TP LONG OU

LE GRAND BAIN DU TRAVAIL COLLABORATIF

ENSEM 3A ISA – SECTION SERTR

REMERCIEMENTS

Nos plus profonds remerciements s'adressent bien sûr en priorité à toute l'équipe qui a joué le jeu du mieux qu'elle le pouvait, qui a travaillé jusqu'au bout pour obtenir des résultats, qui a su aussi faire preuve d'humour certaines fois mais surtout prendre ses responsabilités lors des moments difficiles.

Merci à Jean-Luc Metzger et Christophe Bouthier d'avoir été là et de nous avoir aidé spontannement lorsque nous n'osions pas demander de l'aide ou lorsque nous étions trop plongés dans nos problèmes pour y penser.

Enfin, un grand merci à Jean-Claude Derniame pour l'instauration de ce cours, ses astuces, conseils et anecdotes personnels qui nous seront sûrement très utiles pour notre future entrée dans le monde du travail et notamment, celui du travail collaboratif dans le développement informatique.

"Un petit programme est un petit paquet d'erreurs.
Un gros programme est un gros paquet d'erreurs."
Jean Claude Derniame

"Le projet a son avenir derrière lui."
Nicolas Griedlich

"Un Didion vaut mieux que deux tu l'auras."
Pensée collective

SOMMAIRE

PRÉSENTATION

Dans cette partie, nous parlerons de tout ce qui atrait à la situation initiale du projet. En reprenant le sujet proposé initialement (celui distribué sous format papier), nous verrons quelles simplifications ont été adoptées, pourquoi, et nous finirons par présenter les choix techniques pour lesquels nous avons opté.

1. Description du sujet :

Nous avons reçu, lors de la première séance, le sujet de cette année sous la forme du cahier des charges suivant :

Cahier des charges :

Objectif : mettre en œuvre un outil d'aide à la décision pour une politique de gestion des feus de circulation dans une ville, décrite plus loin.

Objectifs pédagogiques :

- Mise en œuvre de technologies de développement
- Approche du travail en équipe de taille conséquente et conduite de projet logiciel

Critères d'évaluation :

- Respect des contraintes
- Bon fonctionnement de la gestion de l'ensemble des feux et proposition de politique efficace selon l'importance du trafic
- Agrément et sophistication de l'interface graphique réalisée
- Communication entre les sous-groupes de travail

Contraintes :

- Rendre le projet avant le 17 mars 2008 17h17 sous forme d'un rapport collectif, d'une démonstration et des rapports individuels.
- Mettre en place une simulation discrète permettant de vérifier la validité du système réalisé pour des politiques correspondantes à différentes charges de circulation, par exemple selon les moments de la journée
- Avoir une interface permettant de visualiser l'évolution en pas à pas de la simulation d'un part et un synoptique montrant les chiffres clef pour les politiques retenues (trafic, arrivées, écoulements, files d'attentes au différents feux)
- Avoir une application répartie su au moins 3 sites.
- Utiliser les technologies UML, Java, RMI

La ville :

Pour ce qui concerne ce projet, la ville se limitera à deux grands axes de circulation à sens uniques : NS, SN, EO et OE

- Sur chacun de ces axes : des feux. La distance entre ces feux est connue :
    - □ Pour NS : 300m avant l'arrivée au centre, 200m pour les 3 feux suivants
    - □ Pour OE, 200m avant l'arrivée au centre, 300m pour les 3 feux suivants
    - □ Pour le centre, 50m entre ces deux feux
- Chacun des axes est doté d'une onde verte réglée à 47km/h
- Les véhicules arrivent aux entrées de la ville selon une loi de poisson
- 65% des véhicules respectent la vitesse de 47km/h, les autres allant plus ou moins vite selon une loi normale
- Chaque véhicule met 1 (80%), 2 (10%), 3 secondes (10%) pour redémarrer, avec une accélération constante jusqu'à la vitesse fixée
- Tous les véhicules mesurent 4 mètre de long, et sont tous blancs
- Les feux du centre sont équipés de caméras
- Tous les feux sont équipés d'un dispositif de comptage des véhicules en attente devant eux.
- Les voix latérales peuvent évidemment apporter des perturbations que l'on étudiera
- Des véhicules prioritaires peuvent arriver à tout moment

2. Interprétation du sujet

1. Simplifications :

Quelques simplifications et spécifications ont été décidées pour traiter le problème :

- Le freinage des voitures est jugé instantané
- Les routes font 4 mètres de large -> le carré central fait 42 m de côté
- Les voitures sont toutes blanches (ce qui est beau et pur)

L'aspect graphique qui a été adopté pour la ville est l'image haute définition (7500x7500) suivante :

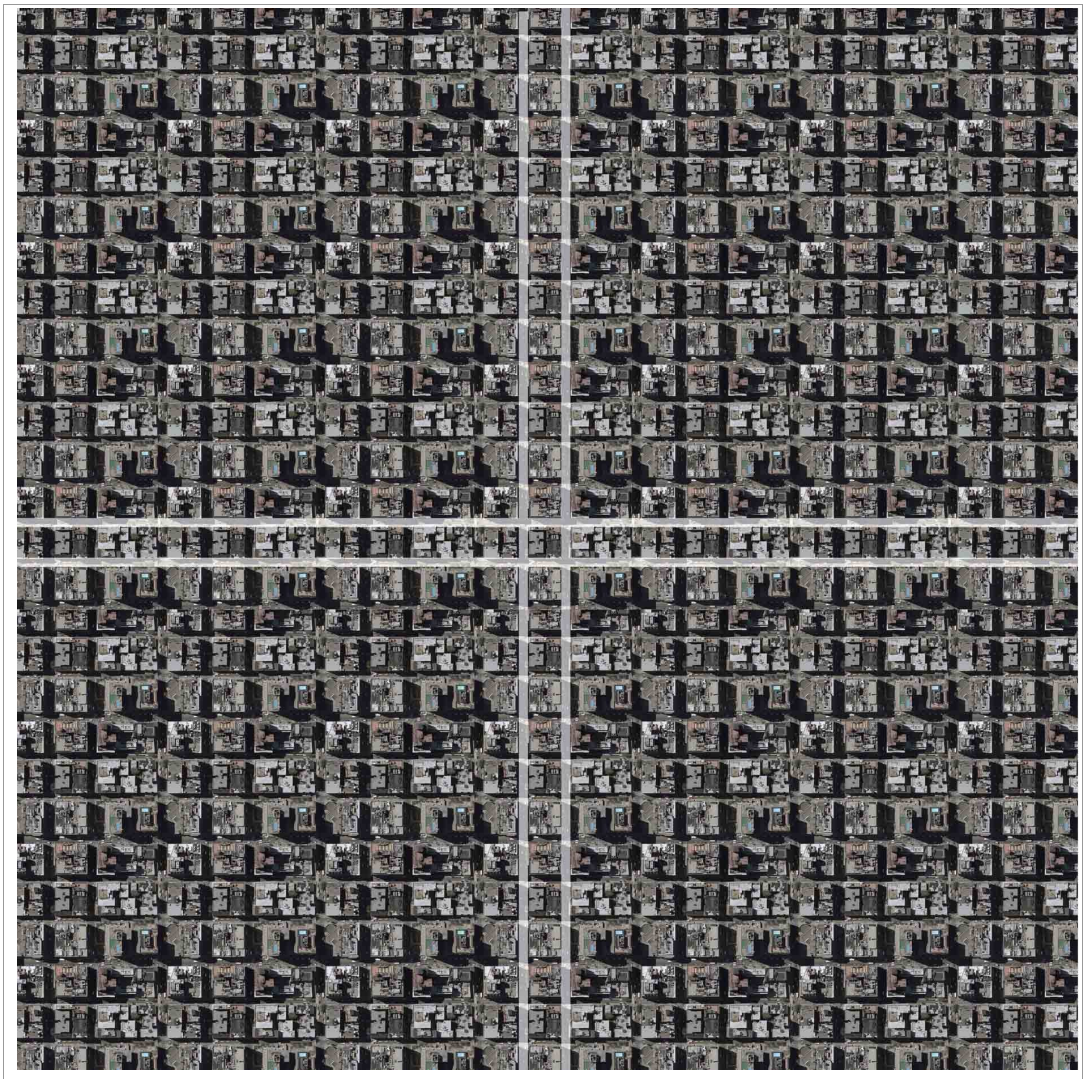


Illustration 1: Fond de carte de la ville finale

2. Produit fini :

Le programme fourni au client doit pouvoir lui montrer la présence de voitures mais aussi les flux qui évoluent dans la ville. Pour pouvoir juger lui-même de l'influence des paramètres du système, on se propose de concevoir une interface interactive qui permettra au client de :

- Jouer sur la rapidité de la simulation
- Faire varier les paramètres de création de voitures
- Programmer des politiques à suivre
- Voir s'afficher les statistiques instantannées concernant le traffic dans la ville

Lorsque le client arrête la simulation, il a accès aux statistiques post-mortem obtenues grâce à toutes les informations retenues. Il a ensuite possibilité de recommencer une simulation snas avoir à relancer le programme.

Vous trouverez en pièce jointe de ce document le contrat client signé qui présente les différentes caractéristiques du produit fini de façon très précise.
Depuis il n'a pas beaucoup évolué, si ce n'est que l'équipe a pris un certain retard (tout à fait normal vis à vis des standards du monde de la programmation) et que certaines fonctionnalités ne pourront jamais être implémentées (accès distant RMI, gestion des véhicules prioritaires, gestion des accidents).
Nous reviendrons là-dessus par la suite, au moment de la recette dans la partie *Avancement du projet* en page 23 où tout celà est précisé.

3. Choix techniques

Pour programmer et finaliser notre logiciel, nous avons décidé d'utiliser les outils suivants, qui ont été installés et mis en place par notre groupe :

- La salle 204 jaune et ses 8 ordinateurs sont à notre disposition, équipés du système d'exploitation Ubuntu
- nous utilisons Eclipse pour la programmation
- pour effectuer un travail coopératif, nous centralisons nos classes grâce au Google Code et SVN
- nous programmons en Java
- nous utilisons pour la conception les standards UML
- nous diposons d'un espace de stockage FTP
- nous avons mis en place un Forum pour faire circuler l'information durant les périodes

où l'équipe ne peut se réunir.

Configuration spatiale de la salle :

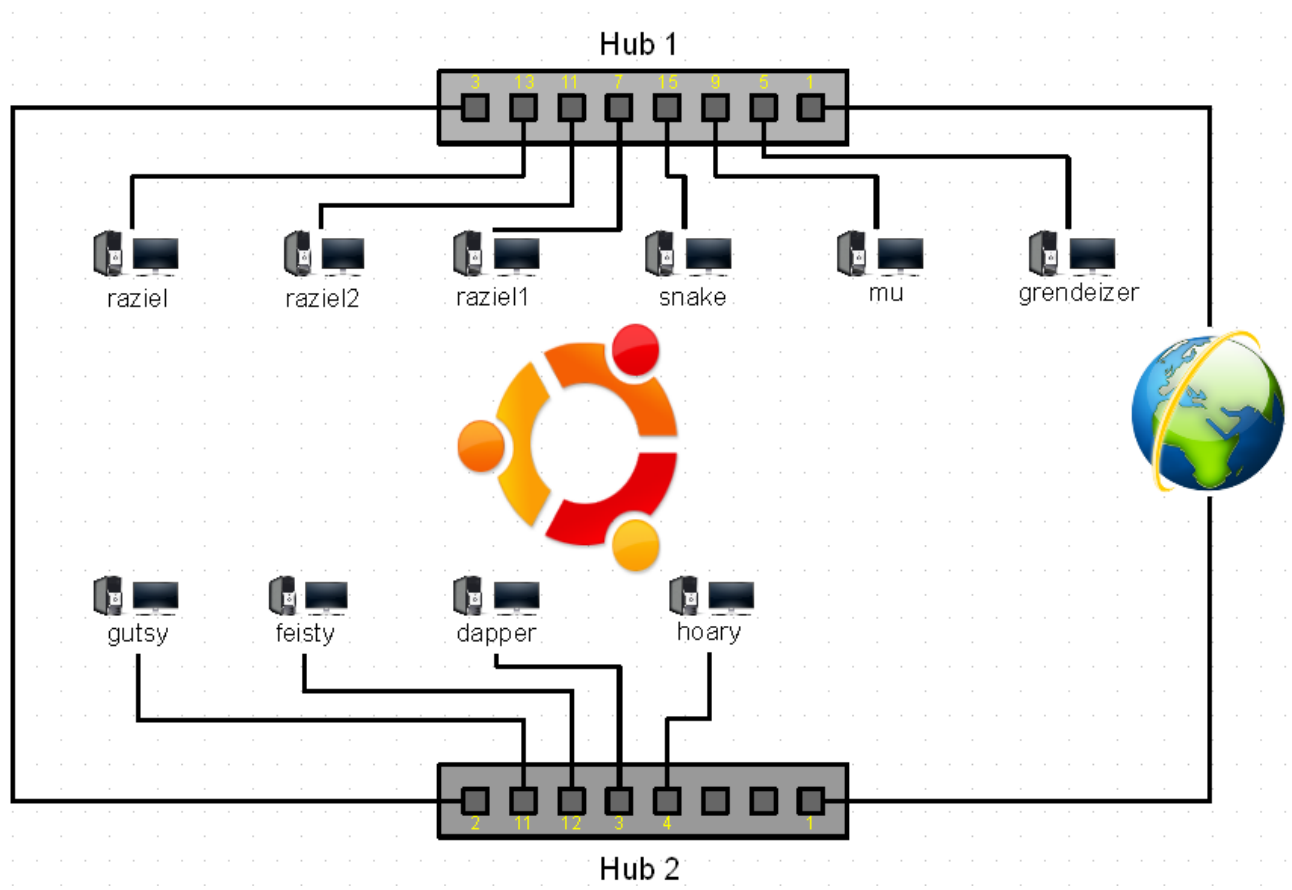


Illustration 2: Organisation du réseau de la salle 204 jaune

DÉMARCHE DE TRAVAIL

Dans cette partie, nous expliquons comment nous avons travaillé selon deux approches : en premier lieu, chronologiquement, c'est à dire quelles ont été les principales phases du développement du projet et comment elles se sont enchainées et en deuxième, au niveau de la méthode, comment s'est réparti le travail au sein de l'équipe.

1. Répartition du travail

1. Organisation des effectifs

Pour développer notre projet, nous avons décidé d'attribuer un rôle à chacun, dans le cadre d'une hiérarchie fonctionnelle. De la cohue générale ambiante de la première séance, deux hommes courageux se sont extirpés pour s'imposer comme leaders charismatiques. Après avoir nominés ces deux flamboyants chefs, nous nous sommes divisés en deux groupes auxquels nous avons décidé de donner :

- Pour l'équipe 1: le travail concernant la conception et la simulation de l'algorithme de traitement des voitures et des feux ; nous la nommons "équipe conception"
- Pour l'équipe 2 : l'élaboration de l'interface graphique par laquelle le client pourrait interagir avec le programme, c'est l'équipe "interface".

Chef de projet : Cédric Bastien
Secrétaire : Pablo Vásquez-Guilhendou

Équipe 1 : Conception	Équipe 2 : Interface
Chef d'équipe : Cécile de Trentinian	Chef d'équipe Nicolas Griedlich
Membres de l'équipe : - Julien Didion - Alexandre Leduc - Jean Philippe Muller - Romain Lannette	Membres de l'équipe : - Laurent Manneville - Jean François Rouyer - Éric Lenepvou - Mohamed Bayoumi

- Guillemenin Seng - Étienne Delecroix	

2. Circulation de l'information

Entre les membres de la section SERTR qui travaillent sur le projet, il a été décidé que nous devions faire circuler l'information comme dans un vrai projet, c'est à dire de manière verticale. Ainsi, l'administrtaion supérieure n'est en communication qu'avec les chefs des équipes et les équipes ne partagent pas non plus d'informations entre elles.

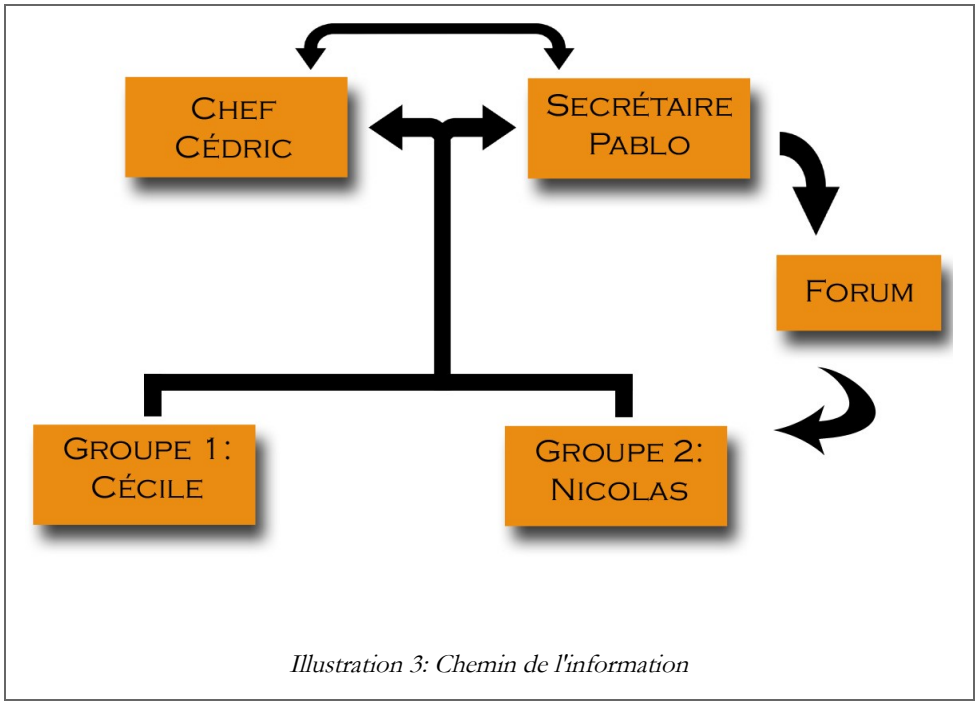


Illustration 3: Chemin de l'information

3. Le forum

Pour pouvoir partager et communiquer facilement, nous avons créé un forum où se centraliseraient les informations dont tout le monde peut avoir besoin. Il comprend une partie publique permettant de garder une certaine transparence quant au client ainsi que des parties privées accessibles à chacune des équipes.

Le Forum est un élément essentiel permettant à chacun de s'informer de l'avancement du travail de son groupe et de signaler aux autres membres et son chef l'avancement des tâches qui lui étaient assignées. On y trouve donc toutes les tâches en cours et le travail à réaliser pour la

prochaine séance, mais aussi les comptes rendus des séances précedentes, entre autres.

Enfin, on y trouve toute la documentation qui peut servir à tout un chacun :

- Comment installer eclipse
- Comment installer les différents plugin dont nous avons besoin (SVN)
- Un dictionnaire de la nomenclature du programme et les diagrammes UML pour être sûr que tout le monde emploie les mêmes noms d'objets, de classes, ...

