ENGEES
ÉCOLE NATIONALE DU GÉNIE DE L'EAU
ET DE L'ENVIRONNEMENT DE STRASBOURG

IUT Robert Schuman
Institut universitaire de technologie
Université de Strasbourg

AMELIORATION D'UN LOGICIEL D'ANALYSE DE DESSINS A MAIN LEVEE

Romaric Mathis

Tuteur professionnel
Florence Le Ber

Tuteur pédagogique
Nicolas Lachiche

TABLE DES MATIÈTRES

REMERCIEMENTS

Je tiens à remercier ma tutrice Florence Le Ber pour m'avoir permis de faire mon stage à l'ENGEES et de m'avoir suivi tout au long de celui-ci.

Je remercie également mon tuteur pédagogique Nicolas Lachiche pour avoir suivi mon travail.

Je tiens aussi à remercier Carine Heitz qui est à l'initiative du projet.

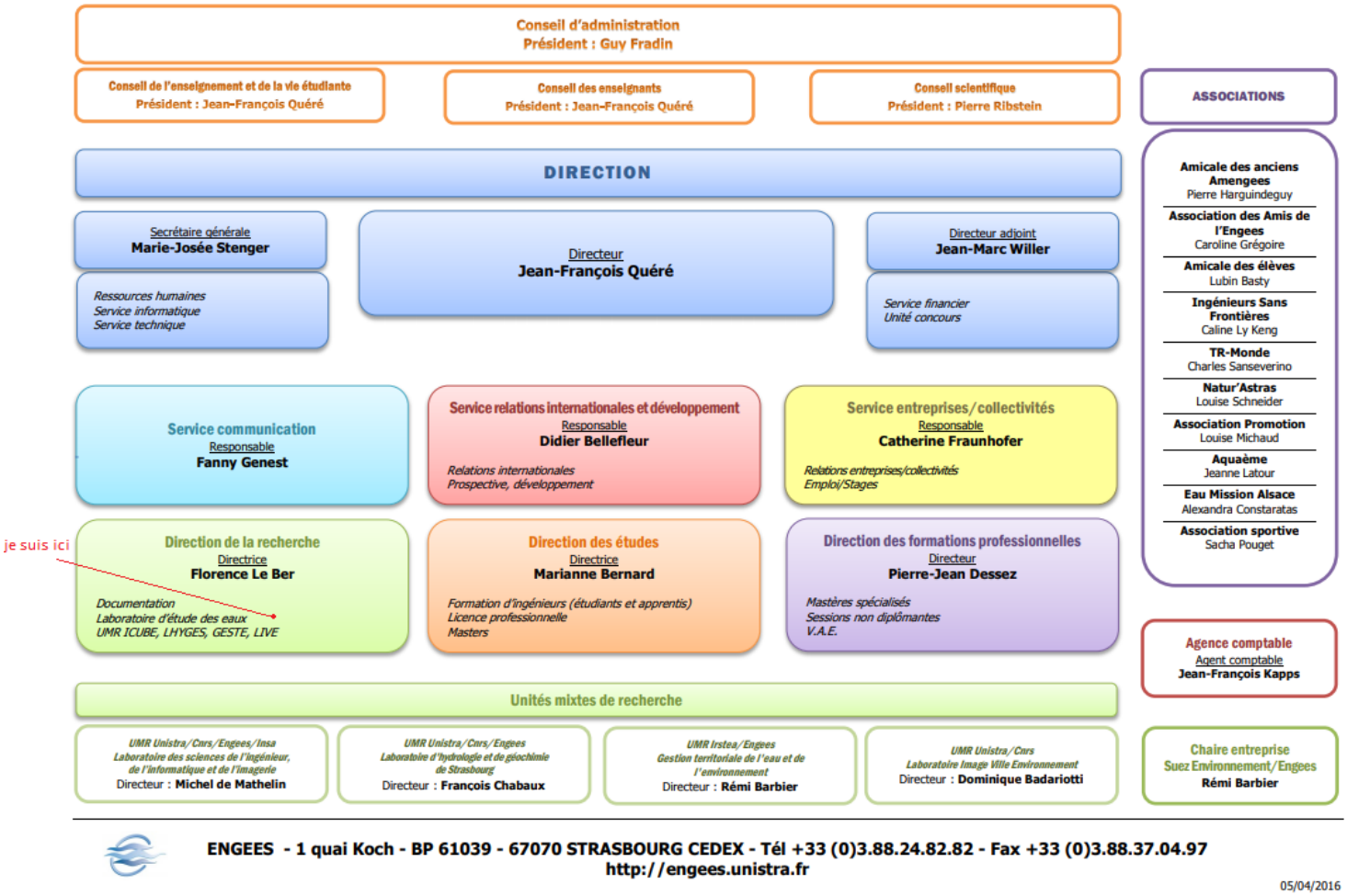
INTRODUCTION

L'Alsace possède plusieurs territoires qui présentent des risques de coulées d'eaux boueuses et sont alors soumis à des aménagements spécifiques telles que des haies ou des fascines bien que "*Ces ouvrages sont la plupart du temps efficaces mais ils modifient le paysage des habitants, souvent attachés à leur cadre de vie*", (Carine Heitz et al., 2018). Cela nous mène à nous poser quelques questions : comment et dans quelle mesure les habitants sont touchés par ces modification de leur environnement ? À quel point sont-ils informés sur les risques de coulées de boue ? Quelles sont les différences de perception du milieu en fonction des caractéristiques sociales de l'habitant ?

