



Analyse VirtuCarrière

Livrable 4

Présenté à Thierry Eude

Par
Équipe 09

Matricule	Nom	Courriel
111 254 476	Maxime Miville Deschênes	maxime.miville-deschenes.1@ulaval.ca
111 158 759	Vincent Masse	vincent.masse.4@ulaval.ca
111 238 923	Francis Boulianne	francis.boulianne.2@ulaval.ca
111 240 093	Philippe Vincent	philippe.vincent.3@ulaval.ca

Énoncé de vision

À l'époque actuelle, la technologie évolue constamment et plusieurs domaines en sont affectés positivement. En effet, la plupart des problèmes d'auparavant sont maintenant résolus grâce aux multiples avancées technologiques. Par exemple, le domaine médical est maintenant muni d'un système informatique permettant de sauvegarder facilement toutes les informations sur les patients, sans avoir besoin de dossiers papier. Effectivement, peu importe dans quel domaine de travail une personne se trouve, il lui est pratiquement toujours possible de trouver une application qui va lui permettre de faciliter son travail.

Même lorsque cela peut sembler un peu anodin, une simple application peut grandement faciliter certaines tâches. Prenons la livraison de matériaux dans une carrière par exemple. Cette tâche qui semble plutôt simple peut devenir assez complexe pour un chauffeur qui ne possède pas d'expérience ou d'outils adéquats. Il serait donc très intéressant pour le chauffeur d'un camion d'avoir accès à une représentation visuelle avant un déchargement.

C'est ici que l'application VirtuCarrière intervient. Celle-ci permettrait à l'utilisateur de créer son trajet d'avance dans une certaine carrière et la personnaliser à son goût.

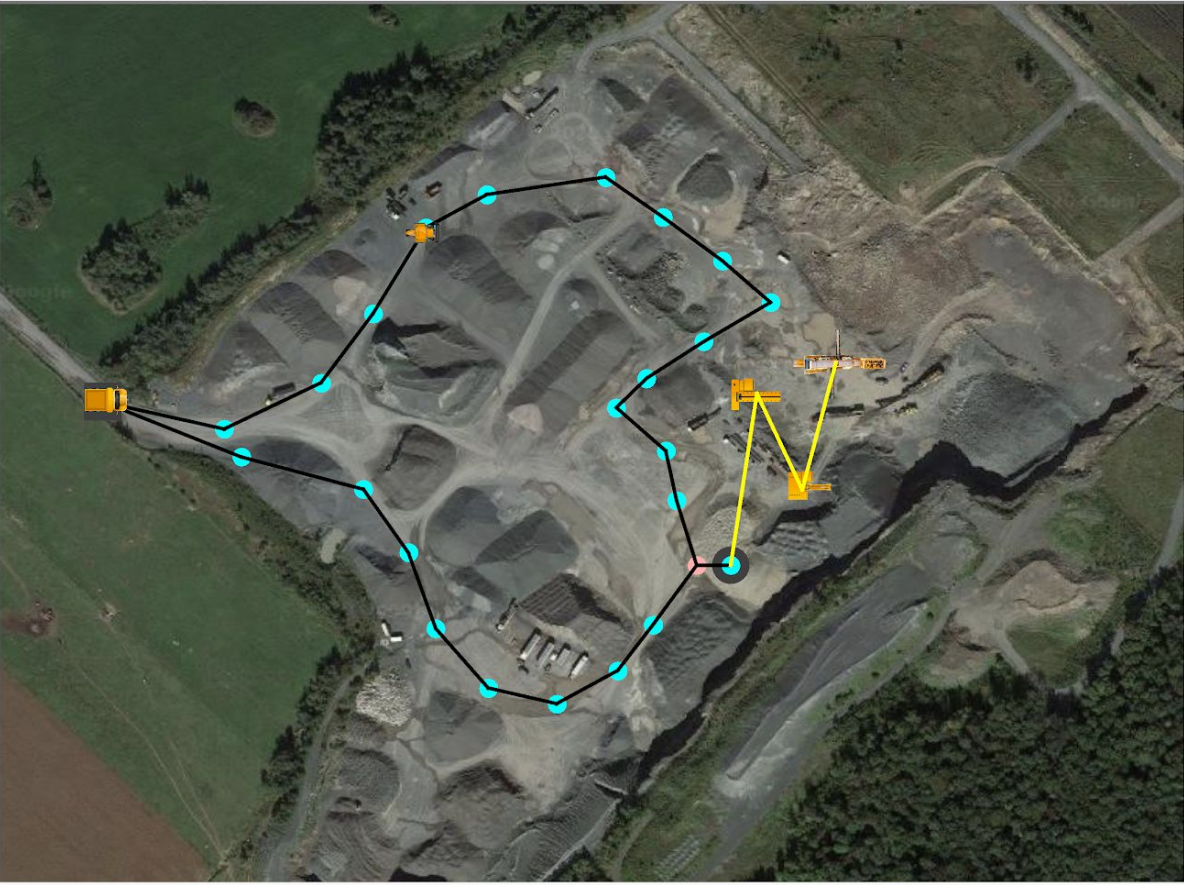
Premièrement, l'utilisateur est invité à créer une carrière ou à en charger une déjà existante. Deux modes sont disponibles, le mode plan ou le mode simulation. Le mode plan permet de modifier la carrière comme on le veut, en y ajoutant des éléments comme l'entrée de celle-ci, des concasseurs, des cribles, des tas de matériaux, des broyeurs ou encore des convoyeurs. Le mode de simulation permet quant à lui de visualiser la chaîne de production de la carrière.

En mode plan, l'utilisateur peut ajouter, modifier ou supprimer les éléments de la carrière à tout moment. Toutes les actions possibles peuvent être réalisées à l'aide de

la souris en symbiose avec le panneau permanent situé à la droite de l'écran. On peut facilement indiquer les principaux carrefours de la carrière, comme les lieux de chargement, avec des noeuds que l'on peut placer n'importe où sur le schéma. Il est possible de relier deux noeuds par un arc, créant ainsi un chemin. À partir du point de référence de la carrière, soit le centre de celle-ci, tous les points possèdent une position relative permettant de les localiser et de les situer par rapport aux autres éléments. Cette position est constamment affichée à l'écran lors du déplacement de la souris. Également, en sélectionnant un élément, l'utilisateur peut effectuer une translation en abscisse ou en ordonnée, mais aussi une rotation sur celui-ci.

Une fois que le plan de la carrière est finalisé, l'utilisateur peut le sauvegarder et quitter l'application ou encore utiliser le mode simulation de l'application pour reproduire le déplacement des camions et des chargeurs. À ce moment, la modification du plan n'est plus possible, l'utilisateur peut cependant placer des chargeurs et des camions où il désire. Après avoir défini les destinations des camions, on peut démarrer la simulation des transactions. On peut également la mettre sur pause, l'accélérer ou encore la ralentir.

