Fabien DONIUS
Chérine KAMEL
Chloé LE GUEVEL
Selim MILED

ING2 Gr6

Lib'R'traj

jeudi 14 février 2008

I. Sommaire :

II. Analyse du sujet :

Il s'agit pour ce projet de suivre une tendance en plein boum sur internet depuis l'été 2007 : les outils liés au désormais célèbre Vélib'. Comme tout moyen de déplacement, la problématique la plus souvent rencontrée est de savoir comment se rendre d'un point A à un point B, aussi rapidement que possible et avec le moins de contraintes de type coût ou stockage.

Face aux vastes offres de logiciels gratuits proposant les mêmes services, Lib'r'traJ se doit de fournir une prestation minimale. L'algorithme de détermination du chemin est bien sûr le point central et principal, mais de nombreux modules additionnels pourront en faire un outil véritablement utile. Les options prioritaires seront donc l'adaptabilité à de nouvelles cartes, prise en compte du sens de circulation, etc. Ces points sont détaillés dans la partie suivante.

Une partie importante de notre travail est également de maintenir notre logiciel dans les limites du bon sens : par exemple une station de Vélib' en pleine Seine ne devra pas être prise en compte… Tout du moins tant que Bertrand Delanoë n'aura pas annoncé un projet «Aqualib'».

III. Les modules envisagés :

Nous allons réaliser le module de base ainsi que les modules détaillés ci-dessous. Pour chacun des modules supplémentaires, des options seront disponibles. Celles-ci seront détaillées dans chaque partie dédiée au module.

1. Module de base

Le module de base permettra à l'utilisateur de trouver le chemin le plus court entre deux stations de Vélib' sélectionnées manuellement sur la carte. Aucun autre facteur que la distance ne sera pris en compte, à part évidement les axes de circulation disponibles.

2. Module sens de circulation

Le module sens de circulation permettra de se rapprocher beaucoup plus de la réalité des usages d'un moyen de transport tel que le Vélib' : le trajet proposé à l'utilisateur sera alors un trajet qu'il peut véritablement emprunter sans risquer de se retrouver bloqué ou à contre-sens.

Le respect des règles de sécurité routière implique la répression : des positions possibles de représentants de l'ordre seront enregistrées, et l'utilisateur averti des risques d'amendes qu'il encourt sur son trajet. Une telle sensibilisation ne peut être que bénéfique à la prévention routière, spécialement si l'utilisateur se rend compte du prix de revient de son trajet en prenant en compte une amende. Cet aspect est détaillé dans le module suivant.

3. Module durée/vitesse/coût

Dans ce module, une option permettra d'exploiter une subtilité du système Velib'. En effet, les trajets dont la durée est inférieure à 30 minutes sont gratuits. Ce module permettra de changer de vélo à intervalle régulier afin de ne pas payer, même si le temps total de trajet dépasse 30 minutes. Bien sûr, cette méthode rallongera la distance et le temps à parcourir, mais c'est une concession que de nombreux utilisateurs de Vélib' font régulièrement. L'emploi des voies sur berges pourra par exemple permettre d'écourter les trajets pour ainsi réaliser des économies.

a. Durée

La durée sera calculée selon la méthode suivante :

Durée = distance / vitesse = nb de pixel / (2pixels / seconde), avec l'option expliqué ci-dessous.

b. Vitesse

Nous considérons que la vitesse de déplacement sera de deux pixels par seconde avec la carte fournie dans le sujet. Nous pourrons de plus envisager une prise en compte d'une échelle de carte permettant d'afficher les distances et temps de parcours réels dans le cas d'une utilisation d'une carte avec une échelle différente.

c. Coût

Les tarifs utilisés dans ce logiciel seront ceux établis par la mairie de Paris :

- 30 premières minutes gratuites
- 30 minutes suivantes 1€
- Les tranches suivantes sont toutes à 2€
- Amende de catégorie C : 90€, circulation sur le trottoir

En raison de la grande inventivité des cyclistes et des pouvoirs publics, le logiciel pourra permettre de rajouter les nouvelles amendes (et infractions !) produites chaque jour…

4. Module affichage progressif du trajet

Un module d'affichage progressif du trajet pourra éventuellement être mis en place. Ce dernier permettra d'afficher progressivement le trajet à une vitesse choisie par l'utilisateur. Il s'agit en quelque sorte d'une simulation du trajet déterminé par le logiciel. L'affichage pourra se faire en temps réel ou plus rapidement.

