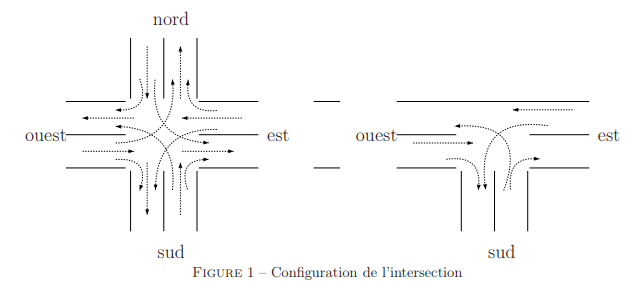
Synchronisation de feux de circulation



**Auteurs**

|  |
| --- |
| Élida Melo - 111 153 228  Ermine Wankpo - 111 067 301  Isabelle Eysseric - 111 189 609  Slim Ben Yahia - 111 150 230 |

**Table des matières**

|  |
| --- |
| [**1. Principaux événements, actions, interactions**](#_2qv0117cqxy6) **3**  [Intersection T](#_b7122gjei14q) 3  [Intersection +](#_wzgekpnbeey9) 3  [Les deux intersections](#_31t31r2e7hw3) 3  [**2. Entités actives**](#_24n938n4jc7b) **4**  [**3. Entités passives**](#_wud4y533b9r) **4**  [**4. Environnement interactif d’affichage**](#_v7moaekzrbod) **6**  [**5. Structure des classes**](#_m921wgu9cewo) **7**  [**6. Annexe**](#_i8910ksoa8q7) **9** |

|  |
| --- |
| 1. Principaux événements, actions, interactions |

Les principaux ensembles identifiés sont :

#### Intersection T

{ est.tourneGauche, sud.tourneGauche, est.continue, ouest.continue, sud.tourneDroite, ouest.tourneDroite }

{ traverse }

{ est.passeauvert, sud.passeauvert, ouest.passeauvert, est.passeaurouge, sud.passeaurouge, ouest.passeaurouge }

#### Intersection +

{ est2.tourneGauche, sud2.tourneGauche, ouest2.tourneGauche, nord2.tourneGauche, est2.continue, sud2.continue, ouest2.continue, nord2.continue, sud2.tourneDroite, sud2.tourneDroite, ouest2.tourneDroite, nord2.tourneDroite }

{ traverse2 }

{ est2.passeauvert, sud2.passeauvert, ouest2.passeauvert, nord2.passeauvert, est2.passeaurouge, sud2.passeaurouge, ouest2.passeaurouge, nord2.passeaurouge }

#### Les deux intersections

{ est.tourneGauche, sud.tourneGauche, est2.tourneGauche, sud2.tourneGauche, ouest2.tourneGauche, nord2.tourneGauche, est.continue, ouest.continue, est2.continue, sud2.continue, ouest2.continue, nord2.continue, sud.tourneDroite, ouest.tourneDroite, sud2.tourneDroite, sud2.tourneDroite, ouest2.tourneDroite, nord2.tourneDroite }

{ traverse,traverse2 }

{ est\_sync.passeauvert, sud\_sync.passeauvert, ouest\_sync.passeauvert, nord\_sync.passeauvert, est\_sync.passeaurouge, sud\_sync.passeaurouge, ouest\_sync.passeaurouge, nord\_sync.passeaurouge }

|  |
| --- |
| 2. Entités actives |

Les processus qui sont des entités actives et implantés comme threads sont :

* Voitures
* Piétons
* Lumières

Pour ce faire nous utilisons les classes :

CarRunnable, LightRunnable, PedestrianRunnable, LightSyncRunnable, (Sync pour la version synchronisée).

|  |
| --- |
| 3. Entités passives |

Six entités passives partagées sont implantées comme moniteurs :

* TrafficController : Classe qui permet le contrôle du trafic des voitures, piétons et lumières en fonction de l’intersection choisie (T, Croix ou Syc).
* LightController : Classe qui permet les actions de changement de couleur des lumières.
* LightSyncController : Classe qui permet de synchroniser les lumières et les voitures des intersections en T et en Croix en utilisant des barrières. Pour la synchronisation des voitures nous avons utilisé l’approche producteur-consommateur où les producteurs représentent les actions quittant d’une intersection et menant à l’autre. Les consommateurs quant à eux, sont les des directions dans les actions dépendant de l’autre intersection.
* AllLightController : Classe qui permet le contrôle de toutes les lumières dans toutes les directions, incluses les méthodes de changement de couleur, vérification de lumières de la direction opposée, vérification s’il y a de lumières vertes pour empêcher le piéton de traverser, vérification des lumières dans les directions voisines pour ne pas permettre deux lumières adjacentes d’être vertes en même temps.
* AllLightSyncController : c’est l’équivalent de la classe AllLightController, mais appliqué au cas des deux intersections synchronisées.
* PedestrianController : Classe permettant le contrôle du passage des piétons lors de l'exécution du thread PedestrianRunnable.