Afin d'apporter une réponse à ces questions, le projet Paysage a été proposé au sein de l'UMR GESTE et a été financés par l'ENGEES, une école d'ingénieurs dépendant du Ministère de l'Agriculture et de l'Alimentation. Ce projet consiste à enquêter les habitants, au moyen d'un questionnaire et de relever des schémas dessinés à main levée représentant la perception de l'environnement de l'habitant et des risques liés aux coulées d'eaux boueuses, puis à les analyser. Les objectifs des analyses sont de déterminer l'impact des aménagements contre les coulées de boue sur la relation des habitants avec leur milieu, d'identifier les différences existantes entre une représentation et une réalité physique et de déterminer les décalages de représentation entre différents groupes sociaux.

Pour ce faire, le projet paysage a besoin d'un logiciel d'analyse de schémas permettant de sortir des statistiques sur la perception des aménagements liés aux caractéristiques des enquêtés. Ce logiciel a été commencé par une apprentie en master informatique et science de l'image et est actuellement capable de segmenter les formes des schémas et de détecter dans certains cas leur nature. Cependant ce logiciel nécessite toujours une intervention humaine afin d'attribuer le type d'infrastructures des formes et de corriger les éventuelles erreurs commises lors de l'analyse de l'image. C'est pour cette partie que l'ENGEES m'a engagé afin que j'améliore l'interface utilisateur et développe des outils de correction d'analyse afin d'obtenir rapidement des statistiques utilisables.

L'ENTREPRISE



ENGEES - 1 quai Koch - BP 61039 - 67070 STRASBOURG CEDEX - Tél +33 (0)3.88.24.82.82 - Fax +33 (0)3.88.37.04.97
http://engees.unistra.fr

05/04/2016

1. ENGEES

L'Ecole nationale du génie de l'eau et de l'environnement a pour vocation de former des ingénieurs dans les domaines de l'équipement des collectivités, de l'aménagement durable du territoire, de la gestion des risques environnementaux et sanitaires et de la gestion des services publics.

- *Historique*

L'ENGEES est une école publique à caractère administratif sous la tutelle du Ministère de l'Agriculture et de l'Alimentation fondée en 1952 à Paris sous le nom de l'Ecole d'application des Ingénieurs des Travaux Ruraux. Elle fut transférée à Strasbourg en 1960 et changea alors sa dénomination en Ecole Nationale des Ingénieurs des Travaux Ruraux et des Techniques Sanitaires (ENITRTS). L'ENGEES prend sa dénomination actuelle en 1992.

- *Quelques chiffres clé*

L'ENGEES forme en continu 450 étudiants et 400 stagiaires et a déjà formé 3000 diplômés ingénieurs. L'ENGEES est composé de 70 agents permanents dont 24 enseignants-chercheurs

et ingénieurs de recherche et fait appel à plus de 200 intervenants extérieurs pour un budget annuel de 5,6 millions d'euros.

2. LES UNITÉS MIXTES DE RECHERCHES

- *GESTE*

L'UMR GEStion Territoriale de l'Eau et de l'environnement (GESTE) conduit des recherches appliquées dans les domaines de la gestion des services publics d'environnement (eau, assainissement, déchets) et de l'action publique environnementale envisagée à différentes échelles territoriales. Les chercheurs de GESTE développent à cet effet des méthodes, outils et concepts relevant de l'aide à la décision, de l'analyse et de l'évaluation de l'action publique, de la régulation économique des comportements et de l'ingénierie sociale.

- *ICUBE*-SDC

L'équipe « Science des Données et Connaissances » effectue des recherches dans les domaines de la fouille de données et de l'intelligence artificielle.

