**Haute école spécialisée bernoise**
**Technique et informatique**

Thèse de Bachelor 2008

# E-voting

Système électronique de vote par Internet


| | |
|---|---|
| Section | Informatique |
| Étudiants | Molina Pablo |
| | Pietronigro Marc |
| Professeurs | Dr. Eric Dubuis |
| | Gerhard Hassenstein |

## Sommaire

Introduction ..... 4
Autres nécessités et contraintes ..... 4
Acteurs ..... 5
Use Cases ..... 6
Use Case UC1 : Génération des codes d'accès ..... 7
Main Success Scenario ..... 7
Extension ..... 7
Use Case UC2 : Obtention des codes d'accès ..... 8
Main Success Scenario ..... 8
Extension ..... 8
Use Case UC3 : Identification, vote ..... 9
Main Success Scenario ..... 9
Extension ..... 9
Use Case UC4 : Permission ..... 10
Main Success Scenario ..... 10
Extension ..... 10
Use Case UC5 : Confirmation ..... 11
Main Success Scenario ..... 11
Evénements système ..... 12
Génération des codes ..... 12
Identification ..... 12
Permission ..... 12
Votation ..... 12
SSD ..... 13
Génération des codes ..... 13
Identification ..... 13
Permission ..... 14
Votation ..... 15
Cas générique ..... 16
Fonctionnement général (étape par étape) ..... 16
Perte des codes d'accès ..... 16
Domain model ..... 17
Concepts ..... 18
Associations ..... 19
Planning ..... 20
RC1 ..... 20
RC2 : ..... 21
Architecture du projet ..... 23
Description des layers ..... 24
Diagramme de communication ..... 24
Génération - AccessGenerator ..... 25
1. Recherche de la liste des personnes, génération et envoie d'e-mail. ..... 25
Point de vue logique ..... 26
Client - AccessInfo ..... 31
2. Récupération des codes d'accès ..... 31
Point de vue logique ..... 31
Serveur - AccessInfo ..... 34
3. Authentification et distribution des codes ..... 34
Point de vue logique ..... 34

Client - l'AccessInfo ..... 37
4. Contrôle ..... 37
Point de vue logique ..... 37
Client ClientVoteModule ..... 40
5. Vote ..... 40
Point de vue logique ..... 40
Serveur VoteServer ..... 43
6 Traitement du vote ..... 43
Point de vue logique ..... 43
Système ..... 48
Generator ..... 49
Client module ..... 50
Web-Service ..... 51
Communication ..... 52
Firewall ..... 53
Gestion des clefs ..... 54
Protocoles ..... 55
Audit ..... 56
Audit : Ordinateur citoyen ..... 57
Système de démonstration (partie codé durant la thèse) ..... 57
Système final (proposition final) ..... 58
Audit : Accès physique des serveurs ..... 59
Système de démonstration (partie codé durant la thèse) ..... 59
Système final (proposition final) ..... 59
Audit des canaux de communication ..... 60
Système de démonstration (partie codé durant la thèse) ..... 60
Système final (proposition final) ..... 60
Audit des serveurs ..... 61
Système de démonstration (partie codé durant la thèse) ..... 61
Système final (proposition final) ..... 61
Audit : Social engineering ..... 61
Etude des protocoles de sécurités choisis ..... 62
Marche à suivre du client ..... 63
Protection du système client ..... 63
Télécharger l'applet ..... 63
Récupérer les informations pour voter ..... 63
Sauvegarder les informations ..... 63
Voter ..... 63
Vérifier un certificat ..... 64
Choses à ne pas faire ..... 64
Marche à suivre de l'école ..... 65
Génération des codes ..... 65
Informations sur les serveurs ..... 65
Information utilisateurs ..... 65
Conclusion ..... 66
Remerciements ..... 66
Spécification supplémentaires ..... 67
Glossaire ..... 68
Sources ..... 70
Annexes Sommaires ..... 72

**Abstract** Les systèmes de votes par Internet réalisés à l'heure actuel rivalisent tous d'ingéniosité. Ceci prouve la difficulté et les possibilités diverses qu'impliquent un système de E-voting. Ce rapport présente une version créée spécialement pour les écoles spécialisés suisses, cette version allie la facilité de mise en place, d'administration avec une sécurité solide et un coût d'exploitation raisonnable. Ce document présente et explique les choix réalisés pour notre système, ainsi que ces faiblesses et points forts.

**Mot clés :** Votation, E-voting, Internet, Sécurisé, Signature, Encryption, XML, Web service

# Introduction

Les systèmes de votes électroniques commencent gentiment à apparaître dans la vie des citoyens. De plus en plus de cantons (niveau Suisse) et pays (niveau international) créent et testent des systèmes d'E-voting. Les études et résultats de ces tests prouvent que le taux de participation augmente grâce à la votation via Internet et que les gens qui ont été sondés, disent apprécier ce nouveau système.
C'est donc dans ce contexte de développement d'une nouvelle technologie de vote que notre thèse de bachelor commence. Nous avons réalisé un système pouvant servir de base pour une implémentation future.

Pour l'école d'ingénieur de Bienne, nous avons créé un système de votation en ligne (*E-voting*) présentant une sécurité optimale afin de pouvoir voter en toute sécurité.
Ce projet est réalisé dans le cadre de notre thèse de bachelor. Le but de cette thèse est de développer des connaissances dans le domaine des votations en ligne, connaissances servant à certains projets de l'école ainsi qu'à notre enrichissement personnel.
Notre projet est *Open Source* afin que la sécurité puisse être correctement testée et évaluée. Le travail a été donné à deux groupes distincts qui produisent deux implémentations différentes du système.

Notre projet consiste donc à réaliser un système de vote sécurisé *end-to-end* n'autorisant aucune manipulation avant, pendant et après la période de votation.

## Autres nécessités et contraintes

Le système doit prendre en compte les coûts en temps et en matériel d'une école, ce qui limite les choix des sécurités à employer. Il doit être le plus *user friendly* possible et portable sous toutes les plates formes.
Les onze *statements* à respecter

1. Les *suffrages* exprimés électroniquement ne doivent pas pouvoir être interceptés, modifiés ou détournés.
2. Le contenu des *suffrages* exprimé électroniquement ne doit pas pouvoir être connu par des tiers avant le *dépouillement*.
3. Seules les personnes ayant le droit de vote doivent pouvoir prendre part au *scrutin*.
4. Chaque électeur ne dispose que d'une voix et ne peut voter qu'une seule fois.

5. En aucun cas, même pendant le *dépouillement*, il ne doit être possible de faire un lien entre un électeur et son *suffrage*.
6. Le site doit être en mesure de résister à une *attaque en déni de service* pouvant aboutir à la saturation du serveur.
7. L'électeur doit être protégé contre toute tentative de vol d'identité.
8. Le nombre de votes émis doit correspondre au nombre de votes reçus, toute différence doit pouvoir être expliquée et corrigée.
9. La preuve qu'un électeur a voté doit pouvoir être faite.
10. Le système n'accepte pas de vote électronique en dehors de la période d'ouverture du *scrutin* électronique.
11. Le bon fonctionnement du système doit pouvoir être vérifié par les autorités désignées à cet effet.

## Acteurs

| | |
|---|---|
| HTI : | Elle génère les codes d'accès et les distribuent. |
| Serveur WEB : | Interface qui lie l'utilisateur à notre système. |
| Urne électronique : | Base de données où sont stockés tous les bulletins de vote cryptés. |
| Registre des électeurs : | Base de données contenant toutes les données personnelles des votants. |
| Registre de contrôle : | Base de données contenant les *ID Unique* des personnes ayant droit de vote. |
| Votant : | Personne ayant le droit de vote |

## Use Cases

**Génération des codes d'accès :**
Permet à l'école de créer à l'aide du *serveur LDAP* les codes qui seront distribués aux votants potentiels

**Obtention des codes d'accès :**
Permet aux votants d'obtenir les *codes d'accès* nécessaire à s'identifier sur notre système.

**Identification :**
Le votant, à l'aide des *codes d'accès* s'identifie sur notre système.

**Permission :**
Fonctionnement lors de l'identification afin de savoir si la personne a le droit de vote.

**Vote :**
Phase de vote point de vue client et serveur

**Confirmation :**
Confirmation que la personne a voté.

## Use Case UC1 : Génération des codes d'accès

Primary Actor : HTI
Precondition : Période de pré-votation, créé depuis un ordinateur se trouvant sur le réseau
Postcondition : Les *codes d'accès* pour les votants sont générés correctement et stockés sur les serveurs (Registre de contrôle + serveur de récupération des codes).

### *Main Success Scenario*

1. Les responsables des votes de l'école lancent le programme de génération.
2. Les informations sur les votants sont récupérées (*LDAP*).
3. Les nombres aléatoires servant de *PIN*, *ID Unique et* l'*ID de référence* sont générés.
4. La base de données de contrôle est créée sur la base de ces nombres stockés sous forme de *hash value*.
5. La base de données de récupération est créée sur la base de ces nombres encryptés.

### *Extension*

1. Les serveurs de bases de données sont indisponibles :
   a. La génération des codes est stoppée.

## Use Case UC2 : Obtention des codes d'accès

Primary Actor : Votant
Precondition : Email d'information reçu, période de votation.
Postcondition : Possession des codes d'accès

### *Main Success Scenario*

1. Le votant lit l'email d'information.
2. Le votant se connecte sur notre serveur et récupère l'applet.
3. Le votant rentre son *login* et *mot de passe*.
4. Le votant obtient les *codes d'accès.*

### *Extension*

1. Mauvais *login*/*mot de passe* :
   a. Le votant est invité après un certain délai à rentrer à nouveau des données valides.
2. L'utilisateur ferme l'application sans avoir pris note des codes :
   a. Il ne pourra plus voter pour cette session de vote.

## Use Case UC3 : Identification, vote

Primary Actor : Votant
Precondition : Il a reçu les *codes d'accès*, période de vote ouverte
Postcondition : Le votant est identifié dans le système

### *Main Success Scenario*

1. Le votant se connecte via un web browser sur l'adresse du site qu'il connaît.
2. Il entre ses *codes d'accès*, une information personnelle (qu'il connaît) et remplit son bulletin de vote.
3. Il envoie le tout.

### *Extension*

I. A tout moment, le browser peut être fermé et le processus d'identification est annulé.
1. L'adresse URL est correcte mais le laps de temps disponible pour voter est dépassé.
   a. Une page d'erreur est affichée.
2. Le votant inscrit mal son numéro *ID Unique*. Le système lui propose deux fois d'insérer son numéro *ID Unique* (trois tentatives).
   a. Après la 3ème erreur, le système se bloque et il est impossible de voter depuis cette machine pendant 30minutes.

## Use Case UC4 : Permission

Primary Actor : Le votant
Precondition : L'utilisateur est identifié dans le système
Postcondition : L'utilisateur sait ce qu'il peut faire sur le site de vote.

### *Main Success Scenario*

1. Le système contrôle les *codes d'accès*.
2. Il contrôle s'ils ont déjà été utilisés.
3. Le vote est sauvegardé.

### *Extension*

1. L'utilisateur a déjà voté :
   a. Voir Use Case *Confirmation* (page 8).
2. Les codes d'accès ne sont pas corrects :
   a. Le vote n'est pas sauvegardé et une erreur est retournée

## Use Case UC5 : Confirmation

Primary Actor : Votant, HTI
Precondition : Le votant a voté
Postcondition : Une confirmation est affichée

### *Main Success Scenario*

1. L'heure, la date et le moyen de vote sont affichés

## Evénements système

### Génération des codes

generate()

### Identification

getInfo(String login, String mdp)

### Permission

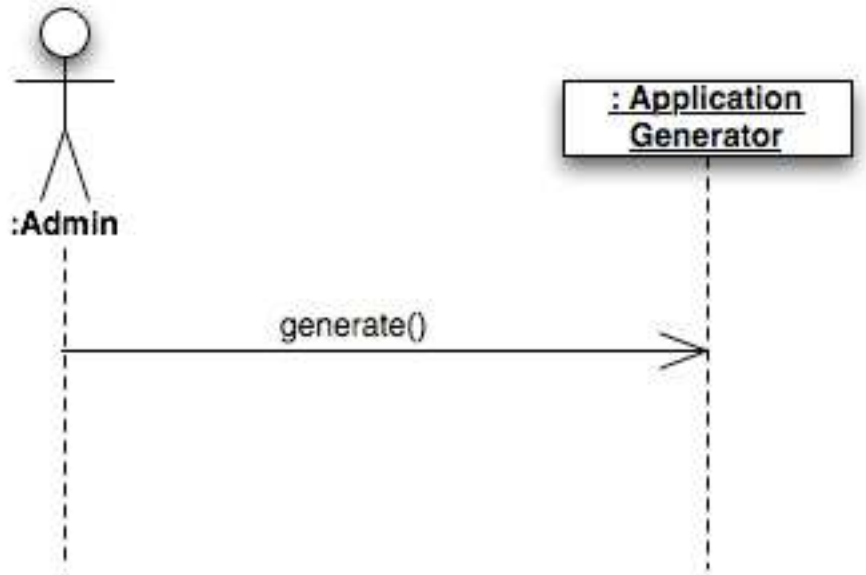
isIdcorrect(String Id)
asIdVoted(String Id)
isPinCorrect(String Pin, String InfoPerso)

### Votation

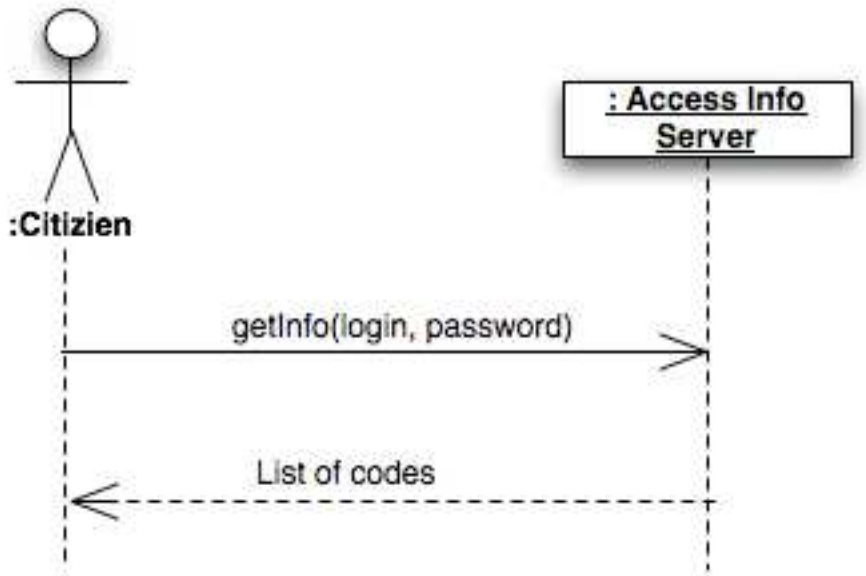
vote(String Pin, String Bulletin, String InfoPerso, String Id, String idRef)

## SSD

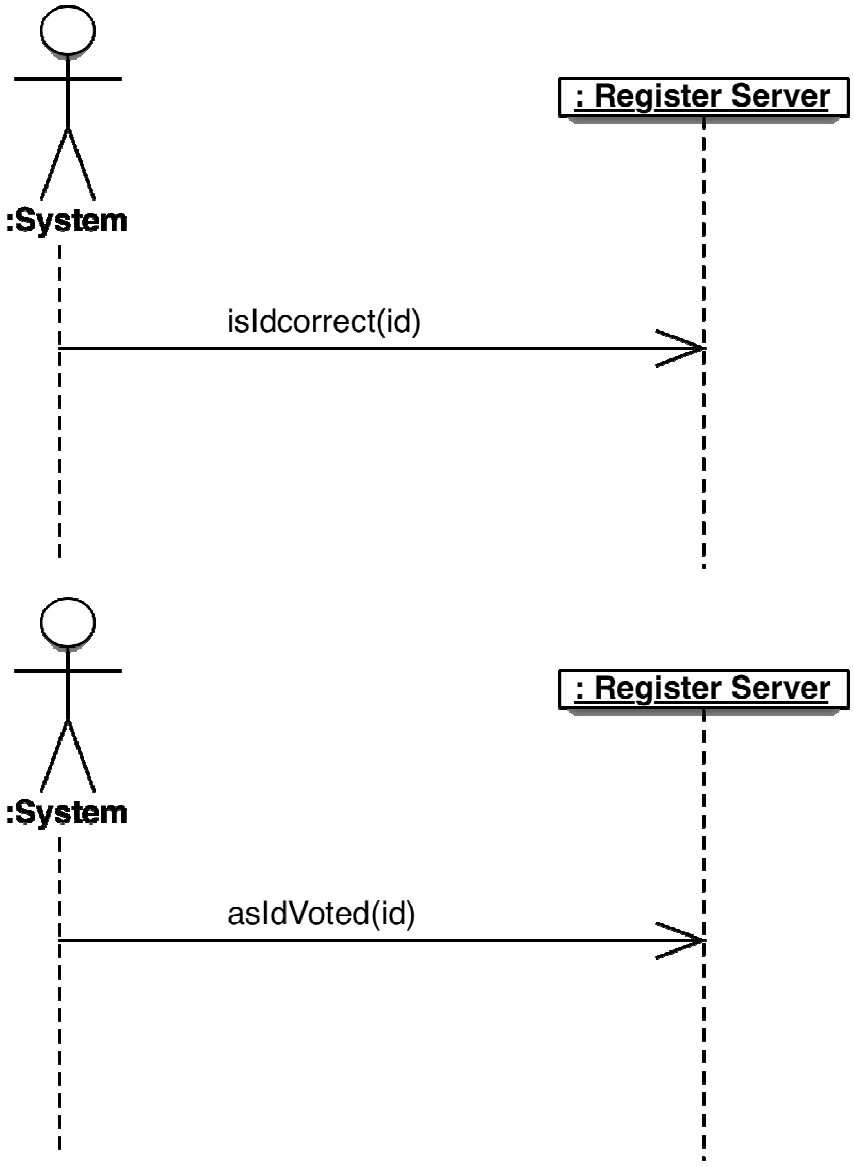
### Génération des codes

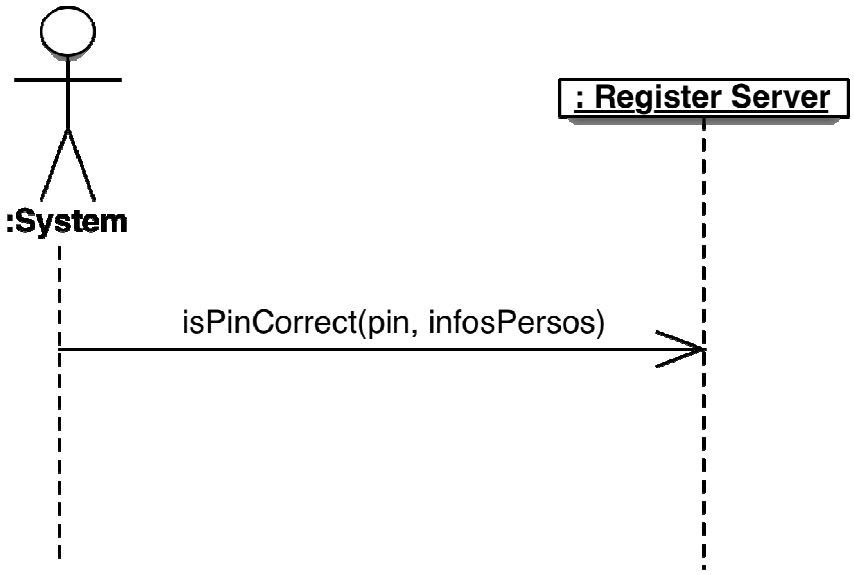



### Identification

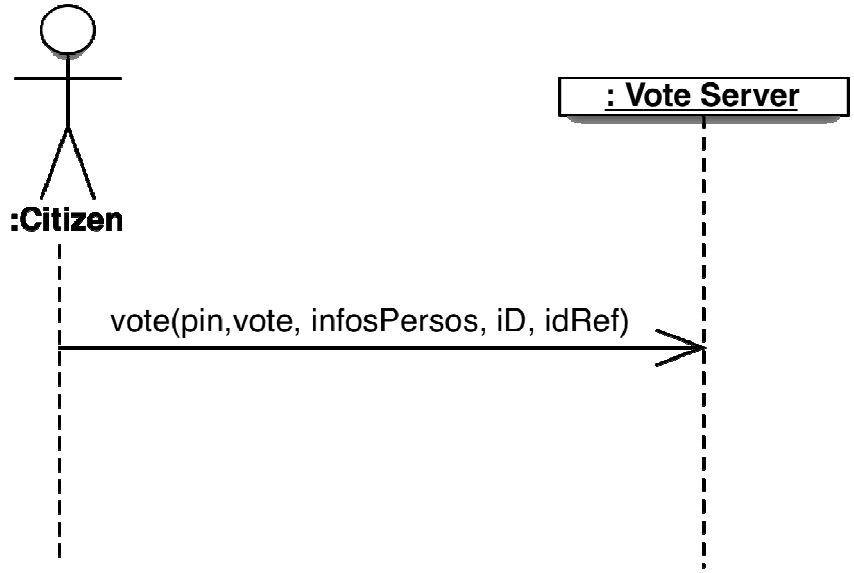


**Permission**


: Register Server
:System
isIdcorrect(id)
: Register Server
:System
asIdVoted(id)


:System
: Register Server
isPinCorrect(pin, infosPersos)

**Votation**


:Citizen
: Vote Server
vote(pin,vote, infosPersos, iD, idRef)

# Cas générique

## Fonctionnement général (étape par étape)

**Le système génère les codes :**

1. L'école génère les *codes d'accès*.
2. L'école génère la base de données de contrôle (Registre de vote) en même temps.
3. L'école génère la base de données de récupération des codes.

**Récupération des codes d'accès :**

4. Le votant ouvre sa boîte mail et voit que la période de vote est ouverte.
5. Le votant télécharge l'applet de récupération des *codes d'accès*, se *log* avec son mot de passe et *login* de l'école et récupère les codes d'accès.

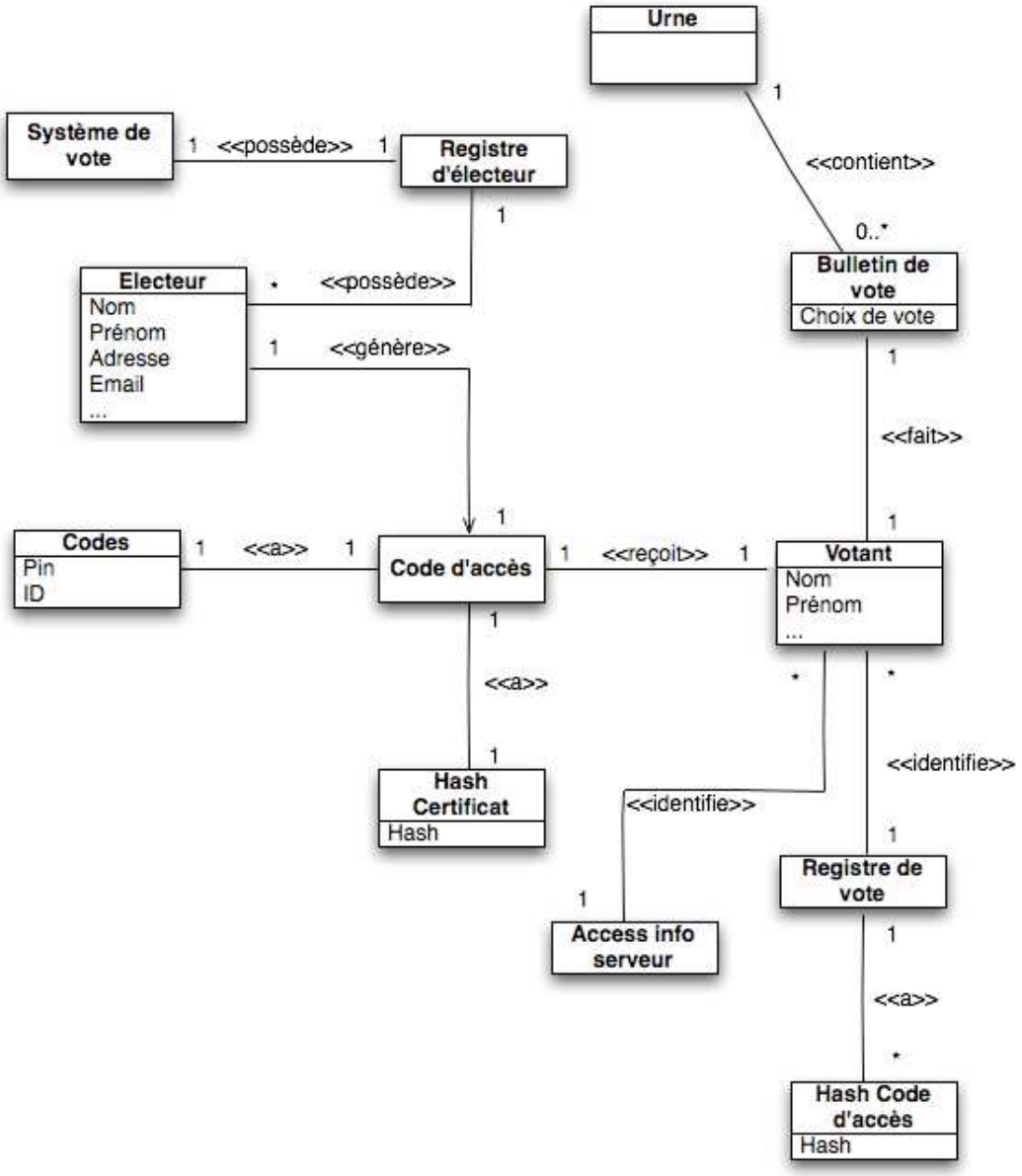
**Phase de vote :**

1. Le votant se connecte au serveur de vote.
2. Il vérifie le certificat à l'aide du *fingerprint* des codes d'accès.
3. Il remplit le bulletin de vote.
4. Il remplit les champs d'identification grâce aux *codes d'accès*
5. Il envoi le tout.
6. Le votant voit le message : « Votre vote a bien été pris en compte ».

## Perte des codes d'accès

Implique que ce votant n'aura pas le droit de vote pour cette session.

## Domain model



## Concepts

**Système de vote**
Application servant de générateur et d'interface de communication entre le système de vote et les utilisateurs.

**Registre d'électeur**
Registre contenant les informations personnelles sur les votants. Ce registre sert de base pour la création des codes d'accès.

**Electeur**
Informations personnelles sur les votants.

**Registre de vote**
Registre contenant les informations servant à l'identification des votants. Aucune donnée personnelle n'y figure, uniquement des *hashs value*.

**Access Info serveur**
Registre contenant les informations servant à la récupération des codes d'accès. Toutes les données sont cryptées.

**Hash code d'accès**
Serveur contenant les *hashs value* des *codes d'accès* qui servent de base pour l'identification.

**Codes**
*Codes d'accès* du votant. *ID Unique*, *PIN* et *ID de référence*.

**Hash Certificat**
Contient la *hash value* du certificat du serveur de vote afin de permettre au votant de vérifier l'identité du serveur.

**Votant**
Personne ayant le droit de vote.

**Bulletin de vote**
Bulletin crypté contenant les choix de vote du votant.

**Urne**
Base de données servant à stocker les bulletins de vote.

## Associations

**Système de vote – Registre d'électeur**
Le système de vote possède un registre d'électeur.

**Registre d'électeur – Electeur**
Le registre d'électeur contient plusieurs électeurs.

**Electeur – Code d'accès**
Les informations des électeurs servent de base pour générer les codes d'accès.

**Code d'accès – Codes**
Chaque *code d'accès* possède un ensemble de codes (*PIN* + *ID unique*, …)

**Code d'accès – Hash Certificat**
Chaque code d'accès possède la *hash value* du certificat du serveur de vote.

**Votant – Code d'accès**
Chaque votant possède un ensemble de *code d'accès*.

**Votant – Access Info serveur**
Chaque votant s'identifie vers *l'access info* serveur.

**Votant – Registre de vote**
Chaque votant s'identifie vers le *registre de vote*.

**Registre de vote – hash code d'accès**
Le registre de vote utilise les hash des codes d'accès pour vérifier l'identité du votant.

