Table des matières

[1. Livrables de capture et analyse des besoins 1](#_Toc434494125)

[1.1 Planning prévisionnel du projet 1](#_Toc434494126)

[1.2 Modèle du domaine 2](#_Toc434494127)

[1.3 Glossaire 2](#_Toc434494128)

[1.4 Diagramme de cas d'utilisation 4](#_Toc434494129)

[1.5 Description textuelle structurée des cas d'utilisation 4](#_Toc434494130)

[2. Livrables de conception: 10](#_Toc434494131)

[2.1 Liste des événements utilisateur et diagramme Etats-transitions 10](#_Toc434494132)

[2.2 Diagrammes de packages et de classe 10](#_Toc434494133)

[2.3 Diagrammes de packages et de classes retro-générés à partir du code 10](#_Toc434494134)

[2.4 Choix architecturaux et design patterns utilisés 10](#_Toc434494135)

[2.5 Diagramme de séquence du calcul de la tournée à partir d'une demande de livraison 11](#_Toc434494136)

[3. Implémentation et tests 11](#_Toc434494137)

[4. Bilan 11](#_Toc434494138)

[4.1 Planning effectif du projet 11](#_Toc434494139)

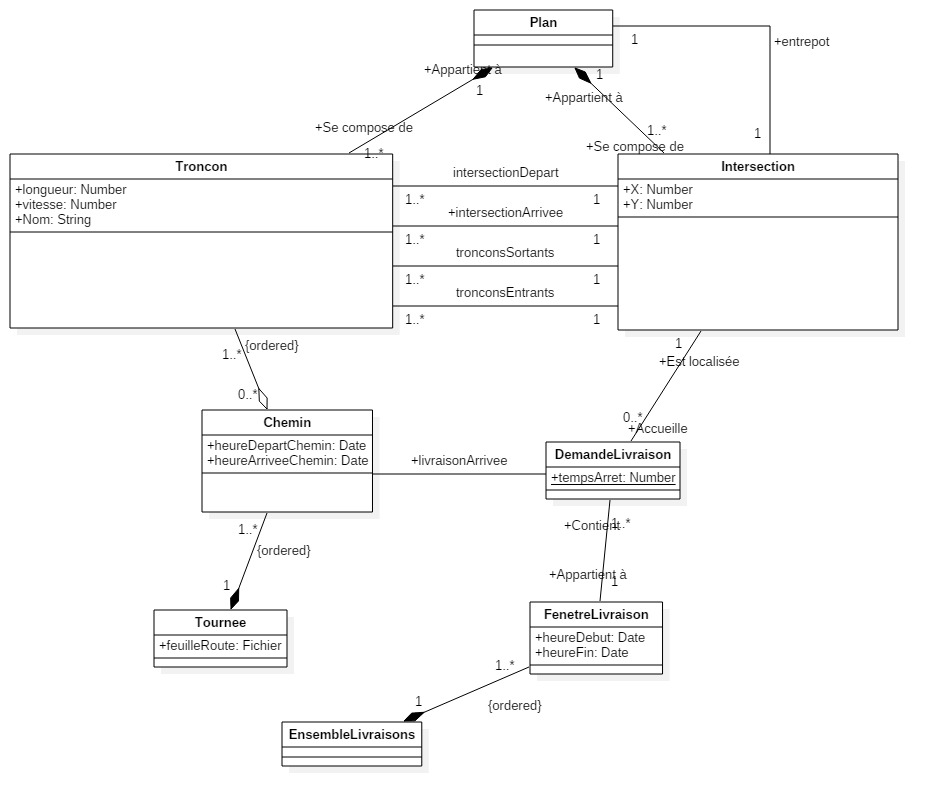
[4.2 Bilan humain et technique 12](#_Toc434494140)

[5. Conclusion 12](#_Toc434494141)

