Projet de session VirtuTuile

Livrable n^{o} 3 – Mise à jour du rapport et des diagrammes

IFT-2007 : Analyse et conception des systèmes orientés objets

Automne 2019

Travail présenté à

Marc Philippe Parent

Réalisé par

Équipe 8

Gabriel Chevrette-Parrot	gabriel.chevrette-parrot.1@ulaval.ca	IFT	111 091 150
Petros Fytilis	petros.fytilis.1@ulaval.ca	IFT	111 074 144
Martin Sasseville	martin.sasseville.5@ulaval.ca	IFT	990 059 038

1- Énoncé de vision

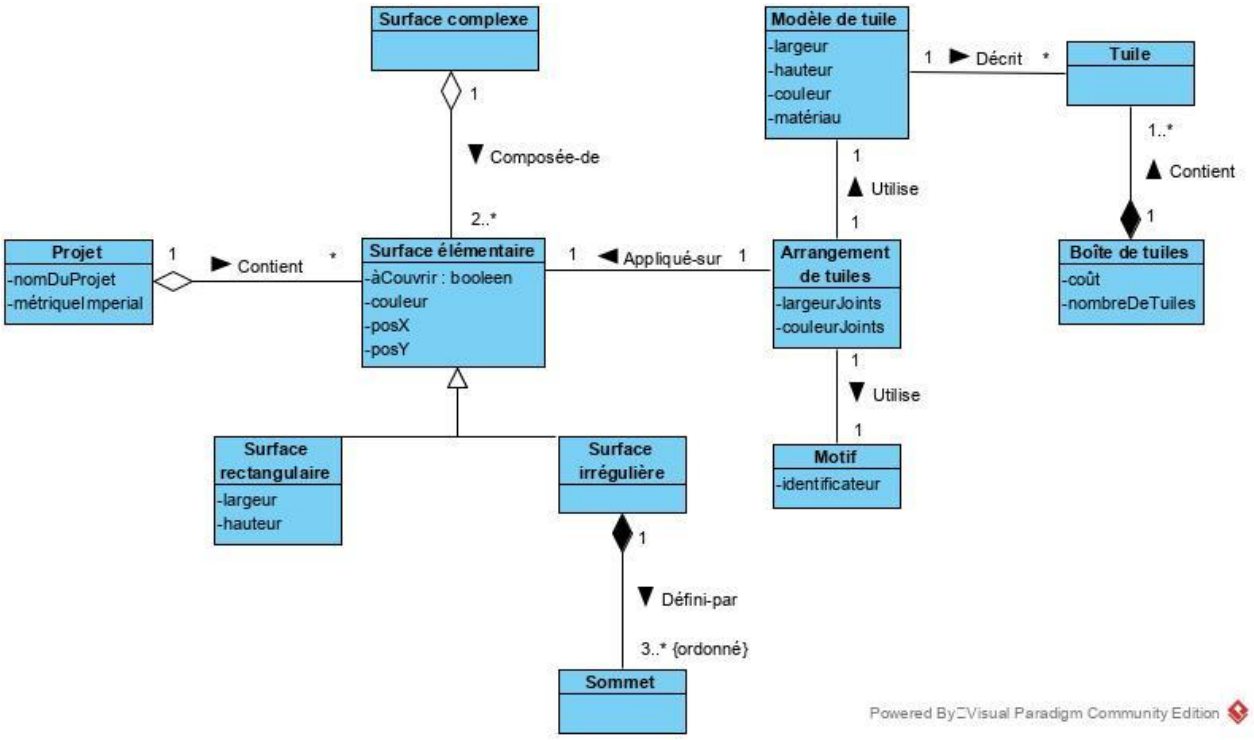
Nous voulons concevoir une application qui aidera les utilisateurs à planifier la pose de revêtement sur leurs planchers et murs. L'application offrira des outils aux utilisateurs qui leur permettront de modéliser ces surfaces en détail. L'application sera en mesure d'automatiquement simuler le recouvrement de ces surfaces par des tuiles dans différents motifs, puis de calculer le nombre de tuiles nécessaires à leur projet.

Le logiciel VirtuTuile pourra être offert aux clients qui désirent procéder à la planification de de leur rénovation de manière autonome, sans devoir ni rencontrer un professionnel ni se déplacer de leur chez-soi. L'application pourra également être disponible aux professionnels pour être utilisé dans leur travail.