**Votant – Bulletin de vote**
Chaque Votant fait un bulletin de vote.

**Urne – Bulletin de vote**
L'urne contient tous les bulletins de vote réalisés.

## Planning

Le projet *E-voting* étant très conséquent et n'ayant pas assez de temps pour tout développer de façon complète, nous avons décidé de couper le travail en deux parties. La première partie (RC1) sera celle réalisée durant le temps imparti pour la thèse. Elle consistera à réaliser le système de vote dans son ensemble de façon à pouvoir voter, la sécurité ne sera pas implémentée dans sa totalité. La seconde partie, que nous n'aurons pas le temps de réaliser, sera simplement documentée et comprendra tout ce qu'il reste à faire afin d'avoir un système de *E-voting* complet et sûr.

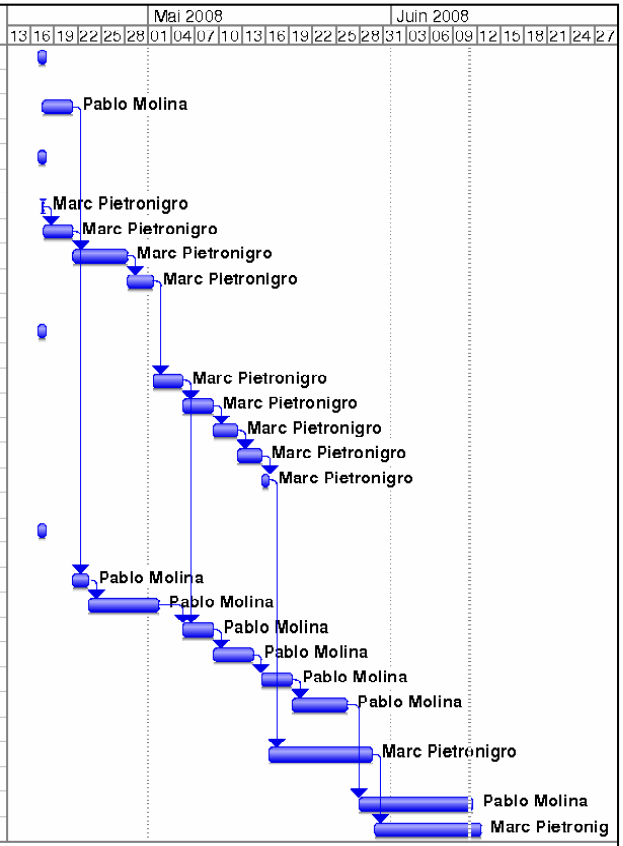
### RC1

- Génération des clefs :
  - Soft de génération des bases de données.
  - Urne doit être créée et opérationnelle.
  - Registre doit être créé et opérationnel.
  - Le serveur de récupération des codes doit être opérationnel.
- Client :
  - L'application doit communiquer avec le Web Service (échange de message SOAP).
  - L'application doit pouvoir créer des documents XML.
  - Elle doit encrypter asymétriquement les documents XML. (clé public dans l'application).
  - Interpréter l'XML. (le message de réponse du web server)
  - Interface de vote (prototype).
- Web Service
  - Communiquer avec le client. (SOAP).
  - Communiquer avec les bases de données.
  - Interpréter l'XML.
  - Gérer les signatures et méthodes d'encryptions.
  - Permettre une démonstration de vote.
- Base de données
  - Base de données doivent être installées.
- Documentation et test

**RC2 :**

- Génération des clefs :
  - Créer un tunnel sécurisé entre le serveur de génération et les deux serveurs de stockages.
  - Brasser (mélanger) le registre des électeurs, afin de garantir la *pseudonymie* (éviter que l'ordre alphabétique du serveur *LDAP* ne corresponde aux entrées dans la base de données).
  - Récupération des informations sur les citoyens via une base de données *LDAP*.
  - Récupération des clefs publiques via *XKMS* (enlever les clefs *hardcode*).
  - Utilisation de clefs privées sécurisées (clef stocker dans des modules hardware).
  - Rajout de la *hash value* du certificat *HTTPS* du serveur de vote comme code d'accès.
  - Utilisation de certificat signé par des instances connues et trust (*Verisign*, …).
- Client :
  - Transformation de l'application java en *applet*.
  - Création d'un tunnel sécurisé entre l'applet client et le serveur.
  - Récupération des clefs publiques via *XKMS*.
  - Amélioration de l'interface graphique.
- Web Service :
  - Communication *HTTPS* pour la récupération des *applets*.
  - Communiquer avec *XKMS* pour la récupération des clefs publiques utilisées lors des différentes communications.
  - Configuration des différents fichiers *Policy* (*web – service Policy*)
  - L'application doit gérer les signatures.
- Base de données :
  - Brassage (mélange) des bases de données afin d'éviter toute comparaison entre l'heure de vote et l'ordre de stockage.
- Documentation et test

| N° | Nom de la tâche | Durée | Début | Fin | Prédécesseurs |
|---|---|---|---|---|---|
| 1 | **Configuration des serveurs** | 1 jour? | Jeu 17.04.08 | Jeu 17.04.08 | |
| 2 | | | | | |
| 3 | installer configuer les db | 4 hr | Jeu 17.04.08 | Lun 21.04.08 | |
| 4 | | | | | |
| 5 | **Coder Génération des mot de passes** | 1 jour? | Jeu 17.04.08 | Jeu 17.04.08 | |
| 6 | | | | | |
| 7 | Générer codes aléatoires unique | 3 hr | Jeu 17.04.08 | Jeu 17.04.08 | |
| 8 | Créer et coder le serveur de récupération des codes | 5 hr | Jeu 17.04.08 | Lun 21.04.08 | 7 |
| 9 | Créer et coder le registre d'électeur | 12 hr | Lun 21.04.08 | Lun 28.04.08 | 8 |
| 10 | Signer les informations | 8 hr | Lun 28.04.08 | Jeu 01.05.08 | 9 |
| 11 | | | | | |
| 12 | **Coder application client** | 1 jour? | Jeu 17.04.08 | Jeu 17.04.08 | |
| 13 | | | | | |
| 14 | Communication avec le WebServer | 6 hr | Jeu 01.05.08 | Lun 05.05.08 | 10 |
| 15 | Génération des documents XML | 8 hr | Lun 05.05.08 | Ven 09.05.08 | 14 |
| 16 | Encrypter le document XML -> symétrique (hard codée) | 4 hr | Ven 09.05.08 | Lun 12.05.08 | 15 |
| 17 | Interpreter l'XML (réponse) | 5 hr | Lun 12.05.08 | Jeu 15.05.08 | 16 |
| 18 | Interface de vote | 5 hr | Jeu 15.05.08 | Ven 16.05.08 | 17 |
| 19 | | | | | |
| 20 | **coder WebService** | 1 jour? | Jeu 17.04.08 | Jeu 17.04.08 | |
| 21 | | | | | |
| 22 | Etablir la connexion WebService - Base de données | 4 hr | Lun 21.04.08 | Mer 23.04.08 | 3 |
| 23 | Créer et coder le vote serveur et l'urne | 13 hr | Mer 23.04.08 | Ven 02.05.08 | 22 |
| 24 | Communiquer avec le client | 5 hr | Lun 05.05.08 | Ven 09.05.08 | 14;23 |
| 25 | Interpetter l'XML | 8 hr | Ven 09.05.08 | Mer 14.05.08 | 24 |
| 26 | Configurer les policies(minimal) | 5 hr | Jeu 15.05.08 | Lun 19.05.08 | 25 |
| 27 | Gestion signature, encryption et décryption | 11 hr | Lun 19.05.08 | Lun 26.05.08 | 26 |
| 28 | | | | | |
| 29 | **Test** | 23 hr | Ven 16.05.08 | Jeu 29.05.08 | 18 |
| 30 | | | | | |
| 31 | **Documentation** | 22 hr | Mer 28.05.08 | Mer 11.06.08 | 27 |
| 32 | Documentation | 22 hr | Ven 30.05.08 | Jeu 12.06.08 | 29 |



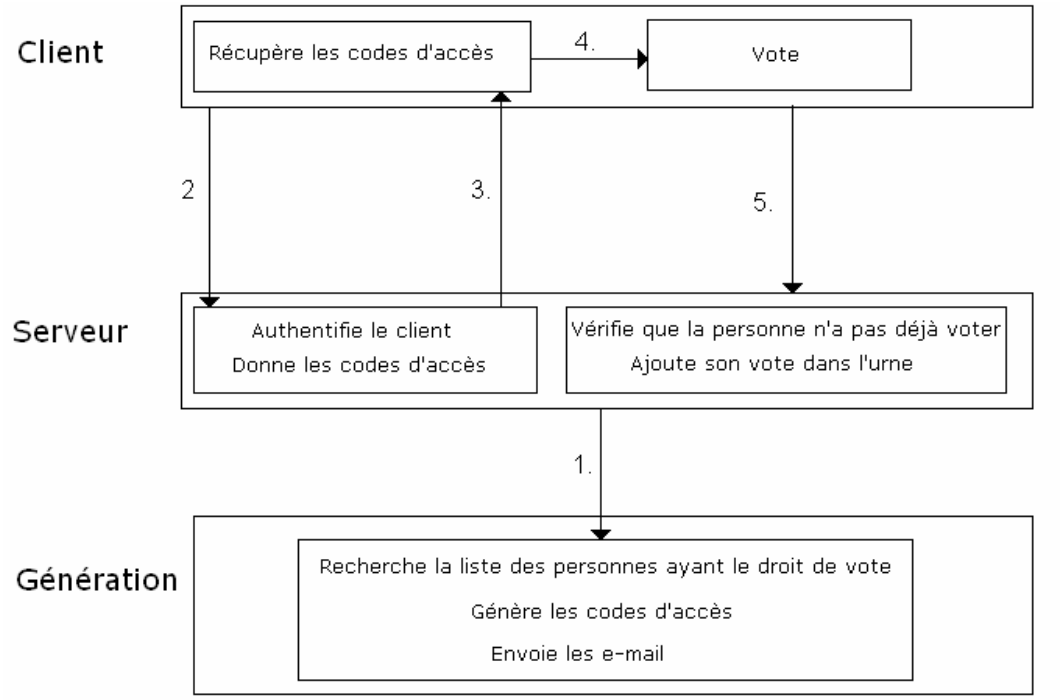

Projet : RC1
Date : Mer 11.06.08

Tâche — Jalon — Tâches externes
Fractionnement — Récapitulative — Jalons externes
Avancement — Récapitulatif du projet — Échéance

Page 1

*Planning de notre RC1*

## Architecture du projet




Ce schéma représente globalement le fonctionnement de notre système. On y voit les trois parties importantes qui sont : le client, le serveur et la génération ainsi que les actions principales qui s'y produisent.

## Description des layers

### *Diagramme de communication*

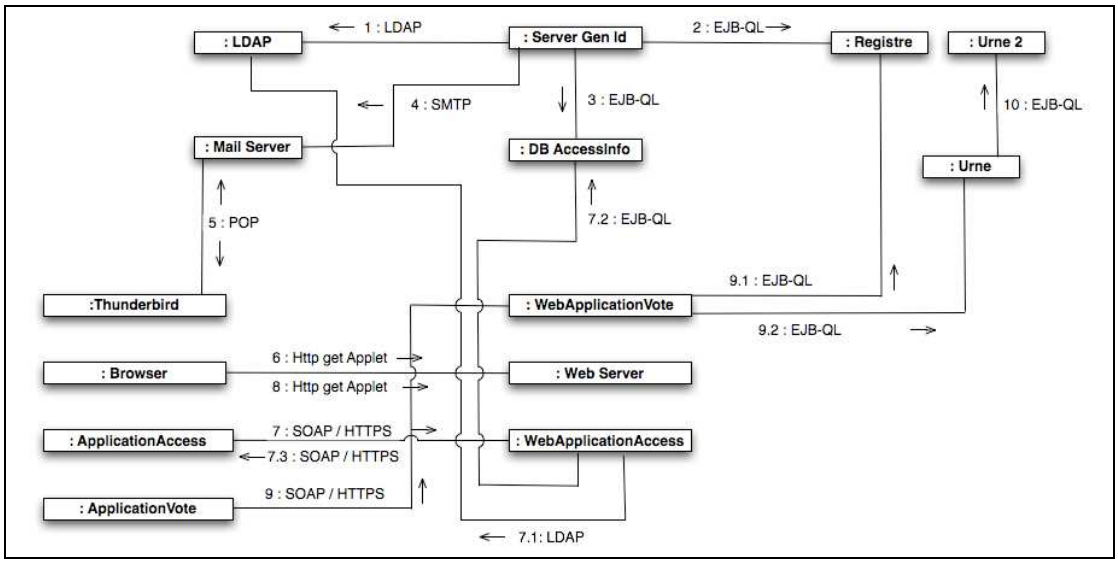



*Diagramme de communication du fonctionnement global*

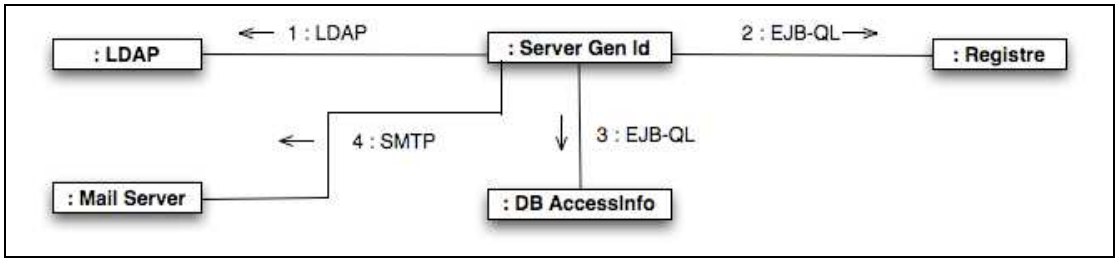



*Diagramme de communication de la génération*

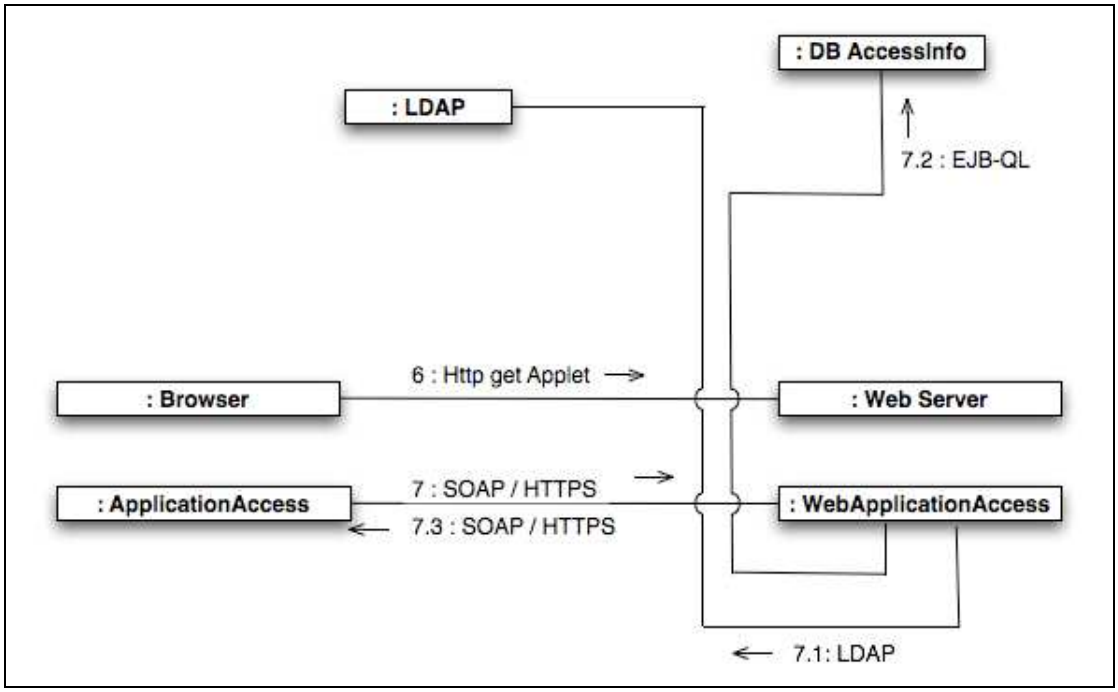



*Diagramme de communication de la récupération des informations*

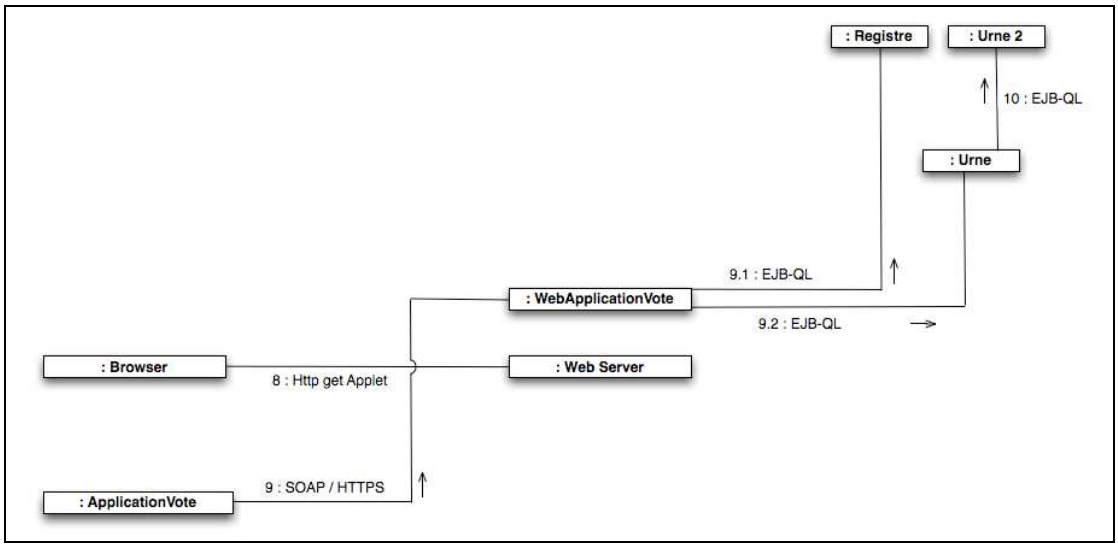



*Diagramme de communication du Vote*

### *Génération - AccessGenerator*

La génération est composée de la classe *AccessGenerator*, elle s'appuie sur d'autres classes afin de traiter les informations XML (*Utils)* et la recherche d'information (*getInfo*).

**1. Recherche de la liste des personnes, génération et envoie d'e-mail.**

La liste des personnes se trouve actuellement dans le code. La liaison avec le serveur *LDAP* n'a pas été implémentée (se reporter à la section *Système* de notre document).
La génération se fait sur la base des informations récupérées. Par la suite, un e-mail est envoyé à chaque client ayant les droits de vote pour les informer qu'un vote est ouvert.

***Point de vue logique***

**Classes, interfaces, relations**

AccessGenerator :

Récupère les informations via la méthode *getInfo()*. Insère ces informations dans la base de données *Info* puis envoie l'email à l'utilisateur (la fonction d'envoi d'email n'a pas été implémentée, se reporter à la section *Système* de notre document).

GetInfo :

Interface du *Factory pattern* qui définie la méthode de recherche des personnes.

GetInfoFactory :

*Factory pattern*

GetInfoHardcode :

Classe du *Factory pattern*. C'est la méthode utilisée dans notre projet, elle contient des informations sur les personnes de façon *hardcoder*.

GetInfoLdap :

Classe prévue pour la recherche des personnes via le serveur, elle n'a pas été implémentée (se reporter à la section *Système* de notre document).

AccessInfo :

Web service appelé lors de la sauvegarde des informations dans la base de données *Info.*

Register :

Web service appelé lors de la sauvegarde des informations dans la base de données.

Utils :

Classe utilitaire contenant les méthodes pour traiter les informations XML.

**Scénario**

Lorsque l'école veut lancer une votation, après avoir démarré les web services, elle exécute notre fichier client *AccessGenerator* qui s'occupera de générer les informations et de les enregistrer dans les bases de données.

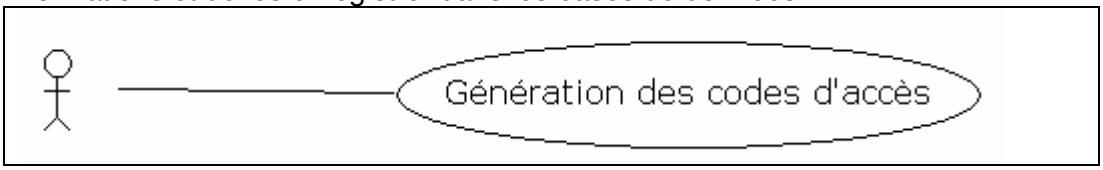



*Diagramme de fonctionnement de la génération des codes.*

**Structure de la base de données Info**

| ID | PK_PIN | MD5_LOGIN | PK_ID | IDREF |
|---|---|---|---|---|
| 52 | <?xml versi... | <?xml versi... | <?xml versi... | <?xml versi... |
| 53 | <?xml versi... | <?xml versi... | <?xml versi... | <?xml versi... |
| 54 | <?xml versi... | <?xml versi... | <?xml versi... | <?xml versi... |
| 56 | <?xml versi... | <?xml versi... | <?xml versi... | <?xml versi... |

*Notre clé primaire est le champ ID, qui lui ne contient pas de XML.*

ID :

Clé primaire de notre base de données.

PK_PIN :

Contient le PIN unique signé, encrypté puis resigné. Ce champ est signé avec la clé privée du *Generator* et encrypté avec une clef de session, qui elle est encryptée avec la clé publique de l'*AccessInfo*.

MD5_login :

Contient le login en *hash value*. Ce champ est signé avec la clé privée de l'*Generator*.

PK_ID :

Contient l'*ID unique* signé, encrypté puis résigné. Ce champ est signé avec la clé privée du *Generator* et encrypté avec une clef de session, qui elle est encryptée avec la clé publique del'*AccessInfo*.

IDREF :

Contient l'*ID de référence* signé, encrypté puis résigné. Ce champ est signé avec la clé privée du *Generator* et encrypté avec une clef de session, qui elle est encryptée avec la clé publique dd l'AccessInfo.

**Structure de la base de données RegisterEntry**

| ID | PERSONALINFO | IDNUMBER | HASVOTED | PIN | REFERENCENUMBER |
|---|---|---|---|---|---|
| 2 | <?xml version="1.0... | <?xml versi... | 0 | <?xml versi... | <?xml version="1.0" ... |
| 3 | <?xml version="1.0... | <?xml versi... | 1 | <?xml versi... | <?xml version="1.0" ... |
| 4 | <?xml version="1.0... | <?xml versi... | 0 | <?xml versi... | <?xml version="1.0" ... |
| 5 | <?xml version="1.0... | <?xml versi... | 0 | <?xml versi... | <?xml version="1.0" ... |
| 6 | <?xml version="1.0... | <?xml versi... | 0 | <?xml versi... | <?xml version="1.0" ... |

*Notre clé primaire est le champ ID, qui lui ne contient pas de XML.*

ID :

Clé primaire de notre base de données.

PERSONALINFO :

Contient la date de naissance signée et en *hash value*. Ce champ est signé avec la clé privée du *Generator*.

IDNUMBER :

Contient l'*ID unique* en *hash value* et signé. Ce champ est signé avec la clé privée du *Generator*.

PIN :

Contient le *PIN* unique en *hash value* et signé. Ce champ est signé avec la clé privée du *Generator*.

REFERENCENUMBER :

Contient l'*ID de référence* en *hash* value et signé. Ce champ est signé avec la clé privée du *Generator*.

**Détails spécifique**

Pour la génération des informations plusieurs algorithmes sont juxtaposés.
N'ayant pas d'autorité externe pour signer nos informations, nous créons un certificat que nous auto-signons (se reporter à la section *Système* pour plus de détails).

Nous générons une paire de clefs basée sur l'algorithme *RSA*. La clef privée est utilisée pour signer les informations et la clef publique est ajoutée dans le certificat.

L'encryption des informations se fait via une clef symétrique en mode *AES*. La clef utilisée pour encrypter est elle-même encryptée par la clef publique du serveur avec lequel on communique, dans notre cas l'*AccessInfo*. Elle est par la suite ajoutée dans notre document *XML* (*<EncryptedKey>*)..

La génération des informations uniques, qui sont le *PIN*, l'*ID unique*, *ID de référence*, sont créées avec le *SecureRandom SHA1PRNG* de SUN. Qui est un algorithme très fréquemment utilisé pour générer des nombres pseudo-aléatoires dit *Strong*.

Le *Register* reçoit le *PIN*, l'*ID unique*, la *Date de naissance* et l'*ID de référence*, sous forme de document *XML* signé (avec notre clef *RSA*) et leur valeur est en *hash value*. De cette façon il nous sera possible par la suite de contrôler lors du vote, les informations saisies par l'utilisateur.

L'*AccessInfo* reçoit le *PIN*, l'*ID unique* et l'*ID de référence*, sous forme de document *XML* signé (*RSA*), crypté(*AES*) puis signé(*RSA*) ainsi que le *login* qui lui est uniquement signé(*RSA)* mais en *hash value*. Nous pourrons de cette façon récupérer les informations encryptées (non en *hash value*), le *login* en *hash value* est là pour être vérifié uniquement.