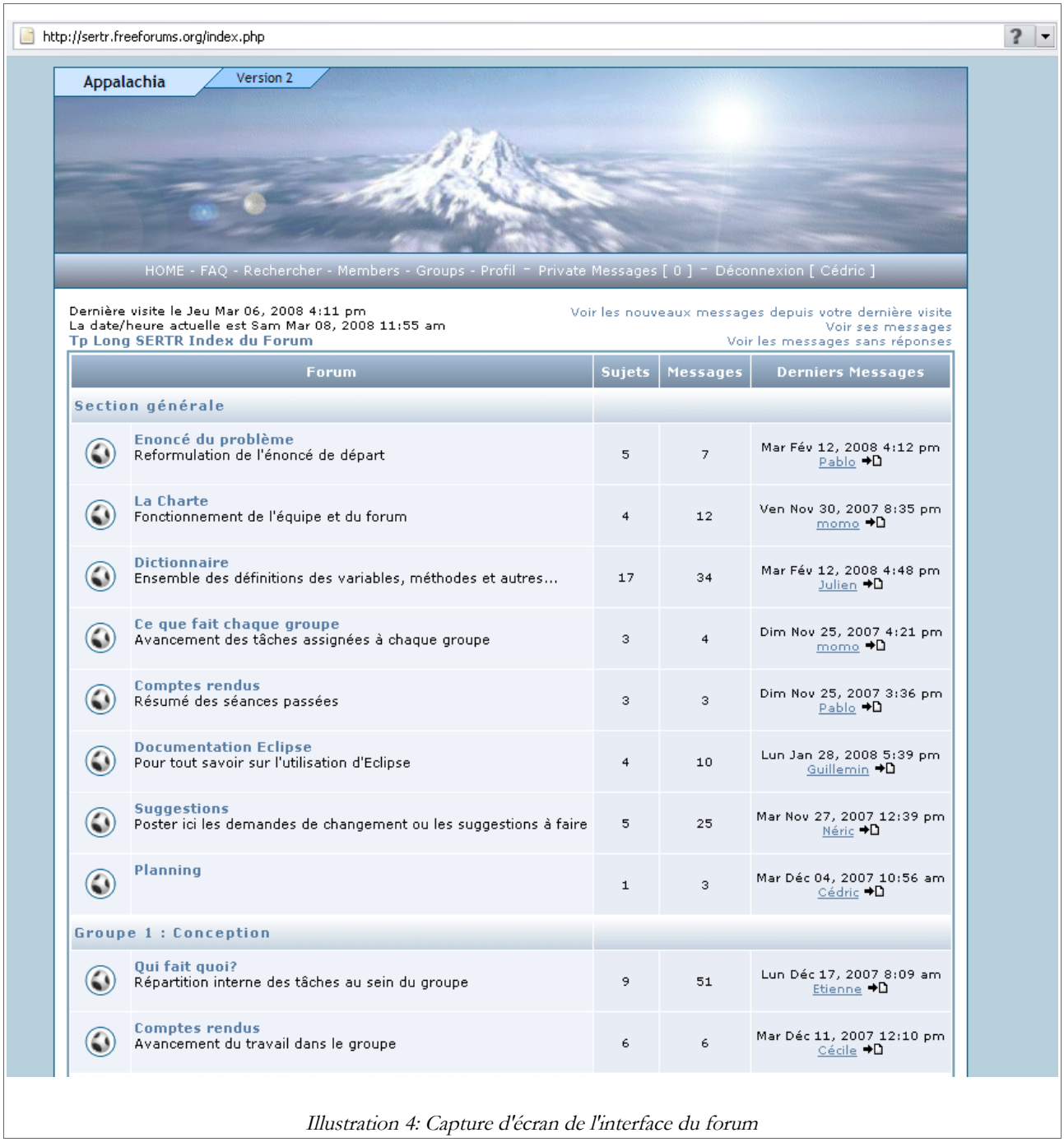


Illustration 4: Capture d'écran de l'interface du forum

2. Organisation de notre temps

1. Choix des rôles

Lors de la première séance, la première chose à faire a été de choisir quelles étaient les deux personnes qui allaient diriger le projet. Suite à celà, nous avons scindé la classe en deux groupes dont l'effectif (7 personnes pour le groupe 1 et 6 pour le groupe 2) semblait correspondre, à première vue, à la quantité de travail inhérente à chaque équipe.

2. Concertation

Nous nous sommes tous entretenus pour analyser le cahier des charges. Nous avons tenté de voir ce que nous évoquait le sujet et de dégager les plus importants problèmes qui pouvaient lui être liés, a priori. L'objectif a été avant tout de se mettre d'accord sur le sujet, pour une bonne compréhension et afin que tout le monde puisse travailler dans la même direction.
C'est aussi au début de ces séances que nous avons désigné les rôles et responsabilités de chacun et que nous avons fait travailler chacun sur des problèmes différents.

3. Proposition de solutions et choix d'une direction

Ce brainstorming nous a offert plusieurs solutions aux problèmes posés, entre autres, comment modéliser de la position des voiture et comment gérer le temps. Après avoir confronté ces solutions, nous avons défini ce que nous attendions en priorité de notre programme et le comportement des objets que nous utiliserions.

Pour traiter le problème nous avons choisi de fonctionner de manière séquentielle avec un Ordonnanceur dans lequel sont placés des évènements datés représentant les occurrences qui nous intéressent dans la ville. Nous avons donc décidé de traiter le problème de manière asynchrone pour diminuer la quantité de cacluls nécessaires à l'exécution de notre programme.

C'est à ce point de développement du projet que nous avons choisi la modélisation finale de la ville.
Elle est composée de différents tronçons, qui représentent des portions de routes entre deux

feux/croisements.
L'existance de cet objet permet de travailler sur le traffic à l'échelle du pâté de maison.
Chaque tronçon est quant à lui composé de cases, qui permettent de situer les voitures les unes par rapport aux autres, de gérer leurs déplacements et aussi d'extraire leurs positions à l'échelle métrique.

4. Description technique UML

Pour partir sur de bonnes bases et clarifier ce que nous exigeons de notre programme, nous avons utilisé le langage de description UML afin de définir l'ensemble des classes et objets que nous intégreront dans le code.
Le but était au moins double : d'une part, nous pouvions être sûr que tout le monde aurait la même description et comprendrait la même chose de ce que nous attendions de chaque objet/classe, d'autre part, nous pouvions générer le squelette de notre programme un peu plus rapidement et sans se soucier des fautes de syntaxe.
Nous avons ainsi construit le diagramme de classes, référence à partir de laquelle nous allions pouvoir programmer.
Le diagramme final est très complexe mais indispensable au développement. Il est en accès libre à tous sur le Forum, et vous pourrez le trouver en annexe A.

5. Programmation

Les deux groupes ont commencé leur travail séparément, et ont chacun codé sans se soucier de ce que faisait l'autre équipe. Leur travail est repris séparément dans la suite de ce rapport quand nous parlerons de la recette en page 23 ainsi que du produit fini.

6. Confrontation et tests

Une fois que les codes ont été prêts à être regroupés (l'interface offrait des petits programmes de test pour que le groupe 1 puisse tester si son code était fonctionnel), les deux équipes se sont rapprochés en s'expliquant mutuellement comment fonctionnaient les processus qui devaient interagir. Après regroupement des deux codes, nous nous sommes rendus compte de l'étendue du travail qu'il nous restait à faire puisqu'un nombre plus que conséquents d'erreurs et bugs en tous genres étaient présents dans notre code lors de son exécution.
À partir de ce moment là, le travail a consisté à revoir tout le code pour comprendre pourquoi

certaines choses ne marchaient pas, et pourquoi le programme pouvait refuser de s'exécuter. L'importance primordiale des listes de classes s'est bien ressentie à cet instant du développement.

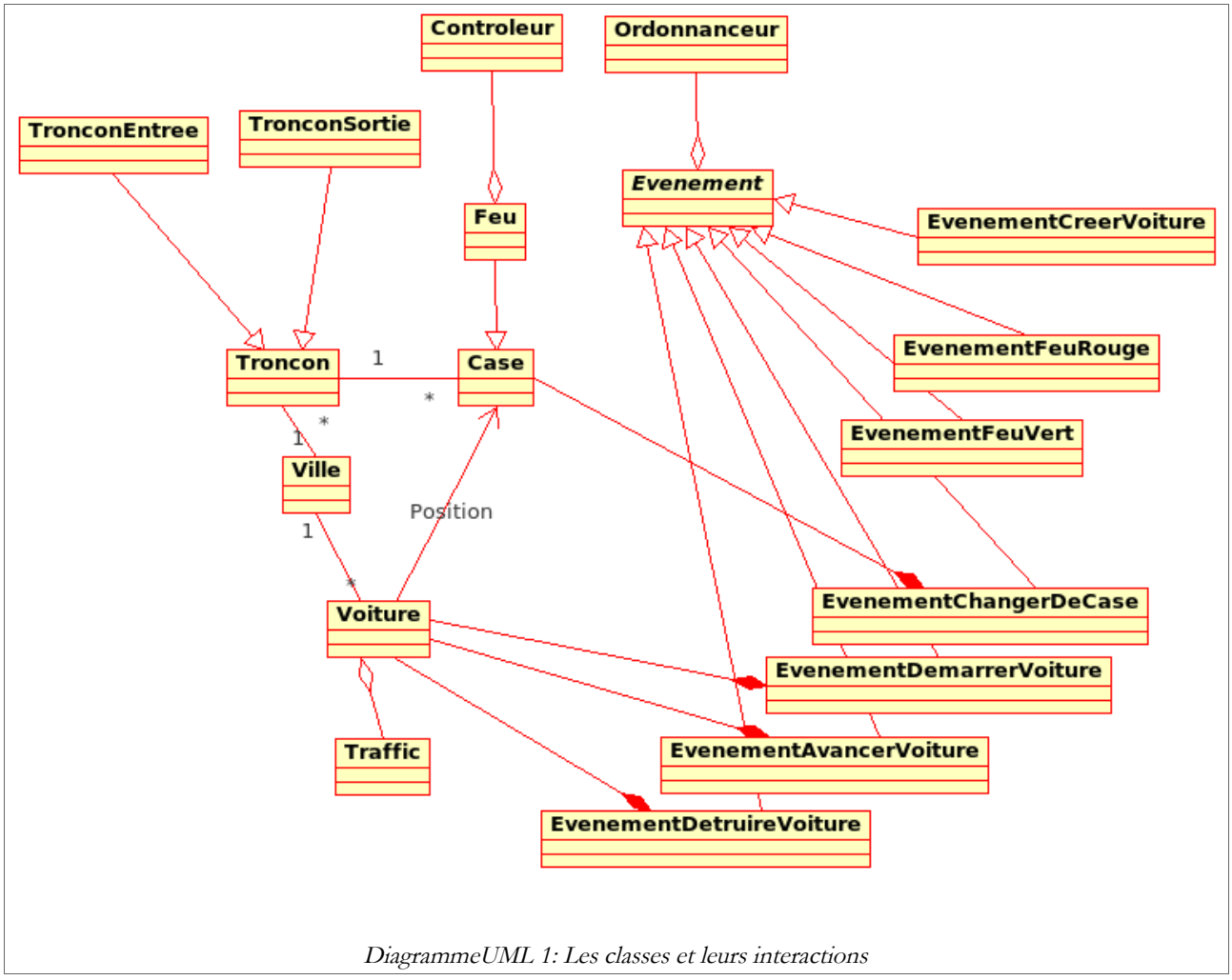
7. Rédaction du rapport et présentation

La rédaction du rapport s'est faite en parallèle avec le développement du logiciel final et le diaporama a été réalisé expressement pour la dernière séance puisque le plus gros du travail a bien évidemment été réalisé dans les derniers instants avant l'évaluation (tout à fait compréhensible vis à vis des standarts du monde du développement informatique).
Une fiche d'évaluation a été concçue pour permettre aux membres des équipes de jauger eux même leur travail, sans oublier quelques lignes d'appréciation du supérieur hierarchique direct, donnant une idée de la place et du rôle du membre au sein de son équipe.

Le produit fini

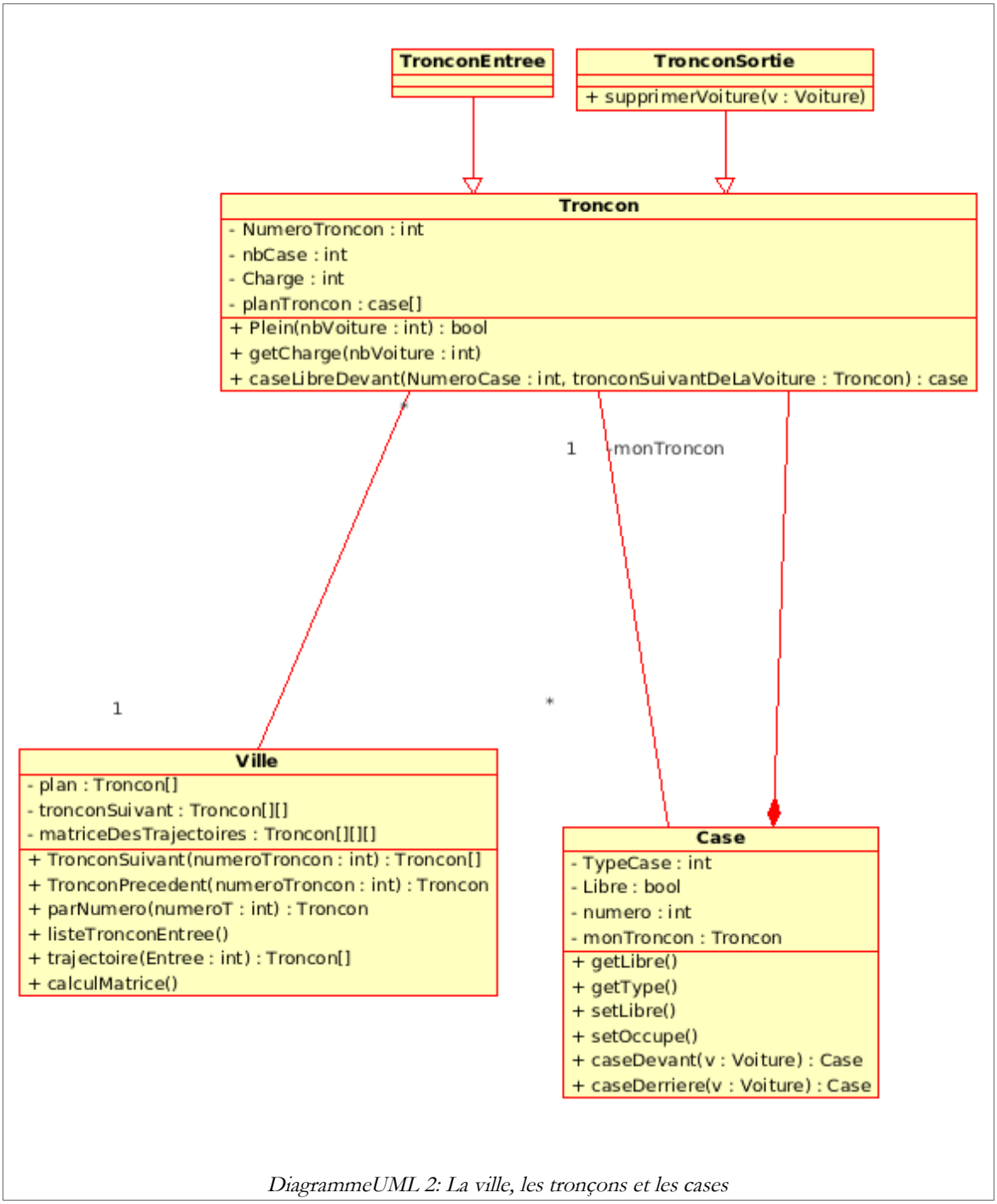
1. Son architecture :

Pour expliquer comment marche notre programme, plutôt que de vous présenter le code et de le commenter, nous allons vous fournir une série de diagrammes UMLafin de vous présenter les différents objets utilisés dans notre programme, leurs méthodes principales et tout ce qu'il faut en connaître pour avoir une bonne vision du programme. Rappelons une fois de plus que le digramme complet est présent en annexe A, ainsi que la liste des classes détaillées en annexes B et C.



DiagrammeUML 1: Les classes et leurs interactions

1. La carte (ville, tronçons, cases)



DiagrammeUML 2: La ville, les tronçons et les cases

La ville s'organise s'organise de la manière suivante :

1. On crée une instance de l'objet ville qui contiendra un tableau de tronçons.
2. On détermine comment sont organisés spatialement les tronçons en les rangeant avec leurs numéros dans le tableau *tronconSuivant*. Cela se fait en considérant que la colonne correspond au numéro de l'antécedent et les numéros de lignes correspondent aux numéros des suivants.
3. On crée les cases dans chaque tronçon.

