



UNIVERSITÉ
LAVAL

Projet de session :
Développement d'une application *VirtuBois*

Livrable 3

présenté à

M. Jonathan Gaudreault

par

Équipe 02

<i>matricule</i>	<i>nom</i>	<i>signature</i>
111 174 511	Martine Deschênes	
111 175 430	Jordan Mayhue	
111 073 283	Yoan Chamberland	
111 178 727	Gabriel St-Pierre	

Université Laval
9 avril 2019

Historique des versions

<i>version</i>	<i>date</i>	<i>description</i>
#0	23 janvier 2019	Création du document et début de la réalisation du projet
#1	5 février 2019	Remise du Livrable 1
#2	6 février 2019	Début de la réalisation du Livrable 2
#3	26 février 2019	Remise du Livrable 2
#4	27 février 2019	Début de la réalisation du Livrable 3
final	9 avril 2019	Remise du Livrable 3

Table des matières

Table des figures	ii
1 Modèle du domaine	1
1.1 Diagramme du modèle du domaine	1
1.2 Texte explicatif	2
2 Modèle des cas d'utilisation	4
3 Diagramme de classe de conception	5
3.1 Diagramme de classe de conception	5
3.2 Explication du digramme de classe de conception	5
3.2.1 Explication des classes du domaine	5
3.2.1.1 Contrôleur de Larman (LarmanController)	5
3.2.1.2 Dtos	6
3.2.1.3 Entités (Entities)	6
3.2.2 Helpers	7
3.2.3 Java	8
3.2.4 Enums	8
3.2.5 Presentation	8
3.2.5.1 views	8
3.2.5.2 javafxControllers	8
3.2.5.3 javafx	9
3.2.5.4 JavafxPresenters	10
4 Diagrammes d'état de la chargeuse	11
5 Contribution des membres de l'équipe	12

Table des figures

1.1	Diagramme des classes conceptuelles	1
2.1	Diagramme des cas d'utilisation	4
3.1	Diagramme de classe de conception	5
4.1	Diagramme d'états de la chargeuse	11