Cette approche est fort intéressante car elle prend avantage de la tendance culturelle DIY («do-it-yourself»). De nos jours de plus en plus de gens désirent accomplir leurs projets de manière autonome pour 1) sauver de l'argent, 2) apprendre de nouvelles compétences et 3) en retirer plus de satisfaction personnelle. Nous avons ainsi une forte raison de croire que le produit sera populaire chez la clientèle des rénovateurs amateurs et professionnels.

Le logiciel permettra de créer les divers types de tuiles nécessaires au projet du client (incluant entre autres leurs dimensions et couleurs). Cette banque personnalisée pourra ensuite être éditée par le client pour tenir compte des ajustements que ce dernier voudrait effectuer.

2- Modèle du domaine



Un **projet** englobe toute la planification de revêtement d'une ou de plusieurs surfaces.

Un projet est ainsi une agrégation de **surfaces élémentaires**, soit des surfaces de plancher ou de mur. Chaque surface possède un attribut qui indique si elle doit être couverte ou pas.

Plus spécifiquement, une surface élémentaire peut soit être une **surface rectangulaire** ou une **surface irrégulière**. Une surface rectangulaire est définie par sa hauteur et sa largeur, tandis qu'une surface irrégulière est définie par ses sommets ordonnés.

Un **sommet** est un simple point cartésien.

Une surface complexe est une agrégation d'au moins deux surfaces élémentaires qui ont été combinées ensemble.

Une tuile est un objet de forme rectangulaire qui sert à couvrir des surfaces de plancher ou de mur.

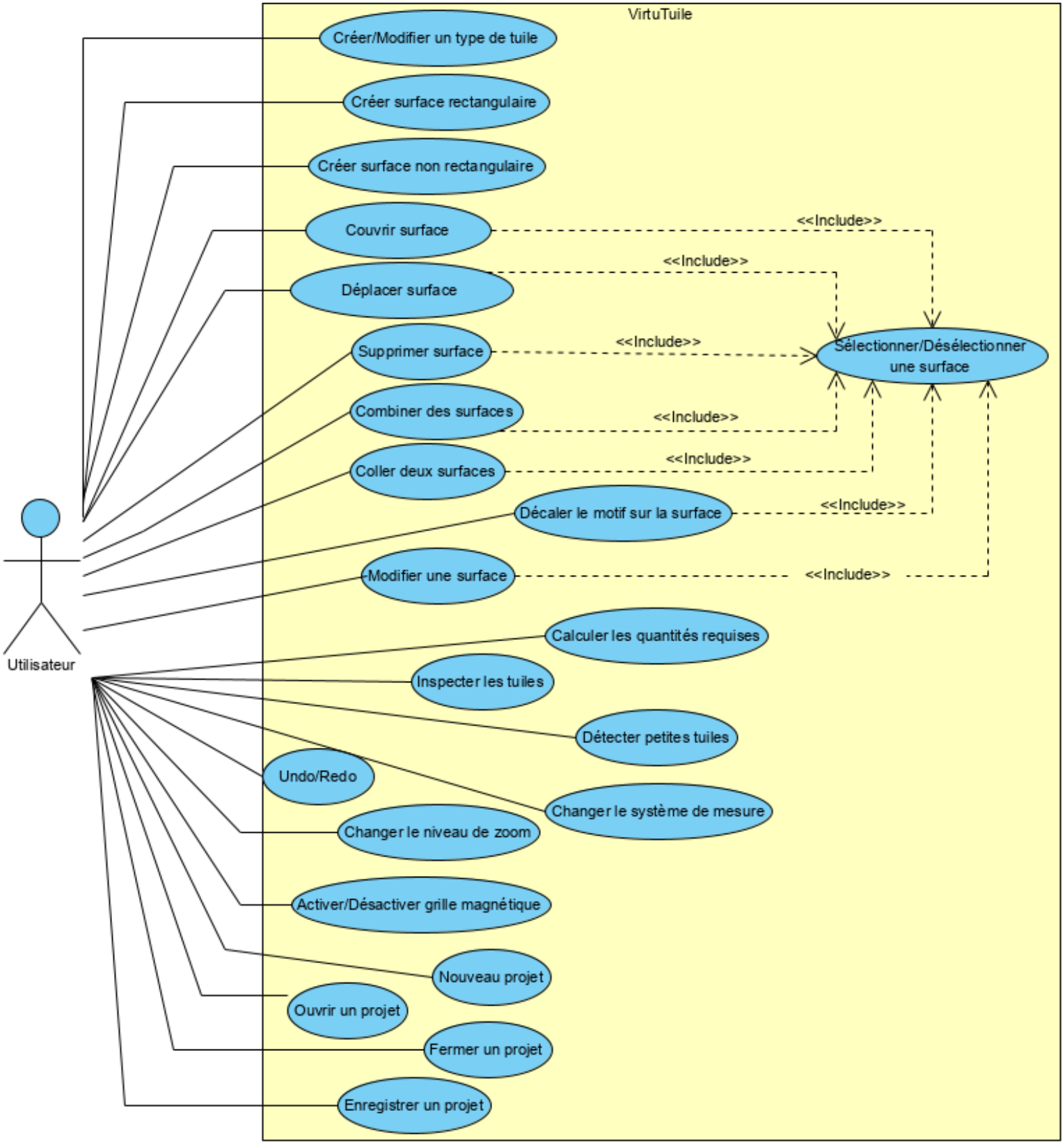
Une boîte de tuiles est un ensemble de tuiles qui peut être acheté à un certain prix. Elle contient un nombre prédéterminé de tuiles d'un modèle précis.

