RAPPORT DE PROJET LO41

Par BISSARI Déromba Serge & KRIFA Khaled

UTBM - P2017

Introduction

Au coeur d'une société en constante évolution technologique, l'innovation est devenue une obligation. Dans cette optique, un acteur majeur de l'e-commerce invente la livraison par les airs avec un drone en moins de 30 minutes. Et pour faciliter le déploiement, il envisage de miser sur des centres aériens déportés de traitement des commandes qui seraient principalement utilisées pour couvrir des zones difficiles d'accès.
Dans le compte de l'UV ayant pour code LO41 et pour intitulé "Système d'Exploitation : Principes et Communication", il nous a été demandé de réaliser le projet de simulation des livraisons par drone.
Ce document décrit d'abord le projet, ensuite notre démarche de conception et enfin la structuration de notre rendu.

1. Présentation du projet

1.1. Description

La description du projet a été donnée dans le sujet du TP. A l'image des essaims de navettes-drones décollant d'un vaisseau mère, les drones auraient pour objectif de livrer des colis. Si le client rempli toutes les conditions nécessaires, les colis peuvent lui être livrés à sa demande. Ces conditions sont les suivantes :

- Être couvert par ce service de livraison ;
- Être installé dans un périmètre où les drones peuvent livrer en moins de 30 minutes ;
- Posséder un jardin pour que l'appareil puisse atterrir et repartir ;
- Avoir posé une cible (sur laquelle l'appareil se pose pour délivrer son paquet) dans son jardin;
- Être présent lors de la livraison.

Avec cela, différentes contraintes sont à prendre en compte à savoir :

- Autonomie du drone fonction de son type, de la distance à couvrir, du poids du colis, des conditions météorologiques ;
- Type du colis (priorités, poids, etc.) ;
- Recharge des batteries ;
- Disponibilité du drone ;
- Gestion de l'espace de vol (couloir aérien, etc.) ;
- Aléas liés aux clients (Absence, Retour du colis, etc.) ;
- Gestion de la livraison par le vaisseau mère avec slots d'approvisionnement.
- Etc.

1.2. Problématique

La problématique serait d'ordonnancer ou de synchroniser les différents processus qui s'enchaînent lors de ces livraisons afin d'assurer un meilleur service aux clients.

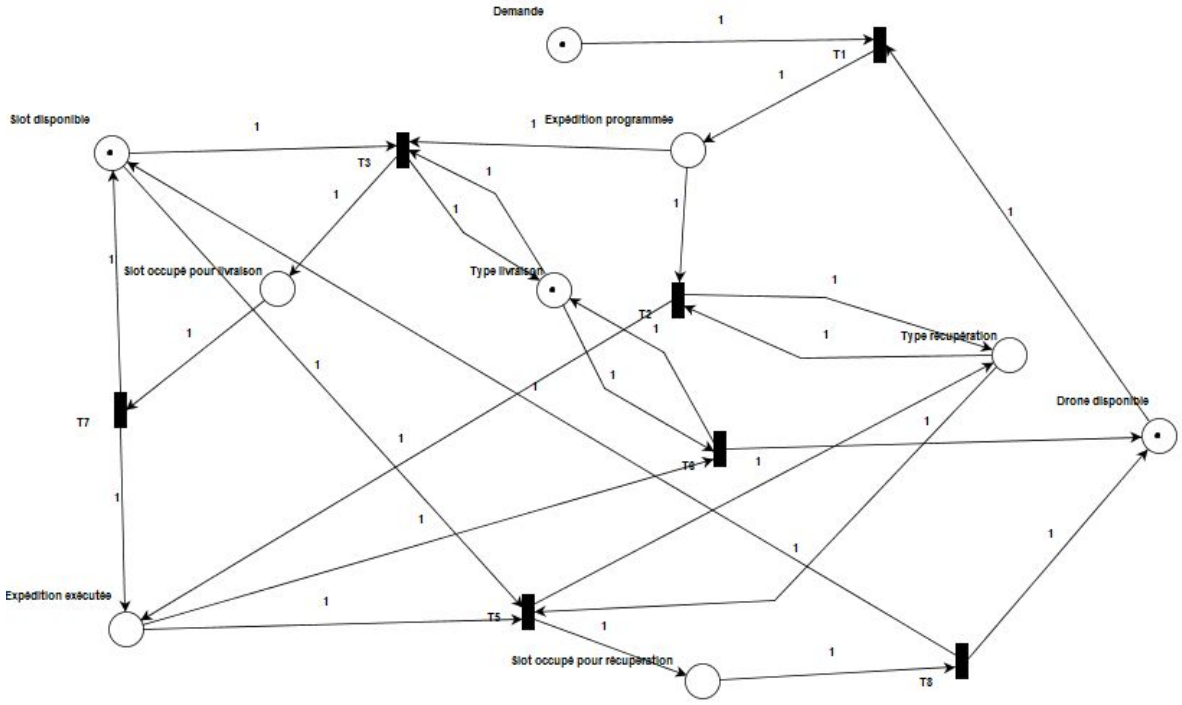
2. Conception

2.1. Les acteurs

Afin de modéliser notre système, il faut identifier les acteurs et leurs tâches et les événements déclencheurs. Après analyse nous avons distingué les acteurs suivants :

