

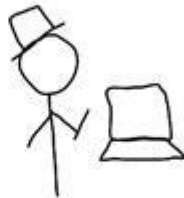
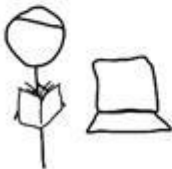




Damos labs

Aurélien David & Adrien  
Ramos

REASONS WHY PEOPLE WHO WORK  
WITH COMPUTERS SEEM TO HAVE  
A LOT OF SPARE TIME... eviljaysm2.com

<p>Web Developer</p>  <p>'Its uploading'</p>	<p>Sysadmin</p>  <p>'Its rebooting'</p>	<p>Hacker</p>  <p>'Its scripted'</p>
<p>3D Artist</p>  <p>'Its rendering'</p>	<p>IT Consultant</p>  <p>'Its your problem now'</p>	<p>Programmer</p>  <p>'Its compiling'</p>

memecenter.com

« spare time »

## [A LONG STOWERIE]

Rapport du projet de C++ portant sur l'optimisation du planning d'une grande école d'informatique.

## Table des Matières

Introduction.....	1
Choix Préliminaires.....	2
Implémentation du programme.....	3
IHM : Les principales fonctionnalités.....	3
Gestion des données .....	7
Fonctionnement détaillé du programme .....	8
Structures de données.....	8
La génération .....	8
L'algorithme .....	10
Améliorations possibles.....	11
Conclusion .....	12
Annexe: .....	13
Qt.....	13
SQLite .....	13
Github.....	13
Sources .....	13

## Introduction

En cette deuxième année de l'EFREI, le cours de Programmation en C++ est venu en remplacement de l'enseignement en programmation C dispensé l'année précédente. Ce changement a impliqué de nouvelles attentes: changer sa façon de concevoir un programme informatique, celui-ci passe d'abord par une phase d'analyse puis de conception.

L'objectif du projet était clair, il s'agissait de créer un système de gestion d'emploi du temps en utilisant les nouvelles notions acquises. C'est un problème qui nécessite la gestion d'un nombre important de contraintes et donc un angle de programmation différent. Il nous a donc fallu étudier longuement la façon dont nous allions aborder le sujet, le concevoir et enfin l'optimiser.

Le dernier projet de l'année précédente, le Problème du Voyageur de Commerce nous avait apporté une première approche de solution lorsque le brute-force n'était pas envisageable avec un algorithme génétique. Il s'agit cette fois-ci de trouver nous même le moyen de palier au brute-force.

Ce projet était ainsi pour nous l'occasion d'étudier la raison de nos parfois trop nombreux problèmes d'emploi du temps, afin d'avoir du recul sur le travail nécessaire à la gestion d'un emploi du temps.

## Choix Préliminaires

Le projet a été réalisé dans notre langage de prédilection, le C++ en utilisant l'IHM (interface homme-machine) Qt, déjà utilisée pour les précédents projets (Voyageur de commerce et Rubiks Cube). Pour la gestion des données nous avons choisi d'utiliser une Base de Données SQLITE. Ces choix nous ont permis de réaliser une interface utilisateur complète et agréable tout en utilisant le modèle MVC qui facilite grandement l'organisation du code. Nous avons également réfléchi à la création d'une application serveur pour l'administrateur et une autre pour les différents utilisateurs, cependant cela risquait de sortir du contexte du projet et difficilement utilisable sur les machines de l'EFREI pour des raisons de sécurités évidentes, c'est pourquoi nous n'avons pas continué sur cette piste.

Afin de concevoir ce projet facilement nous avons utilisé le gestionnaire de version Git. Le système de commit nous permettant de savoir sur quelle partie du code chacun des membres du binôme avait travaillé et de comprendre les modifications. De plus l'utilisation de différentes branches nous a permis d'effectuer des tests pour chercher une meilleure performance sans nuire à la branche principale du programme qui se devait d'être à 100% fonctionnelle. Nous avons ainsi pu concevoir d'un coté le Model et le Controller et de l'autre la Vue sans pour autant ralentir le développement de l'autre membre.

