



# TABLE TENNIS PERFORMANCE ANALYSER

HAMMOUMA Youssef - BOTELLA-BROC Yohann - CANONNE Ludwing - MONTEIRO  
Geoffrey



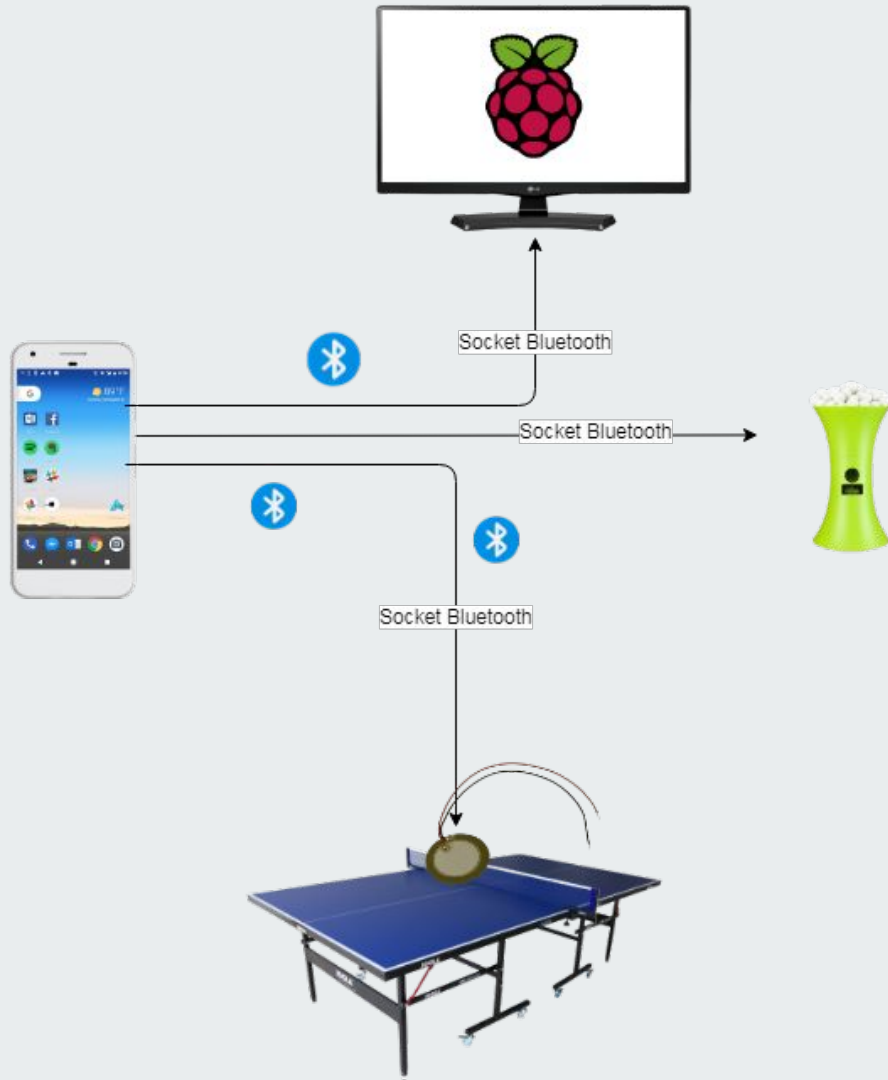
# Systeme TTPA

- **Module de gestion de séance (Mobile-TTPA)** : le joueur paramètre et lance la séance d'entraînement à partir d'une application sur un terminal mobile (sous Android)
- **Module de distribution de balles (Robot-TTPA)** : le robot lanceur envoie des balles, en suivant un rythme fixé par le joueur, avec des effets différents et réalistes.
- **Module de détection des impacts (Table-TTPA)** : la table est équipée de capteurs permettant d'identifier la zone impactée par la balle renvoyée par le joueur ;
- **Module de visualisation de performance (Écran-TTPA)** : le joueur peut visualiser en "temps réel" le rythme de jeu, la précision, le pourcentage de réussite.



