.

b) L'algorithme RSA (Rivest Shamir Adleman)

Inventé à la fin des années 1970 (de ses concepteurs Rivest, Shamir et Adleman), il utilise des clés très longues (jusqu'à 1024 bits) et offre toutes les garanties cryptographiques (confidentialité, intégrité, authentification et non répudiation). La sécurité apportée par le système RSA se fonde sur la difficulté à factoriser le produit de deux grands nombres premiers. Le RSA reste sécurisé face aux attaques, mais on doit employer des nombres premiers de plus en plus grands, car la puissance des microprocesseurs croît sans cesse. Cet algorithme est très largement utilisé, par exemple dans les navigateurs pour les sites sécurisés et pour chiffrer les emails. Il est dans le domaine public.

L'algorithme :

Pour encrypter un message, on fait: $\mathbf{c = m^e \bmod n}$

Pour décrypter: $\mathbf{m = c^d \bmod n}$

m = message en clair
c = message encrypté
(e,n) constitue la clé publique
(d,n) constitue la clé privée
n est le produit de 2 nombres premiers
mod est l'opération de modulo (reste de la division entière)

Créer une paire de clés :

Pour créer une paire de clés, il ne faut pas choisir n'importe comment **e**, **d** et **n**. Voici comment procéder:

1. Prendre deux nombres premiers **p** et **q** (de taille à peu près égale). Calculer **n = pq**.
2. Prendre un nombre **e** qui n'a aucun facteur en commun avec **(p-1)(q-1)**.
3. Calculer **d** tel que **e*d mod (p-1)(q-1) = 1**

Le couple **(e, n)** constitue la clé publique. **(d ,n)** est la clé privée.

Pour crypter on fait : $c = m^e \bmod n$

Pour décrypter on fait : $m = c^d \bmod n$

c) Problème de la cryptographie à clé publique

Les méthodes de chiffrement à clé publique sont jusqu'à 1000 fois plus lentes que les méthodes de chiffrements à la clé secrète.

I-2-4-3 Hachage

a) Principe de hachage

Le hachage (appelé aussi résumé de message ou empreinte numérique) est une représentation plus bref d'un message.

La fonction de hachage reçoit en entrée un message de longueur aléatoire et produit un message de longueur fixe .

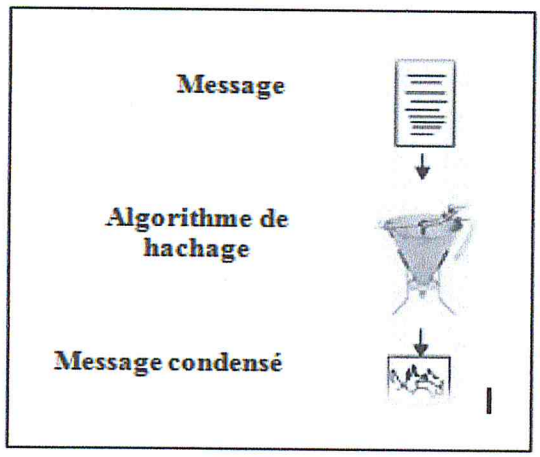


Figure 5 : Principe du hachage.

Un algorithme de hachage doit être :

- Cohérent : le même message en entrée doit toujours produire le même résultat.
- Unique : deux messages différents ne doivent jamais produire le même condensé.
- Non réversible : il doit être extrêmement difficile, voir impossible d'obtenir le message d'origine à partir de son condensé.

Le hachage est généralement utilisé pour fournir une empreinte d'un message ou fichier pour assurer l'intégrité et l'authentification du message .

b) Types de hachage :

Les algorithmes de hachage peuvent être sans clé ou avec, on distingue trois types de hachage :

- **Hachage sans clé : MIC** (Message Integrity Code)

Dans ce type, le message est soumis à un algorithme de hachage sans clé (sans paramètre en entrée). Les algorithmes les plus utilisés sont MD5 et SHA1.

La plupart des signatures numériques à clé publique emploient des résumés de message sans clé.

- **Hachage avec clé : MAC** (Message Authentification Code)

Dans ce type, le message est soumis à une fonction de hachage qui reçoit comme paramètre d'entrée une clé.

- **HMAC :**(keyed-Hash Message Authentication Code)

Combine les deux méthodes précédentes. Le message est concaténé à une clé secrète. Le tout est soumis à une fonction de hachage sans clé.

I-2-4-4 La signature numérique :

a) Définition

Une signature numérique est un **condensé de message crypté** qui joint un document. Elle combine l'utilisation du cryptage à clé publique et d'une fonction de hachage.

Ce ci permet de s'assurer que :

- l'expéditeur est bien l'émetteur de message (identification - authentification).
- le message reçu est bien conforme à celui transmis par l'expéditeur (intégrité).

b) Processus de création d'une signature

- Créer une paire de clés publique/ privée
- Soumettre le message à une fonction de hachage
- Crypter le résultat de hachage avec la clé privée

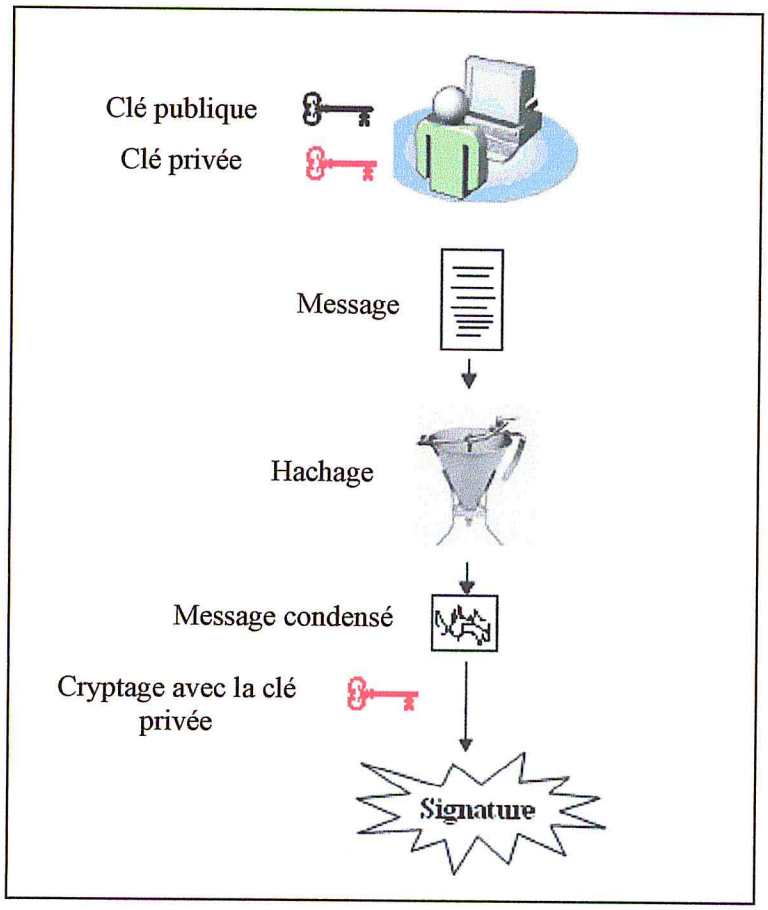


Figure 6: processus de création d'une signature

Remarque :

L'émetteur envoie au destinataire son message en lui concaténant la signature obtenue et la clé publique (ou un certificat numérique contenant la clé publique). La clé publique servira pour la vérification de la signature.

c) Processus de vérification d'une signature :

Le destinataire reçoit un message signé avec la clé privée. La clé publique correspondante à cette dernière est à sa disposition. Pour vérifier cette signature et donc vérifier que le message provient du bon émetteur, il procède ainsi :

1. Séparer la signature du message
2. Décrypter la signature avec la clé publique de l'émetteur (on obtient ainsi le résumé du message originale).
3. Soumettre le message à la même fonction de hachage (les communiquant s'entendent sur l'algorithme de hachage avant de commencer l'échange de message)
4. Comparer le résumé obtenu dans 3 avec celui obtenu dans 2.

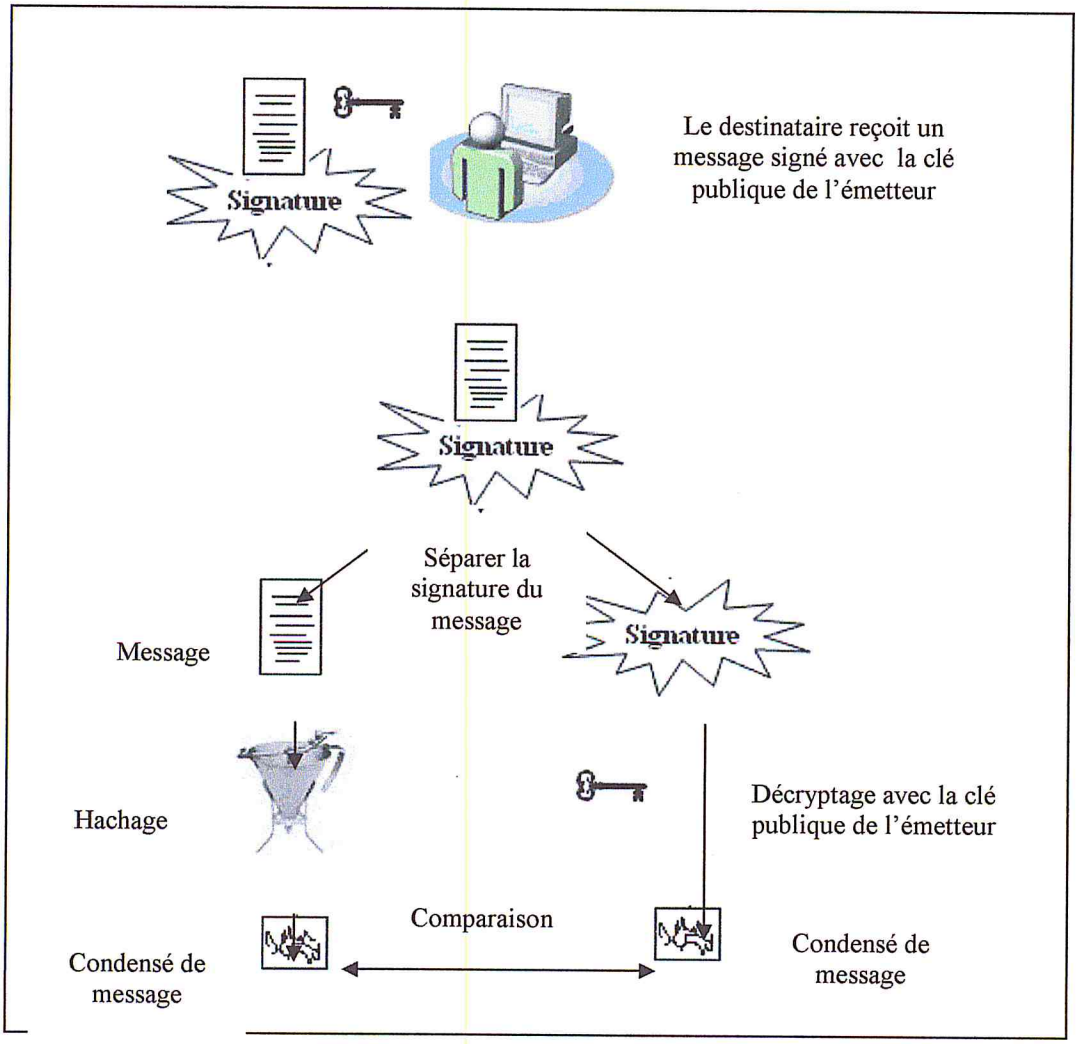


Figure 7: Vérification d'une signature

3. Conclusion

Les méthodes et outils pour la sécurité web deviennent de plus en plus nécessaires pour faire face aux logiciels espions.

Nous avons abordé dans ce chapitre les différentes méthodes cryptographiques, notamment la cryptographie à clé publique.

II- TATOUAGE

II-1 Principe et applications:

Selon Cox et col. [5], le principe du tatouage est de modifier imperceptiblement un contenu pourinsérer une information concernant le document. Le contenu original est appelé le contenu hôte. L'objectif du tatouage est donc de permettre l'accès au contenu hôte, qui pourra être authentifiéou protégé grâce au message inséré.

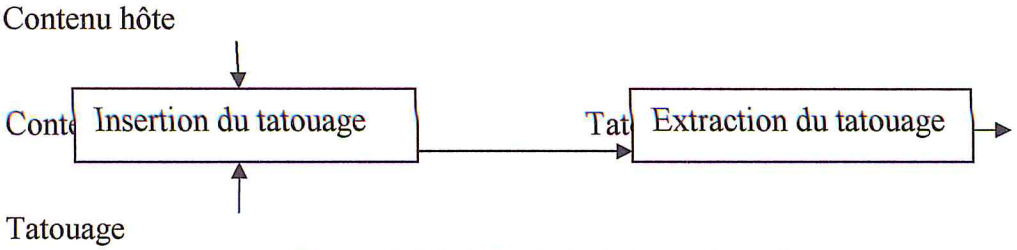


Figure8: Modèlegénéral du système de tatouage .

Le tatouage peut être appliqué dans différents domaines [7], [5] comme la surveillance dediffusion, la protection de copyright, l'authentification de contenu ...

- **Surveillance de diffusion** :

Dans le domaine de la radiodiffusion ou de la télédiffusion, l'annonceur paie une certaine sommepour la diffusion de sa publicité pendant un certain temps. Cela peut être trente minutes, uneheure voire deux heures par jour. Et l'annonceur doit vérifier si le temps payé est bien respecté.
Il existe deux systèmes de surveillance : passif et actif. La surveillance passive essaie dereconnaître le contenu diffusé tandis que la surveillance active se base sur l'information diffuséeavec le contenu. Le tatouage est une des solutions pour la surveillance active : il est implémentédans nombreuses entreprises.

- **Protection de copyright**

En général, un tatouage peut être inséré dans un contenu hôte pour démontrer le droit dupropriétaire du contenu. Pour prouver la propriété du document, ce tatouage devra être robusteet sûr, par exemple si deux personnes insèrent successivement leurs tatouages dans une mêmeimage, on devra pouvoir trouver le vrai propriétaire.

- **Authentification de contenu**

Pour authentifier le contenu, on peut utiliser la signature digitale et le tatouage. L'inconvénient dela signature digitale est d'être stockée dans les métadonnées. Elle est donc facilement perduedans l'usage normal. Etant inséré directement dans le contenu, le tatouage peut éviter cetinconvénient. Le message d'authentification doit être éliminé quand le contenu est un peumodifié. Autrement dit, le tatouage doit être fragile. Le tatouage peut être appliqué sur plusieurs types de données audiovisuelles: signaux audio, signaux vidéo, images naturelles, images ou objets synthétiques de 2 ou 3 dimensions. Ce dernier type inclut les maillages 3D, support dans le cadre de ce stage.

II-2Propriété du tatouage

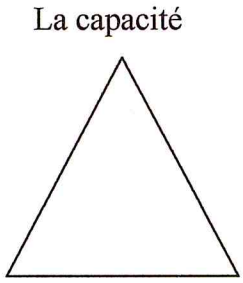
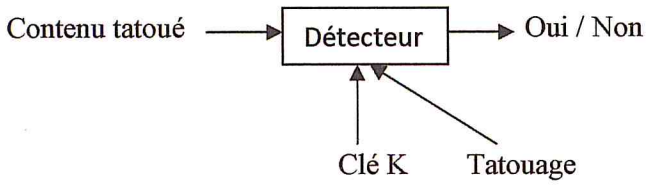


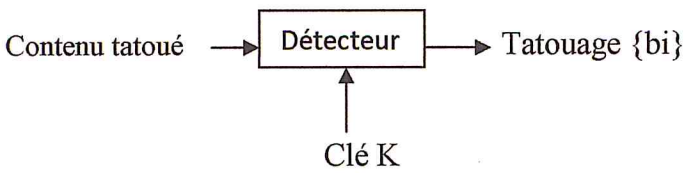
Figure9: Compromis entre trois propriétés importantes du tatouage

❖ **Capacité du tatouage :**

La capacité [6] est le nombre de bits qui pourront être insérés dans un contenu hôte. Il s'agitd'une propriété importante du tatouage.
On parle de tatouage « zéro bit », lorsque la capacité est équivalente à 0. Dans ce cas, on parle detatouage détectable car on va seulement détecter la présence d'une marque dans le document. Dans les autres cas, on parle de tatouage lisible, c'est-à-dire que la marque peut être extraite.



(a) Tatouage détectable : le décodeur vérifie si le contenu est tatoué



(b) Tatouage lisible : le décodeur extrait le tatouage multi-bits {bi}, i = 1..N

Figure10: Illustration des deux systèmes de décodeur du tatouage.

❖ **Insertions multiples :**

Dans quelques applications, on a besoin d'insérer plusieurs tatouages dans un même signal hôte. Un exemple de ces applications est le système de protection des droits d'auteur et du contrôle decopie. Le contenu doit contenir deux tatouages : l'un avec l'identité de l'auteur et l'autreindiquant le nom du client. D'une part, il faut que l'insertion de plusieurs tatouages n'influencepas la qualité du signal hôte. D'autre part, la présence des tatouages suivants ne doit pas affecterl'extraction ou la détection des tatouages précédemment insérés.

❖ **Robustesse :**

La robustesse [6] est l'aptitude du message caché à survivre à des manipulations nonmalveillantes sur le contenu tatoué. Les manipulations non malveillantes sont des traitementsordinaires qui ne visent pas à enlever le tatouage. Elles incluent de nombreuses opérationscomme l'ajout de bruit, la rotation, la compression …

Chapitre II

MODELISATION

I-ANALYSE DES BESOINS

Dans ce chapitre nous allons formuler les objectifs et les besoins de notre système à l'aide des diagrammesde cas d'utilisation d'UML.

I-1-Cas d'utilisation globale :

Dans ce cas, le diagramme de cas d'utilisation utilise deux types d'utilisateurs :

- Trois acteurs principaux représentés par l'administrateur du système, gestionnaire d'une institution, et agent de saisie d'une institution.
- Deux acteurs externe représenté utilisateur et l'anonyme.

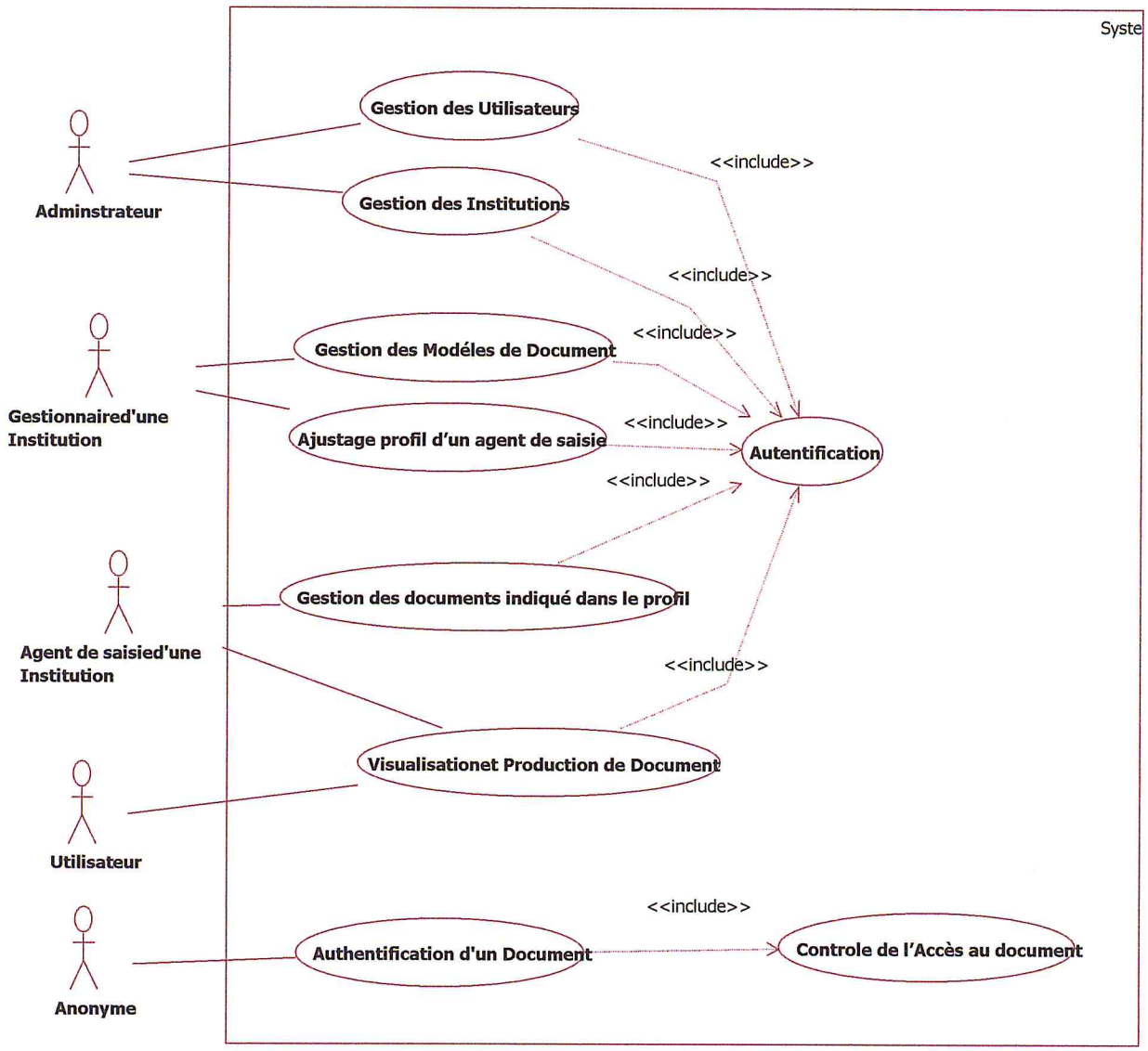


Figure 12 : Cas d'utilisation général

- L'administrateur à la possibilité de gérer les Institutionset les utilisateurs.

