Projet Linked-Crossing

- *3 rendus valant 65% de la note globale:*

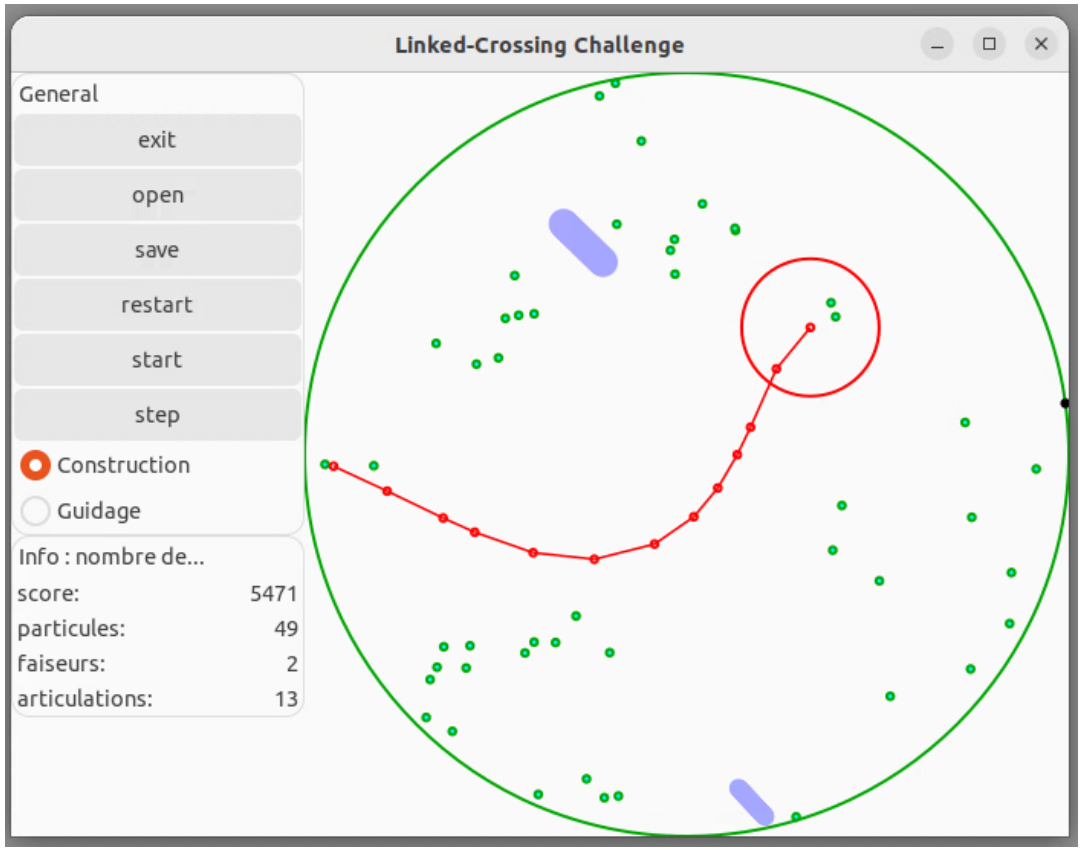
  - *Rendu1 : 30 mars ~22%*
    - *mise au point d'un module de bas niveau **tools** et initialisation par lecture d'un fichier avec detection d'erreurs.*
    - *Syntaxe de test: passage d'un nom de fichier*

      `./projet t01.txt`

  - *Rendu2 : 27 avril ~21%*
    - *Affichage graphique GTKmm et fonctionnement minimal de l'interface (code GTKmm partiellement fourni)*
    - *Simulation limitée aux particules et faiseurs*
    - *Syntaxe de test: passage d'un nom de fichier*

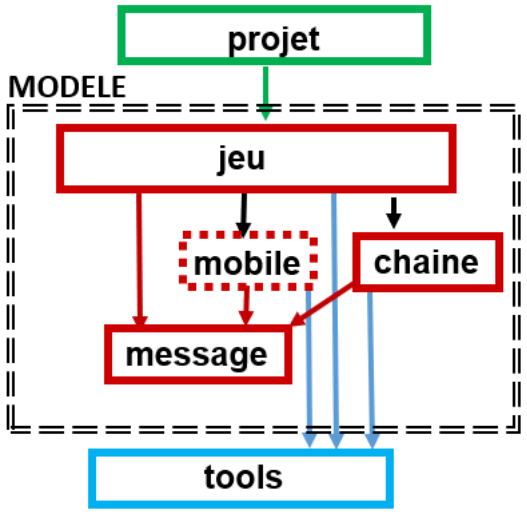
      `./projet test1.txt`

  - *Rendu3 : 25 mai ~22%*
    - *Simulation complete avec interface graphique GTKmm*



