Smartcity

par

Tano Iannetta

Loan Lassalle

Luana Martelli

Wojciech Myszkorowski

Camilo Pineda

Jeremie Zanone

sous la direction du Professeur René Rentsch

Semestre de Printemps 2017

# Table des matières

[Table des matières 2](#_Toc482732809)

[1 Introduction 4](#_Toc482732810)

[1.1 Objectifs 4](#_Toc482732811)

[1.2 Description 4](#_Toc482732812)

[1.3 Organisation 4](#_Toc482732813)

[1.4 Répartition des tâches 5](#_Toc482732814)

[2 Conception / Architecture 6](#_Toc482732815)

[2.1 Fonctionnalités principales 6](#_Toc482732816)

[2.1.1 Cas d’utilisations 6](#_Toc482732817)

[2.1.2 Gestion, dans une base de données, de toutes les requêtes remontées par les citoyens 6](#_Toc482732818)

[2.1.3 Gestion d'un agenda 7](#_Toc482732819)

[2.1.4 Gestion des requêtes faites par les utilisateurs 7](#_Toc482732820)

[2.1.5 Filtrage des requêtes 7](#_Toc482732821)

[2.1.6 Ajout d'événements de la ville 7](#_Toc482732822)

[2.1.7 Implémentation d'une carte interactive 7](#_Toc482732823)

[2.1.8 Generation d'un PDF 8](#_Toc482732824)

[2.2 Technologies utilisées 9](#_Toc482732825)

[2.2.1 Maven 9](#_Toc482732826)

[2.2.2 Carte 9](#_Toc482732827)

[2.2.3 Base de données 10](#_Toc482732828)

[2.2.4 PDF 10](#_Toc482732829)

[2.2.5 GUI 10](#_Toc482732830)

[3 Implémentation 11](#_Toc482732831)

[3.1 Carte 11](#_Toc482732832)

[3.2 Base de données 12](#_Toc482732833)

[3.3 PDF 13](#_Toc482732834)

[3.4 GUI 14](#_Toc482732835)

[4 Tests de l’application 15](#_Toc482732836)

[5 Conclusion 16](#_Toc482732837)

[5.1 Etat des lieux 16](#_Toc482732838)

[5.1.1 Situation initiale 16](#_Toc482732839)

[5.1.2 Situation actuelle 16](#_Toc482732840)

[5.2 Problèmes rencontrés durant le projet 16](#_Toc482732841)

[5.3 Propositions d’améliorations 16](#_Toc482732842)

[6 Annexes 17](#_Toc482732843)

[6.1 Cahier des charges 17](#_Toc482732844)

[6.2 Journal de travail 17](#_Toc482732845)

[6.3 Planification 17](#_Toc482732846)

[6.3.1 Initiale 17](#_Toc482732847)

[6.3.2 Evolution réelle 17](#_Toc482732848)

[6.4 Tests 17](#_Toc482732849)

[7 Bibliographie 18](#_Toc482732850)

[7.1 Sites internet 18](#_Toc482732851)

[7.2 Livres 18](#_Toc482732852)

# Introduction

## Objectifs

Dans le cadre de ce cours il nous était demandé de remplir un certains nombres d’exigences afin de pouvoir rendre le projet viable .

## Description

Dans le cadre de ce projet, nous avons décidé après plusieurs choix d’idée de projets, de partir sur un projet de gestion d’une ville. Nous voulons implémenter une application permettant à l'administration d'une ville (dans notre cas Lausanne) d'organiser les requêtes (de réparations, d'évènements, de projet, etc.) venant de ses citoyens.

Dans notre application , plusieurs opérations s’offre à l’administrateur qui utilise cette plateforme. Dans un premier temps il pourra consulter les événements déjà présent dans la base de données. Dans un deuxième temps il devra gérer les requêtes qui arrivent dans l’application

Chaque requête doit être validée par l’administrateur et ajoutée à une rubrique dédiée. Il est ensuite possible de consulter ces différentes rubriques et de visualiser, au moyen d'une carte interactive, les endroits concernés. Pour une meilleure visibilité, chaque rubrique est associée à un filtre, rendant ainsi plus aisée la lecture sur la carte. La carte est interactive ce qui permet à la personne gérant cette dernière à la positionner selon ces besoins. Il est aussi possible pour l'administrateur de donner des priorités aux évènements (haute ou à titre informatif) afin de traiter plus efficacement les requêtes. L'administrateur pourra aussi consulter la carte selon une date précise. Finalement, l'administrateur pourra générer des PDF contenant des informations relatives aux évènements et aux rubriques qu'il aura précédemment choisies.