Bien sûr, les cas de circulation illicite seront pris en compte et l'affichage progressif du trajet se fera même dans le cas d'une circulation à contre sens.

IV. Choix méthodologique :

1. Dijkstra

a. Pourquoi Dijkstra ?

L'algorithme de Dijkstra semble le plus adapté pour le parcours de l'arbre créé par traitement et reconnaissance d'image. Cet algorithme sera appliqué sur un graphe pondéré par la distance et orienté selon les sens de circulation des rues de la capitale.

b. Analyse descendante

Voici une analyse descendante de l'algorithme de Dijkstra qui sera utilisé pour le calcul des parcours entre deux sommets dans ce logiciel.

- On initialise la distance du sommet initial à 0 et à l'infini (où avec une valeur négative) pour les sommets non adjacents au sommet initial
- On ajoute le sommet s dans la file d'attente
- Tant que la file n'est pas vide
    - On retire le premier sommet de la file (x)
    - On marque le sommet retiré de la file
    - Pour tout sommet adjacent au sommet retiré de la file (x)
        - Si non marqué et D[y] supérieur à D[x] + poids c(x,y)
            - Mettre à jour D[y]

- Ajouter y dans la file de priorité puis la trier par ordre croissant des poids
- Stocker le prédécesseur x de y

- Pour lire le trajet partir de l'arrivée et remonter avec les prédécesseurs vers le départ (inverser à l'affichage).

2. Algorithme de pondération

Cet algorithme prend en compte toutes les options de pondération liées à la distance, donc au coût, à la durée, etc.

Avec l'algorithme de reconnaissance d'image sera implémenté un algorithme permettant de rajouter des données au graphe comme la pondération par la distance, le sens de circulation, et même éventuellement la fluidité de la circulation.

Nous pourrons également introduire une probabilité de présence policière permettant de calculer le montant des amendes en fonction des infractions éventuellement commises (pour que l'on considère une infraction comme commise, elle doit avoir été constatée par un policier, d'où la prise en compte de la probabilité de présence policière). A supposer que l'utilisateur indique au logiciel de ne pas tenir compte des sens de circulation, ce dernier calculera le montant du trajet en incluant les amendes.

3. Traitement d'image et détections

a. Traitement d'image

Pour traiter les points nous ne conserverons que le calque de couleur rouge. Pour le traitement des rues et sens de circulation nous conserverons les couleurs jaune, blanc et noir. Tout cela prenant bien sûr en compte une plage de couleurs qui sera déterminée expérimentalement : il est certain de ne pas toujours avoir en source, comme dans le cas présent, une carte parfaite dont toutes les entités de même nature sont monochromes.

L'image de base représentant la carte des stations de Vélib' sera divisée en plusieurs calques.

Les stations de Vélib' seront représentées sur un calque de couleur rouge, ce qui nous permettra de ne pas perturber la détection par d'autres données également présentes sur la carte.

Concernant le traitement des rues et des sens de circulation, nous utiliserons un calque jaune.

b. Détection

La détection des stations de Vélib' pourra se faire à l'aide d'un simple balayage de la carte afin de repérer les pixels rouges groupés en un amas assez important (la taille sera déterminée expérimentalement).

En ce qui concerne la détection des rues (et donc des axes qui relient les stations de Vélib'), nous pouvons partir de toutes les stations de Vélib', et scanner les rues qui en partent ou qui en viennent pour ensuite voir vers quelle station elles aboutissent. Cette méthode nous permettra d'établir d'un seul coup le graphe orienté utilisé dans la suite du logiciel.

V. Structure de données :

Cette partie détaille les structures de données que nous allons employer dans les algorithmes utilisés dans le développement du logiciel (comme des piles ou encore des files).

1. Graphe principal

Ce graphe représentera notre plan. Les sommets de ce graphe seront les stations Vélib' et les arêtes seront les rues reliant les stations Vélib' entre elles.

En mémoire, ce graphe sera représenté par une succession de listes chaînées. La liste des sommets sera une liste chaînée, et les arêtes aussi. Ainsi, le graphe sera entièrement dynamique et permettra d'adapter le programme à une carte différente, tout en fonctionnant normalement.