# 1. Livrables de capture et analyse des besoins

## 1.1 Planning prévisionnel du projet

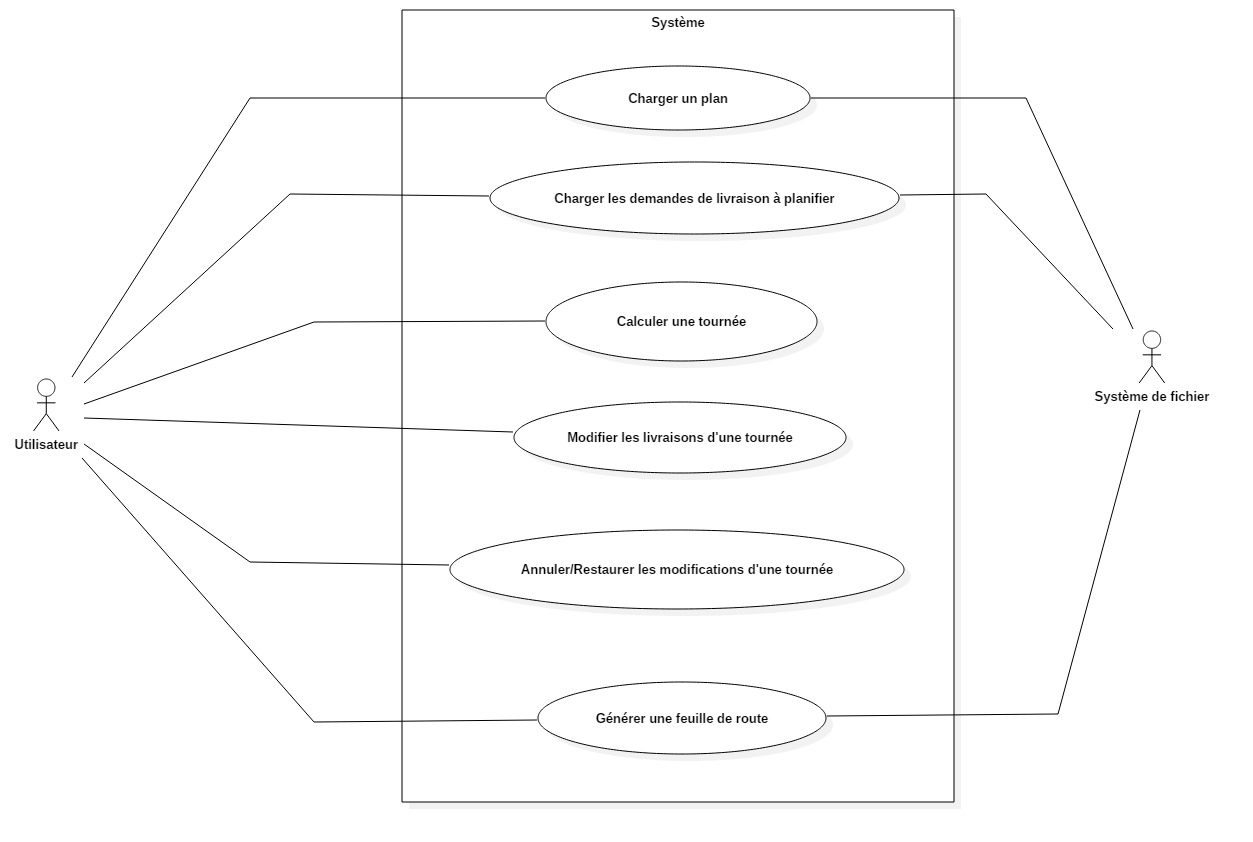
## 1.2 Modèle du domaine



## 1.3 Glossaire

* Chemin : Chemin entre deux livraisons.
* Heure de départ (d’un Chemin) : Heure à laquelle le livreur quitte la livraison précédente (donc temps de livraison non compris) ou l’entrepôt.
* Heure d’arrivée (d’un Chemin): Heure à laquelle le livreur arrive à la livraison suivante, nommée livraison arrivée (temps de livraison non compris).
* livraisonArrivee : La dernière livraison d'un chemin.
* Demande de livraison : Une demande de livraison.
* Temps d’arrêt (d’une demande de livraison) : Temps d'arrêt pour une livraison, ici fixé à 10 minutes pour chacune d'entre elles.
* Fenêtre de livraison : Regroupement des demandes de livraisons partageant la même fenêtre temporelle de livraison.
* Heure de début (d’une fenêtre de livraison) : Heure à partir de laquelle les demandes de livraison d’une fenêtre de livraison sont attendues.
* Heure de fin (d’une fenêtre de livraison) : Heure jusqu’à laquelle les demandes de livraison d’une fenêtre de livraison sont attendues.
* Ensemble de livraisons : Regroupement, ordonné temporellement, des fenêtres de livraison, qui contiennent-elles même les demandes de livraison.
* Intersection : Intersection de deux ou plusieurs tronçons. Une intersection peut correspondre à un point de livraison.
* x (d’une intersection) : Coordonnée X d’une l'intersection dans le plan.
* Y (d’une intersection): Coordonnée Y d’une l'intersection dans le plan.
* Tronçons sortants (d’une intersection) : Ensemble des tronçons quittant une intersection.
* Tronçons entrants (d’une intersection) : Ensemble des tronçons arrivant sur une intersection.
* Tronçon : Route empruntable entre deux intersections.
* Longueur (d’un tronçon) : longueur physique d'un tronçon.
* Vitesse (d’un tronçon) : vitesse moyenne du livreur sur le tronçon.
* Intersection de départ (d’un tronçon) : intersection à partir de laquelle commence un tronçon
* Intersection d’arrivée (d’un tronçon) : intersection sur laquelle arrive un tronçon.
* Plan : Cartographie les intersections et les tronçons.
* Tournée : Ensemble ordonné de chemins. Détermine la totalité du trajet du livreur depuis l'entrepôt jusqu'à l'entrepôt, en passant par les points de livraison.