MISSION

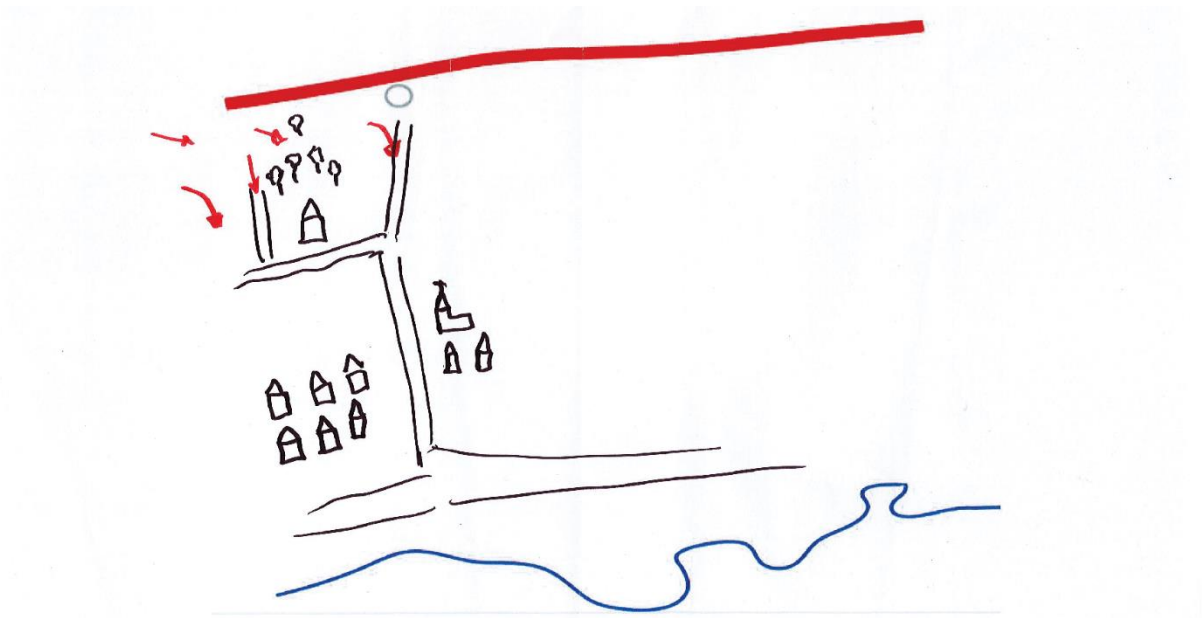
1. CONTEXTE

De plus en plus de communes en Alsace sont touchées par des coulées d'eaux boueuses qui engendrent des dégâts matériaux et écologiques. Afin de mieux comprendre et maitriser ces risques, les chercheurs s'intéressent en particulier aux perceptions des habitants. Le projet Paysage, proposé par Carine Heitz, géographe à l'ENGEES, en collaboration avec Florence Le Ber, consiste à analyser les représentations de coulées boueuses en Alsace. C'est dans ce projet qu'a été développé une application d'*"Extraction d'informations et reconnaissances de formes sur des cartes réalisées à main levée ", (Marine Ciron, 2017)*.

2. SUJET

La mission consiste à améliorer le logiciel d'analyse de schéma, avec reconnaissance de formes dessinées à la main sur feuille ou sur tablette par des habitants des zones enquêtées et qui représentent leur vision des risques liées aux coulées d'eaux boueuses.

Exemple de schéma



3. CAHIER DES CHARGES

Ma première tâche lors de ce stage a été d'analyser les besoins et d'établir un cahier des charges qui se trouve en annexe.

4. TECHNOLOGIES UTILISÉES

Le projet est codé en C++ et utilise le Framework Qt et une base de données SQLite.

ETAT DU LOGICIEL (v.0)

1. CHARGEMENT

a. Chargement initial

Pour créer une nouvelle session, il faut donner en entrée :

1. Un dossier source
2. L'image de fond de carte
3. Un tableau contenant la liste des esquisses avec les infos sur les auteurs

A partir de là, le logiciel charge toutes les images et une fois qu'elles sont toutes chargées, l'utilisateur peut commencer à travailler.

2. UTILISATION

a. Sélection

Le système de sélection consiste à cliquer sur une forme de l'image et à faire un choix parmi ceux proposés par une fenêtre dialogue, cela peut être une demande de saisie de type d'infrastructure (maison, église, …) si l'utilisateur a sélectionné avec un clic gauche ou une demande de saisie de type de forme (rectangle, …) si l'utilisateur a fait un clic droit.

ANALYSE DE L'EXISTANT ET PROPOSITION DE SOLUTIONS

Schéma d'exemple : tous les chiffres seront basés sur le traitement de cette image.

3. CHARGEMENT

a. Chargement initial

Le problème avec la méthode de la v.0 c'est que le chargement peut être long (environ 5 minutes pour 6 images), et qu'il est impossible de rajouter une esquisse après coup ; pour en rajouter une il faut créer une autre session et refaire le chargement de toutes les images.

Afin de réduire le temps d'attente et d'augmenter la flexibilité sur le chargement des esquisses, j'ai proposé un système qui permet de charger les images une par une et en arrière-plan pour ne pas bloquer l'utilisation de l'application pendant le chargement. De cette manière, l'utilisateur peut commencer à traiter les premières images pendant que les autres chargent. Ainsi le temps d'attente est réduit à environ 30 secondes, quelque soit le nombre d'esquisses.

b. Drag-and-drop

Pour le chargement des images en aval, j'ai proposé une méthode de drag-and-drop d'image, on sélectionne les images à charger dans l'Explorer Windows[1] et on les glisse dans la fenêtre du logiciel qui commencera à les charger.

4. ERGONOMIE

a. Sélection

Le problème avec la v.0 c'est que chaque forme non identifiée nécessite une saisie de l'utilisateur et que cela devient long et répétitif, environ 3 minutes pour le schéma donné en exemple.