## Diagrammes de classe

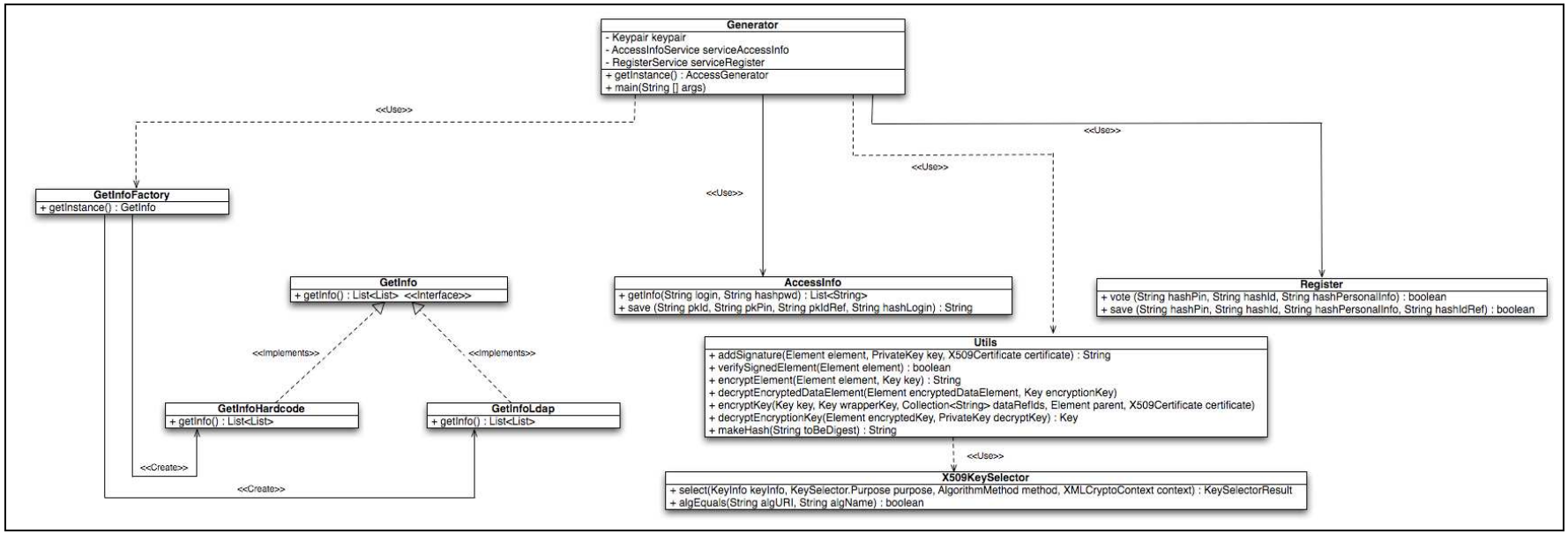



*Diagramme de la classe AccessGenerator*

### Diagramme de séquence

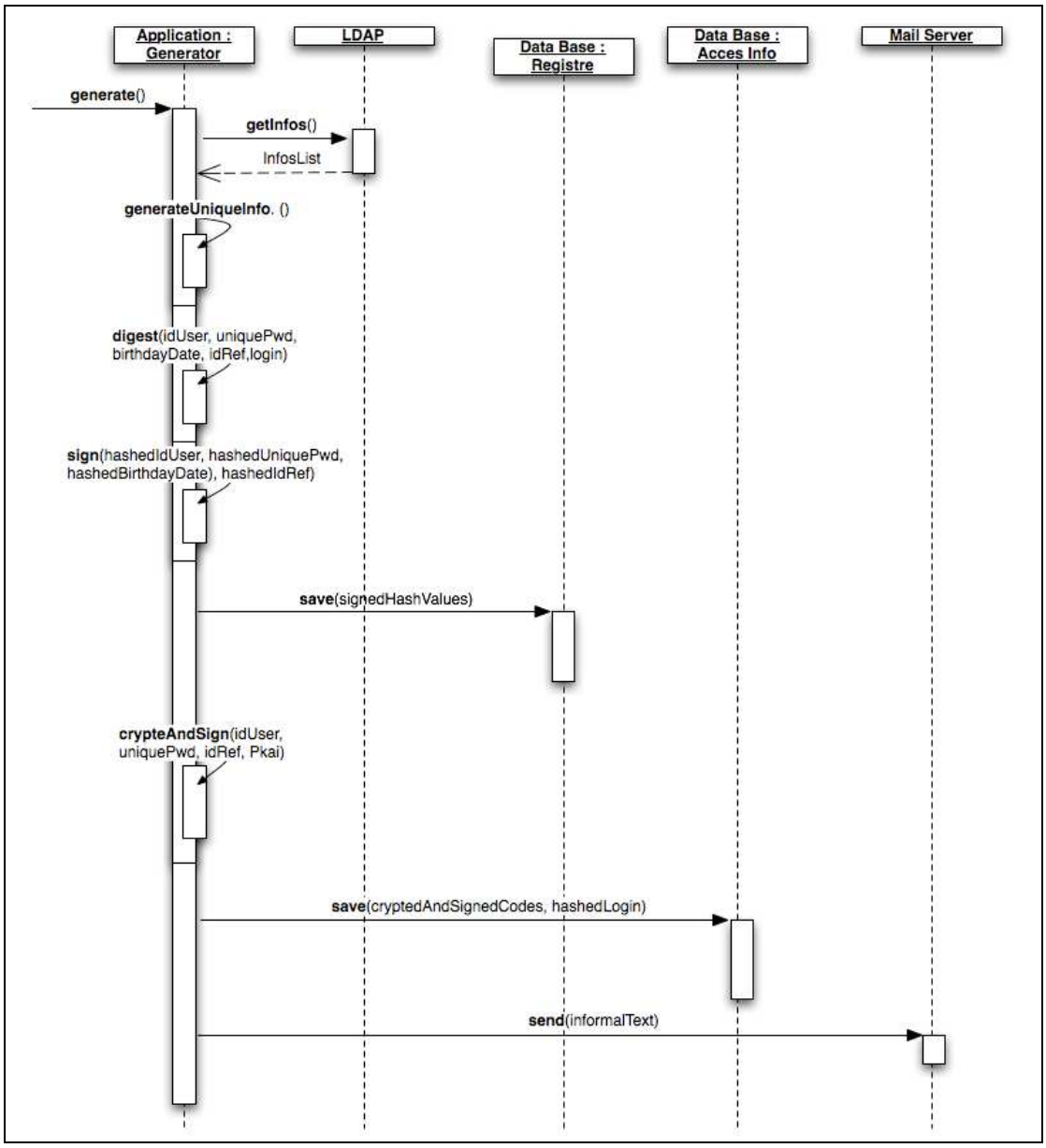



*Diagramme de séquence AccessGenerator. Génère les informations, les sauvegardes puis envoi l'email aux personnes ayant le droit de vote.*

### *Client - AccessInfo*

La partie *Client* est composée de deux éléments. L'obtention des codes d'accès et le vote. Le premier élément se fait tout d'abord par le téléchargement de l'*applet* (*l'applet* n'a pas été implémenté, se reporter à la section *Système*), puis via une authentification par *login*, *mot de passe* (dans notre solution, le *mot de passe* n'est pas contrôlé, se reporter à la section *Système*) sur ce même *applet*. Le client utilise les informations reçues via l'applet pour voter de façon anonyme sans employer ses identifiants propres (dans notre cas le *login* de l'école). Ceci garantie une **pseudonymie**, car il n'y a aucun lien entre l'*ID unique* utilisé pour s'authentifier lors du vote et le *login*.

#### 2. Récupération des codes d'accès

La première action du client, lorsqu'il désire voter, est de récupérer ses *codes d'accès personnels* qui se trouvent, une fois générés, dans la base de données *Info*. La communication se fait via le web service *AccessInfo*.

#### *Point de vue logique*

#### Classes, interfaces, relations

ClientModule :
Ce module est un *GUI* prototype, il propose à l'utilisateur d'insérer son login et son *mot de passe* afin de récupérer ces informations personnelles. Elles ont été générées par l'*AccessGenerator* et enregistrées dans la base de données *Info*. La récupération des informations se fait via la méthode *getInfo(…)* du web service *AccessInfo*.

#### Scénario

Paramètre :

| | |
|---|---|
| Login | Identifiant de l'école, il est connu de tous. |
| Mot de passe | Mot de passe de l'école, c'est un mot de passe de type **fort**, il est composé de majuscule minuscule symbole et numéro. Ceci est très important, le mot de passe doit être **résistant**. |

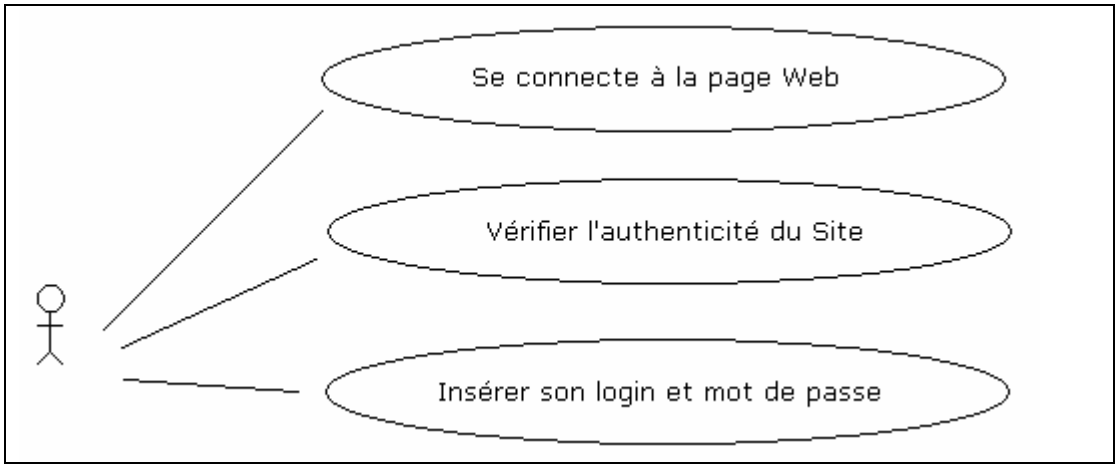



*Diagramme de fonctionnement de la récupération des codes.*

Déroulement

- Le client a reçu l'email l'avertissant qu'un vote est ouvert, il décide d'aller voter.
- Il se connecte sur la page web qu'il connaît (l'adresse n'est pas transmise par e-mail pour éviter les *fishing attack*). Arrivé sur cette page il vérifie le certificat de la page *https* (la connexion https n'a pas été implémentée, se reporter à la section *Système* de notre document).
- Il télécharge l'*applet*.
- Une fois l'*applet* lancé, il insert son *login* et son *mot de passe*.

**Détails spécifiques**

l'*AccessInfo* possède deux méthodes, une sauvegarde des informations *Save(…)* et la seconde qui récupère des informations *getInfo(…)*.

La sauvegarde vérifie la signature de chaque élément afin de garantir qu'aucune modification n'a eu lieu pendant l'envoi avant d'enregistrer l'élément dans la base de données.

La récupération va chercher, en fonction du *login,* les informations dans la base de données. Leurs signatures sont contrôlées avant qu'elles ne soient décryptées puis retournées à l'utilisateur.

Pour vérifier les signatures nous utilisons la clef se trouvant dans l'*EncryptedKey*.

## Diagrammes de classe

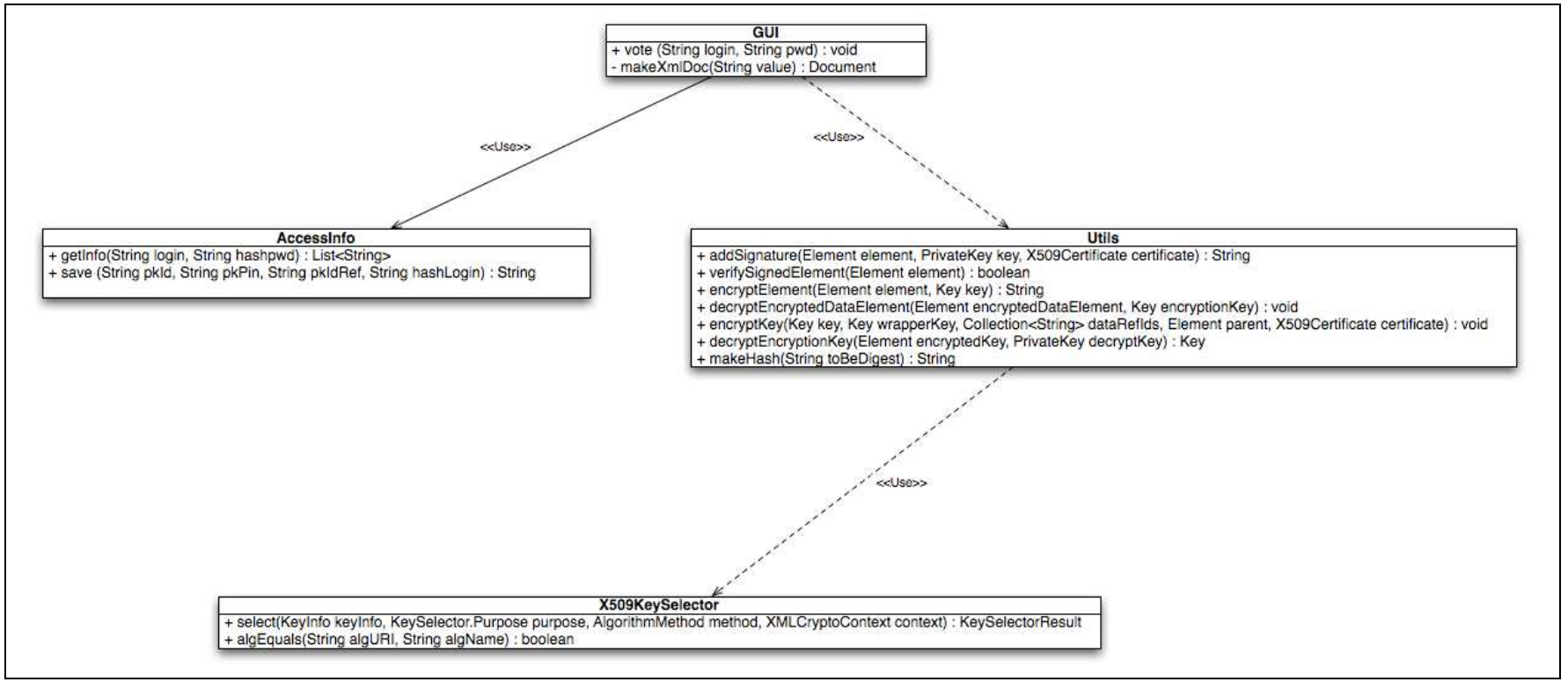



*Diagramme de la classe GUI du client pour la recherche de ces informations*

### *Serveur - AccessInfo*

La partie *Serveur* possède deux rôles. Le premier et d'authentifier le client et de lui distribuer les *codes d'accès*. Le deuxième sera expliqué au point *Contrôle et vote*.
Dans cette première phase la partie serveur va communiquer avec le web service *AccessInfo* pour vérifier que le client n'a pas déjà récupéré ses informations et dans ce cas, lui donner les informations dont il aura besoin pour voter.

#### 3. Authentification et distribution des codes

La première action du web service *AccessInfo* et de récupérer les informations dans la base de données *Info* avec comme paramètres, le *login* et le *mot de passe* (dans notre implémentation le mot de passe n'est pas contrôlé, se reporter à la partie *Système*).
Dans le cas où l'opération est un succès. Le web service décryptera les informations *XML* récupérées dans la base de données et les enverra au client.

#### *Point de vue logique*

#### Classes, interfaces, relations

AccessInfo :

Classe de type web service, qui récupère les informations *XML* dans la base de données en fonction du *login* et du *mot de passe*. Elle renvoie au client les informations récupérées et décryptées.

Info :

Classe de type *Entity* qui crée la table Info.

Utils :

Classe utilitaire contenant les méthodes pour traiter les informations *XML*.

X509KeySelector :

Classe qui sert à récupérer la clef publique se trouvant dans le certificat.

#### Scénario

La méthode *login* du web service est appelée par le client, avec comme paramètre le *login* et le *mot de passe*.
Une recherche se fait dans la base de données *Info*. Si la recherche est fructueuse, on vérifie la signature des éléments récupérés. Une fois les éléments contrôlés, on les décryptent. Ils sont ensuite supprimés de la base de données. La suppression évite que lorsque tout le monde a voté un administrateur arrive pour x ou y raisons à décrypter les informations et à faire le lien entre les votes dans l'urne et les ID présents dans la base de données *Info*. Ces informations sont ensuite envoyées au client sous forme décryptée et signée.

## Diagramme de classe

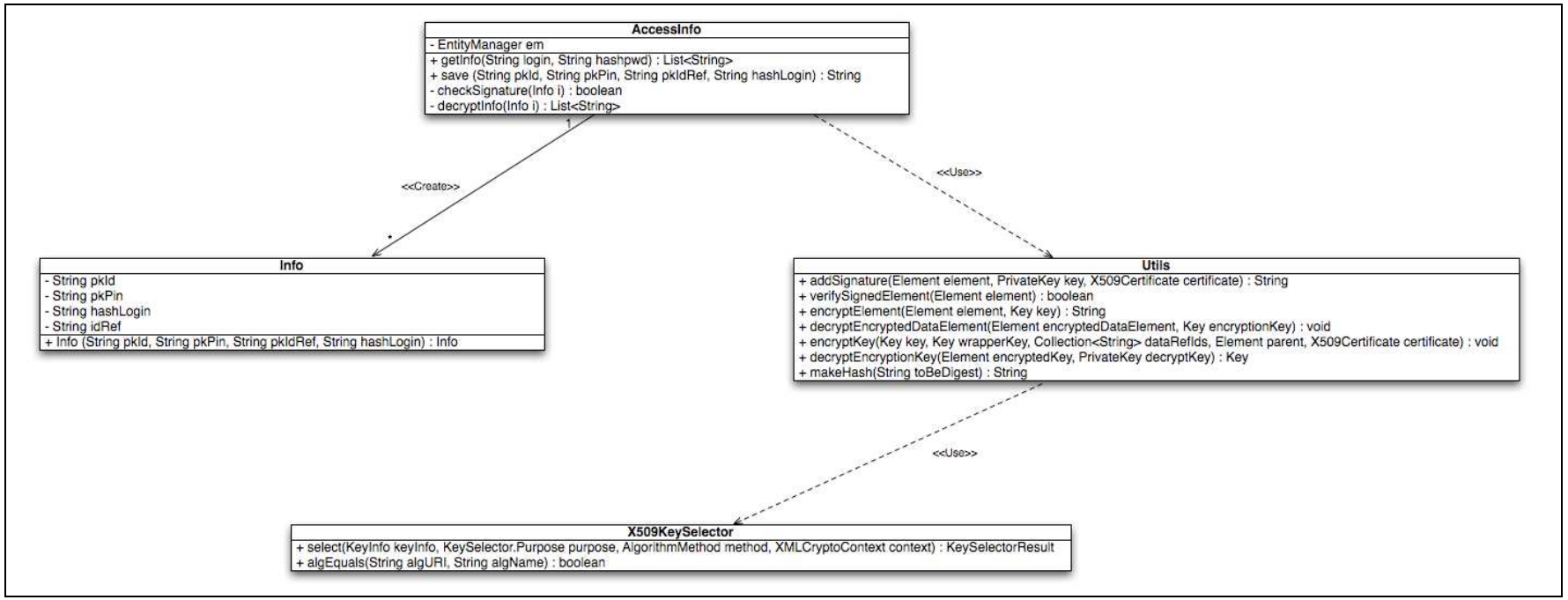



*Diagramme de la classe AccessInfo.*

## Diagramme de séquence

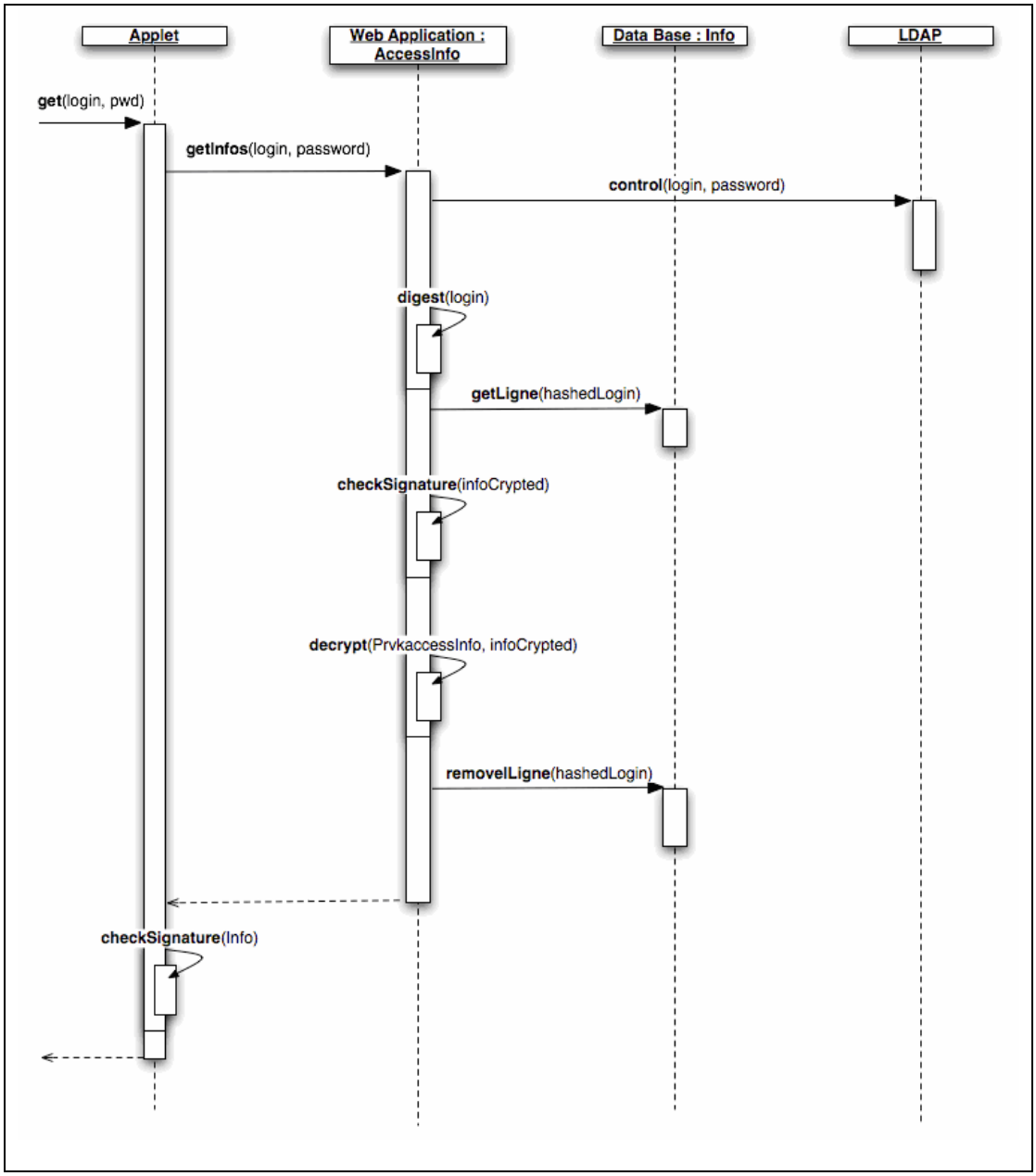



*Diagramme de séquence illustrant la récupération des codes. Depuis l'applet client via le web service AccessInfo*

### *Client - l'AccessInfo*

Cette partie du client est atteinte après avoir inséré son *login* et *mot de passe* dans *l'applet* et avoir reçu une réponse.

#### 4. Contrôle

Le contrôle se fait sur les informations renvoyées par l'appel de la méthode *getInfo()*.

#### *Point de vue logique*

#### Classes, interfaces, relations

ClientModule :

> Ce module est un *GUI* prototype, il propose à l'utilisateur d'insérer son *login* et son *mot de passe* afin de récupérer ces informations personnelles. Elles ont été générées par le *AccessGenerator* et enregistrées dans la base de donnée *Info*. La récupération des informations se fait via la méthode *getInfo(…)* du web service *AccessInfo*. Il vérifie la signature des éléments retournés par la méthode *getInfo* et les traitent.

#### Scénario

Après l'exécution du scénario de la partie 2 récupération des codes.
*L'applet* client vérifie la signature des éléments obtenus depuis la méthode *getInfo()* et dans le cas ou la vérification est un succès, affiche les informations reçues. Dans le cas contraire, un message d'erreur est affiché. (Le contrôle de la signature n'a pas été implémenté, se reporter à la section *Système*).

#### Détails spécifiques

Le client appelle la méthode *getInfo* de l'*AccessInfo* avec sont *login* en clair et le mot de passe. Le login doit être en claire pour la recherche sur le serveur *LDAP*.

## Diagramme de classe

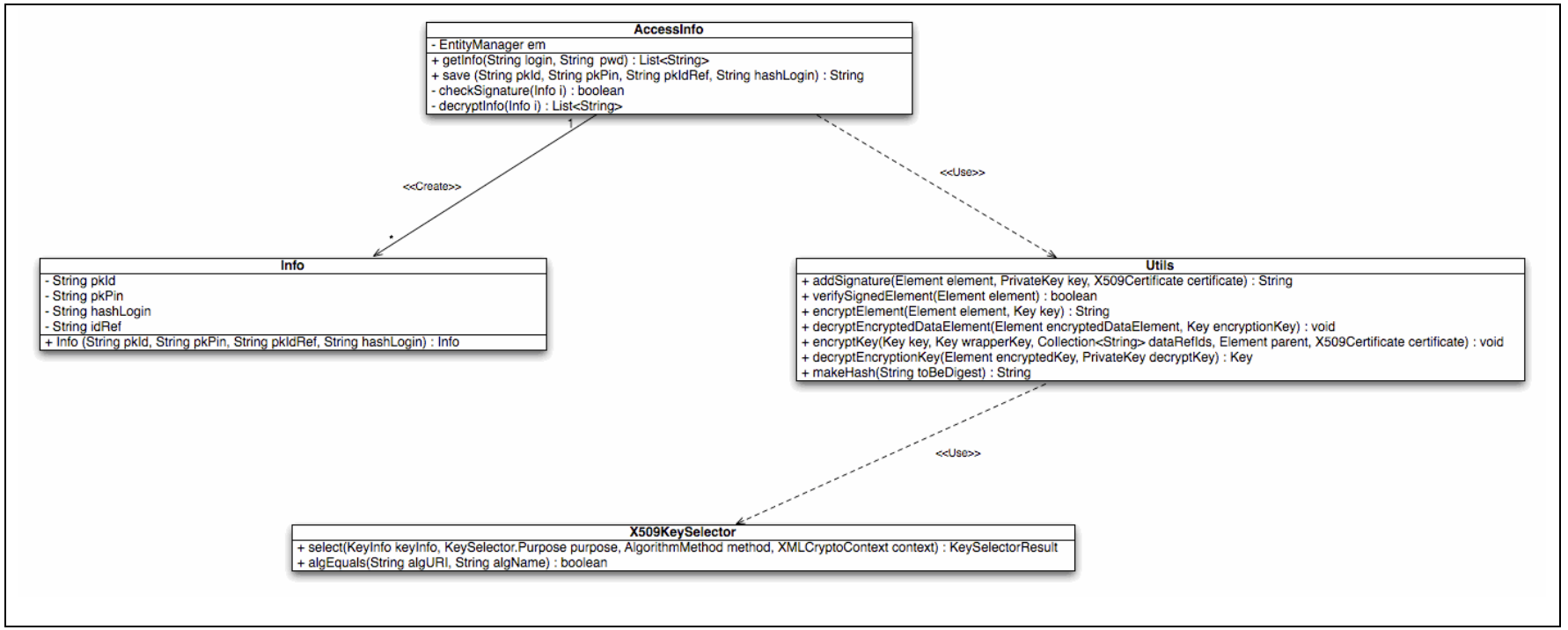



*Diagramme de la classe AccessInfo.*

## Diagramme de séquence

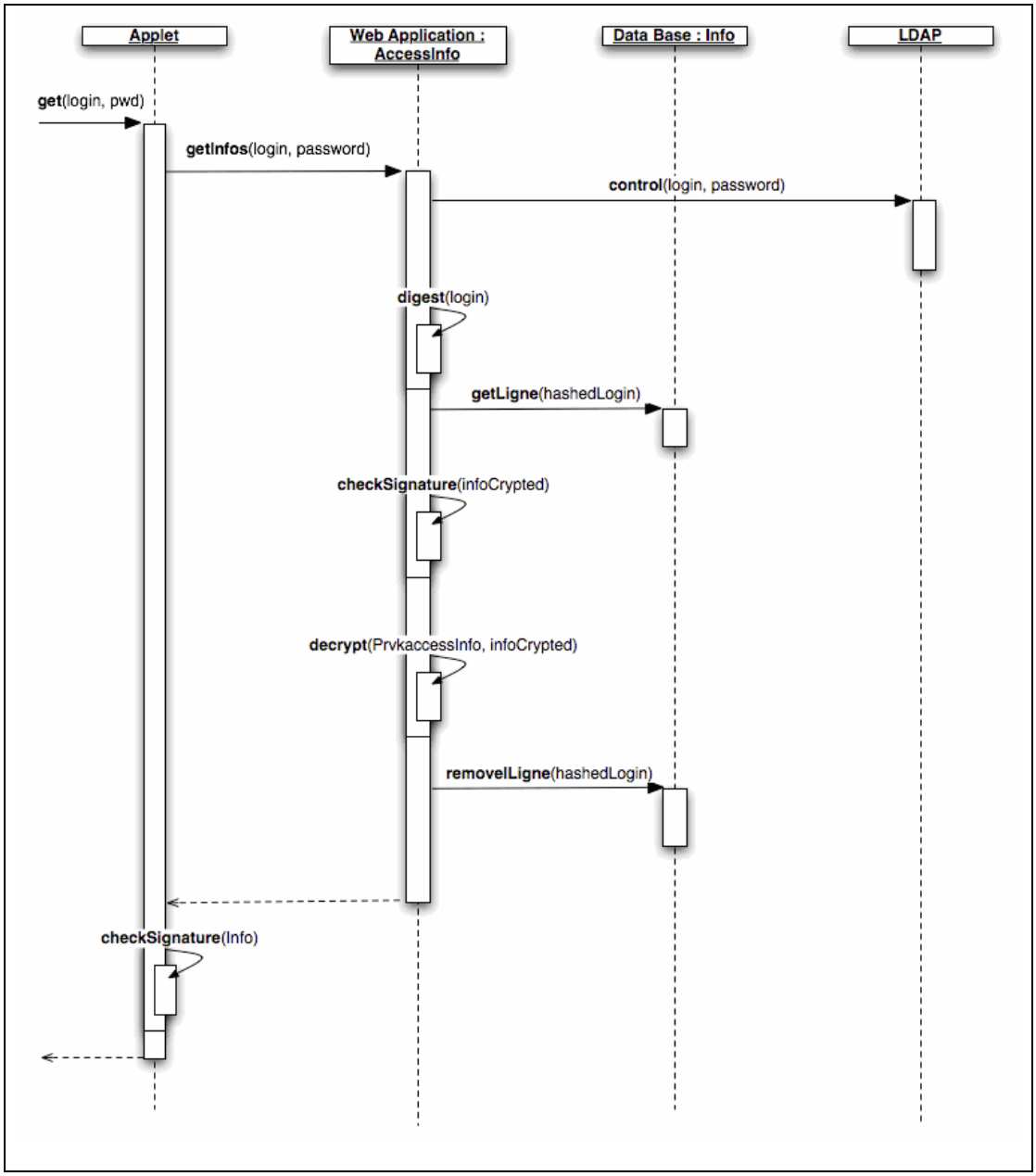



*Diagramme de séquence illustrant la récupération des codes. Depuis l'applet client via le web service AccessInfo*

### *Client ClientVoteModule*

Cette partie utilise un deuxième *applet* que le client télécharge sur un site prévu à cet effet, comme lors de la récupération des codes. (Cet *applet* n'est pas implémenté, se reporter à la section *Système*). Il lui permet de voter.

