Haute école spécialisée bernoise
Technique et informatique

Thèse de Bachelor 2012

Division Informatique

# SwissiVi

## Proof-of-Concept for a Novel E-Voting Platform

Etudiants : Andrea Pellegrini
Philémon von Bergen

Professeurs : Prof. Dr. Eric Dubuis
Prof. Dr. Rolf Haenni
Prof. Reto E. Koenig

Expert : Dr. Andreas Spichiger

Date : 14 juin 2012

Bien qu'il existe aujourd'hui plusieurs protocoles cryptographiques qui satisfont de nombreuses exigences suggérées pour des systèmes de vote électronique, une interface sûre et facile à utiliser entre le votant et la cryptographie sur un PC ou smartphone potentiellement vulnérable manque toujours à l'appel. Le *Research Institute for Security in the Information Society* de la HESB a mis en place un nouveau concept de vote électronique résolvant ce problème. Le but de ce travail de bachelor est de réaliser une démonstration de faisabilité (*Proof of Concept*, POC) de ce concept.

Par souci de lisibilité, seule la forme masculine est utilisée dans ce document. La forme féminine est bien sûr toujours sous-entendue.

**Berner Fachhochschule**
**Haute école spécialisée bernoise**
Technik und Informatik
Technique et informatique

Abteilung Informatik
Division informatique

# Erklärung der Diplomandinnen und Diplomanden
# Déclaration des diplômé-e-s

## Selbständige Arbeit / Travail autonome

Ich bestätige mit meiner Unterschrift, dass ich meine Bachelor Thesis selbständig durchgeführt habe. Alle Informationsquellen (Fachliteratur, Besprechungen mit Fachleuten, usw.), die wesentlich zu meiner Arbeit beigetragen haben, sind in meinem Arbeitsbericht im Anhang vollständig aufgeführt.

Par ma signature, je confirme avoir effectué mon mémoire de Bachelor de manière autonome. Toutes les sources d'information (littérature spécialisée, discussions avec spécialistes etc.), qui m'ont fortement aidées dans mon travail, sont intégralement mentionnées dans l'annexe de mon mémoire.

Name/Nom, Vorname/Prénom ..... Pellegrini Andrea .....

Datum/Date ..... 14.06.2012 .....

Unterschrift/Signature ..... APelli .....

Dieses Formular ist dem Bachelor Thesis Bericht beizulegen.

Ce formulaire doit être joint au rapport du mémoire de Bachelor.

**Berner Fachhochschule**
**Haute école spécialisée bernoise**
Technik und Informatik
Technique et informatique

Abteilung Informatik
Division informatique

# Erklärung der Diplomandinnen und Diplomanden
# Déclaration des diplômé-e-s

## Selbständige Arbeit / Travail autonome

Ich bestätige mit meiner Unterschrift, dass ich meine Bachelor Thesis selbständig durchgeführt habe. Alle Informationsquellen (Fachliteratur, Besprechungen mit Fachleuten, usw.), die wesentlich zu meiner Arbeit beigetragen haben, sind in meinem Arbeitsbericht im Anhang vollständig aufgeführt.

Par ma signature, je confirme avoir effectué mon mémoire de Bachelor de manière autonome. Toutes les sources d'information (littérature spécialisée, discussions avec spécialistes etc.), qui m'ont fortement aidées dans mon travail, sont intégralement mentionnées dans l'annexe de mon mémoire.

Name/Nom, Vorname/Prénom ..... von Bergen Philémon .....

Datum/Date ..... 14 juin 2012 .....

Unterschrift/Signature ..... P. von Bergen .....

Dieses Formular ist dem Bachelor Thesis Bericht beizulegen.

Ce formulaire doit être joint au rapport du mémoire de Bachelor.

# Table des matières

# Table des figures

# 1. Introduction

L'internet se répand de plus en plus dans notre société. Les gouvernements essaient aussi de mettre à profit les avantages offerts par ce moyen de communication.
En Suisse, des études sont menées pour développer l'eGovernment. La possibilité de voter par le moyen d'internet est, entre autres, un dossier qui est approfondi en ce moment. Plusieurs cantons ont déjà mis en place des projets pilotes et ont effectué différents essais. La Chancellerie Fédérale désire cependant mettre en place un système uniforme qui pourrait être utilisé dans toute la Suisse. La Haute Ecole Spécialisée Bernoise a été mandatée pour développer un concept qui pourrait satisfaire aux exigences du système suisse tout en respectant les critères de sécurité requis pour un tel système. C'est dans ce contexte que se place le présent travail.

Bien qu'il existe aujourd'hui plusieurs protocoles cryptographiques qui satisfont de nombreuses exigences suggérées pour des systèmes de vote électronique, une interface sûre et facile à utiliser entre le votant et la cryptographie sur un PC ou smartphone potentiellement vulnérable manque toujours à l'appel. Le groupe d'E-Voting de RISIS [1] a mis en place un nouveau concept de vote électronique résolvant ce problème. Les composants essentiels de ce concept sont la plateforme de vote affichant les choix du votant, un appareil de vote sûr, et une carte de vote personnalisée.

Le but de cette thèse de bachelor est la réalisation de ce concept ciblant une solution pour le vote électronique en Suisse. Plus spécifiquement, la réalisation comprend l'interface utilisateur complète pour le votant pour un système de vote électronique lors de votations et élections suisses au niveau fédéral, cantonal et communal. Les composants hardware devront être simulés sur des smartphones. Le centre d'attention principal est placé sur les aspects à première vue contradictoires que sont la convivialité et la sécurité au niveau application. Le but de cette thèse est atteint si le résultat obtenu rencontre un accueil favorable chez tous les intéressés impliqués dans l'e-voting.

Ce projet porte le nom Swissivi pour les raisons suivantes : « Swiss » parce que le projet est développé en Suisse et pour la Suisse, et « iVi » en référence aux initiales e (se prononçant « i » en anglais) et V (prononcé « Vi ») de « e-Voting ».

## 1.1. Le concept en bref

Le groupe de l'institut RISIS a écrit une documentation décrivant en détail le concept qu'il a mis en place [7].

La plateforme sécurisée est un problème récurrent dans le domaine de l'e-voting. Elle doit permettre au votant de réaliser son vote de manière confidentielle, même si le support utilisé pour afficher la plateforme est infecté par un malware. Par définition, un malware possède un contrôle complet sur le support infecté, par conséquent, il peut s'ingérer dans toutes les actions exécutées par ledit support. La plateforme peut alors être aussi sécurisée que possible, la confidentialité ainsi que l'intégrité du vote ne peuvent pas être garanties. Pour éviter ce problème, la HESB a développé un concept mettant en jeu un petit appareil supplémentaire permettant de confirmer son vote, comme cela est déjà utilisé dans certains systèmes d'e-banking pour confirmer les transactions. Cet appareil n'a pas de moyen de communication direct avec l'ordinateur ou internet, et ne peut donc pas être infecté par un malware, il est donc totalement sûr. Malgré cet appareil, un malware installé sur le support affichant la plateforme peut toujours modifier le vote, mais cela sera détecté par l'utilisateur lorsqu'il devra confirmer son choix sur l'appareil de vote.

Trois acteurs principaux interviennent dans ce concept. Premièrement, la plateforme de vote permet à l'utilisateur de consulter les objets de vote. Les résultats pouvant être choisis par le votant (oui ou non pour une votation, une liste de candidats pour des élections) sont représentés sous forme de codes-barres bidimensionnels. A l'aide du second acteur, l'appareil de vote muni d'une caméra, l'utilisateur pourra lire ces codes-barres. Sur l'écran de l'appareil apparaîtra le résultat choisi. L'utilisateur devra alors confirmer son choix. L'appareil de vote peut être

1. Research Institute for Security in the Information Society, institut de recherche de la HESB, Haute Ecole Spécialisée Bernoise

partagé entre plusieurs votants. Le troisième acteur permet de vérifier la légitimation de vote, ainsi que de récupérer le vote créé par l'appareil de vote afin de l'envoyer à l'urne électronique. Il s'agit d'une carte de vote qui doit être insérée dans l'appareil de vote. Cette carte est protégée contre une utilisation non autorisée (par un code PIN par exemple). Une fois le vote terminé, la carte de vote peut être connectée à l'ordinateur pour y copier le fichier de vote généré par l'appareil de vote. Ce fichier peut ensuite être chargé dans l'urne électronique.

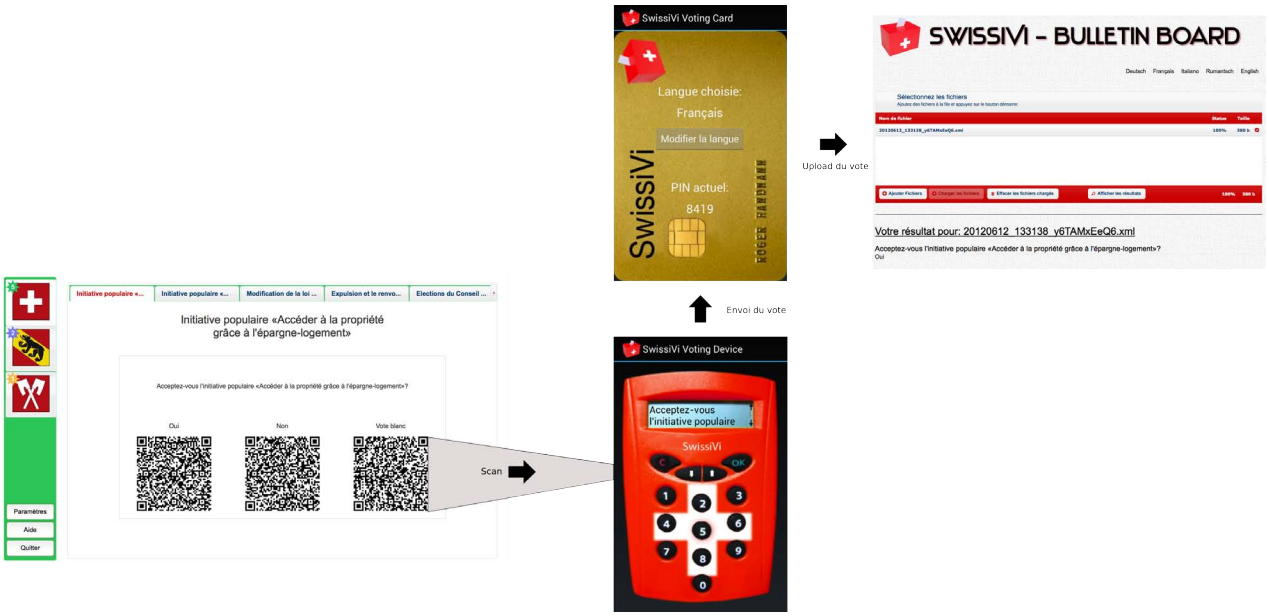



Figure 1.1.: Le concept schématisé

L'utilisation de l'appareil de vote et de la carte de vote garantit la confidentialité du vote. En effet, le support utilisé pour afficher la plateforme de vote ne peut pas savoir quel code-barres a été scanné. Une fois le vote confirmé sur l'appareil, celui-ci est chiffré de façon à ce qu'il ne soit lisible que par l'urne électronique. Ces deux aspects offrent donc la confidentialité. La demande de confirmation affichée sur l'appareil de vote permet de garantir l'intégrité du vote. Si un malware installé sur le support utilisé pour afficher la plateforme modifie le contenu des codes-barres, le votant s'en rendra compte lorsqu'il devra vérifier son choix sur l'appareil de vote. S'il confirme, le vote sera alors signé numériquement par la carte de vote de façon à ce qu'il ne puisse plus être modifié ultérieurement. Chacune de ces exigences est réalisée grâce à la cryptographie, garantissant ainsi la sécurité du système. Le document cité en introduction en décrit le fonctionnement dans les détails [7].

## 1.2. Portée du projet

Dans ce projet, il s'agit de développer un système de simulation de ce concept à des fins de présentations. Ce travail s'articule autour de deux axes principaux :
– la simulation de l'appareil et de la carte de vote
– la plateforme de vote
Comme l'appareil et la carte de vote n'existent pas encore, il faut les simuler. Cela se fait à l'aide de deux smartphones. L'insertion de la carte dans l'appareil est simulée par une communication entre les deux smartphones.

La deuxième partie comprend la réalisation d'une plateforme de vote qui soit facilement utilisable pour le votant et qui supporte les types de votes de la Suisse, notamment les votations, les initiatives et les élections, ainsi que la séparation dans les trois niveaux : fédéral, cantonal, communal.

D'autres tâches moins prioritaires sont également à réaliser en fonction du temps disponible. L'appendice B contient un cahier des charges et l'annexe C une liste de use-cases détaillant plus précisément les travaux à réaliser.

Une étude de ce projet et la réalisation des spécifications des différents acteurs ont déjà été faites dans le cadre d'un projet pour le module de projet 2. Certaines particularités, lesquelles ne seront pas réexpliquées dans ce document, y sont décrites [16].

### 1.2.1. Simulation de l'appareil et de la carte de vote

**L'appareil de vote** Il doit permettre au votant de scanner les codes-barres présents sur la plateforme de vote. Il s'agit donc de décoder le contenu des codes-barres. Ensuite, l'appareil doit permettre au votant de confirmer le choix qu'il a scanné.
Parallèlement, l'utilisateur doit pouvoir changer le code PIN de la carte.

**La carte de vote** Elle contient un certain nombre de données propres à son propriétaire, comme par exemple, la langue dans laquelle seront affichés les messages sur l'appareil de vote, ou le code PIN verrouillant la carte.

**La communication** L'échange de messages entre la carte de vote et l'appareil doit permettre de simuler l'introduction de la carte dans l'appareil. Les informations transmises dans cette communication sont, par exemple, le code PIN, la langue ou le résultat du vote.

### 1.2.2. La plateforme de vote

Le rôle de la plateforme est d'afficher les objets pour lesquels l'utilisateur doit voter. Ces objets diffèrent en fonction du canton et de la commune de résidence du votant.
Elle doit supporter les votations normales, les initiatives où trois questions sont posées, ainsi que des élections où le votant doit pouvoir choisir les candidats qu'il désire élire. Ces différents types de votations doivent être supportés au niveau fédéral tout comme aux niveaux cantonal et communal. Tous les résultats qui peuvent être choisis doivent être représentés sous forme de codes-barres bidimensionnels afin qu'ils puissent être lus par l'appareil de vote.

**La sélection de la commune** La sélection de la commune est une fonctionnalité essentielle pour cette plateforme. Il est indispensable de connaître la provenance du votant afin de pouvoir lui afficher les objets communaux auxquels il doit répondre. Le canton de résidence est également nécessaire pour pouvoir afficher les objets cantonaux ou pour les élections fédérales qui fonctionnent sur le principe des circonscriptions, chaque canton étant une circonscription. Il s'agira donc de trouver un moyen facile pour sélectionner le canton et la commune d'origine.

**La création des codes-barres** Un autre défi pour la plateforme de vote est la génération des codes-barres. Pour des votations standard, les codes peuvent être créés au chargement de la page, puisque leur contenu est statique. Cependant, pour une initiative ou une élection, il est impossible de générer à l'avance les codes-barres contenant tous les résultats imaginables. Les codes doivent donc être générés en fonction du choix du votant.

**Une interface simple d'utilisation** L'interface pour les élections doit être conçue pour permettre à l'utilisateur de choisir une liste prédéfinie ou de composer sa propre liste, au besoin de panacher ou cumuler les candidats. Il s'agit donc de réaliser un outil qui permette à l'utilisateur de faire cela facilement. D'autre part, l'interface générale doit être optimisée pour afficher tous les objets de vote de façon pratique et facile à utiliser.

# 2. Planification

## 2.1. Planification prévisionnelle

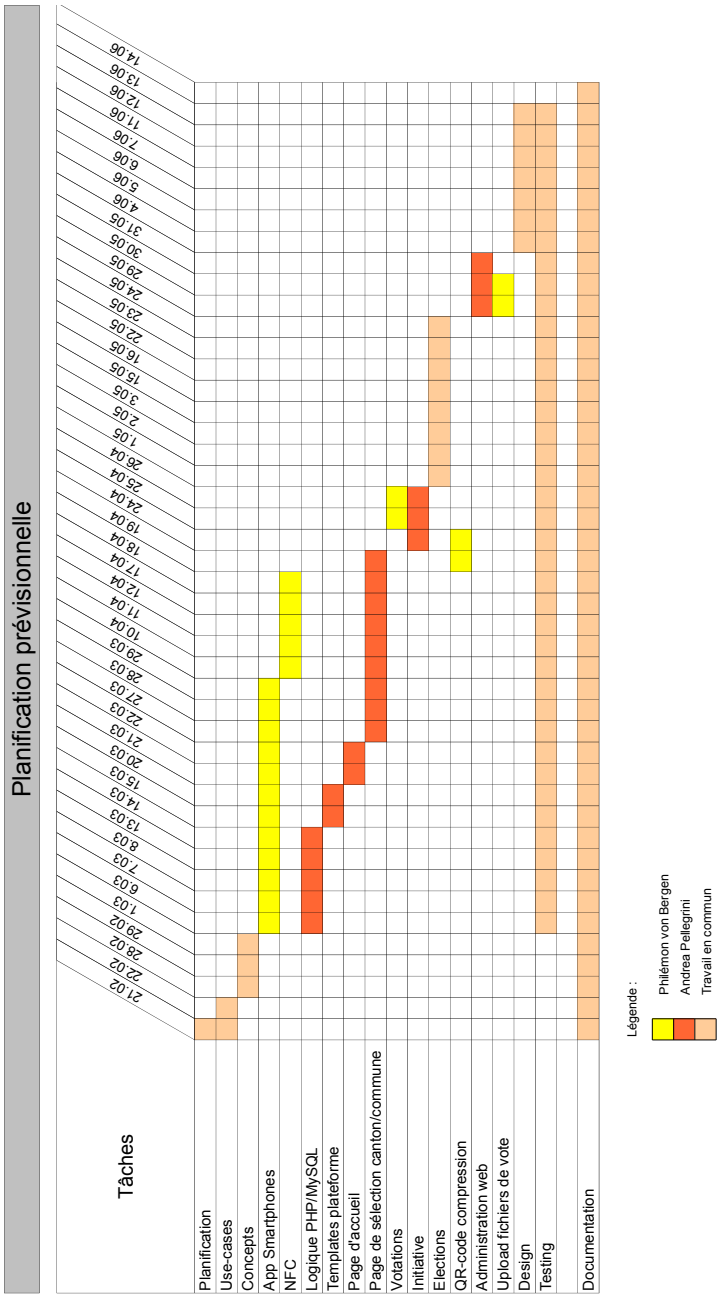



Figure 2.1.: Planning prévisionnel

## 2.2. Travail effectif

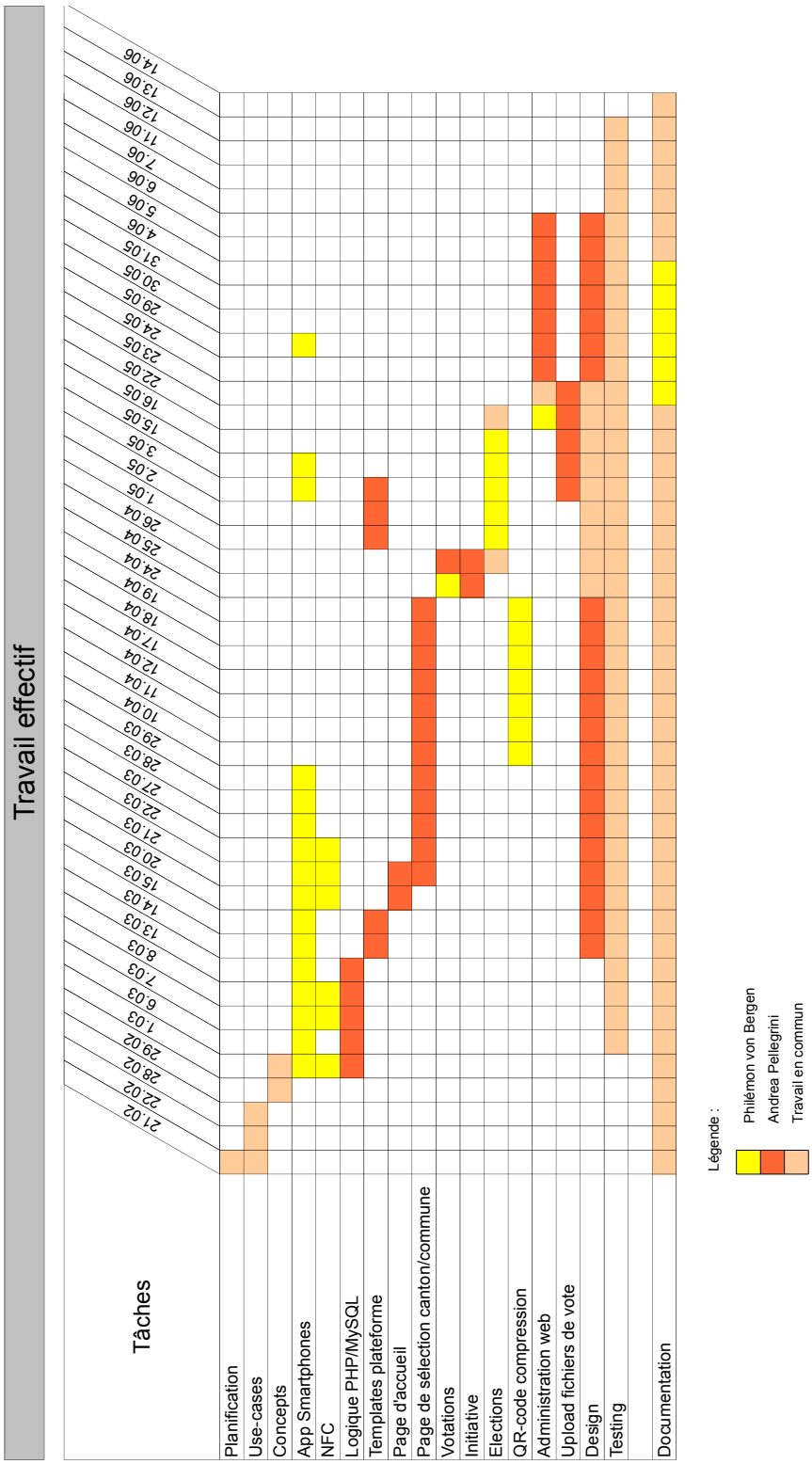



Figure 2.2.: Planning effectif

## 2.3. Commentaires

Voici quelques explications se rapportant aux différences que l'on voit apparaître dans les deux figures précédentes.

On remarque tout d'abord que les tâches de planification, de définition des use-cases et des concepts se sont déroulées plus ou moins comme prévu.

Pour le développement des applications de simulation de l'appareil et de la carte de vote, l'implémentation de la technologie NFC [1] s'est faite dès le départ, tout d'abord sous forme de tests, puis de manière adaptée au présent cas. Cela a permis de réduire le temps nécessaire par rapport au temps prévu. Suite à des essais d'utilisation de l'appareil et à de nouvelles idées, quelques heures ont été nécessaires plus tard pour l'adaptation des applications.

En ce qui concerne la plateforme de vote, la page de sélection de la commune a pris plus de temps que planifié, à cause de la logique et des effets qu'elle contient. De plus, c'était la première page qui utilisait le framework jQuery. Il a donc d'abord fallu acquérir un peu d'expérience avec cet outil.
Grâce à une synergie avec un autre projet, il a été possible de réutiliser du code pour la page d'élections, ce qui a permis d'éviter de dépasser le temps prévu. Autre différence encore : il était prévu que cette page soit réalisée par Andrea Pellegrini, mais elle a finalement été programmée par Philémon von Bergen.

Suite à quelques problèmes rencontrés dans la création des codes-barres, le temps planifié était considérablement insuffisant. Ce retard a, par chance, pu être compensé par le temps gagné lors du développement des applications pour smartphones.

L'upload du fichier de vote et la partie d'administration de la plateforme ont également pris un peu plus de temps que prévu. Cela a été compensé grâce au fait que le design de la plateforme a été réalisé tout au long de la création des pages, et non à la fin comme cela était pensé initialement. Ces deux parties ont été principalement réalisées par Andrea Pellegrini, alors que Philémon von Bergen s'est plutôt occupé de la documentation pendant ces quelques jours, notamment pour la rédaction des textes pour le livre de travaux de bachelor ainsi que pour le poster de présentation.

En conclusion, on retrouve donc, dans la plan du travail effectué, la structure prévue au départ, avec, certes, quelques variations.

1. Near Field Communication, voir chapitre 3

# 3. Simulation de l'appareil de vote

Comme mentionné précédemment, l'appareil de vote ainsi que la carte de vote doivent être simulés avec des smartphones. Tous deux doivent communiquer ensemble pour obtenir des informations comme la langue d'affichage ou pour transmettre les votes. Pour cela, il a été décidé d'utiliser la technologie NFC, Near Field Communication, en français « Communication en champ proche » qui permet le transfert de données entre deux appareils distants d'au maximum quelques centimètres. Le NFC est implémenté dans plusieurs nouveaux téléphones portables et convient bien pour simuler l'insertion de la carte dans l'appareil de vote.

L'appareil de vote a deux fonctions principales. Il doit :
– permettre de voter
– offrir la possibilité de changer le code PIN de la carte

## 3.1. Description du cycle

Le cycle de vote peut être divisé en plusieurs parties :
– Accueil
– Scan de codes-barres
– Confirmation du choix
– Introduction du PIN
– Finalisation du vote

**Accueil** Pour démarrer l'application, il faut envoyer le premier message NFC de la carte à l'appareil de vote (simulation de l'introduction de la carte dans l'appareil). Si la carte n'est pas bloquée, l'appareil affiche le menu principal où l'utilisateur peut choisir de voter ou de changer son code PIN.

**Scan de codes-barres** La partie de scan débute avec un message invitant l'utilisateur à scanner un code-barres. Une fois cela réalisé, l'appareil analyse le contenu du codes-barres. Le résultat du vote peut être réparti sur plusieurs codes-barres (voir chapitre 5.4). Dans ce cas, il faut scanner les codes restants. A cette étape, la signature numérique de la question doit être vérifiée (voir également chapitre 5.4). Si des erreurs surviennent pendant ces contrôles, des messages sont affichés.

**Confirmation du choix** Afin de vérifier son choix, l'utilisateur doit contrôler si le texte affiché sur l'écran de l'appareil correspond bien à ce qu'il désirait voter. Pour cela, il doit faire défiler le texte jusqu'à la dernière ligne. Il peut ensuite confirmer ou annuler son choix. S'il le valide, l'appareil va vérifier que le vote n'existe pas déjà sur l'appareil, auquel cas il en informe l'utilisateur, puis il va chiffrer le vote. Dans ce projet, le chiffrement est simulé, car la cryptographie n'a pas été implémentée.
Après cela, le choix est offert à l'utilisateur de continuer ou de terminer de voter. S'il choisit de continuer à voter, le cycle reprend à l'étape de scan de codes-barres.