Le code source a été rédigé en anglais, sauf pour les commentaires qui adoptent eux la convention utilisée en TP, semblable à Javadoc.

Pour réaliser notre méthode de résolution nous avons commencé par effectuer des recherches sur les différents algorithmes utilisés pour optimiser un emploi du temps. Après coup nous avons alors décidé de suivre nos propres idées et de réaliser notre algorithme "from scratch" et d'observer le résultat afin de chercher à l'optimiser.

## Implémentation du programme

Profitant du modèle MVC nous avons structuré notre programme de façon à séparer complètement la Vue du Model et du Controller.

### IHM : Les principales fonctionnalités.

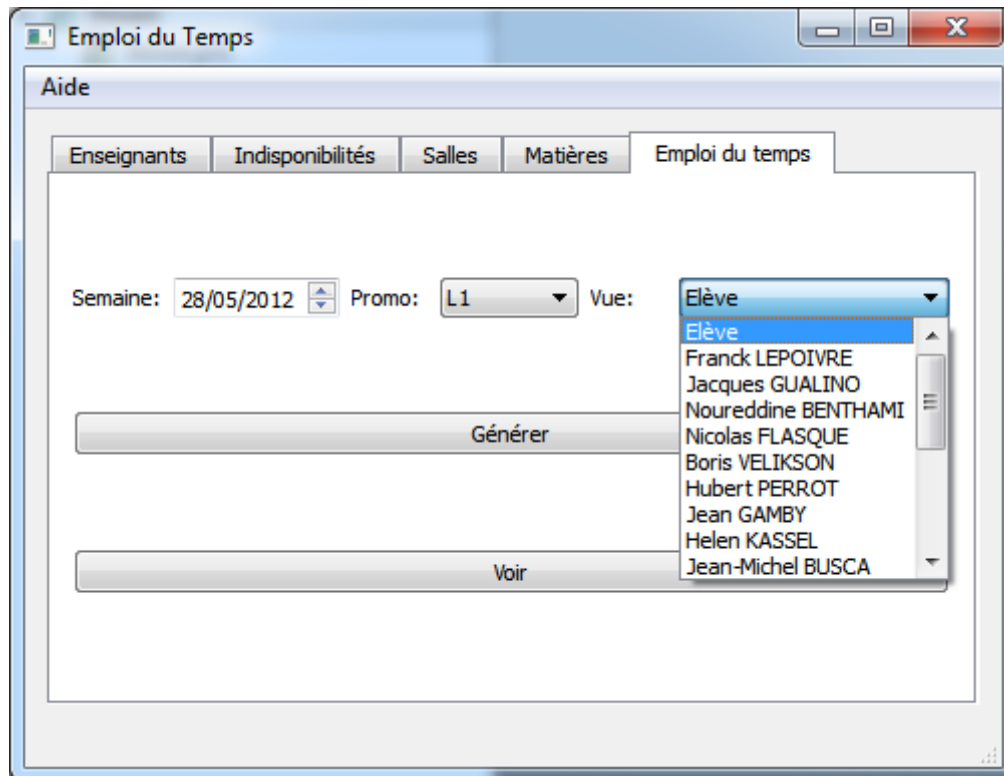


Figure 1- Fenêtre Principale

Au lancement du programme l'emploi du temps est vide, suivant la philosophie KISS (Keep it Simple, Stupid), nous avons choisi un menu restreint proposant le simple nécessaire, la sélection de la vue (prof ou élève) et la génération.

Nous avons ajouté une série d'options pour étendre les fonctionnalités du programme. Nous proposons ainsi la saisie des différentes contraintes via les onglets.