Un modèle de tuile sert à décrire certaines tuiles. Il possède comme attributs une hauteur et une largeur (soit les dimensions de la tuile), une couleur et un matériau (e.g. céramique).

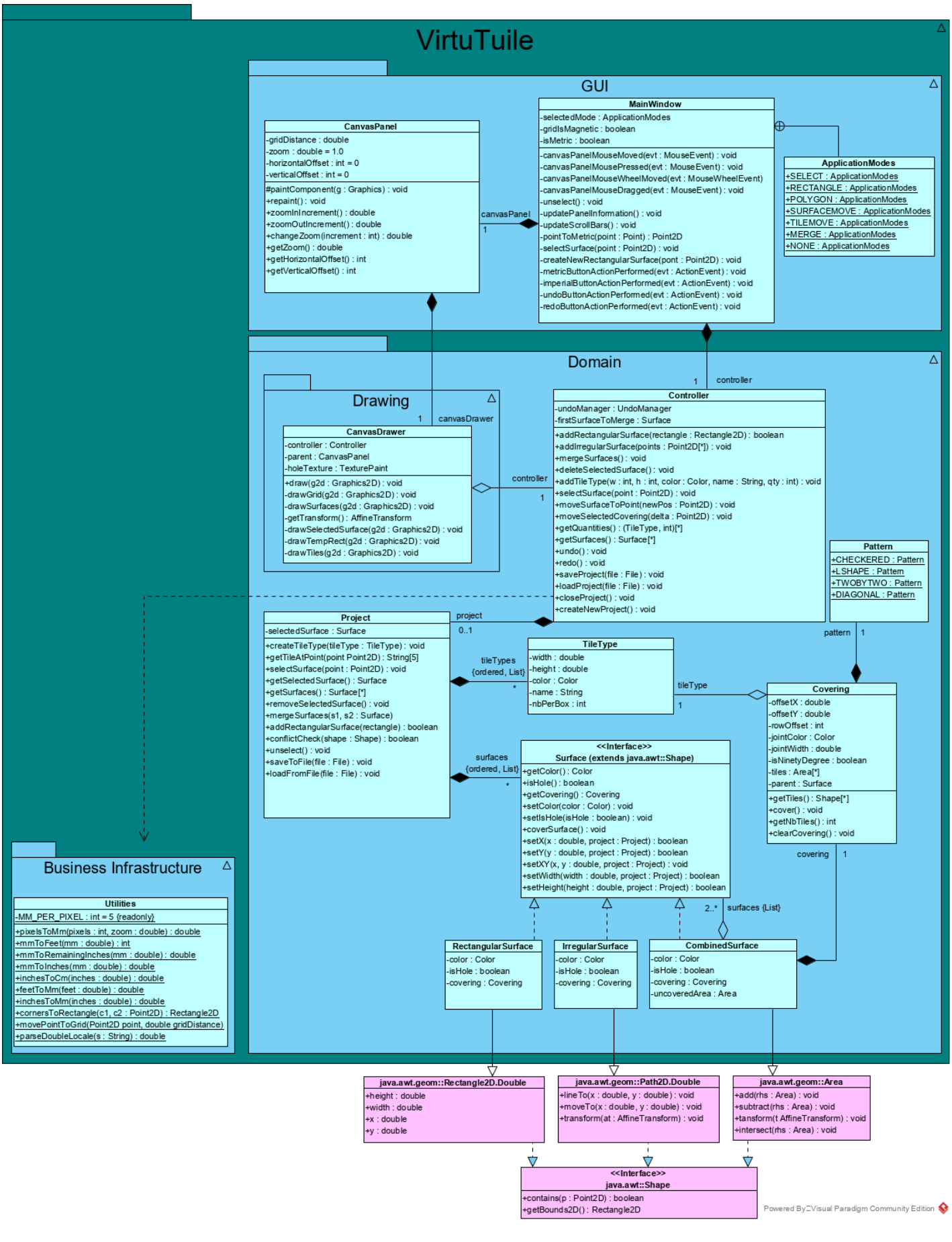
Un motif correspond à la façon d'arranger des tuiles ensemble. Nous travaillons avec 4 différents motifs, qui ont chacun un identificateur alphabétique.

