Simulateur de carrefour à sens giratoire : spécifications

Nom du projet	**SimulateurRP**
Livrables	

Spécificités

	Nom et prénom	Affiliation	Contact
Auteur principal	Vincent P.	Etudiant	Pierre.vincent1@etu.univ-amu.fr
Chef du projet	Deutsch R.	Etudiant	Remi.deutsch@etu.univ-amu.fr
Approbateur	Loignon L.	Etudiant	Lucas.loignon@etu.univ-amu.fr
Approbateur	Cauet C.	Etudiant	Christopher.cauet@etu.unv-amu.fr

Livré le	Approuvé le	Validé le
11.10.2017	30.09.2017	30.09.2017

Entité	Nom et prénom	Mode de distribution
à :	Amine Hamri	dépôt github
copie à :		

Nom du fichier	Etat	Nombre de pages
SimulateurRP.zip		

Evolution (objet)	Date de l'évolution	Numéro de version
Ajout choix 3 à 6 voies	29.09.2017	1.2

Table des matières

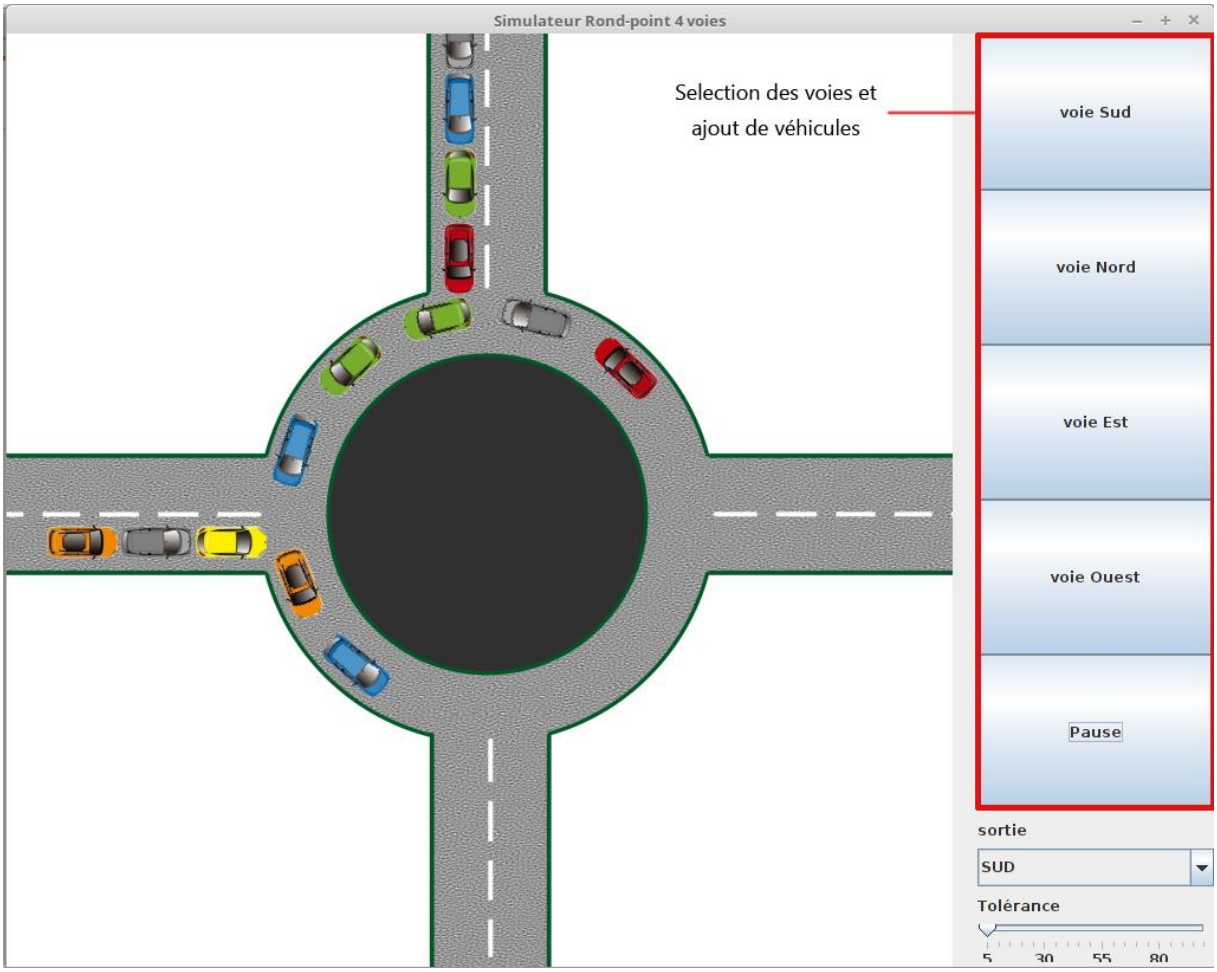
Le logiciel livré est un simulateur de carrefour à sens giratoire, il permet de simuler un rondpoint avec un trafic plus ou moins dense en fonction de l'utilisateur. Une interface graphique de test est livrée avec le produit et permet de visualiser, rapidement et efficacement le fonctionnement du simulateur.