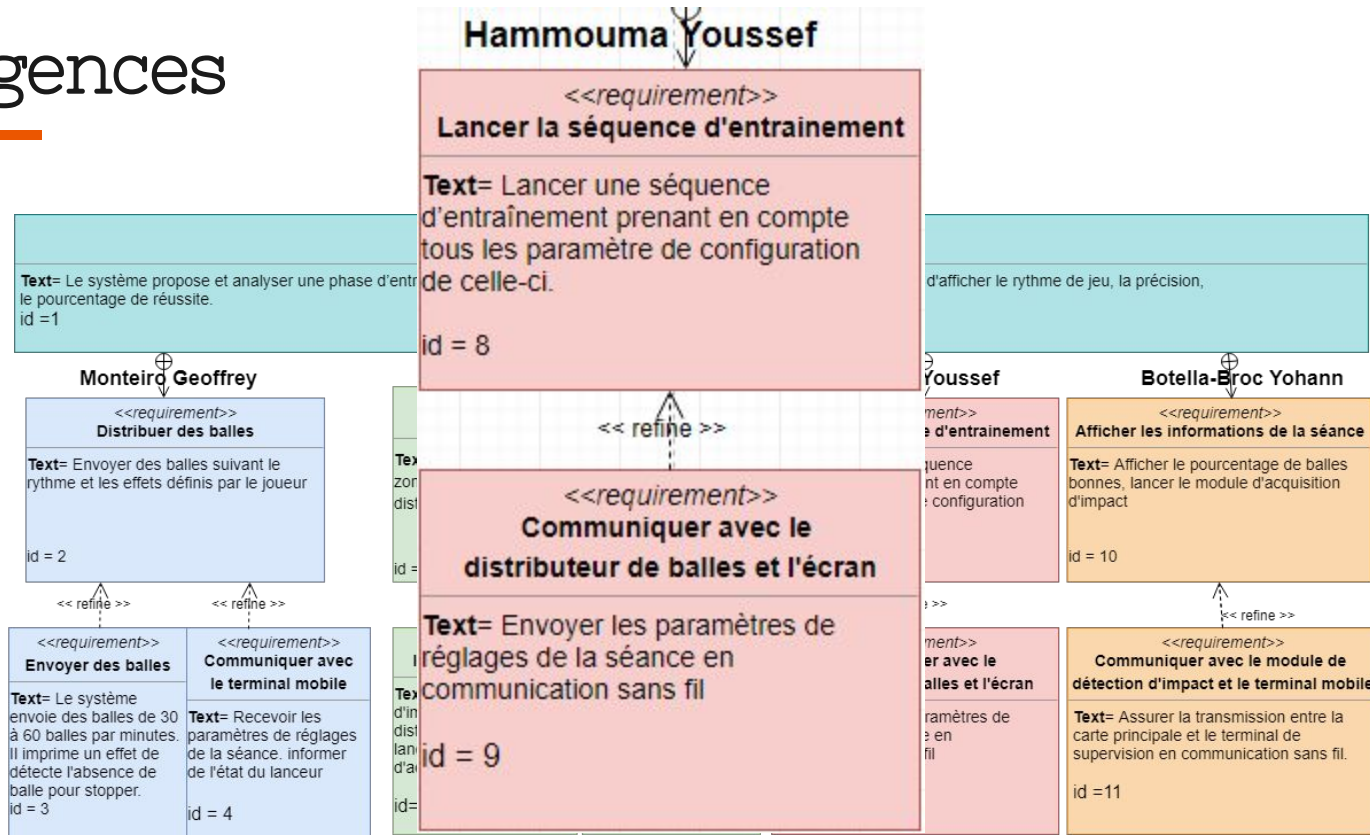
# Présentation générale

- Permettre une analyse des performances du joueur
- Proposer une phase d'entraînement adaptée au niveau du joueur
- Détecter l'impact des balles afin d'afficher le rythme de jeu, la précision et le pourcentage de réussite
- Identifier la zone d'impact sur un écran de télévision en fin d'exercice



# Synoptique du système

# Exigences





# Répartition des tâches (IR)

Botella-Broc Yohann (IR 1)

