



Dossier Personnel

BTS CIEL SESSION 2025 - IR

E6 – PROJET TECHNIQUE

Système de Surveillance météorologique connecté



Lycée : Touchard Washington		Ville : LE MANS	
Nom du projet :	Système de surveillance météorologique connecté		
N° du projet :	TW3		
Projet nouveau	Oui <input type="checkbox"/> Non <input type="checkbox"/>	Projet interne : Oui <input type="checkbox"/> Non <input type="checkbox"/>	
Délai de réalisation	150 heures	Statut des étudiants : Formation initiale <input type="checkbox"/> Apprentissage <input type="checkbox"/>	
Spécialité des étudiants	ER <input type="checkbox"/> IR <input type="checkbox"/> Mixte <input type="checkbox"/>	Nombre d'étudiants : 3 étudiants	
Professeurs responsables	Philippe CRUCHET, François MARTIN, Anthony LE CREN, Jilali KHAMLACH, Saïd LAHSIKA.		

Sommaire

1. Cas d'utilisation.....	4
1.1. Traiter les données en provenance des capteurs.....	4
1.2. Stocker les données météorologiques.....	5
1.3. Diffuser les données météorologiques en temps réel.....	6
2. Protocoles.....	7
2.1. JSON.....	7
2.2. WebSocket.....	8
3. Outils.....	8
3.1. QTCreator.....	8
3.2. MariaDB.....	10
3.3. RTL_433.....	10
4. Présentation de GitHub.....	10
5. Diagrammes de séquences.....	12
5.1. Traiter les données.....	12
5.2. Stocker les données.....	13
5.3. Diffuser les données.....	16
6. Diagramme de classe.....	17
7. Fiche de test.....	18
8. Diagramme de Gantt.....	25
9. Compétences acquises.....	25
10. Perspectives d'améliorations.....	26

Avant de présenter mon projet en précision, j'ai d'abord décidé de représenter l'objectif de mon projet sous la forme d'un diagramme de séquence. Celui-ci reprend les tâches que j'ai pu réaliser mais de manière globale, aucune classe fournie par les outils et seulement les méthodes les plus importantes y sont représentées.

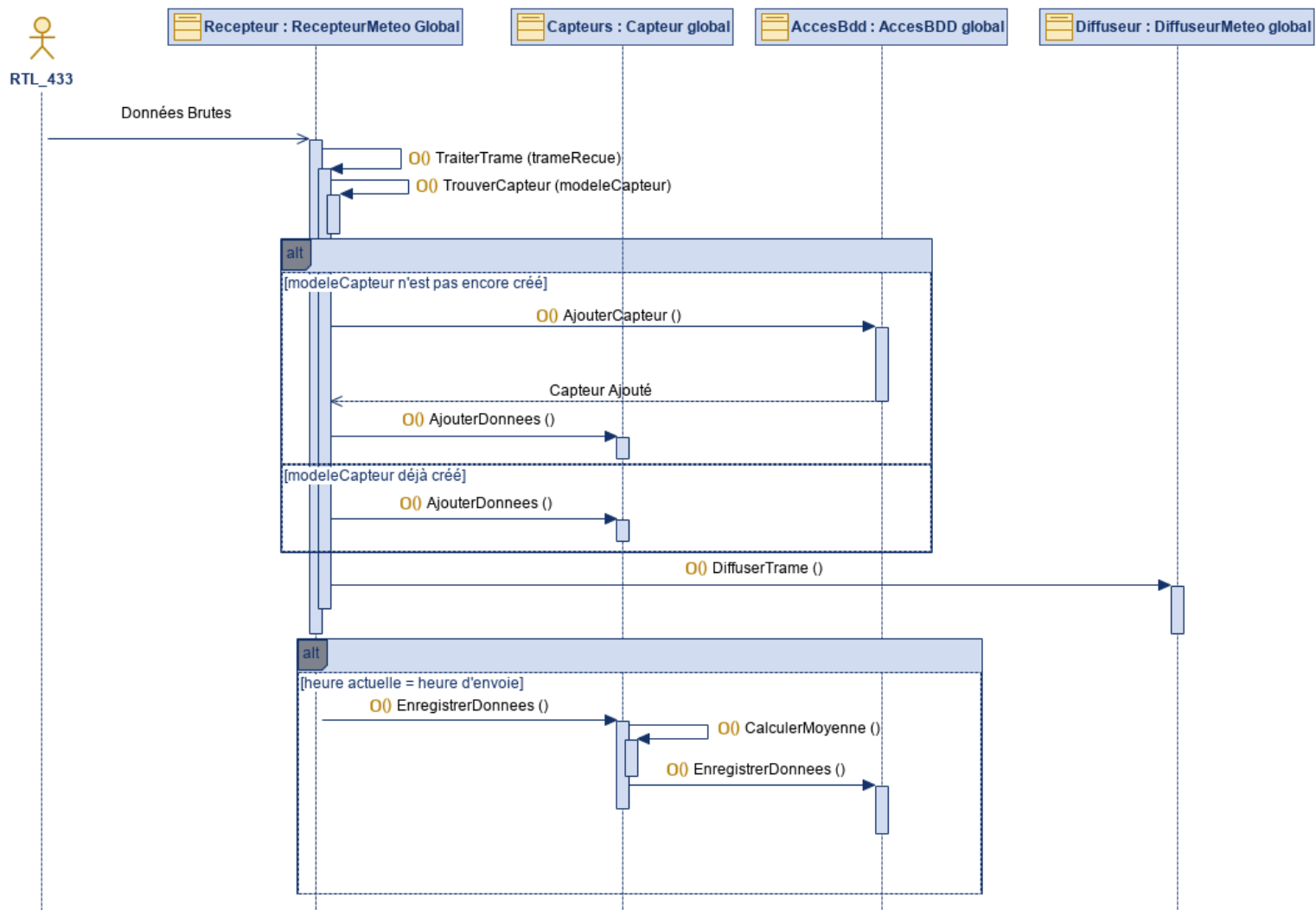


Image 1: Diagramme de séquence global

On peut donc voir que le récepteur va traiter la trame après avoir reçu les données des capteurs. Après cela il va identifier le capteur à l'origine de la trame et en fonction de celui-ci il va soit faire une demande d'ajout à la classe AccesBDD (ajoute le capteur dans la base de données) et ajouté les données dans la classe Capteurs, ou seulement demander à les ajouter dans la classe. Et après cela il va diffuser cette trame aux clients websockets.

Et lorsque il est l'heure que les données soient enregistrées il va faire une demande d'enregistrement à la classe capteur. Celle-ci va alors calculer la moyenne des données qu'elle à accumulée durant la dernière demi-heure et ensuite les faire enregistrer dans la BDD par le biais de la classe AccesBDD.