Dans la suite de ces spécifications nous expliquerons le fonctionnement et les diverses fonctionnalités offertes par notre produit, puis nous détaillerons le schéma de production avec des scénarios de cas d'utilisation, ainsi que le patron de conception UML et un diagramme de gant.

I. Documentation

1. Ajout de véhicules dans le carrefour

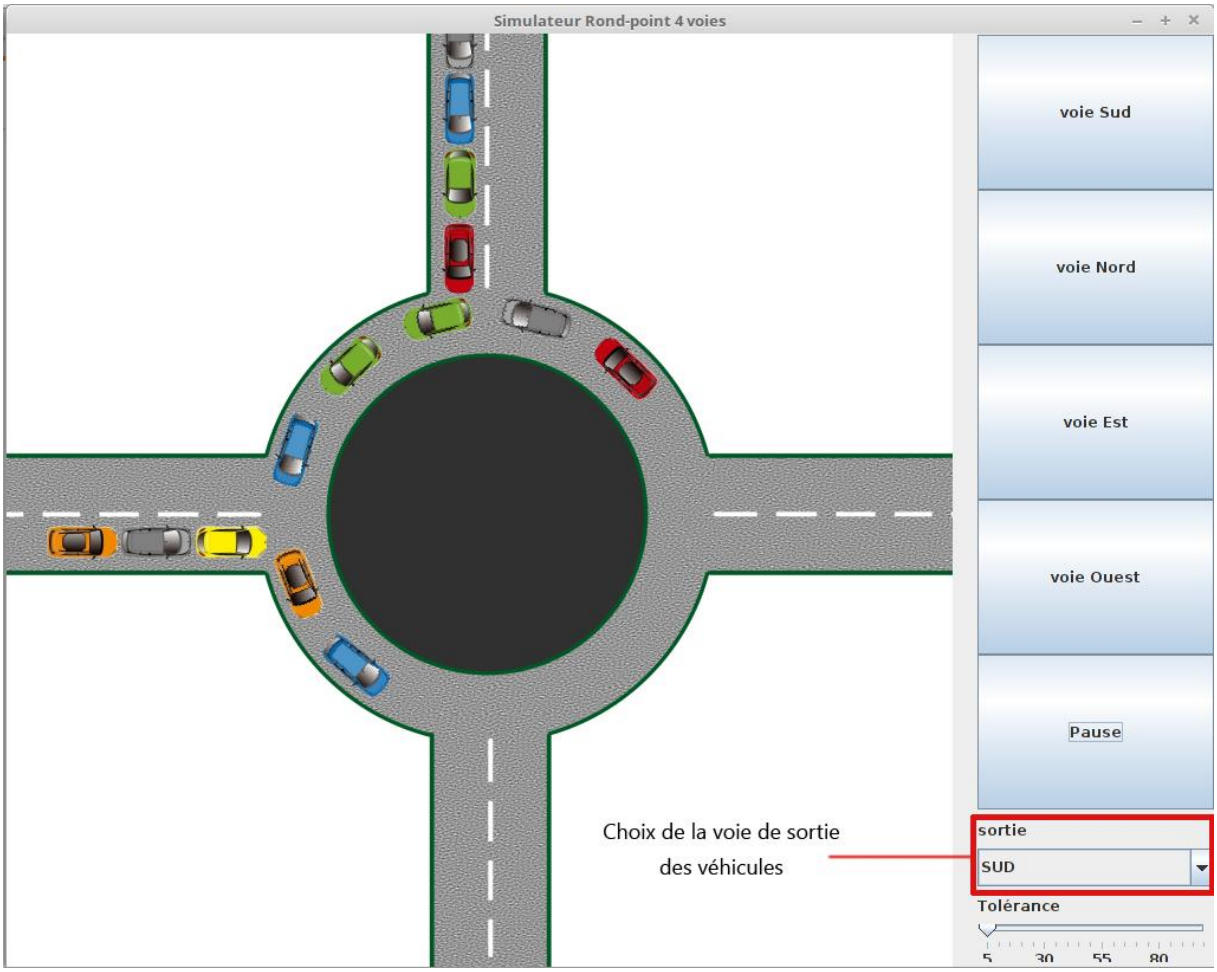
L'ajout de véhicules dans le carrefour se fait sur l'une des voies grâce à des boutons situés en haut à droite de la fenêtre de l'application, le bouton choisi permet de définir la voie d'entrée du véhicule. Suite au clique sur l'un des boutons d'ajout, un véhicule apparaitra