Chapitre 1

Modèle du domaine

1.1 Diagramme du modèle du domaine

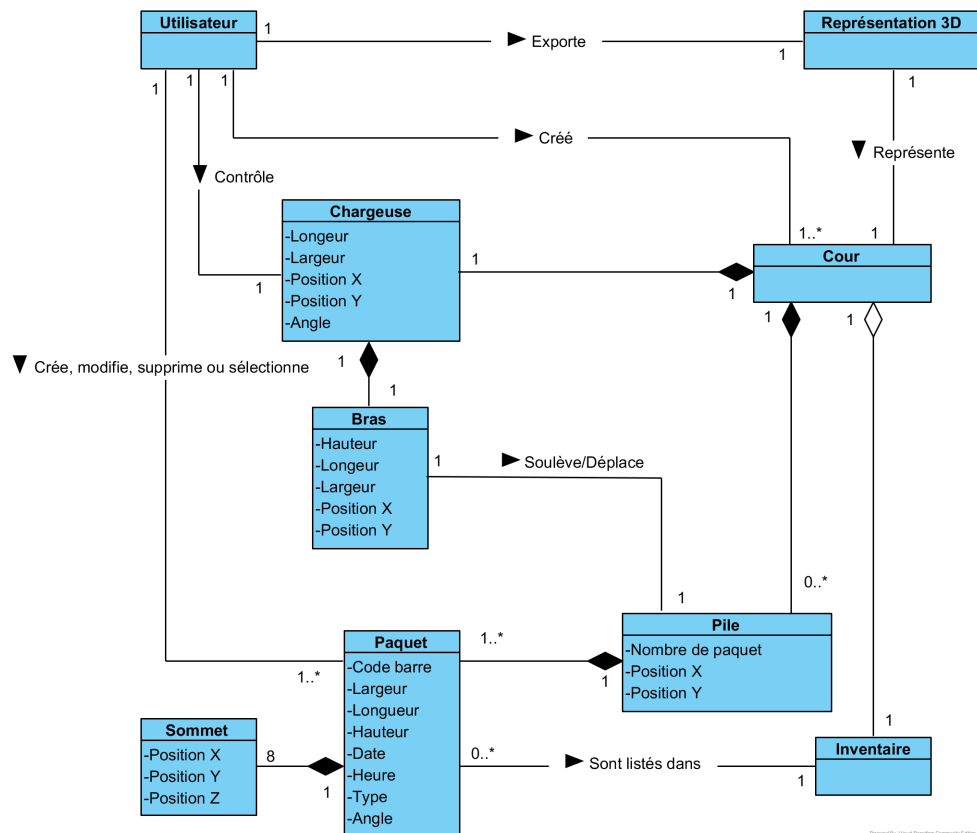


FIGURE 1.1 – Diagramme des classes conceptuelles

1.2 Texte explicatif

Utilisateur : Il s'agit tout simplement de l'utilisateur du logiciel. Cet utilisateur peut créer une cour à bois, dans un projet distinct, dans lequel il crée différents paquets de bois qu'il peut déplacer de manière manuelle ou avec une chargeuse à bras. Il peut aussi modifier, à sa guise, les attributs des paquets comme ses dimensions, sa localisation, son orientation, etc. En outre, il peut exporter en 3D une représentation de la cour.

Cour : Ceci représente en quelque sorte un projet : un objet contenant toutes les informations pour une cour à bois. Il est possible d'avoir plusieurs cours qui contiennent des informations différentes les unes des autres. Il faudra cependant charger des fichiers indépendants pour avoir accès à l'une ou l'autre des cours. Chaque cour possède des dimensions infinies, c'est-à-dire qu'on peut se déplacer indéfiniment dans une cour sans jamais atteindre de limite. De plus, une cour possède une instance de la classe Chargeuse et autant d'instances de la classe Pile que l'utilisateur souhaite.

Paquet : Ce sont les paquets de bois qui sont contenus dans la cour. Ils ont des propriétés qui permettent de les différencier les uns des autres, soit un code barre, une largeur, une longueur, une hauteur, une date et heure de production ainsi qu'un type (p.ex. 2x6 Érable). Tous les paquets ont la forme d'un prisme à base rectangulaire. L'utilisateur peut les déplacer dans la cour en tout temps, soit manuellement ou à l'aide de la chargeuse. Également, l'utilisateur peut modifier l'angle et les attributs des paquets en tout temps. On peut retrouver l'emplacement d'un paquet avec une simple recherche par code à barre dans l'inventaire.

Pile : Une pile est constituée d'au moins un paquet. C'est ce qui représente la position de un ou plusieurs paquets superposés. L'utilisateur peut, en tout temps, connaître le nombre de paquets contenu dans la pile, l'ordre de ces paquets dans la pile ainsi que les différentes informations sur chacun des paquets. Toute cette information est accessible à partir de la vue d'élévation. Il peut aussi utiliser la chargeuse pour déplacer une partie des paquets de la pile ou encore la pile au complet, tout dépendant de la hauteur des bras de la chargeuse.

Inventaire : Il s'agit d'une liste de tous les paquets présents dans la cour à bois. Grâce à l'inventaire, un utilisateur pourra en tout temps rechercher un paquet par code à barre pour retrouver son emplacement dans la cour et connaître ses propriétés respectives.

Chargeuse : C'est une entité qui permet de se déplacer dans la cour et de déplacer les paquets d'une pile à une autre. Elle contient des coordonnées qui permettent de la localiser. En tout temps, l'utilisateur peut prendre le contrôle de la chargeuse simplement en cliquant dessus. Il peut ainsi la déplacer dans la cour à travers les différentes piles de paquets. La chargeuse est composée d'un bras pour lui permettre d'effectuer son travail.

Bras : C'est ce qui soulève les paquets. On utilise des coordonnées décalées de celles de la chargeuse. Grâce à certaines touches du clavier, les bras peuvent monter ou descendre.

Ainsi, les coordonnées horizontales des bras sont dépendantes des coordonnées horizontales de la chargeuse tandis que les coordonnées verticales sont indépendantes de la chargeuse.

Sommet : Il s'agit d'une coordonnée utilisée pour représenter un des coins du polyèdre qui représente un paquet.

Représentation 3D : C'est la représentation 3D au format STL de la cour et ses paquets. On peut voir la cour en version 3D en exportant celle-ci.