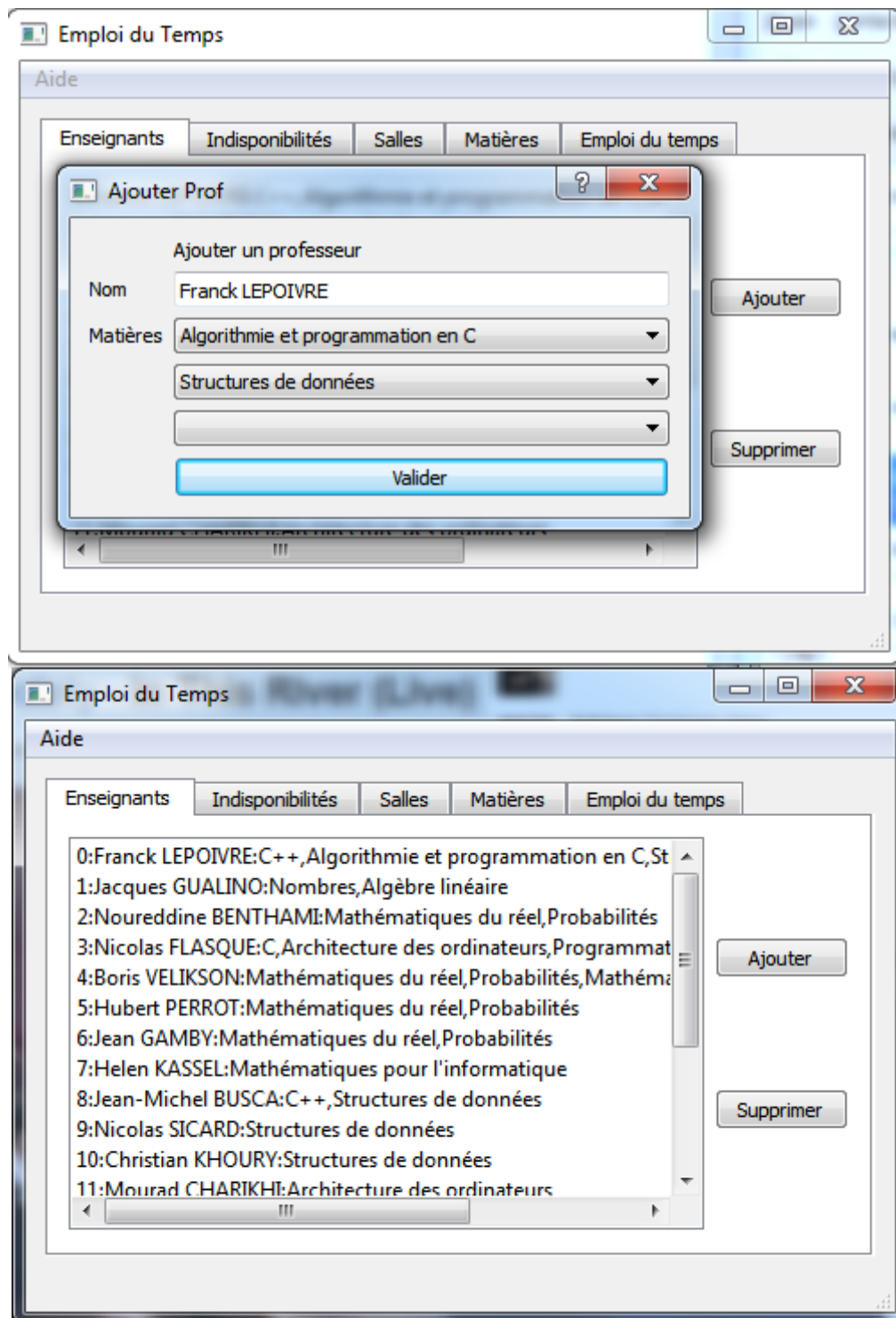


Figure 2- Gestion des professeurs

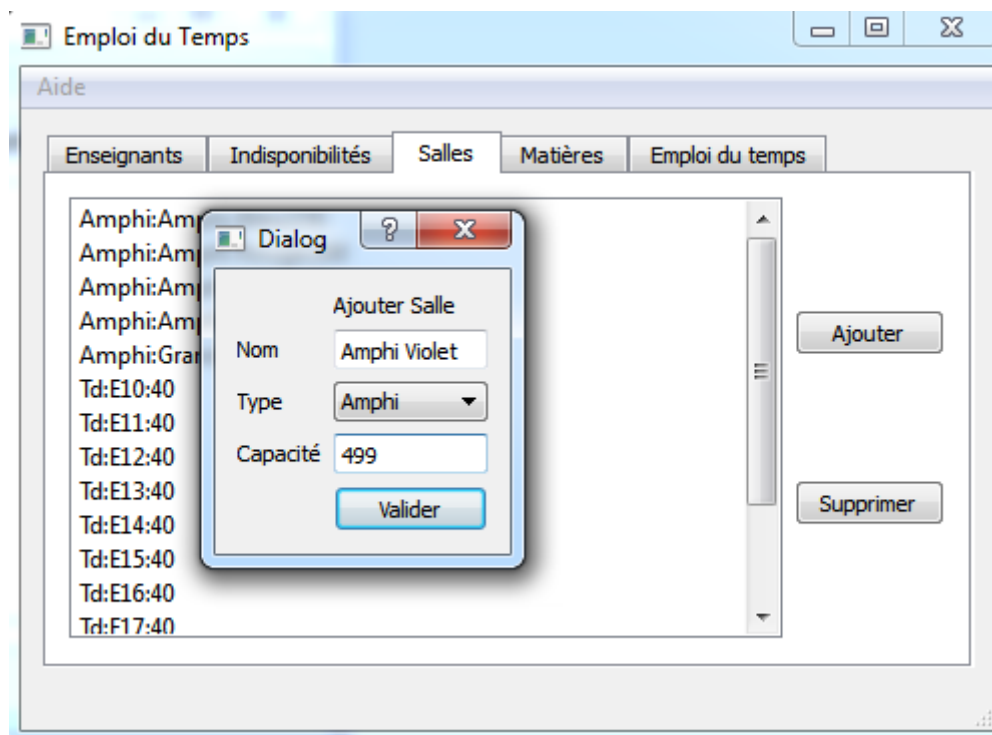


Figure 3 - Ajout d'une salle

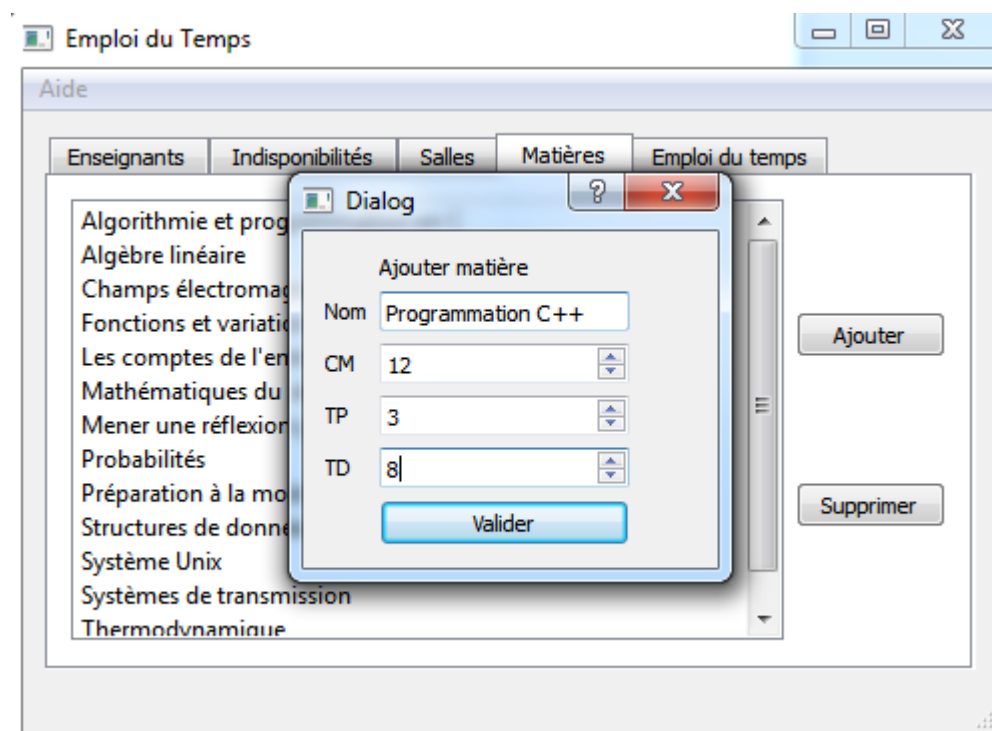


Figure 4 - Ajout d'une matière

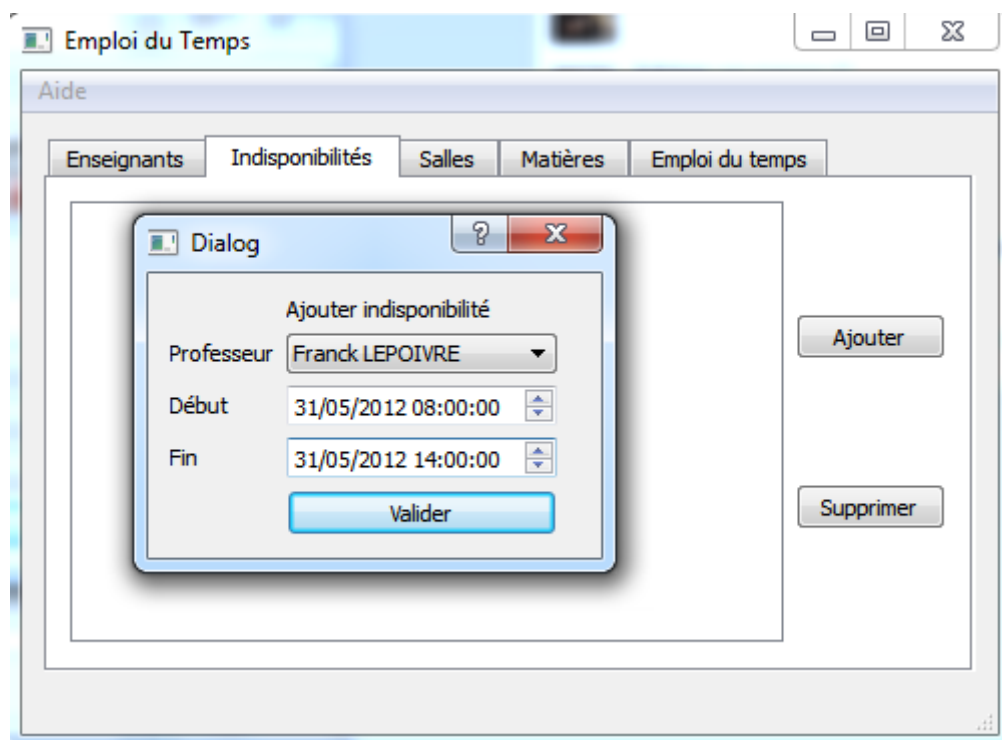


Figure 5 - Ajout d'une indisponibilité

Toute référence à notre date de soutenance est fortuite.

Nous arrivons à notre objectif qui est la génération de l'emploi du temps:

The screenshot shows a window titled 'Emploi du temps' displaying a timetable for 'mercredi 30/05/12'. The timetable is organized into columns for different rooms (L1D, L1A, L1B, L1C, L1D) and rows for different time slots (from 8h00 to 11h45). The content of the timetable is as follows:

	L1D	L1A	L1B	L1C	L1D
8h00					
8h15		Cours n°0 de Al...	Cours n°0 de Al...	Cours n°0 de Al...	Cours n°0 d
8h30					
8h45					
9h00					
9h15					
9h30		Amphi Bleu	Amphi Bleu	Amphi Bleu	Amphi Bleu
9h45					
10h00		Cours n°1 de Al...	Cours n°1 de Al...	Cours n°1 de Al...	Cours n°1 d
10h15					
10h30					
10h45					
11h00					
11h15		Amphi Bleu	Amphi Bleu	Amphi Bleu	Amphi Bleu
11h30					
11h45		Cours n°2 de Al	Cours n°2 de Al	Cours n°2 de Al	Cours n°2 d

Figure 6 - Emploi du temps



## Gestion des données

Le sujet nous proposait de fournir un jeu de données en entrée, c'est pourquoi nous avons choisi de créer une Base de Données SQLite utilisée par de nombreux logiciels comme Firefox.