* Feuille de route : fichier au format texte généré donnant la liste des livraisons à faire, dans l'ordre, avec l'adresse, horaires de livraison (arrivée et départ) et l'itinéraire entre chaque livraison.
* Entrepôt : Intersection de départ et d'arrivée d'une tournée.

## 1.4 Diagramme de cas d'utilisation



## 

## 1.5 Description textuelle structurée des cas d'utilisation

**Cas d'utilisation:**

Charger un plan

**Description abrégée:**

L’utilisateur demande au système de charger un plan. L'utilisateur choisit un fichier XML décrivant le plan à charger. Le système affiche le plan.

**Précondition:**

L'application est lancée

**Scénario principal:**

1) Le système demande à l’utilisateur de choisir un fichier XML décrivant le plan à charger.

*2)* L'utilisateur choisit le fichier xml valide.

*3)* Le système affiche le plan et rend la fonctionnalité "Charger demandes de livraison" active.

**Alternatives:**

*2a)* Le fichier choisit par l’utilisateur n’est pas valide (se référer à la liste des erreurs pour la définition des cas de non validité du fichier). Le système indique que le fichier n'est pas valide et retourne à l'étape 1.

*1-3a)* L'utilisateur indique au système qu'il souhaite annuler le chargement du fichier. Le système annule la demande de saisie.

Liste d’erreurs : - Le fichier xml n’est pas syntaxiquement valide.

- L’un des tronçons décrit dans le fichier xml référence une intersection inconnue.

**Cas d'utilisation:**

Charger les demandes de livraisons à planifier

**Description abrégée:**

L’utilisateur demande au système de charger des demandes de livraison à planifier. L'utilisateur choisit un fichier XML décrivant les demandes de livraisons à planifier. Le système affiche la position de chaque demande de livraison sur le plan, ainsi que, pour chaque demande de livraison, sa plage horaire.

**Précondition:**

Un plan est chargé.

**Scénario principal:**

1) Le système demande à l’utilisateur de choisir un fichier XML décrivant les demandes de livraisons à planifier.

2) L'utilisateur choisit le fichier contenant les demandes de livraisons à planifier.

3) Le système affiche le plan où chaque demande de livraison est mise en évidence.

**Alternatives:**

*2a)* Le fichier choisit par l’utilisateur n’est pas valide (se référer à la liste des erreurs pour la définition des cas de non validité du fichier). Le système indique que le fichier n'est pas valide et retourne à l'étape 1.

