Daix Théo

Dubrulle Allan

Verhoye Victor

Rapport de modélisation

Génie Logiciel

Projet de modélisation et d'implémentation

Simulateur pour un distributeur de billet de train

Faculté des Sciences

UMONS Université de Mons

Enseignants : Mr Mens et Mr Hauweele
Année Scolaire 2017-2018

Table des matières

Introduction :

Voici le rapport de modélisation du projet d'un simulateur pour un distributeur de billets de train, à destination des étudiants en deuxième année de bachelier en math-info. Son but est de montrer et d'expliquer les différents diagrammes servant à décrire le fonctionnement de la future application. Dans l'introduction, nous aimerions parler de certains choix de conception que nous avons fait :

- Lorsqu'une quelconque panne sera détectée, toutes les fonctionnalités qui sont impactées par ces pannes seront désactivés (cela sera fait grâce aux boutons concernés qui seront indisponible dans la fenêtre de simulation).

- De même, si des composants optionnels ne sont pas activés, ils ne seront pas utilisables. Tout ce qui les concerne sera donc indisponible.

Diagramme de cas d'utilisation :

Ce diagramme (voir figure 1) représente non pas un simulateur de distributeur, mais bien un distributeur. Nous avons pris cette décision car il est ainsi plus évident de comprendre les réelles interactions entre les utilisateurs et le système. Etant donné que ce projet est en réalité un simulateur, le technicien et le client sont bien sûr la même personne (dans la suite, j'appellerai cette même personne l'utilisateur). En effet, celle-ci pourra à la fois interagir avec le distributeur, mais pourra aussi gérer elle-même les pannes (par exemple, recharger en encre et en papier le distributeur quand il n'y en aura plus). A la suite de ce diagramme, vous pourrez comprendre exactement les cas d'utilisation à l'aide des descriptions qui sont données. Le système bancaire, lui, sera en réalité (dans la suite du projet) la combinaison entre la base de données et le gestionnaire de base de données (dans le diagramme de classe, vous pouvez le voir apparaitre à travers la classe GestionBaseDeDonnees).

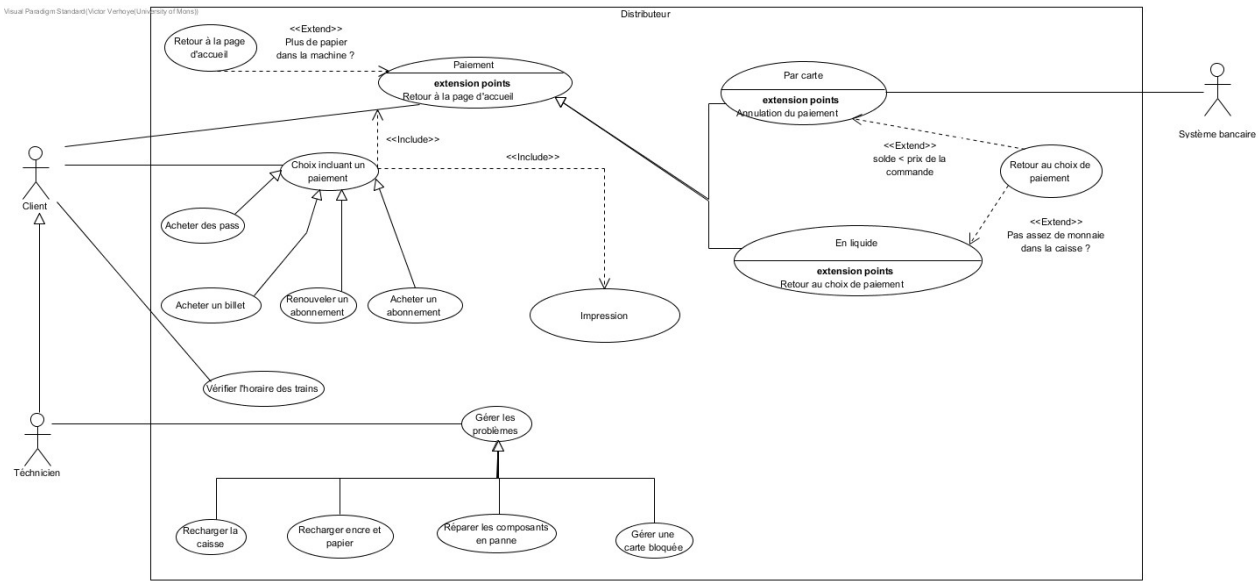

Figure 1 - Diagramme de cas d'utilisation d'un distributeur de billet de train

Description semi-formelle des cas d'utilisation :

Cas d'utilisation : Vérifier l'horaire des trains
Résumé : Client vérifie l'horaire d'un train.
Acteur : Client
Hypothèses : /
Préconditions : La connexion avec la base de données est opérationnelle.
Etapes du déroulement normal :

1. Ecran demande d'inscrire les informations relatives au trajet.
2. Client entre les informations.
3. Ecran affiche l'horaire en question.

Postconditions : Client est au courant de l'horaires des trains désirés.
Déroulements alternatifs : /

Cas d'utilisation : Choix incluant un paiement
Résumé : *Cela dépend du cas d'utilisation spécialisé.*
Acteur : Client
Hypothèses : LecteurCarte fonctionne ou la fente à pièces fonctionne. Imprimante fonctionne. Client a assez d'argent.
Préconditions : *Cela dépend du cas d'utilisation spécialisé.*
Etapes du déroulement normal :

1. *Début : cela dépend du cas d'utilisation spécialisé.*
2. Client confirme et est redirigé vers Paiement (voir description du cas d'utilisation).
3. Redirection vers Impression (voir description du cas d'utilisation).

Postconditions : *Cela dépend du cas d'utilisation spécialisé.*
Déroulements alternatifs : *Cela dépend du cas d'utilisation spécialisé.*
1a. S'il n'y a plus de papier ou d'encre, Ecran affiche un message d'erreur.
2a. Redirection vers le menu principal.

Cas d'utilisation : Acheter un billet
Résumé : Client achète un billet.
Acteur : Client
Hypothèses : LecteurCarte fonctionne ou la fente à pièces fonctionne. Imprimante fonctionne.
Préconditions : Client a assez d'argent.
Etapes du déroulement normal :

1. Ecran demande d'inscrire les informations relatives au pass.
2. Client entre les informations.
3. Ecran affiche un récapitulatif du billet.
4. *Fin : voir description du cas d'utilisation Choix incluant un paiement.*

Postconditions : Un billet est acheté.
Déroulements alternatifs :
3a. S'il n'y a plus de train le jour de la commande, Ecran affiche un message informant qu'il n'y a plus de train avant le lendemain et demande confirmation du billet pour le lendemain.
3b. Ecran affiche un récapitulatif de l'abonnement et Client annule.

Cas d'utilisation : Acheter un abonnement
Résumé : Client achète un abonnement.
Acteur : Client
Hypothèses : LecteurCarte fonctionne ou la fente à pièces fonctionne. Imprimante fonctionne.
Préconditions : Client a assez d'argent.
Etapes du déroulement normal :

1. Ecran demande d'inscrire les informations relatives à l'abonnement.
2. Client entre les informations.
3. Ecran affiche un récapitulatif de l'abonnement.
4. *Fin : voir description du cas d'utilisation Choix incluant un paiement.*

Postconditions : Un abonnement est acheté.
Déroulements alternatifs :

3a. Ecran affiche un récapitulatif de l'abonnement et Client annule.
4a. Redirection vers le menu principal.

Cas d'utilisation : Acheter un pass
Résumé : Client achète un pass.
Acteur : Client
Hypothèses : LecteurCarte fonctionne ou la fente à pièces fonctionne. Imprimante fonctionne.
Préconditions : Client a assez d'argent.
Etapes du déroulement normal :

1. Ecran demande le choix du type de pass.
2. Client choisit le type de pass.
3. Ecran demande d'inscrire les informations relatives au pass.
4. Client entre les informations.
5. Ecran affiche un récapitulatif du pass.
6. *Fin : voir description du cas d'utilisation Choix incluant un paiement.*

Postconditions : Un pass est acheté.
Déroulements alternatifs :

3a. Ecran affiche un récapitulatif de l'abonnement et Client annule.
4a. Redirection vers le menu principal.

