**CRAMILLET Bastien (GI 02) GAVIGNET Adrien (GI 02) MALTIS Jérémy (GI 02)**

Printemps 2010

**Rapport de projet LO45**

**1er sujet : Gestion d’un emploi du temps**

**LO45 : Programmation objet avancée**

**Responsable de l'UV : Claude RENAUD**

Sommaire

[Introduction 3](#_Toc264481688)

[Description du sujet 4](#_Toc264481689)

[1. Sujet 4](#_Toc264481690)

[2. Cahier des charges 4](#_Toc264481691)

[Choix Technologiques 6](#_Toc264481692)

[1. SQLite 6](#_Toc264481693)

[2. Qt 6](#_Toc264481694)

[3. CSS 6](#_Toc264481695)

[Interface Graphique 7](#_Toc264481696)

[Fonctionnement de l’application 8](#_Toc264481697)

[Difficultés rencontrées 9](#_Toc264481698)

[Conclusion 10](#_Toc264481699)

Introduction

De nos jours, l’informatique tient une place de plus en plus importante dans nos vies, que cela soit au niveau privé, ou même dans notre travail. Nous passons de plus en plus de temps devant nos ordinateurs, et ils ont tendance à remplacer le papier. Même nos habitudes les plus tenaces telles que les prises de note manuelles ou les agendas papiers disparaissent au profit de l’informatique : nous tapons désormais tout à l’ordinateur, et les palms, les téléphones portables … remplacent les agendas papiers.

Notre projet, qui consistait à développer un agenda s’inscrit donc parfaitement dans cette tendance. L’apprentissage des techniques nécessaires à la réalisation de ce projet était non seulement intéressant mais aussi utiles dans le cadre de notre formation d’ingénieurs informatiques dans le sens où l’essor grandissant des programmes informatiques de ce genre nous amènera à réutiliser les techniques.

Le but était de créer un programme permettant la gestion d’un emploi du temps. Aucune contrainte d’ergonomie n’était imposée, et nous aurions pu créer ce programme sans utiliser d’interface graphique. Cependant, le but d’un agenda est de prendre connaissance rapidement de son planning, et de pouvoir y insérer des rendez vous. Nous avons donc jugé nécessaire la création d’une interface graphique, dans le but que le programme soit plus pratique, mais aussi plus agréable à utiliser. Pour ce faire, nous avons ajouté des couleurs dans le planning, pour que l’utilisateur distingue mieux ses rendez vous.

Nous allons donc expliquer la façon dont nous avons procédé pour réaliser ce projet : le respect du cahier des charges, la mise en place d’une interface, et les différentes fonctionnalités proposées.

Description du sujet

## Sujet

Le projet se devait d’être conçu en utilisant le langage C++ afin de mettre en application les différents outils vus au cours de l’UV LO45. L’utilisation de la STL et des bitset était donc obligatoire à la réalisation du programme.

Comme l’indique le titre du projet, notre but était de développer une application permettant de gérer un agenda, c'est-à-dire : organiser une réunion, y inviter des personnes, visualiser son emploi du temps, le tout suivant certains critères.

Cependant le sujet possédait certaines bornes, un cahier des charges a été mis en place avant de démarrer le projet.

## Cahier des charges

Les principales fonctions à intégrer sont les suivantes :

- Gérer les horaires à l’heure, la demi-heure, le quart d’heure ou toutes autres divisions

- Gérer des salles de capacités différentes et contenant des équipements

- Gérer les dirigeants de réunion

- Gérer les participants aux réunions

- Le planning peut se faire à la journée, la semaine, au mois avec des réservations ponctuelles ou des annulations

Des fonctions annexes sont indispensables au bon fonctionnement de l’application :

- Recherche d’une salle adaptée libre sur un créneau

- Recherche d’un créneau disponible pour un groupe de personnes

- Signaler les impossibilités d’un groupe de personnes sur un créneau

- Annulation d’une réunion ou modification de durée d’une réunion par le responsable

- Calculer le taux d’occupation à la semaine d’une personne ou d’une salle

En plus de ces fonctions à intégrer, certaines règles sont à respecter :

- Une seule réunion par salle à une heure précise

- La salle doit satisfaire la demande : salle précise ou offrant les moyens

- Une personne ne peut pas être présente dans deux salles différentes au même horaire

- Un délai d’une demi heure est nécessaire entre deux réunions pour qu’un

participant soit prêt à participer à la suivante

- Les durées de réunion peuvent être "quelconques"

- La journée est gérée en continue

- La pause déjeuner est nécessaire entre deux réunions.

- Il peut y avoir des réunions régulières et des réunions ponctuelles

- On ne peut changer une réunion moins d’une semaine à l’avance

- Le planning se fait sur un mois

- Des demandes ponctuelles peuvent arriver à tout moment

- Une salle affectée doit être compatible avec le nombre de participants à une

réunion

- Lorsqu’un dirigeant fait des demandes de salle il fournit ses disponibilités horaires,

les noms des participants prévus, sa demande de salle (nommément ou

équipement)

Pour finir ce cahier des charges, une contrainte de développement était imposée :

- Gérer les créneaux occupés à l’aide des champs de bits (bitset) pour accélérer les

comparaisons

En ce qui concerne la partie graphique, aucune indications n’était fournit. Par défaut, une application en console suffit donc.

