



INSA Lyon
20, avenue Albert Einstein
69621 Villeurbanne Cedex

LIVRABLE DE PROJET

Développement Orienté Objet

du 27 novembre au 20 décembre 2013



Hexanôme H4404 :

Guillaume ABADIE
Nicolas BUISSON
Louise CRÉPET
Rémi DOMINGUES
Aline MARTIN
Martin WETTERWALD

Enseignants :

Christine SOLNON
Elöd EGYED-ZSIGMOND

Année scolaire 2013-2014

Sommaire

Introduction	1
1 Capture et analyse des besoins	2
1.1 Rôles	2
1.2 Planning prévisionnel du projet	2
1.3 Modèle du domaine	4
1.4 Diagramme de cas d'utilisation	5
1.5 Description abrégée des cas d'utilisation	5
2 Conception	6
2.1 Description détaillée des cas d'utilisation	6
2.1.1 Charger un plan	6
2.1.2 Charger une demande de livraison	7
2.1.3 Calculer une tournée	7
2.1.4 Visualiser une tournée	8
2.1.5 Générer d'une feuille de route	8
2.1.6 Modifier une tournée en temps réel	9
2.1.7 Pré-modifier une tournée	10
2.2 Diagrammes de packages et de classes	11
2.2.1 Package Modèle (Model)	11
2.2.2 Package Vue (View)	15
2.2.3 Package Controlleur (Controller)	17
2.3 Diagrammes de séquences	19
2.3.1 Chargement d'une tournée	19
2.3.2 Calcul d'une tournée	21
2.3.3 Selection d'un noeud de la map	22
2.3.4 Ajout d'une livraison	23
2.3.5 Suppression d'une livraison	24
3 Implementation	25

3.1	Captures d'écran de l'application	25
3.2	Diagrammes rétrogénérés de packages et de classes	28
3.2.1	Architecture générale de l'application	28
3.2.2	Package Utilitaires (Utils)	29
3.2.3	Package Modèle (Model)	30
3.2.4	Package Vue (View)	34
3.2.5	Package Controlleur (Controller)	36
4	Bilans	38
4.1	Planning effectif du projet	38
4.2	Bilan humain	42
4.2.1	Méthodologie	42
4.2.2	Respect du planning et adaptations	42
4.2.3	Ressenti	42
4.3	Bilan technique	43
4.3.1	Sujet	43
4.3.2	Compétences acquises	43
	Glossaire	44

Introduction

Ce dossier a pour but de vous présenter le système de gestion des livraisons du Grand Lyon, Opti_fret_COURLY. Nous nous attarderons ici plus spécifiquement sur l'interface du superviseur, qui lui permet de gérer les demandes de livraisons des clients utilisant le système.

Vous retrouverez dans ce dossier la démarche détaillée de la capture et de l'analyse des besoins, ainsi que la description complète de la conception de l'application. Nous espérons que notre système Optifret_COURLY vous donnera une entière satisfaction.

1. Capture et analyse des besoins

1.1 Rôles

ABADIE Guillaume Concepteur / Développeur

BUISSON Nicolas Concepteur / Développeur

CREPET Louise Concepteur / Développeur

DOMINGUES Rémi Chef de projet / Concepteur / Développeur

MARTIN Aline Responsable qualité / Concepteur / Développeur

WETTERWALD MARTIN Responsable Git / Concepteur / Développeur

1.2 Planning prévisionnel du projet

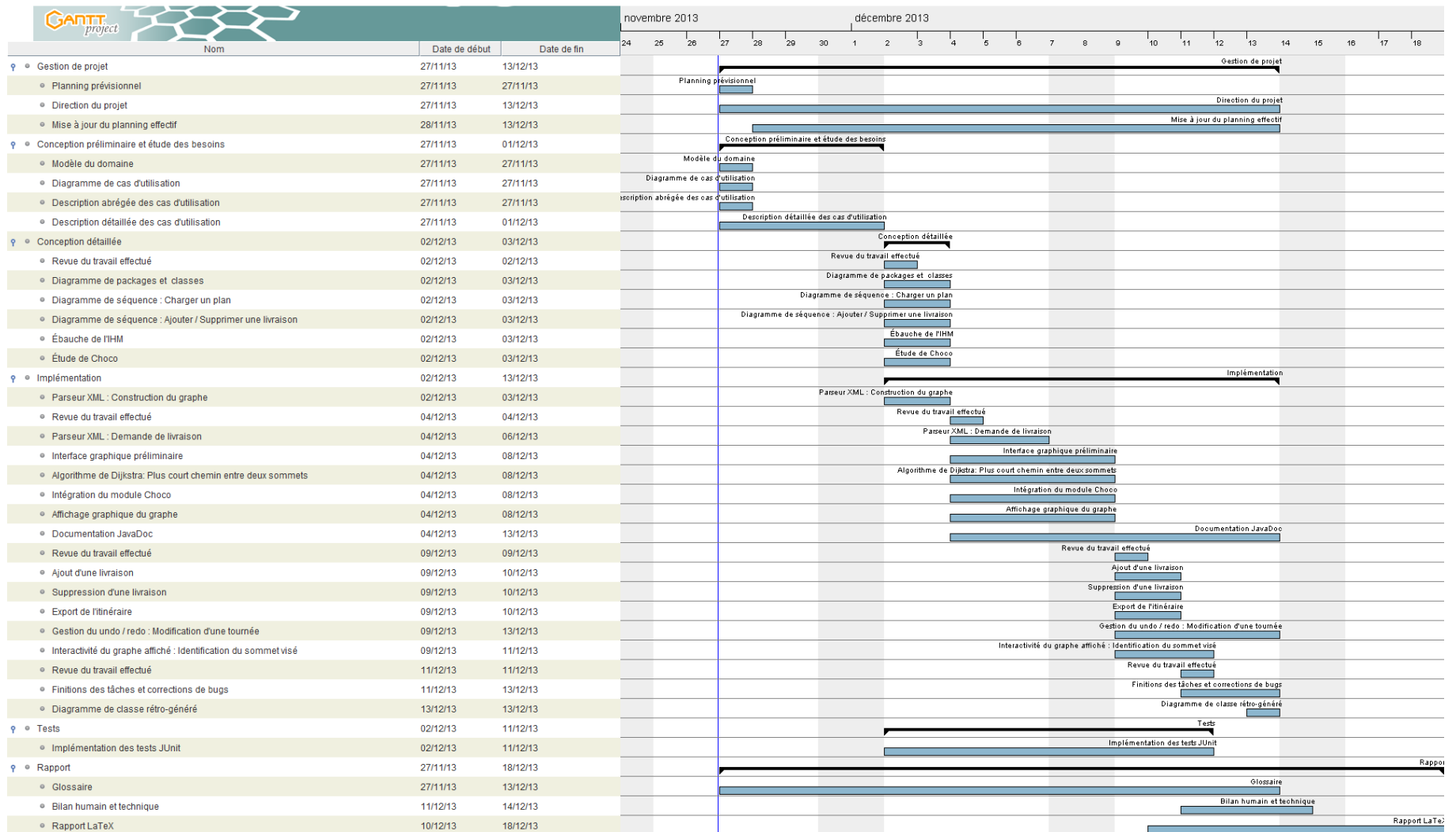


FIGURE 1.1 – Planning prévisionnel du projet

1.3 Modèle du domaine

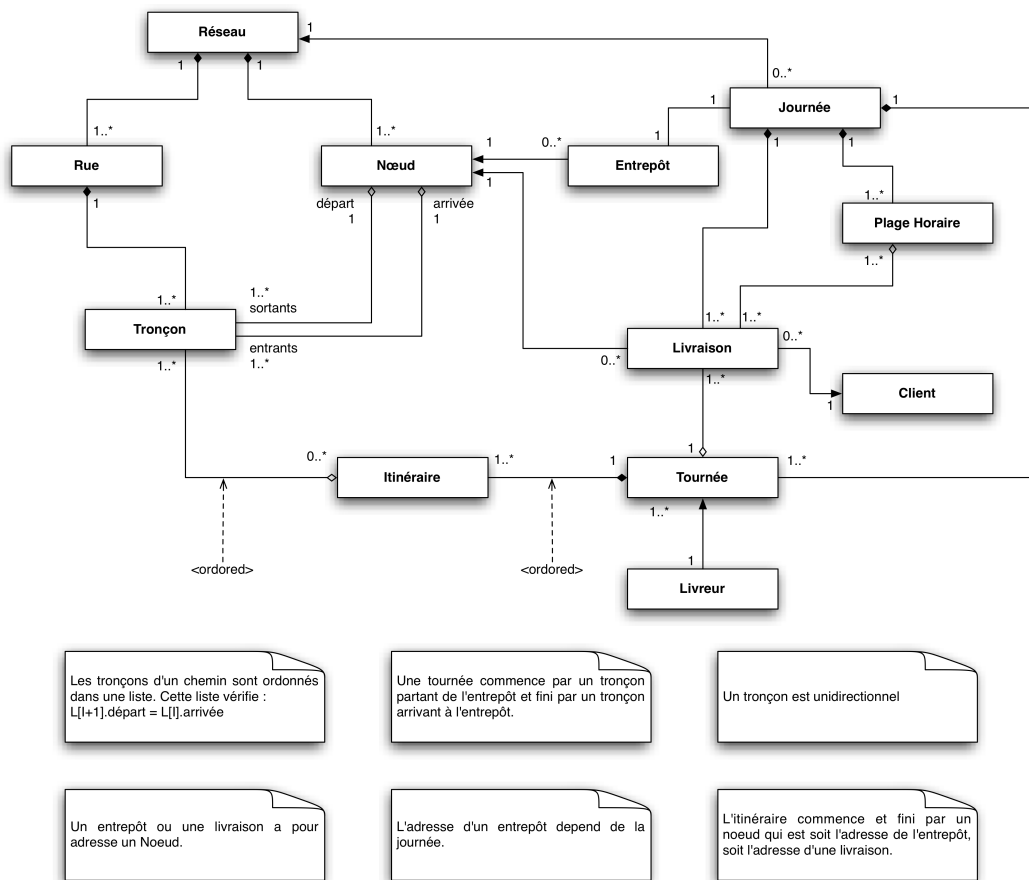


FIGURE 1.2 – Modèle du domaine

1.4 Diagramme de cas d'utilisation

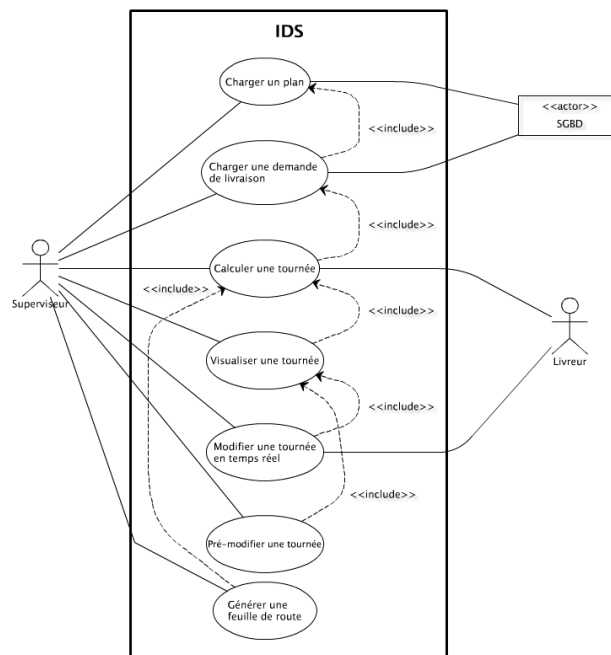


FIGURE 1.3 – Diagramme de cas d'utilisation

1.5 Description abrégée des cas d'utilisation

Le système doit permettre plusieurs ensembles d'actions :

- Le superviseur peut à tout moment visualiser le plan d'une zone de la ville, il peut alors y voir les points de livraison de la zone, ainsi que leur détails (adresse, état de livraison, etc ...).
- D'autre part, il peut charger une demande de livraison, celle ci est ajoutée au système et sera traitée dans les futures tournées.
- Enfin il peut générer une feuille de route multi-support (papier et électronique) à la destination d'un des livreurs. Pour cela il peut demander au système de calculer une tournée, et de la visualiser sur un plan. Il peut alors y faire d'éventuelles modifications avant la génération de la feuille et lancer cette dernière.
- Une fois la feuille de route générée, le superviseur pourra à tout moment modifier une tournée en cours. Le livreur de cette tournée modifiée en sera alors informé par le système.

2. Conception

2.1 Description détaillée des cas d'utilisation

2.1.1 Charger un plan

Contexte : le superviseur veut visualiser le plan d'une zone de la ville

Acteur principal : Superviseur

Precondition : -

Postconditions : Le plan doit être affiché

Scenario principal :

1. Le superviseur demande le chargement d'un plan pour une zone de la ville
2. Le système demande au SGBD si la zone existe bien, ainsi que les données correspondantes
 - (a) La zone n'existe pas :
 - i. Le système renvoie une erreur, le CdU reprend à l'étape 1, le cas d'utilisation se termine par un échec.
 - (b) Erreur syntaxique, le cas d'utilisation se termine par un échec.
 - (c) Erreur sémantique, le cas d'utilisation se termine par un échec.
3. Le système affiche le plan à l'écran, le cas d'utilisation se termine par un succès.

2.1.2 Charger une demande de livraison

Contexte : Le superviseur veut charger une demande de livraison, c'est-à-dire un ensemble de points à livrer avec, pour chaque point, sa plage horaire

Acteur principal : Superviseur

Precondition : Le plan doit être chargé

Postconditions : Les points doivent être affichés sur le plan

Scenario principal :

1. Le superviseur sélectionne une demande de livraison
2. Le système demande au SGBD si la demande de livraison existe bien, ainsi que les données correspondantes
 - (a) La demande de livraison n'existe pas :
 - i. Le système renvoie une erreur, le CdU reprend à l'étape 1, le cas d'utilisation se termine par un échec.
 - (b) Erreur syntaxique, le cas d'utilisation se termine par un échec.
 - (c) Erreur sémantique, le cas d'utilisation se termine par un échec.
3. Le système affiche les points de livraison sur le plan, le cas d'utilisation se termine par un succès.