#### 5. Vote

Le vote se fait en insérant les informations récupérées au point 4 contrôle ainsi qu'une information personnelle, la date de naissance du votant.

#### *Point de vue logique*

#### Classes, interfaces, relations

ClientVoteModule :

Ce module est un *GUI* prototype, il propose à l'utilisateur d'insérer les informations récupérées depuis le *ClientModule* ainsi que sa date de naissance et bien sûr son vote.

#### Scénario

Paramètre :

PIN :

Code d'accès à 8 chiffres uniques servant de *mot de passe* à l'utilisateur.

ID unique :

Code d'accès à 10 chiffres uniques servant de *login* à l'utilisateur.

ID référence :

Code de référence permettant lors du comptage de vérifier qu'aucune modification n'a été effectuée pendant la décryption de la base de données. Dans ce cas, il sert également de mot de passe.

Date de naissance :

Informations personnelles de l'utilisateur qui permet d'augmenter la sécurité. L'utilisateur à besoin de trois codes qu'il possède (*PIN, ID unique*, *ID de référence*) et d'un élément qu'il connaît (sa date de naissance).

Vote :

Vote de l'utilisateur

L'applet reçoit les informations ci-dessus en paramètre, il crée une *hash value* du *PIN*, de l'*ID unique* et de la *date de naissance*. Il fait une encryption *XML* du *Vote* et de l'*ID référence* avec la clef publique de l'*Urne*. Ce qui garanti une sécurité *END-TO-END* car le vote ne peut être décrypté que par l'urne.

#### Détails spécifiques

Les paramètres, qui sont le *vote* et l'*ID de référence,* sont encryptés avec une clef de session *AES*. Cette même clef est encryptée avec la clef publique du serveur avec lequel on communique, dans notre cas celle du *VoteServer.*

### Diagramme de classe

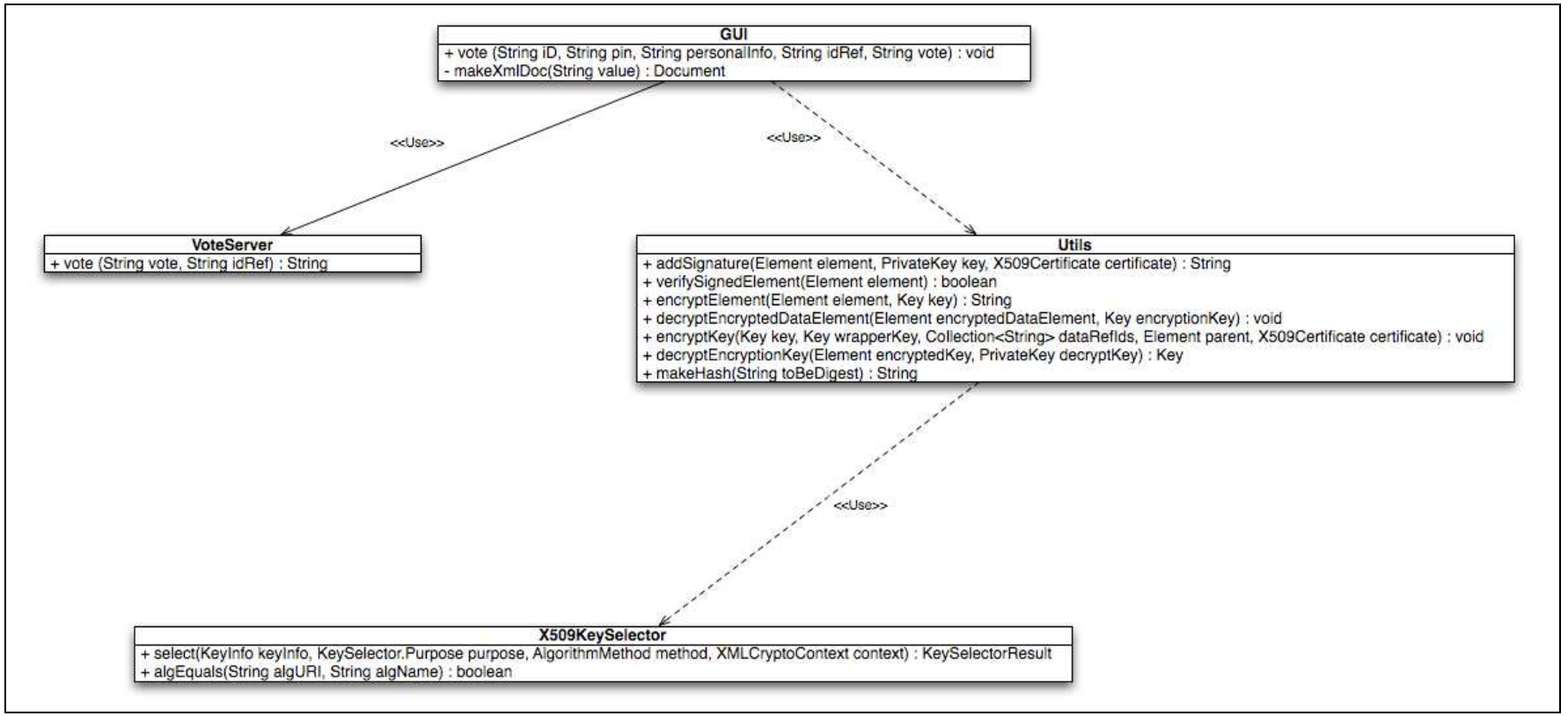



*Diagramme de la classe GUI du client pour le vote*

### Diagramme de séquence

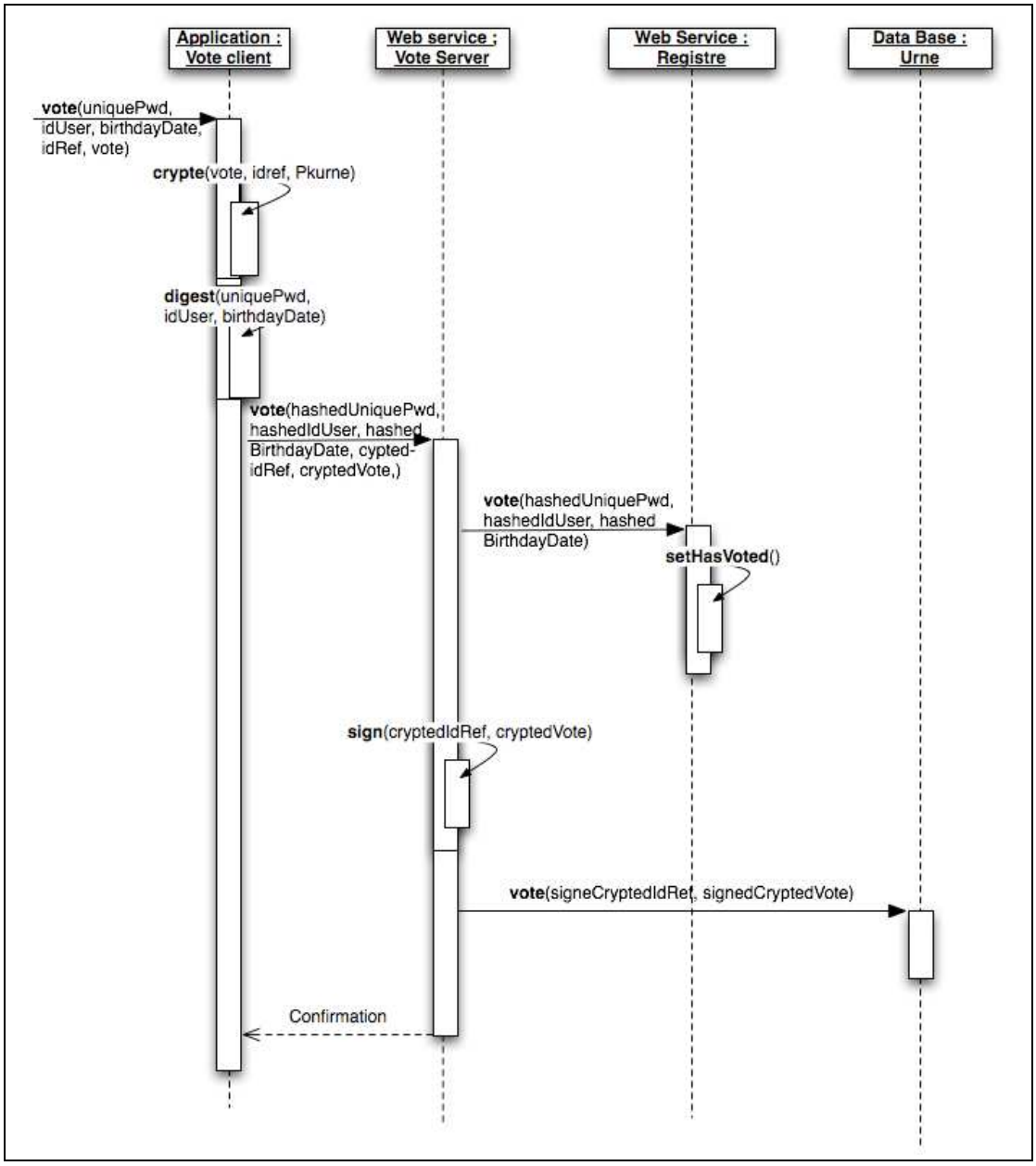



*Diagramme de séquence du GUI client qui vote en communiquant avec le VoteServer*

### *Serveur VoteServer*

La partie *VoteServer* est la partie externe avec laquelle le *GUI* client communique. Derrière cela le *VoteServer* à son tour entre en contact avec deux web service.

#### 6 Traitement du vote

Le web service *VoteServer* reçoit les informations qu'il transmet au *Register* puis à l'*Urne*.

#### *Point de vue logique*

#### Classes, interfaces, relations

VoteServer :

Classe qui est appelée par le client avec l'*ID unique*, le *PIN*, la *Date de naissance*, le *vote* et l'*ID de référence*. Elle communique avec le *Register* en lui envoyant le hash du *PIN*, hash de l'*ID unique* et le hash de la *Date de naissance*, pour savoir si le client a déjà voté. Si le client n'a pas encore voté, il transmet le *vote* et l'*ID de référence* à l'urne pour la sauvegarde.

Utils :

Classe utilitaire contenant les méthodes pour traiter les informations XML.

X509KeySelector :

Classe qui sert à récupérer la clef publique se trouvant dans le certificat.

#### Scénario

La méthode *vote(…)* du web service *VoteServer*est est appelée par le *GUI* client. Le *VoteServer* envoie le *PIN,* l' *ID unique* et la *date de naissance* au web service *Register*. Pour savoir s'il a déjà voté. Ensuite, s'il n'a pas encore voté, il transmet le *vote* et l'*ID référence* au web service *Urne* pour son enregistrement dans la base de données.

#### Détails spécifiques

Le vote serveur communique avec deux autres web services le *Register* et l'*Urne*.
Il transmet le *PIN*, l'*ID unique* et la *Date de naissance* au *Register* afin de contrôler si il a le droit de vote.
Si il a le droit de vote, il signe le *vote* et l'*ID de référence* avant de les envoyer à l'*Urne* pour les enregistrer.

**Diagramme de classe**

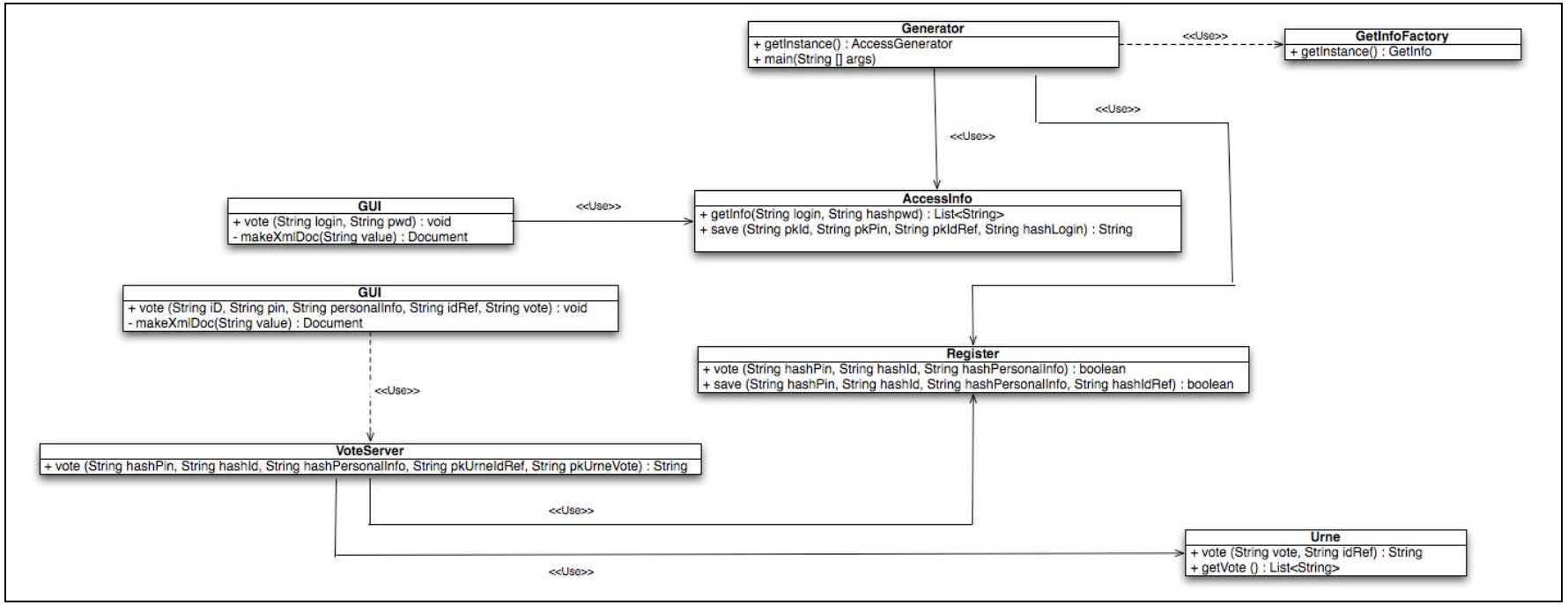



*Diagramme de la classe générale montrant l'interaction du VoteServer avec l'Urne et le Register*

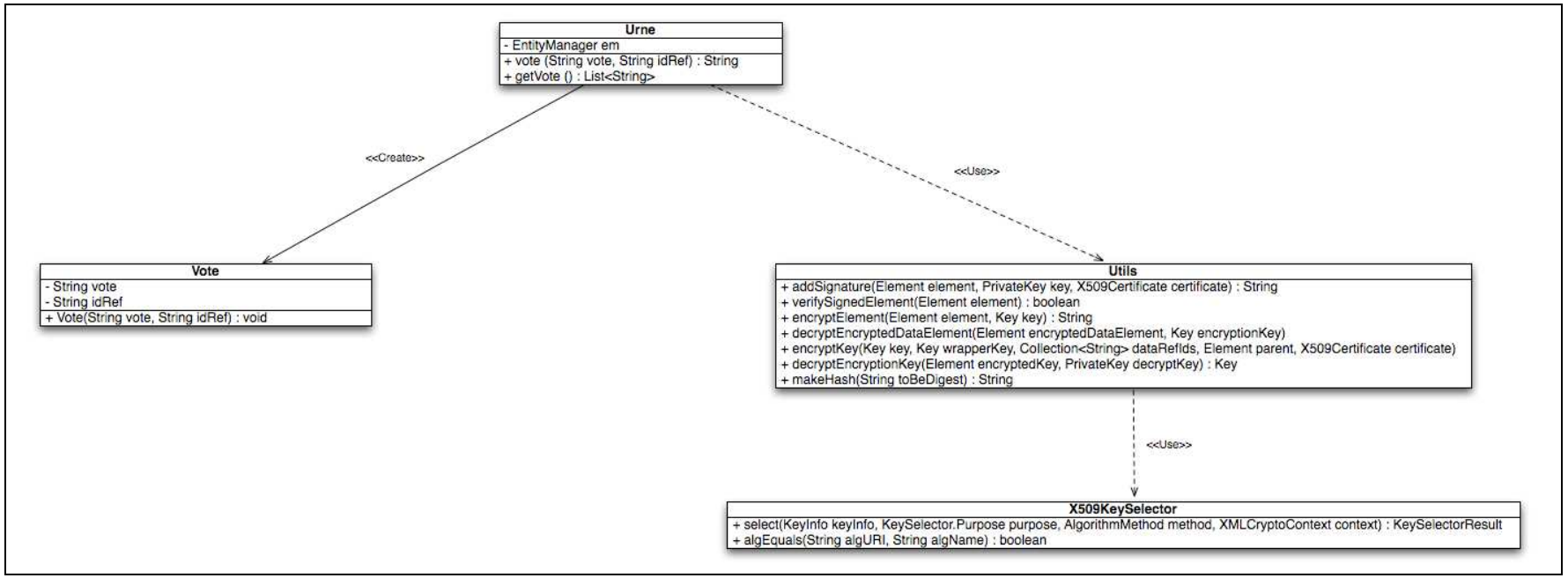



*Diagramme de la classe Urne*

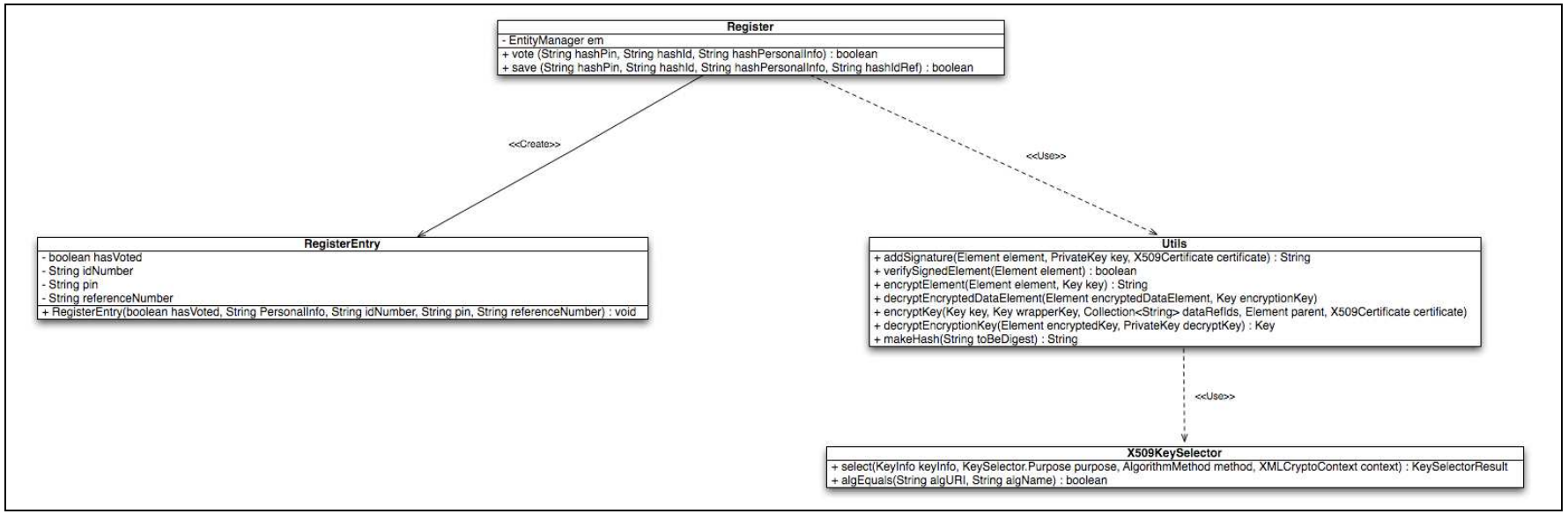



*Diagramme de la classe Register*

### Diagramme de séquence

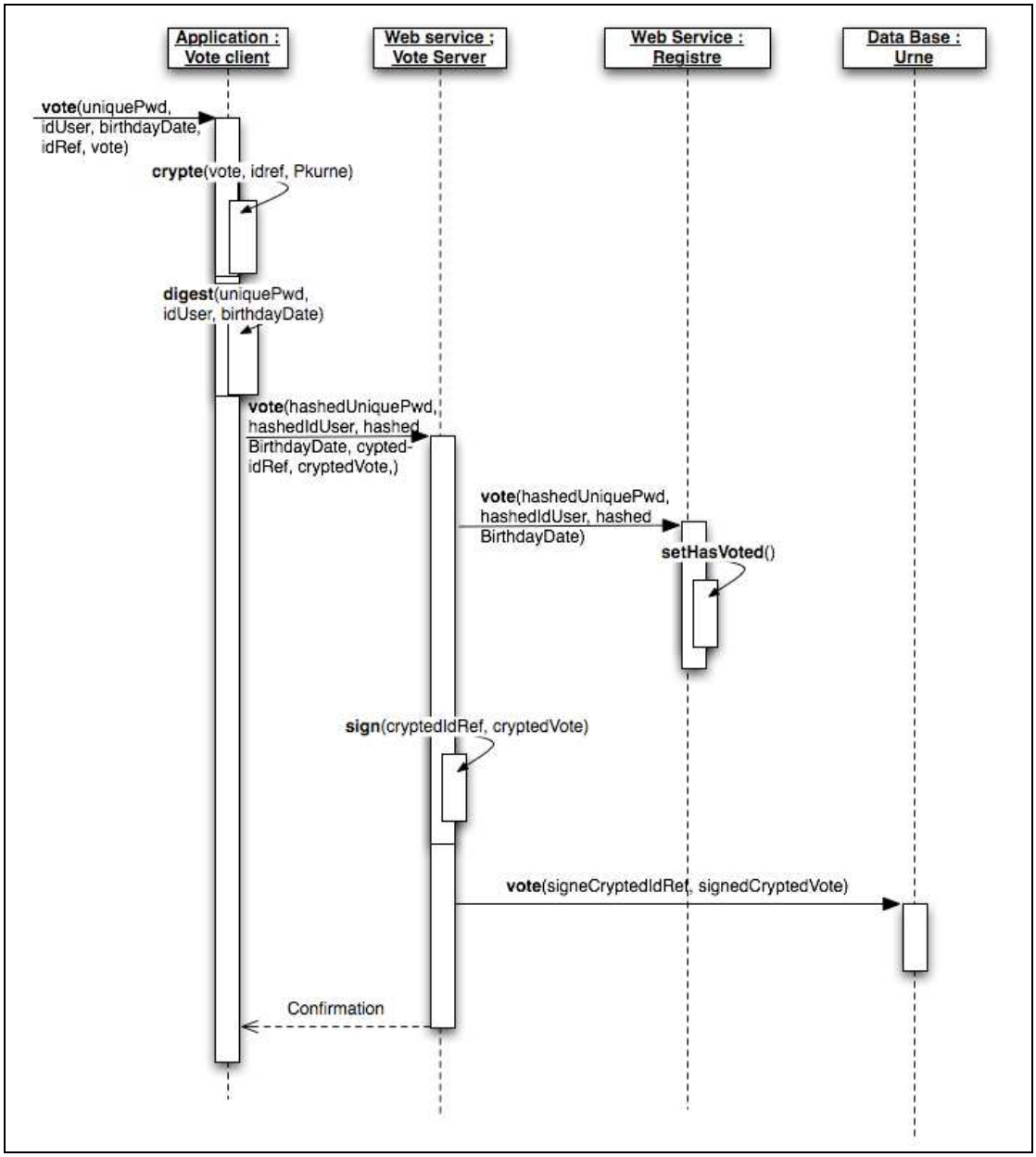



*Diagramme de séquence d'un vote depuis l'applet client via le VoteServer*

## Système

Cette partie du document décrit les fonctions qui n'ont pas été implémentées dans notre solution, puisque comme décrit dans la section *Planning*, il était impossible de tout réaliser dans le temps imparti. Cette section a donc pour but de décrire ce qu'il reste à faire, le fonctionnement des différents protocoles de sécurité qu'il reste à mettre en place et donner une vue globale du projet terminé.
Avant de développer en détail le fonctionnement des parties non implémentées, voici un bref résumé des choses qu'il reste à faire dans le projet :

- Génération des clefs :
  - Créer un tunnel sécurisé entre le serveur de génération et les deux serveurs de stockages.
  - Brasser (mélanger) le registre des électeurs, afin de garantir la *pseudonymie* (éviter que l'ordre alphabétique du serveur LDAP ne corresponde aux entrées dans la base de données).
  - Récupération des informations sur les citoyens via une base de données *LDAP*.
  - Récupération des clefs publiques via *XKMS* (enlever les clefs *hardcoder*).
  - Utilisation de clefs privées sécurisées (clef stocker dans des modules hardware).
  - Rajout de *l'hash value* du certificat *HTTPS* du serveur de vote comme code d'accès.
  - Utilisation de certificat signé par des instances connues et *trusts* (*Verisign*, …).
- Client :
  - Transformation de l'application java en *applet*.
  - Création d'un tunnel sécurisé entre *l'applet* client et le serveur.
  - Récupération des clefs publiques via *XKMS*.
  - Amélioration de l'interface graphique.
- Web Service :
  - Communication *HTTPS* pour la récupération des *applets*.
  - Communiquer avec *XKMS* pour la récupération des clefs publiques utilisées lors des différentes communications.
  - Configuration des différents fichiers *Policy (web – service Policy*)
- Base de données :
  - Brassage (mélange) des bases de données afin d'éviter toute comparaison recoupage entre l'heure de vote et l'ordre de stockage.

Il reste donc un certains nombres de choses à réaliser afin d'avoir une version complète et utilisable. Cette partie du document sert donc de guide pour la suite des opérations. Ce qui arrive va décrire et expliquer dans les détails les implémentations qu'il reste à faire, le tout classé par section :

- Generator
- Client module
- Web – service
- Communication
- Firewall
- Gestion des clefs
- Protocoles de sécurité divers

## Generator

Comme dit plus haut, le générateur (traduction française) est le programme servant à la génération des codes d'accès et à la transmission de ces informations dans le but de les stocker dans des bases de données. Son rôle est donc de la plus haute importance, les codes générés se doivent d'être le plus aléatoire possible, les protocoles d'encryptions, les clefs et les algorithmes choisis nécessites une grande résistance, puisque les informations en sortant sont stockées sur des médias permanemment.
Actuellement, le *generator* crée des codes d'accès comprenant les informations suivantes :

- Id unique
- Pin unique
- Id référence

Ces trois informations sont donc celles qu'un citoyen peut récupérer lors de la phase de récupération des codes. A ces informations, il faudra **rajouter le fingerprint** du certificat du serveur de vote, afin que les votants puissent vérifier cette information et ainsi réduire les chances d'attaques *phising* ou *Man-In-The-Middle*. Le *Generator* devra donc créer et stocker dans la base de données *Info* :

- Image sécurisée comprenant (un *ID Unique*, un *PIN Unique*, un *ID de référence*)
- *Fingerprint* du certificat (même pour tous)

**L'envoi d'image sécurisé** au lieu de simple String encrypté, permet d'accroître la sécurité des informations se trouvant sur l'ordinateur du client, empêchant ainsi à un virus quelconque d'interpréter les informations envoyées.

*Exemple d'image sécurisé (tiré de Windows live Messenger)*

**Le brassage des bases de données** *RegistreEntry* et *Info* doivent aussi être implémentés lors de la génération des codes. Cette manipulation a pour but d'éviter toutes correspondances entre l'ordre des codes se trouvant dans les DB et l'ordre alphabétique des citoyens. Une petite manipulation comme celle-ci peut paraître négligeable, mais pourrait mettre en danger toute la *pseudonymie* de l'application si elle n'était pas mise en place.
**La communication avec le serveur *LDAP*** doit être créée, car elle n'est pas encore mise en place et que c'est sur ce serveur que se trouvent les informations sur les votants. Cette communication doit aussi être réalisée via un canal sécurisé, garantissant la *confidentialité* et *l'intégrité* des données transmises.
Un tunnel *SSLI/TLS* 128 bits est tout à fait adéquat pour cette sécurisation du transfert. *L'intégrité* et la confidentialité sont requises afin d'éviter que des informations, servant à l'identification lors de la phase de vote, ne soient transmises en *clear text* (Exemple : la date de naissance, servant de secret connu et non transmis), ainsi modifiés ou lus par des attaquants potentiels.
**La gestion des clefs** est sujet très important pour lequel il faut apporter beaucoup de modifications entre la version actuelle et la version finale du système. Pour en savoir plus sur ces modifications, référez-vous au point ci-dessous *Gestion des clefs.* Le comportement, la formation et les droits des administrateurs ainsi que l'utilisation des serveurs sont détaillés dans la section ***Protocoles de sécurités** divers.*

## Client module

Comme précisé dans l'architecture, le client module comprend deux parties, une servant à la récupération des codes d'accès et une autre à la réalisation du vote. Chacune de ces deux parties possède sa propre sécurité ainsi que ses besoins en sécurité.
Actuellement, ces deux parties sont des applications Java, celle-ci seront **transformées en applet** Java pour des raisons de confort d'utilisation et d'aspect de sécurité. Lors de ce transfert, le code réalisé dans l'application devra être le même dans l'applet, seul **l'interface graphique proposé devra être optimisée** afin d'augmenter le confort et l'ergonomie de son utilisation.
La communication réalisée lors de la phase de récupération des codes, étant pour l'instant envoyé en mode *clear text*. Ceci doit impérativement être changé pour la version finale. Le tunnel de sécurité devant être mis en place, doit respecter les points suivants :

- Solide encryption
- Intégrité des messages garantie
- Mise en place des *time stamp* permettant d'éviter les *reply attack*.