Name	Object	Type	Schema
profes	table		CREATE TABLE profes (matieres TEXT, name TEXT)
matieres	field	TEXT	
name	field	TEXT	
promos	table		CREATE TABLE promos (name TEXT, groups TEXT)
name	field	TEXT	
groups	field	TEXT	
rooms	table		CREATE TABLE rooms (type TEXT, name TEXT, capacity NUMERIC)
type	field	TEXT	
name	field	TEXT	
capacity	field	NUMERIC	
programme	table		CREATE TABLE programme (promo TEXT, matieres TEXT)
promo	field	TEXT	
matieres	field	TEXT	
matieres	table		CREATE TABLE matieres (name TEXT, CM NUMERIC, TD NUMERIC, TP NUMERIC)
name	field	TEXT	
CM	field	NUMERIC	
TD	field	NUMERIC	
TP	field	NUMERIC	

Figure 7 - Structure de la base de données

	matieres	name
1	C++,Algorithmie et programmation en C,Structures de données	Franck LEPOIVRE
2	Nombres,Algèbre linéaire	Jacques GUALINO
3	Mathématiques du réel,Probabilités	Noureddine BENTHAMI
4	C,Architecture des ordinateurs,Programmation en C++ et UML	Nicolas FLASQUE
5	Mathématiques du réel,Probabilités,Mathématiques pour l'informatique	Boris VELIKSON
6	Mathématiques du réel,Probabilités,Statistiques	Hubert PERROT
7	Mathématiques du réel,Probabilités,Statistiques	Jean GAMBY
8	Mathématiques pour l'informatique	Helen KASSEL
9	C++,Structures de données	Jean-Michel BUSCA
10	Structures de données	Nicolas SICARD
11	Structures de données	Christian KHOURY
12	Architecture des ordinateurs	Mourad CHARIKHI
13	Analyse des données,Statistiques	Hervé BERTRAND
14	Champs électromagnétiques,Propagation de l'information,Systèmes numéri	Isabelle SIROT
15	Champs électromagnétiques,Propagation de l'information	Aissa TABBI
16	Champs électromagnétiques	Anica LEKIC
17	Systèmes de transmission	Pierre PROT

Figure 8 - Extrait de la table "profes"

La base complète peut être explorée et modifiée à l'aide d'un logiciel tel que SQLite Database Browser (lien en annexe).

## Fonctionnement détaillé du programme

Nous avons vu la structure générale du projet, passons maintenant en l'étude complète de nos structures de données ainsi que la génération d'emploi du temps.

