

Projet d'AL

V3: drone autonomy from truck:
allocation for driver, scheduling, mobile possibly disconnected



Team G

Martin Bruel, Thibaut Esteve, David Lebrisse,
Nathan Meulle, Kévin Ushaka

Sommaire

Acteurs	3
Personas	3
Scénarios	4
Epics	7
Diagramme d'architecture	9
Diagramme de composants	11
Déroulement du scénario 1.1 dans notre architecture	12
Choix Technos	14

Acteurs

- Drivers
- Operators
- Drone
- Customer
- Service de carte

Personas

- Sam (rôle : Customer) est un retraité de Genouillac. Il commande souvent des produits depuis des sites de e-commerces car aucune grande ville n'est à côté de sa maison et qu'il ne peut pas se déplacer facilement. Il souhaite connaître le statut de la livraison pour être prêt à le recevoir à sa porte.
- Christine (rôle : Operators) est opératrice de traitement de colis. Elle travaille dans un dépôt pour une société de livraison de colis par drones. Elle est chargée de récupérer et redistribuer les colis qu'elle reçoit. Elle veut charger les camions qui partent pour livraison avec le colis des clients.
- Marcel (rôle : Driver) est livreur pour une agence de livraison à domicile. Sa société se spécialise dans la livraison par drones. Il habite à Genouillac dans la Creuse. Sa région est très faiblement peuplée et très peu couverte par le réseau 4G ou 5G. Il est chargé de livrer chaque matin les colis qui sont déposés dans son camion. Il souhaite réaliser facilement toutes ses livraisons avec les drones qui lui sont fournis par sa société. Une fois arrivé au point de déploiement, il est chargé de la synchronisation des drones avec le camion puis doit se tenir prêt à intervenir en cas de problème.

Scénarios

Scénario 1.1 [Obligatoire]: Chaque drone livre un colis à la fois

- Le conducteur arrive sur le lieu de déploiement des drones et démarre sa tablette.
- Le conducteur lit sur sa tablette le colis qu'il doit associer au drone libre.
- Il charge le colis sur le drone.
- Le conducteur indique que le colis est chargé sur la tablette.
- Le camion lui communique l'itinéraire de livraison.
- Le drone signale qu'il est parti pour sa livraison et commence la navigation.
- Il arrive à l'adresse et dépose le colis, il n'est alors pas connecté au camion
- Il repart pour rejoindre le camion.
- Arrivé dans la zone de couverture du camion, il signale qu'il a livré le colis.
- Le camion signale que le colis est livré dès qu'il peut se connecter à internet
- Le drone signale qu'il est retourné sur le camion et qu'il est libre
- Le système recalcule l'allocation des colis pour savoir quel colis il lui donne

Scénario 1.2 [Obligatoire]: Un grand drone livre plusieurs colis ou un seul.

- Le camion comporte des drones classiques capables de soulever 1 colis léger et des drones lourds pouvant supporter 1 colis lourd ou plusieurs colis légers.
- Arrivé sur le site de livraison, le conducteur lit sur sa tablette le colis qu'il doit associer au drone libre.
- Le conducteur a un colis lourd en priorité et d'autres colis légers qui doivent être livrés ensuite.
- Le conducteur charge le colis lourd et envoie le drone effectuer sa livraison (cf Scénario 1).
- Le drone revient, il ne reste que des colis légers. Le conducteur charge tous les colis dans ce drone et l'envoie pour sa dernière livraison.

Scénario 1.3 [Optionnel]: Couverture étendue : le camion effectue plusieurs livraisons

- Le conducteur lit sur sa tablette la première position où doit se rendre le camion
- Le conducteur indique sur sa tablette qu'il est arrivé à la première position.
- Une fois arrivé, il déploie ses drones, ceux-ci vont livrer les colis dans un périmètre autour du camion. (*taille du périmètre à déterminer*)(itération du scénario 1).
- Quand les drones ont fini leurs livraisons et sont revenus au camion, le conducteur lit sur sa tablette la seconde position où doit se rendre le camion.