## *Module de visualisation de performance*

- ❑ Afficher un écran d'accueil
- ❑ Visualiser l'impact de balles dans les zones de la table
- ❑ Visualiser les données de la séance en temps réel et en fin de séances
- ❑ Dialoguer avec le terminal mobile

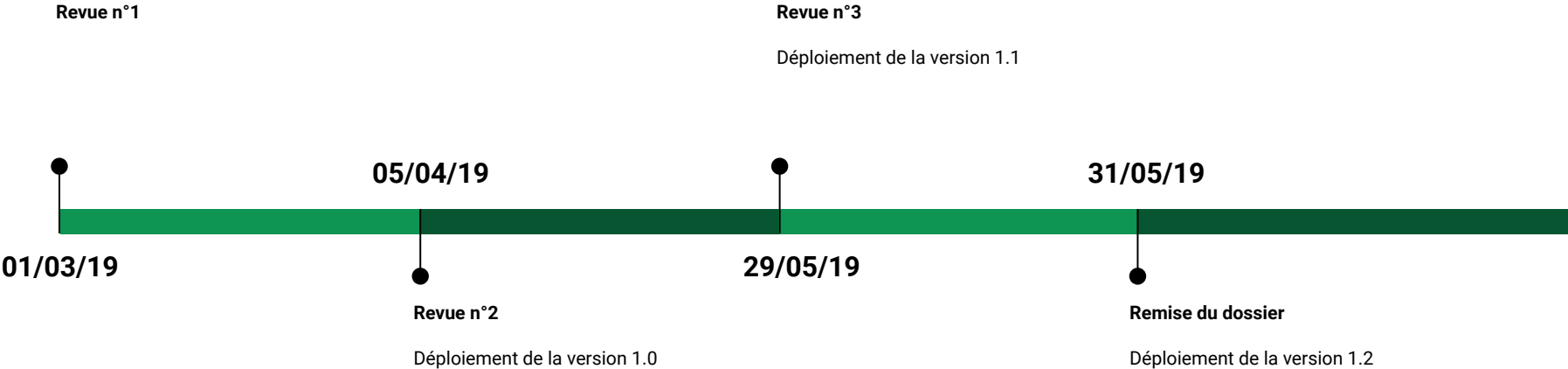
Hammouma Youssef (IR 2)

## *Module de gestion de séance*

- ❑ Créer un profil de joueur
- ❑ Paramétrer une séance d'entraînement
- ❑ Enregistrer les données des séances
- ❑ Consulter l'historique des séances d'un joueur
- ❑ Purger les séances
- ❑ Dialoguer avec les module



# Planification





# Ressources de développement

Fonctionnalité	Élément utilisé	Version
Système d'exploitation du poste de développement	GNU/LINUX	4.8.0
Logiciel de planification	Trello + TrelloGantt	
Génération des diagrammes	BOUML + Draw.io	7.8
Gestion de versions	Subversion	1.9.3
Environnement de développement	Qt/Qml	5.10.1
Génération de la documentation	Doxygen	1.8.11
Tablette test	Tablette Samsung	Android 7.0





# Organisation

## Convention de nommage

- **Nom de Classe** : NomDeClasse
- **Nom de variable**: nomDeVariable
- **Nom de fichier**: NomDeFichier