Cas d'utilisation : Renouveler un abonnement
Résumé : Client renouvelle son abonnement.
Acteur : Client
Hypothèses : LecteurCarte fonctionne ou la fente à pièces fonctionne. Imprimante fonctionne.
Préconditions : Client est en possession d'un abonnement. Client a assez d'argent.
Etapes du déroulement normal :

1. Ecran propose de scanner le code barre ou de taper le code barre de l'abonnement.
2. Client tape ou scanne le code barre.
3. GestionBaseDeDonnees vérifie l'existence de l'abonnement et donne les informations à Ecran.
4. Ecran affiche les informations relatives à l'abonnement.
5. *Fin : voir description du cas d'utilisation Choix incluant un paiement.*

Postconditions : L'abonnement est renouvelé.
Déroulements alternatifs :

1a : Si le scanneur de code est en panne ou pas activé, Client est obligé de taper le code à la main.

Cas d'utilisation : Paiement
Résumé : Client paie sa commande.
Acteur : Client
Hypothèses : LecteurCarte fonctionne ou la fente à pièces fonctionne. Client a assez d'argent.
Préconditions : Client a confirmé une commande d'un titre de transport. I mprimante fonctionne.
Etapes du déroulement normal :

1. Ecran affiche le prix de la commande et demande le choix de paiement.
2. Client sélectionne la méthode de paiement souhaitée (en liquide ou par carte).
3. *Suite : voir cas d'utilisation En liquide ou Par carte.*

Postconditions : Client a payé sa commande.
Déroulements alternatifs :

2a. Client annule sa commande.
3a. Redirection vers le menu principal.
1b. Si LecteurCarte et la fente à pièces ne fonctionnent pas, Ecran affiche un message d'erreur.
2b. Redirection vers le menu principal.

Cas d'utilisation : En liquide
Résumé : Client paie sa commande en liquide.
Acteur : Client
Hypothèses : /
Préconditions : La fente à pièces fonctionne. Client a confirmé une commande et désire payer sa commande en liquide.
Etapes du déroulement normal :

1. 2. *Début : voir cas d'utilisation Paiement.*
3. Client insère de l'argent dans les fentes mises à disposition.
4. Reception rend le surplus d'argent donné.

Postconditions : Client a payé sa commande.
Déroulements alternatifs :

4a : Si la machine n'a pas assez de monnaie dans la caisse pour rendre le surplus d'argent donné, Ecran affiche un message d'erreur et rend le montant introduit.

Cas d'utilisation : Par carte
Résumé : Client paie sa commande par carte.
Acteur : Client, Système bancaire
Hypothèses : /
Préconditions : LecteurCarte fonctionne. Client a confirmé une commande et désire payer sa commande par carte.
Etapes du déroulement normal :

1. 2. *Début : voir cas d'utilisation Paiement.*
3. Client insère sa carte bancaire dans LecteurCarte.
4. Ecran demande à Client d'introduire le code PIN.
5. Client introduit son code PIN et Système bancaire le vérifie.
6. Système bancaire confirme le paiement. LecteurCarte rend la carte.

Postconditions : Client a payé sa commande.
Déroulements alternatifs :

5a : Client introduit un mauvais code PIN. Retour à l'étape 4.
6a : Système bancaire refuse le paiement (si solde insuffisant). Retour à l'étape 1.

Cas d'utilisation : Impression
Résumé : Imprimante imprime le titre de transport et le reçu si demandé. Reception donne le résultat de l'impression.
Acteur : Client
Hypothèses : /
Préconditions : Client a payé sa commande. Imprimante en marche.
Etapes du déroulement normal :
1. Imprimante imprime le titre de transport.
2. Ecran demande à Client s'il souhaite un reçu.
3. Imprimante l'imprime ou non en fonction du choix du Client.
4. Reception donne ce qui a été imprimé.

Postconditions : Client obtient son titre de transport et éventuellement un reçu.
Déroulements alternatifs : /

Cas d'utilisation : Retour à la page d'accueil
Résumé : Un message d'erreur est affiché à l'écran et Client est redirigé vers la page d'accueil.
Acteur : Client
Hypothèses : /
Préconditions : Il n'y a plus de papier et/ou d'encre.
Etapes du déroulement normal :
1. La commande est annulée.
2. La machine revient à la page d'accueil.

Postconditions : Client est informé de la panne et les options nécessitant un paiement ne sont plus disponibles.
Déroulements alternatifs : /

Cas d'utilisation : Retour au choix de paiement
Résumé : Client est redirigé vers le choix de paiement
Acteur : Client
Hypothèses : /
Préconditions : Le solde du Client n'est pas suffisant pour payer la commande ou la machine n'a pas assez de monnaie pour rendre au Client
Etapes du déroulement normal :
1. Affiche un message d'erreur.
2. La machine redirige Client vers le choix de paiement.

Postconditions : Client est redirigé vers la page d'accueil.
Déroulements alternatifs : /

Cas d'utilisation : Gérer les problèmes
Résumé : Technicien règle le problème en question.
Acteur : Technicien
Hypothèses : /
Préconditions : /
Etapes du déroulement normal :
1. *Cela dépend de la panne à gérer*

Postconditions : La panne de la machine est connue.
Déroulements alternatifs : /

Cas d'utilisation : Recharger encre et papier
Résumé : Technicien recharge le stock de papier et d'encre.
Acteur : Technicien
Hypothèses : /
Préconditions : /
Etapes du déroulement normal :
1. Technicien introduit du papier et de l'encre dans la machine.

Postconditions : Les stocks d'encre de de papier de la machine sont rechargés.
Déroulements alternatifs : /

Cas d'utilisation : Recharger la caisse
Résumé : Technicien recharge la caisse de la machine.
Acteur : Technicien
Hypothèses : /
Préconditions : /
Etapes du déroulement normal :
2. Technicien introduit de l'argent dans la caisse.

Postconditions : La caisse de la machine est rechargée.
Déroulements alternatifs : /

Cas d'utilisation : Gérer carte bloquée
Résumé : Technicien débloque une carte bloquée dans la machine.
Acteur : Technicien
Hypothèses : /
Préconditions : Une carte est bloquée dans la machine.
Etapes du déroulement normal :
1. Technicien débloque la carte bloquée dans la machine.

Postconditions : La carte est débloquée.
Déroulements alternatifs : /

Cas d'utilisation : Réparer les composants en panne
Résumé : Rend opérationnel(s) le(s) composant(s) optionnel(s) défectueux.
Acteur : Technicien
Hypothèses : /
Préconditions
Etapes du déroulement normal :
1. Technicien répare le(s) composant(s) défectueux.

Postconditions : La machine est opérationnelle.
Déroulements alternatifs : /

Diagramme de classes :

Ce diagramme (voir figure 2) représente les classes principales et leurs associations. Étant donné qu'il y a des croisements, nous avons décidé pour plus de clarté de faire un détour au croisement pour insister sur le fait que sont bien des associations différentes. Une partie de leur comportement sera explicité dans la suite de ce document à travers les diagrammes de séquences. Nous avons décidé de représenter chaque type de titre de transport à l'aide d'une classe différente (car chaque titre est un objet). La classe abstraite TitreDeTransport qui généralise Billet, Pass et Abonnement sert à éviter la redondance des attributs communs (le montant à payer, les dates de validité et d'expiration, …). Ces titres de transports sont rassemblés avec la classe Recu (la preuve de paiement d'un titre) dans le package Imprimable, car ceux sont tous les objets que l'on peut imprimer.

Un autre package InterfaceGraphique rassemble ce qui sera représenté (voir maquette de l'interface graphique) visuellement lors de l'implémentation. On peut y trouver Reception (l'endroit où on peut récupérer son argent, ainsi que tout ce qui aura été imprimé), LecteurCarte (le lecteur de carte de crédit), Ecran (ce qui affichera les messages, ainsi que les différents menus, l'utilisateur va principalement être en interaction avec celui-ci), et finalement FenetreConfiguration et FenetreSimulation (la fenêtre de configuration va permettre à l'utilisateur de choisir le type de distributeur qu'il souhaite, et la fenêtre de simulation va être le simulateur du distributeur). Il est important de préciser que toutes ces classes se trouvant dans le package InterfaceGraphique vont utiliser les classes de JavaFX (héritage, associations, …). Nous avons décidé de ne pas le représenter pour ne pas alourdir le diagramme.