**Un tunnel sécurisé** utilisant *SSL/TLS* 2.0 128 bits ou *SAML* 2.0 encryption (avec *time stamp*) répond parfaitement à ces critères et devra être mis en place.
**La gestion des clefs** est tout aussi importante pour le client module que pour le générateur, il faut donc lui apporter de grandes modifications afin d'augmenter la sécurité quasi inexistante dans la version actuelle (version de démonstration), pour plus de détails, voir ci-dessous *Gestion des clefs*. En distribuant les modules de votes aux **votants,** il est très **important de les former** à une utilisation correct de ces outils ainsi que de leur fournir un enseignement plus important sur la sécurité. La section *Marche à suivre client* apporte une partie de cette enseignement au votant, mais une formation régulière serait encore plus adaptée et permettrait d'éviter de potentiels hacks.
Il est évident qu'une telle formation est quasie impossible dans une école puisqu'elle implique des coups et du temps supplémentaires. Un compromis serait un bulletin régulier, rappelant les concepts et protocoles de sécurité à suivre, ce bulletin peut très bien prendre la forme d'e-mail ou de bulletin papier.

## Web-Service

Dans la version actuelle, il y a quatre web service, un *AccessInfo* (servant à récupérer les codes d'accès), un *Register* (servant de base de registre pour identifier les votants), Un *VoteServer* (servant de service de communication pour les votes) et une *Urne* (servant de service s'occupant du stockage des votes).
La sécurité de ces services est dépendante de la qualité du code, des algorithmes choisis et de l'utilisation des clefs. Ceci représente réellement le cœur de la sécurité de notre système de vote, c'est ici que le code pour les signatures et les encryptions sont réalisés. Il est donc primordial que ce code soit le plus optimisé possible.
Les algorithmes choisit pour l'encryption sont *RSA* 1024 bits et *RSA* signature pour réaliser les diverses signatures. Ces algorithmes correspondent à la sécurité nécessaire pour réaliser un système de *E-voting*, une plus grande taille pour les clefs ralentirait les calculs de l'application et ainsi le système de vote dans son ensemble. Niveau code, il faut par contre absolument améliorer la gestion des clefs qui pour l'instant utilise des clefs *hardcoder* (ce qui ne correspond pas du tout aux règles de sécurité nécessaire, réalisé ainsi par manque de temps).
Il est nécessaire de modifier le code des signatures et le contrôle des signatures de tous les web service les utilisant, puisque pour l'instant les certificats sont auto-signés via la même clef privée utilisée pour la signature du document.
Il faut donc **utiliser des certificats *X509* signés par** des instances connues et trusts tels que ***Verisign*** ou autres institutions. Grâce à ces certificats signés proprement, il est possible de contrôler la signature de la façon la plus sécurisée qui soit.
Ces modifications impliquent les classes suivantes :

*AccesGenerator* : méthode *signInfo* ()
*Utils* : méthode *verifySignedElement* ()
*VoteServer* : méthode *signInfo* ()

**La gestion des clefs** est primordiale pour les web service, il est donc de la plus grande importance que les clefs utilisées par les web service soient faites de la façon la plus propre et sûre qu'il soit (pour plus de détails voir ci-dessous *Gestion des clefs)*. La partie *protocoles* ci-dessous va définir comment doivent être utilisés et administrés les différents web-services.

## Communication

La version actuelle ne comprend pas de canaux de communication sécurisés, seul *l'XML* encryption et signature sont pour l'instant implémentés. Il est donc primordial pour la version finale de rajouter des canaux de communication sécurisés, ceci dans le but d'apporter :

- *Confidentialité*
- *Intégrité*
- *Time stamp* ou équivalent

L'*XML* encryption et signature mis en place garantissent déjà la confidentialité et l'intégrité mais ne garantissent pas une sécurité équivalente aux *time stamp*. Il est donc très important de mettre en place de tels canaux.

Pour se faire les choix sont relativement vastes mais nous en n'avons retenus deux :

- *SLL/TLS* 128bits
- *SAML* encryption avec *time stamp*

Lequel des deux choisir n'est pas très important, sachant que les deux proposent au final, la même chose. Ce qui est par contre primordial, c'est où ces canaux de communication seront mis en place.

Afin d'avoir une sécurité optimale, ils seront mis en place pour chaque communication entre les différents acteurs du E-voting (client , serveur, générateur, …). Le schéma ci-dessous montre ces quelques canaux :

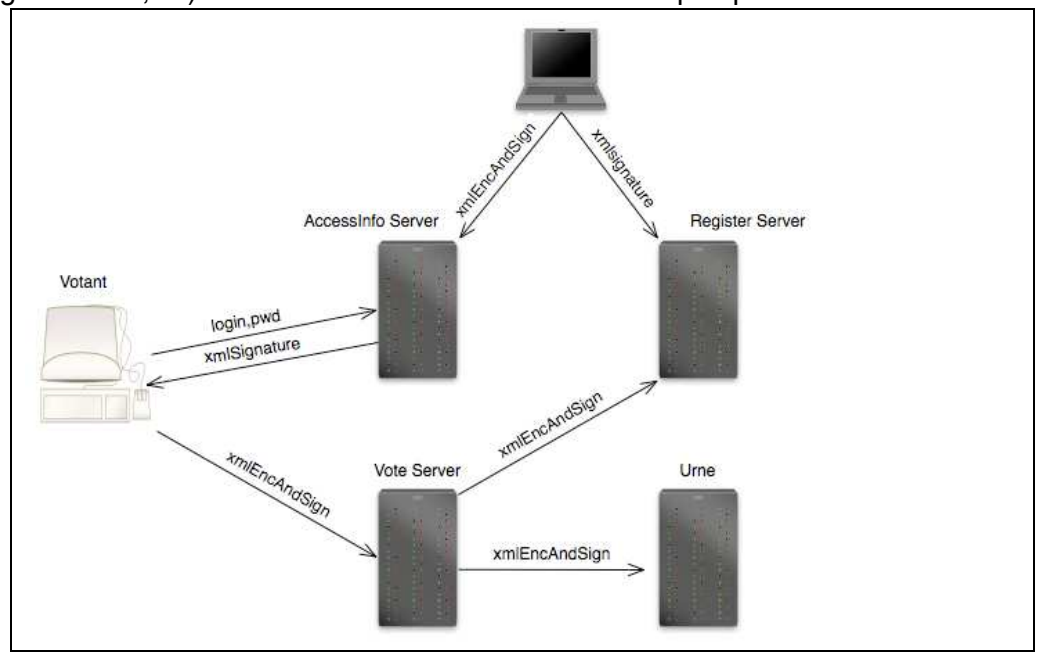



*Image montrant les différents canaux sécurisés. Les flèches représentent des canaux SSL.*

## Firewall

Les firewalls sont des éléments importants de la sécurité, qu'il ne faut pas négligé. Pour notre système *d'E-voting* ces éléments jouent un rôle important et permettent de réaliser un premier aspect de sécurité en coupant les zones en zones de plus en plus sécurisé (la zone la plus sécurisé étant celle contenant le registre et l'urne.
Le schéma si dessous montre la disposition des firewall dans notre système :

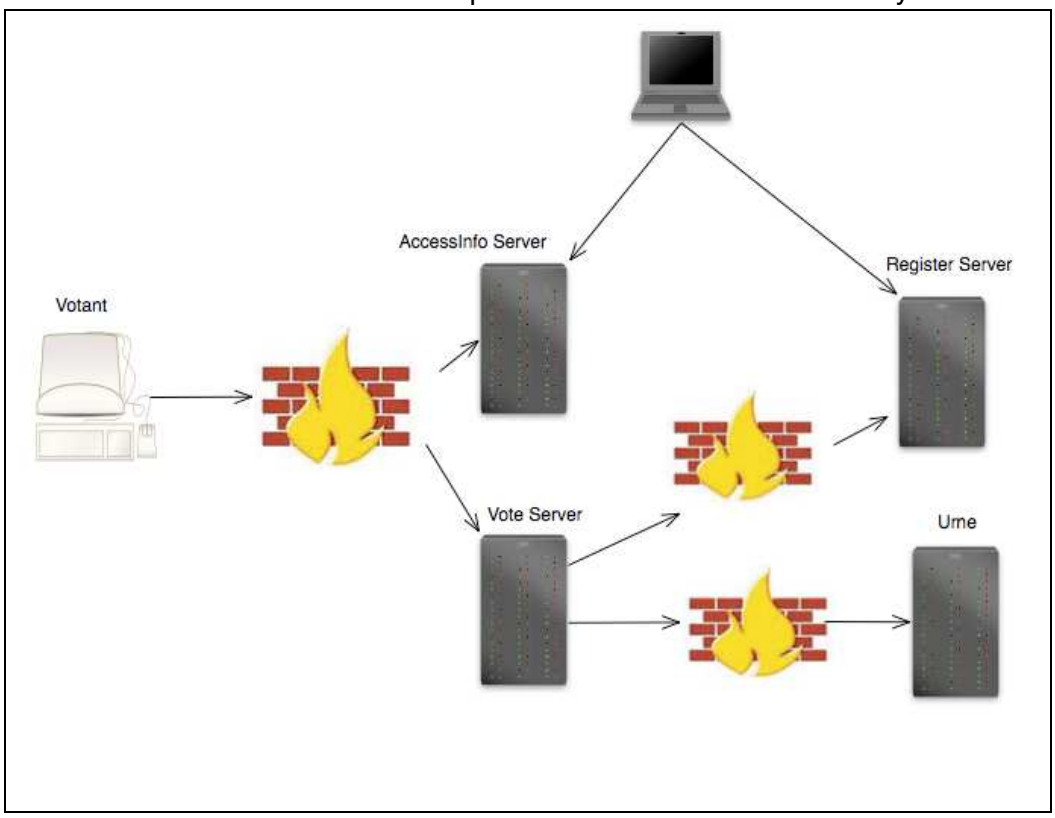



*Image montrant le placement des différents firewalls.*

Comme le montre l'image ci-dessus, le système final sera protégé par trois firewalls. Le premier se trouvant entre le votant les *AccessInfo* et *Vote* a pour tâche d'autoriser et protéger l'accès aux serveurs publics. Ce *pare-feu* permet aussi d'autoriser l'accès au serveur de vote uniquement en période de votation (voir section *Protocoles* pour plus d'information).
Les deux autres *firewalls* se trouvant entre le *VoteServer* et le registre respectivement l'*Urne*, ont pour but d'accroître la sécurité de ces deux serveurs qui ont un rôle critique dans les votations. Ils ont aussi pour but d'autoriser l'accès uniquement durant les périodes de votation.

## Gestion des clefs

La gestion des clefs est de la plus haute importance, les clefs doivent être stockées dans un endroit sûr et restreint d'accès. Les certificats mis dans les signatures doivent être signés par des *trusted* serveurs et être utilisés de façon convenable. Ces *trusted* serveurs peuvent être :

- Digital signature trust
- Verisign
- GlobalSign
- ….
- Serveur trust de l'école

Le stockage des clefs et des certificats ne doit pas se faire de façon *hardcoder* comme réalisé dans la version actuelle mais via un moyen plus sécurisé. Cette façon sécurisée est soit le stockage des clefs dans des fichiers à droit d'accès très restreints ou l'utilisation de matériel externe.
L'avantage de l'utilisation de matériel externe est que les clefs privées ne peuvent en aucun cas être accédées. Il suffit de limiter l'accès au matériel du système *d'E-voting* et ainsi la gestion des clefs est sûre.
Le système de fichiers à des droit d'accès restreint et a comme avantage de n'impliquer aucun cout supplémentaire, mais en cas d'élévation de privilège la clef pourrait être compromise alors que dans le système externe, seule son utilisation serait compromise.
Les deux solutions offrent une solution viable et sûre.
Le système d'E-voting que nous avons créé, utilise beaucoup de clefs privées, certificats et clefs publiques. Le texte qui suit résume l'utilisation de chaque clef pour chaque entité :

- Le générateur possède une clef privée servant à signer les informations qu'il stocke dans les bases de données *RegisterEntry* et *Info*. Il utilise la clef publique du serveur AccessInfo afin de crypter les informations qu'il enregistre.
- Le registre utilise le certificat se trouvant dans les informations signées pour contrôler la signature. Ce certificat doit être signé par un trust serveur afin d'être valable.
- *L'AccessInfo* serveur utilise aussi le certificat se trouvant dans les informations signées afin de contrôler les signatures. Il utilise aussi sa propre clef privée pour décrypter les informations se trouvant dans la base Info.
- *L'applet* de récupération des codes utilise le certificat se trouvant dans les informations reçu pour contrôler les signatures des informations reçus.
- *L'applet* de vote utilise la clef publique de l'urne préalablement récupérée afin de signer l'i*ID de référence* et le *bulletin de vote*.
- Le vote serveur utilise sa propre clef privée pour signer les informations qui seront stockées dans l'*Urne*.

## Protocoles

Les protocoles définis ici sont des conseils et marches à suivre afin d'accroître la sécurité du système de E-voting, leur importance est primordiale et apportent l'équivalent en sécurité que pourrait le faire un firewall ou autre système.
En effet, le comportement et l'utilisation d'un système sont très importants, une mauvaise utilisation ou définition des droits peut engendrer de grandes vulnérabilités. Les protocoles définit sont les suivants :

- Utilisation, placement et droit d'accès physique du système
- Droit des administrateurs
- Que faire en cas de fraude ?

L'endroit où placer les serveurs est très important, cela doit se trouver dans une salle spécialement conçue pour les serveurs (refroidie et aérée). L'accès à cette salle doit être contrôlée et filmée. Une personne seule ne doit pas pouvoir y entrer, ce qui signifie que les administrateurs ou techniciens devant y accéder doivent toujours être par groupe de deux. Ce protocole peut paraître sévère et non approprié, mais le *E-voting* requiert des mesures de sécurité élevées qu'il ne faut pas négliger.
Les droits des administrateurs ne doivent en aucun cas être l'équivalent du *root*, mais des droits d'utilisation restreints. Ceci permet d'empêcher toutes installations de logiciel par des tiers ayant un accès physique au serveur, ainsi que de n'autoriser l'accès aux clefs uniquement au système de vote et à personne d'autre. L'administration se fait donc via des scripts préconçus que les administrateurs peuvent utiliser selon les situations.
Si un problème devait demander des accès plus importants pour être corriger, les administrateurs devraient demander les mots de passes *root* à la direction. Cette dernière, après avoir étudié le cas, récupèrera les mots de passes stockés dans un coffre fort, nommera deux personnes indépendantes des administrateurs et leur transmettra les codes sous scellés. Les deux personnes indépendantes auront pour tâche de transmettre les codes aux administrateurs et de surveiller leur travail. Ceci implique que les deux personnes choisies devront avoir de bonnes connaissances en informatique. Une fois le problème résolu, de nouveaux mots de passes seront mis pour les droits *root*, mis sous scellés et stockés dans le coffre fort de l'école.
Si une fraude venait à être détectée durant la période de vote, l'ampleur et l'impact de cette fraude devront être transmises à une autorité compétente (Ex : direction) afin que les décisions appropriées soient prises. Durant la période de vote, les fraudes peuvent apparaître sous différentes formes, virus sur un serveur de vote, clef compromise,…
L'autorité compétente agira en fonction de la gravité de l'attaque, en cas extrême tous les votes seront supprimés, les bases de données effacées, les serveurs sécurisés et une nouvelle session de vote sera ouverte (recréation des codes d'accès).
Si une fraude est détectée lors du comptage celle-ci sera signalée. Son impact sera nul puisque le système à été conçu dans le but d'empêcher toute personne non autorisée de voter, ceci grâce à l'*ID de référence* et aux votes signés. Les votes frauduleux auront donc une mauvaise signature et un *ID de référence* ne correspondant pas avec la table Registre, ils pourront donc être écartés du comptage facilement.

## Audit

Ce document présente différentes attaques possibles et la résistance du système de vote développé lors de notre thèse (système codé & solution finale). Un audit est une évaluation des risques du système traité, dans le but de montrer sa résistance aux attaques ou d'apporter une diminution des risques.
Les tests que nous avons réalisé pour cette audit peuvent être catégorisés de la façon suivante :

- *Audit* : Ordinateur citoyen
- *Audit* : Accès physique des serveurs
- *Audit* des canaux de communication
- *Audit* des serveurs
- *Audit* : Social engineering
- Etude des protocoles de sécurités choisit

Ces différents tests permettent d'avoir une vue globale de la sécurité du projet proposé, mais ne garantissent pas une sécurité 100% (cela n'existe pas). Cette réflexion sur les faiblesses potentielles ont été réalisées par Molina Pablo et Pietronigro Marc, autrement dit les développeurs de la solution auditée. Pour un résultat optimum, les tests de sécurité devraient être réalisés par d'autres personnes non-impliqués dans le développement du système.

Les tests ont pour but de contrôler si les *statements* suivants sont respectés :

1. Les *suffrages* exprimés électroniquement ne doivent pas pouvoir être interceptés, modifiés ou détournés.
2. Le contenu des *suffrages* exprimés électroniquement ne doit pas pouvoir être connu par des tiers avant le *dépouillement*.
3. Seules les personnes ayant le droit de vote doivent pouvoir prendre part au *scrutin*.
4. Chaque électeur ne dispose que d'une voix et ne peux voter qu'une seule fois.
5. En aucun cas, même pendant le *dépouillement*, il ne doit être possible de faire un lien entre un électeur et son *suffrage*.
6. Le site doit être en mesure de résister à une *attaque en déni de service* pouvant aboutir à la saturation du serveur.
7. L'électeur doit être protégé contre toute tentative de vol d'identité.
8. Le nombre de votes émis doit correspondre au nombre de votes reçus, toute différence doit pouvoir être expliquée et corrigée.
9. La preuve qu'un électeur a votée doit pouvoir être faite.
10. Le système n'accepte pas de vote électronique en dehors de la période d'ouverture du *scrutin* électronique.
11. Le bon fonctionnement du système doit pouvoir être vérifié par les autorités désignées à cet effet.

## Audit : Ordinateur citoyen

La sécurité des ordinateurs des différents citoyens ne peut être garantie, de part leur nombre et l'utilisation qu'ils en ont. C'est donc un élément à haut risque pour notre système de vote, pour lequel virus et malware en feront partie.
A noter que pour la thèse, il a été décidé que la sécurtié de l'ordinateur du citoyen ne sera pas prise en compte et considérée comme sûre. Même si le développement du système n'a donc pas été prévu pour sécuriser cette partie, nous allons tout de même la tester. Nous allons donc vérifier les différents moyens qu'un ordinateur infecté a pour manipuler les votes de quelques manières qui soit.
Parmi les onze *statements* que doit respecter le système d'E-voting, les *statements* suivant pourraient être mis en danger sur l'ordinateur du client :

- Les suffrages exprimés électroniquement ne doivent pas pouvoir être interceptés, modifiés ou détournés.
- Le contenu des suffrages exprimés électroniquement ne doit pas pouvoir être connu par des tiers avant le dépouillement.
- Seules les personnes ayant le droit de vote doivent pouvoir prendre part au scrutin.
- En aucun cas, même pendant le dépouillement, il ne doit être possible de faire un lien entre un électeur et son suffrage.
- L'électeur doit être protégé contre toute tentative de vol d'identité.

### *Système de démonstration (partie codé durant la thèse)*

Dans la version codée, la communication entre le client et le serveur de vote ou de récupération des codes d'accès, est réalisée via une application java qui communique avec le serveur via un canal non sécurisé.

Lors de la récupération des codes d'accès, un *login* et un *mot de passe* sont demandés au citoyen, ces informations sont des accès permanents pour le votant. Si ces informations venaient à être connues avant une phase de vote, alors un attaquant pourrait voter à la place de quelqu'un d'autre.
De plus un *Keylogger* permettrait aussi de connaître le contenu du vote avant le dépouillement (à petite échelle, puisqu'il faut un ordinateur infecté) mais aussi de faire le lien entre une personne et un vote (pseudonymie non garantie). En effet le *keylogger* peut récupérer le login et les codes d'accès de la personne et si le vote est réalisé sur un ordinateur infecté, à ces informations se rajoute le contenu du vote. Un lien entre le vote et son votant peut donc être réalisé.
Un virus à généralement le contrôle total de la machine, il est donc possible qu'il modifie le contenu du vote se trouvant dans la mémoire sans même que le votant ne s'en aperçoive.

Récapitulation des faiblesses :

- *Keylogger* → non respect des *statements* :
  - Seules les personnes ayant le droit de vote doivent pouvoir prendre part au *scrutin*.
  - L'électeur doit être protégé contre toute tentative de vol d'identité.
  - Le contenu des *suffrages* exprimé électroniquement ne doit pas pouvoir être connu par des tiers avant le *dépouillement*.
  - En aucun cas, même pendant le *dépouillement*, il ne doit être possible de faire un lien entre un électeur et son *suffrage*.
- Virus → non respect des *statements* :

- Les *suffrages* exprimés électroniquement ne doivent pas pouvoir être interceptés, modifiés ou détournés.

Une bonne éducation des citoyens sur la sécurité permettrait, non pas de faire disparaître ces vulnérabilités, mais de grandement les limités. En effet un bon anti-virus et anti-spyware permettent de limiter le risque d'infection.

### *Système final (proposition final)*

Cette version est une amélioration de la précédente, la récupération des codes se fait via un applet se trouvant sur un site certifié et sécurisé par *HTTPS*, la communication entre *l'applet* et le serveur est cryptée et *maccée* afin de garantir *l'intégrité* et la *confidentialité*. Les codes sont envoyés sous forme d'images traités de façon à les rendre illisibles par des logiciels mais lisibles pour le citoyen.
La faiblesse reste le *mot de passe* qui est permanent, même si comme la documentation *Système* le propose, il est possible de mettre des *Policy* obligeant ainsi le citoyen à changer de mot de passe tous les X temps. Cette solution n'est jamais mise en place car trop encombrante pour les utilisateurs. Il est donc possible grâce à un *Keylogger* de voter avec les accès d'un autre citoyen. Grâce au système d'image, il n'est par contre plus possible pour un logiciel de faire un lien entre le votant et son vote de façon automatique (pas de lien login-vote possible pour un logiciel). Un virus peut toujours changer le vote sans que le votant ne s'en aperçoive.
Récapitulation des faiblesses :

- *Keylogger* → non respect des *statements* :
  - Seules les personnes ayant le droit de vote doivent pouvoir prendre part au *scrutin*.
  - L'électeur doit être protégé contre toute tentative de vol d'identité.
  - Le contenu des suffrages exprimé électroniquement ne doit pas pouvoir être connu par des tiers avant le *dépouillement*.
- Virus → non respect des *statements* :
  - Les *suffrages* exprimés électroniquement ne doivent pas pouvoir être interceptés, modifiés ou détournés.

La menace que les codes d'accès soient récupérés par un attaquant grâce un *Keylogger* est moins grave que celle de la falsification par virus. En effet, si une personne vote avec les codes d'accès d'une autre, il y a de forte chance que la victime (votant) fasse remarquer le piratage lorsqu'elle essaiera de voter à son tour (instruction des votant à la sécurité). Le virus par contre est très dangereux parce qu'il travail dans l'ombre sans que personne ne s'en aperçoivent.
Comme dit plusieurs fois la sécurité de l'ordinateur d'un citoyen n'est pas prise en compte dans la solution d'E-voting proposé. Cette décision avait été prise pour des raisons de temps à disposition pour réaliser le travail.

## Audit : Accès physique des serveurs

Avoir un accès physique à une machine rime généralement avec avoir un accès et un contrôle total de cette machine. Il est en effet très facile de faire une élévation de privilège ou autre piratage si un accès physique est donné à un attaquant. C'est pourquoi les accès physiques aux machines sont restreints selon le protocole de sécurité proposé dans la version finale.
Parmi les onze *statements* que doit respecter le système d'E-voting, tous pourraient être mis en danger par une attaque provenant d'une personne ayant un accès physique. Le fait que les *statements* pourraient être mis en danger ne signifie pas que c'est le cas pour notre système mais simplement qu'un accès physique au serveur permet beaucoup de choses.

### *Système de démonstration (partie codé durant la thèse)*

Dans la version codée actuelle, les clefs privés et publiques servant à l'encryption, décryption, signature sont *hardcoder*. Ce n'est de loin pas une solution optimum, mais réalisée uniquement afin de pouvoir montrer une solution fonctionnant dans son ensemble malgré le peu de temps à disposition pour la réalisation. Aucun logiciel de code *obfuscator* n'a été appliqué à cette version, ce qui rendrait le reverse engineering très simple.
La sécurité de l'accès physique aux serveurs ne peut être jugée sur la version actuellement codée, puisque cette version est en cours de développement.

### *Système final (proposition final)*

Dans cette version, des protocoles de sécurité pour l'accès physique ont été définis, la gestion des clefs a été revu et la sécurité dans l'ensemble a été augmentée. Nous pouvons donc réfléchir aux différents types d'attaques possibles venant d'une personne ayant accès direct aux serveurs.
Si les protocoles définis sont respectés, aucune personne non – autorisée ne peut avoir un accès aux différents serveurs et ordinateurs servant aux systèmes de votation. Seul les administrateurs dûement autorisés ont un accès et ils ne peuvent pas y accéder seul (groupe de deux requis). Si malgré ces définitions de sécurité, deux administrateurs collaborent afin de pirater le système, ils se heurteront à certaines mesures qui les en empêcheront.
L'accès aux clefs est très difficile, puisque celles-ci se trouvent sur un matériel hardware externe, lequel requiert certains droits d'accès pour y accéder. Comme les administrateurs ne possèdent pas les droits *root*, mais des droits d'utilisateurs restreints leur permettant juste de lancer les scripts nécessaires (similaire au système des banques), ils ne peuvent installer de logiciels malveillants.
Si dans le but de réaliser une élévation de privilège un serveur est redémarré (boot cd afin de supprimer le mot de passe *root*) une alerte alors est automatiquement déclenchée et l'information transmise aux autorités compétentes, ce qui réduit les chances que de telles attaques aient lieu.
Si malgré tous les protocoles définis dans la section *Système* un pirate arrive tout de même à accéder aux différentes clefs privées et ainsi générer de faux votes signés proprement qu'il ajoute dans l'urne, ces votes ne seront pas pris en compte lors du décompte des votes puisqu'il est quasi impossible que l'*ID de référence* choisit correspondent à un existant.
La clef privée servant à la décryption des bulletins de vote se trouvant dans un coffre fort. Il est impossible de décrypter les votes avant le dépouillement. Si un virus, qui aurait pour but de manipuler les votes décryptés, venait à être installé sur la machine de dépouillement, cette manipulation serait détectée grâce au système de ré-encodage/*hash value* (voir section *Système*).

## Audit des canaux de communication

Les canaux de communications représentent un danger pour les données qui y transitent. Il est en effet possible que ces données soient lues, modifiées, altérées,… par des personnes mal intentionnées. C'est pourquoi une attention toute particulière est donnée pour cette section de la sécurité.
Les points principaux pouvant être mis en danger lors des différentes communications :

- Les suffrages exprimés électroniquement ne doivent pas pouvoir être interceptés, modifiés ou détournés.
- Le contenu des suffrages exprimés électroniquement ne doit pas pouvoir être connu par des tiers avant le dépouillement.
- Seules les personnes ayant le droit de vote doivent pouvoir prendre part au scrutin.
- Chaque électeur ne dispose que d'une voix et ne peux voter qu'une seule fois.
- En aucun cas, même pendant le dépouillement, il ne doit être possible de faire un lien entre un électeur et son suffrage.