- Le gestionnaire d'une Institution à la possibilité de gérer les Modèles de Document et le rôle des agents de saisie d'une institution (indication du type de document que l'agent peut saisir).
- L'agent des saisie d'une institutionà la possibilité de saisir et modifier les documents et produire les documents à fournir aux concernés (ou bénéficiaire).
- Un utilisateur est un acteur qui possède un compte dans le système. Son profil ne lui permet que la visualisation et la production des documents pour lesquels il a le droit d'accès. L'administrateur, le gestionnaire, ou l'agent de saisiepeuvent indiquer, chacun selon ses privilèges, les documents auxquels peut accéder un utilisateur particulier.
- Une personne anonyme (accès public), peut accéder aux aspects publics du système. La fonctionnalité d'authentification de document est une fonctionnalité publique. L'accès a cette fonctionnalité nécessite la spécification du numéro de document à authentifier et un numéro secret associé au numéro du document.

I-2- Diagrammes de Cas d'utilisation détaillés :

Pour illustrer notre démarche, nous allons détailler les cas d'utilisation précédents.

I-2-1 Cas d'utilisations de la Gestion des Utilisateurs :

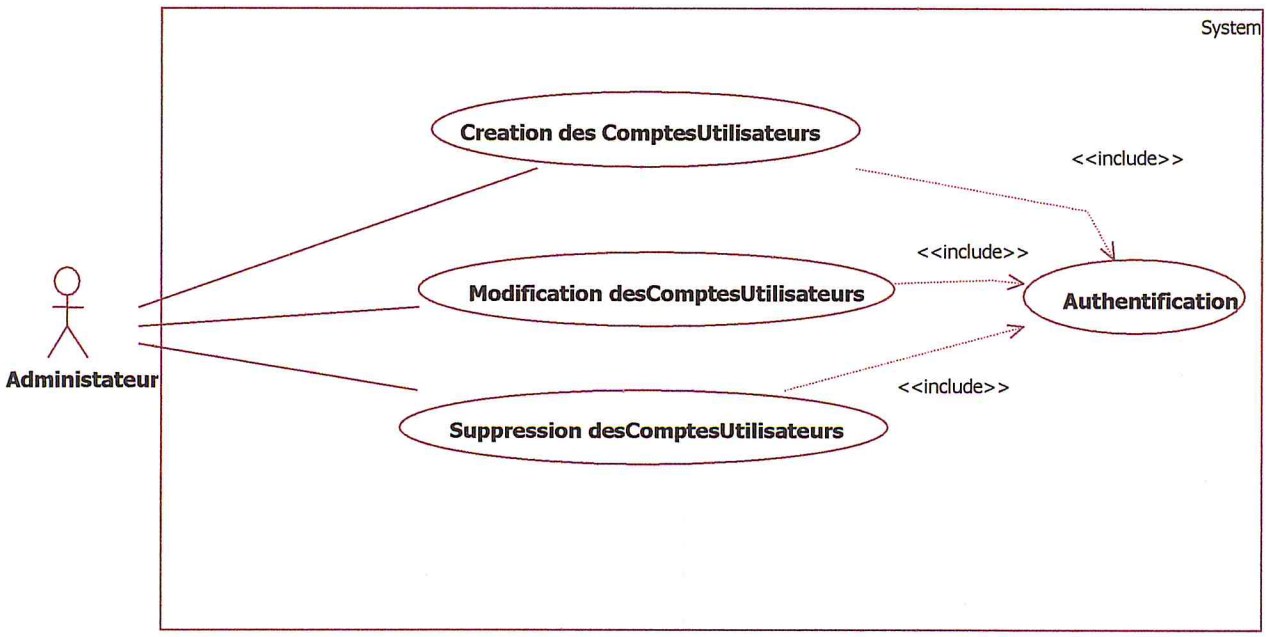


Figure13 : Cas d'utilisations de la Gestion des Utilisateurs

L'administrateur pourra à tout moment faire :

- La Mise à jour des comptes utilisateurs (Créer, Modifier, Supprimer).

I-2-2 Cas d'utilisation de la Gestion des Institutions :

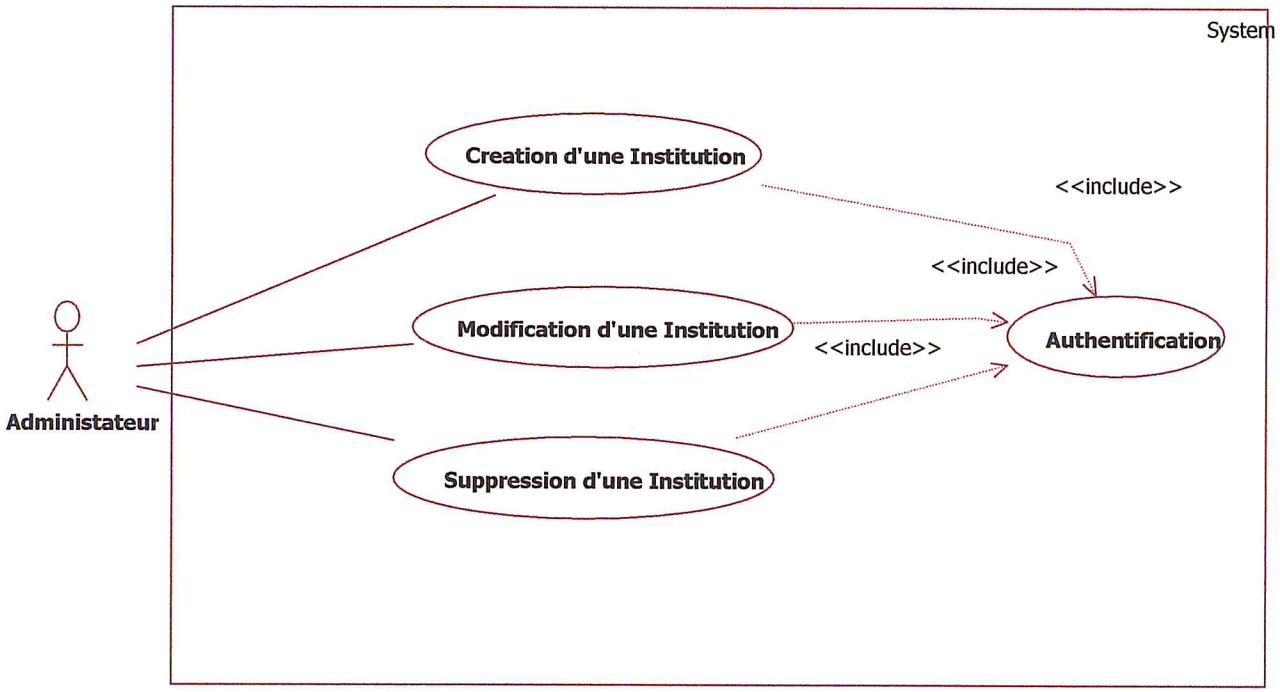


Figure14 : Cas d'utilisation de la Gestion des Institutions

L'administrateur pourra à tout moment faire :

- La Mise à jour d'une institution (Créer, Modifier, Supprimer).

I-2-3 Cas d'utilisation de la Gestion des Modèles de Document:

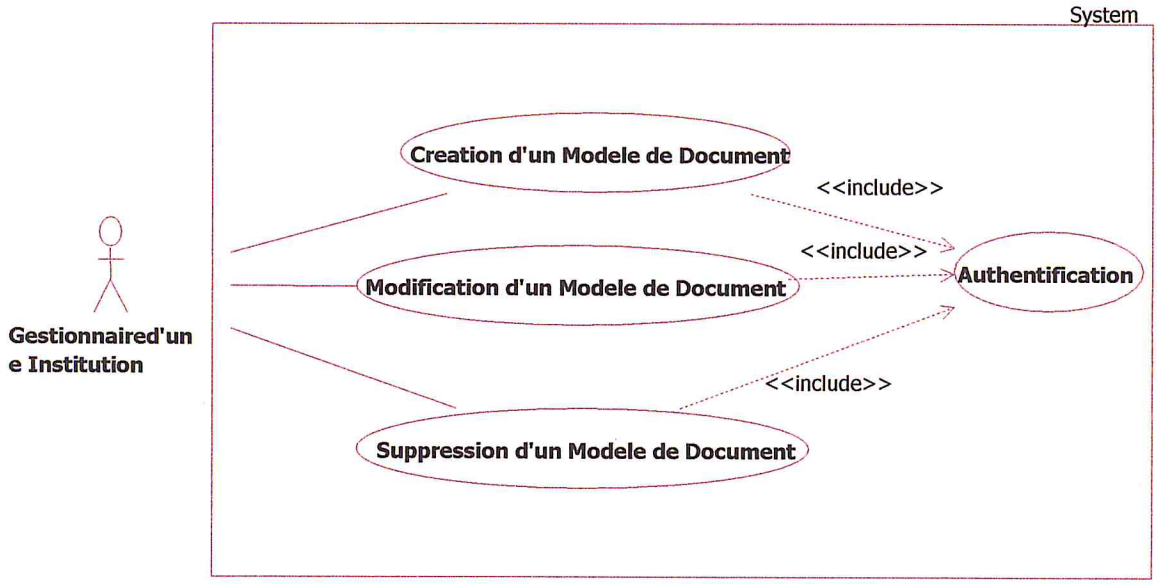


Figure 15: Cas d'utilisation de la Gestion des Modèles de Document

Le gestionnaire d'une institution à la possibilité de faire :

- La Mise à jour d'unmodèle de document (créer, Modifier, Supprimer).

I-2-4 Cas d'utilisation de la Gestion des Documents indiqué dans le profil:

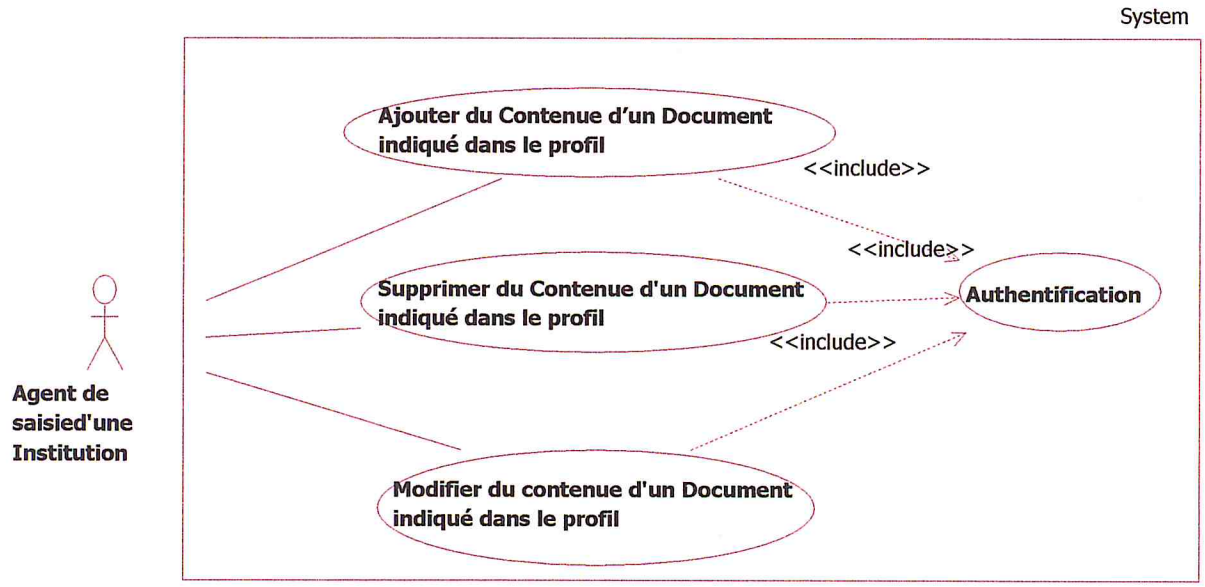


Figure 16: Cas d'utilisation de la Gestion des Documents indiqué dans le profil

Pour la gestion des documents indiqué dans le profil, l'agent de saisied'une institution fait :

- La Mise à jour du contenu d'un document (ajouter, supprimer, modifier)

I-2-5 Cas d'utilisation de l'Authentification d'un Documents:

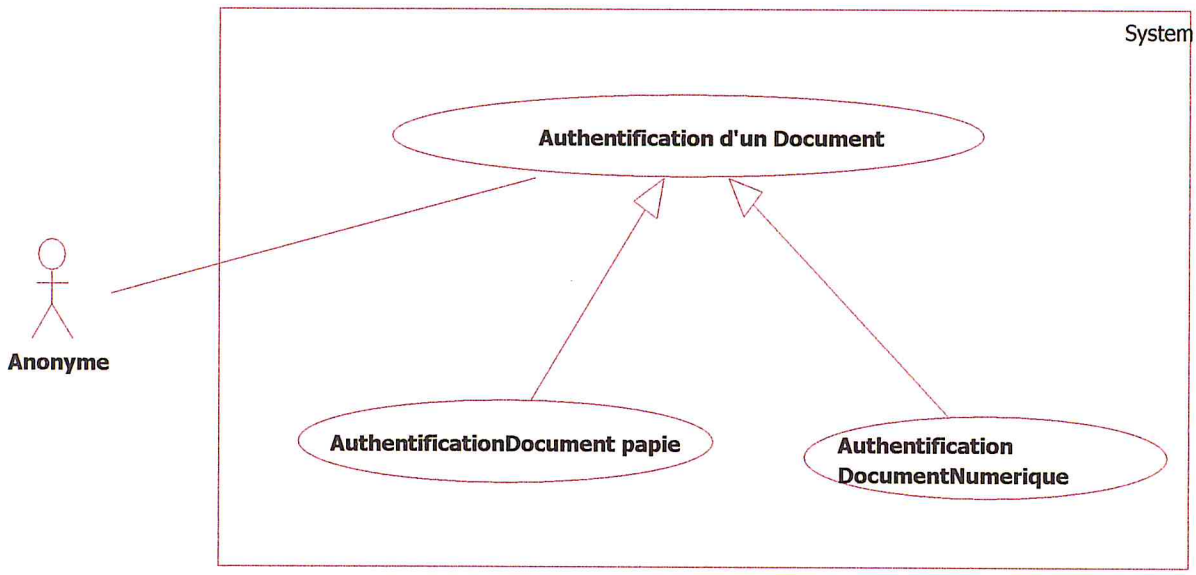


Figure 17: Cas d'utilisation de l'Authentification d'un Document

L'authentification des documents est divisée en deux types d'authentification :

- L'authentification des documents papier
- L'authentification des documents numériques

II- CONCEPTION DU SYSTEME

II-1Introduction :

Modéliser un système avant sa réalisation permet de mieux comprendre son fonctionnement , c'est également un bon moyen de maîtriser la complexité d'un système et d'assurer sa cohérence.

Un diagramme donne à l'utilisateur un moyen de visualiser et de manipuler des éléments de modélisation. UML définit des diagrammes structurels et comportementaux pour représenter respectivement des vues statiques et dynamiques d'un système.

Nous avons utilisé cinq diagrammes :

- Les cas d'utilisation : pour définir les besoin de notre système (voir partie analyse des besoins).
- Le diagramme de classe : pour exprimer de manière générale la structure du système.
- Le diagramme de séquence : pour représenter les interactions entre les objets.
- Le diagramme d'activités : pour représenter le comportement d'une opération en termes d'actions.

II-2 Diagramme de Classe :

Les diagrammes de classe expriment de manière générale la structure statique d'un système, en termes de classe et de relation entre les classes**[8]**.

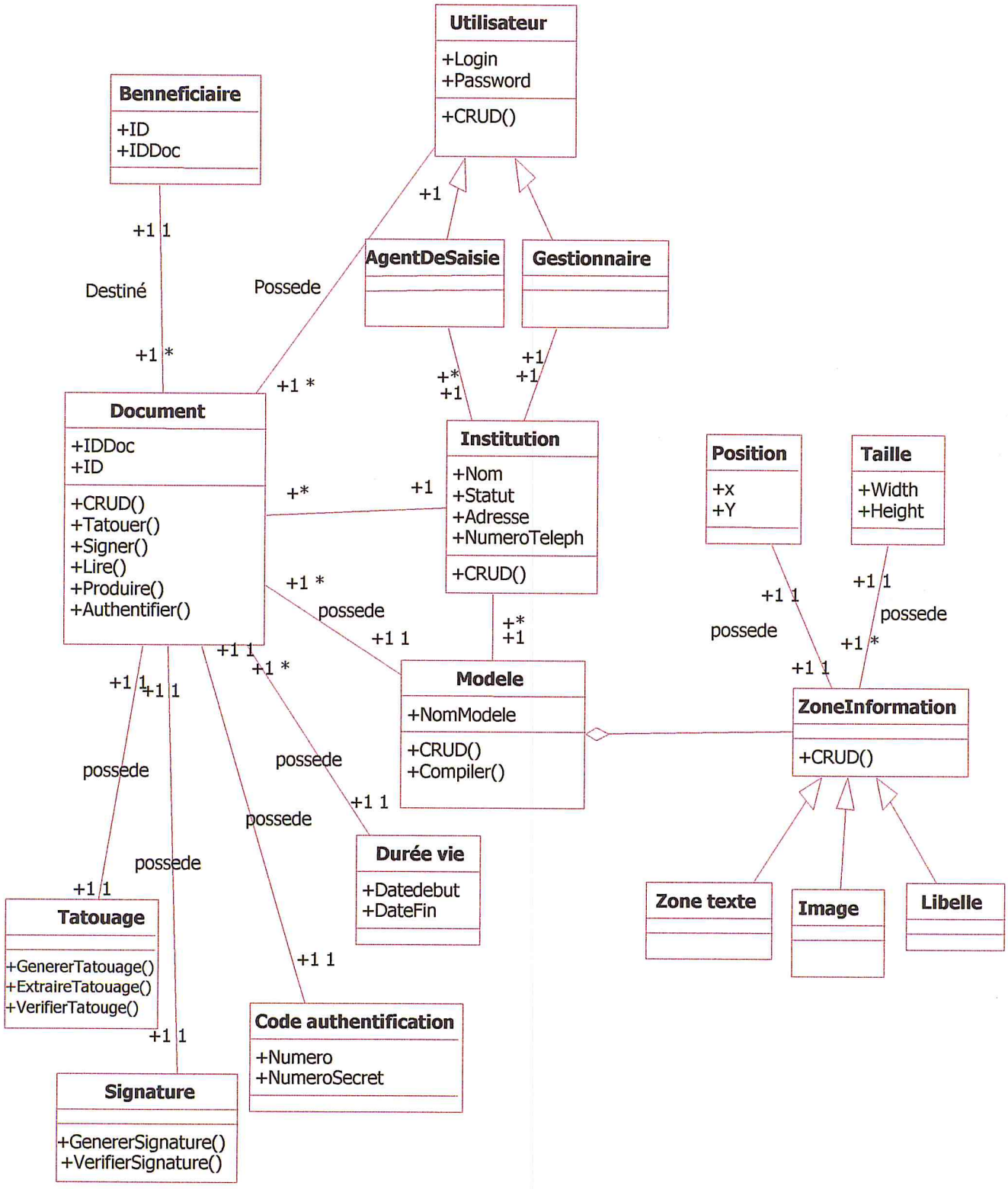


Figure 18: Diagramme de Classe

II -2-1 Description des classes :

Classe	Description
Utilisateur	Contient les informations sur les utilisateurs
Beneficiaire	Contient les informations sur toutes personnes bénéficie d'un document sur le Système
Institution	Contient des informations d'une Institution
ZoneInformation	Contient les informations sur les zones d'un modèle de document
Position	Contient les informations sur la position d'une zone d'Information
Taille	Contient les informations sur la taille d'une zone d'Information
Modèle	Contient les Information sur le modèle de document
Document	Contient les Informations du document
DureeDeVie	Contient les informations sur la durée de vie d'un document
CodeAuthentification	Contient les informations sur les Codes d'authentification de chaque exemplaire du document
Tatouage	Contient les informations sur le tatouage appliqué sur chaque document
Signature	Contient les informations sur la signature numérique appliquée sur chaque document

Tableau 1 : Description des classes

II -2-2 Description des attributs et des méthodes des classes :

1-Classe Institution :

Attributs		
Attribut	Type	Signification
Nom	Varchar(50)	Le nom de l'institution
Adresse	Varchar(50)	Adresse de l'institution
N°Téléphone	Varchar(50)	Numérotéléphone de l'institution
Statut Sociale	Varchar(50)	Statut Sociale de l'institution

Méthodes	
Méthode	Signification
CRUD()	Creat ,Read,Update,Delete signifiant ajouter,lire,modifier et supprimer les informations de l'insitution

2-Classe Utilisateur :

Attributs		
Attribut	Type	Signification
Login	Varchar(50)	Le nom d'utilisateur
Password	Varchar(50)	Le mot de passe d'un utilisateur
Disponibilité	Varchar(50)	La disponibilité du compte

Méthodes	
Méthodes	Signification
CRUD()	Creat ,Read,Update,Delete signifiant ajouter,lire,modifier et supprimer les informations d'un utilisateur

3-Classe Bénéficiaire :

Attributs		
Attribut	Type	Signification
ID	Varchar(50)	L'identifiant d'un bénéficiaire

4-Classe Document :

Attributs		
Attribut	Type	Signification
ID	Varchar(50)	L'identifiant d'un document
Méthodes		
Méthodes	Signification	
CRUD()	Creat ,Read,Update,Delete signifiant ajouter,lire,modifier et supprimer les informations d'un document	
Tatouer()	Appliquer un tatouage numérique au document	
Signer()	Appliquer une signature numérique sur le Document	
Produire()	Production d'une version PDF du Document	
Authentifier()	Vérification de l'authenticité d'un Document	

5-Classe Modèle :

Attributs		
Attribut	Type	signification
NomModéle	Varchar(50)	Nom du modèle de document
Méthodes		
Méthode		Signification
CRUD()		Creat ,Read,Update,Delete signifiant ajouter,lire,modifier et supprimer les informations d'un Modèle

6-Classe Dureedevie :

Attributs		
Attribut	Type	signification
Datedebut	Date	Date de production d'un document
Datefin	Date	Date d'expiration d'un document
Méthodes		
Methode		Signification
CRUD()		Creat ,Read,Update,Delete signifiant ajouter, lire, modifier et supprimer les informations de la durée de vie d'un Document

7-Classe CodeAutentification :

Attributs		
Attribut	Type	Signification
Numéro	Varchar(50)	Numéro d'identification associé a chaque exemplaire de document produit par système

10-Classe Signature :

Méthodes	
Methode	Signification
GenererSignature	Generation d'une signature numerique
VerifierSigneture	Verification d'une Signature numerique

11-Classe Position :

Attributs		
Attribut	Type	Signification
X	Integer(50)	Indique la position d'une zone sur l'axe des x
Y	Integer(50)	Indique la position d'une zone sur l'axe des y

12-Classe Taille :

Attributs		
Attribut	Type	Signification
width	Integer(50)	Indique la largeur d'une zone du modèle
height	Integer(50)	Indique la hauteur d'une zone du modèle

II-3- Diagramme d'Activité :

Le diagramme d'activité visualise un graphe d'activité qui modélise le comportement interne d'une méthode (la réalisation d'une opération), d'un cas d'utilisation ou plus généralement d'un processus impliquant l'utilisation d'un ou plusieurs classificateurs**[8]**.

