Tableau de hockey interactif

Document de conception – Remise 2

présenté à

Martin Savoie

par

The Javangers

matricule	*nom*	*signature*
111 127 868	Jérémie Bolduc	
111 126 228	Simon-Pierre Deschênes	
111 121 082	Émile Grégoire	
111 130 693	Alexandre McCune	

Université Laval

6 novembre 2016

Historique des versions		
version	*date*	*description*
0.0	19 septembre 2016	Création du document
0.1	2 octobre 2016	Première version après la phase d'inception et le début de l'élaboration
1.0	6 novembre 2016	Version pour la remise 2

Table des matières

Table des figures

Chapitre 1

Diagramme de classe de conception

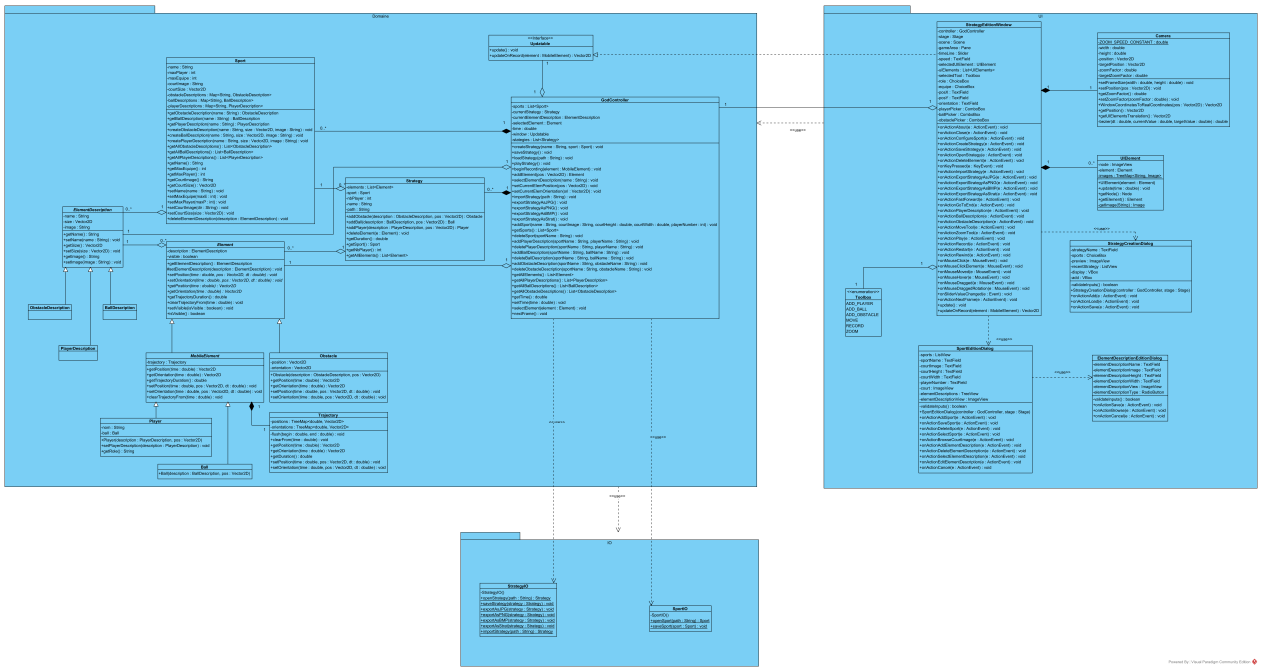

FIGURE 1.1 – Diagramme de classe de conception avec packages

Diagramme de classes de conception

Le diagramme de classes de conception est divisé en trois packages : le package UI, le package Domaine et le package IO. Ces trois package suivent le modèle en couches : la couche du UI utilise la couche du domaine, et la couche du domaine, quant à elle, utilise la couche utilitaire (IO dans ce cas-ci). Il n'y a donc pas de communications directes entre les couches inférieures et les couches supérieures. De plus, il y a le moins possible de communications entre les couches supérieures et les couches qui ne sont pas directement en-dessous d'elles.

UI

Le package du UI contient toutes les classes qui servent à faire l'affichage de l'interface graphique. Ce package est composé d'une classe servant de fenêtre principale, nommée StrategyEditionWindow et de classes servant à faire l'affichage des différents menus de configuration. Ces dernières se nomment StrategyCreation-

Dialog, SportEditionDialog et ElementDescritpionEditionDialog. De plus, ce package est aussi composé de deux classes métiers, servant à faciliter l'affichage, qui sont nommées UIElement et Camera. Finalement, il y a une énumération nommée Toolbox, qui contient tous les outils qu'il est possible de sélectionner dans l'interface graphique. Il est possible de voir dans le diagramme qu'à partir de la classe SportEditionDialog, il sera possible d'accéder à la classe ElementDescritpionEditionDialog. Cela est dû au fait que les éléments sont classés par sport : il est donc primordial de sélectionner le sport, et, par la suite, de sélectionner l'élément à modifier dans le sport. Les différents éléments d'un sport qui peuvent être modifiés sont les rôles des joueurs, les obstacles et la ou les balles de ce sport. Au moment de l'ouverture de ElementDescritpionEditionDialog, selon l'élément sélectionné dans l'interface graphique, on pourra modifier l'élément voulu. Il sera aussi possible d'ajouter l'élément voulu dans le sport sélectionné.

Domaine

Le package du domaine contient toutes les classes qui font partie de la logique de l'application. Ce package est composé de classes métiers, comme les classes Element, MobileElement, Obstacle, Player, Ball, Trajectory, ElementDescription, ObstacleDescription, PlayerDescription, BallDescription, Sport et Strategy. Ce package est aussi composé d'un contrôleur, qui, dans le cas présent, est nommé GodController. Finalement, il y a une interface servant à faire le pont entre le package du domaine et le package du UI, qui s'appelle Updatable. Les principes qui ont été utilisés dans ce package sont les suivants :

- Le contrôleur : Dans notre cas, c'est la classe GodController qui joue le rôle de contrôleur. Elle permet aux autres packages d'utiliser le package du domaine. C'est la seule classe qui est utilisée directement par les autres packages. Elle permet de relayer les appels de l'extérieur aux bons objets du domaine.

- L'indirection et la protection contre les variations : Dans notre cas, l'interface Updatable permet de faire des appels à l'interface graphique, tout en n'ayant aucune idée de comment cette dernière est faite. Pour pouvoir fonctionner correctement, la classe de la fenêtre n'a qu'à implémenter cette interface. Cela permet d'immuniser notre package du domaine des changements pouvant survenir dans le package du UI.

- Le polymorphisme : Dans le package du domaine, il y a de l'héritage partout où c'est possible. Cela permet d'utiliser le plus possible de polymorphisme. Effectivement, le principal exemple d'héritage dans notre programme est l'arborescence des Element. En effet, à la racine de l'arborescence, il y a la classe Element, dont les classes MobileElement et Obstacle héritent. Ensuite, les classes Player et Ball héritent de la classe MobileElement. À chaque niveau de l'arborescence, des méthodes de plus en plus spécifiques sont ajoutées aux classes.

- Les autres principes : De plus, nous essayons le plus possible d'utiliser les autres principes dans le package du domaine. En effet, dès que cela est applicable, nous essayons d'utiliser les principes d'expert en information, de créateur, de forte cohésion et de faible couplage.

IO

Le package IO est un petit package qui ne sert qu'à enregistrer différentes choses dans des fichiers ou à charger différentes choses à partir de fichiers. Ce package est composé de deux classes servant à l'enregistrement, au chargement, à l'importation et à l'exportation de fichiers : il s'agit des classes StrategyIO et SportIO. Puisque les méthodes de ces classes sont statiques, il est possible de les utiliser sans avoir besoin de créer d'instance. C'est pourquoi il n'y a pas de relation de composition ou d'agrégation avec ces classes.

Chapitre 2

Diagramme de séquence de conception

2.1 Conversion de coordonnées

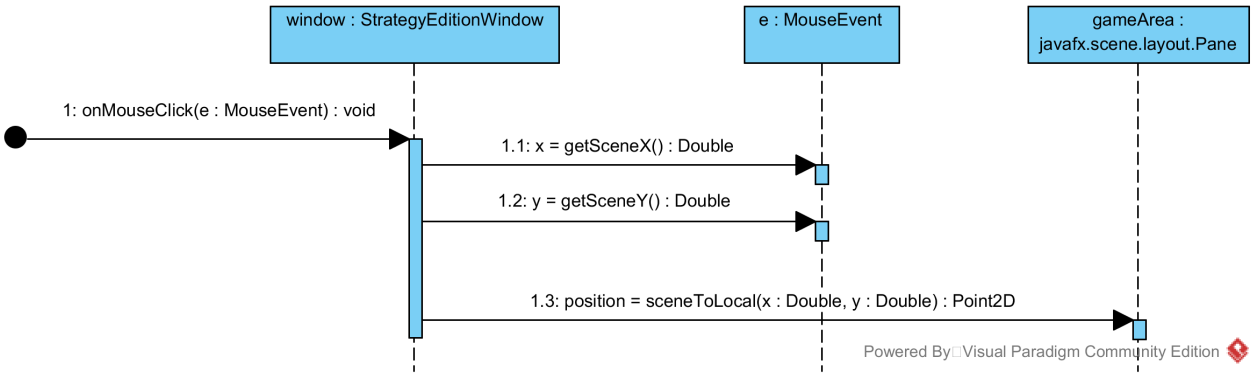

FIGURE 2.1 – Diagramme de séquence de conversion de coordonnées

Puisque les éléments sont ajoutés dans un Pane de JavaFX avec les dimensions réelles, la conversion s'en trouve fortement simplifiée. Par exemple, lorsque l'on ajoute le sprite qui représente le terrain, on l'ajoute avec ses vrais dimensions dans le Pane (ex : 100m par 60m). Pour pouvoir l'afficher avec des dimensions raisonnables à l'écran, il suffit de jouer avec les dimensions du Pane intérieur pour faire un zoom.

Ainsi, on procède en ajoutant un écouteur sur le Pane qui contient la scène et en écoutant les événements de clic de la souris. Par la suite, on obtient la position en X et la position en Y à partir de l'événement JavaFX généré. Ces coordonnées seront en pixels par rapport au sommet supérieur gauche du Pane. Finalement, on utilise la méthode `sceneToLocal` du Pane pour convertir ces coordonnées en unités réelles.