Somme toute, malgré que VirtuCarrière reste une application assez simple d'utilisation, les nombreuses fonctionnalités offertes permettent sans aucun doute d'optimiser les livraisons de matériaux dans une carrière.



PlanSimulation

SélectionSupprimer

Ajouter un équipement

Convoyeur

Ajouter un noeud

Noeud

Ajouter un tas

Tas1-110

Modifier rayon du Tas

Ajouter/Modifier un noeud ou tas

X :Y :

NoeudTas

AjouterAjouter

ModifierModifier

Angle : 0

Modifier

Ajouter un Arc

Arc

Tracer un chemin

Chemin

Ajouter une entrée

Entrée

Élément(s) sélectionné(s) : 0

Modèle du domaine

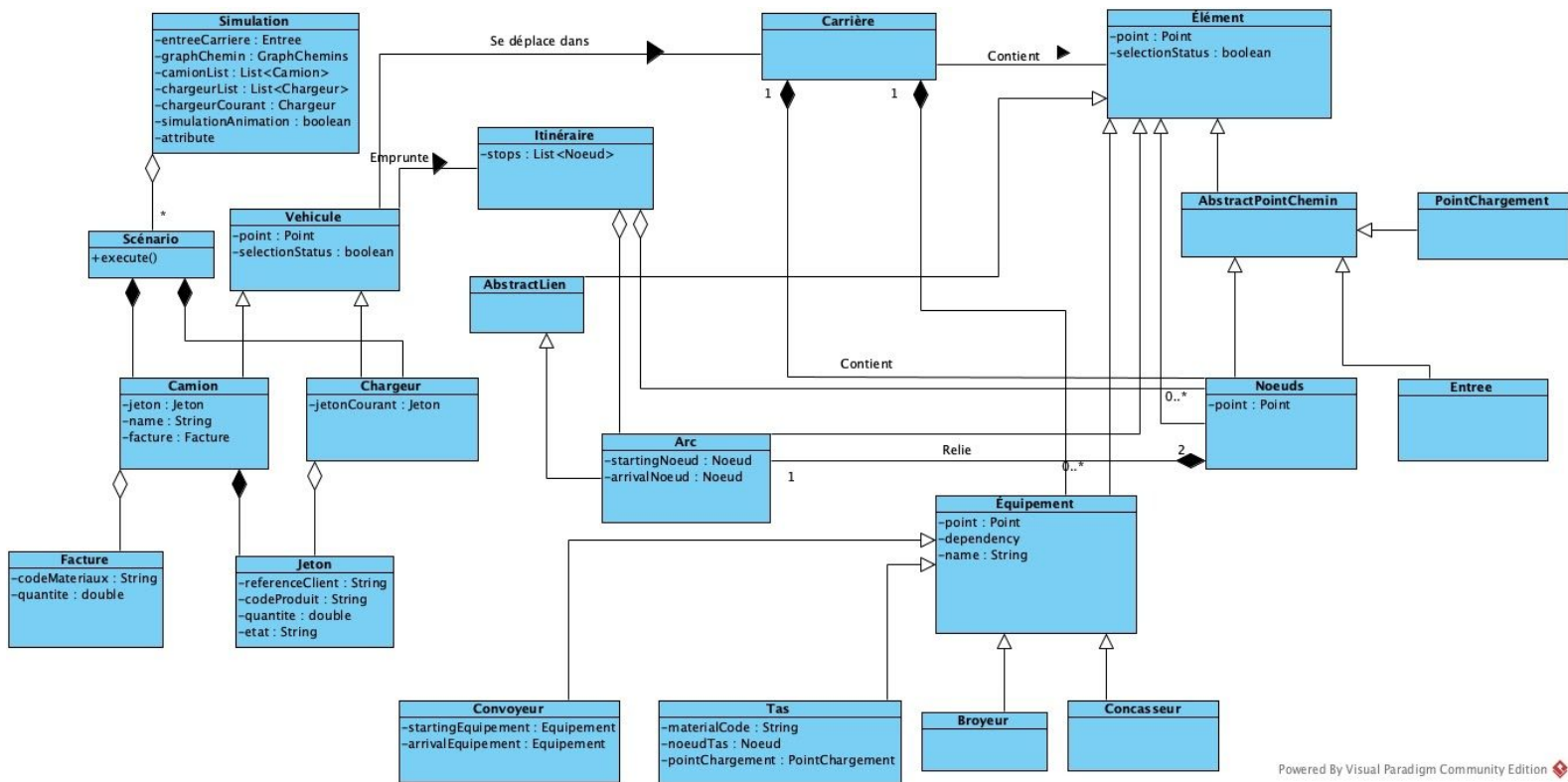
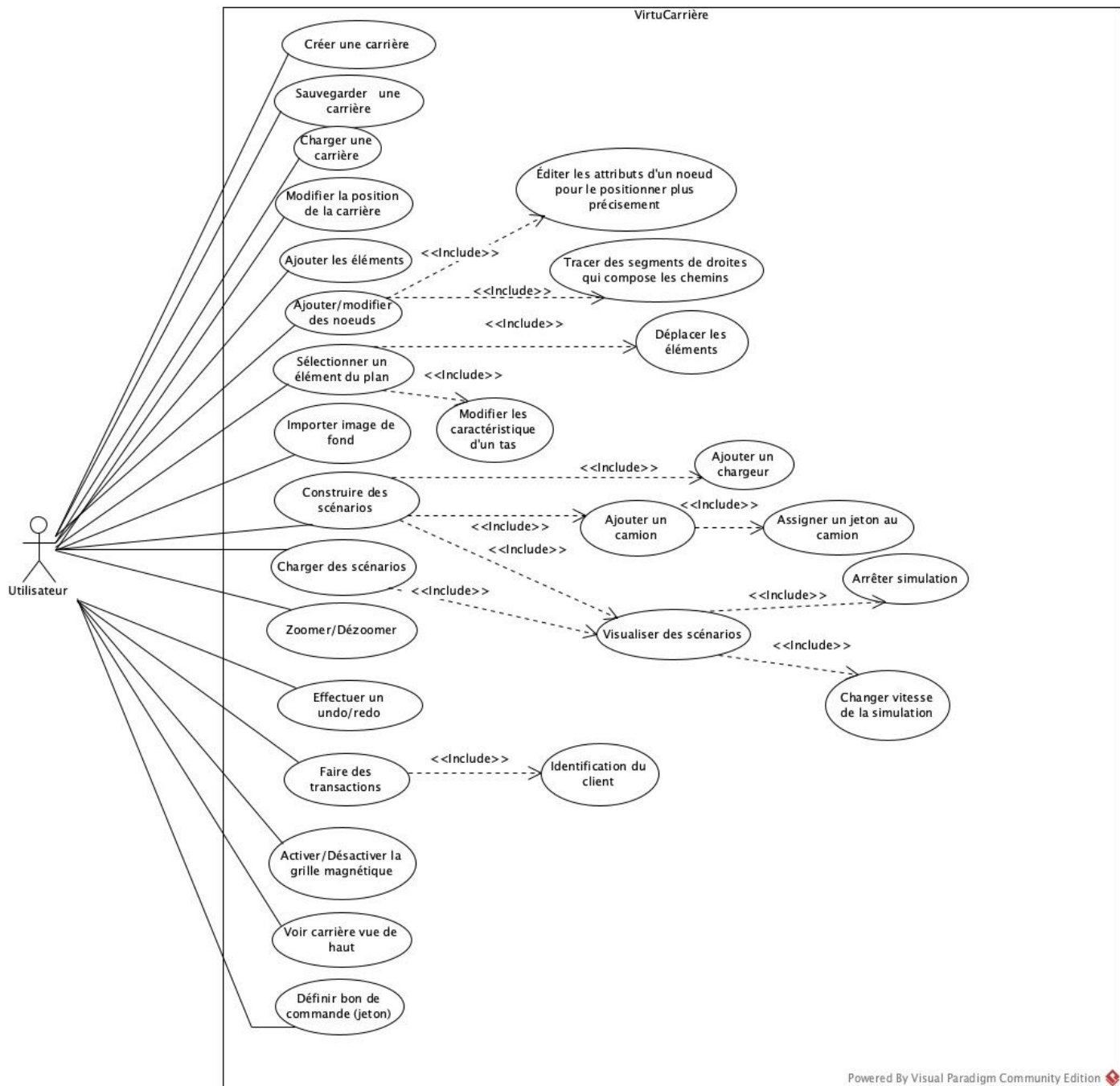