[1] Explorer Windows : Explorateur de fichier de Windows.

La solution envisagée est de réduire les demandes de saisie, qui sont longues et désagréables. Pour ce faire, j'ai proposé un système de sélection qui prend en compte des listes de formes qui peuvent être complétées en faisant plusieurs sélections (en gardant la touche CTRL enfoncée). Dans la même optique de gain de temps en limitant le nombre d'actions, j'ai proposé un système de sélection de zones, en gardant le clic enfoncé et en le déplaçant sur l'image, le logiciel sélectionnera toutes les formes qui sont dans le rectangle formé par le point où le clic a été enfoncé et celui où il a été relâché. Cependant, il se peut que pendant la sélection multiple, avec CTRL enfoncé, l'utilisateur sélectionne une forme par erreur ; afin de lui éviter de devoir recommencer la sélection, j'ai fait en sorte que lorsqu'on resélectionne une forme sélectionnée, elle est désélectionnée.

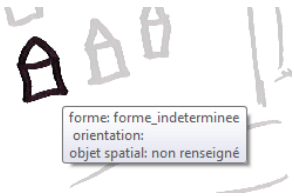
Ce système permet de faire toutes les saisies de type d'infrastructure sur l'image en 40 secondes au lieu de 3 minutes et en 6 saisies au lieu de 38.

b. Modification

Le logiciel ne possède pas d'outils pour modifier les esquisses ce qui peut poser problème quand par exemple une maison est coupée en deux parties car elle sera comptée comme deux maisons lors de l'analyse statistique. Dans la même idée, certaines esquisses possèdent du texte, ce qui est contraire aux consignes de l'enquêteur et cela peut être gênant dans la lecture ou l'analyse du schéma.

Pour régler ces problèmes j'ai proposé deux outils, un pour fusionner des formes et l'autre pour les supprimer.

c. Affichage



Actuellement, pour avoir des informations sur une forme, il faut mettre la souris sur la forme en question et attendre une info bulle qui contient la liste des caractéristiques. Cette action est pénible, si on veut vérifier que le schéma est bien analysé il faut passer sur chaque forme et attendre une seconde à chaque fois pour que l'info bulle apparaisse.

La solution choisie est d'assigner une couleur par type d'infrastructure et d'afficher une légende dans un panneau latéral.

5. ANALYSE STATISTIQUE

a. Carte moyenne

Dans un objectif de synthèse des données récoltées sur les esquisses, il est demandé de développer une fonction de génération de carte moyenne afin d'avoir un aperçu rapide de la représentation du milieu de vie par les enquêtés.

b. Sélection d'auteur

Afin de ressortir les différences de représentation du milieu par les enquêtés en fonction de leurs caractéristiques sociales, un sélectionneur d'auteur en fonction des caractéristiques sera implémenté afin de pouvoir comparer les cartes moyennes des hommes et des femmes par exemple.

c. Statistique avancée

Pour ressortir les différences de représentation des groupes sociologique, il a été demandé d'ajouter un outil de génération de statistiques à partir d'une sélection qui peut déterminer quelle catégorie sociale a majoritairement représenté un certain type d'infrastructure.

DEVELOPPEMENT (de la v.1)

0. LOGIQUE Qt

Qt est un Framework[2] multiplateforme permettant développer des applications en C++ avec interface graphique (GUI). Les deux grands principes sont les composants de la fenêtre et les signaux transmis entre eux.

Les composants de la fenêtre comme les boutons, les lignes de saisie de texte ou les panneaux graphique ont tous un fonctionnement de base commun (hérité du même objet source appelé QWidget) qui définit que chaque composant est fils d'un autre, dans une hiérarchie des composants, et que chaque composant peut en contenir d'autres.

Afin que chaque composant communique avec le programme, il y a un système de signaux et de slots. Un slot est une fonction qui peut être appelée par un signal de signature identique (qui utilise les mêmes paramètres que ceux du slot). Ainsi, lors de la création d'un composant on peut lier/connecter un signal du composant, par exemple un signal *void pushed()*, à un slot du programme qui ne prend pas de paramètre comme par exemple une fonction *void onButtonPushed()*. A partir de ce moment, la fonction *onButtonPushed()* sera lancée à chaque fois que le composant émettra le signal *pushed()*.

Ces mécanismes permettent d'avoir une application complexe tout en manipulant uniquement des objets simples.

1. CHARGEMENT

a. Objet de chargement (gestionnaire[3])

Pour pouvoir charger des esquisses à n'importe quel moment de l'utilisation de l'application, j'ai besoin d'avoir un objet qui s'occupe de mémoriser les paramètres de chargement initial et qui s'occupe de stocker une liste d'images à charger et qui les lance une par une. Pour cela j'ai développé un objet 'loader', géré en singleton[4], auquel on passe lors de sa création les paramètres de chargement dont le tableau avec les informations sur les auteurs et les chemins vers les esquisses (voir figure 1.a.1), le chemin vers le fond de carte et le chemin vers le dossier père.