2.2 Ajout d'un élément à l'aide de la souris

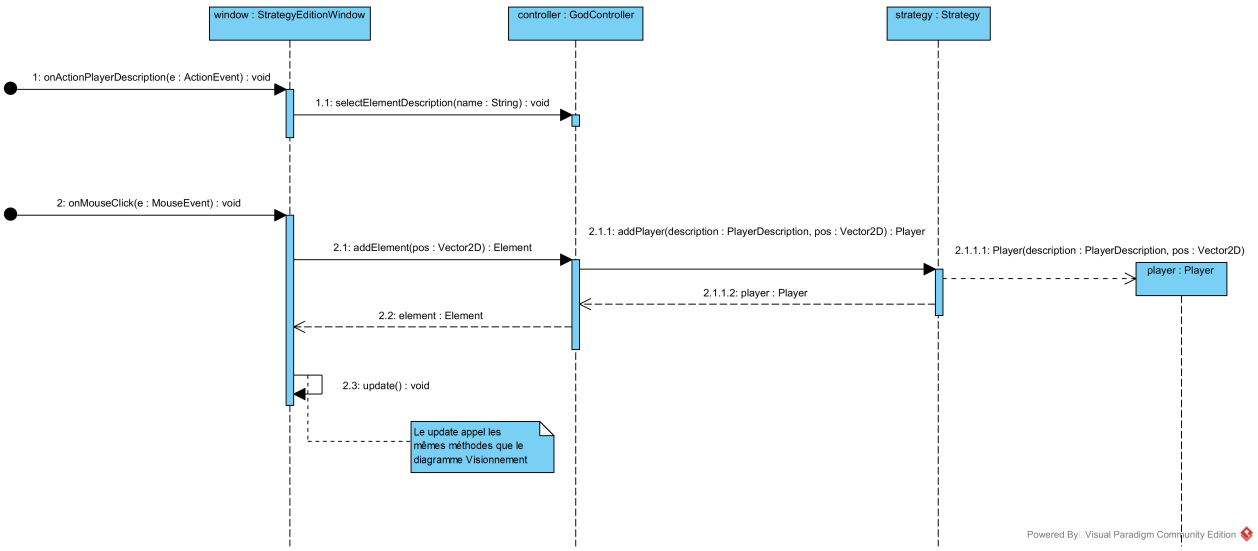

FIGURE 2.2 – Diagramme de séquence d'ajout d'un élément

Le premier appel se fait à partir d'un bouton dans le menu de gauche. On y sélectionne l'outil de création d'élément, et par le fait même, on indique au contrôleur qu'on veux créer des éléments du type qu'on lui passe en paramètre. Le second appel viens de la scène, on passe au contrôleur la position où on veux créer l'élément. Ensuite, il ne reste plus qu'à demander à la strategie de créer l'élément, et mettre à jour l'interface graphique.

2.3 Déterminer si le clic de la souris est sur un joueur

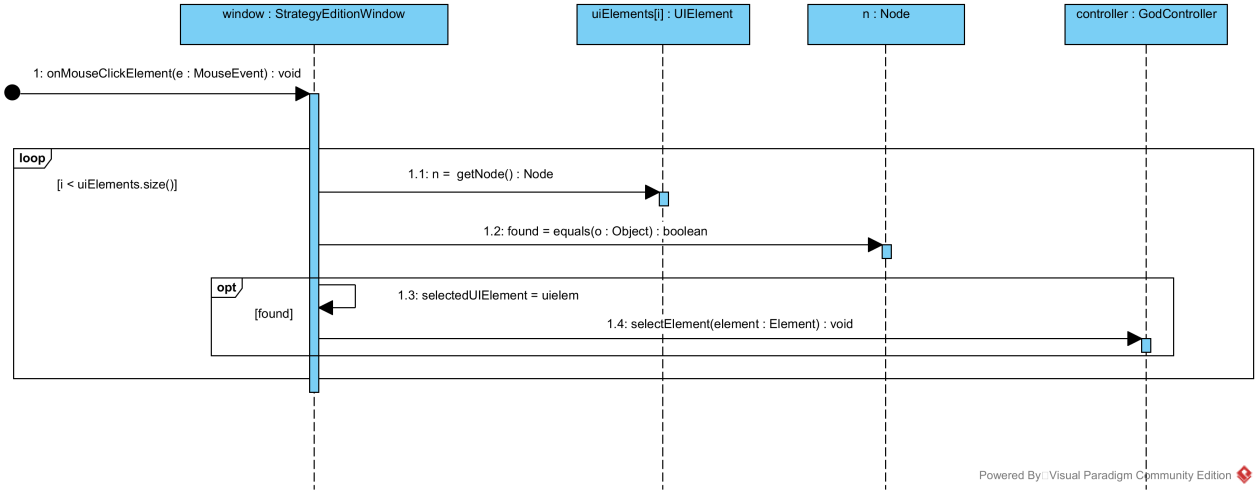

FIGURE 2.3 – Diagramme de séquence pour déterminer si le clic de la souris est sur un joueur

Avant toute chose, notez qu'avant d'ajouter un élément dans la scène, on place un écouteur sur son *Node JavaFX*. L'écouteur appel la méthode *onMouseClickElement*. Lorsque la méthode est appelée, il faut trouver quel élément correspond à la source de l'événement en itérant parmi les *UIElement* et en comparant le *Node* avec celui qui a généré l'événement.

2.4 Édition de la stratégie image par image

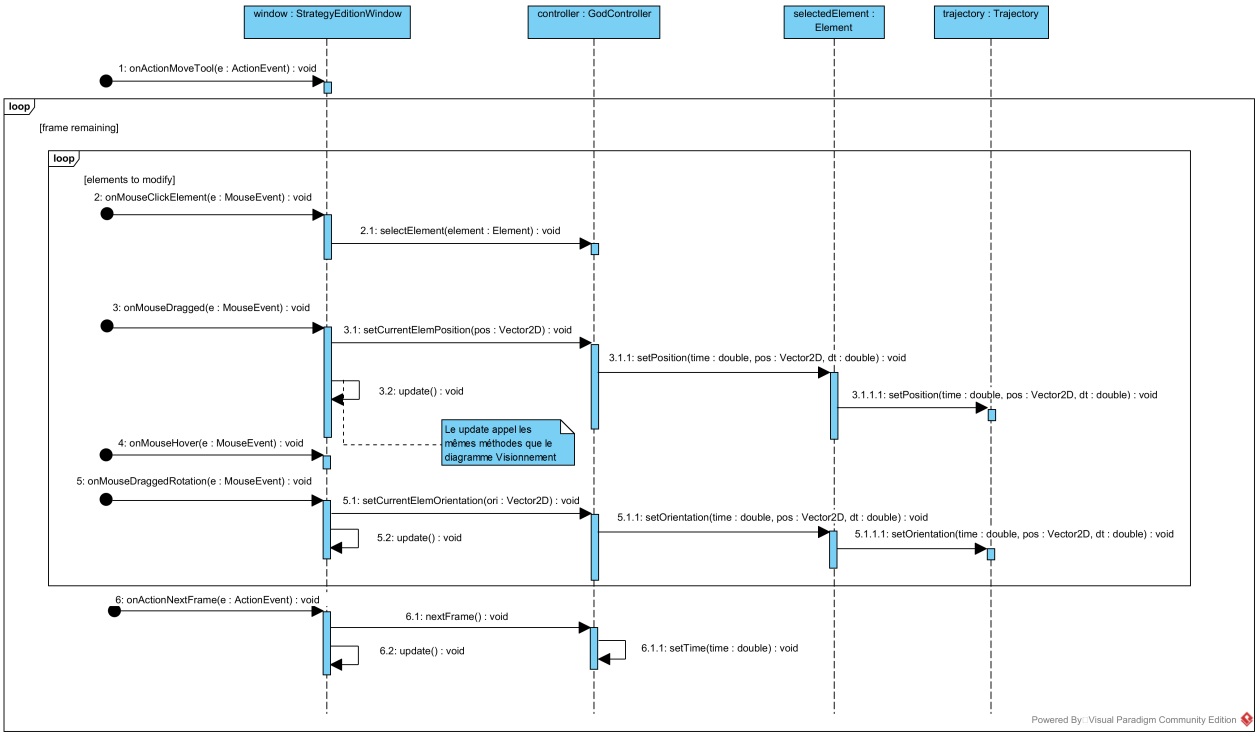

FIGURE 2.4 – Diagramme de séquence d'édition de stratégie image par image

La première étape pour l'édition image par image est de sélectionner l'outil de déplacement dans la barre d'outil. Pour cela, un simple écouteur est mis sur le bouton en question. Ceci permet à l'interface de comprendre que l'utilisateur veut pouvoir déplacer des éléments.

Ensuite, lorsque l'utilisateur clique sur un élément, un écouteur permet d'informer l'interface de l'action. L'interface informe le contrôleur de l'élément sélectionné. À la fin du déplacement (le *drag*), un écouteur nous informe de l'événement. L'interface informe donc le contrôleur de la nouvelle position du joueur. Le contrôleur met alors à jour l'élément actuel. Dans le diagramme ci-dessus, on illustre le cas le plus complexe où l'élément est un *MobileElement*. Dans ce cas, la position est mise à jour dans la trajectoire. Dans le cas d'un Obstacle, la position est simplement mise à jour dans une variable interne, sans gestion de trajectoire. Par la suite, l'interface se met à jour pour représenter le nouveau domaine.

Si l'utilisateur place sa souris au-dessus d'un élément, un écouteur avertit l'interface qui affiche une petite flèche pour indiquer la possibilité de rotation de l'élément. Si cette petite flèche est déplacée, une rotation est appliquée à l'élément de la même façon que pour la position.

L'utilisateur peut ainsi modifier autant d'éléments qu'il le souhaite. Une fois qu'il a fini d'éditer la trame actuelle, il peut passer à la prochaine trame en cliquant sur le bouton approprié. L'interface va alors notifier le contrôleur qui va mettre à jour sa variable interne de temps. Puis, l'interface se mettra à jour pour présenter

la prochaine trame.

2.5 Édition de la stratégie en temps réel

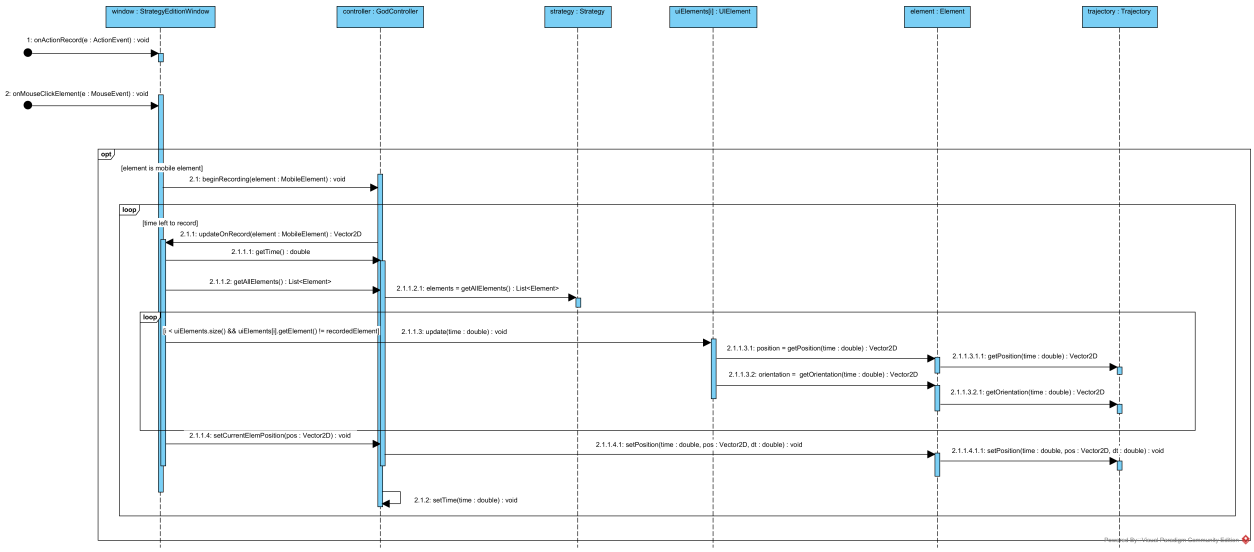

FIGURE 2.5 – Diagramme de séquence d'édition de stratégie en temps réel

L'édition en temps réel débute par le clic sur le bouton d'enregistrement en temps réel. Par la suite, dès que l'utilisateur cliquera sur un élément de la stratégie, l'édition en temps réel débutera. Ainsi, lors du clic sur un élément, l'interface avertit le contrôleur du début de l'édition en temps réel. Le contrôleur va alors démarrer un fil d'exécution pour mettre à jour les éléments et enregistrer la position de l'élément en édition. Ainsi, à chaque trame, le contrôleur va appeler la méthode *update* de l'interface. Cet appel se fait par l'interface *Updatable*. L'interface se met à jour en récupérant le temps actuel et tous les éléments, puis en modifiant ses *UIElement* pour refléter le domaine. À la fin, il met à jour la position du joueur en édition de la même façon que pour le mode image par image. Finalement, le contrôleur met à jour sa variable de temps et le tout se répète jusqu'à la fin de la stratégie.