**Introduction du PIN** Suit alors l'introduction du code PIN qui permet de débloquer la carte afin de lui envoyer les fichiers. Si l'utilisateur entre trois PIN erronés, la carte est alors bloquée.

**Finalisation du vote** Si le PIN a été entré correctement, l'appareil va vérifier si les votes à envoyer n'existent pas déjà sur la carte. Le cas échéant, il le signale à l'utilisateur. Sinon, les votes chiffrés sont transférés vers la carte qui va les signer. Une fois la signature réalisée, la carte envoie une confirmation à l'appareil, puis elle peut être retirée (simuler par une pression sur le bouton OK) et l'appareil s'éteint.

Le changement du PIN, quant à lui, comprend les étapes suivantes :
– Accueil
– Introduction du PIN
– Introduction du nouveau code PIN
– Confirmation du nouveau code PIN
– Ecriture du nouveau code PIN sur la carte

Les étapes d'accueil et d'introduction du code PIN correspondent aux états portant le même nom dans le cycle de vote.

Une fois l'utilisateur authentifié grâce à l'introduction du code PIN, il peut entrer le code PIN qu'il désire. Il devra le confirmer une seconde fois, et si les codes correspondent et contiennent plus de quatre chiffres, le PIN sera écrit sur la carte.

La figure suivante montre un diagramme présentant en détail chacune des étapes décrites ci-dessus ainsi que les transitions possibles d'un état à l'autre. Cette image est une représentation complète du fonctionnement de l'appareil de vote.

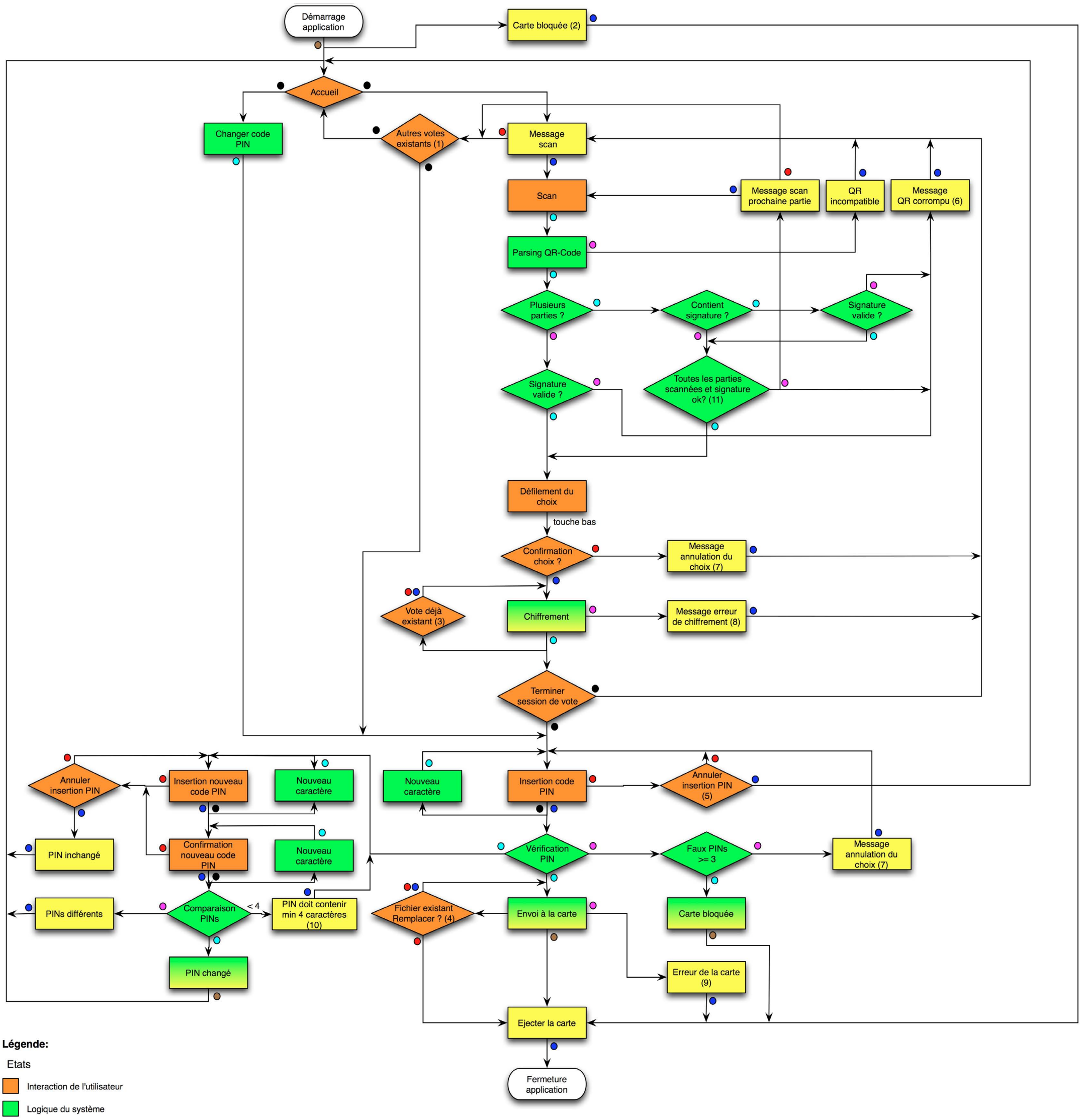



Figure 3.1.: Diagramme des états de l'appareil de vote

**Annotations dans le diagramme** Certains états nécessitent une petite explication. Il sont annotés avec un chiffre dans le diagramme. Voici les compléments d'information :
(1) L'utilisateur peut annuler le scan d'un code-barres et avoir déjà précédemment voté pour un autre objet. Dans ce cas, un message est affiché lui demandant s'il veut terminer ces votes ou s'il désire aussi les annuler. Ce message n'est évidemment affiché que si de tels votes existent.

(2) Il se peut que l'utilisateur bloque sa carte en entrant trois fois un code PIN erroné. Si l'utilisateur introduit une carte bloquée dans l'appareil, celui-ci la refusera. Dans notre cas, cette information se trouve dans le premier message NFC envoyé par la carte. Lors de la lecture de ce message, cette information est vérifiée et si la carte est bloquée, l'appareil passe directement dans l'état « Ejecter la carte ».

L'utilisateur peut voter plusieurs fois pour un même objet. Il faut différencier deux cas :
(3) L'utilisateur réalise un premier cycle de vote. Il décide de voter pour un second objet avant de copier le tout sur la carte. Le deuxième objet est le même que le premier pour lequel il a déjà voté. Dans ce cas, l'appareil doit lui demander s'il veut écraser le vote existant ou non.
(4) L'utilisateur peut avoir voté pour un objet qui se trouve déjà sur la carte. Dans ce cas, après qu'il ait entré son code PIN, le système doit lui demander s'il veut écraser le fichier déjà existant sur la carte. La liste des fichiers déjà existant sur la carte est transmise dans le premier message envoyé de la carte à l'appareil. Si l'utilisateur renonce à écraser le seul vote qu'il a fait, le système n'envoie pas de message vide, mais passe directement dans l'état « éjecter la carte ».

(5) Lorsque l'utilisateur annule l'introduction du code PIN, un message est affiché. Si l'utilisateur était en train d'introduire son code pour changer le PIN alors le message sera différent de celui qui s'affiche quand l'utilisateur est dans un cycle de vote.

(6) Si cette erreur survient, l'utilisateur doit scanner à nouveau tous les codes-barres correspondant à l'objet auquel il répondait quand l'erreur est survenue.

(7) Dans ce cas, seul l'objet annulé est effacé. Les autres objets précédemment confirmés ne sont pas éliminés.

(8) Si cette erreur survient, seul l'objet en cours est effacé, ce qui comprend toutes les parties du code-barres de cet objet.

(9) Si cette erreur survient, aucun vote n'est enregistré.

(10) Message affiché seulement si le nouveau code PIN contient moins de 4 caractères

(11) Si toutes les parties ont été scannées mais qu'il manque toujours des informations, le système indique que le code-barres est corrompu.

## 3.2. Architecture de l'application

Dans le cadre du projet 2, nous avions déjà réalisé une petite application pour simuler le déroulement du processus de vote. Dans cette simulation, seules les étapes principales avaient été prises en compte sans entrer trop dans les détails de façon à avoir un bon aperçu.
Pour la réalisation de l'application définitive dans le cadre du travail de bachelor, l'application du projet 2 a été reprise et ses fonctionnalités ont été étendues.
Il s'est rapidement avéré que l'architecture utilisée ne satisfaisait plus les nouvelles exigences. Le cycle de vote est très séquentiel. Lors du projet 2, il n'y avait que quelques états à simuler. L'application finale est cependant bien plus complexe. Lors du projet 2, la technique du switch-case sur un enum avait été choisie pour simuler la séquence, mais cette architecture était trop restrictive pour l'application finale.
Sur le conseil d'un superviseur, une machine d'état (state machine) a été mise en place. Elle permet un fondement plus modulable et bien plus facilement entretenable. C'est une state machine mise à disposition dans la bibliothèque Tungsten [21] qui a été utilisée.

Voici un diagramme des classes essentielles utilisées dans cette state-machine :

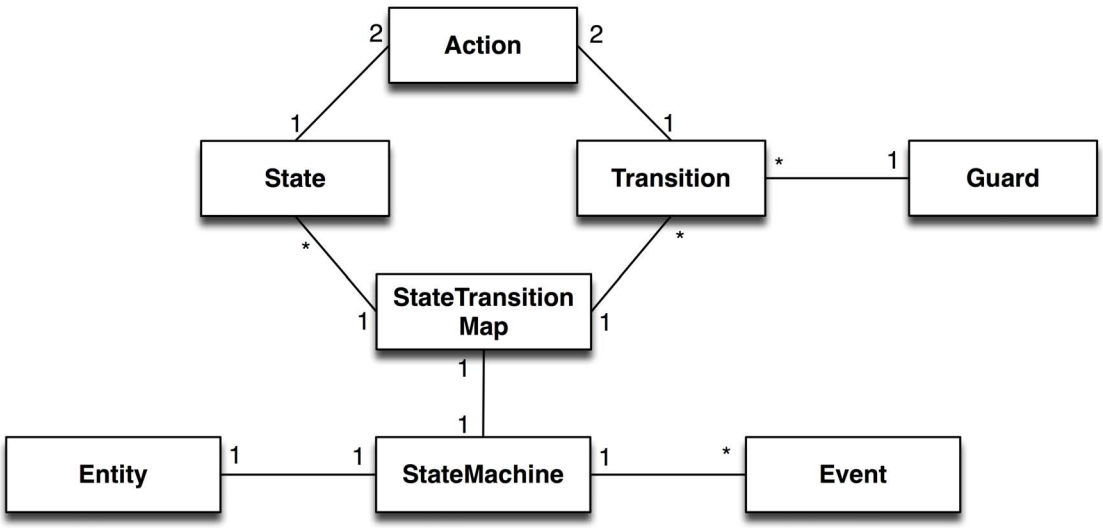



Figure 3.2.: Diagramme de classes de la machine d'états

La classe principale est la classe « State Machine ». C'est elle qui est responsable de changer d'état à chaque événement reçu. Elle fait référence à une entité (« Entity »), objet qui utilise la state machine et auquel on peut accéder pour exécuter des actions.
La classe « Event » représente les événements envoyés à la state machine. A chaque nouvel événement reçu, la machine considère l'état dans lequel elle se trouve et les transitions correspondantes à l'événement reçu. En fonction des transitions définies, la machine change d'état ou reste dans l'état actuel.
La classe « State Transition Map » est la classe dans laquelle on doit définir tout les états de notre machine (classe « State ») et les transitions possibles (classe « Transition »). Cette carte des états et des transitions doit être passée à la machine lorsque celle-ci est démarrée.
Chaque état et chaque transition peuvent effectuer une action (classe « Action »).
Lorsque la machine reçoit un événement, elle considère toutes les transitions sortant de l'état où elle se trouve momentanément. Pour chaque transition, elle considère si l'événement satisfait la condition définie dans la transition. Cette condition est décrite par la classe « Guard ».

Cette machine a donc été utilisée pour créer la séquence de l'appareil de vote. L'application du projet précédent à dû être retranscrite en reprenant les fonctionnalités déjà implémentées et les introduisant dans les états correspondants de la state machine.

## 3.3. Développement et fonctionnalités

Notre expérience dans le développement sur Android au début de ce projet se limitait à quelques essais préparatoires que nous avions faits le semestre passé dans le cadre du projet 2. La programmation de l'application de l'appareil de vote et de la carte de vote a donc pris passablement de temps, puisqu'il nous a fallu beaucoup de recherches dans la documentation d'Android [3] et sur des forums [18]. Ce chapitre traite de la façon dont nous avons implémenté certaines fonctionnalités de l'appareil de vote.

### Logging

Android possède un outil de logging pouvant être visualisé dans un environnement de développement. Cependant, la bibliothèque de la machine d'état nécessite Log4J [4]. Log4J ne fonctionne pas sur Android parce qu'il se base sur des classes qui ne sont pas supportées dans Android. Il a donc fallu rechercher une solution permettant d'allier

ces deux outils. Android-logging-log4j [14] permet de diffuser les logs sur le système de logging d'Android ainsi que dans un fichier de logs. Cet outil nous a été très utile pour debugger nos applications.

## Communication NFC

Une autre tâche était de comprendre le fonctionnement de la technologie NFC sur Android et de l'appliquer à notre projet. Pour cela, nous avons suivi un tutoriel [17] qui nous a grandement facilité la tâche. Combiné avec la documentation d'Android, nous sommes finalement parvenus à nos fins.
Sur les smartphones qui nous ont été mis à disposition par l'école, la puce NFC se trouve au dos de l'appareil. Pour les faire communiquer, il faut donc les coller dos à dos à une distance d'au maximum un centimètre et demi. Cette contrainte ne nous permet donc pas d'avoir une communication en continu entre les appareils. Il faut donc anticiper l'envoi d'informations qui seront nécessaires pour le bon déroulement d'un cycle de vote. Voici ci-dessous un tableau listant les données envoyées en NFC. Les données présentes sur la carte sont détaillées dans le chapitre 4.

| Données | Direction | Moment d'envoi | Description |
|---|---|---|---|
| Langue | Carte vers appareil | A l'allumage | Le premier message NFC correspond à la simulation de l'insertion de la carte dans l'appareil. La langue est transmise à ce moment de façon à afficher les textes de l'appareil dans la langue désirée. |
| Code PIN | Carte vers appareil | A l'allumage | Le code PIN doit être connu de l'appareil pour qu'il puisse le vérifier. Il est transmis au début afin d'éviter une communication supplémentaire avec la carte. |
| Salt [1] pour nom des fichiers | Carte vers appareil | A l'allumage | Salt utilisé dans le nom des fichiers créés par l'appareil de vote |
| Noms des fichiers présents sur la carte | Carte vers appareil | A l'allumage | L'appareil doit savoir quels fichiers sont déjà présents sur la carte de vote afin de pouvoir demander à l'utilisateur s'il veut les remplacer. |
| Fichiers de vote | Appareil vers carte | Après confirmation | Quand l'utilisateur a fini de voter les fichiers sont transmis vers la carte. |
| Confirmation de réception | Carte vers appareil | Après envoi des fichiers | La carte confirme qu'elle a bien reçu les fichiers. |
| Nouveau PIN | Appareil vers carte | Après changement du PIN | L'appareil envoie le nouveau PIN choisi par l'utilisateur à la carte. |
| Carte bloquée | Appareil vers carte | Après 3 faux PINs | L'appareil informe la carte que l'utilisateur a entré trois faux PINs et que la carte doit donc se bloquer. |
| Carte bloquée | Carte vers appareil | A l'allumage | La carte informe l'appareil qu'elle est bloquée. |

Table 3.1.: Messages transmis par NFC

L'envoi de données à l'allumage de l'appareil permet d'obtenir des informations qui sont nécessaires pour le déroulement du cycle de vote. Par exemple, si le PIN n'était pas transmis à ce moment-là, cela impliquerait une communication de plus avec la carte pour pouvoir le vérifier. Il faudrait donc une fois de plus coller les deux téléphones ensemble, ce qui n'est pas très pratique. Ces données sont donc mises à disposition de l'appareil dès le début. Ceci n'est utile que pour la simulation puisqu'il n'y a pas de communication continue. Sur l'appareil réel ce problème ne se posera pas.

## Internationalisation

Cette application doit impérativement supporter l'internationalisation. En effet, la Suisse possède quatre langues nationales et l'anglais est certainement un atout.

Android possède un système pour l'internationalisation d'applications. C'est la langue du téléphone qui est utilisée par défaut. Il est possible de spécifier quelle langue doit être utilisée, mais pour cela, il faut pourtant relancer l'application pour que la modification soit prise en compte. Cette contrainte posait un problème dans le cas présent, car la langue n'est reçue qu'un fois l'application lancée, elle ne peut donc pas être redémarrée. C'est la raison pour laquelle le système d'internationalisation proposé par Java avec l'utilisation des RessourceBundle a été utilisé.

## Scanner

Une autre problématique à laquelle nous avons été confrontés est celle du scan des codes-barres. L'appareil doit pouvoir lire les codes-barres bidimensionnels. Pour cela nous avons utilisé l'application Barcode Scanner de ZXing [22] qui peut être appelée depuis une autre application afin de réaliser le scan d'un code-barres. Elle retourne ensuite le résultat à l'appelant. Par ce moyen, nous pouvons obtenir le contenu du code-barres.
Pour installer cette application, nous avons dû configurer un compte Google dans les smartphones mis à disposition par l'école. Nous avons créé le compte « swissivi@gmail.com » avec, pour mot de passe, « uXHtxYkH ».

Comme cela est décrit dans le chapitre 5.4, le contenu des codes-barres est compressé afin de prendre le moins de place possible. Il s'agit d'une compression Zip. Les classes fournies par Java dans java.util.zip ont été utilisées pour décompresser les données.

Une fois le texte original obtenu, il reste à récupérer les données. Selon ce qui est mentionné au chapitre 5.4, elles sont codées au format JSON[2] . Nous avons donc utilisé une bibliothèque d'analyse JSON [6] qui nous permet de récupérer ces données.

## Multipart scanning

Il peut arriver que, pour un même objet (particulièrement pour les élections), un seul code-barres ne suffise pas pour contenir toutes les données. Dans ce cas, plusieurs codes-barres pour le même objet sont affichés sur la plateforme. Cela implique que l'appareil doit détecter ce cas de figure et attendre que tous les codes aient été scannés avant de soumettre le vote à confirmation.

Il avait été décidé que, pour faciliter l'utilisation du système, l'utilisateur peut scanner les différentes parties du code-barres dans n'importe quel ordre. L'appareil lui indique après chaque scan les parties manquantes.

Nous avions également pensé vérifier la signature électronique de la question aussi tôt que possible, de façon à ce que, si la signature est incorrecte, cela puisse être indiqué à l'utilisateur et lui évite de scanner inutilement d'autres parties.
Pour pouvoir effectuer ce contrôle le plus tôt possible, il faut savoir à quelle moment toutes les parties contenant un morceau de la question ont été scannées. Cela n'est pas chose aisée, car nous ne savons pas dans quel ordre seront scannées les différentes parties, ni sur combien de parties s'étend la question. Il a donc fallu trouver un moyen de contrôler si toutes les parties de la question ont été scannées.
Lors de la lecture de la première partie, le nombre total de parties est décodé et deux tableaux de cette taille sont créés. Dans le premier, les parties de la questions vont être stockées, et dans le deuxième, ce sont les parties de la réponses qui seront sauvegardées. Chaque partie des textes est enregistrée dans le tableau à l'index correspondant au numéro de la partie scannée. Ainsi, si une partie de la question est trouvée dans la partie 1 du code-barres, elle sera enregistrée à l'index 1 du tableau des réponses. Cette technique permet d'utiliser l'algorithme suivant :
– on parcourt le tableau des réponses jusqu'à ce qu'on rencontre la première partie de la réponse
– on mémorise l'index (1 dans le schéma ci-dessous)
– on parcourt le tableau des questions et on vérifie que, jusqu'à l'index mémorisé, aucun emplacement ne soit vide (2)

2. JSON (JavaScript Object Notation) est un format de données textuel, générique, dérivé de la notation des objets du langage ECMAScript. Il permet de représenter de l'information structurée.

Si c'est le cas, la question a été scannée complètement. Ceci repose sur la règle que la question apparaît toujours avant la réponse dans les codes-barres. Il reste encore à différencier le cas où une des parties contient un bout de la question et un bout de la réponse du cas où la fin de la question et le début de la réponse se trouvent dans deux parties distinctes. Cela influe sur le contrôle, dans le premier cas, l'algorithme se poursuit jusqu'à l'index mémorisé, dans le deuxième cas, jusqu'à l'index-1.
Ci-dessous une illustration schématisant l'algorithme :

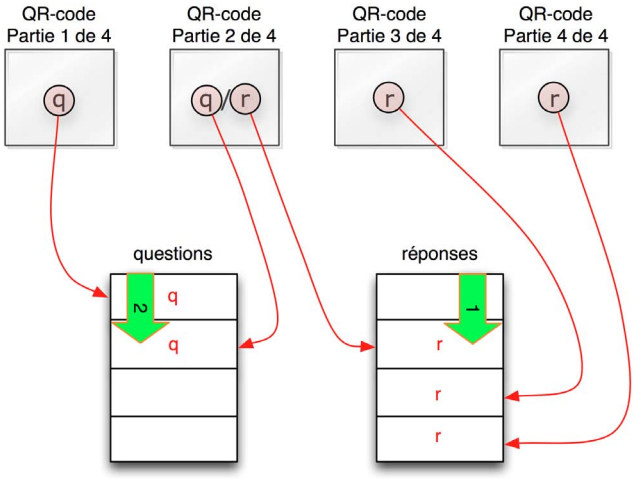



Figure 3.3.: Algorithme de détection de la complétude de la question

Grâce à ce moyen, il est possible de savoir quand la question complète a été scannée. On peut alors vérifier la signature et ainsi éviter des scans inutiles si celle-ci est erronée.

## Fichier de réponse

Une fois que le votant a scanné un code-barres et a confirmé son choix, le résultat doit être écrit dans un fichier qui sera transmis à la carte de vote, puis chargé dans l'urne électronique. Ce fichier va subir plusieurs opérations cryptographiques et devra ensuite être analysé une fois arrivé à sa destination finale.

Le langage XML, qui est facilement analysable et qui peut être chiffré et signé numériquement de façon très pratique, semblait être le plus adapté à cette situation. Pour cela, nous avons utilisé une bibliothèque Java nommée JDom [8] qui permet de créer un fichier XML en Java.

## Remplacement de fichiers existants

Il se peut qu'un utilisateur vote deux fois pour le même objet. Si c'est le cas, il faut l'en avertir et lui demander s'il désire conserver l'ancien ou le nouveau vote.
Ce cas de figure peut apparaître dans deux situations. La première serait celle ou l'utilisateur vote pour un premier objet. Lorsque la question lui est posée s'il désire finaliser son vote ou continuer à voter, il choisit de continuer. C'est alors qu'il vote à nouveau pour le même objet. Le résultat de son premier vote est toujours sur l'appareil puisqu'il n'a pas encore finalisé ce vote. L'appareil doit donc lui demander lequel il veut conserver.
Le second cas de figure apparaît quand un vote a déjà été finalisé et copié sur la carte. Si l'utilisateur vote à nouveau pour le même objet, un message doit lui indiquer, lors de la copie du vote sur la carte, que celui-ci y est déjà présent. De cette façon, il peut décider lequel il désire garder.

Ces deux événements se ressemblent beaucoup, la seule différence est qu'une fois le vote provoquant le conflit se trouve sur l'appareil et dans le deuxième cas, il se trouve sur la carte. Ce sont pourtant deux situations à gérer différemment.

## Convivialité d'utilisation

**Design** Pour obtenir un outil de démonstration convainquant, nous avons mis un certain poids sur l'apparence de l'appareil et de la carte de vote. Pour l'appareil de vote, nous avons repris une photo de l'ancien appareil d'e-banking utilisé par la Poste que nous avons un peu modifié pour satisfaire à nos besoins. Pour la carte de vote, c'est une carte de crédit qui a été légèrement adaptée.


Figure 3.4.: L'appareil de vote

**Vibration à l'appui des touches** Pendant la phase de test, nous avons remarqué qu'appuyer sur l'écran du smartphone ne donnait pas l'impression de presser sur des boutons, et qu'il était parfois difficile de savoir si la pression a bien été reconnue par le smartphone. Pour cela, nous avons ajouté une petite vibration à chaque appui d'une touche, pour autant que cette touche ait une fonctionnalité dans cet état du cycle.

**Défilement des textes** Un autre inconvénient que nous avons découvert pendant le testing est que l'écran de l'appareil de vote ne contient que deux lignes alors que certains textes dépassent cette limite. Nous avons donc dû mettre en place un moyen visuel permettant d'informer l'utilisateur qu'une partie du texte est masquée. Nous avons pensé à afficher une petite flèche en direction du bas quand il faut faire défiler le texte et une flèche en direction du haut si des lignes précédentes sont masquées. De cette façon l'utilisateur comprend rapidement s'il doit faire défiler des textes.

**Textes** Les textes affichés sur l'écran de l'appareil de vote jouent également un grand rôle pour la convivialité d'utilisation. En effet, si ceux-ci sont trop longs, il faudra utiliser le défilement pour les lire ce qui n'est pas pratique. S'ils ne sont pas clairs, l'utilisateur ne saura pas ce qu'il doit faire et va se décourager, entraînant ainsi le risque qu'il abandonne l'utilisation de l'appareil. Il est donc primordial pour l'acceptation du système, d'avoir des textes clairs et concis. Il est également utile d'avoir une structure dans les messages. Pour cela, nous avons classé les textes de l'appareil selon les catégories suivantes :

- **Messages informatifs** : ces textes donnent une information à l'utilisateur et doivent être quittancés par une pression sur la touche OK.
  Exemple : « Code-barres invalide »
- **Messages invitant à l'action** : ces textes informent l'utilisateur qu'il doit réaliser une action.
  Exemple : « Scanner le code-barres »
- **Questions OK/Annuler** : ces textes représentent les questions auxquelles on peut répondre par oui ou non, respectivement par OK ou Annuler.
  Exemple : « Ce vote existe déjà ! Ecraser ? »

– **Questions numériques** : ces textes s'appliquent au cas où l'utilisateur peut choisir des actions. La sélection se fait à l'aide des touches numériques.
  Exemple : « 1 : Voter 2 : Changer PIN »
– **Messages sans validation** : ces textes informent l'utilisateur qu'une action est en cours (principalement utilisé pour les messages NFC).
  Exemple : « Préparation du vote... »

Dans chacune de ces catégories, nous avons essayé, autant que possible, d'avoir une logique grammaticale, c'est à dire de construire les phrases de la même façon. Cela donne à l'utilisateur une impression de déjà vu, et le met à l'aise.