Un arrangement de tuiles représente la manière précise qu'une surface de travail est couverte. Il utilise un certain modèle de tuile et un certain motif, puis est appliqué sur une certaine surface. Il possède également, comme attributs, une largeur de joints (soit l'espace entre chaque tuile individuelle) et une couleur de coulis.

3- Diagramme des cas d'utilisation



4- Diagramme de classe de conception



MainWindow: La fenêtre principale de l'application. Hérite de JFrame. C'est elle qui est responsable de créer tous les autres JComponents, en plus du Controller.

ApplicationModes: Une énumération (enum) de différents modes pour indiquer si l'utilisateur veut sélectionner une surface, créer un rectangle, créer un polygone, combiner des surfaces, déplacer une surface ou un motif. Cette énumération est définie dans le MainWindow.

CanvasPanel: Le canevas de dessin où les surfaces sont dessinées. Hérite de JPanel et est instancié dans le MainWindow.

CanvasDrawer: Il est responsable d'afficher le domaine sur le CanvasPanel en utilisant Graphics et Graphics2D. Il est instancié dans le CanvasPanel et possède une référence au Controller.

Controller: Il contrôle tous les accès entre l'interface graphique et le domaine. C'est sa responsabilité d'appeler les fonctions respectives du domaine lorsque l'utilisateur fait une requête à partir du GUI.

Pattern: Une énumération (enum) des quatre différents motifs de tuiles. Cette énumération est définie dans le Controller.

Project: Représente le projet courant. Il est composé d'une liste de TileType est d'une liste de Surface. Il est responsable de la sauvegarde et du chargement de projet.

TileType: Représente un type (donc une description) de tuile.

Covering: Permet de couvrir de tuiles une surface. Il possède une référence à un TileType et est instancié dans une Surface. Après avoir couvert une surface, il enregistre les tuiles sous la forme d'une liste d'objets java.awt.geom::Area.

Surface: L'interface de toutes nos surfaces. Possède une liste de méthodes qui devront être implémentées dans toutes nos classes de surfaces. Hérite de java.awt::Shape.

RectangularSurface: Une simple surface rectangulaire. Hérite de java.awt.geom::Rectangle2D.Double et implémente ElementarySurface.

IrregularSurface: Une surface irrégulière. Hérite de java.awt.geom::Path2D.Double et implémente ElementarySurface.