Chapitre 2

Modèle des cas d'utilisation

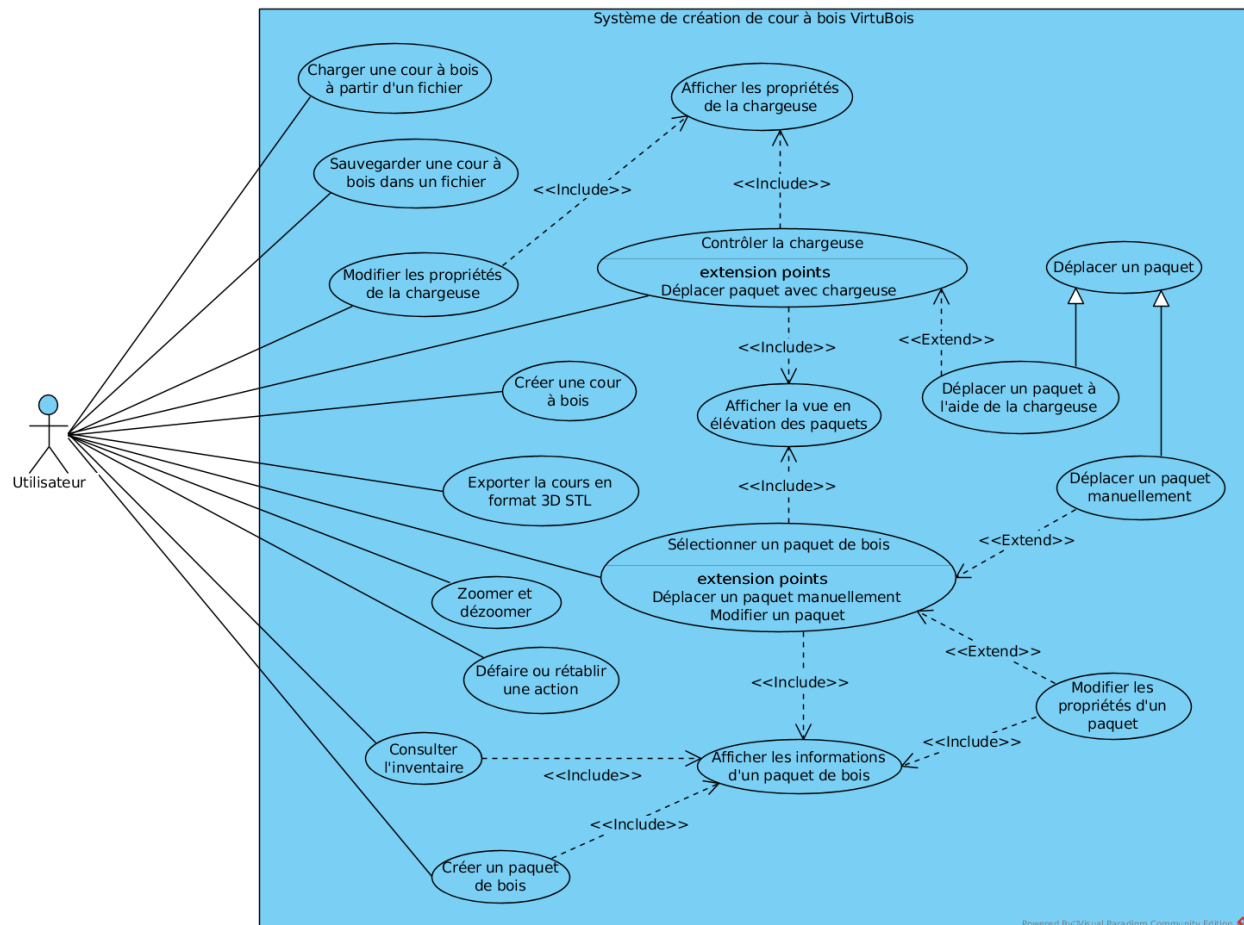


FIGURE 2.1 – Diagramme des cas d'utilisation

Diagramme de classe de conception

3.1 Diagramme de classe de conception

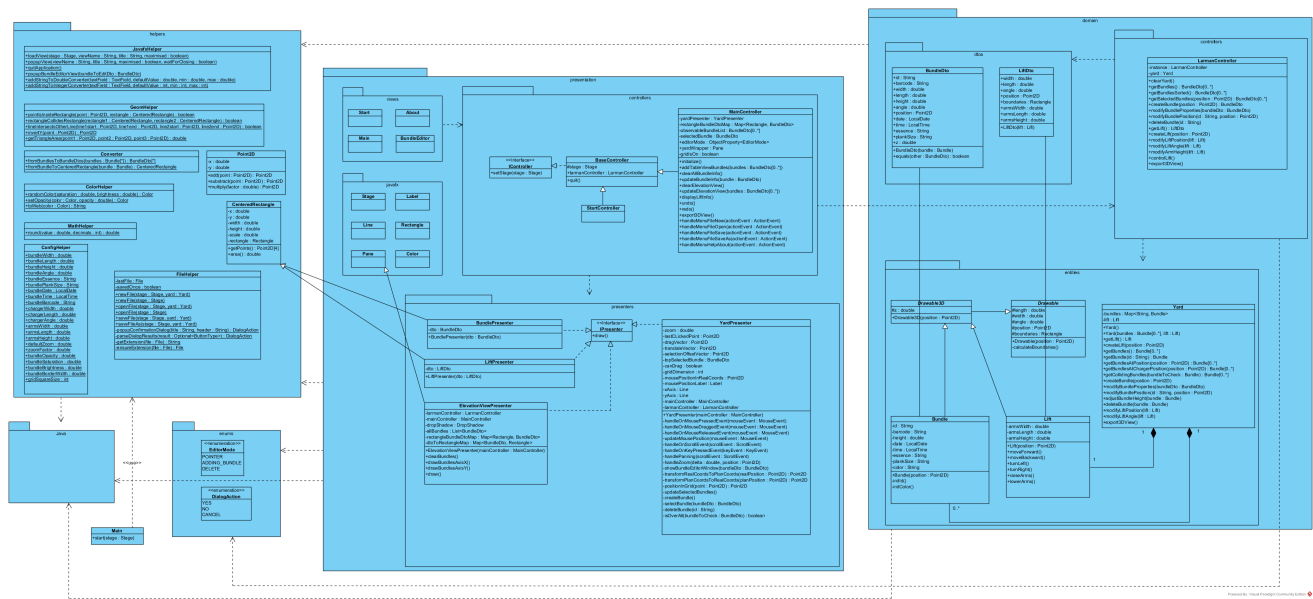


FIGURE 3.1 – Diagramme de classe de conception

3.2 Explication du digramme de classe de conception

3.2.1 Explication des classes du domaine

3.2.1.1 Contrôleur de Larman (LarmanController)

Le contrôleur de Larman est la passerelle entre l'interface utilisateur et le domaine de l'application. Ce contrôleur gère, à l'aide de diverses méthodes, toutes les interactions entre ces

deux entités. Du côté de l'interface utilisateur (*presentation*), plusieurs contrôleurs JavaFX interagissent avec ce contrôleur pour faire le lien.

3.2.1.2 Dtos

Les attributs de chacune des classes (Bundle et Lift) servent à contenir les données qui seront échangées entre l'interface et le domaine. On peut accéder à ces Dtos grâce aux méthodes *getLift* et *getBundles* du contrôleur de Larman. Les entités BundleDto et LiftDto contiennent chacun un constructeur BundleDto et LiftDto qui crée l'objet avec les données actuelles qu'elles contiennent.

3.2.1.3 Entités (Entities)