Dans le dernier package Systeme, nous avons rassemblé toutes les classes qui touchent de près ou de loin à la logique du système. Il est composé de Controleur (comprend la logique principale du système), GestionBaseDeDonnees (qui permettra d'apporter des modifications à la base de données) et HoraireTrains (qui se chargera d'aller chercher les informations sur des horaires de train). Nous pouvons aussi y trouver PaiementLiquide (qui se chargera principalement de vérifier que l'utilisateur donne le bon montant lors d'un paiement liquide), Imprimante (qui se chargera, entre autres, de vérifier qu'il reste suffisamment d'encre et de papier pour imprimer un reçu ou un titre de transport) et Carte (cela représente les cartes de crédit, les données des cartes seront stockées dans la base de données). Finalement, la classe abstraite ComposantPanne est une généralisation des composants (du moins ceux qui sont représentés sous forme de classe) qui peuvent tomber en panne.

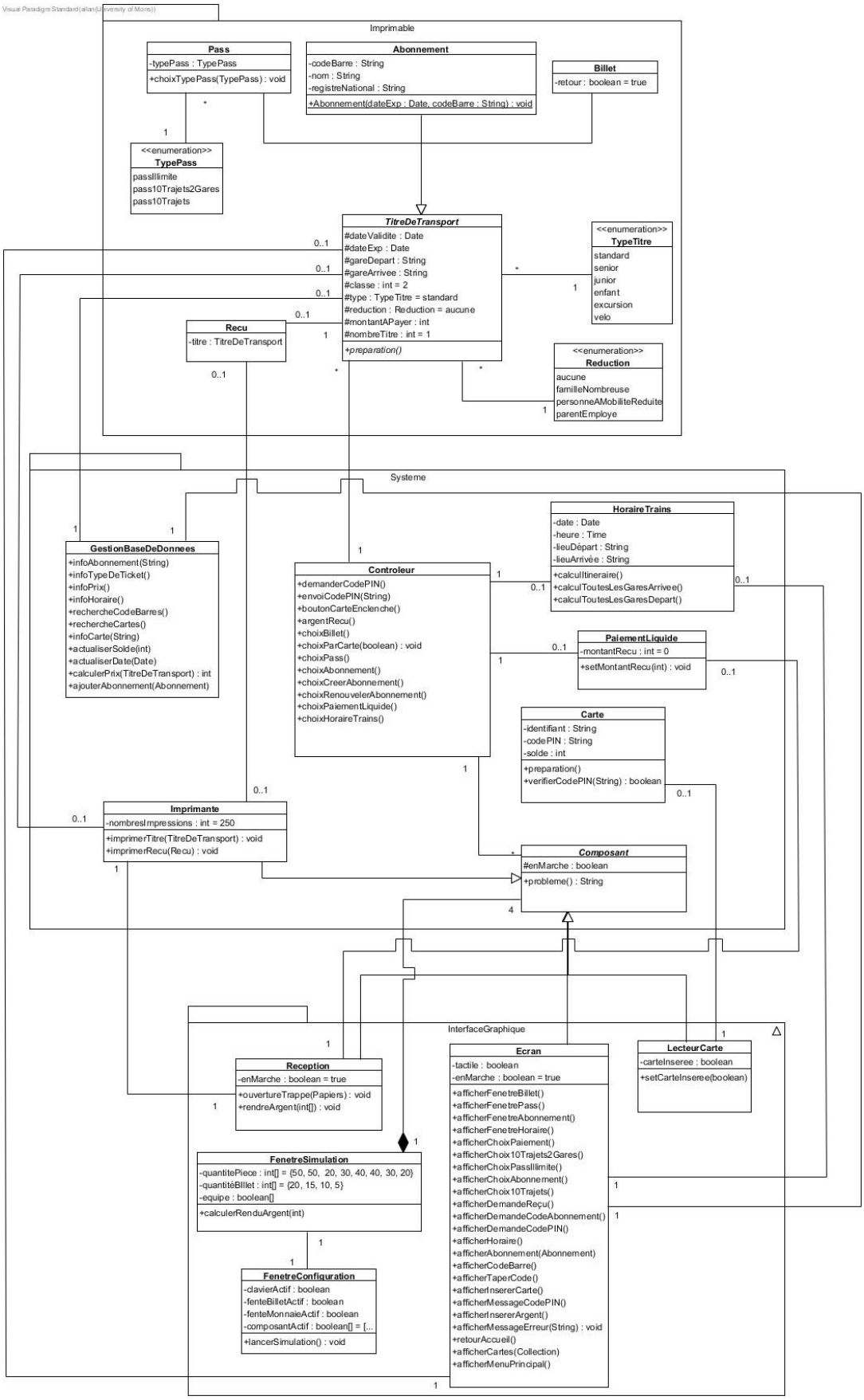

Figure 2 - Diagramme de classes

Diagrammes de séquences :

1. Acheter billet :

Ce diagramme (voir figure 3) représente le cas où, dans le menu principal, l'utilisateur appuie sur le bouton « Acheter un billet ». S'en suivra un appel d'une méthode qui permettra à Controleur de connaitre le choix de l'utilisateur, afin qu'il crée une instance de Billet. Ecran affichera un nouveau menu où l'utilisateur pourra taper toutes les informations sur le billet qu'il désire. *La suite est généralisée dans les diagrammes Paiement et Impression.*

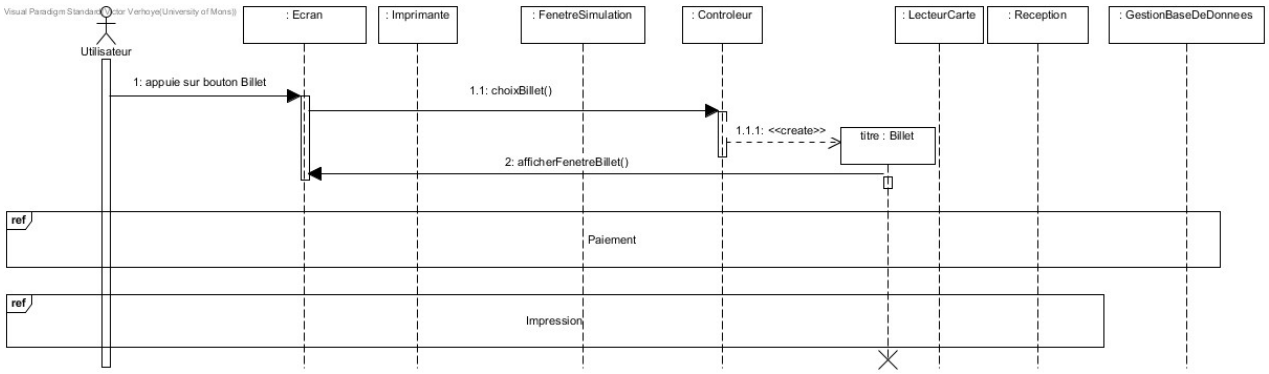

Figure 3 - Acheter billet

2. Acheter abonnement :

Ce diagramme (voir figure 4) représente le cas où, dans le menu principal, l'utilisateur appuie sur le bouton « Acheter un abonnement ». S'en suivra un appel d'une méthode qui permettra à Controleur de connaitre le choix de l'utilisateur, afin qu'il crée une instance de Abonnement. Ecran affichera un nouveau menu où l'utilisateur pourra taper toutes les informations sur l'abonnement qu'il désire. *La suite est généralisée dans les diagrammes Paiement et Impression.* Lorsque toute la procédure est finie, ce nouvel abonnement sera ajouté à la base de données.