2. Structure image

Pour faire le traitement d'image nous devons définir une structure d'image avec conversion du JPEG en image BMP pour les traitements.

Pour le traitement d'images, les images seront ouvertes par le programme en JPEG ou en BMP. Cette méthode permet une plus grande souplesse pour l'utilisateur final. En revanche, nous devrons définir une méthode de traitement des images avec conversion du JPEG vers le BMP (ce dernier étant beaucoup plus simple à manipuler).

3. Structure option

Cette structure reprendra toutes les options et leurs états. Elle pourra être enregistrée pour retrouver ses options personnelles favorites à chaque lancement du logiciel, après enregistrement dans un fichier.

4. Structure nœud

C'est la structure principale du programme qui nous servira pour construire notre graphe mais aussi pour récupérer toutes les informations pour notre pondération.

VI. L'organisation :

Pour l'organisation de ce projet, nous utilisons Microsoft© Office Project 2007.

1. Les membres

Chaque membre aura une fonction bien définie dans le projet mais nous travaillerons tout de même ensemble sur sa totalité pour accélérer le processus, impliquant une connaissance de l'intégralité du projet par chacun.

Fabien sera chargé des algorithmes de Dijkstra et de pondération. Il aura aussi une vision complète du projet et assistera les autres en cas de besoin.

Chérine s'occupera de l'interface graphique, ainsi que de la détection des routes et sens de circulation.

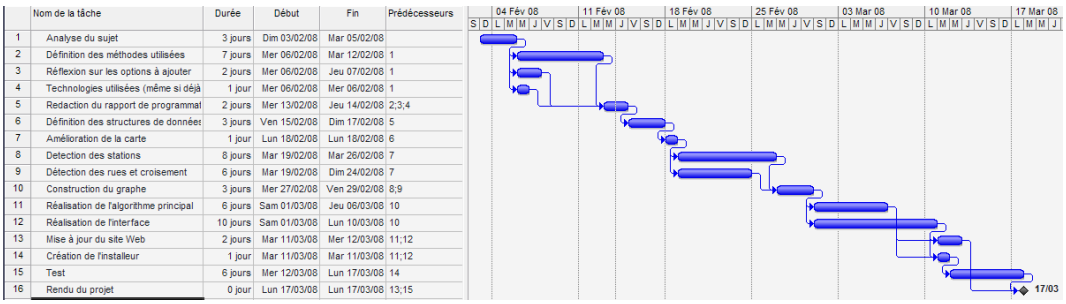
Chloé aura pour rôle la gestion des options et la création d'un design pour l'interface utilisateur.

Enfin Selim s'occupera de la détection des stations, et en collaboration avec Chérine, de la détection des rues et de la construction du graphe (les nœuds compris).

2. Les tâches

Vous pouvez retrouver l'organisation de notre projet en vous rendant sur notre site et en téléchargent notre fichier Office Project 2007.

Vous trouverez ci-dessous la liste des taches (tirée de notre fichier) ainsi qu'un graphe Pert et un organigramme de Grantt.