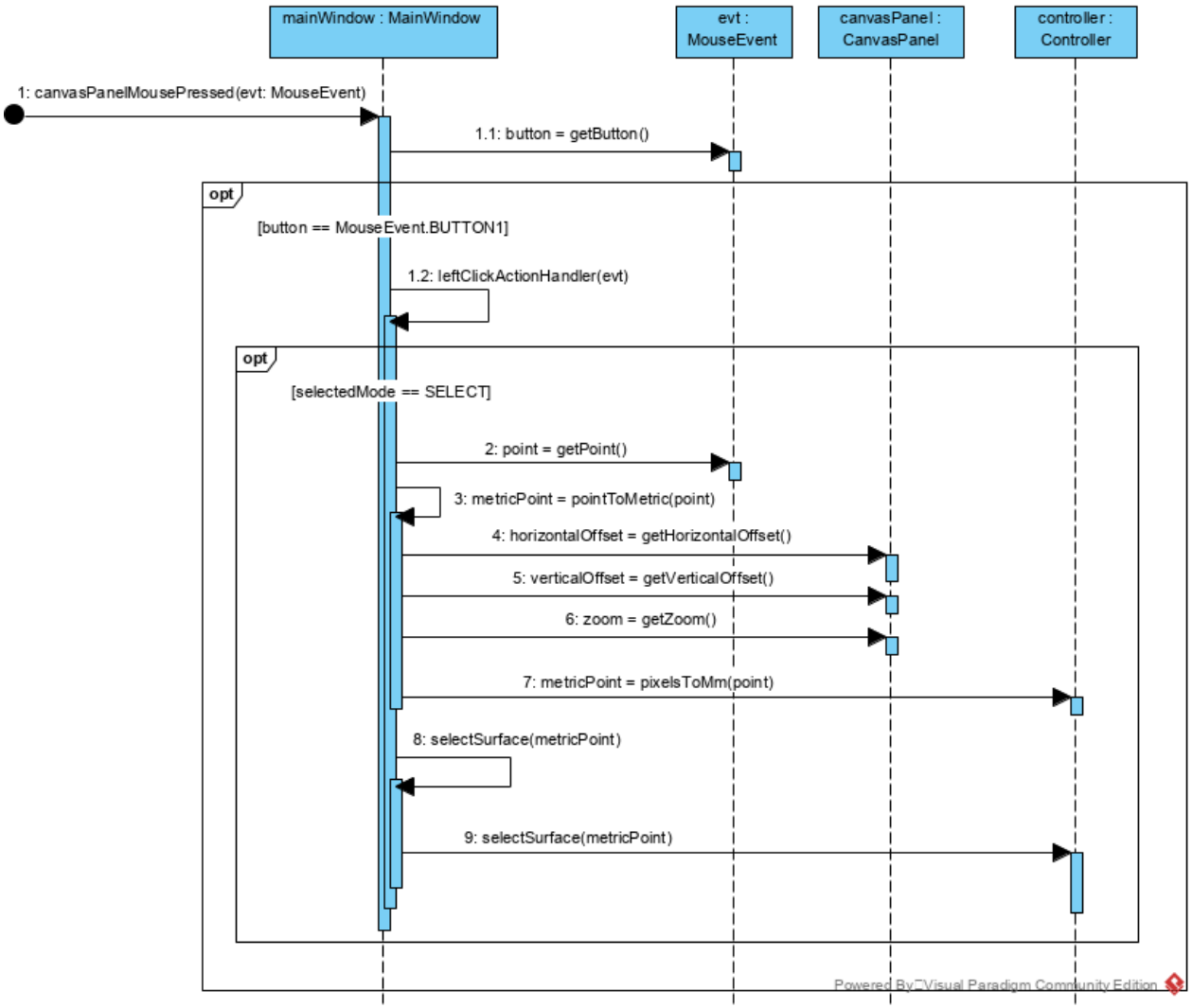
CombinedSurface: Une combinaison de surfaces. Hérite de java.awt.geom::Area et implémente Surface.

Utilities: Contient des fonctions statiques utilitaires pour, entre autres, la conversion entre les pixels, les mesures métriques et les mesures impériales.

3. Diagrammes de séquence de conception

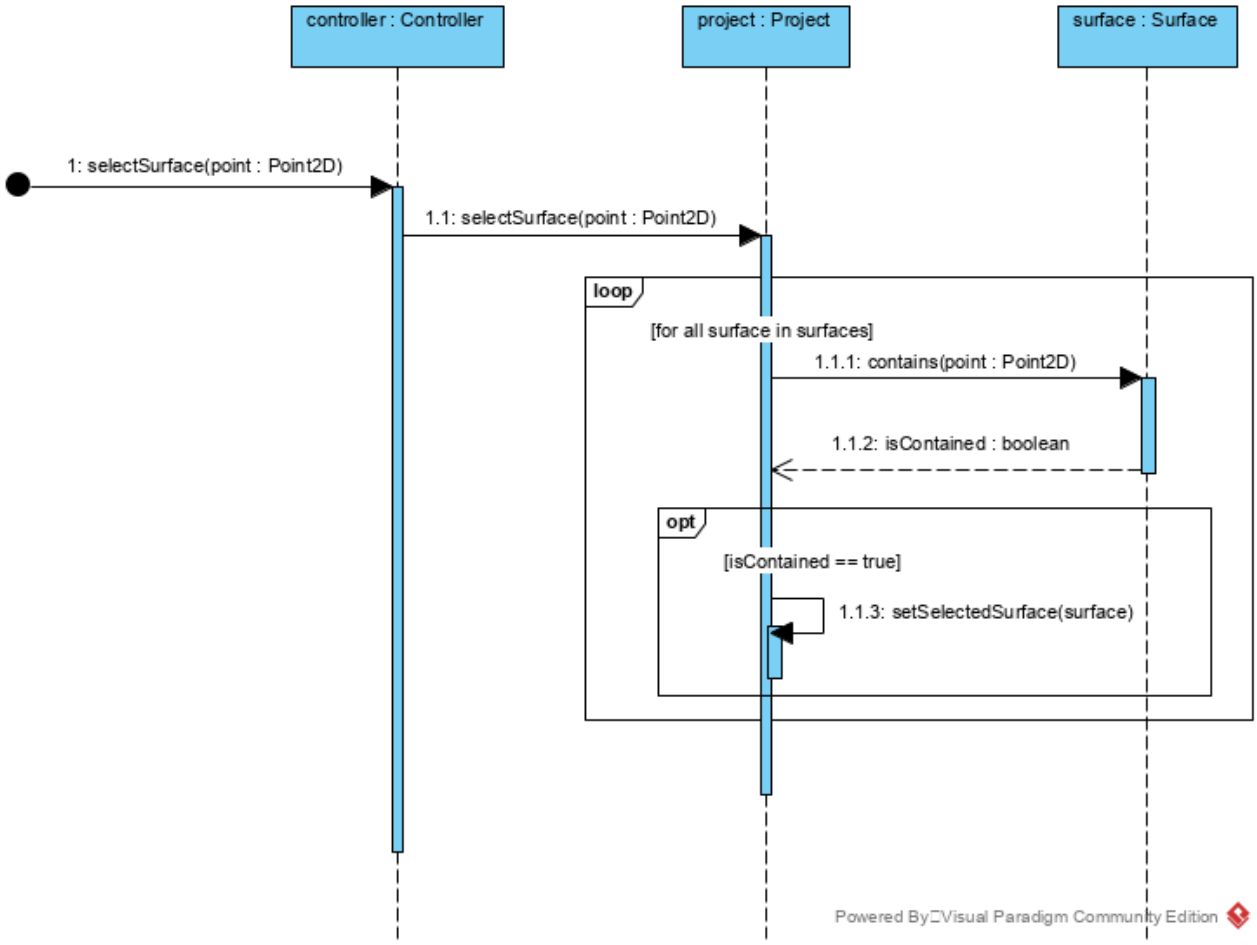
3.1. Déterminer la surface sélectionnée lors d'un clic de souris dans la vue en plan

