

**PlanIFTicateur**

**Travail pratique 4**

GLO-2004 – Génie Logiciel Orienté Objet – Hiver 2015

Travail présenté à

M. Jonathan Gaudreault

Chayer, Philippe [Philippe.chayer.1@ulaval.ca](mailto:Philippe.chayer.1@ulaval.ca) IFT PHCHA47

Khouma, Abdou [abdou.khouma.1@ulaval.ca](mailto:abdou.khouma.1@ulaval.ca) GIF ABKHO9

Gadoury, Gabriel [Gabriel.gadoury.1@ulaval.ca](mailto:Gabriel.gadoury.1@ulaval.ca) IFT GAGAD1

Yéo, Clotioloman [Clotioloman.yeo.1@ulaval.ca](mailto:Clotioloman.yeo.1@ulaval.ca) GLO CLYEO1

Table des matières

[Introduction 3](#_Toc416948900)

[Interfaces utilisateurs 4](#_Toc416948901)

[Modèle du domaine 6](#_Toc416948902)

[Modèle des cas d’utilisation 8](#_Toc416948903)

[Architecture logique 10](#_Toc416948904)

[Diagrammes de classes de conception 11](#_Toc416948905)

[Diagrammes de séquences 12](#_Toc416948906)

[Explications du design 13](#_Toc416948907)

[Gestion de projet 14](#_Toc416948908)

[Conclusion 15](#_Toc416948909)

# Introduction

Ce rapport consiste en une analyse de la faisabilité pour la conception d’un logiciel permettant la gestion et la création d’horaire. Cette analyse sera basée sur les informations fournies par le client. Le but du logiciel est de faciliter la gestion des horaires de session effectuée par la direction de programme.

- ***PlanIFTicateur*** est un programme apte à construire un horaire de session de façon interactive et peut, si souhaité, faire la génération de votre horaire de façon optimale (rapide, efficace et flexible). Le but primaire est de rendre agréable la construction de l’horaire tout en réduisant le temps consacré à ce dernier.

- Une planification automatique optimale grâce à un algorithme de recherche efficace se basant sur les restrictions et les statistiques de ce qu’est un bon horaire.

- Une interface simple et intuitive visant à aider l’utilisateur dans son travail.

- Un suivi simple et efficace grâce à une fonctionnalité permettant de prendre des notes en lien avec la grille horaire que vous construisez.

- Les horaires créés seront en tous points valides grâce à des fonctionnalités de validations optimales. Ces fonctions rendront facile et stimulante la création d’un horaire grâce à son aspect visuel et dynamique.

- Une importation des données et restrictions des cours instantanée facilitant le travail et limitant les erreurs ou les oublis.

- Des statistiques pertinentes disponibles pour la grille horaire en construction visant à l’optimisation de votre horaire.

- Une exportation de fichier simple et facile à des fins d’utilisation hors programme.

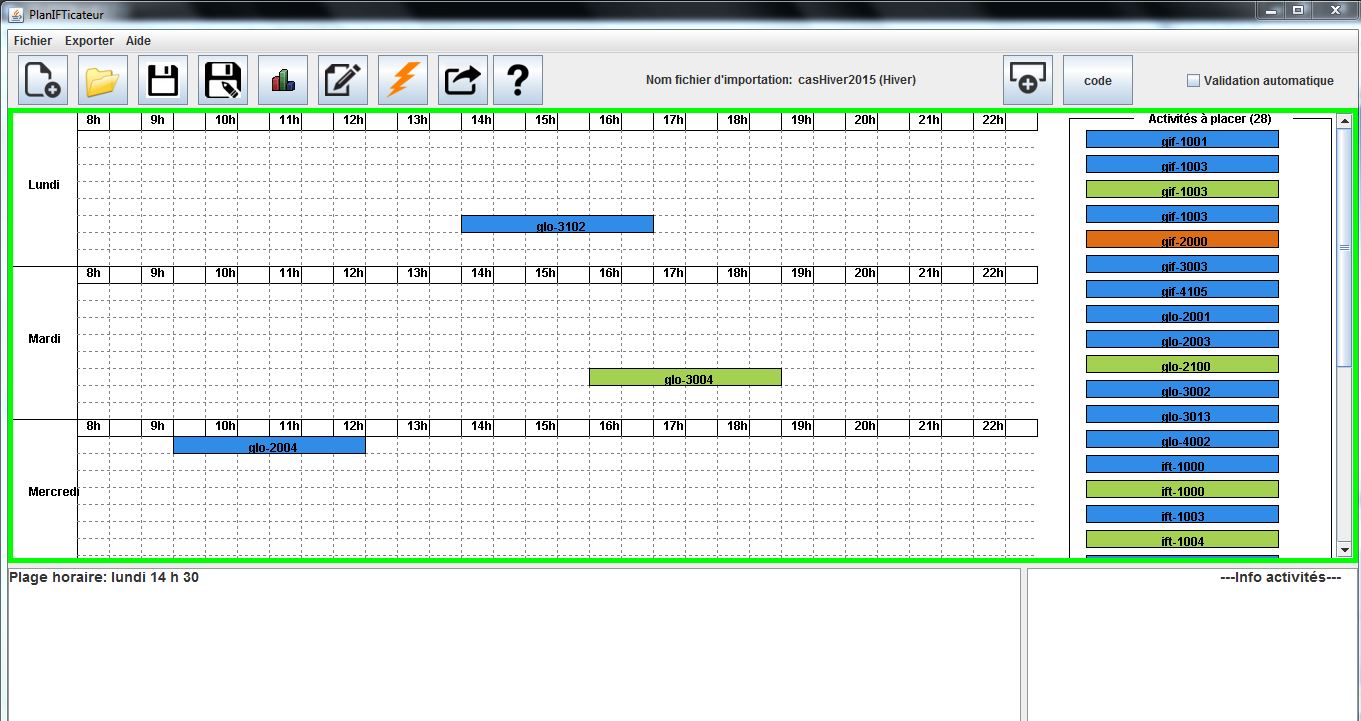
- Un historique est disponible pour un suivi efficace de toutes les modifications réalisées sur votre planification.