## Codage

- **Indentation** : 4 espaces
- **Documentation** : Doxygen



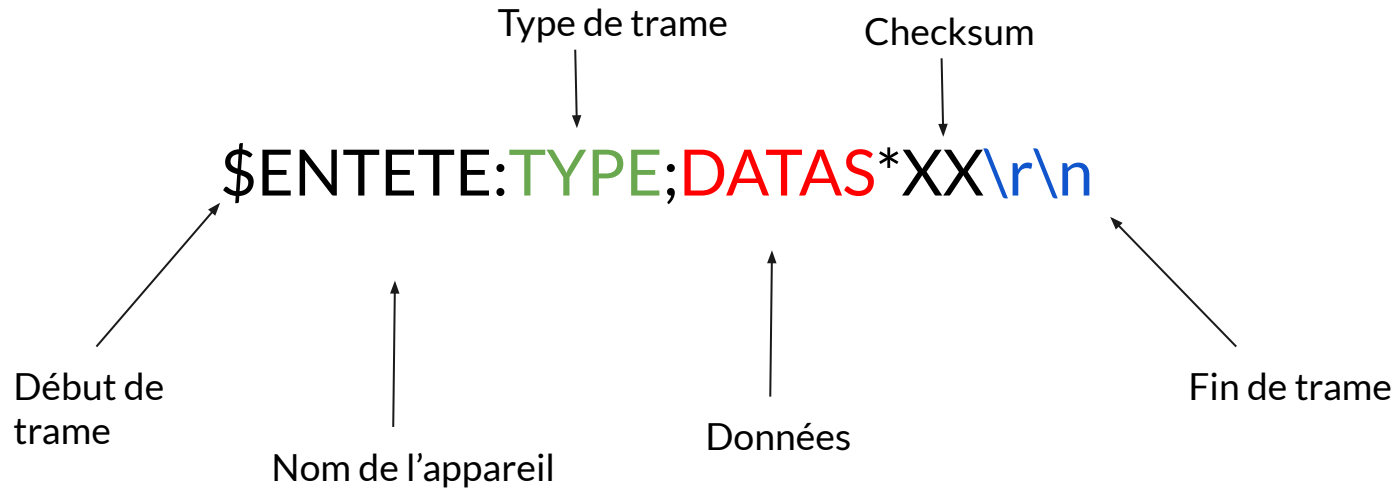
# Organisation

Gestionnaire de version (IR) : **Subversion**, pour les documents sources du projet

Espace de stockage (IR + EC) : **Google Drive**, pour les documents ressources du projet

Protocole de communication

# Protocole de communication

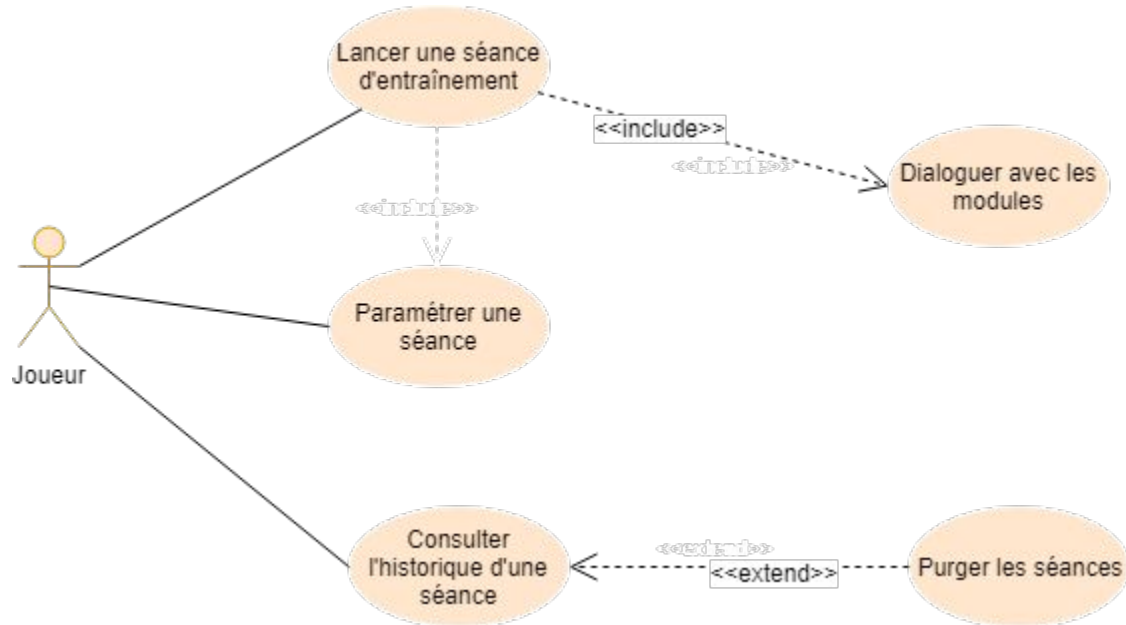





# Bluetooth

Module	Version du Bluetooth
Gestion de séance (terminal-mobile)	4.1
Visualisation de performance (écran-ttpa)	4.1
Distribution de balles (robot-ttpa)	4.2
Détection d'impacts (table-ttpa)	3.0