Dans le cadre de ce rapport nous allons vous exposer les technologies utilisées qui nous ont permis de rendre l’application tel qu'elle est. Nous allons aussi décrire les différentes fonctionnalité implémenté afin de mieux comprendre ce que permet notre application. Les problèmes rencontrés ainsi que les changement de conception seront aussi abordé afin d’expliquer nos choix quand au produit final . Nous sommes dans cette idée car le projet regroupés différentes technologies et permettait une bonne séparation des tâches au niveau de l'implémentation

## Organisation

Membres du projet:

* Chef de projet: Tano Iannetta
* Chef en second: Wojciech Myszkorowski
* Membre : Loan Lassalle
* Membre : Luana Martelli
* Membre : Jeremie Zanone
* Membre : Camilo Pineda

## Répartition des tâches

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Phases/membres | 1 | 2 | 3 | 4 | 5 | 6 |
| Dossier finale | 10% | 40% | 20% | 10% | 10% | 10% |
| Analyse | 22% | 22% | 22% | 22% | 22% | 22% |
| Conception | 5% | 5% | 5% | 5% | 5% | 5% |
| Réalisation | 25% | 15% |  |  |  |  |
| Administration |  |  |  |  |  |  |
| Tests |  |  |  |  |  |  |
| Documention de code | 5% | 5% | 5% | 5% | 5% | 5% |

# Conception / Architecture

## Fonctionnalités principales

### Cas d’utilisations

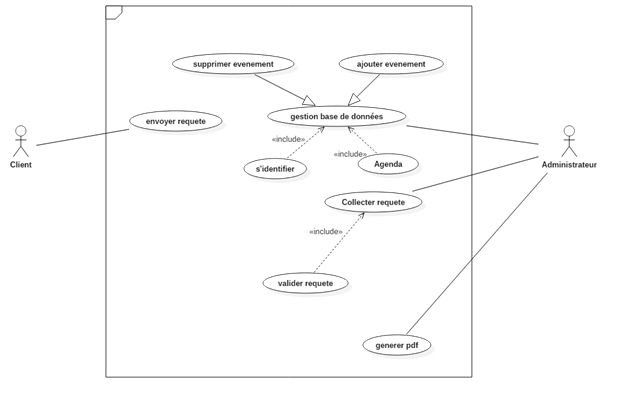


Diagramme des cas d’utilisations

Dans la figure ci-dessus figure les cas d’utilisations. L’administrateur est ici la personne qui va utiliser l’application que nous avons mis en place. Cette personne a la possibilité d’effectuer plusieurs actions. Elle peut tout d’abord valider des requêtes qui sont figurent dans une base de données. Cette personne peut aussi elle-même rajouter des événements dans la base de données alors modifier ceux y figurant déjà.

### Gestion, dans une base de données, de toutes les requêtes remontées par les citoyens

* Les requêtes appartiennent à une rubrique
* Travaux
* Accident
* Manifestations
* L'administrateur de l'application pourra gérer les rubriques.
* Un évènement aura une priorité donnée par l'administrateur.

### Gestion d'un agenda

* Pour toutes les requêtes, chacune aura une date de création et dans la mesure du possible une date de fin estimée.
* Organisation de la carte selon les filtres des rubriques et de la date (granularité au jour principalement).

### Gestion des requêtes faites par les utilisateurs

Les utilisateurs de l'application pourront effectuer plusieurs types de requêtes, classées selon les rubriques suivantes:

* Traffic
* Accidents
* Travaux
* Chantier
* Constructions
* Rénovations
* Culture
* Manifestations
* Doléances

### Filtrage des requêtes

L'administrateur s'occupera de filtrer les requêtes faites par les utilisateurs. Pour certains comptes privilégiés, il n'y aura pas besoin de l'intervention de l'administrateur.

* Gestion des évènements (acceptation de requêtes, refus)
* Comptes privilégiés (par exemple, les accidents proposés par les TCS sont directement validés)

### Ajout d'événements de la ville

Les utilisateurs pourront émettre une localisation d'une nouvelle manifestation qui sera approuvée ou non par l’administrateur.