La ville finale aura la configuration suivante :

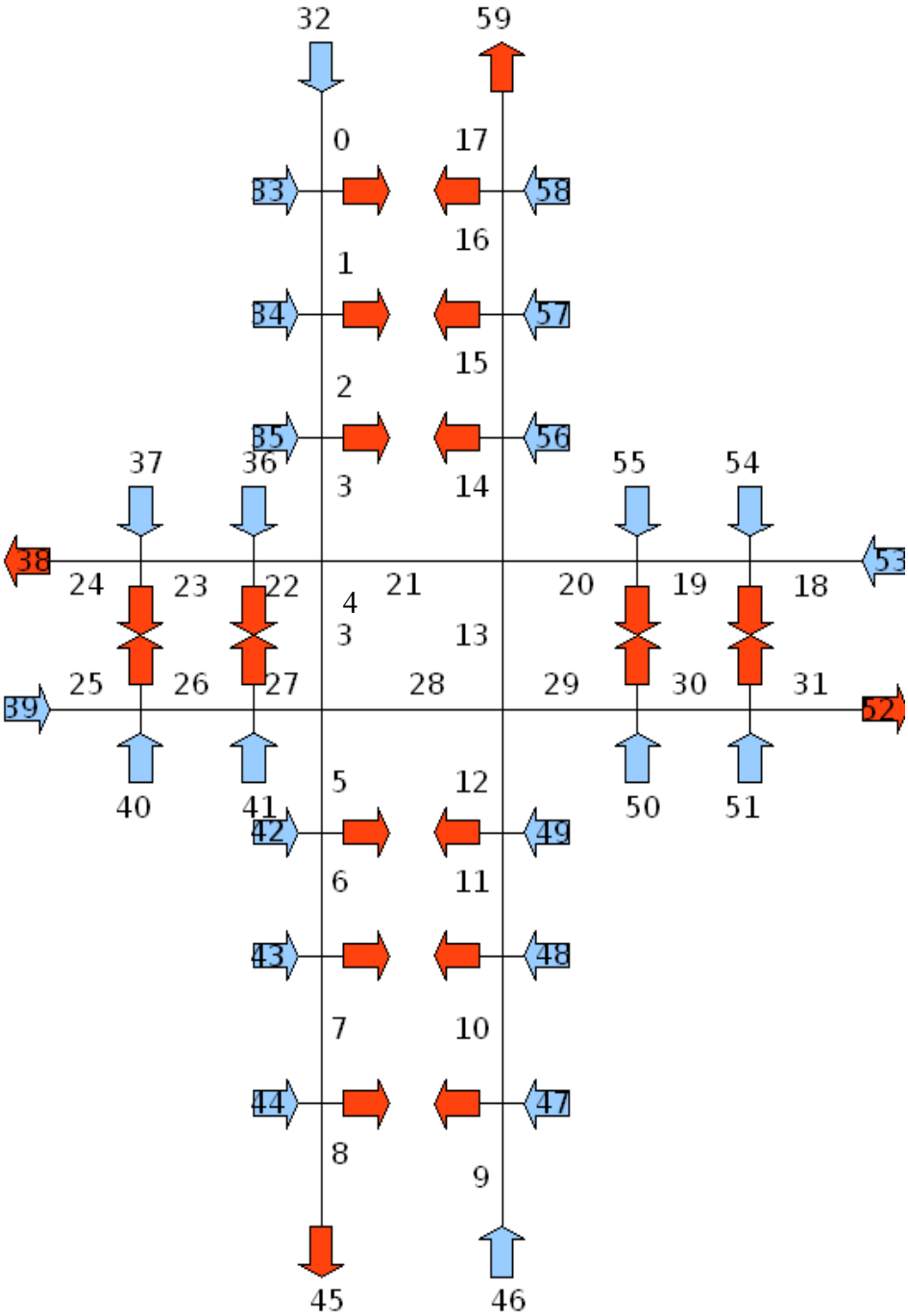


Illustration 5: Schéma de la ville avec les numéros des tronçons

Il y a différents types de tronçons, les tronçons normaux en bleu, les tronçons d'entrée en vert et les tronçons de sortie en rouge.
Ces différents types de tronçons ont l'utilité suivante :

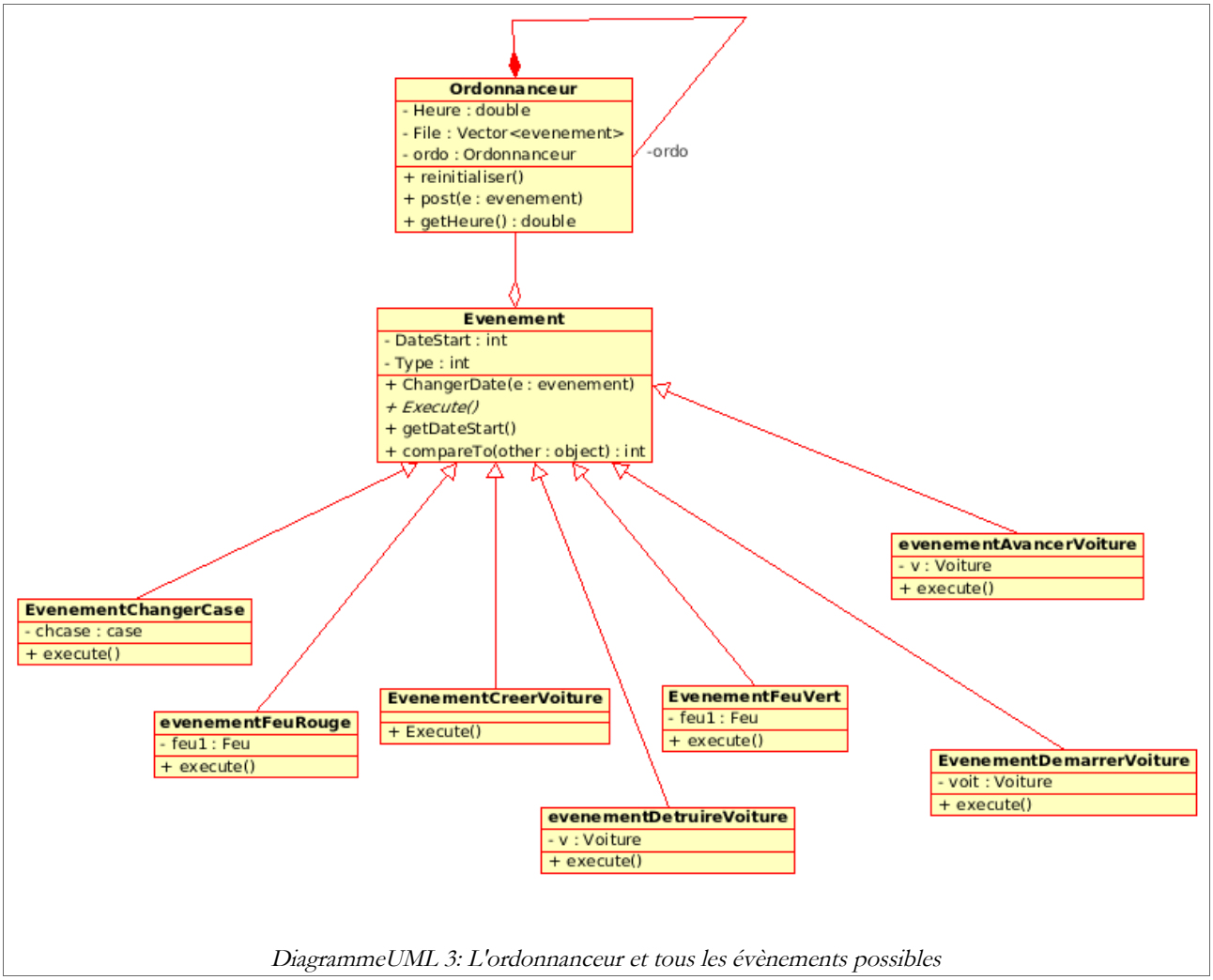
- Sur les tronçons d'entrée, les voitures sont créées avec une vitesse et des paramètres propres : on définit un type de conducteur (nerveux, passif, personne âgée, ...) et en fonction de ça, on a une plage de valeurs pour les paramètres aléatoires qui vont être générés. Ces paramètres serviront à définir son accélération et sa vitesse nominale.
- Sur les tronçons de sortie, les voitures sont supprimées de la ville (on leur donne une date de décès, mais on en conserve la trace dans l'historique pour les statistiques). Dès qu'elles atteignent la dernière case elle sont déplacées dans le tronçon numéros 60, sur lequel débouchent toutes les sorties.
- Sur les tronçons normaux, les voitures se déplacent de manière tout à fait normale.

On peut noter ici qu'il existe un certain nombre de méthodes nécessaires au déplacement des voitures sur les cases, notamment celle qui calcule comment trouver la trajectoire d'une voiture, c'est à dire savoir dans quelle direction elle va tourner à chaque carrefour.
Il existe un objet *matriceDesTrajectoires[][][]* qui est un tableau à trois dimensions, dans lequel on lira en lignes le numéro du tronçons d'entrée dans la ville, en colonnes le numéro du tronçon de sortie et en profondeur la succession des numéros des tronçons nécessaires à parcourir pour atteindre ce chemin.
Quand une voiture est créée, elle va choisir son tronçon d'entrée en fonction des probabilités données par les slides de densité de création, et donc la ligne de sa matrice, la colonne sera elle déterminée aléatoirement de manière équiprobable. Ainsi, à chaque fois qu'une voiture doit changer de tronçon, vu qu'elle connaît les coordonnées de la case dans la matrice, elle n'aura plus qu'à regarder le vecteur dans cette case pour connaître l'ordre des tronçons qu'elle va devoir emprunter.

On a aussi plusieurs méthodes qui permettent de retrouver certaines cases, certains tronçons par rapport à d'autres, lorsque l'on cherche à savoir jusqu'où une voiture pourra avancer.

Enfin, l'objet case contient deux attributs primordiaux :

1. Le boolean *libre* qui indique si une voiture est présente ou non sur cette case
2. L'entier *type* qui indique s'il s'agit d'une case spéciale ou non. Si une voiture arrive sur une case qui n'est pas de type 0, alors elle devra regarder ce qu'elle fait. Les différents types sont :
    - 0 : Case normale
    - 1 : Case feu
    - 2 : Case virage



DiagrammeUML 3: L'ordonnanceur et tous les évènements possibles

L'ordonnanceur est un objet qu'on instancie qu'une seule fois. Il va être composé d'un tableau d'objets de type *Evenement* qui correspondront à toutes les choses à éxécuter dans le futur. Notre programme se veut donc asynchrone, et chaque évènement est associé à la date à laquelle il doit être exécuté, sans qu'on ait tout à regénerer à chaque fois.

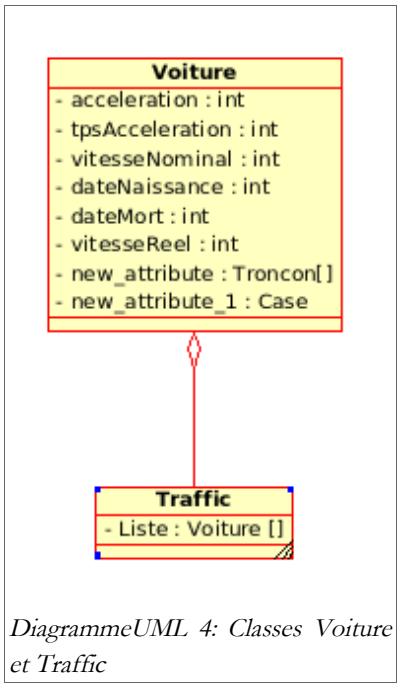
La méthode *execute* fait appelle aux méthodes des sous évenements associés qui exécuteront chacun une fonciton propre :

- *ChangerCase* : celà déplace la voiture, c'est à dire que la voiture va scruter toutes les cases libres jusqu'où elle peut aller (c'est à dire jusqu'à la prochaine case libre, ou d'un type qui indique qu'il faut regarder ce qu'il se passe : un carrefour ou un feu, par exemple). Elle va ensuite considérer être sur toutes ces cases à la fois (elle met leur attribut *libre* à *false*) et va ranger dans l'ordonnanceur des évènements correspondant aux moments où chacune de ces cases va être libérée. À la fin, on ajoute le prochain évenement *ChangerCase* qui correspond au moment où on devrait libérer la dernière case pour que la voiture regarde à

nouveau là où elle peut se déplacer.

- FeuVert et FeuRouge : on indique la prochaine date à laquelle un feu va changer de couleur. Dès que celà arrive on range à nouveau dans l'ordonnanceur la prochaine date de changement de couleur de feu.
- CréerVoiture : on génère une voiture avec des caractéristiques aléatoire propres. Comme on a défini plusieurs types de comportements (agressif, lent, ...), on génere déjà le comportement qui offre une plage de valeurs aléatoires qu'on va à nouveau tirer au sort. On range ensuite dans l'ordonnanceur le prochain évènement CréerVoiture daté grâce à une loi de poisson. Il faut savoir que la voiture sera générée à une position qui dépend des probabilités données par les curseurs de création de voitures. Ainsi, par exemple, si les 4 premiers curseurs sont à 10 et le dernier curseur est à 60, la voiture aura 10% de chances d'être créée à un des 4 premiers tronçons et 60% au dernier. On place la voiture créée au plus près du feu d'un TronconEntree (on vérifie bien sûr qu'il n'est pas déjà plein) et on range dans l'ordonnanceur un AvancerVoiture pour que la voiture puisse se déplacer par la suite.
- DetruireVoiture : on supprime la voiture de la ville en libérant les dernières cases et on met toutes les statistiques à jour.
- *DemarrerVoiture* : on fait appel à la classe *AvancerVoiture* si cela est possible, sinon on reposte un évènement *DémarrerVoiture* plus tard dans l'ordonnanceur.

3. Les voitures

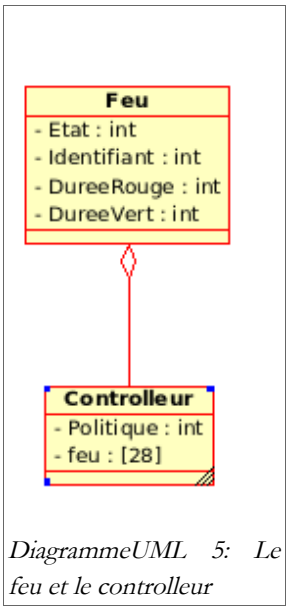


DiagrammeUML 4: Classes Voiture et Traffic

L'unique instance de l'objet *Traffic* contient un tableau de toutes les voitures qui ont été créées. Celà permet à la fois d'avoir un historique utile à toutes les statistiques qu'on désire calculer et afficher, mais aussi d'avoir un accès rapide à toutes les voitures d'après leurs identifiants.

Lorsque l'on crée une voiture, comme expliqué précèdement, on va déterminer tous ses paramètres de conduite selon un comportement aléatoire appartenant à une certaine plage, suivant des comportements types.

4. Les feux et leur contrôleur



DiagrammeUML 5: Le feu et le controlleur

Le contrôleur est un tableau de feux : il référence tous les feux de notre ville. On fait appel à une de ses méthodes *politique* pour changer de politique, et à ce moment là, il va changer toutes les durées de tous les feux suivant la politique choisie.

2. L'interface :

La réalisation de l'interface s'est déroulée en plusieurs étapes :

- recherche d'idées, brainstorming et prosition de quelque-chose de concret par l'équipe interface
- retouche par les chefs de projets pour proposer quelque-chose qui nous semblait plus

correspondre aux attentes du client

- concertation avec celui-ci pour voir si cela lui correspondait ; c'est à ce moment que nous avons pris en compte toutes les modifications qu'il désirait
- validation d'une interface finale pour le client au travers du contrat client que vous pourrez retrouver en pièce jointe à ce document.

Voici, repris du contrat client, les élèments essentiels de l'interface :

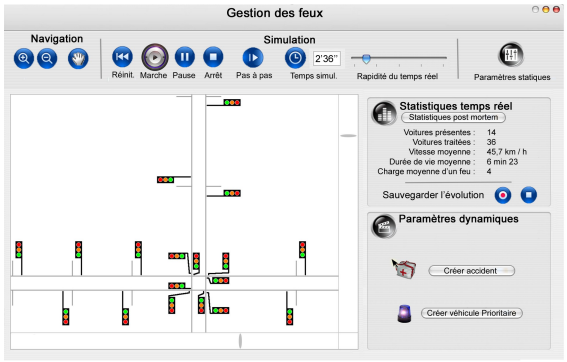


Fig1. Statistiques temps réel

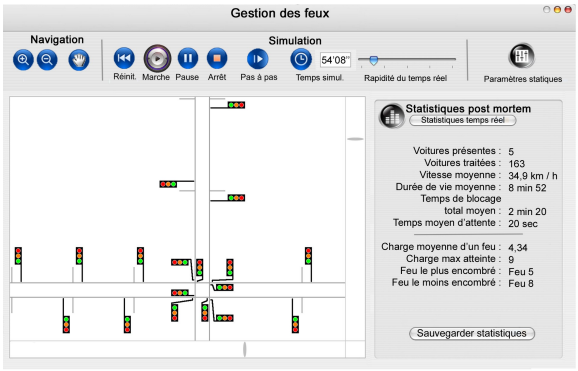


Fig2. Statistiques post-mortem

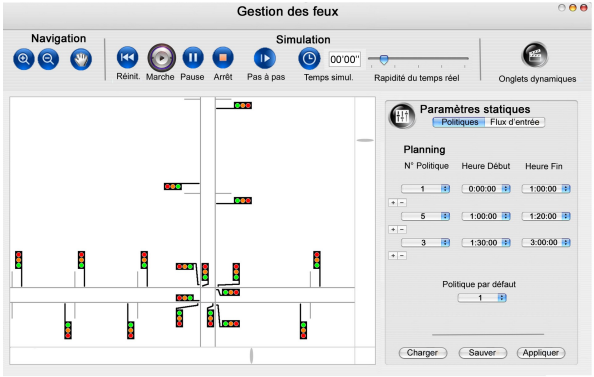


Fig3. Flux d'entrée

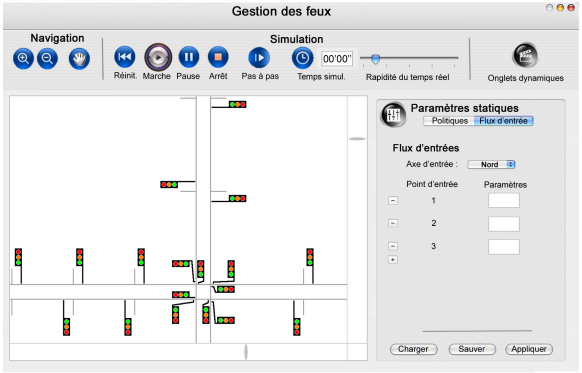


Fig4. Politiques

Dans la version finale de l'interface que nous vous proposons, nous retrouvons les points clés de cette présentation, avec toutefois quelques changements mineurs notables. Vous trouverez la liste des ces changements dans les parties suivantes où nous expliqu'ons quels choix nous avons fait et pourquoi.

Vous trouverez ci dessous une capture de l'interface actuelle, telle qu'elle est présentée dans la

version finale de notre projet :

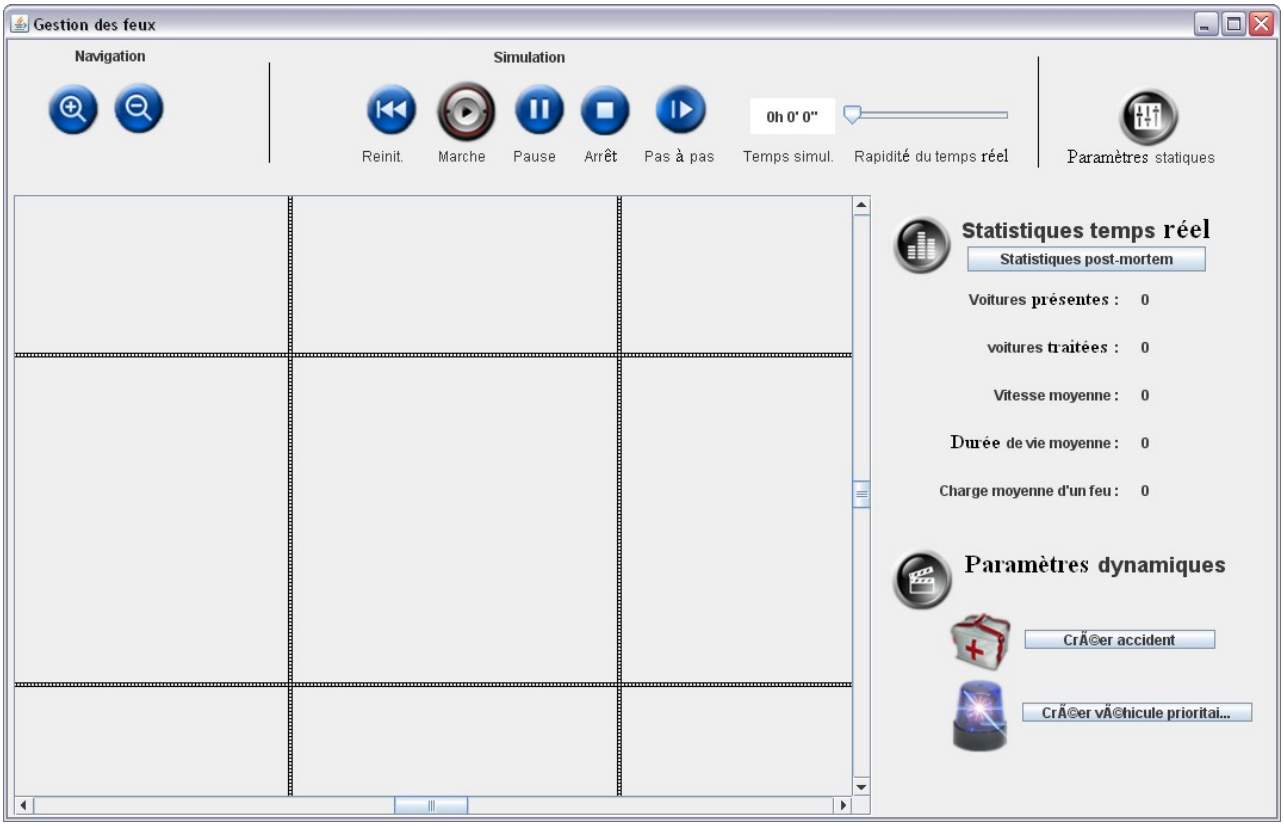


Illustration 6: Version finale de l'interface

La justification des choix

Dans cette partie, nous allons vous expliquer comment ont été décidé les choix techniques, c'est à dire selon quels critères. Nous verrons dans un premier temps la partie conception et ses choix algorithmiques, et ensuite la partie interface qui présentera plus des choix d'esthétisme.