II-3-1 Diagramme d'Activité pour l'authentification d'un documentpapier :

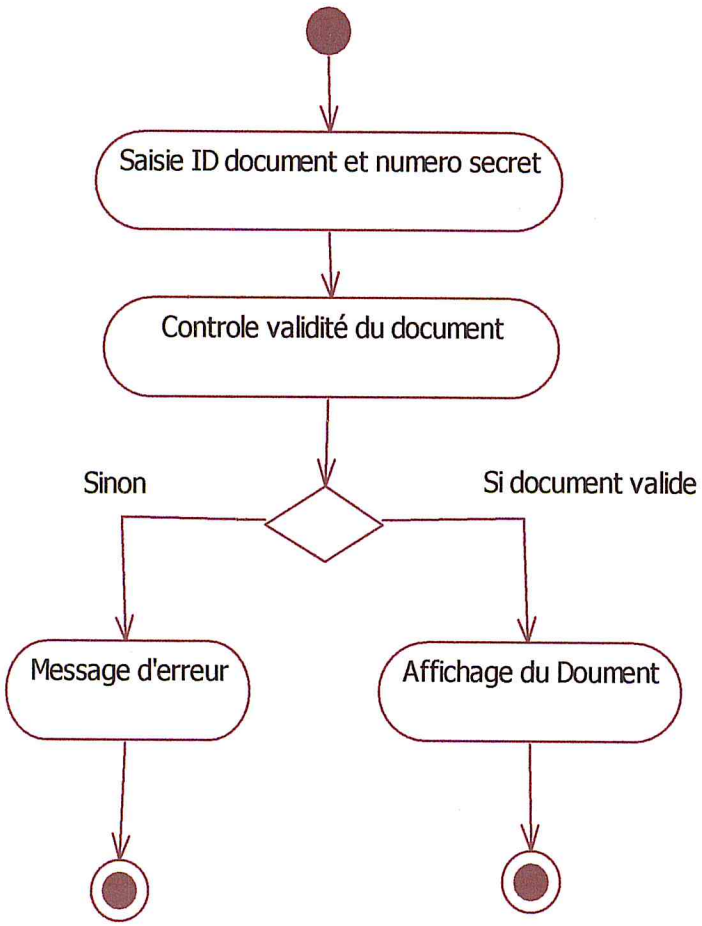


Figure 19: Diagramme d'Activité pour l'authentification d'un document

II-3-2 Diagramme d'Activité pour l'authentification d'un document Numérique :

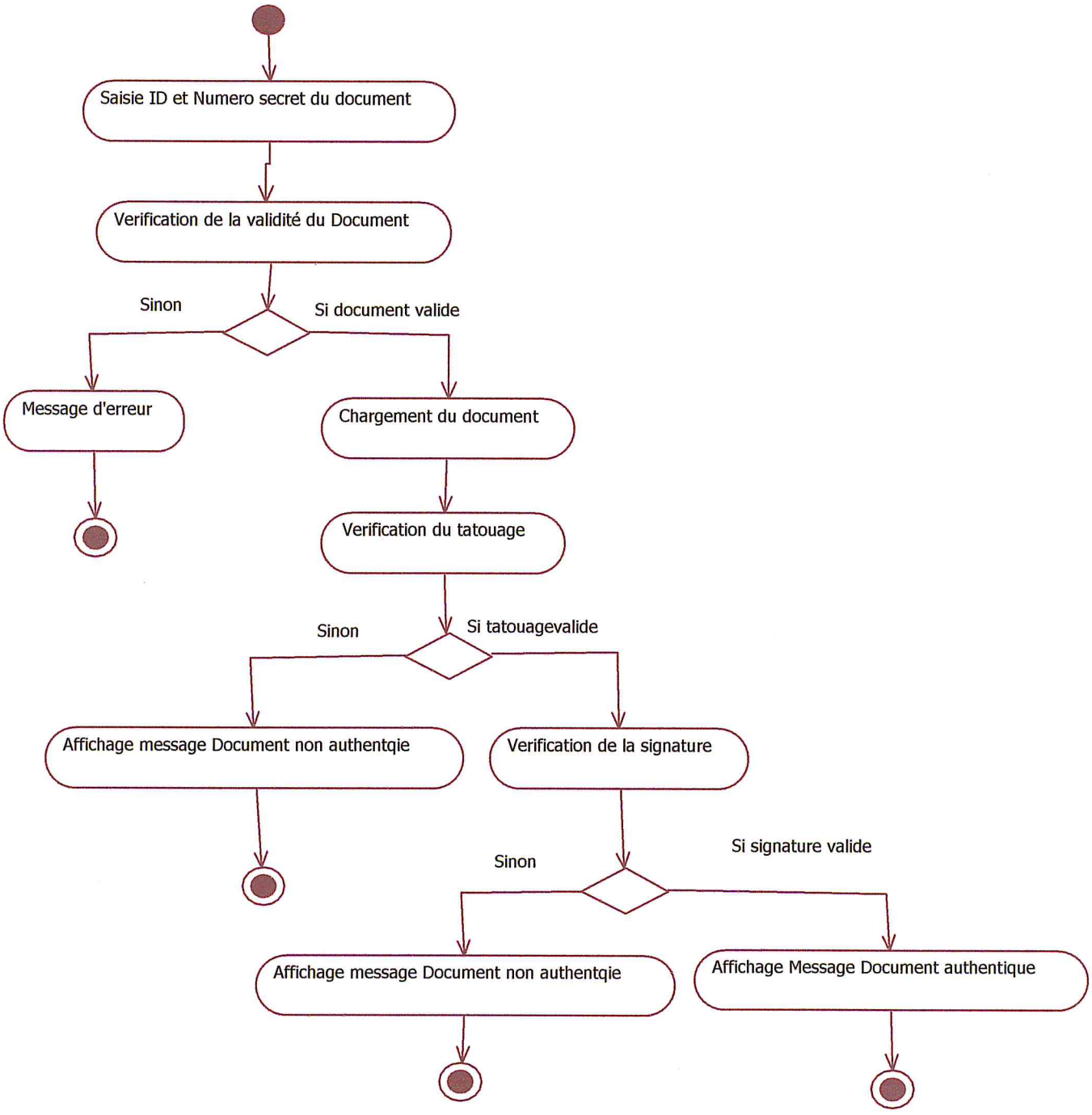


Figure 20 : Diagramme d'authentification de document numérique

II-3-3 Diagramme d'Activité pour la gestion des modèles de document :

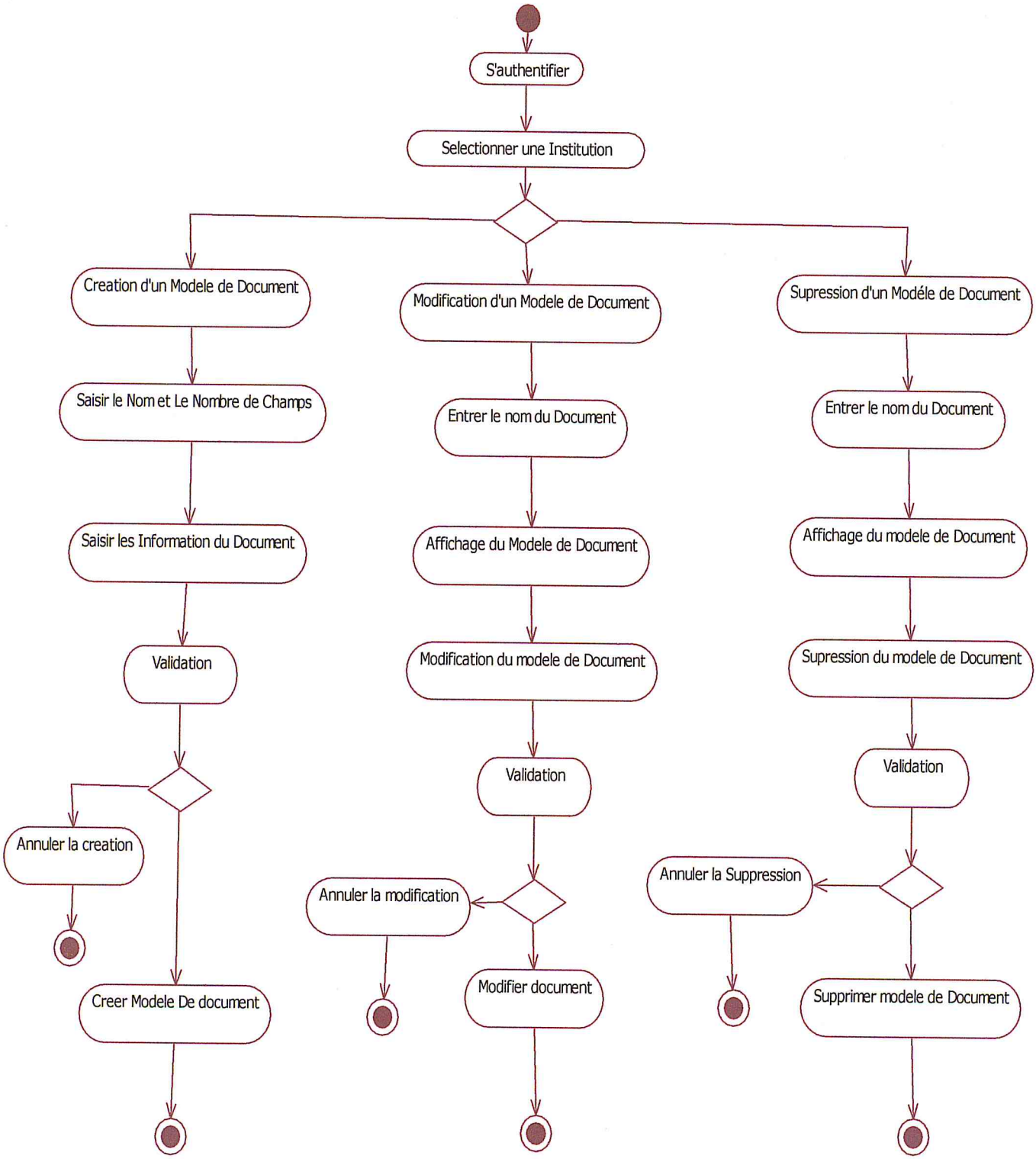


Figure 21: Diagramme d'Activité pour la gestion des modèles de document

II-3-4 Diagramme d'Activité pour la production de document :

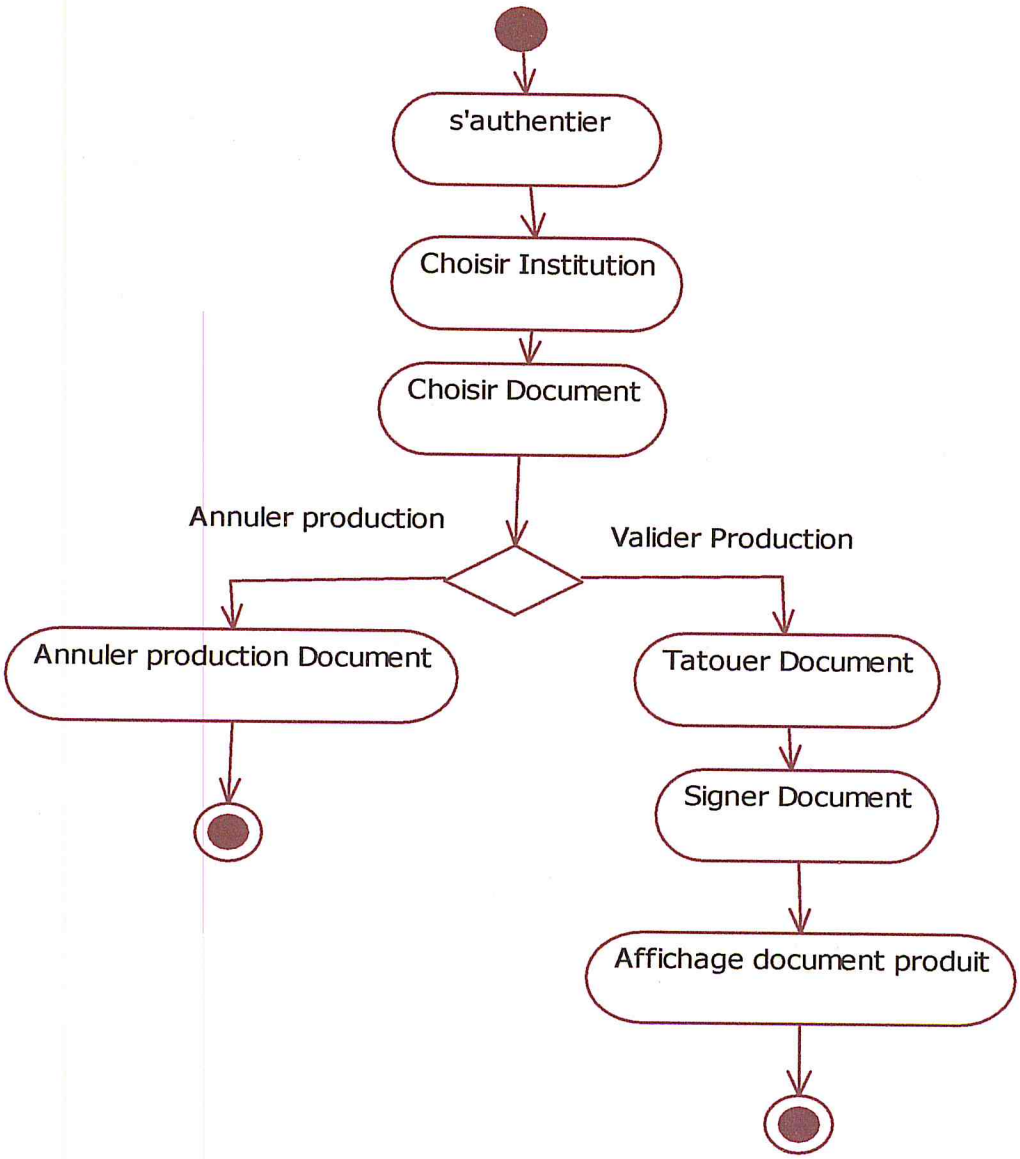


Figure 22: Diagramme d'Activité pour la production de document

II-4-Diagramme de Séquence :

Les diagrammes de séquence montrent des interactions entre les objets.
Toutefois, la présentation se concentre sur la séquence des interactions selon un point de vue temporel[8].

II-4-1 Diagramme de Séquence pour la création des modèles de document :

1- Saisie Nombre de champs et le nom du document
2- Creation d'un formulaire de saisie
3- Affichage du Formulaire
4- Saisie des Infromations du Document
5- Creation du document
6- Création du Model de Document
7- Compilation du modèle
8- Reporting des données
9- Production d'un Exemplaire du document
10- AffichageExemplaire du Document

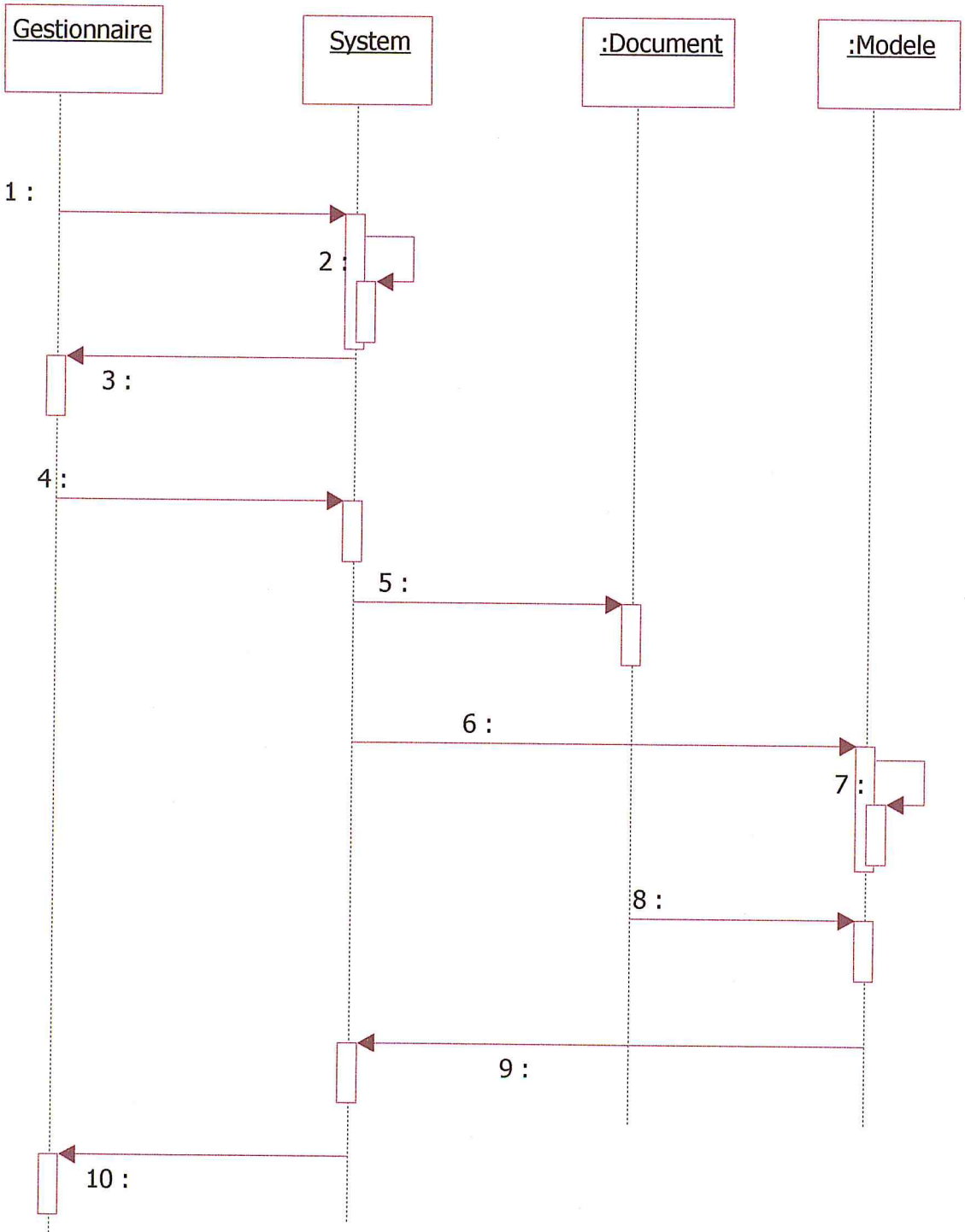


Figure 23: Diagramme de Séquence pour la création des modèles de documents

II-4- 2 Diagramme de Séquence pour la saisie du contenu des documents :

1- Choisir document de saisie
2- Création d'un formulaire de saisie
3- Affichage du formulaire
4- Saisie de la clé de recherche
5- Création d'un nouveau formulaire de saisie
6- Affichage du formulaire
7- Saisie des restes des informations du Document