Nom de l'image	Nom	Prenom	Age	Sexe	estHabitant
enquete23-05-1.png	Potter	Harry	42	homme	0
enquete23-05-2.png	Granger	Hermione	43	femme	0
enquete23-05-3.png	Weasley	Ron	44	homme	0

Figure 1.a.1

Une fois tous les paramètres entrés, l'objet de chargement récupère le format des descriptions des auteurs (la première ligne du tableau) et le mémorise pour l'appliquer aux esquisses qui seront chargées depuis le drag and drop car elles ne sont pas présentes dans le tableau d'entrée et donc ne sont pas décrites, puis charge le fond de carte et ajoute les chemins des images à charger qui se trouvent dans la première colonne du

[2] Framework :

[3] Gestionnaire : Module qui traite les actions de l'utilisateur, modifie les données des objets modèle (ex : une esquisse) ainsi que les données de la vue.

[4] Singleton : Gestionnaire sous forme d'objet.

tableau dans la liste des esquisses à charger et lance la fonction de chargement d'esquisses en parallèle dans un thread séparé.

b. Chargement d'une image

Le chargement d'une image se divise en 3 étapes : l'initialisation avec la lecture de l'image sur le disque et la suppression du fond de carte (si l'image provient d'un papier), la segmentation qui crée la liste des différentes formes de l'esquisse puis l'analyse des formes pour détecter les rectangles, flèches, courbes, etc. Afin d'optimiser l'utilisation du processeur, je dois diviser la charge de calcul de certains algorithmes pour les lancer en parallèle[5]. J'ai séparé ainsi l'analyse de chaque forme.

Lorsque le chargement de l'esquisse est terminé, la fonction se relance s'il reste des éléments dans la liste de chargement sinon elle se termine.

c. Drag and Drop

Grâce aux outils développé par Qt, le passage de fichier par drag and drop est assez simple, il suffit de définir que la fenêtre accepte les drops, d'écrire la fonction `dropEvent(QDropEvent *e)` et de récupérer les chemins vers les fichiers sélectionné dans l'objet e. Comme mon objet `loader` n'utilise que le chemin d'une image pour charger une esquisse, il me suffit d'appeler la fonction `loader.load(QString path)` pour charger les images sélectionnées.

2. ERGONOMIE

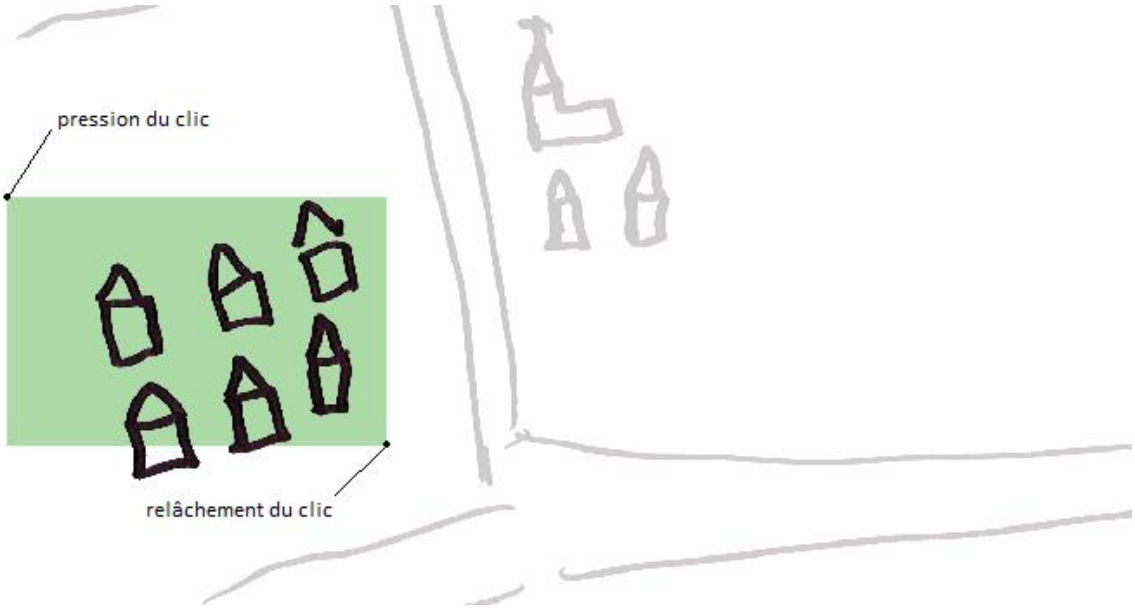
a. Sélection

Le mode de sélection de la v.0 consiste à cliquer sur un pixel dans le composant central, qui affiche l'image de l'esquisse, puis à regarder si une forme se trouve sur ce pixel, et enfin à envoyer un signal qui transmet l'identifiant de la forme qui se trouve sous la souris. Le message est reçu par un slot de type `recevoirSelection(entier)` qui demande ensuite à l'utilisateur d'entrer le type choisi pour la forme.

