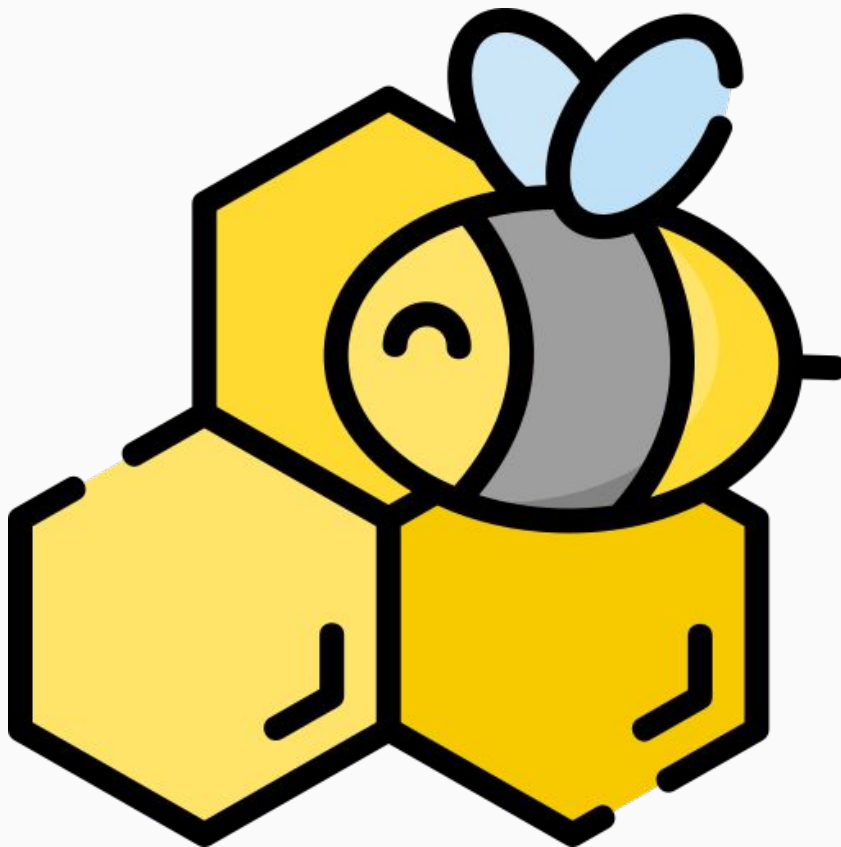


Projet BTS SN 2019

Bee-Honey-T

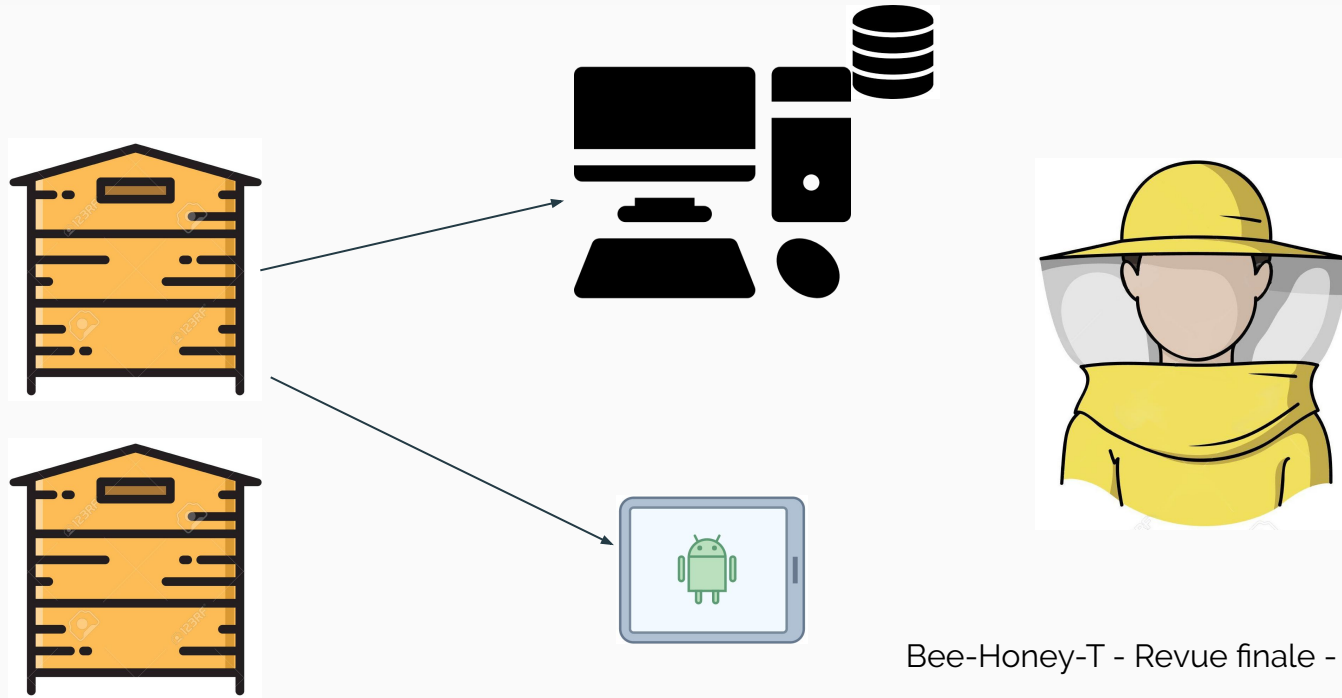
Revue Finale



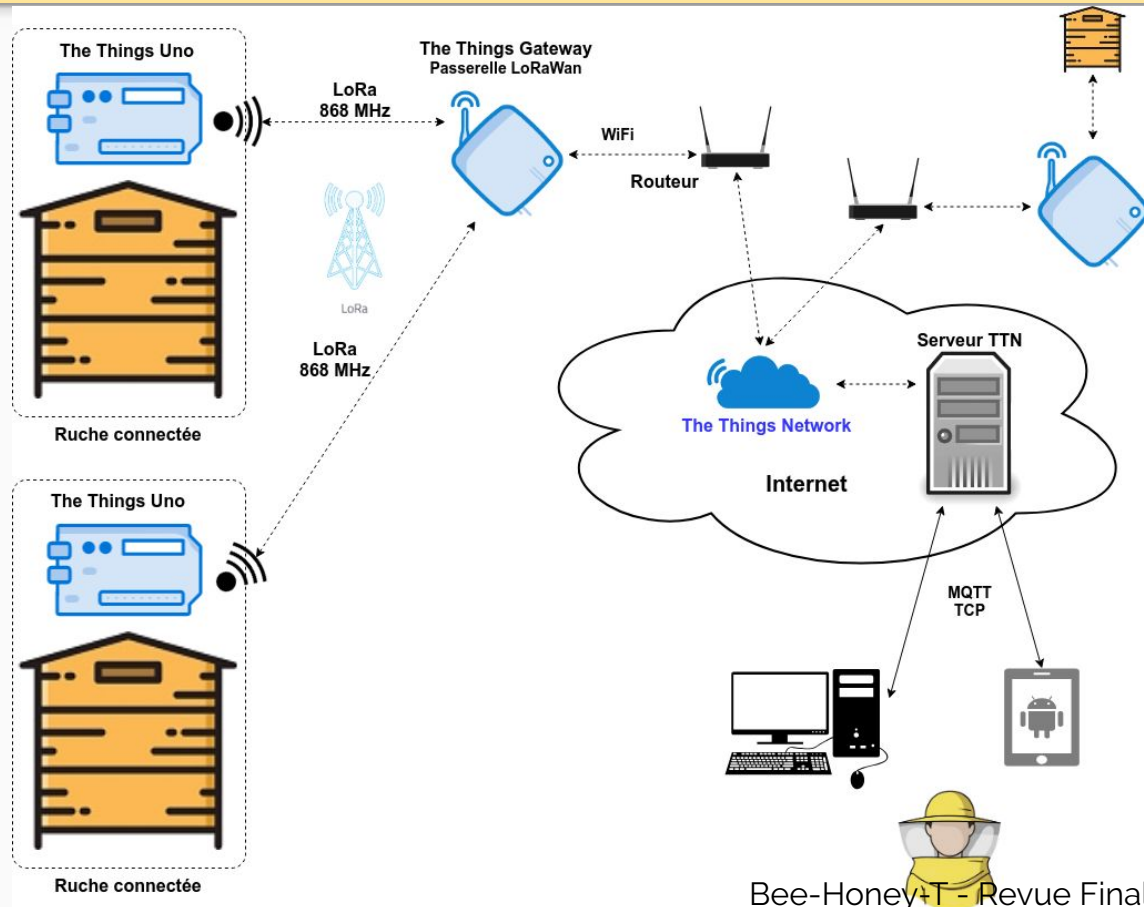
Sommaire

- ❏ Expression du besoin
- ❏ Fonctionnement du système
- ❏ Gestion de projet
- ❏ Ressources
- ❏ Spécifications techniques
- ❏ IHM
- ❏ Application *Desktop*
- ❏ Les classes
- ❏ Base de Données
- ❏ Gestion des alertes
- ❏ Conclusion

Expression du besoin



Principe de fonctionnement du système



Planification

The screenshot displays a task planning application interface with three columns: "A faire IR", "En cours IR", and "Terminé IR". Each column contains task cards with progress bars, titles, descriptions, dates, and status icons.

A faire IR

- Gérer les ruches : Paramétrer les alertes (27 mai, F)
- Créer les classes alertes et configurer les différentes alertes (F)
- Consulter les données d'une ruche (poids, niveau de charge, tension et courant de la batterie) (27 mai, F)
- Enregistrer les données collectées (27 mai, F)
- Déclencher les alertes (27 mai, F)
- Gérer les ruches (27 mai, LC)

En cours IR

- Gérer la planification des tâches (27 mai)
- Gérer les ruches : Paramétrer une nouvelle ruche (27 mai, E)
- Consulter les données d'une ruche (température, humidité, pression atmosphérique, et ensoleillement) (25 mars, E)
- Consulter les données d'une ruche (27 mai, LC)

Terminé IR

- Réalisation d'une maquette IHM PC (F)
- Recevoir les données des ruches (22 mars, E)
- Création de la convention de nommage (E)