2.1.3 Calculer une tournée

Contexte : Le superviseur veut calculer une tournée pour la dernière demande de livraisons chargée

Acteur principal : Superviseur

Precondition : Une tournée doit être sélectionnée, les demandes de livraisons doivent être chargées

Postconditions : -

Scenario principal :

1. Le superviseur clique sur "calculer tournée" pour la demande de livraison
2. Le système calcule la tournée et l'envoie au SGBD
 - (a) Il n'est pas possible de tout faire dans les temps, le système notifie l'utilisateur qu'il y aura forcément des retards, le cas d'utilisation se termine par un succès.
 - (b) Une livraison n'est pas accessible, le cas d'utilisation se termine par un échec.
3. Le système affiche la tournée sur le plan, le cas d'utilisation se termine par un succès.

2.1.4 Visualiser une tournée

Contexte : Le superviseur veut visualiser une tournée sur un plan avec les différents états des livraisons (fait, à l'heure, en retard, etc.). Possibilité de visualiser les détails de la tournée.

Acteur principal : Superviseur

Precondition : La tournée existe et est dans une liste de tournées disponibles

Postconditions : La tournée existe, est dans une liste de tournées disponibles, est sélectionnée et est affichée

Scenario principal :

1. Le superviseur clique sur la tournée dans la liste
2. La tournée s'affiche dans la zone prévue à cet effet

Extension :

1. Le superviseur clique sur "détail"
 - (a) Les informations complémentaires s'affichent dans la zone prévue à cet effet, le cas d'utilisation se termine par un succès.
 - (b) Les informations par défaut s'affichent si la tournée est vide, le cas d'utilisation se termine par un succès.

2.1.5 Générer d'une feuille de route

Contexte : Le superviseur veut générer une feuille de route d'un livreur contenant les itinéraires de chaque livraison

Acteur principal : Superviseur

Precondition : La tournée du livreur est prête.

Postconditions : La feuille de route a été générée et la version papier imprimée

Scenario principal :

1. Le superviseur clique sur "générer feuille de route"
2. La feuille de route est générée et sauvegardée, le cas d'utilisation se termine par un succès.
 - (a) Pas de livraison pour la tournée
 - (b) Problème avec l'ordinateur pendant la sauvegarde
 - (c) Le fichier existe déjà dans les sauvegardes
 - i. Ecraser l'ancien, le cas d'utilisation se termine par un succès.
 - ii. Annuler le nouveau, le cas d'utilisation se termine par un échec.
 - iii. Renommer l'un des deux, le cas d'utilisation se termine par un succès.

2.1.6 Modifier une tournée en temps réel

Contexte : Le superviseur veut modifier une tournée le jour même en ajoutant ou retirant une livraison. Il doit être possible d'annuler la dernière action, mais aussi de la refaire.

Acteur principal : Superviseur

Preconditions :

Générale : Il y a une tournée sélectionnée, il est possible de communiquer avec le livreur

Pour ajouter : Il y a au moins un noeud libre sur le plan

Pour enlever : Il y a au moins une livraison dans la tournée

Postconditions :

Générale : La tournée sélectionnée est modifiée et les changements ont été envoyés au livreur

Pour ajouter : La livraison est ajoutée à la tournée à la bonne place

Pour enlever : La livraison est enlevée de la tournée qui est automatiquement recalculée

Scenario principal :

Ajouter

1. Le superviseur sélectionne un noeud disponible
2. Le superviseur clique sur "ajouter" et remplit le formulaire d'ajout (plage horaire, client, etc.)
3. Le superviseur clique sur "annuler" pendant qu'il remplit le formulaire
 - (a) L'ajout est annulé et le superviseur perd ce qu'il avait pré-rempli, le cas d'utilisation se termine par un échec.
4. Le superviseur clique sur "ok"
5. La livraison est ajoutée et la tournée recalculée
6. L'ajout est envoyé au livreur, le cas d'utilisation se termine par un succès.
 - (a) La connexion au livreur échoue, le cas d'utilisation se termine par un échec.

Enlever

1. Le superviseur sélectionne une livraison sur la tournée
2. Le superviseur clique sur "enlever"
3. La livraison est enlevée et la tournée recalculée
 - (a) La livraison est enlevée et la tournée est vide, le système continue à l'étape enlever :4
4. La suppression est envoyée au livreur, le cas d'utilisation se termine par un succès.
 - (a) La connexion au livreur échoue, le cas d'utilisation se termine par un échec.

2.1.7 Pré-modifier une tournée

Contexte : Le superviseur veut modifier une tournée la veille en ajoutant ou retirant une livraison. Il doit être possible d'annuler la dernière action, mais aussi de la refaire.

Acteur principal : Superviseur

Preconditions :

Générale : Il y a une tournée sélectionnée, il est possible de communiquer avec le livreur

Pour ajouter : Il y a au moins un noeud libre sur le plan

Pour enlever : Il y a au moins une livraison dans la tournée

Postconditions :

Générale : La tournée sélectionnée est modifiée

Pour ajouter : La livraison est ajoutée à la tournée à la bonne place

Pour enlever : La livraison est enlevée de la tournée qui est automatiquement recalculée

Scenario principal :

Ajouter

1. Le superviseur selectionne un noeud disponible
2. Le superviseur clique sur "ajouter" et remplit le formulaire d'ajout (plage horaire, client). La plage horaire se decompose en deux champs chiffrés qui doivent être cohérents avec la notion de plage horaire
3. Le superviseur clique sur "annuler" pendant qu'il remplit le formulaire
 - (a) L'ajout est annulé et le superviseur perd ce qu'il avait pré-rempli, le cas d'utilisation se termine par un échec.
4. Le superviseur clique sur "ok"
 - (a) Une aberration est dans la livraison à ajouter, le système signale une erreur, le cas d'utilisation se termine par un échec.
5. La livraison est ajoutée et la tournée recalculée, le cas d'utilisation se termine par un succès.

Enlever

1. Le superviseur selectionne une livraison sur la tournée
2. Le superviseur clique sur "enlever"
3. La livraison est enlevée et la tournée recalculée, le cas d'utilisation se termine par un succès.
 - (a) La livraison est enlevée et la tournée est vide, le cas d'utilisation se termine par un succès.

2.2 Diagrammes de packages et de classes

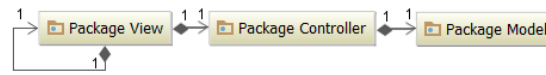


FIGURE 2.1 – Diagramme UML de l'architecture générale

2.2.1 Package Modèle (Model)

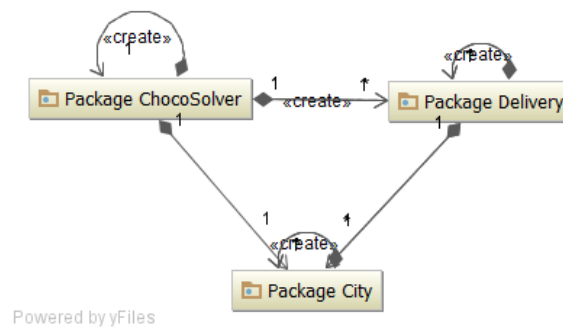
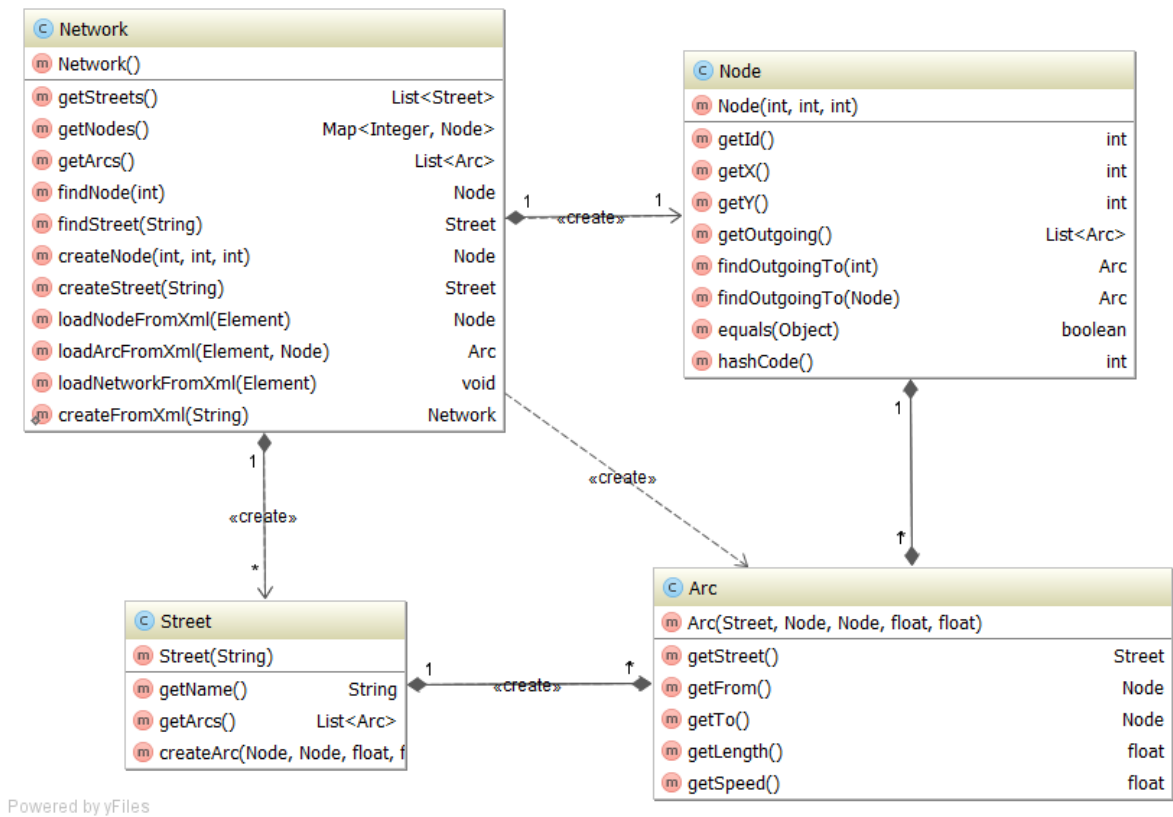


FIGURE 2.2 – Diagramme UML du package Model

Package Modèle.Ville (Model.City)



Powered by yFiles

FIGURE 2.3 – Diagramme UML du package Model.City

Package Modèle.Livraison (Model.Delivery)

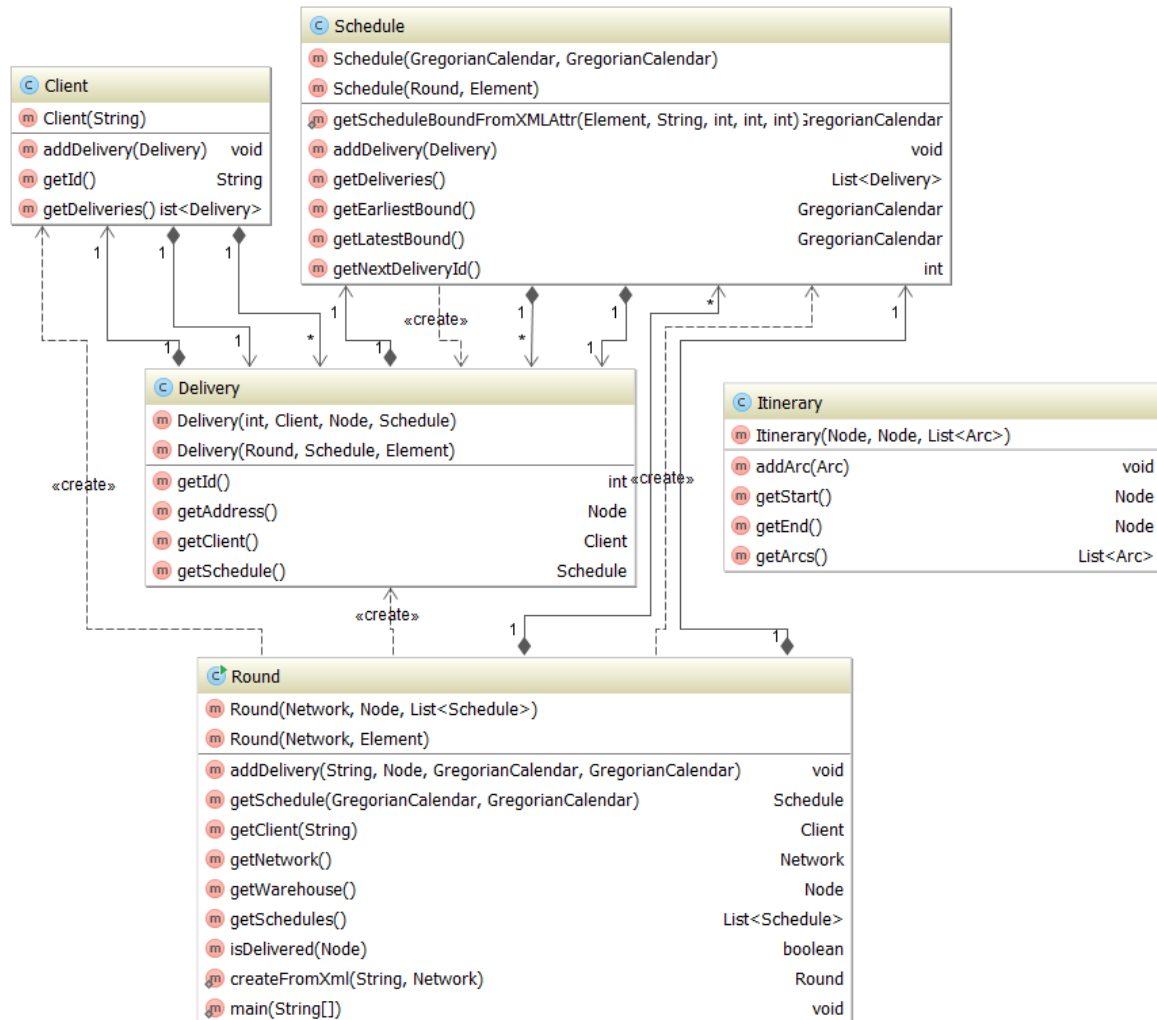


FIGURE 2.4 – Diagramme UML du package Model.Delivery

Package Modèle.ChocoSolver (Model.ChocoSolver)