# Diagramme des cas d'utilisation



# Application




▼

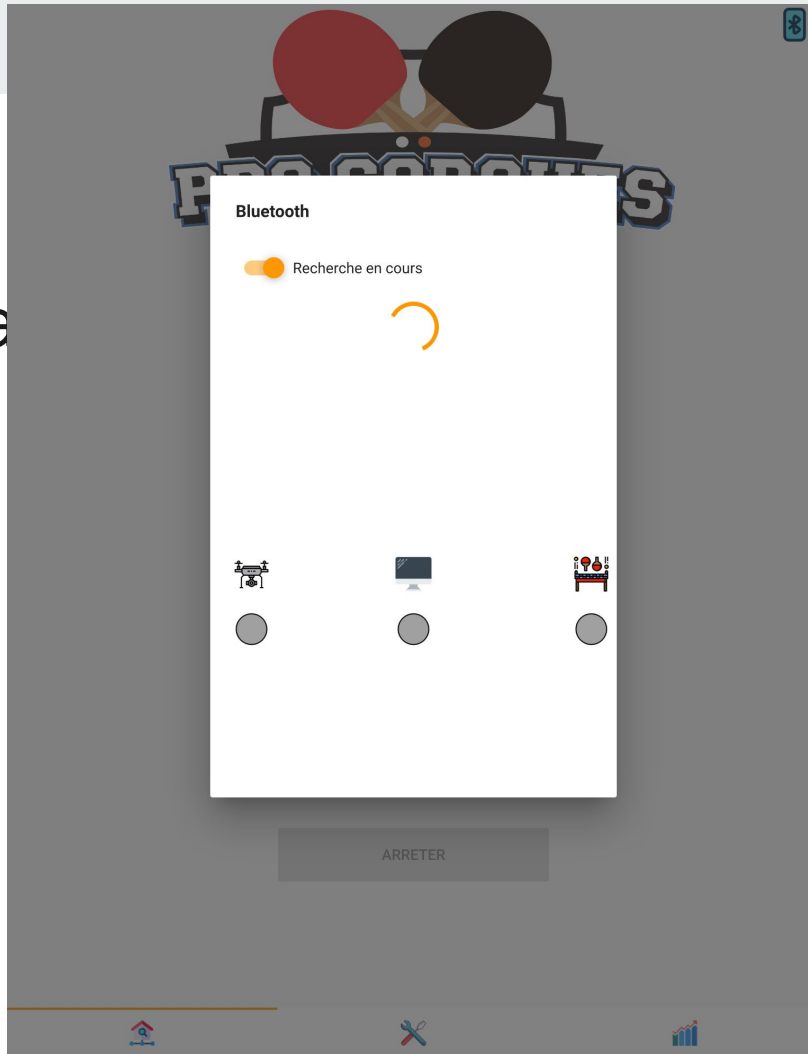
Nom du Joueur

JOUER

ARRETER



# Connexion a



# Paramétrage

Choisir la zone du robot

ZONE 1	ZONE 2	ZONE 3
ZONE 4		
ZONE 5	ZONE 6	ZONE 7

CANCEL SAVE





# Diagram

```

- peripheriqueBluetoothLocal : QBluetoothLocalDevice
- nomPeripheriqueBluetoothLocal : QString
- adressePeripheriqueBluetoothLocal : QString
- <<QList>> peripheriquesTTPA : QObject*
- discoveryAgentDevice : QBluetoothDeviceDiscoveryAgent
- etatRecherche : bool
- peripheriqueRobotTTPA : bool
- peripheriqueEcranTTPA : bool
- peripheriqueTableTTPA : bool
+ CommunicationBluetooth()
+ ~CommunicationBluetooth()
+ rechercherPeripheriquesTTPA() : void
+ arreterRecherche() : void
+ estValide() : bool
+ connecterPeripheriquesTTPA() : void
+ deconnecterPeripheriquesTTPA() : void
+ envoyerPeripheriqueTTPA(in nom : QString, in message : QString) : bool
+ demarrerSeance(in frequenceBalle : int, in nbBalles : int, in effet : int, in puissance : int, in rotation : bool) : bool
+ getNomPeripheriqueBluetoothLocal() : QString
+ getAdressePeripheriqueBluetoothLocal() : QString
+ getPeripheriquesTTPA() : QVariant
+ ajouterPeripheriqueTTPA(in info : QBluetoothDeviceInfo) : void
+ rechercheTerminee() : void
+ rechercheTerminee() : void
+ rechercheErreur(in erreur : QBluetoothDeviceDiscoveryAgent::Error) : void
+ recherche() : void
+ detecte() : void
+ robotDetecte() : void
+ ecranDetecte() : void
+ tableDetecte() : void
+ peripheriquesTTPAUpdated() : void
+ CommunicationBluetooth()