3.1.1. Appel externe



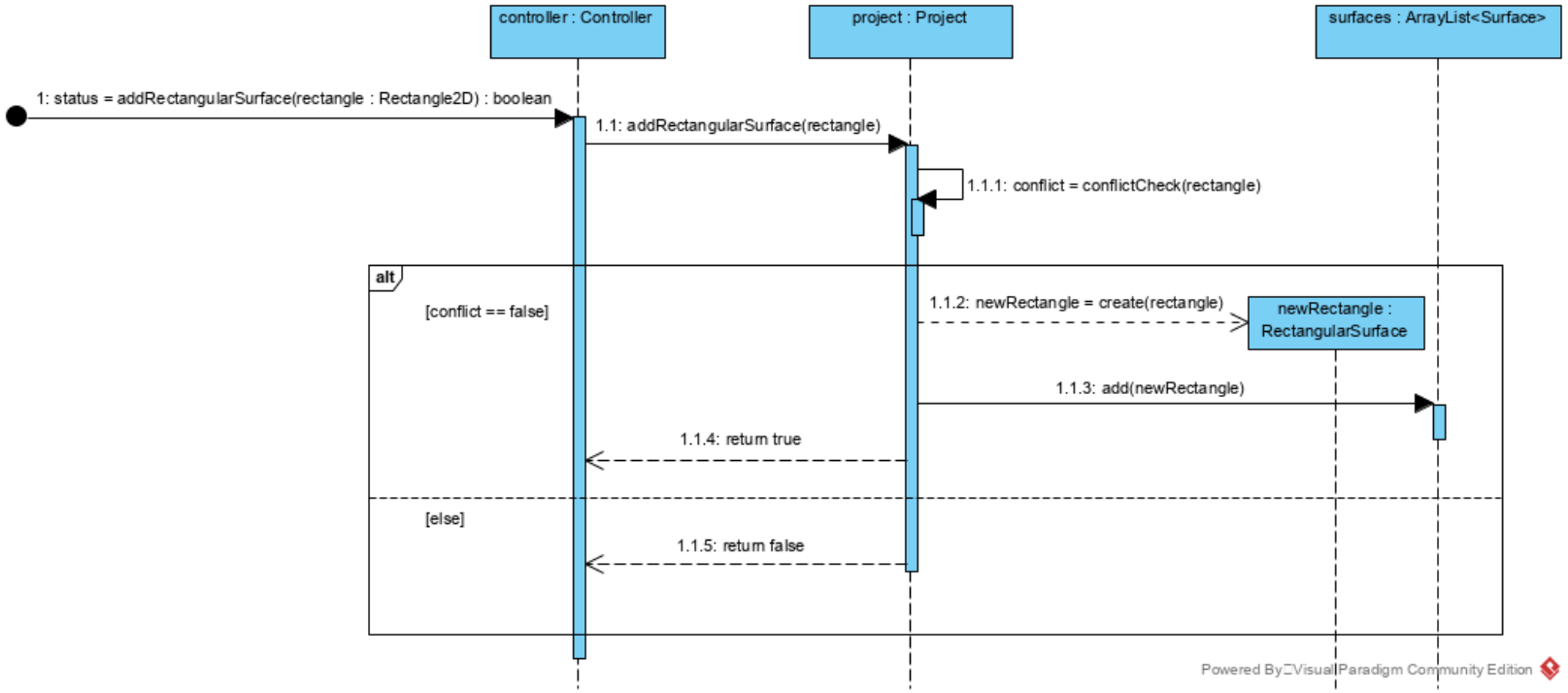
Le diagramme débute par un appel contenant un MouseEvent au MainWindow. Il vérifie premièrement que le MouseEvent provient d'un clic gauche de la souris, puis que la fenêtre est en mode SELECT. Il fait des appels au CanvasPanel pour obtenir les offsets horizontal et vertical du canevas (connectés aux scrollbars de la fenêtre) et le zoom actuel. Il demande au Controller de convertir le point en mesures métriques, puis envoie une requête de sélection de surface au Controller.