1. Les choix dans la conception

1. La ville

La discrétisation spatialement

Nous avons décidé de discretiser l'espace de notre ville en cases, que la voiture occuperait, car en première approche cela nous a semblé plus simple à la fois pour afficher les voitures, et aussi pour pouvoir les déplacer. En effet, nous n'avions pensé qu'à la solution qui consiste à regarder le plus loin devant soi et poster l'évenement "je vais jusque là" dans l'ordonnanceur, mais entre temps, il faut prendre et libérer toutes les cases entre le point de départ et le point d'arrivée. Avec le système des cases, cela nous semblait facile, alors que si nous nous étions contenté de mettre à jour des positions pour les voitures, il aurait déjà fallu aller les chercher dans des objets de type files d'attentes FIFO (qui représentent nos tronçons actuels, pour pouvoir savoir quelle voiture est devant nous) et il aurait ensuite fallu comparer les positions de nos deux voitures.

La solution a donc été retenue, grâce à cet aspect plus simpliste, et peut être plus proche que ce que nous avons l'habitude de voir dans des jeux ou des simulateurs, où tout est toujours discrétisé.

Spécification de trois villes différentes

Les choix concernant la distance entre les feux et largeur de la route ont été faits pour simplifier le projet, notamment pour les carrefours. On a donc choisi de faire des carrefours carrés.

Le choix de trois villes a été faites pour progresser dans le projet à l'aide d'étapes successives. La

première ville est donc constituée de simples troncons, sans virages, et au départ sans feux. On ajoute par la suite les feux, puis on passe à la seconde ville qui intègre cette fois un virage. La dernière est la ville complète, qui sera spécifiée plus tard, après plusieurs changements qui l'ont considérablement modifiée au cours du projet.

Les carrefours

Problème des carrefours : comment suivre une voiture si elle change de tronçon? Si on a juste quatre axes on retombe sur le même probleme avec les croisements.

Une longue réflexion a été faite concernant le choix de l'objet qui constiturait les routes. Nous pouvions soit avoir quatre axes, qui se croiseraient aux carrefours, des axes à sens unique. Soit avoir des tronçons de routes, qui seraient délimités par les feux, et qui se chevaucheraient sur les carrefours. Dans les deux cas, il existe un point épineux au niveau des carrefours, car si la voiture change d'axe ou de troncon, il faut pouvoir la suivre, et elle occupe à un moment à la fois ce qui se trouve devant elle et se qui se trouve derrière. Le système de tronçons a été retenu car il permettait de reproduire pour chaque zone entre feux la même schéma. Les axes auraient été plus simples à implémenter mais par la suite plus complexes à utiliser, car il aurait fallu savoir quel niveau elles se situent, à quel feu, ...

Spécifications des cases

Vient ensuite le problème des cases qui composent chaque tronçon. Nous savions qu'une voiture fait quatre mètres de long (et nous avons aussi choisi quatre mètres de large, pour que tout aille bien au niveau des carrefour). Nous avons donc choisi des cases carrées. En évaluant la vitesse des voitures entre 50km/h et 90km/h, il fallait que l'on puisse observer leur mouvement dans la ville assez précisément. Pour simplifier toute la suite, comme les cases sont d'un ordre inférieur au mètre et que le temps s'écoule en secondes, on choisit d'exprimer la vitesse des voitures en mètres par secondes. Pour déterminer la taille idéale des cases il faut prendre le cas le plus défavorable soit celui qui nous oblige à prendre le plus de cases, ou bien le moins de case. Ile problème à résoudre est si deux voitures qui se suivent ont des vitesses différentes mais doivent être les plus proches possibles. Si on considère qu'une voiture A avance à V1, et une voiture B à une vitesse V2 supérieure à V1 d'un mètre/seconde, et que l'on prend la seconde comme unité de temps, alors B doit rattraper A d'un mètre toutes les secondes. Si la précision de la vitesse est le mètre, alors des cases d'un mètre suffisent. Avec le même raisonnement, si la précision augmente, la taille des cases diminue. En considérant que la vitesse serait plus précise, et que le temps serait d'un ordr eplus court que la seconde, il est apparu que les cases devraient faire 50 centimètres pour bien modéliser les voitures.

Une voiture fait donc 8 cases, qu'elle occupe toutes, mais le choix est fait de déterminer sa position par la première cases occupée uniquement. Ainsi, on choisit pour avancer dans la ville non pas de tester où sont les voitures, mais plutôt si les cases sont occupées ou non. De même

pour les feux, ce sont les cases qui nous dirons si un feu est présent ou non. On ramène comme cela tout aux cases, au lieu de considérer chaque feu et chaque voiture. Avec cette conception, il a donc fallu choisir un "type" de case, selon si l'on est sur une case feu, une case occupée ou libre (par une voiture) ou bien si l'on se trouve sur une case virage (indiquant que l'on entre sur un carrefour, donc sur une zone où deux tronçons se chevauchent).

2. Les voitures

Les contraintes clients

Plusieurs spécifications concernant les voitures n'ont pas fait l'objet de recherches car elles nous ont été imposées par le client. L'accélération des voitures par exemple. Pour plus de simplicté, nous avons choisi de ne pas mettre d'espacement entre les voitures, et de ne modéliser qu'une case sur la largeur de la route. En effet, par la suite nous devons inclure les véhicule prioritaires qui pourront doubler les voitures mais comme ceux -ci sont des véhicules que l'on suppose rares, il serait inutile de créer des "cases de dépassement" pour chaque route.

Réflexion sur le réveil des voitures

Lorsqu'une voiture est arrêtée, soit derrière un feu soit derrière une voiture elle aussi à l'arrêt, elle doit savoir quand se réveiller pour démarrer. Deux solutions s'offrent à nous : elles se réveillent de manière autonome ou bien leur réveil est provoqué par l'objet les précédent (feu ou voiture).

Au début du projet, avant le début du codage, il avait été décidé de gérer ce réveil de façon autonome. Chaque voiture arrêtée derrière une voiture ou un feu rouge regardait dans l'ordonnanceur l'évenement associé à la case de devant qui l'empêchait d'avancer (voiture ou feu). Ainsi elle pouvait donc savoir à quelle date elle pourrait redémarrer.. Ce choix a été fait car des classes toutes faites permettaient de coder apriori très simplement un "appel à démarrer".

En commençant à coder cette partie, il nous a paru intéressant et plus pratique d'effectuer une gestion généralisée des réveils en utilisant l'implémentation des observables. Chaque feu serait un *observable* et aurait comme *observer* toutes les voitures qui entreraient dans son troncon. Il réveillerait donc tous ses *observer* dès qu'il passerait au vert.

Cependant ce choix a été abandonné par la suite car il etait compliqué d'assigner des *observer* à une classe *observable* (a savoir la classe evenement) : en effet les observer ne sont jamais les mêmes pour un feu donné, ils changent au cours du temps...

Finalement il a été décidé de revenir à la première solution, de façon encore plus simplifiée. Chaque voiture devient autonome mais gère son réveil de façon passive. Chaque fois qu'elle est à l'arrêt, elle poste dans l'ordonnanceur un évènement AvancerVoiture.

C'est l'exécution de cet évènement qui regarde dans le parcours de la voiture si celle-ci peut

avancer ou non. Si elle ne peut pas, cet évènement poste alors un nouvel évènement AvancerVoiture toutes les 1 seconde (ce qui est un temps acceptable quant à la configuration de la ville) jusqu'à ce que la voiture puisse avancer.

Les politiques associées

Nous avons réflechi aux diverses politiques que l'on pouvait mettre en place dans la ville, recherché des solutions à l'engorgement des véhicules et trouvé des politiques dépendant du traffic. Le cahier de charges nous demandait notament la mise en place d'ondes vertes à 47km/h.

Nous avons rapidement conclu que les feux d'une même route en dehors du carrefour devaient avoir le même fonctionnement, décalé au temps de roulement entre feux près.

La mise en place d'une onde verte implique entre deux feux distants :

- de 50 mètres : un écart de fonctionnement de 3,8 secondes
- de 200 mètres : un écart de fonctionnement de 15,3 secondes
- de 300 mètres : un écart de fonctionnement de 23 secondes

Nous avons décidé de garder le feu vert pour une durée telle que le carrefour ne s'engorge pas si de trop nombreuses voitures tournent. Ensuite nous avons décidé d'utiliser la capacité des feux à compter les voitures devant eux et de changer de mode de fonctionnement quand il y a neuf voitures devant les feux du centre.

Enfin, nous avons pensé que chaque feu pourrait compter le nombre de voitures devant lui et de passer au vert quand il y a plus de voitures devant lui que devant l'autre feu associé au carrefour. Mais il n'y a alors pas de garantie d'onde verte. Il faut alors utiliser une communication entre les feux pour suivre les ondes vertes. Et le feu qui a la plus grande envie de passer au vert dirige l'onde verte.

Les détails concernant les durées de feu vert ou de feu rouge ne sont pas déterminées tout de suite, car ils dépendent de la circulation des voitures dans la ville. Il faudra donc faire un paramétrage à la toute fin, lorsque les voitures s'afficheront.

3. Les évènements

Gestion du temps synchrone vs gestion par évènements

Examen des deux solutions :

Politique de gestion du temps par évènements :

Pour appliquer cette politique, il faut prendre en compte tous les évènements qui vont arriver (prédictibles donc) et les ordonnancer dans l'ordre croissant de leur date d'éxécution.

Ensuite l'évenement en tête de liste est éxécuté puis supprimé et on recommence pour pouvoir prendre en compte d'éventuels nouveaux évènements qui n'existaient pas auparavant (création d'une nouvelle voiture par exemple). S'il n'y a pas de nouveaux élements, il suffit de poursuivre l'éxécution des évènements dans l'ordre établit.

Ainsi à chaque exécution d'un événement, on met à jour la valeur du temps :

temps = temps + Date d'exécution(évènement précédent).

Exemple : *T*=53 : (Feu n°2 ROUGE =>VERT à T=54s) / (Suppression Voiture n°36 à T=56) / ... / ...

T=*T+1 =54 : Le feu n°2 passe au vert*

T=54 : (Suppression Voiture n°36 à T=56s) / ... / ... / ...

Politique de gestion du temps par discrétisation :

Cette politique consiste à établir une discrétisation constante du temps. Il faut pour cela que cette discrétisation soit suffisamment courte pour que le système soit fonctionnel.

Ainsi à chaque période il faut recalculer tous les paramètres du système (positions des véhicules) et prendre de nouvelles décisions (vitesse des voitures, états des feux de signalisation).

Conclusion et déduction sur la spécification

Le type de gestion du temps par discrétisation serait efficace si le système était soumis à une charge plus ou moins constante et suffisamment importante. En effet dans ce cas, la notion de périodicité serait justifiée.

Dans notre cas, la charge du système varie avec le temps. (suivant que l'on se trouve dans les heures de pointes ou non) Les heures de pointes (environ 7-9h / 12-14h / 17-20h) ne représentent qu'un peu moins du tiers de la journée. Un système de gestion par événement

serait plus approprié et permettrai d'assouplir la charge de calculs en fonction du trafic, que ce soit à ces heures de pointe ou en période normale. C'est donc cette solution qui a été choisie.

Par la suite, lorsque nous avons rassemblé le travail des deux équipes, notre gestion du temps est restée la même mais a été limitée par l'horloge du groupe Interface. Tous les évènements devaient s'éxecuter si leur date d'éxecution est inférieure à la date de l'interface.

2. Les choix dans l'interface

L'interface graphique a été divisée en plusieurs panneaux. Ces panneaux sont le résultat d'une concertation collective puis d'une validation du client. Nous allons voir en trois points le pourquoi de ces panneaux.

1. La barre d'outils

Nous avons voulu l'interface la plus intuitive et la plus ergonomique possible. Ainsi, nous avons privilégié une interface comprenant des boutons indiquant d'eux-mêmes, de par leurs formes et de par leur graphisme, leur utilité, plutôt qu'une interface comprenant des menus déroulants telles qu'on peut en voir dans des application comme Word ou Excel. Ceci a été possible par le faible nombre de fonctions générales (tel que le zoom, ou l'avance des voitures) auxquelles l'utilisateur peut accéder. Si ce nombre de fonctions avait été plus grand, il nous aurait fallu implanter beaucoup d'icônes sur le panneau du haut, et cela l'aurait surchargé et rendu illisible.

Étant donné que la zone d'affichage de la carte est carrée, et qu'elle ne couvre pas la largeur totale de l'écran, nous avons pu utiliser la place restante sur l'écran pour afficher divers panneaux, tels que ceux des *Statistiques temps réel*, *Paramètres dynamiques*, *Statistiques post-mortem* et *Paramètres statiques* (flux d'entrée et politiques). L'accès à ces panneaux se fait grâce un simple bouton sur le panneau du haut. D'un simple clic, on accède aux panneaux concernant les statistiques, ou aux panneaux concernant les données que l'utilisateur peut entrer.

Afin que l'utilisateur n'entre pas de données incohérentes dans les flux d'entrée, nous avons préféré implanter un système de sliders à une entrée manuelle de chiffres.

De même, le système de programmation des politiques laisse un choix vaste, mais restreint.

L'utilisateur choisit les politiques dans celles qu'on lui propose dans le menu déroulant, et ne peut donc pas se tromper. Néanmoins, il faut savoir que dans le code du programme cela ne change pas grand chose puisque les politiques n'ont pas été implémentées, et en réalité chaque choix revient à choisir la même politique.

2. La carte

La carte est carrée. De ce fait, nous avons implémenté un panneau carré. Il devient ainsi plus facile de gérer le zoom et le déplacement de la carte avec la main.

Nous avons choisi de mettre en place un zoom sur seulement trois niveaux.(x1, x2, x3). En effet, nous n'avons pas jugé pertinent d'implémenter un zoom progressif qui croît indéfiniment, car à partir d'un certain niveau, le panneau qui contient la carte n'est de toute façon plus adapté pour l'affichage de celle-ci. D'autre part, on aurait constaté une certaine lourdeur en terme de performance pour notre application et son utilité relative. C'est donc pour cette raison que nous nous sommes dirigés vers cette solution.

Nous avons également facilité l'accès à la main qui permet de bouger la partie visible de la ville. A l'origine, la fonction main était accessible via un bouton ad hoc dans la barre d'outils sur lequel l'utilisateur devait cliquer pour que le curseur de la souris se transforme en une main, mais nous avons finalement opté pour une mise en fonction automatique. Ainsi, dés que le curseur de la souris survole la carte de la ville, il change d'apparence pour prendre celle de la main. Cette fonctionnalité est ainsi directement accessible et son utilisation plus intuitive. Il suffit alors de garder le bouton gauche de la souris pressé pour pouvoir bouger la carte et se rendre vers la partie de la carte qui intéresse l'utilisateur.

En plus de cette fonctionnalité, la carte est munie de barres de défilement, en bas et à droite, permettant à l'utilisateur d'accéder plus rapidement à ce qu'il veut voir et aussi de pouvoir se repérer et connaître sa position. Pour garder une certaine logique, les boutons de défilement de chaque barre dépendent de la position de la main. Ainsi, lorsque l'utilisateur utilise la main, les curseurs se placent automatiquement à la position attendue.

Par défaut, à l'initialisation de la carte (donc au lancement de l'application), la carte est automatiquement centrée ce qui permet de voir directement le centre de la ville sans avoir à le chercher en déplaçant la main. Nous avons fait en sorte que lorsque l'utilisateur clique sur le zoom, la carte zoomée soit également centrée (les barres de défilement se placent automatiquement au centre). De même lorsqu'on dézoome.

3. L'affichage des informations

L'affichage des informations se fait à côté de la carte de la ville. Ainsi, l'utilisateur accède facilement aux différentes informations rescencées par l'application. Les différents informations disponibles ont été sélectionnées à la suite d'une réflexion cherchant à trouver des statistiques pertinentes à montrer à l'utilisateur, cela bien sûr après suggestion au client qui a validé nos choix.

AVANCEMENT DU PROJET

1. Recette :

Cette partie reprend tout ce que nous avions prévu de faire et pourquoi il n'a pas été implémenté.