```

## terminal-TTPA::CommunicationBluetooth

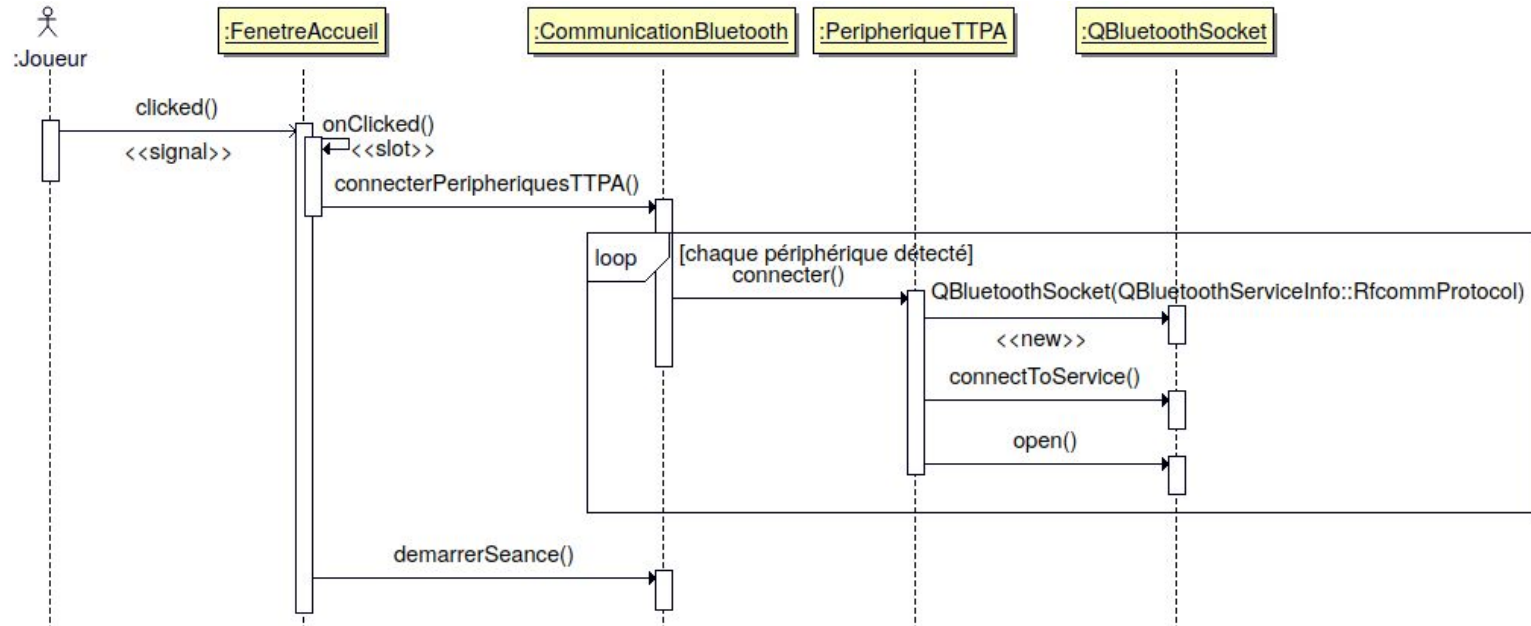
```

- peripheriqueBluetoothLocal : QBluetoothLocalDevice
- nomPeripheriqueBluetoothLocal : QString
- adressePeripheriqueBluetoothLocal : QString
- <<QList>> peripheriquesTTPA : QObject*
- discoveryAgentDevice : QBluetoothDeviceDiscoveryAgent
- etatRecherche : bool
- peripheriqueRobotTTPADetecte : bool
- peripheriqueEcranTTPADetecte : bool
- peripheriqueTableTTPADetecte : bool
+ CommunicationBluetooth()
+ ~CommunicationBluetooth()
+ rechercherPeripheriquesTTPA() : void
+ arreterRecherche() : void
+ estValide() : bool
+ connecterPeripheriquesTTPA() : void
+ deconnecterPeripheriquesTTPA() : void