- Tests de mise en œuvre de MQTT (3/3, E, F)
- Mettre en œuvre MQTT : ajout des différents répertoire / mise en place du système de requête et de suscribe (27 mai, F)
- Créer la base de donnée (15 mars, F)

+ Ajouter une autre carte

Répartition des tâches

Répartition des Tâches		
ROSSI Enzo	LAURAIN Clement	MELLAH Florentin
<ul style="list-style-type: none"><input type="checkbox"/> Gérer les ruches : Paramétrer une nouvelle<input type="checkbox"/> Consulter les données d'une ruche (température, humidité, pression atmosphérique et ensoleillement)<input type="checkbox"/> Recevoir les données des ruches	<ul style="list-style-type: none"><input type="checkbox"/> Gérer les ruches<input type="checkbox"/> Consulter les données d'une ruche<input type="checkbox"/> Lire les données à partir de la base de données	<ul style="list-style-type: none"><input type="checkbox"/> Gérer les ruches : Paramétrer les alertes<input type="checkbox"/> Consulter les données d'une ruche (poids, niveau de charge, tension et courant de la batterie)<input type="checkbox"/> Enregistrer les données collectées<input type="checkbox"/> Déclencher les alertes

Ressources techniques

IDE : Qt



Modules Qt :

Qwt

QtSQL

QtMQTT



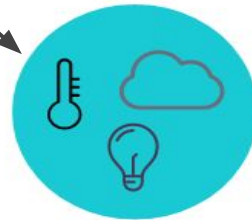
Spécifications techniques

MQTT & TTN

Protocole MQTT

Les capteurs de l'objet connecté envoient (**publish**) des données sur un **Topic** sur le **broker MQTT**

Les clients MQTT qui ont fait un **subscribe** à un **Topic** reçoivent les données des capteurs