### *Système de démonstration (partie codé durant la thèse)*

Cette version du projet *E-voting* comprend de l'encryption et des signatures pour quasi toutes les communications réalisées. Les seules choses manquantes sont les tunnels sécurisés visant à empêcher les *reply attack* et à augmenter la *confidentialité*. A noter que lors de la récupération des *codes d'accès*, ceux-ci sont envoyés signés mais en *clear text* (tunnel sécurisé non – implémenté par manque de temps).
Le fait qu'aucune protection contre les *reply attack* ne soit implémentée, il serait possible de réaliser plusieurs fois le même vote en captant et en envoyant un paquet à *l'Urne*. Lors du décomptage, cette manipulation serait détectée grâce aux *ID de références* et serait aussi tôt corrigée. De la même façon une *reply attack* serait envisageable lors de la phase de récupération des codes d'accès (*reply* envoyé lors de la prochaine session de vote, à condition que le mot de passe n'ait pas changé).
L'attaque passive par *sniffing* permettrait de récupérer les codes d'accès demandés par l'utilisateur puisque ceux-ci sont envoyés en *clear text*. Les certificats x509 joint dans les document *XML* sont auto-signés ce qui ne permet pas un contrôle rigoureux de l'intégrité des paquets et ouvre ainsi une possibilité d'attaque par *forging* de paquet.

### *Système final (proposition final)*

Cette version comprend des canaux de communication sécurisés pour chaque échange réalisé, ces cannaux sécurisés viennent compléter la sécurité faîtes grâce à L'*XML* encryption et signature. Le système se voit ainsi protégé contre tout type de *reply attack* (les canaux sécurisés implémentent des *time stamp*). Le *forging* est quand à lui impossible sans avoir accès aux différentes clefs privées servant aux signatures.
Les systèmes de sécurité suivants ont pour but de protéger les onze *statements* :

- Les suffrages exprimés électroniquement ne doivent pas pouvoir être interceptés, modifiés ou détournés.
    a. Protégé grâce aux signatures et aux canaux sécurisés servant à chaque transmission
- Le contenu des *suffrages* exprimés électroniquement ne doit pas pouvoir être connu par des tiers avant le *dépouillement*.

b. Protégé grâce à *l'end-to-end* encryption, faîte depuis l'ordinateur du citoyen via la clef publique de l'urne.

- Seules les personnes ayant le droit de vote doivent pouvoir prendre part au *scrutin*.
  c. Le *forging* de paquets n'étant pas possible, seules les personnes ayant récupéré leurs codes d'accès ont le droit de vote.
- Chaque électeur ne dispose que d'une voix et ne peux voter qu'une seule fois.
  d. Les *reply attack* n'étant pas possible grâce aux canaux sécurisés réalisés, les votants ne peuvent voter qu'une fois.
- En aucun cas, même pendant le *dépouillement*, il ne doit être possible de faire un lien entre un électeur et son *suffrage*.
  e. La confidentialité fait par l'encryption de chaque communication et la *pseudonymie* réalisée grâce aux codes d'accès, empêchent toutes personnes de savoir qui à voter quoi.

## Audit des serveurs

Un serveur requiert un certain nombre de points à respecter. Il doit en effet être résistant aux grands nombres de connexions simultanées et aux attaques de *Denial Of Service*. La sécurité présente sur l'ordinateur doit et est irréprochable, de solides logiciels de protection se doivent d'être installés, mis à jour et configurés correctement.
Les points principaux pouvant être mis en danger coté serveur :

- Le site doit être en mesure de résister à une attaque en déni de service pouvant aboutir à la saturation du serveur.
- Le système n'accepte pas de vote électronique en dehors de la période d'ouverture du scrutin électronique.
- Le bon fonctionnement du système doit pouvoir être vérifié par les autorités désignées à cet effet.

### *Système de démonstration (partie codé durant la thèse)*

Comme la version actuellement codée et développée ne se trouve pas sur un ou des serveurs, il est impossible de réaliser un audit.

### *Système final (proposition final)*

Si les serveurs choisis pour héberger le système de vote sont suffisamment résistants et configurer de façon à détecter et prévenir les attaques de Denial *Of Service*, les serveurs auront une sécurité acceptable.

## Audit : Social engineering

Le social engineering est un type d'attaque qui n'a de limite que l'imagination et les talents d'acteurs de l'attaquant. Le seul moyen de lutter contre ce type d'attaque est la formation du personnel.
Les sections marche à suivre sont là pour faire une partie de cette éducation, l'autre partie doit venir de l'institution, en envoyant les personnes suivre des cours sur ce sujet.
Les chances d'attaques par *Man-In-The Middle* sont réduites grâce à l'éduction réalisées aux votants (contrôle de l'hash value du certificat avant de voter ou récupérer les codes d'accès).

## Etude des protocoles de sécurités choisis

Dans cette section nous allons vous lister tous les protocoles choisies ainsi que les raisons de leur choix.
Nous avons choisi **RSA** comme algorithme de signature car nous l'avons étudié et nous savons que nous pouvons nous fier à son implémentation. C'est également un standard très fréquemment utilisé. Dans notre cas seul la longueur de la clef pourrait être discutée qui est de 1024 bits (un standard dans les *XML* signature).

Le choix d'**AES** a été simple, c'est actuellement le standard cryptographique dans énormément de domaine (NIST aux USA). Il est pour le moment resté incassable seul le brute force est envisageable.

**SHA-256** est notre algorithme de hachage, nous l'avons choisi à la place de *MD5* et *SHA1*, car il est plus sûr et le *one-way function* est plus respecté.

**SHAPRNG1** implémenter par SUN est utilisé pour générer des nombres pseudo aléatoires forts.

# Marche à suivre du client

Cette section regroupe les conditions que le client doit remplir afin de voter avec la plus grande sécurité possible. Elle contient des explications sur ce que le client doit faire ou ne pas faire et la marche à suivre étape par étape pour voter. Il a pour but d'instruire l'utilisateur du point de vue de la sécurité informatique.

## Protection du système client

Afin que la machine soit le moins vulnérable possible, nous recommandons d'installer un anti-virus ainsi qu'un anti-spyware sur l'ordinateur du client. Ils doivent bien sur être à jour, et la machine doit être désinfectée régulièrement. Si vous ne nous connaissez pas de tels logiciels contactez votre administrateur.
Il est recommandé que votre système d'exploitation soit également à jour et que vous soyez protégé derrière un pare-feu.

## Télécharger l'applet

Pour se connecter à la page de téléchargement, il ne faut en aucun cas cliquer sur un lien mais bien entrer l'adresse soi-même à la main dans votre explorateur. Cela empêche les attaques de type *fishing.*
Une fois arrivé sur la page de téléchargement, le certificat doit être vérifié afin d'être sûr qu'on se trouve sur la bonne page (plus d'explication dans la section *Vérifier un certificat*).

Tous ces contrôles effectués, on peut s'identifier pour récupérer les informations nécessaires à la prochaine étape.

## Récupérer les informations pour voter

L'applet demande le nom d'utilisateur et le mot de passe de l'école afin de garantir votre autorisation. Il est conseillé de changer son mot de passe régulièrement pour augmenter sa sécurité. Une fois les informations récupérées il est possible de voter dans l'immédiat ou de les sauvegarder.

### *Sauvegarder les informations*

Pour sauvegarder les informations deux techniques vous sont proposée :

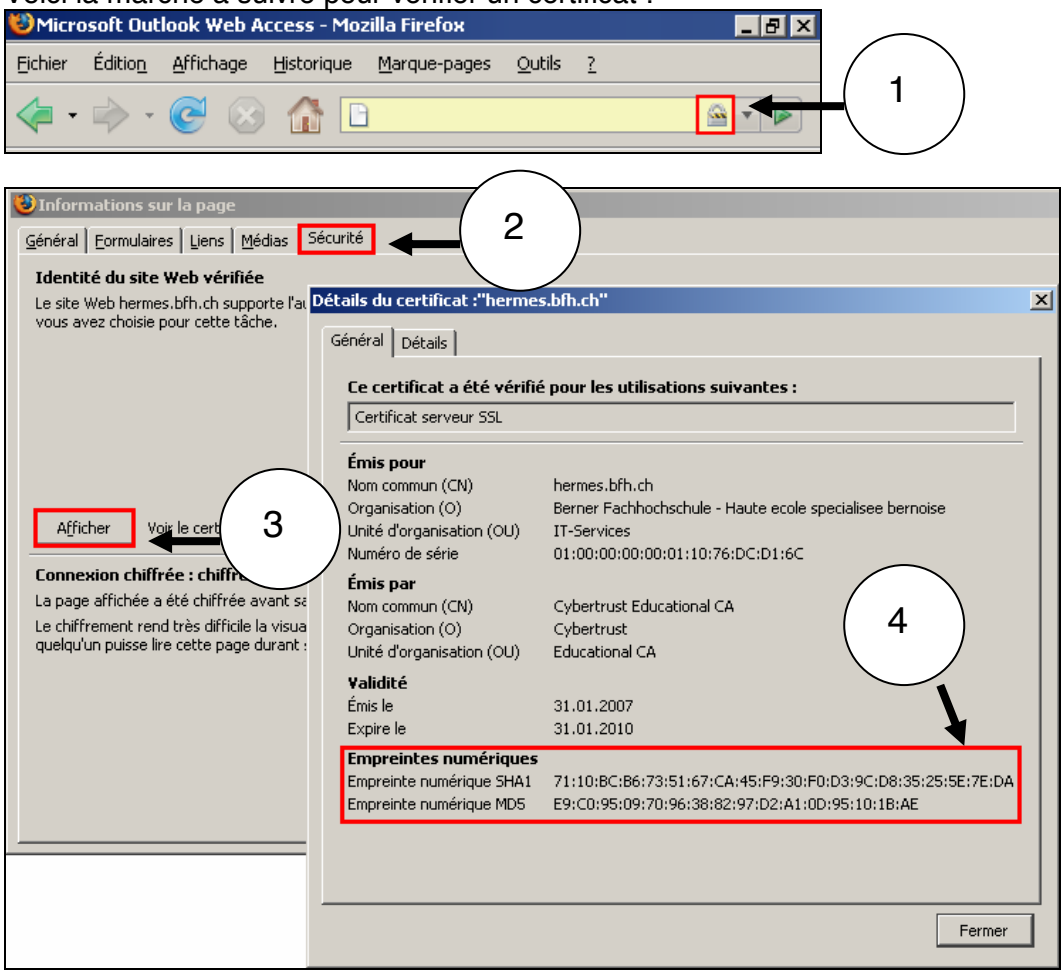
1. Ecrire les informations sur un morceau de papier, le garder sur soi et le détruire après avoir voté.
2. Le copier dans un fichier texte et utiliser un programme de cryptage pour protéger le contenu du fichier.

## Voter

Après s'être identifié sur *l'applet* et avoir récupéré les informations de vote, il est possible de se connecter à la *page de votation* et de re-télécharger un nouvel applet qui permettra de voter (suivez toujours les consignes de la section *télécharger l'applet*). Il est également permis de sauvegarder les informations pour une future utilisation. Attention, car le temps de vote est limité, veillez à ne pas dépasser ce délai, car une fois dépassé il est impossible de voter.
Il est fortement recommandé de voter dans l'immédiat afin que ces informations ne soient pas récupérées par un tiers ou perdues. Dans le cas d'une perte, il est **impossible** de voter !

## Vérifier un certificat

Voici la marche à suivre pour vérifier un certificat :




- Se connecter au site Internet.
- Cliquer sur le cadenas **( 1 )**.
- Sélectionner l'onglet *Sécurité* **( 2 )** puis cliquer sur *Afficher* **( 3 )**.
- Comparer les empreintes numériques **( 4 )**. Dans le cas où elles sont différentes, contacter le numéro d'urgence*.

## Choses à ne pas faire

Il y plusieurs choses à ne pas faire afin de ne pas compromettre la sécurité. Voici une petite liste :

- Ne pas respecter un des points cité ci-dessus.
- Ne pas cliquer sur des liens qui indiqueraient le site de vote.
- Ne jamais donner son login et son mot de passe. Un administrateur ne vous demandera jamais votre mot de passe.
- Ne pas remplir des formulaires où l'on vous demanderait votre login et votre mot de passe.
- Si le vote n'est pas réalisé correctement conserver les informations de votes dans un endroit sûr.
- Ne pas faire voter un tiers à sa place. Voter soi-même
- Ne pas vendre son droit de vote.
- Ne pas insérer les informations récupérées ailleurs que dans l'applet prévu à cet effet.

**Actuellement aucun numéro d'urgence n'a été défini.*

# Marche à suivre de l'école

Ce document contient les informations nécessaires à l'école afin de générer les codes correctement, un protocole y est décrit.

## Génération des codes

La génération des codes d'accès doivent se faire sur un ordinateur « saint ».
Il doit remplir plusieurs conditions :

- Il ne doit être connecté au réseau que lors de la génération des codes.
- Au moment de la génération, il doit être protégé derrière une infrastructure munie d'un *pare-feu* et d'un *IDS*.
- L'anti-virus, l'anti-spyware et les mises à jour de système d'exploitation doivent être d'actualité.
- Les personnes utilisant cet ordinateur doivent être formées pour son utilisation.
- Avant et après la génération il doit être rangé dans un lieu sûr (coffre-fort).
  Certaines informations sont paramétrables comme le *range* des *ID*'s et du *PIN* (se reporter à la section *Système*).

## Informations sur les serveurs

Les serveurs *Register*, *Urne*, *VoteServer* ne doivent être visible que lors de la période de vote. La période dépassée, leur accès depuis Internet doit être bloqué (par une pare-feu ou simplement débranché).
Leur accès physique doit être contrôlé et hors d'accès des personnes non-autorisées. Dans le meilleur des cas, il faut au moins deux administrateurs pour accéder aux machines. Pour éviter tout types de modifications, les administrateurs n'auront que peu de droits. Pour avoir plus de droit il faudra suivre un protocole (se reporter à la section *Système*).

## Information utilisateurs

Il est du devoir de l'école d'informer et d'éduquer les utilisateurs à l'utilisation du système (la section *marche à suivre client* contribue à leur formation).
Le *mot de passe* que les utilisateurs possèdent pour s'identifier ainsi que celui pour se connecter au serveur doit être **fort**, c'est-à-dire au moins 8 caractères contenant des majuscules, des minuscules ainsi que des symboles et des numéros.

## Conclusion

Après ces mois de développement et de réflexions, nous sommes globalement content de notre projet, même si nous voulions, au départ, pousser son implémentation plus loin que son état actuel.

Nous avons trouvé dans ce projet, les éléments que nous recherchions au départ. L'apprentissage d'une nouvelle technologie, que nous n'avions encore jamais traité autant en détails : *les web services.*
Le stress d'un projet, se fixer un but à atteindre avec tous les contre temps rencontrés et les délais à respecter. Le travail de groupe, car nous étions quatre au départ à faire des recherches sur le sujet et cela à demandé de la coordination et de la communication. Innover, devoir développer quelque chose qui n'était pas encore mis en place dans cette école et faire appel à notre imagination.

Nous n'avons pas pu réaliser un projet égal à notre idée de départ pour plusieurs raisons. La première et celle qui nous aura fait perdre le plus de temps : l'outil de développement. L'utilisation de *Netbeans* nous a posé passablement de problèmes, d'une part le changement de version de 6.0 à 6.1 avec tous les bugs rencontrés avec le *svn* ou le rafraichissement des références entre web service. D'autre part nous avons rencontré énormément de problème avec le serveur d'application *Glassfish*. Cela nous a beaucoup handicapé dans les tests et a passablement retardé l'implémentation du projet.
Deuxièmement, le fait de devoir travailler avec un double groupe au départ (deux groupes de deux personnes) n'a pas favorisé l'avancement, nous avions des plannings très différents et trouver des dates rapprochées pour les réunions n'étaient pas choses faciles, nous manquions également d'expérience dans se genre de tâche.

Si nous devions refaire ce projet, nous commencerions plutôt à implémenter afin d'avoir plus de temps pour réagir lors de problèmes qui nous étaient inconnus et inexplicables. Nous réfléchirions aussi aux différentes plates-formes de développement disponible afin de faire un choix plus judicieux.

Intégralement, le projet nous aura beaucoup apporté et nous avons eu beaucoup de plaisir à le faire. Nous trouvons le thème *E-voting* extrêmement intéressant avec une grande perspective d'avenir et nous avons apprécié faire des recherches dans ce domaine. Nous sommes restés un peu sur notre faim car nous n'avons pas pu développer le projet dans son intégralité. Couper le projet en deux étaient nécessaires vu le peu de temps que nous avions à disposition.

## Remerciements

Nous tenons particulièrement à remercier les professeurs Eric Dubuis, Gerhard Hassenstein et Endre Bangerter pour leur soutien, leur aide et leurs conseils apportés tout au long du projet.
Nous voulons également remercier les responsables des communes de Bienne et de Moutier pour leurs accueils et leurs explications lors de notre phase de recherche.

## Spécification supplémentaires

Spécification de Java 5 ou 6 sont a respecter afin de faire tourner un applet.
Il faut un serveur de base de données, web. Il faut un système de certificat (https, encryptions à clef publique).

## Glossaire

| | |
|---|---|
| *AccessInfo* | Server qui permet de récupérer les codes d'accès |
| *AES* | Algorithme d'encryption symétrique |
| *Applet* | Application java pour Internet |
| *Attaque en déni de service, Denial of service* | Attaque visant à rendre une application informatique incapable de répondre aux requêtes des utilisateurs. |
| *Audit* | Ensemble de test visant à contrôler la sécurité d'un système |
| *Clé primaire, primary key* | Identifiant de la table dans une base de données |
| *Code de contrôle* | Code affiché sur l'image qui permet au votant d'identifier le serveur |
| *Codes d'accès* | Ensemble des informations permettant à la personne de s'identifier et de valider son vote (*ID Unique*, *PIN*). |
| *Déloger* | Le votant n'est plus identifié sur le système. |
| *Dépouillement* | Phase de comptage des bulletins de votes |
| *Donnée personnel* | Informations personnels du votant (date de naissance, commune, …) |
| *Encryption* | Rend un texte impossible à comprendre. |
| *End-to-end* | Encryption du début à la fin. De l'utilisateur à l'urne. |
| *Entity* | Entité |
| *E-voting* | Vote en ligne sur Internet. |
| *Factory pattern* | Description d'une architecture java |
| *Fingerprint* | Empreinte unique d'un certificat. |
| *firewall, pare-feu* | Unité filtrant des paquets et protégeant un système informatique. |
| *Forging* | Création. (*Forging de packet*, création de paquet) |
| *GUI* | Interface graphique |
| *Hash value* | Fonction mathématique transformant un texte en une valeur. Le sens inverse valeur → texte. |
| *Https* | Protocole Internet sécurisé. |
| *ID Unique* | Numéro d'identification unique pour chaque votant. |
| *IDS* | Système de détection d'intrusions |
| *Intégrité, integrity* | Contenu non modifié |
| *Key logger* | Programme enregistrant les frappes du clavier |
| *Login* | Identifiant connue par le votant et le serveur. |
| *Maccée* | Crée un code d'authentification de message. Garantit l'intégrité |
| *Malware* | Programme malveillant |
| *Mot de passe, password* | Code de sécurité |
| *Numéro de contact :* | Numéro de téléphone à appeler en cas de problème. |
| *Obfuscator* | Programme empêchant de retrouver le code source |
| *Open Source* | Possibilité de libre redistribution et d'accès au code source. |
| *PIN* | Nombre secret permettant de confirmer le vote |
| *Policy* | Règle de sécurité |
| *Pseudonymie* | Anonymat via des pseudonymes |
| *Register* | Server du registre |
| *Replay attack* | Attaque utilisant le renvoie de paquet |

| | |
|---|---|
| *RSA* | Algorithme de signature asymétrique |
| *SAML* | Standard qui définit un protocole d'échange d'information lié à la sécurité. |
| *Script* | Code réalisant des tâches prédéfinies. |
| *Scrutin :* | Bulletin de vote |
| *Serveur LDAP :* | Serveur où sont stockés les informations des utilisateurs |
| *SHA* | Protocole qui crée une valeur digest |
| *Shared secret :* | Mot de passe secret partagé entre le serveur et le votant. Sert à encrypter. |
| *Signature* | Principe pour garantir l'intégrité. |
| *SOAP* | Permettant la transmission de messages XML entre objet distant. |
| *Social engineering* | Attaque visant l'utilisateur d'un système. |
| *SSL* | Protocole de sécurisation d'échange sur Internet |
| *Suffrages :* | Déclaration d'opinion dans une élection. |
| *Time stamp* | Marqué le temps sur un message pour vérifier sa fraicheur. Il est signé. |
| *Trust* | Système de confiance. |
| *Url unique de vote* | Adresse du site web unique pour chaque votant. |
| *Urne* | Server où sont stockés les votes |
| *User friendly* | Simple à prendre en main par un utilisateur. |
| *Verysign* | Entreprise de création de certificat possédant des trust's serveur |
| *VoteServer* | Serveur qui communique avec le client. Il s'occupe de parler avec l'urne et le registre. |
| *X509* | Définit un standard de certificat électronique. |
| *XKMS* | Protocole d'enregistrement et de validation des clefs |
| *XML* | Langage de balisage extensible, utilisé comme standard pour créer des documents |

## Sources

**Système de Genève :**
http://www.geneve.ch/evoting/
http://www.geneve.ch/evoting/doc/rapports/200409_rapport_carouge_meyrin.pdf
http://www.geneve.ch/evoting/doc/rapports/rapport_version_internet.pdf

**Informations sur les élections bernoises :**
http://www.sta.be.ch/site/fr/wahlenabstimmungen-wahlen06-anleitung

**Informations sur le vote électronique Français :**
http://www.sciences.univ-nantes.fr/info/perso/permanents/enguehard/perso/RI_halshs-00085041.pdf
http://membres.lycos.fr/isabellebreil/vote.html

**SSL / TLS :**
http://www.securiteinfo.com/cryptographie/ssl.shtml
https://wsit-docs.dev.java.net/releases/m6/Overview2.html#wp149787

**Secure Random :**
http://java.sun.com/j2se/1.4.2/docs/api/java/security/SecureRandom.html
http://www.developerzone.biz/content/view/95/36/

**Tutoriel gestion des clefs, signature et encryption :**
http://java.sun.com/docs/books/tutorial/security/apisign/step2.html
http://exampledepot.com/egs/java.security/GenKeyPair.html
http://java.sun.com/developer/technicalArticles/xml/dig_signatures/
http://java.sun.com/webservices/docs/1.6/tutorial/doc/XMLDigitalSignatureAPI8.html
http://72.5.124.55/webservices/docs/2.0/xmldsig/api/javax/xml/crypto/doc-files/X509KeySelector.java

**Tutoriel netbeans :**
http://www.netbeans.org/kb/60/websvc/wsit.html
http://www.netbeans.org/kb/60/websvc/jax-ws.html
http://www.netbeans.org/kb/60/websvc/client.html
http://www.netbeans.org/kb/60/web/web-jpa.html
http://www.netbeans.org/kb/60/web/mysql-webapp.html
http://ui.netbeans.org/usability/java_ee_March_2006/ejb30-persistence-one-pager.pdf

**Tutoriel LDAP :**
http://forum.topflood.com/flood-site/tutorial-ldap-3888.html

**Parsing XML :**
http://www.informit.com/articles/article.aspx?p=31349&seqNum=3

Tutoriel wsit et java :
http://java.sun.com/webservices/reference/tutorials/wsit/doc/
http://java.sun.com/webservices/docs/1.6/tutorial/doc/index.html

**Documentation papier :**
Secure and Easy Internet Voting par Giampiero E.G Beroggi

An anonymous Electronic Voting Protocol for Voting Over The Internet par Indrajit Ray et Indrakshi Ray

**Personnes :**
Prof. Dr. Eric Dubuis
Prof. Gerhard Hassenstein
Prof. Dr. Endre Bangerter

## Annexes Sommaires

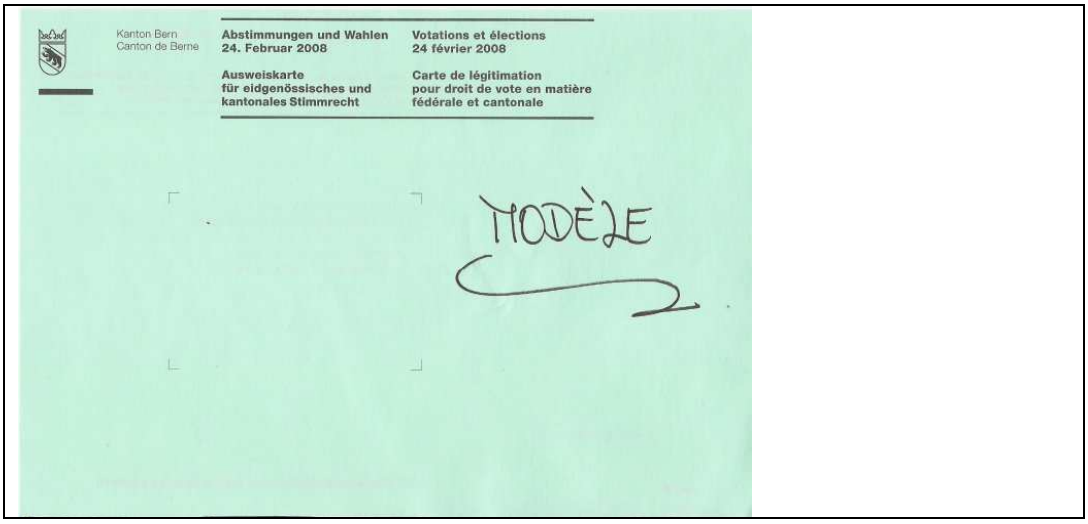
Votation papier du canton de Berne. ..... 74
Illustration du matériel de vote ..... 74
Fonctionnement des votations ..... 76
Sécurités mises en œuvres ..... 76
Attaques possibles ..... 77
Comparaison avec le système Français ..... 77
Fonctionnement des votations Française ..... 77
Fonctionnement de la machine de vote (système Français) ..... 78
Système Français plus sûre ? ..... 78
E-voting Genève ..... 80
Fonctionnement globale ..... 80
Points de sécurité ..... 81
L'architecture des serveurs est la suivante : ..... 82
Les 11 conditions en détails ..... 82
Internet et la résolution des noms de domaine ..... 83
Faq sur le site Genevois : ..... 85
Analyse système physique ..... 86
Vision ..... 86
Autre nécessité et contraintes ..... 86
Acteurs ..... 87
Use Cases ..... 88
Use Case UC1 : Génération des codes d'accès ..... 89
Main Success Scenario ..... 89
Question ..... 89
Use Case UC2 : Distribution des codes d'accès ..... 90
Main Success Scenario ..... 90
Extension ..... 90
Question ..... 90
Use Case UC3 : Obtention des codes d'accès ..... 91
Main Success Scenario ..... 91
Extension ..... 91
Use Case UC4 : Identification ..... 92
Main Success Scenario ..... 92
Extension ..... 92
Question ..... 92
Combien de temps faut t'il bloquer la machine ? ..... 92
Use Case UC5 : Permission ..... 93
Main Success Scenario ..... 93
Extension ..... 93
Use Case UC6 : Vote ..... 94
Main Success Scenario ..... 94
Extension ..... 94
Use Case UC7 : Confirmation ..... 95
Main Success Scenario ..... 95
Spécification supplémentaires ..... 96
Evénements système ..... 97
Génération des codes ..... 97
Envoi ..... 97
Identification ..... 97
Permission ..... 97
Votation ..... 97
Cas générique ..... 98

Fonctionement général (étape par étape) ..... 98
Erreur d'envois de l'email ..... 98
Feuille de sécurité restante au secrétariat à l'ouverture des votes ..... 98
Domain model ..... 99
Concepts ..... 100
Associations ..... 101
Informations supplémentaires ..... 102
Point de vu du système ..... 102
Première base de données ..... 102
Deuxième base de données (registre de vote) ..... 102
Urne ..... 102
Création et transfert de la feuille de sécurité ..... 102

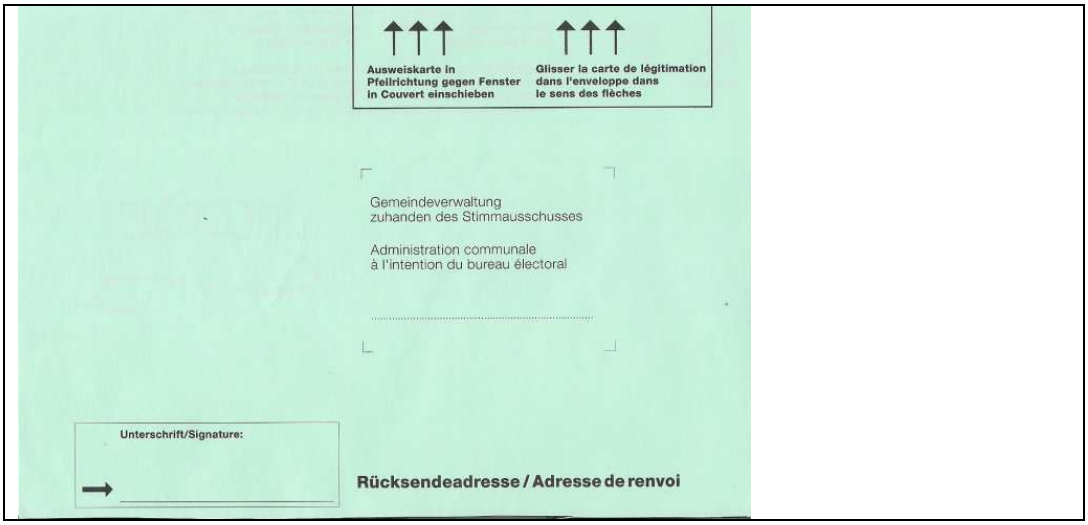
## Votation papier du canton de Berne.