# 4. Simulation de la carte de vote

La structure de l'application de simulation de la carte de vote est plus simple que celle de l'appareil de vote. La carte contient un certain nombre d'informations comme la langue préférée de l'utilisateur, un code PIN, et d'autres encore. Ces données, ainsi que le nom des fichiers de vote déjà présents sur la carte de vote, sont transmis à la simulation de l'appareil de vote lors de la première communication. Cela permet d'afficher l'interface de l'appareil de vote dans la langue de l'utilisateur, d'éviter une communication supplémentaire pour la vérification du code PIN ou de l'existence de fichiers sur la carte.

**Description du cycle** Dès que la langue a été sélectionnée, la simulation de la carte émet un message NFC contenant les données citées ci-dessus. Une fois ces informations envoyées, l'application attend de recevoir un message NFC de la part de la simulation de l'appareil de vote. Ce message peut commencer par « change pin » si l'utilisateur a changé son PIN, par « sig request » si l'utilisateur a voté et veut faire signer ses votes par la carte ou par« block card » si l'utilisateur a entré trois fois un faux code PIN. Un message débutant par « change pin » sera suivi du nouveau PIN, une communication commençant par « sig request » contiendra également le contenu des fichiers de résultats à signer. En fonction du message reçu, l'application réalise l'action demandée. Si elle doit signer des votes, elle génère un message NFC « sig ok » dès que la signature est terminée. Dans notre cas, la cryptographie n'est pas implémentée, mais seulement simulée.
Le smartphone simulant la carte peut alors être connecté à l'ordinateur et les fichiers de résultats peuvent y être copiés pour l'envoi à l'urne électronique.

## Données stockées sur la carte de vote

Comme déjà mentionné précédemment, la carte contient un certain nombre de données. Chacune d'elles est propre à son propriétaire, car la carte de vote est une carte personnalisée. Voici une liste des informations qu'elle renferme :

– la langue que parle son possesseur qui permet d'afficher l'appareil de vote dans cette langue.
– un code PIN, afin que seul son détenteur légitime puisse l'utiliser.
– un salt pour le nom des fichiers
– une clé privée lui permettant de signer numériquement le vote. Cette clé ne doit jamais sortir de la carte.
– les fichiers contenant les résultats de vote.

L'utilisateur devrait supprimer les votes présents sur la carte, une fois qu'il les a chargés dans l'urne électronique. Cela permettrait de libérer l'espace pour les prochaines votations et éviterait également que, lors de la prochaine votation, l'utilisateur charge des fichiers datant d'une votation précédente dans l'urne électronique. Cet effacement pourrait, dans le futur, être géré par un logiciel permettant l'envoi des fichiers au bulletin board [1]. Dans notre simulation, la non suppression des fichiers ne pose cependant pas de problème.

Sur la carte réelle, il faudra certainement aussi inclure d'autre données comme le lieu de résidence du propriétaire de la carte par exemple. Cette information serait alors inclue dans le fichier de résultat et permettrait de réaliser des statistiques communales ou cantonales.

## Noms des fichiers de vote

Une fois les votes cryptés par l'appareil de vote, ils sont envoyés à la carte qui les signe et en crée des fichiers. Ces fichiers doivent porter un nom. Pour simplifier la vie à l'utilisateur, nous avons pensé faire apparaître dans le nom, la date de création dudit fichier. Ceci offre la possibilité à l'utilisateur de voir rapidement quels sont les fichiers actuels et quels sont les fichiers de votes d'une session précédente (s'ils n'ont pas été effacés).

1. Autre nom pour « urne électronique »

La date et l'heure de création (précise à la microseconde pour éviter d'avoir deux fichiers portant le même nom) ne suffisent pas, car il sera impossible à la carte de vérifier si un vote existe déjà pour l'objet que l'appareil est en train d'envoyer. Celui-ci ne pourra, par conséquent, pas en informer l'utilisateur. Nous avons donc pensé ajouter « l'event id » (l'id de la session de vote) et « l'object id » (l'id de l'objet) dans le nom de fichier afin de pouvoir réaliser cette vérification.
Le problème avec cette solution est la perte de confidentialité. En effet, un attaquant observant le trafic réseau lorsque le votant envoie son fichier au Bulletion Board peut savoir pour quel objet le votant a voté et pour lequel pas, simplement en lisant le nom des fichiers.
Pour éviter cela, nous avons décidé de hacher[2] « l'event id » et « l'object id » avec un salt prédéfini sur la carte. Le nom du fichier est donc constitué de la date de création suivie du hash. De cette façon, la tâche est rendue plus difficile à l'attaquant, qui devrait précédemment créer un dictionnaire de hash de tous les « event id » et « object id » avec tous les salts imaginables. Pour rendre cette attaque plus difficile, il faut choisir un salt avec un minimum de 80 bits pour atteindre la limite de calculabilité qui se trouve actuellement autour de $2^{80}$. Pour cette raison, il n'est pas possible d'utiliser le code PIN pour cela. D'autre part, le code PIN peut être modifié, ce qui perturberait le fonctionnement de l'écrasement des fichiers.
Dans le cadre de ce projet, nous avons utilisé un salt d'une force d'environ 120 bits. Nous avons généré un salt composé des caractères A-Za-z0-9 (62 caractères) d'une longueur de 20 caractères ce qui équivaut environ à une force de 120 bits car : $2^{120} = 62^{x} \Rightarrow x \cong 20$


Figure 4.1.: La carte de vote

2. Une fonction de hachage une fonction particulière qui, à partir d'une donnée fournie en entrée, calcule une empreinte servant à identifier rapidement, bien qu'incomplètement, la donnée initiale. C'est une fonction à sens unique.

# 5. Structure de la plateforme de vote

La plateforme de vote est, en réalité, un site web. Dans le cadre du projet 2, nous avions défini les interfaces, les pages du site et les langages de programmation que nous utiliserions. Pendant le travail de bachelor, nous avons dû mettre en place l'architecture du site.
L'architecture du site est réalisé en php et utilise le modèle de conception MVC (Model View Controller). Les effets graphiques (drag and drop, fenêtres de popup, etc) sont codés en javascript avec l'aide du framework jQuery.
Pour la création de l'architecture du site, nous nous sommes basés sur un tutoriel [11] qui explique comment créer un MVC en php. La figure ci-dessous illustre la philosophie du modèle de conception MVC :

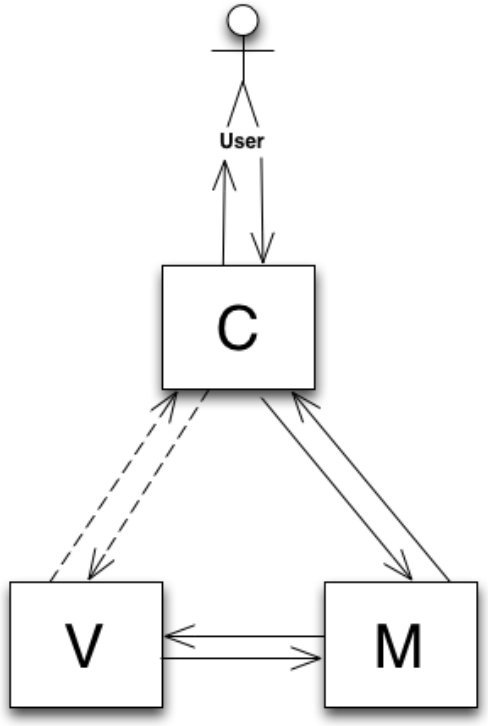



Figure 5.1.: Philosophie du modèle de conception MVC

Les lignes entre le Controller et la View sont traitillées, car il est possible d'appeler directement une View à partir du Controller, mais dans notre projet, nous utilisons toujours un Model nommé Template pour afficher correctement les pages.

L'utilisation d'un modèle de conception MVC permet de simplifier et d'accélérer la construction d'une page en utilisant des ressources déjà existantes. On peut par exemple réutiliser une view dans plusieurs pages et la positionner au bon endroit avec l'aide d'un modèle « template », lequel intègre des views dans d'autres views. On peut également utiliser les fonctions présentes dans les modèles dans plusieurs contrôleurs (par exemple, les fonctions du modèle « database » permettant de faire des requêtes à la base des données peuvent être réutilisées à différents endroits).

## 5.1. Architecture du site

Afin de mieux comprendre le fonctionnement du modèle de conception MVC mentionné ci-dessus, voici un exemple pratique.

Un utilisateur veut visiter la page appelée `https://swissivi.ch/index.php?rt=controller/method/var1`. Comme l'url nous l'indique, la page index.php est appelée et des paramètres y sont passés. Ceux-ci permettent de générer le contenu désiré. Toutes les requêtes passent par la page index.php, quelle que soit le contenu qu'on désire afficher. Cela permet de centraliser la logique dans la page index.php, qui délègue ensuite les différentes tâches. Ainsi, les contrôleurs sont toujours appelés par la page index.php. Il n'est pas possible d'appeler directement un contrôleur (avec son url, ex : `https://swissivi.ch/controller/myController.php`)
La page index.php, en plus d'appeler le bon contrôleur, charge aussi tous les fichiers nécessaires au fonctionnement du modèle de conception. Dans la figure ci-dessous, on peut voir la structure des objets de notre MVC et leur utilisation :

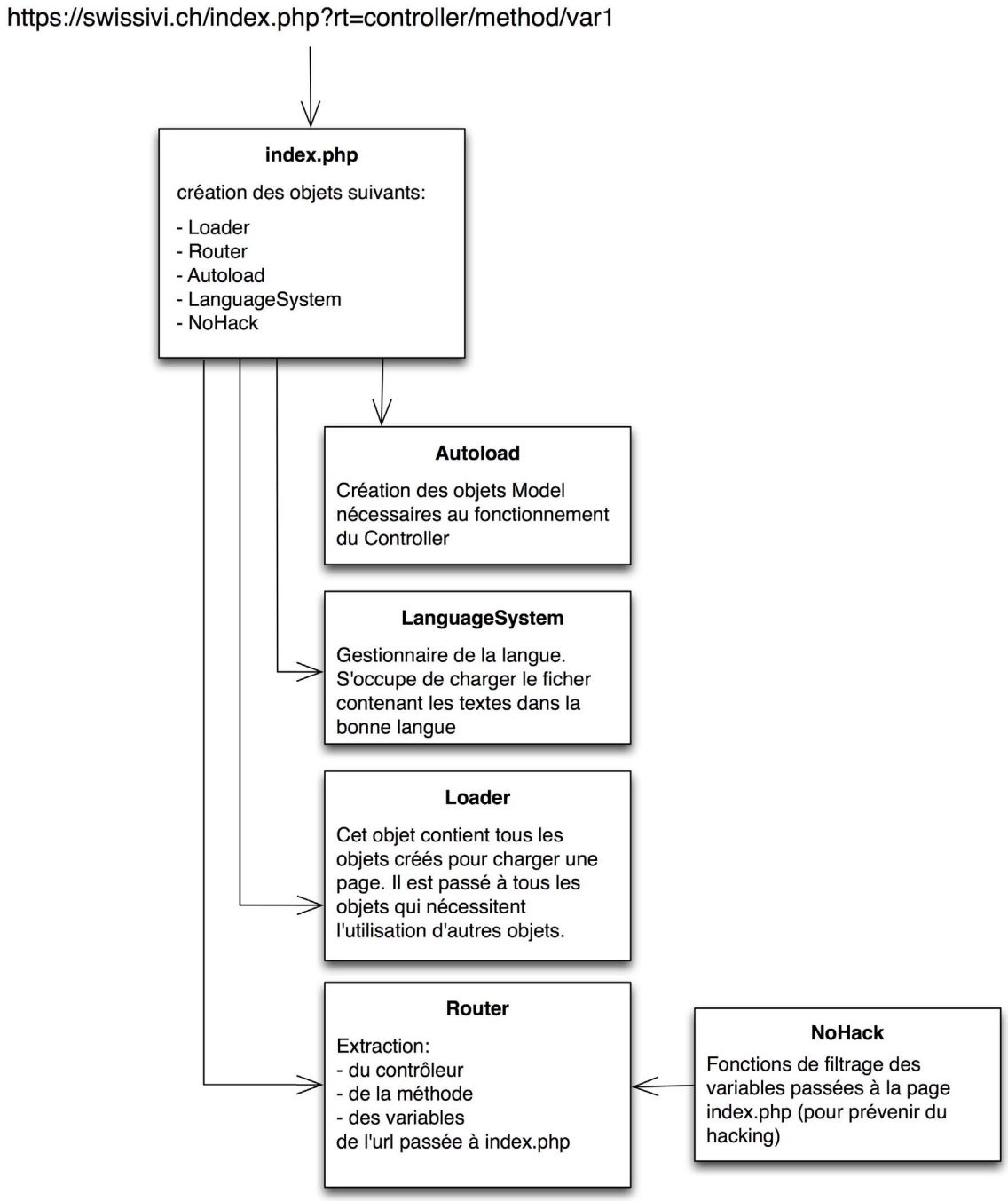



Figure 5.2.: Design de la structure des objets principaux du site

La page index.php s'occupe aussi du chargement du fichier de configuration où sont enregistrées toutes les confi-

gurations nécessaires au bon fonctionnement du site.
L'objet Autoload crée automatiquement les objets Model pour permettre au programmeur de ne pas devoir les créer à chaque utilisation de son contrôleur.
L'objet LanguageSystem est fondamental pour ce projet, vu que notre site doit supporter plusieurs langues. Il s'occupe de charger les bons fichiers de langue.
Le Loader est appelé dans chaque contrôleur, modèle ou view qui a besoin de récupérer un objet nécessaire à la logique de la page.
L'objet le plus important est le Router. Ce dernier s'occupe de créer le bon objet contrôleur qui est appelé via l'url. À l'aide de l'objet NoHack, le Router contrôle que dans la variable « controller/method/var1 », il n'y ait pas de code malicieux qui peut être utilisé pour attaquer le site.
Dans le schéma ci-dessous, on peut voir ce que l'objet Router est chargé de créer :

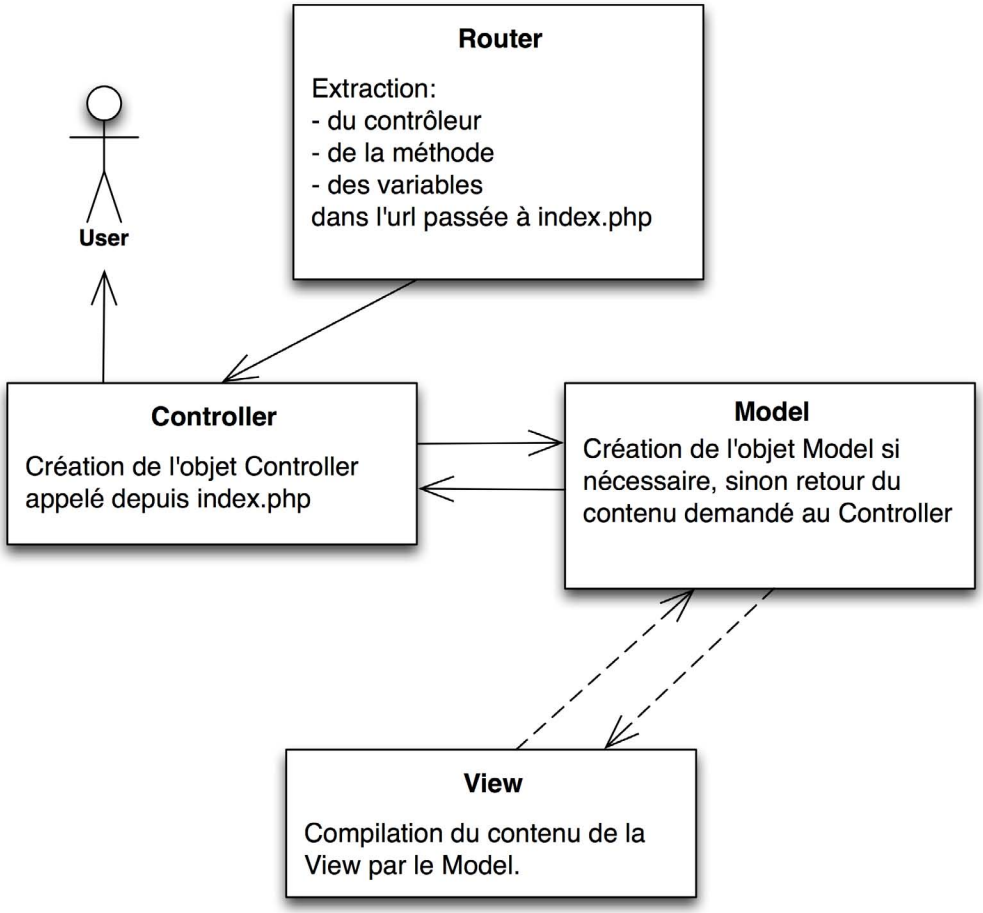



Figure 5.3.: Le devoir de l'objet Router

L'objet Controller, dans notre site, utilise toujours au minimum un objet Model, le « template ». Entre Model et View, on voit des lignes traitillées, car seul le modèle « template » appelle une ou plusieurs views. La création de l'objet Model est faite par le contrôleur seulement dans le cas où il n'a pas été créé dans l'objet « Autoload ». Le modèle « template » s'occupe de la compilation des Views pour les redonner au contrôleur qui s'occupe de les afficher à l'utilisateur.

Comme on peut l'observer dans l'image 5.4, le dossier « view » contient deux autres dossiers : « content » et « template ». Le dossier « content » renferme toutes les views qui sont appelées dans les « templates ». Autrement dit, le modèle « template » construit la page que l'utilisateur voit en utilisant un template (structure de la page) et une ou plusieurs views qui remplissent ce template.

Dans le site, nous avons utilisé beaucoup de requêtes « Ajax »[1] qui permettent de charger des contenus, autrement dit des views, sans devoir recharger la page. Ces requêtes appellent le contrôleur correspondant de la même façon que si on visite la page normalement. Le chapitre 6 contient des applications de cette technique.

La structure des fichiers constituants le site est la suivante :

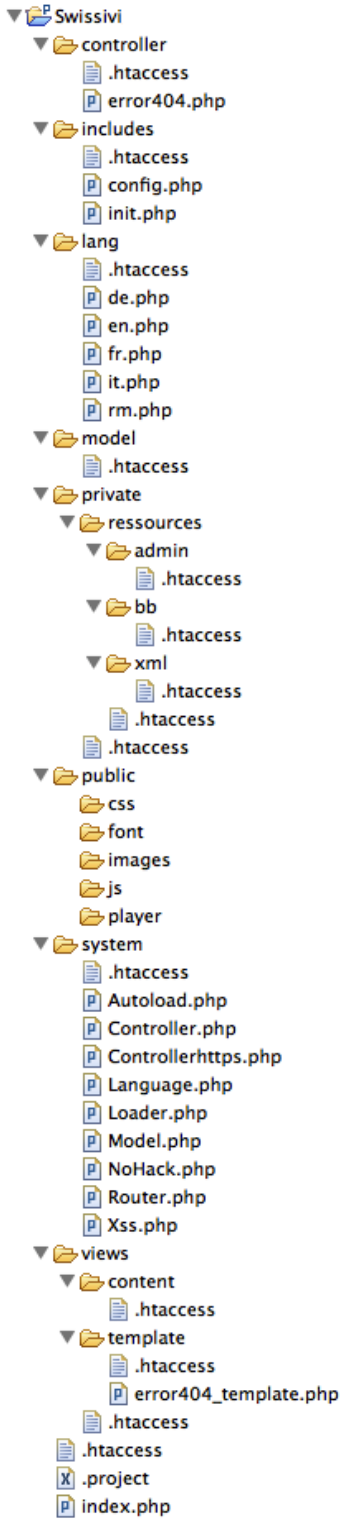



Figure 5.4.: Structure des dossiers du site

1. Asynchronous Javascript and XML.

Comme on peut le voir dans l'image ci-dessus, dans tous les dossiers se trouve, sauf le « public » qui contient les images et autres fichiers qui peuvent être publics, un fichier « .htaccess ». Ce ficher interdit l'accès aux fichiers depuis l'extérieur. Ce système est nécessaire afin de donner l'accès au site seulement en passant par la page index.php, laquelle s'occupe de générer la page désirée.

## 5.2. Réécriture de l'URL

Dans le chapitre 5.1, l'utilisation d'url du type `https://swissivi.ch/index.php?rt=controller/method/var1` est décrite. Afin de masquer la structure du site et pour simplifier l'apparence de l'url, nous avons pensé utiliser la fonction de réécriture de l'url.
La technique de réécriture de l'url permet de modifier la syntaxe des urls. Cela permet de donner un meilleur aspect esthétique aux adresses des pages. Nous avons décidé de donner la structure suivante aux nôtres : `https://swissivi.ch/controller/method/var1.html`. Le .html à la fin de l'adresse est un détail apprécié par les moteurs de recherches qui indique comment analyser la page. Il permet également de faire croire que `https://swissivi.ch/controller/method/var1.html` est une page simple et non une construction de plusieurs objets.
La réécriture de l'url est faite par le fichier « .htaccess » qui se trouve à la racine du site (même niveau de l'index.php). La règle utilisée pour est la suivante :

```
RewriteRule ^(.*)\.html$ index.php?rt=$1 [L,QSA]
```

La première partie « ^(.*)\.html$ » sert à informer le serveur que s'il trouve une syntaxe de ce type dans une requête, il doit la transformer en « index.php?rt=$1 » avant de l'analyser. La syntaxe utilisée pour créer cette règle est la suivante :
- ^ : début de la chaîne de caractères se trouvant après l'url de base du site
- **$** : fin de l'url
- **(.*)** : tous les caractères après l'adresse de base du site (sans le « .html ») sont mis dans la variable **$1**
- **.html** : l'url se termine en « .html »
- **index.php?rt=$1** : syntaxe dans laquelle l'url doit être transformée
- **[L,QSA]** : sont des paramètres. « L » indique au Rewrite Engine de ne pas exécuter les règles qui suivent. « QSA » est utile quand l'url contient plus de paramètres que prédéfini. Avec cette condition, le Rewrite Engine les ajoute à la suite. Ils peuvent ainsi être utilisés sur le serveur. Cette règle est importante, car elle est utilisée dans le script d'auto-complétion des noms des communes où un paramètre supplémentaire est passé dans l'url (plus de détails dans le chapitre 6.2).

Un autre avantage de l'utilisation de la réécriture de l'url est la possibilité de gérer les « erreurs 404 »[2]. Cette gestion permet d'afficher une page « d'erreur 404 » personnalisée. Les règles utilisées sont les suivantes :

```
RewriteRule ^public/(.*)$ public/$1 [L,QSA]
RewriteRule ^$ index.php?rt=home [L,QSA]
RewriteRule ^(.*)$ index.php?rt=error404 [L,QSA]
```

Les deux premières règles permettent l'accès aux ressources publiques et à la page index.php.

Cette technique a été réutilisée pour simplifier les adresses d'accès au bulletin board ainsi qu'à la plateforme d'administration. L'url `https://swissivi/bulletinboard/index.html` devient `https://swissivi/bb`. De même `https://swissivi/administrator/index.html` se transforme en `https://swissivi/administrator` ou `https://swissivi/admin`. Ces réécritures de l'url sont réalisées grâce aux règles suivantes :

```
RewriteRule ^bb$ index.php?rt=bulletinboard [L,QSA]
RewriteRule ^administrator$ index.php?rt=administrator [L,QSA]
RewriteRule ^admin$ index.php?rt=administrator [L,QSA]
```

2. Erreur 404 : erreur survenant quand une page inexistante est appelée

## 5.3. La base des données

Pour ce projet, la base des données est fondamentale. Elle doit contenir toutes les informations des votes. De plus, elle doit être conçue de façon à ce que la recherche pour la création des objets de vote soit performante. Pour cela, nous avons établi la structure suivante :

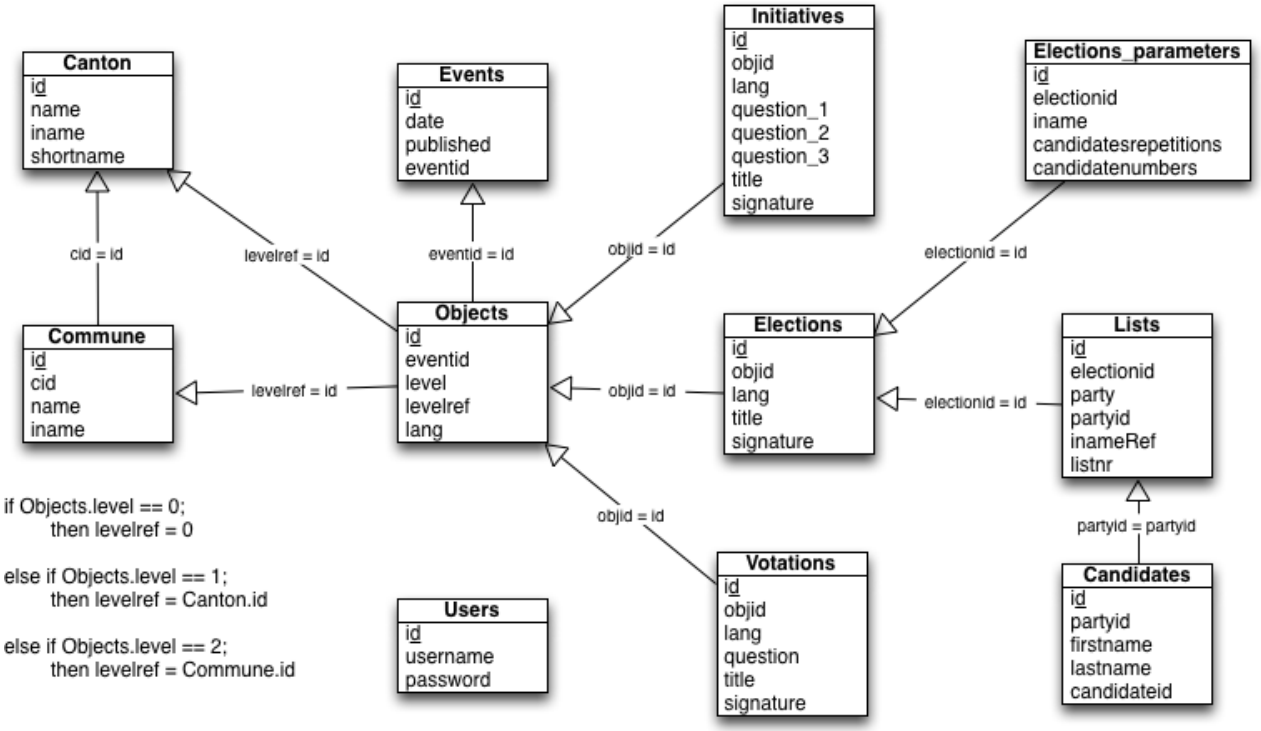



Figure 5.5.: Structure de la base des données

La table « Events » contient tous les événements qui sont ou qui ont été publiés. Un événement (qu'on pourrait aussi appeler « session de vote ») correspond à l'ensemble des votes qui doivent être réaliser pendant une journée. Par exemple, lors de l'événement du 23 octobre 2011, des élections fédérales étaient organisées.

