

RAPPORT DE PROJET

Qualité de programmation :

« Gestion d’emploi du temps »

Alexis ROOS

Daniel WETTEL

Jean-Philippe DUBUC

Romain TEULÉ

L3 Informatique

Année universitaire 2016-2017

SOMMAIRE

[Introduction 2](#_Toc471680087)

[Présentation du projet 3](#_Toc471680088)

[Les contraintes : 3](#_Toc471680089)

[Les objectifs : 3](#_Toc471680090)

[Élaboration du projet 5](#_Toc471680091)

[Partage du travail : 5](#_Toc471680092)

[Choix techniques : 5](#_Toc471680093)

[Résultats obtenus 7](#_Toc471680094)

[État final du projet : 7](#_Toc471680095)

[Améliorations possibles : 8](#_Toc471680096)

[CONCLUSION 9](#_Toc471680097)

# Introduction

Nous avons choisi le projet « Gestion d’emploi du temps » parmi les sujets proposés car il nous paraissait intéressant d’essayer de programmer un emploi du temps comme celui du site e-services de l’UHA. Ce projet est pour nous l’occasion de concrétiser toutes les connaissances accumulées au cours de l’année en programmation C++. Qualité de programmation oblige, l’accent sera évidemment mis sur la création de code propre, notion fondamentale pour travailler le plus professionnellement possible en entreprise.

L’un des objectifs de ce projet étant également de manipuler un outil de gestion de versions (Git), nous avons décidé de recourir à GitHub pour son efficacité et sa facilité d’utilisation. Ainsi, l’ensemble de notre travail est donc consultable sur le dépôt GitHub disponible à cette adresse : <https://github.com/DanielLeThug/CalendrierCpp>. Notre équipe est composé de quatre étudiants, dont voici les noms et pseudo sur GitHub :

* Alexis ROOS -> allefou
* Daniel WETTEL -> DanielLeThug
* Jean-Philippe DUBUC -> teezul
* Romain TEULÉ -> romainTeule

Dans ce rapport, nous verrons dans une première partie une courte présentation du projet et des principaux objectifs. Dans une seconde partie nous détaillerons les moyens mis en œuvre pour répondre à ces objectifs. Enfin, nous présenterons les résultats obtenus ainsi que les possibles améliorations de notre programme.

# Présentation du projet

## Les contraintes :

Le programme doit être écrit en C++ et l’utilisation du gestionnaire de versions Git est obligatoire. Comme indiqué dans l’introduction ci-dessus, nous avons choisi d’utiliser GitHub pour faciliter notre travail. Le projet devra fonctionner en mode console dans un premier temps, puis une interface graphique devra être mis en place.

Le choix de l’infrastructure utilisée pour cette interface étant libre, nous avons décidé de l’implémenter avec Qt Creator qui permet de créer rapidement une interface sans avoir à coder une grosse quantité de lignes de code.

Pour nous assurer du bon fonctionnement de notre programme, nous devons également réaliser des tests unitaires avec Catch et documenter notre code avec Doxygen.

## Les objectifs :

Le sujet était la création et l’implémentation d’un programme permettant la gestion d’emploi du temps dans un établissement scolaire, donc prenant en compte les notions de filière, de professeurs et d’étudiants, de salles, afin de gérer (créer, manipuler et supprimer) et d’afficher les cours des différentes filières.

L’objectif principal de ce projet était d’appliquer les notions vues dans le cours de qualité de programmation, matière ayant pour but de nous apprendre les manières de travailler et les principes à respecter afin de produire un code propre, robuste, maintenable, agréable à lire et remanier.

Ainsi, il fallait, en plus de rendre un code fonctionnel et remplissant les attentes du sujet, faire un effort tout particulier à la propreté de celui. Les classes et méthodes devaient réfléchis et le rôle de chacune ne pas être confus, le code lisible et claire, chaque ligne devant être comprise rapidement.

Il fallait également implémenter des tests unitaires, autre notion importante de la matière. Ces tests avaient pour but non seulement de s’assurer du bon fonctionnement des fonctions mais surtout devaient permettre de définir leur rôle, en réfléchissant à ces tests avant de commencer à coder les fonctions .

Finalement, une documentation permettant une meilleure compréhension du programme était attendue, pour cela, nous avons utilisé doxygen qui permet non seulement de commenter les classes et méthodes mais propose aussi la création de graphe permettant de mieux visualiser l’implémentation du programme.