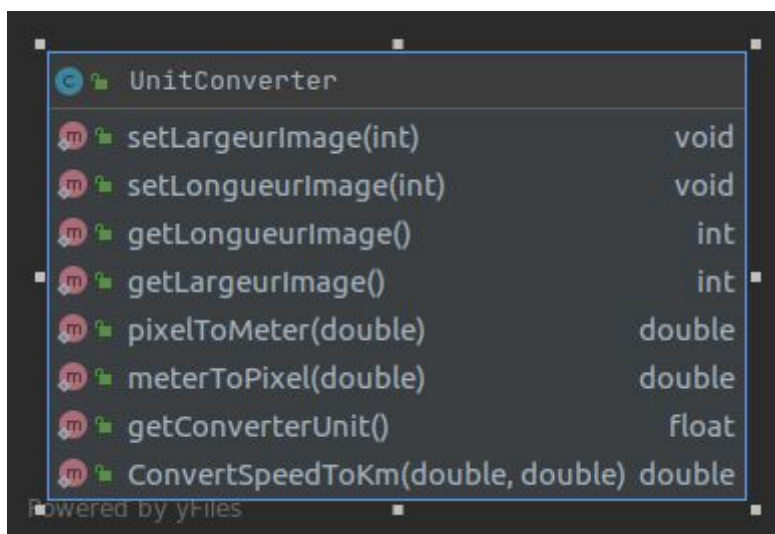
Diagramme des cas d'utilisation



Modèle de conception mis à jour

Comme pour le livrable 3, pour le modèle de conception, nous avons eu l'autorisation de Thierry Eude pour générer notre diagramme de classe grâce à IntelliJ. Nous avons choisi de présenter le modèle de conception par package pour faciliter le visionnement et nous avons inclus le modèle de package à la suite pour permettre de voir la composition de notre application. De plus, nous allons joindre les images ci-dessous en .png pour faciliter votre visionnement, si désiré.