1.Cas d'utilisation

1.1.Traiter les données en provenance des capteurs

Description du cas d'utilisation	<p>Ce cas d'utilisation permet de recevoir les trames JSON issues de différents transmetteurs météorologiques via le programme RTL_433, de les analyser pour identifier les différents capteurs et en extraire les données météorologiques, puis d'exécuter deux actions principales :</p> <ul style="list-style-type: none"> - Alimenter une base de données en appelant le cas d'utilisation "Stocker les données météorologiques". - Transmettre les données aux clients Android en appelant le cas d'utilisation "Diffuser les données météorologiques en temps réel".
Pré-conditions	<p>Le programme RTL_433 est opérationnel et transmet les données JSON au récepteur météo.</p> <p>La configuration des capteurs (type, identifiant unique) est enregistrée dans le système, le format de la trame pour chaque transmetteur est connu.</p>
Scénario principal	<p>Réception des données:</p> <ul style="list-style-type: none"> ➤ Le système reçoit une trame JSON depuis RTL_433. ➤ La trame est analysée pour identifier le capteur à l'origine des données (via son identifiant ou un champ spécifique). Exemple de trame : {"time" : "2025-01-15 10:43:49", "model" : "Nexus-TH", "id" : 130, "channel" : 3, "battery_ok" : 1, "temperature_C" : 22.100, "humidity" : 31} <p>Extraction des données:</p> <ul style="list-style-type: none"> ➤ En fonction du type de transmetteur météo, les éléments de la trame (température, humidité, précipitations, vent, etc.) sont extraits et convertis si nécessaire (unités, formats). <p>Stockage des données</p> <ul style="list-style-type: none"> ➤ Pour chaque grandeur extraite (température, humidité, précipitations, vent), le système appelle le cas d'utilisation "Stocker les données météorologiques". <p>Diffusion des données</p> <ul style="list-style-type: none"> ➤ Les données extraites sont transformées en une trame JSON standardisée pour diffusion. ➤ Le système appelle le cas d'utilisation "Diffuser les données météorologiques en temps réel". Les données sont alors envoyées à tous les clients Android connectés.
Scénario alternatif	<p>Si une trame JSON est non reconnue (ne provient pas d'un de nos capteurs).</p>

1.2. Stocker les données météorologiques

Description du cas d'utilisation	Ce cas d'utilisation permet de gérer l'enregistrement des données météorologiques extraites dans une base de données locale pour chaque grandeur mesurée (température, humidité, précipitations, vent). Les données stockées dans la base sont utilisées pour le cas d'utilisation « Afficher des données historiques ».
Pré-conditions	La base de données locale est accessible et opérationnelle La structure des tables est correctement configurée: une table par type de grandeur, incluant les colonnes suivantes: <ul style="list-style-type: none"> • Identifiant du capteur (entier) • Valeur mesurée (type selon la grandeur) • Horodatage
Post-conditions	Les données météorologiques valides sont stockées dans la base de données. Les erreurs éventuelles sont enregistrées dans les journaux appropriés.
Scénario Principal	<p>Réception des données</p> <ul style="list-style-type: none"> ➤ Les données extraites (grandeur, valeur, identifiant du capteur) sont reçues du cas d'utilisation "Traiter les données en provenance des capteurs". Le système identifie les tables qui vont faire l'objet d'une mise à jour. <p>Vérification de la dernière valeur enregistrée</p> <ul style="list-style-type: none"> ➤ Le système interroge chaque table concernée par le transmetteur météo afin de vérifier si la dernière valeur enregistrée est identique à celle reçue (même identifiant de du capteur et même valeur). <p>Enregistrement ou mise à jour des données</p> <ul style="list-style-type: none"> ➤ Si la nouvelle valeur diffère de la dernière enregistrée : Un nouvel enregistrement est ajouté à la table correspondante, incluant l'horodatage et l'identifiant du capteur et la nouvelle valeur. ➤ Si la nouvelle valeur est identique à la dernière enregistrée : L'horodatage de l'enregistrement existant est simplement mis à jour.
Scénarios Alternatifs	<p>Échec de la connexion à la base de données:</p> <ul style="list-style-type: none"> ➤ Si la base de données est inaccessible, le système retente la connexion après un délai défini. ➤ En cas d'échec persistant, les données sont temporairement stockées dans un fichier local ou une file d'attente. <p>Échec de l'insertion des données :</p> <ul style="list-style-type: none"> ➤ Si une erreur survient lors de l'insertion, le système l'enregistre dans un fichier local. <p>Incohérence dans les données reçues</p> <ul style="list-style-type: none"> ➤ Si les données reçues sont incomplètes, elles sont rejetées et enregistrées dans un journal d'erreurs pour analyse.

1.3. Diffuser les données météorologiques en temps réel

Description du cas d'utilisation	Ce cas d'utilisation permet au système de réception météorologique d'envoyer, en temps réel, des trames standardisées au format JSON à tous les clients Android connectés.
Pré-conditions	Les données météorologiques doivent être standardisées dans un format JSON qui est connu des clients Android. Les clients abonnés doivent être connectés à la plateforme de diffusion, et disposés dans une liste active du système.
Scénario principal	<p>Préparation des données à diffuser :</p> <ul style="list-style-type: none">➤ Les données extraites (grandeurs, identifiant du capteur, horodatage) sont reçues du cas d'utilisation "Traiter les données en provenance des capteurs". <p>Diffusion des données :</p> <ul style="list-style-type: none">➤ Le système parcourt la liste des clients Android connectés.➤ La trame JSON est envoyée à chaque client via une connexion réseau.

2. Protocoles

2.1. JSON

En ce qui concerne l'envoi des données nous avons choisi le type de format JSON que nous avons estimé le plus approprié. Pour faciliter la construction nous allons nous baser sur le format de la trame reçue du RTL_433. Nous allons simplement retirer les informations inutiles aux clients. Nous allons donc garder les paramètres suivant:

Date (format: AAAA-MM-JJ HH-MM-SS)

Identifiant

État de la batterie (batterie ok : true; batterie faible : false)

Le reste des données seront les informations transmises par les capteurs. Si une des informations n'est pas reçue alors le message ne contiendra pas champ contenant cette données

format : {<Date>, <Identifiant>, <Batterie> [, <Données>, <Données>]}

Lors de la diffusion de données nous allons aussi utiliser le format JSON. Afin de faciliter l'écriture et la lecture des données, nous allons réduire le nombre d'informations dans une trame, car certaines sont superflues. La nouvelle structure est représentée ci-dessous.

{Date,Identifiant,[<Données>,<Données>]}

Champs (peu changer en fonction de la trame)	Nom	Exemple de valeur
Date (Obligatoire)	Date	"2025-05-02 1:30:34"
État de la batterie (Obligatoire)	EtatBatterie	true
Identifiant du Capteur (Obligatoire)	Identifiant	4
Température	Temperature	26.3
Humidité	Humidite	51
Vitesse du vent	Vent_Moyen	0
Vitesse Maximum	Rafale	0
Direction du vent	Direction_Vent	90
Pression	pression	35
Précipitation	Precipitations	114

Exemple : {"Date":"2025-05-02 1:30:34", "Direction_Vent":90, "Etat_Batterie":"true", "Humidite":51, "Identifiant":4, "Precipitations":114, "Rafale":0, "Temperature":26.3, "Vent_Moyen":0}

2.2.WebSocket

Les WebSockets permettent d'établir une connexion persistante et bidirectionnelle entre la station météo (jouant le rôle de serveur) et l'application mobile (client). Contrairement aux requêtes HTTP classiques, cette technologie évite le besoin d'interroger régulièrement le serveur (polling) en maintenant la connexion ouverte. Ainsi, dès que de nouvelles données sont disponibles (température, humidité, pression, etc.), elles sont immédiatement envoyées à l'application.

Cette solution présente plusieurs avantages :

- Réception instantanée des données : les informations météorologiques sont transmises dès leur acquisition, sans délai.
- Optimisation des ressources : réduction de la consommation de bande passante et d'énergie, ce qui est important pour les appareils mobiles.
- Expérience utilisateur fluide : l'interface de l'application peut afficher les données en continu, ce qui est idéal pour des fonctions comme les alertes météo.

3.Outils

3.1.QTCreator



Qt est un framework de développement logiciel multiplateforme qui peut être utilisé pour créer des applications non-gui comme des outils de traitement de données ou des serveurs. Développé par The Qt Company, Qt est particulièrement apprécié pour sa capacité à fonctionner sur plusieurs systèmes d'exploitation dont le système d'exploitation Raspbian OS. Celui-ci nous offre aussi la possibilité de

coder en C++ et d'utiliser le SQL par le biais [de bibliothèques](#).

Sortie Console

Qt permet de gérer les sorties vers la console en utilisant des classes telles que QDebug, qui facilitent l'envoi de messages de débogage ou d'informations générales directement dans la console. Grâce à cette fonctionnalité, nous allons afficher les données reçues/envoyées, les erreurs et les informations liées au traitement des données. Cette fonctionnalité nous offre un outil de suivi fiable.