Publisher

message



message



Subscriber

message



Subscriber

Virtual channel
Topic

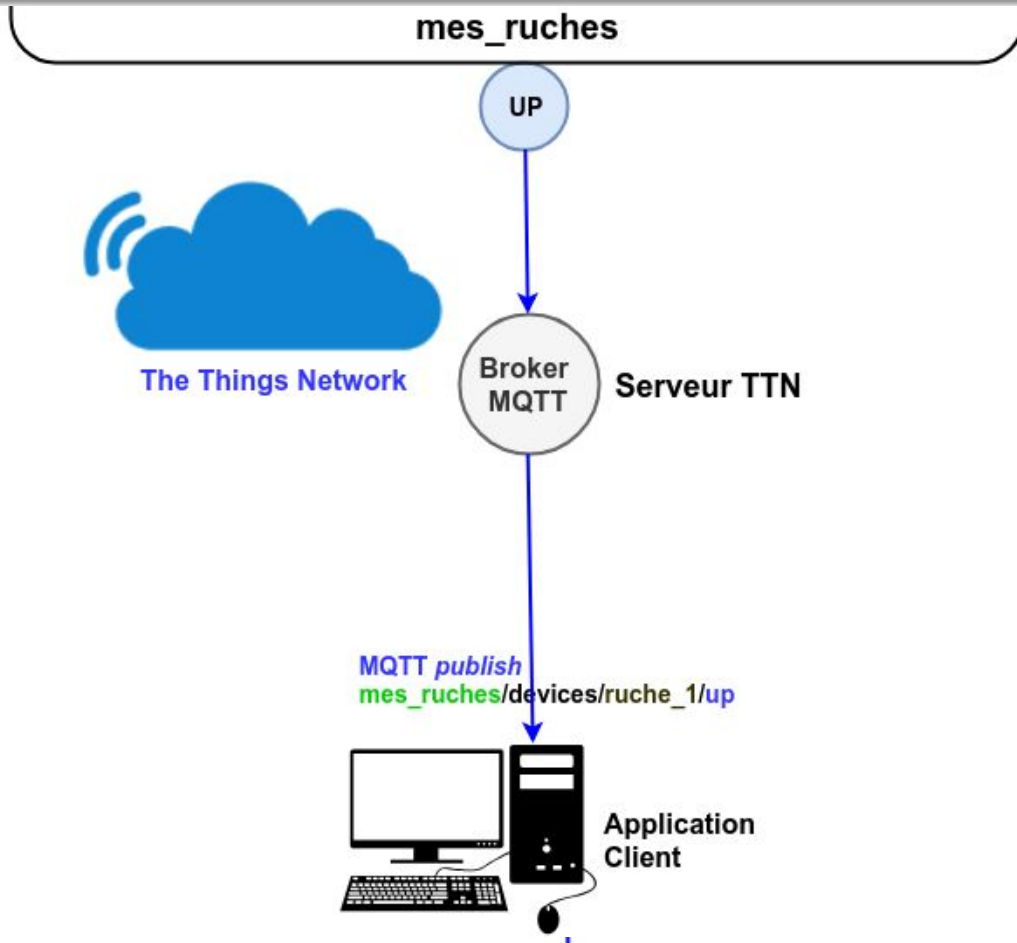
Le **broker** MQTT fait des **publish** des données du **Topic** vers les abonnés (**subscriber**)



THE THINGS
N E T W O R K

UN RÉSEAU
LoRaWAN
COMMUNAUTAIRE
OPEN SOURCE

MQTT / The Things Network

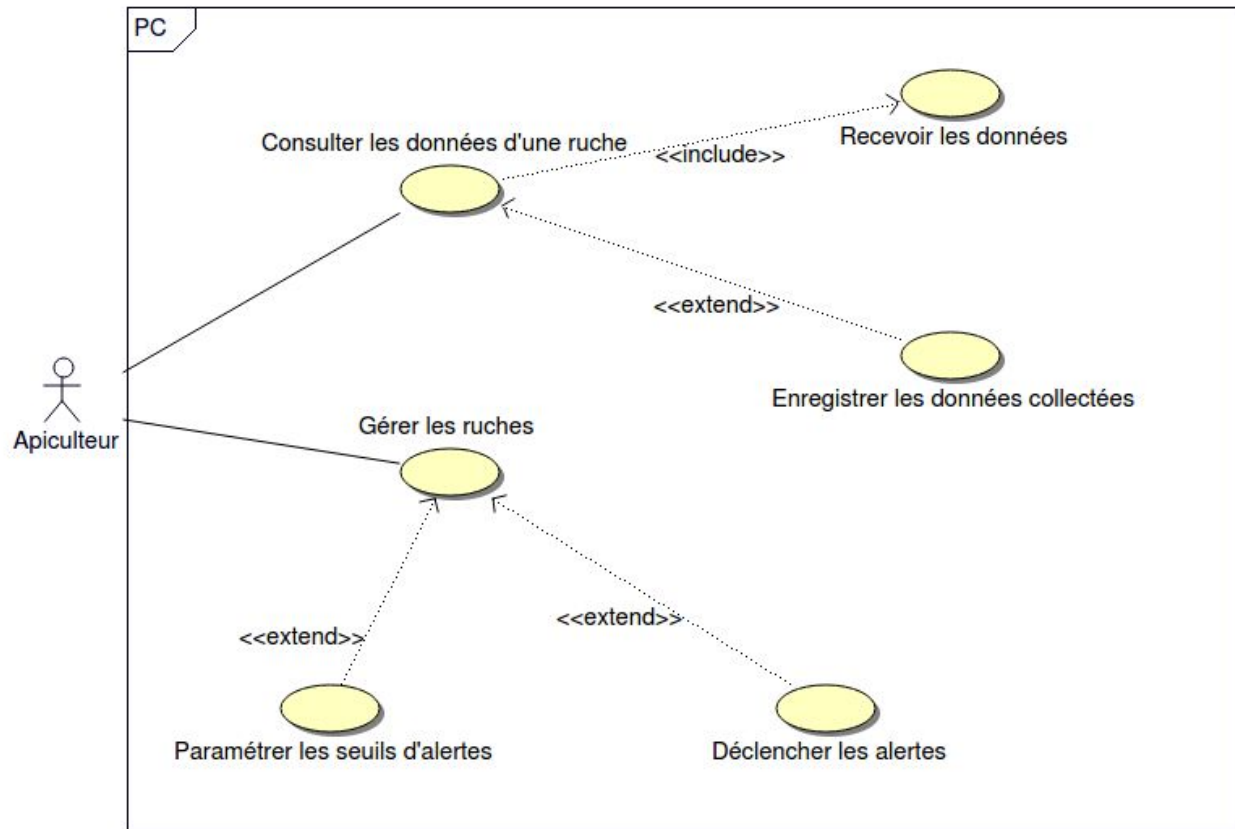


- **AppID** : Identifie le parc des objets connectés
- **DevID** : Identifie l'objet connecté
- **Up** : Sens Ruche vers Client
- **Topic** : "Sujet" auquel s'abonner où sont envoyés les données ou où récupérer les données