*1-3a)* L'utilisateur indique au système qu'il souhaite annuler le chargement du fichier. Le système annule la demande de saisie.

**Liste des erreurs** :

*-* Le fichier n'existe pas.

*-* Le fichier décrit des fenêtres de livraisons qui se chevauchent.

*-* Le fichier contient une adresse qui n'existe pas dans le plan.

*-* Le fichier décrit une fenêtre de livraison dont la date de fin est antérieure à la date de début. *-* Le fichier n’est pas syntaxiquement valide.

- Le fichier ne contient pas d'entrepot

**Cas d'utilisation:**

Calculer une tournée.

**Description abrégée:**

L'utilisateur demande au système de calculer la tournée de livraison. Le système calcule la tournée à partir des données qu'il possède sur les points de livraison puis affiche l'itinéraire à emprunter sur le plan. Le système affiche aussi, dans une autre partie de l'écran, les livraisons dans l'ordre dans lesquelles elles vont être réalisées.

**Préconditions:**

Un plan est chargé.

Un ensemble de points de livraison est chargé.

**Scénario principal:**

1) Le système calcule l'itinéraire à partir des données qu'il possède sur les points de livraison (fenêtre temporelle et localisation).

2) Le système affiche sur le plan, l'itinéraire à emprunter.

3) Le système affiche la liste des livraisons dans un autre cadre de l'écran. Cette liste respecte le même ordre que celui de la tournée.

**Alternatives:**

*1a)* Le temps de calcul de la tournée par le système dépasse le temps limite de calcul fixé. On passe à l’étape 2, mais l’itinéraire affiché par le système n’est pas l’itinéraire optimum, c’est le meilleur itinéraire que le système ait pu calculer dans le temps limite.

*3a)* Certaines livraisons ne respectent pas la fenêtre temporelle qui leur imposée.

Ces livraisons sont signalées à l'utilisateur par un indicateur visuel.

**Cas d’utilisation :**

Modifier les livraisons d’une tournée

Ce cas d’utilisation est détaillé ici en plusieurs scénarios possibles de modification

***Cas d’utilisation 1:***

Supprimer des livraisons

**Description abrégée:**

L'utilisateur sélectionne des points de livraison d'une tournée qu’il souhaite supprimer de la tournée et demande au système de supprimer ces livraisons. Le système effectue la suppression et affiche la nouvelle tournée.

**Précondition:**

Le plan est chargé, la demande est chargée et une tournée est calculée.

**Scénario principal:**

*1*) L'utilisateur choisit des points de livraison qui appartiennent à la tournée calculée.

*2)* L'utilisateur demande au système de supprimer les points de livraisons choisis.

3) Le système enlève les points de livraison choisis de la tournée.

*4)* Le système affiche la nouvelle tournée modifiée.

**Alternatives:**

*2a)* Un point de livraison choisi n'appartient pas à la tournée calculée. Le système indique que le point n'est pas valide et retourne à l'étape 1.

*2b)* Un point sélectionné est l'entrepôt. Le système indique qu'il n'est pas possible de supprimer l'entrepôt.

***Cas d’utilisation 2:***

Ajouter des livraisons

**Description abrégée:**

L'utilisateur choisit des points de livraison sur le plan et demande au système d’ajouter ces points de livraison. Le système affiche la nouvelle tournée contenant les points de livraison sélectionnés.

**Précondition:**

Le plan est chargé, la demande est chargée et une tournée est calculée.

**Scénario principal:**

*1)* L'utilisateur choisit les points de livraison à ajouter à la tournée.

*2)* L'utilisateur demande au système d’ajouter les points de livraison choisis.

3) Le système demande à l’utilisateur de choisir un point de livraison de la tournée après lequel les points de livraison choisis précédemment doivent être ajoutés.

4) L’utilisateur choisit le point demandé.

5) Le système ajoute à la tournée les points de livraison choisis.

6) Le système affiche la nouvelle tournée.