WebSockets

Les WebSockets sont utilisés pour permettre la transmission des données en temps réel vers l'application Android destinée aux utilisateurs.

Lecture/Écriture JSON

Dans ce projet de station météorologique connectée, la classe QJsonObject de Qt est utilisée pour gérer les trames de données échangées entre la station et les différents équipements connectés, notamment :

- la lecture des données reçues depuis les capteurs (via la clé RTL-SDR et le logiciel RTL_433),
- et la construction des messages envoyés en temps réel aux clients Android via WebSocket.

Le format JSON (JavaScript Object Notation) est particulièrement adapté pour représenter les mesures météorologiques sous forme structurée, facilement interprétable par l'application mobile.

QJsonObject joue un rôle central dans la gestion des échanges de données structurées entre la station météo et les clients, assurant une communication claire, efficace et fiable dans le système global.

Lecture de logiciel externe

La classe QProcess de Qt est utilisée pour interfacer l'application Qt avec un programme externe; rtl_433 dans notre cas. Ce dernier est un outil en ligne de commande qui permet de recevoir les données transmises par les capteurs sans fil (433 MHz), comme ceux de température, d'humidité ou de pression.

Plutôt que de réimplémenter un système de décodage complexe, le projet s'appuie sur rtl_433 pour capter et décoder les trames envoyées par les capteurs. QProcess permet alors :

- de lancer automatiquement rtl_433 en arrière-plan depuis l'application Qt,
- de lire en temps réel la sortie standard (texte brut) générée par rtl_433,
- et de récupérer les données des capteurs pour les traiter ensuite dans le programme principal.

Avantages de QProcess :

- Intégration simple de programmes externes : évite de réécrire un outil existant performant comme rtl_433.
- Lecture en temps réel des données : permet une acquisition continue et automatique.

Lecture/Écriture dans une base de données

Les classes QSqlDatabase et QSqlQuery de Qt sont utilisées pour enregistrer les données reçues des capteurs dans une base de données locale. Ce stockage permet de conserver un historique complet des mesures météorologiques, qui pourra ensuite être consulté et affiché sur une interface web.

Objectifs de l'enregistrement en base de données :

- Archiver les mesures (température, humidité, pression, etc.) avec leur horodatage. Permettre une consultation différée ou statistique des données (par heure, jour, semaine...).
- Offrir une visualisation web de l'évolution des conditions météorologiques, via un tableau de bord ou des graphiques.

Rôle des classes Qt :

- QSqlDatabase permet de configurer et d'ouvrir la connexion avec une base de données (par exemple SQLite, MySQL, etc.).
- QSqlQuery est utilisée pour exécuter les commandes SQL nécessaires : insertion des données, création de tables, requêtes de lecture pour affichage ou traitement.

L'utilisation conjointe de QSqlDatabase et QSqlQuery permet à l'application Qt de transformer un flux de données en une base de données consultable, répondant ainsi aux

besoins de traçabilité, analyse et visualisation à long terme dans ce projet de station météorologique connectée.

3.2.MariaDB

MariaDB est un SGBD relationnel open source, dérivé de MySQL, compatible avec le langage SQL. Il est reconnu pour sa stabilité, sa performance et sa compatibilité multiplateforme. Il convient parfaitement à une utilisation embarquée ou serveur dans des projets de type IoT, comme une station météo.

MariaDB est utilisé comme système de gestion de base de données (SGBD) pour stocker les données météorologiques collectées par la station. Ce stockage permet de conserver un historique complet, exploitable depuis une interface web ou une application.

MariaDB est utilisé pour :

- Stocker durablement les données des capteurs: température, humidité, pression, etc.
- Gérer les horodatages de chaque mesure pour permettre un tri chronologique.
- Permettre l'accès aux données par d'autres interfaces, comme une application web, sans dépendre de l'application Qt.
- Assurer la fiabilité des données, même en cas de coupure de l'application Qt.



3.3.RTL 433

rtl_433 est un outil open source en ligne de commande conçu pour décoder automatiquement les signaux radio transmis par des capteurs domestiques courants utilisant la fréquence 433 MHz. Il est compatible avec de nombreux protocoles propriétaires de capteurs météo, et convertit les signaux reçus en données lisibles, souvent au format JSON.

Le logiciel rtl_433 est utilisé pour recevoir les données des capteurs sans fil (température, humidité, pression, etc.) émettant sur la fréquence 433 MHz. Il fonctionne en association avec une clé RTL-SDR, qui permet de capter les signaux radio émis par les capteurs.

Dans le projet, rtl_433 joue un rôle fondamental dans la réception des données capteurs. Il est utilisé pour :

- Décoder automatiquement les signaux 433MHz reçus par la clé RTL-SDR.
- Extraire les mesures météorologiques et les afficher sous forme de texte ou de JSON.
- Fournir un flux de données en temps réel que l'application Qt lit ensuite via QProcess.

Avantages de rtl_433 :

- Large compatibilité avec des dizaines de modèles de capteurs météo 433 MHz.
- Installation simple sur Linux (Raspberry Pi, PC, etc.).
- Sortie directe en JSON, facilement exploitable dans Qt.
- Fiabilité et performance : permet une acquisition continue sans développement bas niveau.

4.Présentation de GitHub

GitHub est utilisé comme plateforme de gestion de version et de collaboration. Cet outil s’est révélé particulièrement utile pour **organiser, suivre et sécuriser le développement du code** tout au long du projet.

L’utilisation de GitHub présente plusieurs avantages concrets :

- **Historique des modifications** : chaque changement apporté au code source est enregistré, ce qui permet de revenir à une version antérieure en cas d’erreur ou de dysfonctionnement.
- **Travail collaboratif** : GitHub facilite le travail en équipe grâce au partage de dépôts, à la gestion des branches, et à l’intégration de contributions multiples sans conflit.
Sauvegarde sécurisée : le code est stocké en ligne, évitant toute perte liée à une défaillance locale.
- **Professionnalisation** : GitHub est un outil couramment utilisé dans le milieu professionnel. Son utilisation dans le cadre de ce projet permet d’acquérir de bonnes pratiques de développement logiciel.



Grâce à GitHub, la gestion du code a été plus fluide, plus structurée et mieux adaptée à la collaboration entre les membres du projet. Cela a contribué à la **réussite technique et organisationnelle** de la station météorologique connectée. Pour argumenter son utilisation voici une capture d’écran du github utilisé par le groupe:

nbertault build		7975464 · 22 minutes ago	52 Commits
Appli_Meteo	Application Android	2 weeks ago	
Document_Ressource	ajout nécessaire Web	last week	
Gantt	ajout nécessaire Web	last week	
IHM/ClientWeb	IHM Web v1.1	5 months ago	
Mocodo/StationMeteoAnalyse	Mocodo modifications	3 months ago	
Modelio	Mise a jour diagrammes	23 minutes ago	
QT	Code Final	23 minutes ago	
Site_Web	ajout nécessaire Web	last week	
build-Appli_Meteo-Desktop_Qt_6_7_1-Debug	build	22 minutes ago	
CommandesSqlCreationDatabase.sql	Renommage Commandes Database	2 weeks ago	
DossierAnalyse.odt	Dossier Analyse Final	4 months ago	
DossierAnalyse.pdf	Dossier Analyse Final	4 months ago	
FicheTestUnitaireExemple.odt	changement fiche unitaire (exemple + vierge)	3 months ago	
FicheTestUnitaireVierge.odt	changement fiche unitaire (exemple + vierge)	3 months ago	
Informations_RTL.md	Update Informations_RTL.md	5 months ago	

Et pour accéder au projet nous vous conseillons d’utiliser ce lien où nous avons regroupé la totalité du projet final et des dossiers finaux: <https://github.com/nbertault/StationMeteoFinal>

5. Diagrammes de séquences

Après avoir présenté l'ensemble de mes cas d'utilisation et les outils que j'ai choisis, il est désormais possible de représenter des diagrammes de séquences complets, représentant le vrai programme.