Pour pouvoir faire une opération sur plusieurs formes en même temps, je dois changer ce `signal(entier)/slot(entier)` en `signal(liste<entier>)/slot(liste<entier>)` et modifier la fonction du slot afin d'effectuer la modification sur toutes les formes de la liste d'entiers passée en paramètre.

Maintenant je peux envoyer des listes de formes entre le composant qui fait interface avec l'esquisse et le programme, il ne manque plus qu'à développer un objet qui permet de gérer facilement une sélection. Cet objet possède une matrice d'identifiants de forme qui permet de récupérer l'identifiant de la forme à partir des coordonnées du pixel cliqué. A partir de là il ne me reste plus qu'à envoyer deux points à l'objet de sélection pour qu'il puisse faire la liste des identifiants différents qu'il peut trouver dans le rectangle formé par ces deux points dans la matrice.

[5] Threading :



b. Affichage

L'affichage des types de formes par couleur se divise en deux étapes. La première consiste à pourvoir modifier l'image d'une esquisse afin de changer la couleur de tous les pixels d'une forme. La deuxième est la génération d'une légende des couleurs, permettant de modifier la couleur attribuée à un type d'infrastructure.

Pour colorier une forme je dois récupérer tous les pixels de celle-ci, pour cela j'accède à l'objet Forme qui se trouve dans l'objet Esquisse et j'y récupère la liste des points de la forme qui est la liste des pixels de la forme. Ensuite je parcours les points pour changer la couleur du pixel qui s'y trouve (voir résultat en figure 2.b.1).

Figure 2.b.1

Pour la modification des couleurs, l'utilisateur doit avoir accès à une liste de types d'infrastructure associé à une couleur et avoir la possibilité d'agir avec cette couleur. J'ai donc développé un composant `legende` qui s'occupe de générer un tableau à deux colonnes (QGridLayout) de widgets afin d'afficher et de lier des paires de composant (ici un label et un pushbutton, voir figure 2.b.2).

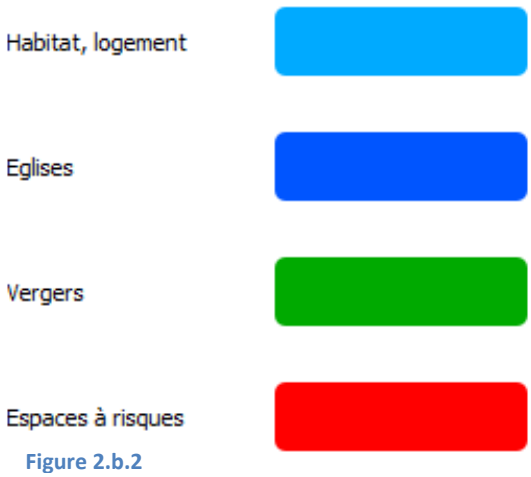


Figure 2.b.2

Maintenant qu'on possède un composant qui affiche la légende et qui permet de sélectionner des couleurs via un bouton, il ne reste plus qu'à opter pour une politique de transmission des signaux. Pour une utilisation plus simple dans le reste du programme, j'ai décidé de faire passer tous les signaux des composants de la légende par l'objet légende qui les retransmet au slot choisi lors de la création de l'objet.

Exemple d'instanciation de légende avec connexion du signal de modification de la valeur (sur la figure 2.b.1 la valeur est le bouton pour choisir la couleur)

```
couleurDessin = new Legende(this);
connect(legende, SIGNAL(valueSignal(QWidget*)), this, SLOT(changementCouleurType(QWidget*)));
```

Une fois les deux principales tâches réalisées, il faut les lier afin que lorsqu'on modifie le type d'une forme vers un type qui ne se trouve pas dans la légende, on génère une nouvelle paire `<key,value>` (ici `<label,bouton>`) dans la légende et que lorsqu'on modifie la couleur d'un type, on modifie la couleur de toutes les formes du type changé (ce qui se fait dans la fonction `changementCouleurType(QWidget*)` vue plus haut) en la couleur du bouton de la légende passé en paramètre (à travers le paramètre `QWidget*`).