dans la voie sélectionnée. Les boutons d'ajout de véhicules sont illustrés dans la figure suivante :

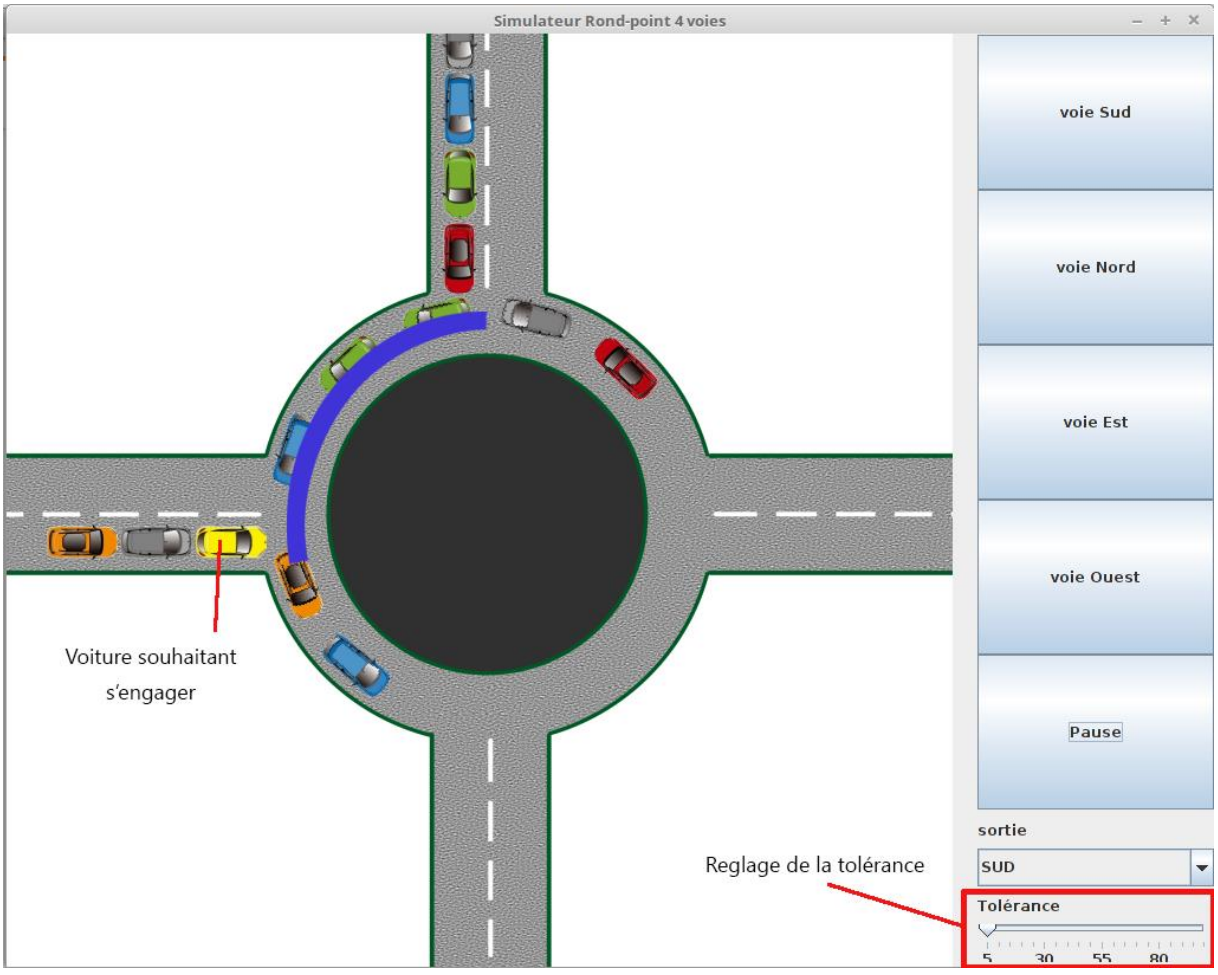
2. Sélection de la voie de sortie

L'utilisateur peut choisir la voie de sortie pour les véhicules qu'il insère dans le carrefour, cette sélection se fait grâce à un menu déroulant en bas à droite de la fenêtre de l'application. Cette sélection sera effective sur tous les véhicules que l'utilisateur ajoutera jusqu'à ce qu'il choisisse une autre voie. Ce menu déroulant est illustré à la figure suivante :



3. Modification de la tolérance

La tolérance est une variable qui détermine la distance sur laquelle aucun véhicule ne doit être présent pour permettre l'insertion dans le carrefour. Par exemple une tolérance de 5 représente 5% du rondpoint, si i n'y a pas de véhicules dans les 5% du rondpoint situé à la gauche du véhicule celui-ci s'insérera. Cette valeur va impacter directement la densité du trafic. Voici un schéma qui illustre ce principe :