Dans un premier temps, voici au niveau de l'équipe conception (dont le travail est limitant, puisque si un module ne fonctionne pas, c'est tout le programme qui refusera de s'exécuter) l'état actuel des différents élèments qui devaient être intégrés :

Intitulé	État	Explication
Créer une route	OK	Route créée
Créer route avec virage	En cours	Pas d'erreur lors de la création, tests réalisés, mais sans le projet complet
Créer la ville	En cours	Pas d'erreur lors de la création, tests réalisés, mais sans le projet complet
Créer les voitures (loi de poisson)	OK	Voitures créées aux tronçons d'entrée avec un paramètre d'arrivée donné par l'équipe 2
Choisir une trajectoire	OK	Trajectoire entrée à la main au lieu d'utilisation de la méthode *trajet*
Faire avancer les voitures	En cours	Les voitures avancent dans la première ville, jusqu'au dernier tronçon de leur trajectoire
Détruire les voitures	OK	Les voitures sont bien détruites.
Stocker les voitures	En cours	Tableau des voitures créées avec leurs attributs (dates, position et état)
Gérer les politiques	OK	Quatre politiques implémentées, mais non paramétrées, fonctionnent mais ne sont pas utilisées
Gérer les évènements et le temps	OK	Tous les évènements sont traités et le temps s'écoule suivant les appels de l'équipe 2
Changer l'état des feux	OK	Les feux changent d'état

Changer l'état des cases	OK	Les cases ont le comportement attendu
Fonctionnement général	En cours	On peut voir l'évolution des voitures sur la première ville jusqu'au dernier tronçon de leur trajectoire, il manque encore leur destruction

Voici maintenant la recette d'un point de vue un peu plus global, présentable vraiment au client et intégrant les élèments du contrat client :

Prévu initialement	État actuel
RMI, application répartie	Deux personnes y ont particulièrement réfléchi pour pouvoir appréhender le fonctionnement, mais le manque de temps n'a pas permis de le mettre en place.
Accidents	Partie avortée par manque de temps.
Véhicules d'urgence	Partie avortée par manque de temps.
Modes pas à pas et auto	Fonctionne bien, reste à voir comment va se comporter le mode auto avec beaucoup de voitures et si le rafraichissement est acceptable.
Tracer des routes	OK pour les villes tests et la ville finale.
Fond de carte esthétique	Reste à être implémenté, ne sera surement pas achevé et peut être un peu lourd à implémenter.
Main de déplacement pour la carte	État actuel inachevé, fonctionne mais pas de manière optimale.
Zoom	Fait, reste à régler quelques problèmes avec la vue globale de la carte au niveau de l'interpolation de la couleur de la route.
Interface globale	Tout est réalisé selon les attentes du contrat client.
Statistiques	Normalement fonctionnel, mais pas encore testé car aucune ville viable n'a été proposée par le groupe 1.
Renvoi des paramètres sélectionnés par l'utilisateur	OK

2. Étude des plannings :

Nous avions établi, lors des premières séances, le planning suivant qui devait résumer comment s'organiserait le travail tout au long de l'année, tout en se gardant une certaine marge d'avance : (vous pourrez retrouver ce planning zoomé en annexe D)

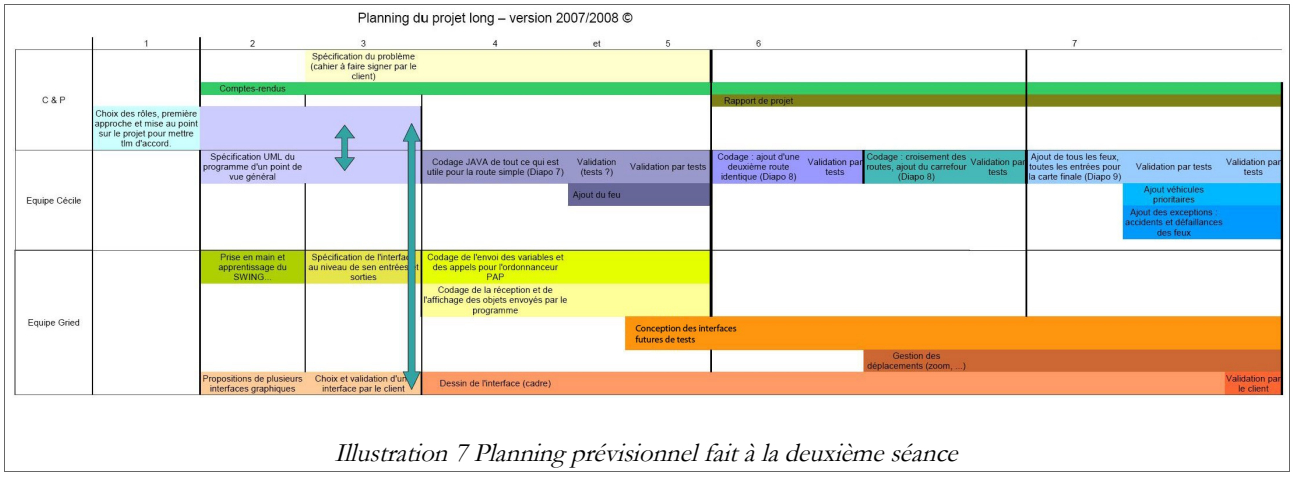


Illustration 7 Planning prévisionnel fait à la deuxième séance

Voici maintenant comment s'est effectivement déroulé le travail d'après les deux chefs d'équipe à qui nous avons régulièrement demandé de rendre des comptes (vous pourrez retrouver ces plannings dans une taille plus convenable en annexes E et F) :

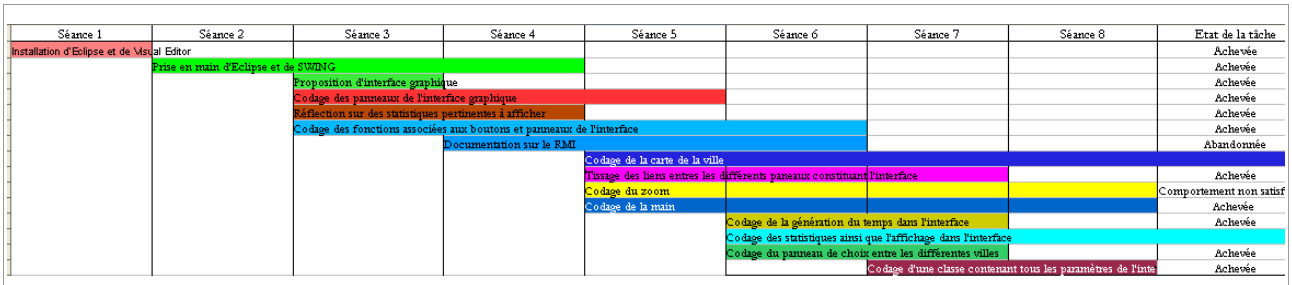


Illustration 8 Planning équipe interface

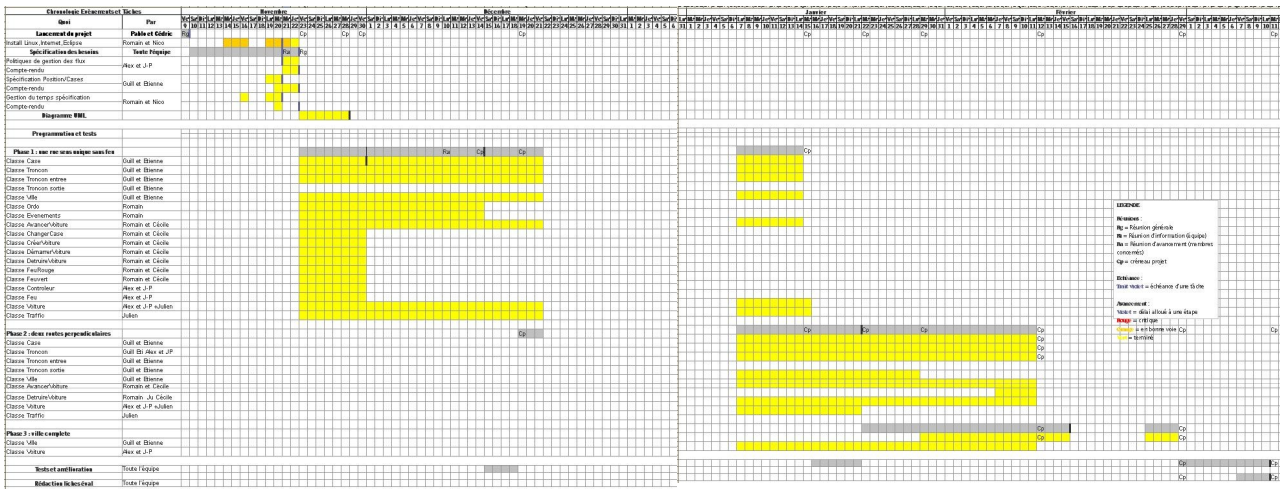


Illustration 9: Planning de l'équipe conception

La première chose que l'on peut constater est que le projet ne s'est pas du tout déroulé comme nous l'entendions. En effet, à toutes les étapes de la conception du retard s'est accumulé.

Nous avions aussi prévu de pouvoir tester nos méthodes *AvancerVoiture* et toutes les méthodes essentielles dès le début sur une ville simple, mais finalement cela n'a pas pu être fait pour plusieurs raisons :

- ces méthodes ont été finalisées après tous les programmes de tests et non parallèlement à ce qui était nécessaire
- nous avons pris beaucoup de temps pour associer le travail des deux équipes alors que l'équipe 2 proposait les interfaces de test des programmes de l'équipe 1 (dont vous trouverez les diagrammes des villes en annexes G et H.

ENSEIGNEMENTS HUMAINS

Le TP Long n'étant pas un TP comme les autres où seules des compétences techniques suffisent, nous pensons qu'il est important d'exprimer comment nous avons vécu ces séances de travail collaboratif et quels ont pû être les problèmes rencontrés. Dans cette partie, nous allons dans un premier temps reprendre tous les points qui nous on posé problème au cours du TP Long et nous tirerons ensuite quelques enseignements philosophiques de cette expérience scolaire hors du commun.

1. Problèmes divers

1. Hétérogénéité des compétences

Les élèves de notre filière participant au projet TP Long possèdent chacun leurs compétences, leurs points forts et leurs carrences. Comme on pouvait s'y attendre, pour plusieurs personnes, les domaines abordés lors du travail autour de ce projet ont posé problème. Parallèlement, pour d'autres, ce projet a permis d'affûter leur efficacité sur plusieurs sujets.
Nous avons donc pris conscience très vite de la difficulté de travailler avec des gens aux compétences différentes et dont les investissements personnels étaient extrêmement variables.

Il est apparu que se laisser les groupes se former par affinités était une erreur importante. En effet, on a pu constater qu'il est déjà apparu de fortes différences de niveau au moment de la scission des effectifs. Mais le pire s'est peut être produit au niveau des binômes où on pouvait clairement constater que le travail allait du simple au décuple.

2. Centralisation de l'information

Un des plus gros problèmes qui est apparu au cours du TP Long a été de diffuser une information claire et cohérente à chacun. Nous pouvons avouer sans trembler que nous avons pêché à ce niveau.

En effet, nous avons clairement sur-estimé notre capacité à communiquer puisque malgré l'instauration d'un forum, la tradition orale a été beaucoup trop présente ce qui a entrainé un certain nombre d'incompréhensions qui ont amené des tensions et une surcharge de travail due à certaines tâches redondantes.

Pour ce qui est de la centralisation des sources, cela semblait tout d'abord constituer un problème, beaucoup de personnes travaillant sur les même packages ou les même classes. Après quelques conseils enrichissants du personnel encadrant, nous avons pu nous tourner vers la solution de l'utilisation de GoogleCode qui permet de fonctionner avec des fichiers en commun.

2. Apprendre par le travail en groupe

1. Une expérience nouvelle

Même s'il est évident que chacun d'entre nous avait déjà travaillé en petits groupes (maximum quatre personnes) dans le cadre de l'école, ce travail à quatorze a été une première pour la plupart d'entre nous. Le travail en équipe n'est pas à prendre à la légère car il peut très bien se passer tout comme mener à un désastre. Ce fut pour chacun d'entre nous un réel apprentissage quel que soit le rôle que l'on a joué dans ce TP long.
Ce projet nous a contraint à travailler en autonomie mais aussi à confronter nos idées à celles de nos collaborateurs.

Le moteur de ce projet a aussi été la motivation éprouvée par chacun au moment de mesurer et de développer ses compétences autour d'un produit créé communement à partir d'un cahier des charges peu exhaustif, qui laissait donc un bonne part de liberté et d'initiative aux membres de l'équipe.
Il est clair que tout le monde a été motivé par sa réalisation. C'est en effet le travail de l'année qui se rapproche probablement le plus de ce que l'on sera tous amenés à faire au cours de notre parcours professionnel. Après coup, cette expérience nous a semblé indispensable avant une possible immersion parmi une équipe de travail ou de projet dans le cadre de notre futur emploi.

2. Une dimension sociale indéniable

Puisque le travail en groupe implique coproductions, répartition des tâches, échange d'idées et aide mutuelle, la réalisation de ce projet a également apporté une composante sociale. Les personnes plutôt discrètes se sont révélées et cela a permis à l'ensemble du groupe d'utiliser au mieux les compétences de chacun. Il n'est en effet pas possible lors d'un tel travail de rester à l'écart, chacun propose ses idées et aide le groupe à faire avancer le projet. Cette réalisation nous a permis de nous rendre compte de la réelle complémentarité de nos compétences.

Les étudiants qui ont pris en charge la responsabilité du projet ou même d'un groupe, ont eu l'occasion d'apprendre sur le terrain les subtilités de l'organisation et de la coordination d'une équipe. Même si nous ne sommes pas très nombreux, je pense que chacun a pu se rendre compte de la difficulté de cette tâche. Les étudiants qui ont travaillé dans un groupe ont appris à ne pas faire « bêtement » le travail que leur donnait leur « chef » mais au contraire à porter un avis critique dessus. Ainsi, de nombreuses fois au cours de ce projet, des suggestions faites aux chefs se sont révélées être pertinentes et ont été mises en application par ces derniers.
Quant au binômes en difficulté, ils ont souvent pu recevoir l'aide de camarades de leur équipe ou même de supérieurs hiérarchiques.
La leçon à retenir de cette expérience est la suivante: dans un groupe, chacun a quelque chose de nouveau et d'innovant à apporter, que ce soit sur le plan technique ou sur le plan social. Le travail d'équipe puise sa force et sa productivité dans la réunion et la contribution des meilleures aptitudes de chacun.

3. La communication, point clef de la réussite

Par ailleurs, ce projet a nécessité de nombreuses réunions, au début d'abord pour se mettre d'accord sur la méthode de résolution du problème posé, puis au cours du projet pour faire le point sur l'avancement des deux parties, et finalement s'entendre pour la réunion des sources pour la reconstitution du produit final.
Cette notion de réunion a été pour nous un réel apprentissage puisque c'était une grande première. En entreprise, les réunions sont quasi quotidiennes et peuvent ne pas être toujours bien menées. Nous avons constaté une amélioration notable du déroulement de nos réunions au cours de l'avancée du travail. Les réunions organisées au commencement du projet étaient longues et pas assez cadrées, ce qui aboutissait en fin de compte à une perte de temps. Au fur et à mesure elles sont devenues plus brèves et efficaces.
On pourra en retenir que le plus important dans une réunion est sa préparation, mais aussi la concision et la simplicité des propos. Si l'ordre du jour est bien déterminé et communiqué à

l'avance aux participants, chacun peut y réfléchir et la réunion est constructive. Nous avons aussi noté l'importance de rédiger les décisions prises lors de ces discussions et de prendre des notes.

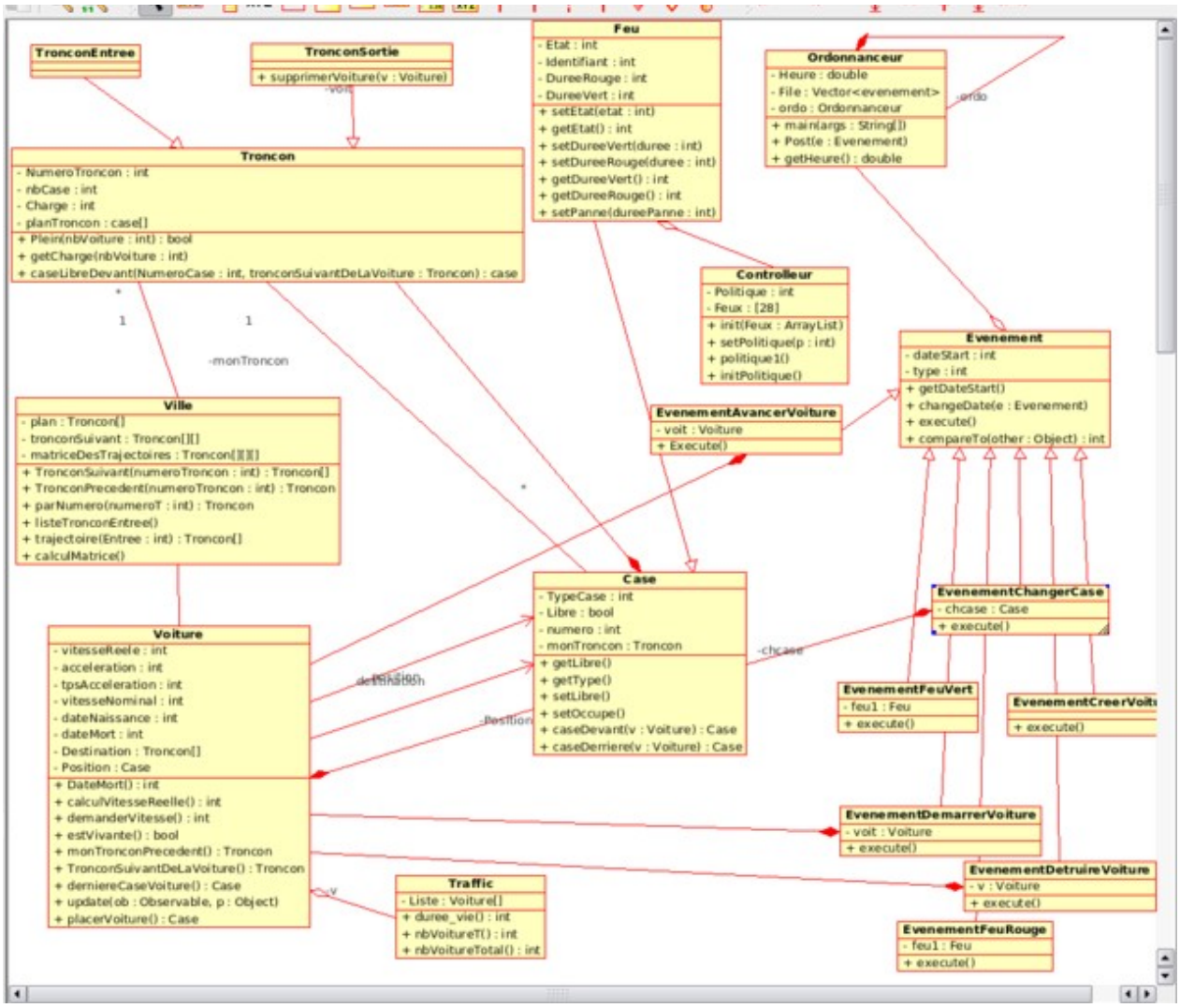
CONCLUSION

Ce super projet nous a donné un avant goût du travail collaboratif, et du monde du développement de logiciels grâce à une approche plus humaine que celle de la simple réflexion autour d'un cahier des charges.
Même si le résultat n'est pas celui que nous attendions au commencement, les problèmes qui sont survenus et notre réactivité face à eux nous ont permis de prendre conscience de la complexité à faire cohabiter plusieurs difficultés : garder un bon contact humain avec les autres membres de l'équipe sans pour autant être laxiste, se concentrer sur le travail qui nous est alloué tout en restant informés, tenir des délais assez stricts, et surtout organiser et planifier le travail de chacun.