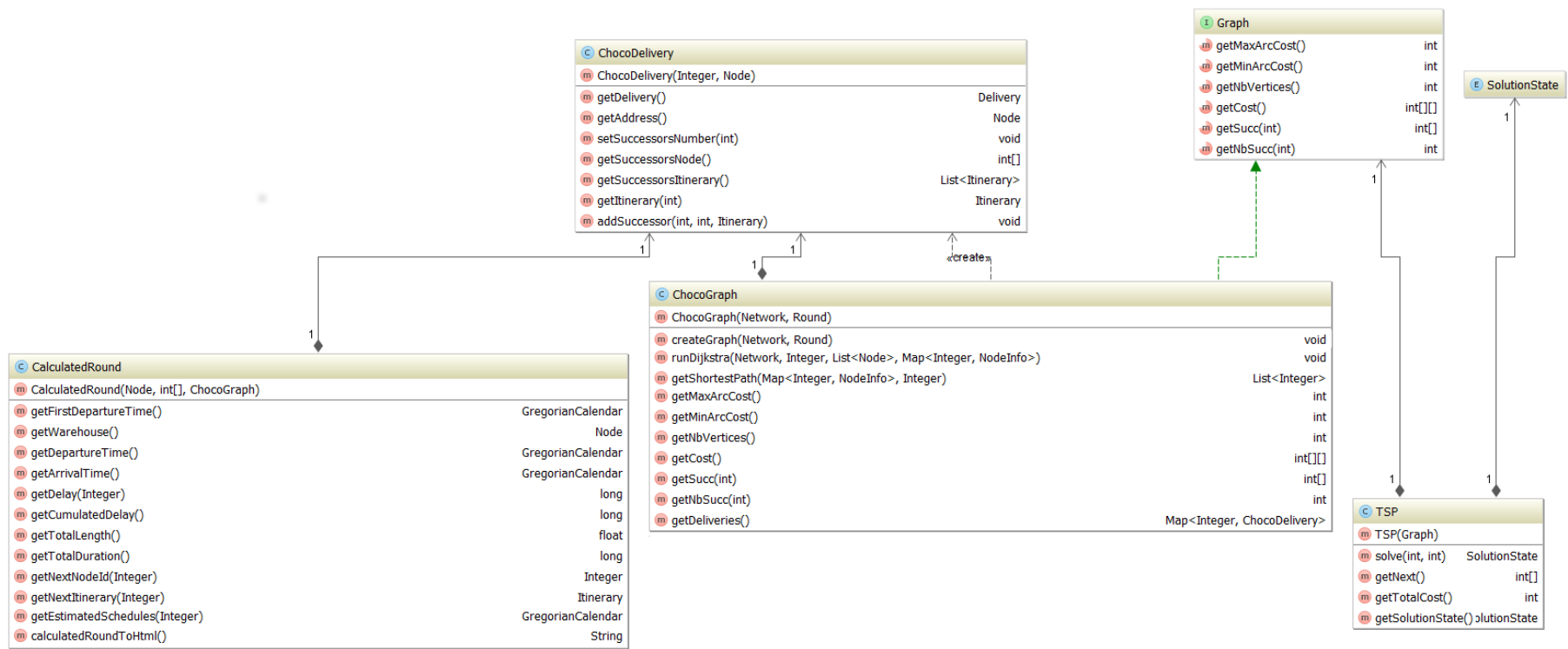


FIGURE 2.5 – Diagramme UML du package Model.ChocoSolver

2.2.2 Package Vue (View)

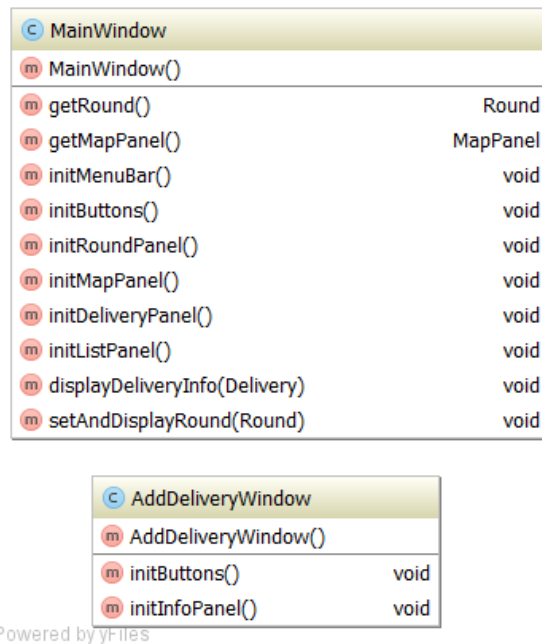
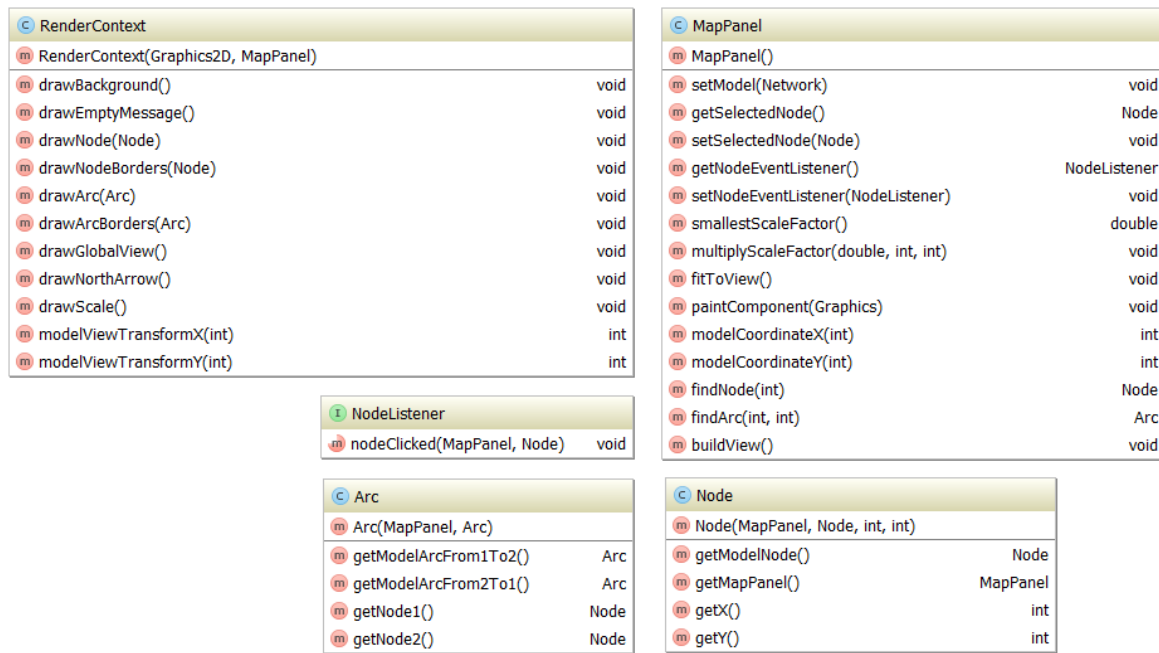


FIGURE 2.6 – Diagramme UML du package View

Package Vue.Carte (View.MapPanel)



Powered by yFiles

FIGURE 2.7 – Diagramme UML du package View.MapPanel

2.2.3 Package Contrôleur (Controller)

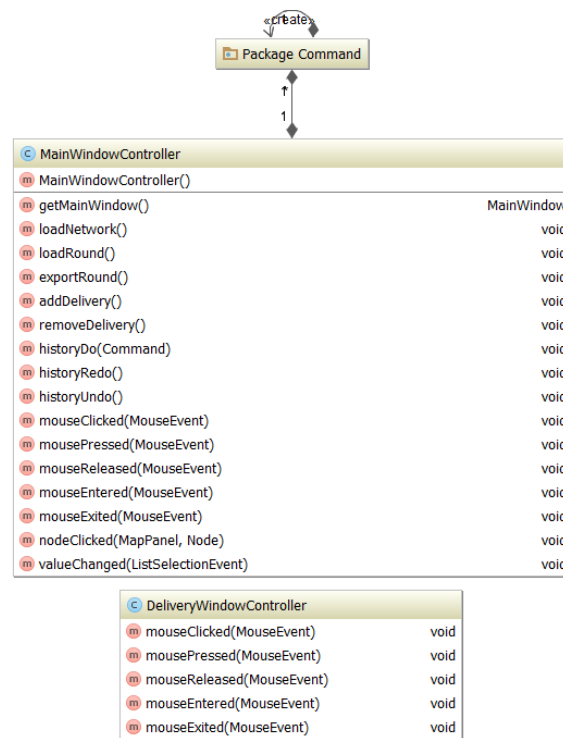


FIGURE 2.8 – Diagramme UML du package Contrôleur

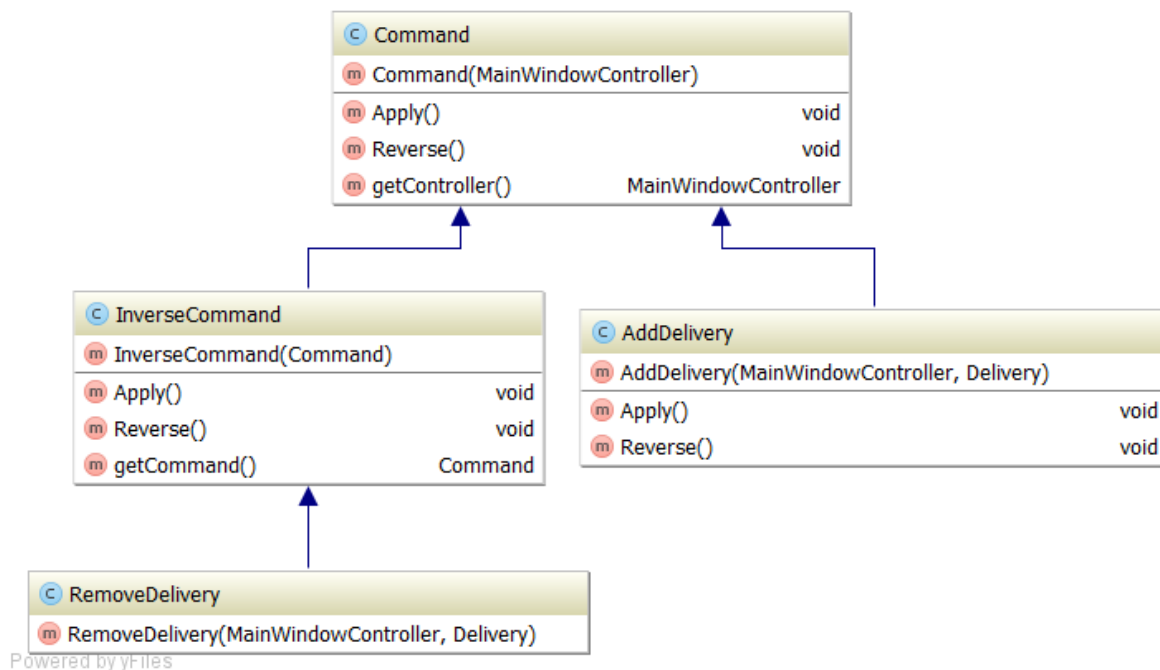
Package Controleur.Commande (Controller.Command)

FIGURE 2.9 – Diagramme UML du package Controller.Command

Ce package implémente le design pattern Commande afin de gérer le undo/redo pour l'ajout et suppression d'une livraison dans la tournée.

On remarque alors que les classes `AddDelivery` et `RemoveDelivery` ont un code pour les méthodes `Apply()` et `Reverse()` semblable à la différence qu'ils sont inversés (la fonction `Apply()` de `RemoveDelivery` est identique à `Reverse()` de `AddDelivery` par exemple). Dans un but de factorisation de code pour l'ajout et la suppression, nous avons donc choisi d'implémenter la classe `InverseCommand`. Celle-ci est donc une commande effectuant les opérations inverses d'une autre qu'on lui passe à l'instanciation.

Ainsi nous pouvons implémenter `RemoveDelivery` en faisant simplement l'inverse de la commande `AddDelivery`. Autrement dit, pour supprimer une livraison, il suffit de la "désajouter".