**Alternatives:**

*2a)* Un point sélectionné appartient à la tournée calculée. Le système indique que le point n'est pas valide et retourne à l'étape 1.

*4a)* Le point sélectionné n’appartient pas à la tournée calculée. Le système indique que le point n'est pas valide et retourne à l'étape 3.

***Cas d’utilisation:***

Echanger deux livraisons

**Description abrégée**:

L'utilisateur choisit deux points de livraison de la tournée calculée et demande au système d’échanger les points de livraison. Le système effectue le changement et affiche la nouvelle tournée avec les deux points de livraison échangés.

**Préconditions:**

Le plan est chargé, les points de livraison sont chargés et une tournée est calculée.

**Scénario principal:**

*1)* L'utilisateur choisit deux points de livraison.

*2)* L'utilisateur demande au système d’échanger les deux points de livraison choisis.

3) Le système échange les deux points de livraison choisis.

*4)* Le système affiche la nouvelle tournée.

**Alternatives:**

*2a)* Un point sélectionné n'appartient pas à la tournée calculée. Le système indique que le point n'est pas valide et retourne à l'étape 1.

*2b)* Un point sélectionné est l'entrepôt. Le système indique qu'il n'est pas possible d'échanger un point de livraison avec l'entrepôt.

**Cas d'utilisation:**

Annuler/Restaurer les modifications d'une tournée

Ce cas d’utilisation est détaillé ici en plusieurs scenarios possibles

***Cas d’utilisation 1:***

Annuler la dernière modification de la tournée.

**Description abrégée:**

L'utilisateur demande au logiciel d'annuler la dernière modification. La tournée revient dans l'état antérieur à la dernière modification.

**Précondition:**

Une modification a été apportée à la tournée proposée par le logiciel.

**Scénario principal:**

*1)* L'utilisateur demande au logiciel d'annuler la dernière modification.

*2)* Le système revient dans l'état où il était avant la dernière modification.

**Alternatives:**

*2a)* Aucune modification à annuler. Le système ne fait rien et l’étape 2) est ignorée

***Cas d’utilisation 2:***

Restaurer la dernière modification annulée de la tournée.

**Description abrégée:**

L'utilisateur demande au logiciel de restaurer la dernière modification annulée.

La tournée revient dans l'état antérieur à la dernière annulation.

**Précondition:**

Une modification apportée à la tournée proposée par le logiciel a été annulée.

**Scénario principal:**

*1)* L'utilisateur demande au logiciel de restaurer la dernière modification annulée.

*2)* Le système revient dans l'état où il était avant la dernière annulation.

**Alternatives:**

*2a)* Aucune modification à restaurer. Le système ne fait rien et l’étape 2) est ignorée

**Cas d'utilisation:**

Générer la feuille de route.

**Description abrégée:**

L'utilisateur valide la tournée et demande au système de générer la feuille de route correspondante. Le système génère la feuille de route à partir de la tournée validée. Le système enregistre la feuille de route calculée dans un fichier texte.

**Préconditions:**

Un plan est chargé.

Un ensemble de points de livraison est chargé.

La tournée a été calculée.

**Scénario principal:**

*1)* L'utilisateur valide la tournée et demande au système de calculer la feuille de route.

*2)* Le système demande à l'utilisateur de choisir un emplacement et un nom de fichier pour la sauvegarde de la feuille de route.

*3)* L'utilisateur indique au système un nom et un emplacement de fichier.

*4)* Le système génère la feuille de route.

*5)* Le système sauvegarde la feuille de route dans le fichier renseigné à l'étape 3.

**Alternatives:**

*2a)* L'utilisateur choisit d'annuler la génération de la feuille de route. Le système termine la procédure de génération de la feuille de route.

*3a)* Il n'est pas possible de créer ou d'écrire à l'emplacement renseigné. Le système indique la nature du problème et retourne à l'étape 2.

*3b)* Il existe déjà un fichier à l'emplacement renseigné par l'utilisateur.

*3b-1)* Le système demande à l'utilisateur s'il doit écraser l'instance déjà existante du fichier.