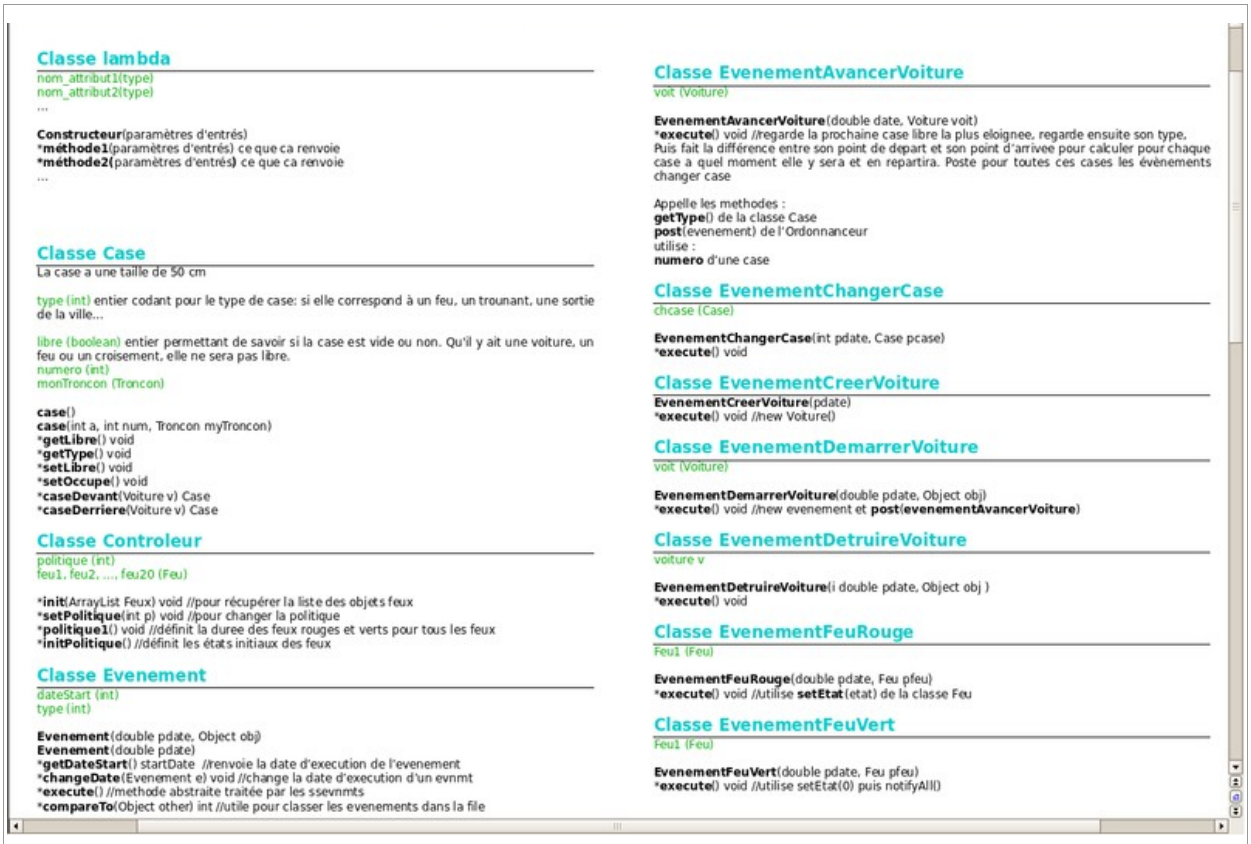
Participer à cette expérience a permis à chacun de nous de faire un travail d'auto évaluation de son profil en entreprise et nous a appris à nous prémunir de certains des pièges du travail d'équipe .

Au final, la satisfaction d'avoir tous contribué à l'élaboration d'un produit fini fonctionnel constitue une satisfaction scientifique indéniable.

ANNEXES



Annexe A: Diagramme UML complet de la partie de l'équipe 1



Classe lambda

nom_attribut1(type)
nom_attribut2(type)
...

Constructeur(paramètres d'entrés)
***méthode1**(paramètres d'entrés) ce que ca renvoie
***méthode2**(paramètres d'entrés) ce que ca renvoie
...

Classe Case

La case a une taille de 50 cm

type (int) entier codant pour le type de case: si elle correspond à un feu, un trounant, une sortie de la ville...

libre (boolean) entier permettant de savoir si la case est vide ou non. Qu'il y ait une voiture, un feu ou un croisement, elle ne sera pas libre.
numero (int)
monTroncon (Troncon)

case()
case(int a, int num, Troncon myTroncon)
***getLibre**() void
***getType**() void
***setLibre**() void
***setOccupe**() void
***caseDevant**(Voiture v) Case
***caseDerriere**(Voiture v) Case

Classe Controleur

politique (int)
feu1, feu2, ..., feu20 (Feu)

***init**(ArrayList Feux) void //pour récupérer la liste des objets feux
***setPolitique**(int p) void //pour changer la politique
***politique1**() void //définit la duree des feux rouges et verts pour tous les feux
***initPolitique**() //définit les états initiaux des feux

Classe Evenement

dateStart (int)
type (int)

Evenement(double pdate, Object obj)
Evenement(double pdate)
***getDateStart**() startDate //renvoie la date d'execution de l'evenement
***changeDate**(Evenement e) void //change la date d'execution d'un evnmt
***execute**() //methode abstraite traitée par les ssevnmts
***compareTo**(Object other) int //utile pour classer les evenements dans la file

Classe EvenementAvancerVoiture

voit (Voiture)

EvenementAvancerVoiture(double date, Voiture voit)
***execute**() void //regarde la prochaine case libre la plus eloignee, regarde ensuite son type. Puis fait la différence entre son point de depart et son point d'arrivee pour calculer pour chaque case a quel moment elle y sera et en repartira. Poste pour toutes ces cases les évènements changer case

Appelle les methodes :
getType() de la classe Case
post(evenement) de l'Ordonnanceur
utilise :
numero d'une case

Classe EvenementChangerCase

chcase (Case)

EvenementChangerCase(int pdate, Case pcase)
***execute**() void

Classe EvenementCreerVoiture

EvenementCreerVoiture(pdate)
***execute**() void //new Voiture()

Classe EvenementDemarrerVoiture

voit (Voiture)

EvenementDemarrerVoiture(double pdate, Object obj)
***execute**() void //new evenement et **post**(**evenementAvancerVoiture**)

Classe EvenementDetruireVoiture

voiture v

EvenementDetruireVoiture(i double pdate, Object obj)
***execute**() void

Classe EvenementFeuRouge

Feu1 (Feu)

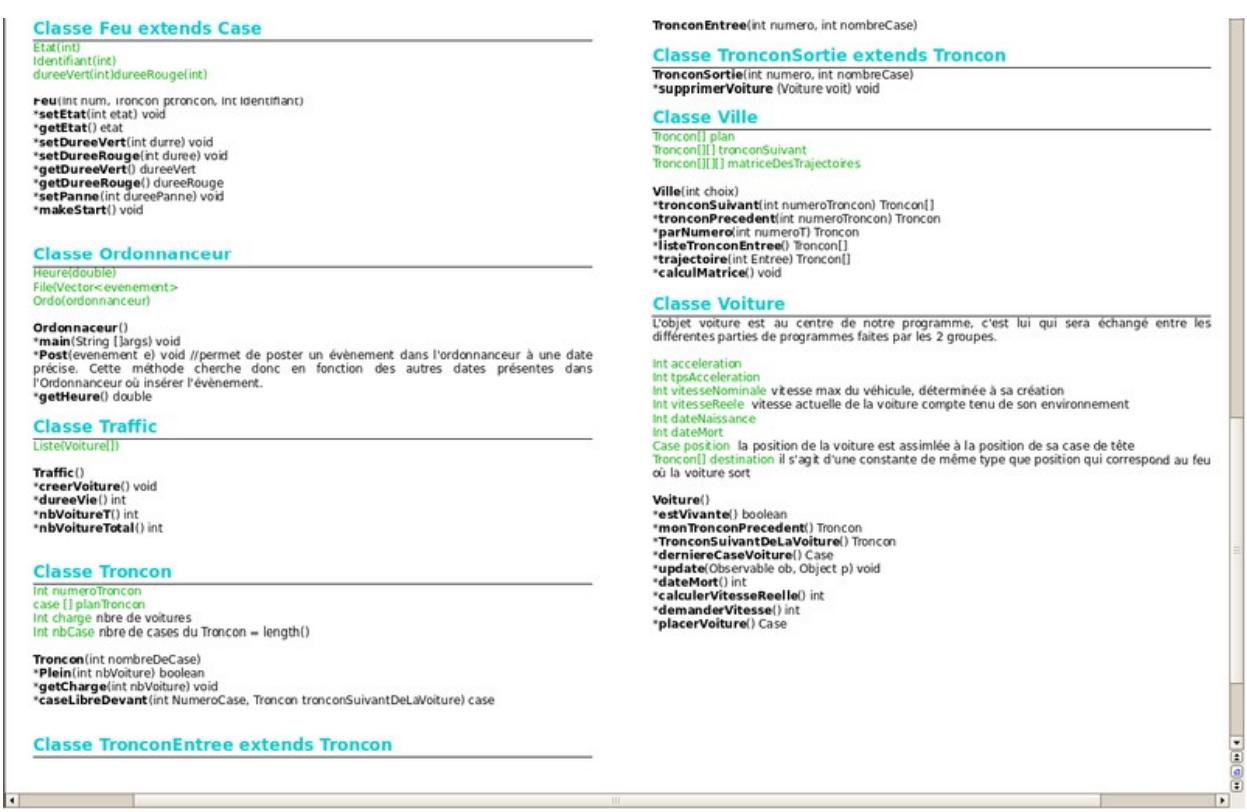
EvenementFeuRouge(double pdate, Feu pfeu)
***execute**() void //utilise **setEtat**(etat) de la classe Feu

Classe EvenementFeuVert

Feu1 (Feu)

EvenementFeuVert(double pdate, Feu pfeu)
***execute**() void //utilise setEtat(0) puis notifyAll()

Annexe B: Suite de la liste des classes, méthodes et attributs de la partie de l'équipe 1



Classe Feu extends Case

Etat(int)
Identifiant(int)
dureeVert(int)dureeRouge(int)

Feu(int num, Troncon ptroncon, int Identifiant)
***setEtat**(int etat) void
***getEtat**() etat
***setDureeVert**(int durre) void
***setDureeRouge**(int duree) void
***getDureeVert**() dureeVert
***getDureeRouge**() dureeRouge
***setPanne**(int dureePanne) void
***makeStart**() void

Classe Ordonnanceur

Heure(double)
File(Vector<evenement>
Ordo(ordonnanceur)

Ordonnaceur()
***main**(String []args) void
***Post**(evenement e) void //permet de poster un évènement dans l'ordonnanceur à une date précise. Cette méthode cherche donc en fonction des autres dates présentes dans l'Ordonnanceur où insérer l'évènement.
***getHeure**() double

Classe Traffic

Liste(Voiture[])

Traffic()
***creerVoiture**() void
***dureeVie**() int
***nbVoitureT**() int
***nbVoitureTotal**() int

Classe Troncon

Int numeroTroncon
case [] planTroncon
Int charge nbre de voitures
Int nbCase nbre de cases du Troncon = length()

Troncon(int nombreDeCase)
***Plein**(int nbVoiture) boolean
***getCharge**(int nbVoiture) void
***caseLibreDevant**(int NumeroCase, Troncon tronconSuivantDeLaVoiture) case

Classe TronconEntree extends Troncon

TronconEntree(int numero, int nombreCase)

Classe TronconSortie extends Troncon

TronconSortie(int numero, int nombreCase)
***supprimerVoiture** (Voiture voit) void

Classe Ville

Troncon[] plan
Troncon[][] tronconSuivant
Troncon[][][] matriceDesTrajectoires

Ville(int choix)
***tronconSuivant**(int numeroTroncon) Troncon[]
***tronconPrecedent**(int numeroTroncon) Troncon
***parNumero**(int numeroT) Troncon
***listeTronconEntree**() Troncon[]
***trajectoire**(int Entree) Troncon[]
***calculMatrice**() void

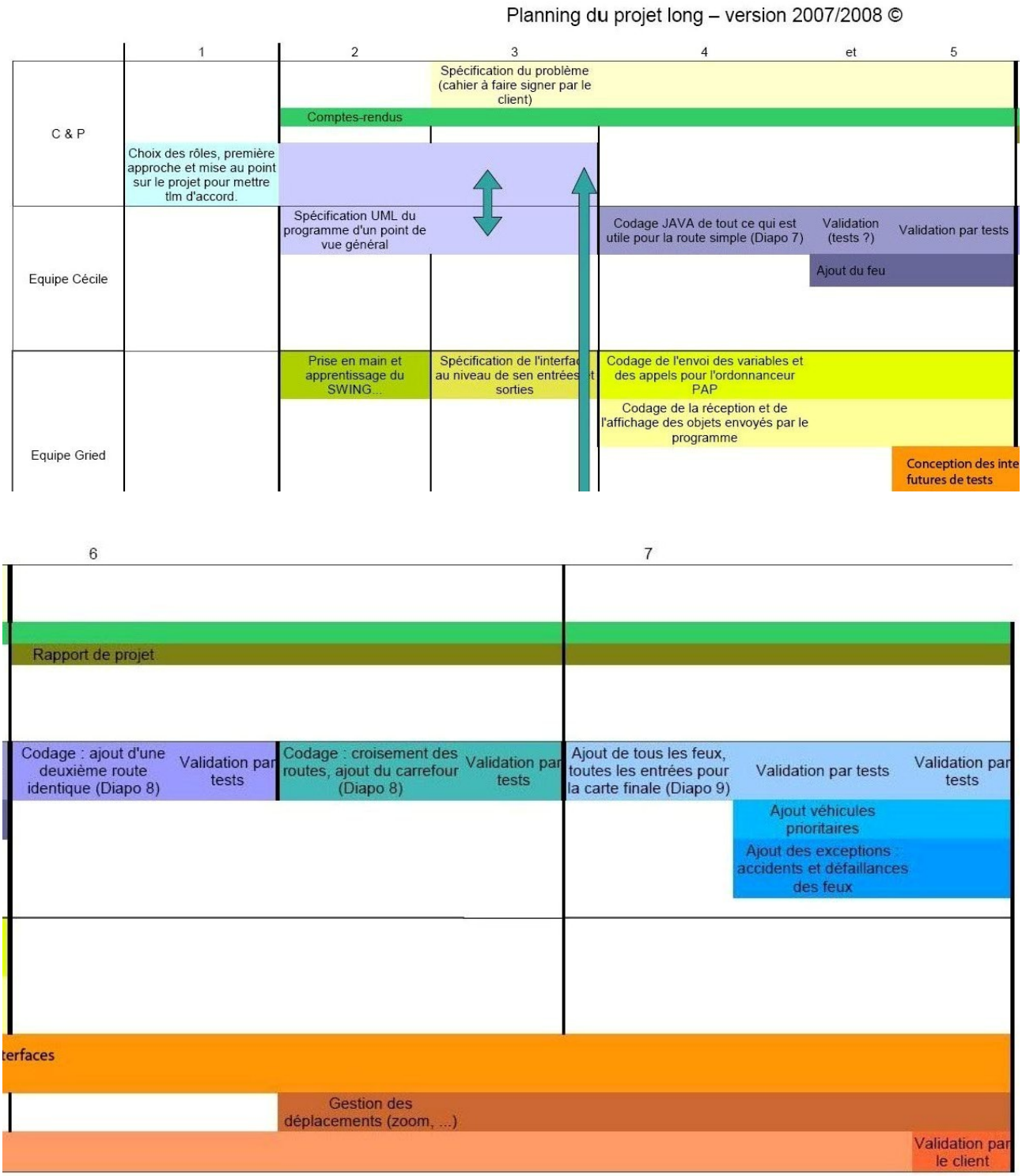
Classe Voiture

L'objet voiture est au centre de notre programme, c'est lui qui sera échangé entre les différentes parties de programmes faites par les 2 groupes.

Int acceleration
Int tpsAcceleration
Int vitesseNominale vitesse max du véhicule, déterminée à sa création
Int vitesseReele vitesse actuelle de la voiture compte tenu de son environnement
Int dateNaissance
Int dateMort
Case position la position de la voiture est assimilée à la position de sa case de tête
Troncon[] destination il s'agit d'une constante de même type que position qui correspond au feu où la voiture sort

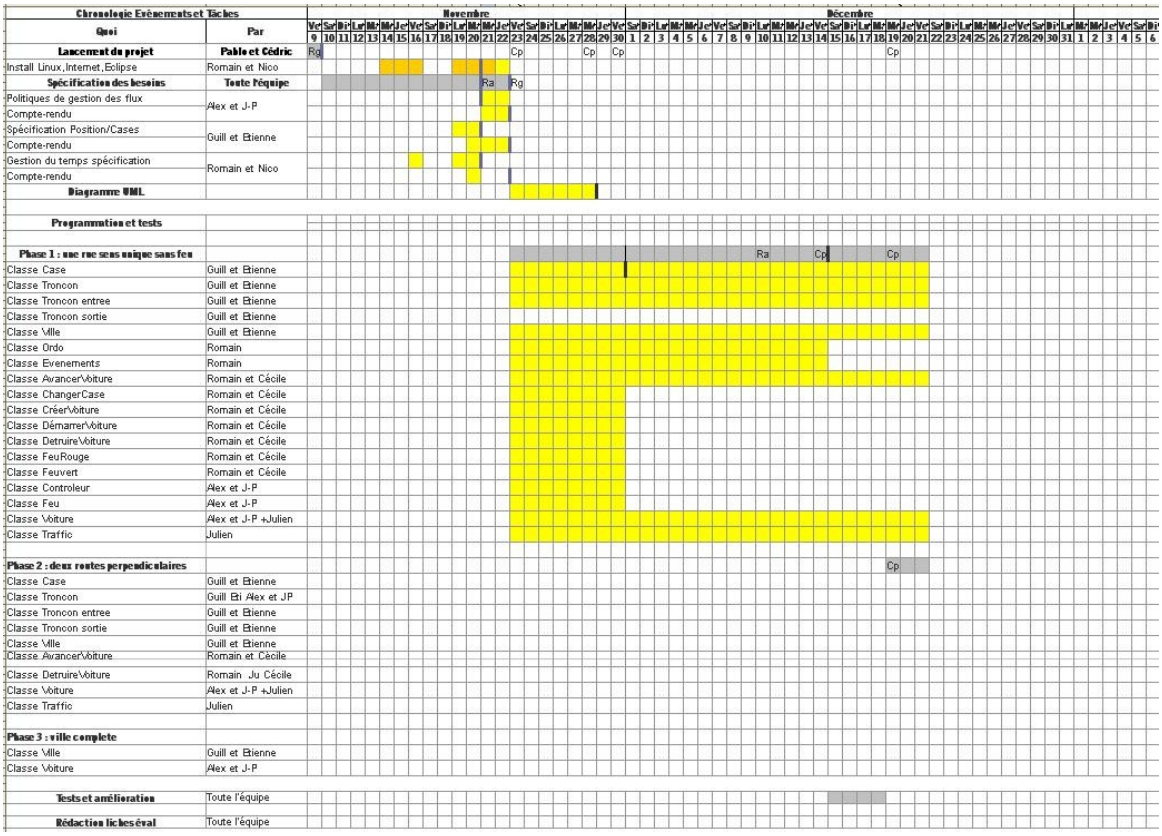
Voiture()
***estVivante**() boolean
***monTronconPrecedent**() Troncon
***TronconSuivantDeLaVoiture**() Troncon
***derniereCaseVoiture**() Case
***update**(Observable ob, Object p) void
***dateMort**() int
***calculerVitesseReelle**() int
***demanderVitesse**() int
***placerVoiture**() Case