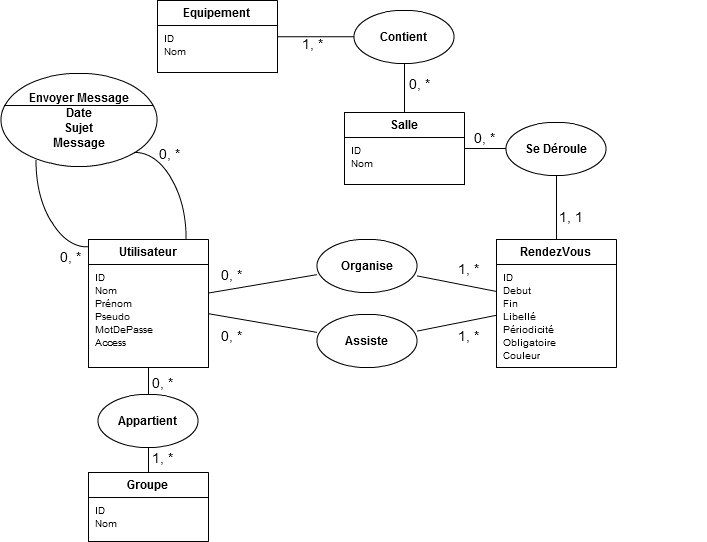
Choix Technologiques

## SQLite

Afin de stocker les différentes informations contenues dans l’application (c'est-à-dire les personnes inscrites, les groupes créés, les réunions organisées…) une base de données était indispensable. Celle-ci devait être facilement exportable dans l’application afin de pouvoir effectuer des requêtes SQL directement en C++.

Un logiciel répondait à toutes ces attentes, il s’agit de SQLite. SQLite est une bibliothèque écrite en C qui propose un moteur de base de données relationnelle accessible par le langage SQL. Contrairement aux serveurs de bases de données traditionnels, comme MySQL ou PostgreSQL, sa particularité est de ne pas reproduire le schéma habituel client-serveur mais d'être directement intégrée aux programmes, ce qui convient parfaitement dans notre cas.

Afin de gérer au mieux les différentes fonctionnalités de l’agenda, nous nous devions de mettre en place différentes tables intégrées à notre base de données (qui sera utilisé par l’application grâce à SQLite). Nous avons donc mis en place un Modèle Conceptuel des Données semblables à celui-ci :



## Qt

Etant parti du principe qu’un agenda ne pouvait pas être développé en console, nous nous sommes imposé dès le début de créer une interface graphique. Lors de notre cursus universitaire, nous avions appris les bases de Qt. Qt est un framework orienté objet et développé par Nokia. Elle offre des composants d'interface graphique (widgets), d'accès aux données, de connexions réseaux… L’intérêt est aussi de pouvoir développer une application portable sous Windows, Linux et Mac Os puisque Qt est une bibliothèque multiplateforme. C’est pourquoi nous avons choisi ce framework, qui comble tout à fait nos attentes.

De plus, Qt intègre les QBitArray. Ceux-ci fonctionnent exactement comme les bitsets vu en cours, mais possèdent un nom différent puisqu’ils appartiennent à Qt. Grâce à ces QBitArray, le cahier des charges est respecté sur ce point.

Enfin Qt permet la mise en place d’une interface « esthétique » et ergonomique est des plus simple grâce aux widgets tels QComboBox, QMenu, QTableView par exemple. Ceci nous a permis de concevoir une application pratique à utiliser. De plus, l’utilisation du CSS est autorisée dans Qt : ceci a permis d’ajouter de la couleur et des formes à l’agenda…

## CSS

Le CSS (Cascading Style Sheets) est un langage informatique qui sert à décrire la présentation des documents HTML et XML…habituellement. Dans notre cas, comme expliqué dans le paragraphe précédent, nous l’avons utilisé afin d’appliquer un style à notre application Qt, qui est totalement compatible avec le CSS.

Ceci se fait de façon très simple : on applique un style à un type de widget, par exemple :

QTableWidget

{

selection-background-color: #cfe7fe;

selection-color : black;

background-color : #E1F0FF;

}

## SVN

Etant un trinôme à travailler sur le projet, nous nous devions de mettre en place un système de gestion de versions, afin d’être un maximum efficace. Pour cela, nous avons décidé d’utilisé SVN, qui est certainement le plus connus des systèmes. Très simple d’utilisation, il nous a permis de ne pas nous perdre parmi toutes les différentes versions du projet, et de pouvoir travailler chacun de son côté les jours où nous ne pouvions pas nous voir.

Interface Graphique

Fonctionnement de l’application

Difficultés rencontrées

Conclusion

En conclusion, on peut dire que ce projet répond au cahier des charges à tous les niveaux. Les fonctionnalités suivantes y sont présentes : créer des réunions en y invitant des personnes ou un groupe de personnes, de chercher un créneau qui correspond à tous les invités d’une réunion, de consulter son emploi du temps, de modifier les réunions que l’on a organisé… De plus, nous avons ajouté une fonction bien utile permettant d’envoyer un message électronique aux différents utilisateurs de l’application. Les principales règles sont toutes respectées, de ce fait, l’application est utilisable sans problème, ne possédant pas de bugs apparents.

De plus, une interface graphique a été mise en place. Celle-ci permet un confort d’utilisation du programme et donne vraiment envie de se servir de l’application, contrairement à une application en console par exemple. Les couleurs ont été choisies de manière à ce qu’elles n’agressent pas l’œil humain, et offrent une lisibilité parfaite.

Ce projet fut un très bon exercice de mise en application des connaissances théoriques acquises au cours des cours, des travaux dirigés et des travaux pratiques. Grâce à celui-ci, nous maîtrisons désormais les grands principes du C++ et de Qt principalement.

Avec un temps de développement plus grand, quelques améliorations auraient pu être ajoutées à cette application, telles que la génération d’un document PDF de l’emploi du temps, qui serait ainsi imprimable ; une interface graphique interactive (déplacement de réunion, création de nouvelles…) ; une gestion des statistiques pour chaque utilisateur et chaque salle ; ou encore pouvoir envoyer de vrais mails sur les boites personnelles des utilisateurs.