## Structures de données

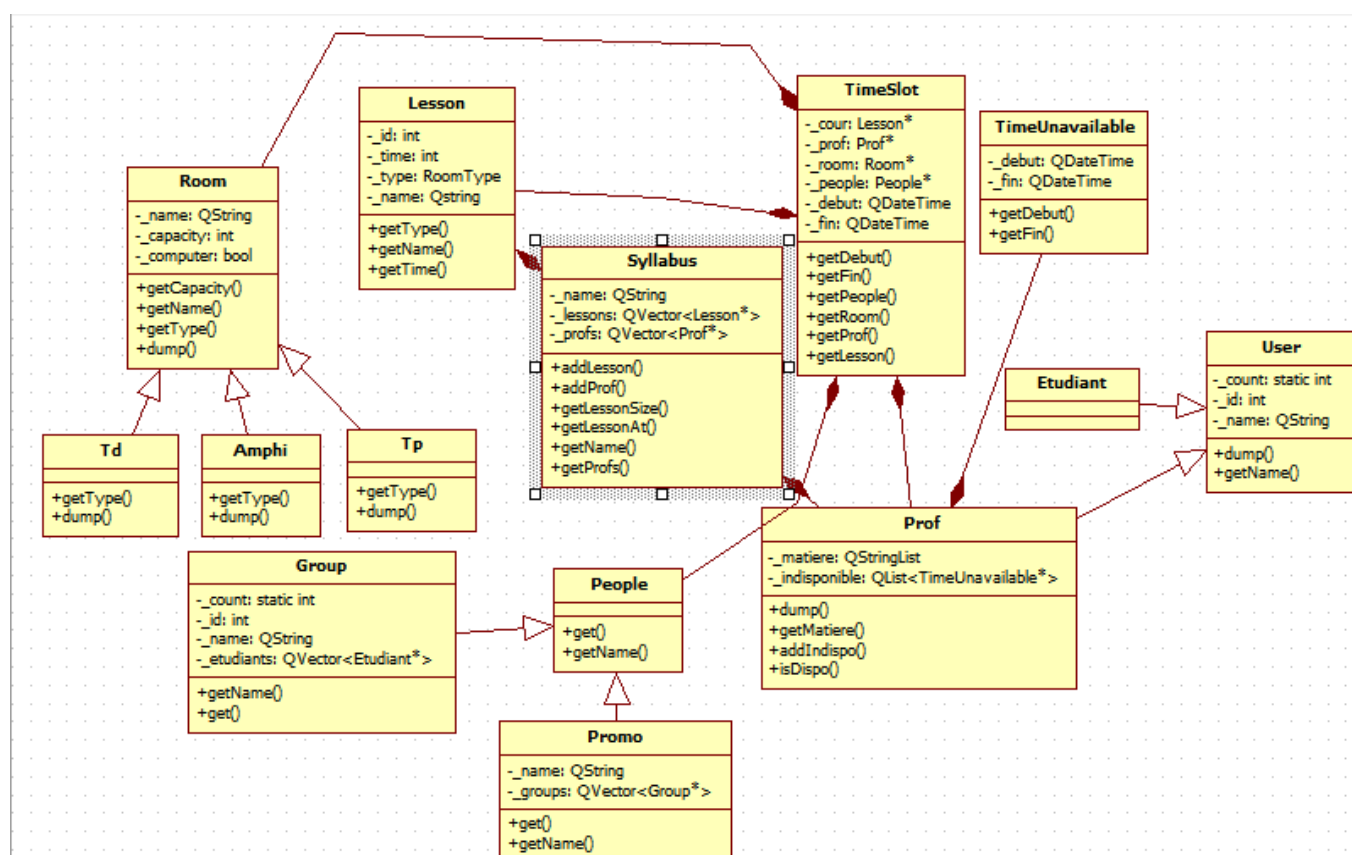


Figure 9 - Diagramme de classes du Model

## La génération

La première étape consiste à récupérer les contraintes depuis la Base de Données ainsi que des modifications apportées par l'utilisateur au programme.

Lorsque le module de génération est appelé, le Controller initialise la génération qui utilise le design pattern "Strategie" (pattern permettant de changer l'algorithme utilisé de façon dynamique.)