Afin d'être au courant des technologies utilisées dans les votations sur papier du canton de Berne, nous sommes allés nous renseigner dans les chancelleries de Bienne et de Moutier.
Nous y avons demandés le fonctionnement des votations et quels types de méthodes y sont utilisés afin de rendre ce système de vote sûr.

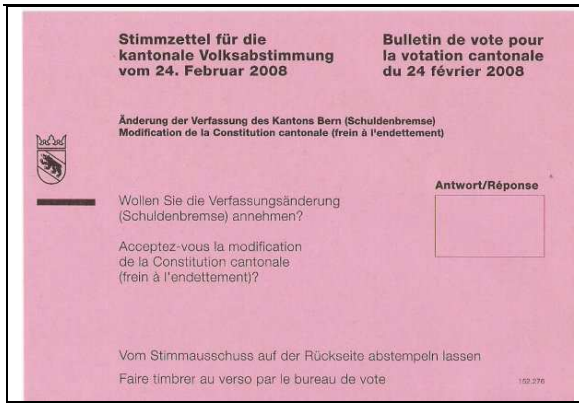
## Illustration du matériel de vote




*Carte de légitimation recto*




*Carte de légitimation verso*



Stimmzettel für die kantonale Volksabstimmung vom 24. Februar 2008

Bulletin de vote pour la votation cantonale du 24 février 2008

Änderung der Verfassung des Kantons Bern (Schuldenbremse)
Modification de la Constitution cantonale (frein à l'endettement)

Antwort/Réponse

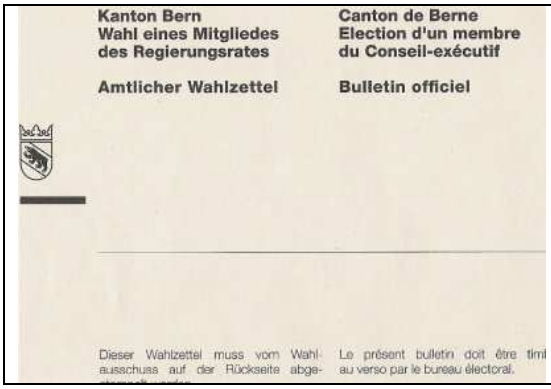
Wollen Sie die Verfassungsänderung (Schuldenbremse) annehmen?

Acceptez-vous la modification de la Constitution cantonale (frein à l'endettement)?

Vom Stimmausschuss auf der Rückseite abstempeln lassen

Faire timbrer au verso par le bureau de vote

*Sujet soumis à votation*



Kanton Bern
Wahl eines Mitgliedes des Regierungsrates

Amtlicher Wahlzettel

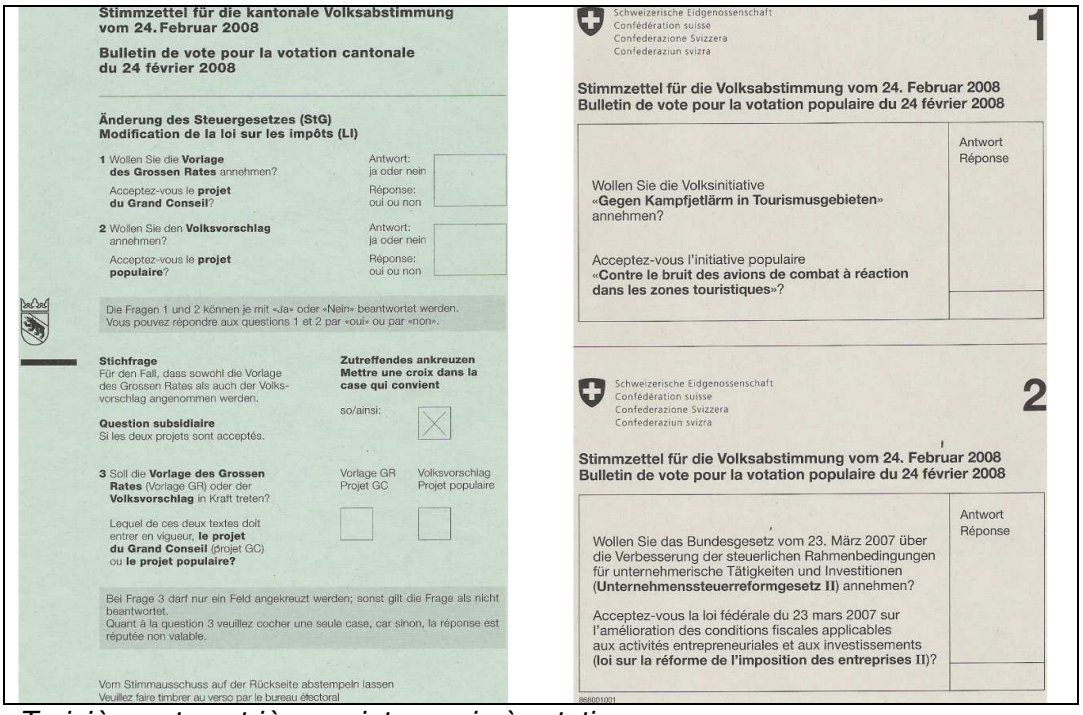
Canton de Berne
Election d'un membre du Conseil-exécutif

Bulletin officiel

Dieser Wahlzettel muss vom Wahlausschuss auf der Rückseite abgestempelt werden.

Le présent bulletin doit être timbré au verso par le bureau électoral.

*Deuxième sujet soumis à votation*



Stimmzettel für die kantonale Volksabstimmung vom 24. Februar 2008

Bulletin de vote pour la votation cantonale du 24 février 2008

Änderung des Steuergesetzes (StG)
Modification de la loi sur les impôts (LI)

1 Wollen Sie die **Vorlage des Grossen Rates** annehmen?
Acceptez-vous le **projet du Grand Conseil**?
Antwort: ja oder nein
Réponse: oui ou non

2 Wollen Sie den **Volksvorschlag** annehmen?
Acceptez-vous le **projet populaire**?
Antwort: ja oder nein
Réponse: oui ou non

Die Fragen 1 und 2 können je mit «Ja» oder «Nein» beantwortet werden.
Vous pouvez répondre aux questions 1 et 2 par «oui» ou par «non».

**Stichfrage**
Für den Fall, dass sowohl die Vorlage des Grossen Rates als auch der Volksvorschlag angenommen werden.

**Question subsidiaire**
Si les deux projets sont acceptés.

**Zutreffendes ankreuzen**
**Mettre une croix dans la case qui convient**
so/ainsi:

3 Soll die **Vorlage des Grossen Rates** (Vorlage GR) oder der **Volksvorschlag** in Kraft treten?
Lequel de ces deux textes doit entrer en vigueur, **le projet du Grand Conseil** (projet GC) ou **le projet populaire?**

Vorlage GR / Projet GC
Volksvorschlag / Projet populaire

Bei Frage 3 darf nur ein Feld angekreuzt werden; sonst gilt die Frage als nicht beantwortet.
Quant à la question 3 veuillez cocher une seule case, car sinon, la réponse est réputée non valable.

Vom Stimmausschuss auf der Rückseite abstempeln lassen
Veuillez faire timbrer au verso par le bureau électoral

Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

1

Stimmzettel für die Volksabstimmung vom 24. Februar 2008
Bulletin de vote pour la votation populaire du 24 février 2008

Antwort
Réponse

Wollen Sie die Volksinitiative «**Gegen Kampfjetlärm in Tourismusgebieten**» annehmen?

Acceptez-vous l'initiative populaire «**Contre le bruit des avions de combat à réaction dans les zones touristiques**»?

Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

2

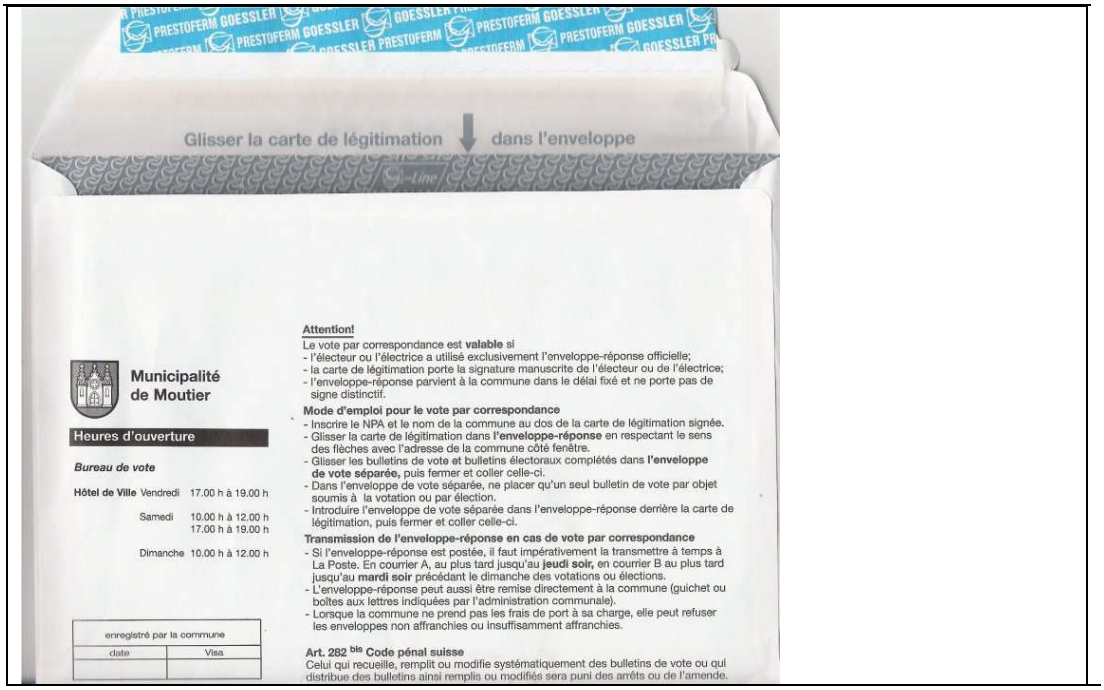
Stimmzettel für die Volksabstimmung vom 24. Februar 2008
Bulletin de vote pour la votation populaire du 24 février 2008

Antwort
Réponse

Wollen Sie das Bundesgesetz vom 23. März 2007 über die Verbesserung der steuerlichen Rahmenbedingungen für unternehmerische Tätigkeiten und Investitionen (**Unternehmenssteuerreformgesetz II**) annehmen?

Acceptez-vous la loi fédérale du 23 mars 2007 sur l'amélioration des conditions fiscales applicables aux activités entrepreneuriales et aux investissements (**loi sur la réforme de l'imposition des entreprises II**)?

*Troisième et quatrième sujet soumis à votation*




*Enveloppe d'envoi :*

## Fonctionnement des votations

Lorsqu'une votation est annoncée, des personnes sont engagées afin d'envoyer les enveloppes aux citoyens ayant droit de vote. Ces enveloppes contiennent cinq éléments : *La carte de légitimation*, *les bulletins de vote*, *une enveloppe de vote*, *une enveloppe d'expédition* et des *documents explicatifs* sur les votations en cours.

- Carte de légitimation : Elle contient les données personnelles et un champ qui doit être signé par le votant au verso. La carte de légitimation se met dans l'enveloppe d'expédition.
- Bulletins de vote : Ils contiennent les questions des votations en cours ou des noms de personnes à être élues. Ils doivent être remplis selon un schéma strict afin d'être valide. Ils sont à mettre dans l'*enveloppe de vote*.
- Enveloppe de vote : Elle contient uniquement *les bulletins de vote* (pas de *carte de légitimation*). Elle permet de séparer l'identité de la personne de ces votes. Elle est faite de telle façon qu'il est impossible de voir à l'intérieur une fois l'enveloppe fermée. Sur cette enveloppe aucun nom de *votant* ne doit figurer.
- Enveloppe d'expédition : Cette enveloppe est la seule à être renvoyée. Elle contient la *carte de légitimation* et l'*enveloppe de vote*.
- Documents explicatifs : Ils contiennent toutes les explications concernant le vote, les articles, les lois en votation et/ou la présentation des parties.

## Sécurités mises en œuvres

La sécurité des votations est assez faible, nous avons été surpris lors de son analyse. Aucune identité n'est contrôlée lors du dépouillement. La seule pièce qui fait office d'identification est la *carte de légitimation*, qui est loin d'être autant sécurisée qu'un passeport ou qu'un billet de banque. La copie n'est pas une chose compliquée, les paramètres à reproduire sont la *carte de légitimation*, les *enveloppes de vote et d'expédition* (qualité du papier, couleur et format). Un autre problème est qu'aucun registre contenant les personnes ayant voté n'est tenu. Par

contre un contrôle qui est fait de temps en temps, est l'appel d'une cinquantaine de personnes afin de vérifier leur signature pour être sûr qu'elles correspondent à celle qui est sur la *carte de légitimation*.

## Attaques possibles

Toutes les attaques possibles à grandes échelles sont basées sur le vote par correspondance via la reproduction illégale du matériel de vote.
Premier cas : L'attaqueur acquière des enveloppes vierges similaires aux *enveloppes d'expédition* et de *vote*. Grâce à une imprimante, il reproduit à l'identique celles utilisées officiellement. Il se procure également un papier de couleur et de qualité semblable à celui utilisé pour créer la *carte de légitimation* et à nouveau, grâce à une imprimante et un base de données qui contient des noms non-existants, crée des fausses *cartes de légitimations*. Comme aucun registre des personnes ayant le droit de vote n'est tenu, il est possible de faire voter des personnes non-existantes, donc de fausser la votation. L'attaqueur doit tout de-même signer toutes les *cartes de légitimation* à la main, une deuxième contrainte est que le taux de participation ne doit pas dépasser les 100%.

Deuxième cas : Un proche ou un parent ayant accès au matériel de vote d'une personne peut voter pour lui en imitant ou en inventant une signature sans forcément lui demander son avis.

On remarque que les attaques sont basées sur le vote par correspondance et que le fait de tenir un registre avec les personnes ayant le droit de vote limiterait voir supprimerait le risque de tricherie à grande échelle.

## Comparaison avec le système Français

Le système de votation français est bien différent de celui réalisé dans le canton de Berne. Comme nous allons le voir, il est plus stricte et sécurisé sur certains points, mais ouvre des portes à d'autres failles.

## Fonctionnement des votations Française

Les citoyens Français ayant le droit de vote possèdent une *carte d'électeur* valable durant un certain laps de temps (5ans en général). Cette carte contient les données personnelles du votant :

- Nom et prénom
- Adresse
- Date de naissance
- Commune de naissance
- Lieu de vote
- Département
- Signature du maire
- Signature du titulaire
- Tampon de la mairie
- N°du bureau de vote

Pendant les votations la personne désirant voter doit se présenter en personne (sauf si procuration), présenter sa *carte d'électeur*, sa *carte d'identité* et signer un *registre* afin de pouvoir aller voter. Ce *registre* permet de savoir qui a voté.
Pour voter le citoyen Français a le choix entre la *machine électronique de vote* et le vote normal par scrutin mis dans une enveloppe puis dans l'urne, à noter que le vote par correspondance a été interdit en 1975 pour des raisons de sécurité. En effet trop de fraudes étaient réalisées via ce système, il a donc été supprimé.

## Fonctionnement de la machine de vote (système Français)

Une fois que le votant a passé les formalités du contrôle d'identité, il indique si il souhaite voter de façon électronique ou normale. Si il choisit la méthode électronique, il reçoit une carte magnétique qu'il devra introduire dans la machine afin de pouvoir voter.
La votation se fait via un écran tactil qui permet aussi bien le vote blanc, que le vote standard.
La carte magnétique doit être rendue après avoir effectuée la votation, cette carte sert de moyen de contrôle pour le dépouillement de la votation, elle contient le choix du votant. Les machines électroniques doivent correspondre à des critères stricts de sécurités définis par l'état.

## Système Français plus sûr ?

De première abord on pourrait répondre à la question par l'affirmative, mais en regardant de façon plus approfondit on remarque qu'il contient tout de même un certain nombre de failles.
Le fait que le vote par correspondance soit interdit, qu'il faille présenter sa carte d'identité et signer un registre empêche déjà toutes les attaques possibles que nous avons cité pour le système bernois. En effet ces attaques étaient basées sur le système de vote par correspondance et sur la non tenu d'un registre des personnes ayant voté.
Mais la France autorise les votations par machine électronique, ce qui marque une évolution positive mais qui apporte son lot de problèmes. Exemple tiré d'un document réalisé par Chantal Enguehard, LINA, Université de Nantes :

1. États-Unis : en novembre 2003, dans le comté de Boone (Indiana), un ordinateur de vote a enregistré plus de 144 000 votes alors qu'il n'y avait que 19 000 électeurs [Simons 2004]
2. États-Unis : en mars 2002, dans la ville de Wellington, une élection visant à départager deux candidats se déroule sur des ordinateurs de vote. Les résultats sont de 1263 voix pour un candidat contre 1259 voix pour l'autre, mais 78 voix n'ont pas été enregistrées alors que les électeurs ont émargé. La directrice des élections a conclu que ces personnes n'ont simplement pas voté lorsqu'ils étaient en présence de l'ordinateur, ce qui n'est vraiment pas prouvé. Il est bien plus probable que ces votes n'ont pas été enregistrés par l'ordinateur. Des incidents similaires ont eu lieu à Palm Beach, et Miami [Mercuri 2002].
3. En Belgique chaque vote sur un ordinateur de vote est enregistré sur une carte magnétique anonyme. Lors du vote, il faut insérer la carte dans l'ordinateur puis procéder au choix. Celui-ci est mémorisé dans la carte que l'on dépose ensuite dans une urne électronique qui décomptera les voix. Ce système a le mérite de prendre en compte la nécessité absolue de mémoriser le vote sur un support qui permette un dépouillement indépendant de celui de l'ordinateur. Il présente cependant le défaut majeur d'utiliser un support qui n'est pas directement lisible par un humain (comme un bulletin imprimé avec le nom du candidat) : il n'est pas possible de voir ce qui est inscrit sur sa carte magnétique (est-ce bien mon vote ?). Lors des élections du 18 mai 2003, à Schaerbeek, un candidat d'une liste obtient plus de voix qu'il n'est possible d'en obtenir. Le recomptage manuel à partir des cartes magnétiques a montré une erreur de 4096 voix, erreur qu'il a été impossible d'expliquer ou de reproduire lors de nombreux tests menés sur le même ordinateur [Rapport Chambre et Sénat belge 2004].

Aucun système de vote n'est sûr à 100%, que se soit par papier ou de façon électronique, il existe toujours un moyen de tricher. Chaque pays adapte la sécurité en fonction de ces besoins et de ces coutumes. Comme nous l'avons écrit plus haut, le canton de Berne utilise un système papier laissant une place à la confiance comme aspect de sécurité.
La France quand a elle sécurise énormément son système de vote, forçant les gens à se déplacer en personne (sauf si procuration) et permet aussi la nouveauté avec les machines de votes électronique. La raison principale de la présence des machines de votes en France est de permettre le développement de ce secteur industriel dans le pays.

## E-voting Genève

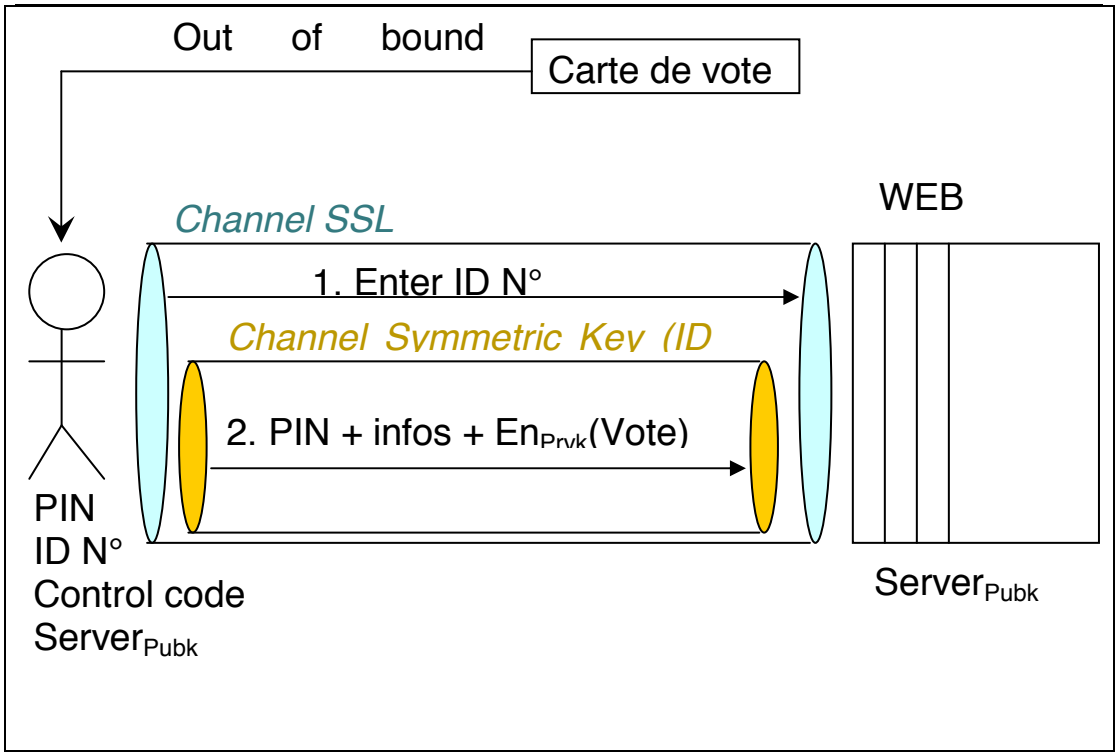
Lors de nos recherches pour trouver la meilleure solution à implémenter pour notre école. Nous avons étudié le système de Genève afin de savoir ce qui avait déjà été mis en place. Ce document décrit ce que nous avons trouvé ainsi que son fonctionnement, appuyé par un schéma.
Une partie reste encore floue. Le coté inter-serveur n'est pas documenté et même après avoir pris contact avec les responsables du projet, à ce jour nous attendons toujours des réponses.

### Fonctionnement globale

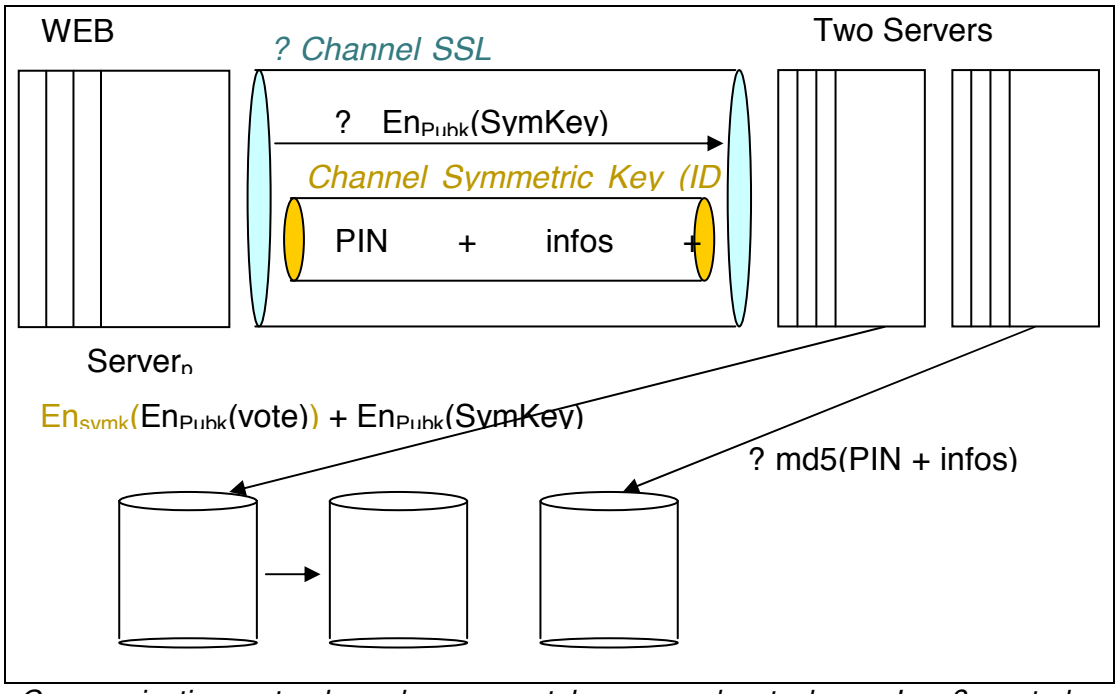
Le citoyen tape le site web de vote à la main afin d'éviter des attaques telles que le fishing. Il arrive sur la page d'accueil du site de vote, cette page est un *https* crypter grâce à *SSL / TLS* 128 bits. Il entre son numéro de carte de vote (nombre de seize chiffres écrient sur sa carte de vote), si le numéro existe, il arrive sur la page permettant de voter. Le citoyen vote puis doit remplir divers champs tels que :

- Le numéro code PIN (A gratter sur sa carte de vote).
- Sa date de naissance.
- Sa commune d'origine.
- Reproduire et contrôler le code alphanumérique se trouvant dans une image affichée et reproduite sur sa carte de vote. (en cas de problème appeler la ligne d'assistance téléphonique et mettre fin à la session).

Afin d'éviter les manipulations de scrutin par un cheval de Troie ou autre virus pouvant se trouver sur la machine du citoyen, les votes se font via un applet java se trouvant sur le serveur et étant exécuté par le serveur. L'intégrité du transfert entre le PC du citoyen et le serveur d'enregistrement est garantie par un tunnel *SSL* 128 bits, la confidentialité du scrutin est réalisée via un triple cryptage lors du transfert. Le vote est crypté via la clef publique du serveur de l'Etat, une clef symétrique générée à partir du numéro du bulletin de vote et une troisième fois par les tunnels SSL. Le contenu du vote ne peut être décrypté que grâce à la clef symétrique et la clef privée du serveur (cette dernière est obtenue que lors du dépouillement, elle-même est cryptée par des mots de passes).




*Communication entre le client et le web server*




*Communication entre le web server et le server de stockage. Les ? sont des informations que nous supposons mais nous n'avons eu aucune confirmation de la part de Genève.*

## Points de sécurité

La sécurité de l'E-voting genevois (http://www.geneve.ch/evoting/) est basée sur onze conditions qui sont :

1. Les suffrages exprimés électroniquement ne doivent pas pouvoir être interceptés, modifiés ou détournés.

2. Le contenu des suffrages exprimés électroniquement ne doit pas pouvoir être connu par des tiers avant le dépouillement.
3. Seules les personnes ayant le droit de vote doivent pouvoir prendre part au scrutin.
4. Chaque électeur ne dispose que d'une voix et ne peux voter qu'une seule fois.
5. En aucun cas, même pendant le dépouillement, il ne doit être possible de faire un lien entre un électeur et son suffrage.
6. Le site doit être en mesure de résister à une attaque en déni de service pouvant aboutir à la saturation du serveur.
7. L'électeur doit être protégé contre toute tentative de vol d'identité.
8. Le nombre de votes émis doit correspondre au nombre de votes reçus, toute différence doit pouvoir être expliquée et corrigée.
9. La preuve qu'un électeur a voté doit pouvoir être faite.
10. Le système n'accepte pas de vote électronique en dehors de la période d'ouverture du scrutin électronique.
11. Le bon fonctionnement du système doit pouvoir être vérifié par les autorités désignées à cet effet.