Les entités représentent les objets dans le domaine qui seront affichés à l'utilisateur. En effet, dans ce paquet, nous retrouvons les entités cour, paquet et chargeuse que l'utilisateur voit sur l'interface. Ce paquet contient aussi des entités de dessins afin de créer en 2D ou en 3D les entités de ce paquet. Finalement, il est possible de remarquer que chacune des entités sont liées entre elles. En effet, la cour contient les paquets et la chargeuse qui héritent eux-même de *Drawable*. *Drawable3D* hérite lui aussi de *Drawable*.

Bundle

Cette classe représente les paquets de bois qui sont contenus dans la cour. Chaque paquet a pour attributs une hauteur bien définie, une date et une heure de création, une essence spécifique, la taille des planches contenues dans le paquet, un code à barre entré par l'utilisateur ainsi qu'un identifiant unique. Elle possède aussi des attributs qu'elle hérite de sa classe de base *drawable* permettant de lui spécifier une longueur, une largeur, un angle, une position dans la cour à bois ainsi qu'une frontière. On peut accéder aux attributs et les modifier à partir de *getters* et de *setters* qui seront implantés. Il est également à noter que le constructeur de la classe ne prend en paramètre qu'une coordonnée en *x* et *y* dans l'espace de la cours à bois. Cela est dû au fait que, par défaut, tous les attributs à l'exception de l'identifiant et de la date et l'heure seront initialisés aux valeurs du dernier paquet créé. Pour ce qui est de l'identifiant d'un paquet, il sera initialisé à partir de la méthode *initId*.