Affichage Util



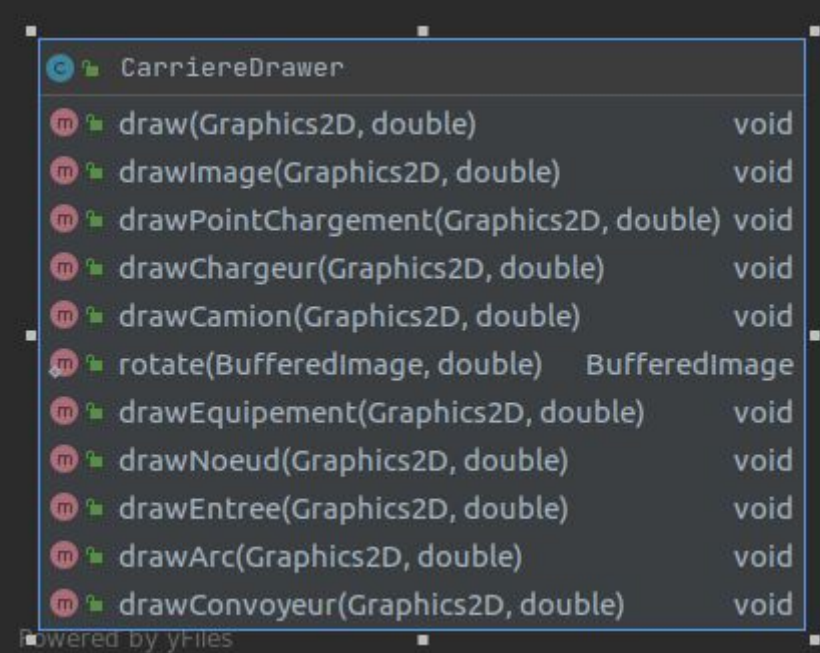
Controller

Controller	
initObserver()	void
update(String, Object)	void
addElementToStack(Action)	void
deleteElementsAfterPointer(int)	void
undo()	void
redo()	void
copy(Object)	Object
addConvoyeur(Point, EquipementModes, double)	void
addConvoyeurForPopup(Equipement, Equipement)	void
addCrible(Point, EquipementModes, double)	void
addConcasreur(Point, EquipementModes, double)	void
addBroyeur(Point, EquipementModes, double)	void
TrouverTasCorrespondant(List<Tas>, String)	Tas
EditCamion(Camion, String, String, double)	void
cheminDuCamion(Tas)	Vector<AbstractPointChemin>
addChargeur(Point)	void
addCamion(Point, String, String, double, int)	void
cheminDuCamionRetour(Tas)	Vector<AbstractPointChemin>
genererFacture(Camion)	Facture
createToken(String, String, double)	void
removeChargeur(Chargeur)	void
removeCamion(Camion)	void
clearEquipementConv()	void
choisirChargeurCorrespondant(Tas)	Chargeur
ChargeurCheminToPath(Chargeur, Tas, List<Noeud>)	Vector<AbstractPointChemin>
verificationJeton(Camion, Chargeur)	boolean
removeEquipement(Equipement)	void
addEquipement(EquipementModes, Point, double)	void
addTas(Point, String, double)	void
addNoeud(Point, EquipementModes, double)	void
changementSelectionStatus(double, double)	void
addChemin(Point)	void
removeNoeud(AbstractPointChemin)	void
removeArc(Arc)	void
removeConvoyeur(Convoyeur)	void
addEntree(Point, EquipementModes, double)	void
removeEntree(Entree)	void
switchSelectionStatus(double, double, boolean)	void
addArc(Point)	void
updateSelectedItemsPositions(double, double)	void
saveSimulation()	void
openSimulation()	void
openFile()	void
saveAs()	void
save()	void
newProject()	void
validateDependencies()	boolean
entree	Entree
convoyeurList	ArrayList<List<Convoyeur>>
element	ElementContainer
arcList	ArrayList<List<Arc>>
graphChemin	GraphChemins
camionList	List<Camion>
equipementList	List<Equipement>
allNoeuds	List<Noeud>
graphCheminSimulation	GraphChemins
urlBackground	URL
chargeurList	List<Chargeur>
noeudForArcList	List<Noeud>
pointsForArcList	List<AbstractPointChemin>
noeudList	List<AbstractPointChemin>













Powered by yFiles

ElementContainer	
initObserver(Controller)	void
update(String, Object)	void
notifyObservers(String, Object)	void
addObserver(Observer)	void
removeObserver(Observer)	void
switchSelectionStatus(double, double, boolean)	void
updateSelectedItemsPosition(double, double)	void
trouverChargeurCorrespondant(Tas)	Chargeur
ChargeurCheminToPath(Chargeur, Tas, List<Noeud>)	Vector<AbstractPointChemin>
trouverTasCorrespondant(List<Tas>, String)	Tas
cheminDuCamion(Tas)	Vector<AbstractPointChemin>
isNoeudPresent(Noeud)	void
isEquipementPresent(Equipement)	void
addArc(Point)	void
quickAddNoeud(AbstractPointChemin)	void
quickAddConvoyeur(Convoyeur)	void
quickAddArc(Arc)	void
addChemin(Point)	void
clearEquipementConv()	void
addElement(Point, EquipementModes, double)	void
removePlan(Element)	void
addTas(Point, String, double)	void
removeCamion(Camion)	void
quickAddCamion(Camion)	void
addCamion(Point, String, String, double, int)	void
changementSelectionStatus(double, double)	void
removeArc(Arc)	void
removeConvoyeur(Convoyeur)	void
quickAddChargeur(Chargeur)	void
addChargeur(Point)	void
EditCamion(Camion, String, String, double)	void
generateFacture(Camion)	void
verificationJeton(Camion, Chargeur)	boolean
cheminDuCamionRetour(Tas)	Vector<AbstractPointChemin>
removeChargeur(Chargeur)	void
genererFacture(Camion)	Facture
createToken(String, String, double)	void
removeEquipement(Equipement)	void
addEquipement(Equipement)	void
addConvoyeurForPopup(Equipement, Equipement)	void
removeNoeud(AbstractPointChemin)	void
snapSelectedElementToGrid(double)	void
getElement(Point)	Element
validateDependencies()	boolean
entree	Entree
backgroundUrl	URL
convoyeurList	ArrayList<List<Convoyeur>>
urlBackground	URL
arcList	ArrayList<List<Arc>>
file	File
graphChemin	GraphChemins
simulation	Simulation
camionList	List<Camion>
allNoeuds	List<Noeud>
equipementList	List<Equipement>
graphCheminSimulation	GraphChemins
chargeurList	List<Chargeur>
noeudForArcList	List<Noeud>
pointsForArcList	List<AbstractPointChemin>
noeudList	List<AbstractPointChemin>

