

### **Lundi 3 septembre :**

Création de la class Vector3D, des méthodes associés, des tests unitaires de celle-ci et de quelques fonctions utiles. Aucun événements majeurs à signaler. Quelques tests ne passaient pas suite à des erreurs bêtes dans le code mais les erreurs associées ont rapidement été corrigées.

### **Mercredi 5 septembre :**

Ajout de la class Particule et des méthodes associées.

### **Problème rencontré avec GLUT (FT#1) :**

La fonction glutMainLoop() ne retourne jamais → Impossibilité de faire une boucle de jeu classique.

### **Jeudi 6 septembre :**

Résolution du **FT#1** (Problème de la boucle de jeu avec GLUT). La solution consiste à utiliser la fonction glutIdleFunc() fournit par l'API. La fonction passée en paramètre de cette fonction est appelée « dès que GLUT le peu ».

Pour résoudre le **FT#1** il fallait donc placer une référence de la fonction gameloop dans cette fonction, puis ajouter un temps d'attente dans la gameloop pour rendre le frame-rate constant.

Aucun autre fait marquant ce jour.

### **Vendredi 7 septembre :**

Auto formation à openGL .

### **Dimanche 9 septembre :**

Ajout d'une nouvelle architecture comprenant :

- Game : class gérant la démo
- Graphics : class gérant le rendu graphique du moteur
- Input : class Abstraite définissant le standard d'un contrôleur
- Mouse : class permettant de supporter une souris dans la démo
- Keyboard : class permettant de supporter un clavier dans la démo

### **Mercredi 19 septembre :**

Ajout du système de registre force.

### **Problème rencontré (FT#2) :**

Apparition d'un problème sur l'accélération des particules (particules partant à l'infini).

### **Mercredi 26 septembre :**

Résolution du **FT#2** : Le problème était dut à une erreur dans la formule de drag, les particules acculait une erreurs et partait à l'infini.

### **Vendredi 28 septembre :**

Ajout des ressorts. Rien à signaler d'autre.

## Mercredi 3 octobre :

Début du système de collision. Implémentation des classes ParticuleContact, ParticuleContactGenerator et ParticuleContactResolver. Les classes n'ont pas été testées du au découpage du cours en deux séances.

## Jeudi 18 octobre :

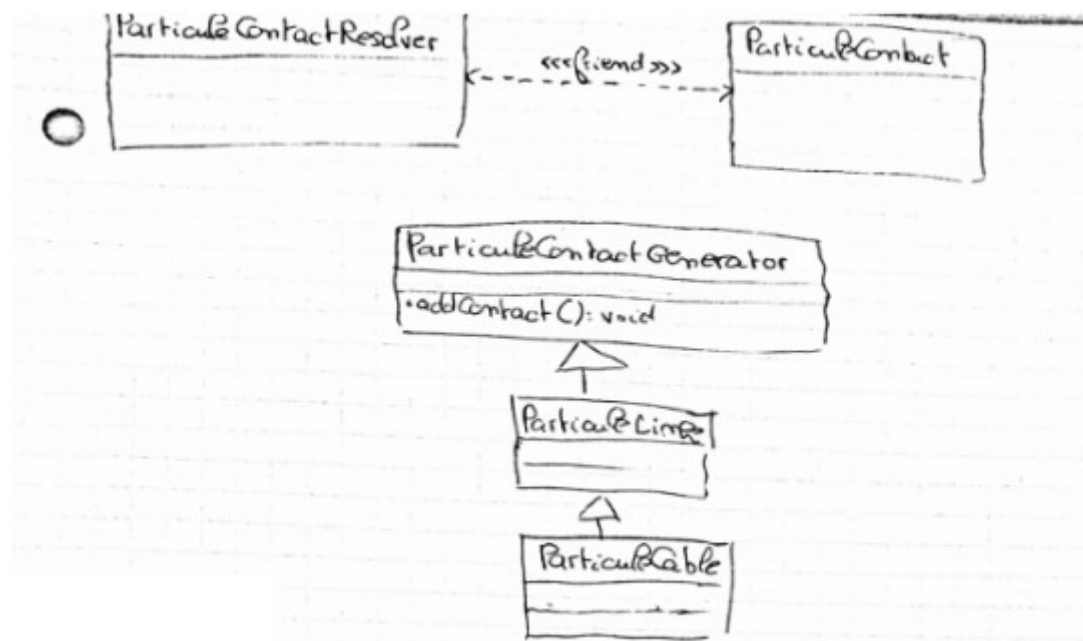
Tests des classes implémentés précédemment et résolution des légères FT associées. Ajout du sol et des rebonds associés.

