Université de Bordeaux

Sciences et Technologies

Master de Bioinformatique - Parcours Biologie Computationnelle

Année 2019-2020

Projet de Programmation Orientée Objet

MinimesEnUnClic

Abdelghani Neuhaus

Sommaire

Introduction

Le Port des Minimes est le port de plaisance de la Rochelle. Il a été crée en 1972 et compose actuellement l'une des trois parties du port. Il s'agit du premier port de plaisance d'Europe et un des plus grand d'Europe. En effet, avec 4500 places à flot dont 440 pour les visiteurs, c'est presque 10 millions de visiteurs qui s'y rendent chaque année. Divers services y sont proposés : gestion des déchets, mise à l'eau, levage mais aussi location de vélo, connexion Wi-Fi gratuite, etc...

Figure 1 : Photo du Port des Minimes

Ce jeu ancien nous vient du plus profond des montagnes de l'Asie centrale. Il servait d'épreuve d'initiation, non pas pour pas devenir un valeureux guerrier, mais pour révéler qui serait le plus fin stratège du village.

Dans ce projet, il nous ait proposé de simuler la gestion du port et d'informatiser celle-ci au travers du projet **LesMinimesEnUnClic**.

Partie 1

Analyse du projet

Dans cette partie, nous discuterons et exposeront nos idées et pensées après la lecture du sujet.

1.1 Principes et utilisation du programme

Le programme doit permettre la gestion du port. Le port peut accueillir deux types d'usagers :

— **Abonnés** : ils disposent d'un bateau à l'année (c'est-à-dire une place permanente) et ils règlent chaque mois une cotisation ;

— **Client de passage** : ils viennent pour seulement quelques jours et règlent une taxe par jour de présence dans le port

Les clients qui viennent dans le port peuvent posséder trois types de voiliers :

— **Non-habitable** : bateau d'une taille inférieure à 10 mètres de long. Il ne possède pas de cabine et n'utilise pas les services du port (eau et électricité) ;

— **Type 1** : bateau d'une taille comprise entre 10 et 25 mètres. Il possède des cabines et utilise les services du port ;

— **Type 2** : bateau d'une taille supérieure à 25 mètres. Comme le voilier de type 1, il possède des cabines et utilise les services du port, mais, en plus, il peut se placer uniquement dans des places spécifiques du port.

Lors de la saisie d'un client (ajout ou retrait), l'utilisateur doit donc pouvoir aisément choisir chaque spécificité (type de client, type de voilier, etc...) qui caractérise l'usager. Il doit aussi être possible de voir diverses informations à tout moment comme par exemple la liste des places disponibles pour un certain type de voilier, les informations concernant un client, etc...

1.2 Contraintes à la réalisation

La lecture du sujet a soulevé plusieurs contraintes à gérer et à prendre en compte avant de commencer à concevoir le programme :

- Tout d'abord, il faut, lors de la création du port, générer quatre types de places (pour les trois types de voiliers et des corps morts) ;
- Ensuite, il faut gérer l'usager : le type de formule, la durée du séjour, la facture et son départ ;
- Aussi, lorsque le client quitte le port, il faut penser à éditer sa facture et à libérer la place occupée par le voilier, tout en gardant en mémoire les informations sur le client (montant de la facture payée notamment) ;
- Également, une place ne peut être attribuée qu'une seule fois et si toutes les places sont occupées, il faut indiquer à l'utilisateur que c'est le cas ;
- Un autre problème soulevé est la gestion des informations sauvegardées : chaque client doit avoir un numéro de dossier unique pour pouvoir le retrouver facilement dans la base de données. Il est primordial que l'utilisateur puisse accéder aisément aux données enregistrées.

Ces éléments ont donc été pris en compte avant de réaliser le programme.

1.3 Affichage et lisibilité pour l'utilisateur

Le programme sera affiché sur la console d'un terminal via la commande ***"./MinimesEnUnClic"***, en se trouvant dans le dossier contenant le fichier. Un menu sera affiché lors du lancement du programme, permettant à l'utilisateur de choisir ce qu'il veut faire. Après chaque action (saisie, consultation de données), le menu réapparaît afin de faciliter l'utilisation.

Partie 2

Conception du programme

Dans cette partie, nous discuterons de comment nous avons agencé le programme et les différents éléments le constituant.