Rendu1

Rendu1 : *Ce rendu SANS GTKmm sera toujours testé en indiquant un nom de fichier de test sur la ligne de commande. L'architecture du programme doit respecter cette organisation:*



L'exécution doit :

- *lire un fichier et vérifier la validité des données (sect. 4.2)*
- *construire les structures de données*

Nous fournissons 15 fichiers publics, de `t00.txt` à `t14.txt`

Execution avec affichage d'un message dans le terminal selon le contenu du fichier lu. ex :

```
./projet t00.txt
```

Les messages d'erreur standardisés sont fournis

Un rapport d'activité (pdf) de 0.5 page documente l'activité des deux membres du groupe:
1) brève description de la méthode de travail choisie
***2) pour chaque personne**: liste des fonctions développées/testées seul(e) ou ensemble*

Rendu1 (suite)

Nous essaierons de fournir un autograder ; en attendant on peut utiliser la commande **`diff` :**

1) Mémoriser l'affichage par redirection de la sortie :

```
./projet t00.txt > aff00.txt
```

2) un fichier out00.txt est fourni pour chaque fichier de test t00.txt. La comparaison entre ce fichier out00.txt et votre affichage aff00.txt permet de valider chaque test.

3) l'execution de: `diff -s out00.txt aff00.txt`

produit en cas d'égalité: `Files out00.txt and aff00.txt are identical`

cf exécution sur exemples.

Rendu2

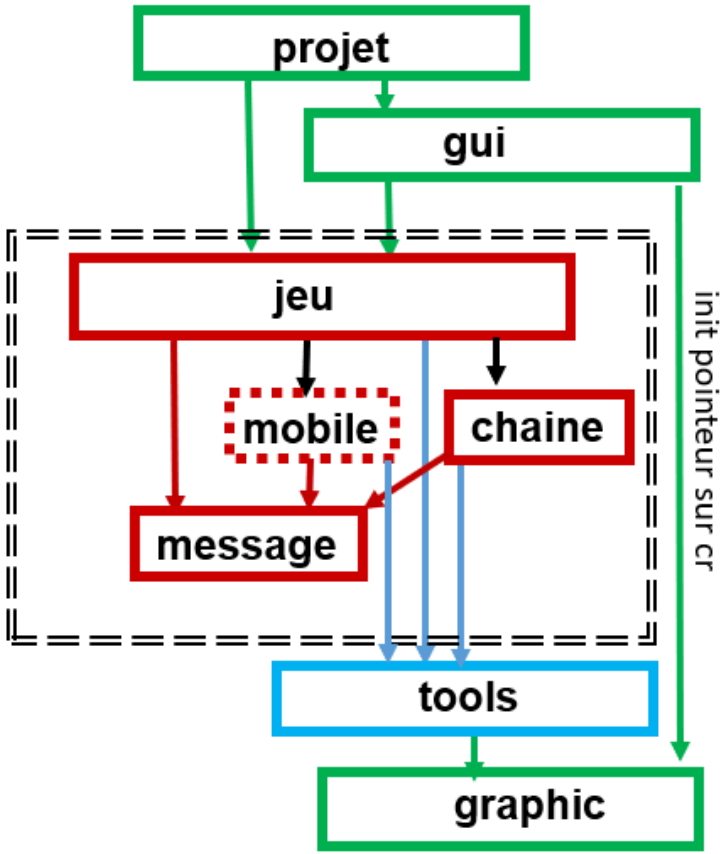
Rendu2 : *Ce rendu avec GTKmm sera testé avec un nom de fichier de test sur la ligne de commande. L'essentiel du code GTKmm sera fourni ; il faudra connecter ce code des sous-systèmes de Contrôle et de Visualisation (en vert dans Fig 11b) au reste du projet.*

le programme continue de vérifier les erreurs à la lecture de fichier mais initialise aussi le dessin. ***La robustesse de l'initialisation sera vérifiée en effectuant plusieurs lectures/sauvegarde/relecture avec/sans erreur et cela sans quitter le programme****.*

L'interface graphique est testée (boutons, radio-button)

L'évolution du jeu est limitée à la gestion des particules + faiseurs

Un rapport de 1 page résume le choix final de structuration des données et décrit l'activité des membres du groupe.



Rendu3

Rendu3 : *Ce rendu avec GTKmm sera (optionnellement) testé en indiquant un nom de fichier de test sur la ligne de commande.*

Exécution complète du jeu avec illustration de tous les mécanismes décrits dans la donnée.

Complété par un rapport de 2 pages documentant un scénario de jeu, les performances, et une synthèse sur la méthode de travail et l'activité de chaque membre du groupe.