### Implémentation d'une carte interactive

L'application comportera une carte interactive avec laquelle il sera possible d'interagir. Cette carte référence à l'aide de "tags" les événements ayant lieu dans la ville

* Affichage interactif de la carte (zoom, déplacement manuel)
* Ajout de différentes icônes (pins) sur la carte pour localiser les différents types d'événements
* Filtre des événements par date et par rubriques

### Generation d'un PDF

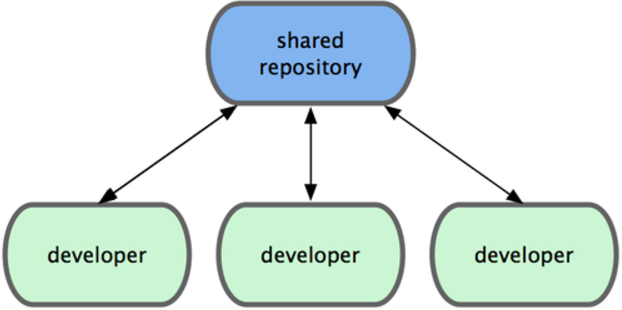
* Génération d'un PDF selon une rubrique choisie
* Ces PDF seront principalement destinés aux départements reliés aux rubriques, afin de garder un historique des événements. On peut imaginer qu'une fois générés, ils seront envoyés aux personnes concernées.
* Un PDF sera organisé en deux parties
* La première partie sera commune à tous les types de rubriques et contiendra les informations principales (nom de la rubrique, nom de l'événement, lieu, date, priorité et texte détaillant l'événement)
* La deuxième partie sera présentée sous la forme de statistiques. Elles seront personnalisées selon la rubrique. Par exemple, s'il s'agit d'un événement relatif à des travaux, on affichera le temps moyen de la durée des chantiers. S'il s'agit en revanche de doléances, on affichera plutôt le nombre de commentaires qui ont été publiés à ce sujet, et ainsi de suite pour les autres rubriques.
* Si plusieurs filtres ont été sélectionnés, alors une option sera de générer automatiquement plusieurs PDF.

## Technologies utilisées

### GitHub

Dans ce projet nous avons utilisé la plateforme GitHub comme moyen de mettre facilement en commun le travail de chacun dans un projet commun et de pouvoir voir les travaux des autres membres. Git est un logiciel de contrôle de version, ce qui signifie qu'il gère les modifications apportées à un projet sans écraser aucune partie de ce projet.

Les projets sur GitHub peuvent être consultés et manipulés à l'aide de l'interface de ligne de commande Git standard et toutes les commandes Git standard fonctionnent avec lui. GitHub permet également aux utilisateurs enregistrés et non enregistrés de parcourir les dépôts publics sur le site

Ce qui est vraiment pratique c'est que le répertoire de base situé sur un répertoire a distance peut être consulté depuis n'importe quel machine et n'importe quel moment.  
  
  
  
Plusieurs personnes peuvent être en train d'essayer d'accéder a la ressource en même temps et cela ne vas pas poser probleme.  
Le seul cas qui peut poser problème c'est lorsque 2 personnes ou plus travaillent sur la même ressources en même temps et décident de la remettre sur le serveur distant. Il y aura a ce moment la un conflit a résoudre. Il faudra décider quel parties garder de celles qui ont été modifié par les personnes

### Maven

Nous avons utilisé Maven pour la conception architecturale de nos fichiers.

Le fichier principal, le *pom.xml*, va donc décrire le projet. Nous y trouvons entre autres :

* l'endroit où se situent le code source et les fichiers de ressources
* la version de Java utilisée pour compiler le code source
* les dépendances directes du projet
* les plugins utilisés
* les rapports techniques demandés

Maven 2 insiste sur le principe de favoriser l'utilisation des conventions plutôt que de configurer son projet. Ainsi, si l'on respecte certaines conventions définies par Maven 2, il devient inutile de préciser certaines informations dans son *pom.xml*. Par exemple, Maven 2 préconise l'utilisation du répertoire *src/main/java* pour stocker les fichiers source du projet. En respectant ceci, il devient alors inutile de spécifier à Maven 2 où se trouvent les sources Java, ce qui allège d'autant l'écriture du *pom.xml*.¶

Avec ce simple *pom.xml* il devient possible de réaliser tout le processus de construction d'un projet avec Maven 2, à savoir :

* Gérer les fichiers de ressources
* Compiler les sources Java
* Compiler et exécuter les tests unitaires
* Créer le fichier JAR du projet
* Déployer le JAR

### Carte

L’utilisation des tuiles de openStreepMap nous a dirigé vers l’implementation de ..