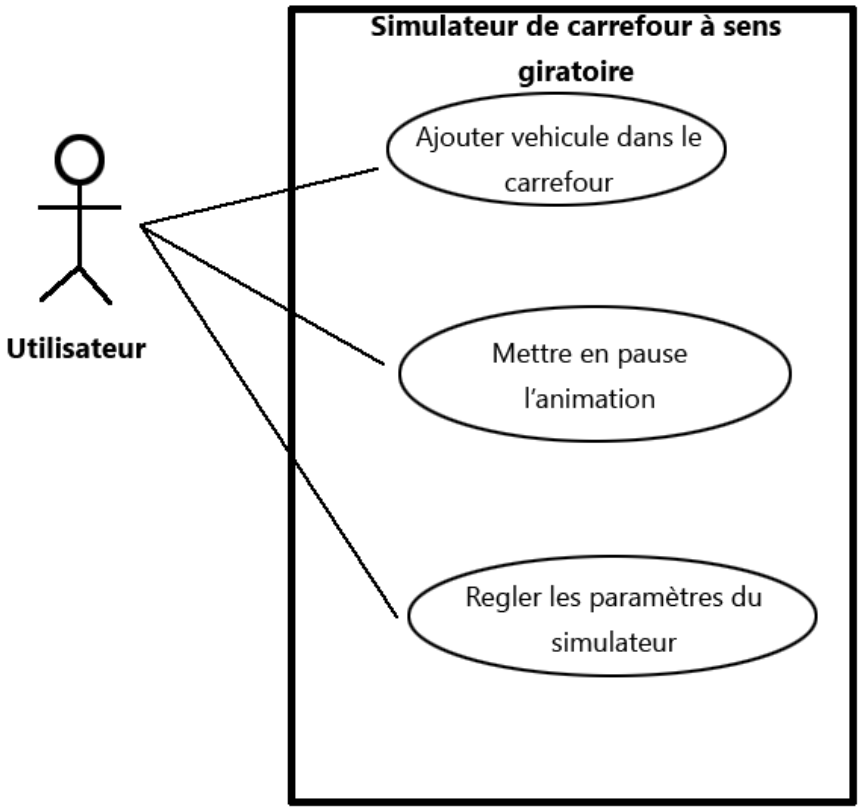
Sur la figure ci-dessus nous pouvons voir que la tolérance s'ajuste grâce à un curseur présent en bas à droite de la fenêtre de l'application. Le tracé bleu symbolise une tolérance de 25%, en effet la voiture jaune qui souhaite s'engager dans le carrefour ne s'engagera pas tant qu'il y aura des véhicules dans la zone bleue (25% du carrefour).

II. Détails techniques

1. Cas d'utilisations

Maintenant nous allons aborder les aspects techniques du produit. Dans un premier temps nous présenterons un diagramme de cas d'utilisation avec quelques scénarios de mise en situations, puis nous montrerons la solution adoptée pour la conception du produit.

Voici le diagramme de cas d'utilisation :



Les fonctionnalités prévues pour le produit sont donc :

- Permettre à l'utilisateur l'ajout de véhicule dans le carrefour.
- Permettre la mise en pause de l'animation.
- Permettre le réglage des paramètres du simulateur.

2. Scénarios

Titre : Utilisation du simulateur de ronds-points par un utilisateur.

Acteur : Utilisateur.

Condition(s) : Lancer le simulateur.

- **Scénario 0** : L'utilisateur veut configurer le simulateur pour le type de rond-point souhaité :

  1) L'utilisateur lance le programme ;
  2) Une fenêtre s'ouvre, proposant les différents types de ronds-points ;
  3) L'utilisateur choisit un type de rond-point ;
  4) Une interface d'utilisation appropriée au choix de l'utilisateur s'ouvre.

- **Scénario 1** : L'utilisateur veut ajouter un véhicule au rond-point :

  1) L'utilisateur configure la sortie prévue pour le véhicule ;
  2) L'utilisateur clique sur le bouton correspondant à la voie par laquelle il veut ajouter le véhicule.
  3) Le simulateur vérifie s'il reste de la place dans le rond-point ;
  4) Le véhicule est inséré dans la voie choisie ;
  5) Le simulateur vérifie si le véhicule peut s'insérer ;
  6) Si oui, le véhicule s'insère. Sinon, il attend dans la file jusqu'à que le simulateur le permette.

- **Scénario 1.b** : L'utilisateur veut arrêter momentanément l'animation :

Conditions : Le programme est lancé et configuré et l'animation n'est pas en pause.

  1) L'utilisateur clique sur le bouton Pause ;
  2) L'animation s'arrête.

- **Scénario 1.c** : L'utilisateur veut relancer l'animation :

Conditions : Le programme est lancé et configuré et L'animation est en pause.

  1) L'utilisateur clique sur le bouton Play ;
  2) L'animation redémarre.

