Rapport du projet informatique

Florentin Brisebard, Bemoît Guillot, Charles d'Andigné

Table des matières

1 Introduction

Notre projet consistait à développer une application à destination d'une compagie aérienne. Elle doit permettre à cette dernière de gérer les réservations en optimisant les coûts par le choix judicieux des avions affrétés.

Au regard de notre rapport d'analyse, notre diagramme de classe a beaucoup évolué. Nous avons pu essayer d'aller un peu plus loin que le sujet de départ malgrès la difficulté que cela a engendré notament pour pouvoir proposer en même temps l'interface de réservation pour les clients.

2 Organisation du travail

L'organisation et la répartition des tâches se sont faites de manière naturelle et équitable. Chacun d'entre nous a remplis sa mission et même si nous n'avons pas tout le temps travaillés simultannément de notre côté. En effet notre avancée pouvais dépendre d'une tâche en cour de réalisation d'un de nos camarade. Ce fut surtout le cas pour l'interface. Celle-ci ne pouvait avancer sans l'implémentation des classes or cette implémentation nécéssitait une maîtrise complète des classes ce qui empêchais à une même personne de faire tout elle-même. Néanmoins il était possible travailler en simultané notemment lors de l'implémentation de notre Main .

3 Dévellopement

3.1 Développement général

Dans un premier temps, nous avons dû faire de nombreux choix quant aux limites de notre programme, en effet il était impensable à notre niveau d'avoir une application en temps réel,

ou bien de gérer un trop grand nombre de paramètre sur les détails techniques des vols et des réservations (gestion des trajets, système d'escale ou encore paiement en ligne).

Néanmoins nous voulions dépasser les limites du sujet initiale en créant un interface voyageur qui générait les réservations de la compagnie, créant les vols qui conviennent et de pouvoir gérer notre compagnie en instanciant nos objets.

C'est pourquoi nous avons décidé réellement s'occuper de la gestion de nos objets ; un vol n'est pas possible si l'aéroport de départ ne contient pas d'avion (par exemple), ou encore un avion ne peut pas effectuer 2 vols en même temps (comprendre le même jour). De plus, nos avions sont très variés et certains ne sont pas adaptés à certains vols, donc nous avons dû implémenter des fonctions vérifiant les capacités d'un avion a réaliser un vol.

La première difficulté réelle a été la gestion du temps : nous nous sommes dans un premier temps lancé dans une gestion au jour le jour avec un système qui permettait de passer au jour suivant. Le problème étant que nous avons besoin de prévoir de nombreux paramètres a l'avance, comme la disponibilité de nos avions dans les aéroports concernés. Il était donc nécessaire de pouvoir accéder aux vols et aux informations des aéroports à n'importe quel moment. Nous avons donc opté pour une gestion avec des dictionnaires (HashMap) comprenant les dates en clés et les différentes listes d'objets en valeurs.

La deuxième grande difficulté a été de gérer les changements induits par un vol d'avion, en effet, il faut prendre en compte non seulement que l'avion du vol quitte son aéroport à la date donné, mais aussi qu'il ne sera plus présent pour les jours suivants, mais qu'il pourra être utilisé ensuite par l'aéroport d'arrivé pour un vol futur, même si celui-ci a été programmé précédemment !

Nous avons fais le choix de ne pas programmer les vols à l'avance, mais plutôt de laisser l'utilisateur engendrer la création d'un vol, si aucun des vols déjà programmé ne répondait à ses attentes. Ainsi l'interface ne permet pas à la compagnie de programmer un vol, mais cela est fait automatiquement lorsqu'un voyageur veut faire une réservation.

Notre Main est de taille très importante, en effet nous avons été obligé d'instancier l'intégralité de nos objets à la main (aéroports, avions, équipages) ainsi que d'effectuer la répartition de ces derniers dans les listes contenues dans les dictionnaires. Il aurait été beaucoup plus judicieux d'utiliser une base de données pour réaliser cela, mais nous aurions perdu beaucoup trop de temps à revenir en arrière.