On peut représenter un « Event » comme un ensemble qui contient un ou plusieurs objets « Objects » de vote. Un objet de vote fait toujours référence à un canton ou à un commune ou s'applique à toute la suisse. Cette référence nous permet, dès que le canton et la commune ont été choisis, de savoir quels objets de vote on doit afficher au votant.
Un « Object » peut être soit une votation, soit une initiative ou encore une élection. Ci-dessous, les paramètres nécessaires à chacun de ces types :

- **Votation** : La référence à l'objet de vote, la langue de la question, le titre et la signature de la question.
- **Initiative** : La référence à l'objet de vote, la langue des questions, le titre et la signature des questions.
- **Election** : La référence à l'objet de vote, la langue, le titre et la signature du titre. Une élection, au contraire des votations ou des initiatives, nécessite d'autres informations :
  - Lists : liste de candidats pour les élections. Une liste appartient à une seule élection. Une élection fédérale aura des listes différentes pour chaque canton. C'est à dire que pour la même élection, un habitant du canton de Berne verra des listes différentes s'afficher qu'un habitant du canton de Fribourg.
  - Candidates : les candidats des différentes listes. Un candidat appartient à une seule liste.
  - Elections_parameters : cette table contient les paramètres nécessaires à une élection. Ces sont le nombre maximum de candidats qui peuvent être élus, ou le nombre de répétitions admises de chaque candidat. Chaque entrée de cette table appartient à une élection et, dans le cas d'une élection fédérale, une entrée par canton est nécessaire, car, le nombre de candidats pouvant être élus dans le canton de Berne n'est pas le même que dans le canton de Vaud.

Toutes les communes et tous les cantons sont également stockés dans la base de données. De cette façon, chaque objet de vote peut avoir une référence sur une commune/un canton (attribut levelref de la table Objects). La table des communes est également utilisée dans la page de sélection de la commune (voir chapitre 6.2).

La table « Users » sert au contrôle d'accès à la page d'administration.

## 5.4. Génération des QR-codes

L'interface entre la plateforme de vote et l'appareil de vote se fait par l'intermédiaire de codes-barres bidimensionnels, plus précisément, à l'aide de QR-codes [3]. Ces codes contiennent le choix de l'utilisateur. Pour une votation par exemple, le votant a le choix entre trois QR-codes présents sur la plateforme. Le premier contient la réponse oui, le second la réponse non, et le troisième ne contient aucune réponse représentant ainsi le vote blanc.
Pour une élection, le QR-code contient les candidats choisis par le votant. Etant donné que l'appareil de vote n'a pas d'accès à internet et que les noms des candidats devront être affichés sur l'écran de l'appareil de vote afin de demander au votant de confirmer son choix, il est impératif que le nom figure en entier dans le code-barres.
Dans le canton du Tessin, lors des élections cantonales, jusqu'à 90 candidats peuvent être choisis. Si nous admettons un moyenne de 30 caractères par candidats (10 pour l'identification et 20 pour le nom) avec une moyenne de 7 bits par caractère, nous obtenons un total de 18 900 bits. A cela s'ajoute un certain nombre de données générales qui doivent également être intégrées dans le code-barres.
Le QR-code peut contenir jusqu'à 23 648 bits dans sa version la plus grande (version 40). Cependant, cette version de QR-code devient très difficile à lire avec une simple caméra de smartphone. Par conséquent, nous avons dû limiter la taille du QR-code à la version 18 (5768 bits) pour que la lecture du QR-code reste relativement facile. On se rend donc rapidement compte de la nécessité de générer plusieurs QR-codes et de répartir les données entre eux.
Il faut cependant rester conscient que la lecture d'un code-barres est quelque chose de relativement pénible pour le votant, il faut donc limiter au maximum le nombre de codes à scanner. Pour cela, il a été décidé de compresser (sans pertes) les données avant de les injecter dans les QR-codes. Nous avons utilisé l'algorithme Deflate qui est utilisé dans la compression Zip. Cette compression est assez efficace du fait que les données sont, pour la plus grande partie, du texte.

La génération de ces codes-barres doit se faire du côté client (au moins pour les initiatives et pour les élections) de façon à éviter que des informations sur le choix du votant transitent par internet. Il nous a donc fallu trouvé des bibliothèques permettant la création de QR-codes [5] et la compression Zip [9] en Javascript.

### 5.4.1. Contenu des QR-codes

Comme déjà cité précédemment, le(s) code(s)-barres devra(ont) contenir la réponse choisie par le votant. Mais à cela s'ajoutent beaucoup d'autres données.
Nous avons vu qu'un découpage des données en plusieurs parties était nécessaire. Par conséquent, chaque code doit savoir quelle partie des données il contient et combien de parties il y a, ceci afin que l'appareil de vote puisse reconstituer les données de façon consistante.
D'autre part, chaque partie des données doit savoir à quel objet de vote elle se rapporte. Ceci est défini par « l'event id » (identification de l'événement de vote, par exemple la date de la votation) et par « l'object id » (identification de l'objet de vote). De cette façon, on évite qu'un utilisateur scanne une première partie des données d'un objet, puis une deuxième partie se rapportant à un autre objet.
Il est également utile pour l'appareil de vote de savoir de quel type de votation il s'agit : une simple votation, une initiative ou une élection.
Ensuite, il est évident que la question et la réponse choisie doivent être contenues dans ces données afin qu'on puisse demander au votant de confirmer son choix. Si la question manquait, le votant ne pourrait pas savoir à quel objet se rapporte sa réponse. Par conséquent, la question est nécessaire.
A cela s'ajoute encore un contenu cryptographique : la signature numérique de la question. On pourrait imaginer qu'un attaquant modifie le texte d'une question de façon à ce que le votant réponde non, alors qu'il voudrait répondre oui à l'objet. Par exemple, l'attaquant peut modifier la question « Acceptez-vous l'initiative “Six semaines

3. Le code QR (Quick Response) est un type de code-barres en deux dimensions (ou code matriciel datamatrix) constitué de modules noirs disposés dans un carré à fond blanc.

de vacances pour tous" » en « Refusez-vous l'initiative "Six semaines de vacances pour tous" ». Le votant qui voudrait répondre oui à la première question répondra non à la seconde. Le résultat serait donc faussé. Pour cela, il faut que la question officielle soit signée numériquement, et que cette signature soit transmise à l'appareil de vote. Ce dernier pourra ainsi contrôler que la question n'a pas été modifiée.
En résumé, voici les données nécessaires dans un code-barres :

- le numéro de la partie
- le nombre total de parties
- le type de votation
- l'identification de l'événement
- l'identification de l'objet
- la signature de la question
- la question
- la réponse du votant

Pour faciliter le découpage de ces données, nous avons émis quelques règles :

- le numéro de la partie, le nombre total de parties, le type de votation, l'identification de l'événement, l'identification de l'objet, la signature de la question et le début de la question doivent être contenus dans la première partie. Cela implique que seules la question et la réponse peuvent être coupées.
- le numéro de la partie et le nombre total de parties doivent être placés tout au début de la partie
- la signature de la question doit être placée seulement dans la première partie

### 5.4.2. Découpage et compression

Pour générer les codes-barres, deux solutions se présentent à nous. La première, que nous nommeront Zip & Cut se base sur le principe qui consiste à compresser le contenu puis à le couper en fonction de la quantité de données pouvant être introduite dans un QR-code. La seconde, Cut & Zip, coupe le texte puis le compresse et l'injecte dans le code-barres.

**Zip & Cut** Le premier problème auquel nous sommes confrontés avec cette solution est que le numéro du code-barres et le nombre de code-barres sont zippés, et on ne peut donc pas, dans le cas d'un code-barre en plusieurs parties, indiquer à l'utilisateur quelles sont les parties qu'il reste à scanner.

Une solution à ce problème serait de ne pas compresser ces informations, mais de seulement compresser la question et la réponse, afin que les informations générales soient lisibles sans décompression.

Nous avons décidé d'utiliser la notation JSON pour encoder les données à injecter dans le code-barres. Cependant, la solution mentionnée ci-dessus n'est pas applicable avec JSON, car JSON n'accepte pas de données sous forme de byte array, forme correspondant à nos données compressées. La raison de cette incompatibilité est relativement facile à comprendre. JSON n'accepte que des caractères Unicode. Certaines valeurs de bytes sont converties en caractères illégaux et l'analyseur syntaxique JSON ne peut donc plus les décoder. Pour éviter ce cas de figure, on pourrait utiliser l'encodage en Base64 des données binaires.

Cet encodage représente 6 bits par un caractère alphanumérique déterminé. On obtient donc une string de caractères légaux pour JSON cette fois. Cependant, en passant cette chaîne de caractères à notre bibliothèque de génération de QR-codes, chacun de ces caractères va être représenté sur 8 bits. Par conséquent, l'encodage en Base64 est, dans notre cas, une expansion appliquée après une compression, autrement dit, deux actions qui s'annulent. Ce n'est donc pas adaptée à notre utilisation.

Une variante serait de ne pas utiliser JSON et de définir un parser nous-mêmes. De cette façon, nous aurions la possibilité d'avoir une partie du QR-code qui est compressée et une autre pas. Cependant, si, par malchance, dans la partie compressée, la suite de caractères que nous utilisons comme tag pour le parsing apparaît, le parser sera trompé et les données seront corrompues, ce qui ne doit en aucun cas arriver. L'encodage Base64 résoudrait ici encore le problème, mais pour la même raison que cité précédemment, nous ne pouvons pas l'utiliser dans notre cas.

**Cut & Zip** Nous nous sommes donc tournés vers la solution Cut & Zip, qui, elle, est compatible avec JSON, car le texte clair coupé est d'abord codé au format JSON, puis le tout est compressé et injecté dans le QR-code. Cette variante présente cependant un autre inconvénient. Il s'agit ici de couper le texte avant de le compresser. Cependant, la compression du texte dépend de la compressibilité de celui-ci. Comment alors savoir où couper le texte pour qu'il ait la taille désirée une fois compressé ?

A cette question, il n'existe pas de réponse exacte, il est toutefois possible d'utiliser la méthode Cut & Zip de façon satisfaisante à l'aide de quelques statistiques. Si nous voulons obtenir un QR-code de version 18 (5768 bits), quelques essais ont démontré qu'on pouvait accepter environ 8500 bits de données non-compressées. En fonction de la compressibilité du texte, la version du QR-code généré sera peut-être un peu différente de 18. Dans le pire des cas, lorsqu'aucune compression ne peut être appliquée au texte, la version serait de 23, ce qui, avec nos smartphones, est encore lisible. Pour notre simulation, ceci est donc encore acceptable et nous avons décidé d'implémenter ce système.

Pour essayer de palier au mieux à l'inconvénient mentionné ci-dessus, nous avons décider d'exécuter une compression de test sur le texte original entier afin d'estimer sa compressibilité. Ainsi, nous pouvons dynamiquement adapter la limite de bits de texte non-compressé autorisée. Cette compression demande, certes, quelques ressources, mais cela est tout à fait acceptable en comparaison du temps nécessaire pour la génération des QR-codes.
Cette compression de test n'est toutefois qu'une estimation de la réalité, car la compression finale sera exécutée sur les textes découpés et non sur le texte entier, ce qui pourra provoquer de légères différences. Par ailleurs, avec Cut & Zip, tout le contenu est compressé, y compris la signature qui est très peu compressible et qui peut influencer négativement la compression des autres données.

La méthode Cut & Zip nous offre toutefois l'avantage de pouvoir utiliser l'encodage JSON, qui est une méthode largement répandue sur le Web améliorant ainsi la compatibilité et la compréhensibilité de notre application, tout en simplifiant la méthode de parsing.

### 5.4.3. Génération en temps réel

Les QR-codes pour les votations peuvent être générés au moment de la création de la page. Les codes pour les initiatives et pour les élections doivent cependant être générés en fonction du choix de l'utilisateur (voir chapitre 6.5 et 6.6).
Les tests ont démontré que la génération d'un QR-code avec découpage et compression peut prendre quelques secondes (ce temps dépend des performances de l'ordinateur de l'utilisateur). Ce cas peut être rencontré surtout dans des élections. Le problème est que pendant cette génération des codes-barres, l'interface du navigateur se bloque et attend la fin de la fonction. Ceci implique qu'à chaque modification de l'élection (ajout ou suppression d'un candidat par exemple), l'utilisateur doit attendre que les codes-barres soient générés avant d'entreprendre une nouvelle modification. Ceci ralentit beaucoup l'interface et diminue sa convivialité d'utilisation.

Nous avons donc cherché une solution à ce problème et nous nous sommes orientés vers les Web Workers. Ils permettent l'exécution de scripts en tâche de fond. Cela nous permettrait de générer les codes pendant que l'utilisateur continue de travailler. L'inconvénient des workers est qu'ils ne sont pas supportés sur tous les navigateurs internet, en particulier Internet Explorer.
Comme nous développons ici un système de simulation, nous avons décidé avec nos superviseurs, de ne pas supporter ces navigateurs. La plateforme reste utilisable avec ce type de navigateurs, mais elle n'offre pas les performances optimales. Internet Explorer fait toutefois exception à cette règle. Ses différences d'interprétation du Javascript et du CSS[4] auraient demandé un trop grand effort d'adaptation qui n'est pas utile pour un Proof of Concept. Les utilisateurs de ces navigateurs recevront donc un message les informant que le navigateur n'est pas conseillé ou pas supporté.

4. Cascading Style Sheet, langage utilisé pour la mise en page d'une page HTML

# 6. Interfaces de la plateforme de vote

Ce chapitre décrit le fonctionnement de la plateforme basé sur la logique décrite au chapitre 5.

Les langages de programmation utilisés dans la partie graphique sont HTML, CSS, JavaScript. Les frameworks jQuery [1] [19] et jQuery-ui [2] [10] ont été utilisé pour la création des effets présents sur la plateforme.

**Programmation JQuery** Le code javascript jQuery doit être écrit dans une page HTML (entre le tag « <script> </script> »). Il ne peut pas être importé depuis un fichier dédié. Cette restriction est très handicapante pour ce projet, car certaines pages contiennent beaucoup de ligne de code jQuery. Ce code devrait être écrit directement dans les templates, ce qui n'est pas acceptable pour un projet comme celui-ci. Heureusement, il existe un plugin jQuery [20] qui permet de séparer le code jQuery et de le placer dans des fichiers à part. Grâce à lui, il nous a été possible de mieux organiser notre code.
Il existe également des techniques qui permettent d'optimiser le code jQuery afin de le rendre plus performant [13]. Nous avons essayé d'appliquer ces méthodes sur notre code afin d'accélérer l'exécution des scripts de notre plateforme.

**Le support des navigateurs** Plusieurs navigateurs peuvent être utilisés pour afficher la plateforme. Cependant, deux conditions minimales doivent être remplies. La navigateur doit accepter les cookies [3] afin que l'identifiant de la session puisse y être mémorisé.
De plus, le support JavaScript doit être activé, car une grande partie de la plateforme repose sur cette technologie. Si une de ces conditions n'est pas satisfaite, un message s'affiche à l'arrivée sur la plateforme informant l'utilisateur de cette nécessité. Le contrôle du support des Web Workers (voir chapitre 5.4.3) est également vérifié à ce moment.


Figure 6.1.: Message informatif sur les exigences de la plateforme

1. Framework Javascript qui porte sur l'interaction entre JavaScript (comprenant Ajax) et HTML, et a pour but de simplifier des commandes communes de JavaScript.
2. Partie graphique du framework jQuery.
3. Cookie : En informatique, un cookie constitue une suite d'informations envoyée par un serveur à un client, que ce dernier retourne lors de chaque interrogation du même serveur sous certaines conditions.

## 6.1. La page d'accueil

La page d'accueil d'un site web est très importante, car c'est la première page à laquelle l'utilisateur est confronté. Pour cette raison, nous avons cherché à implémenter quelque chose de simple et intuitif. La langue préférée du navigateur est automatiquement choisie comme langue d'affichage de la page d'accueil. Si l'utilisateur désire une autre langue, il peut la choisir dans la liste. Si le navigateur ne contient pas de langue préférée, c'est l'allemand qui est utilisé.

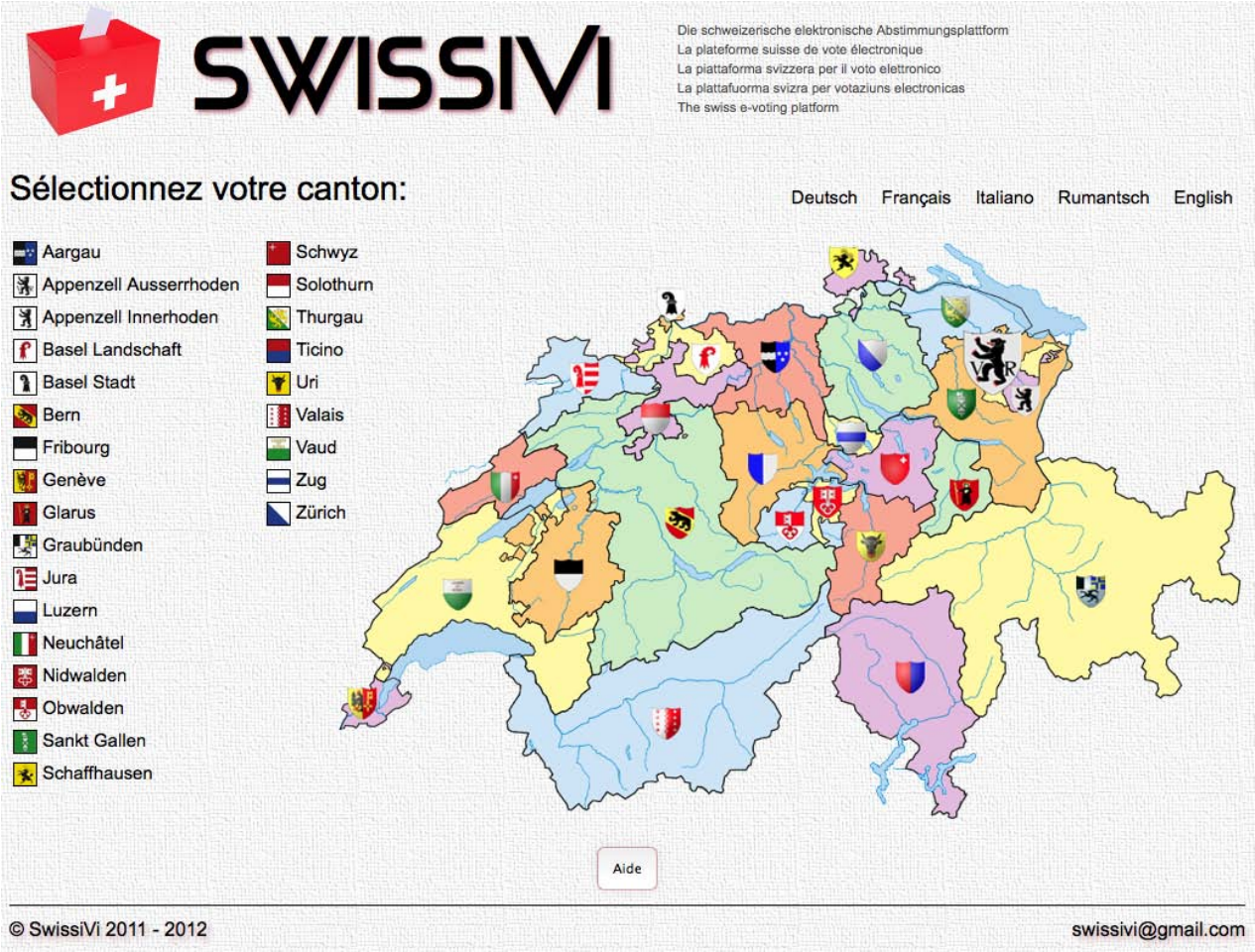



Figure 6.2.: La page d'accueil

La carte de la Suisse permet à l'utilisateur de choisir son canton. S'il préfère, il peut également le choisir dans la liste de gauche. Cette liste est principalement prévu pour les personnes à visibilité réduite. En passant la souris sur le nom d'un canton, le drapeau du canton correspondant est animé sur la carte. Un clic sur le canton affiche la page de sélection de la commune.

En dessous de la carte se trouve un bouton permettant d'afficher un guide d'utilisation de la plateforme. Nous avons réalisé une petite vidéo pour chaque étape démontrant comment utiliser le site.

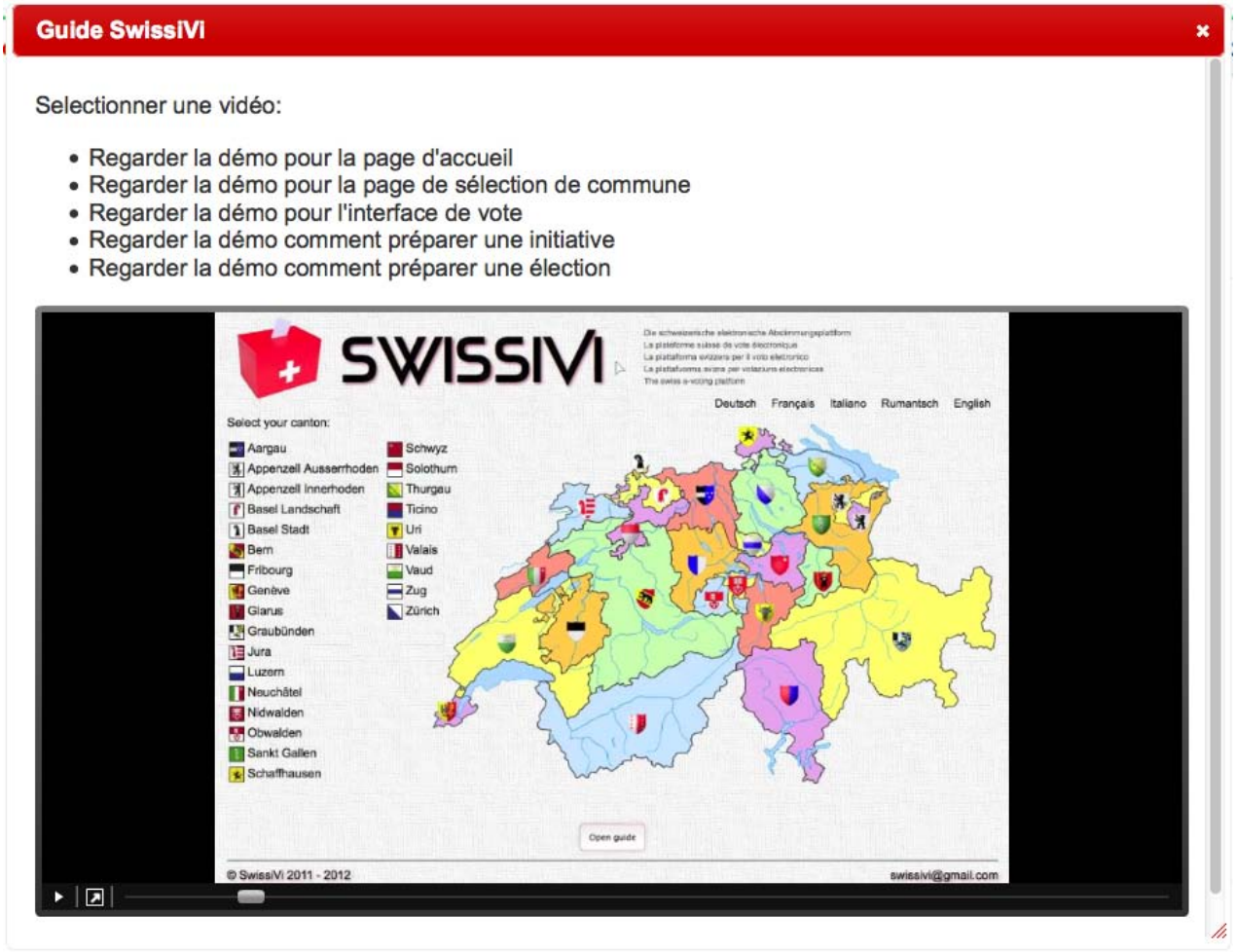



Figure 6.3.: Pop-up contenant le guide

A l'ouverture du guide, une requête Ajax permet de charger son contenu. Cela permet d'éviter de charger tout le guide à l'appel de chaque page. De même, les vidéos ne sont chargées qu'au moment où elles sont visionnées.

## 6.2. La page de sélection de la commune

La page de sélection de la commune a pour but de permettre à l'utilisateur de choisir sa commune de résidence. Cela permettra d'afficher les objets de vote correspondant à la commune et au canton sélectionnés. Cette page a également comme mission de donner un aperçu des objets pour lesquels les visiteurs devra voter. Chaque niveau est affiché dans un encadré séparé, et chaque encadré possède une animation à l'affichage.


Figure 6.4.: Pop-up de sélection de la commune

La page est bloquée de façon à ce que l'utilisateur soit obligé de choisir une commune. Un bouton d'aide est également disponible à cet endroit.

Le champ texte de gauche contient déjà le canton que l'utilisateur a sélectionné sur la page d'accueil. S'il le désire, il peut toutefois encore le modifier.
Le champ texte de droite permet de choisir la commune de résidence. C'est un champ à auto-complétion, c'est-à-dire que le visiteur commence à écrire le nom de sa commune, et des propositions de communes lui sont faites en fonction des lettres entrées. Les propositions dépendent du canton sélectionné, seules les communes de ce canton seront proposées. La liste des communes est générée à partir de la base de données. Pour ce champ, le plugin « Autocomplete » de jQuery a été utilisé. A chaque nouvelle lettre, une requête Ajax est générée, retournant la liste des communes.


Figure 6.5.: Exemple d'auto-complétion du nom de la commune

En passant la souris sur un des choix proposé, la commune est survolée est sélectionnée.