Lift

Cette classe gère tout ce qui touche à la chargeuse. Il ne peut exister plus d'une instance de cette classe, car il n'y a qu'une chargeuse dans la cour. Tout comme la classe Pack, elle hérite de la classe de base *drawable*. La chargeuse possède pour attributs la longueur de ses bras, la largeur de ses bras et la hauteur de ceux-ci. Elle hérite également des attributs *drawable*, soit une longueur, une largeur, un angle, une position dans la cour à bois ainsi qu'une frontière délimitant l'objet dans l'espace. On peut accéder aux attributs et les modifier à partir de *getters* et de *setters* qui seront implantés. Les méthodes *moveForward*, *moveBackward*, *turnLeft* et *turnRight*, permettront de déplacer la chargeuse dans la cour. Les méthodes *raiseArms* et *lowerArms* permettront, quant à elles, de modifier la hauteur des bras de la chargeuse.

Yard

Cette classe représente une cour à bois. Elle est composée d'une instance de *Lift* et d'une map contenant tous les paquets associés à son identifiant. La classe implémente des méthodes permettant d'obtenir une liste des paquets présents dans la cour, mais également d'obtenir une liste des paquets présents à une position précise dans la cour. C'est aussi cette classe qui est en charge de créer les paquets, d'en modifier les différents attributs selon les instructions du contrôleur de Larman, mais également de créer une chargeuse et d'en modifier certaines propriétés.

Drawable

Drawable est une interface des entités qui soutient l'affichage des paquets et de la chargeuse à bois. En effet, *Bundle* et *Lift*, deux des classes de ce paquet héritent de certains attributs essentiels au dessin dans l'interface des objets qui les composent. Il est possible d'y retrouver entre autres, la position, la longueur, l'angle, la largeur, etc. De plus, cette classe contient une méthode permettant de connaître les arêtes du rectangle qui composent le paquet.

Drawable3D

Drawable3D est très semblable à l'interface précédemment mentionnée. En effet, *Drawable3D*, qui hérite de *Drawable* permet tout simplement, grâce à sa méthode et à son attribut, de dessiner les objets en 3D lors de l'affichage en 3D de la cour à bois.

3.2.2 Helpers

Le paquet *Helpers* du diagramme contient toutes les fonctions utilitaires simples qui sont utilisées par le domaine et par l'interface utilisateur pour accomplir des tâches mathématiques ou pour ce qui à trait à la vue. Dans le premier cas, c'est la classe *GeomHelper* qui contient toutes les méthodes mathématiques pour, par exemple, savoir si les coordonnées d'un point sont dans un rectangle, ce qui remplacera l'algorithme implanté dans la partie 4 du livrable 2, si deux rectangles se touchent, etc. Les *JavafxHelpers*, quant à eux, aideront à l'affichage de l'application. *Converter* fait la conversion des paquets en dtos et vice versa. *ColorHelper* s'occupe d'attribuer une couleur aux paquets de la cour à bois. *ConfigHelper* contient toutes les valeurs initiales de différents objets comme les paquets ou la chargeuse. *FileHelper* gère tout ce qui à trait à la sauvegarde, à l'ouverture ou à la création d'une nouvelle cour à bois. *MathHelper* contient une méthode qui arrondi les chiffres qui doivent l'être. *CenteredRectangle* gère tout ce qui à trait aux rectangles des paquets. Finalement, *Point2D* s'occupe de créer toutes les variables de type *Point2D*, car la sérialisation ne permettait pas d'utiliser *Point2D* directement.

3.2.3 Java

Ce paquet est très simple, il s'agit du langage de programmation qui sera utilisé par le domaine, les *helpers* et l'interface utilisateur.

3.2.4 Enums

Le paquet *enums* contient la classe *EditorMode* de type *enumerate* qui énumère les différents modes dans lesquels l'utilisateur peut se retrouver. Également, le paquet contient la classe *DialogAction* qui contient les actions possibles au sein des fenêtres de dialogue.

3.2.5 Presentation

3.2.5.1 views

Start

La classe *Start* du paquet *view* sert à l'affichage de la première fenêtre lorsque l'application est lancée. À son lancement, une fenêtre *pop-up* apparaît à l'utilisateur pour qu'il crée une nouvelle cour, qu'il en charge une ou qu'il quitte tout simplement.