+ envoyerPeripheriqueTTPA(in nom : QString, in message : QString) : bool
+ demarrerSeance(in frequenceBalle : int, in nbBalles : int, in effet : int, in puissance : int, in rotation : bool) : bool
+ getNomPeripheriqueBluetoothLocal() : QString
+ getAdressePeripheriqueBluetoothLocal() : QString
+ getPeripheriquesTTPA() : QVariant
+ ajouterPeripheriqueTTPA(in info : QBluetoothDeviceInfo) : void
+ rechercheTerminee() : void
+ rechercheErreur(in erreur : QBluetoothDeviceDiscoveryAgent::Error) : void
+ recherche() : void
+ detecte() : void
+ robotDetecte() : void
+ ecranDetecte() : void
+ tableDetecte() : void
+ peripheriquesTTPAUpdated() : void
+ CommunicationBluetooth()

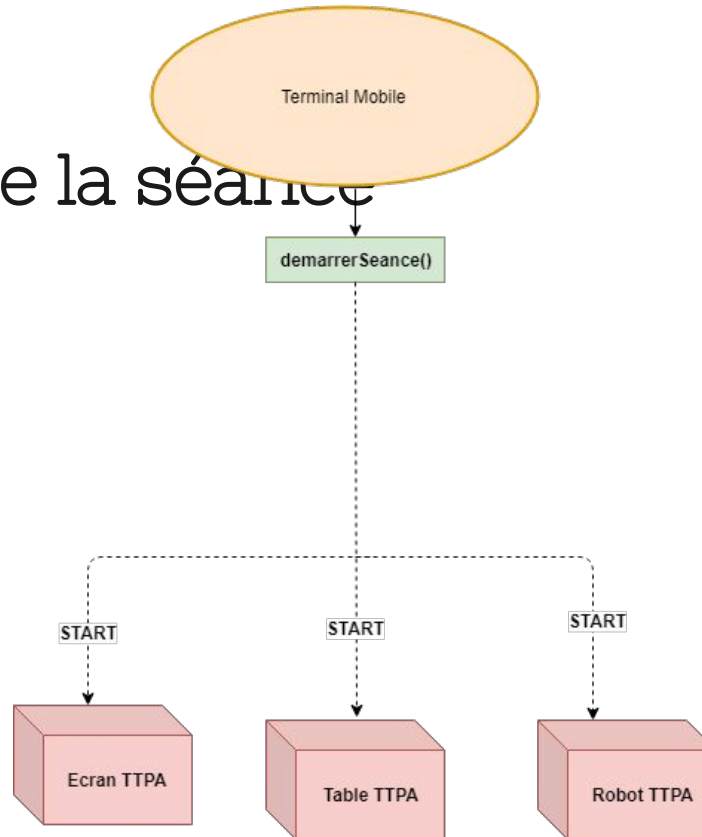
```