Drawing

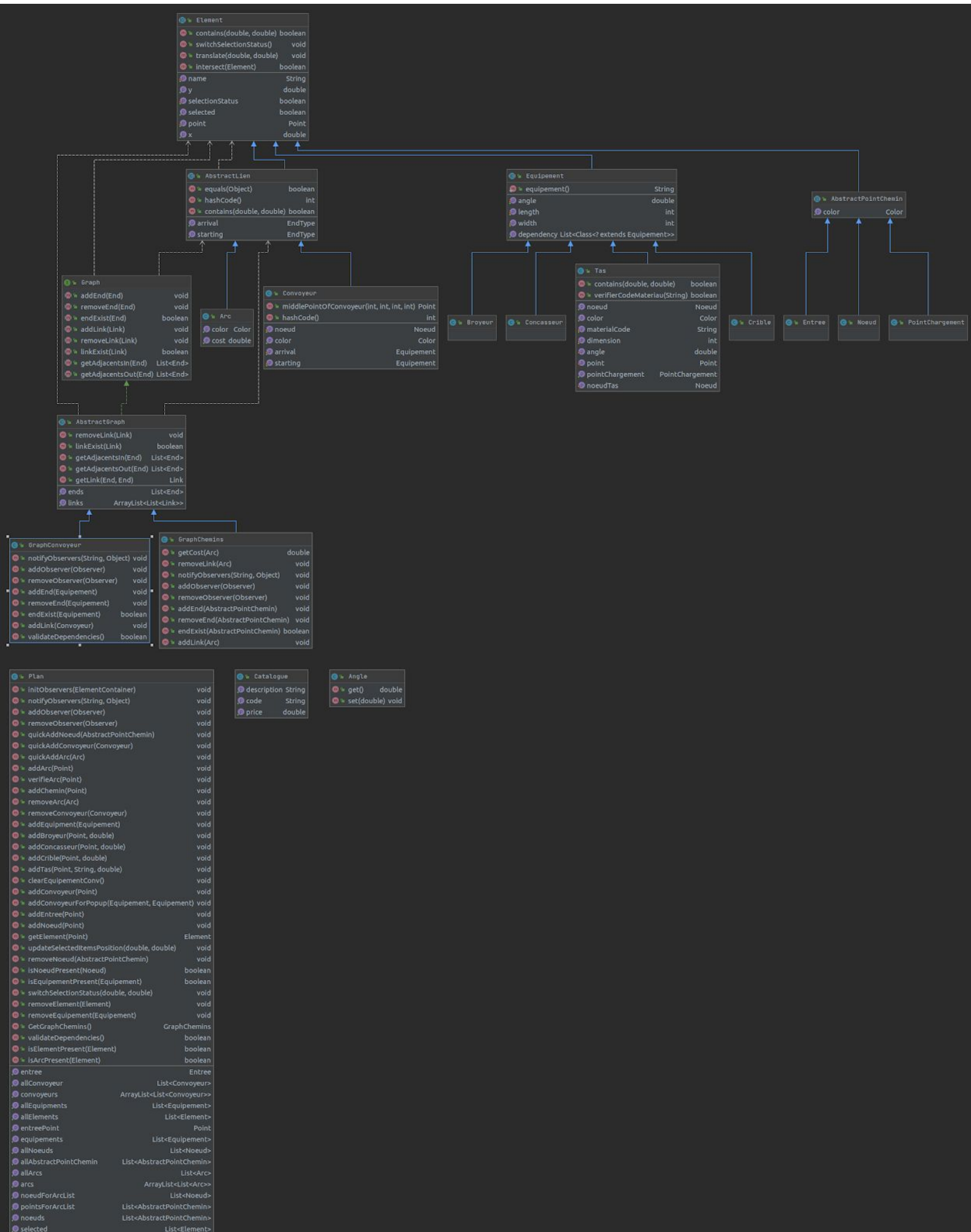


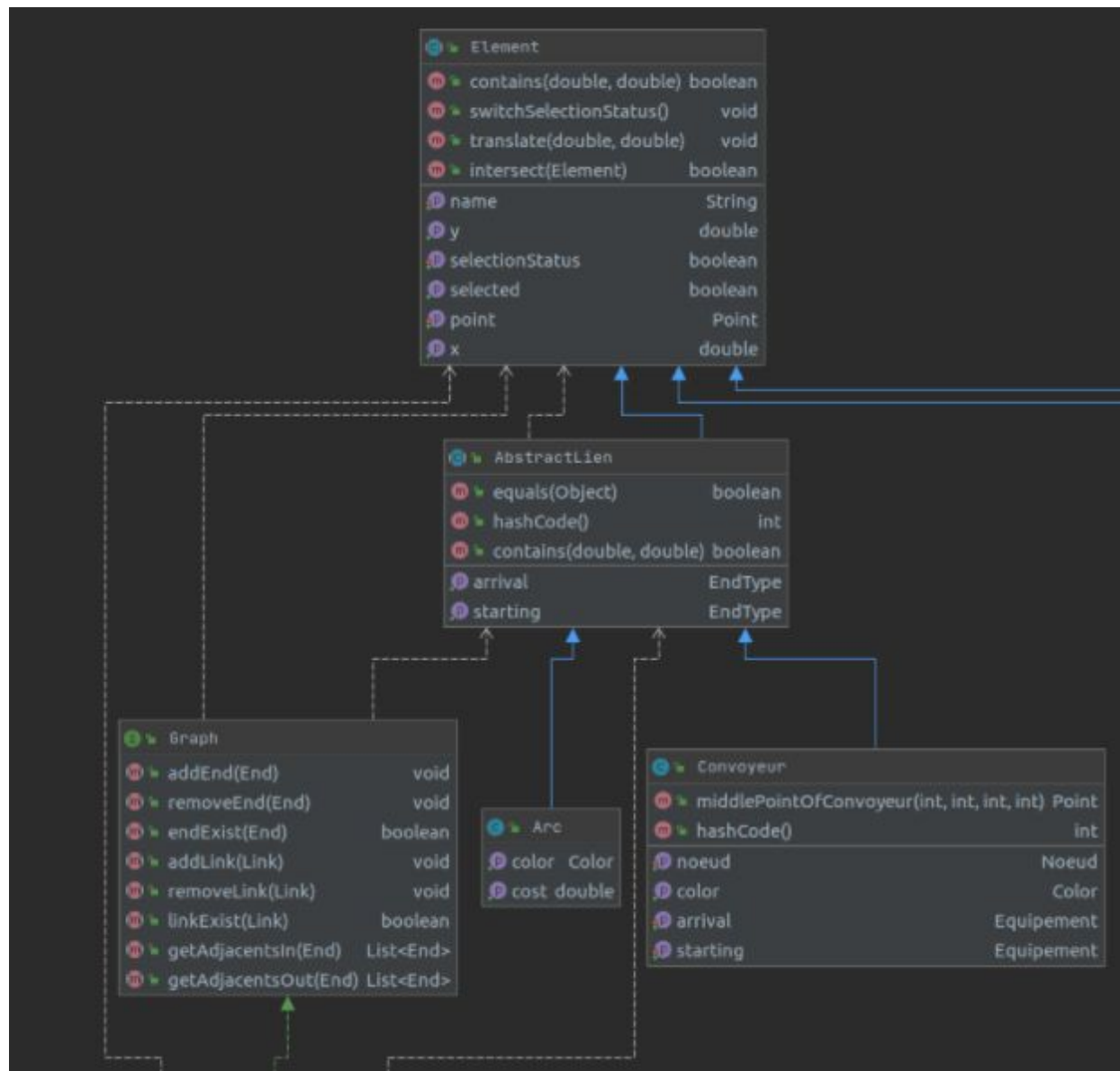
A screenshot of a Java IDE's method list for the `CarriereDrawer` class. The list contains 11 methods, each with a red 'm' icon, a green folder icon, the method signature, and the return type. The methods are: `draw(Graphics2D, double)` (void), `drawImage(Graphics2D, double)` (void), `drawPointChargement(Graphics2D, double)` (void), `drawChargeur(Graphics2D, double)` (void), `drawCamion(Graphics2D, double)` (void), `rotate(BufferedImage, double)` (BufferedImage), `drawEquipement(Graphics2D, double)` (void), `drawNoeud(Graphics2D, double)` (void), `drawEntree(Graphics2D, double)` (void), `drawArc(Graphics2D, double)` (void), and `drawConvoyeur(Graphics2D, double)` (void). The IDE interface is dark-themed with a blue border around the method list.