2.6 Visualisation d'une stratégie

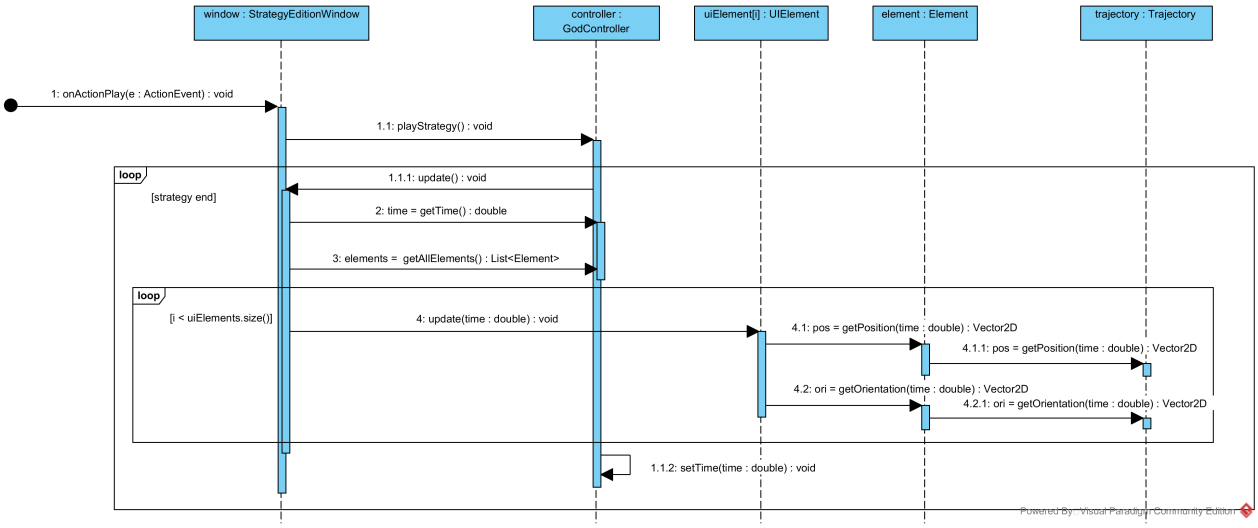

FIGURE 2.6 – Diagramme de séquence de visualisation d'une stratégie

Dans ce diagramme, on a représenté uniquement le cas du bouton Jouer puisque c'est le plus complexe. Lorsqu'on utilise un autre bouton pour changer d'image, on peut imaginer le même diagramme, mais sans la boucle «strategy end».

De la même manière, seul le cas où l'élément est un élément mobile a été représenté dans le diagramme, car c'est aussi le plus complexe. Dans le cas où l'élément est statique, il n'y a simplement pas d'appel à la classe Trajectory.

La visualisation de la stratégie fonctionne de la même manière que pour l'édition de la stratégie en temps réel, sauf que la position actuelle de la souris n'est pas renvoyée au contrôleur. Veuillez vous référer aux explications du diagramme précédent pour plus de détails.

Chapitre 3

Plan de travail

La figure 3.1 présente l'échéancier actuel du projet sous la forme d'un diagramme de Gantt. Les principales activités ainsi que les livrables y sont identifiés.

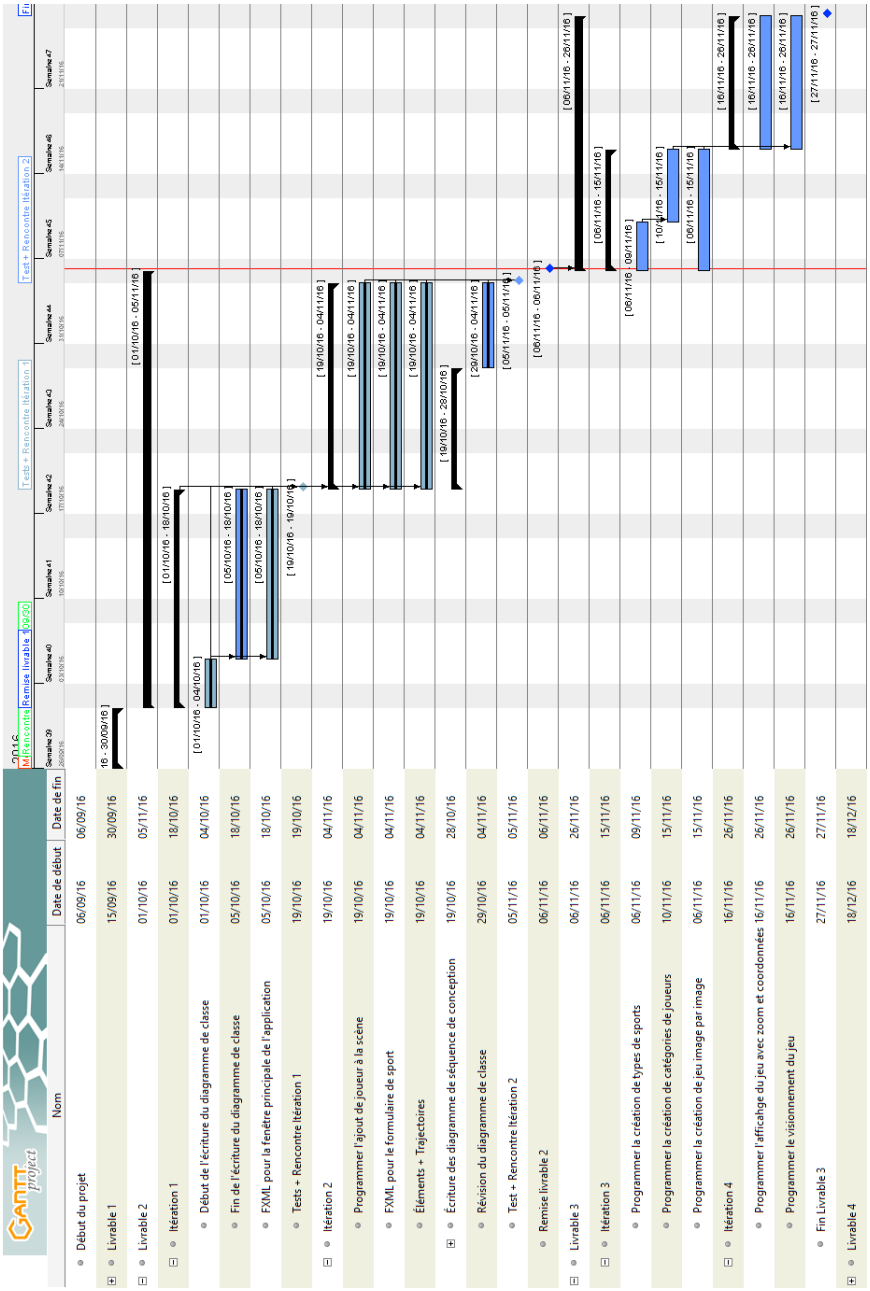

FIGURE 3.1 – Échéancier du projet

Annexe A

Vision

Cette section présente la vision du projet VisuaLigue, un outil de communication de stratégies sportives numérique. La lecture de cette partie est fortement recommandée pour un néophyte du projet, car elle présente le positionnement du produit, une description générale du logiciel et un sommaire des fonctionnalités. Ainsi, la lecture de cette section permet une compréhension globale du projet et une meilleure orientation dans son développement.

A.1 Positionnement

A.1.1 Énoncé du problème

Présentement, la majorité des entraineurs utilisent un tableau blanc avec un arrière-plan de patinoire pour dessiner et expliquer les stratégies. Or, les esquisses sont perdues après chaque entraînement et la visualisation des dessins n'est pas toujours simple.

De plus, le partage et le répertoriage des stratégies sont actuellement difficiles. Généralement, la façon de faire est de photographier le tableau blanc, puis d'envoyer l'image par courriel aux membres intéressés, ou encore de refaire le dessin sur papier ou sur une tablette graphique pour la répertorier.

L'autre problème majeur est la visualisation des stratégies. Souvent, lorsqu'un joueur est présent à une partie, il assiste à l'élaboration de la stratégie et comprend mieux le dessin. Généralement, le mouvement des joueurs est dessiné en temps réel et la gestuelle de l'entraineur est primordiale pour la compréhension. De plus, les trajectoires des joueurs sont souvent effacées pour libérer de l'espace. Or, pour un joueur qui était absent à l'entraînement ou pour un parent qui n'était pas sur la glace lors de l'élaboration de la stratégie, il est souvent difficile de comprendre la stratégie seulement avec le dessin.

A.1.2 Opportunité

Le produit est né d'une demande d'un entraineur de hockey junior qui se plaignait des méthodes traditionnelles d'esquisses de stratégies.

Non seulement un entraineur a manifesté un enthousiasme pour le produit, mais l'Association des entraineurs mineurs du Québec (AEMQ) est aussi intéressée par le produit. D'ailleurs, les spécifications actuelles ont été établies en collaboration avec le président de l'AEMQ.

Outre le milieu du hockey junior, le produit est facilement extensible au milieu professionnel. On remarque notamment l'usage des tableaux blancs dans le domaine professionnel du hockey. De plus, en rendant le logiciel suffisamment flexible, il serait possible d'étendre l'idée pour d'autres sports, notamment le soccer, le football, le volleyball, l'ultimate frisbee, le handball, le kinball, le curling et bien d'autres.

Ce projet pourrait donc avoir des répercussions sur un vaste éventail de sports, dont certaines ligues professionnelles qui engrangent des quantités impressionnantes d'argent. Plusieurs de ces sports sont présents

un peu partout dans le monde, tant au niveau amateur que professionnel. La démarcation du produit pourrait d'ailleurs se faire à de nombreux événements sportifs tels que des tournois, des championnats internationaux et même les Jeux olympiques.

A.1.3 Alternatives et compétition

Les alternatives présentement sur le marché représentent davantage des outils de dessin sur ordinateur. Notamment, le logiciel ConceptDraw PRO [1] offre des extensions pour le dessin de stratégies de hockey.

De nombreux logiciels sont disponibles sur les tablettes pour le dessin. Certains de ces logiciels permettent la projection simultanée pour une meilleure visualisation par les joueurs.

Or, ces solutions permettent seulement de réaliser des images statiques. Celles-ci sont plus faciles à répertorier considérant leur support numérique. Toutefois, le problème de visualisation est toujours présent. Il est difficile de visualiser la stratégie à partir d'une image fixe.

A.1.4 Résumé du produit

VisuaLigue est une application qui permet la création et la visualisation de stratégies sportives. Un entraineur peut donc facilement placer les joueurs, les obstacles et les objets sur un terrain virtuel. Ces éléments peuvent ensuite être modifiés facilement grâce à une interface utilisateur conviviale. Le logiciel permet aussi de visualiser la stratégie de manière dynamique. Ainsi, l'entraineur peut démarrer la visualisation et montrer à tout le monde le déplacement des joueurs en temps réel. Il peut aussi mettre la visualisation sur pause, avancer, reculer et regarder image par image. Finalement, l'application permet la sauvegarde des jeux pour permettre le partage et le répertoriage.

Le tableau A.1 présente un sommaire non exhaustif des fonctionnalités du logiciel ainsi que les avantages offerts pour les parties prenantes.