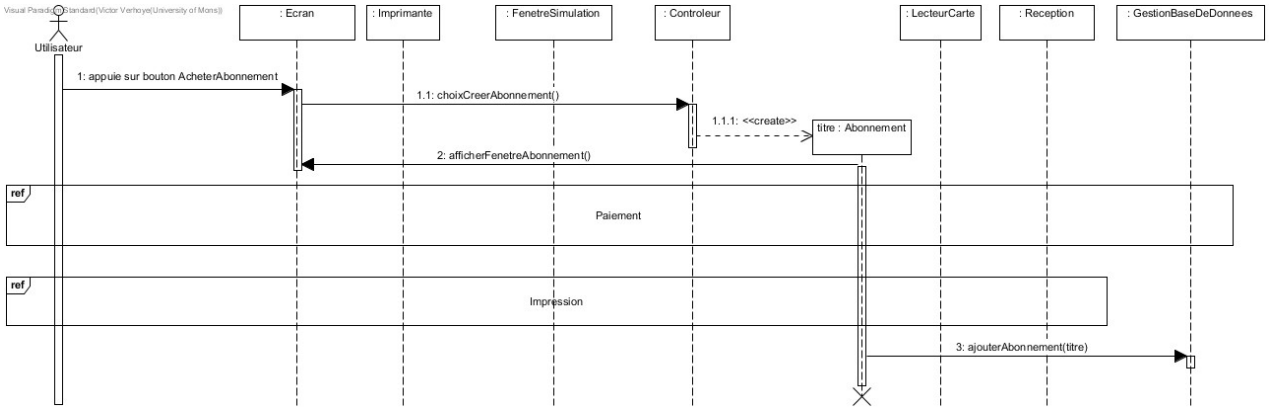

Figure 4 - Acheter abonnement

3. Renouveler abonnement :

Ce diagramme (voir figure 5) représente le cas où, dans le menu principal, l'utilisateur appuie sur le bouton « Renouveler un abonnement ». S'en suivra un appel d'une méthode qui permettra à Controleur de connaitre le choix de l'utilisateur, afin qu'il crée une instance de Abonnement. Ecran affichera un nouveau menu où l'utilisateur choisira s'il veut taper ou scanner le code de l'abonnement. S'il choisit de scanner le code, GestionBaseDeDonnees ira chercher tous les codes des abonnements existants dans la base de données, et Ecran les affichera. L'utilisateur pourra donc choisir un code parmi ceux affichés. S'il a choisi de taper lui-même son code, Ecran affichera une fenêtre où l'utilisateur pourra taper son code. Qu'il ait tapé ou choisi son code, la suite est similaire. GestionBaseDeDonnees va aller chercher les informations de l'abonnement en fonction du code, et Ecran va afficher un nouveau menu qui affichera l'abonnement en question et où l'utilisateur pourra taper les derniers détails du renouvellement. *La suite est généralisée dans les diagrammes Paiement et Impression.* Lorsque toute la procédure est finie, la nouvelle date d'expiration de l'abonnement sera mise à jour dans la base de données.

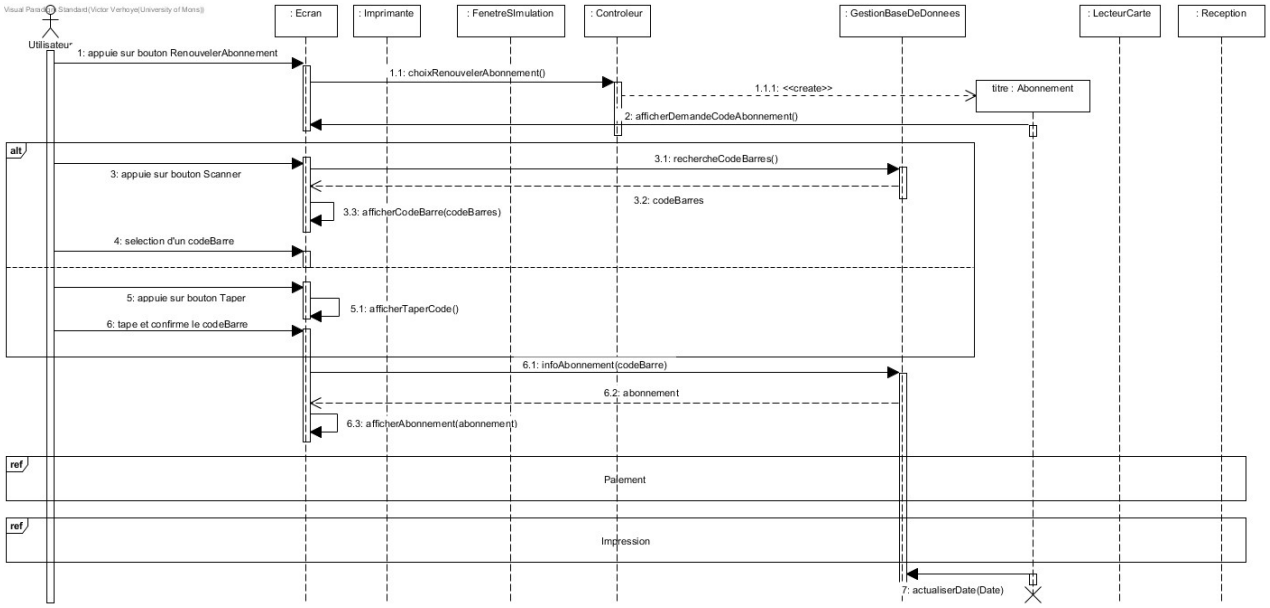

Figure 5 - Renouveler abonnement

4. Acheter pass :

Ce diagramme (voir figure 6) représente le cas où, dans le menu principal, l'utilisateur appuie sur le bouton « Acheter un pass ». S'en suivra un appel d'une méthode qui permettra à Controleur de connaitre le choix de l'utilisateur, afin qu'il crée une instance de Pass. Ecran affichera un nouveau menu où l'utilisateur pourra choisir le type de pass qu'il souhaite (10 trajets, 10 trajets entre 2 gares prédéfinies ou pass illimité). Une méthode donnera l'information à Pass, qui demandera à Ecran d'afficher un nouveau menu, qui permettra à l'utilisateur de taper toutes les informations sur le pass qu'il désire. *La suite est généralisée dans les diagrammes Paiement et Impression.*