5.1. Traiter les données

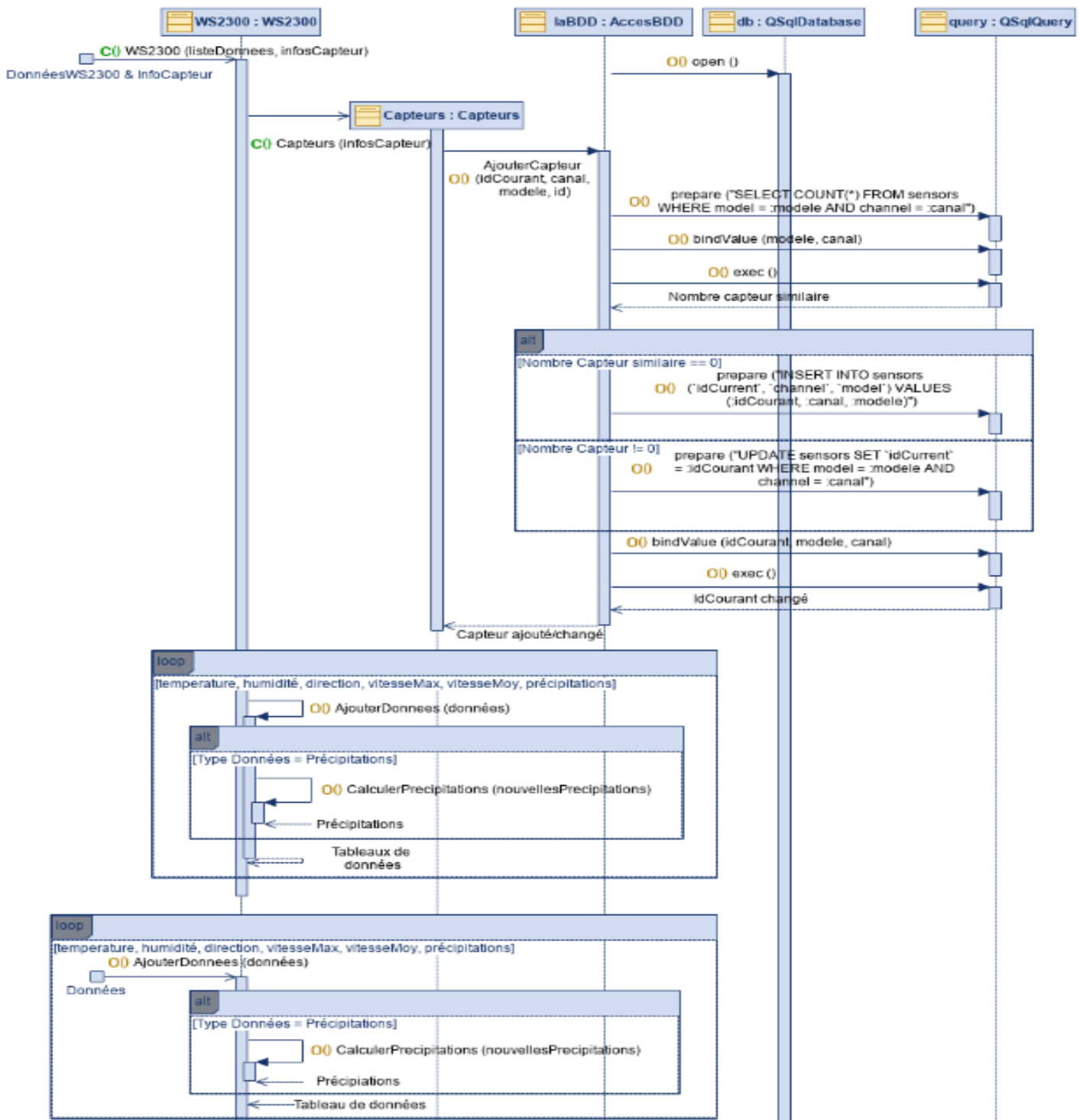


Image 2: Diagramme de séquence "Traiter les données"