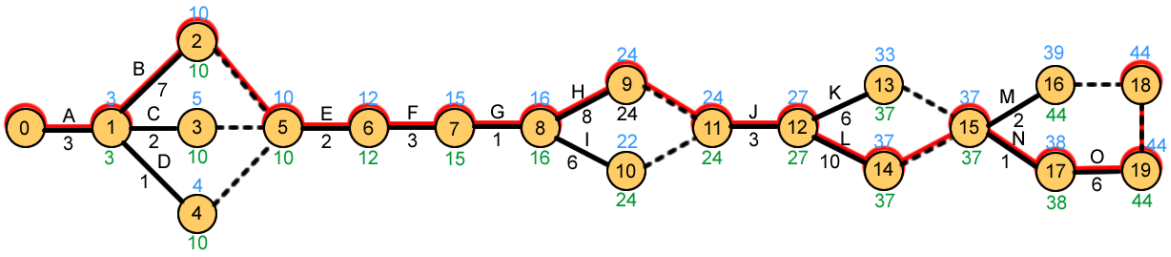


	Nom de la tâche	Durée	Début	Fin	Prédécesseurs
1	Analyse du sujet	3 jours	Dim 03/02/08	Mar 05/02/08	
2	Définition des méthodes utilisées	7 jours	Mer 06/02/08	Mar 12/02/08	1
3	Réflexion sur les options à ajouter	2 jours	Mer 06/02/08	Jeu 07/02/08	1
4	Technologies utilisées (même si déjà	1 jour	Mer 06/02/08	Mer 06/02/08	1
5	Redaction du rapport de programmat	2 jours	Mer 13/02/08	Jeu 14/02/08	2;3;4
6	Définition des structures de donnée	3 jours	Ven 15/02/08	Dim 17/02/08	5
7	Amélioration de la carte	1 jour	Lun 18/02/08	Lun 18/02/08	6
8	Detection des stations	8 jours	Mar 19/02/08	Mar 26/02/08	7
9	Détection des rues et croisement	6 jours	Mar 19/02/08	Dim 24/02/08	7
10	Construction du graphe	3 jours	Mer 27/02/08	Ven 29/02/08	8;9
11	Réalisation de l'algorithme principal	6 jours	Sam 01/03/08	Jeu 06/03/08	10
12	Réalisation de l'interface	10 jours	Sam 01/03/08	Lun 10/03/08	10
13	Mise à jour du site Web	2 jours	Mar 11/03/08	Mer 12/03/08	11;12
14	Création de l'installeur	1 jour	Mar 11/03/08	Mar 11/03/08	11;12
15	Test	6 jours	Mer 12/03/08	Lun 17/03/08	14
16	Rendu du projet	0 jour	Lun 17/03/08	Lun 17/03/08	13;15

3. Graphe de Pert

Graphe de Pert :

Prédesseceurs	N°	Tache	Durée	Successeurs
	A	Analyse du sujet	3 jours	B, C, D
A	B	Définition des méthodes utilisées	7 jours	E
A	C	Réflexion sur les options à ajouter	2 jours	E
A	D	Technologies utilisées (même si déjà très limité par le sujet)	1 jour	E
B, C, D	E	Redaction du rapport de programmation	2 jours	F
E	F	Définition des structures de données utilisées précise	3 jours	G
F	G	Amélioration de la carte	1 jour	H, I
G	H	Detection des stations	8 jours	J
G	I	Détection des rues et croisement	6 jours	J
H, I	J	Construction du graphe	3 jours	K, L
J	K	Réalisation de l'algorithme principal	6 jours	M, N
J	L	Réalisation de l'interface	10 jours	M, N
K, L	M	Mise à jour du site Web	2 jours	P
K, L	N	Création de l'installeur	1 jour	O
N	O	Test	6 jours	P
M, O	P	Rendu du projet	0 jour	

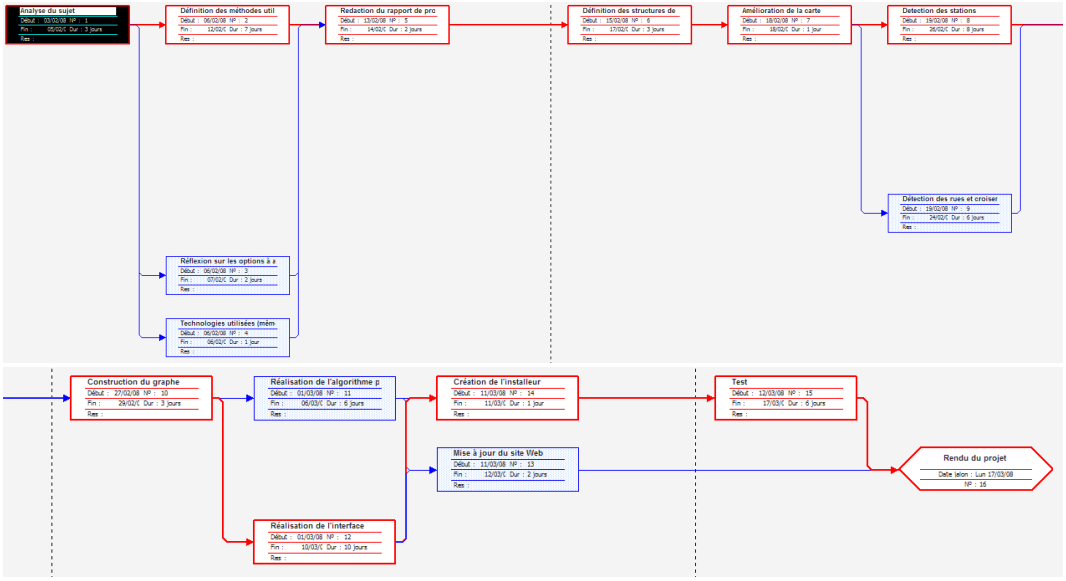


Marges totales :