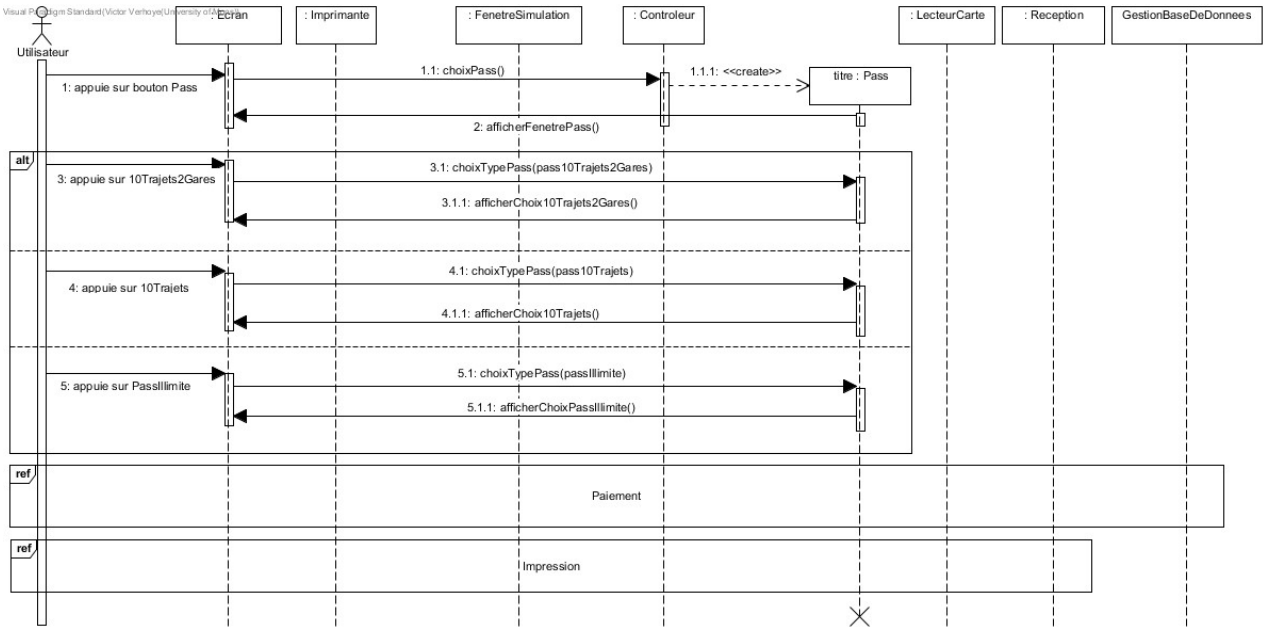

Figure 6 - Acheter pass

5. Paiement :

Ce diagramme (voir figure 7) représente non seulement le paiement d'un titre de transport, mais aussi la fin de la création de ce titre (cette décision a été prise afin d'éviter une quelconque redondance). Il est important de savoir que le paiement ne se fait qu'à la condition que nombresImpressions soit supérieure ou égale à nombreTitre. Ce choix a été pris pour éviter que l'utilisateur paie sans pouvoir recevoir ses billets. Lorsque l'utilisateur aura tapé toutes les informations concernant son titre de transport, il va confirmer. A ce moment-là, la méthode preparation() va aller attribuer à chaque variable de titre de transport une des données tapées par l'utilisateur auparavant. GestionBaseDeDonnees va, lui, calculer le prix de ce titre de transport à l'aide de la base de données (qui stockera pareillement les réductions, les types de titres de transport, …), et ensuite attribuer cette valeur en tant que montantAPayer du titre. Ecran va alors afficher un nouveau menu où l'utilisateur pourra choisir son type de paiement. S'il choisit par carte, Controleur va créer une instance de Carte, et Ecran va demander à l'utilisateur d'insérer sa carte. Lorsque l'utilisateur appuie sur « Insérer carte », GestionBaseDeDonnees va faire une recherche dans la base de données de toutes les cartes stockées, et Ecran va les afficher. Lorsque l'utilisateur va choisir une carte, LecteurCarte va passer son attribut carteInseree à vrai, et les informations sur la carte (code PIN, …) vont être ajouté à l'instance de Carte qui a été créée auparavant. Ecran va alors demander à l'utilisateur son code PIN, et tant que ça ne sera pas le même que celui de l'instance de Carte, il devra recommencer. S'il tape le code correctement, ça modifiera le solde sur sa carte et dans la base de données. S'il décide de payer en liquide, une instance de PaiementLiquide sera créée, qui aura comme attribut le prix du titre de transport. L'utilisateur va alors pour insérer des pièces ou des billets comme bon lui semble. Quand le montant reçu excède le montant à payer, l'argent donnée en trop sera rendue à travers Reception. Nous portons votre attention sur notre choix de « créer une instance de la classe abstraite TitreDeTransport ». Nous sommes bien conscients que cela n'a pas de sens dans la programmation même, mais cette décision a été prise afin de pouvoir généraliser le comportement de Paiement pour un Abonnement, un Billet, ou un Pass.

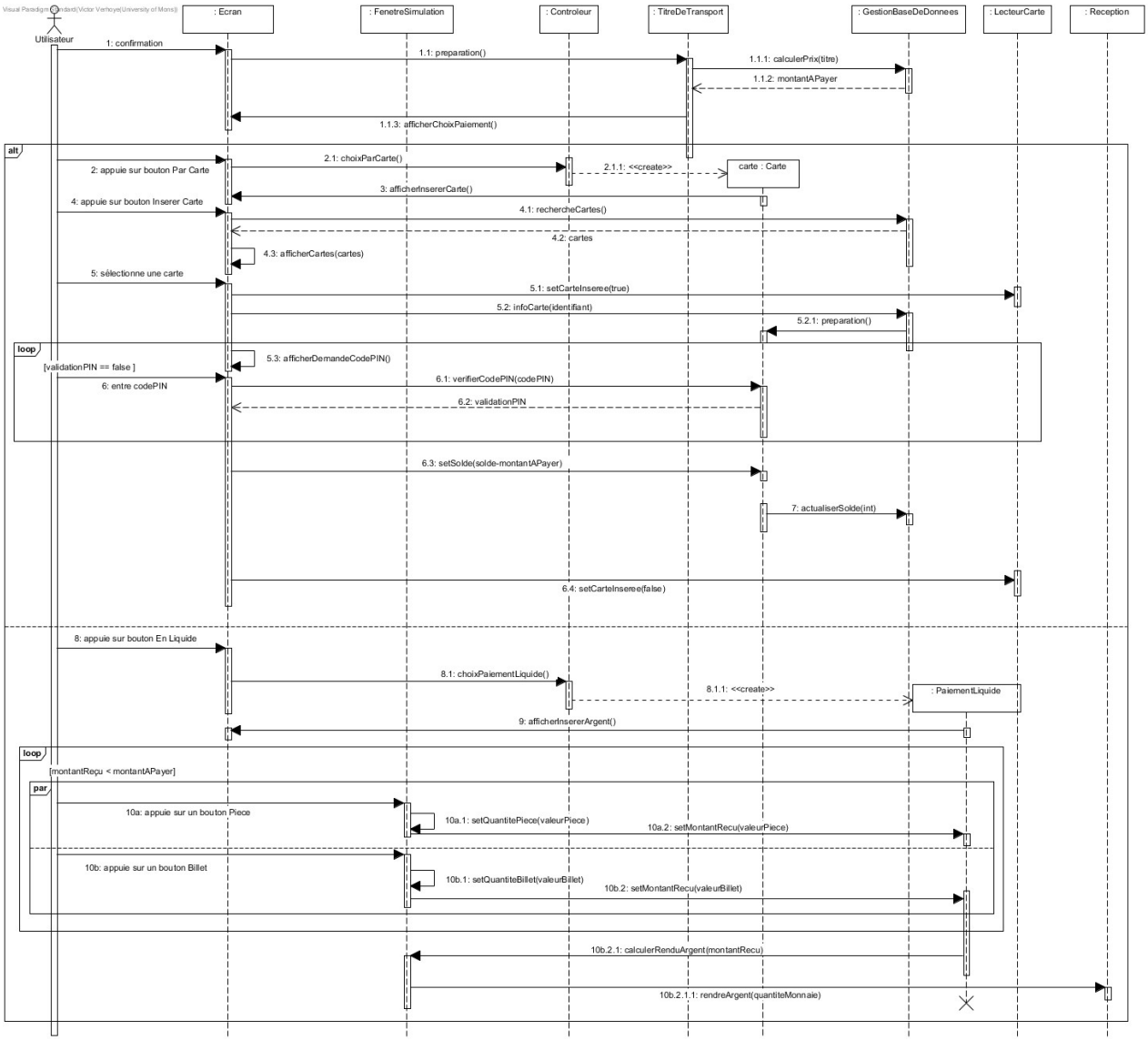

Figure 7 - Paiement

6. Impression :

Ce diagramme (voir figure 8) représente le fait d'imprimer le titre de transport que l'utilisateur vient de payer. Ce dernier peut, s'il le désire, imprimer un reçu. A chaque impression, l'attribut nombresImpressions est décrémenté. Il est important de rappeler qu'il n'y a pas de soucis pour imprimer les titres car le nombre d'impressions a été vérifié au préalable (voir la description du diagramme de séquences Paiement). Cependant, pour le reçu, il se peut que nombresImpressions soit égal à 0, à ce moment-là un message d'erreur est affiché et on revient à la page d'accueil.