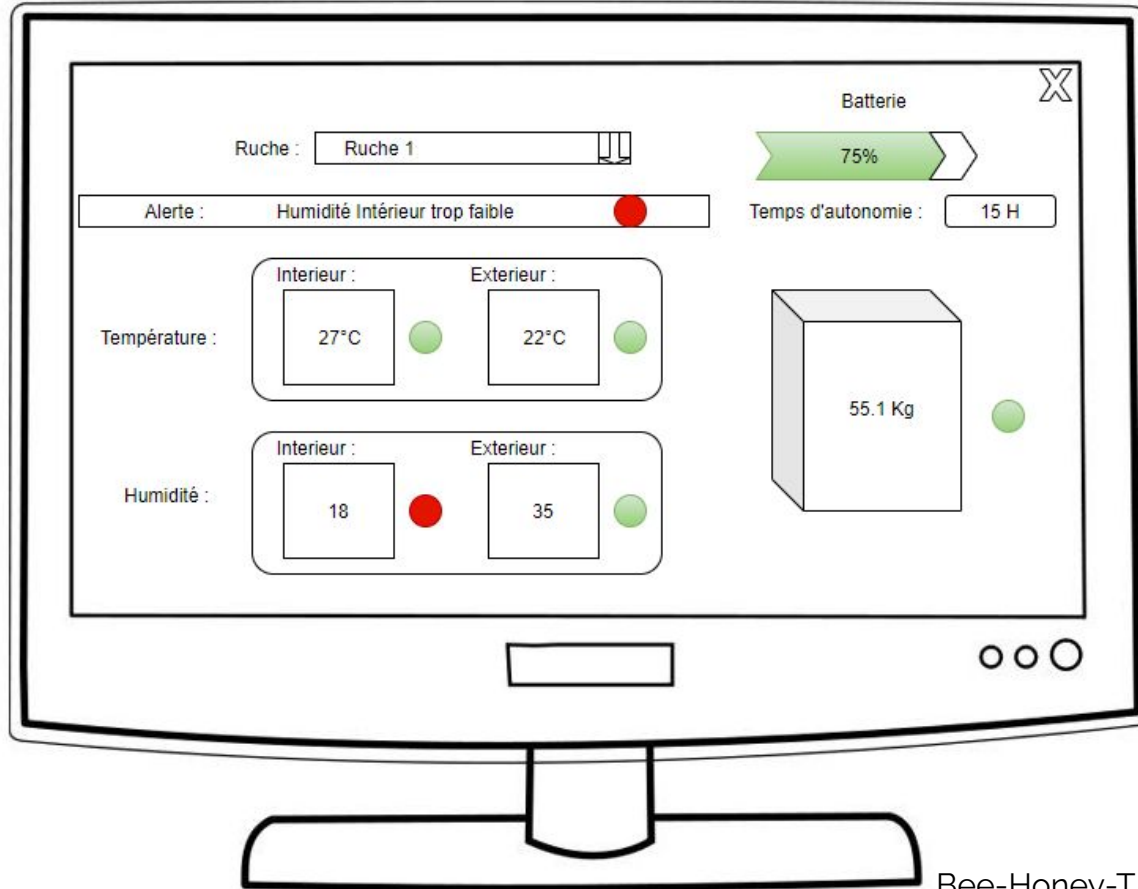
Les données

▲ 14:21:00	84822	6	payload: 00 vol: <i>false</i>
▲ 14:19:58	84821	5	payload: 01 BB ensoleillement: 443
▲ 14:19:46	84820	4	payload: 09 EC 11 08 04 0A humidite: 43.6 pression: 1034 temperature: 25.4
▲ 14:19:34	84819	3	payload: 0A 28 10 FE humidite: 43.5 temperature: 26
▲ 14:19:22	84818	2	payload: 43 80 poids: 17152
▲ 14:19:10	84817	1	payload: 00 5D 00 2A 3A charge: 58 courant: 0.205078125 tension: 0.4541015625

