Lycée TOUCHARD WASHINGTON

Page de garde – Dossier d'analyse

Session 2025 – BTS CIEL (IR)

Titre du projet : Ballon stratosphérique

Référence interne : TW-5
Épreuve : E6 – Projet technique

Établissement : Lycée Touchard-Washington, Le Mans
Commanditaire : CNES & Planète Sciences Espace

Budget indicatif : 2 000 €

Encadrants pédagogiques :
• Philippe Simier
• Jilali Khamlach

Équipe étudiante :
• Étudiant 1 – Module télémétrie CSV – MAINFRAY Camille
• Étudiant 2 – Interface Qt & LoRa – STANISLAWSKI Erwän
• Étudiant 3 – Chaîne image SSTV & Web – BENIANI Ilyès
• Étudiant 4 – Détection événements & messaging – LANCIEN Dorian

Date de dépôt de la présente version : 23 avril 2025
Historique des révisions : Version 1.0 — première émission du dossier.

Lycée TOUCHARD WASHINGTON

1. Analyse détaillée

1.1 Contexte et périmètre personnel

Le projet **Ballon stratosphérique TW-5** vise à lancer, suivre et exploiter une nacelle scientifique depuis le lycée Touchard-Washington avec l'appui du CNES et de Planète Sciences. Deux ambitions structurent l'édition 2025 :

- (i) vérifier la loi des gaz parfaits en mesurant le diamètre du ballon par traitement d'images ;
- (ii) mettre en place une **liaison radio LoRa bidirectionnelle** capable d'émettre la télémétrie et de répondre à des requêtes issues du sol .

Positionnement dans l'équipe

Sur les **quatre étudiants** mobilisés, ma contribution (référencée *Étudiant 2* dans le cahier des charges) couvre l'intégralité du **segment sol LoRa / APRS** :

- développement de l'**interface Qt** permettant la saisie de messages et la visualisation des retours ;
- gestion de la **liaison LoRa montante & descendante** (module RA-02 + protocole AX.25/TNC2) ;
- réalisation de la **passerelle serveur** qui :
    - reformate les trames en APRS Weather,
    - les publie sur le réseau mondial **APRS-IS / aprs.fi**,
    - archive chaque échange dans une base **MySQL** (tables `machines`, `trames`).
      Ces attributions sont officialisées dans la rubrique « Répartition des fonctions » du cahier des charges .

Interfaces avec les autres sous-systèmes

- **Nacelle (Étudiant 1 & 4)** : réception des trames télémétriques toutes les 2 min et renvoi des réponses aux codes Q (*QSA*, *QTR*, *QSL*).
- **Chaîne photo + site Web (Étudiant 3)** : le serveur MySQL alimente également l'application PHP/Highcharts afin d'afficher en temps réel trajectoire, compteurs de trames et historique des messages.
- **Suiveur mobile** : un client ESP32 TTGO optionnel, paramétré en **KISS** sur port série, peut injecter des requêtes dans la passerelle lors de la phase de récupération .

Contraintes techniques et réglementaires

- **Bande 433 MHz**, puissance < 10 mW ERP et paramètre LoRa SF 7, afin de respecter la réglementation française Cahier des chargesCahier des charges.

- **Robustesse** : aucune tâche bloquante ; toutes les trames sont journalisées localement avant d'être poussées sur APRS-IS, conformément aux exigences de fiabilité du dossier commun Dossier Ballon CommunDossier Ballon Commun.

- **Budget global** 2 000 € et fenêtre de lancement fin mai 2025 ; mon périmètre se limite à un **Raspberry Pi 4 B** + HAT LoRa, un PC Linux pour l'interface Qt et un VPS Debian pour le serveur web/MySQL.

Objectifs de réussite

1. **Pas de perte de trame** entre la passerelle et aprs.fi (taux de réussite ≥ 99 %).