Fonctionnalité	**Avantage pour les parties prenantes**
Création de stratégies numériquement	Facile à partager, schéma plus clair, notation standardisée
Visualisation des stratégies (lecture, pause, avancer, reculer, image par image, etc.)	Meilleure compréhension des joueurs, surtout s'ils n'étaient pas présents lors de l'élaboration de la stratégie
Création de nouveaux types de sports	Plus grande flexibilité pour les entraineurs. Extension du projet vers des sports autres que le hockey
Création de nouveaux types d'obstacles	Flexibilité du logiciel pour différents types d'entraînements

TABLE A.1 – Fonctionnalités et avantages pour les parties prenantes

D'autres fonctionnalités plus techniques ont été énoncées durant les discussions. Les points suivants ont notamment été soulevés :

— Fonctionnalité d'annuler/rétablir
— Exporter les stratégies sous un format d'image (PNG, JPEG, etc.)
— Zoom
— Affichage des coordonnées de la souris lors du déplacement sur l'aire de jeu
— Option pour montrer/cacher le rôle des joueurs

Une liste exhaustive des fonctionnalités sera détaillée plus loin dans ce rapport.

1. http://www.conceptdraw.com/How-To-Guide/ice-hockey-diagram-defensive-strategy-neutral-zone-trap

Annexe B

Modèle des cas d'utilisation

La figure B.1 représente l'interaction des acteurs avec le système VisuaLigue. Le diagramme est relativement simple, car l'entraineur exécute les différentes actions avec le système, alors que les parents et les joueurs n'ont accès qu'à certaines fonctionnalités.

On note une extension entre les deux modes de saisie de trajectoires. En effet, le client a spécifié qu'il voulait pouvoir modifier la trajectoire des éléments mobiles en mode image par image, ainsi qu'en temps réel avec souris. Le mode image par image est le plus simple à implémenter, donc il a été spécifié en premier. Le mode en temps réel est en fait une extension du premier mode à cause de son interactivité. Ceci explique le lien entre les deux dans le diagramme.

Chaque cas d'utilisation est détaillé par une ficher complète dans les prochains paragraphes. Un diagramme de séquence système suit chaque cas d'utilisation afin de décrire visuellement le scénario principal.

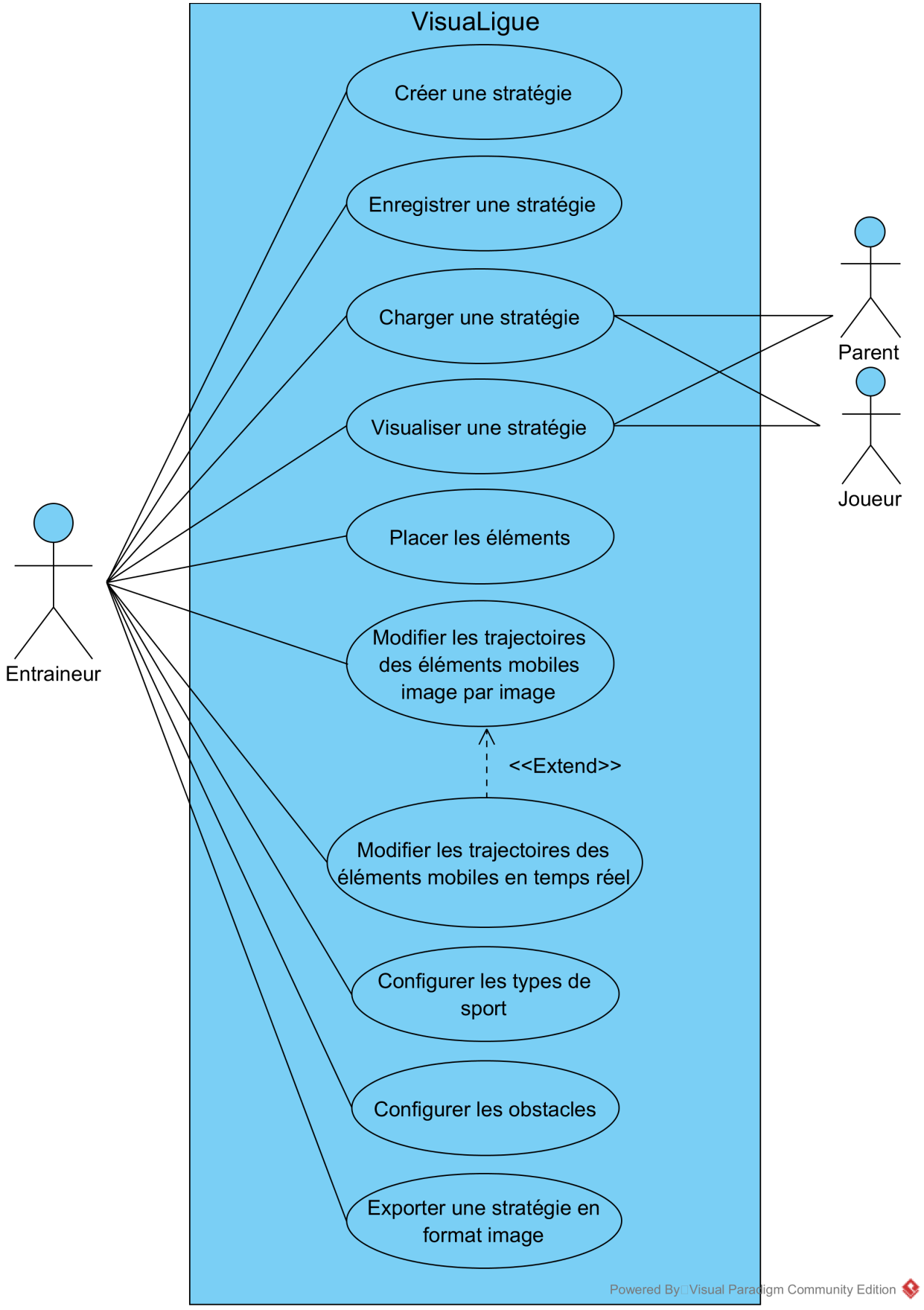

FIGURE B.1 – Diagramme des cas d'utilisation

B.1 Cas d'utilisation : Créer une stratégie

Projet : VisuaLigue

Niveau : But d'utilisateur

Acteurs primaires : Entraineur

Figurants et intérêts :
- Entraineur : Veut pouvoir créer des fichiers qui contiendront éventuellement des stratégies.

Préconditions :
Un type de sport doit déjà avoir été créé.

Garantie en cas de succès : L'entraineur aura un fichier qui pourra être utilisé pour élaborer une stratégie.

Principal scénario de succès :
1. Entraineur démarre le processus de création d'une stratégie.
2. Système demande les informations en lien avec la nouvelle stratégie.
3. Entraineur fournit à Système les informations nécessaires.
4. Système demande l'endroit où le fichier devra être enregistré ainsi que le nom de ce dernier.
5. Entraineur choisi l'endroit où le fichier devra être enregistré ainsi que son nom.
6. Système crée le fichier et Entraineur continu dans le cas d'utilisation Placer les éléments.

Scénarios alternatifs :
*a. Entraineur annule le processus de création d'une stratégie.
    1. Système retourne à la page où il se trouvait avant que le processus de création d'une stratégie ne soit démarré.
1a. Entraineur était au milieu de l'édition d'un fichier.
    1. Système demande à Entraineur s'il veut sauvegarder le fichier dont l'édition était en cours.
    2. Entraineur choisit s'il veut sauvegarder ou non le fichier dont l'édition était en cours.
5a. Entraineur entre un nom de fichier invalide.
    1. Système informe Entraineur que le nom entré est invalide.
    2. Entraineur entre un nom valide pour le fichier.
5b. Entraineur choisit un emplacement invalide pour le fichier.
    1. Système informe Entraineur que l'emplacement choisi est invalide.
    2. Entraineur choisit un emplacement valide pour le fichier.

Fréquence d'occurrence : Régulièrement

Questions ouvertes : Quelles seront les informations en lien avec la nouvelle stratégie que l'entraineur devra entrer ?

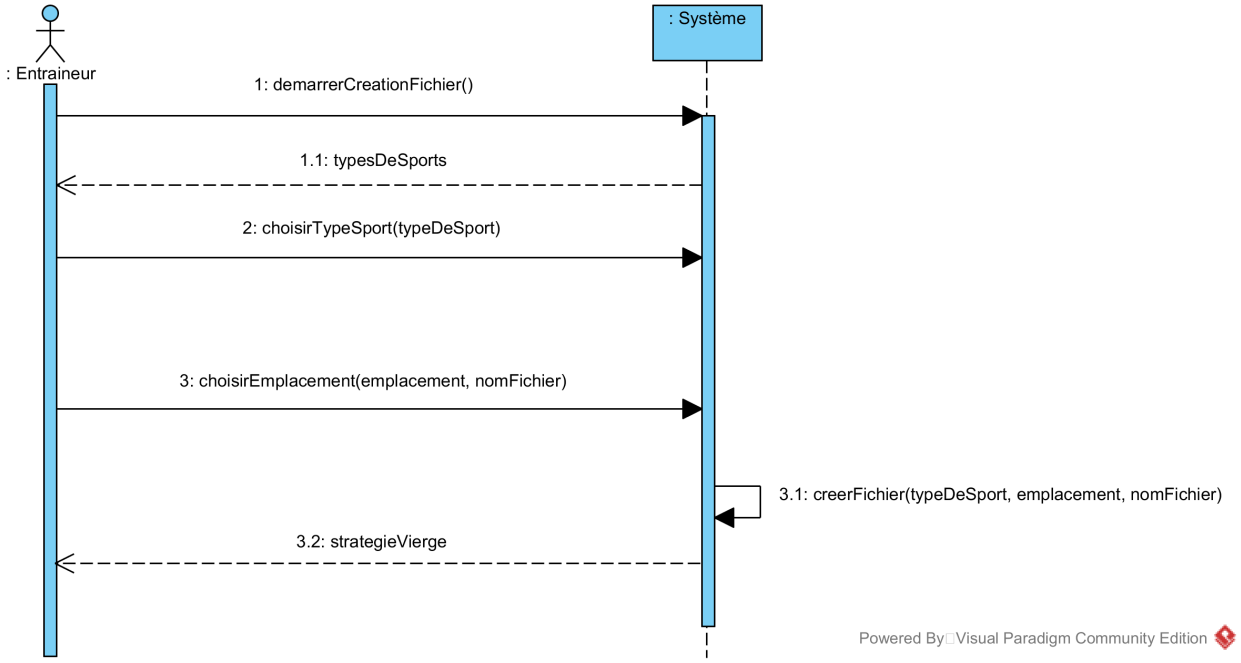

FIGURE B.2 – Diagramme du cas d'utilisation « Créer une stratégie »

B.2 Cas d'utilisation : Enregistrer une stratégie

Projet : VisuaLigue

Niveau : But d'utilisateur

Acteurs primaires : Entraineur

Figurants et intérêts :
- Entraineur : Veut pouvoir enregistrer une stratégie qu'il a élaborée.

Préconditions :
Un fichier de stratégie doit préalablement avoir été chargé.

Garantie en cas de succès : La stratégie est enregistrée et peut être chargée lors d'une prochaine utilisation du logiciel.

Principal scénario de succès :
1. Entraineur démarre le processus d'enregistrement de la stratégie.
2. Système enregistre la stratégie dans le fichier de stratégie.