## L'architecture des serveurs est la suivante :

Plusieurs types de serveurs sont impliqués dans le vote par internet :

- le serveur d'accueil de l'Etat de Genève donne accès aux informations relatives aux scrutins mais ne contient aucun lien vers les autres serveurs, le votant doit taper l'adresse complète du site de vote afin d'éviter les détournements vers des sites pirates;
- le serveur de pages internet est protégé par le certificat SSL, maîtrisé par l'Etat, durcit et optimisé par configuration, qui n'est accessible que durant la période de vote, le reste du temps une page d'erreur est affichée;
- le serveur d'application, maîtrisé, durcit et optimisé par configuration, héberge la logique métier du vote ainsi qu'une partie de la logique de chiffrement utilisée par le canal sécurisé;
- enfin le serveur de base de données, lui aussi maîtrisé, durcit et optimisé par configuration, contient les tables du registre des électeurs et la table de l'urne électronique incluant plusieurs mécanismes de confidentialité et de contrôle d'intégrité.

## Les 11 conditions en détails

### *1. Les suffrages exprimés électroniquement ne doivent pas pouvoir être interceptés, modifiés ou détournés*

Pour sécuriser la communication entre le citoyen et le serveur une connexion utilisant le protocole SSL est utilisée.
Pour garantir à l'utilisateur qu'il communique bien avec le serveur une tierse partie, on associe un « certificat digital » au serveur. Ce certificat est renouvelé tous les trois mois au maximum (rythme des votations fédérales). Il permet à l'utilisateur de vérifier l'empreinte digitale du certificat en la comparant avec celle qui est imprimée sur la carte de vote.

A l'intérieur de la connexion SSL, est construit pour la session de vote, un *canal sécurisé spécifique* utilisant les dernières avancées en matière de cryptographie.
Afin de rester dans le périmètre contrôlé, un applet java est activé par le numéro de la carte de vote.

## Internet et la résolution des noms de domaine

La navigation sur Internet dépendant des serveurs de noms de domaine (« DNS »), la mise à jour de ces derniers est forcée à une fréquence de quelques minutes au lieu de quelques jours. Ainsi toute tentative de détournement se verra immédiatement détectée et contrée.
Le serveur est « automatiquement » identifié lors de la phase d'authentification de l'électeur. Lorsque l'électeur s'identifie sur le système, il affiche un code alphabétique unique, reproduit sur la carte de vote et lié au numéro de cette carte. Seul le serveur a les moyens d'associer le numéro de la carte de vote à son code correspondant. Il est impossible pour un espion qui surveillerait l'échange sur Internet de lire et de reproduire ce code car il est traité et crypté comme une image. Dans le cas où le code affiché ne correspondrait pas à celui de la carte l'électeur doit mettre fin à la session de vote et appeler l'assistance en ligne.

***2. Le contenu des suffrages exprimés électroniquement ne doit pas pouvoir être connu par des tiers avant le dépouillement***

Le serveur est placé dans l'hôtel de police sous un accès contrôlé et il faut impérativement être deux dans la salle pour y accéder.

Grâce à l'architecture à clé publique, chaque suffrage est crypté individuellement par une clé asymétrique détenue par les contrôleurs. Une clef publique qui se trouve dans l'application est utilisée de manière transparente afin de crypter chaque vote des citoyens. La clé privée pour déchiffrer, est protégée par deux mots de passe choisit par les contrôleurs et qu'eux seuls connaissent. Sans leur présence il est impossible de procéder au dépouillement de l'urne électronique.

Chaque vote en donc doublement crypté, premièrement par la clef symétrique (clef de session pour le canal sécurisé) et deuxièmement par le système de clef asymétrique. Ils seront déchiffrés au moment du dépouillement.

Les réponses possibles n'étant que « oui », « non » ou « blanc », il serait possible d'analyser le code pour en découvrir une réponse. C'est pour cela qu'un texte aléatoire y est ajouté avec le cryptage, afin de rendre cette analyse par comparaison impossible.

***3. Seules les personnes ayant le droit de vote doivent pouvoir prendre part au scrutin***

***4. Chaque électeur ne dispose que d'une voix et ne peux voter qu'une seule fois***

Le canton de Genève dispose d'un registre des habitants informatisés. Il permet de définir exactement la liste des citoyens ayant le droit de vote.

Chaque électeur reçoit à son domicile le matériel de vote contenant une carte de vote, un bulletin de vote et la documentation officielle. Cela trois semaines avant le début du scrutin. La carte de vote est personnelle et ne peut être utilisée qu'une seul fois.

Afin de rendre le vote aussi aisé qu'avec la méthode « normale », la carte de vote contient un code PIN à gratter. Ce code alphanumérique est différent pour chaque électeur et change à chaque scrutin. La carte comporte également un code à seize chiffres qui permet d'identifier le votant. Le code PIN est utilisé pour l'identification lors du login et le code à seize chiffres pour confirmer l'authentification après le vote.

***5. En aucun cas, même pendant le dépouillement, il ne doit être possible de faire un lien entre un électeur et son suffrage***

L'urne électronique où sont stockés les votes cryptés ne contient aucun lien avec le registre. Le registre est tenu afin d'éviter qu'une personne puisse voter deux fois.
Le fichier des votes est mélangé afin d'empêcher la reconstitution à l'aide des logs de la base de données des scrutins.
Aucune identité n'est enregistrée dans le registre, elle ne contient que des identifiants numérique. Un accès au registre ne permet pas de connaître les utilisateurs.

***6. Le site doit être en mesure de résister à une attaque en déni de service pouvant aboutir à la saturation du serveur***

Afin de protéger le système contre des attaques de masse ou une attaque deni de service (DoS), des sondes ont été installées. Elles réagissent sous certaines conditions :

- Si une page est trop souvent demandée par la même adresse IP.
- Si des tentatives d'identification trop rapide sont faîtes (séquence de numéro semblable ou inséré trop rapidement).
- Lorsque certaines pages subissent des requêtes anormales.
- Si un équipement tombe en panne (serveurs, disque dur, pare-feu, problème de la base de données, arrêt du logiciel).
- Si le système de fichier est modifié anormalement.

Si une sonde déclenche l'alarme, un opérateur est immédiatement appelé et une procédure d'urgence est lancée. Selon la gravité de l'alarme plusieurs méthodes existent :

- Le problème est bénin, les autorités en sont informés, voir les citoyens.
- Si le vote par Internet est mis en danger, dans ce cas le vote par Internet est stoppé et il se poursuit de façon standard (façon papier et postale).
- Dans le cas de votes erronés enregistrés et sans possibilité de suppression, tout est annulé et on redistribue des cartes de vote aux électeurs concernés.

***7. L'électeur doit être protégé contre toute tentative de vol d'identité***

Des moyens fort ont été mis en place pour protéger au possible le vol d'identité :

- Le numéro d'identification est composé de seize chiffres ce qui rend quasiment improbable la possibilité de découvrir un tel nombre.
- Le fichier qui contient toutes les informations des votants avec les codes d'accès ne quitte jamais les bâtiments de l'administration. Il est crypté et il faut une dizaine de personne pour se l'approprier.
- L'électeur pour s'identifier doit utiliser une information qu'il possède (son PIN), deux informations qu'il connaît (date de naissance et commune d'origine) et confirmer avec un code à seize chiffre (qu'il possède sur la carte). Posséder uniquement la carte ne suffit pas pour voter pour un tiers.
- Un appel de 4000 à 8000 citoyens est effectué lors d'élection pour vérifier que ce sont bien eux qui ont votés

***8. Le nombre de votes émis doit correspondre au nombre de votes reçus, toute différence doit pouvoir être expliquée et corrigée.***

Pour garantir une grande sécurité de l'information tous les données sont doublées et stockées sur des disques différents.

***9. La preuve qu'un électeur a voté doit pouvoir être faite***

Pour garantir l'anonymat et la liberté de vote, aucune information concernant les votes ne peut être divulguée. Cependant pendant les 3 semaines de scrutin un citoyen peut vérifier que son vote a bien été pris en compte en insérant son numéro d'identification (qui figure sur la carte de vote). Le système redonnera alors la date et l'heure de son vote ainsi que la méthode utilisée.

***10. Le système n'accepte pas de vote électronique en dehors de la période d'ouverture du scrutin électronique***

Pour éviter ce problème le serveur web est accessible uniquement pendant une période déterminée qui commence et fini selon la loi. Après cette période le serveur bloque lui-même la communication ainsi que les *firewall.*

***11. Le bon fonctionnement du système doit pouvoir être vérifié par les autorités désignées à cet effet***

Des urnes de contrôles identiques aux 68 urnes officielles du canton de Genève sont utilisés par des citoyens désignés par les partis politiques et nommés par le Conseil d'Etat pour voter afin de tester le système. De cette manière une vérification simple et fiable peut être effectuée.

## Faq sur le site Genevois :

**Un cryptage et une signature électronique basés sur un système PKI ne sont-ils pas une solution plus sûre que la solution par code pin, mise en œuvre ?**

La solution actuelle implique déjà un cryptage des votes de bout en bout, jusqu'au moment du dépouillement. Elle offre en outre l'avantage d'ouvrir le vote par Internet à tous.
Il est prévu d'offrir la possibilité d'une authentification par un système PKI dès que le principe en aura été accepté dans la législation fédérale et qu'une telle infrastructure aura été implantée dans le cadre des projets fédéraux de carte d'identité électronique.

## Analyse système physique

Lorsque la méthode à utiliser a été décidée, nous avons fait deux propositions. Une au format électronique et une au format physique. Ce document décrit le fonctionnement de la méthode physique ainsi que ce qui est nécessaire pour son implémentation.
Cette méthode n'a pas été appliquée au projet car l'école ne peut pas mettre une personne à disposition pour s'occuper de générer et de diffuser les mots de passe. Cela demande aussi beaucoup de travail au niveau du secrétariat et il faut mettre un ordinateur à disposition uniquement pour s'occuper de la génération des mots de passe. Cet ordinateur doit être dans un endroit sur (coffre-fort) lorsqu'il n'est pas utilisé. De plus, un système pour une école doit être souple dû au-va-et-vient des élèves et ce système n'y est pas adapté du fait que pour chaque nouvel élève, il aurait fallu tout régénérer.

### Vision

Nous allons créer pour l'école d'ingénieur de Bienne un système de votation en ligne (*E-voting*) présentant une sécurité optimale afin de pouvoir voter en toute sécurité.
Ce projet est réalisé dans le cadre de notre thèse de bachelor. Le but est que ce système soit mis en pratique dans la HTI qui est une école qui contient plusieurs corps de métier dont une section informatique relativement importante.
Notre projet sera *Open Source* afin que la sécurité puisse être correctement testée et évaluée. Le travail a été donné à deux groupes distincts qui produisent deux implémentations différentes du système. La meilleure solution des deux sera retenue.

Notre travail consiste à réaliser un système de vote sécurisé *end-to-end* ne permettant aucune manipulation avant, pendant et après la période de votation.

### Autre nécessité et contraintes

Le système doit prendre en compte les coups en temps et en matériel d'une école, ce qui limite les choix des sécurités à employer. Il doit le plus *user friendly* possible et être portable sous toutes les plates formes.
Les onze *statements* à respectés

1. Les *suffrages* exprimés électroniquement ne doivent pas pouvoir être interceptés, modifiés ou détournés.
2. Le contenu des *suffrages* exprimé électroniquement ne doit pas pouvoir être connu par des tiers avant le *dépouillement*.
3. Seules les personnes ayant le droit de vote doivent pouvoir prendre part au *scrutin*.
4. Chaque électeur ne dispose que d'une voix et ne peux voter qu'une seule fois.
5. En aucun cas, même pendant le *dépouillement*, il ne doit être possible de faire un lien entre un électeur et son *suffrage*.
6. Le site doit être en mesure de résister à une *attaque en déni de service* pouvant aboutir à la saturation du serveur.
7. L'électeur doit être protégé contre toute tentative de vol d'identité.
8. Le nombre de votes émis doit correspondre au nombre de votes reçus, toute différence doit pouvoir être expliquée et corrigée.
9. La preuve qu'un électeur à voté doit pouvoir être faite.
10. Le système n'accepte pas de vote électronique en dehors de la période d'ouverture du *scrutin* électronique.

11. Le bon fonctionnement du système doit pouvoir être vérifié par les autorités désignées à cet effet.

## Acteurs

| | |
|---|---|
| HTI : | Elle génère les codes d'accès et les distribuent. |
| Serveur WEB : | Interface qui lie l'utilisateur à notre système. |
| Urne électronique : | Base de données ou sont stockées tous les bulletins de vote cryptés. |
| Registre des électeurs : | Base de données contenant toutes les données personnelles des votants. |
| Registre de contrôle : | Base de données contenant les *ID Unique* des personnes ayants droit de vote. |
| Votant : | Personne ayant le droit de vote |

## Use Cases

| | |
|---|---|
| Génération des codes d'accès : | Permet à l'école de créer à l'aide du *serveur LDAP* les codes qui seront distribués aux votants potentiels |
| Distribution des codes d'accès : | L'école distribue les *codes d'accès* aux votants |
| Obtention des codes d'accès : | Permet aux votants d'obtenir les *codes d'accès* nécessaire à s'identifier sur notre système. |
| Identification : s'identifie sur | Le votant, à l'aide des *codes d'accès* notre système. |
| Permission : savoir | Fonctionnement lors de l'identification afin de si la personne a le droit de vote |
| Vote : | Phase de vote point de vue client et serveur |
| Confirmation : | Confirmation que la personne à voter. |

## Use Case UC1 : Génération des codes d'accès

Primary Actor : HTI
Precondition : Début du semestre, fait depuis un ordinateur *Out-of-band*
Postcondition : Les *codes d'accès* pour les votants sont générés correctement

### *Main Success Scenario*

1. Les responsables des votes de l'école lancent le programme de génération.
2. Les responsables rentrent les mots de passes qui serviront à encrypter le registre des électeurs.
3. Les informations sur les votants sont récupérées *(LDAP* en local).
4. Les nombres aléatoires servant de *PIN*, *ID Unique*… (Plusieurs par votant) sont générés et stockés dans la base des électeurs.
5. La base de données de contrôle est créée sur la base de ces nombres.
6. Les nombres aléatoires ainsi que le nom du votant sont imprimés sur papier et mis dans une enveloppe.
7. La base de donnée est cryptée et transférée sur un serveur réseau.

### *Question*

De quelles manières la base de contrôle est transférée sur le serveur réseau ?
Elle est cryptée et mise sur un support physique (Clef USB, …) afin d'être décryptée et copiée sur le serveur réseau.

## Use Case UC2 : Distribution des codes d'accès

Primary Actor : HTI
Precondition : Les codes d'accès ont été générés, imprimés et mis sous enveloppe
Postcondition : Les votants possèdent leurs codes d'accès

### *Main Success Scenario*

1. Les enveloppes sont remises à chaque secrétariat de section.
2. Un email est envoyé à tous les étudiants.
3. Les secrétaires vérifient l'identité des votants et leur donnent l'enveloppe.

### *Extension*

4. L'identité ne peut pas être vérifiée
    a. La secrétaire demande au votant de repasser avec une carte de légitimation.
5. Toutes les enveloppes n'ont pas été distribuées avant la première votation
    a. Les enveloppes sont détruites

### *Question*

Combien de temps doivent être conservées les enveloppes ?

Le temps aux étudiants de venir les chercher et au plus tard jusqu'à la premières période de vote.

## Use Case UC3 : Obtention des codes d'accès

Primary Actor : Votant
Precondition : Début du semestre, feuilles d'accès imprimées, Email du secrétariat reçu.
Postcondition : Possession des codes d'accès

### *Main Success Scenario*

1. Le votant se présente au secrétariat avec sa carte d'étudiant
2. La secrétaire contrôle l'identité
3. Le votant reçoit l'enveloppe de vote

### *Extension*

3. Le votant ne présente pas une carte d'étudiant valide
    a. La secrétaire lui demande de repasser plus tard avec une carte valide.

## Use Case UC4 : Identification

Primary Actor : Votant
Precondition : Il a reçu les codes d'accès, il est dans la période de vote, il a reçu le mail comportant les informations de vote reçu.
Postcondition : Le votant est identifié dans le système

### *Main Success Scenario*

1. Le votant se connecte via un web browser sur l'adresse du site qui se trouve sur sa feuille comportant les codes d'accès.
2. Il entre son numéro *ID Unique* présent sur les *codes d'accès* (ligne x)
3. Le système confirme son identification
4. Le système montre une image avec un *code de contrôle*. L'utilisateur vérifie l'exactitude du code avec celui qu'il possède dans les *codes d'accès*.

### *Extension*

II. A tout moment le browser peut être fermé et le processus d'identification est annulé.
1. L'adresse Url est correcte mais le laps de temps disponible pour voter est dépassé.
   a. Une page d'erreur est affichée.
4. Le votant inscrit mal son numéro *ID Unique*. Le système lui propose 2 fois d'insérer son numéro *ID Unique* (3 tentatives). Après la 3ème erreur
   a. Le système se bloque et il est impossible de voter depuis cette machine pendant 30minutes.
5. L'image ne correspond pas.
   a. L'utilisateur stoppe son vote et appelle le *numéro de contact* qu'il retrouve dans les listes contenant les codes d'accès.

### *Question*

Combien de temps faut-il bloquer la machine ?

La machine est bloquée 30 minutes.

## Use Case UC5 : Permission

Primary Actor : Le votant
Precondition : L'utilisateur est identifié dans le système
Postcondition : L'utilisateur sait se qu'il peut faire sur le site de vote.

### *Main Success Scenario*

1. Le système contrôle si l'*ID Unique* a déjà été utilisée.
2. L'utilisateur voit le bulletin de vote et a la possibilité de voter.

### *Extension*

1. L'utilisateur a déjà voté.
    a. Voir Use case *Confirmation* (page 9).

## Use Case UC6 : Vote

Primary Actor : Le votant
Precondition : Le votant est identifié et a la droit de vote.
Postcondition : Le votant a voté

### *Main Success Scenario*

1. L'utilisateur coche ou ne coche pas les votes (oui, non). Il valide.
2. Le système récapitule ses votes.
3. L'utilisateur insère son *PIN* plus une *donnée personnel*. Il valide.
4. Le système vérifie la validité des informations et enregistre le vote.
5. Le système confirme son vote.

### *Extension*

I A tout moment le système peut être arrêté. Aucune information n'est sauvegardée.

1. Le votant voit une erreur dans son choix.
    a. Il revient en arrière et modifie son vote.
2. Le *PIN* ou la *donné personnel* est fausse.
    a. Le système affiche un message, puis après 3 essais le votant est *délogger*.

## Use Case UC7 : Confirmation

Primary Actor : Votant, HTI
Precondition : Le votant a voté
Postcondition : Une confirmation est affichée

### *Main Success Scenario*

6. L'heure, la date et le moyen de vote sont affichés

## Spécification supplémentaires

Spécification de Java 5 ou 6 sont à respecter afin de faire tourner un applet.
Il faut un serveur de base de données, web. Il faut un système de certificat (https, encryptions à clef publique).
Un ordinateur *out-of-band* dédié à la génération et à l'impression des codes d'accès.

## Evénements système

### *Génération des codes*

getElecteurs()
encrypte(String mdp[])

### *Envoi*

sendMail(String contenu)

### *Identification*

log(String Pin)
controleImage(String code)

### *Permission*

IsIdcorrect(String Id)
AsIdVoted(String Id)
IsPinCorrect(String Pin, String InfoPerso)

### *Votation*

Vote(String Pin, String Bulletin, String InfoPerso)
getConfirmation()

## Cas générique

### *Fonctionement général (étape par étape)*

Le système génère les codes :

1. L'école génère les *codes d'accès*.
2. L'école imprime les *codes d'accès* et les mets sous envelopes.
3. L'école confie les enveloppes aux différents secrétariats en fonction des départements.

Distribution des feuilles de sécurités :

1. Le votant se présente au secrétariat en prouvant son identité.
2. L'identité est contrôlée.
3. Le votant reçoit la feuille de sécurité.

Phase de vote :

1. Le votant écrit à la main l'adresse de la page de vote qui se trouve sur la feuille de sécurité.
2. Il vérifie le certificat à l'aide du *fingerprint* de la feuille de sécurité.
3. Il insère son *ID Unique* qui se trouve à la ligne x.
4. Le système lui demande de vérifier que l'image à la ligne x de sa feuille de sécurité correspond bien à celle à l'écran. (Identification du serveur au client)
5. Il choisit son vote.
6. Il valide son vote en insérant les informations personnelles demandées et son *PIN* se trouvant à la ligne x sur sa feuille de sécurité.
7. Le votant voit le message : « Votre vote a bien été pris en compte ».

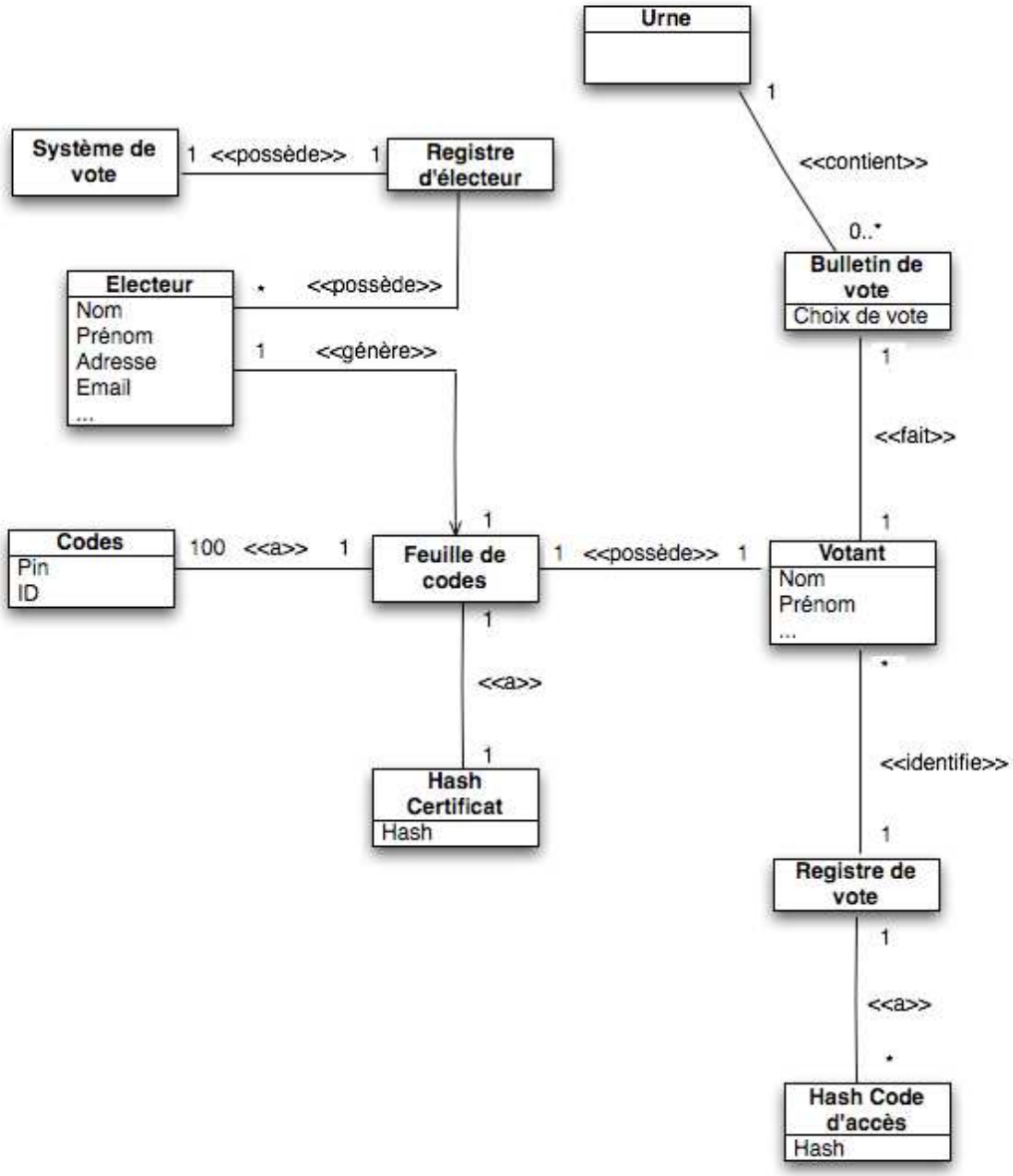
### *Erreur d'envoi de l'email*

L'email envoyé sert à informer les personnes ayant le droit de vote qu'une phase d'élection arrive. Si une erreur survient lors de l'envoi, l'email est renvoyé et ceci trois fois au maxium. Si malgré ces trois renvois une erreur survient toujours, le système passe au votant suivant.

### *Feuille de sécurité restante au secrétariat à l'ouverture des votes*

Les feuilles de sécurité restantes au secrétariat lors de la première ouverture d'une phase de vote sont détruites. Ceci afin d'éviter qu'une personne volant ces feuilles puissent voter plusieurs fois. Cette façon de faire implique que si un votant ne récupère pas dans les temps sa feuille de sécurtié, il ne pourra pas voter durant le semestre.

## Domain model



## Concepts

**Système de vote**
Application servant de générateur et d'interface de communication entre le système de vote et les utilisateurs.

**Registre d'électeur**
Registre contenant les informations personnelles sur les votant. Ce registre contient aussi tous les codes d'accès, il est donc crypté et gardé de la façon la plus sûre possible.

**Registre de vote**
Registre contenant les informations servant à l'identification des votants. Aucune donnée personnelle n'y figure, uniquement des hashs.

**Feuille de code**
Ensemble des *codes d'accès* livrés à un votant.

**Codes**
*Codes d'accès* du votant. *Id Unique, PIN*, …

**Hash Certificat**
Contient l'hash value du certificat du serveur de vote afin de permettre au votant de vérifier l'identité du serveur.

**Votant**
Personne ayant le droit de vote.

**Bulletin de vote**
Bulletin crypté contenant les choix de vote du votant.

**Urne**
Base de données servant à stocker les bulletins de vote.

## Associations

**Système de vote – Registre d'électeur**
Le système de vote possède un registre d'électeur.

**Registre d'électeur – Feuille de codes**
Le registre d'électeur sert à générer toutes les feuilles de codes et à les stocker.

**Feuille de codes – Codes**
Chaque feuille de codes a cent codes d'accès (*Pin + ID Unique*, …)

**Feuille de codes – Hash Certificat**
Chaque feuille de codes a l'hash value du certificat du serveur de vote.

**Votant – Feuille de codes**
Chaque votant possède une feuille de codes.

**Votant – Registre de vote**
Chaque votant s'identifie vers le registre de vote.

**Votant – Bulletin de vote**
Chaque Votant fait un bulletin de vote.

**Urne – Bulletin de vote**
L'urne contient tous les bulletins de votes réalisés (1…*)

**Registre de vote – Feuille de codes**
Le registre de vote contient les feuilles de codes sous forme de hash value.

## Informations supplémentaires

### *Point de vu du système*

Le système est composé en deux parties. La première contenant les bases de données pour générer la feuille de sécurité et la deuxième contient toute les informations permettant d'enregistrer et de contrôler les votants. A noter que la deuxième base de données ne contient aucun nom mais des identifiants numériques.

### *Première base de données*

Elle contient les noms, prénoms et informations personnelles de chaque votant potentielle ainsi que la liaison entre le votant et la liste de ces mot de passe, l'identifiant unique et l'image unique. La base de données est extrêmement sensible est doit être encrypter de la façon la plus rigoureuse possible et enregistré sur un ordinateur *out-of-bound*.

### *Deuxième base de données (registre de vote)*

Elle est générée à partir de la première base de données et avant chaque votation. Elle contient le md5 de la date de naissance, la commune du domicile, le numéro d'identification, le mot de passe unique et l'image unique.
Le transfert de la première à la deuxième base de données se fait via un support physique sur lequel la deuxième base de données (générée sur la machine qui contient la première) est cryptée jusqu'au transfert sur le serveur de base de données.

### *Urne*

L'urne est vide avant chaque élection. Les informations contenus dans l'urne électronique sont encrypté doublement. Premièrement par la clé de session utilisée lors de la communication votant-système (la base de données doit garder cette clef pour chaque vote) et deuxièmement par la clé publique du serveur. Pour garantir une sécurité maximal la clé privée du serveur et crée à partir de 3 mot de passes de 3 personnes différentes. Le couple de clef privée/publique change à chaque votation. A noter que les deux bases de données (urnes et registre de vote) sont brassés régulièrement.

### *Création et transfert de la feuille de sécurité*

La feuille de sécurité est générée depuis l'ordinateur out-of-band qui contient la première base de données. Elles sont imprimées et insérées dans une enveloppe opaque sur laquelle est écrit le nom, prénom et le login de l'école de la personne. Cette enveloppe est distribuée par le corps enseignant au début de chaque semestre en spécifiant que se sont des données confidentielles.