Dans notre projet nous utilisons une carte interactive. Afin de dessiner cette carte, nous avons utilisé un système de coordonnées. Nous avons utilisé 2 systèmes différents :

* Un système de coordonnée sphérique (3 dimensions, nommé WGS84, ce système est utilisé par les GPS notamment).
* Un système de coordonnée cartésien qui s’appellera ici OSM (système utilise par les principaux outils en lignes comme OpenStreepMap).

Le Système WGS84 est pratique et l’un des plus courant pour représenter les positions dans l’espace ainsi que calculer des positions. On utilise le système OSM car celui-ci est très utile pour l’affichage à l’écran.

#### Système WGS84

Étant le système le plus utilisé de nos jours, nous avons décidé de l’utiliser afin de localiser nos différents évènements sur notre carte, c’est le système principal de notre projet.

Un point est représenté par trois caractéristiques :

* **La longitude** est une coordonnée géographique représentée par une valeur angulaire, expression du positionnement est-ouest d'un point sur Terre.
* **La longitude** est une coordonnée géographique représentée par une valeur angulaire, expression du positionnement est-ouest d'un point sur Terre (ou sur une autre sphère).
* **L'altitude** est l'élévation verticale d'un lieu ou d'un objet par rapport à un niveau de base. (dans ce projet ne tenant pas compte de cette composante car elle ne nous ait pas utile).

Le méridien de référence celui de Greenwich. La longitude est comprise entre -180o et + 180o les valeurs positives se trouvent à l’est et à l’ouest pour les valeurs négatives. La latitude se situe entre -90o et + 90o les valeurs positives se trouvant au nord et les valeurs négatives au sud. Dans ce projet nous n’avons pas tenu compte de l’altitude car ce paramètre ne n’affecte pas nos résultats d’affichages.

#### Système OSM

Dans ce projet nous avons décidé d’utiliser les cartes de OpenStreepMap du fait qu’elles sont gratuites et que nous pouvons les utiliser de manière ouverte. OpenStreepMap offre un avantage considérable aussi, celui d’utiliser le système OSM qui est le système cartésien le plus utilisé actuellement ce qui nous a permis d’avoir de la facilité quand l’implémentation de ce système cartésien. Principalement ce système nous permet de visualiser la carte du monde avec différentes échelles (Zoom dans notre cas).

Le zoom dans disponible est celui qui est compris dans un intervalle de [0 – 20] (le 20 est approximatif)

Le zoom 0 correspond au zoom le plus éloigné et nous permet donc de voir la carte dans son entier. A l’inverse le zoom 20 est celui qui nous permet de nous rapprocher et de centraliser une zone très précise. Nous avons décidé d’utiliser une taille de 256 pixels pour chaque côté. A chaque niveau de zoom la carte est deux fois plus grande que l’image précédente. De manière général le niveau de zoom permet de calculer les dimensions de la manière suivante : 2x x 256 ( taille en pixel d’un coté) = 2x+8

#### Conversion Système WGS84 en OSM

Pour passer d’un système à l’autre nous utilisons la projection cartographique, il existe une variété de projections permettant de passer d’un système à l’autre. Nous avons utilisé de Mercator. Pour chaque projection il existe plusieurs avantages et inconvénients, le désavantage de celui de Mercator est qu’il considère la terre comme étant une sphère parfaite. Ce qui n’est pas le cas dans la réalité.

### Base de données

### PDF

### GUI

# Implémentation

## Carte

Dans le cadre de notre projet, OpenStreepMap nous permet d’avoir un zoom d’un certain niveau dans notre cas un zoom de niveau 19 qui représente le zoom le plus proche. Si on prend la carte entière cela représente un nombre de pixels énorme de l’ordre 1815 pixels. Du cette quantité d’information, nous ne voulons utiliser qu’une infime partie. Afin de résoudre ce problèmes les gens utilisent des tuiles de petites tailles de 256 pixels de côté. Donc lors de l’initiation de la carte, on a seulement les tuiles visibles dans cette zone qui seront transmises au programme et assemblé au moment de l’affichage. Grâce à OpenStreetMap on identifie les tuiles avec 3 coordonnées le zoom, le x et le y.

Dans notre projet nous avons effectué plusieurs recherches afin de décider notre stratégie d’affichage de points colorié qui représentent les différents évènements. La création de tuile qui se superposent à notre carte nous permettent de tout simplement faire un calque sur notre carte et d’afficher les points par-dessus notre carte qui est une option plus facile à implémenter que le coloriage direct sur une carte.