# Principales interfaces utilisateurs

Tout d’abord, une interface utilisateur doit miser sur l’aspect ergonomique. Tout en respectant les critères du client, nous avons réussi à rendre simple et intuitif l’interface graphique de notre programme. Les informations généralement recherchées par les utilisateurs sont concentrées au haut de la fenêtre, tandis que les autres sont positionnées de manière à respecter les contraintes du client.

Une fenêtre pour la grille horaire encadrée par une couleur vivante et représentative de l’état du travail indiquera à l’utilisateur s’il y a un problème ou si tout va bien, un champ au bas communiquera avec précision les différentes erreurs au besoin ou encore des messages de confirmation et une barre de menu semblable à toutes les barres représentées dans les différents logiciels connus de tous. Voyez maintenant les différents croquis :

## Fenêtre principale



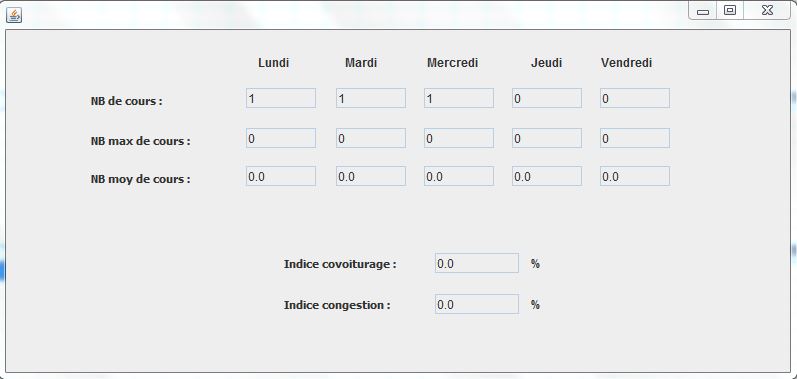
## La fenêtre principale est essentiellement composée des parties suivantes :

## Une barre de menu au haut de l’écran, permettant l’accès à des fonctionnalités triviales

* Une grille horaire qui suivra la même logique, c'est-à-dire simple et intuitive.
* Une liste des activités non attribuée (située à droite)
* Un champ pour l’indication d’erreurs
* Un bouton planification automatique permettant de faire l’essai de la fonctionnalité de planification automatique de l’application
* Un bouton note permettant à l’utilisateur de prendre des notes
* Un bouton statistique dont le contenu est détaillé ici-bas.