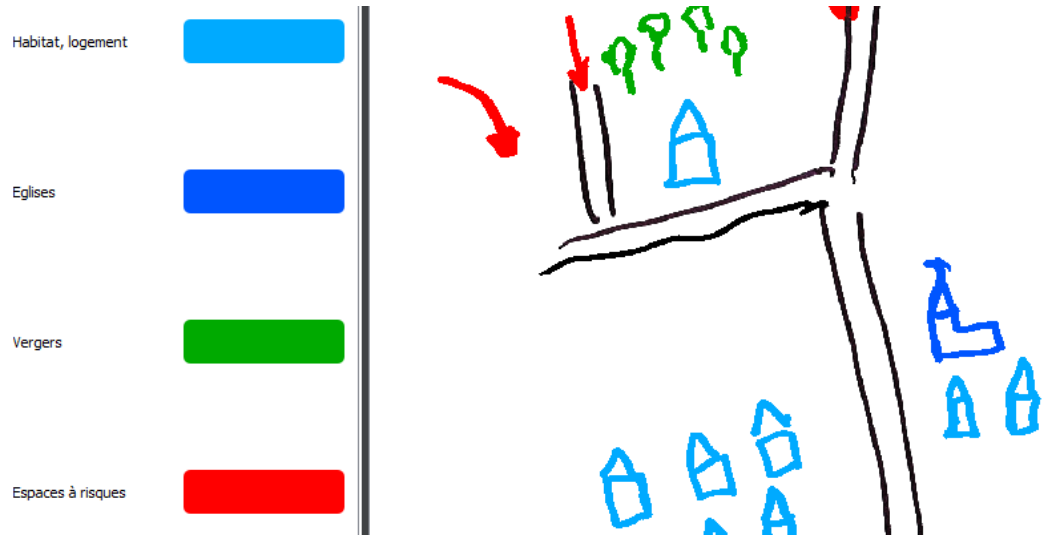


Figure 3 : Affichage des esquisses en fonction du type d'infrastructure des formes

c. Modification

- *Fusion de forme :*

  Les formes sont composées de plusieurs sous formes (nommé `fuzzyforme`) qui ensemble constituent l'ensemble des points de la forme. Pour fusionner deux formes, je dois ajouter les `fuzzyformes` d'une forme à la liste des `fuzzyformes` d'une autre puis supprimer la première forme.

- *Suppression de forme :*

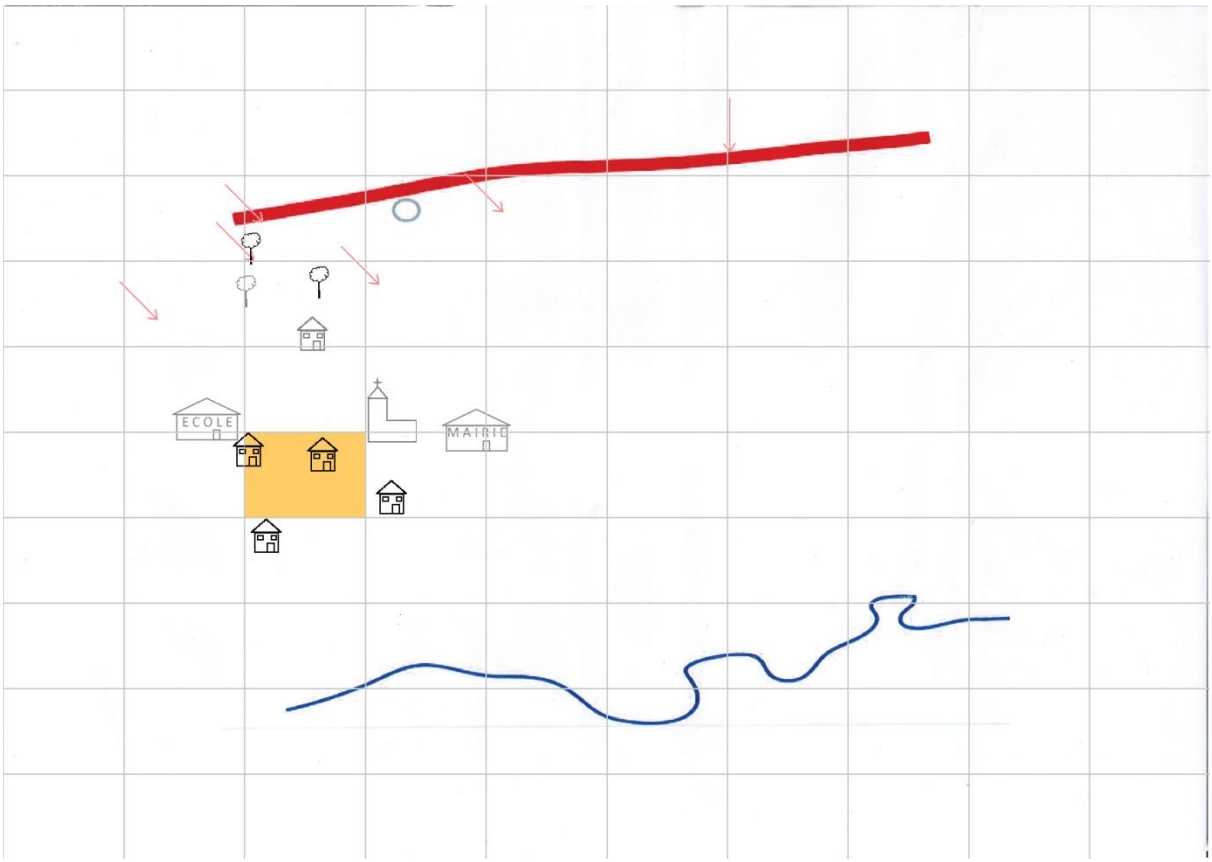
  Pour supprimer une forme, je dois dans un premier temps colorer en blanc tous les pixels de la forme sur l'image de l'esquisse puis ensuite l'enlever de la liste des formes de l'esquisse.