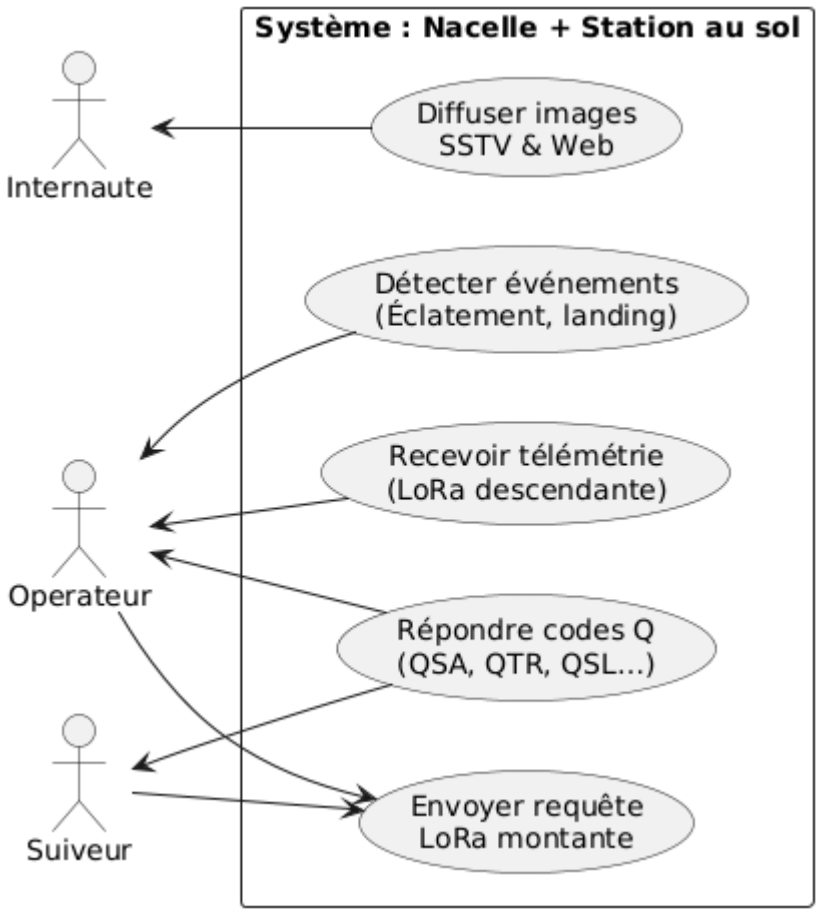
2. **Latence aller-retour** (requête → réponse) < 5 s dans un rayon de 50 km.

3. **Visualisation temps réel** : carte mise à jour < 30 s après réception d'une trame.

4. **Traçabilité** : 100 % des messages LoRa archivés avec horodatage en base.

En résumé, ma mission s'articule autour de la **chaîne sol "LoRa → APRS → Web"**, pivot indispensable pour valider en vol la fiabilité des communications et offrir, à l'équipe comme au public, une vision instantanée de la trajectoire et de l'état du ballon.

1.2 Cas d'utilisation pris en charge



Pourquoi seulement cinq ? Ce sont les interactions directement sous ma responsabilité (segment sol « LoRa → APRS → Web »), tels qu'assignés dans le cahier des charges et la répartition des fonctions étudiants.

1.3 Protocoles et flux de communication utilisés

Pour assurer la continuité entre la nacelle, la station au sol et Internet, trois couches de communication s'enchaînent : **LoRa → KISS/AX.25 → APRS-IS**.
L'objectif de cette section est de clarifier, sans jargon superflu, le rôle de chaque protocole et la façon dont ils s'imbriquent dans mon périmètre sol.

1. Liaison LoRa (433 MHz) – *couche radio physique*
La nacelle émet toutes les 2 minutes une trame LoRa modulée en 433 MHz (SF 7, CRC activé). Chaque trame transporte soit :

- un paquet **télémétrie** (latitude, longitude, altitude, capteurs),
- soit une **réponse** à une requête montante (QSA, QTR, QSL).

Le même canal est utilisé en sens inverse lorsque l'opérateur envoie une requête ; la station au sol relaie simplement la trame sans altération. Ces contraintes (bande, puissance < 10 mW ERP) sont imposées par le cahier des charges afin de rester conformes à la réglementation française .

2. Encapsulation KISS + AX.25 / TNC2 – *couche liaison de données*
Dès qu'une trame LoRa est reçue par la gateway, le flux d'octets entre dans la **pile radio-amateur** :

1. **KISS** (Keep It Simple Stupid) pose un en-tête minimal pour transporter les données entre le port série du module LoRa et l'application Qt.
2. **AX.25** fournit ensuite les champs *Source*, *Destination*, *Control* et *PID* indispensables pour que la trame soit reconnue sur le réseau APRS.
3. Pour la lisibilité et la compatibilité réseau, la trame est finalement convertie au format texte **TNC2** (ex. `F4KMN-8>APRS,WIDE1-1:!4803.45N/00113.56E_...`).

Cette chaîne de conversions est implantée dans les modules `KISSHandler`, `AX25Converter` et `Interface` de mon application Qt.

3. APRS-IS – *couche application, diffusion Internet*
Une fois au format TNC2, la trame est ouverte sur le monde :

- La passerelle établit un socket TCP vers un serveur **APRS-IS** (port 14580).

- Après la séquence de login (`user F4KMN pass -1 vers …`), chaque ligne est poussée en temps réel.

- Le site **aprs.fi** l'indexe aussitôt, offrant au public la position du ballon ou le message Q-code retourné.

Tout upload fait l'objet d'un log simultané dans la base MySQL (`tables machines / trames`) pour garantir la traçabilité, même si la connexion Internet est interrompue ; la passerelle rejoue alors les INSERT manquants à la reconnexion .