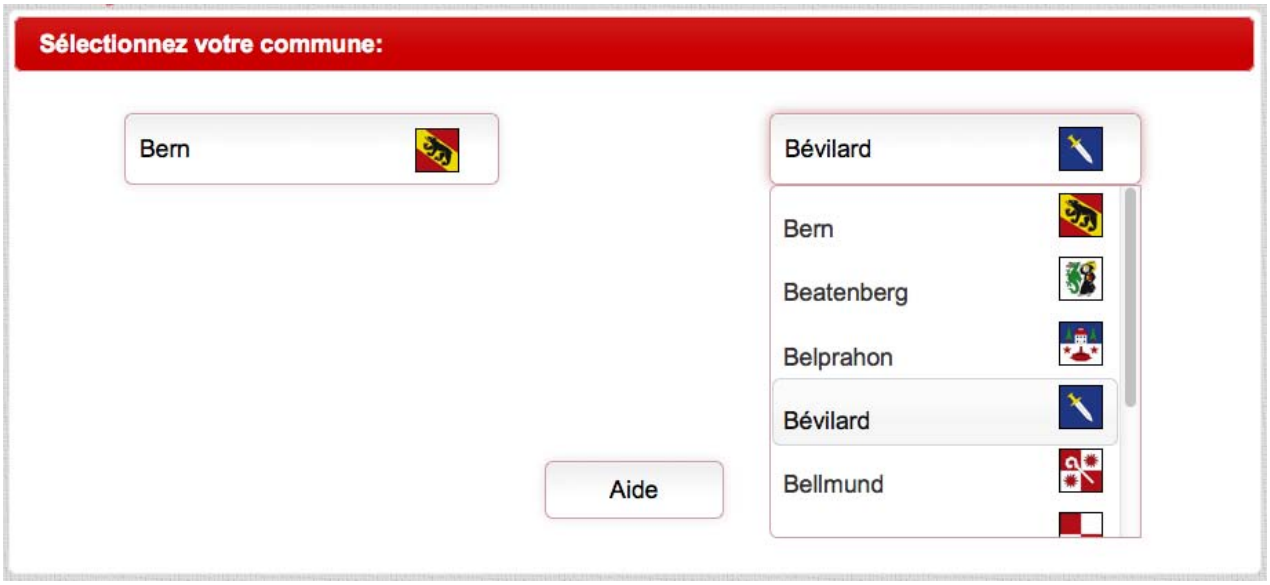



Figure 6.6.: Exemple d'auto-complétion du nom de la commune, avec le nom sélectionné

En cliquant avec la souris sur un des choix, ou en appuyant sur « Enter », la commune est enregistrée dans une variable de session. Ensuite, la boîte de dialogue s'agrandit pour afficher un aperçu des objets de vote. Les niveaux apparaissent l'un après l'autre afin de rendre l'utilisateur attentif à la répartition des objets en trois niveaux, fédéral, cantonal et communal.

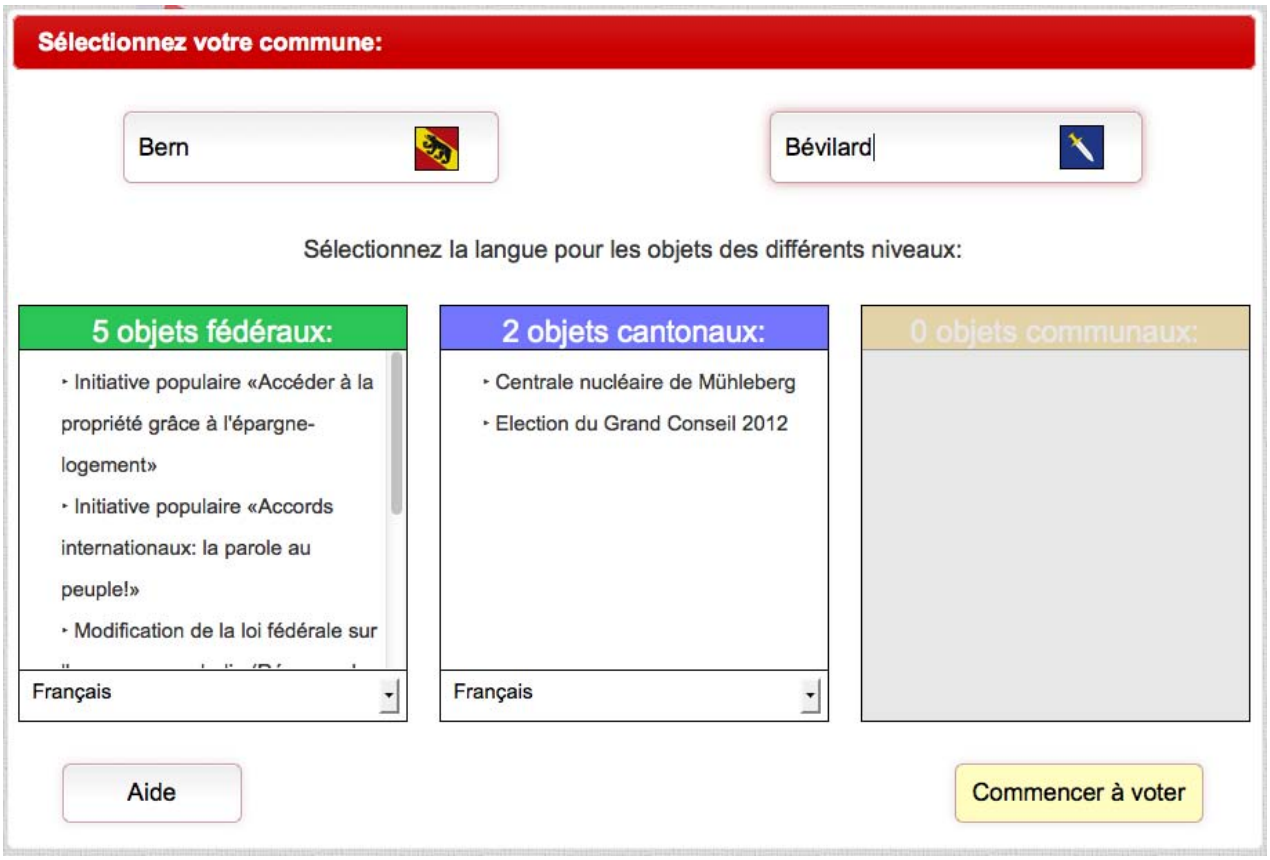



Figure 6.7.: Exemple de l'aperçu des votes

Le bouton « Commencer à voter » est montré à l'utilisateur seulement après que les sujets ont été affichés. Dans

l'image ci-dessus on peut bien voir que, si on n'a pas d'objet à voter pour un certain niveau, l'encadré correspondant est grisé.
Dans le titre de chaque niveau apparaît le nombre d'objets. Cela permet de rendre l'utilisateur attentif au nombre d'objets qu'il aura à voter pour chaque niveau.
Si l'utilisateur change de commune, la vue d'ensemble est effacée et le pop-up original est restauré.

**Les couleurs** Les couleurs de fond dans le titre des encadrés permettent de bien différencier les trois niveaux. Ces couleurs sont réutilisées plus tard dans l'interface de vote.

**Le changement de la langue** La langue choisie sur la page d'accueil correspond à la langue dans laquelle l'interface sera affichée. Les listes déroulantes visibles dans les encadrés des niveaux dans l'image ci-dessus permettent de choisir la langue d'affichage des objets de vote.
Ceci peut être utile dans le cas où les objets de vote n'existent pas dans la même langue que l'interface. Par exemple, un tessinois domicilié à Bienne pourrait choisir italien comme langue de l'interface. Les objets de vote pour le canton de Berne et pour la ville de Bienne n'existent cependant qu'en français et en allemand. Le tessinois pourra alors choisir parmi ces deux langues laquelle il préfère. Il peut également choisir la langue pour les objets fédéraux qui eux sont disponibles en italien. Par défaut, les objets s'affichent dans la même langue que l'interface, s'ils sont disponibles dans cette langue, sinon c'est l'allemand qui est choisi.

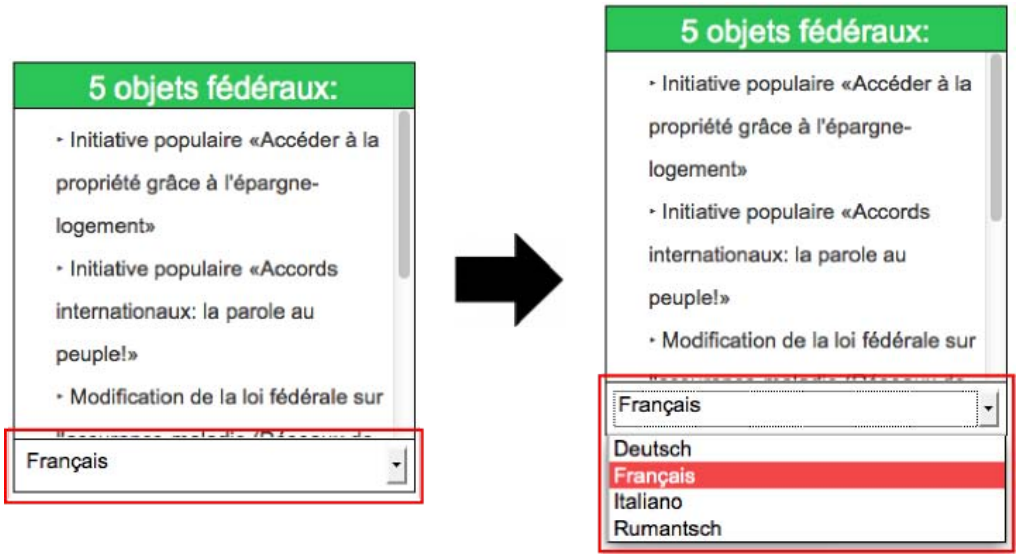



Figure 6.8.: Changement de la langue pour le niveau fédéral

**Les variables de session** Elles sont utilisées dans le cas de requêtes Ajax pour savoir pour quels canton et commune les réponses doivent être générées.

## 6.3. L'interface de vote

Cette page est celle où l'utilisateur va pouvoir voter. Elle est générée de façon un peu particulière. En effet, chaque niveau est réparti dans un onglet vertical apparaissant sur la gauche. Afin de rendre l'utilisateur attentif à cette répartition, l'affichage de l'interface se fait pas à pas. D'abord c'est le niveau communal qui est affiché, masqué ensuite par le niveau cantonal, et finalement c'est le niveau fédéral qui s'affiche.

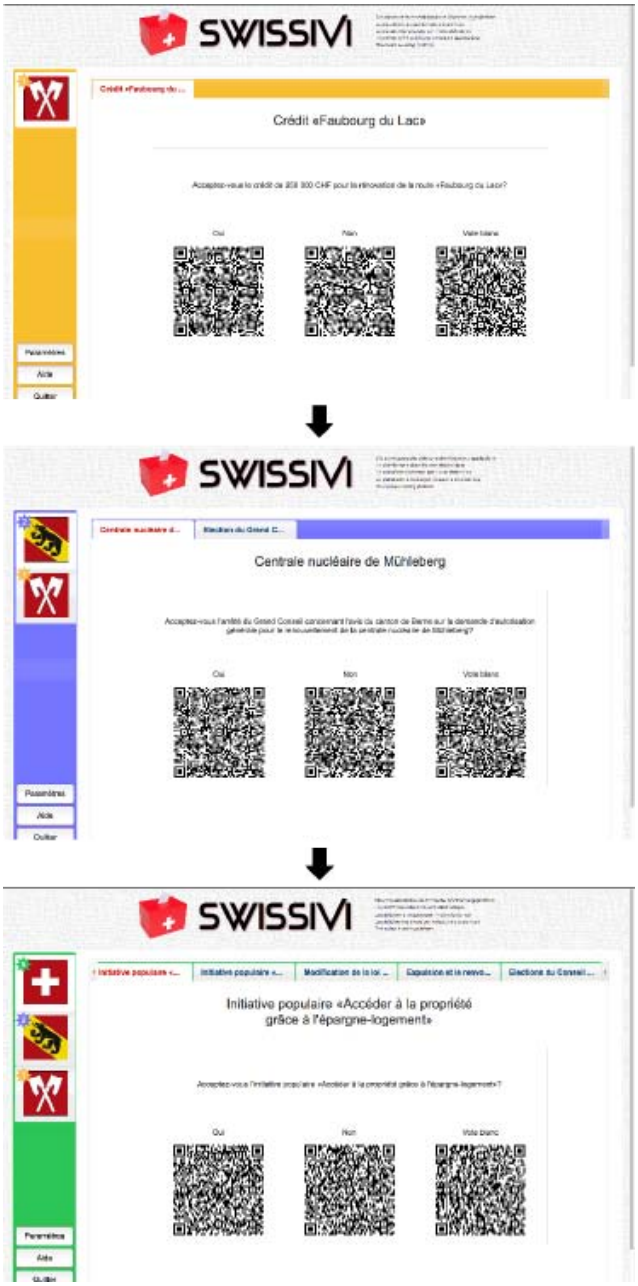



Figure 6.9.: Séquence de l'apparition des niveaux de vote

On remarque ici la même utilisation des couleurs que dans la page de sélection de la commune.

La navigation entre ces niveaux se fait à l'aide des onglets de gauche où on voit apparaître le drapeau suisse, le drapeau du canton et celui de la commune. Sur chaque onglet s'affiche le nombre d'objets dans ce niveau. Les onglets horizontaux séparent les différents objets de vote pour le niveau sélectionné.


Figure 6.10.: Page avec les objets de vote

Si un niveau n'a aucun objet de vote, l'onglet n'est pas sélectionnable. Le drapeau est toujours visible mais il est grisé, comme on peut le voir dans l'image ci-dessous.


Figure 6.11.: Page avec les objets de vote et un niveau vide

**Les boutons** présents en bas à gauche de la page permettent les actions suivantes :

– Paramètres : ouvre la page de « sélection de la commune » pour permettre au votant de changer de commune ou de canton ou de changer la langue des objets de vote.
– Aide : affiche le guide d'utilisation.
– Quitter : permet à l'utilisateur de retourner à la page d'accueil

**Les onglets des objets de vote** Le titre des objets de vote est généralement plus long que la taille de l'onglet. Nous avons donc dû trouver une solution pour raccourcir ces titres. Pour cela, un script nous permet de calculer combien de caractères peuvent être affichés dans l'onglet. On ajoute alors les points de suspension pour éviter de couper une lettre.
Pour toutefois pouvoir lire le titre en entier, si l'utilisateur positionne la souris pendant une seconde sur un onglet, une bulle [12] renfermant le titre complet s'affiche.


Figure 6.12.: Bulle avec le titre complet de l'objet de vote

Il peut arriver que le nombre d'objets disponibles pour un certain niveau soit trop grand pour que tous les onglets puisse être affichés sur la largeur de la fenêtre. Par défaut, jQuery affiche alors les onglets sur deux lignes, ce qui n'est pas très convivial.
Grâce à un plugin [1], ce problème peut être évité. Il permet de faire défiler les onglets pour atteindre ceux qui n'ont pas pu être affichés.


Figure 6.13.: Flèche de défilement dans les onglet

Une fois que cette page a été générée, toutes les informations nécessaires aux votes sont présentes du côté client. La connexion pourrait donc être interrompue à ce moment sans altérer le fonctionnement de la plateforme. Toutefois, si l'utilisateur désire consulter l'aide, ou qu'il change des paramètres comme la commune ou le canton de résidence ou la langue, une connexion à internet est à nouveau nécessaire.

## 6.4. La page de votation

Les pages qui contiennent une votation se présentent de la façon suivante :


Figure 6.14.: Page de votation

En plus du titre, on voit apparaître la question complète ainsi que les trois résultats possibles sous forme de codes-barres :
– Oui
– Non
– Vote blanc
Pour voter, le votant doit scanner le code-barres représentant la réponse qu'il désire.
Le « vote blanc » a été ajouté pour que les votes réalisés sur la plateforme aient exactement les mêmes possibilités que les votes sur papier.

## 6.5. La page d'initiative

Les pages qui contiennent une initiative sont construites comme suit :

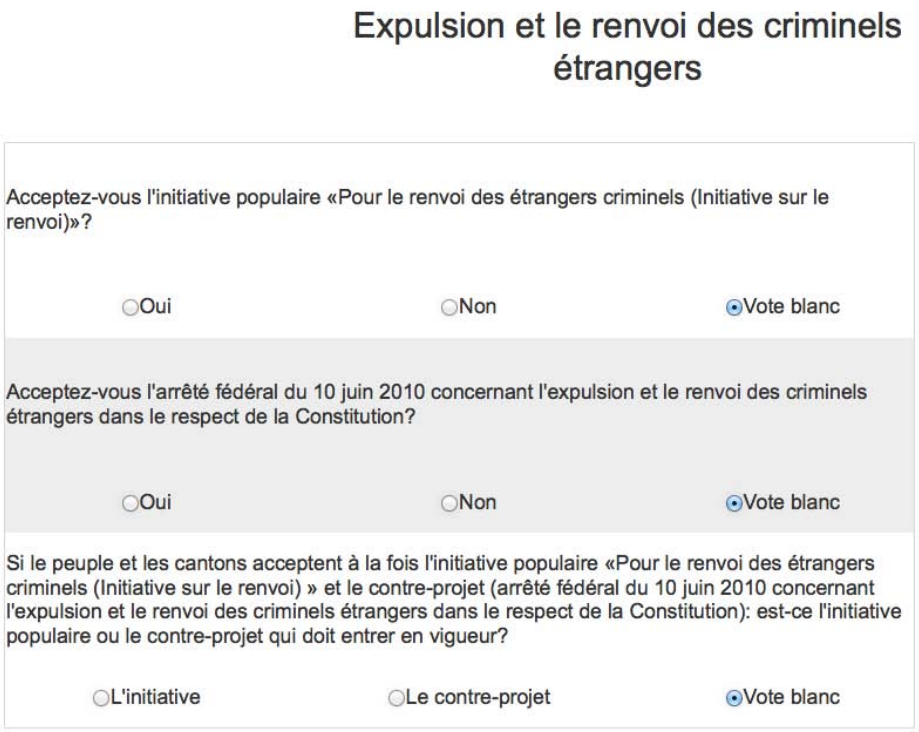



Figure 6.15.: Page d'initiative

Une initiative se compose généralement de trois parties : l'initiative proprement dite, le contre-projet et une dernière question permettant d'indiquer sa préférence entre les deux. Pour chacune de ces questions, les réponses possibles sont les mêmes que pour une votation, à savoir « oui », « non » ou « vote blanc ».
On remarque qu'il y a un seul code-barres à scanner pour trois questions. Celui-ci est généré en fonction des choix sélectionnés par le votant. Cela permet d'éviter à l'utilisateur de scanner un code-barres par question. L'inconvénient apporté par cette solution est que la plateforme pourrait savoir ce que le votant a choisi comme réponse. Pour diminuer ce risque, le code-barres est automatiquement généré à chaque modification. Il suffit donc de sélectionner différents choix avant de réaliser le choix définitif et de scanner le code-barres. La plateforme ne peut pas savoir lequel de tous les codes générés a été scanné. Ainsi, on peut facilement déjouer un éventuel malware qui analyserait les choix réalisés sur la plateforme.

## 6.6. La page d'élection

Une page contenant une élection se présente ainsi :


Figure 6.16.: Page d'élection

Cette page est construite de la façon suivante : à gauche, on trouve les listes prédéfinies avec tous les candidats, la partie médiane est consacrée à liste personnalisée et à droite on trouve le(s) code(s)-barres à scanner. Celui-ci contient toutes les données présentes dans la liste personnalisée. Le code affiché avant toute modification contient une liste vide, représentant ainsi un vote blanc. Ce code peut être affiché à tous moments en réinitialisant le contenu de la liste personnalisée. Au départ, seul un code est affiché. En fonction du nombre de candidats choisis, jusqu'à trois codes peuvent être nécessaires pour contenir toutes les données. La raison est décrite au chapitre 5.4.

**Les listes prédéfinies** Les listes prédéfinies sont les listes de candidats telles qu'elles sont créées par le parti. Elles correspondent aux listes préimprimées qu'on reçoit actuellement par la poste. Aucune modification ne peut être entreprise dans ces listes.

**La liste personnalisée** La liste personnalisée permet de composer son propre choix de candidats et le choix d'un parti. Elle correspond à la liste préimprimée vide.
Pour ajouter des candidats à cette liste, on a deux possibilités : soit on les ajoute un après l'autre ou on copie une liste entière.
Deux fonctions de base permettent de réaliser cela :

– **drag & drop** : on prend un candidat ou une liste de la partie de gauche et on la glisse dans la partie médiane. Une zone de couleur jaune indique où déposer la liste ou le candidat. L'image suivante illustre la séquence décrite ci-dessus :

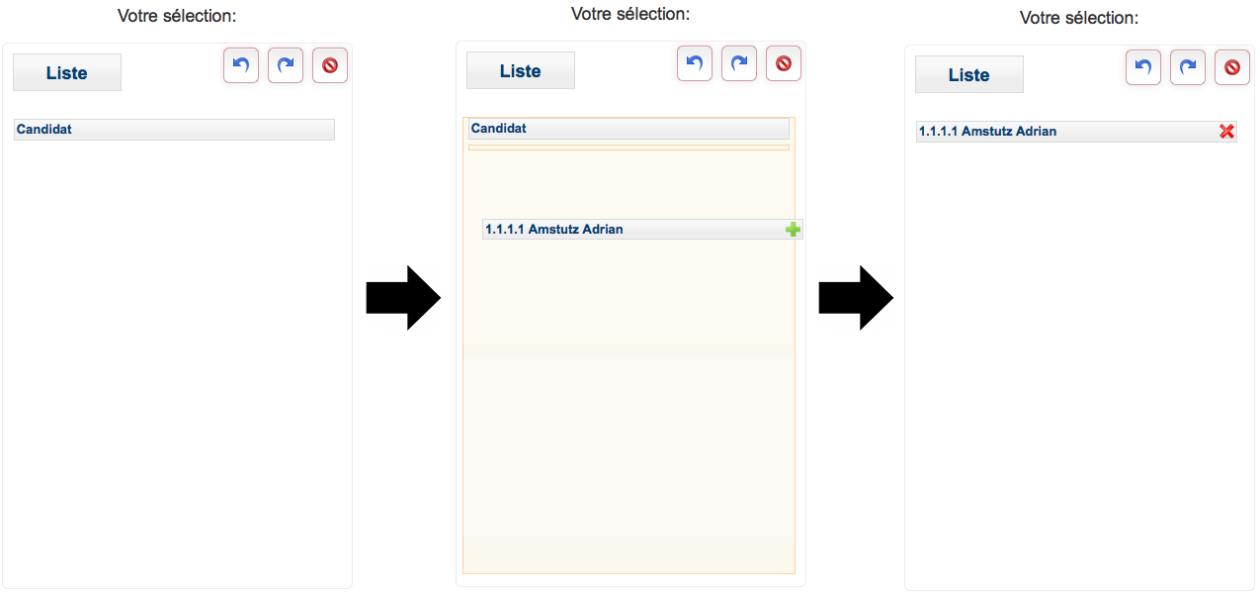



Figure 6.17.: Procédure d'ajout d'un candidat ou d'une liste par « drag & drop »

Lors de l'ajout d'une liste, une question supplémentaire nous est posée. Il y a en effet deux possibilités. L'utilisateur peut vouloir copier la liste entière (y compris les candidats) ou seulement indiquer ce parti et choisir des candidats d'autres liste. La demande lui est alors faite pour savoir s'il désire copier les candidats, ou s'il désire seulement utiliser le titre de la liste (numéro de liste et parti).


Figure 6.18.: Question lors du glissage d'une liste

Dans le chapitre 5.3, il est parlé des paramètres nécessaires à une élection. Ces paramètres sont utiles pour informer l'utilisateur lorsqu'il désire choisir un nombre trop grand de candidats, ou qu'il répète trop de fois le même candidat.


Figure 6.19.: Messages montrant les erreurs que l'utilisateur peut faire pendant la création d'une élection

Pour effacer un candidat de la sélection personnelle, il faut le glisser en dehors de la liste personnalisée. Une image de corbeille apparaît et les listes prédéfinies sont grisées. Pour réaliser cet effet, nous avons utilisé un plugin jQuery [2].

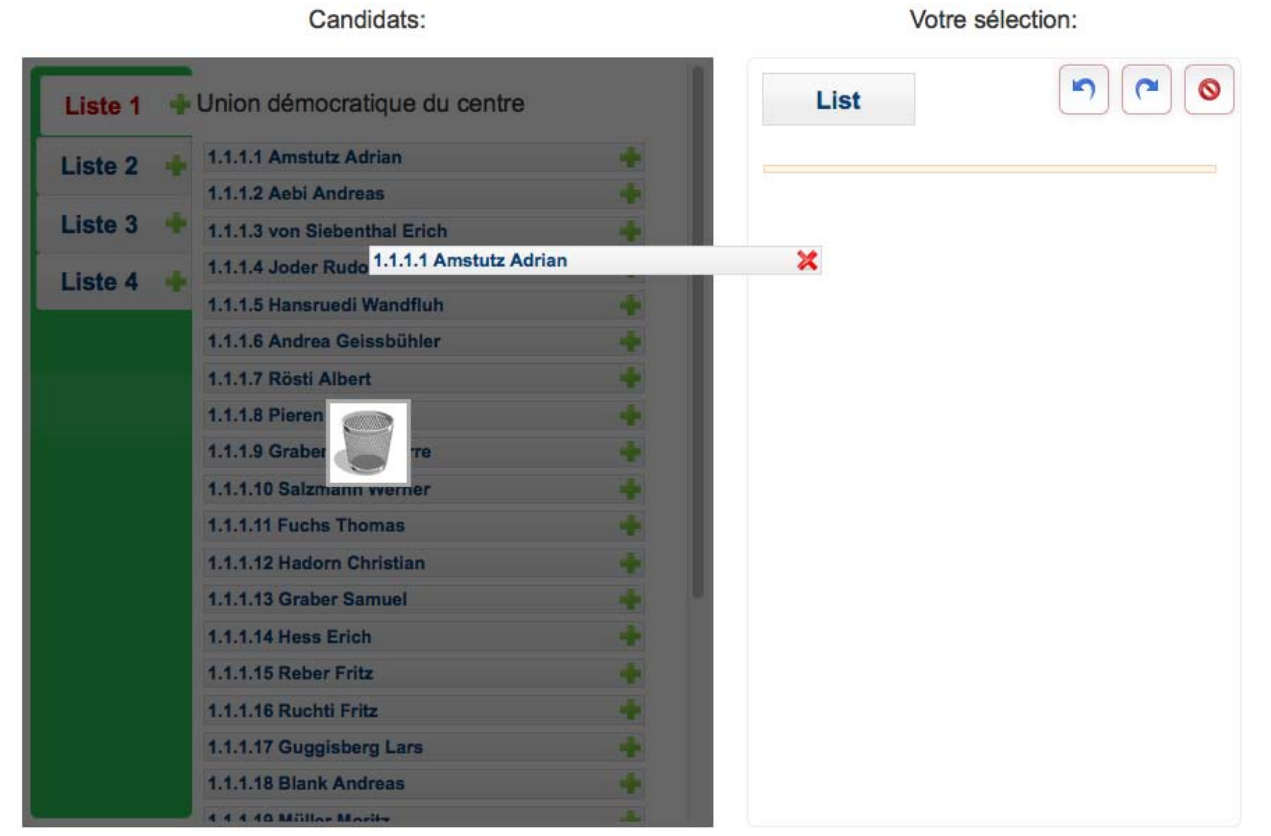



Figure 6.20.: Affichage de la corbeille lors de l'effacement d'un candidat

– **le clic sur l'image « plus vert »** :la deuxième possibilité pour ajouter un candidat ou une liste est de cliquer sur l'image représentant un plus vert se trouvant à droite de chaque liste ou candidat. L'image d'une croix rouge présente à coté de chaque candidat choisi permet d'effacer celui-ci de la sélection personnelle. Dans l'image ci-dessous, on voit apparaître ces images :

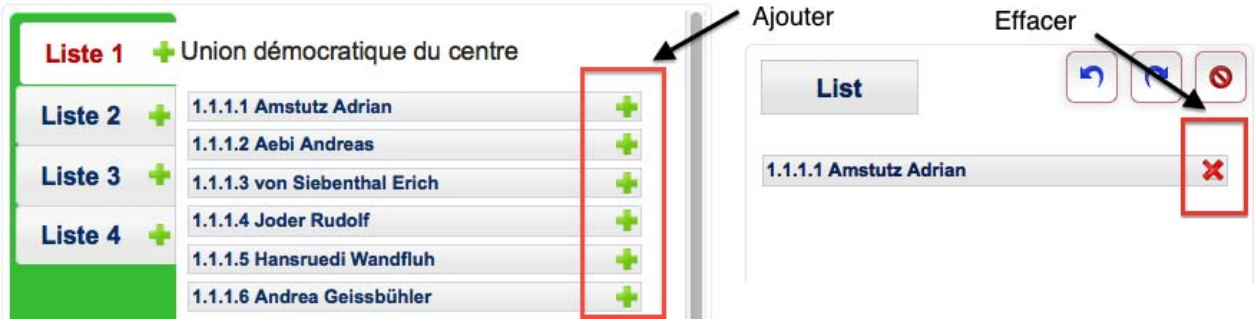



Figure 6.21.: Disposition des boutons verts et rouges

**Le code-barres** À chaque modification de la liste personnalisée, le code-barres va être généré à nouveau. Cela implique le même inconvénient concernant la récolte d'informations que pour une initiative. Mais ici également, le code-barres étant généré à chaque modification, la plateforme ne peut pas savoir lequel a finalement été scanné. A ce stade, l'aide des Web Workers est très utile pour améliorer les performances.