Diagramme des cas d'utilisation



Maquette IHM



Application Desktop

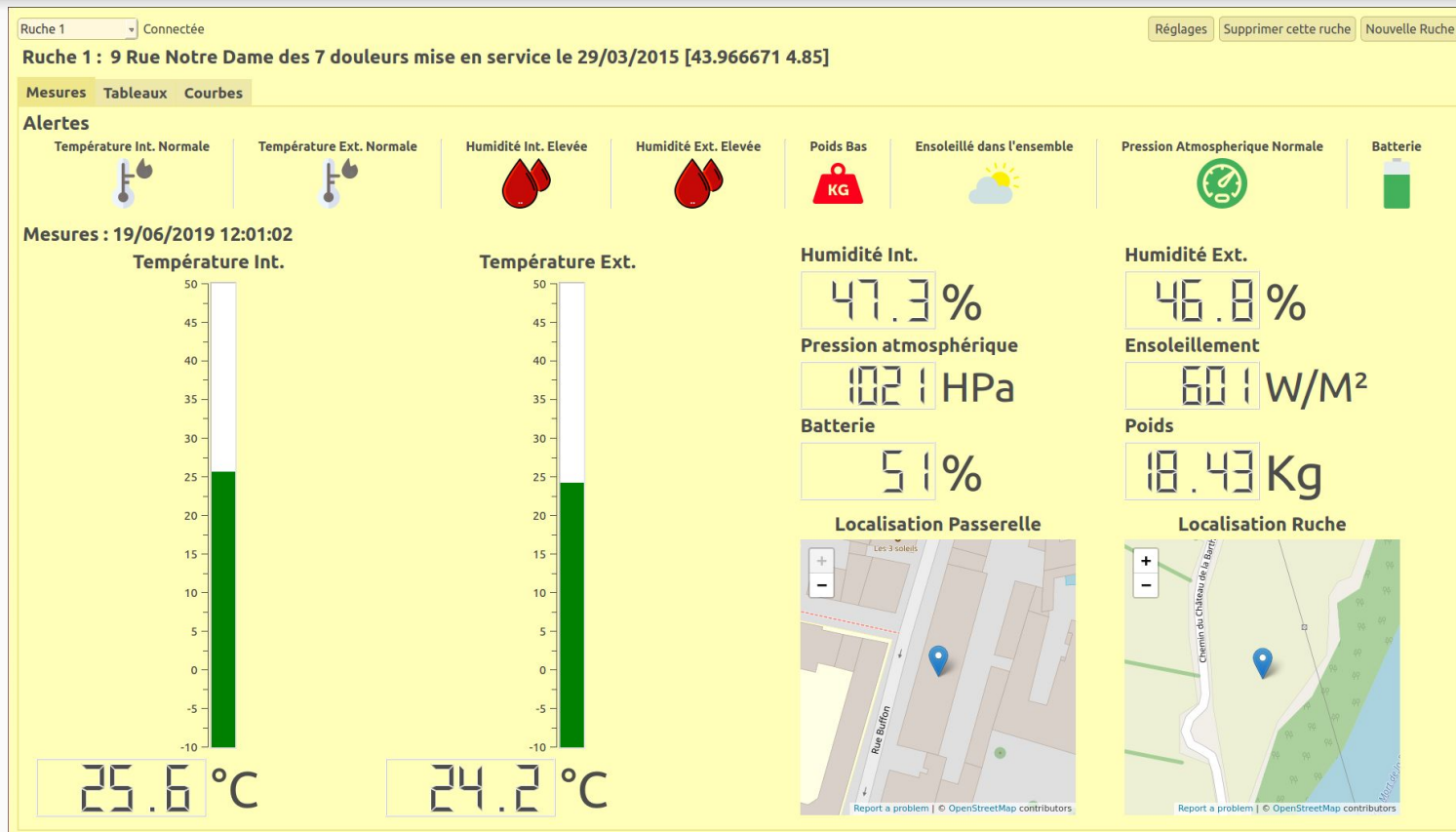