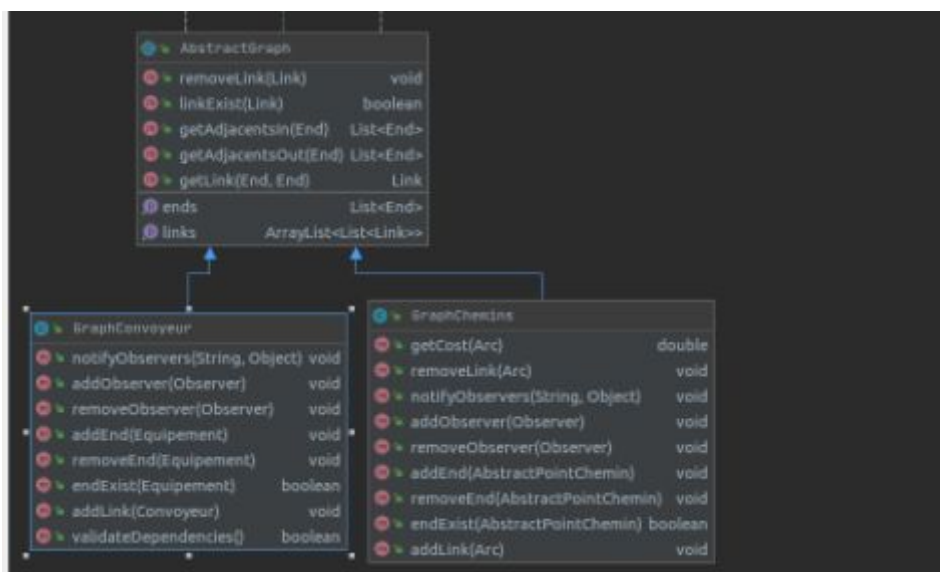
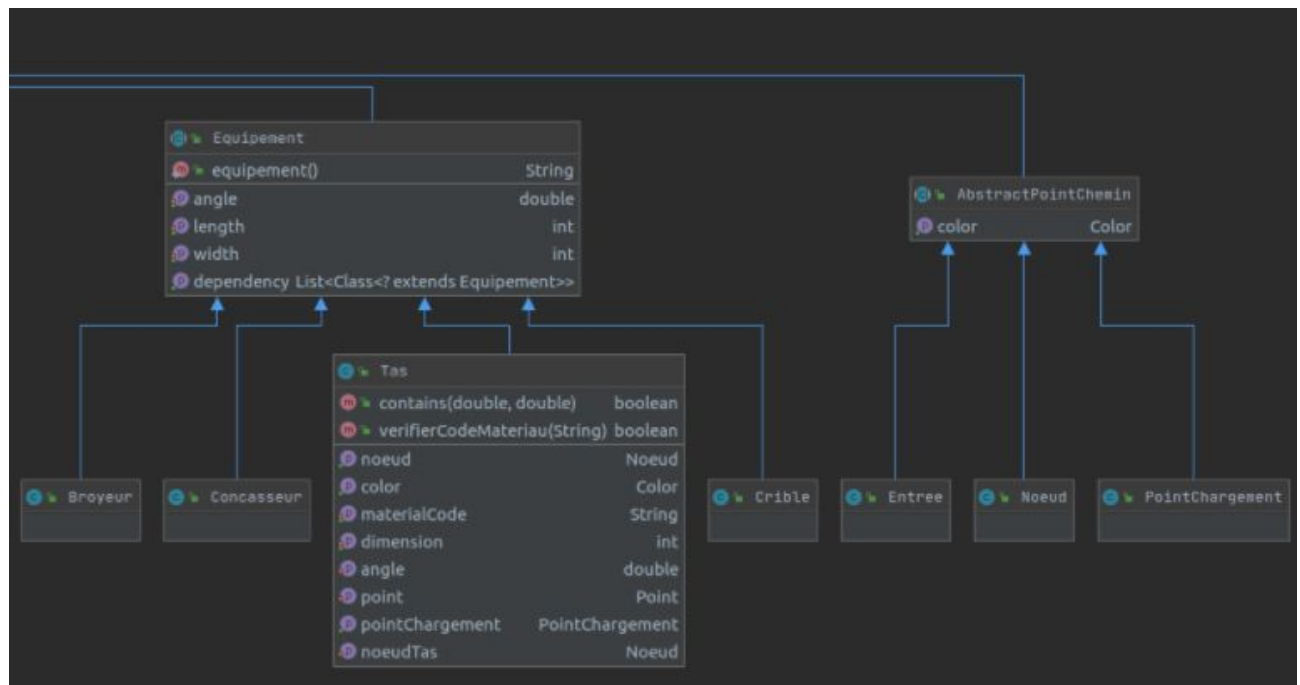
	<code>CarriereDrawer</code>	
	<code>draw(Graphics2D, double)</code>	<code>void</code>
	<code>drawImage(Graphics2D, double)</code>	<code>void</code>
	<code>drawPointChargement(Graphics2D, double)</code>	<code>void</code>
	<code>drawChargeur(Graphics2D, double)</code>	<code>void</code>
	<code>drawCamion(Graphics2D, double)</code>	<code>void</code>
	<code>rotate(BufferedImage, double)</code>	<code>BufferedImage</code>
	<code>drawEquipement(Graphics2D, double)</code>	<code>void</code>
	<code>drawNoeud(Graphics2D, double)</code>	<code>void</code>
	<code>drawEntree(Graphics2D, double)</code>	<code>void</code>
	<code>drawArc(Graphics2D, double)</code>	<code>void</code>
	<code>drawConvoyeur(Graphics2D, double)</code>	<code>void</code>