5.2.Stocker les données

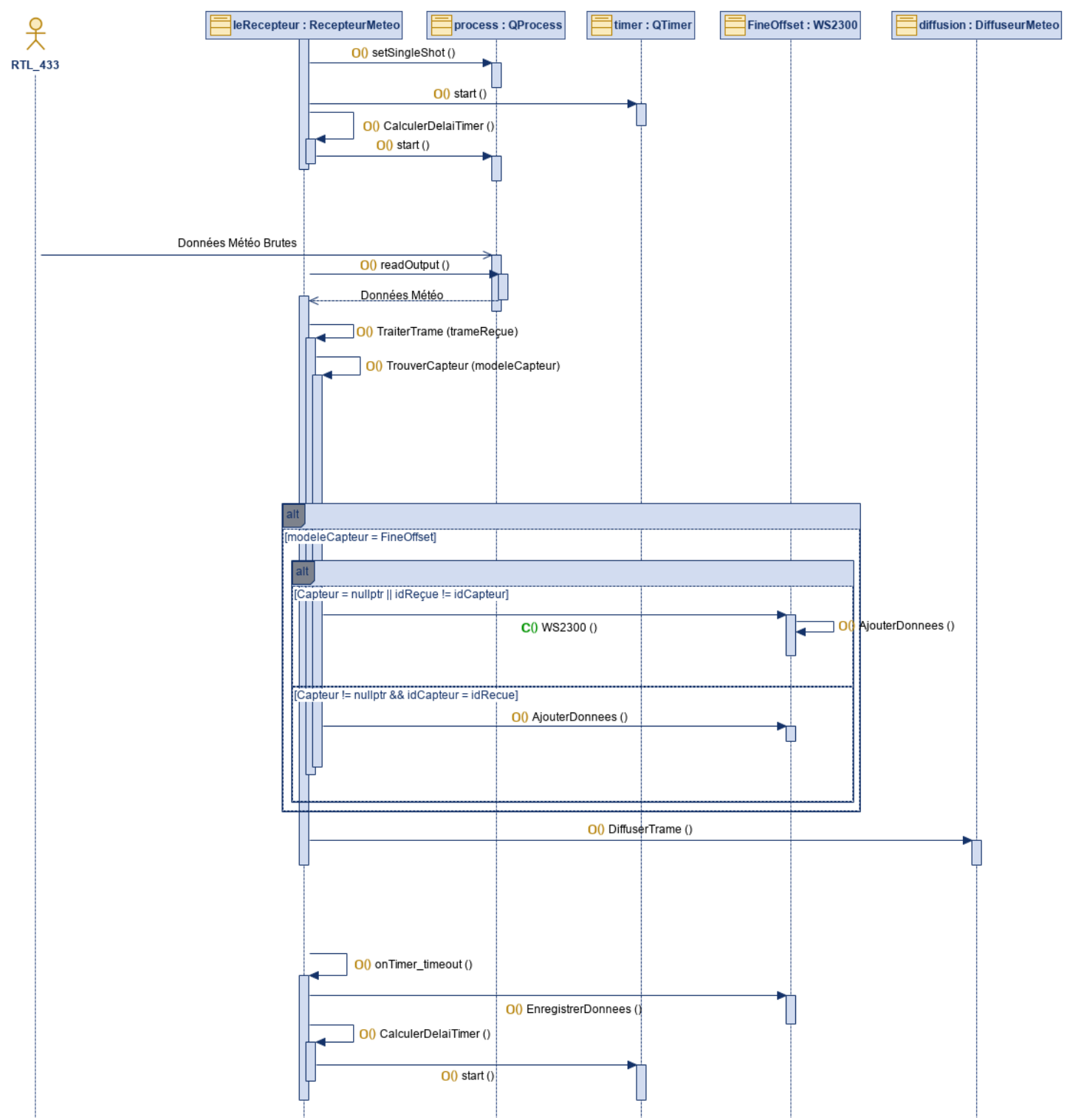


Image 3: Diagramme de séquence “Stocker les données” partie 1

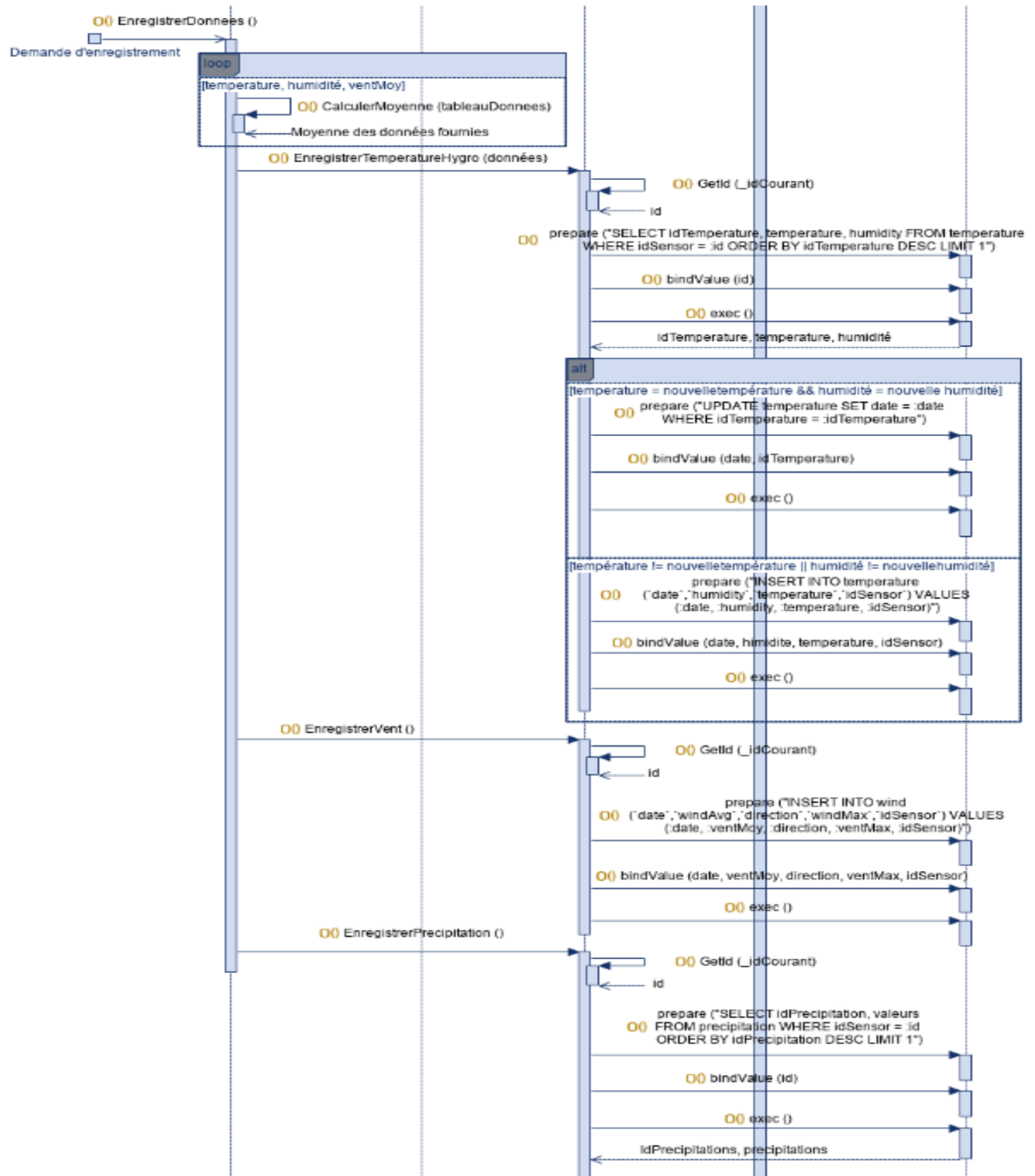


Image 4: Diagramme de séquence "Stocker les données" partie 2

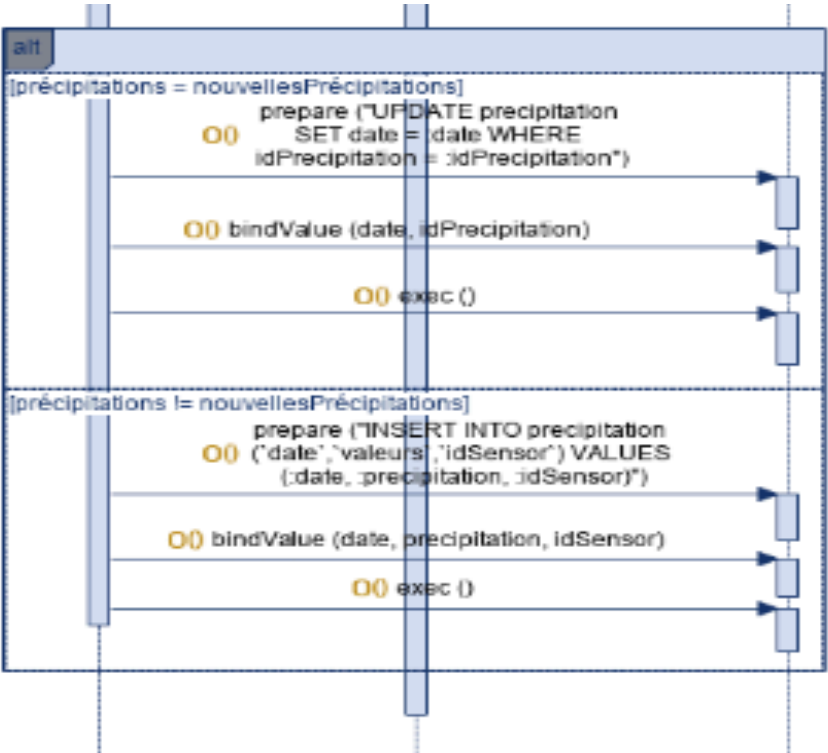


Image 5: Diagramme de séquence “Stocker les données” partie 3

5.3.Diffuser les données

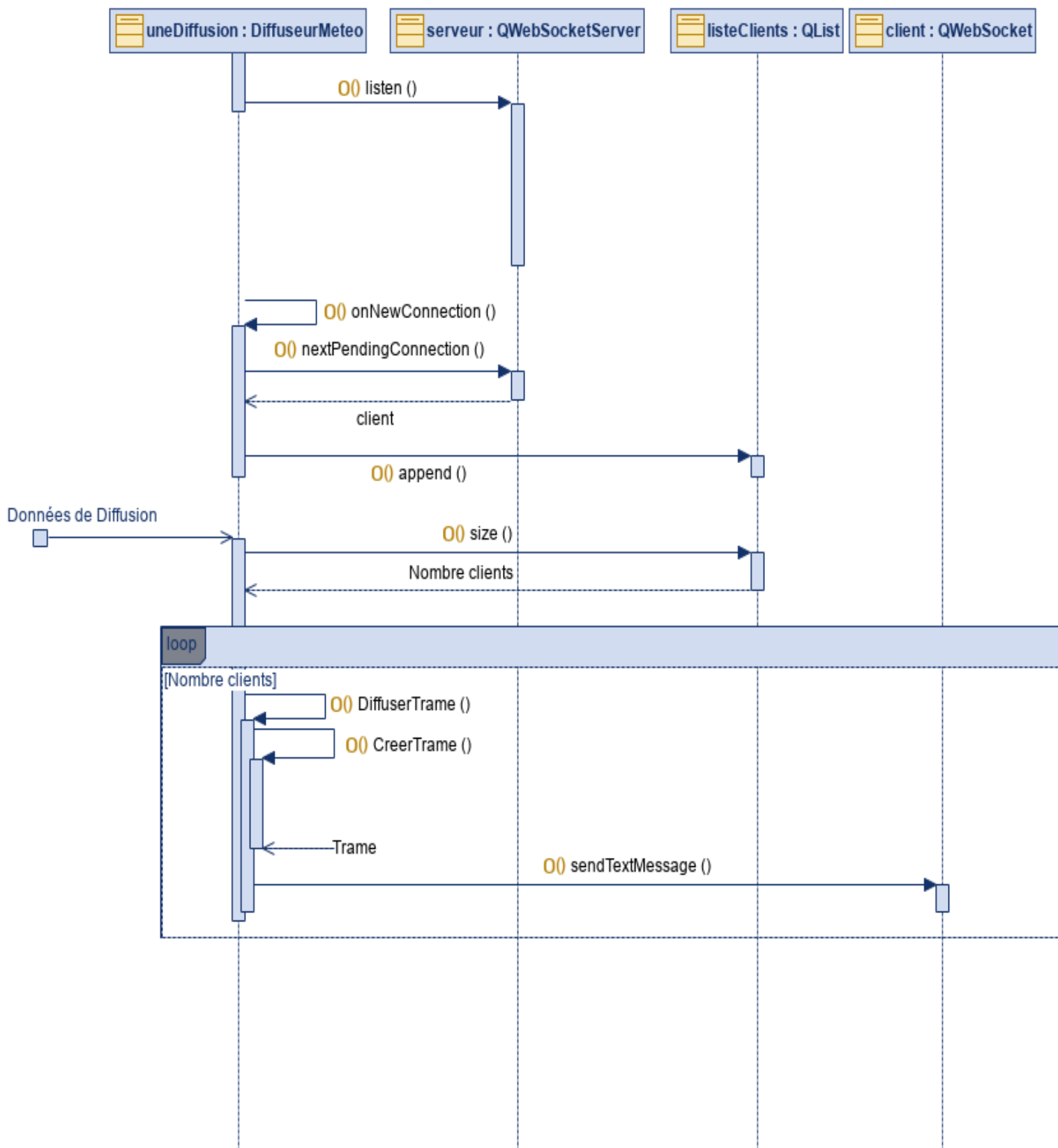


Image 6: Diagramme de séquence “Diffuser les données”

7.Fiche de test

Cette fiche de test permet de vérifier le stockage des données au sein de la BDD. La série de tests effectuée permettra de prendre en compte toutes les éventualités. Lors de l'élaboration de cette fiche de test seulement certaines classes étaient créées, une interface graphique a dû être créée afin de pouvoir réaliser les tests.

Fiche de tests					
Nature :		Fonctionnel	Référence : TW3 – ET1 – 001		
Module :		Stocker les données météorologiques	Auteur : BERTAULT Nolhan		
Date de création / mise à jour :			11/03/2025 - 11/03/2025		
Objectif :		Vérifier que la classe AccesBDD() stock les données dans la base de données			
Condition du test					
État initial du module :			Environnement du test :		
Ordinateur		PC	Linux		
Programme		RecepteurMeteo	QT Creator		
Base de données		Nom de la base : Meteorologie login: MeteorologieCiel2 mot de passe : E49KG1g2oaBXbYligH	Version et état initial La base de données contient toutes les tables nécessaires mais aucune donnée.		
Procédure de test					
Lancement de l'application :					
Repère	Opérations		Résultats attendus		
1	Envoyer des données traitées avec l'idCourant du capteur WS2300 et des données différentes des anciennes données. [date : 2020-02-12 16:14:46 ; modele : « Fineoffset-WHx080 » ; idCourant : 71 ; etatBatterie : 1 ; donnees : « temperature : 10.700, humidite : 56, directionVent : 22.5, ventMoyen : 7.344, ventMax : 8.568, precipitations : 136.800 »]		Les données temperatures et humidite sont ajoutées à la table Temperature. Les données liées aux vents sont ajoutées à la table Wind. Les données precipitations sont ajoutées à la table Precipitation.		
			Precipitation		
			date		2020-02-12 16:14:46
			idPressure		1
			valeurs		136.800
			idSensor		1
			Wind		
			date		2020-02-12 16:14:46
			idWind		1
			windAvg		7.344
			direction		22.5
			windMax		8.568
			idSensor		1
			Temperature		
			date		2020-02-12 16:14:46
			idTemperature		1
			humidity		56
			temperature		10.700
IdSensor		1			
Les informations du capteur sont enregistrées dans la table Sensors (le canal est défini à 0 par défaut)					

		<table><tr><td>Sensor</td><td>idSensor</td><td>id Current</td><td>battery State</td><td>channel</td><td>model</td></tr><tr><td></td><td>1</td><td>71</td><td>1</td><td>0</td><td>Fineoffset-WHx080</td></tr></table> <p>Les données temperature et humidite sont ajoutées à la table Temperature. Les données liées aux vents sont ajoutées à la table Wind. Les données precipitations sont ajoutées à la table Precipitation.</p>	Sensor	idSensor	id Current	battery State	channel	model		1	71	1	0	Fineoffset-WHx080																										
Sensor	idSensor	id Current	battery State	channel	model																																			
	1	71	1	0	Fineoffset-WHx080																																			