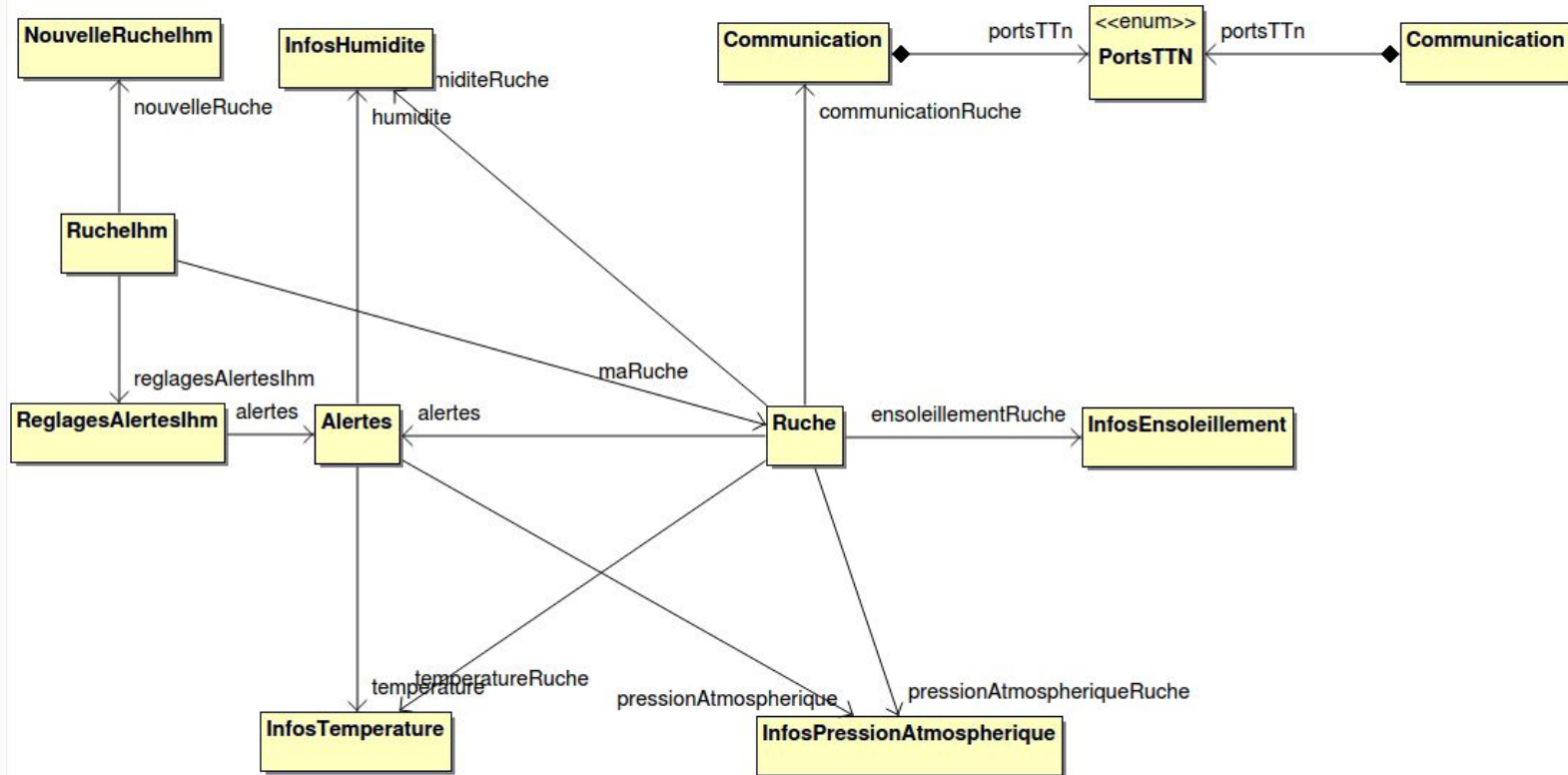
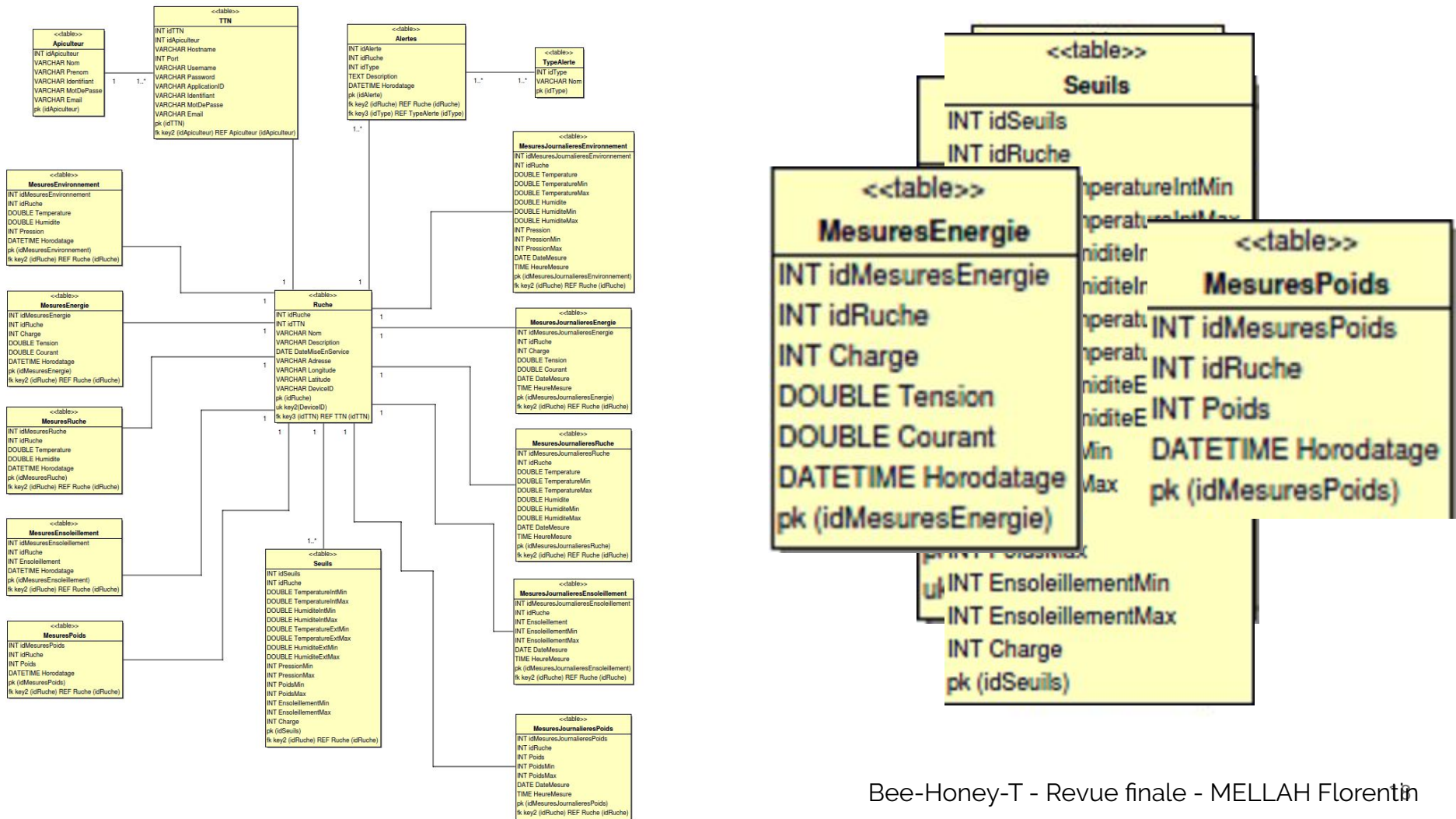