2.3 Diagrammes de séquences

2.3.1 Chargement d'une tournée

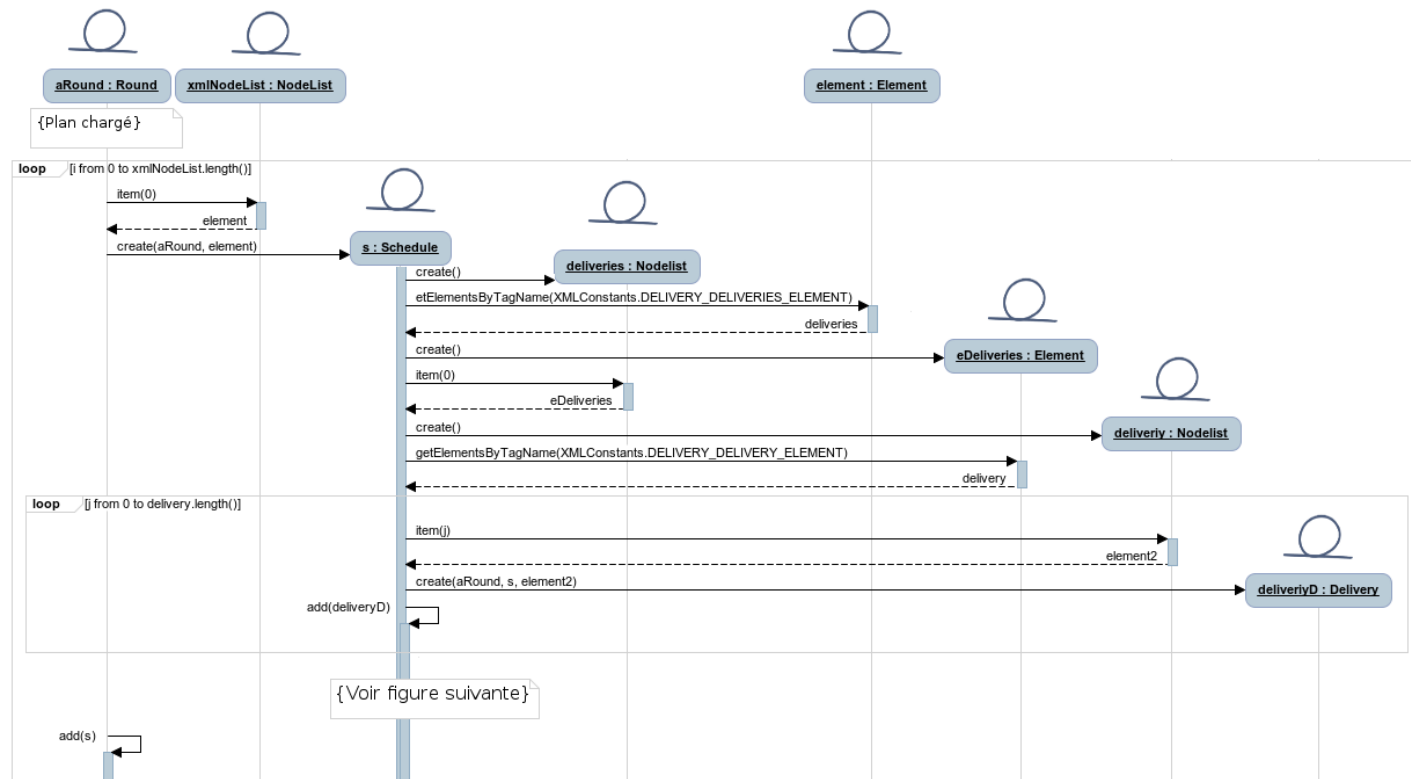


FIGURE 2.10 – Diagramme de sequence du chargement de la tournée

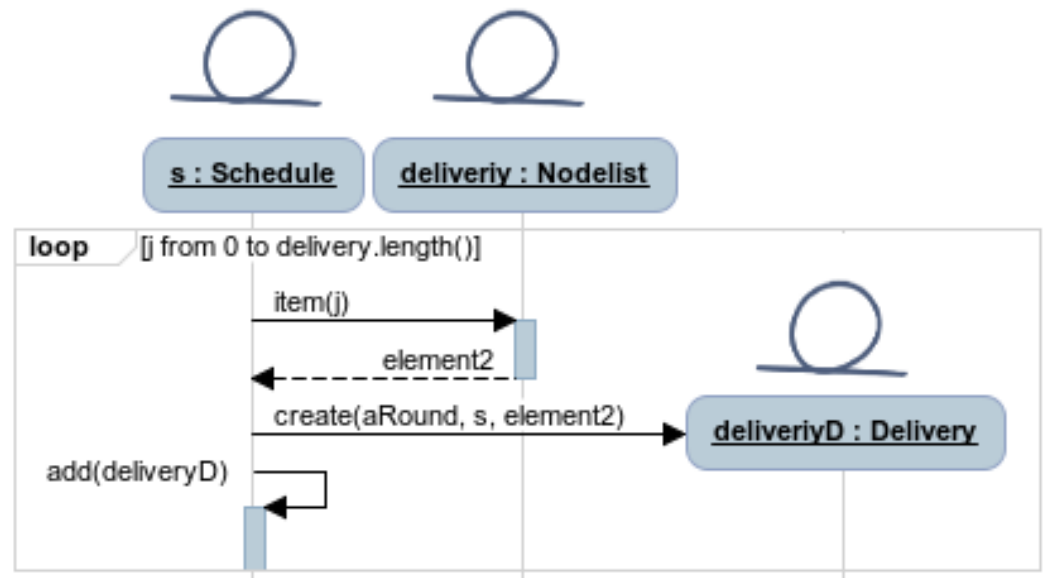


FIGURE 2.11 – Additif au diagramme de sequence du chargement de la tournée

2.3.2 Calcul d'une tournée

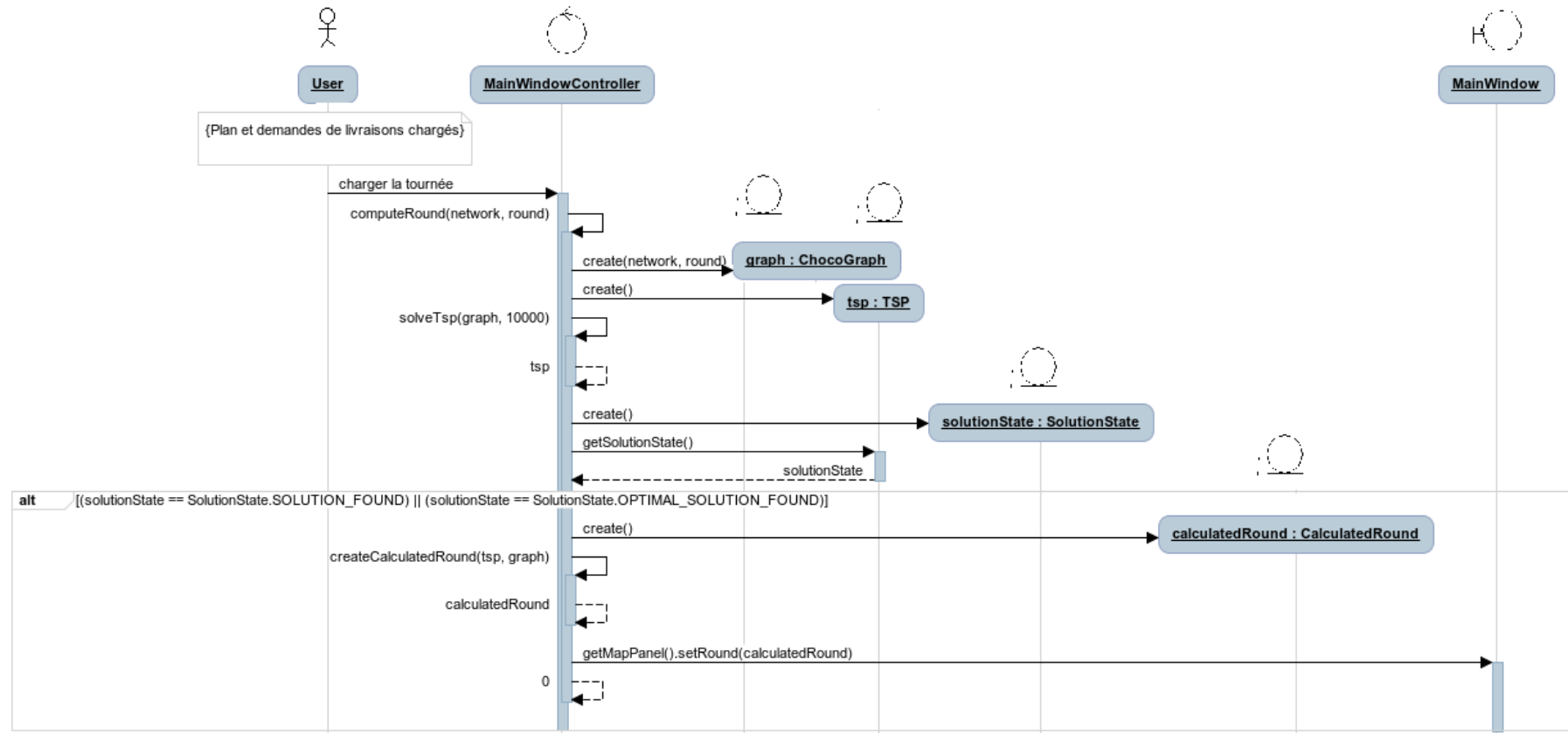


FIGURE 2.12 – Diagramme de séquence de calcul de la tournée

2.3.3 Selection d'un noeud de la map

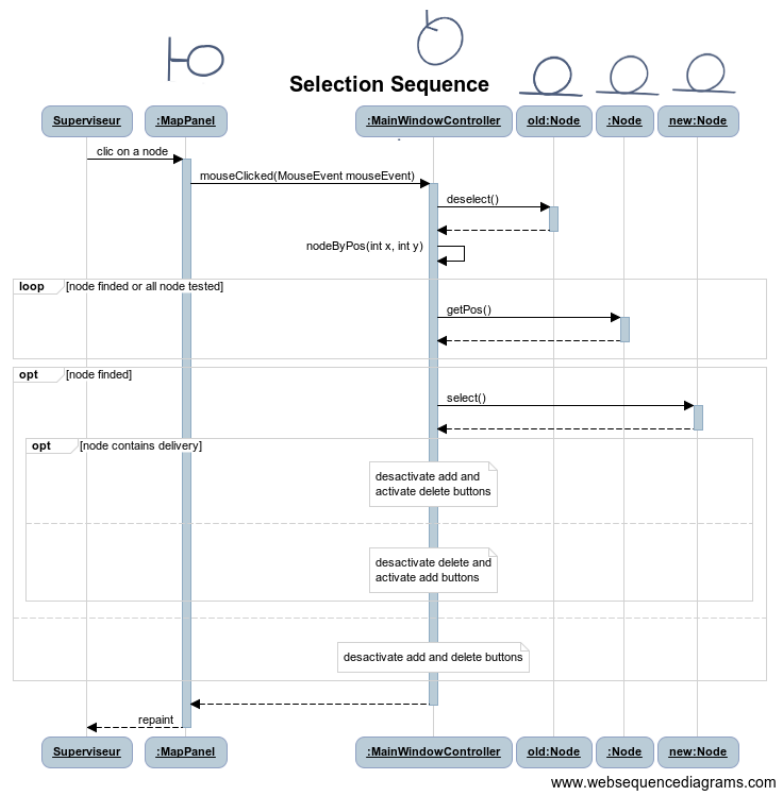


FIGURE 2.13 – Diagramme de sequence d'ajout d'une livraison

2.3.4 Ajout d'une livraison

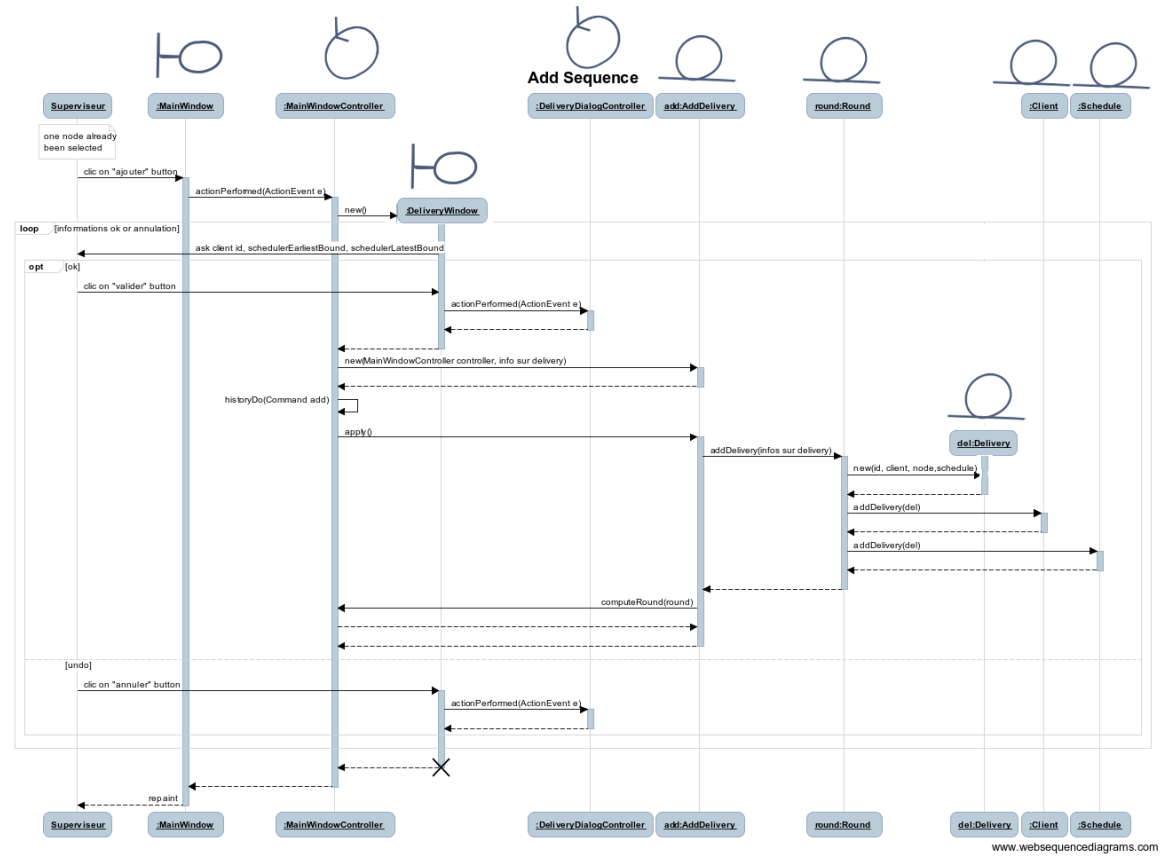