4. Gestion des Q-codes – *procédure requête ↔ réponse*
Les requêtes montantes se limitent volontairement aux trois codes :

- **QSA ?** → force du signal (RSSI / SNR) ;

- **QTR ?** → heure locale de la nacelle ;

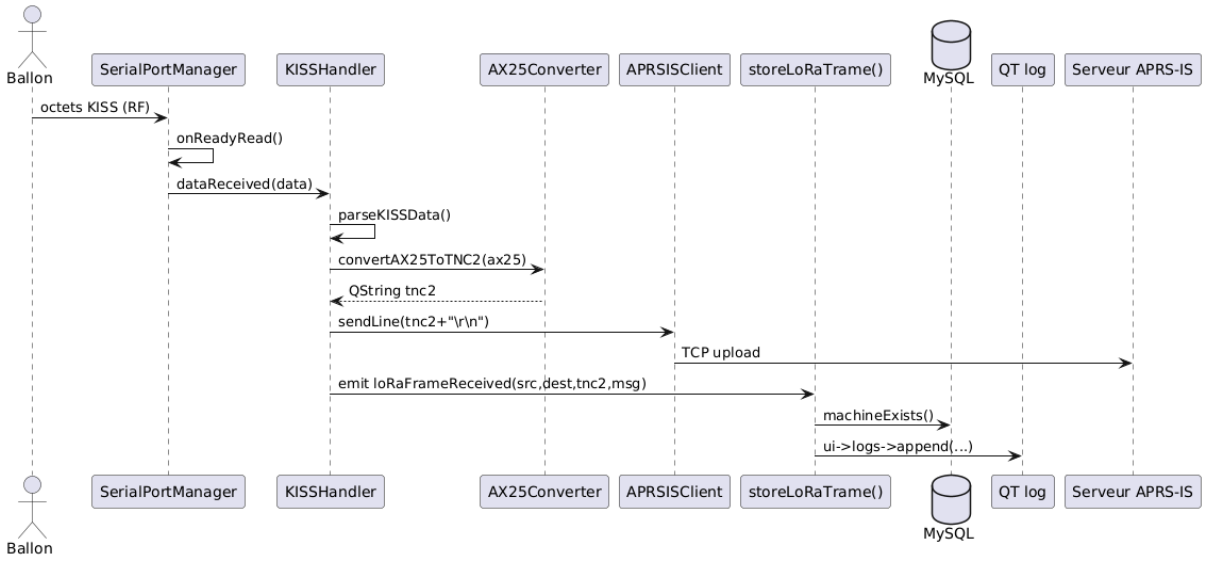
- **QSL** → accusé de réception.

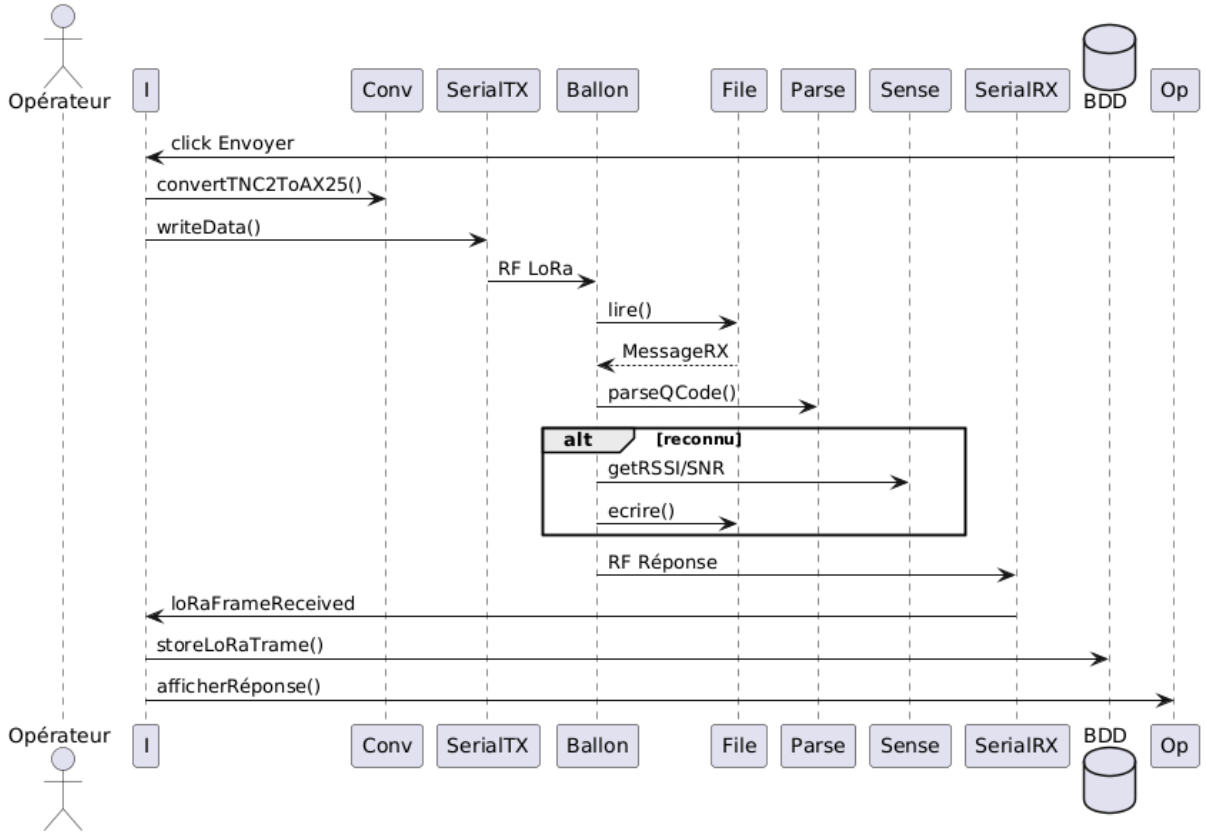
La nacelle ne répond que si :
(1) le message termine par ? pour QSA ou QTR,
(2) la destination est F4KMN-8,
(3) la source fait partie de la *whitelist* (F4KMN, F4LTZ).
Ces règles, codées dans `parseQCode()` sur la nacelle, évitent toute pollution radio ou réponse fantôme.