2.1 Initialisation du programme et accueil de l'utilisateur

Au moment où le programme est ouvert, comme dit précédemment, un menu accueille l'utilisateur, lui proposant cinq choix : saisie d'un client, consultation des données d'un client, saisie du départ d'un client, consultation des places disponibles et sauvegarde des données saisies.

Nous avons pensé au fait que l'utilisateur puisse avoir des données qu'il voudrait charger (d'une précédente utilisation du programme). Ainsi, lors du démarrage, deux fichiers (un stockant les usagers et un stockant les voiliers) sont chargés et placés dans des conteneurs. L'utilisateur peut donc travailler avec des données déjà existantes (ce qui est utile pour consulter les factures des clients partis par exemple).

2.2 Création des éléments du port

a) Port

La création du port est la plus importante car il faut prendre en compte plusieurs éléments. Tout d'abord, discutons des places disponibles. Elles sont au nombre de cent et ont été réparties en quatre types. Des places sont disponibles pour :

— Les voiliers non-habitables ;

— Les voiliers de type 1 ;

— Les voiliers de type 2 ;

— Des corps morts (au nombre de 10).

Nous avons pensé que le port s'occuperait de sa gestion. Ainsi, les places disponibles sont visibles au sein du port. Un autre élément concerne la gestion des usagers : chaque usager est stocké dans le port via un numéro de dossier qui commence à 0 et qui s'incrémente de 1 à chaque ajout d'un nouveau client. L'ajout de voilier dans la base de données se fait au même moment que le client, mais dans une

autre base de données. Les saisies sont néanmoins synchronisés (les numéros seront les mêmes).

Le port a aussi la possibilité de retirer un bateau lors du départ du client : la place est libérée mais les données sont conservées pour pouvoir les consulter même après le départ (notamment pour accéder à la facture et aux informations sur le voilier). Le service de gestion du port est donc l'élément qui s'occupe de tout ceci et la classe associée s'appelle **GestionPort** *(cf.partie 3)*.

b) Usagers du port

Les usagers seront saisies dans les éléments de gestion du port. Un usager a comme caractéristiques :

— Un nom et un prénom ;

— Un voilier (sous forme de pointeur, *cf.plus bas*) ;

— Une formule (abonné ou passager) et une facture ;

— Un état de présence sur le port (le client est-il présent ou est-il parti ?)

L'usager possède aussi un numéro de dossier, mais qui n'est pas dans ses attributs : il n'a pas à connaître son numéro de dossier. En effet, à partir du moment où il a accès son voilier, il peut retrouver la place de ce dernier ; il connaît aussi sa facture et sait s'il est sur le port ou non.

c) Voiliers

Le port peut accueillir trois types de voiliers cités plus haut. Ces voiliers peuvent néanmoins prendre deux places : celle qui leurs sont réservées et les corps morts. Un voilier ne pouvant connaître son usager, seul l'usager et le port peuvent accéder à ces informations. Les voiliers possèdent des attributs qui sont les suivants :

— Un nom ;

— Une place ;

— Une longueur et donc un type ;

— La présence de cabine(s) et un besoin à l'accès de service ou non ;

Étant donné la présence de trois types de voiliers, il y'a quatre classes de voiliers (*cf. diagramme UML plus bas*)

2.3 Éléments supplémentaires

La gestion des données sur les clients et les voiliers est effectuée par le port : seul lui peut sauvegarder, charger et avoir accès aux données (l'usager a accès uniquement à ses données et celle du voilier).

Partie 3

Réalisation

Nous discuterons dans cette partie de comment nous avons codé et réalisé le programme. Le diagramme UML ci-dessous indique les attributs, les méthodes et les liens des classes :