FIGURE 2.14 – Diagramme de sequence d'ajout d'une livraison

2.3.5 Suppression d'une livraison

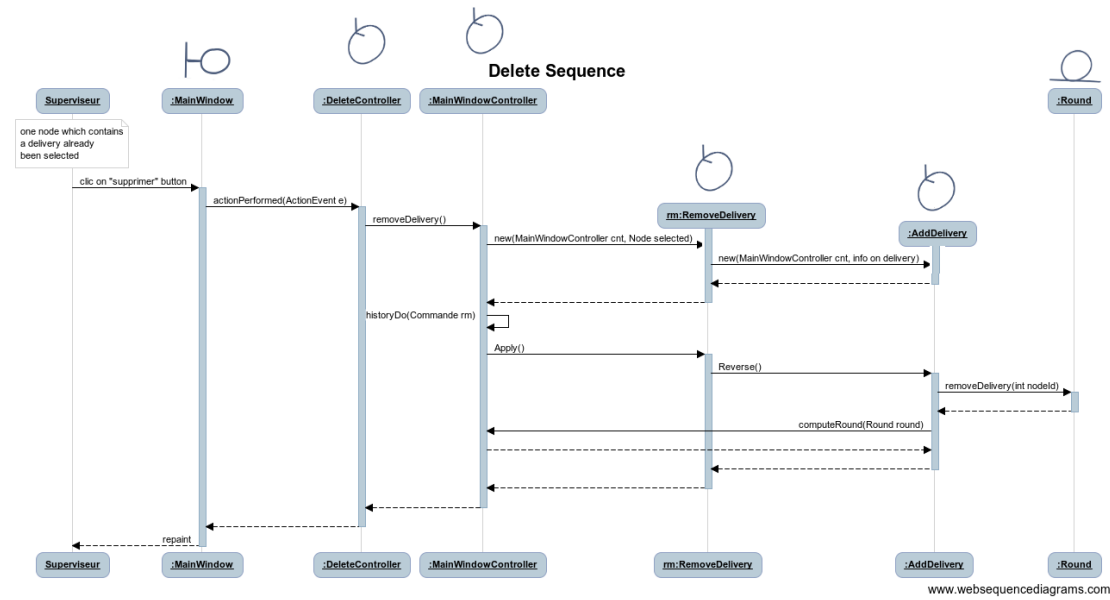


FIGURE 2.15 – Diagramme de sequence de suppression d'une livraison

3. Implementation

3.1 Captures d'écran de l'application

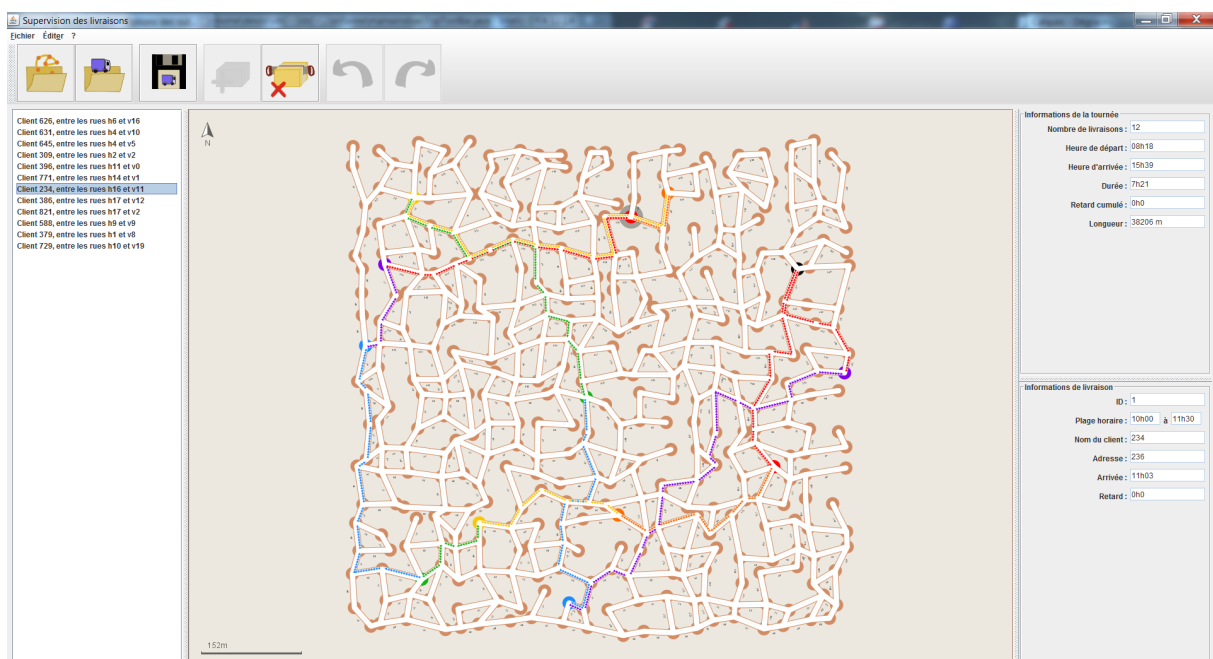


FIGURE 3.1 – Capture d'écran de la fenêtre principale

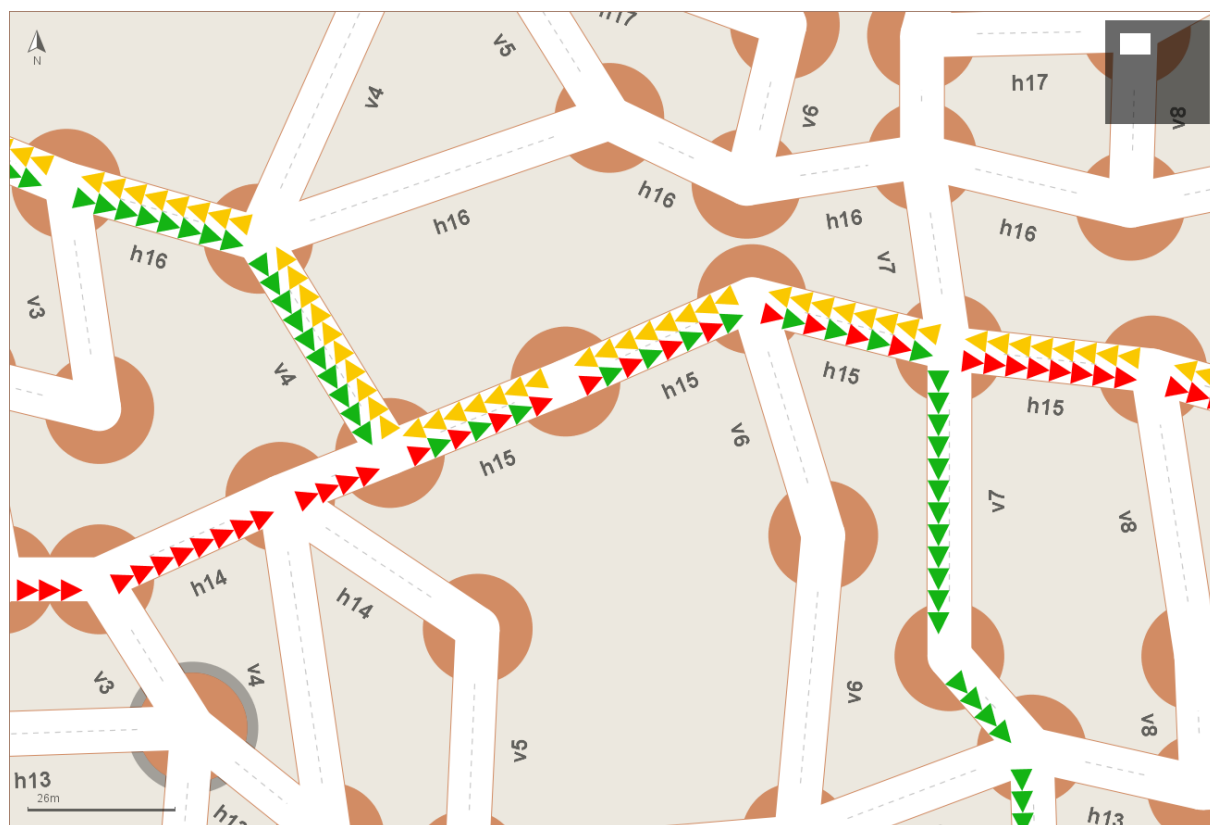
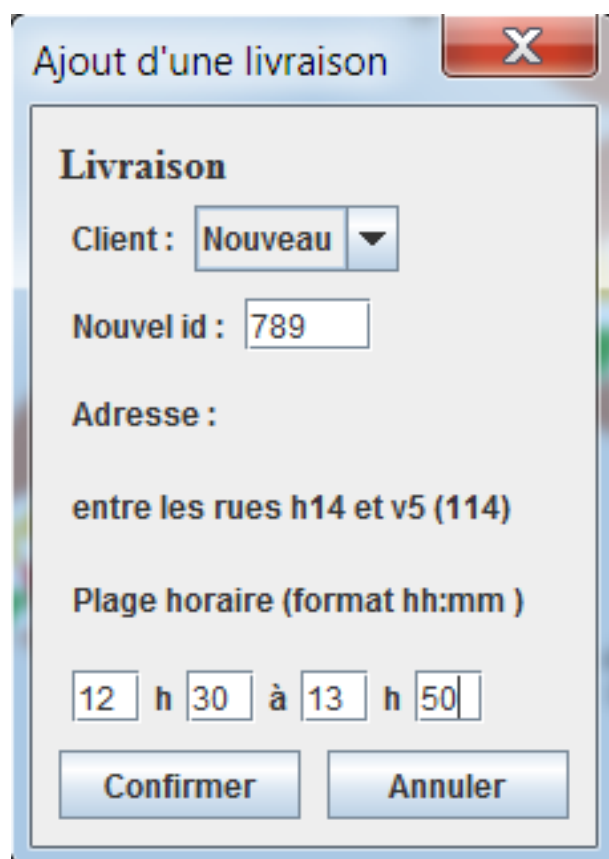


FIGURE 3.2 – Zoom sur le graphe



Ajout d'une livraison [X]

Livraison

Client : Nouveau ▼

Nouvel id : 789

Adresse :

entre les rues h14 et v5 (114)

Plage horaire (format hh:mm)