**Les fonctionnalités** Les possibilités d'ajout ou de suppression de candidats ou de listes ont déjà été vues précédemment. Une autre fonctionnalité qui peut être très utile est la possibilité d'annuler la ou les dernières opérations effectuées ou de les rétablir. Il est également possible de réinitialiser le contenu de la sélection personnelle. L'image ci-dessous montre l'emplacement des boutons offrant ces fonctionnalités :


Figure 6.22.: Boutons de modification de l'état de la liste personnalisée

Il est, naturellement, possible de naviguer entre les différentes listes prédéfinies à l'aide des onglets. La couleur apparaissant derrière ces onglets est dépendante du niveau de l'élection (fédéral, cantonal ou communal).

Pour remplacer le numéro de la liste choisie, il suffit de copier un autre titre de liste. Pour l'effacer, il faut réinitialiser la liste.

## 6.7. Bulletin board

Le bulletin board, en français « urne électronique », développé pour ce projet n'est pas prévu pour compter les votes. Il sert d'outil de vérification, c'est à dire qu'il permet de contrôler si le vote généré par l'appareil de vote correspond réellement au vote scanné sur la plateforme.
Pour la création de l'interface, nous avons utilisé un plugin de jQuery appelé « plupload »[4][15], qui a dû être adapté pour s'intégrer au mieux à notre plateforme.

4. Plupload est un plugin de jQuery qui permet de sélectionner et de charger plusieurs fichiers en même temps sur un serveur.

Le déroulement est le suivant : une fois le vote réalisé, on branche la carte de vote à l'ordinateur et on récupère le(s) ficher(s) de résultat. Ensuite, on le(s) charge sur le bulletin board. Seuls des fichiers au format XML peuvent être chargés. Un bouton permet alors d'afficher le contenu de ce(s) ficher(s).
Si le résultat affiché correspond à ce qu'on a voté, tout est alors en ordre. Si on obtient un résultat différent, c'est que la phase de confirmation du vote sur l'appareil de vote n'a pas été réalisée correctement. Si on obtient une erreur d'analyse, alors un mauvais fichier XML a été chargé ou la copie de la carte à l'ordinateur s'est mal passée.

La page du bulletin board se présente de la façon suivante :

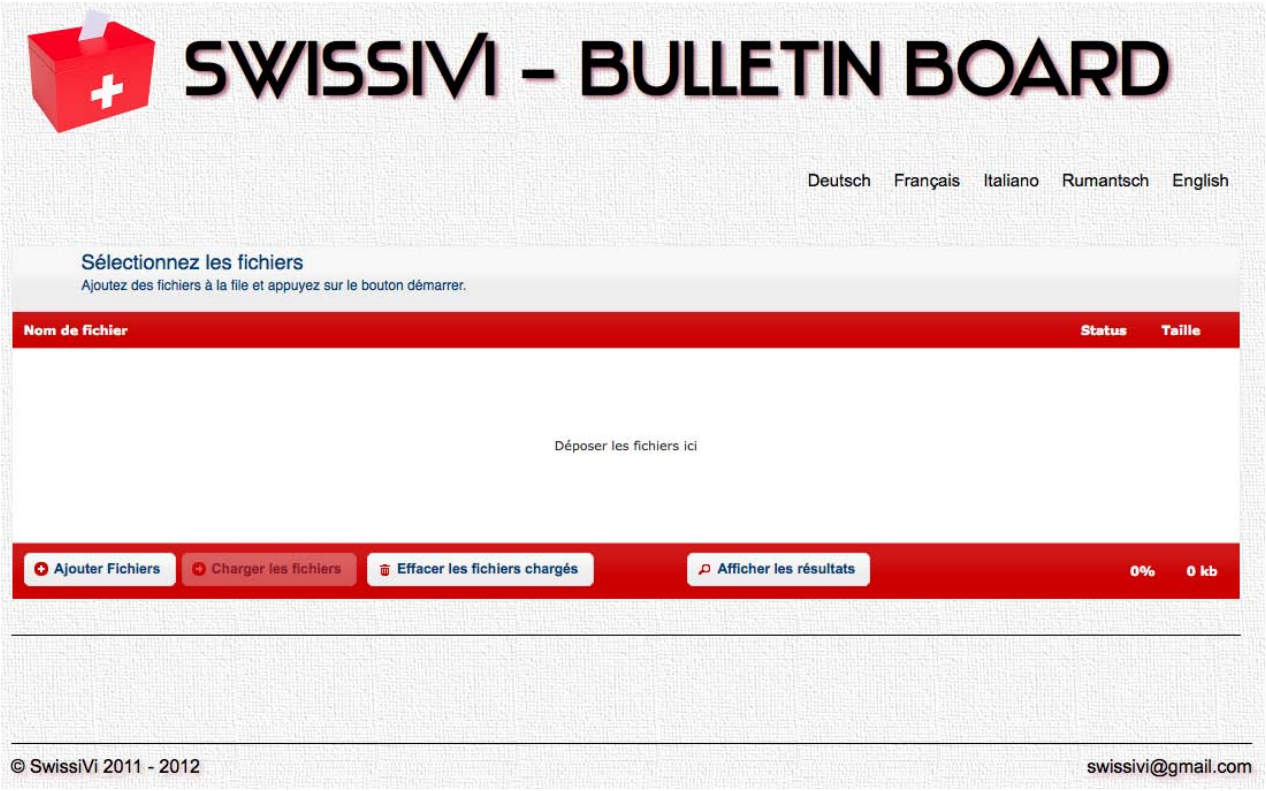



Figure 6.23.: Page du bulletin board

Le changement de la langue se fait comme dans la page d'accueil. Une interface qui nous permet de charger les fichiers de vote et d'afficher les résultats qu'ils contiennent.
L'ajout de fichiers peut être fait en cliquant sur le bouton « Ajouter Fichiers » ou en utilisant du drag & drop. Le chargement sur le serveur des fichiers sélectionnés est fait à l'aide du bouton « Charger les fichiers » qui devient cliquable dès qu'un fichier se trouve dans la liste. Pour afficher les résultats, il suffit ensuite de cliquer sur le bouton « Afficher les résultats ».
Dans l'image ci-dessous, on peut voir le résultat obtenu à la fin de la procédure qui vient d'être décrite :


Figure 6.24.: Résultat du chargement et de la visualisation d'un vote

Si, à ce moment, on ajoute encore un autre ficher, le résultat va se placer au dessous des résultats déjà affichés. Enfin, si on désire effacer les résultats il suffit de cliquer sur le bouton« Effacer les fichiers chargés ».

Le serveur garde les fichiers seulement jusqu'à ce qu'on clique sur « Afficher les résultats », ensuite il les efface. Si plusieurs utilisateurs chargent leurs fichiers simultanément, le serveur est capable de montrer à l'utilisateur seulement le ou les fichiers que celui-ci a chargé et pas les fichiers des autres.

## 6.8. Page d'administration

Cette page permet à l'administrateur de la plateforme d'ajouter ou d'effacer des objets de vote.
Comme toute page d'administration, son accès est protégé par un login. Dans l'image ci-dessous, on peut voir comment se présente la page lors de la première visite (authentification pas encore faite) :

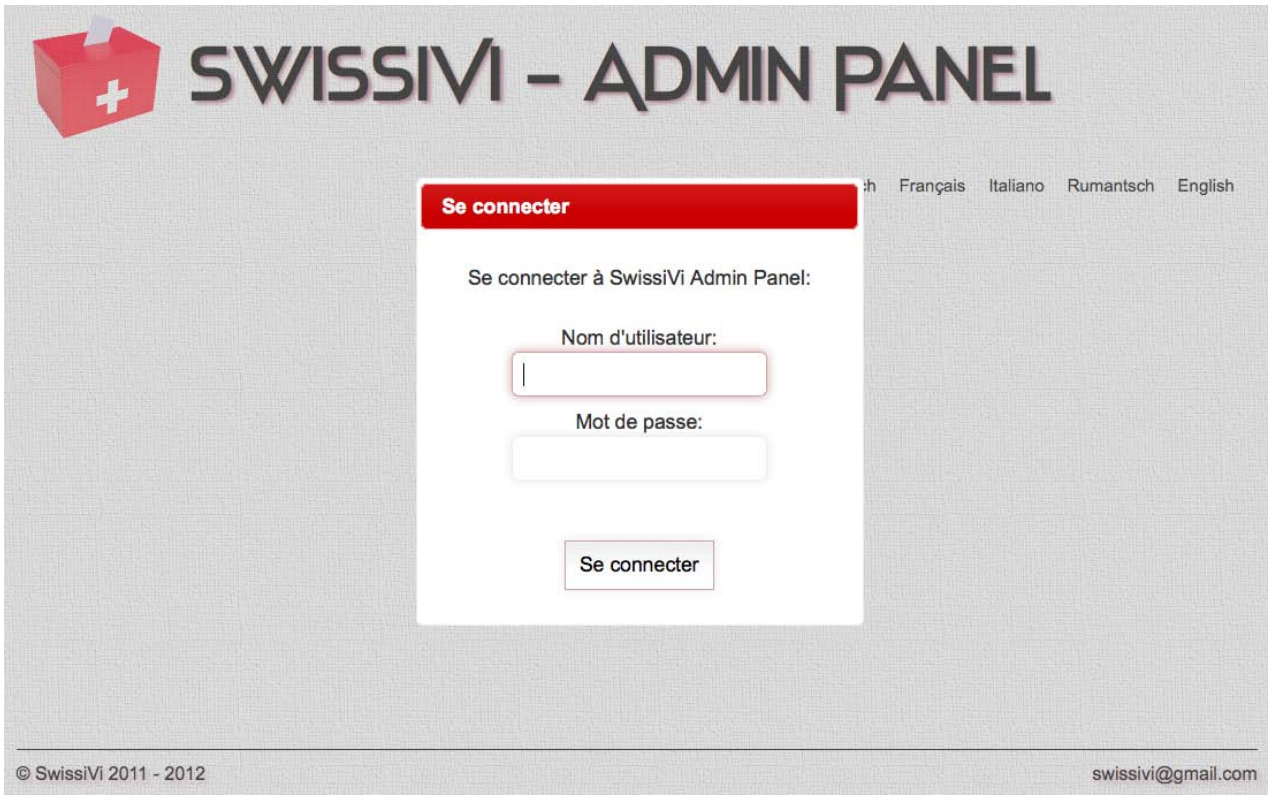



Figure 6.25.: Page d'administration avant l'authentification

Le reste de la page est grisée, car l'authentification est obligatoire.
Si un champ n'est pas rempli, un message informatif apparaît et le champ devient rouge.

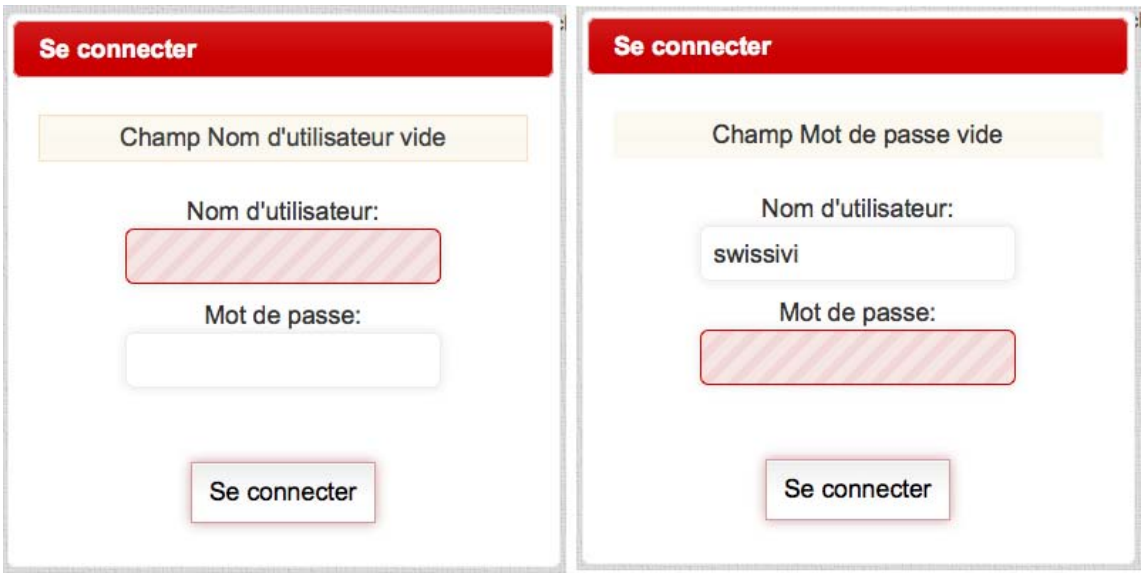



Figure 6.26.: Champs de login vides

L'authentification est faite via Ajax, il n'y a donc pas de rechargement de la page.
Nous avons créé compte administrateur avec le nom d'utilisateur « swissivi » et le mot de passe « swissivi ».

**Les fonctionnalités** Après l'authentification, une page contenant un menu s'affiche. Le menu permet d'ouvrir la page où l'administrateur peut ajouter un événement ou d'appeler la page qui permet de visualiser, modifier ou supprimer les événements et les objets de vote présents sur la plateforme. En plus, un bouton permet de se déconnecter.


Figure 6.27.: Page d'administration après l'authentification

**Ajouter un événement** L'ajout d'événement de vote se fait par le moyen d'un fichier XML. Un standard a été défini, décrivant la structure que doit avoir un tel fichier. Ce fichier doit être chargé sur le serveur. Il est ensuite analysé et les informations trouvées sont stockées dans la base de données. Si une erreur d'analyse survient, celle-ci sera affichée à l'écran.
La page d'ajout d'événements se présente sous la même forme que la page du bulletin board. A la place où étaient affichés les résultats, seuls les messages indiquant le succès ou les erreurs d'importation sont affichés.

**Modifier, supprimer un événement ou un objet** Dans cette page, deux tableaux sont affichés. Le premier permet de voir les événements présents sur la plateforme, de les modifier et de les effacer. Dans le deuxième, on a la liste de tous les objets de vote. Un objet ne peut pas être modifié mais seulement effacé.

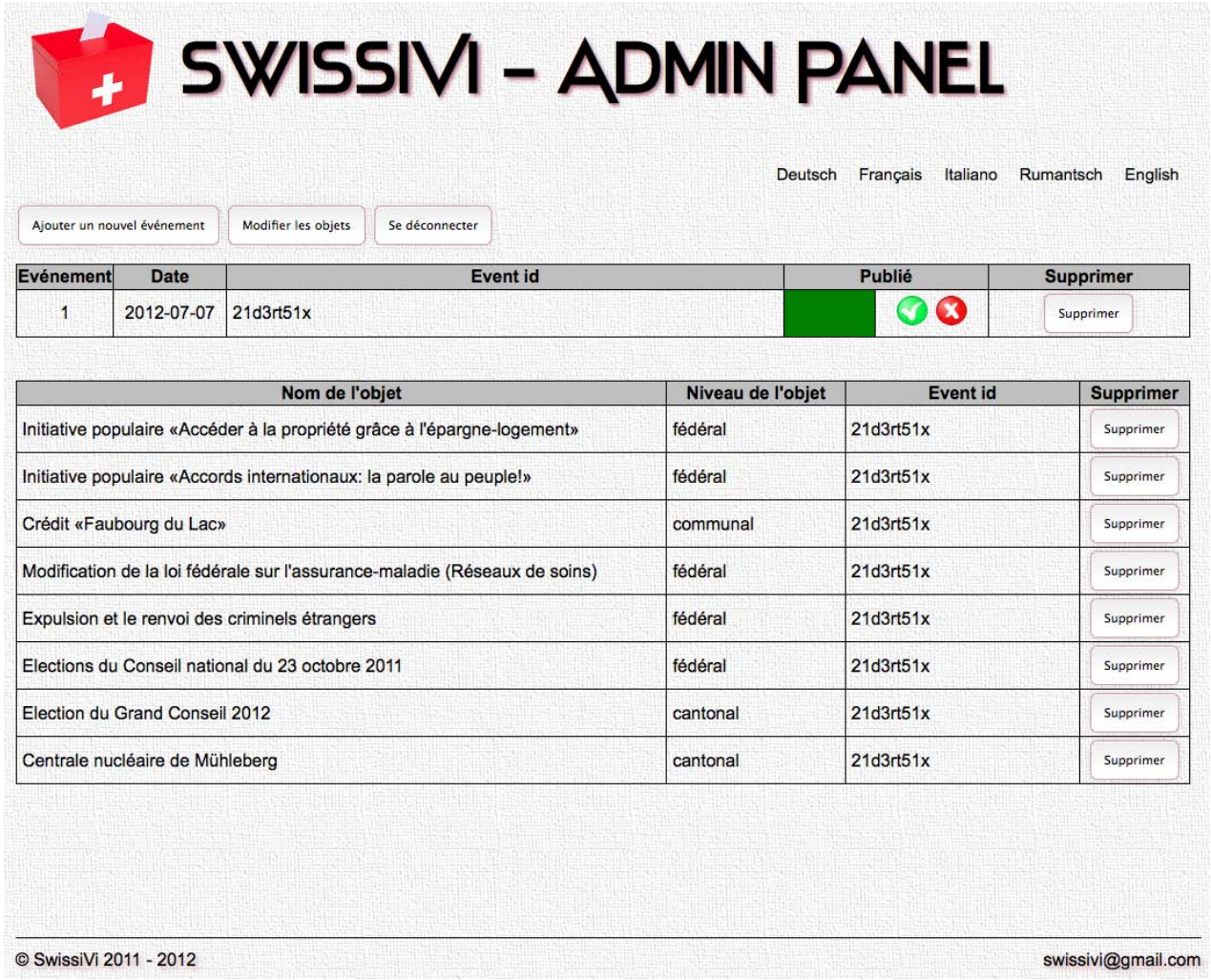



Figure 6.28.: Modification des événements et objets de vote

Lorsqu'on efface un objet, un message de confirmation empêchant toute autre interaction avec la page est affiché à l'utilisateur.


Figure 6.29.: Message de confirmation de la suppression d'un objet de vote

**Déconnexion** En cliquant sur le bouton « Se déconnecter », la session d'administration est terminée et la page de login est affichée.

# 7. Conclusion

Le but de ce travail était de réaliser un preuve de faisabilité du concept développé par RISIS. Le présent document décrit un possibilité d'implémentation et démontre par là que le concept est applicable. Pour ce qui est de l'accueil qui lui sera fait auprès des personnes spécialisée dans ce domaine, cela est plus difficile à estimer. Le point clef de ce concept, à savoir l'appareil de vote sûr, est également son talon d'Achille, car, autant il améliore la sécurité du système, autant il pourrait le handicaper en fonction de ce que les utilisateurs en penseront et de leur disposition à utiliser cet appareil. De notre côté, nous avons essayé de rendre le tout le plus attractif possible et d'en simplifier l'utilisation. Nous pensons et espérons que ce produit pourra être utilisé pour présenter ce concept et convaincre les spécialistes.

## Tâches restantes et potentiel d'amélioration

En ce qui concerne le déroulement de ce travail, la plupart des tâches à effectuer ont pu être réalisées. Une tâche annexe a, en effet, dû être laissée de côté par manque de temps. Il s'agit de l'implémentation de la cryptographie. Le produit a toutefois été conçu pour pouvoir l'y implémenter facilement. La cryptographie n'étant pas un point essentiel pour cet outil de démonstration, son absence ne détruit donc pas le projet, bien que sa présence y apporterait certainement un plus.

Notre bulletin board et notre panneau d'administration sont également des outils de démonstration. Une vraie urne électronique est bien plus complexe que cela. La nôtre a pour simple but d'afficher le contenu des fichiers de vote afin d'en vérifier visuellement le contenu. Cela n'a rien à voir avec une vraie urne électronique.
Le panneau d'administration a également un grand potentiel d'amélioration. Actuellement, il n'est pas très convivial à utiliser. Il sert seulement à ajouter des données pour les démonstrations. Un grand travail peut et doit encore être fourni dans cette partie si on voulait l'utiliser dans un produit soumis au grand public.

D'autres améliorations pourraient certainement encore être apportées au code jQuery principalement et au code Android afin d'en améliorer les performances. Ce sont, en effet, deux langages que nous avons découverts au cours de ce projet, nous ne sommes, par conséquent, pas des spécialistes dans ces domaines. Plusieurs choses pourraient vraisemblablement être améliorées par des connaisseurs.
Des spécialistes des interfaces graphiques pourraient certainement aussi apporter des commentaires utiles afin d'optimiser les interfaces. Nous avons essayé de faire au mieux avec les connaissances dont nous disposons sans être spécialisés dans le domaine.

**Forme du rendu** La plateforme de vote est disponible via internet à l'adresse suivante :
`https://projects.ti.bfh.ch/swissivi/`.
Une machine virtuelle contenant un serveur web avec la plateforme est également disponible à l'adresse :
`https://projects.ti.bfh.ch/swissivi/public/ressources/vm/swissivi.zip` (nom d'utilisateur : « swissivi », mot de passe : « swissivi »).
Les applications de simulation de l'appareil de vote et de la carte de vote peuvent être installées sur de téléphones portables équipés du système Android en version 4.0.3. Ils doivent également avoir une puce NFC. Elles peuvent être obtenues ici :
`https://projects.ti.bfh.ch/swissivi/public/ressources/apps/VotingDevice.apk` et
`https://projects.ti.bfh.ch/swissivi/public/ressources/apps/VotingCard.apk`.
Le code source est disponible, pour les personnes autorisées, dans le répertoire Subversion suivant :
`https://svn.bfh.ch/repos/projects/swissivi/bachelor/final`.

## Connaissances acquises

Notre bilan sur ce travail est positif. Nous avons eu du plaisir à réaliser ce projet intéressant touchant à plusieurs domaines notamment la sécurité, la programmation web et le développement Android. Nous avons aussi apprécié le fait de pouvoir travailler sur un projet très concret.

Ce projet nous a permis d'élargir nos connaissances grâce à la découverte de la programmation pour Android même si l'utilisation que nous en avons faite reste relativement superficielle. Le développement Android offre en effet beaucoup plus de possibilités que celles que nous avons utilisées pour nos simulations. Il n'en est pas moins vrai que la philosophie d'Android nous est maintenant connue, ce qui est déjà un grand bagage que nous sommes heureux d'avoir pu acquérir dans le cadre de ce travail.

Ce travail nous a confronté au framework jQuery. jQuery est un outil très intéressant simplifiant énormément l'implémentation de certaines fonctionnalités. Il est très utile par exemple pour appliquer des effets à une page internet. Cependant, jQuery n'est pas très flexible quant aux possibilités de structurer le code. Une utilisation importante de ce framework engendre une grande quantité de code dans un même fichier et le rend difficilement maintenable. jQuery est donc, à notre avis, plus adapté pour de petites applications.

Nous avons également apprécié le fait de pouvoir nous plonger dans l'univers de la conception d'interfaces graphiques intuitives, et nous avons dû remarquer qu'il s'agit d'un domaine qui n'est pas aussi anodin qu'il y paraît à première vue.

Enfin, la découverte de nouvelles technologies comme le NFC ou le QR-code ainsi que leur utilisation ont également été un enrichissement.

Toutes ces connaissances que nous avons pu acquérir durant ces quatre derniers mois nous seront certainement très utiles dans le futur. Au final, nous sommes satisfaits du résultat obtenu et espérons qu'il pourra être utile.

# Remerciements

Nous aimerions adresser nos remerciements à toutes les personnes qui nous ont aidés d'une quelconque façon dans ce travail. Nous aimerions mentionner :

- **MM. Reto Koenig, Rolf Haenni et Eric Dubuis**, nos superviseurs de projet, pour leurs précieux conseils tout au long de ce travail, ainsi que pour la traduction allemande des textes de la plateforme et de l'appareil de vote
- **M. Andreas Spichiger**, expert, pour ses conseils lors de notre séance
- **Mme Lydia von Bergen** pour la relecture de ce dossier
- **M. Josué von Bergen** pour ses critiques constructives sur la plateforme et l'appareil de vote, ainsi que pour le traitement de certaines images
- **M. Pietro Bottani** pour ses critiques constructives sur la plateforme et l'appareil de vote
- **M. Curdin Maissen** pour la traduction romanche des textes de la plateforme et de l'appareil de vote
- **Les propriétaires du site www.gemeindefahnen.ch** pour la mise à disposition des images des drapeaux des cantons et des communes pour ce travail de Bachelor
- **La commune de Carouge** pour la mise à disposition de l'image utilisée dans le logo

# Bibliographie


[1] A. Afridi, "Jquery scrollable tabs." [Online]. Available : http://jquery.aamirafridi.com/jst/

[2] M. Alsup, "Jquery blockui plugin." [Online]. Available : http://malsup.com/jquery/block/

[3] "Android developers." [Online]. Available : http://developer.android.com

[4] Apache Software Foundation, "Log4j." [Online]. Available : http://logging.apache.org/log4j/1.2/

[5] K. Arase, "Qr code generator for javascript." [Online]. Available : http://www.d-project.com/qrcode/index.html

[6] J. DeRegnaucourt, "Perfect java serialization to and from json format." [Online]. Available : http://code.google.com/p/json-io/

[7] R. Haenni, R. Koenig, and E. Dubuis, "Secure internet voting with a trusted voting device," *EVOTE12 Bregenz*, 2012.