About

La classe *About* contient tant qu'à elle, un guide d'utilisation pour l'application.

Main

La classe *Main* de ce paquet contient toute l'information relative à la cour à bois. Elle contient tous les boutons pour le fonctionnement de la cour (grille magnétique, *undo*, *redo*, créer paquet, etc. Elle contient aussi toutes les zones d'informations pertinentes (sauf la zone pour éditer un paquet) ainsi que la cour à bois où il est possible de voir la chargeuse en action et la disposition des paquets de bois.

BundleEditor

La classe *BundleEditor* contient toute l'information relative à l'édition des informations d'un paquet mais aussi à l'affichage de ses propriétés en tout temps.

3.2.5.2 javafxControllers

IController

Ce contrôleur de JavaFX de type *Interface* fera guise d'interface pour l'application. C'est le contrôleur qui gère l'affichage et dont le contrôleur *BaseController* dépend pour fonctionner correctement. En d'autres mots, il met l'interface en place.

StartController

Tant qu'à lui, le *StartController* ne contient aucune méthode en soi. Comme JavaFX est

utilisé, le contrôleur est créé par défaut et est utilisé pour l’affichage de la fenêtre.

MainController

Le *MainController* gère tout ce qui touche l’affichage de l’information pertinente et des boutons. Ce contrôleur s’occupe de mettre à jour l’inventaire, lorsqu’un changement est fait à un paquet, il affiche toutes les informations sur le paquet et de la vue d’élévation lorsqu’un paquet est sélectionné et des informations de la chargeuse et de la souris en tout temps. En ce qui a trait aux boutons, il contrôle leur fonctionnement. Il s’assure que *undo*, *redo*, Créer Paquet et Grille Magnétique fonctionne de la bonne façon.

BaseController

Ce contrôleur, qui prend en argument l’objet créé par le *IController* s’assure du bon fonctionnement de la création, de l’ouverture ainsi que la fermeture de l’application. C’est de ce contrôleur qu’hérite le contrôleur *Main*, *Start* et *JavaFX*

3.2.5.3 javafx

Pane

Pane de JavaFX est l’équivalent de JPanel en Swing. Il s’agit d’une classe qui s’occupe de la disposition des éléments dans l’application. Il existe plusieurs types de Pane qui offre une expérience différente à l’utilisateur.

Rectangle

La classe *Rectangle* définit un rectangle avec une position et une forme prédéfinie par l’utilisateur. Ici, il s’agit des paquets qui seront créés par l’utilisateur qui sont des objets de la classe *Rectangle*.

Stage

La classe *Stage* constitue le plus gros conteneur défini par JavaFx permettant d’englober d’autres objets pour l’affichage.

Label

Cette classe permet de contrôler les zones de texte non éditables.

Line

La classe *Line* permet de créer des lignes à partir d’un point initial et d’un point final. Elle nous a surtout été utile lors de l’implantation de la grille magnétique.

Color

Cette classe est utilisée pour encapsuler des couleurs dans l’espace de couleur RGB.

3.2.5.4 JavafxPresenters

Les modèles JavaFX (JavaFXPresenters), aussi appelés *objets JavaFX*, sont des classes dérivées de certains *Nodes* de la librairie. Ce sont des objets affichables et qui possèdent en plus leurs DTOs associés.

IPresenter

Le *IPresenter* est le gestionnaire de l’affichage des *Presenters*. Il agit comme interface pour les modèles.

BundlePresenter

Le *BundlePresenter* contient toutes les informations des paquets afin de facilement pouvoir les afficher à l’écran. Ce modèle contient donc les DTOs de la classe comme attribut. Il contient un constructeur qui crée un paquet à partir de son Dto.

LiftPresenter

Le LiftPresenter contient toutes les informations de la chargeuse présente afin de faciliter l’affichage à l’écran. Ce modèle contient donc les DTOs de la classe comme attribut. Il contient la méthode privée *LiftPresenter* qui gère de façon individuelle ses propres événements, allégeant ainsi grandement les contrôleurs JavaFX.

YardPresenter

Le YardPresenter contient tous les attributs qui sont en lien avec l’affichage de la cours. Il contient plusieurs méthodes autant publiques que privées qui gèrent l’affichage des différents éléments que contient la cour.