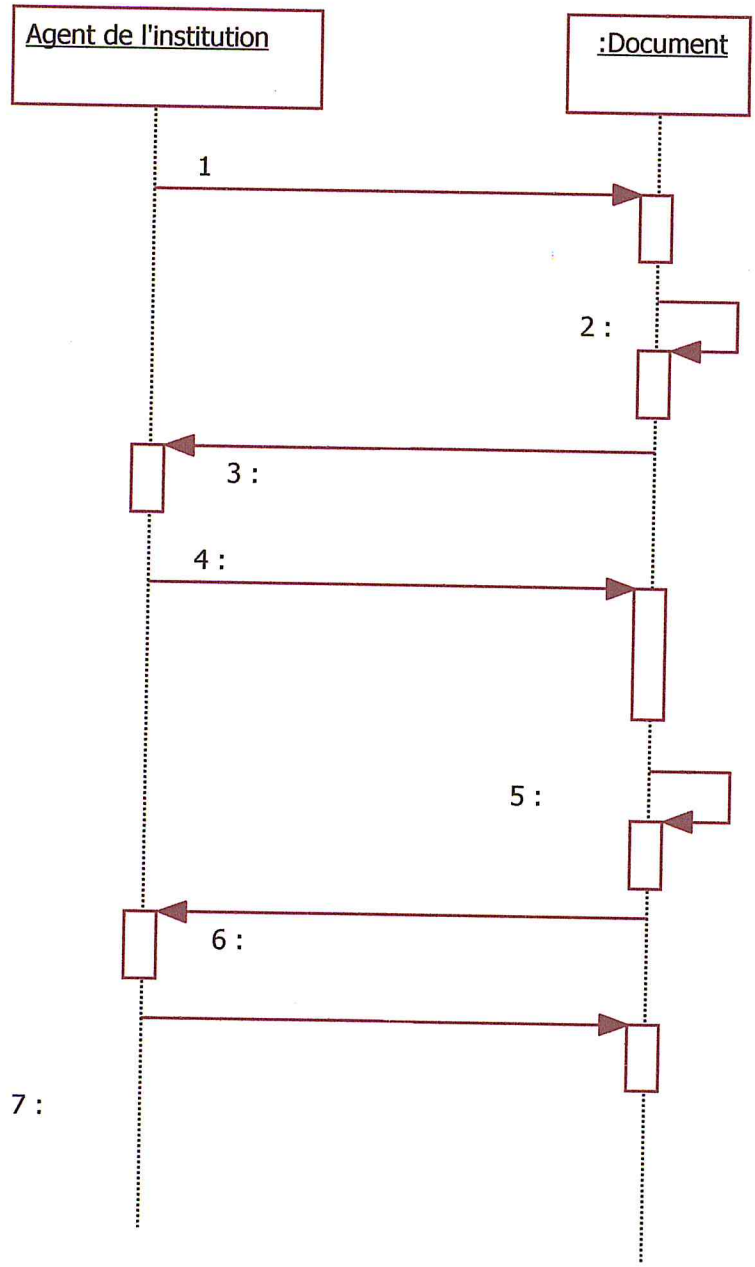


Figure 24: Diagramme de Séquence pour le saisi du contenu des documents

II-4- 3 Diagramme de séquence pour la production de document :

1- Choisir document
2- Compilation du model de document
3- Extraction selective des Informations
4- Production de version de document Non Tatoué Non Signé
5- Extraction du contenue de la version Non Tatoué Non signé
6- Generation d'empreinte à partir du contenue extrait
7- Ajout de l'empreinte dans le modéle
8- Enregistrement du modeletatoue
9- Compilation du modeletatoué
10- Extraction selective des Informations
11- Production de Document tatoué
12- Signature du document tatoué
13- Affichage du PDF tatoué signer

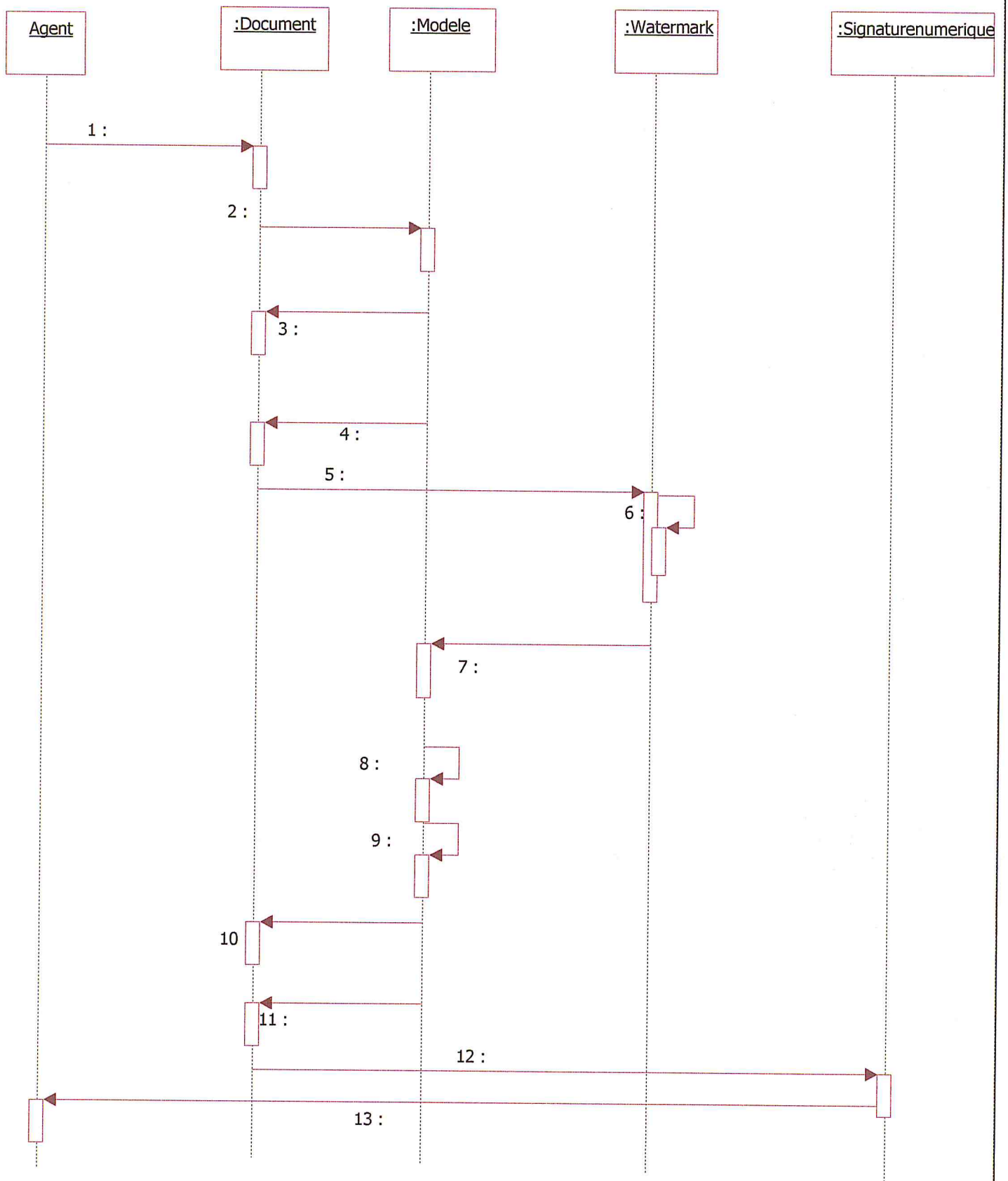


Figure 25 : Diagramme de séquence pour la production de document

II-4- 4 Diagramme de séquence pour l'authentification des documents numérique:

1- Choisir Institution
2- Selection des modeles de document de l'institution
3- Affichage des modele de document
4- Choisir document
5- Affichageformulaired'authentification de document
6- Saisie du ID et du IDs du document à authentifier
7- Document non existant
8- Verification de la validité de l'exemplaire
9- Document non valide
10- Affichage d'un formulaire de chargement
11- Extraction du tatouage
12- Charger le document
13- Extraction du text
14- Generation d'une empreinte du text
15- Verification de la corespondance entre le tatouage extrait et le l'empreinte produit
16- Document non authentique
17- Verification de la signature
18- Document non authentique
19- Document authentique a 100%

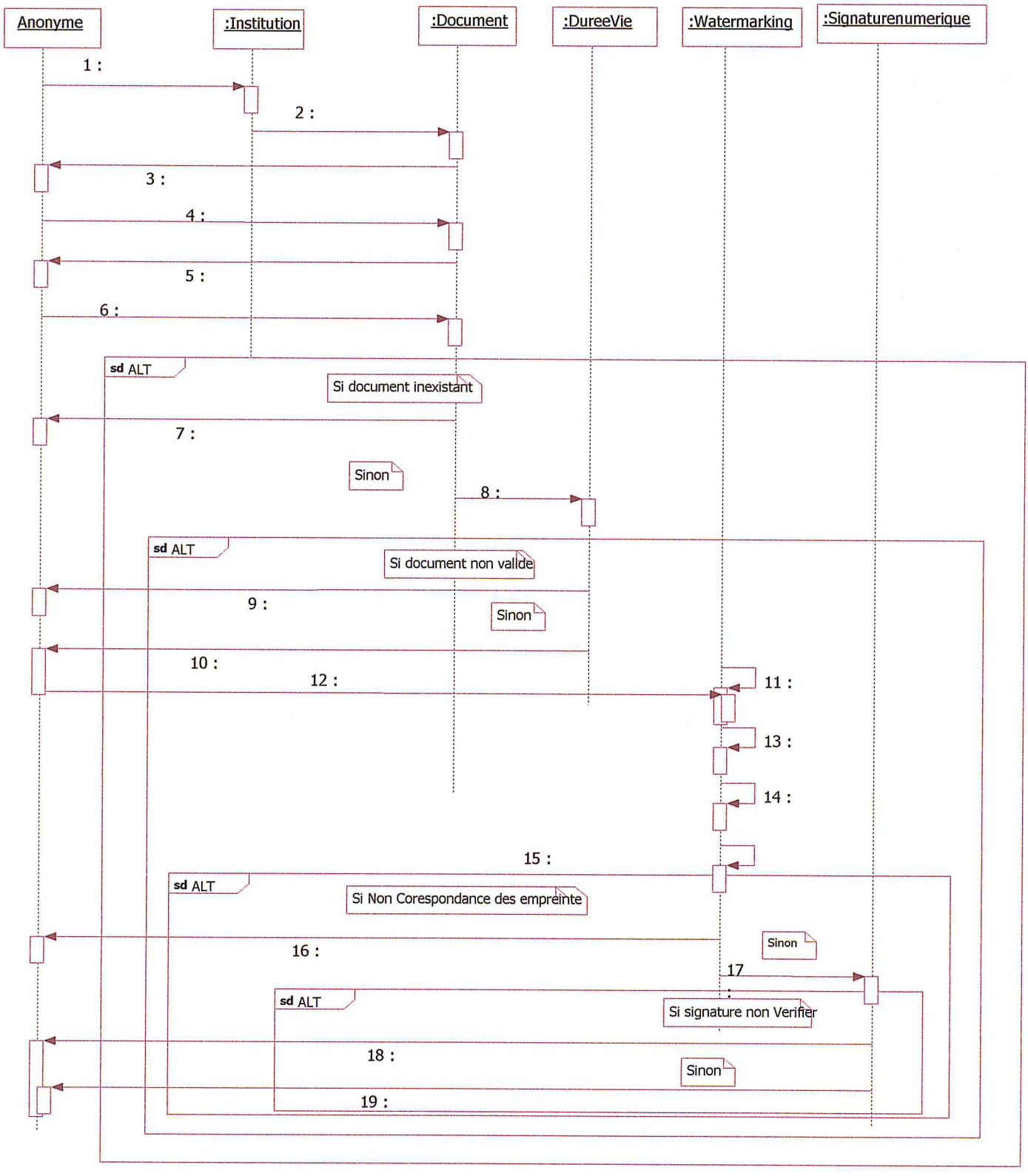


Figure 26: Diagramme de séquence pour l'authentification des documents numerique

II-4-5 Diagramme de séquence pour l'authentification Des documents papiers :

1- Choixd'institution
2- Sélectiondes modèle de doc de l'institution
3- Affichage des modèles de documents de l'Institution
4- Choisir document
5- Affichage d'un formulariesd'authentification de documents
6- Saisi du ID et IDs du document à authentifier
7- vérification de l'existence du document
8- Document n'existe pas
9- Vérification de la validité du document
10- Document non valide
11- Compilation du Modèle
12- Reportingdes information du document
13- Production du document
14- Affichage document
15- Verification visuelle.

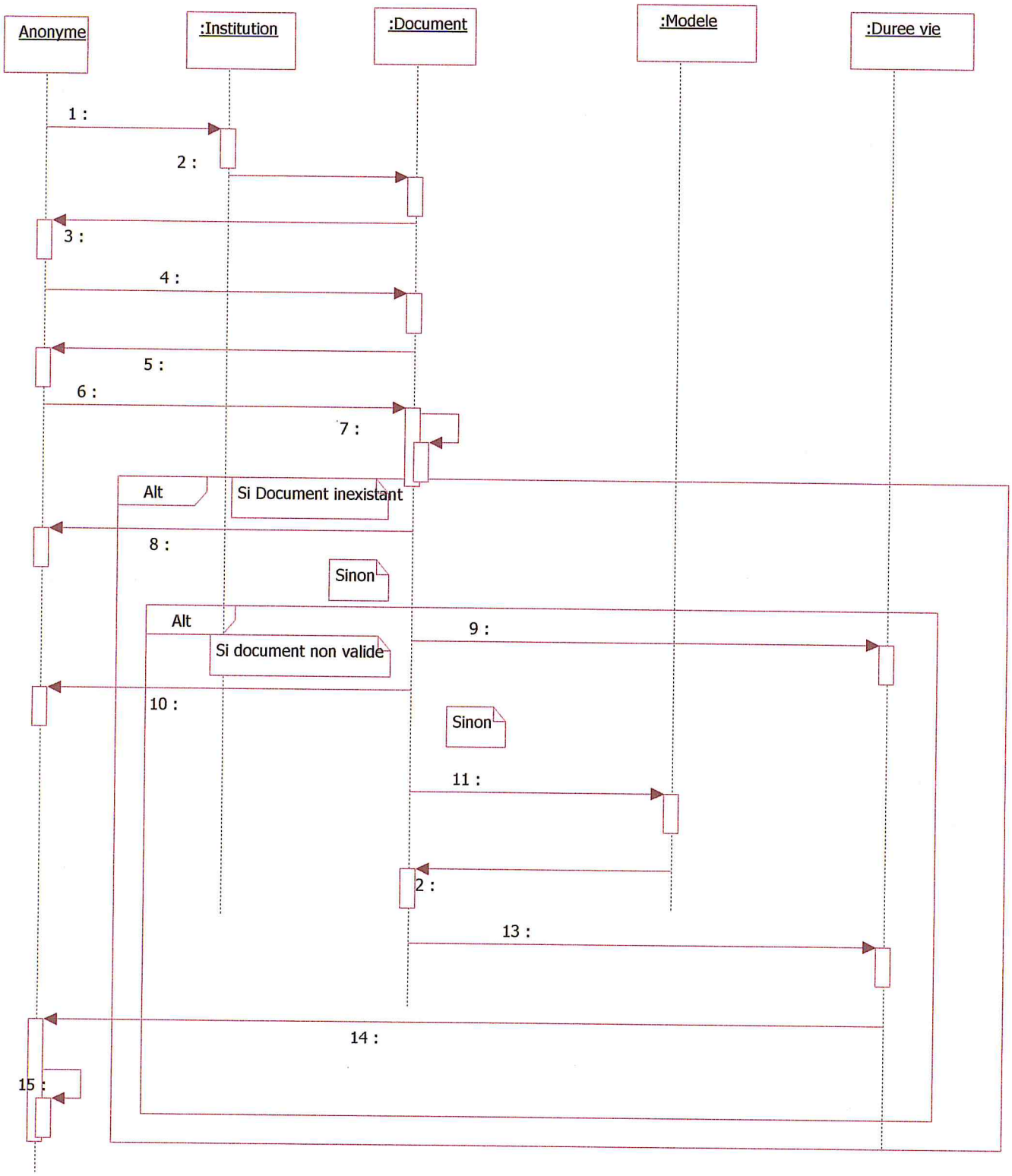


Figure 27 : Diagramme de séquence pour l'authentification des Documents papier

Chapitre III

REALISATION

I- Introduction :

Dans cette partie nous allons mettre en œuvre notre système tel qu'il est décrit dans la partie conception. Pour cela, nous justifions nos choix en matière d'environnement de développement , d'architecture technique, et de fonctionnement de notre application . Tout sera illustré et développé par la suite .

II-Environnement de développement :

Nous présentons dans cette section les outils que nous avons utilisés pour réaliser notre travail (serveur web, SGBD, langages de programmation) tout en justifiant nos choix.

Les logiciels que nous avons utilisés sont :

- Apache-Tomcat-6.0.32
- MySQL version 5.0.18
- Eclipse J2EE

II-1 Le serveur web Apache Tomcat :

Les serveurs web les plus populaires aujourd'hui sont : Apache, Microsoft IIS, Zeus et Sun One, etc. Nous avons choisit Apache pour les avantages suivants :

- Apache est gratuit.
- La configuration d'Apache s'effectue en modifiant ses fichiers de configuration au sein desquels des directives permettent de définir son comportement. Cette méthode de configuration lui procure une souplesse permettant à l'administrateur du serveur un contrôle sur les fonctionnalités et la sécurité offerte par Apache.
- Apache implémente SSL grâce au module mod-ssl et la bibliothèque Openssl (uneboite à outils cryptographiques implémentant le protocole SSL) que nous avons utilisé pour les manipulations cryptographiques nécessaires.
- Tomcat a été écrit en langage Java. Il peut donc s'exécuter via la machine virtuelle Java sur n'importe quel système d'exploitation la supportant

Toutes ces caractéristiques nous ont permit l'utilisation facile de Openssl et configurer facilement notre serveur pour sécuriser notre application.

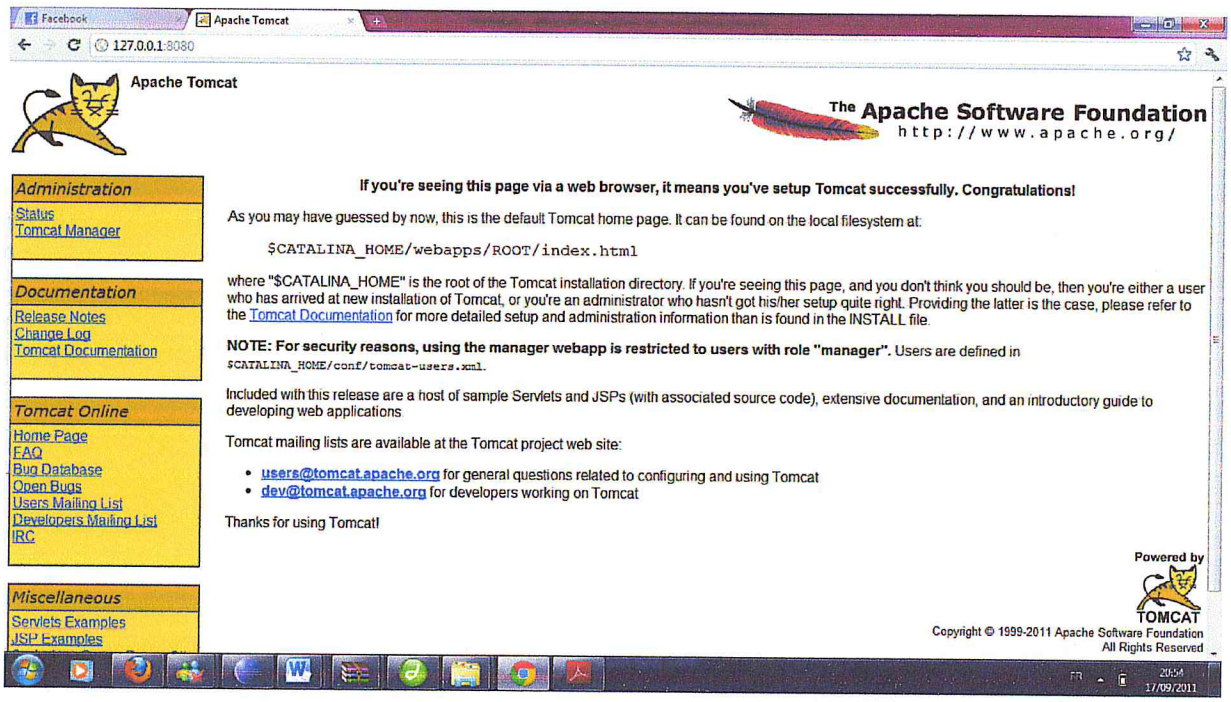


Figure 28: Interface Apache Tomcat

II-2 MYSQL :

Les SGBD libres et gratuits sont nombreux. MYSQL, mSQL, Postgres sont des exemples. Si nous avons choisit MYSQL, c'est plus pour des raisons de performances et fonctionnalités offertes. Nous citerons dans la suite ses principaux avantages :

- MYSQL est beaucoup moins complexes à installer et à administrer que d'autres systèmes.
- MYSQL permet des connexions multiples en même temps et utiliser différentes bases de données simultanément.
- MYSQL dispose d'un système de contrôle intégré qui interdit la consultation de données à ceux qui n'en ont pas l'autorisation.