N°	Tache	Marge
A	Analyse du sujet	0
B	Définition des méthodes utilisées	0
C	Réflexion sur les options à ajouter	5
D	Technologies utilisées (même si déjà très limité par le sujet)	6

E	Redaction du rapport de programmation	0
F	Définition des structures de données utilisées précise	0
G	Amélioration de la carte	0
H	Detection des stations	0
I	Détection des rues et croisement	2
J	Construction du graphe	0
K	Réalisation de l'algorithme principal	4
L	Réalisation de l'interface	0
M	Mise à jour du site Web	5
N	Création de l'installeur	0
O	Test	0
P	Rendu du projet	0

Réseau de tâches :



En rouge il s'agit du chemin critique.

VII. L'interface :

1. Pourquoi Allegro ?

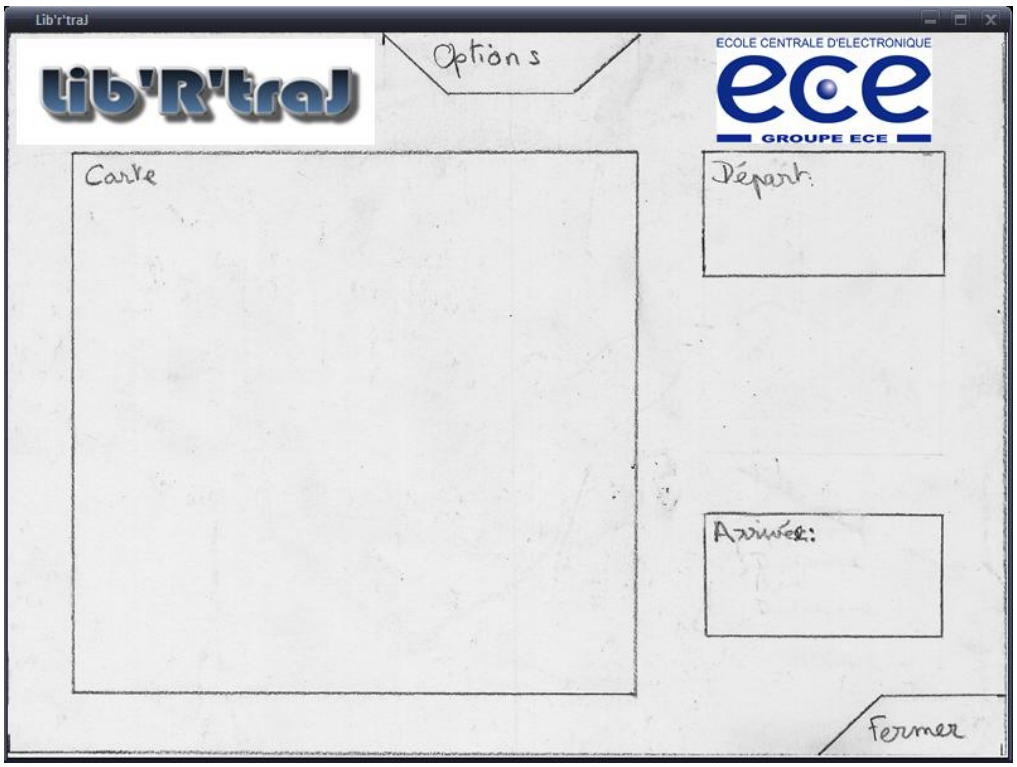
Pour permettre l'interactivité, le traitement des images et la facilité de sélection.

Allegro nous permettra de dessiner l'interface exactement comme nous le désirons. En effet, Allegro étant une librairie purement graphique, les interfaces pourront alors être dessinées au pixel près, ce qui rend très pratique la création d'un design propre poussé.