1.4 Diagrammes de séquence

> Les messages sont nommés **fonctionnellement**, sans préjuger de classes C++ ou de sockets.

DS-01 – « Réception & push APRS »





2. Conception

Après avoir précisé le besoin fonctionnel et les principales interactions dans l'analyse détaillée, cette partie décrit les choix qui transforment ces exigences en un système opérationnel pour le segment sol : **LoRa ➜ passerelle ➜ APRS-IS ➜ web**.

Objectif	**Concrètement, cela implique…**
Simplicité d'intégration	S'appuyer sur des composants techniques déjà maîtrisés afin de concentrer l'effort sur la logique métier plutôt que sur l'infrastructure.
Robustesse temps réel	Acheminer, journaliser et, au besoin, réémettre chaque trame en quelques secondes, sans perte.
Traçabilité & ouverture	Conserver l'historique complet en SQL pour permettre statistiques, visualisation publique et relecture post-vol.

Ces trois axes guident tous les choix décrits ci-dessous.

2.1 Choix technologiques et justifications

Brique	Décision	Pourquoi ?
Chaîne radio embarquée	Arduino Uno + module LoRa RA-02 (SX1278) conservés tels quels. Le firmware existant émet du **KISS** à 115 200 bauds.	Réécrire le firmware (TNC2 ou ASCII brut) aurait dépassé le planning ; garder la pile radio intacte limite les risques.
Passerelle applicative (station sol)	Application **Qt 6 / C++** sous Debian 12, découpée en trois couches : 1. **QSerialPort** : lecture du flux KISS, décapsulation AX.25, normalisation des champs (indicatif, payload…). 2. **QtSql** : requêtes préparées vers MariaDB. 3. **Service REST** diffusant les données vers le front web.	Qt propose des API multiplateformes robustes (série, SQL, réseau) déjà connues de l'équipe ; la segmentation clarifie la maintenance.
Base de données	**MariaDB 10.11** installée sur le serveur du lycée.	Solution SQL libre, familière à l'administrateur ; options de réplication et de sauvegarde simples.
Publication APRS-IS	Lorsqu'une trame contient une position valide, elle est convertie côté Qt en chaîne **TNC2** puis envoyée via socket TCP vers un serveur APRS-IS.	Le protocole KISS reste cantonné au lien USB local ; la communauté radio-amateur continue de recevoir le format standard.
Visualisation web	Front HTML / CSS / JavaScript / PHP maison, hébergé sur un second serveur du lycée.	Développer un visualisateur sur mesure répond mieux aux besoins pédagogiques et permet d'apprendre une stack JS complète.
Non-utilisation de Leaflet	Leaflet a été écarté.	Choix volontaire : concevoir la cartographie "from scratch" pour renforcer la compréhension des projections, du rendu canvas et de l'optimisation front-end.

Synthèse

En combinant des briques éprouvées (Qt, MariaDB, protocoles KISS / TNC2) avec un développement ciblé sur la logique métier et la visualisation, la solution atteint :

- une **mise en œuvre rapide** (réemploi maximal),
- une **fiabilité** adaptée aux contraintes temps réel du vol,
- une **ouverture** des données pour l'exploitation scientifique et la communication grand public.