2	<p>Envoyer des données traitées avec l'idCourant du capteur WS2300 et les données de température similaires aux données précédentes.</p> <p>[date : 2020-02-12 16:14:47 ; modele : Fineoffset-WHx080 » ; idCourant : 71 ; etatBatterie : 1 ; donnees : « temperature : 10.700, humidite : 57, directionVent : 45, ventMoyen : 8.344, ventMax : 9.568, precipitations : 137.800 »]</p>	<table><tr><td colspan="2">Precipitation</td></tr><tr><td>date</td><td>2020-02-12 16:14:47</td></tr><tr><td>idPressure</td><td>2</td></tr><tr><td>valeurs</td><td>137.800</td></tr><tr><td>idSensor</td><td>1</td></tr><tr><td colspan="2">Wind</td></tr><tr><td>date</td><td>2020-02-12 16:14:47</td></tr><tr><td>idWind</td><td>2</td></tr><tr><td>windAvg</td><td>8.344</td></tr><tr><td>direction</td><td>45</td></tr><tr><td>windMax</td><td>9.568</td></tr><tr><td>idSensor</td><td>1</td></tr><tr><td colspan="2">Temperature</td></tr><tr><td>date</td><td>2020-02-12 16:14:47</td></tr><tr><td>idTemperature</td><td>2</td></tr><tr><td>humidity</td><td>57</td></tr><tr><td>temperature</td><td>10.700</td></tr><tr><td>IdSensor</td><td>1</td></tr><tr><td colspan="2">La date des dernières données ajoutées à la table Temperature est mise à jour.</td></tr></table>	Precipitation		date	2020-02-12 16:14:47	idPressure	2	valeurs	137.800	idSensor	1	Wind		date	2020-02-12 16:14:47	idWind	2	windAvg	8.344	direction	45	windMax	9.568	idSensor	1	Temperature		date	2020-02-12 16:14:47	idTemperature	2	humidity	57	temperature	10.700	IdSensor	1	La date des dernières données ajoutées à la table Temperature est mise à jour.	
Precipitation																																								
date	2020-02-12 16:14:47																																							
idPressure	2																																							
valeurs	137.800																																							
idSensor	1																																							
Wind																																								
date	2020-02-12 16:14:47																																							
idWind	2																																							
windAvg	8.344																																							
direction	45																																							
windMax	9.568																																							
idSensor	1																																							
Temperature																																								
date	2020-02-12 16:14:47																																							
idTemperature	2																																							
humidity	57																																							
temperature	10.700																																							
IdSensor	1																																							
La date des dernières données ajoutées à la table Temperature est mise à jour.																																								
3	<p>Envoyer des données traitées avec l'idCourant du capteur WS2300 et des données température et hygro similaires aux données précédentes.</p> <p>[date : 2020-02-12 16:14:48 ; modele : « Fineoffset-WHx080 » ; idCourant : 71 ; etatBatterie : 1 ; donnees : « temperature : 10.700, humidite : 57, directionVent : 180, ventMoyen : 9.344, ventMax : 10.568, precipitations : 138.800 »]</p>	<table><tr><td colspan="2">Temperature</td></tr><tr><td>date</td><td>2020-02-12 16:14:48</td></tr><tr><td>idTemperature</td><td>2</td></tr><tr><td>humidity</td><td>57</td></tr><tr><td>temperature</td><td>10.700</td></tr><tr><td>IdSensor</td><td>1</td></tr><tr><td colspan="2">Les données liées aux vents ajoutées à la table dans la table Wind. Les données precipitations sont ajoutées à la table Precipitation.</td></tr><tr><td colspan="2">Wind</td></tr></table>	Temperature		date	2020-02-12 16:14:48	idTemperature	2	humidity	57	temperature	10.700	IdSensor	1	Les données liées aux vents ajoutées à la table dans la table Wind. Les données precipitations sont ajoutées à la table Precipitation.		Wind																							
Temperature																																								
date	2020-02-12 16:14:48																																							
idTemperature	2																																							
humidity	57																																							
temperature	10.700																																							
IdSensor	1																																							
Les données liées aux vents ajoutées à la table dans la table Wind. Les données precipitations sont ajoutées à la table Precipitation.																																								
Wind																																								