Enfin, l'interface ne contient pas toutes les fonctions que nous avons codées, en effet le lien compagnie/réservation n'a pas été abouti et notre interface paraît quelque peu vide par rapport à l'idée initiale que nous en avions. Mais il est possible de passer par la console et le main pour visualiser d'autres possibilités telles que le calcul des bénéfices d'un vol, la modification d'un vol pour une réservation existante ou encore l'annulation et le remboursement complet de celui-ci. Ceci vous sera présenté brièvement dans notre présentation.

3.2 Cas particulier de l'interface

Rapport Interface :

Notre interface graphique est réalisé en utilisant les fonctionnalités de base de java en matière d'interface.

Ainsi, nous avons utilisé une fenetre (JFrame) principale avec deux onglets dessus. L'un étant l'interface Compagnie-Application, l'autre l'interface Voyageur-Application.

Une fenetre principale avec un bouton et le logo de la compagnie sert d'ouverture de l'application.

Interface Compagnie :

Elle se divise en 3 onglets :

- Onglet aéroport :

L'utilisateur de la compagnie peut via un bouton (utilisation du Jbutton) ouvrir un menu défilant affichant la totalité des aéroports utilisés par la compagnie. Un simple clic sur l'aéroport (via un ActionListener) va ouvrir les dates des vols à venir dans une nouvelle fenetre avec menu défilant. Puis, un clic sur la date séléctionée ouvrira une nouvelle fenetre affichant les avions presents à cette date.

-Onglet avion :

L'utilisateur de la compagnie peut via un premier bouton afficher les différents modèles d'avion utlisés par la compagnie. Un clic sur un modèle ouvrira une nouvelle fenetre avec un gif affichant l'appareil correspondant.

Un second bouton permet à l'utilisateur d'afficher tous le avions dans un menu déroulant (JlistScroller) et, par un clic sur l'appareil, afficher toutes ses caractéristiques (nom, capacité, rayon d'action...etc)

- Onglet vol :

L'utilisateur peut via un bouton afficher les dates des vols à venir dans un menu déroulant(JlistScroller) et par un clic sur la date, voir les vols à cette date dans une nouvelle fenetre (Jframe).

Cette interface compagnie nous semble complète, mis à part le fait que nous n'avons pas eu le temps de faire un lien entre la compagnie et les voyageurs dans l'interface (Affichage de la liste des voyageurs sur un vol par exemple)

Interface Voyageur :

Elle est composéede 2 boutons.

Un bouton identification qui ouvre une fenetre avec des zones de saisie de texte (JtextField) ou doivent etre ecrits les noms/prénoms/date de naissance/ poids des bagages du voyageur s'enregistrant. Ces données sont stockées en string et en double pour le poids afin de génerer le voyageur dans l'application et de pouvoir l'ajouter à un ou des vols ensuite.

Le second bouton fait apparaitre un menu déroulant qui doit afficher les voyageurs enregistrés afin que le voyageur concerné selectionne son profil. Mais faute de temps, cela n'a pu etre realisé.

Pour conclure, l'interface ne peut permettre à un voyageur de s'enregistrer sur un vol. Cela aurait été fait facilement avec du temps parce que toutes les méthodes nécessaires sont opérationnelles.

L'interface pour la compagnie est efficace et permet bien à la compagnie de gérer ses aéroports, ses avions, et ses vols ! Mais de même, nous aurions pu facilement ajouter d'autres méthodes fonctionnant dans la console afin de rendre notre interface plus ludique et efficace.

4 Conclusion

Nous avons beaucoup apris cela va sans dire sur la programation en elle même mais surtout sur l'organisation d'un vrais projet de groupe. En terme d'organisation, d'interaction. Nous avons appris à prendre notre place au sein du projet. A composer avec les caractères de chacun.