Remarque : les coordonnées indiquées au camion sont calculées de manière à optimiser la vitesse de livraison des colis ainsi que la qualité du signal entre le camion et les drones.

Scénario 2.1 [Optionnel]: Un drone n'est pas en mesure de décoller

ex : La batterie est tombée, On ne peut plus charger

Scheduling des autres drones pour compenser.

- Le conducteur lit sur sa tablette le gros colis qu'il doit associer au gros drone libre.
- Il charge le gros colis sur le gros drone.
- Le conducteur indique que le colis est chargé sur la tablette.
- Le camion lui communique l'itinéraire de livraison.
- Le drone ne répond pas
- Le système réalloue les colis aux drones qu'il reste en fonction de leur taille
- Le camion signale les colis qui ne pourront pas être livrés

Scénario 2.2 [Optionnel]: Le drone rencontre un obstacle

- Le drone est parti pour sa livraison.
- Il échange sa position avec le camion.
- Il rencontre un obstacle (entraînement des pompiers).
- Il revient au dernier point de connexion avec le camion.
- Il envoie sa position au camion et demande un recalcul du plan de vol.
- Le camion calcule et renvoie un premier petit itinéraire alternatif.
- Le drone met à jour son plan de vol.
- En cas d'impossibilité à franchir l'obstacle, le drone revient pour récupérer un tout autre itinéraire.

Scénario 2.3 [Obligatoire]: Le drone ne revient pas au camion

- Le drone est parti pour sa livraison.
- Il échange sa position avec le camion.
- Le camion ne reçoit plus de données du drone.
- Le camion détermine un timeout en fonction de la distance qui reste à parcourir par le drone.
- Le timeout est terminé, le camion considère que le drone est inopérant.
- Le système réalloue les colis aux drones qu'il reste en fonction de leur taille.
- Le camion signale les colis qui ne pourront pas être livrés

Scénario 3 [Obligatoire]: Route accidentée

- Le camion ne peut pas atteindre la position prévue.
- Le chauffeur s'en approche le plus possible.
- Il relance le calcul des plans de vols à partir de sa nouvelle position.
- Une livraison n'est pas réalisable (beaucoup trop loin), elle est annulée.
- Le conducteur lance la livraison des autres colis avec les drones.

Scénario 4 [Obligatoire]: Connection perdue entre le camion et le drone

- Le camion n'a pas reçu de signal du drone sur sa position (envoyée normalement à intervalle régulier).
- Le camion enregistre la dernière position connue du drone et lance un timeout (calculé en s'appuyant sur la distance). Une fois dépassé le drone est considéré comme perdu (la dernière position connue sera envoyée au site pour permettre l'envoi d'une équipe de récupération).
- Le drone poursuit son plan de vol. Il effectue une synchronisation lors de sa reconnexion. Celle-ci lui permet de corriger sa trajectoire et obtenir les dernières informations quant aux imprévus de parcours (météo, commande annulée...)

Scénario 5 [Optionnel]: Meilleure zone de connection pour le camion

- Au cours de son vol, le drone trouve une meilleure zone de connexion pour le camion.
- Le camion notifie chaque drone de sa nouvelle position

Epics

Epic #1 pour 12/10

Nom : Minimal Viable Product

Description : Livraison de plusieurs colis, chaque drone a un colis à la fois.

- En tant que livreur, je veux consulter la liste d'affectation des colis afin de recharger les drones et débiter la livraison.
- En tant que camion, je veux affecter le colis le plus prioritaire au prochain drone libre afin d'optimiser mes horaires de livraison.
- En tant que drone, je veux télécharger l'itinéraire de livraison afin de démarrer la navigation.
- En tant que drone, je veux suivre l'itinéraire de livraison afin de déposer le colis à l'adresse du client et retourner au camion.
- En tant que livreur, je veux que le camion me notifie d'un drone libre et du prochain colis à livrer afin de démarrer une autre livraison.
- En tant que camion, je veux être notifié régulièrement de l'activité des drones, afin de maintenir mon planning de livraison.
- En tant que développeur, je veux que le camion enregistre l'état de mes livraisons afin de notifier le client.
- En tant que développeur, je veux que le drone notifie le camion de sa fin de livraison afin de changer l'état de la livraison.