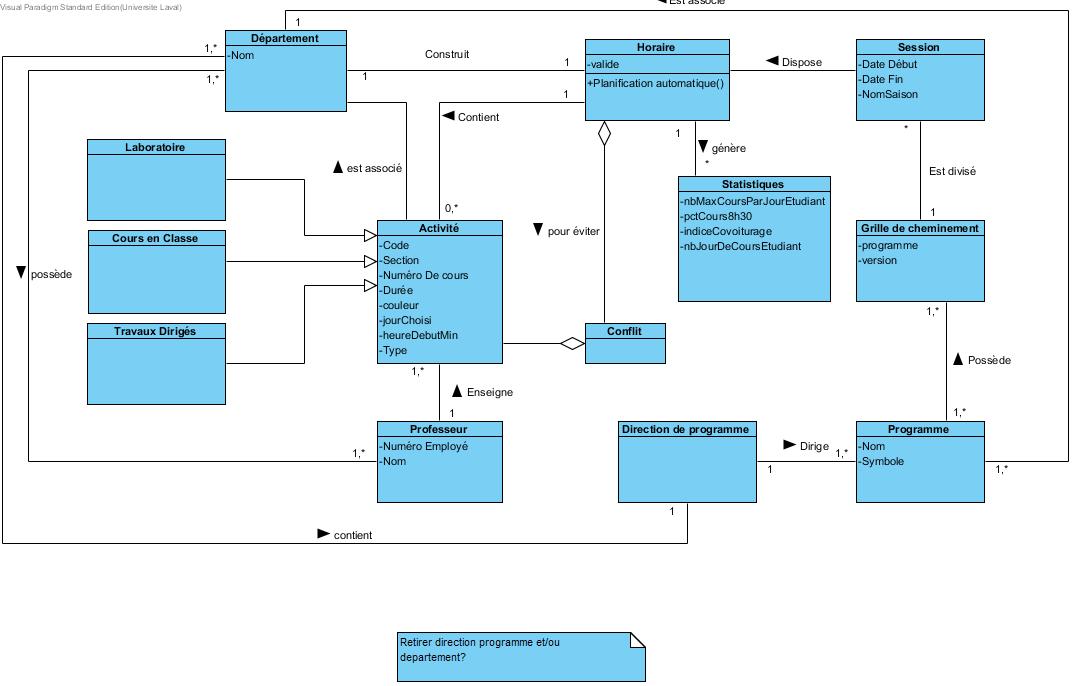
## Fenêtre statistiques

La liste des statistiques est automatiquement affichée pour tous les jours de la semaine.



# Modèle du domaine

Cette section consiste à vous présenter une perspective générale du fonctionnement de *PlanIFTicateur*. Comme tout bon modèle d’affaires, le schéma ci-dessous contient l’ensemble des informations nécessaires à la compréhension de notre projet, voire notre logiciel. Suite au visionnement du dît schéma, nous pourrons constater l’ampleur du travail accomplit.



***Détails du diagramme* :**

Pour le diagramme de modèle de domaine, plusieurs choix ont été faits afin d’obtenir le diagramme ci-dessus. Tout d'abord, le département est présent puisque c'est ce dernier qui utilisera l'application (ou quelqu'un de mandaté par la direction du département). Ensuite, la direction de programme est directement liée aux programmes puisque c'est la direction qui dirige ces programmes et qui construit les grilles de cheminement.

Ces grilles sont alors divisées en sessions qui ont tous un horaire à respecter selon certains critères, c'est-à-dire d'éviter les conflits d'horaires entre les cours d’une même session (facteurs fournis par les grilles de cheminement). Dans notre diagramme, l’horaire peut être vu de deux façons :

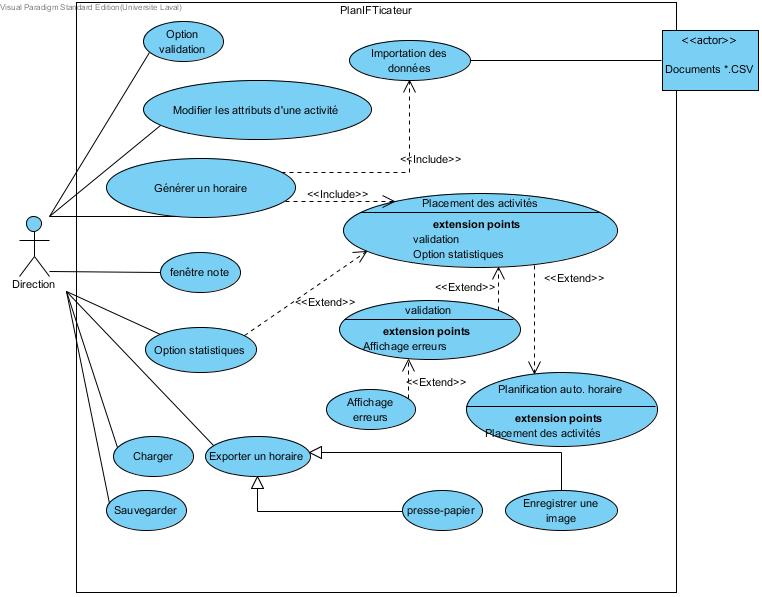
-Soit comme étant un horaire d'étudiant qui contient normalement cinq cours avec, possiblement des laboratoires;

-Soit comme étant un horaire de tous les cours qui sont donnés dans un département. D’ailleurs, c’est de cette façon que la direction planifie les horaires des sessions qui permettra ensuite d’obtenir celle des étudiants.