-

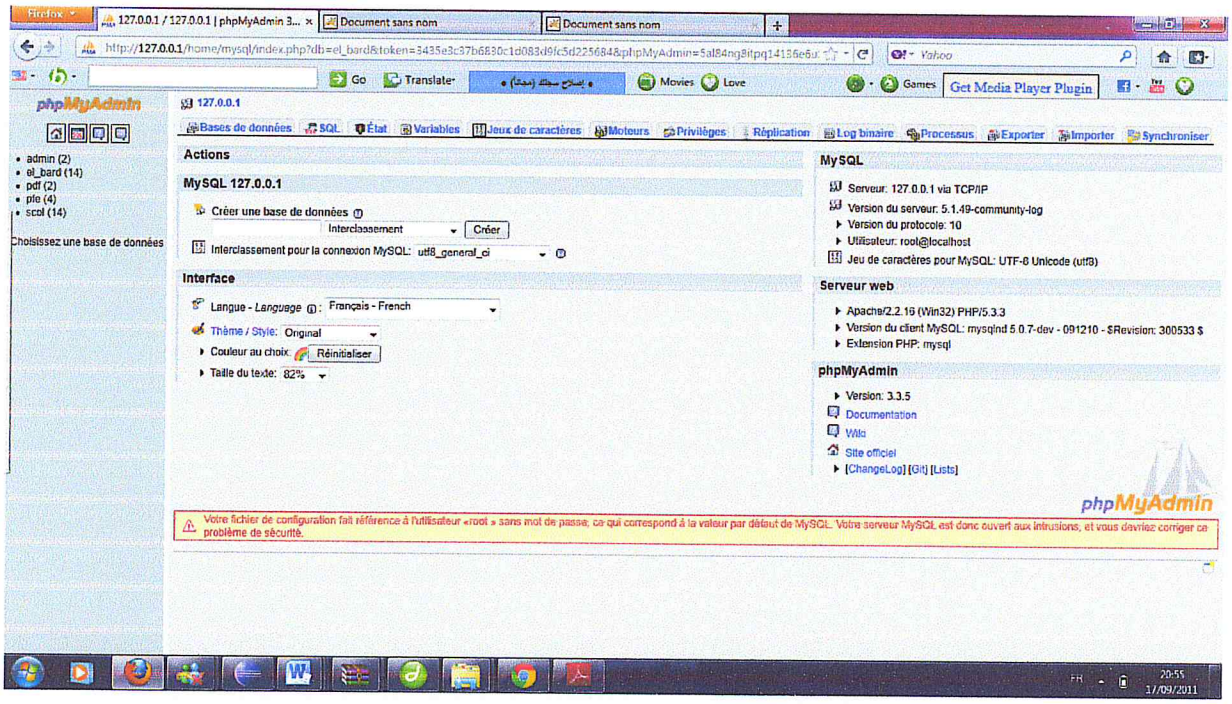


Figure 29 : Interface MySQL

II-3Eclipse JEE :

Java Platform, Enterprise Edition ou Java EE est une plate-forme largement utilisé pour la programmation web dans le langage de programmation Java. La plate-forme Java (Enterprise Edition) est différente de la plate-forme Java Standard Edition (Java SE) en ce qu'il ajoute les bibliothèques qui fournissent des fonctionnalités pour déployer à tolérance de pannes, distribués, à plusieurs niveaux du logiciel Java, largement basée sur des composants modulaires fonctionnant sur un serveur d'application.

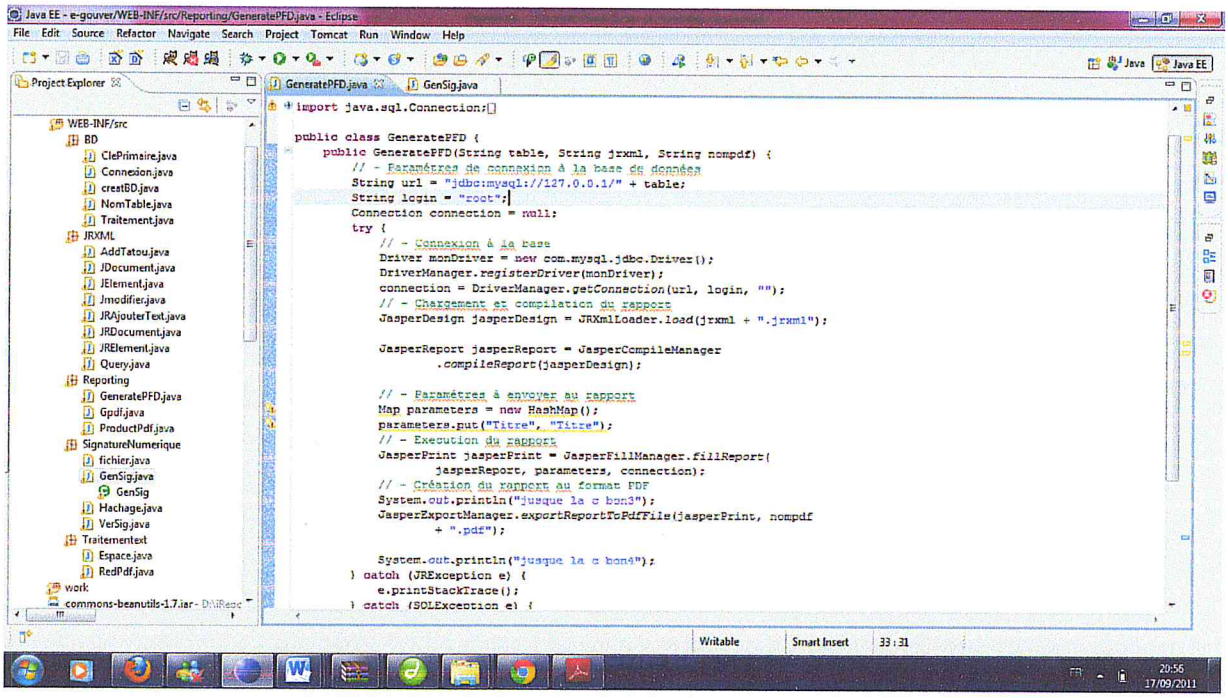


Figure 30 : Interface Eclipse JEE

III- Architecture technique du Système :

Les communications entre l'administrateur, client et le serveur web se font avec le protocole HTTPS.

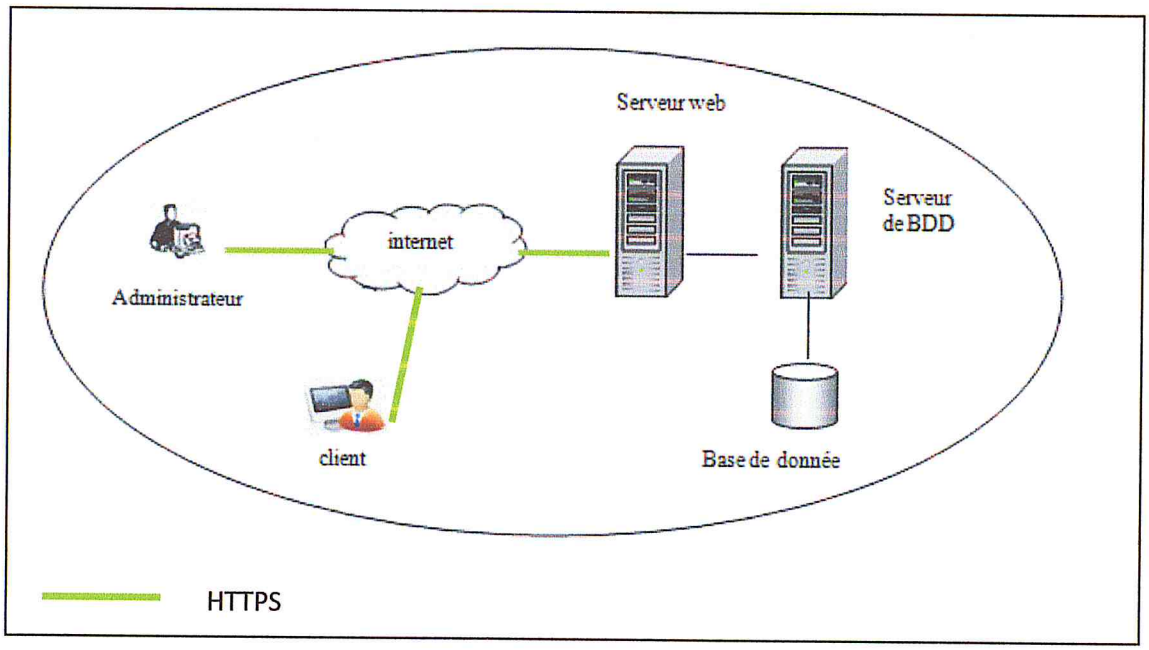


Figure 31: Architecture technique du Système

IV-Implémentation:

L'implémentation du système s'est deroulé en trois phases :

- l'implementation des classes .
- l'implémentation de la base de données.
- l'implementation des page JSP.

Les classes utilisées sont nombreuses,elles sont reparties sur quatre paquetages :

- **Paquetage BaseDonnées :**
  les classes portant sur le traitement des bases de données ont été créées dans ce paquetage.
- **Paquetage JRXML** :

les classes qui portent surla gestion des modeles JRXML sont presents sur ce paquetage.

- **Paquetage Reporting**
  ce Paquetage regroupe les classes qui s'occupent de la generation du rapport a partir du modele JRXML.

- **Paquetage Signature numerique**
  Dans ce paquetage sont present les classes qui permettent la signature numerique , le tatouage numerique ainsi que la verification des documents.

- **Paquetage Traitement**
  Ce dernier paquetage porte sur les differents traitements necessaires au bon fonctionnement du système

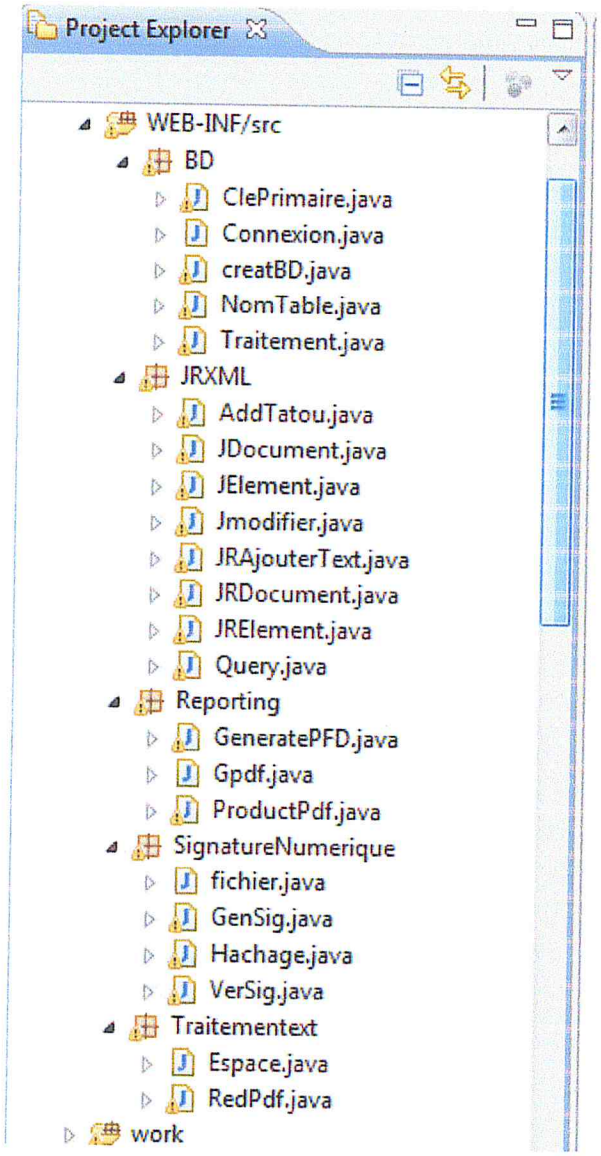


Figure 32 : Liste Des Paquetages

V- Les Classes Principales :

❖ Classe GeneratePDF

La classe GeneratePDF permet de generer un rapport au format pdf à partir d'un modele JRXML en utilisantJasperReports comme outil (librairie) pour la génération d'états.

elle prend en parametre le nom de la base de données et le modele jrxml.

Sonfonctionnement est relativement simple. Une fois le modèle jrxml (JasperDesign) compilé, il est chargé dans un objet Java (JasperReport) qui peut lui-même être sérialisé et stocké dans un fichier (avec l'extension .jasper). Cet objet sérialisé est alors utilisé pour compléter le rapport avec des données. En fait, la définition du rapport nécessite la compilation de toutes les expressions Java déclarées dans le modèle XML. Le résultat obtenu après le processus de remplissage des champs est un nouvel objet Java (JasperPrint) qui représente le document final.
Celui-ci est directement transformé dans un format lisible PDF.

Afin de pouvoir employer correctement ces différents objets, nous devons utiliser un certain nombre de bibliothèques (.jar) indispensables au bon fonctionnement du programme Java:

– **jasperreports-1.0.2.jar** : cette bibliothèque possède toutes les fonctionnalités propres àJasperReports.

– **commons-digester.jar** : cette bibliothèque permet de tranformer un fichier XML en objets Java.

– **commons-collections-3.1.jar** : cette bibliothèque introduit de nouvelles classes, de nouvelles interfaces et apporte des fonctionnalités supplémentaires aux classes de base

– **commons-beanutils.jar** : cette bibliothèque apporte une aide à la génération d'objets Java.

– **commons-logging.jar** : cette bibliothèque fournit, de manière transparente, des fonctionnalités de log en utilisant n'importe lequel des frameworks de log existants.

– **jdt-compiler.jar** : cette bibliothèque est obligatoire pour développer avec Eclipse.

– **itext-1.3.jar** : cette bibliothèque permet de générer des fichiers PDF à la volée.

– **hsqldb.jar, ojdbc14.jar, ...** : ces bibliothèques permettent de communiquer avec la base de données.

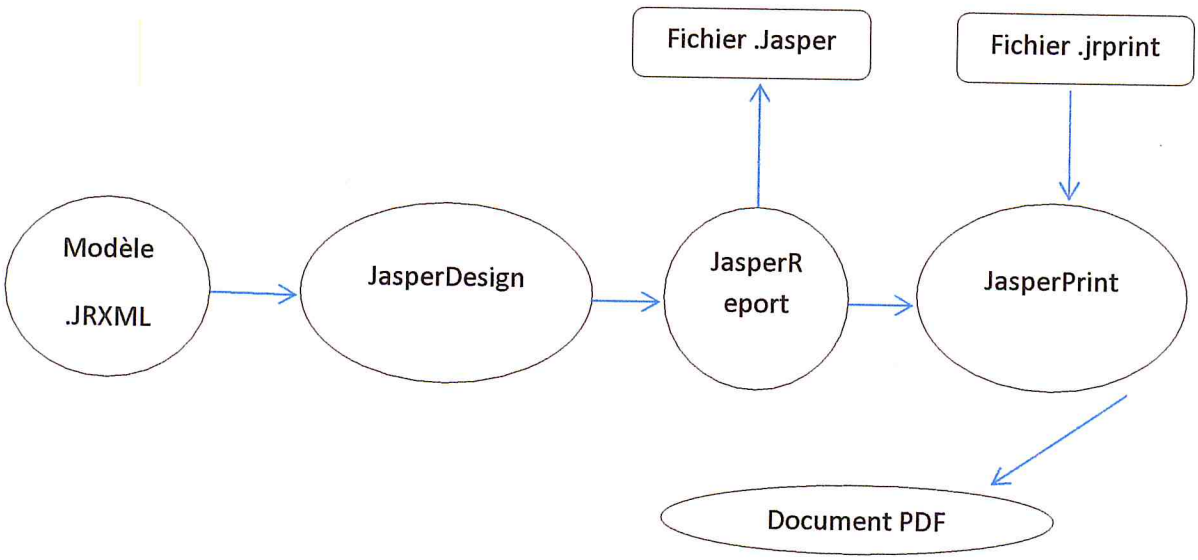


Figure 33 : Processus de Génération d'état avec Jasper Report

Extrait de la classe :

```
publicclassGeneratePFD {
      publicGeneratePFD(String table, String jrxml, String nompdf) {

                                                .....


// - Chargementet compilation durapport

                  JasperDesignjasperDesign = JRXmlLoader.load(Modele.jrxml);
      //-Creation d'un objet jasperReport
                  JasperReportjasperReport = JasperCompileManager
                              .compileReport(jasperDesign);

                  // - Paramétres à envoyeraurapport
                  Map parameters = newHashMap();
                  parameters.put("Titre", "Titre");
                  // - Executiondurapport
                  JasperPrintjasperPrint = JasperFillManager.fillReport(
                              jasperReport, parameters, connection);
                  // - Créationdurapportau format PDF

            JasperExportManager.exportReportToPdfFile(jasperPrint, nompdf.pdf);


                                                .....

                                                }}
```

Figure 34: Extrait de la classe GeneratePDF

❖ Classe GenSig :

La classe GenSig est responsable de la signature numerique du document à l'aide des deux l'algorithmes RSA et SHA1 elle prend en parametre le document PDF .

à chaque appel de la classe GenSig une paire de clé (privé , publique) est générée :

KeyPairGeneratorkeyGen = KeyPairGenerator.*getInstance*("RSA")

SecureRandom random = SecureRandom.*getInstance*("SHA1PRNG","SUN")

Pour signer le document PDF,un objet algorithme de signature est instancié de la classe Factory Signature en prenant comme paramètre le nom de l'algorithme « RSA » :

Signature signalg = Signature.getInstance("RSA")

Pour préparer l'objet à la signature du message, la méthode initSign est utiliséeen prenant en paramètre la clé privée :

signalg.initSign(privkey)

La méthode update est utiliséepour ajouter des octets aux objets algorithme :

byte[] bytes = ...
while (...) signalg.update(bytes)

Le calcule de la signature se fait à l'aide de la méthode sign(). La signature est renvoyée sous la forme d'un tableau d'octets.

byte[] signature = signalg.sign();

Enfin les clé public et clé privé sont enregistrées dans des fichiers sous format .txt

```
byte[] realSig = sig.sign()
FileOutputStreamsigfos = newFileOutputStream(fichier.txt)
sigfos.write(realSig)
```

Extrait de la classe :

```
publicclassGenSig {

publicvoidgenererSignatur(String chemin,String nom){
      String cle=null;
      try{
KeyPairGeneratorkeyGen = KeyPairGenerator.getInstance("RSA");
SecureRandom random = SecureRandom.getInstance("SHA1PRNG", "SUN");
.....

KeyPair pair = keyGen.generateKeyPair();
PrivateKeypriv = pair.getPrivate();
PublicKey pub = pair.getPublic();
.....

Signature sig = Signature.getInstance("SHA1withRSA");

sig.initSign(priv);
.....

FileInputStreamfis = newFileInputStream(pdf);
BufferedInputStreambufin = newBufferedInputStream(fis);
byte[] buffer = newbyte[1024];
intlen;
while (bufin.available() != 0) {
len = bufin.read(buffer);
sig.update(buffer, 0, len);
};

bufin.close();
.....

}}
```

Figure 35 : Extrait de la classe GenSig

❖ Classe VerSig :

La classe VerSig execute le processus de verification de la validité d'une signature numerique,elle prend en parametre le document signé et la pair de clé(privé,publique) et revoie un booelan indiquant l'etat de la signature numerique.

La classe doit générer un objet d'algorithme de signature RSA et le préparer pour la vérification de signature en appelant la méthode initVerifyavec la clé publique comme paramètre.

```
Signature verifyalg = Signature.getInstance("RSA");
verifyalg.initVerify(pubkey);
```

Puis envoyez le message à l'objet algorithme.

```
byte[] bytes = ...;
while (...) verifyalg.update(bytes);
```

Enfin, vérifiez la signature.

```
boolean check = verifyalg.verify(signature);
```

Extrait de la classe :

```
publicclassVerSig {


publicbooleanVerSig(String chemin,String nom){
      boolean verifies = false;
try{
/* import encoded public key */


FileInputStreamkeyfis = newFileInputStream(Clepublic);
byte[] encKey = newbyte[keyfis.available()];
keyfis.read(encKey);

keyfis.close();

      X509EncodedKeySpec pubKeySpec = newX509EncodedKeySpec(encKey);

KeyFactorykeyFactory = KeyFactory.getInstance("RSA");
PublicKeypubKey = keyFactory.generatePublic(pubKeySpec);


    Signature sig = Signature.getInstance("SHA1withRSA");
sig.initVerify(pubKey);
……..
verifies = sig.verify(sigToVerify);
……..
Return verifies ;}}
```

Figure 36 : Extrait de la Classe VirSig

VI-Présentation du système realiser :

Dans ce qui suit nous allons présenter les fonctionnalités les plus importantes du système à travers leurs interfaces.

VI-1 Page d'acceuil :

Figure 37 : Page d'accueil

Description figure :

La page d'accueil offre deux principaux espaces

- un espace pour la connexion des personnesayant un compte sur le système
- un espace pour l'authentification des documents et lavérification de la validité des documents

VI-2Interface Authentification Des Documents:

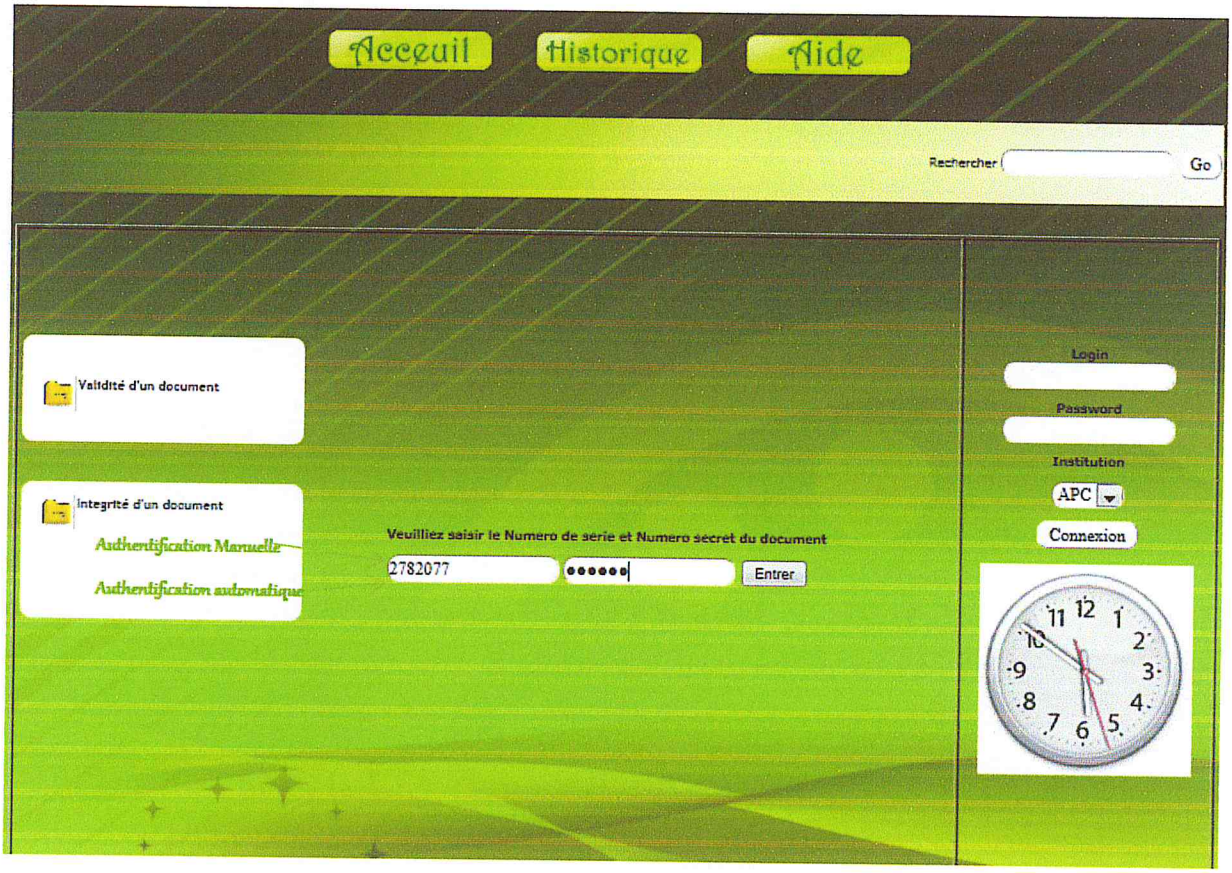


Figure 38: Interface Authentification des documents

Description figure :

Les deux types d'authentification de document requiert la saisie du numéro du Document et de sonnuméro secret comme le montre la figure ci- dessus.