- La navette : reçoit les demandes de livraison et les déclenche en fonction des disponibilités des drones ;
- Les drones :
    - Récupèrent les colis à travers les slots lorsque ceux-ci sont disponibles et se chargent de les livrer ;
    - Vont récupérer les colis chez le client et les déposent dans les slots lorsque ceux-ci sont disponibles ;
- Les clients : effectuent les demandes de livraison comme de retour de colis.

2.2. Réseau de Pétri



3. Implémentation

3.1. Les outils

- **PIPE**

Pour modéliser le réseau de pétri de notre processus principal et faire une simulation nous avons utilisé PIPE (Platform Independent Petri Net Editor) qui est gratuit, facile à utiliser.

- **Le Langage C**

Pour réaliser la solution, nous avons utilisé le langage C. Nous avons eu à utiliser divers outils dudit langage, que nous avons appris à utiliser au cours de l'UV LO41. Il s'agit en l'occurrence des threads et des sémaphores.

- **L'environnement de développement**

Notre implémentation a été faite dans un environnement Linux Ubuntu. On s'est servi des éditeurs "Gedit" et "Sublime Text" ainsi que le compilateur C "gcc".

3.2. La structure

- **Programme principal (main) :**
  - **Initialise les données de simulation à partir des fichiers mis à disposition ;**
  - **Initialise les sémaphores utilisées pour la synchronisation des threads ;**
  - **Lance le thread de la navette ;**
  - **Attend la fin de l'exécution du thread de la navette ;**
  - **Détruit les sémaphores ;**
  - **Ferme le programme.**
- **Thread navette :**
  - **Lance le thread de recharge des drones ;**
  - **Réorganise les demandes par priorité ;**
  - **Lance un thread d'expédition pour chaque demande ;**
  - **Attend la fin de l'exécution des threads d'expédition ;**
  - **Arrête l'exécution du thread de recharge des drones ;**
  - **S'arrête.**
- **Thread expédition :**
  - **Vérifie la disponibilité d'un drone pour l'expédition ;**
  - **Prend un drone quand disponible ;**
  - **Si expédition de type livraison :**
    - **Vérifie disponibilité d'un slot ;**
    - **Occupe un slot quand disponible ;**
    - **Libère le slot occupé ;**
    - **Effectue la livraison ;**
  - **Si expédition de type récupération :**
    - **Effectue la récupération ;**
    - **Vérifie disponibilité d'un slot ;**
    - **Occupe un slot quand disponible ;**
    - **Libère le slot occupé ;**
  - **Dépose le drone ;**
  - **S'arrête.**
- **Thread recharge drone :**
  - **Incrémente les autonomies de tous les drones présents sur la navette de leur vitesse de rechargement ;**
  - **Si un drone atteint son autonomie maximale, il n'est plus rechargé.**

De plus, il a été ajouté un traitant pour le signal de sortie du programme pour détruire les sémaphores avant sortie du programme.

4. Perspectives

Faute de temps, nous n'avons pu finir toutes les fonctionnalités que nous avons envisagé pour ce simulateur. Mais voici une liste, non exhaustive de celles que nous trouvons pertinentes :

- **La livraison de plusieurs colis par un seul drone : Avec des coordonnées GPS nous pourrions implémenter un algorithme de recherche opérationnelle permettant de lister les livraisons que peuvent effectuer un drone en un seul voyage.**
- **L'approvisionnement de la navette : Gérer le stock de la navette et à l'épuisement, ravitailler celle-ci.**
- **Maintien de la navette : On pourrait faire du thread de la navette un thread qui ne s'arrête que sur la demande de l'utilisateur et ainsi implémenter un thread client qui enverrait les demandes à la navette qui se chargera de les traiter.**
- **Gérer l'absence du client lors du voyage du drone : Le client pourrait être absent à l'arrivée du drone, le drone attendrait donc un moment et s'en irait si le client ne se présente pas.**

Conclusion

Ce projet a été pour nous un exemple de réalisation possible grâce aux connaissances acquises au cours de l'UV LO41, un approfondissement. La modélisation, la conception et la réalisation ont été pour nous un défi à relever face à la problématique d'ordonnancement posée. Et notre démarche a surtout porté sur les synchronisations entre les threads de sorte à éviter les problèmes d'accès concurrent possible.