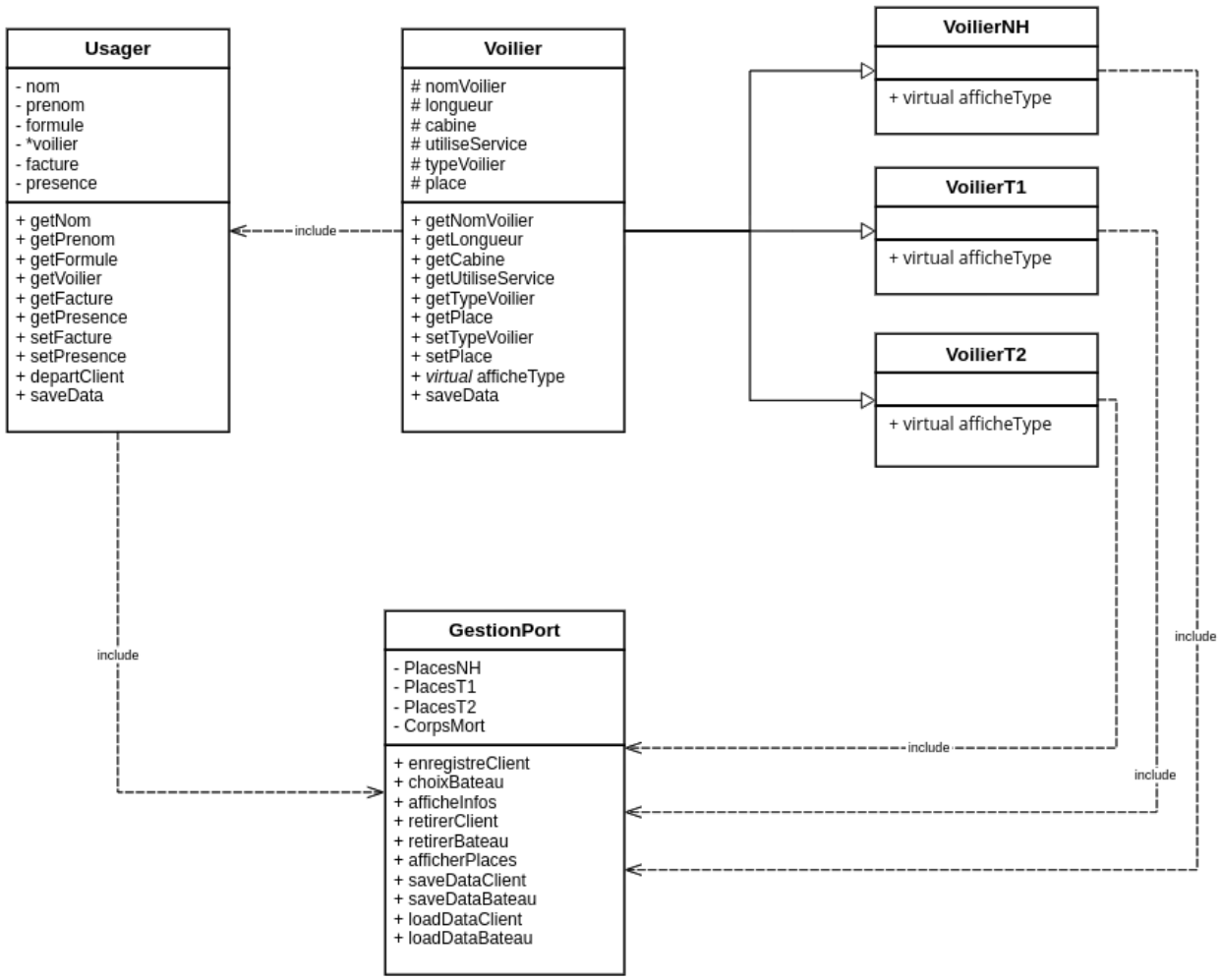

Figure 2 : Diagramme UML des classes du projet

3.1 Initialisation

L'initialisation du programme lors de son lancement consiste en la création de :

— Deux conteneurs de type vector pour stocker les usagers et les voiliers (celui stockants des voiliers stockent des pointeurs) ;

— Une instance de la classe **GestionPort** nommée *GestionLaRochelle* ;

— Places qui sont générées avec *GestionLaRochelle*. Ces places étant visibles partout - (un touriste peut voir les places disponibles dans le port en se promenant) et par le service de gestion du port - et étant immuables (les numéros ne changeront pas), elles ont été défini comme étant des attributs *static* et les places sont contenues dans un vector stockant des *int*. On a donc quatre vectors crées lors de la création de l'instance *GestionLaRochelle* (*static vector<int>*) ;

— Le chargement des données enregistrées est automatiquement effectué après la création des conteneurs nécessaires. Les données des usagers et voiliers sont lus et leurs contenus stockés.

3.2 Stockage des données des usagers et des voiliers

Les données des clients et des voiliers sont stockées dans deux conteneurs vector de la bibliothèque STL (le premier étant *Clients* et le second *Voiliers*). Les numéros de dossier des clients correspondent à l'indice où ils sont stockées (par exemple, le premier client aura le numéro de dossier 0 et les informations sur son voilier seront stockées à l'indice 0 du vector *Voiliers*).