Epic #2 pour 19/10

Nom : Connection issue

Description: Le drone peut subir une déconnexion voire être perdu.

- En tant que drone, je veux transmettre ma position au camion afin de me reconnecter avec le camion.
- En tant que camion, je veux enregistrer le(s) derniers points de connexion afin de déterminer un timeout en fonction de la distance qui reste à parcourir par le drone.
- En tant que développeur, je veux annuler les livraisons qui sont beaucoup trop éloignées du camion afin de réussir mes livraisons.
- En tant que développeur, je veux réassigner les colis rattachés au drone qui est perdu afin de garantir les livraisons restantes dans les meilleurs délais.

Epic #3 pour 26/10

Nom : Drone optimisation

Description : On ajoute un grand drone qui livre plusieurs colis.

- En tant que camion, je veux affecter les colis en fonction de leur poids afin de ne pas surcharger les drones.
- En tant que camion, je veux déterminer une nouvelle zone de décollage afin d'optimiser mon plan de route.
- En tant que livreur, je veux lancer le calcul des plans de vols à partir d'une nouvelle position afin de m'adapter aux accidents de la route.

Epic #4 pour 02/10

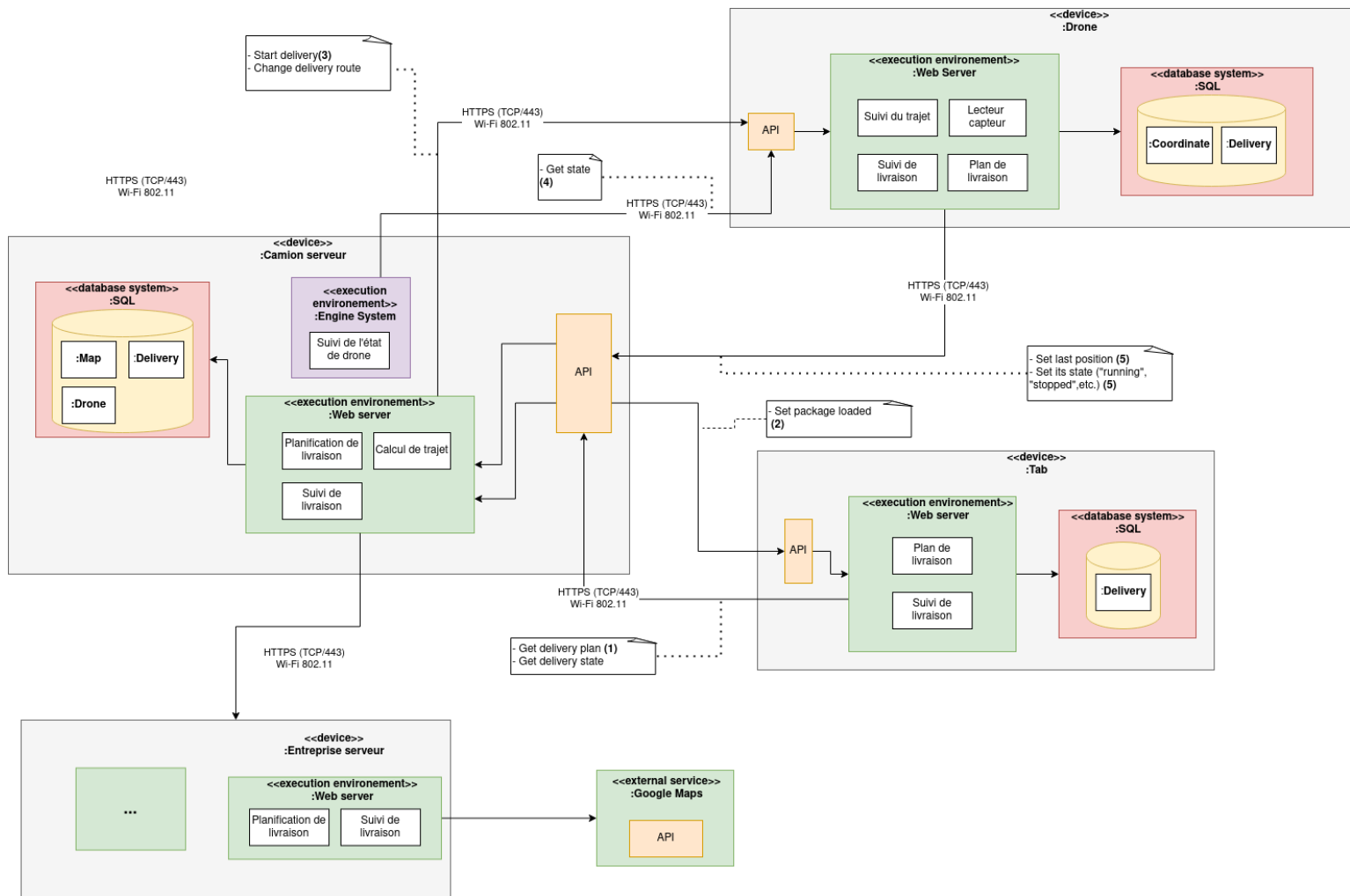
Nom : Vésuvis

Description : Le camion ne peut pas atteindre le point de déploiement