Powered by yFiles

Plan

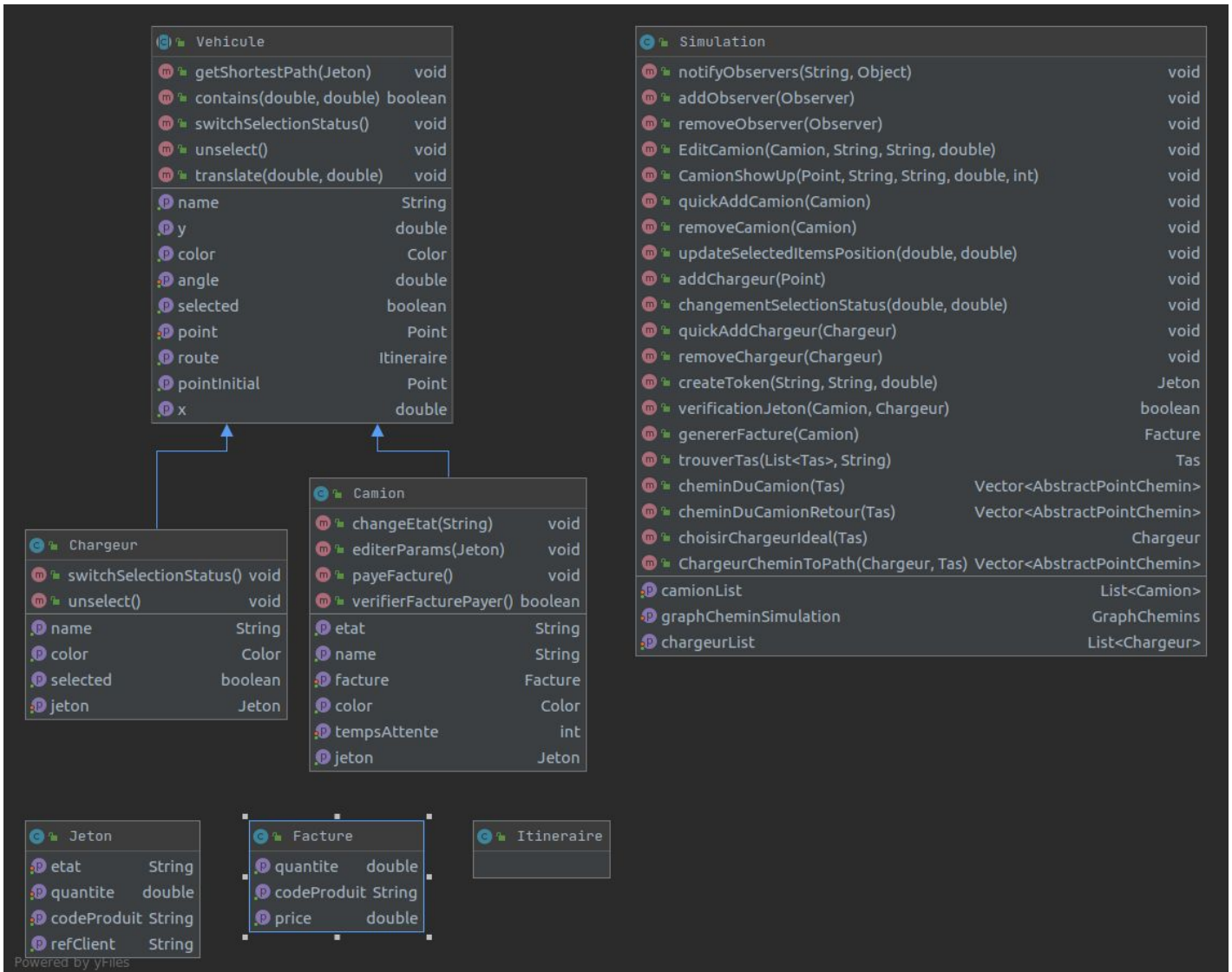






<div>Plan</div> <div> <div>initObservers(ElementContainer)</div> <div>void</div> </div> <div> <div>notifyObservers(String, Object)</div> <div>void</div> </div> <div> <div>addObserver(Observer)</div> <div>void</div> </div> <div> <div>removeObserver(Observer)</div> <div>void</div> </div> <div> <div>quickAddNoeud(AbstractPointChemin)</div> <div>void</div> </div> <div> <div>quickAddConvoyeur(Convoyeur)</div> <div>void</div> </div> <div> <div>quickAddArc(Arc)</div> <div>void</div> </div> <div> <div>addArc(Point)</div> <div>void</div> </div> <div> <div>verifieArc(Point)</div> <div>void</div> </div> <div> <div>addChemin(Point)</div> <div>void</div> </div> <div> <div>removeArc(Arc)</div> <div>void</div> </div> <div> <div>removeConvoyeur(Convoyeur)</div> <div>void</div> </div> <div> <div>addEquipment(Equipement)</div> <div>void</div> </div> <div> <div>addBroyeur(Point, double)</div> <div>void</div> </div> <div> <div>addConcasseur(Point, double)</div> <div>void</div> </div> <div> <div>addCrible(Point, double)</div> <div>void</div> </div> <div> <div>addTas(Point, String, double)</div> <div>void</div> </div> <div> <div>clearEquipmentConv()</div> <div>void</div> </div> <div> <div>addConvoyeur(Point)</div> <div>void</div> </div> <div> <div>addConvoyeurForPopup(Equipement, Equipement)</div> <div>void</div> </div> <div> <div>addEntree(Point)</div> <div>void</div> </div> <div> <div>addNoeud(Point)</div> <div>void</div> </div> <div> <div>getElement(Point)</div> <div>Element</div> </div> <div> <div>updateSelectedItemsPosition(double, double)</div> <div>void</div> </div> <div> <div>removeNoeud(AbstractPointChemin)</div> <div>void</div> </div> <div> <div>isNoeudPresent(Noeud)</div> <div>boolean</div> </div> <div> <div>isEquipmentPresent(Equipement)</div> <div>boolean</div> </div> <div> <div>switchSelectionStatus(double, double)</div> <div>void</div> </div> <div> <div>removeElement(Element)</div> <div>void</div> </div> <div> <div>removeEquipment(Equipement)</div> <div>void</div> </div> <div> <div>GetGraphChemins()</div> <div>GraphChemins</div> </div> <div> <div>validateDependencies()</div> <div>boolean</div> </div> <div> <div>isElementPresent(Element)</div> <div>boolean</div> </div> <div> <div>isArcPresent(Element)</div> <div>boolean</div> </div> <div> <div>entree</div> <div>Entree</div> </div> <div> <div>allConvoyeur</div> <div>List<Convoyeur></div> </div> <div> <div>convoyeurs</div> <div>ArrayList<List<Convoyeur>></div> </div> <div> <div>allEquipments</div> <div>List<Equipement></div> </div> <div> <div>allElements</div> <div>List<Element></div> </div> <div> <div>entreePoint</div> <div>Point</div> </div> <div> <div>equipements</div> <div>List<Equipement></div> </div> <div> <div>allNoeuds</div> <div>List<Noeud></div> </div> <div> <div>allAbstractPointChemin</div> <div>List<AbstractPointChemin></div> </div> <div> <div>allArcs</div> <div>List<Arc></div> </div> <div> <div>arcs</div> <div>ArrayList<List<Arc>></div> </div> <div> <div>noeudForArcList</div> <div>List<Noeud></div> </div> <div> <div>pointsForArcList</div> <div>List<AbstractPointChemin></div> </div> <div> <div>noeuds</div> <div>List<AbstractPointChemin></div> </div> <div> <div>selected</div> <div>List<Element></div> </div>	<div>Catalogue</div> <div> <div>description String</div> </div> <div> <div>code String</div> </div> <div> <div>price double</div> </div>	<div>Angle</div> <div> <div>get() double</div> </div> <div> <div>set(double) void</div> </div>
--	--	--