		<table><tr><td>date</td><td>2020-02-12 16:14:48</td></tr><tr><td>idWind</td><td>3</td></tr><tr><td>windAvg</td><td>9.344</td></tr><tr><td>direction</td><td>180</td></tr><tr><td>windMax</td><td>10.568</td></tr><tr><td>idSensor</td><td>1</td></tr><tr><td colspan="2">Precipitation</td></tr><tr><td>date</td><td>2020-02-12 16:14:48</td></tr><tr><td>idPressure</td><td>3</td></tr><tr><td>valeurs</td><td>138.800</td></tr><tr><td>idSensor</td><td>1</td></tr></table>	date	2020-02-12 16:14:48	idWind	3	windAvg	9.344	direction	180	windMax	10.568	idSensor	1	Precipitation		date	2020-02-12 16:14:48	idPressure	3	valeurs	138.800	idSensor	1														
date	2020-02-12 16:14:48																																					
idWind	3																																					
windAvg	9.344																																					
direction	180																																					
windMax	10.568																																					
idSensor	1																																					
Precipitation																																						
date	2020-02-12 16:14:48																																					
idPressure	3																																					
valeurs	138.800																																					
idSensor	1																																					
4	<p>Envoyer des données traitées avec l'idCourant du capteur WS2300 et des données de précipitation similaires aux données précédentes. [date : 2020-02-12 16:14:49 ; modele : « Fineoffset-WHx080 » ; idCourant : 71 ; etatBatterie : 1 ; donnees : « temperature : 12.700, humidite : 59, directionVent : 0, ventMoyen : 10.344, ventMax : 11.568, precipitations : 138.800 »]</p>	<p>La date des dernières données ajoutées à la table Precipitaion est mise à jour.</p> <table><tr><td colspan="2">Precipitation</td></tr><tr><td>date</td><td>2020-02-12 16:14:49</td></tr><tr><td>idPressure</td><td>3</td></tr><tr><td>valeurs</td><td>138.800</td></tr><tr><td>idSensor</td><td>1</td></tr></table> <p>Les données temperatures et humidite sont ajoutées à la table Temperature. Les données liées aux vents sont ajoutées à la table Precipitation.</p> <table><tr><td colspan="2">Temperature</td></tr><tr><td>date</td><td>2020-02-12 16:14:49</td></tr><tr><td>temperature</td><td>12.700</td></tr><tr><td>idTemperature</td><td>3</td></tr><tr><td>humidity</td><td>59</td></tr><tr><td>IdSensor</td><td>1</td></tr></table> <table><tr><td colspan="2">Wind</td></tr><tr><td>date</td><td>2020-02-12 16:14:49</td></tr><tr><td>idWind</td><td>4</td></tr><tr><td>windAvg</td><td>10.344</td></tr><tr><td>direction</td><td>0</td></tr><tr><td>windMax</td><td>11.568</td></tr><tr><td>idSensor</td><td>1</td></tr></table>	Precipitation		date	2020-02-12 16:14:49	idPressure	3	valeurs	138.800	idSensor	1	Temperature		date	2020-02-12 16:14:49	temperature	12.700	idTemperature	3	humidity	59	IdSensor	1	Wind		date	2020-02-12 16:14:49	idWind	4	windAvg	10.344	direction	0	windMax	11.568	idSensor	1
Precipitation																																						
date	2020-02-12 16:14:49																																					
idPressure	3																																					
valeurs	138.800																																					
idSensor	1																																					
Temperature																																						
date	2020-02-12 16:14:49																																					
temperature	12.700																																					
idTemperature	3																																					
humidity	59																																					
IdSensor	1																																					
Wind																																						
date	2020-02-12 16:14:49																																					
idWind	4																																					
windAvg	10.344																																					
direction	0																																					
windMax	11.568																																					
idSensor	1																																					

5

Envoyer des données traitées avec un idCourant différent du capteur WS2300 et un modèle similaire.
[date : 2020-02-12 16:14:50 ;
modele : « Fineoffset-WHx080 » ;
idCourant : 81 ;
etatBatterie : 1 ;
donnees : « temperature : 10.700, humidite : 56,
directionVent : 22.5,
ventMoyen : 7.344,
ventMax : 8.568,
precipitations : 136.800 »]

Sensor	idSensor	idCurrent	battery State	channel	model
	1	81	1	0	Fineoffset-WHx080
Les données temperatures et humidite sont ajoutées à la table Temperature. Les données liées aux vents sont ajoutées à la table Wind. Les données precipitations sont ajoutées à la table Precipitation.					
Temperature					
date		2020-02-12 16:14:50			
temperature		10.700			
idTemperature		4			
humidity		56			
IdSensor		1			
Wind					
date		2020-02-12 16:14:50			
idWind		5			
windAvg		7.344			
direction		22.5			
windMax		8.568			
idSensor		1			
Precipitation					
date		2020-02-12 16:14:50			
idPressure		4			
valeurs		136.800			
idSensor		1			

6

Envoyer des données traitées avec un idCourant différent du capteur WS2300 et un modèle similaire.
[date : 2020-02-12 16:14:50 ;
modele : « Fineoffset-WHx080 » ;
idCourant : 81 ;
etatBatterie : 1 ;
donnees : « temperature : 10.700, humidite : 56,
directionVent : 22.5,
ventMoyen : 7.344,
ventMax : 8.568,
precipitations : 136.800 »]

Sensor	idSensor	idCurrent	battery State	channel	model
	1	81	1	0	Fineoffset- WHx080
Les données temperatures et humidite sont ajoutées à la table Temperature. Les données liées aux vents sont ajoutées à la table Wind. Les données precipitations sont ajoutées à la table Precipitation.					
Temperature					
date		2020-02-12 16:14:50			
temperature		10.700			
idTemperature		4			
humidity		56			
IdSensor		1			
Wind					
date		2020-02-12 16:14:50			
idWind		5			
windAvg		7.344			
direction		22.5			
windMax		8.568			
idSensor		1			
Precipitation					
date		2020-02-12 16:14:50			
idPressure		4			
valeurs		136.800			
idSensor		1			

7	<div>Envoyer des données traitées avec l'idCourant du capteur WS2300 et des données manquantes.</div> <div>[date : 2020-02-12 16:14:51 ; modele : « Fineoffset-WHx080 » ; idCourant : 81 ; etatBatterie : 1 ; donnees : « temperature : 11.700, humidite : 57, directionVent : 45, ventMoyen : 8.344, ventMax : 9.568 »]</div>	Les données temperatures et humidite sont ajoutées à la table Temperature. Les données liées aux vents sont ajoutées à la table Wind.	
		Temperature	
		date	2020-02-12 16:14:51
		temperature	11.700
		idTemperature	5
		humidity	57
		IdSensor	1
		Wind	
		date	2020-02-12 16:14:51
		idWind	6
		windAvg	8.344
		direction	45
		windMax	9.568
		idSensor	1
		Aucune modification n'a été apportées à la table Precipitation	
		Precipitation	
		date	2020-02-12 16:14:50
		idPressure	4
		valeurs	136.800
		idSensor	1

Le test unitaire réalisé sur la fonctionnalité d'ajout de données dans la base de données a été effectué avec succès. Les différents cas de saisie ont été validés, et les données insérées ont été retrouvées conformes dans les tables concernées, sans erreur ni comportement inattendu.