Fréquence d'occurrence : Régulièrement

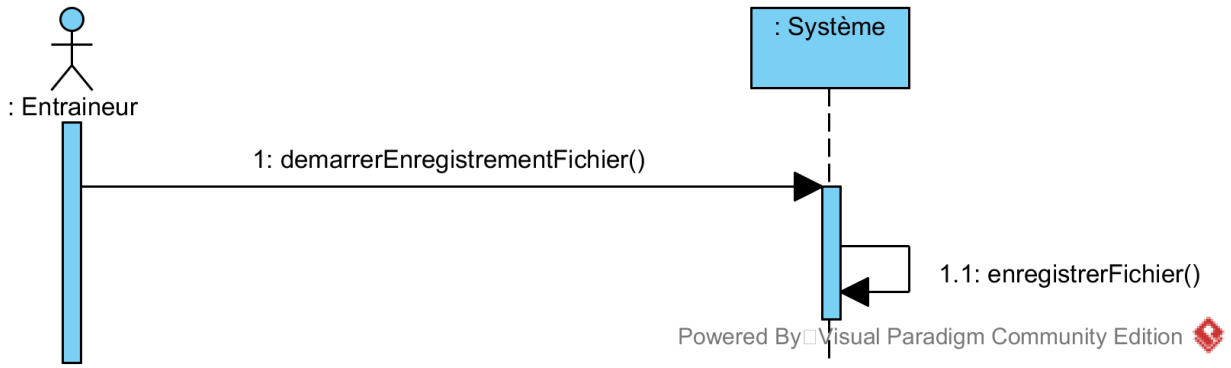

FIGURE B.3 – Diagramme du cas d'utilisation « Enregistrer une stratégie »

B.3 Cas d'utilisation : Charger une stratégie

Projet : VisuaLigue

Niveau : But d'utilisateur

Acteurs primaires : Entraineur et Joueur

Figurants et intérêts :
- Entraineur : Veut pouvoir charger une stratégie qu'il a élaborée.
- Joueur : Veut pouvoir charger une stratégie élaborée par l'entraineur.

Préconditions : Une stratégie doit préalablement avoir été créée.

Garantie en cas de succès : La stratégie est chargée et elle peut être modifiée ou visualisée.

Principal scénario de succès :
1. Entraineur ou Joueur démarre le processus de chargement de la stratégie.
2. Système demande le fichier de stratégie à charger.
3. Entraineur ou Joueur choisit le fichier de stratégie charger.
4. Système charge le fichier.

Scénarios alternatifs :
*a. Entraineur ou Joueur annule le processus de chargement d'une stratégie.
    1. Système retourne à la page où il se trouvait avant que le processus de chargement d'une stratégie ne soit démarré.
1a. Entraineur était au milieu de l'édition d'un fichier.
    1. Système demande à Entraineur s'il veut sauvegarder le fichier dont l'édition était en cours.
    2. Entraineur choisit s'il veut sauvegarder ou non le fichier dont l'édition était en cours.
3a. Entraineur ou Joueur choisit un fichier invalide.
    1. Système informe Entraineur ou Joueur que le fichier choisi est invalide.
    2. Entraineur ou Joueur choisit un fichier valide.

Fréquence d'occurrence : Régulièrement

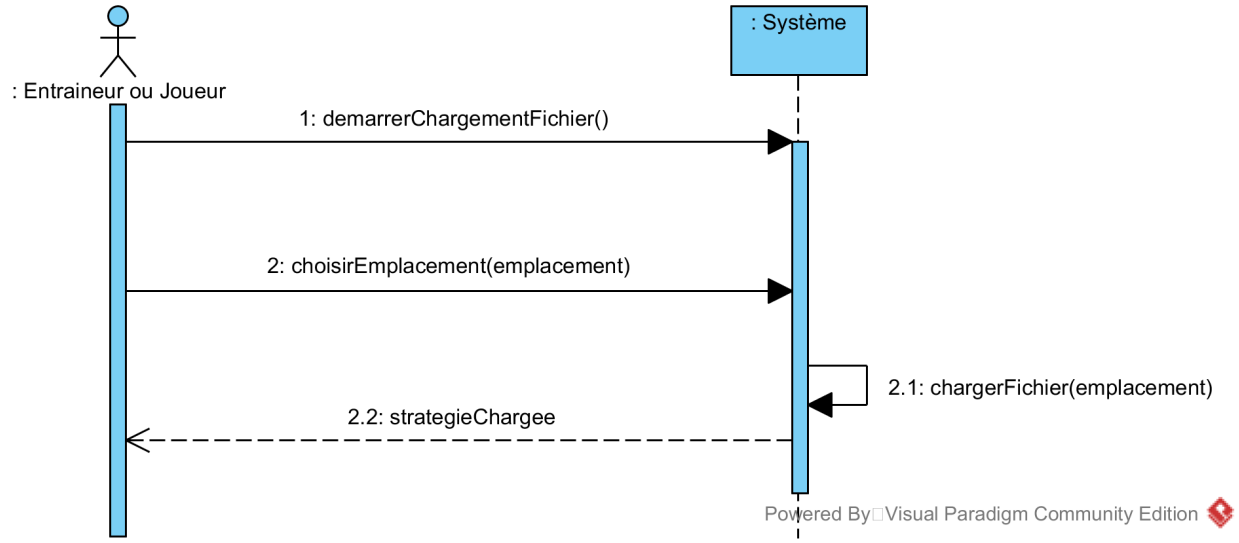

FIGURE B.4 – Diagramme du cas d'utilisation « Charger une stratégie »

B.4 Cas d'utilisation : Visualiser une stratégie

Projet : VisuaLigue

Niveau : But d'utilisateur

Acteurs primaires : Entraineur et Joueur

Figurants et intérêts :
- Entraineur : Veut pouvoir visualiser une stratégie qu'il a créée.
- Joueur : Veut pouvoir visualiser une stratégie à apprendre.

Garantie en cas de succès : La stratégie a été affichée à l'écran.

Principal scénario de succès :
1. Entraineur ou Joueur démarre le processus de visualisation de la stratégie.
2. Système calcule la position des éléments et les affiche.
   Système répète l'action 2 jusqu'à la fin de la stratégie.

Scénarios alternatifs :
2a. Entraineur ou Joueur déplace le curseur sur la ligne du temps.
   1. Système arrête l'exécution de la stratégie et affiche l'image sur laquelle le curseur est placé.
2b. Entraineur ou Joueur annule l'exécution de la stratégie.
   1. Système arrête l'exécution de la stratégie et affiche la dernière image sur laquelle le curseur est placé.
2c. Entraineur ou Joueur appuie sur le bouton de retour au début.
   1. Système place le curseur sur la première image et reprend l'exécution de la stratégie.
2d. Entraineur ou Joueur démarre le processus d'avance rapide.
   1. Système continue l'exécution de la stratégie plus rapidement.
2e. Entraineur ou Joueur démarre le processus de retour en arrière.
   1. Système exécute la stratégie à l'envers plus rapidement que la vitesse normale à partir de l'image à laquelle le curseur était positionné.
2f. Entraineur ou Joueur démarre le processus fin de la stratégie.
   1. Système arrête la stratégie et place le curseur sur la dernière image.

Fréquence d'occurrence : Régulièrement

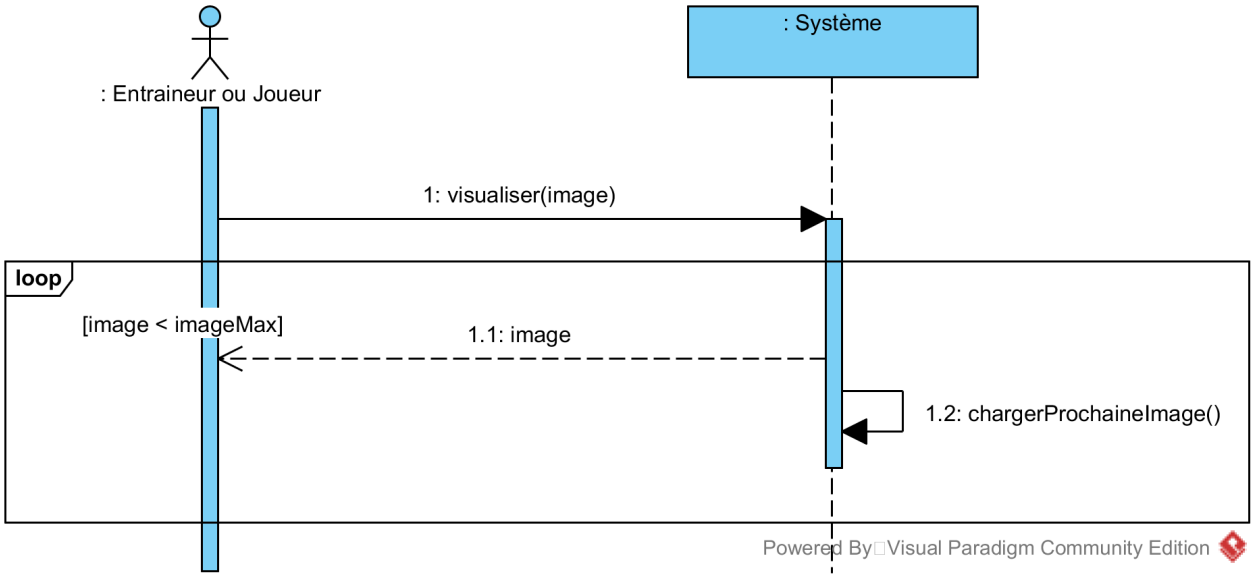

FIGURE B.5 – Diagramme du cas d'utilisation « Visualiser une stratégie »

B.5 Cas d'utilisation : Placer les éléments

Projet : VisuaLigue

Niveau : But d'utilisateur

Acteurs primaires : Entraineur

Figurants et intérêts :
- Entraineur : Veut pouvoir placer les éléments sur la scène et les modifier à sa guise.

Garantie en cas de succès : Les éléments sont placés selon ce que l'entraineur souhaite.

Principal scénario de succès :
1. Entraineur sélectionne un élément de la liste d'éléments disponibles.
2. Système crée un élément selon les spécifications définies selon le type de sport.
3. Entraineur clique dans la scène à l'endroit où l'élément doit être placé.
4. Système place l'élément dans la scène et affiche l'élément.
5. Système met à jour la disponibilité de l'élément dans la liste d'éléments disponibles (si nécessaire)

Scénarios alternatifs :
5a. Entraineur souhaite modifier la position de l'élément après l'avoir placé.
1. Entraineur clique sur l'élément à modifier.
2. Système indique visuellement que l'élément est sélectionné et affiche les paramètres associés.
3. Entraineur déplace avec la souris l'élément en question ou modifie la propriété "Position" des paramètres.
4. Système déplace l'élément selon les spécifications de l'Entraineur.

5b. Entraineur souhaite modifier la rotation de l'élément.
1. Entraineur clique sur l'élément à modifier.
2. Système indique visuellement que l'élément est sélectionné et affiche les paramètres associés.
3. Entraineur modifie la propriété "Rotation" des paramètres.
4. Système oriente l'élément selon les spécifications de l'Entraineur.

5c. Entraineur souhaite modifier la rotation de l'élément avec la souris.
1. Entraineur déplace sa souris sur l'élément à modifier.
2. Système affiche une flèche de rotation près de l'élément sélectionné.
3. Entraineur déplace la flèche de rotation jusqu'à l'angle souhaité.
4. Système oriente l'élément selon les spécifications de l'Entraineur.

5d. Entraineur souhaite modifier le rôle d'un l'élément de type "Joueur".
1. Entraineur clique sur l'élément à modifier.
2. Système indique visuellement que l'élément est sélectionné et affiche les paramètres associés.
3. Entraineur sélectionne le rôle du joueur.
4. Système modifie l'élément pour correspondre aux spécifications du rôle du joueur.