[8] J. Hunter and R. Lear, "Jdom." [Online]. Available : http://www.jdom.org/

[9] M. Izumo, "Zip library javascript." [Online]. Available : http://www.onicos.com/staff/iz/amuse/javascript/expert/deflate.txt

[10] jQuery Foundation and jQuery UI Team, "jquery user interface." [Online]. Available : http://jqueryui.com/

[11] Kevin Waterson, "Model view controller mvc." [Online]. Available : http://www.phpro.org/tutorials/Model-View-Controller-MVC.html

[12] M. Leigeber, "Javascript tooltip." [Online]. Available : http://sixrevisions.com/tutorials/javascript_tutorial/create_lightweight_javascript_tooltip/

[13] Marco Solazzi, "Ottimizzare le performance di jquery : cache e concatenazione." [Online]. Available : http://javascript.html.it/articoli/leggi/3897/ottimizzare-le-performance-di-jquery-cache-e-concatenazione/

[14] Mind|Pipe, "Logging with log4j in androidj." [Online]. Available : http://code.google.com/p/android-logging-log4j/

[15] Moxiecode Systems AB, "Plupload." [Online]. Available : http://www.plupload.com/

[16] A. Pellegrini and P. von Bergen, "Swissivi, spécifications de l'appareil et de la carte de vote, étude de la plateforme de vote," *Travail de semestre, projet 2*, 2011. [Online]. Available : https://projects.ti.bfh.ch/swissivi/public/ressources/pdf/swissiVi_Projet_2.pdf

[17] S.Berfini, "Introduction à l'utilisation du nfc dans une application android." [Online]. Available : http://sberfini.developpez.com/tutoriaux/android/nfc/

[18] "Stackoverflow." [Online]. Available : http://stackoverflow.com/

[19] The jQuery Foundation, "jquery." [Online]. Available : http://jquery.com/

[20] Tobiasz Cudnik, "$.include() ? script inclusion jQuery plugin." [Online]. Available : http://tobiasz123.wordpress.com/2007/08/01/include-script-inclusion-jquery-plugin/

[21] "Tungsten." [Online]. Available : http://sourceforge.net/apps/mediawiki/tungsten/index.php

[22] "Zxing (zebra crossing)." [Online]. Available : http://code.google.com/p/zxing/

# A. Donnée du travail

Bern University of Applied Sciences
Engineering and Information Technology

## Definition of Bachelor Thesis Project

| | |
|---|---|
| for | Andrea Pellegrini<br>Philémon von Bergen |
| Division | Computer Science |
| Advisor | Prof. Dr. Eric Dubuis<br>Prof. Dr. Rolf Haenni<br>Prof. Reto Koenig |

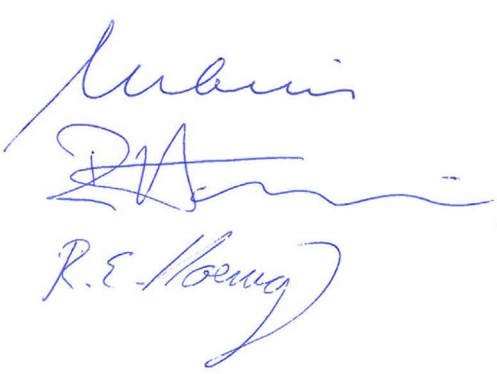
### SwissiVi: Proof-of-Concept for a Novel E-Voting Platform

While several cryptographic protocols exist today which satisfy many requirements postulated for e-voting systems, a secure and easy-to-use interface between a voter and the cryptography on a potentially insecure PC, laptop or smartphone is still missing. The E-Voting Group of RISIS worked out a novel concept of an e-voting system addressing this gap. Core components of this concept are the e-voting platform displaying the voter's choices, a tamper-proof voting device, and a personalized voting card.

The goal of this bachelor thesis is the realization of the novel concept targeting a Swiss e-voting solution. More specifically, the realization comprises the complete user interface for the voter of an upcoming e-voting system for Swiss elections and referendums on federal, cantonal and communal levels. Hardware devices shall be simulated on smartphones. The main focus is set on the fusion of the seemingly contradictory aspects namely 'usability' and 'security' on application level. The goal of this thesis is reached if there is a very high acceptance for all stakeholders involved in e-voting.

| | |
|---|---|
| Start of Project | 20. February 2012 |
| End of Project | 15. June 2012 |

Adivsor: Head of Division:



# B. Cahier des charges

## Introduction

Bien qu'il existe aujourd'hui plusieurs protocoles cryptographiques qui satisfont de nombreuses exigences suggérées pour des systèmes de vote électronique, une interface sûre et facile à utiliser entre le votant et la cryptographie sur un PC ou smartphone potentiellement vulnérable manque toujours à l'appel. Le groupe d'E-Voting de RISIS [1] a mis en place un nouveau concept de vote électronique résolvant ce problème. Les composants essentiels de ce concept sont la plateforme de vote affichant les choix du votant, un appareil de vote sûr, et une carte de vote personnalisée.

## Objectifs

Le but de cette thèse de bachelor est la réalisation de ce concept ciblant une solution pour le vote électronique en Suisse. Plus spécifiquement, la réalisation comprend l'interface utilisateur complète pour le votant pour un système de vote électronique lors de votations et élections suisses au niveau fédéral, cantonal et communal. Les composants hardware devront être simulés sur des smartphones. Le centre d'attention principal est placé sur les aspects à première vue contradictoires que sont la convivialité et la sécurité au niveau application. Le but de cette thèse est atteint si le résultat obtenu rencontre un accueil favorable chez tous les intéressés impliqués dans l'e-voting.

## Description des Stakeholders

- Développeurs : Andrea Pellegrini et Philémon von Bergen
- Superviseurs du projet : Dr. Rolf Haenni, Dr. Eric Dubuis, Reto Eric Koenig
- Expert : Dr. Andreas Spichiger
- E-Voting Group de RISIS qui pourrait utiliser ce projet à des fins de présentation auprès des autorités

## Echéances

Début du projet : 20.02.2012
Fin du projet : 15.06.2012

Rendu du résumé du travail pour le livre des travaux de bachelor : 15.06.2012
Rendu de la version électronique du poster pour l'exposition : 03.07.2012

Défense du travail : 25.06.2012
Présentation du travail lors de la journée finale : 13.07.2012

1. Research Institute for Security in the Information Society

## Tâches à réaliser

Les tâches listées ci-dessous seront réparties entre deux étudiants. A chaque tâche est attribuée une priorité, la priorité 1 signifiant que la tâche est importante et qu'elle doit impérativement être implémentée, la priorité 2 représentant une moins grande importance.

- **Application de simulation de l'appareil de vote** (Priorité 1)
  selon chap. 4.1.2. de la documentation d'étude du projet 2 et selon le document « Secure Internet Voting with a Trusted Voting Device » de R.Haenni, R.Koenig, E.Dubuis.
- **Application de simulation de la carte de vote** (Priorité 1)
  Cette application est responsable pour la signature des fichiers de votes générés par l'application de simulation de l'appareil de vote. La communication entre les deux applications se fait à l'aide d'une technologie wireless, de préférence NFC [2]
- **Plateforme de vote** (Priorité 1)
  L'implémentation de la plateforme de vote se fera selon le chapitre 5.6 de la documentation d'étude du projet 2 et selon le document « Secure Internet Voting with a Trusted Voting Device » de R.Haenni, R.Koenig, E.Dubuis. Cette plateforme sera une application web optimisée pour les ordinateurs personnels.
- **QR code et compression** (Priorité 1)
  La génération du code-barres bidimensionnel doit être réalisé du côté client, de façon à ne pas générer de trafic réseau qui pourrait être audité.
  Pour optimiser la taille du(des) code(s)-barres, le contenu est compressé avant d'être introduit dans le code-barres.
- **Internationalisation** (Priorité 1)
  Les applications pour smartphones ainsi que la plateforme de vote doivent supporter plusieurs langues.
- **Cryptographie** (Priorité 2)
  Le focus pour ce projet est placé sur la convivialité, de ce fait la cryptographie n'est pas le point le plus important. Elle ajoute toutefois de la crédibilité au projet si elle peut être implémentée.
- **Administration de la plateforme** (Priorité 2)
  La partie d'administration n'est pas un point principal du projet. Elle permettra de publier de nouveaux objets de votes ou d'en supprimer.
- **Upload du fichier de vote** (Priorité 2)
  Une simulation simplifiée du bulletin board devra être mise en place afin de permettre l'affichage du résultat du vote. Ceci n'est pas un point important de ce projet.
- **Documentation** (Priorité 1)
  Plusieurs documents devront être rendus. Premièrement, le rapport du travail de Bachelor, deuxièmement, un résumé pour le livre de travail de Bachelor et finalement le poster de présentation du travail.
- **Présentations** (Priorité 1)
  Deux présentations devront être préparées : la défense du travail et une petite présentation pour la journée finale.

## Ressources nécessaires

Afin de pouvoir utiliser et tester la plateforme, nous avons besoin d'un espace sur un serveur web et une base de données mis à disposition par l'école.

Afin de faciliter le partage des fichiers entre nous, nous avons besoin d'un répertoire SVN sur le serveur de l'école.

Afin de simuler l'introduction de la carte de vote dans l'appareil de vote, nous utiliserons deux smartphones, l'un représentant la carte de vote et l'autre l'appareil de vote. La communication entre ces deux smartphones se fera à l'aide de la technologie NFC. Pour cela, nous aurons besoin de deux smartphones supportant le NFC. Ceux-ci seront mis à disposition par l'école.

2. Near field communication : communication en champ proche

## Signature

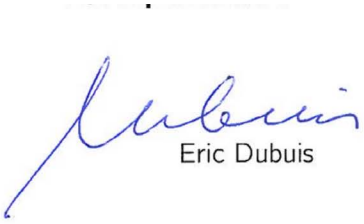
Par l'apposition de sa signature, le signataire confirme être d'accord avec le contenu du présent document.

**Les étudiants :**

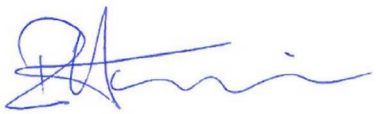
Andrea Pellegrini

Philémon von Bergen

**Les superviseurs :**



Eric Dubuis



Rolf Haenni

Reto Koenig

# C. Use-cases

## Liste des acteurs

| Acteurs | Type | Description |
|---|---|---|
| Votant | Primaire | Personne désirant voter |
| Administrateur | Primaire | Administrateur de la plateforme de vote |
| Appareil de vote | Supporting | Appareil permettant de confirmer son choix |
| Carte de vote | Supporting | Carte de légitimation de vote |

Description des acteurs

## Description des acteurs primaires

Le **votant** affiche la plateforme de vote et prépare son vote. A l'aide de l'appareil de vote dans lequel il a précédemment inséré la carte de vote, il scanne son vote et confirme son choix. Ensuite, il transfère le fichier généré par l'appareil de vote et le charge sur la plateforme de vote.
L'**administrateur** affiche la partie d'administration de la plateforme. Il peut y effacer des objets de vote existants ou en créer de nouveaux en transférant un fichier XML avec un contenu correspondant à une syntaxe définie.

## Use cases

### Plateforme de vote

1. **Choix de la langue**
   Par défaut le système sélectionne automatiquement la langue en prenant la langue du navigateur. L'utilisateur peut changer la langue de l'interface de la plateforme de vote.
2. **Sélection du canton**
   Le votant doit sélectionner son canton, cela peut être fait en cliquant sur la carte de la Suisse à l'endroit désiré ou sur la liste avec le nom de tous les cantons.
3. **Afficher l'aide**
   Pour pouvoir comprendre le fonctionnement de la plateforme de vote, on peut afficher une page d'aide.
4. **Retour à la page d'accueil**
   Ce lien charge la page d'accueil.
5. **Changement du canton**
   Si l'utilisateur s'est trompé de canton, il peut toujours le changer.
6. **Sélection de la commune**
   Pour pouvoir commencer à voter, il faut sélectionner la bonne commune.
7. **Sélection de la langue d'affichage pour les objets de chaque niveau**
   Pour chaque niveau de vote, on peut changer la langue des objets de vote.
8. **Commencer à voter**
   En choisissant cette option, les différents niveaux de vote s'affichent (superposition).
9. **Navigation entre les niveaux**
   On peut changer de niveau de vote en cliquant sur le drapeau de la confédération, du canton ou de la commune.

10. **Navigation entre les objets**
    On peut naviguer entre des objets de même niveau en cliquant sur les onglets visibles.
11. **Changement de la commune**
    Affiche la fenêtre de sélection de commune.
12. **Préparation d'une initiative**
    Le votant doit sélectionner les oui ou les non pour permettre l'affichage du code-barres à scanner.
13. **Sélection d'un candidat**
    Dans la page d'élections, on peut glisser un ou plusieurs candidats dans la zone prédéfinie.
14. **Sélection d'une liste**
    Dans la page d'élections, on peut sélectionner une liste pour déplacer tous les candidats en une seule fois.
15. **Suppression d'un candidat**
    Dans la page d'élections, on peut glisser un candidat hors de la zone prédéfinie pour l'effacer.
16. **Réinitialisation de la liste de choix**
    En cliquant sur le bouton de Reset, on réinitialise la liste des candidats.
17. **Annuler la dernière opération**
    En cliquant sur le bouton de Undo, on annule la dernière modification faite à la liste des élections.
18. **Rétablir l'opération annulée**
    En cliquant sur le bouton de Redo, on rétablit l'opération annulée.
19. **Upload du ficher de résultat**
    Le votant peut uploader les fichers de résultat sur la plateforme pour vérifier la signature et voir le contenu du ficher de vote.

## Administration de la plateforme de vote

20. **Login**
    L'administrateur peut se connecter pour administrer la plateforme.
21. **Logout**
    L'administrateur doit se déconnecter après avoir terminé ses travaux.
22. **Ajout d'un objet de vote**
    L'administrateur peut ajouter un objet de vote.
23. **Suppression d'un objet de vote**
    L'administrateur peut supprimer un objet de vote.
24. **Lister tous les objets de vote**
    L'administrateur peut lister tous les objets de vote disponibles sur la plateforme.

## Applications pour smartphones

25. **Insertion de la carte**
    L'utilisateur insère la carte. Dans notre cas, ce use-case sera simulé par le rapprochement du smartphone simulant la carte de vote du smartphone simulant l'appareil de vote. La communication se fera par NFC.
26. **Choix de la langue**
    L'utilisateur peut choisir la langue de l'interface.
27. **Scan du code-barres**
    L'utilisateur scanne un code-barres.
28. **Faire défiler le choix**
    L'utilisateur fait défiler son choix à l'écran
29. **Confirmer le choix**
    L'utilisateur confirme le choix scanné dans le code-barres.
30. **Annuler le choix**
    L'utilisateur annule le choix scanné dans le code-barres.

31. **Continuer à voter**
    L'utilisateur décide de continuer à voter, c'est à dire de scanner un autre code-barres.
32. **Terminer de voter**
    L'utilisateur décide de terminer de voter.
33. **Insérer code PIN**
    L'utilisateur insère son code PIN.
34. **Effacer code PIN**
    L'utilisateur efface son code PIN, caractère par caractère.
35. **Annulation de l'introduction du code PIN**
    L'utilisateur annule l'introduction du code PIN.
36. **Quitter l'application**
    Si l'utilisateur annule l'introduction du code PIN, l'appareil demande une confirmation avant de s'éteindre.
37. **Valider code PIN**
    L'utilisateur valide son code PIN.
38. **Retirer la carte**
    L'utilisateur retire la carte. Dans notre cas, ce use case sera simulé par l'éloignement du smartphone simulant la carte de vote du smartphone simulant l'appareil de vote.
39. **Modifier le code PIN**
    L'utilisateur entre son nouveau code PIN et le confirme une deuxième fois.

# Fully-dressed Format

## Plateforme de vote

## Use case 1

**Use Case UC1 :** Choix de la langue
**Acteur primaire :** Votant
**Preconditions :**
– L'utilisateur doit être sur la page d'accueil
**Success Guarantee :** L'utilisateur a choisi la langue desirée.

**Main Success Scenario :**

1. L'utilisateur clique sur le bouton de la langue desirée
2. La page est rechargée avec la nouvelle langue

## Use case 2

**Use Case UC2 :** Sélection du canton
**Acteur primaire :** Votant
**Preconditions :**
– L'utilisateur doit être sur la page d'accueil
**Success Guarantee :** L'utilisateur peut ensuite continuer la navigation dans le site

**Main Success Scenario :**

1. L'utilisateur choisit le canton en cliquant sur l'image de la Suisse à l'endroit désiré ou dans la liste avec le nom des cantons
2. La page de sélection de la commune est affichée

## Use case 3

**Use Case UC3 :** Afficher l'aide
**Acteur primaire :** Votant
**Success Guarantee :** L'utilisateur peut voir la page d'aide

**Main Success Scenario :**

1. L'utilisateur clique sur le lien « Afficher l'aide »
2. La page d"aide est affichée

## Use case 4

**Use Case UC4 :** Retour à la page d'accueil
**Acteur primaire :** Votant
**Preconditions :**
– L'utilisateur doit être sur une des pages de vote
**Success Guarantee :** L'utilisateur retourne à la page d'accueil

**Main Success Scenario :**

1. L'utilisateur clique sur le lien
2. L'utilisateur est redirigé à la page d'accueil

## Use case 5

**Use Case UC5 :** Changement du canton
**Acteur primaire :** Votant
**Preconditions :**
– L'utilisateur doit d'abord avoir sélectionné un canton et puis être sur la page de l'« overview » des votes possibles
**Success Guarantee :** Le canton a été changé

**Main Success Scenario :**

1. L'utilisateur clique sur le lien « changer la commune »
2. Il change le canton dans la fenêtre affichée

## Use case 6

**Use Case UC6 :** Sélection de la commune
**Acteur primaire :** Votant
**Preconditions :**
– L'utilisateur doit d'abord avoir sélectionné le canton
**Success Guarantee :** Les objets de vote pour la commune désirée sont affichés

**Main Success Scenario :**

1. L'utilisateur écrit le nom de sa commune, et après chaque lettre, les noms entiers lui sont proposés automatiquement
2. Les objets de vote pour la commune désirée sont affichés

## Use case 7

**Use Case UC7 :** Sélection de la langue d'affichage pour les objets de chaque niveau
**Acteur primaire :** Votant
**Preconditions :**
– L'utilisateur doit être sur la page de « overview » et avoir sélectionné le canton et la commune
**Success Guarantee :** L'utilisateur peut changer la langue pour les objets de chaque niveau

**Main Success Scenario :**

1. L'utilisateur sélectionne la langue pour chaque niveau de vote
2. Les objets de vote sont affichés dans la langue choisie

## Use case 8

**Use Case UC8 :** Commencer à voter
**Acteur primaire :** Votant
**Preconditions :**
– L'utilisateur doit avoir sélectionné son canton et sa commune
**Success Guarantee :** L'utilisateur peut commencer à voter

**Main Success Scenario :**

1. L'utilisateur clique sur le bouton « commencer à voter »
2. Les pages avec les objets de vote sont affichées

## Use case 9

**Use Case UC9 :** Navigation entre les niveaux
**Acteur primaire :** Votant
**Preconditions :**
– UC8 réalisé
**Success Guarantee :** L'utilisateur peut changer le niveau de vote

**Main Success Scenario :**

1. L'utilisateur clique soit sur le drapeau de la commune, du canton ou de la Confédération
2. La page avec le premier objet de vote du niveau de vote demandé sera affichée

**Extensions :** Si l'utilisateur quitte une page d'élections, une confirmation est demandée.

## Use case 10

**Use Case UC10 :** Navigation entre les objets
**Acteur primaire :** Votant
**Preconditions :**
– UC8 réalisé
**Success Guarantee :** L'utilisateur peut naviguer entre les objets de vote du niveau auquel il se trouve

**Main Success Scenario :**

1. L'utilisateur clique sur l'onglet avec le titre de l'objet de vote désiré
2. L'objet de vote désiré est affiché

**Extensions :** Si l'utilisateur quitte une page d'élections, une confirmation est demandée.

## Use case 11

**Use Case UC11 :** Changement de la commune
**Acteur primaire :** Votant
**Preconditions :**
– L'utilisateur doit avoir cliqué sur le lien « changer la commune »
**Success Guarantee :** L'utilisateur peut recommencer à voter avec les objets de vote d'une autre commune

**Main Success Scenario :**

1. L'utilisateur clique sur le lien « changer la commune »
2. La page de « overview » est affichée

## Use case 12

**Use Case UC12 :** Préparation d'une initiative
**Acteur primaire :** Votant
**Preconditions :**
– UC8 réalisé
– être sur la bonne page
**Success Guarantee :** L'utilisateur a préparé son propre vote pour l'initiative pour laquelle il veut voter. Il peut scanner son propre vote

**Main Success Scenario :**

1. L'utilisateur sélectionne la réponse oui ou non aux questions posées
2. Le code-barres est généré

## Use case 13

**Use Case UC13 :** Sélection d'un candidat
**Acteur primaire :** Votant
**Preconditions :**
– UC8 réalisé
– être sur une page d'élections
**Success Guarantee :** L'utilisateur a déplacé le nom d'un ou plusieurs candidats dans la zone de vote

**Main Success Scenario :**

1. L'utilisateur prend le ou les candidats qu'il veut et les glisse dans la zone de vote.
2. Le ou les noms des candidats sont déplacés dans la zone de vote
3. Un code-barres est généré

## Use case 14

**Use Case UC14 :** Sélection d'une liste
**Acteur primaire :** Votant
**Preconditions :**
– UC8 réalisé
– être sur une page d'élections
**Success Guarantee :** L'utilisateur a déplacé une liste entière dans la zone de vote. Tous les noms des candidats de la liste sont écrits dans la zone de vote

**Main Success Scenario :**

1. L'utilisateur prend une liste et la glisse dans la zone de vote
2. La page de sélection de la commune est affichée

## Use case 15

**Use Case UC15 :** Suppression d'un candidat
**Acteur primaire :** Votant
**Preconditions :**
– UC8 réalisé
– être sur une page d'élections
– avoir au moins un candidat dans la zone de vote
**Success Guarantee :** L'utilisateur a effacé un ou plusieurs candidats de la zone de vote

**Main Success Scenario :**

1. L'utilisateur prend un ou plusieurs candidats et les glisse hors de la zone de vote
2. Le candidat n'est plus dans la zone de vote

## Use case 16

**Use Case UC16 :** Réinitialisation de la liste de choix
**Acteur primaire :** Votant
**Preconditions :**
– UC8 réalisé
– être sur une page d'élections
– avoir fait au moins une modification dans la page des élections
**Success Guarantee :** L'utilisateur réinitialise la page d'élections

**Main Success Scenario :**

1. L'utilisateur clique sur le bouton « reset »
2. La page d'élections est réinitialisée

## Use case 17

**Use Case UC17 :** Annuler la dernière opération
**Acteur primaire :** Votant
**Preconditions :**
– UC8 réalisé
– être sur une page d'élections
– avoir fait au moins une modification dans la page des élections
**Success Guarantee :** La dernière opération que l'utilisateur a fait est annulée. L'état de la page est restauré à l'état précédent

**Main Success Scenario :**

1. L'utilisateur clique sur le bouton « undo »
2. L'état de la page est restauré à l'état précédent

## Use case 18

**Use Case UC18 :** Rétablir l'opération annulée
**Acteur primaire :** Votant
**Preconditions :**
– UC8 réalisé
– être sur une page d'élections
– avoir au moins une fois réalisé le UC17
**Success Guarantee :** L'état de la page d'élections est restauré à l'état précédent l'opération qu'on vient d'annuler

**Main Success Scenario :**

1. L'utilisateur clique sur le bouton « redo »
2. La dernière opération annulée est rétablie

## Use case 19

**Use Case UC19 :** Upload du ficher de résultat
**Acteur primaire :** Votant
**Preconditions :**
– Avoir un ficher de vote correct
– être sur la bonne page
**Success Guarantee :** Le votant peut vérifier que le ficher que l'appareil de vote a crée est correct. Il peut vérifier la signature et regarder ce qu'il a voté.

**Main Success Scenario :**

1. L'utilisateur upload le fichier
2. Le ficher est uploadé sur le site et le votant peut vérifier la signature et regarder ce qu'il a voté

## Administration de la plateforme de vote

## Use case 20

**Use Case UC20 :** Login
**Acteur primaire :** Administrateur
**Preconditions :**
– Etre sur la page de login
**Success Guarantee :** L'administrateur est connecté.

**Main Success Scenario :**

1. L'administrateur indique son nom d'utilisateur et son mot de passe.
2. L'application contrôle et donne l'accès

## Use case 21

**Use Case UC21 :** Logout
**Acteur primaire :** Administrateur
**Preconditions :**
– UC20 réalisé
**Success Guarantee :** L'administrateur est déconnecté.

**Main Success Scenario :**

1. L'administrateur se déconnecte

## Use case 22

**Use Case UC22 :** Ajout d'un objet de vote
**Acteur primaire :** Administrateur
**Preconditions :**
– UC20 réalisé
**Success Guarantee :** L'objet est affiché sur la plateforme.

**Main Success Scenario :**

1. L'administrateur upload le fichier XML contenant les données de l'objet
2. L'objet est copié et publié

## Use case 23

**Use Case UC23 :** Suppression d'un objet de vote
**Acteur primaire :** Administrateur
**Preconditions :**
– UC20 réalisé
– au moins un objet doit avoir été créé
**Success Guarantee :** L'objet de vote sélectionné est supprimé.

**Main Success Scenario :**

1. L'administrateur sélectionne un objet de vote
2. Il le supprime
3. L'objet n'est plus visible

## Use case 24

**Use Case UC24 :** Lister les objets de vote
**Acteur primaire :** Administrateur
**Preconditions :**
– UC20 réalisé
**Success Guarantee :** Une liste avec tous les objets disponibles est affichée.

**Main Success Scenario :**

1. Une liste avec tous les objets disponibles est affichée

# Applications pour smartphones

## Use case 25

**Use Case UC25 :** Insertion de la carte
**Acteur primaire :** Votant
**Preconditions :**
– Démarrer l'application
**Success Guarantee :** Le votant peut utiliser l'appareil de vote.

**Main Success Scenario :**

1. L'utilisateur approche le smartphone simulant la carte de vote
2. L'application de simulation de l'appareil de vote passe dans l'état suivant (choix de la langue)

## Use case 26

**Use Case UC26 :** Choix de la langue
**Acteur primaire :** Votant
**Preconditions :**
– UC25 réalisé
**Success Guarantee :** La langue de l'appareil de vote a été choisie.

**Main Success Scenario :**

1. Le votant choisit sa langue
2. L'interface de l'appareil apparaît dans la langue choisie
3. L'application de simulation de l'appareil de vote passe dans l'état suivant (scan du code-barres)

## Use case 27

**Use Case UC27 :** Scan du code-barres
**Acteur primaire :** Votant
**Preconditions :**
– UC26 réalisé
**Success Guarantee :** Le code-barres a été lu correctement.