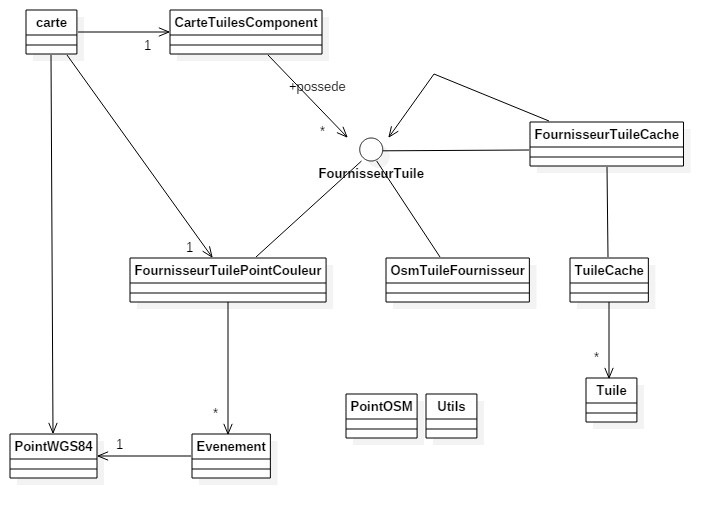


Schéma de la structure des classes utilisé pour construire notre carte

Nous avons une interface qui nous permet de redéfinir une méthode de calcul de tuile.

## Base de données

## PDF

Dans ce projet, il s’agit de garder une trace des données. Pour ce faire, nous avons décidé de donner la possibilité de générer un PDF contenant les informations relatives à la rubrique choisie. Ce document est composé de deux parties majeures. La première donne les informations importantes écrites des événements passés (comme le lieu et les dates). La deuxième partie est plus générale et comporte des statistiques.

Pour générer le PDF, nous avons choisi d’utiliser le kit de développement iText version 7. Ce support offre en effet plusieurs librairies qui permettent de créer facilement des PDF et laisse un grand degrés de liberté sur la gestion de la mise en page.

Pour la partie graphique, nous avons utilisé JfreeChart. Ce kit permet de créer toutes sortes de graphiques.

Les documents doivent pouvoir interagir avec la base de données pour récupérer toutes les informations nécessaires à sa création. Pour ce faire, nous avons créé une classe permettant de se connecter à la base de données et de lier les requêtes SQL avec des fonctions Java.

Finalement, un PDF est généré à partir d’un bouton sur l’interface graphique. Nous avons donc du lier les classes entre elles. De plus, si une erreur lors de la création du PDF est détectée, il faut pouvoir en informer l’utilisateur via l’interface graphique.

## GUI

# Tests de l’application

# Conclusion

## Etat des lieux

### Situation initiale

### Situation actuelle

#### Fonctionnalités non implémentées

## Problèmes rencontrés durant le projet

Durant ce projet nous avons rencontrés plusieurs problèmes auxquels nous n’étions pas préparé ou que nous avons sous estimé au niveau de la charge de travail. Lors de l'élaboration initiale nous avions conscience de la difficulté de certaines tâches. Le problème fut qu'à ce moment nous ne savions pas exactement quels technologies nous allions utiliser pour développer par exemple la base de données.

Tout d’abord la mise en place de la base de données fut beaucoup plus longue que prévu ce qui nous a retardé dans l'accès aux données afin de pouvoir les intégrer au projet centrale. La carte nous a posé d’avantages de problèmes que prévus. Nous nous sommes vites rendus compte qu’il fallait trouver un moyen de pouvoir les charger et les afficher rapidement. Après des recherches nous avons vu qu’il était possible de les mettre en cache , processus permettant des les faire apparaître plus rapidement sur Jpanel de l’application centrale. Un autre problème fut celui de mettre les grandes fonctionnalités sépare ensemble. Lors de l’élaboration nous avons décidé de partir chacun de notre côté et d’utiliser nos libraires respectives afin de faire fonctionner nos parties individuellement dans un 1er temps. Le problème fut lorsqu'il a fallu tout mettre ensemble. Il ne suffisait pas de créer un nouveau projet de rajouter les classes ou les different fichier simplement. Afin de remédier à ce problème , nous avons décidé de partir sur la création d’un projet Maven qui permet de lier différentes fonctionnalité très facilement.

## Propositions d’améliorations

# Annexes

## Cahier des charges

## Journal de travail

## Planification

### Initiale

### Evolution réelle

## Tests

# Bibliographie

## Sites internet

## Ouvrages