- **Scénario 1.d** : L'utilisateur veut régler la tolérance des véhicules :

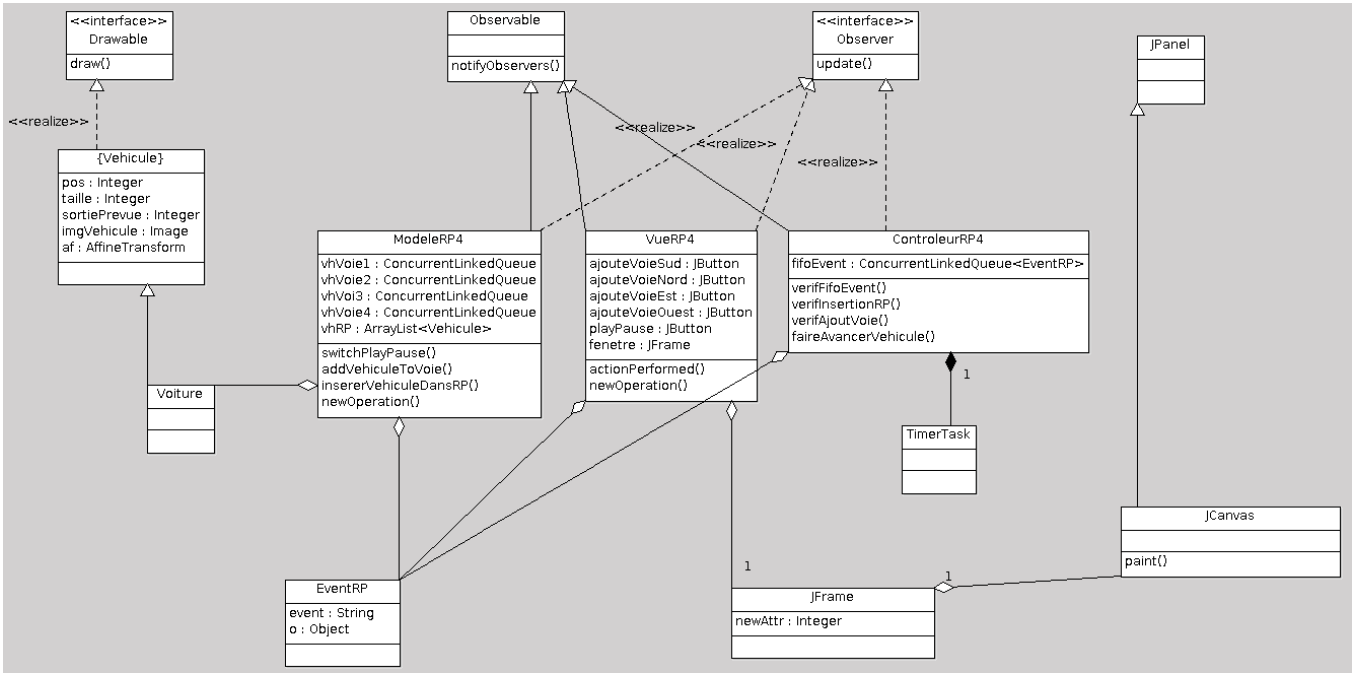
  1) L'utilisateur règle la tolérance en utilisant le curseur destiné à cet effet ;
  2) Le simulateur modifie la tolérance pour les prochaines insertions dans le rond-point.

- **Scénario 2** : L'utilisateur veut fermer l'application :

1) L'utilisateur clique sur la croix ;
2) Le simulateur s'arrête.

3. Diagramme de classes

Le produit est fourni avec le diagramme de classes complet suivant (une version PNG sera disponible attaché à ce rapport) :



Afin de répondre au problème posé de façon optimale nous avons utilisé une conception de type Modèle Vue Contrôleur. L'interface graphique a été réalisé avec la bibliothèque Swing de Java.

4. Compatibilité

L'application a été entièrement réalisé avec java 8, l'application est donc compatible avec tout système du moment que la JVM d'oracle version 8 est installé sur la machine cible aussi bien Windows que Linux. Testé sous Linux Mint Rosa, De, Windows 10 et Windows 8