Simulation



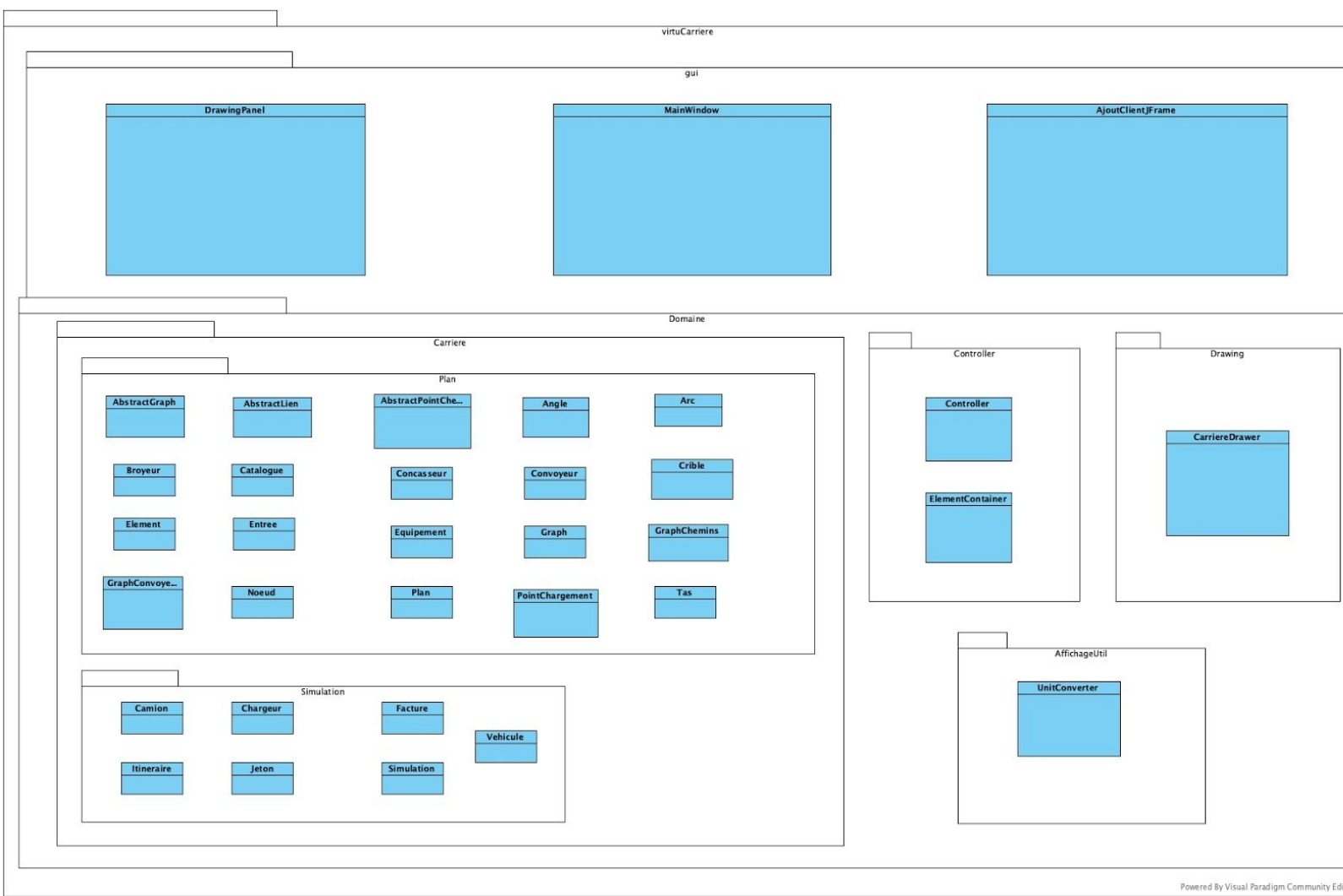
GUI

MainWindow		
f	controller	Controller
f	currentMousePoint	Point
f	initMousePoint	Point
f	simulationSpeed	int
f	intervalle	int
m	accelererSimulation()	void
m	ralentirSimulation()	void
m	pauseRestartSimulation()	void
m	angleOf(Point, Point)	double
m	cheminAllerSimulation(Camion)	void
m	simulationCamionRetour(Camion, Tas)	void
m	genererFacture(Camion, Jeton)	void
m	simulationTextField()	void
m	rafraichissementTextField()	void
m	main(String[])	void
p	mainScrollPane	JScrollPane
p	mode	EquipementModes
p	currentMeasurementUnitMode	MeasurementUnitMode
p	drawingPanel	DrawingPanel
p	mainScrollPanePosition	Point
p	vehicule	VehiculeModes
p	appMode	ApplicationMode
p	mainScrollPaneDimension	Dimension

Powered by yFiles

DrawingPanel		
f	graphics	Graphics
m	setGridLines()	void
m	zoomIn(Point)	void
m	zoomOut(Point)	void
m	setDrawingPanelDimensions()	void
p	gapGrille	double
p	mouseY	double
p	mouseX	double
p	initialDimension	Dimension
p	gridlines	boolean
p	mainWindow	MainWindow
p	zoom	double

Diagramme de packages



Conclusion

Point forts

Premièrement, l'interface de notre application est très intuitive pour l'utilisateur. En effet, parmi les plusieurs possibilités qu'elle offre, son utilisation est assez simple à comprendre. De plus, nous considérons comme un point fort le fait que l'utilisateur a beaucoup d'informations sur sa carrière personnalisée lors de son utilisation, comme l'emplacement des éléments sélectionnés, l'angle des équipements, etc., dans le panneau permanent. Ceci lui est affiché d'une façon claire et concise. Notre application possède donc toutes les fonctionnalités demandées et fonctionne particulièrement bien.

Points faibles

En général, l'application possède encore quelques "bugs" qui pourrait nuire partiellement à l'utilisation dans un contexte industriel. Ces "bugs" arrivent souvent lorsque l'utilisateur pousse un peu l'utilisation de l'application. Cependant toutes les fonctionnalités demandées sont présentes. Afin de régler ces bugs, il faudrait un peu plus de temps à notre équipe pour les régler et peaufiner notre application. Ceci permettrait à l'utilisateur une meilleure expérience.

Contribution de chacun des membres de l'équipe

Les différentes tâches du livrable 4 ont tous été complétées individuellement. Chacun des membres à contribué au projet sensiblement également. Certains ont plus de “commits” que d’autres, mais c’est en grande partie à la taille des commits. Vous trouverez les contributions de chacun des membres dans le tableau suivant.

Tâches	Personne qui l'a complété
Modèle du domaine	Maxime
Modèle de conception	Vincent
Diagramme des cas d'utilisations	Francis
GUI (Affichage)	Majoritairement Maxime avec l'aide des autres membres
Sauvegarde et ouverture de projets	Francis
Plan (où se retrouve tous les éléments du plan)	Vincent
Algorithme du plus court chemin	Vincent
Grille magnétique et zoom	Maxime
Exécuter une simulation	Philippe
Création d'équipement à l'aide de la souris	Maxime
Création d'un tas à l'aide de la souris	Maxime
Éditer les paramètres d'un matériaux à l'aide du panneau d'édition	Maxime
Sélection/désélection d'équipements, de tas	Maxime
Supprimer/déplacer un équipement/un tas à l'aide de la souris	Maxime
Équipements enchaînés reliés par des convoyeurs	Francis
Lorsqu'on supprimer un équipement, tous les convoyeurs en sortie le seront aussi	Francis
Ajouter un camion	Philippe

Ajouter un chargeur	Philippe
Créer, déplacer et supprimer des noeuds avec la souris	Maxime
Créer des arcs	Francis
Créer des convoyeurs	Francis
Éditer les paramètres d'un camion/chargeur à l'aide du panneau à édition	Maxime
Undo/Redo	Francis
Importation d'image de fonds	Philippe
Images pour équipements, véhicules	Maxime
Modification des angles des équipements, véhicules, tas	Maxime/Vincent
Pause/Play, Vitesse Simulation	Philippe