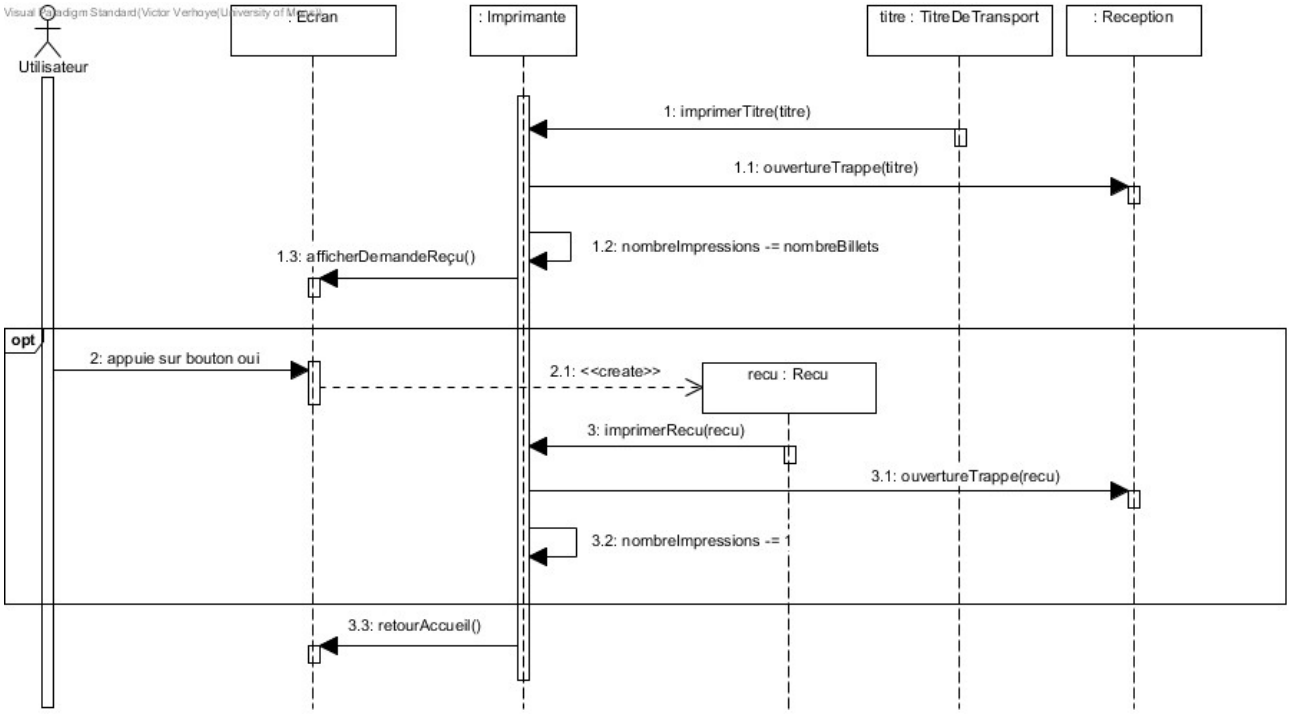

Figure 8 – Impression

7. Sortie de veille :

Ce diagramme (voir figure 9) représente tout simplement le fait que lorsque l'utilisateur va appuyer sur le bouton « Démarrer », Ecran va afficher le menu principal (voir maquette de l'interface graphique).

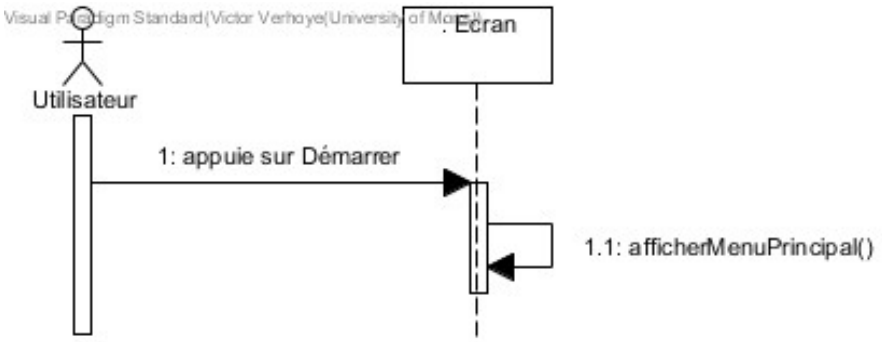

Figure 9 - Sortie de veille

8. Vérifier l'horaire d'un train :

Ce diagramme (voir figure 10) représente le choix de l'utilisateur de vérifier l'horaire d'un train. Ecran va afficher une fenêtre où l'utilisateur pourra taper les informations sur le trajet qu'il désire. En fonction des données entrées, HoraireTrains va calculer les trajets, et Ecran va les afficher.

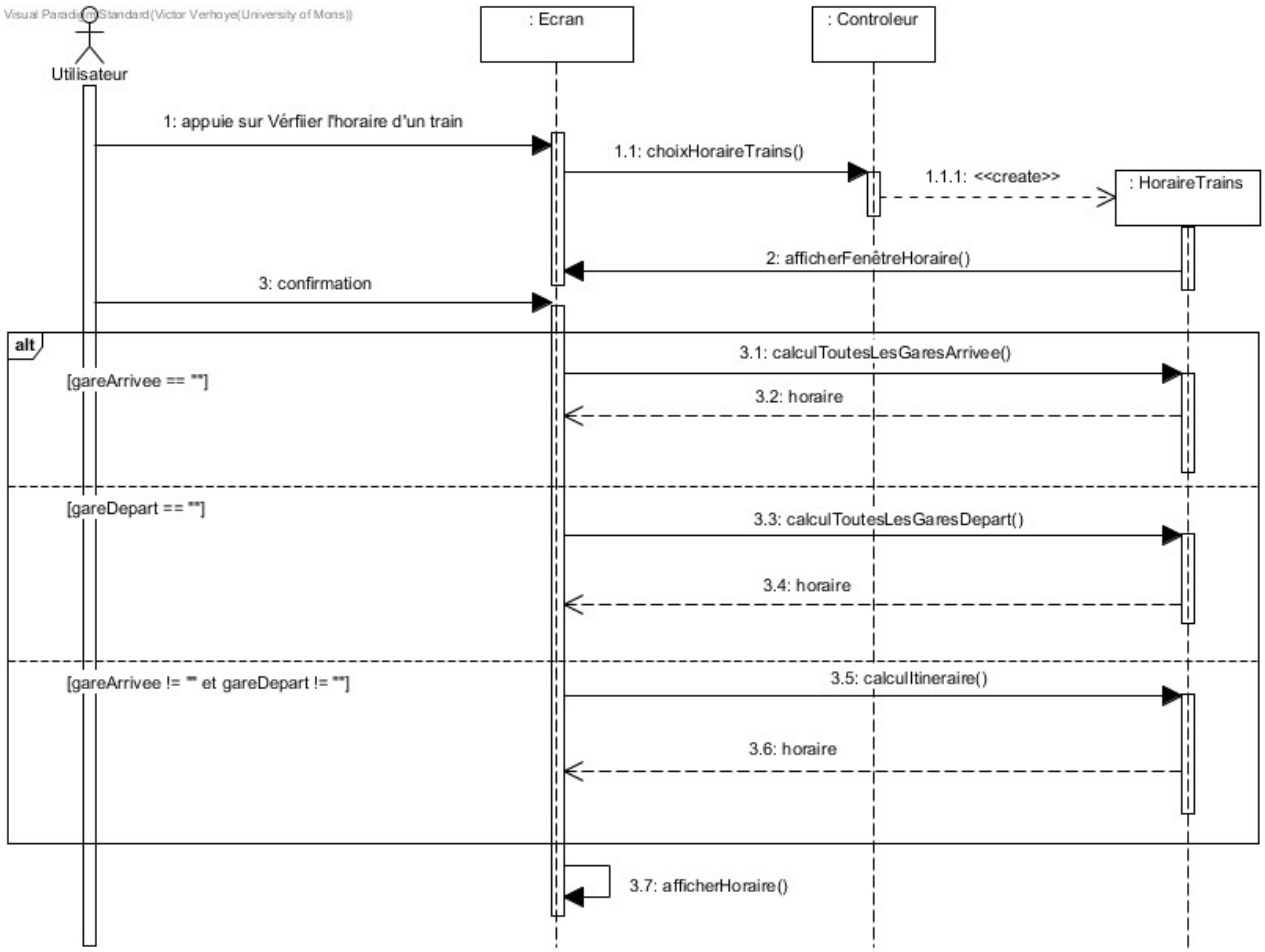

Figure 10 - Vérifier l'horaire d'un train

9. Créer/gérer une panne :

Ces diagrammes (voir figures 11 et 12) passent l'attribut enMarche d'un composant à false/true. Cet attribut est la représentation du fonctionnement ou du dysfonctionnement du composant en question. Nous ne représentons pas ici les mises en panne des composants qui ne sont pas représentés sous forme de classe.