5e. Entraineur souhaite supprimer un élément.
1. Entraineur clique sur l'élément à supprimer.
2. Système indique visuellement que l'élément est sélectionné et affiche avec les paramètres associés.
3. Entraineur démarre le processus de suppression.
4. Système demande à Entraineur s'il veut vraiment supprimer l'élément.
5. Entraineur confirme la suppression de l'élément.
5a. Entraineur annule la suppression de l'élément.

1. Système revient où il était dans le processus de placement d'un élément.

6. Système supprime l'élément.

Fréquence d'occurrence : Régulièrement

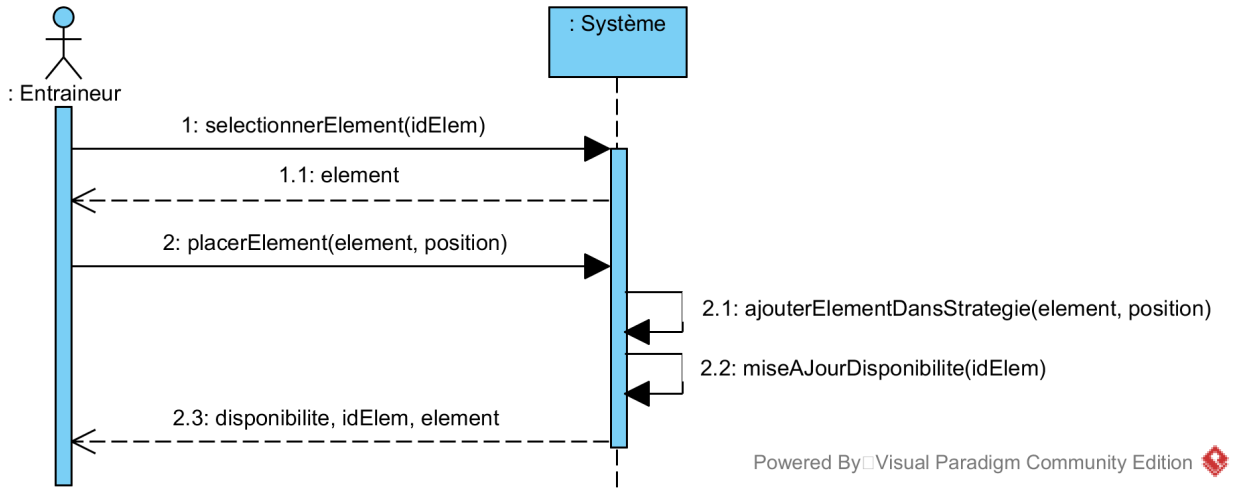

FIGURE B.6 – Diagramme du cas d'utilisation « Placer les éléments »

B.6 Cas d'utilisation : Modifier les trajectoires des éléments mobiles image par image

Projet : VisuaLigue

Niveau : But d'utilisateur

Acteurs primaires : Entraineur

Figurants et intérêts :
- Entraineur : Veut pouvoir modifier la trajectoire des éléments mobiles à sa guise.

Garantie en cas de succès : Les trajectoires des éléments mobiles correspondent à ce que l'entraineur souhaite.

Préconditions :
Un élément mobile a déjà été placé sur la scène.

Principal scénario de succès :
1. Entraineur déplace les éléments mobiles aux positions souhaitées. Voir cas d'utilisation Placer les éléments.
2. Entraineur passe à la prochaine image.
3. Système enregistre toutes les positions des éléments mobiles.
4. Système affiche la prochaine image et les éléments mobiles de l'image précédente en transparence.
Répéter les étapes 1 à 4 jusqu'à la fin de la stratégie.

Scénarios alternatifs :
2a. Entraineur souhaite revenir à l'image précédente pour la modifier.
   1. Entraineur passe à l'image précédente.
   2. Système affiche l'image précédente et les éléments mobiles de l'image qui la précède en transparence.
4a. La prochaine image n'existe pas.
   1. Système génère une nouvelle image à partir de la précédente.
   2. Système affiche la nouvelle image et les éléments mobiles de l'image précédente en transparence.

Fréquence d'occurrence : Régulièrement

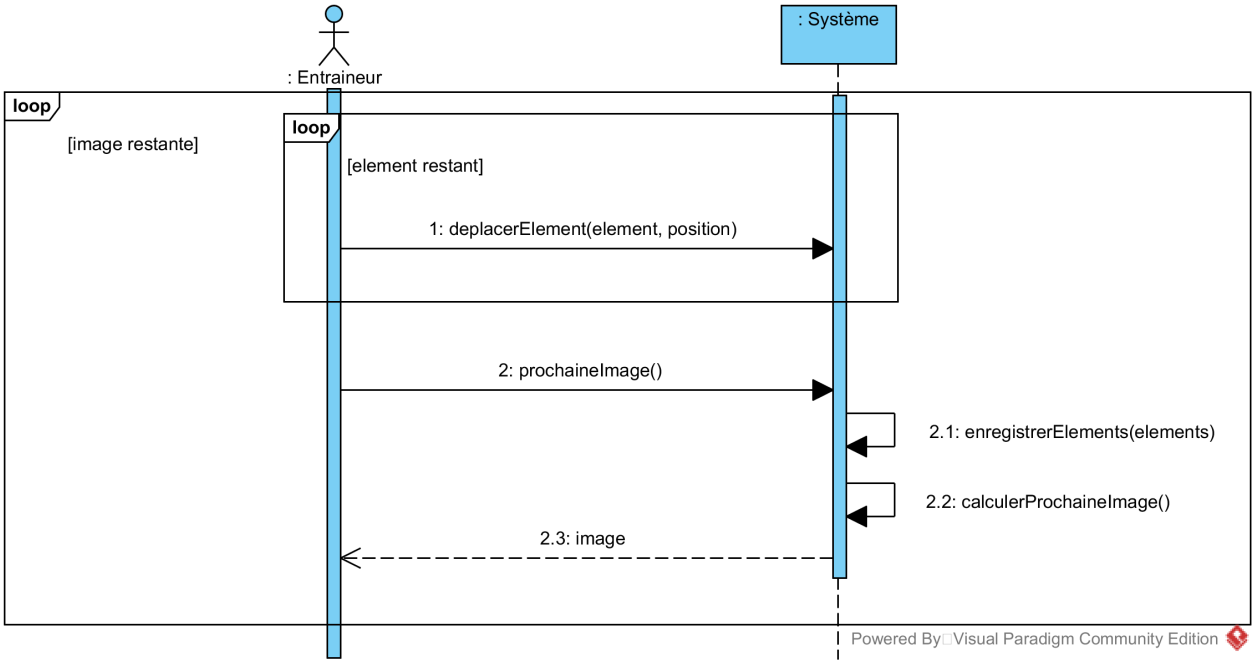

FIGURE B.7 – Diagramme du cas d'utilisation « Modifier les trajectoires des éléments mobiles image par image »

B.7 Cas d'utilisation : Modifier les trajectoires des éléments mobiles en temps réel

Projet : VisuaLigue

Niveau : But d'utilisateur

Acteurs primaires : Entraineur

Figurants et intérêts :
- Entraineur : Veut pouvoir modifier la trajectoire des éléments mobiles à sa guise et en temps réel.

Garantie en cas de succès : Les trajectoires des éléments mobiles correspondent à ce que l'entraineur souhaite.

Principal scénario de succès :
1. Entraineur démarre le processus d'enregistrement en temps réel.
2. Entraineur clique sur un élément mobile.
3. Système déplace les autres éléments mobiles en temps réel tout en déplaçant l'élément mobile sélectionné en suivant la position de la souris.
4. Entraineur visualise la stratégie ainsi obtenue selon le cas d'utilisation Visualiser une stratégie.

Scénarios alternatifs :
*a. Entraineur souhaite arrêter l'enregistrement.
   1. Entraineur signale à Système qu'il souhaite arrêter l'enregistrement.
   2. Système arrête l'enregistrement.
4a. Entraineur souhaite réenregistrer la trajectoire.
   1. Entraineur revient à l'image désirée.
   2. Entraineur répète les étapes 1 à 4 en prenant soin de sélectionner le même élément mobile.

Fréquence d'occurrence : Régulièrement

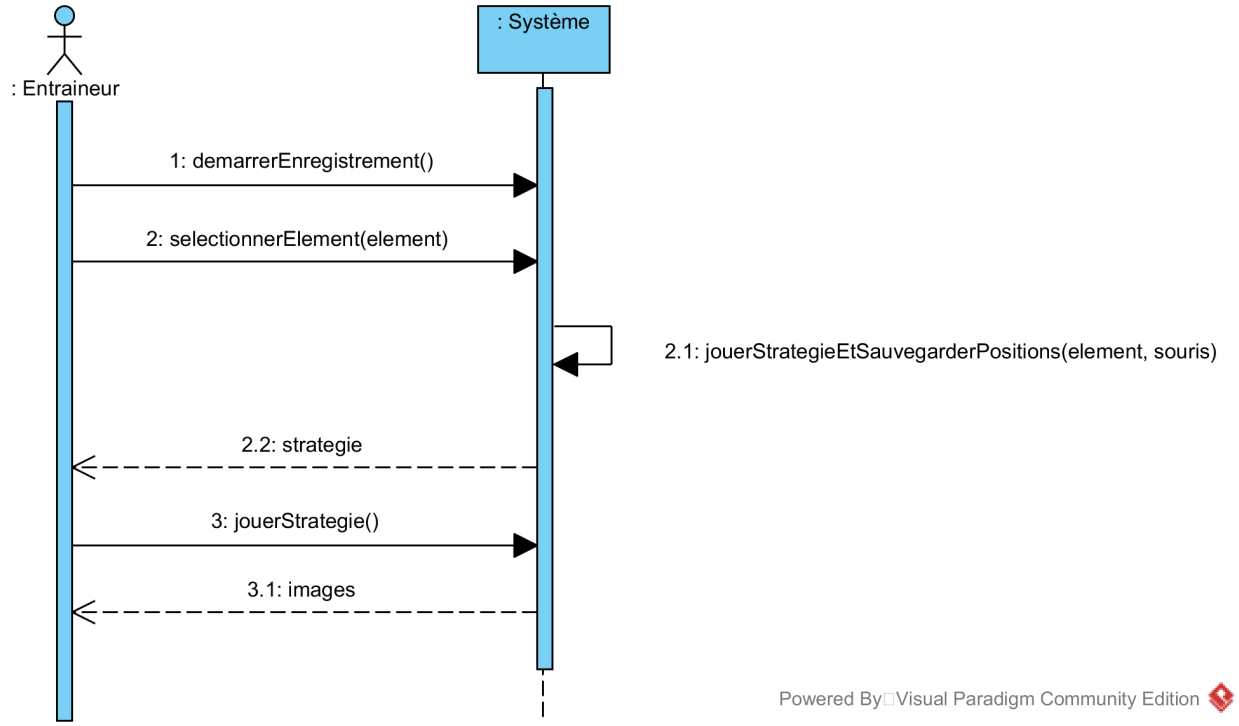

FIGURE B.8 – Diagramme du cas d'utilisation « Modifier les trajectoires des éléments mobiles en temps réel »

B.8 Cas d'utilisation : Configurer les types de sports

Projet : VisuaLigue

Niveau : But d'utilisateur

Acteurs primaires : Entraineur

Figurants et intérêts :
- Entraineur : Veut pouvoir créer et modifier les types de sports supportés par le logiciel.

Garantie en cas de succès : Une fois qu'un type de sport est créé, il est possible de créer une stratégie pour ce type de sport et de configurer les obstacles associés à ce type de sport.