VI-3 Interface de chargement du document :

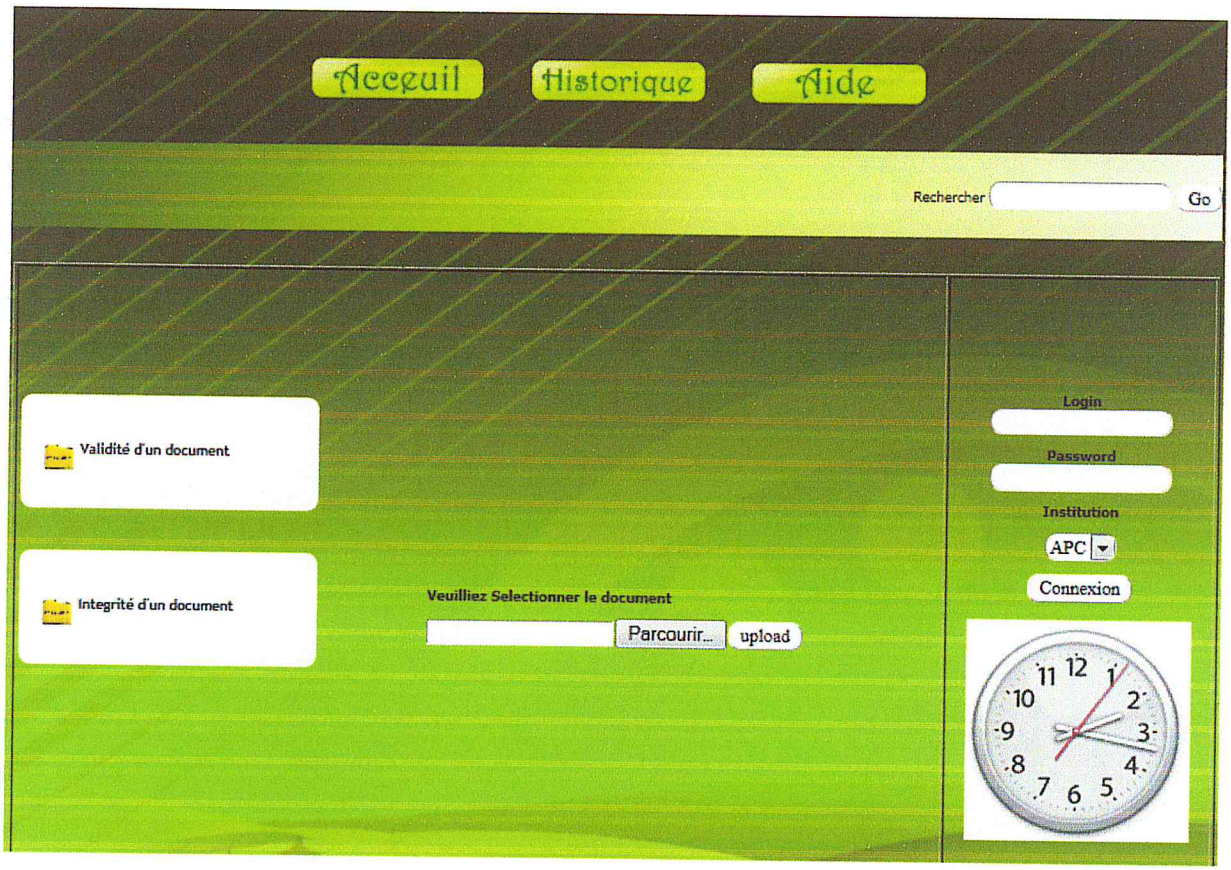


Figure 39 : Interface chargement du document

Description figure :

La figure ci-dessus est affichée à l'écran au moment où une personne veut authentifier un document numérique après la saisie du numéro et du numéro secret du document.

Une fois le chargement terminé , une boite de dialogue est affichée à l'écran indiquant l'état du document chargé sur le système.

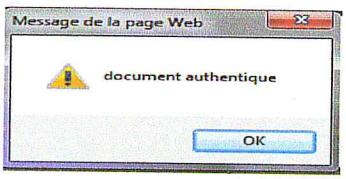


Figure 40: Message état document

VI-4 Interface Gestionnaire d'une institution :

Figure 41 : Interface Gestionnaire D'une Institution

Description figure :

Une fois le gestionnaire authentifié il accède àsa propre interface illustrée par la figure ci-dessus lui permettant de procéder a :

- La Gestion des modèles de document.

-Le contrôle d'accès des agents de saisie aux documents

-L'authentification des documents.

-La production des documents.

VI-5 Interfaces de création des documents :

Figure 42 : Interface s de création des documents

Description figure :

La figure ci-dessus permet au gestionnaire de créer un nouveau modèle de document à travers ce formulaire qui permet la saisie des informations sur le modèle de document qui sont :

-Le Titre du document

-Les attributs et les champs du document

-La position de chaque zone de texte dans le document

-La police de chaque zone de texte du document.

-La taille de chaque zone de texte du document.

-La couleur de chaque zone texte du document.

VI-6 Interface affichage modèle de document crée :

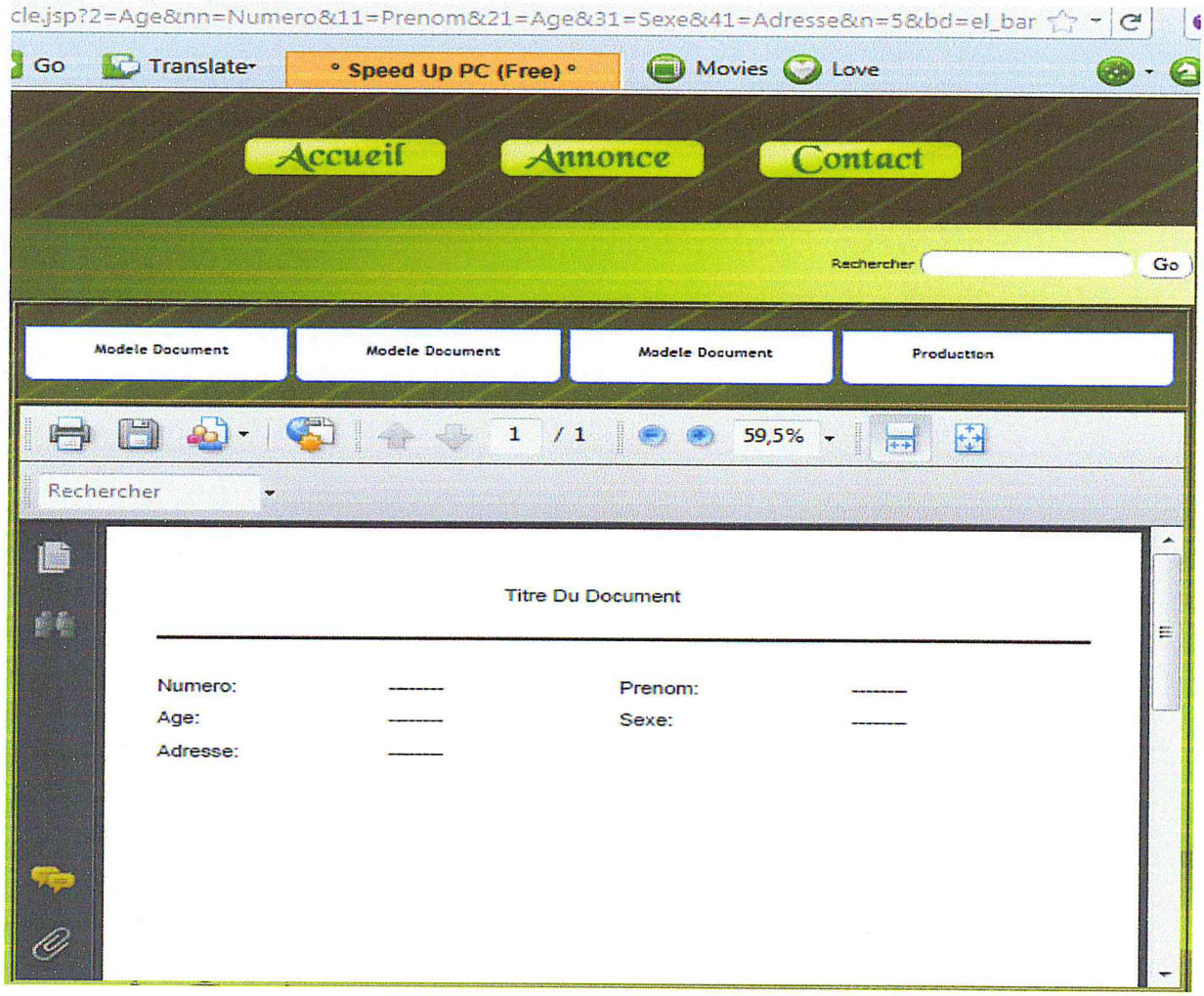


Figure 43 : Interface affichage modèle de document crée

VI-7 Interface de l'agent de saisie :

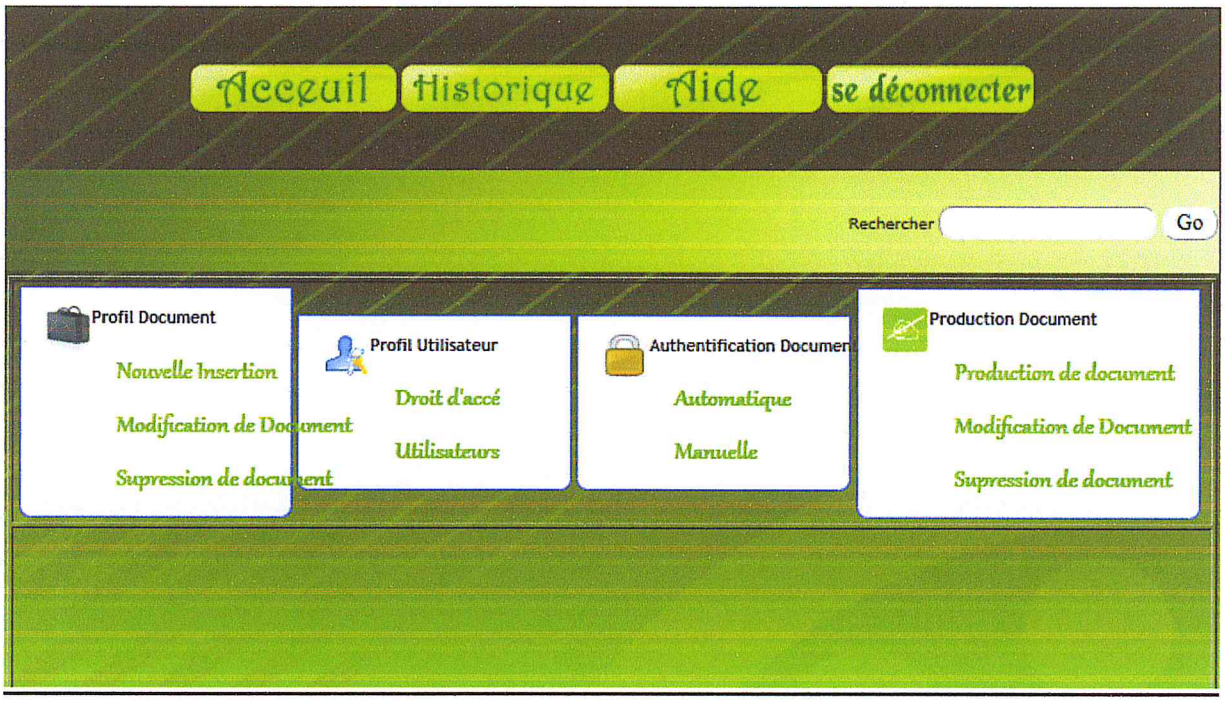


Figure 44 : Interface de l'agent de saisie

Description figure :

Apres authentification l'agent de saisie accède à sa propre interface illustrée par la figure ci-dessus lui permettant de faire :

-La gestion des profils des documents auxquel il peut accéder.

-La production de document

-L'authentification de document.

-La gestion du contrôle d'accès des utilisateurs.

VI-8 Interfaces Production du document :

Figure 45 : Interface production de document

Description figure :

La figure ci- dessus illustre le processus de production de document.

VI-9 Interface Utilisateur :

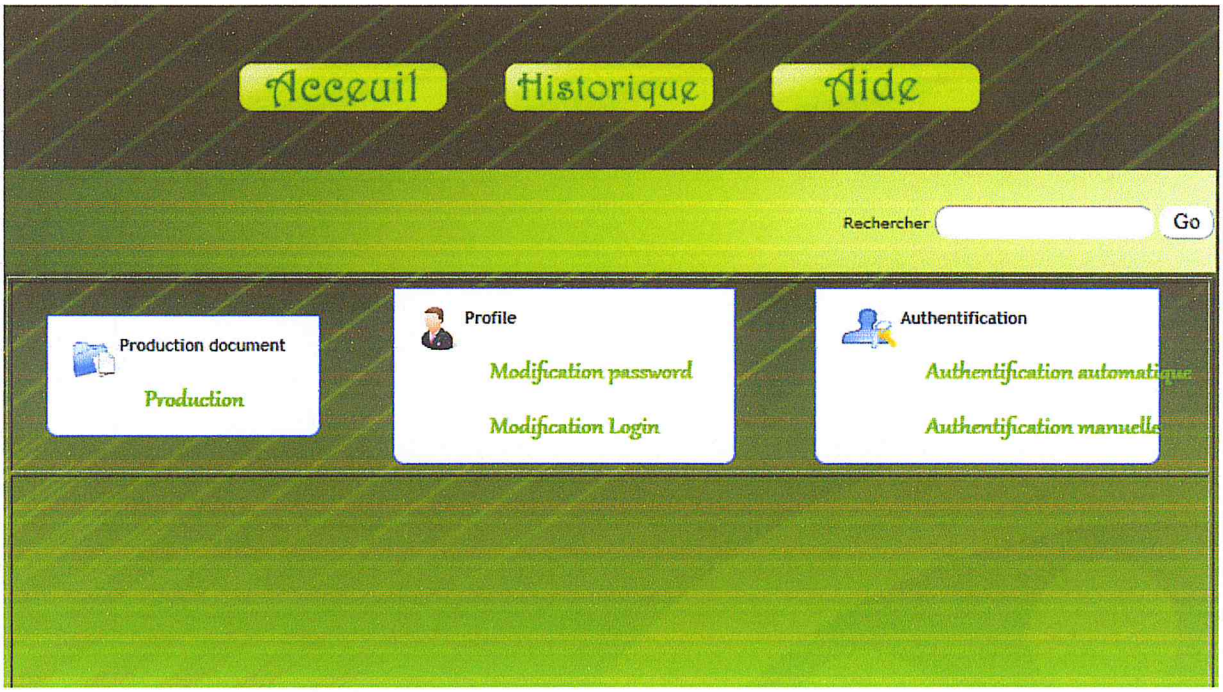


Figure 46: Interface utilisateur

Description figure :

L'interface des utilisateurs externes est illustrée par la figure ci-dessus , elle permet à chaque utilisateur d'accéder à sonprofile, d'accèder au document auquel il a le droit , de les produire et d'authentifier un document .

VI-10 Interface production de document d'un utilisateur :

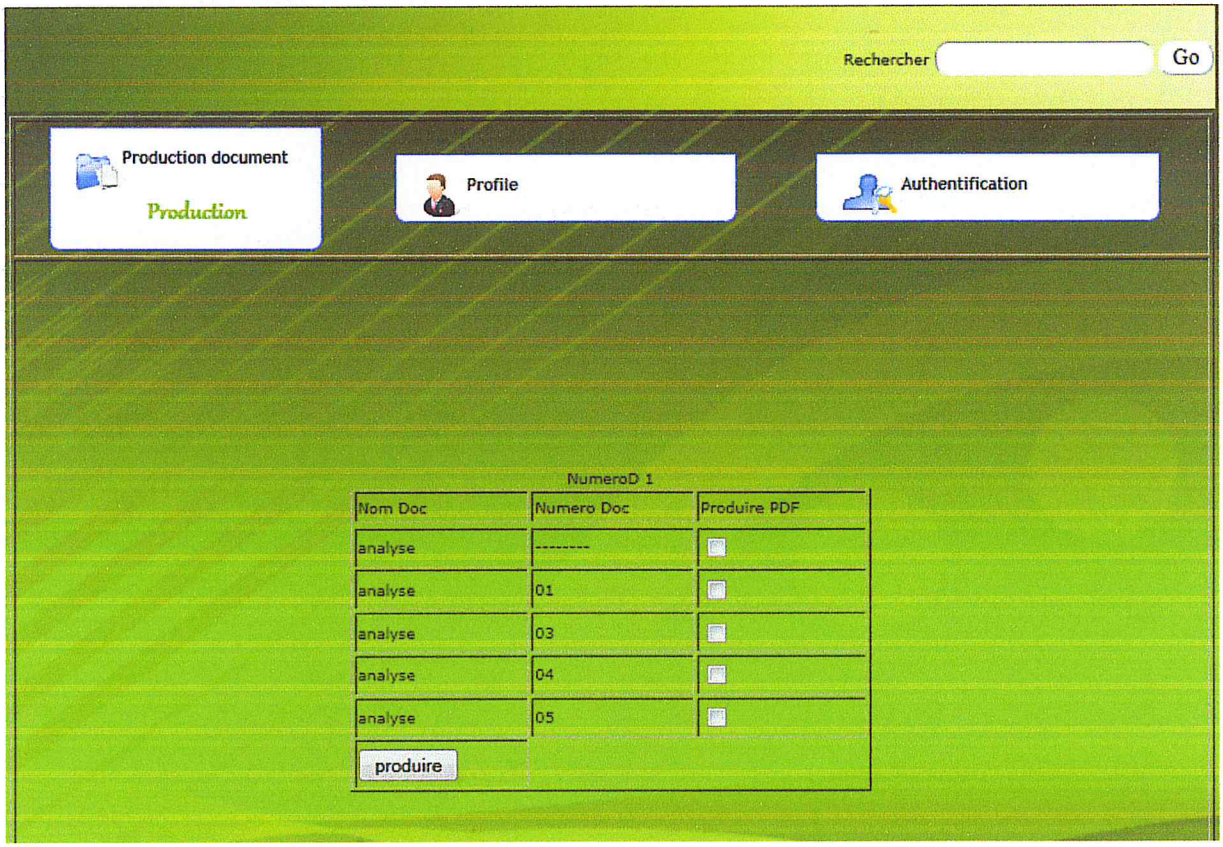


Figure 47 : Interface production de document d'un utilisateur

Description Figure :

Lorsqu'un utilisateurveut produire un document , la liste de tous les documents à laquelle il a droit d'accéder est affichée sur un tableau comme il est illustré sur la figure ci-dessus, l'utilisateur choisitparmi la liste un document à produire en cochant la case correspondante ; toute suite le document est affiché à l'écran comme il aété illustré dans la figure 45.

VII-Conclusion :

Ce chapitre, le dernier de ce rapport, a présenté les choix qui ont été faits pour élaborernotre système, les outils utilisés, ainsi que le résultat de notre travail.

CONCLUSION

CONCLUSION GENERALE

L'objectif de ce projetest de concevoir et réaliser un système de production de documents authentifiables permettant au propriétaire de ces derniers de se les procurer à tout moment et à n'importe quel endroit tout en garantissant leur authenticité.

Pour atteindre cette objectif ,nous avons étudiéles différentes techniques de Reporting existantes ce qui nous a permis de choisir les meilleurs approches pour permettre la production des documents.

Nous avons côtoyé de trèsprès le domaine de la cryptographie et du Tatouage Numérique ,ce qui nous a permis d'assurer l'authenticité des documents produits .

Nous avons développé la conception et la modélisation de notre système par le langage UML, vu sa capacité à donner une vue globale et intégrale sur le système ,ceci nous a permis de donner une image plus claire sur le problème,pour élaborer les concepts et les éléments de base constituant notre application,et de dégager les fonctionnalités qu'elle devrait couvrir.