3.3 Boucle principale

La boucle principale consiste en l'appelle des méthodes de la classe **GestionPort** via l'instance *GestionLaRochelle*. Le menu est affiché à chaque fois qu'une saisie est effectuée (qu'elle soit valide ou non). Il y'a cinq choix possibles uniquement et un message d'erreur apparaît si l'utilisateur entre un mauvais nombre. L'utilisateur peut quitter le programme en tapant 0. Il devra avoir sauvegarder ses données au préalable.

3.4 Gestion de la saisie d'un nouveau client

Lors de la saisie d'un nouveau client, il est d'abord crée le voilier. L'utilisateur renseigne les caractéristiques le nom et la longueur du voilier, et selon la longueur, il est automatiquement crée une instance d'une des trois classes (*par exemple, si la longueur est supérieure à 25, il sera crée une instance de la classe VoilierT2*). Le constructeur est remplie avec les valeurs des attributs saisies précédemment. Ensuite, selon le type de voilier crée, une place est attribuée au voilier. Une fois la place attribuée, elle est retirée du static vector correspondant.

Ensuite, on appelle la méthode de création de client de *GestionLaRochelle*. On saisie les informations de l'usager (nom, prénom, abonnement ou pas). Le voilier associé est récupéré dans la liste des voiliers : c'est pour cela que le voilier est crée en premier. De plus, ce choix permet d'apporter

une touche de vraisemblance : on va d'abord entrer les informations sur le voilier et voir si les places sont disponibles plutôt que de saisir l'usager, puis le voilier et se rendre compte qu'il n'y a plus de places. Une fois que l'usager est saisi, selon son type de formule, sa facture est calculée et est renseignée.

3.5 Gestion de l'affichage des informations d'un client

L'affichage des informations d'un client est réalisée via le numéro de dossier (on accède donc à l'indice des conteneurs). Les deux conteneurs sont requis.Il est affiché les caractéristiques du client (nom, prénom, voilier, facture et place). Il y'a un affichage prévu pour un client étant parti.

3.6 Gestion du départ d'un client

Lors du départ d'un client, on utilise deux méthodes de la classe *GestionLaRochelle* qui sont *retirerBateau* et *retirerClient*. Ces méthodes vont respectivement changer l'état de la présence du client à *false* et mettre la place du bateau à une valeur par défaut de -1, et rendre l'ancienne place du bateau disponible.

3.7 Gestion de l'affichage des places disponibles

L'utilisateur peut à tout moment voir les places disponibles pour un type de voilier (ou corps mort). Un menu est affiché, indiquant quatre choix (pour les quatre types de places). Une fois que le choix est réalisé, on affiche succinctement tous les numéros contenus dans le *static* vector choisi.

3.8 Gestion de la sauvegarde des données

La sauvegarde des données appelle deux méthode de la classe *GestionLaRochelle* qui sont similaires dans leurs fonctionnements. Elles utilisent une méthode de sauvegarde appartenant à la classe Voilier et à la classe Usager. Par exemple, on va utiliser la méthode *saveData* du Voilier (dans *saveDataVoilier* appartenant à *GestionLaRochelle*) pour sauvegarder tous les voiliers.

3.9 Gestion du chargement des données

Pour faciliter la récupération des données par l'utilisateur, nous avons fait en sorte que les méthodes de chargement de données (une pour les clients et une pour les voiliers) soient appelées automatiquement et que l'utilisateur n'est pas à le faire.

3.10 Résultat après réalisation

Le résultat du programme est visible en annexe (pour le code), mais, concrètement, l'organigramme ci-dessous résume la réalisation de nos idées et correspond aux actions qu'un utilisateur effectue du début jusqu'à la fin de sa partie :