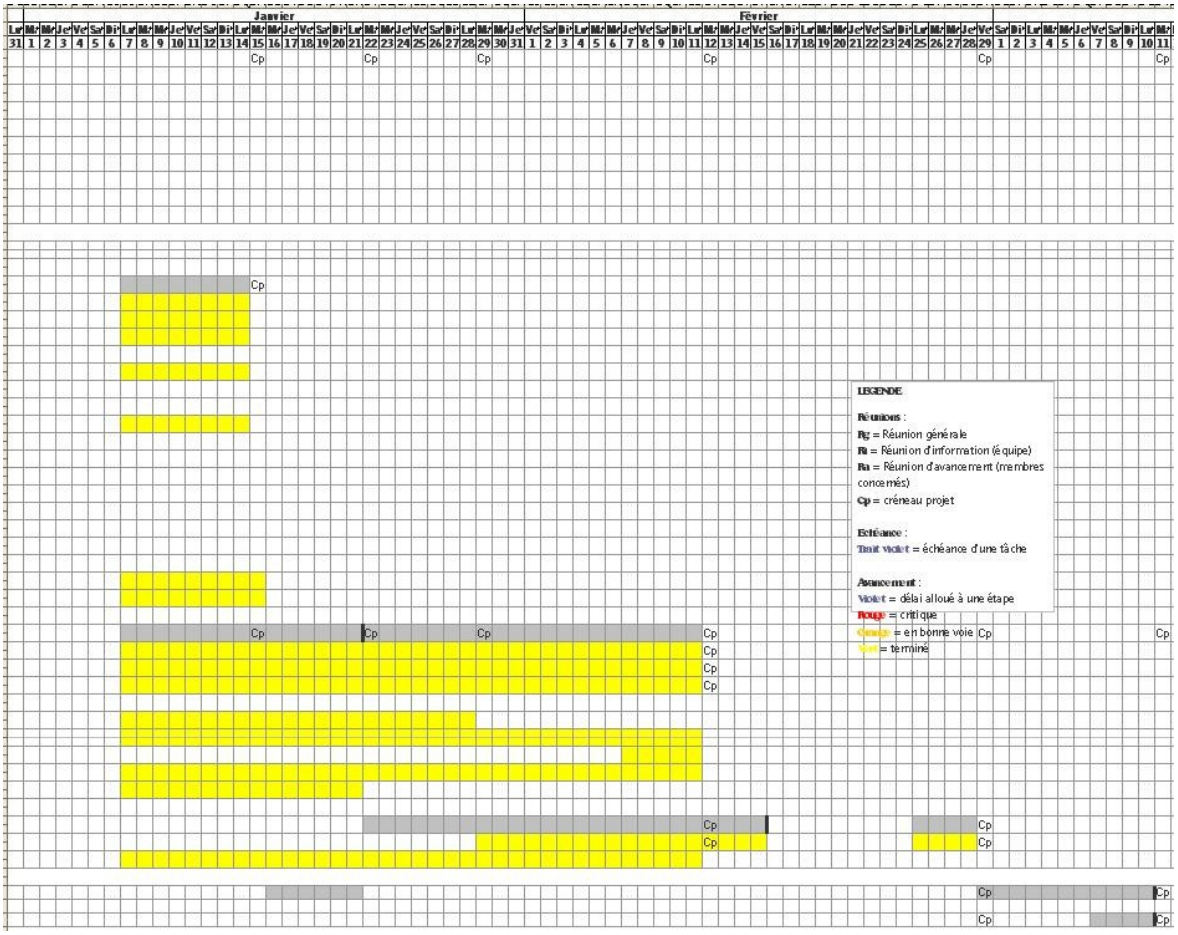
Annexe C: Liste des classes, méthodes et attributs de la partie de l'équipe 1



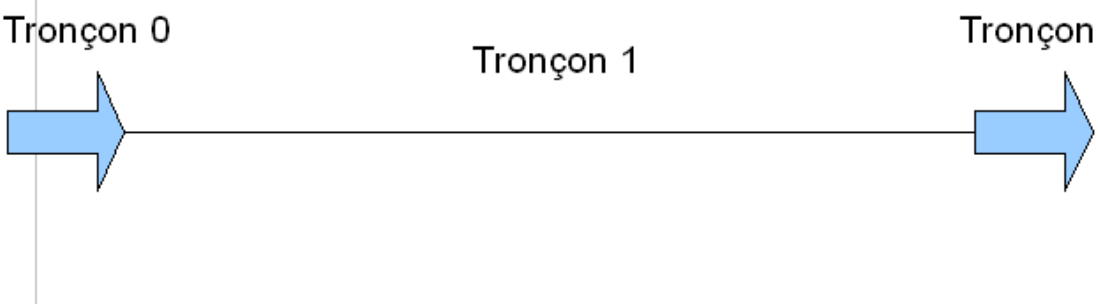
Annexe D : Planning de début d'année

Annexe E: Planning de l'équipe interface



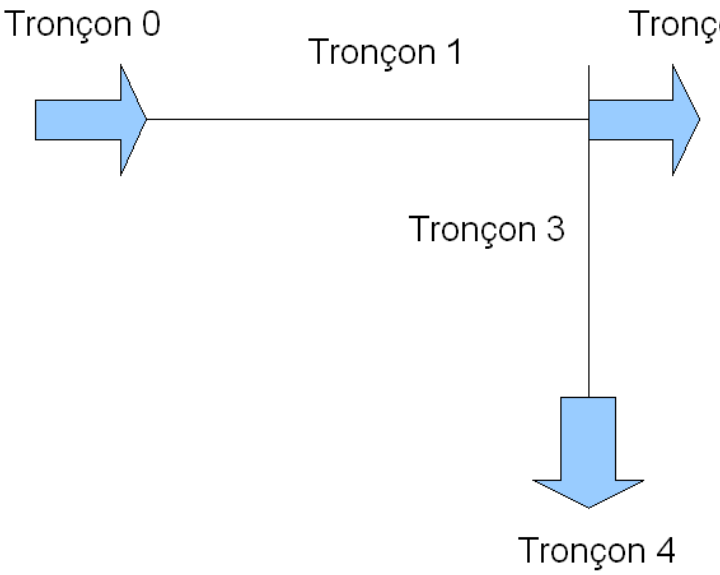


Annexe F : Planning de l'équipe conception



Le tronçon d'entrée(tronçon 0)fait 10 mètre soit 20 cases et celui de sortie fait 4 mètres soit 8 cases (tronçon 2).
Le tronçon 1 fait tous les deux 8 mètres soit 16 cases.

Annexe G: Ville test constituée d'une route simple



Le tronçon d'entrée(tronçon 0)fait 10 mètre soit 20 cases et ceux de sorties font 4 mètres soit 8 cases (tronçon 2;4).
Les tronçons 1 et 3 font tous les deux 10 mètres soit 20 cases.

Annexe H: Ville Test avec un carrefour

Fiches d'évaluation individuelles

Comme demandé par notre M. Derniame, nous avons distribué à chacun la fiche d'évaluation suivante, que les gens ont retransmis à leur supérieur pour qu'il ajoute un commentaire.

Dupont Martin, dit « Blabla » **Equipe X**

Travail préliminaire

Participation aux discussion sur l'architecture du programme et solutions apportées :

Blabla.

Réalisation du programme

Éléments réalisés:

Blabla.

Travail supplémentaire réalisé sur le code :

Blabla.

Travaux complémentaires ou projets d'investissement personnel :

Blabla.

Impressions personnelles

Sur votre travail en général, au sein du groupe

Blabla.

Ce que vous retiendrez du TP Long

Blabla.

Commentaires personnels

Blabla.

Appréciation du supérieur hiérarchique

Blabla.

Travail préliminaire

Participation aux discussion sur l'architecture du programme et solutions apportées :

J'ai conçu la base de l'architecture du programme en proposant un grand nombre de solutions à l'ensemble du groupe de travail. J'ai été à l'origine de l'ordonnanceur asynchrone, des méthodes de déplacement des voitures, d'organisation topographique de la ville, ...

Réalisation du programme

Éléments réalisés:

J'ai aidé ponctuellement à coder quelques classes, notamment *Voiture* et *Traffic.*

Travail supplémentaire réalisé sur le code :

Aide à la résolution de quelques bugs techniques.

Travaux complémentaires ou projets d'investissement personnel :

En collobaration avec mon secrétaire, nous avons redigé ce rapport et le contrat client que j'ai présentée, et il y a bien sûr les deux présentations orales.

J'ai aussi créé plusieurs plannings pour l'organisation du travail au fil des séances, et pris contact avec les chefs d'équipes pour savoir où ils en étaient, leur dire ce que j'attendais d'eux et quels étaient leurs prochains objectifs à atteindre en précisant les délais.

Impressions personnelles

Sur votre travail en général, au sein du groupe

J'ai personnellement adoré avoir à gérer le projet dans sa globalité, sans avoir à me soucier des détails du code, mais plus d'un point de vue management et relation avec le client.

Je regrette simplement que tout le monde n'ait pas joué le jeu, mais je peux comprendre les gens étant donné qu'il s'agit là d'un projet scolaire et que tout le monde ne s'est pas vu confié un travail aussi enrichissant.

Ce que vous retiendrez du TP Long

Qu'il faut toujours garder une trace du travail qu'on exige de ses collaborateurs, l'oral étant vraiment sujet à des rixes inutles pour savoir si oui ou non on a dit ça.

Je retiendrai bien sûr aussi que Chef de Projet et tous les métiers affiliés où l'on se voit donner des responsabilités sont vraiment quelque chose qui m'intéresse pour ma future vie professionnelle.

Commentaires personnels

Si j'avais eu une vraie équipe, dans le cadre d'un emploi, je pense que j"aurais pris la décision de remplacer certaines personnes qui n'accomplissaient pas le travail demandé. J'aurais aimé savoir si celà est une bonne décision, et si ça n'aurait pas trop diminué le moral du reste de l'équipe...

Pablo VASQUEZ-GUILENDHOU, dit « Le bras gauche » — Sécrétaire

TRAVAIL PRÉLIMINAIRE

Participation aux discussion sur l'architecture du programme et solutions apportées :

Lors du premier brainstorming autour du sujet, j'ai structuré et classé les solutions et les problèmes soulevés par la discussion. J'ai activement participé à l'organisation des classes et leur définition, au choix et l'aproffondissement de la solution asynchrone.

Mise en place et gestion du Forum en collaboration étroite avec Mohammed Bayoumi.

Construction commune de l'interface graphique avec Cédric Bastien.

RÉALISATION DU PROGRAMME

Travail supplémentaire réalisé sur le code :

Elaboration de l'image servant d'arrière plan à l'affichage du traffic. Elaboration des éléments graphiques repris lors du choix des boutons et autres composantes de l'interface.

Travaux complémentaires ou projets d'investissement personnel :

Rédaction des comptes rendus des premières séances et adaptation au forum. Mise en ligne des premières définitions des classes avant l'utilisation concrète des diagrammes UML.

Travail important sur l'interface graphique avant son implémentation par le code en vue d'une validation client dans le contrat. Il a fallut très tôt présenter une image précise de l'interface pour savoir dans quelle direction allait avancer l'équipe associée.

Lors de l'absence du chef de projet, j'ai fait le lien entre les deux groupes et me suis tenu au courant de leurs avancées respectives.

Rédaction du présent rapport, des documents PowerPoint destinées aux présentations orales.

IMPRESSIONS PERSONNELLES

Sur votre travail en général, au sein du groupe

Au sein de l'équipe, ma place m'a obligé à rester au courant de l'avancement précis des deux groupes, ce qui n'a pas toujours été facile, étant donné le morcellement des tâches entre leurs membres. J'ai pu réfléchir avec le Chef de projet concernant les décisions à prendre et le travail à donner.

Voir l'avancement d'un projet sans avoir à coder soi-même m'a semblé une façon de mieux savoir dans quel état d'avancement le travail était. Ce recul était nouveau et très instructif.

Ce que vous retiendrez du TP Long

L'importance de la communication avec tous les membres des équipes de travail, même si l'information ne devait remonter que par les chefs de ces équipes. La nécessité de la proche présence des supérieurs hierarchiques lors du développement tant pour guider que pour participer. Le côté humain.

Commentaires personnels

Le projet n'a pas été tous les jours facile, étant donné le manque d'entrain de certains ou leurs carrences en compétences. Cependant, plusieurs éléments moteurs ont pu assurer au projet un avenir concret. La dureté apparente des directives données par le Chef a porté ses fruits.

APPRÉCIATION DU SUPÉRIEUR HIÉRARCHIQUE (CÉDRIC)

Même si Pablo n'a pas forcément rempli les fonctions qu'on attendait d'un secrétaire, notamment sur ce rapport, il a sû être présent sur tous les fronts et se montrer d'une aide indispensable et de grande qualité dans l'accomplissement de ce projet au niveau de la direction, de la logistique et du contact avec le client. Cet homme a sans aucun doute l'âme d'un chef de projet.

Cécile DE TRENTINIAN — Chef équipe 1

TRAVAIL PRÉLIMINAIRE

Participation aux discussion sur l'architecture du programme et solutions apportées :

Avec Romain, nous avons réfléchi aux différentes gestions du temps possibles et avons retenu la solution de la gestion par évènements. Nous nous sommes interrogés sur le « réveil » des voitures, et conclut qu'elles seraient réveillées par l'objet qui les précède.

RÉALISATION DU PROGRAMME

Éléments réalisés:

Avec Romain nous nous sommes occupés de la classe *Evenement* et des sous-évènements (c'est-à-dire en grande partie la classe *EvenementAvancerVoiture*).

Travail supplémentaire réalisé sur le code :

J'ai pu travailler sur toutes les parties du code afin de répondre aux questions de l'équipe et/ou de corriger des erreurs. A la fin notamment, avec la fusion du travail des deux équipes, il y a eu un gros travail de débuggage pour notre partie, que j'ai réalisé avec différents membres de l'équipe.

Travaux complémentaires ou projets d'investissement personnel :

Gestion et organisation de l'équipe 1 (tant bien que mal), avancement du projet pour l'équipe 1.

IMPRESSIONS PERSONNELLES

Sur votre travail en général, au sein du groupe

Malgré une grande motivation les premiers mois, une mauvaise organisation et une répartition floue des rôles a entraîné un retard dans le travail.

J'ai concilié programmation et gestion de l'équipe pour plus de crédibilité, mais à la fin ça faisait trop à gérer avec les autres matières.

Ce que vous retiendrez du TP Long

Qu'il faut déléguer, et que c'est difficile. Qu'il faut tout noter, surtout ce qui semble le moins important... Qu'il faut être crédible techniquement pour manager, et rester motivé. Qu'on a besoin de directives claires et précises pour travailler.

Le versionnage avec Subclipse.

Commentaires personnels

Le TP long s'est étalé sur toute l'année, il fallait à chaque fois se replonger dans le projet. Des séances plus raprochées auraient été mieux.

APPRÉCIATION DU SUPÉRIEUR HIÉRARCHIQUE (CÉDRIC)

Un manque de vision du projet, des connaissances en Java pas toujours affûtées et une sur-présence de ma part, voila quelques raisons qui ont pu initiallement handicaper Cécile dans son rôle de chef d'équipe. Elle a sû néanmoins s'imposer avec un investissement personnel qui surclasse tous les autres acteurs du TP Long, une motivation à toutes épreuves et d'appréciables qualités de communiquante.

Julien DIDION, dit « Le référent UML » — Membre équipe 1

TRAVAIL PRÉLIMINAIRE

Participation aux discussion sur l'architecture du programme et solutions apportées :

J'ai été responsable de l'établissement du diagramme UML. J'ai donc mené les discussions concernant sa création. Ce diagramme est primordial car il définit la structure du programme. J'ai également activement participé aux discussions sur l'architecture depuis le début du projet.

RÉALISATION DU PROGRAMME

Éléments réalisés:

J'ai réalisé la classe *Trafic* et une partie de laclasse *Voiture*.

Travail supplémentaire réalisé sur le code :

J'ai corrigé les erreurs des classes *EvenementCreerVoiture* et *EvenementDetruireVoiture*. J'ai codé la méthode *monTronconPrecedent* qui renvoie le troncon précedent sur le trajet de la voiture. J'ai vérifié le bon fonctionnement de la classe *Voiture*.

Travaux complémentaires ou projets d'investissement personnel :

J'ai été sollicité plusieurs fois par les « chefs » pour faire : la description de l'interface graphique, le récapitulatif de toutes les classes avec méthodes et attributs (différent de ce qui était prévu à l'origine) et la rédaction du bilan de notre travail en équipe. J'ai fait tous les diagrammes UML, que ce soit le diagramme complet ou les diagrammes intermédiaires.

IMPRESSIONS PERSONNELLES

Sur votre travail en général, au sein du groupe

Je n'ai rencontré aucun problème au sein du groupe, je faisais ce que Cécile me disait de faire (vérification de classes, ajout de méthodes...). J'ai aussi fait plusieurs tâches qui m'ont été demandé directement par Cédric et Pablo.

Ce que vous retiendrez du TP Long

Lorsque l'on pense finir un travail en x heures, il faut bien en compter le double car pendant que vous êtes en train de le réaliser, on peut vous demander de l'aide sur autre chose, ou bien encore modifier ce que l'on vous a demandé... Il est très important de bien préparer à l'avance les réunions pour qu'elles soient efficaces.

Commentaires personnels

Comme les mousquetaires, on est plus fort à plusieurs.

APPRÉCIATION DU SUPÉRIEUR HIÉRARCHIQUE (CÉDRIC & CÉCILE)

Des connaissances en JAVA un peu légères qui ont été plus que compensées par une motivation et un investissemnt personnel très appréciables, Julien était toujours là pour réaliser avec panache les tâches qu'on lui assignait, et ce, dans les plus brefs délais.

Alexandre LEDUC, dit « Alex » — Membre équipe 1

TRAVAIL PRÉLIMINAIRE

Participation aux discussion sur l'architecture du programme et solutions apportées :

On a réalisé en binôme un dossier sur les politiques des feux, mais il n'a pas servi par la suite car on était en retard sur l'avancement du projet. Ce travail préliminaire correspondait plus à une anticipation sur notre travail.

J'ai aussi participé à la réflexion sur les moyens à mettre en oeuvre pour faire fonctionner le programme. Enfin, j'ai fait un fichier qui expliquait comment le programme était censé fonctionner en mode pas à pas.

RÉALISATION DU PROGRAMME

Éléments réalisés

On a réalisé avec Jean-Philippe des éléments dans les classes *Feu, Politique, Controleur, Ville* et *Voiture*.

IMPRESSIONS PERSONNELLES

Sur votre travail en général, au sein du groupe

Lorsqu'on me donnait du travail, j'essayais de le réaliser au plus vite. On est restés trop figés dans l'organisation générale du groupe alors que c'était peut-être l'occasion d'en profiter..