Principal scénario de succès :
1. Entraineur démarre le processus de configuration d'un type de sport.
2. Système demande quel type de sport doit être configuré.
3. Entraineur indique le type à configurer.
4. Système offre de modifier les propriétés du type de sport, telles que le nom du sport, l'image pour représenter le terrain, les dimensions réelles du terrain, le nombre de joueurs et les catégories de joueurs associées au sport.
5. Entraineur modifie des propriétés du type de sport.
6. Entraineur démarre le processus d'enregistrement du type de sport.
7. Système valide et enregistre les modifications du type de sport.

Scénarios alternatifs :
*a. Entraineur annule le processus de configuration du type de sport.
    1. Système demande à Entraineur s'il est certain de vouloir annuler la configuration du type de sport.
    2. Entraineur confirme qu'il veut bien annuler la configuration du type de sport.
        2a. Entraineur indique qu'il ne veut plus annuler la configuration du type de sport.
            1. Système revient où il était dans le processus de configuration.
    3. Système retourne à la page où il se trouvait avant que le processus de configuration ne soit démarré.
3a. Entraineur indique qu'il veut créer un nouveau type de sport.
    1. Système demande à Entraineur de remplir tous les champs de propriétés du type de sport.
    2. Entraineur entre toutes les spécifications du type de sport.
    3. Entraineur démarre le processus d'enregistrement du type de sport.
    4. Système valide et enregistre le nouveau type de sport.
7a. Système valide les modifications et constate des données invalides.
    1. Système affiche un message d'erreur et demande de corriger les données invalides.
    2. Entraineur corrige les données.

Fréquence d'occurrence : Parfois

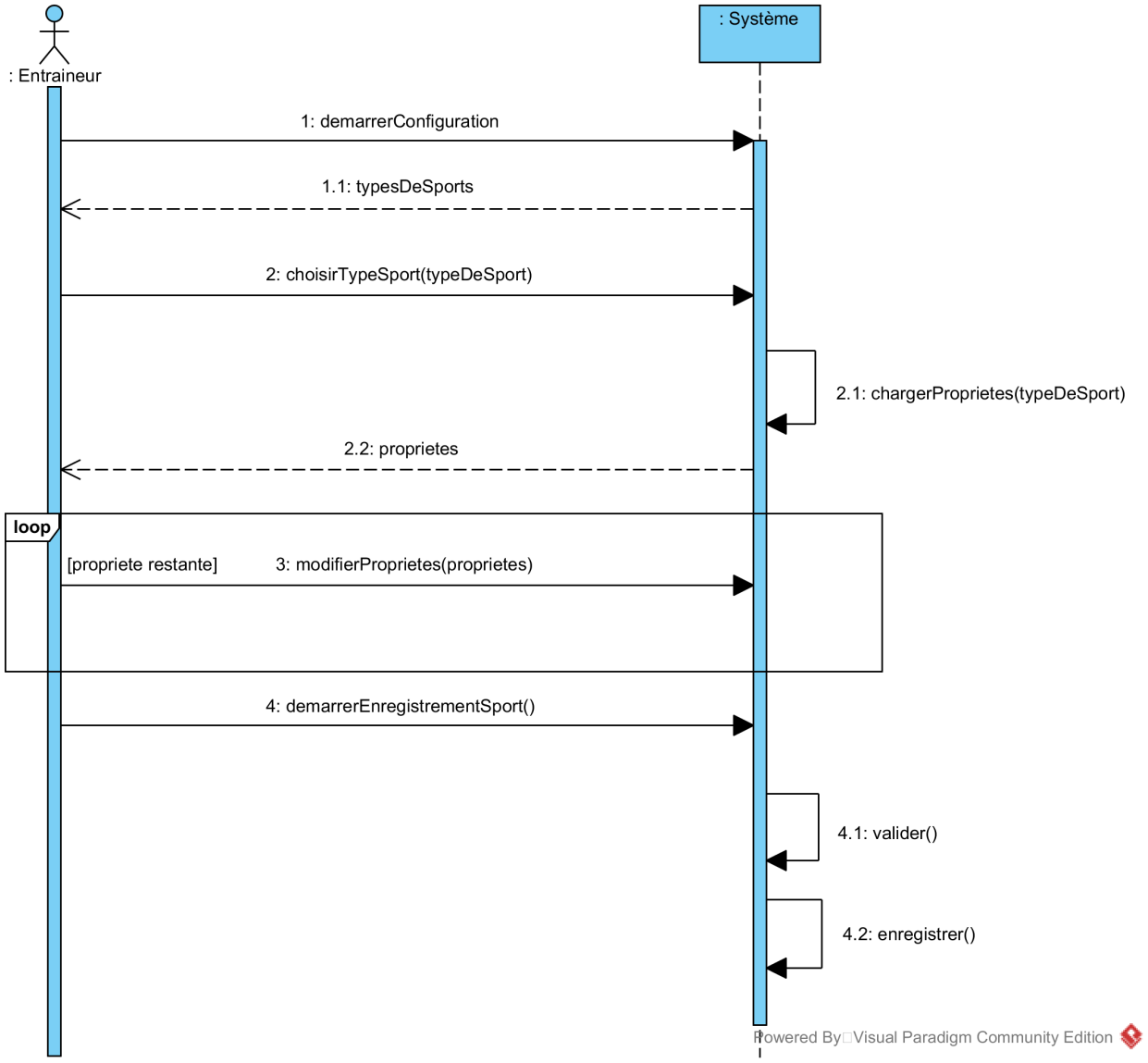

FIGURE B.9 – Diagramme du cas d'utilisation « Configurer les types de sports »

B.9 Cas d'utilisation : Configurer les obstacles

Projet : VisuaLigue

Niveau : But d'utilisateur

Acteurs primaires : Entraineur

Figurants et intérêts :
- Entraineur : Veut pouvoir créer et modifier les obstacles associés à un type de sport.

Préconditions : Un type de sport a été créé.

Garanties en cas de succès : Les obstacles peuvent être utilisés dans une stratégie de ce type de sport.

Principal scénario de succès :
1. Entraineur démarre le processus de configuration des obstacles.
2. Système demande de sélectionner un type de sport pour lequel les obstacles doivent être configurés.
3. Entraineur indique le type de sport désiré.
4. Système offre d'ajouter, de modifier ou de supprimer un obstacle.
5. Entraineur choisit de modifier les propriétés d'un obstacle.
6. Système offre de modifier les propriétés de l'obstacle, telles que le nom de l'obstacle, l'image pour représenter l'obstacle et les dimensions réelles de l'obstacle.
7. Entraineur modifie des propriétés de l'obstacle.
8. Entraineur démarre le processus d'enregistrement.
9. Système valide et enregistre les modifications de l'obstacle.

Scénarios alternatifs :
*a. Entraineur souhaite interrompre le processus de configuration des obstacles.
    1. Système retourne à l'interface précédente.
5a. Entraineur indique qu'il veut créer un nouvel obstacle.
    1. Système demande à Entraineur de remplir tous les champs de propriétés de l'obstacle.
    2. Entraineur entre toutes les propriétés de l'obstacle.
    3. Entraineur démarre le processus d'enregistrement.
    4. Système valide et enregistre le nouvel obstacle.
5b. Entraineur indique qu'il veut supprimer un obstacle.
    1. Système demande à Entraineur de confirmer son choix.
    2. Entraineur confirme qu'il veut bien supprimer l'obstacle.
        2a. Entraineur indique qu'il ne veut plus supprimer l'obstacle.
            1. Système revient à l'interface d'ajout/modification/suppression.
    3. Système supprime l'obstacle et revient à l'interface d'ajout/modification/suppression.
7a. Entraineur annule le processus de configuration de l'obstacle.
    1. Système demande à Entraineur s'il est certain de vouloir annuler la configuration de l'obstacle.
    2. Entraineur confirme qu'il veut bien annuler la configuration de l'obstacle.
        2a. Entraineur indique qu'il ne veut plus annuler la configuration de l'obstacle.
            1. Système revient où il était dans le processus de configuration.
    3. Système retourne à la page où il se trouvait avant que le processus de configuration ne soit démarré.
9a. Système valide les modifications et constate des données invalides.

1. Système affiche un message d'erreur et demande de corriger les données invalides.
2. Entraineur corrige les données.

Fréquence d'occurrence : Parfois

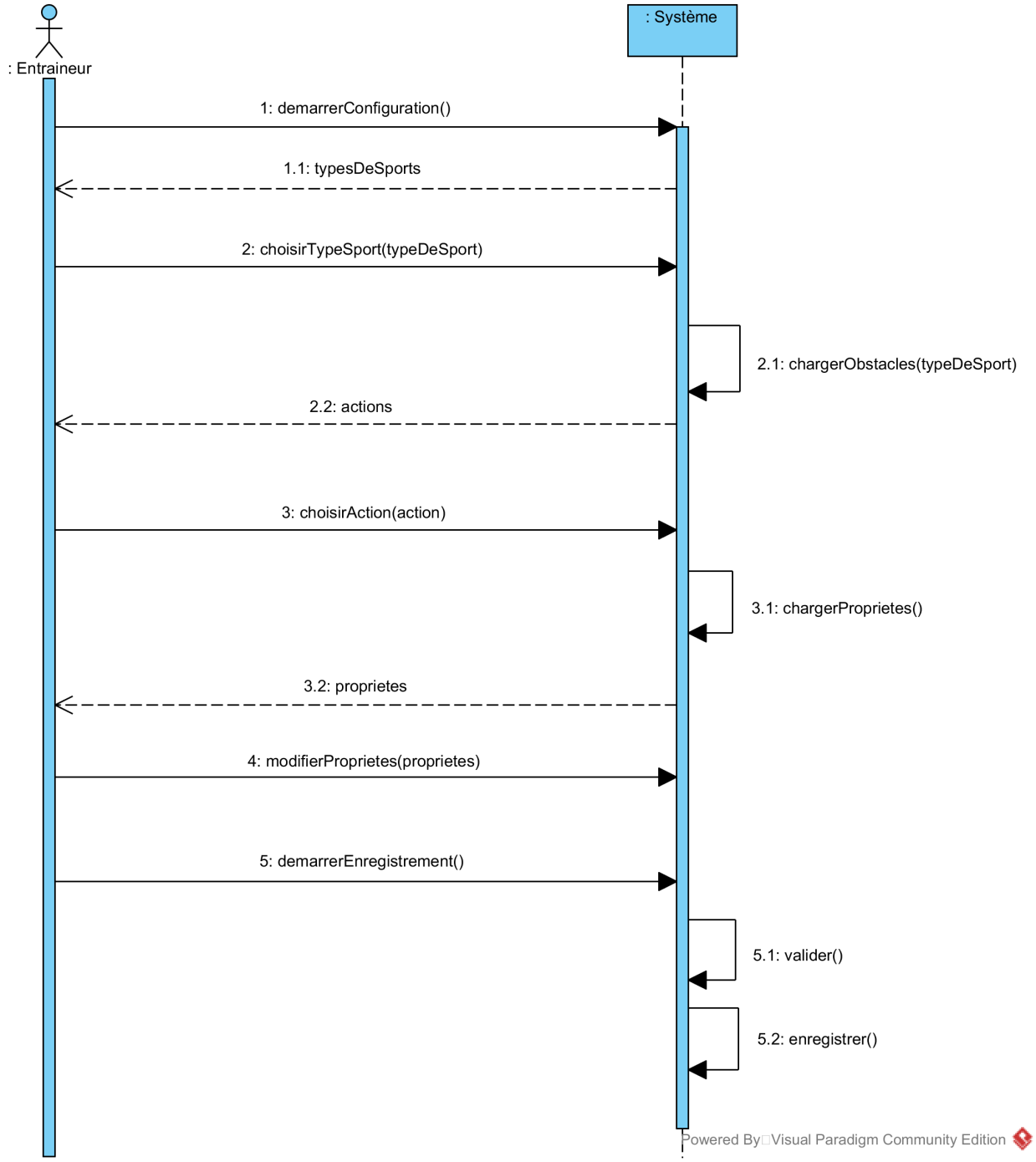

FIGURE B.10 – Diagramme du cas d'utilisation « Configurer les obstacles »

B.10 Cas d'utilisation : Exporter une stratégie en format image

Projet : VisuaLigue

Niveau : But d'utilisateur

Acteurs primaires : Entraineur

Figurants et intérêts :
- Entraineur : Veut pouvoir exporter les fichiers de l'application en image.