Par la suite, il y a les activités, qui sont réunies dans une superclasse pour ce diagramme et qui se divise en trois sous-classes : les laboratoires, les cours en classe et les travaux dirigés. Ces sous-classes correspondent aux différents types d'activités qui seront mises à l'horaire. Finalement, il y a les étudiants qui font partie du modèle puisque l'horaire construit concerne directement les étudiants. D'ailleurs, les statistiques générées par l'application sont basés sur ces étudiants.

# Modèle des cas d’utilisation

Maintenant que nous avons une bonne vue d’ensemble, il sera plus simple de comprendre la section ci-présente. Le modèle qui suivra représente le système, *PlanIFTicateur*, abritant tous ses cas d’utilisation. Il y aura également une représentation sommaire des acteurs du système. Suite à ce modèle, suivra l’architecture logique du logiciel.



# Architecture logique

# Diagrammes de classes de conception

Le diagramme qui suit représente le corps des classes utilisées par notre application. Par rapport précédent, nous avons ajouté plusieurs méthodes et attributs dans les différentes classes, voir toutes. Certains attributs ont ajoutés suite à plusieurs constations lors de l’implémentation du code.

Nous avons aussi ajouté une classe **ImageExporter** qui est utilisé pourexporter les images de l’horaire dans un format prédéfini. Le contrôleur, nommé ici **HoraireController**, est la seule classe qui échange avec l’interface.

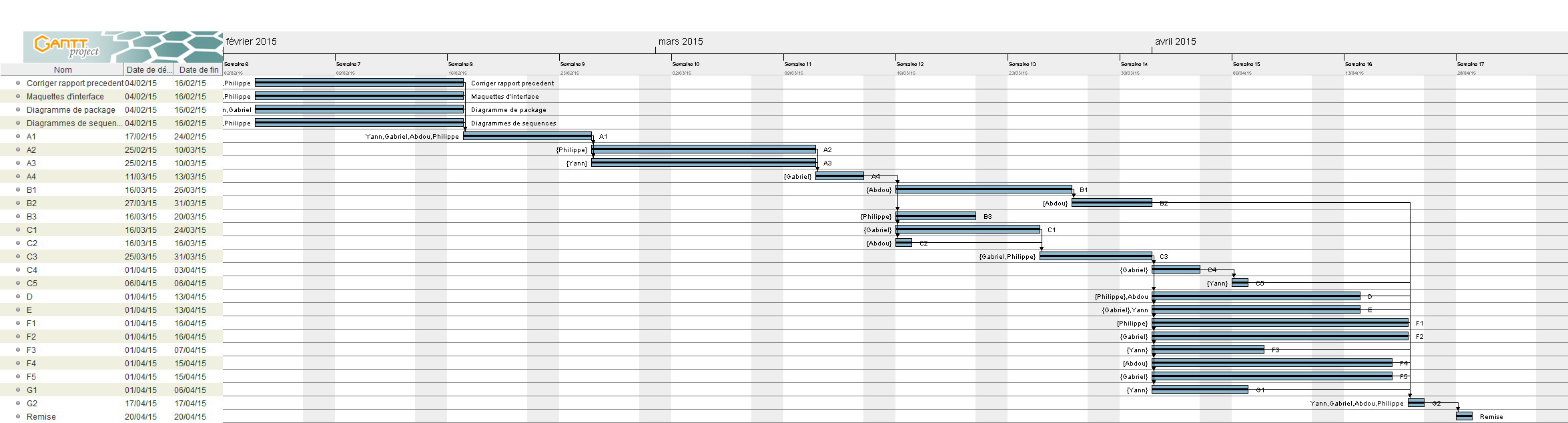
Tout élément de l’interface qui souhaite accéder à une fonction ou un événement doit passer par ce contrôleur qui s’assurera, par la suite, de faire la relation vers le code approprié. Le même phénomène se produit à l’inverse, c’est-à-dire lorsque les éléments du domaine, qui contient le code fonctionnel, désirent interagir avec l’interface utilisateur.

# Diagrammes de séquences

# Explications du design

# Gestion de projet

***Voir diagrammeDeGant.gan .***



# Conclusion