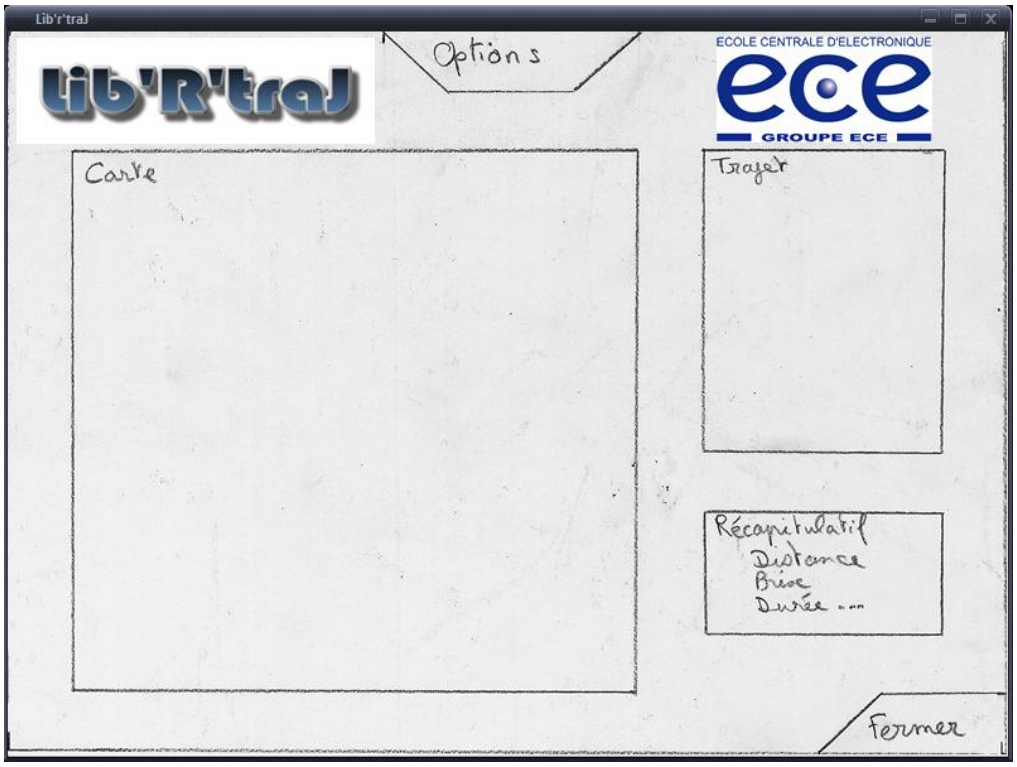
De plus, Allegro permet une bonne manipulation des images au format BMP (et également des JPEG avec l'extension JPGAlleg), ce qui sera nécessaire pour la détection des éléments sur la carte.

2. Maquette / User board

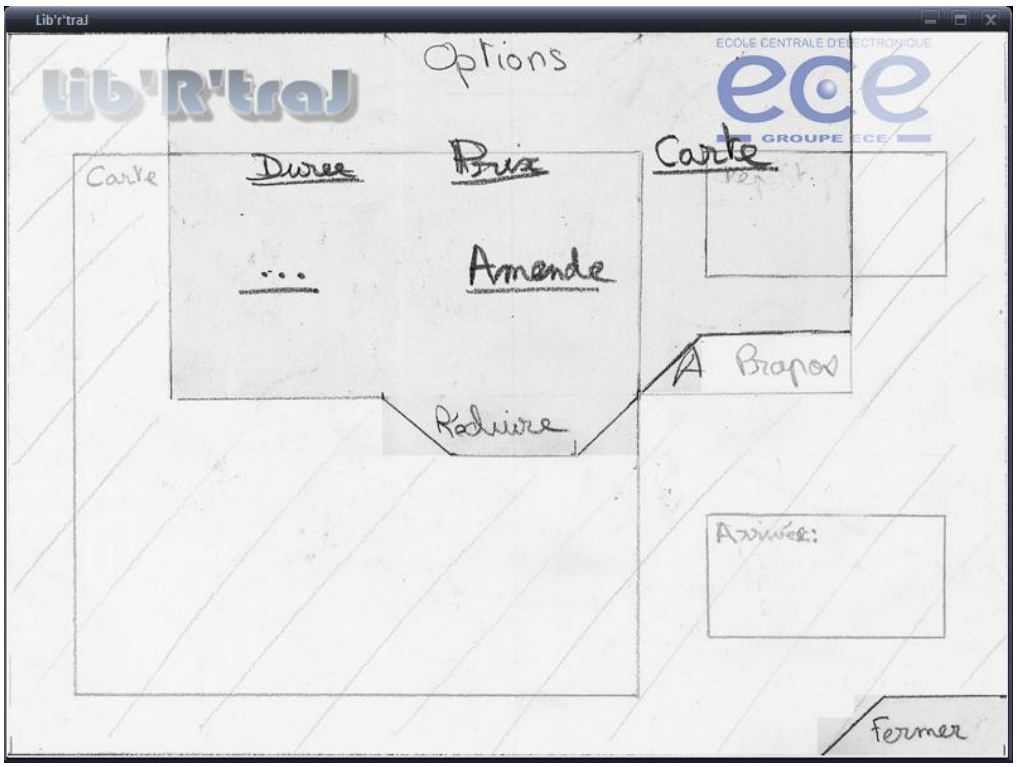
Voici un aperçu de l'interface utilisateur de Lib'r'traJ.