*3b-2a)* L'utilisateur confirme l'écrasement du fichier déjà existant. Le système écrasera l'instance existante du fichier à l'étape 5. Le système passe à l'étape 4.

*3b-2b)* L'utilisateur refuse l'écrasement du fichier déjà existant. Le système retourne à l'étape 2.

# 2. Livrables de conception:

## 2.1 Liste des événements utilisateur et diagramme Etats-transitions

## 2.2 Diagrammes de packages et de classe

## 2.3 Diagrammes de packages et de classes retro-générés à partir du code

## 2.4 Choix architecturaux et design patterns utilisés

Au niveau de l’architecture générale du projet, nous avons construit l’application sur une architecture MVC (Modèle-Vue-Contrôleur). Premièrement car cette architecture était imposée, et ensuite car cela correspondait au type d’application développé ici.

**Pattern Observer**

Lorsqu’une tournée a été calculée par le système, l’utilisateur doit pouvoir modifier cette tournée (ajout, suppression, échange de points de livraisons). Afin que la vue puisse communiquer avec le modèle, (par exemple, que la vue soit informée de l’ajout d’une livraison à l’ensemble de livraison), nous avons utilisé le pattern *observer*. Ainsi la vue peut-être *observer* des objets Tournée et EnsembleLivraisons, qui sont quant à eux *observable*.

Pattern undo/redo

Dans le cahier des charges, il est spécifié qu’une modification de la tournée doit pouvoir être annulée. Une telle spécification appelle clairement à l’utilisation du pattern Commande, qui permettra de faire aisément des undo/redo sur les modifications.

ModeleManager

Lorsque le contrôleur communique avec le modèle, il est souhaitable que la communication se fasse au moyen d’un unique point d’entrée sur le modèle. Nous avons fait le choix d’une classe ModeleManager, qui contiendrait alors le plan actuellement chargé, les demandes de livraison actuellement chargées, et la tournée actuellement générée. C’est aussi cette classe qui contient la méthode calculerTournee. À noter que le modeleManager aurait pu être implémenté en utilisant le pattern singleton (une seule et unique instance de ModeleManager dans l’application). Cependant, dans un souci de réutilisabilité, nous avons écarté ce choix. En effet, dans le cas où, par exemple, l’application permettrait d’avoir plusieurs onglets contenant plusieurs plans différents en même temps, il serait pratique d’avoir plusieurs instances de ModeleManager.

Package XML

Les descriptions des plans de villes et les descriptions des demandes de livraisons sont réalisées au moyen de fichiers au format xml. Ces fichiers doivent être parsés, et les informations contenues doivent être analysées, puis envoyées dans le modèle. Nous avons décidé de placer le composant applicatif relatif au parsing dans un package particulier (différents des trois packages principaux), le package xml.

Pattern State

L’application que nous développons possède plusieurs états de fonctionnement (Le plan est chargé, puis les demandes de livraisons sont chargées, etc.) Les possibilités d’interaction utilisateur et les effets des interactions ne sont pas les même selon l’état de l’application. Afin que le contrôleur puisse gérer les différents états, nous avons implémenté le pattern state.

**Gestion des fonctionnalités actives en fonction des états**

Selon l’état du contrôleur, toutes les fonctionnalités de l’application ne doive pas pouvoir être appelée (Lorsqu’un plan n’est pas chargé, par exemple, l’utilisateur ne devrait pas pouvoir demander à l’application de charger des demandes de livraison). C’est pourquoi nous avons mis en place un système d’activation/désactivation de fonctionnalité. La fenêtre graphique de l’application implémente une méthode d’activation/désactivation de fonctionnalité pour chaque fonctionnalité (on entend ici les boutons, les items des menus, .etc.). A chaque changement d’état, l’état désactive l’ensemble des fonctionnalités et réactive les fonctionnalités permises dans cet état.

## 2.5 Diagramme de séquence du calcul de la tournée à partir d'une demande de livraison

# 3. Implémentation et tests

## 

# 4. Bilan

## 4.1 Planning effectif du projet

## 4.2 Bilan humain et technique

# 5. Conclusion