- Aucun problème n'a été constaté :
- Les connexions à la base de données ont été établies correctement.
 - Les requêtes SQL exécutées ont produit les résultats attendus.
 - L'intégrité des données (types, contraintes, clés étrangères) a été respectée.

Ce test confirme que la fonction Stocker les données météorologiques est opérationnelle et fiable, et peut être intégrée au projet principal.

8. Diagramme de Gantt

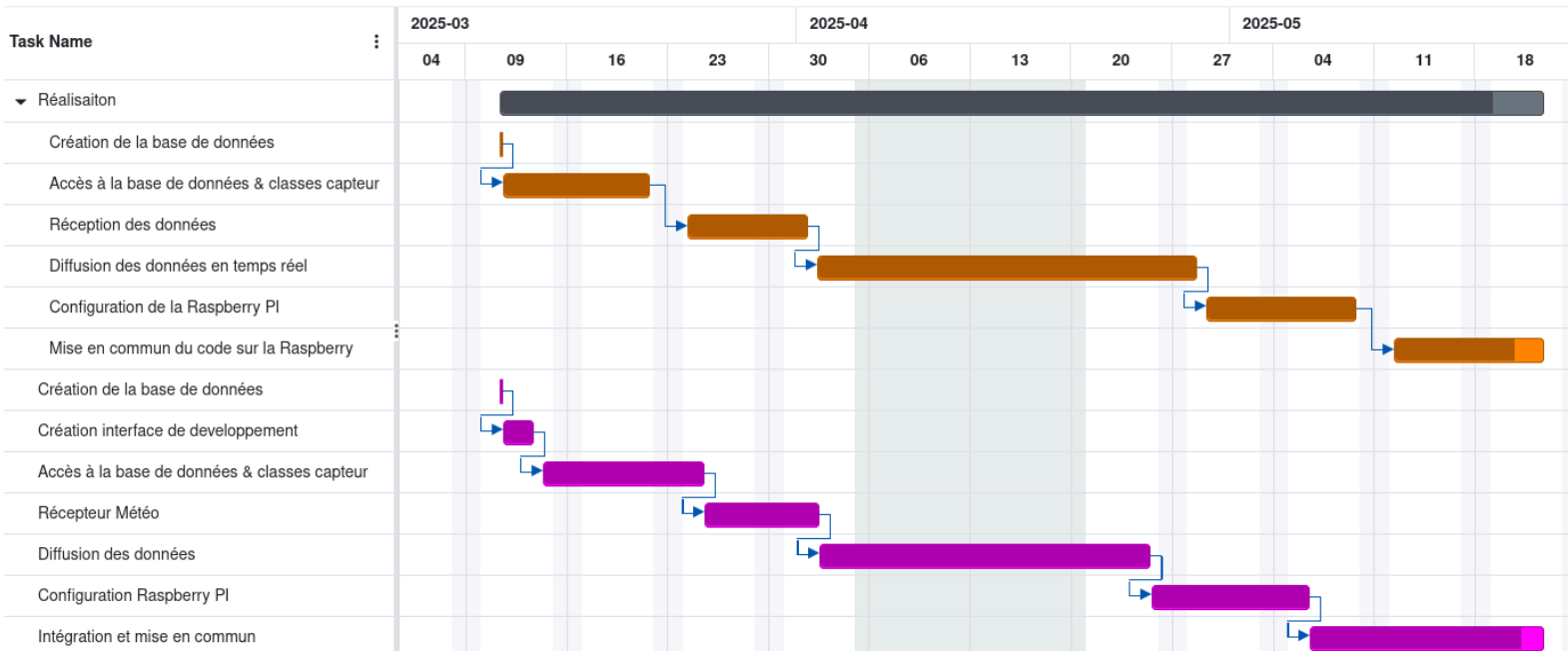


Image 8: Comparaison Gantt prévisionnel et Gantt réel

Cette comparaison entre le diagramme de Gantt prévisionnel et le diagramme de Gantt réel, nous montre que lors de la mise en place nous n'avions pas prévu l'intégration d'une interface de test, cette omission nous a mis quelque peu en retard sur le plan initial, cependant les tâches ayant été effectuées plus rapidement que prévu ce retard a été compensé durant toute la durée du projet et nous a même permis d'avoir plus de temps pour la mise en commun des parties de chacun.

9. Compétences acquises

La réalisation du projet de station météorologique connectée a permis d'acquérir et d'approfondir un large panel de compétences techniques et professionnelles, en phase avec les réalités du milieu des systèmes numériques et embarqués.

En matière de technique, le projet a permis :

- De manipuler des outils de communication tels que les WebSockets pour la diffusion de données en temps réel.
- D'utiliser et intégrer des formats d'échanges de données tels que JSON via QObject.
- D'interagir via des outils externes grâce à QProcess, pour notamment exploiter les données issues du logiciel rtl_433 via une clé RTL-SDR.
- D'implémenter et de gérer une base de données relationnelle MariaDB, avec des requêtes SQL via QSqlDatabase et QSqlQuery.
- De concevoir une base de données structurée et normalisée pour la mise en historique des mesures météorologiques.
- De travailler sur un environnement embarqué, avec un Raspberry Pi configuré en serveur central.

D'autre part au niveau professionnel, ce projet a permis aussi :

- D'appliquer une gestion de projet rigoureuse (planification via Gantt, répartition de tâches, suivi d'avancement).
- D'entretenir un esprit d'analyse, d'autonomie, et de résolution de problèmes concrets.
- De pouvoir élaborer une synergie efficace au sein d'une équipe technique.
- De documenter correctement les étapes du projet, les choix techniques, et les résultats obtenus.

Pour résumer, ce projet se veut être l'illustration d'une expérience complète ayant permis de mobiliser les compétences du référentiel BTS CIEL et de s'initier à des situations techniques « professionnelles » proches du réel.

10.Perspectives d'améliorations

Bien que ce projet se soit globalement déroulé de manière satisfaisante certains aspects de celui-ci pourrait être améliorable. Voici certains exemples pouvant être travaillé dans l'optique d'une optimisation du code:

- Tout d'abord le système n'est pas 100% autonome, en effet celui-ci est fait pour fonctionner avec les capteurs qu'y nous ont été fournis, il est donc possible de l'améliorer afin de prendre en compte tous les capteurs émettant sur du 433MH.
- De plus il était prévu que l'on y installe un BME280, par manque de temps celui-ci n'a pas put être installé ce qui fait un second point d'amélioration, bien que l'installation d'un tel capteur en local nécessite l'implémentation d'une nouvelle bibliothèque et la modification de la classe ayant été prévue.
- En ce qui concerne la diffusion, les clés permettant l'identification des capteurs par les clients sont définies au préalable et écrites dans le code. Il nous faudrait trouver un moyen de rendre ces clés uniques, les communiquer aux clients de manière automatique.