3. ANALYSE ET STATISTIQUE

a. Carte moyenne

Pour générer la carte moyenne j'utilise le quadrillage entré par l'utilisateur au lancement du logiciel et j'y récupère les carreaux afin de les utiliser comme rectangles de sélection. Une fois les formes de la sélection récupérées, je compte le nombre d'apparition de chaque type d'infrastructure que je stocke dans une map<identifiant, nombre d'apparition> pour obtenir la somme d'apparitions de chaque type d'infrastructure de toutes les esquisses. Ensuite je récupère la moyenne d'apparitions (somme / nombre d'esquisse) et je redessine les formes sur l'image de sortie. Lorsqu'un type n'a pas une moyenne entière, je dessine la dernière apparition en nuance de gris.

Figure 4 : Exemple de carte moyenne



b. Sélection d'auteurs

La sélection des auteurs est divisée en 2 parties : les conditions syntaxiques (pour le texte) et les conditions algébriques (pour les nombres). Pour la partie syntaxique il m'a suffi d'utiliser le module d'expression régulière de Qt. Pour les conditions algébriques, j'ai développé un lecteur syntaxique qui lit des expressions telle que >50&<70, une documentation pour les écrire est en annexe.

c. Statistiques avancées

[en cour de développement et fera partie de ma prolongation en CDD]

4. PERFORMANCE ET CORRECTION DE BUG

a. Calcul des contours d'une image

Dans la v.0, l'application s'arrête de fonctionner lors du chargement de certaines images. Après recherche, il se trouve que le plantage était causé par la fonction de détection des contours d'une forme. Comme la fonction pose problème uniquement sur certaines images, il s'agit sûrement d'un problème de gestion des cas limites. Pour calculer les contours, la fonction parcourt l'ensemble des points de la forme et considère comme contour, tous les points qui ont un voisin blanc. Le problème c'est qu'un point peut être en bordure d'image et lors de l'accès à son voisin il y a un dépassement de tableau et donc un plantage. Pour résoudre cela, il m'a suffi de considérer les bordures comme des pixels blancs.

CONCLUSION

Les améliorations que j'ai fourni au logiciel permettent de réduire le temps nécessaire pour réaliser une tâche du logiciel ainsi que de sortir des statistiques affichées de manière plus visuelle. De plus, j'ai ajouté plusieurs outils de modification de l'esquisse afin de simplifier les formes dessinées par les auteurs.

La plus grande difficulté était de rentrer dans le programme, de comprendre comment il a été pensé, afin de pourvoir faire des modifications et des ajouts sans risquer d'interférer avec les fonctions préexistantes. Cette tâche était d'autant plus difficile du fait que le C++ ne permet pas d'avoir des informations relatives à un plantage (même en mode debug) comme le font les autres langages orientés objet ce qui empêche tout débogage rapide. Il aurait été préférable pour le développement d'utiliser une technologie plus récente comme le C# qui est un langage de plus en plus utilisé et largement plus limpide que le C++ et qui permet aussi de manipuler la mémoire ou d'utiliser des fonctions codées en C.

J'ai beaucoup apprécié l'autonomie qui m'a été laissée pour développer le projet ainsi que la possibilité de proposer des solutions aux problèmes rencontrés. Le stage m'a permis d'apprendre un langage et à travailler en laboratoire.

Je suis content de mon stage et j'espère avoir satisfait les attentes de ma tutrice, Florence Le Ber, ainsi que la demandeuse du logiciel, Carine Heitz.

BIBLIOGRAPHIE

- Marine Ciron, « Extractions d'informations et reconnaissances de formes sur des cartes réalisées à main levée », mémoire de master ISI, université de Strasbourg 30 août 2017

- Carine Heitz et al., « Etude exploratoire des représentations de coulées d'eau boueuse en Alsace », soumis à SAGEO, Juin 2018.

- Carine Heitz, Projet paysage, financé par le CS de l'ENGEES, 2015

- ENGEES, Historique [en ligne]. ENGEES, [consulté en avril-mai 2018].
  Disponible sur : https://engees.unistra.fr/lengees/presentation/historique/

- ICUBE-SDC, Accueil [en ligne]. ICUBE-SDC, dernière mise à jour : 2016 [consulté en avril-mai 2018].
  Disponible sur : http://icube-sdc.unistra.fr/fr/index.php/Accueil

- GESTE, Projet paysage [en ligne]. GESTE, 2015 [consulté en avril-mai 2018].
  Disponible sur : http://geste.engees.eu/projet-Paysage

- GESTE, Identité [en ligne]. GESTE, [consulté en avril-mai 2018].
  Disponible sur : http://geste.engees.eu/identite

- Qt, Qt Documentation [en ligne]. The Qt Company Ltd, dernière mise à jour : 2018 [consulté en avril-juin 2018].
  Disponible sur : http://doc.qt.io/

- Mathieu Nebra et Matthieu Schaller, Programmez avec le langage C++, openclassroom.com, [consulté en avril 2018]
  Disponible sur : https://openclassrooms.com/courses/programmez-avec-le-langage-c