3.1.2. Appel au contrôleur



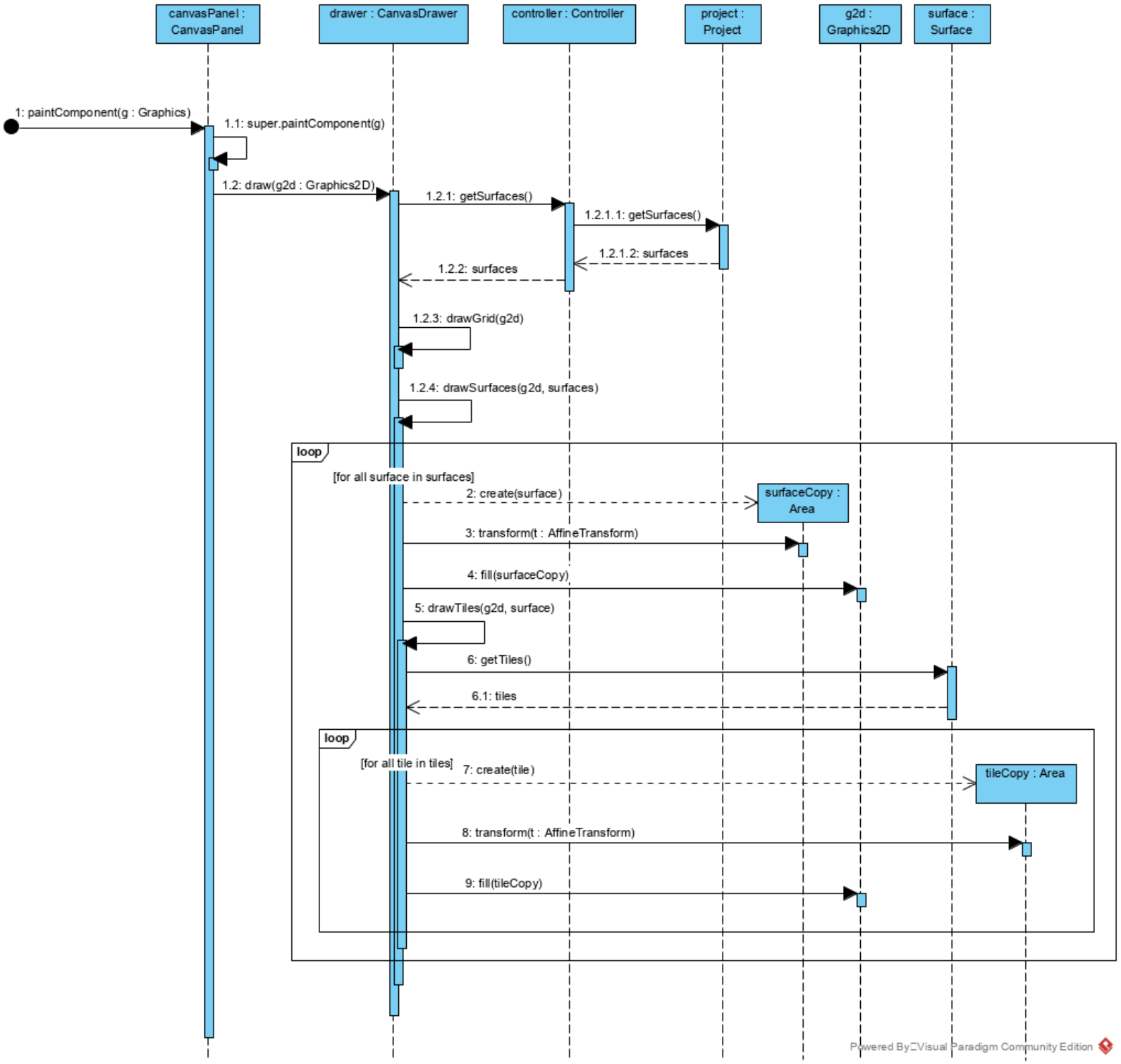
Le diagramme débute par un appel au Controller contenant les mesures métriques, représenté par un objet Point2D. Le Controller envoie ces mesures au Project. Le Project fait alors une boucle pour chaque surface de sa liste de surfaces. Si le point est contenu dans la surface, il désigne la surface comme étant la surface sélectionnée. Sinon, il ne fait rien.