- En tant que camion, je ne peux pas accéder au point de déploiement et indique donc la position alternative trouvée
- En tant que drone je veux télécharger les nouveaux itinéraires.

Diagramme d'architecture

(cf : <https://github.com/pns-si5-al-course/al-drone-21-22-al-drone-21-22-g/blob/main/archi1.png>)



Notre système propose la livraison de colis à partir d'un entrepôt. La première partie de la livraison est effectuée en camion par nos conducteurs, la seconde est réalisée par drones lancés depuis le camion. On envisage que l'entreprise dispose de différents microservices pour gérer son activité. Quand le camion est chargé, l'opérateur lance le chargement des cartes géographiques et des plans de livraisons. C'est le microservice de suivi de livraison qui récupère les cartes depuis un service externe et les envoie au camion. Il envoie également les données sur les livraisons qu'il faut effectuer. Le serveur du camion stocke donc les livraisons, les cartes et les informations sur les drones (poids, capacité de la batterie, etc.).

Le camion dispose d'une application sur un serveur local pour la planification de livraison, le calcul de trajet et le suivi de livraison, et d'un certain nombre de tâches CRON pour contrôler l'état des drones régulièrement. Quand le livreur arrive à son point d'arrivée, il se connecte au réseau du serveur en *Wi-Fi* et appelle le serveur en HTTP