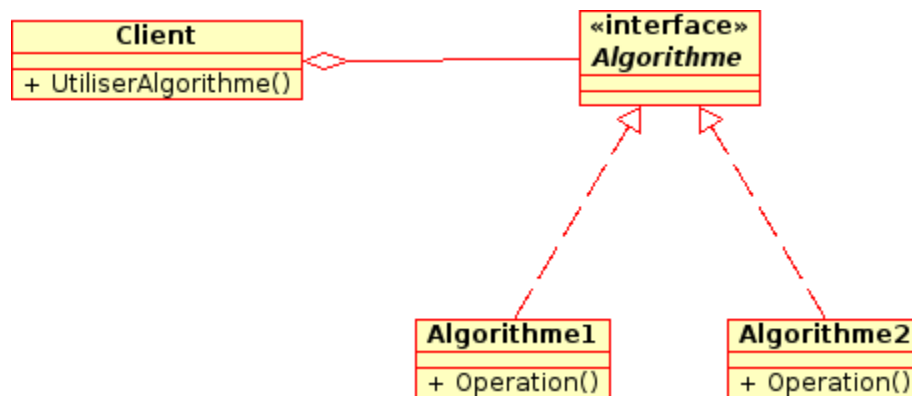


Figure 10 - Diagramme UML du pattern stratégie

Bien que très simple ce pattern nous simplifie la génération et surtout nous permettra d'envisager d'étudier d'autres algorithmes pour poursuivre notre projet et en les mélangeant obtenir le meilleur résultat possible.

### *L'algorithme*

Voici la partie principale du projet, devant le nombre de contraintes imposées nous avons choisi de générer les cours un par un et de les placer au premier endroit qui respectait toutes les conditions. Grace au nombre important de contraintes nous obtenons malgré tout un résultat convenable, l'emploi du temps est suffisamment varié.

La première étape consiste à récupérer les différentes matières, ensuite pour chacune des promos possédant cette matière (en effet des promos différentes peuvent étudier la même matière comme les L3 et les L1), on récupère ainsi les différents cours à ajouter.

L'étape suivante est de chercher la première disponibilité du cours en parcourant la liste des cours précédents, on vérifie qu'une salle et un prof sont disponibles en cherchant la prochaine disponibilité dans le cas inverse.

Une fois que le cours respecte toutes les contraintes on l'ajoute simplement à la liste.

### ***Améliorations possibles***

L'état actuel du projet est tout à fait fonctionnel, cependant nous espérons de meilleures performances, nous générons un emploi du temps qui respecte toutes les contraintes que nous nous sommes imposés. Il reste cependant des possibilités d'améliorations auxquelles nous avons réfléchi mais pas encore trouvé le temps d'implanter.

La première est que le choix d'un algorithme génétique aurait sans doute été judicieux pour raccourcir le temps de génération, un mélange peut être envisagé via le design pattern que nous utilisons.

La seconde est de générer l'emploi du temps en favorisant les professeurs plutôt que les élèves comme nous le faisons actuellement. En effet les professeurs présentent plus d'indisponibilité de part leur fonction que les élèves.

## Conclusion

Nous avons finalement réussi à réaliser notre propre gestionnaire d'emploi du temps. Notre programme permet de produire un emploi du temps honorable. Bien que d'autres algorithmes auraient pu permettre d'obtenir de meilleures performances, nous avons cherché à réfléchir nous même plutôt que de nous baser sur des algorithmes tiers.

Programmer ce logiciel nous a permis avant tout de consolider notre programmation en C++ mais aussi d'avoir un certain recul sur la difficulté à produire un emploi du temps, étant donné le nombre de contraintes importants.

Pour terminer, il est dommage que la période de réalisation du projet ai été si chargée, car c'était certainement le projet le plus ambitieux que nous ayons eu la chance de commencer à réaliser...

## Annexe:

### Qt

*“Qt est un framework orienté objet et développé en C++. Il offre des composants d’interface graphique, d’accès aux données, de connexions réseaux...”*

Qt permet la portabilité entre systèmes d’exploitation.

### SQLite

SQLite est une bibliothèque écrite en C qui propose un moteur de base de données relationnelles accessible par le langage SQL.

<http://sqlitebrowser.sourceforge.net/>

### Github

L’intégralité du gestionnaire de version du projet est disponible de façon open source à l’adresse suivante: (Pour des raisons évidentes de protection du code source vis à vis des autres binômes, prévoir un délais de 1 ou 2 jours après la date de rendu officielle)

<https://github.com/spyl94/Stowe>

### Sources

*Documentation interne à Qt Creator.*

*Etude des Design Pattern:*

- <http://abrillant.developpez.com/tutoriel/java/design/pattern/introduction/>
- <http://come-david.developpez.com/tutoriels/dps/>