Garanties en cas de succès : La stratégie est exportée en tant qu'image.

Principal scénario de succès :
1. Entraineur démarre le processus d'exportation.
2. Système demande le format d'exportation du fichier, l'emplacement où il devra être exporté ainsi que le nom du fichier.
3. Entraineur sélectionne un format d'image, indique l'emplacement du fichier exporté ainsi que son nom.
4. Système enregistre la stratégie sous forme d'image avec des flèches représentants les trajectoires.

Scénarios alternatifs :
*a. Entraineur annule le processus d'exportation en image.
1. Système retourne à la page où il se trouvait avant que le processus d'exportation en image soit démarré.

Fréquence d'occurrence : Parfois

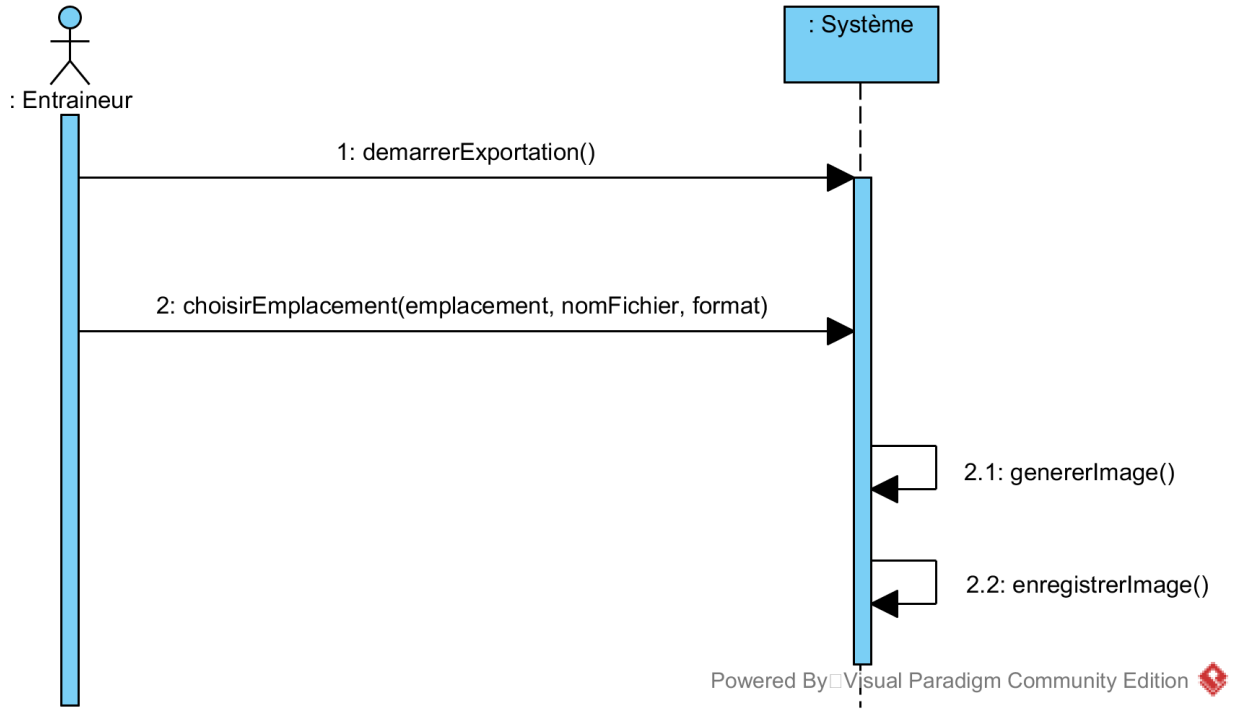

FIGURE B.11 – Diagramme du cas d'utilisation « Exporter une stratégie en format image »

Annexe C

Modèle du domaine

Cette section présente le modèle du domaine de VisuaLigue. La figure C.1 présente une vue globale de l'application.

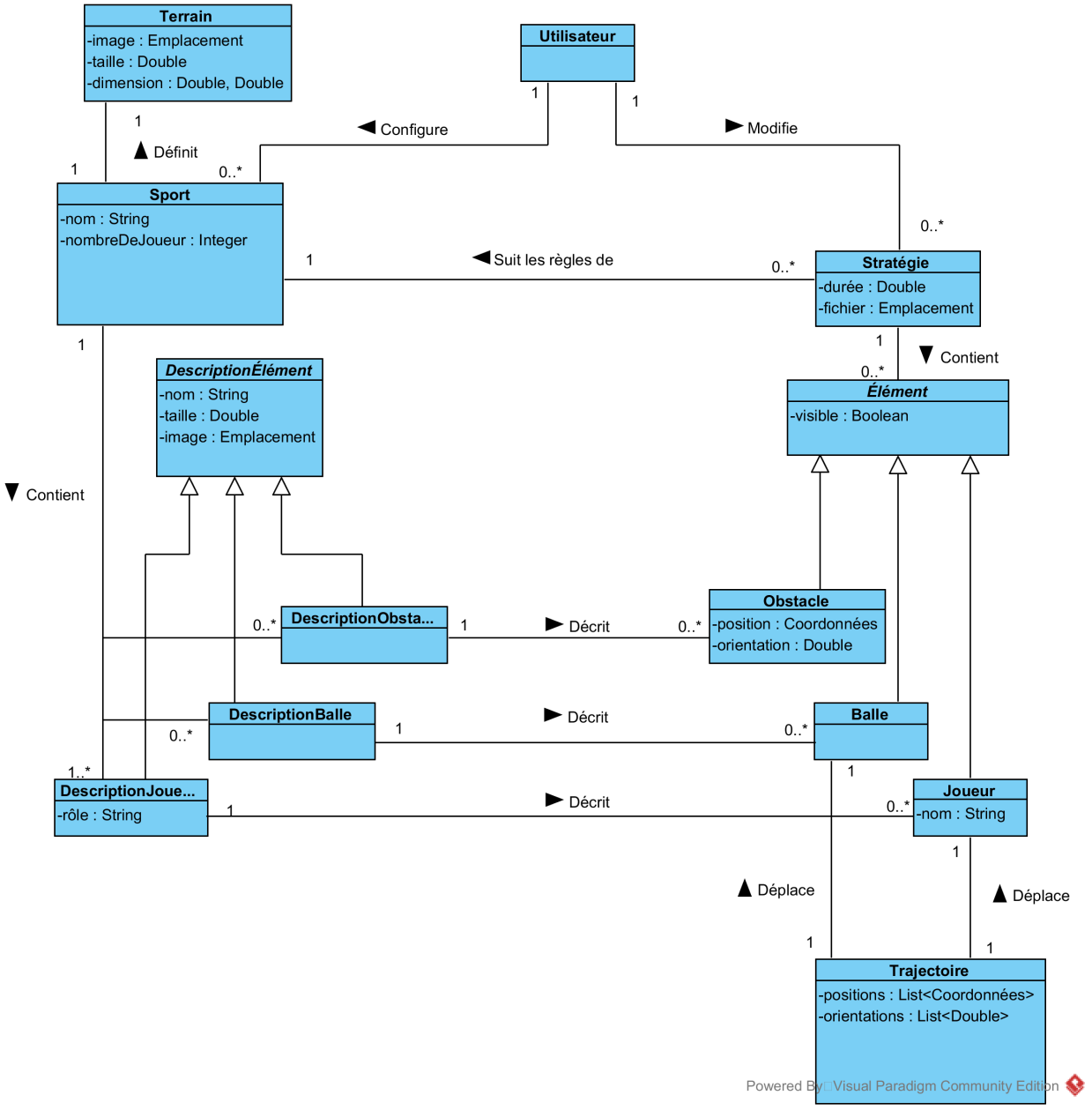

FIGURE C.1 – Diagramme du modèle du domaine

Le point d'entrée est l'utilisateur. Celui-ci peut être un Entraineur, un Joueur ou un Parent. Dans ce diagramme, on prend en considération que l'utilisateur est un Entraineur, car il peut exécuter toutes les actions.

L'utilisateur peut configurer des sports. Un sport est constitué d'un terrain, de plusieurs descriptions de joueurs, de balles et d'obstacles. Le terrain contient l'image, la taille et les dimensions. Il ne peut y avoir qu'un seul terrain par sport. Les descriptions des joueurs, des balles et des obstacles contiennent les propriétés qui seront utilisées lors de l'ajout de ceux-ci dans une stratégie. Par exemple, c'est ici que l'on définit le rôle des joueurs. Puisque plusieurs propriétés sont partagées entre ces classes, une généralisation

appelée *DescriptionÉlément* a été créée. Afin d'alléger le diagramme, la classe utilisateur n'a pas été reliée aux descriptions ni au terrain. Toutefois, on comprendra que lorsque l'utilisateur configure un sport, il modifie aussi les descriptions et le terrain. Il est aussi à noter que le terme « balle » est utilisé pour décrire tout élément mobile principal d'un sport. Par exemple, il peut représenter une rondelle de hockey, un Frisbee ou même une pierre de curling. Voir le glossaire à la page 37 pour plus de détails.

Une fois que l'utilisateur a configuré un sport, il peut créer et modifier une stratégie. Chaque stratégie est reliée à un sport qui représente les règles, le terrain ainsi que les descriptions des éléments. À l'intérieur d'une stratégie, on retrouve des obstacles, des balles et des joueurs. Ceux-ci partagent des propriétés en commun, d'où la généralisation *Élément*. Chaque élément est relié à sa description afin de refléter dans la stratégie les changements apportés dans le sport. Pour les éléments mobiles, une classe Trajectoire permet d'enregistrer leur déplacement. Encore une fois, pour alléger le diagramme, l'utilisateur n'a pas été relié aux éléments. On comprendra que lorsque l'on modifie une stratégie, on modifie également les éléments qui la composent.

Annexe D

Glossaire

Le glossaire présente toutes les définitions des termes simplificateurs utilisés dans le rapport. Il sert à réduire l'ambiguïté du texte.

Nom	Définition
Scène	Section principale de l'écran. C'est dans cette section que l'aire de jeu et les éléments sont affichés.
Stratégie	Ensemble des joueurs et de leurs déplacements dans la scène.
Élément	Objet qui peut être placé dans la scène.
Élément statique	Élément qui ne se déplace jamais dans la scène.
Élément mobile	Élément qui peut se déplacer dans la scène (ex. : un joueur ou une rondelle).
Joueur	Élément mobile représentant un membre d'une équipe sportive.
Type de sport	Ensemble de propriétés définissant un sport telles que les dimensions du terrain ou le nombre de joueurs.
Curseur	Élément visuel se déplaçant horizontalement sur la ligne du temps indiquant l'image actuellement affichée sur la scène.
Ligne du temps	Section de l'écran où l'on retrouve une représentation visuelle de la progression temporelle de la stratégie.
Image	Rendu de la scène à un moment donné dans le temps.
Barre d'outils	Section de l'écran où sont situés les boutons permettant de placer ou déplacer les éléments.
Balle	Élément mobile d'un sport qui est l'objet principal du sport. Par exemple, un ballon, une rondelle, une pierre de curling, etc.