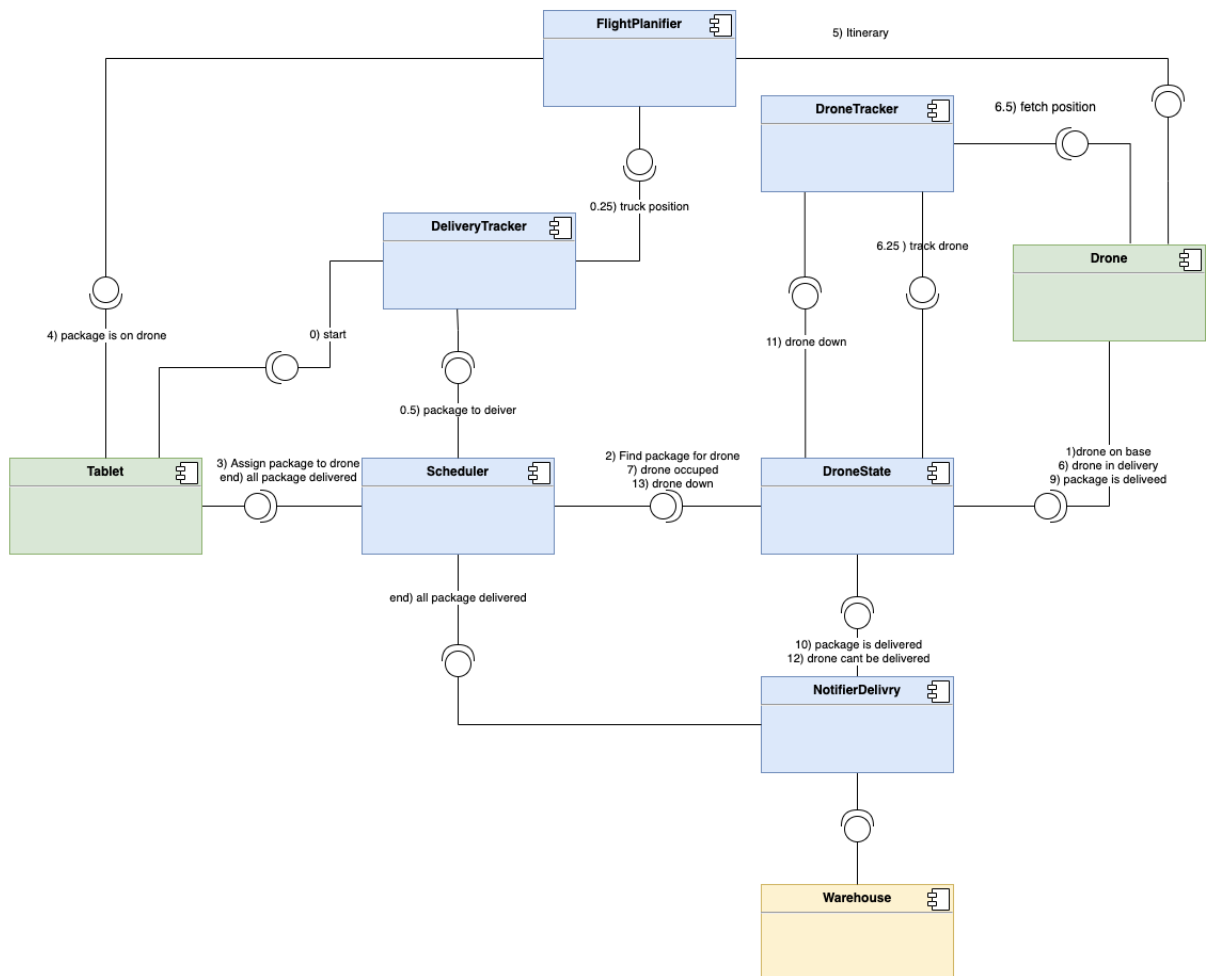
pour démarrer la planification des trajets et l'assignation des drones. Il récupère les différentes assignations et les informations sur la livraison. Le serveur du camion peut rétablir les données de livraison à partir des données sur la tablette. Après le chargement, le livreur démarre la livraison, le camion est connecté en *filaire et wifi* au drone et envoie les positions à suivre au drone ce qui démarre le trajet du drone.

Le camion demande régulièrement l'état du drone. En prenant en compte la possibilité d'une perte de connexion, l'on requiert plusieurs fois le drone et mettons en place un timeout avant de définir le drone comme perdu. De son côté, le drone enregistre ses dernières positions pour gérer le scénario du drone perdu. Il peut revenir à son dernier point de livraison. Il a également les informations de la livraison pour mettre à jour son état. Il a des tâches en concurrence. Il lit les données de ses capteurs, il les traite, suit le trajet et envoie son état au camion. Quand le drone revient au camion, on le rebranche pour le recharger et lui transférer sa prochaine livraison. L'état des livraisons sur le serveur du camion sont également mis à jour.

Le retour sur camion à l'entrepôt synchronise les données de livraison avec les données du serveur de l'entreprise.