**Main Success Scenario :**

1. Le votant lit le code-barres à l'aide de l'appareil
2. Le code est lu correctement
3. L'application de simulation de l'appareil de vote passe dans l'état suivant (défilement du choix)

**Extensions :** Si le code-barres est en plusieurs parties, chaque partie devra être lue.

## Use case 28

**Use Case UC28 :** Faire défiler le choix
**Acteur primaire :** Votant
**Preconditions :**
– UC27 réalisé
**Success Guarantee :** Le votant a fait défiler tout le choix

**Main Success Scenario :**

1. Le votant fait défiler son choix

## Use case 29

**Use Case UC29 :** Confirmer le choix
**Acteur primaire :** Votant
**Preconditions :**
– UC28 réalisé
**Success Guarantee :** Le votant a confirmé son choix.

**Main Success Scenario :**

1. Le votant confirme son choix
2. L'application de simulation de l'appareil de vote passe dans l'état suivant (terminer ou continuer à voter)

## Use case 30

**Use Case UC30 :** Annuler le choix
**Acteur primaire :** Votant
**Preconditions :**
– UC27 réalisé
**Success Guarantee :** Le votant a annulé son choix.

**Main Success Scenario :**

1. Le votant annule son choix
2. L'application de simulation de l'appareil de vote passe dans l'état suivant (scan du code-barres)

## Use case 31

**Use Case UC31 :** Continuer à voter
**Acteur primaire :** Votant
**Preconditions :**
– UC29 réalisé
**Success Guarantee :** Le votant peut voter pour un autre objet.

**Main Success Scenario :**

1. Le votant décide de continuer à voter
2. L'application de simulation de l'appareil de vote passe dans l'état suivant (scan du code-barres)

## Use case 32

**Use Case UC32 :** Terminer de voter
**Acteur primaire :** Votant
**Preconditions :**
– UC29 réalisé
**Success Guarantee :** L'utilisateur a choisi de terminer de voter.

**Main Success Scenario :**

1. Le votant décide de terminer de voter
2. L'application de simulation de l'appareil de vote passe dans l'état suivant (introduction du code PIN)

## Use case 33

**Use Case UC33 :** Insérer code PIN
**Acteur primaire :** Votant
**Preconditions :**
– UC32 réalisé
**Success Guarantee :** L'utilisateur a inséré son code PIN.

**Main Success Scenario :**

1. Le votant insère son code PIN
2. L'application de simulation de l'appareil de vote passe dans l'état suivant (valider code PIN)

## Use case 34

**Use Case UC34 :** Effacer code PIN
**Acteur primaire :** Votant
**Preconditions :**
– UC32 réalisé
– Le votant a introduit au moins un caractère
**Success Guarantee :** L'utilisateur a effacé un ou plusieurs caractères.

**Main Success Scenario :**

1. Le votant efface un ou plusieurs caractères entrés

## Use case 35

**Use Case UC35 :** Annulation l'introduction du code PIN
**Acteur primaire :** Votant
**Preconditions :**
– UC32 réalisé
**Success Guarantee :** Le votant a annulé l'introduction du code PIN.

**Main Success Scenario :**

1. Le votant efface tous les caractères entrés et appuie encore sur « effacer »
2. Une demande de confirmation s'affiche

**Extensions :** La confirmation de cette manipulation quitte l'application.

## Use case 36

**Use Case UC36 :** Quitter l'application
**Acteur primaire :** Votant
**Preconditions :**
– UC35 réalisé
**Success Guarantee :** Le votant a quitté l'application.

**Main Success Scenario :**

1. Le votant confirme vouloir annuler l'introduction du PIN
2. L'application quitte

## Use case 37

**Use Case UC37 :** Valider code PIN
**Acteur primaire :** Votant
**Preconditions :**
– UC33 réalisé
**Success Guarantee :** Le votant a validé le code PIN et celui-ci est correct.

**Main Success Scenario :**

1. Le votant valide le code PIN entré
2. L'application vérifie le code PIN
3. L'application de simulation de l'appareil de vote passe dans l'état suivant (retirer la carte)

## Use case 38

**Use Case UC38 :** Retirer la carte
**Acteur primaire :** Votant
**Preconditions :**
– UC37 réalisé
**Success Guarantee :** Le votant a retiré la carte et l'application a quitté.

**Main Success Scenario :**

1. Le votant retire la carte
2. L'application quitte

## Use case 39

**Use Case UC39 :** Modifier le code PIN
**Acteur primaire :** Votant
**Preconditions :**
– UC33 réalisé
**Success Guarantee :** Le code PIN a été changé.

**Main Success Scenario :**

1. Le votant entre le nouveau PIN
2. Il confirme son nouveau PIN

# D. Protocole de tests

La plateforme ainsi que de l'appareil et la carte de vote ne se prêtent pas bien à un testing automatisé. Les effets jQuery par exemple ne peuvent être testés que manuellement et le fonctionnement de l'appareil et de la carte de vote se prêtent mieux à un testing manuel.
Pour cette raison, nous avons réalisé un protocole de tests permettant de vérifier le bon fonctionnement des différents acteurs.

## Plateforme de vote

### UC 1 : Choix de la langue

| Description | Succès |
|---|---|
| Sur la page d'accueil, l'utilisateur peut choisir sa langue | Oui |
| La langue sélectionnée reste la même pour toute la durée de la session de vote ou jusqu'à modification de la part de l'utilisateur | Oui |

### UC 2 : Sélection du canton

| Description | Succès |
|---|---|
| L'utilisateur peut choisir son canton de résidence sur la carte topographique de la page d'accueil | Oui |
| L'utilisateur peut choisir son canton de résidence dans la liste sur la page d'accueil | Oui |

### UC 3 : Afficher l'aide

| Description | Succès |
|---|---|
| Sur chaque page, une page d'aide peut être affichée | Oui |

### UC 4 : Retour à la page d'accueil

| Description | Succès |
|---|---|
| A partir de la page de vote, l'utilisateur peut retourner à la page d'accueil | Oui |

### UC 5 : Changement du canton

| Description | Succès |
|---|---|
| L'utilisateur peut changer son canton de résidence dans la page de sélection de commune | Oui |
| Le changement du canton réinitialise la vue d'ensemble des objets et la commune sélectionnée | Oui |

## UC 6 : Sélection de la commune

| Description | Succès |
|---|---|
| L'utilisateur peut choisir sa commune en commençant à en taper le nom | Oui |
| Un nom de commune mal orthographié n'est pas accepté | Oui |
| Seules les communes du canton sélectionné sont proposées et acceptées | Oui |

## UC 7 : Sélection de la langue d'affichage pour les objets de chaque niveau

| Description | Succès |
|---|---|
| Pour chaque niveau, l'utilisateur peut choisir dans quelle langue les objets doivent être affichés | Oui |
| Seules les langues dans lesquels les objets sont disponibles sont proposées | Oui |

## UC 8 : Commencer à voter

| Description | Succès |
|---|---|
| En cliquant sur le bouton correspondant, la fenêtre de sélection de commune se ferme et la page de vote s'affiche | Oui |
| La construction de l'interface se fait niveau par niveau | Oui |

## UC 9 : Navigation entre les niveaux

| Description | Succès |
|---|---|
| La navigation entre les niveaux à l'aide des onglets verticaux est fonctionnelle | Oui |
| Si un niveau ne contient pas d'objet, l'onglet correspondant n'est pas sélectionnable | Oui |

## UC 10 : Navigation entre les objets

| Description | Succès |
|---|---|
| La navigation entre les objets à l'aide des onglets horizontaux est fonctionnelle | Oui |
| Le titre complet s'affiche dans des infobulles | Oui |
| Si les onglets nécessitent plus de place que la largeur de l'écran, on peut faire défiler les onglets | Oui |

## UC 11 : Changement de la commune

| Description | Succès |
|---|---|
| La commune d'origine ou les langues d'affichage des objets peuvent être modifiées | Oui |
| Lors du changement de la commune, la vue d'ensemble est effacée et recréée pour la nouvelle commune | Oui |
| Les contenus devant être rechargés sont remplacés correctement | Oui |

## UC 12 : Préparation d'une initiative

| Description | Succès |
|---|---|
| Les résultats d'une initiative peuvent être choisis à l'aide de boutons radio | Oui |

## UC 13 : Sélection d'un candidat

| Description | Succès |
|---|---|
| Un candidat peut être ajouté à la sélection par drag and drop | Oui |
| Un candidat peut être ajouté à la sélection à l'aide d'un bouton | Oui |

## UC 14 : Sélection d'une liste

| Description | Succès |
|---|---|
| Une liste entière peut être copiée par drag and drop | Oui |
| Une liste entière peut être copiée à l'aide d'un bouton | Oui |
| Le titre d'une liste peut être copié par drag and drop | Oui |
| Le titre d'une liste peut être copié à l'aide d'un bouton | Oui |

## UC 15 : Suppression d'un candidat

| Description | Succès |
|---|---|
| Un candidat peut être supprimé de la sélection par drag and drop | Oui |
| Un candidat peut être supprimé de la sélection à l'aide du bouton | Oui |

## UC 16 : Réinitialisation de la liste de choix

| Description | Succès |
|---|---|
| La sélection de candidats peut être effacée | Oui |

## UC 17 : Annuler la dernière opération

| Description | Succès |
|---|---|
| Les dernières actions peuvent être annulées | Oui |

## UC 18 : Rétablir l'opération annulée

| Description | Succès |
|---|---|
| Les actions annulées peuvent être rétablies | Oui |

## UC 19 : Upload du ficher de résultat

| Description | Succès |
|---|---|
| Le fichier de vote récupéré sur la carte de vote peut être chargé sur la plateforme | Oui |
| Le résultat contenu peut être affiché | Oui |
| Un message indique si une erreur est survenue | Oui |
| Les fichiers chargés sont effacés du serveur dès qu'ils ne sont plus utilisés | Oui |

### Autres tests

| Description | Succès |
|---|---|
| Les différentes animations jQuery fonctionnent | Oui |
| Dans l'interface des élections, le nombre maximum de candidats est respecté | Oui |
| Dans l'interface des élections, le nombre maximum de répétitions d'un candidat est respecté | Oui |
| Les codes-barres contiennent des contenus correctement formatés | Oui |
| Les codes-barres en plusieurs parties sont générés correctement | Oui |
| Les codes-barres sont générés à chaque modification (élections et initiative) | Oui |

## Administration de la plateforme de vote

### UC 20 : Login

| Description | Succès |
|---|---|
| L'administrateur peut se connecter à la partie d'administration | Oui |
| Impossible d'afficher la partie d'administration sans se connecter | Oui |

### UC 21 : Logout

| Description | Succès |
|---|---|
| L'administrateur peut se déconnecter | Oui |

### UC 22 : Ajout d'un objet de vote

| Description | Succès |
|---|---|
| L'administrateur peut ajouter un événement de vote contenant un ou plusieurs objets de vote | Oui |
| Un message d'erreur lui indique si l'importation échoue | Oui |

### UC 23 : Suppression d'un objet de vote

| Description | Succès |
|---|---|
| L'administrateur peut supprimer un événement ou un objet de vote | Oui |
| La suppression d'un événement de vote entraîne la suppression de tous les objets de vote qui lui sont associés | Oui |

### UC 24 : Lister tous les objets de vote

| Description | Succès |
|---|---|
| L'administrateur peut afficher tous les événements ou objets de vote présents dans la base de données | Oui |
| L'administrateur peut modifier l'état de publication d'un événement de vote | Oui |

## Bulletin board

| Description | Succès |
|---|---|
| L'utilisateur peut charger un ou plusieurs fichiers de résultat | Oui |
| Seuls les fichiers au format XML sont acceptés | Oui |
| Le résultat contenu dans le fichier est affiché | Oui |
| Les erreurs d'analyse sont affichées à l'utilisateur | Oui |
| L'utilisateur peut supprimer les fichiers chargés, ou ils sont supprimés automatiquement | Oui |

## Appareil et carte de vote

### UC 25 : Insertion de la carte

| Description | Succès |
|---|---|
| Lors de l'insertion de la carte, la langue est transmise de la carte à l'appareil | Oui |
| Lors de l'insertion de la carte, le PIN est transmis de la carte à l'appareil | Oui |
| Lors de l'insertion de la carte, les noms des fichiers présents sur la carte sont transmis de la carte à l'appareil de vote | Oui |
| L'insertion d'une carte bloquée affiche un message | Oui |

### UC 26 : Choix de la langue

| Description | Succès |
|---|---|
| Les textes de la carte de vote s'affichent dans la langue choisie | Oui |
| La langue de l'appareil de vote correspond à la langue choisie sur la carte de vote | Oui |
| Tous les textes sont affichés dans la langue choisie | Oui |

### UC 27 : Scan du code-barres

| Description | Succès |
|---|---|
| L'application de scan de code-barres se lance | Oui |
| Le scan d'un code-barres différent de ceux de la plateforme affiche un message d'erreur | Oui |
| Le scan d'un code-barres corrompu affiche un message | Oui |
| Le scan d'un code-barres en plusieurs parties affiche un message indiquant les parties manquantes | Oui |
| La décompression du contenu du code-barres est exécutée correctement | Oui |
| La reconstitution du contenu de codes-barres en plusieurs parties est exécutée correctement | Oui |
| La vérification de la signature est exécutée dès que la questions a été scannée dans sa totalité | Oui |
| Si une erreur survient dans l'une des trois actions mentionnées ci-dessus, un message d'erreur correspondant est affiché | Oui |

### UC 28 : Faire défiler le choix

| Description | Succès |
|---|---|
| Le contenu du code-barres scanné est affiché à l'écran | Oui |
| Les touches Haut et Bas font défiler le texte | Oui |
| Impossible de confirmer le choix avant d'avoir fait défiler tout le texte | Oui |

## UC 29 : Confirmer le choix

| Description | Succès |
|---|---|
| La touche OK confirme le choix | Oui |
| Impossible de confirmer le choix avant d'avoir fait défiler tout le texte | Oui |

## UC 30 : Annuler le choix

| Description | Succès |
|---|---|
| La touche C annule le choix | Oui |

## UC 31 : Continuer à voter

| Description | Succès |
|---|---|
| Si l'utilisateur choisit de continuer à voter, il peut scanner un nouveau code-barres | Oui |

## UC 32 : Terminer de voter

| Description | Succès |
|---|---|
| Si l'utilisateur décide de terminer à voter, l'appareil demande le code PIN | Oui |

## UC 33 : Insérer code PIN

| Description | Succès |
|---|---|
| Les touches numériques permettent d'introduire le code PIN | Oui |

## UC 34 : Effacer code PIN

| Description | Succès |
|---|---|
| La touche C efface le dernier chiffre | Oui |

## UC 35 : Annulation de l'introduction du code PIN

| Description | Succès |
|---|---|
| Quand il n'y a plus de chiffre, la touche C annule l'introduction du code PIN | Oui |
| Une confirmation est demandée avant de quitter l'introduction du code PIN | Oui |

## UC 36 : Quitter l'application

| Description | Succès |
|---|---|
| La touche OK confirme la décision de quitter | Oui |

## UC 37 : Valider code PIN

| Description | Succès |
|---|---|
| La touche OK valide l'introduction du code PIN | Oui |
| Si le code PIN est erroné, le code correct est demandé | Oui |
| Suite à la validation, les votes sont envoyés à la carte | Oui |
| Une fois que la carte a reçu les votes, elle envoie une confirmation | Oui |

## UC 38 : Retirer la carte

| Description | Succès |
|---|---|
| Le retrait de la carte (simuler par un appui sur OK dans l'état correspondant) quitte l'application | Oui |

## UC 39 : Modifier le code PIN

| Description | Succès |
|---|---|
| L'utilisateur entre son nouveau code PIN | Oui |
| Il confirme son nouveau code PIN | Oui |
| Le nouveau code PIN est écrit sur la carte | Oui |
| Si la confirmation du nouveau code PIN diverge du premier code PIN, la modification est annulée | Oui |
| Le nouveau code PIN doit contenir au moins 4 chiffres | Oui |

## Autres tests

| Description | Succès |
|---|---|
| L'introduction de trois PINs erronés bloque la carte de vote | Oui |
| Les différentes communications NFC transmettent les messages attendus correctement | Oui |
| Le fichier XML contenant la réponse est généré correctement | Oui |
| Si un vote existe déjà sur l'appareil ou la carte, un message informatif s'affiche | Oui |
| La décision de ne pas écraser un fichier présent sur la carte retire ce fichier de la liste des fichier à envoyer à la carte | Oui |
| Si, suite à cela, il n'y a plus de fichiers à transférer, aucun message NFC n'est émis | Oui |
| Lors d'une interruption du cycle de vote, un contrôle est effectué si des votes attendent d'être transmis à la carte | Oui |
| A chaque redémarrage de l'application de la carte, un nouveau code PIN est généré | Oui |

Toutes les transitions apparaissant dans le diagramme de la machine d'état de l'appareil de vote ont également été testées sans être relevées en détails ici.

# E. Structure XML pour l'ajout de nouveaux événements de vote

L'ajout de nouveaux événements de vote sur la plateforme se fait par le moyen d'un fichier XML contenant toutes les informations nécessaires. Actuellement, la création de ce fichier se fait à la main. Ce descriptif contient les règles à respecter pour la composition de ce fichier, afin d'éviter des erreurs d'analyse lors de l'ajout dans la base de données.

## Structure générale

L'élément racine du fichier est l'élément « event ». Il ne peut être présent qu'une fois et il contient tous les autres éléments.

```
<swissivi:event xmlns:swissivi="http://projects.ti.bfh.ch/swissivi/ns/" id="event_id_1234"
date="2012-06-17" published="1">...</swissivi:event>
```

Ses attributs :
- xmlns :swissivi : namespace utilisé
- id : c'est l'identifiant de l'événement, il doit être unique pour tous les événements présents dans la base de données
- date : contient la date à laquelle cet événement à lieu (format YYYY-MM-DD)
- published : indique si l'événement est publié ou non (1=oui, 0=non)

Cet élément peut contenir zéro, un ou plusieurs des éléments suivants :
- votation
- initiative
- election

Les éléments doivent impérativement apparaître dans l'ordre mentionné ci-dessus et ne peuvent pas être mélangés.

## L'élément votation

```
<swissivi:votation id="17-06-2012-v1" level="0" levelref="0" lang="11111">...</swissivi:votation>
```

L'élément « votation » possède quatre attributs :
- id : identifiant de l'objet, il doit être unique pour tous les objets d'un même événement
- level : niveau auquel s'applique cette votation (0 : fédéral, 1 : cantonal, 2 : communal)
- levelref : référence à l'entité dans le niveau. Si level=0, alors levelref=0, si level=1, alors levelref contiendra le nom du canton auquel l'objet s'applique, si level=2, alors levelref contiendra le nom de la commune à laquelle l'objet s'applique.
- lang : indique dans quelles langues l'objet est disponible (1=disponible, 0=non disponible). La suite de bits représente, dans l'ordre, l'allemand, le français, l'italien, le romanche et l'anglais.

L'élément « votation » peut contenir zéro ou une occurrence de chacun de ces éléments :
- de : pour les textes allemands
- fr : pour les textes français
- it : pour les textes italiens
- rm : pour les textes romanches
- en : pour les textes anglais

L'ordre doit être le même que mentionné ci-dessus. L'apparition de ces éléments doit être consistante avec l'attribut « lang » de l'élément « votation ».

Chacun des éléments listés ci-dessus contient les trois éléments suivants :
– question : la question de la votation
– title : un titre pour cette votation
– signature : la signature numérique du texte de la question (codée en base64)

```
<swissivi:de>
    <swissivi:question>...</swissivi:question>
    <swissivi:title>...</swissivi:title>
    <swissivi:signature>...</swissivi:signature>
</swissivi:de>
<swissivi:fr>
    <swissivi:question>...</swissivi:question>
    <swissivi:title>...</swissivi:title>
    <swissivi:signature>...</swissivi:signature>
</swissivi:fr>
<swissivi:it>
    <swissivi:question>...</swissivi:question>
    <swissivi:title>...</swissivi:title>
    <swissivi:signature>...</swissivi:signature>
</swissivi:it>
<swissivi:rm>
    <swissivi:question>...</swissivi:question>
    <swissivi:title>...</swissivi:title>
    <swissivi:signature>...</swissivi:signature>
</swissivi:rm>
<swissivi:en>
    <swissivi:question>...</swissivi:question>
    <swissivi:title>...</swissivi:title>
    <swissivi:signature>...</swissivi:signature>
</swissivi:en>
```

## L'élément initiative

```
<swissivi:initiative id="17-06-2012-i1" level="0" levelref="0" lang="11110">...</swissivi:initiative>
```

L'élément « initiative » contient les mêmes attributs et le mêmes enfants directs que l'élément « votation ». Chaque élément de langue possède cependant trois questions, différenciées à l'aide d'un attribut « id ».

```
<swissivi:fr>
    <swissivi:question id="1">Initiative</swissivi:question>
    <swissivi:question id="2">Contre-projet</swissivi:question>
    <swissivi:question id="3">Préférence initiative - contre-projet</swissivi:question>
    <swissivi:title>Titre</swissivi:title>
    <swissivi:signature>signature codée base64</swissivi:signature>
</swissivi:fr>
```

## L'élément élection

L'élément « election » est l'élément le plus complexe. Il contient cependant les mêmes attributs que l'élément « votation » et l'élément « initiative ».

```
<swissivi:election id="17-06-2012-e1" level="0" levelref="0" lang="11111">...</swissivi:election>
```

Les élections cantonales ou communales ne possèdent qu'une seule circonscription électorale, le canton lui-même, respectivement la commune elle-même. Pour les élections fédérales, par contre, chaque canton forme un circonscription. La structure suivante permet de gérer ces deux cas.

L'élément « election » contient un élément « iname » qui permet de définir les circonscriptions. Celui-ci possède un attribut « ref » qui contient le nom de la circonscription. Cet élément n'est utile que pour les élections fédérales

et son attribut contiendra le nom de chaque canton. Dans le cas d'une élection cantonale ou communale, l'attribut « ref » sera une répétition de l'attribut « levelref » de l'élément « election ».

L'élément « iname » contient les descendants directs suivants :

- parameters : contient les paramètres des élections pour cette circonscription
- de : pour les textes allemands
- fr : pour les textes français
- it : pour les textes italiens
- rm : pour les textes romanches
- en : pour les textes anglais
- candidates : contient les candidats d'une liste, cet élément doit être répété autant fois qu'il y a de listes pour cette cirsconscription

L'élément « parameters » contient, dans l'ordre, ces deux éléments :

- candidate_repetiton : nombre de répétitions autorisées dans le bulletin d'élection pour un candidat
- candidates_number : nombre de candidats pouvant être élus

Chaque élément de langue contient ensuite :

- title : un titre pour cette votation
- signature : la signature numérique du titre (codée en base64)
- list : un élément représentant une liste, cet élément est répété autant de fois qu'il y a de listes pour cette circonscription

L'élément « list » apparaît dans chaque langue, car le titre d'une liste est différent pour chaque langue. Les candidats, eux, ne changent pas en fonction de la langue, c'est pourquoi on les retrouve dans l'élément « iname ».

L'élément « list » contient l'élément « party » contenant le nom du parti et l'élément « listnr » contenant le numéro de la liste.

Pour chaque élément de langue, on obtient donc le contenu suivant :

```
<swissivi:fr>
    <swissivi:title>...</swissivi:title>
    <swissivi:signature>...</swissivi:signature>
    <!-- Titles of the lists of candidates-->
    <swissivi:list>
        <swissivi:party>Union démocratique du centre</swissivi:party>
        <swissivi:listnr>1</swissivi:listnr>
    </swissivi:list>
    <swissivi:list>
        <swissivi:party>Parti socialist</swissivi:party>
        <swissivi:listnr>2</swissivi:listnr>
    </swissivi:list>
</swissivi:fr>
```

Il reste encore l'élément « candidates », descendant direct de l'élément « iname ». Il contient un ensemble de candidats d'une liste. Le numéro de cette liste est répertorié dans son attribut « listnr ». Il contient ensuite autant d'éléments « candidate » que la liste qu'il référence contient de candidats.

L'élément « candidate » possède trois descendants :
- firstname : prénom du candidat
- lastname : nom du candidat
- candidateid : identifiant du candidat

Cela nous donne donc le contenu suivant pour l'élément « iname » :

```
<swissivi:iname ref="bern">
    <swissivi:parameters>
        <swissivi:candidate_repetiton>2</swissivi:candidate_repetiton>
        <swissivi:candidates_number>26</swissivi:candidates_number>
    </swissivi:parameters>
     <!-- Beware of the order: german, french, italian, rumanch, english -->
    <swissivi:de>
       contenu vu précédemment
    </swissivi:de>
    <swissivi:fr>
       contenu vu précédemment
    </swissivi:fr>
```

```
    <swissivi:it>
       contenu vu précédemment
    </swissivi:it>
    <swissivi:rm>
        contenu vu précédemment
    </swissivi:rm>
    <swissivi:en>
        contenu vu précédemment
    </swissivi:en>
    <!-- List of candidates -->
    <swissivi:candidates listnr="1">
        <swissivi:candidate>
            <swissivi:firstname>Paul</swissivi:firstname>
            <swissivi:lastname>Exemple</swissivi:lastname>
            <swissivi:candidateid>1.1.1</swissivi:candidateid>
        </swissivi:candidate>
        <swissivi:candidate>
            <swissivi:firstname>Pierre</swissivi:firstname>
            <swissivi:lastname>Lapierre</swissivi:lastname>
            <swissivi:candidateid>1.1.2</swissivi:candidateid>
        </swissivi:candidate>
        <swissivi:candidate>
            <swissivi:firstname>Albert</swissivi:firstname>
            <swissivi:lastname>Einstein</swissivi:lastname>
            <swissivi:candidateid>1.1.3</swissivi:candidateid>
        </swissivi:candidate>
    </swissivi:candidates>
    <swissivi:candidates listnr="2">
        <swissivi:candidate>
            <swissivi:firstname>Ingrid</swissivi:firstname>
            <swissivi:lastname>De La Fontaine</swissivi:lastname>
            <swissivi:candidateid>2.1.1</swissivi:candidateid>
        </swissivi:candidate>
        <swissivi:candidate>
            <swissivi:firstname>Iris</swissivi:firstname>
            <swissivi:lastname>Lafeuille</swissivi:lastname>
            <swissivi:candidateid>2.1.2</swissivi:candidateid>
        </swissivi:candidate>
        <swissivi:candidate>
            <swissivi:firstname>Anthoine</swissivi:firstname>
            <swissivi:lastname>Cailler</swissivi:lastname>
            <swissivi:candidateid>2.1.3</swissivi:candidateid>
        </swissivi:candidate>
    </swissivi:candidates>
</swissivi:iname>
```

L'ordre des éléments joue également un rôle ici. L'élément « iname » est répété autant de fois qu'il y a de circonscriptions, autrement dit 26 fois pour des élections fédérales, une fois pour des élections cantonales et communales.

Cette structure est contrôlée lorsque le fichier est chargé dans la partie d'administration de la plateforme. La moindre erreur sera signalée et empêchera l'importation des informations dans la base de données. Un fichier d'exemple est disponible dans le dossier « private/xml/ » de la plateforme.