J'ai dû recommencer mon travail pour passer des vecteurs aux listes, mais à la fin l'utilisation des listes s'est avérée problèmatique.

Ce que vous retiendrez du TP Long

Le fonctionnement de SVN.

Commentaires personnels

Les temps entre séances étaient un handicap supplémentaire à la réalisation du programme.

Le travail en équipe est délicat lorsque l'on doit travailler sur une même classe. La mise en place d'une communication efficace tant pour les membres de l'équipe que pour le chef était délicate.

APPRÉCIATION DU SUPÉRIEUR HIÉRARCHIQUE (CÉCILE)

Bon programmeur, tête en l'air mais sérieux quand il faut.

Romain LANETTE — Membre équipe 1

TRAVAIL PRÉLIMINAIRE

Participation aux discussion sur l'architecture du programme et solutions apportées :

Avec Cécile, nous avons étudié deux possibilités de fonctionnement pour la gestion du temps dans la partie Conception : une gestion périodique qui demande plus de calculs et une gestion par événement en utilisant un ordonnanceur. Il a été choisi d'utiliser la seconde solution.

RÉALISATION DU PROGRAMME

Éléments réalisés:

J'ai réalisé la classe *Ordonnanceur* : son constructeur et la méthode *post* qui ajoute un élément dans l'ordonnanceur. Pour ce faire, j'ai réalisé une méthode qui utilise l'implémentation *Comparable* qui permet de classer automatiquement les évènements de l'ordonnanceur dans l'ordre chronologique de leur exécution.

Avec Cécile, nous avons réalisé la classe *Evenement* ainsi que toutes les sous-classes *Evenement*. La plus importante et la plus difficile à coder ayant été la classe *EvenementAvancerVoiture*.

Travail supplémentaire réalisé sur le code :

Lors de la dernière séance avec Cécile, nous avons corrigé les erreurs de code de nos classes ainsi que des classes Troncon et Fenetre.

Travaux complémentaires ou projets d'investissement personnel :

Avec Nicolas, nous avons monté les nouvelles machines de la salle 204 jaune et installé Ubuntu 7.10 sur toutes les machines. Nous avons installé Eclipse ainsi que tous les plugins nécessaires. (RMI, Visual Editor, SVN)

Sur le forum du projet, j'ai réalisé deux tutoriels pour apprendre à installer et utiliser Google Code avec le plugin Subclipse (SVN pour Eclipse):
http://sertr.freeforums.org/how-to-installer-le-plug-in-svn-subclipse-t69.html
http://sertr.freeforums.org/how-to-utiliser-le-plug-in-svn-subclipse-avec-google-code-t70.html

IMPRESSIONS PERSONNELLES

Sur votre travail en général, au sein du groupe

Ce projet nous a clairement montré la difficulté de travailler en équipe. Le principal problème ayant été la communication au sein de l'équipe. Nous avons sûrement abusé de la communication orale au lieu d'instructions écrites, définitives et claires pour tout le monde.

Commentaires personnels

Malgré les difficultés, ce TP aura été intéressant. L'inconvéniant majeur d'un tel projet dans un parcours scolaire est le fait qu'il est forcément réparti tout au long de l'année et qu'entre deux séances on perd forcément un peu de motivation et de concentration. Serait-il possible de rassembler les horaires sur une certaine partie de l'année ?

APPRÉCIATION DU SUPÉRIEUR HIÉRARCHIQUE (CÉCILE)

Expert technique de l'équipe, travail dont le sérieux et la qualité le rendent exemplaire. Sa motivation et son investissement en font sans aucun doute le meilleur membre de l'équipe.

Guillemin Seng et Étienne Delecroix — Membres équipe 1

Travail préliminaire

Participation aux discussion sur l'architecture du programme et solutions apportées :

Nous avons réalisé l'étude des cases qui a abouti à la définition de ce qu'est une case et surtout la taille de ces dernières. Nous avons participé aux discussions sur la définition de ce qu'est un objet carrefour(son intérêt si il est utile d'en créer un) , l'objet voiture et surtout l'objet Tronçon et l'objet Ville.

Nous avons également réalisé le premier diagramme UML qui permet de visualiser l'ensemble du programme et les liens entres les différentes classes et méthodes.

Réalisation du programme

Éléments réalisés:

Nous avons codé les classes Ville , Tronçon, Case, Tronçons d'entrée et de sortie.

Travail supplémentaire réalisé sur le code :

Nous avons réalisé une classe test qui nous a permis de vérifier l'ensemble de notre code.

Différentes Villes pour tester le bon fonctionnement du programme.

Travaux complémentaires ou projets d'investissement personnel :

Nous avons installé Ubuntu sur nos machines personnelles ainsi qu'eclipse et Umbrello(pour faire le diagramme UML). Le lien vers google code.

Impressions personnelles

Sur votre travail en général, au sein du groupe

Nous avons découvert l'intérêt de travailler en pair-programming qui nous a permis d'avoir un avis différent sur notre code ce qui est toujours intéressant, de plus le pair-programming nous a permis de faire vérifier notre code par l'autre et éviter ainsi beaucoup d'erreurs de compréhension.

Le principe de travail hiérarchique est un bon système, mais il faut qu'il soit respecté sinon cela ralentit le bon déroulement du projet. En effet nous avions un chef direct et nos managers sont venus directement nous dire ce que l'on devait faire, puis lors des séances suivantes, n'ayant pas mis tout cela par écrit, il fallait modifier notre travail.

Ce que vous retiendrez du TP Long

Qu'il est plus utile de communiquer par écrit même si cela peut nous paraître plus long dans un premier temps mais au final c'est nettement plus rapide pour l'exécution du projet, car cela nous évite de revenir sur ce qui a été fait.

Commentaires personnels

L'organisation en chef/sous-chef n'était pas nécessaire dans le cadre de notre projet car les équipes sont relativements petites (5-6 membres) et le travail du chef de projet en entreprises consiste beaucoup plus en une vision économique, administrative et temporelle du projet vis à vis du client. Or ce coté n'a pas ou très peu été présent dans notre projet (surtout l'aspect économique).

Le supérieur hérarchique direct était très impliqué dans le projet. En effet, elle passait beaucoup de temps à lire nos programmes et à les comprendre afin d'avoir une vue globale du travail du groupe. C'est cette motivation de sa part qui nous a beaucoup poussés à avancer dans notre travail.

Appréciation du supérieur hiérarchique (Cécile)

De bons éléments qui ont vu leurs connaissances Java s'améliorer avec le projet. Ce sont eux qui ont le plus pâti des problèmes de communication, ils sont tout de même restés motivés jusqu'au bout.

Jean Philippe MULLER, dit « Papa » — Membre équipe 1

TRAVAIL PRÉLIMINAIRE

Participation aux discussion sur l'architecture du programme et solutions apportées :

Réalisation en binome d'un dossier sur les politiques des feux.

RÉALISATION DU PROGRAMME

Éléments réalisés:

Codage classes *Feu*, *Controleur*, certaines méthodes de la classe *Ville* et *Voiture*.

Travail supplémentaire réalisé sur le code :

Correction d'erreurs lors des essais

TRAVAUX COMPLÉMENTAIRES OU PROJETS D'INVESTISSEMENT PERSONNEL :

Recherche sur le fonctionnement des feux de signalisation existants

IMPRESSIONS PERSONNELLES

Sur votre travail en général, au sein du groupe

Ce que vous retiendrez du TP Long

Les méthodes de mise en oeuvre d'un projet collectif (élaboration des premiers diagrammes UML, flux des informations, répartition du travail...)

L'utilisation de SVN.

Commentaires personnels

APPRÉCIATION DU SUPÉRIEUR HIÉRARCHIQUE (CÉCILE)

Bon programmeur, qui posait beaucoup de questions aux professeurs. Il a compensé quelques absences par sa volonté de travailler sur le projet chez lui.

TRAVAIL PRÉLIMINAIRE

Participation aux discussions sur l'architecture du programme et solutions apportées :

Lors de mon court séjour dans le groupe 1 j'ai installé l'ensemble des ordinateurs pour le projet avec Romain Lanette pendant les discussions sur l'architecture du programme.

RÉALISATION DU PROGRAMME

Éléments réalisés:

En tant que chef du groupe, j'ai ébauché certains éléments de l'interface graphique comme le dessin de la ville et des voitures afin de faciliter la tâche des membres de mon équipe.

Travail supplémentaire réalisé sur le code :

Néant. La perfection est un état d'esprit...lol

Travaux complémentaires ou projets d'investissement personnel :

J'ai installé l'OS Ubuntu, l'IDE Eclipse et le plugin Visual Editor sur les machines.

IMPRESSIONS PERSONNELLES

Sur votre travail en général, au sein du groupe

J'ai trouvé mon travail très intéressant et très enrichissant. Ayant toute latitude pour gérer mon équipe, j'ai pris beaucoup de plaisir à planifier la partie interface graphique de ce projet et je constate avec satisfaction que le résultat n'est pas si mal. Cela dit, je tiens aussi à souligner que mon « management » n'aurait porté de fruit sans la collaboration complète des membres de mon équipe.

Ce que vous retiendrez du TP Long

Je pense que ce TP aura été de loin le plus intéressant. J'aurai pu y apprendre que le travail d'équipe exige des membres du sérieux et une volonté d'aller de l'avant que j'ai pu trouver dans mon équipe et ce malgrès l'accès possible à un célèbre réseau social fortement dissuasif.

Commentaires personnels

Il faudrait prévoir la prochaine fois une salle avec 4 m2 de surface par personne et des fauteuils parce que les tabourets c'est pas du boulot !

APPRÉCIATION DU SUPÉRIEUR HIÉRARCHIQUE (CÉDRIC)

Nicolas incarne le sous-chef de rêves, il dirige son équipe avec fougue et passion et ensemble ils ont su atteindre les objectifs qu'ont leur avait fixés, avec très peu de délais. Cet ambitieux s'est révelé dans ce cours en montrant de très bonnes aptitudes à diriger et à inspirer la confiance, attention toutefois à ne pas trop en faire, habituer les gens à l'excellence ne peut que les décevoir.

TRAVAIL PRÉLIMINAIRE

Participation aux discussion sur l'architecture du programme et solutions apportées :

J'ai proposé des idées pour l'interface et participé aux discutions préliminaires sur les tronçons et la gestion des voitures.

RÉALISATION DU PROGRAMME

Éléments réalisés:

J'ai été chargé de réaliser avec l'aide de Jean François l'ensemble de l'interface graphique (mis à part la carte) en visual editor. Cela va de la disposition de tous les éléments, au codage des actions réalisées par l'appui sur les boutons en passant par la réalisation graphique de ces boutons.

Travail supplémentaire réalisé sur le code :

J'ai fait fonctionner la fonction zoom et contribué à la réalisation de la main pour déplacer la carte. J'ai également participé au déboguage de certaines parties, notamment lors du rassemblement des fichiers.

Travaux complémentaires ou projets d'investissement personnel :

J'ai fait deux images pour chaque bouton pour que ceux-ci s'enfoncent quand on appuie dessus.

IMPRESSIONS PERSONNELLES

Sur votre travail en général, au sein du groupe

J'ai réalisé un travail efficace en remplissant mes objectifs. J'ai également réussi à assimiler les programmes des autres membres du groupe pour faire fonctionner le zoom par exemple. Le groupe etait motivé et dynamique ce qui à permis un travail en équipe agréable.

Ce que vous retiendrez du TP Long

De l'utilité de faire un travail propre lorsque l'on programme en groupe. En effet la mise en commun d'un fichier sur lequel tout le monde à travaillé de son coté avec ces méthodes de programmation est plutôt compliquée!

APPRÉCIATION DU SUPÉRIEUR HIÉRARCHIQUE (NICOLAS)

Sans aucun doute l'élément le plus actif et le plus investi de l'équipe de par ses connaissances en Java clairement meilleures que celles de ses collègues. Il possède par ailleurs une ardente polyvalence qui a su se faire apprécier tout au long du processus de développement.

Mohamed BAYOUMI, dit « Momo » — Membre équipe 2

TRAVAIL PRÉLIMINAIRE

Participation aux discussion sur l'architecture du programme et solutions apportées

J'ai travaillé sur la première ébauche de l'interface graphique après discussion avec le groupe. On m'a demandé de fournir un dessin clair de l'interface proposée même si celui-ci n'a pas été retenu après discussion avec le client.

RÉALISATION DU PROGRAMME

Éléments réalisés:

J'ai travaillé sur les classes *Cases*, *Forme*, *Map* et *Fenetre* pour mettre en place les méthodes qui dessinent les routes selon ce qu'on a choisi de dessiner (route simple, 2 routes perpendiculaires et la ville finale). J'ai également travaillé sur la méthode permettant de réaliser le zoom et sur celle implémentant la main qui déplace la carte. On m'a aussi demandé de travailler sur la méthode permettant d'afficher ce que nous envoie le groupe 1 (changement de la couleur des cases selon la présence ou non de voiture).

Travail supplémentaire réalisé sur le code :

J'ai dû reprendre plusieurs fois certaines méthodes car le projet avançant, des choses devaient être modifiées au fur et à mesure.

Travaux complémentaires ou projets d'investissement personnel :

J'ai été chargé de mettre en place le forum SERTR pour qu'on puisse se tenir au courant des avancées du projet et j'ai également réalisé un site FTP pour permettre à l'ensemble des membres du projet de s'échanger des fichiers. J'ai aussi crée un compte projet sur le Google Code pour lequel Romain s'est chargé d'y inscrire les membres en tant qu'utilisateurs.

IMPRESSIONS PERSONNELLES

Sur votre travail en général, au sein du groupe

Le travail au sein de notre groupe s'est très bien déroulé. La communication était sans faille. Tout le monde savait ce qu'il avait à faire et tout le monde s'aidait mutuellement. Par contre, la communication en amont n'était pas toujours optimale, ce qui a fait que des choses ont dû être changé en cours de route.

Ce que vous retiendrez du TP Long

Ce TP Long nous a permis de voir comment fonctionne le travail en équipe avec ses avantages et ses inconvénients. Il nous a également permis de prendre en main SVN qui est d'une grande nécessité dans ce genre de projet.

Commentaires personnels

Je pense que la réussite de ce projet a été un succès même si la finalité n'est pas encore, à ce jour, au rendez-vous.. Certaines choses qui nous semblaient faciles au premier abord ont été implémentées avec beaucoup de mal.

APPRÉCIATION DU SUPÉRIEUR HIÉRARCHIQUE (NICOLAS)

Bon élément de l'équipe, il sait apporter un travail de qualité. Un investissement exemplaire qui a su faire face aux difficultés rencontrées.

Travail préliminaire

Participation aux discussion sur l'architecture du programme et solutions apportées :

J'ai réalisé une ébauche d'interface graphique en expliquant les différentes fonctionnalités que j'y avais apportées de façon à mettre en commun mes idées avec celles des autres. J' ai fait de même pour les statistiques.

Réalisation du programme

Éléments réalisés:

J'ai participé à l'élaboration de l'interface graphique en m'occupant principalement de la partie sur les statistiques. Sinon j'ai apporté mon aide aux autres membres du groupe sur différents points du projet (tracé d'une route simple, implémentation des bouton pas à pas et marche arrêt).

Travail supplémentaire réalisé sur le code :

J'ai dû modifier toutes les variables de mon code sur les statistiques car leurs noms ou leurs caractéristiques différaient de ceux qui m'avaient été donné au tout début.

Travaux complémentaires ou projets d'investissement personnel :

J'ai récupéré la licence du RMI, qui n'a servi à rien puisque nous ne l'avons pas utilisée. J'ai également rédigé les compte-rendu des 4 premières réunions et réalisé un bilan des travaux réalisés et inachevés au cours de ce projet.

Impressions personnelles

Sur votre travail en général, au sein du groupe

J'ai essayé de faire à chaque fois ce qui m'était demandé par le chef. La cohésion de notre groupe a débouché sur une étroite collaboration qui nous a permis d'avancer rapidement dans l'élaboration de notre partie.

Ce que vous retiendrez du TP Long

La mise en commun finale s'est rendu des plus difficiles étant donné la mauvaise communication qui existait entre les deux groupes. A part ça, ce fut une expérience enrichissante de travailler au sein du groupe 2, tant au niveau de la conception que de l'organisation de notre projet.

Commentaires personnels

C'était une très bonne expérience malheureusement, ayant d'autres impératifs à côté, il n'était pas toujours évident de tenir les délais exigés.

Appréciation du supérieur hiérarchique (Nicolas)

Une bonne volonté apparente mais Eric aurait gagné à être plus rapide... Le travail fourni reste toutefois très correct.

Jean-François ROUYER — Membre équipe 2

TRAVAIL PRÉLIMINAIRE

Participation aux discussion sur l'architecture du programme et solutions apportées :

Réalisation de la première carte, à l'échelle, au format bitmap. Elle était utile pour s'apercevoir de la petitesse des voitures lorsque la carte était affichée entièrement à l'écran.

RÉALISATION DU PROGRAMME

Éléments réalisés:

Elaboration de la classe Temporisateur avec Laurent,

Elaboration de l'Interface graphique avec Laurent. (création des panneaux droits 1 et 2, construction du panneau flux entrée, construction du popup, connection des différents panels, rattachement de la carte sur l'interface graphique...)

Travail supplémentaire réalisé sur le code :

Débuggage du système de la main (affichage de telle ou telle partie de la carte suivant les coordonnées souris)

Tentative d'incorporation de la carte au format bitmap dans le code avec Nicolas

Travaux complémentaires ou projets d'investissement personnel :

Débuggage du système de la main. Ce dernier fonctionnait, mais pas correctement. Le système ressemble désormais plus à une main, tel que l'on peut en trouver dans acrobat reader. Pour plus de commodité, nous n'avons pas inverser la souris.

IMPRESSIONS PERSONNELLES

Sur votre travail en général, au sein du groupe

Travailler à plusieurs a été une bonne expérience. Nous avons pu voir quelles étaient les forces et faiblesses de chacun, et en tirer parti afin d'avancer au plus vite. Nous avons ainsi changer de chef car Nicola s'y connaissait beaucoup plus que nous tous en interface graphique en java. L'inconvénient de travailler à plusieurs a été de reprendre le code d'autres personnes, code dans lequel il est parfois difficile de se plonger..

En ce qui concerne le travail, on nous a demandé plusieurs fois de refaire des parties de code, car cela ne convenait plus à l'autre groupe (notament en ce qui concerne le tracé des 3 cartes d'essai).

Travailler sur plusieurs ordinateurs a été difficile : les transferts de fichier entre windows et unix ne s'opérant pas correctement.

Ce que vous retiendrez du TP Long

Je sais désormais comment réaliser une interface graphique en java, grâce à Visual editor. De plus, avoir utilisé Eclipse nous a permis de développer dans un univers professionnel.

Commentaires personnels

Le problème a été la communication. Ne possédant pas Internet chez moi, j'ai raté une réunion projet, programmée du jour pour le lendemain.

APPRÉCIATION DU SUPÉRIEUR HIÉRARCHIQUE (NICOLAS)

Un tempérament de feu qui masque parfois des difficultés de compréhension. Toutefois le travail a été effectué sérieusement.