|  |
| --- |
| 4. Environnement interactif d’affichage |

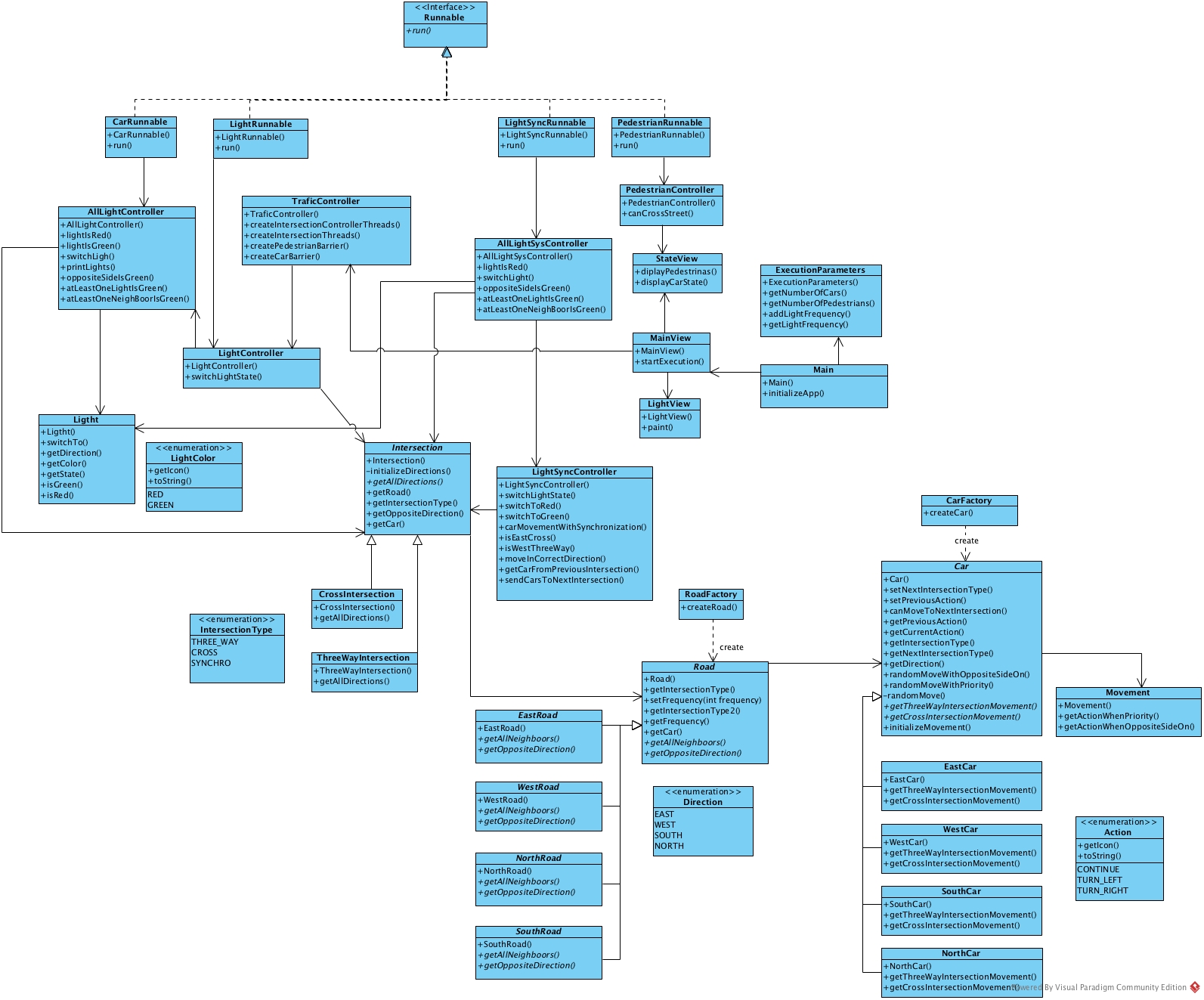
Une interface graphique sera utilisée pour l’affichage des lumières et des actions des voitures et des piétons, ainsi que pour l'interaction avec l'utilisateur. Les paramètres d'exécution tels que le type d’intersection choisie, le nombre de voitures et le nombre des piétons doivent être rentrés par l’utilisateur, bien que les fréquences des lumières pour chaque direction.

L’utilisateur a aussi l’option de mettre l'exécution en pause, ensuite continuer ou recommencer une nouvelle exécution ou quitter l'application. Pour ce faire les boutons PAUSE, RESUME, RESTART et QUIT sont disponibles dans la barre supérieure de l’interface.



Fig 1 - Interface graphique

|  |
| --- |
| 5. Structure des classes |



Dans le diagramme de classe ci-dessus nous retrouvons les principales classes et ses associations :

* Intersection : classe abstraite utilisée pour instancier des intersections du type T et Croix.
* Light : classe utilisée pour représenter une lumière et ses actions de changement de couleur.
* Road : classe abstraite utilisée pour instancier des voix selon ses directions (est, ouest, sud, nord).
* Car : classe abstraite utilisée pour instancier une voiture selon sa direction (est, ouest, sud, nord).
* Les classes utilisées pour l’affichage et initialisation des threads : MainView, StateView et LightView.
* Les contrôleurs : TrafficController, LightController, AllLightController LightSyncController, AllLightSyncController et PedestrianController tel qu’indiqués dans la question 3.
* Les classes qui implémentent les threads : CarRunnable, LightRunnable, PedestrianRunnable, LightSyncRunnable, tel qu’indiquées dans la question 2.
* Autres : des classes auxiliaires tel que factories et énumérations.

|  |
| --- |
| 6. Annexe |

Partie 1 -CODE FSP