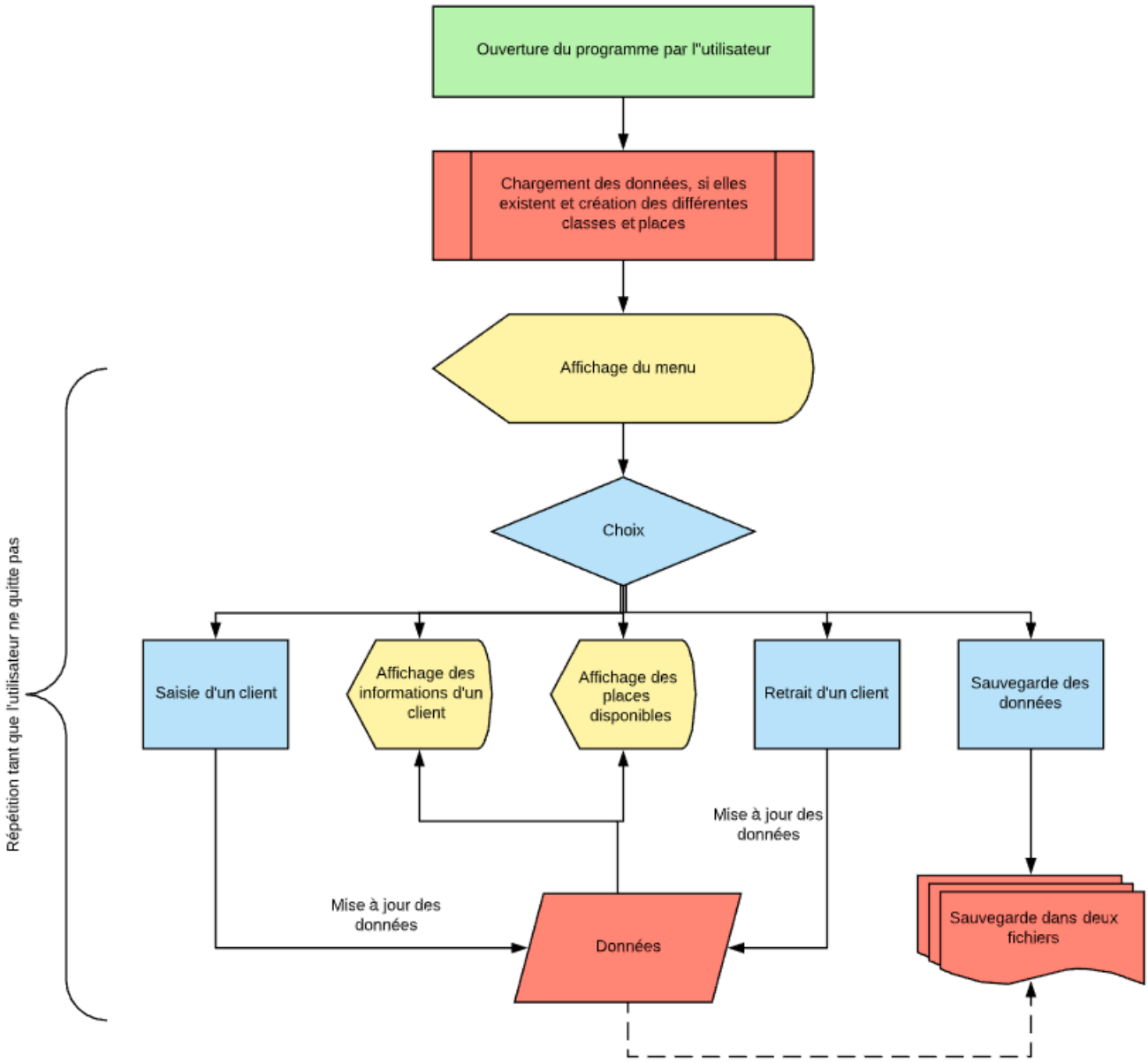

Figure 3 : Organigramme représentant l'exécution du programme

Conclusion

Pour résumer, il a été codé une version d'un programme des gestion pour le port de la Rochelle. Nous avons d'abord implémentées les classes Voilier, Usager et Port. Ensuite, nous avons crée les conteneurs qui stockent les données que l'utilisateur entrera et nous avons crée les places du port. La boucle principale est composée de cinq fonctions : Enfin, l'utilisateur n'a pas à s'occuper de la gestion du chargement : celle-ci est automatique à chaque lancement du programme.

Aussi, dans une version plus développée du programme, nous aurions pu ajouter :

— Un affichage plus poussé des informations sur les factures ;

— Une interface graphique avec des zones de saisies et des boutons pour les cinq choix du menu ;

— La possibilité de sauvegarder les données grâce à des onglets (comme dans un logiciel classique) ;

Annexes

Références webographiques

Figure 1 : https://www.portlarochelle.com/wp-content/uploads/2016/12/port-les-minimes-la-rochelle.jpg