# Élaboration du projet

## Partage du travail :

Le partage du travail s’est fait spontanément, chacun choisissant ce sur quoi il voulait travailler en le signalant sur un groupe de discussion Facebook, en fonction de ses capacités. De plus, ce groupe nous permettait de s’entraider en permanence ainsi que de prendre les décisions vis-à-vis du programme ensemble.

La participation de chaque membre de l’équipe au projet peut être suivi via notre dépôt GitHub.

## Choix techniques :

Tout d’abord, concernant la représentation d’une année, nous avons fait le choix d’utiliser trois classes : Annee, Semaine, Journee.

Ainsi, Annee contient un tableau de pointeur sur Semaine, Semaine un tableau de pointeur sur Journee, et Journee un tableau de pointeur sur des Cours. C’est le choix qui nous semblait le plus logique et le plus adapté à la situation. Ainsi une Journee ne peut exister en dehors d’une semaine et lui est donc liée, et nous n’avons pas la nécessité de garder des informations sur la semaine dans Journee. Même chose pour une Semaine et une Annee : le numéro de la semaine sera obtenu grâce à la position dans le tableau contenu dans Annee.

Ensuite, pour pouvoir gérer l’année scolaire de chaque filière, chaque instance de la classe Filiere contient un pointeur sur une Annee, qui lui est propre. C’est cette Annee qui contiendra les différents cours de la filière ( à travers la structure Annee -> Semaine -> Journee -> Cours vu précédemment ). C’est donc à travers la classe Filiere que sera fait l’accès à une Annee. De plus, pour lier les étudiants et les filières, Filiere contient un tableau de pointeurs sur Etudiant, tableau représentant ainsi les différents étudiants d’une filière donnée .

Concernant la représentation des professeurs et des étudiants, nous nous sommes aperçus des points communs entre les deux entités : les deux possèdent les champs nom et prénoms. Nous avons donc décidé d’utiliser l’héritage en créant une classe Individu, contenant ces deux champs, et en faisant hériter de celle-ci les classe Professeur et Etudiant, qui rajoutent les champs et méthodes propres aux deux roles (matière enseignée et numéro étudiant respectivement).

Nous avons également créé deux classes afficheurs (afficheurConsole et afficheurCSV) héritant d’une classe virtuelle afficheur pour permettre un éventuel usage du polymorphisme. AfficheurConsole permet la sortie console et est utilisé par la classe principale et afficheurCSV permet une éventuelle sortie dans un format lisible par Excel, pour permettre la sauvegarde et l’exportation des emplois du temps.

Chaque classe dispose de getters et de setters permettant un accès clair aux données nécessaires aux fonctionnements des autres classes.

La classe principale, celle qui permet de gérer les emplois du temps des différentes filières c’est-à-dire la classe EmploiDuTemps, contient la liste des filières, salles et profs (sous formes de tableau de pointeur) et les différentes méthodes permettant de choisir une filière et de visualiser ses cours, d’en ajouter ou d’en supprimer, ainsi que les méthodes permettant de gérer les professeurs, salles et filières. Il est ainsi possible de créer de nouvelles filières, de leur ajouter des étudiants, de changer leur intitulé. Pareil pour les professeurs, il est possible d’en ajouter, d’en supprimer ou de les modifier.

Un cours est représenté par la classe Cours, qui contient un pointeur sur un professeur (l’intitulé du cours est déterminé par la matière enseignée par celui-ci), un pointeur sur une Salle, un entier contenant l’horaire de début et un autre contenant la durée (celle-ci étant de deux heures par défaut). Nous pensions au début y ajouter également un pointeur sur une filière mais cela aurait été redondant, du fait de la structure que nous avons choisi. Nous avons pris soin de stocker le minimum nécessaire d’information sur les cours nécessaires, le reste (jour et semaine, étudiants concernés etc.) pouvant être retrouvés en amont.

# Résultats obtenus

## État final du projet :

L’interface graphique n’a pas pu être terminé dans les délais imposés. Voici néanmoins les captures d’écran des fenêtres réalisés pour l’instant :

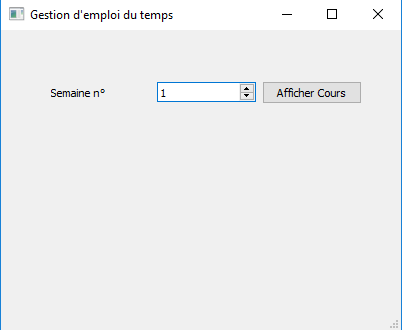


Figure 1 : Menu Principal (non terminée)