Ce travail ne s'est pas déroulé sans difficultés , il nous a permis d'acquérir des connaissances dans le domaine de Reporting et de la Cryptographie .

L'objectif principal de notre projet a été atteint .Nous avons pu réaliser un système simple et ergonomique, qui assure la tache de production et d'authentification des documents.

Annexes

Annexe A

Outils de ReportingJasperreport

1-PANORAMA DES OUTILS DE REPORTING [12]:

Depuis de nombreuses années, les sociétés les plus influentes du marché de l'informatique montrent un intérêt certain au monde du reporting. En effet, afin de concurrencer les leaders mondiaux dans ce domaine (en particulier, *Business Objects*, *Cognos*ou *Hyperion*), ***Microsoft*** et ***Oracle*** proposent dorénavant de véritables solutions venant en complément de leur « Système de Gestion de Bases de Données » (SGBD) respectif. Ainsi, ils ne se contentent plus de la répartition des rôles que l'on connaissait auparavant : aux éditeurs de bases de données les fonctions de stockage, aux éditeurs d'outils de reporting les fonctions d'interrogation. D'un côté, *Microsoft* a développé *Reporting Services*, une gamme de produits intégrés à *SQL Server* et basés sur les
technologies acquises lors du rachat d'*Active Views*, de l'autre, *Oracle* a renforcé son offre de reporting dans la version *10g* de son célèbre SGBD. Face à ces deux offensives, les leaders du reporting contre-attaquent : ***Business Objects***sort sa nouvelle plate-forme, *BusinessObjects XI*, qui reprend en grande partie les éléments de l'infrastructure de *Crystal Decisions*(racheté en 2003) et annonce un accord de partenariat avec *MySQL*, ***Cognos***lance *ReportNet*, qui a remplaçé l'ancien logiciel de reporting*Impromptu.*
A côté de ces « géants » de l'informatique, d'autres outils de reporting tentent de se faire connaître. Certains sont même développés en Open Source sous licence « Apache Software License », « GNU General Public License » (GPL) ou « GNU Lesser General Public License » (LGPL). Citons, par exemple, des logiciels comme *DataVision*, *JasperReports*ou *JfreeReport.*

Descriptif des différentes catégories d'outils ou d'utilitaires de reporting :

– Logiciels de reporting commerciaux (contiennent à la fois les librairies pour générer les rapports, un éditeur graphique de rapports et des options de déploiement) :
– **Crystal Reports** (Business Objects)
– **ReportNet**(Cognos)
– **Hyperion Reports** (Hyperion)
– **Actuate** (Actuate)
– **KSL** (KSL, anciennement Kallisto Informatique)
– **Reporting Services**(Microsoft)
– **Oracle Business Intelligence 10g** (Oracle)
– ...
– Logiciels de reporting en Open Source (tentent d'offrir les mêmes fonctionnalités que les logiciels de reporting propriétaires) :

– **DataVision**(licence *Apache Software license*)
– **Agata Report** (licence *GPL*)
– ...

– Librairies de génération de rapports en Open Source :
– **JasperReports** (licence *LGPL*)
Il doit être couplé avec un éditeur graphique (Graphical User Interface ou GUI) afin de faciliter la création des rapports. Il en existe un certain nombre que nous allons citer ci-dessous :
– **JasperAssistant**: éditeur très complet mais propriétaire, peut s'intégrer dans l'environnement de développement Eclipse
– **iReport** : éditeur complet, pratique et gratuit (licence *GPL*)
– **OpenReportsDesigner** : éditeur complet et gratuit (licence *GPL*)
– ...
– **JFreeReport** (licence *LGPL*) De même que JasperReports, il faut le coupler avec un éditeur graphique :
– **JFreeDesigner**: éditeur gratuit mais encore en cours de développement (licence *GPL*)

2-Jasprereport :

2-1 Présentation Générale :

JasperReports est un outil (librairie) Open Source puissant utilisé pour la génération d'états. Il permet de créer des rapports à partir de fichiers XML. Le résultat peut être affiché à l'écran, imprimé ou stocké dans des fichiers au format PDF, HTML, XLS, CSV ou XML[12].

JasperReports est entièrement développé en Java et peut être intégré dans une gamme très variée d'applications Java (y compris les applications J2EE). Son objectif principal est de fournir un moyen simple et flexible pour la génération de documents[12].

La librairie JasperReports a été conçue en 2001 par *TeodorDanciu*, qui a également participé à de nombreux autres projets Open Source (Hibernate, framework Avalon, ...). Il continue régulièrement à proposer de nouvelles évolutions pour JasperReports. Cependant, il n'imaginait pas un tel succès (plus de 300 000 téléchargements et 11 000 nouveaux téléchargements par mois). C'est pour cette raison qu'une nouvelle companie a vu le jour il y a peu de temps, ***JasperSoft***, qui a été formée afin d'investir dans le développement de JasperReports et afin d'offrir un support, des services et des produits commerciaux destinés à compléterJasperReports. JasperSoft est composée d'un certain nombre de développeurs qui vont pouvoir travailler à plein temps sur JasperReports, mais également sur un outil payant, ***JasperDecisions***, qui va proposer des fonctionnalités supplémentaires à JasperReports. JasperDecisions est composé de deux produits principaux, *Scope Server* (solution de reporting) et *Scope Designer* (outil graphique destiné à produire les rapports).
C'est donc une nouvelle étape importante dans le développement de JasperReports qui pourra ainsi prétendre à devenir l'une des meilleures solutions de reporting du marché. En outre, TeodorDanciu a récemment participé à deux conférences importantes, « MySQL UsersConference 2005 » et « JBoss World 2005 » , ceci afin de présenter son outil à un plus large public et afin de faire une démonstration des capacités d'intégration de JasperReports dans un environnement utilisant ***JBoss***et ***Hibernate***[12].

2-2 Création de Rapport avec JasperReport :

La création de rapports avec JasperReports se déroule généralement en 4 étapes :

• l'obtention d'un fichier modèle XML (à l'aide d'éditeurs graphiques comme iReport ou OpenReports Designer)

• la construction du rapport à partir du modèle

• le remplissage des différents champs du rapport avec les données en provenance de diverses sources (bases de données, classes Java, ...)

• l'exportation du résultat dans plusieurs formats possibles (PDF, HTML, ...)

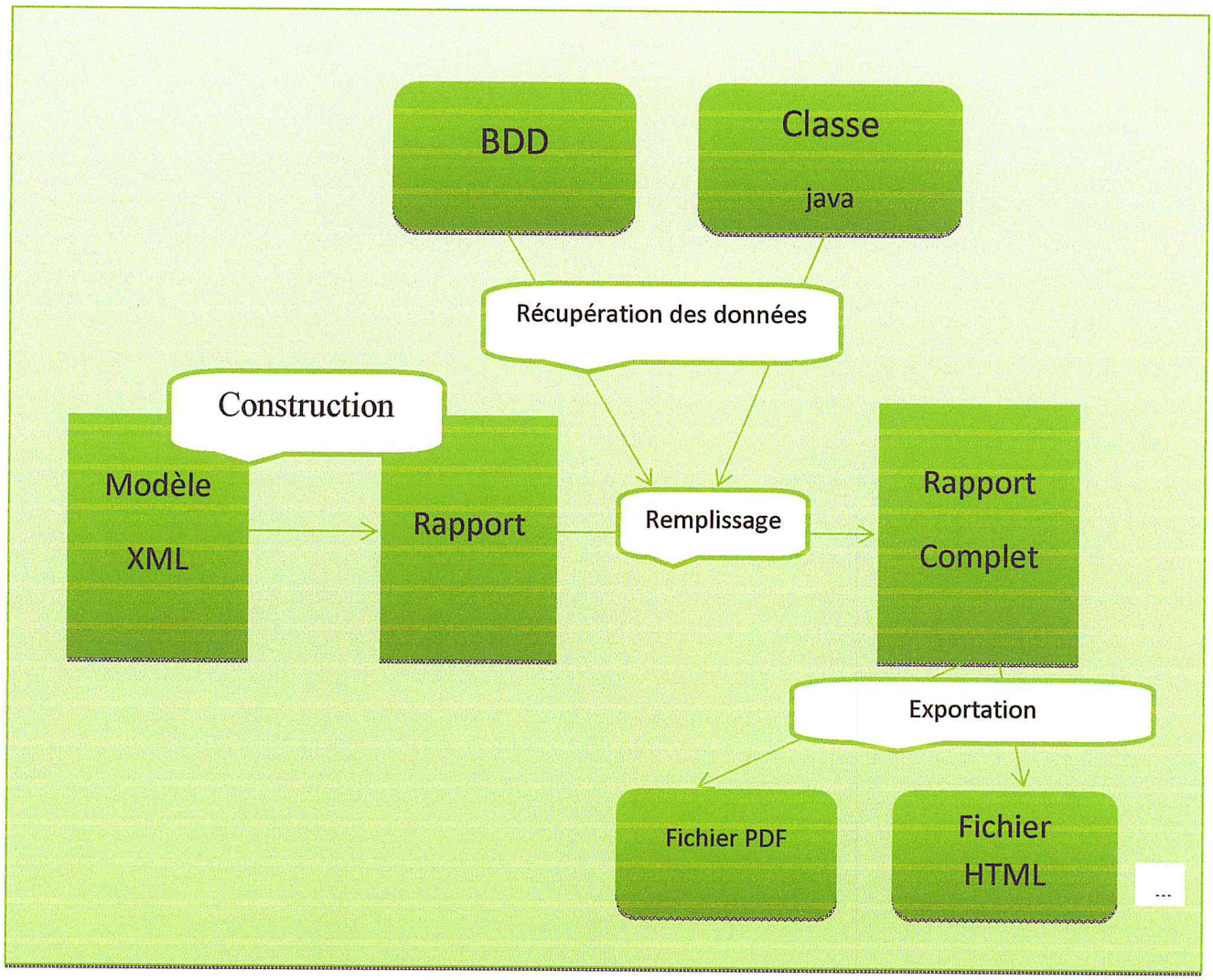


Figure 48:Etapes de creation d'un rapport

2-3 Fonctionnement de JasperReport :

Le fonctionnement de JasperReports est relativement simple. En effet, tous les concepts tournent autour du langage Java. Une fois le modèle XML (JasperDesign) compilé, il est chargé dans un objet Java (JasperReport) qui peut lui-même être sérialisé et stocké dans un fichier (avec l'extension .jasper). Cet objet sérialisé est alors utilisé lorsque l'application désire compléter le rapport avec des données. En fait, la définition du rapport nécessite la compilation de toutes les expressions Java déclarées dans le modèle XML. Le résultat obtenu après le processus de remplissage des champs est un nouvel objet Java (JasperPrint) qui représente le document final.
Celui-ci peut être stocké sur disque pour un usage ultérieur (sous forme sérialisée et avec l'extension .jrprint), directement imprimé ou encore transformé dans un format lisible (PDF, HTML, ...).**[13]**

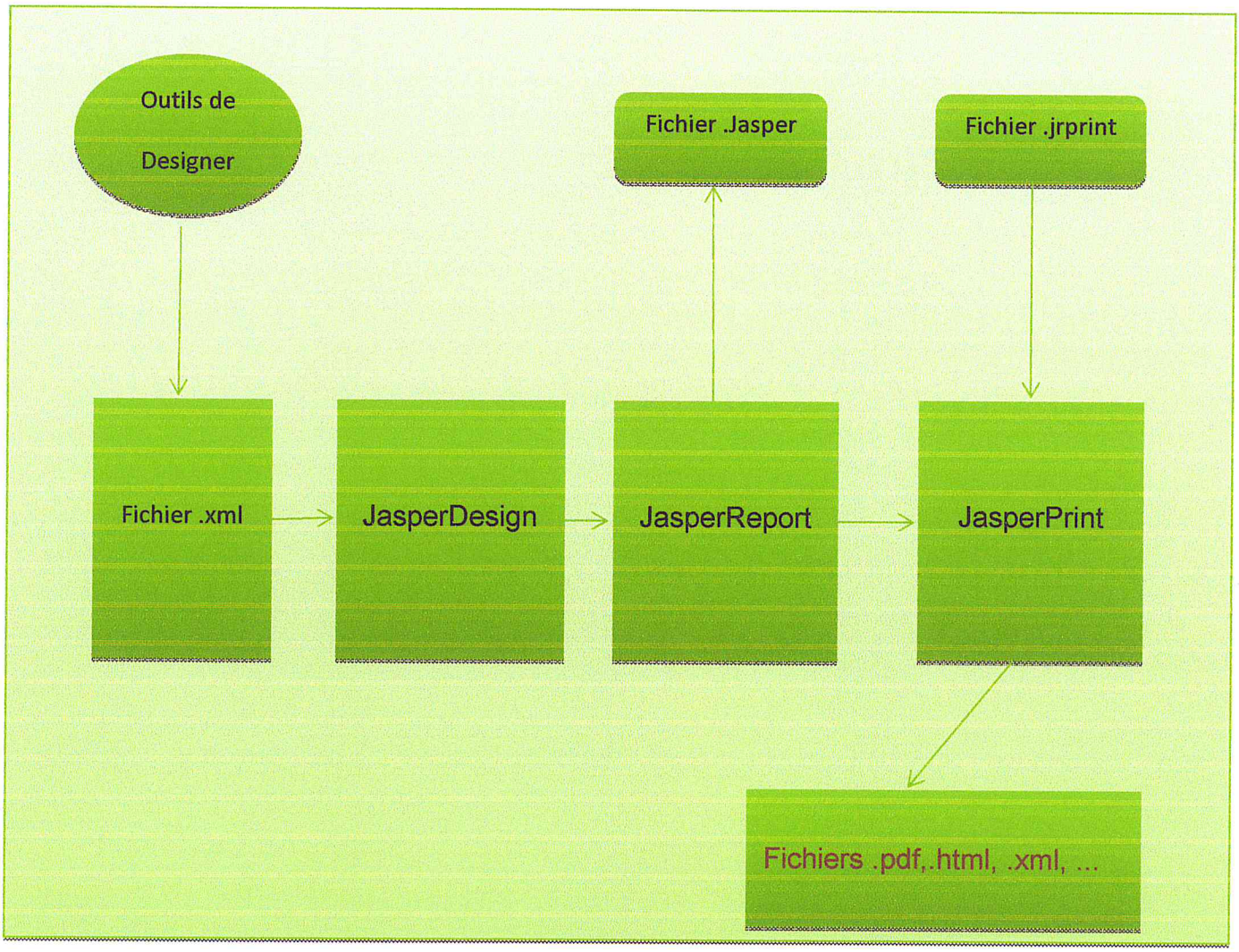


Figure 49 :Fonctionnement de jasperreport

2-4 Autre fonctionnalité de JasperReport :

Durant cette étude, certaines possibilités offertes par JasperReports ne seront pas abordées,vue que nous nous somme limité au Besoin de notre projet .
Il faut savoir que JasperReports dispose d'atouts supplémentaires :

• associer plusieurs rapports afin d'obtenir un unique rapport à exporter

• utiliser des liens dans le rapport ou en direction d'Internet

• élaborer des graphiques complexes à l'intérieur d'un rapport (avec les librairies **jCharts** pour la 2D et **jFreeChart**pour la 3D)

• protéger par mot de passe les fichiers au format PDF.

• transmettre des sous-rapports à un rapport principal au moyen de l'élément *paramètre.*

• employer un fichier XML comme source de données.

• utiliser des *scriptlets* qui apportent des fonctionnalités nouvelles par rapport aux variables (manipulation des données pendant le remplissage des champs).**[12]**

2-5 Présentation Du Modèle JRXML:

Afin d'utiliser la librairie JasperReports, il est nécessaire de créer un fichier modèle JRXML (JasperReport XML). Celui-ci doit avoir une structure bien particulière. Cette structure est déclarée dans un fichier au format DTD (Document Type Definition) [12].

Tous les fichiers XML doivent être définis de la façon suivante :

```
<?xmlversion="1.0" encoding="UTF-8"?>
<!DOCTYPE jasperReportPUBLIC "-//JasperReports//DTD Report Design//EN"
"http://jasperreports.sourceforge.net/dtds/jasperreport.dtd">
<jasperReportname="nom_du_rapport"    >
...
</jasperReport>
```

Les éléments principaux qu'il faut utiliser pour générer le modèle XML sont les suivants :

2-5-1 les paramètres :

Les paramètres sont des références à des objets qui sont passés au rapport au moyen de méthodes de remplissage écrites en Java. Déclarer un paramètre est très simple et requiert uniquement un nom et un type :

```
<parameter name="Titre" class="java.lang.String"></parameter>
```

Les paramètres sont très utiles afin de transmettre des données qui n'apparaissent pas dans les sources de données. Par exemple, nous pouvons passer au rapport le nom de l'utilisateur qui l'a généré ou bien modifier dynamiquement le titre du rapport.

2-5-2 les champs :

Les champs représentent la seule façon de récupérer les données à partir d'une source de données et de les transférer ensuite aux différentes routines gérant le rapport. Ces champs doivent avoir obligatoirement le même nom et le même type que les objets correspondants qui résultent de la requête réalisée sur la source de données.

Par exemple, nous désirons utiliser une table *Employes*qui possède la structure suivante :

Nom	Type	Longueur (Octet)
Numero	integer	4
Nom	varchar	20
Prenom	varchar	10
Salaire	double	8
DateEmbauche	date	4

Nous pouvons alors déclarer les champs :

```
<field name="Numero" class="java.lang.Integer"></field>
<field name="Nom" class="java.lang.String"></field>
<field name="Prenom" class="java.lang.String"></field>
<field name="Salaire" class="java.lang.Double"></field>
<field name="DateEmbauche" class="java.util.Date"></field>
```

2-5-3-Les Expression :

Les expressions sont un dispositif très important de la librairie JasperReports. En effet, elles peuvent être employées pour déclarer des variables qui exécuteront divers calculs, pour spécifier le contenu des champs du rapport ou pour personnaliser l'apparence des différents objets constituant le rapport. Fondamentalement, toutes les expressions sont des expressions Java qui référencent des champs ou des variables du rapport.

Il existe plusieurs manières pour définir des expressions :

```
<variableExpression>, <initialValueExpression>, <groupExpression>,
        <printWhenExpression>, <imageExpression>, <textFieldExpression>
```

Afin d'utiliser des champs dans une expression, le nom du champ doit être placé entre les deux séquences de caractères suivantes : **$F{**et **}.**

Par exemple, nous désirons afficher sur le rapport la concaténation du nom et du prénom des employés :

```
<textFieldExpression>
$F{Nom} + " " + $F{Prenom}
</textFieldExpression>
```

La syntaxe est la même pour les variables et les paramètres sauf qu'il faut les placer entre des séquences de caractères différentes : respectivement **$V{**et **}**, **$P{** et **}.**

2-5-4 les variables :

Les variables sont des objets spéciaux qui sont destinés à être utilisés exclusivement au sein des expressions. Elles peuvent être employées pour simplifier la conception du rapport en déclarant seulement une fois une expression qui sera utilisée à plusieurs reprises tout au long du modèle XML. Elles servent également à réaliser certains calculs dans les expressions où elles sont employées.
Dans une même expression, une variable peut référencer d'autres variables, mais seulement si celles-ci sont définies auparavant. L'ordre de déclaration des variables est donc primordial.
Comme cité précédemment, les variables peuvent exécuter plusieurs types de calculs différents :***count, sum, average, lowest, highest, variance, ...***

• Par exemple, nous désirons afficher sur le rapport la somme totale des salaires des employés :

```
<variable name="Somme" class="java.lang.Double" calculation="Sum">
<variableExpression>$F{Salaire}</variableExpression>
</variable>
```

Il est possible également de spécifier le niveau d'initialisation des variables. Le niveau par défaut est *Report*, ce qui signifie que les variables sont initialisées une seule fois au début du rapport. Il existe d'autres niveaux d'initialisation (*Page*, *Column*, *Group*) qui permettent de réaliser les calculs pour chaque page, chaque colonne ou chaque groupe.

2-5-5 les groupes :

Les groupes représentent une façon simple d'organiser les données à l'intérieur d'un rapport. Lorsqu'il remplit un rapport, le moteur de JasperReports teste toutes les définitions de groupes pour s'assurer qu'il n'y a pas de rupture de groupe et pour voir si les sections suivantes sont bien présentes pour chaque groupe :

```
<groupHeader>, <groupFooter>
```

Comme pour les variables, l'ordre de déclaration des groupes est important. En effet, un groupe contient toujours le groupe suivant et ainsi de suite.

Afin de déclarer un groupe, il faut procéder de la manière suivante :

```
<group name="Employes">
<groupExpression>...</groupExpression>
<groupHeader>
...
</groupHeader>
<groupFooter>
...
```

2-5-6 les requêtes :

Lorsque la source de données est une base de données, il est possible d'effectuer une requête directement dans le fichier modèle XML.