Diagramme de classe



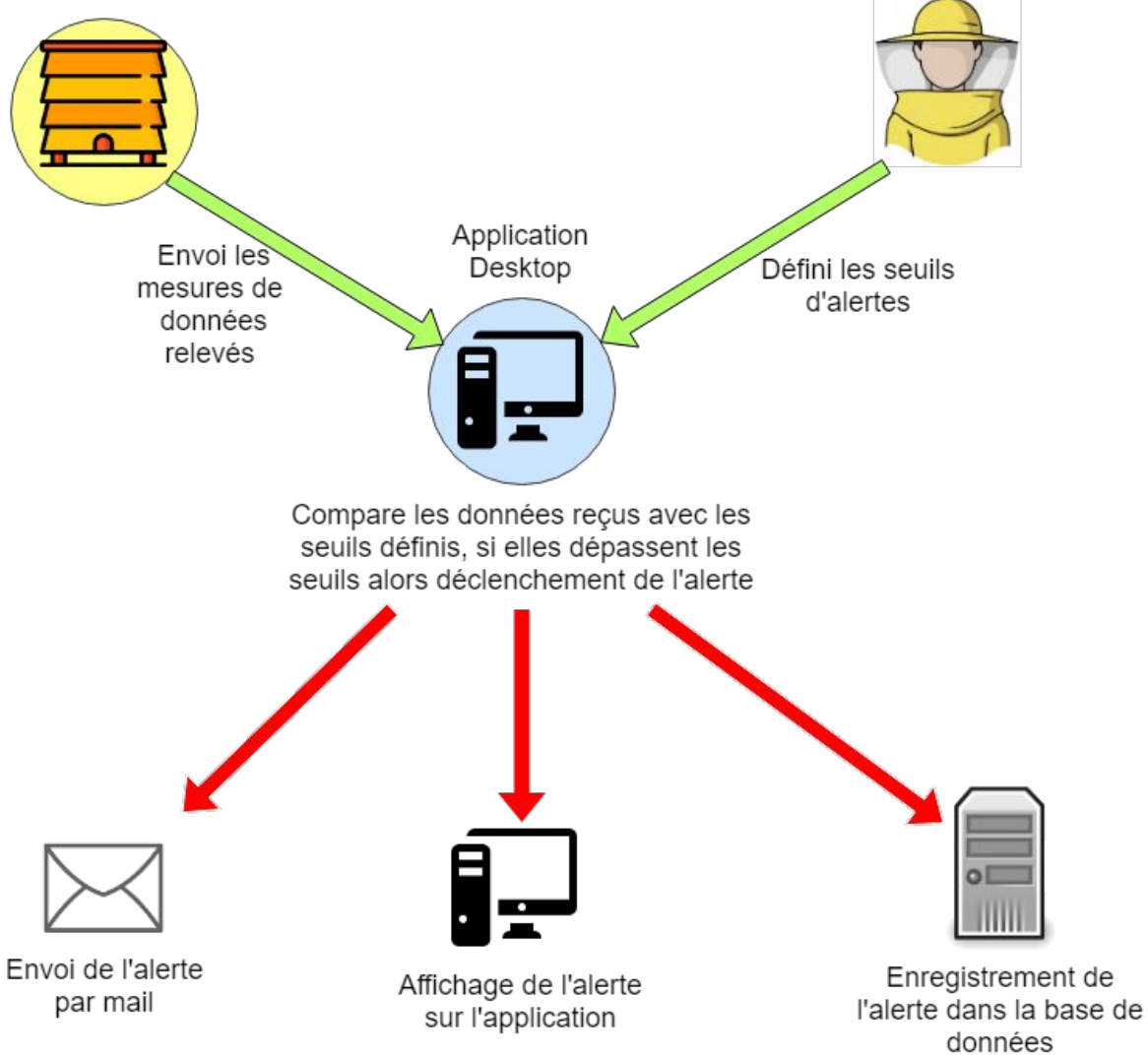
Base de données



Requête SQL

```
QString requete = "INSERT INTO MesuresPoids(idRuche,  
Poids, Horodatage) VALUES ('" + donneesRucheTTN.at(0) +  
"', '" + donneesRuche.poids + "', '" +  
dateTimePortMeasurePoids.toString("yyyy-MM-dd HH:mm:ss")  
+ "')";
```

Gestion des Alertes



Modification des seuils

Ruche 2019 - Réglages des seuils

Température Intérieur Minimum :	<input type="text" value="25"/>	°C
Température Intérieur Maximum :	<input type="text" value="35"/>	°C
Humidité Intérieur Minimum :	<input type="text" value="20"/>	%
Humidité Intérieur Maximum :	<input type="text" value="30"/>	%
Température Extérieur Minimum :	<input type="text" value="5"/>	%
Température Extérieur Maximum :	<input type="text" value="35"/>	°C
Humidité Extérieur Minimum :	<input type="text" value="20"/>	%
Humidité Extérieur Maximum :	<input type="text" value="35"/>	%
Pression Atmosphérique :	<input type="text" value="1000"/>	hPa
Poids Minimum :	<input type="text" value="35"/>	kg
Poids Maximum :	<input type="text" value="100"/>	kg
Ensoleillement :	<input type="text" value="10"/>	W/m ²

OK

Classe Infos

InfosPoids

<ul style="list-style-type: none">- poids : double- horodatagePoids : QString
<ul style="list-style-type: none">+ InfosPoids(inout parent : QObject)+ ~InfosPoids()+ getPoids() : double+ setPoids(in poids : double) : void+ getHorodatagePoids() : QString+ setHorodatagePoids(in horodatagePoids : QString) : void+ poidsEnvoye(in poids : double, in : QString) : void+ traiterNouveauPoids(in nouveauPoids : QString, in horodatagePoids : QString) : void

InfosBatterie

<ul style="list-style-type: none">- tension : double- courant : double- charge : double- horodatageBatterie : QString
<ul style="list-style-type: none">+ InfosBatterie(inout parent : QObject)+ ~InfosBatterie()+ getMesureBatterieCharge() : double+ getMesureBatterieCourant() : double+ getMesureBatterieTension() : double+ getHorodatageBatterie() : QString+ setHorodatageBatterie(in horodatageBatterie : QString) : void+ setMesuresBatterie(in tension : double, in courant : double, in charge : double) : void+ tensionEnvoye(in tension : double, in : QString) : void+ courantEnvoye(in courant : double, in : QString) : void+ chargeEnvoye(in charge : double, in : QString) : void+ traiterNouvelleMesureCourant(in nouveauCourant : QString, in horodatage : QString) : void+ traiterNouvelleMesureCharge(in nouvelleCharge : QString, in horodatage : QString) : void+ traiterNouvelleMesureTension(in nouvelleTension : QString, in horodatage : QString) : void