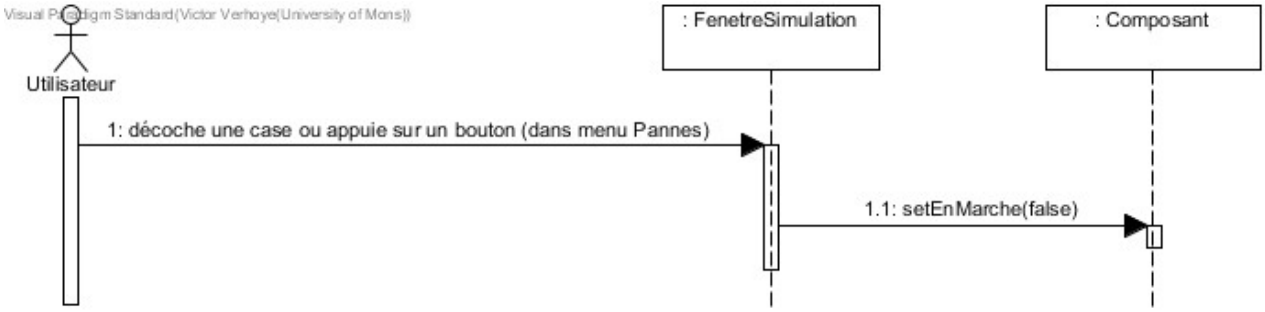

Figure 11 - Créer une panne

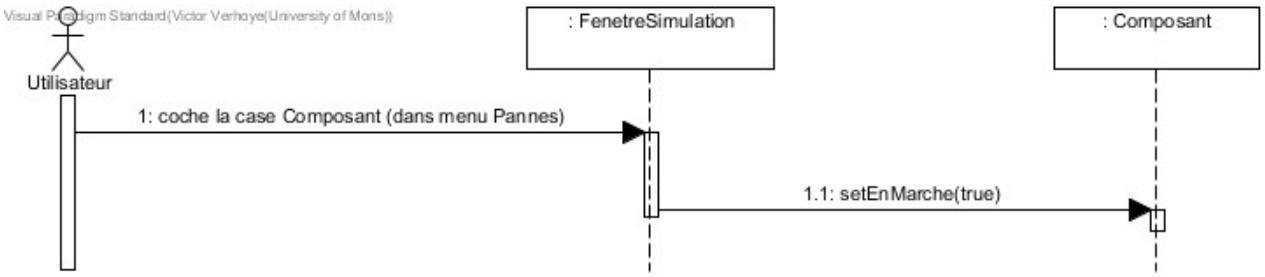

Figure 12 - Gérer une panne

10. Recharger/vider nombre d'impressions :

Ces diagrammes (voir figures 13 et 14) représentent le technicien qui remet de l'encre et du papier/qui vide l'encre et le papier. Ici l'attribut nombresImpressions correspond à la capacité du distributeur d'imprimer des tickets ou des reçus car un distributeur a une capacité limitée en encre et en papier. Ici, nombreImpressions est cette limite (vider nombre d'impressions est surtout là pour permettre à l'utilisateur de créer une panne).

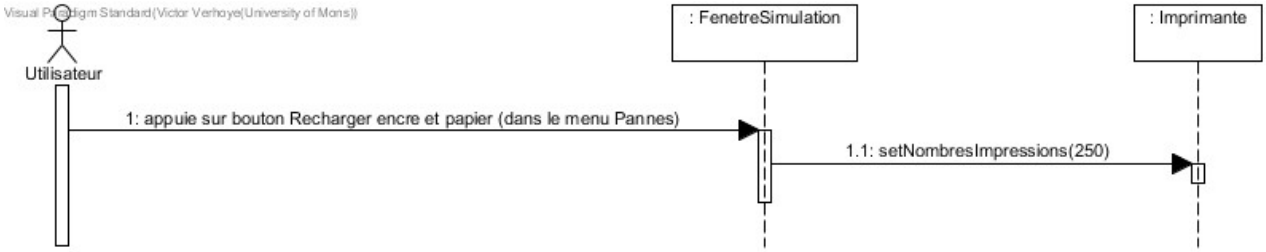

Figure 13 - Recharger nombre d'impressions

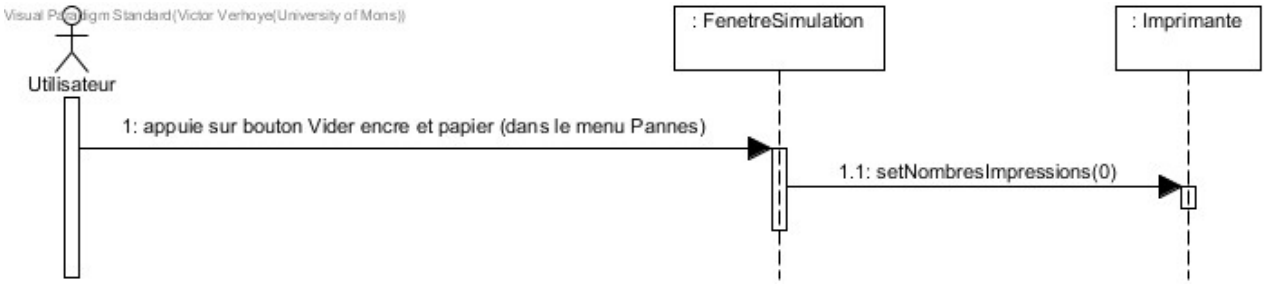

Figure 14 - Vider nombre d'impressions

11. Activer/désactiver composant optionnel :

Ces diagrammes (voir figures 15 et 16) représentent le choix de l'utilisateur de modifier FenetreSimulation quand il le désire. S'il coche/décoche une case dans le menu Composants optionnels, ça modifie l'attribut equipe dans FenetreSimulation, en passant à true/false à l'indice correspondant au composant en question.