Diagramme de composants

(cf : <https://github.com/pns-si5-al-course/al-drone-21-22-al-drone-21-22-q/blob/main/component1.png>)



Nous n'avons pas détaillé le fonctionnement interne du drone et du système de l'entrepôt. Nous faisons ici l'hypothèse que l'entrepôt dispose déjà d'une solution présentant une API dont nous faisons l'usage. Quant à la tablette, c'est une CLI sur laquelle le conducteur effectue ses actions sur le système.

Le début d'une livraison débute avec notre composant **DeliveryTracker**, il récupère la position du camion depuis un capteur et l'envoie au **FlightPlanifier** qui détermine l'itinéraire. Il envoie aussi les paquets à livrer au **Scheduler**.

Le **DroneTracker** demande régulièrement la position de tous les drones et si celui-ci ne lui répond pas lance un timeout. Si le timeout est dépassé, il prévient le **DroneState** que le drone est perdu.

Enfin, le composant **NotifyDelivery** envoie au service externe Warehouse l'état de livraison d'un drone ou si le drone n'a pas pu effectuer sa livraison (afin de notifier les utilisateurs).

Déroulement du scénario 1.1 dans notre architecture :

Le conducteur arrive sur le lieu de déploiement des drones et démarre sa tablette.

La Tablet envoie un signal de départ au composant DeliveryTracker, celui-ci lit la position du camion depuis un capteur et la communique au FlightPlanifier. Ensuite le DeliveryTracker envoie au Scheduler la liste des paquets à livrer.

Le conducteur lit sur sa tablette les colis qu'il doit associer aux drones libres.

Le Scheduler envoie à Tablet les affectations des drones aux paquets.

Il charge le colis sur le drone et indique que le colis est chargé sur la tablette.

Tablet indique l'identifiant du drone et l'adresse de livraison du colis au FlightPlanifier.

Le camion lui communique l'itinéraire de livraison.

Celui-ci va alors calculer l'itinéraire du drone et l'envoyer au Drone.

Le drone signale qu'il est parti pour sa livraison et commence la navigation.

Drone va envoyer un signal au DroneState pour que celui-ci mette à jour l'état du drone à "en livraison".

Le camion surveille la position du drone.

Périodiquement, DroneTracker va envoyer un signal à Drone pour lui demander sa position, il amorce alors un timeout pour considérer ou non un drone comme perdu. Celui-ci lui répond alors avec sa position et DroneTracker met alors à jour le timeout.

Il arrive à l'adresse et dépose le colis, il n'est alors pas connecté au camion.

Drone envoie un signal au DroneState pour l'avertir de changer son état à "colis livré".

Il repart pour rejoindre le camion.

Arrivé dans la zone de couverture du camion, il signale qu'il a livré le colis.

Drone parvient à envoyer son nouveau statut au DroneState qui met alors à jour l'état du drone en base de données. DroneState indique également à NotifierDelivry que le colis livré est arrivé à sa destination.

Le camion signale que le colis est livré dès qu'il peut se connecter à internet

NotifierDelivry effectue une vérification de connexion qui aboutit, il récupère alors les livraisons qui n'ont pas encore été communiquées et les envoie à la Warehouse.

Le drone signale qu'il est retourné sur le camion et qu'il est libre

Drone envoie un signal au DroneState pour l'avertir de changer son état à "prêt à livrer".

Le système recalcule l'allocation des colis pour savoir quel colis il lui donne

DroneState signale au Scheduler qu'il peut assigner un paquet au drone qui vient de revenir.

Choix Technos

Une application en 3-tiers embarquée dans le camion nous paraît une bonne solution car nous avons un certain niveau de complexité autour de nos composants. Nous avons écarté une solution en microservices car nous n'avons pas besoin de rendre l'application scalable au sein du camion (le nombre d'appels est borné). De plus cela ajouterait des coûts de communications entre les composants dû au protocole HTTP. Néanmoins la communication avec les acteurs externes au camion sera réalisée via le protocole HTTP. Nous allons nous orienter vers du java pour l'application du serveur embarqué dans le camion.

Les différents mocks et services externes seront écrits en python : l'aspect interprété du langage nous permettra de tester plus finement les fonctionnalités au cours de notre développement.