Cette image représente le logiciel après son lancement, l’utilisateur aura simplement à cliquer sur son point de départ (qui s’affichera dans le cadre départ) et son point d’arrivé (idem départ) sur la carte.



Ensuite le logiciel pourra automatiquement calculer le trajet optimum pour aller du premier point au second. Il affichera le chemin sur la carte, indiquera les informations telles que le prix, la durée et la distance ainsi que le chemin en mode texte (tourner à gauche droite etc.).



Enfin l'utilisateur pourra régler tous les paramètres du programme dans l'onglet Options.

VIII. Nos plus

1. Les options

Comme expliqué précédemment, les logiciels de navigation Vélib' sont nombreux sur internet, les options que nous nous proposons d'ajouter sont alors un plus indispensable pour tenir la route face à ces produits, souvent également amateurs.

De plus nous avons compris lors de l'élaboration de nos précédents logiciels l'importance d'une grande modularité : l'ajout d'algorithme correcteurs d'erreurs pour CCE testeur, de nouveaux stages pour History of Lemmings, ou la possibilité de modifier le terrain dans Cot Cot Cothello sont autant de raisons d'axer Lib'r'traJ sur l'évolutivité, spécialement lorsque l'on considère le nombre de stations Vélib' en constante augmentation, ou la variation au cours d'une année de la circulation et des prix de revient des moyens de transport.

Les options disponibles seront un grand plus… Selim je veux ton blabla

2. Choix dans la carte (et pas dans la date)

Si notre algorithme de détection des rues est suffisamment performant, on pourra charger une autre carte plus détaillée de la ville. C'est pour cela qu'il est si important de le tester dans des conditions limites et de s'assurer de son bon fonctionnement dans diverses situations. Ainsi, le logiciel ne se limitera plus à la carte fournie avec le sujet, le faisant par la même passer du projet scolaire à un véritable outil pratique.

3. Un projet concret

a. Site web (http://librtraj.crae-prod.info)

Le suivi de notre projet sera fait, avec un site web, qui évoluera en même temps que le logiciel. Celui-ci permettra dans un premier temps d'accéder à nos ressource (Notre fichier Office Project,

le lien vers le SVN et pourquoi pas les versions alpha et beta de notre programme). Ensuite ce site servira à distribuer le logiciel mais aussi fera un lien vers notre bug tracker.

b. Outils professionnels

Nous utiliserons pour faire ce projet des techniques professionnelles de pointe. En effet, la gestion du temps sera effectuée par Office Project 2007. La gestion des versions sera effectuée par un serveur SVN (en utilisant le module AnkhSVN ajouté à Visual Studio ou TortoiseSVN qui s'intègre à Windows) et nous utiliserons un bugtracker (Mantis) pour suivre les bugs et en attribuer la correction à un membre du groupe.

De plus, la programmation du projet s'effectuera sous Microsoft Visual Studio 2008 afin de bénéficier d'un environnement de développement fiable et souple, et d'un débogueur C performant.