Les classes des alertes

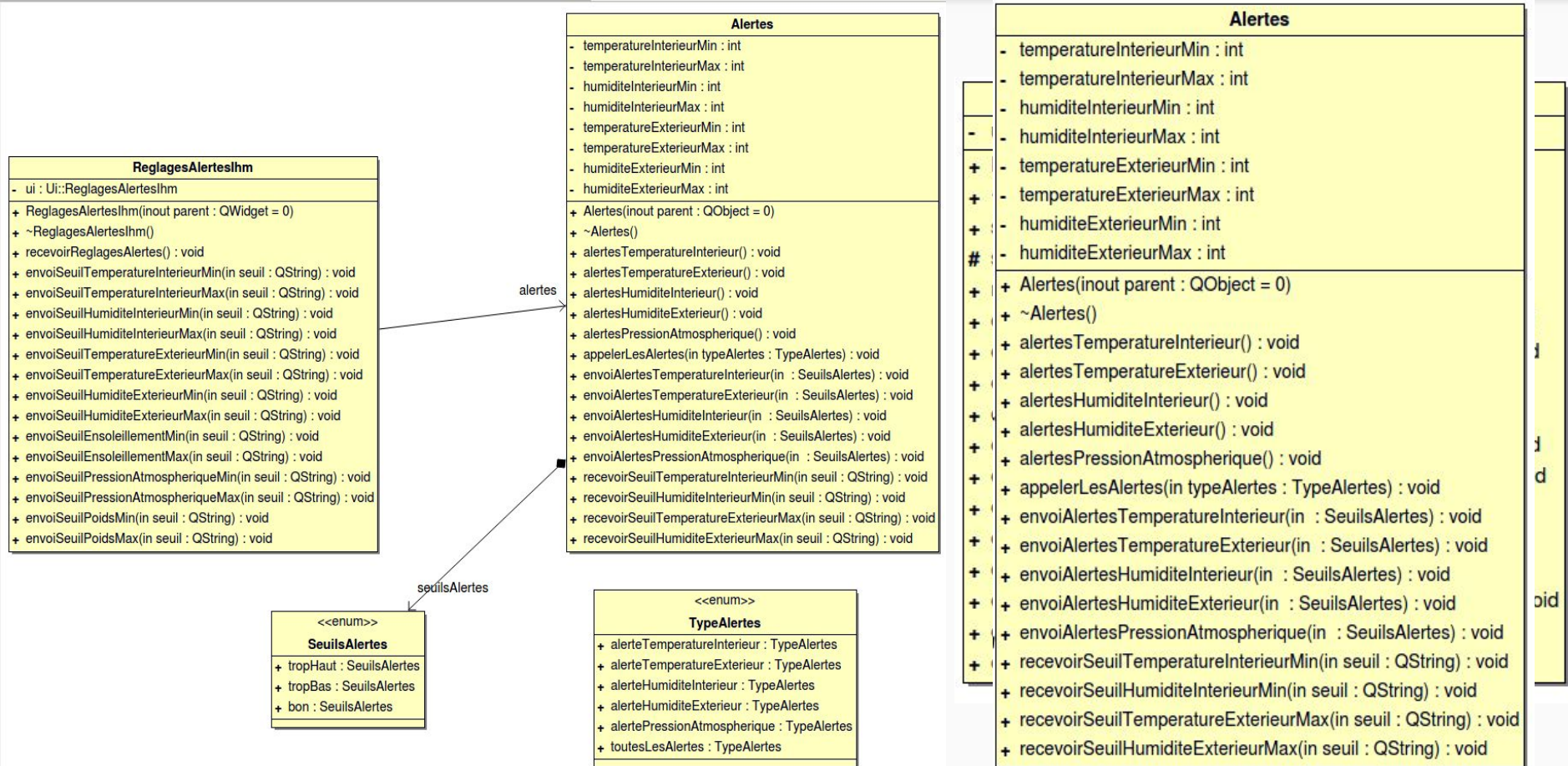
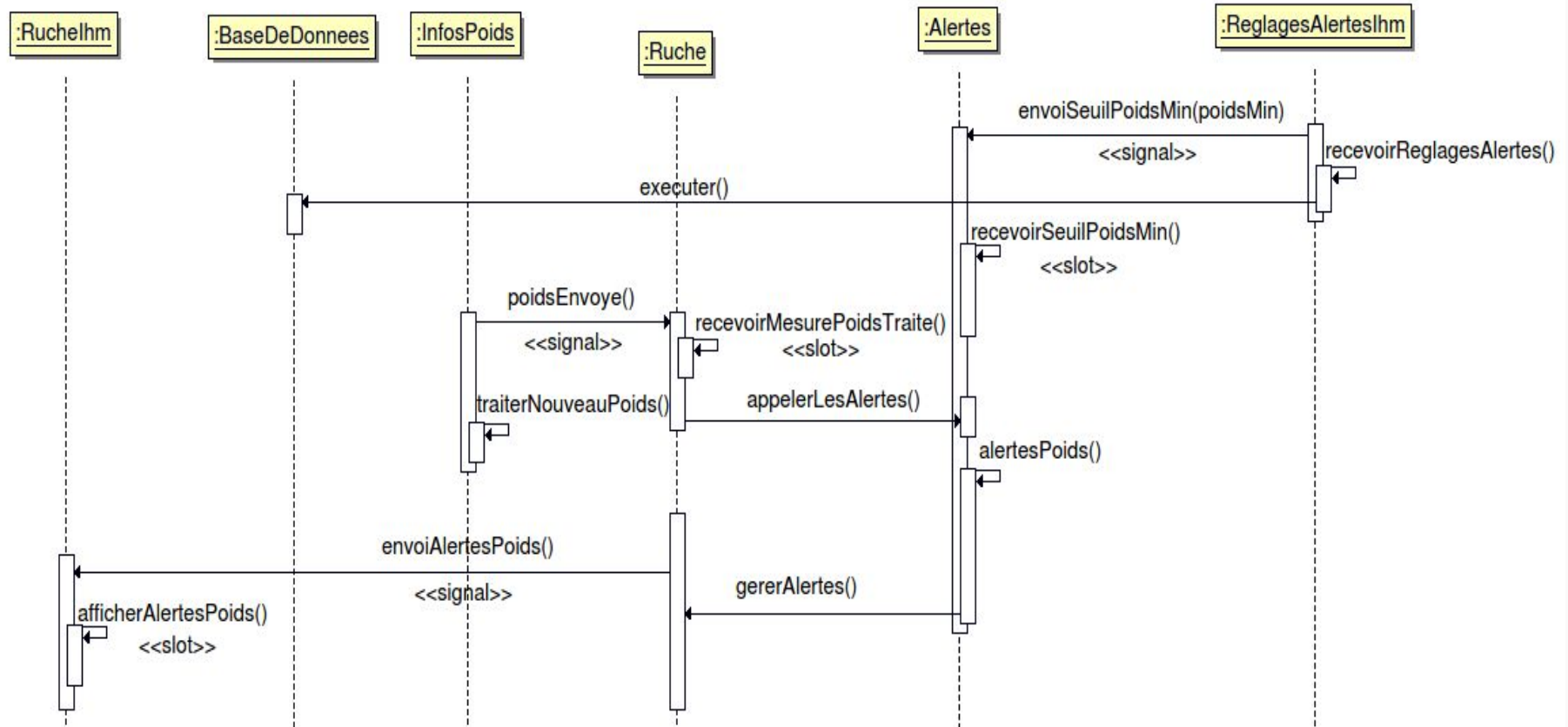


Diagramme de Séquence : Régler les seuils d'alerte et déclencher les alertes



Tests de validation

Désignation	Objectif attendu	Résultat
Gérer les ruches : Paramétrer les alertes	Régler les seuils minimum et maximum de toutes les mesures	Possibilité d'effectuer les réglages en cliquant sur le bouton "Réglages"
Consulter les données d'une ruche (poids, niveau de charge, tension et courant de la batterie)	Visualiser le poids, le niveau de charge de la batterie	Possibilité de visualiser le pourcentage de batterie ainsi que le poids de la ruche en kilogramme
Enregistrer les données collectées	Enregistrement des mesures de poids et de batterie dans la base de données	Les données sont enregistrées dans la base de données, possibilité de les visualiser grâce à l'onglet "Courbe"
Déclencher les alertes	Afficher lorsqu'une mesure dépasse le seuil maximum ou minimum	Visualisation des alertes sur l'interface homme/machine dans la partie spécifique aux alertes

Technologie de télécommunication longue portée



Bande Passante	500 - 125 KHz	100 Hz
Débit	290 bps - 50 Kbps	100 bits/sec
Message Max/jour	Illimité	140 / jour
Puissance Émission	+14 dBm	+14 dBm
Durée de vie de la batterie	105 mois	90 mois
Distance	5 km (urbain), 15 km (rural)	10 km (urbain), 50 km (rural)

Conclusion

Itération 1.1 : ajouter l'anti vol ainsi que l'amélioration de gestion des alertes.

Merci de votre attention