## Dimanche 21 octobre :

Ajout des câbles et des liens et tests de ceux-ci. Révision de la conception données suite à des problème constaté :

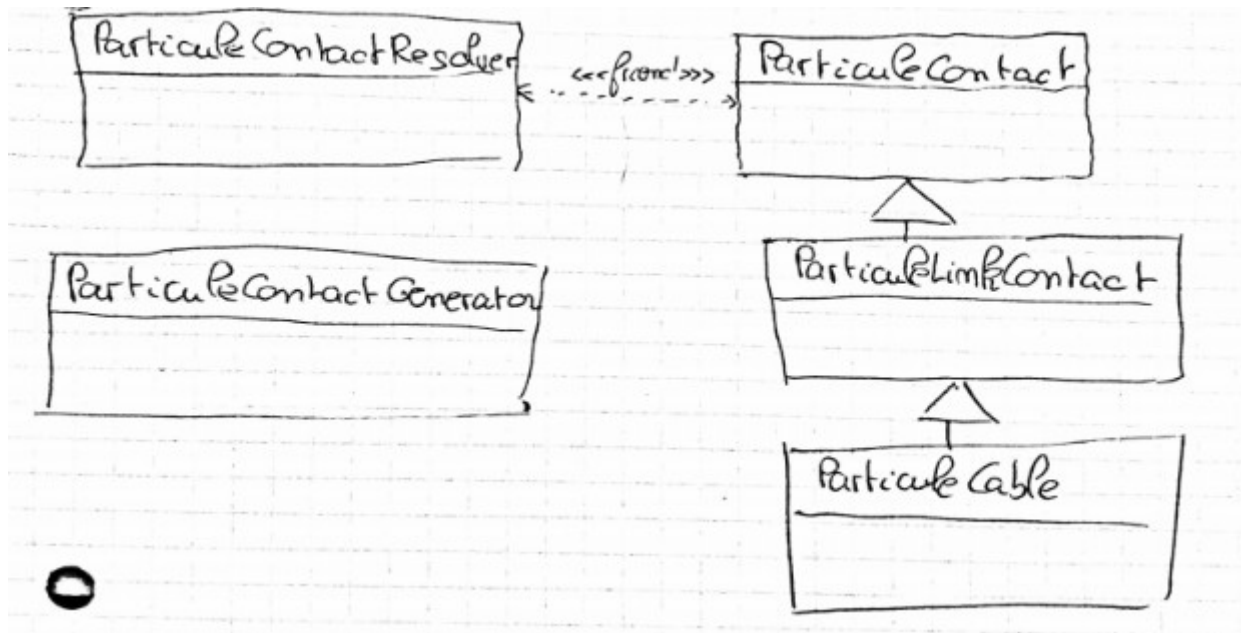
Dans la conception fournit en cour on nous demande de faire hériter la classe ParticuleLink de ParticuleContactGenerator puis de faire hériter ParticuleCable de ParticuleLink. Dans cette conception nous sommes obligés de gérer 3 instances de classe permettant la création de contacts différentes ce qui alourdi et ralenti le code.

### Ancienne conception :



Pour la suite du projet j'ai choisit de créer les classes ParticuleLinkContact et ParticuleLinkCable héritant de la classe ParticuleContact. L'avantage de cette solution est d'utiliser le polymorphisme fournit par le langage C++. Ainsi ParticuleContactResolver n'a besoin de résoudre qu'une seule liste de contacts comprenant à la fois les contacts entre particules, les liens et les câbles.

## Nouvel conception :



## Lundi 22 octobre :

Ajout d'une classe permettant la création de Blob. Classe ajoutée dans la perspective de séparation de Blob. Malheureusement dû à une charge de travail conséquente dans les autres matières cette séparation n'a pas été implémentée.

## Mardi 6 Novembre :

Début de l'implémentation des classes représentant les matrices 4x4 et 3x3.

## Mercredi 13 Novembre :

Finissions des classes représentant les matrices 4x4 et 3x3. Implémentation de la classe Quaternion. Création des tests unitaire pour les matrices.

## Jeudi 15 Novembre :

Ajout des tests unitaires pour les Quaternions. Création de la classe RigidBody et début de l'intégrateur. Implémentation de l'ensemble dans la boucle de jeu.

## Vendredi 16 Novembre :

Ajout du dessins d'un RigidBody avec la bonne orientation.

## Lundi 19 Novembre :

Implémentation des générateur de force. Finissions de l'intégrateur.

## Mardi 20 Novembre :

Implémentation des différents scénarios.

### **Mercredi 28 Novembre :**

Début de l'implémentation de l'octree. Création des classes principales et mini réorganisation du projet

### **Vendredi 30 Novembre :**

Création de la scène de démo. Implémentation et optimisation de l'octree (ajout d'une notion de parent / enfant pour les cellules).

### **Mardi 4 décembre :**

Prédiction de collision implémentée

### **Problème rencontré (FT#3) :**

Dans certain cas la prédiction de collision ne donne rien. Cela est sûrement dû au fait que certain murs ne doivent pas être présent dans l'octree.

### **Mercredi 5 décembre :**

Correction du **FT#3** cela était en effet dû à des murs non pris en compte dans l'octree.

### **Jeudi 6 décembre :**

Implémentation de la détection de collision. Test de celle-ci.

### **Mardi 11 décembre :**

Ajout de l'affichage de la normale lors d'une collision.