12 h 30 à 13 h 50

Confirmer Annuler

FIGURE 3.3 – Fenêtre d'ajout d'une livraison

3.2 Diagrammes rétrogénérés de packages et de classes

3.2.1 Architecture générale de l'application

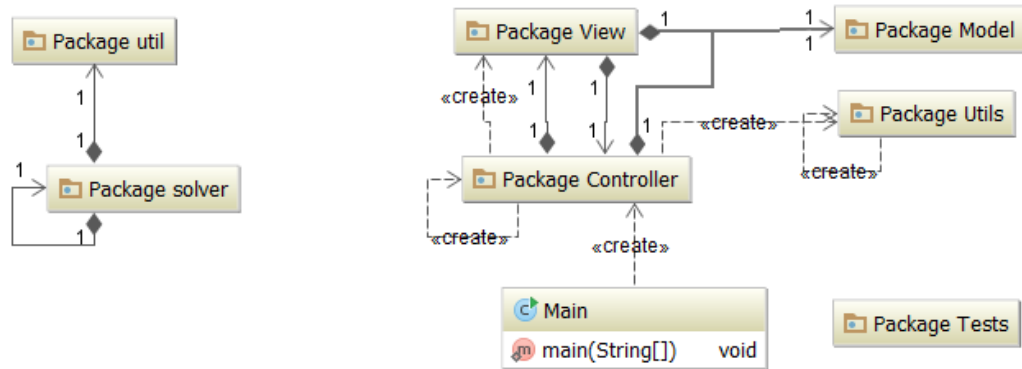


FIGURE 3.4 – Diagramme rétrogénéré UML de package principale

3.2.2 Package Utilitaires (Utils)

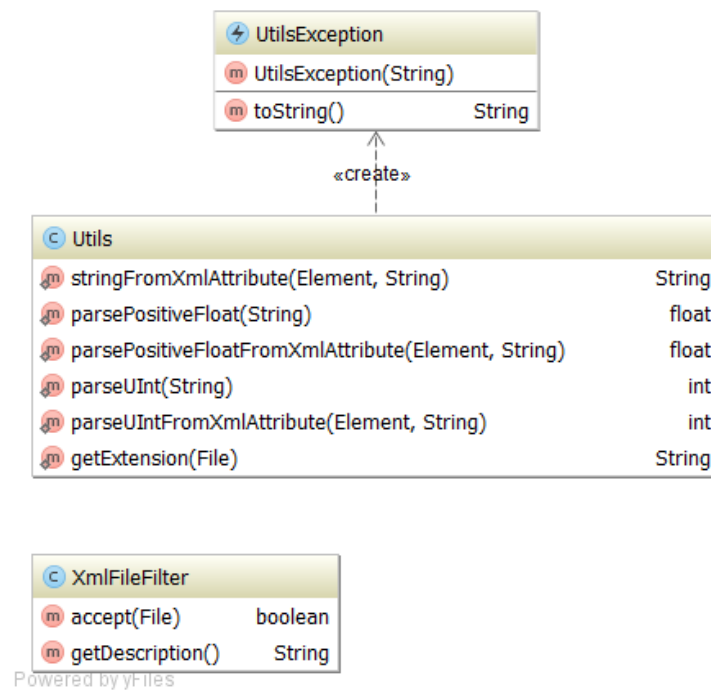


FIGURE 3.5 – Diagramme rétrogénéré UML du package Utils

3.2.3 Package Modèle (Model)

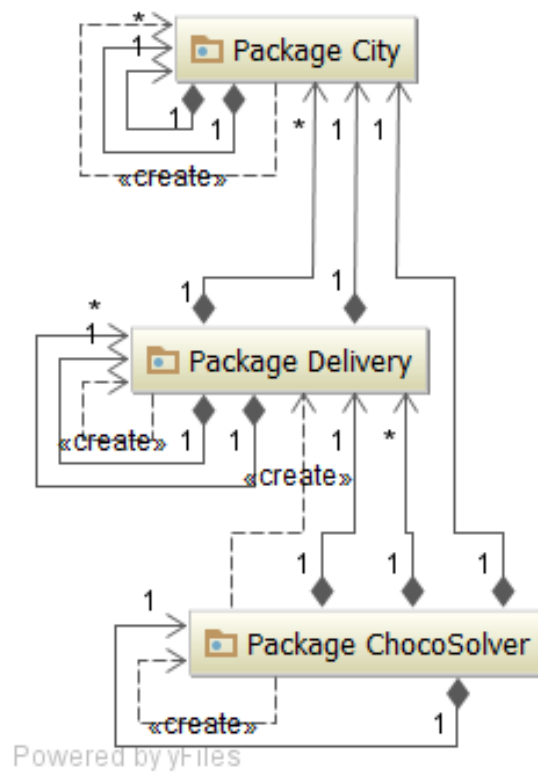
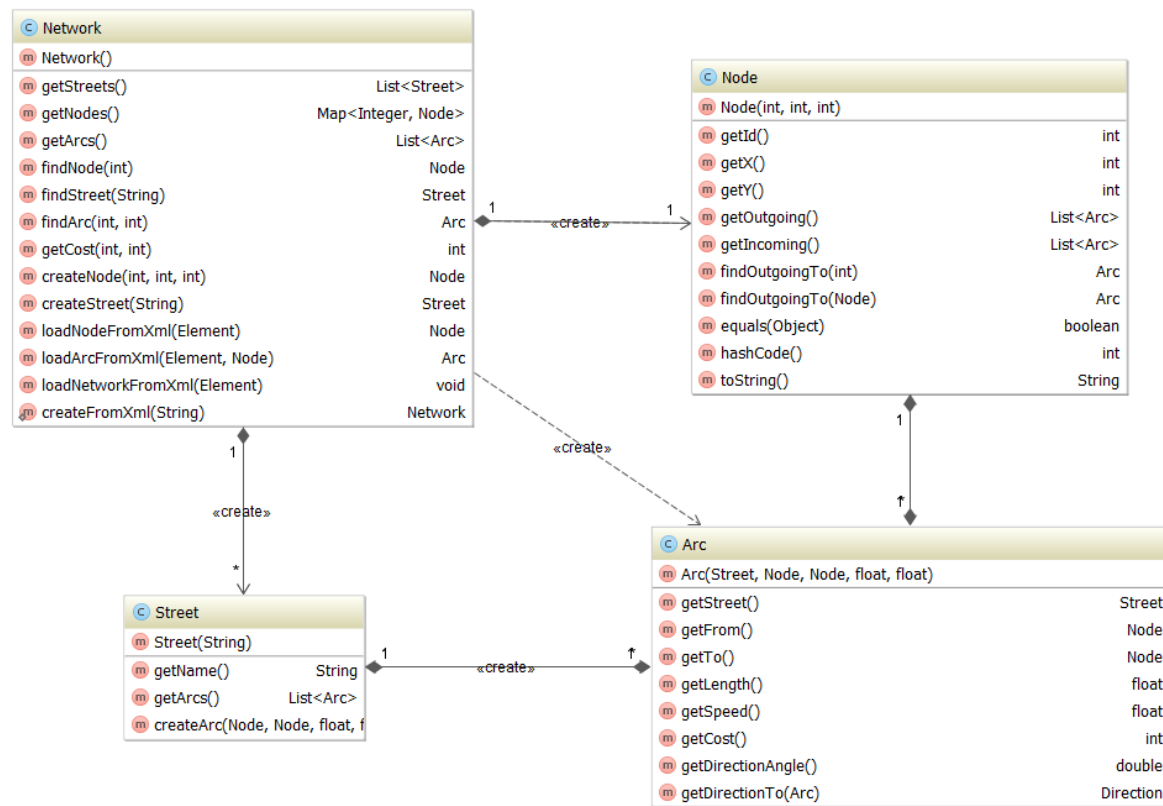


FIGURE 3.6 – Diagramme rétrogénéré UML du package Model

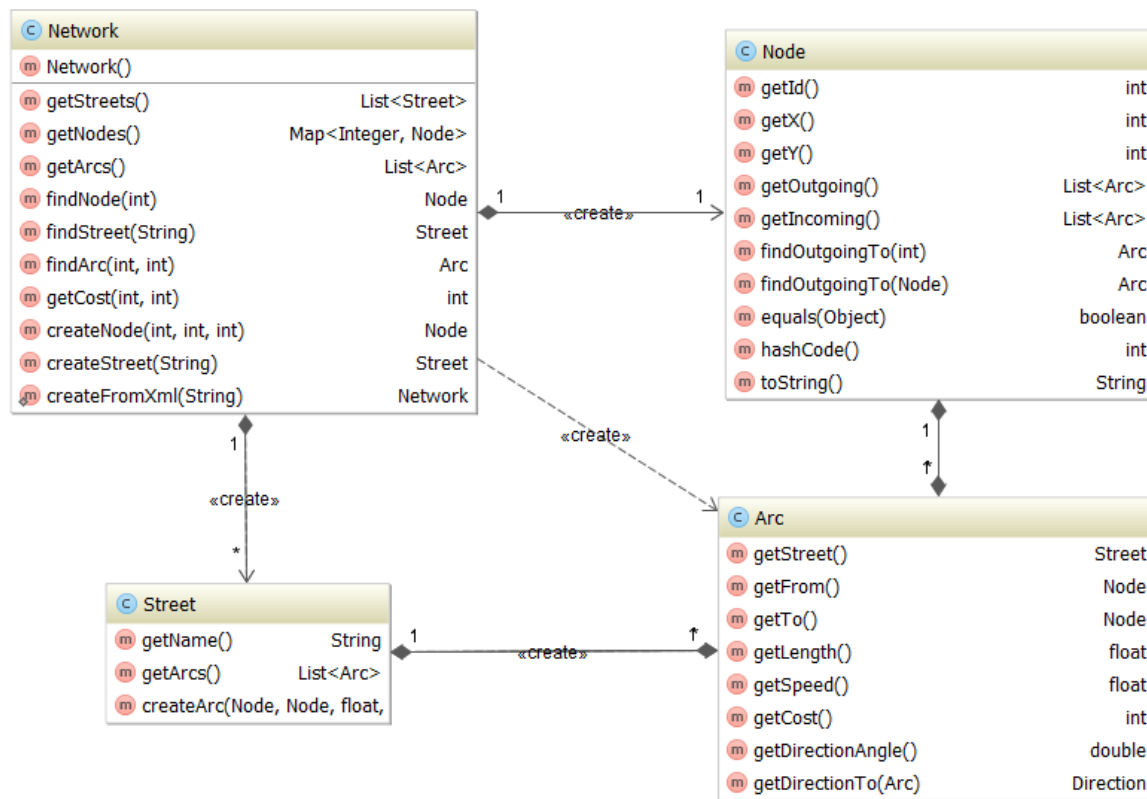
Package Modèle.Ville (Model.City)



Powered by yFiles

FIGURE 3.7 – Diagramme rétrogénéré UML du package Model.City

Package Modèle.Livraison (Model.Delivery)



Powered by yFiles

FIGURE 3.8 – Diagramme rétrogénéré UML du package Model.Delivery

Package Modèle.ChocoSolver (Model.ChocoSolver)

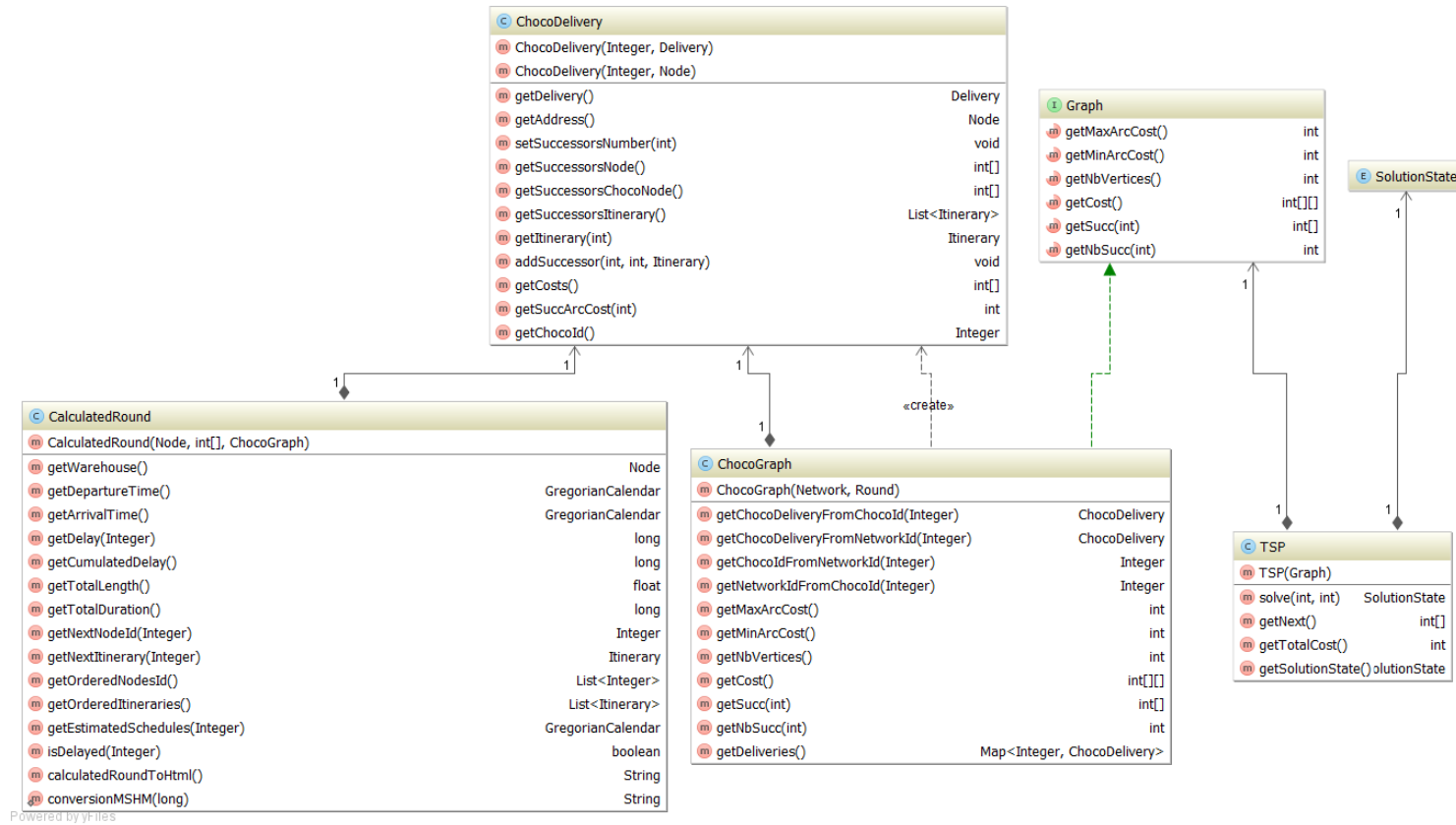
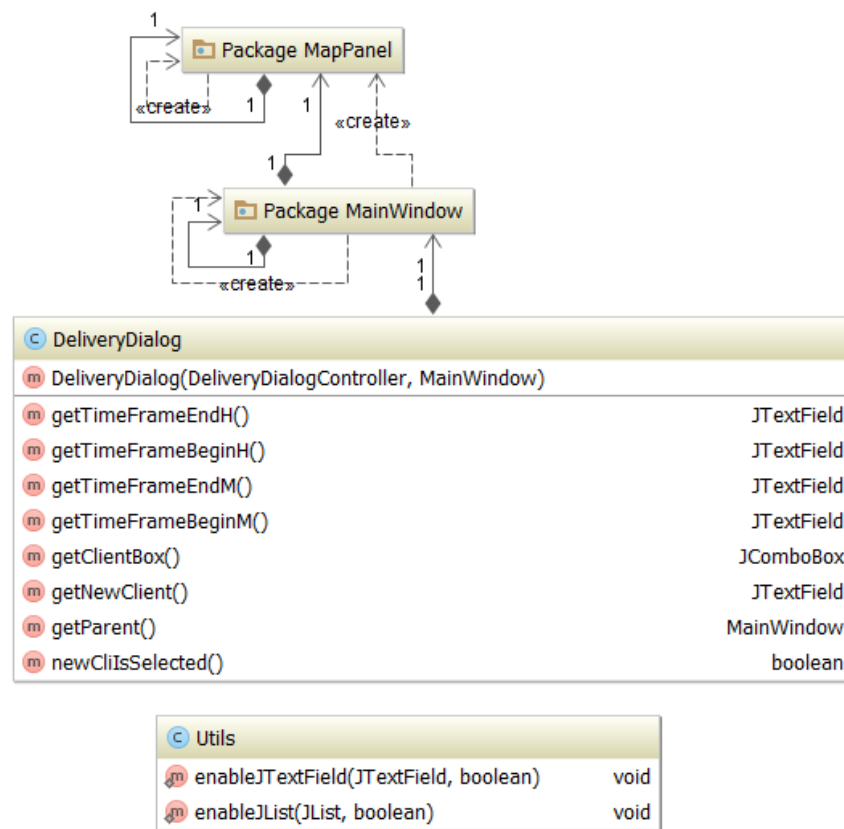