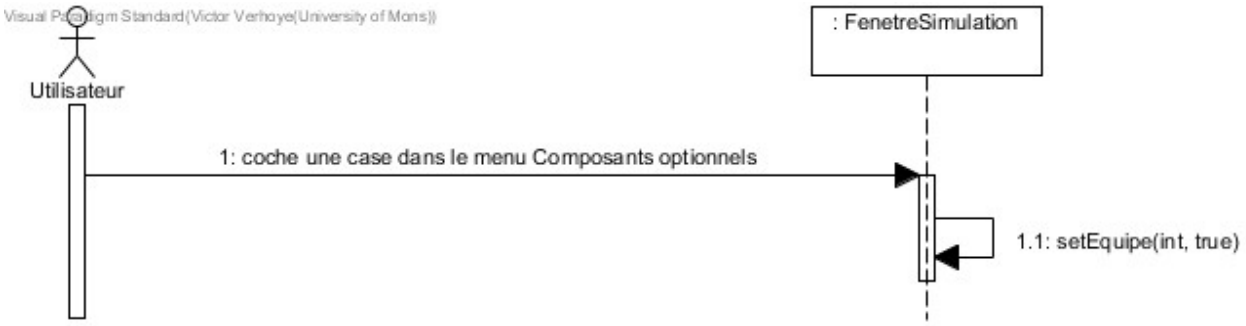

Figure 15 - Activer composant optionnel

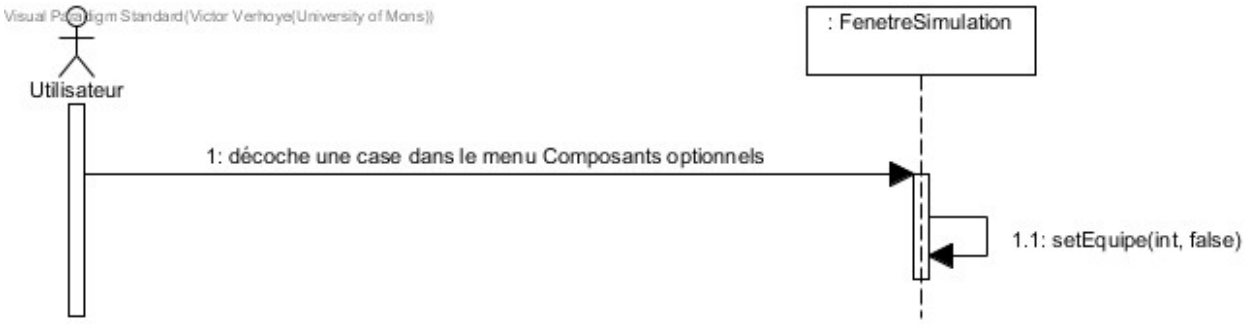

Figure 16 - Désactiver composant optionnel

12. Recharger/vider caisse :

Ces diagrammes (voir figures 17 et 18) représentent le technicien qui recharge/vide la caisse. Ici, les pièces et les billets sont représentés par deux tableaux où chaque élément du tableau représente la quantité restante d'une certaine pièce ou d'un certain billet. Ils sont triés par ordre croissant, donc le premier élément de pièce représente le nombre de pièces de 1 cent et le premier élément de monnaie représente le nombre de billets de 5 euros. Donc Recharger caisse consiste à actualiser le nombre de pièces et de billets que la machine possède, et Vider caisse représente quant à lui le fait de vider la caisse, on n'a donc plus de billets et de monnaies ; d'où le fait que tous les éléments du tableau passent à 0.

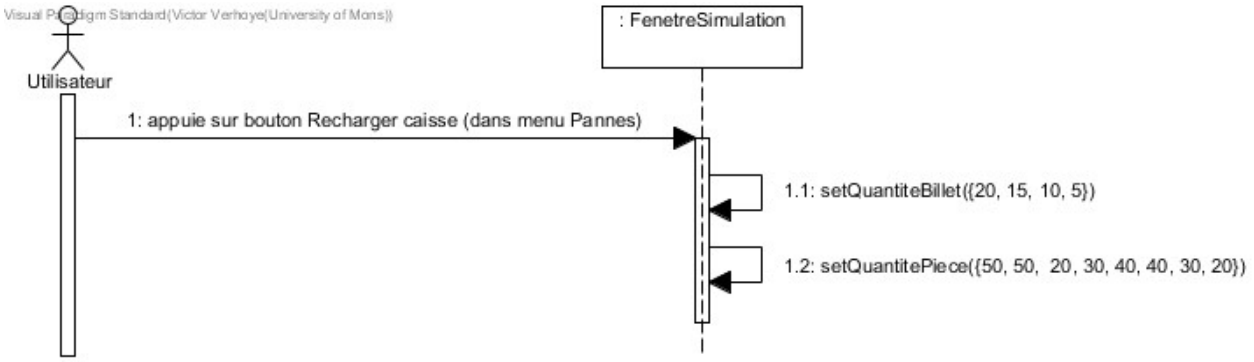

Figure 17 - Recharger caisse

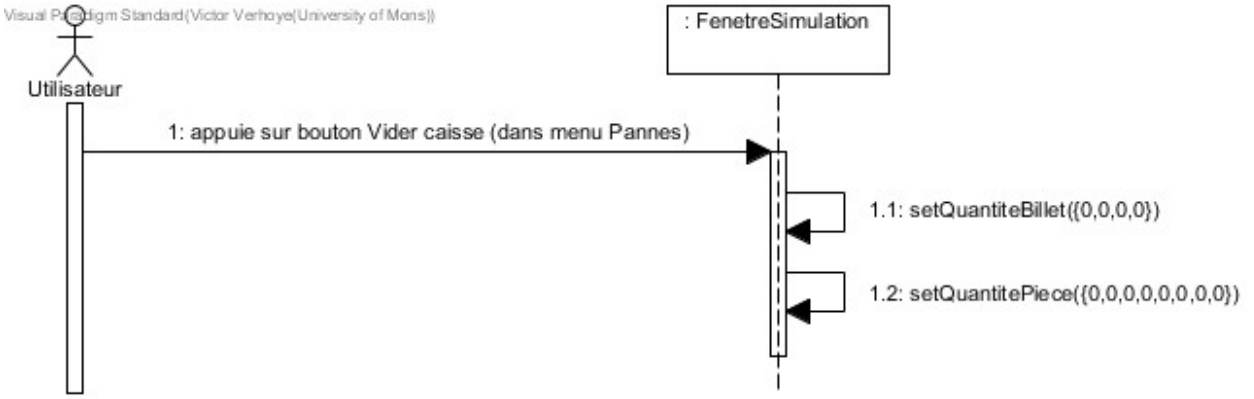

Figure 18 - Vider caisse

Diagramme d'interaction globale (interaction overview diagram) :

Ce diagramme (voir figure 19) représente le fonctionnement typique du distributeur par un utilisateur. Il active la machine, choisit une fonctionnalité proposée par le distributeur, et une fois cette utilisation de l'appareil finie, le distributeur revient à la page d'accueil. Le diagramme comporte une répétition de paiement et d'impression car un achat comprend un paiement et une impression. Le but de cette répétition était d'insister sur l'importance que dans une utilisation normale, on a toujours un paiement et une impression.

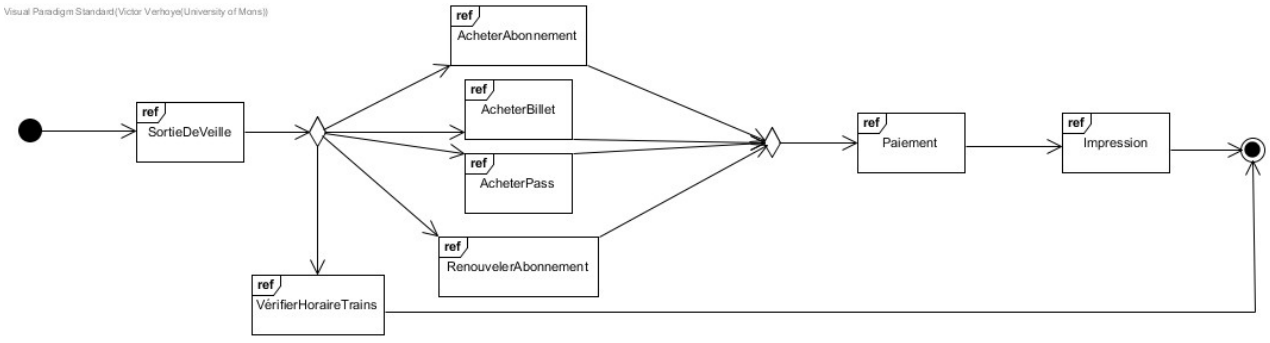

Figure 19 - Diagramme d'interaction globale

Diagramme d'états :

Le diagramme d'états représente les différents états par lesquels transite le distributeur. Il y a quelques points importants à remarquer. Tout d'abord, après l'état Paiement il n'y a plus de possibilités de retour en arrière. Il serait en effet ridicule que l'utilisateur paie son titre de transport, mais que celui-ci ne soit pas imprimé. Ensuite, le fonctionnement de PaiementLiquide repose sur le principe que l'utilisateur entre de la monnaie (représenté ici par les boutons pièces et billets), et ce tant que le montant qu'il a donné ne surpasse pas le montant à payer. Ici, pour le bien de la simulation, le montant à payer est fixe. Il est évident que dans l'implémentation celui variera en fonction du titre de transport choisi. Le choix d'un shallow history dans AchatTitreDeTransport a été pris dans le cas où un utilisateur veut modifier une partie des informations enregistrées, même après confirmation. Pareillement, nous avons laissé 3 secondes après chaque impression afin de simuler le temps d'impression d'une imprimante réelle. Après avoir imprimer le reçu, l'utilisateur ayant terminé sa commande, nous avons décidé de revenir à l'écran d'accueil automatiquement.