Par exemple, nous désirons récupérer le nom et le prénom de tous les employés :

```
<queryString>
SELECT Nom, Prenom FROM Employes
</queryString>
```

2-5-7 Structure d'un Modele JRXML :

Quand on construit un modèle JRXML, il est nécessaire de respecter la disposition des différentes sections. La structure du fichier est divisée en 9 parties :

- **background**
- **title**
- **pageHeader**
- **columnHeader**
- **detail**
- **columnFooter**
- **pageFooter**
- **lastPageFooter**
- **summary**

Chaque section est une partie du rapport bien particulière qui possède une largeur et une hauteur spécifiques et qui peut contenir des objets comme des lignes, des rectangles, des images ou du texte. Pour spécifier le contenu et la disposition d'une section du rapport, il est

indispensable d'utiliser l'élément générique suivant :

```
<band>
```

Les bandes peuvent inclure de multiples éléments, chacun d'eux étant identifié par une position,
une taille et une valeur :

```
<staticText>, <textField>, <line>, <rectangle>, <image>
```

Par exemple, nous désirons afficher, en bas de chaque page du rapport, le numéro de la page correspondante :

```
<pageFooter>
<band height="40">
<line>
<reportElement x="0" y="10" width="500" height="1"/>
<graphicElement/>
</line>
<staticText>
<reportElement x="0" y="20" width="50" height="15"/>
<textElement textAlignment="Right">
<font fontName="Arial" size="14" isItalic="true"/>
</textElement>
<text>Page : </text>
</staticText>
<textField>
<reportElement x="50" y="20" width="30" height="15"/>
<textElement>
<font fontName="Arial" size="14"/>
</textElement>
<textFieldExpression class="java.lang.Integer">
$V{PAGE_NUMBER}
</textFieldExpression>
</textField>
</band>
                                        </pageFooter>
```

2-6 Construction Du rapport :

2-6-1 Objets De Génération de Rapport :

Avant d'employer Jasper Reports, il est nécessaire de détailler les différents objets Java qui entrent en jeu dans le processus de génération d'un rapport, de la conception de celui-ci à la production d'états :

• **JasperDesign :**
Cet objet représente la définition d'un rapport. Dans la plupart des cas, nous devons créer un objet JasperDesign à partir d'un fichier modèle XML bien qu'il soit également possible de générer ce modèle au moyen de code Java**[12]**.

• **JasperReport :**
Cet objet représente un objet JasperDesign compilé. Le processus de compilation vérifie la structure du modèle XML, le compile et le stocke dans un objet JasperReport**[12]**.

• **JasperPrint :**
Cet objet représente le rapport final. Un objet JasperPrint est élaboré à partir d'un objet JasperReport par un processus de remplissage qui consiste à insérer dans le rapport des données en provenance d'une source de données quelconque**[12]**.

La figure suivante illustre le Processus de generation de rapport :

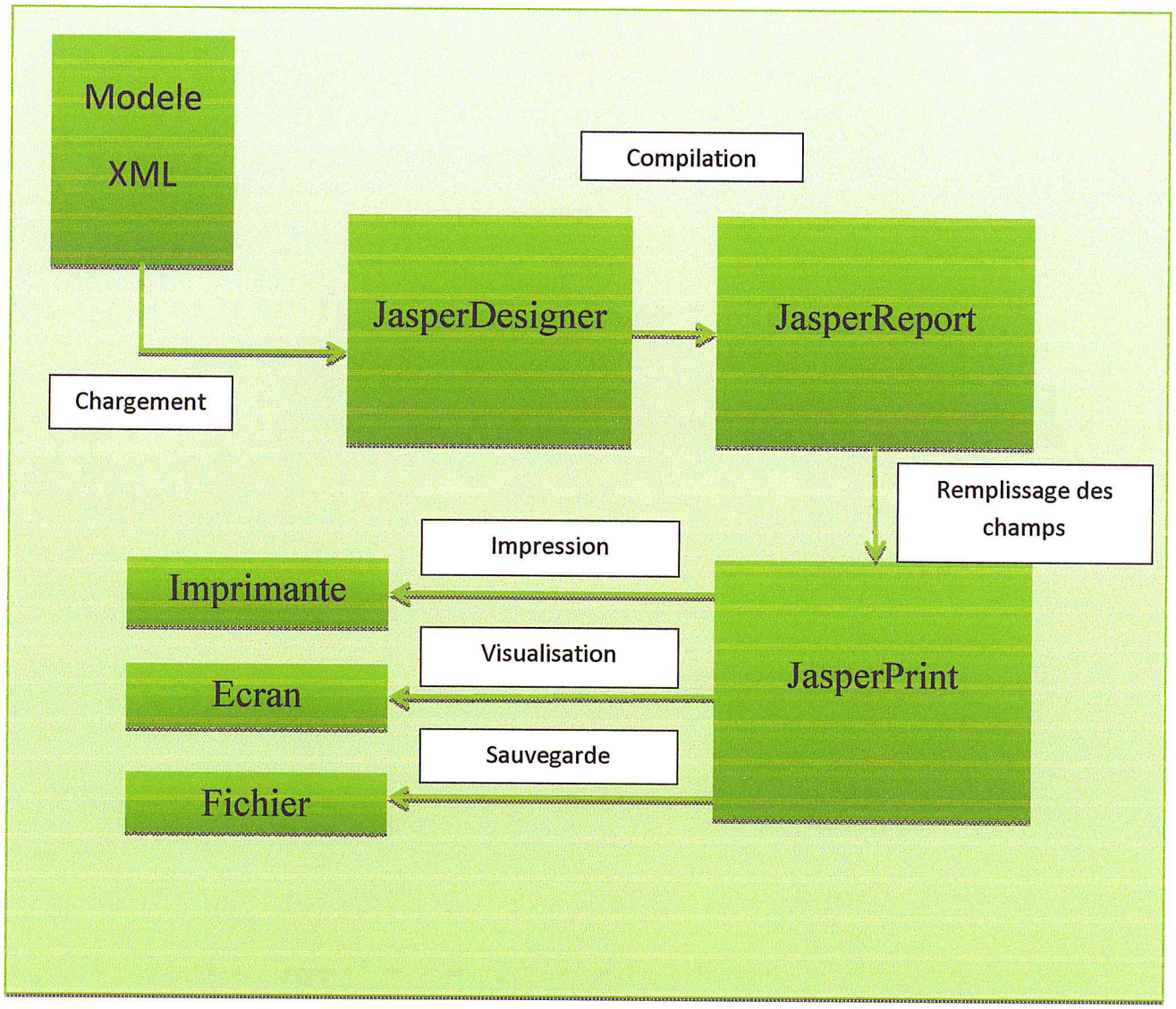


Figure 50 :Processus de génération de rapport

2-7 JasperReport dans une architecture Web :

JasperReports peut être utilisé dans un environnement WEB J2EE. Les traitements sont effectués coté serveur et le résultat est renvoyé au client[14].
Le principe est le même, JasperReports a besoin d'un modèle de rapport, de paramètres et de données :

• le modèle de rapport est stocké sur le serveur sous forme de fichiers xml (JasperDesign) ou sous forme de fichiers jasper (JasperReport)

• les données sont disponibles depuis une base de données

• les paramètres sont récupérés depuis un formulaire web

Exemple de fonctionnement utilisant le Framework JSF [1] :

· l'utilisateur rempli un formulaire web contenant tous les paramètres nécessaires à la génération du rapport.

· à la validation du formulaire, le framework JSF se charge de créer un Bean [2] contenant les paramètres renseignés par l'utilisateur, puis exécute une méthode pour générer le rapport. Cette méthode effectue les actions suivantes :

➢ charge le rapport (le compile si besoin) pour avoir un objet JasperReport

➢ utilise les paramètres enregistrés dans notre Bean et les données issues d'une base de données pour générer un rapport (objet JasperPrint)

➢ génère le rapport sous divers formats (pdf, html, etc.), le format du rapport peut être indiqué par le formulaire web
➢ le rapport est accessible à l'utilisateur depuis l'application web

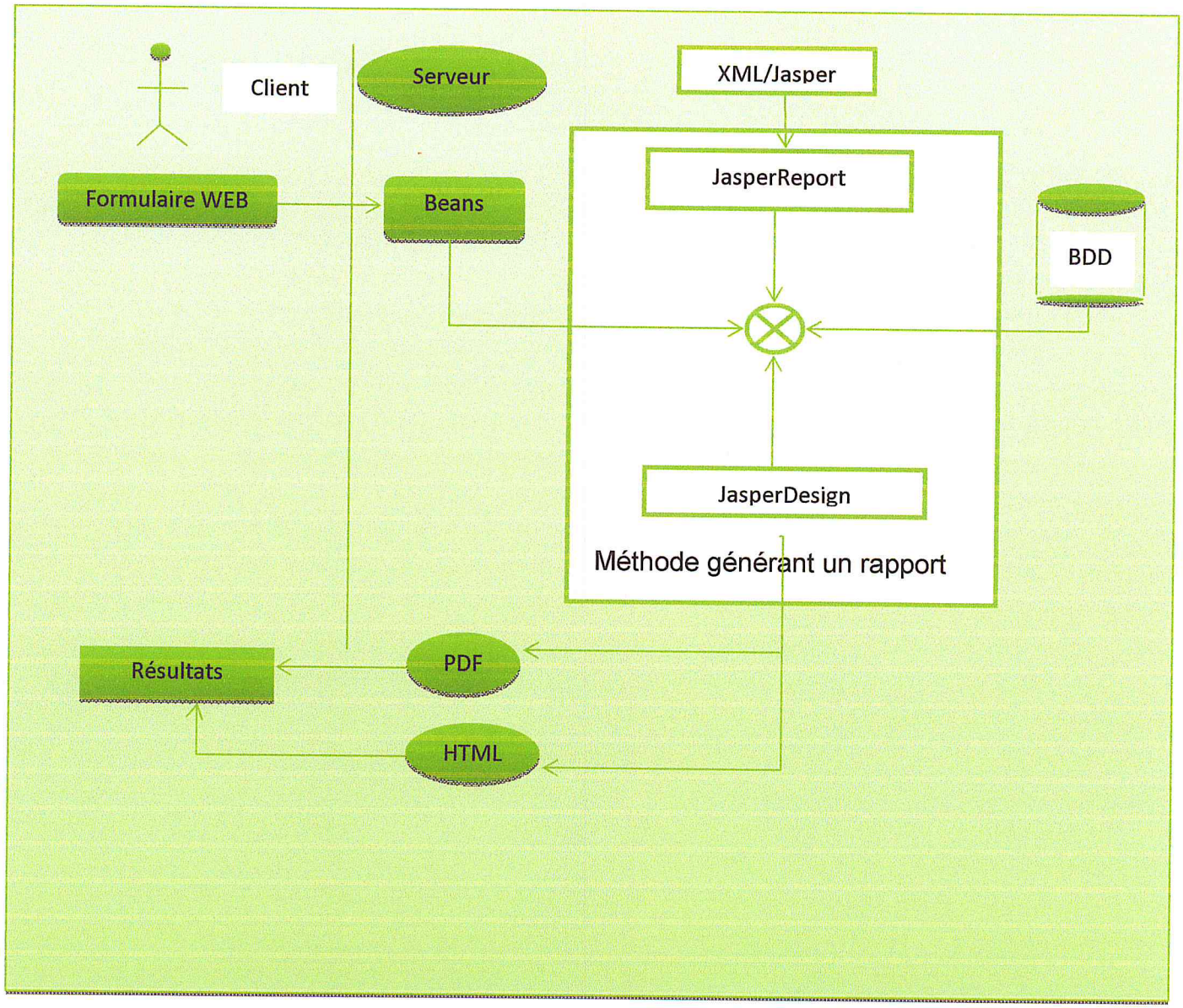


Figure 51 : JasperReport dans une architecture Web

Annexe B
Algorithme a clé publique RSA

Dans la cryptographie à clé publique (ou cryptographie asymétrique) chaque communicant utilisent deux clés, l'une est connue par tous (clé publique), l'autre n'est connue que par lui-même (clé privée). Le message crypté avec l'une ne peut être décrypté qu'avec l'autre [SER 03]. Les deux clés sont reliées mathématiquement entre elles de telle sorte qu'il est impossible de retrouver la clé privée en connaissant la clé publique.

1. L'algorithme RSA (RivestShamirAdleman)

Il existe différents algorithmes asymétriques. L'un des plus connus est le **RSA** (de ses concepteurs Rivest, Shamir et Adleman). Cet algorithme est très largement utilisé, par exemple dans les navigateurs pour les sites sécurisés et pour chiffrer les emails. Il est dans le domaine public.

1.2. Exemple :

Commençons par créer notre paire de clés:

- Prenons 2 nombres premiers au hasard: **p = 29**, **q = 37**
- On calcul **n = pq = 29 * 37 = 1073**
- On doit choisir **e** au hasard tel que **e** n'ai aucun facteur en commun avec **(p-1)(q-1)**:
- **(p-1)(q-1) = (29-1)(37-1) = 1008**
- On prend **e = 71**
- On choisit **d** tel que **71*d mod 1008 = 1**
- On trouve **d = 1079**
- On a maintenant nos clés :

  La clé publique est **(e,n) = (71,1073)** (=clé d'encryptage)

  La clé privée est **(d,n) = (1079,1073)** (=clé de décryptage)

On va encrypter le message '**HELLO**'. On va prendre le code ASCII de chaque caractère et on les met bout à bout:

m = 7269767679

Ensuite, il faut découper le message en blocs qui comportent moins de chiffres que **n**. **n** comporte 4 chiffres, on va donc découper notre message en blocs de 3 chiffres:

726 976 767 900

(on complète avec des zéros)

Ensuite on encrypte chacun de ces blocs:

$$726^{71} \bmod 1073 = 436$$
$$976^{71} \bmod 1073 = 822$$
$$767^{71} \bmod 1073 = 825$$
$$900^{71} \bmod 1073 = 552$$

Le message encrypté est **436 822 825 552**. On peut le décrypter avec d:

$$436^{1079} \bmod 1073 = 726$$
$$822^{1079} \bmod 1073 = 976$$
$$825^{1079} \bmod 1073 = 767$$
$$552^{1079} \bmod 1073 = 900$$

C'est à dire la suite de chiffre **726976767900**.

On retrouve notre message en clair **72 69 76 76 79** : '**HELLO**'.

1.3. Dans la pratique

Dans la pratique, ce n'est pas si simple à programmer:

- Il faut trouver de grands nombres premiers (ça peut être très long à calculer)
- Il faut obtenir des nombres premiers **p** et **q***réellement* aléatoires (ce qui est loin d'être évident).
- On n'utilise pas de blocs aussi petits que dans l'exemple ci-dessus: il faut être capable de calculer des puissances et des modulos sur de très grand nombres.

En fait, on utilise jamais les algorithmes asymétriques pour chiffrer toutes les données, car ils sont trop longs à calculer : on chiffre les données avec un simple algorithme symétrique dont la clé est tirée au hasard, et c'est cette clé qu'on chiffre avec un algorithme asymétrique comme le RSA.

3-Traiter un XML

Deux grandes techniques sont disponibles pour lire et récupérer des informations d'un fichier XML.

3.1 SAX :

La première technique est SAX (Simple Api for XML). Celle-ci se base sur un système producteur/consommateur.
Sans entrer dans les détails, il y a un processus qui lit le fichier XML et dès qu'il a lu entièrement une balise, il produit un événement, par exemple : "'ouverture balise"', "'fermeture balise"', etc.
C'est au consommateur de gérer les différents événements, afin d'effectuer les traitements nécessaires.
L'avantage de cette technique est qu'elle est très rapide et qu'elle ne consomme pas énormément de mémoire. Par contre, elle ne permet pas de récupérer ni de modifier les informations par la suite, ni même de créer un fichier XML**[16]**.

3.2 DOM :

L'autre technique est DOM (Document Object Model). DOM, lui, va lire le document XML et le stocker en mémoire sous la forme d'un arbre. Dès que le traitement est fini, il est tout à fait possible de voyager dans les noeuds de l'arbre, de les modifier ou de les supprimer.
L'avantage est que, dès la fin du traitement, tout le document est facilement accessible en lecture et en écriture, mais l'inconvénient est que cette méthode est beaucoup plus lourde en mémoire et en temps**[16]**.

Les éléments XML peuvent contenir des attributs, comme :

```
<size unit="pt">36</size>
```

Les concepteurs XML ne sont pas tous d'accord sur le moment auquel utiliser des éléments ou des attributs.
Par exemple, il semblerait plus facile de décrire une police comme ceci :

```
<font name="Helvetica" size="36"/>
que comme ceci :
<font>
<name>Helvetica</name>
<size>36</size>
</font>
```

Toutefois, les attributs sont bien moins flexibles. Supposons que vous vouliez ajouter des unités à la valeur de taille. Si vous utilisez des attributs, vous devez ajouter l'unité à la valeur de l'attribut :

```
<font name="Helvetica" size="36 pt"/>
```

Vous devez maintenant analyser la chaîne "36 pt", l'un des inconvénients que XML était censé éviter. L'ajout d'un attribut à l'élément size est bien plus propre :

```
<font>
<name>Helvetica</name>
<size unit="pt">36</size>
</font>
```

Une règle souvent utilisée consiste à n'utiliser les attributs que lorsqu'ils modifient l'interprétation d'une valeur, et non pour spécifier des valeurs. Si vous vous retrouvez engagé dans des discussions métaphysiques sur le fait qu'un paramètre particulier soit une modification de l'interprétation d'une valeur ou non, arrêtez tout et refusez les attributs pour utiliser plutôt des éléments tout du long.

De nombreuses très bonnes DTD n'utilisent pas du tout d'attributs.

Les éléments et le texte sont l'essence des documents XML. Il existe quelques instructions de marquage que vous pouvez rencontrer.

2- Structure d'un document XML[15]

Un document XML doit commencer par un en-tête tel que :

```
<?xml version="1.0"?>
ou
<?xml version="1.0" encoding="UTF-8"?>
```

A strictement parler, l'en-tête est optionnel mais fortement recommandé.L'en-tête peut être suivi d'une *définition du type de document*, comme suit :

```
<!DOCTYPE Web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application
2.2//EN"
"http://java.sun.com/j2ee/dtds/Web-app_2_2.dtd">
```

Les définitions du type de document constituent un mécanisme important pour s'assurer de l'exactitude d'un document, mais ne sont pas obligatoires. Enfin, le corps du document XML comprend *l'élément racine*, qui peut contenir d'autres éléments.
Par exemple :

```
<?xml version="1.0"?>
<!DOCTYPE configuration . . .>
<configuration>
<title>
<font>
<name>Helvetica</name>
<size>36</size>
</font>
</title>
. . .
</configuration>
```

Un élément peut contenir des *éléments enfant*, du texte, voire les deux. Dans l'exemple qui précède, les éléments font possèdent deux éléments enfant, name et size. L'élément name contient le texte :"Helvetica".

Annexe C

XML

eXtensible Markup Language

1-Présentation de XML :

XML pour eXtensible Markup Language, est un langage informatique de balisage générique. Le World Wide Web Consortium (W3C) recommande XML pour exprimer des langages de balisages spécifiques (exemples : XHTML, SVG, XSLT)**[14]**.

Son objectif initial est de faciliter l'échange automatisé de contenus entre systèmes d'informations hétérogènes (interopérabilité), notamment sur Internet. XML est un sous-ensemble de SGML dontil retient plusieurs principes comme :

• la structure d'un document XML est définissable et validable par un schéma.

• un document XML est entièrement transformable dans un autre document XML.

Le format XML vous permet d'exprimer la structure de manière hiérarchique, ainsi que des éléments apparaissant à plusieurs reprises, sans grandes difficultés.
Comme vous pouvez le voir, le format d'un fichier XML est simple. Il ressemble d'ailleurs à un fichier HTML. Il existe une bonne raison à cela, les formats XML et HTML sont des descendants du grand SGML (*Standard Generalized Markup Language*).
SGML existe depuis les années 1970 et permet de décrire la structure de documents complexes. Il est utilisé avec un certain succès sur quelques marchés nécessitant de conserver en continu de grandes masses de documents, particulièrement sur le marché de l'aéronautique. Mais le SGML est assez complexe et n'a donc jamais été amené au premier plan. Une grande partie de cette complexité venaît du fait que le SGML affiche deux objectifs contradictoires. Il veut s'assurer que les documents sont constitués conformément aux règles de leur type de document. Et il veut également faciliter la saisie de données, en autorisant des raccourcis.
XML a en fait été conçu comme une version simplifiée du SGML, à utiliser sur Internet. Comme souvent, le plus simple est le mieux et le XML a immédiatement reçu un accueil enthousiaste et a depuis longtemps supplanté le SGML**[15]**.

BIBLIOGRAPHIE

[3]	Sécurité Optimale. Ressource d'expert. Livre anonyme. Edition S&SM, 1998
[4]	Claude SERVIN,Réseaux et Télécommunications. Cours et exercices corrigés. Edition Dunod, 2003
[5]	I.J. Cox, M.L. Miller, J.A. Bloom, J. Fridrich et T. Kalker « Digital Watermarking and Steganography », Morgan Kaufmann, 2007.
[6]	B. Barni et F. Bartolini, « Watermarking Systems Engineering : Enabling Digital Assets Security and other Applications », Marcel Dekker, 2004.
[7]	I.J. Cox, M.L. Miller et J.A. Bloom, « Digital Watermarking », Morgan Kaufmann, 1999.
[8]	Pierre-Alain Muller ,NathalieGaerthner Modélisation objet avec UML EYROLLES ,2002
[9]	Apache professionnel Peter WAINWRIGHT Edition Eyrolles 2000
[10]	MYSQL ,le serveur SQL de base de données multi API Paul DUBOIS Edition CompusPress
[12]	JasperReports for java Developers David R.Heffelfinger BIRMINHGHAM –MUMBAI 2006
[14]	Creation d'un outil de reporting avancé Manex TFE realiser par Philipe Iodomez Groupe 23 27-Haute Ecole Rennequin Sualem 2006-2007
[15]	Au cœur de java 2 Cay S.Horstmann Gary Cornell CampusPress 2005
[16]	Tutoriel DOM et JDOM Cyril Vidal 2001

WEBOGRAPHIE

[1]	http://securite.developper.com/fac/?page=dispo (Access date: 21 MARS, 2011)
[2]	http://www.hsc.fr/ressources/presentations/pki/img14.htm (Access date : 26 Mars, 2011) PKI et certificats Présentation de G.Labouret. 1999
[11]	http://en.wikipedia.org/wiki/javaPlatform ,Entreprise Edition Année :August 2009,(Access date :23/01/2011)
[13]	http://www.developpez.com (Access date :31/08/2011)