ElevationViewPresenter

Le ElevationViewPresenter contient les attributs qui sont en lien avec l’affichage de la vue d’élévation de l’application. Elle contient des méthodes publiques qui gèrent l’affichage des paquets dans cette vue.

Chapitre 4

Diagrammes d'état de la chargeuse

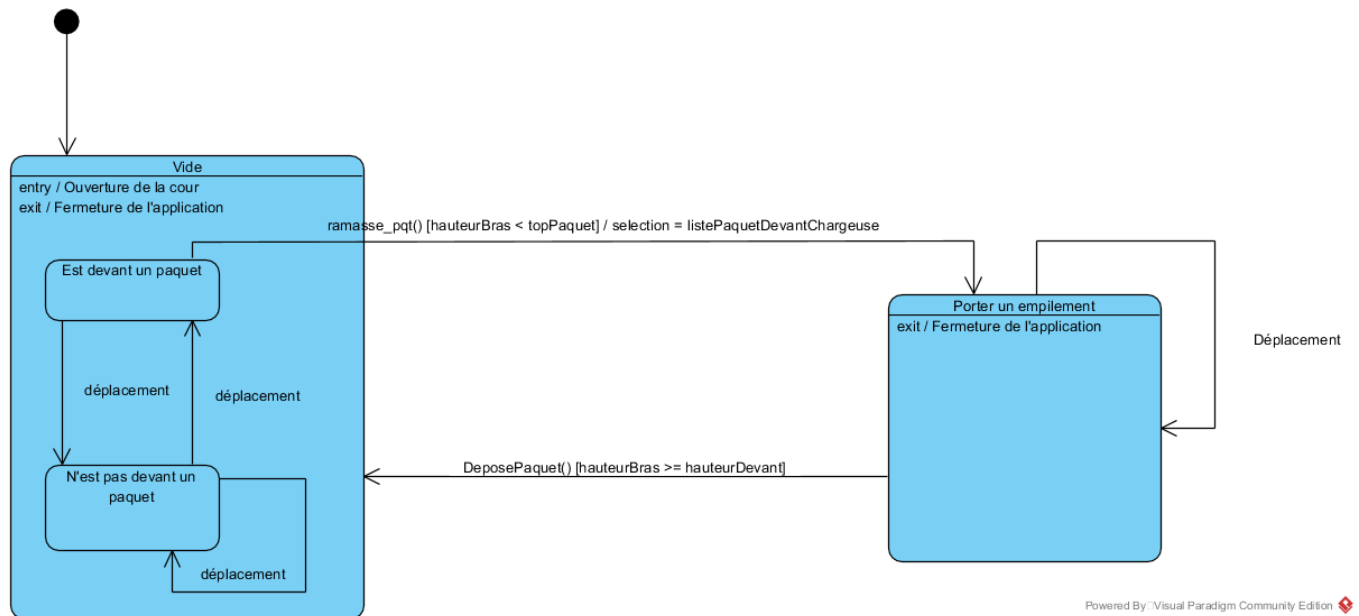


FIGURE 4.1 – Diagramme d'états de la chargeuse

Chapitre 5

Contribution des membres de l'équipe

Voici les différentes fonctionnalités que nous devons implémenter pour ce troisième livrable ainsi qu'une présentation de qui a fait quoi pour chacune d'elles :

- 1. Voir la cour à vol d'oiseau, zoomer / dézoomer à volonté sur différents points de la cour, se déplacer dans la vue :** Gabriel St-Pierre
- 2. Créer un paquet avec la souris :** Gabriel St-Pierre et Yoan Chamberland
- 3. Voir / éditer les infos d'un paquet :** Jordan Mayhue, Martine Deschênes et Yoan Chamberland
- 4. Sélectionner un empilement / paquet :** Gabriel St-Pierre, Martine Deschênes et Jordan Mayhue
- 5. Voir les infos d'un empilement :** Yoan Chamberland et Jordan Mayhue
- 6. Supprimer un paquet :** Jordan Mayhue
- 7. Déplacer un paquet à l'aide de la souris (seul le paquet sur le dessus d'un empilement peut être déplacé) :** Martine Deschênes et Jordan Mayhue
- 8. Voir un empilement en vue d'élévation :** Martine Deschênes et Jordan Mayhue
- 9. Grille magnétique :** Yoan Chamberland et Martine Deschênes
- 10. Sauvegarder / charger un fichier contenant le projet :** Yoan Chamberland et Jordan Mayhue