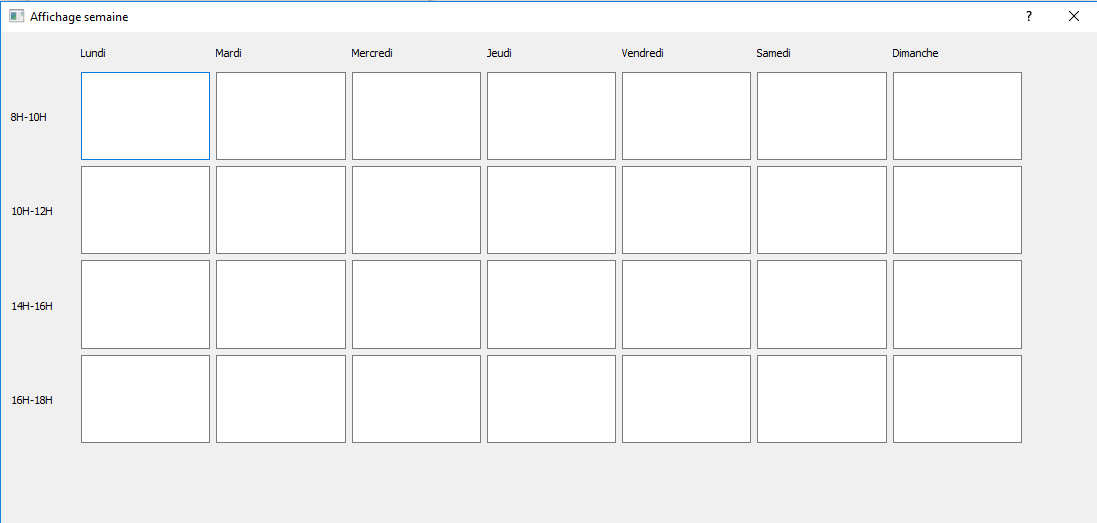


Figure 2 : Fenêtre d'affichage d'une semaine

Concernant le code, toutes les méthodes ont été écrites et la majorité fonctionnent.

Cependant, certaines erreurs subsistent dans des méthodes de la classe EmploiDuTemps , erreurs pouvant provoquer des crash du programme. Il semblerait que ces erreurs soient principalement liées à l’utilisation des pointeurs. Il aurait pu être judicieux de faire plus attention et de réfléchir davantage avant d’écrire certaines méthodes agissant sur ces pointeurs. Il pourrait maintenant être intéressant de remanier ce code en prenant soin de manipuler correctement les différents tableaux de pointeurs

## Améliorations possibles :

Plusieurs méthodes pourraient être intéressantes à rajouter, par exemple :

* Affichage des emplois du temps par salle …
* … ou par professeur
* Pouvoir remplir toute une journée ou semaine avec un cours ou événement (vacances par exemple)
* Permettre l’importation et l’exportation : les méthodes permettant l’exportation au format CSV d’une semaine de cours fonctionnant déjà, il suffirait de rajouter une méthode lisant ces fichiers et créant les cours enregistrés.

Pour l’instant, l’emploi du temps fonctionne avec 4 créneaux horaires, il serait plus réaliste de se défaire de ce modèle. De même, actuellement un professeur enseigne une matière et une seule, pour des soucis de réalisme il serait donc judicieux de permettre à un professeur d’enseigner plusieurs matières.

# CONCLUSION

Dans le cadre de notre projet nous devions réaliser un programme en C++ permettant de gérer l’emploi du temps des étudiants. Nous nous sommes attelés dans un premier temps à réaliser la structure de base de notre application avant de faire l’affichage console et l’export au format CSV. L’interface graphique n’a pas été terminé vue le manque de temps auquel nous avons dû faire face.

Les différentes exigences imposées par le projet nous ont poussé à faire attention à chaque ligne de code, chaque nom de variable afin de pouvoir proposer un code le plus propre possible.

L’utilisation de Git nous a familiarisé avec cet outil très utile, rendant plus facile la collaboration au sein du groupe ainsi que l’archivage.

L’ensemble du groupe pense avoir fait de son mieux pour tenter de faire de ce projet une réussite et même si tous les objectifs n’ont pas été atteint cette expérience aura été positive et nous aura permis de nous familiariser avec le travail de groupe, l’utilisation de Git et des tests unitaires avec Catch. En effet, même si certaines parties du code ne sont pas totalement fonctionnelles, nous avons néanmoins pu mettre en pratique les notions vues en cours et réellement prendre conscience de leur importance dans l’écriture d’un code robuste.

En conclusion, malgré la déception de ne pas avoir réussi à produire un code à 100% fonctionnel, ce projet reste une expérience enrichissante et intéressante.