3.2. Création d'une nouvelle surface rectangulaire



Le diagramme débute par un appel à la fonction **addRectangularSurface** du Controller. Cette fonction prend comme argument un objet de type Rectangle2D, décrivant les dimensions de la surface. Le Controller envoie la requête au Project, qui vérifie si les dimensions de la surface entrent en conflit avec les autres surfaces du projet. S'il n'y a aucun conflit, le Project crée la surface rectangulaire, l'ajoute à sa liste de surfaces, puis retourne **true**. S'il y a conflit, il retourne **false**.

3.3. L'affichage de la vue en plan



Texte explicatif sur la prochaine page.

La procédure d'affichage est déclenchée par l'événement qui appelle la méthode paintComponent du CanvasPanel. Le CanvasPanel appelle la fonction draw du CanvasDrawer avec un objet Graphics2D.

Le CanvasDrawer commence par acquérir la liste des surfaces du projet en faisant une requête au Controller. Ensuite, il affiche la grille avec la méthode drawGrid(). Puis, il affiche les surfaces du projet.

Pour afficher les surfaces du projet, il itère sur une boucle pour chaque surface du projet. Pour chaque surface, il crée une copie de l'aire de la surface, soit un objet Area. Il transforme ensuite l'objet Area avec la méthode transform() de la classe Area, en y passant un objet AffineTransform (décrivant le zoom et les offsets de l'affichage). Puis, il demande au Graphics2D de dessiner la surface avec la méthode fill(surfaceCopy).

Il faut ensuite dessiner les tuiles de la surface. Pour chaque surface encore, il appelle la fonction getTiles() de la surface pour acquérir la liste des tuiles de la surface. Il itère ensuite sur une boucle pour chaque tuile de la surface. Pour chaque tuile, il crée une copie de l'aire de la tuile, soit un objet Area. Il transforme ensuite l'objet Area avec la méthode transform() en y passant le même objet AffineTransform. Puis, il demande au Graphics2D de dessiner la tuile avec la méthode fill(tileCopy).

5- Justification de la contribution de chacun

Élément	Réalisé par
Création de surfaces rectangulaires à l'aide de la souris	Petros
Déplacement de surfaces à l'aide de la souris	Petros
Sélection/désélection de surfaces	Petros
Éditer les paramètres d'une surface à l'aide du panneau/panel d'édition (largeur des joints de coulis, dimensions de la surface, etc.)	Martin
Combiner des surfaces (incluant des trous) pour en faire de nouvelles surfaces	Gabriel
Affichage dans la barre d'état d'informations à propos de la tuile se trouvant sous le curseur de la souris	Martin Gabriel
Éditer les paramètres d'un matériau à l'aide du panneau/panel d'édition	Martin
Associer un matériau à une surface	Gabriel
Associer un motif simple à une surface (tuiles à la verticale ou à l'horizontal – mais pas les 2 dans le même motif - avec ou sans alignement/décalage entre les rangées paires ou impaires)	Gabriel
Quand on associe un matériau et un motif à une surface ou si on change la taille d'une surface par la suite le motif sera automatiquement étendu pour recouvrir celle-ci	Gabriel
Grille magnétique et Zoom (on doit zoomer autour de la position du pointeur de la souris)	Petros
Tout ce qui concerne l'affichage afin de pouvoir démontrer tout ce qui précède	Petros Gabriel