FIGURE 3.9 – Diagramme rétrogénéré UML du package Model.ChocoSolver

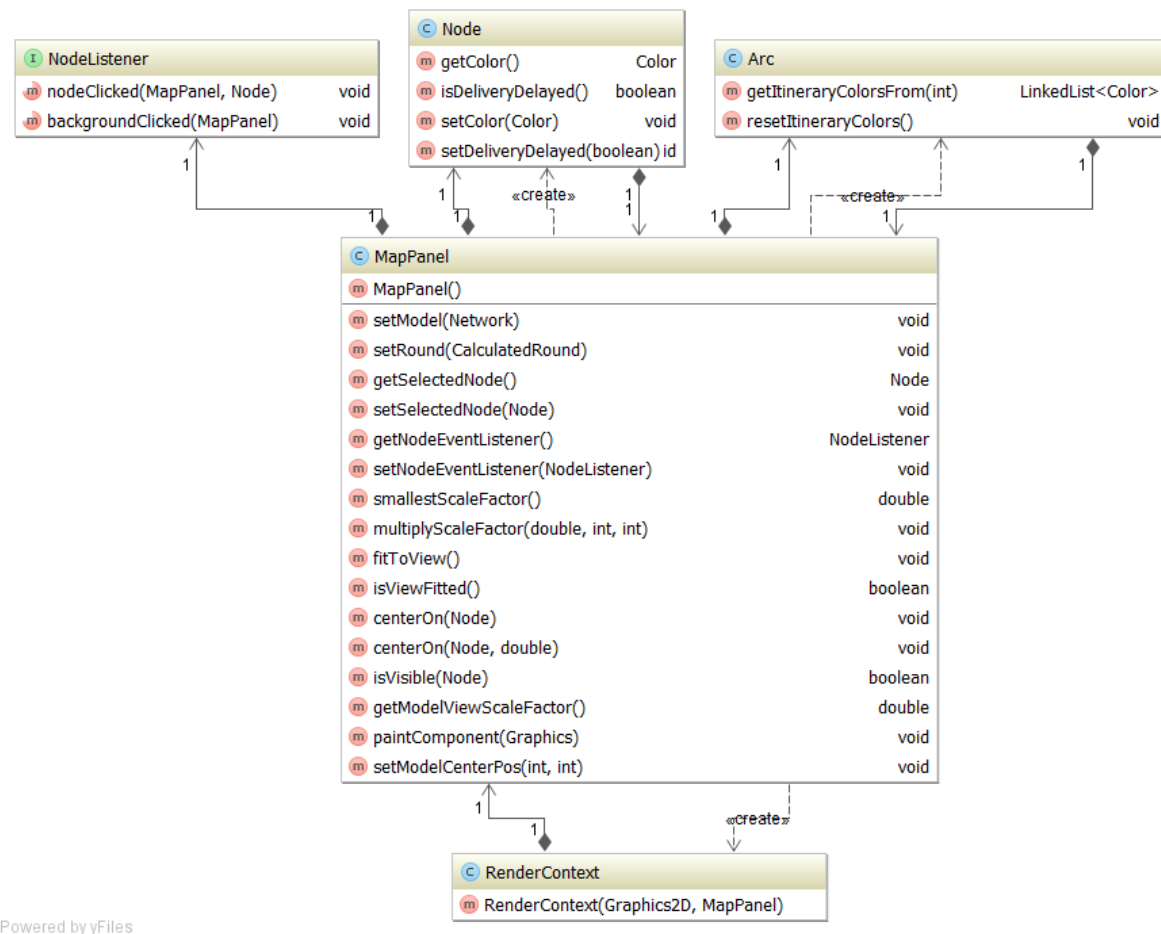
3.2.4 Package Vue (View)



Powered by yFiles

FIGURE 3.10 – Diagramme rétrogénéré UML du package View

Package Vue.Carte (View.MapPanel)



Powered by yFiles

FIGURE 3.11 – Diagramme rétrogénéré UML du package View.MapPanel

3.2.5 Package Controlleur (Controller)

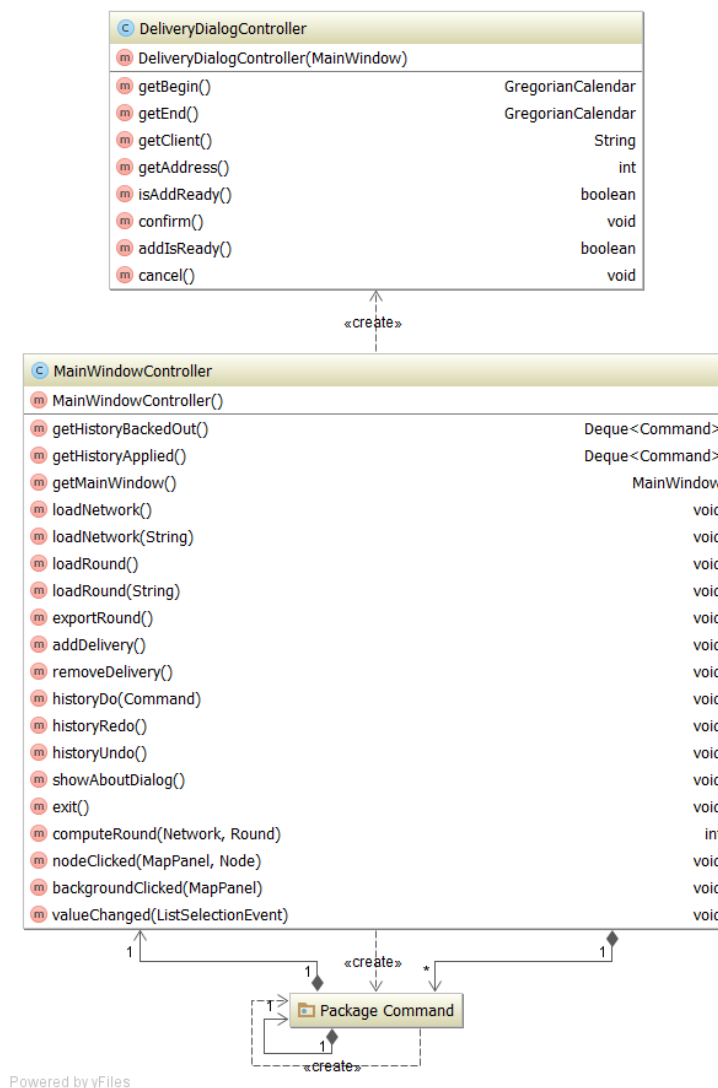


FIGURE 3.12 – Diagramme rétrogénéré UML du package Controller

Package Controlleur.Commande (Controller.Command)

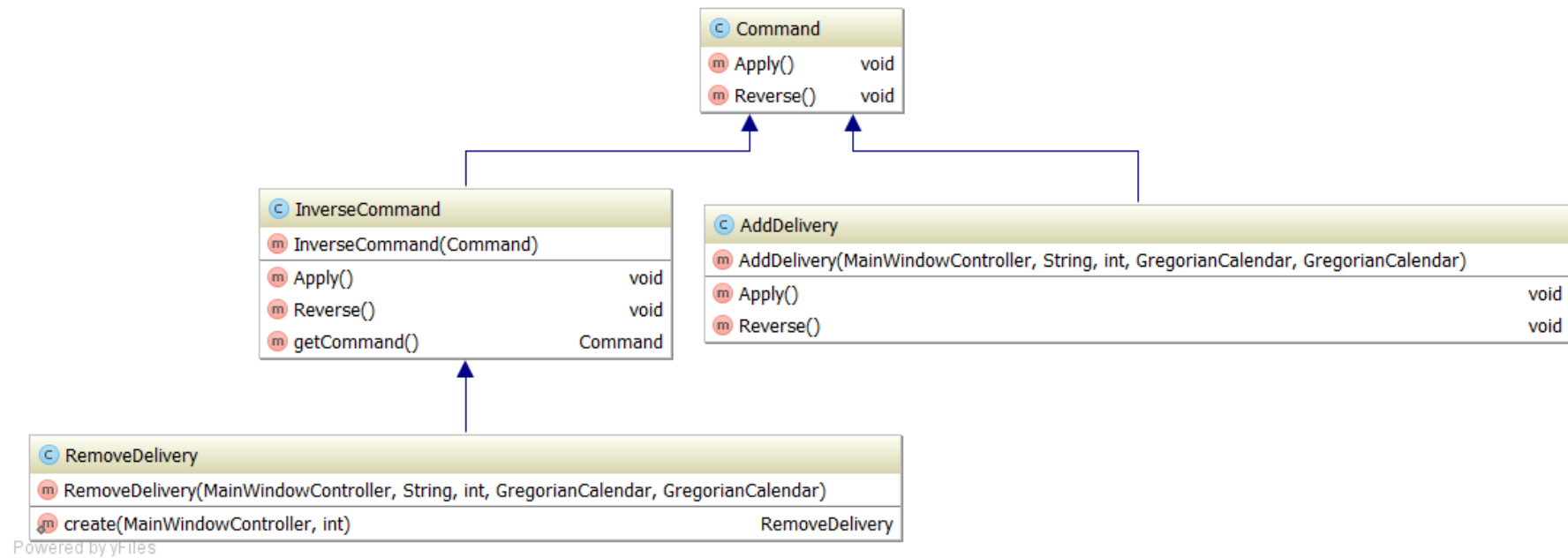


FIGURE 3.13 – Diagramme rétrogénéré UML du package Controller.Command

4. Bilans

4.1 Planning effectif du projet

Ressources	Séance 1	Séance 2	Séance 3	Séance 4	Séance 5
ABADIE Guillaume	Modèle du domaine (3h) Diagramme de classes (Modèle) (2h)	Diagramme de classes (2h) Parseur du plan (1h) Affichage du graphe (4h)	Amélioration de l'affichage du graph (7h)	Finitions undo / redo, ajouter / supprimer (1h) Amélioration + refactoring de l'affichage du graphe (7h)	Rapport LaTeX (4h) Affichage graphique d'une tournée (8h)
BUISSON Nicolas	Diagramme des UC (4h)	Conception IHM (1h30) Étude de Choco Solver (2h)	Étude de l'intégration de choco (1h30) Génération de la tournée (3h)	Binding du ChocoSolver (4h30)	Génération de la feuille de route (3h) Intégration ChocoGraph (5h)
CREPET Louise	Description des UC (4h)	Diagramme de séquence (2h) Conception IHM (1h30)	Diagramme de séquence (3h30)	Junit(5h) Finitions diagramme de séquence (1h)	Diagramme de séquence : tournée (4h) Vérifications Javadoc (2h) Tests Junit (2h)
DOMINGUES Rémi	Planning prévisionnel (2h) Diagramme de classes (Modèle) (3h)	Diagramme de classes (2h) Conception IHM (1h30) Étude de Choco Solver (1h)	Intégration du choco-solver (2h) Parseur d'une demande de livraisons (3h) Gestion de projet (3h) JUnit (1h)	Implémentation Dijkstra (2h) Implémentation de la classe ChocoGraph(8h)	Mise à jour des calendriers (2h) Bilan effectif et technique (2h) Implémentation de la classe ChocoGraph (12h) Tests Junit (1h)
MARTIN Aline	Description des UC (4h)	Diagramme de classes (2h) Diagramme de séquence (2h) Correction des use case (1h)	Diagramme de séquence (3h)	Binding contrôleur / interface (5h) Dessin des icônes (1h30)	Binding contrôleur / interface (8h) Dessin des icônes (2h)
WETTERWALD Martin	Structure du rapport (2h) Glossaire (2h)	Diagramme de classes (2h) Implémentation IHM (4h)	Implémentation IHM (6h)	Binding contrôleur / interface (7h)	Binding contrôleur / interface (9h)
Total	26h	29h30	33h	42h	64h

FIGURE 4.1 – Planning horaire du projet

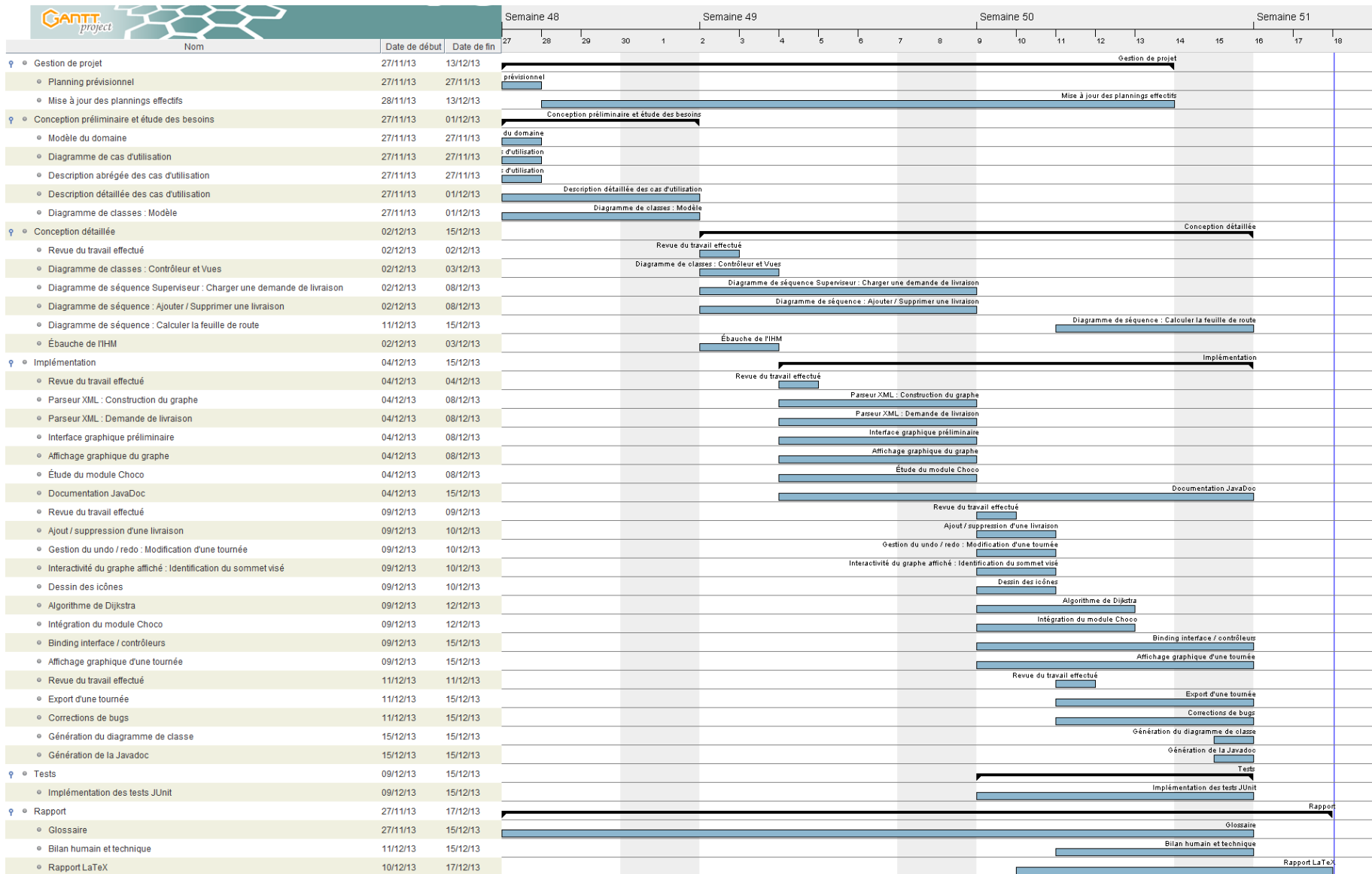


FIGURE 4.2 – Planning effectif du projet

Nom	Date de début	Date de fin	Ressources
Gestion de projet	27/11/13	13/12/13	
Planning prévisionnel	27/11/13	27/11/13	Domingues Rémi
Mise à jour des plannings effectifs	28/11/13	13/12/13	Domingues Rémi
Conception préliminaire et étude des besoins	27/11/13	01/12/13	
Modèle du domaine	27/11/13	27/11/13	Abadie Guillaume
Diagramme de cas d'utilisation	27/11/13	27/11/13	Buisson Nicolas, Crepet Louise
Description abrégée des cas d'utilisation	27/11/13	27/11/13	Crepet Louise, Martin Aline
Description détaillée des cas d'utilisation	27/11/13	01/12/13	Crepet Louise, Martin Aline
Diagramme de classes : Modèle	27/11/13	01/12/13	Abadie Guillaume, Domingues Rémi
Conception détaillée	02/12/13	10/12/13	
Revue du travail effectué	02/12/13	02/12/13	Buisson Nicolas, Crepet Louise, Domingues Rémi, Martin Aline, Wetterwald Martin
Diagramme de classes : Contrôleur et Vues	02/12/13	03/12/13	Abadie Guillaume, Domingues Rémi, Martin Aline, Wetterwald Martin
Diagramme de séquence Superviseur : Charger une demande de livraison	02/12/13	08/12/13	Crepet Louise
Diagramme de séquence : Ajouter / Supprimer une livraison	02/12/13	08/12/13	Martin Aline
Diagramme de séquence : Calculer la feuille de route	02/12/13	10/12/13	Buisson Nicolas
Ébauche de l'IHM	02/12/13	03/12/13	Buisson Nicolas, Crepet Louise, Domingues Rémi
Implémentation	04/12/13	15/12/13	
Revue du travail effectué	04/12/13	04/12/13	Abadie Guillaume, Buisson Nicolas, Crepet Louise, Domingues Rémi, Martin Aline, Wetterwald Martin
Parseur XML : Construction du graphe	04/12/13	08/12/13	Abadie Guillaume
Parseur XML : Demande de livraison	04/12/13	08/12/13	Domingues Rémi
Interface graphique préliminaire	04/12/13	08/12/13	Wetterwald Martin
Affichage graphique du graphe	04/12/13	08/12/13	Abadie Guillaume
Étude du module Choco	04/12/13	08/12/13	Buisson Nicolas, Domingues Rémi
Documentation JavaDoc	04/12/13	15/12/13	Abadie Guillaume, Buisson Nicolas, Crepet Louise, Domingues Rémi, Martin Aline, Wetterwald Martin
Revue du travail effectué	09/12/13	09/12/13	Abadie Guillaume, Buisson Nicolas, Crepet Louise, Domingues Rémi, Martin Aline, Wetterwald Martin
Ajout / suppression d'une livraison	09/12/13	10/12/13	Abadie Guillaume
Gestion du undo / redo : Modification d'une tournée	09/12/13	10/12/13	Abadie Guillaume
Interactivité du graphe affiché : Identification du sommet visé	09/12/13	10/12/13	Abadie Guillaume
Dessin des icônes	09/12/13	10/12/13	Martin Aline
Algorithme de Dijkstra	09/12/13	12/12/13	Domingues Rémi
Intégration du module Choco	09/12/13	12/12/13	Buisson Nicolas, Domingues Rémi
Binding interface / contrôleurs	09/12/13	15/12/13	Martin Aline, Wetterwald Martin
Affichage graphique d'une tournée	09/12/13	15/12/13	Abadie Guillaume
Revue du travail effectué	11/12/13	11/12/13	Abadie Guillaume, Buisson Nicolas, Crepet Louise, Domingues Rémi, Martin Aline, Wetterwald Martin
Export d'une tournée	11/12/13	15/12/13	Buisson Nicolas
Corrections de bugs	11/12/13	15/12/13	
Génération du diagramme de classe	15/12/13	15/12/13	Domingues Rémi
Génération de la Javadoc	15/12/13	15/12/13	Crepet Louise
Tests	09/12/13	15/12/13	
Implémentation des tests JUnit	09/12/13	15/12/13	Abadie Guillaume, Buisson Nicolas, Crepet Louise, Domingues Rémi
Rapport	27/11/13	17/12/13	
Glossaire	27/11/13	15/12/13	
Bilan humain et technique	11/12/13	15/12/13	Domingues Rémi
Rapport LaTeX	10/12/13	17/12/13	Abadie Guillaume

FIGURE 4.3 – Répartition des tâches

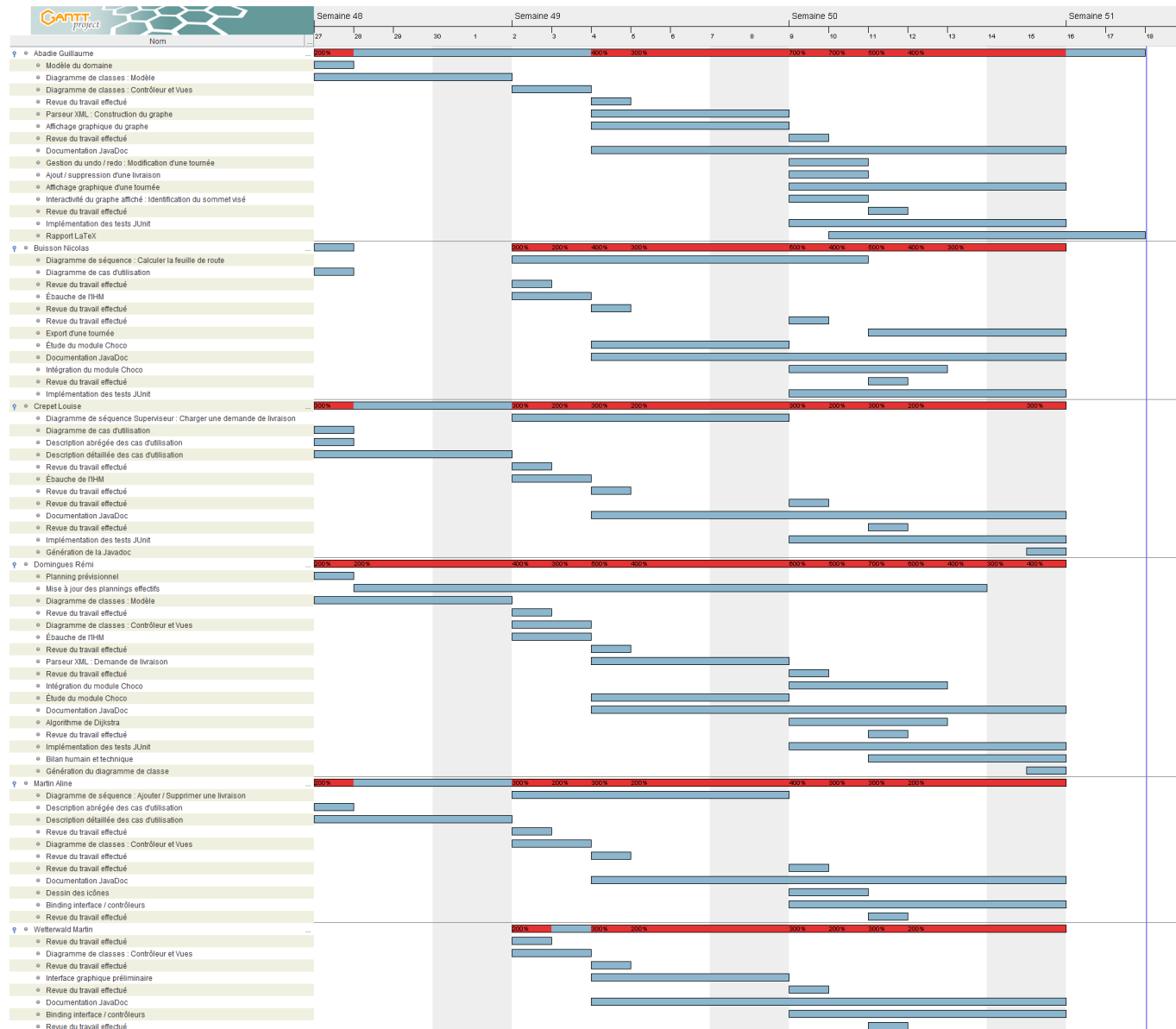


FIGURE 4.4 – Planning des ressources du projet

4.2 Bilan humain

4.2.1 Méthodologie

La réalisation du projet Opti_fret_COURLY est une occasion idéale pour l'exercice des méthodologies enseignées dans la filière Informatique de l'INSA de Lyon. Celle-ci a permis en effet une importante gestion de projet (diagramme de Gantt et de ressources, évaluation des durées des tâches, répartition des rôles, gestion du planning afin de respecter les échéances critiques) et une collaboration entre les différents membres de l'équipe de projet. Ce fut également l'occasion de réaliser un système en collaboration avec un client tenant rôle de maître d'ouvrage, permettant à notre équipe de se rendre compte de l'importance du dialogue avec le client, mais également de la position centrale que doivent occuper ses besoins dans la conception et la réalisation d'une application.

4.2.2 Respect du planning et adaptations

L'efficacité d'une équipe dynamique, sérieuse et bien organisée a permis un respect certain des échéances fixées et du planning général. Si, comme escompté, de nombreuses tâches n'ont pas pu être effectuées dans le cadre d'une séance de travail, celles-ci ont été systématiquement ou presque terminées en dehors des heures pédagogiques.

4.2.3 Ressenti

La première difficulté rencontrée dans la réalisation de ce projet est celle de la continuité du projet IHM. Le projet DevOO semble en effet présenté comme la réalisation du projet précédent, basé sur le même sujet récapitulatif des besoins clients. L'appréhension de nouveaux besoins clients est alors nécessaire.

Il est en outre demandé de réaliser une conception d'application en désaccord avec sa réalisation (diagrammes de cas d'utilisation incluant les applications livreurs et une base de données), ajoutant au sentiment de désarroi de l'équipe de projet.

Enfin, les fichiers XML de description d'un plan et d'une livraison étant livrés sans schéma XML ou DTD associée, la réalisation des parseurs en devient approximative et l'exercice de réalisation d'un tel parseur en accord avec une DTD n'est pas pratiqué.

4.3 Bilan technique

4.3.1 Sujet

La réalisation de ce système tire son intérêt majeur du projet réel dont il est issu. Il s'agit en effet là d'une application utilitariste basée sur un cas d'utilisation concret en accord avec les projets futurs que chacun d'entre nous devra réaliser dans un cadre professionnel.

En outre, ce sujet aborde des domaines techniques d'intérêt tels l'affichage graphique d'un graphe interactif, la résolution d'un plus court chemin dans un graphe et la résolution d'un TSP (traveler salesman problem).

4.3.2 Compétences acquises

Ce projet est en premier lieu l'occasion pour un hexanôme d'approfondir ses compétences en développement dans un langage de programmation de son choix.

Par ailleurs, celui-ci permet la découverte ou l'approfondissement de l'utilisation de bibliothèques graphiques standards dans le cadre du dessin d'un graphe, s'accompagnant de calculs vectoriels et de gestion des événements de la fenêtre parente.

Il en est enfin de même vis-à-vis de l'algorithme de Dijkstra permettant de calculer des plus courts chemins, et de la librairie interfaçant Choco fournie permettant la résolution d'un TSP. L'acquisition et la connaissance du maniement d'une telle librairie sera sans nul doute d'une indéniable utilité dans le cadre de la carrière future de certains membres de notre hexanôme.