ct = nullptr

# Scénario de début de séance



# Démarrage de la séance



## Trames de démarrage

\$robot-ttpa-2:START;45;60;2;30;1\*XX\r\n

paramètres du robot

\$table-ttpa-2:START\*XX\r\n

\$ecran-ttpa-2:START;JULIEN;2;1\*XX\r\n

nom du joueur

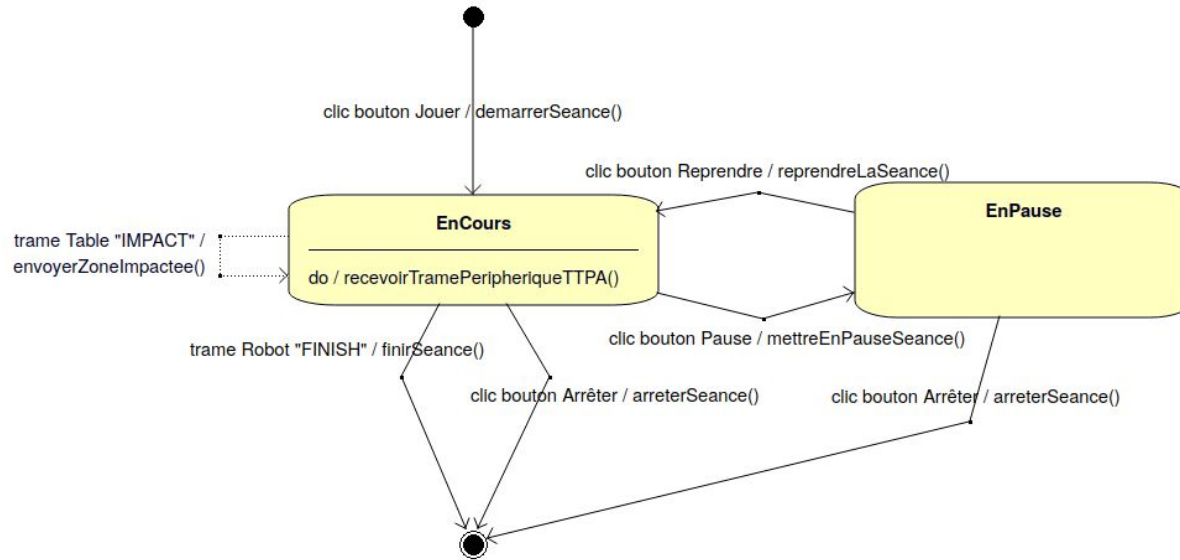
zone objectif

zone robot

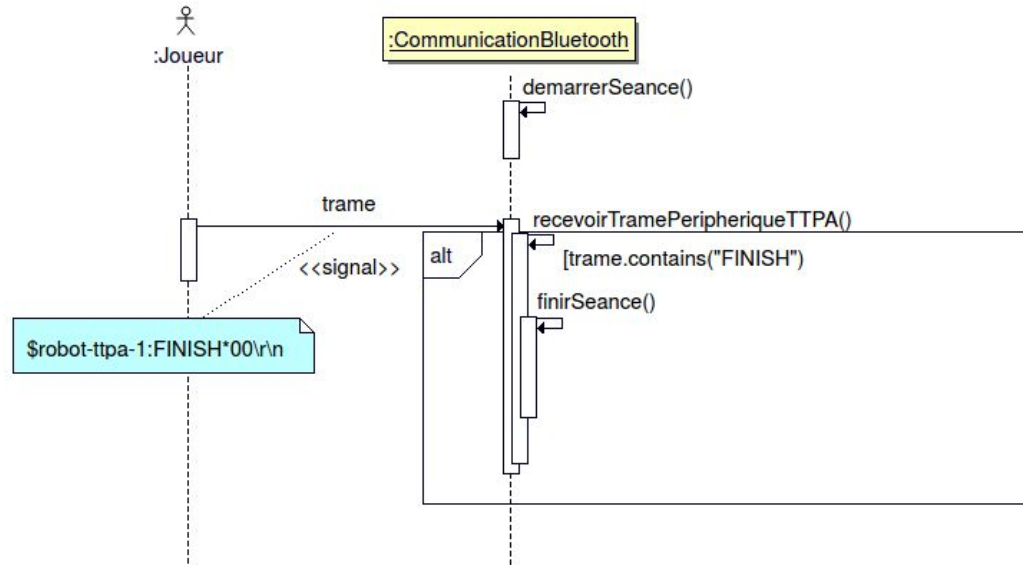
## Les différents types de trames

Trame	Rôle
START	Démarrer une séance
PAUSE	Mettre la séance en pause
RESUME	Reprendre la séance
END	Mettre fin à la séance
RESET	Réinitialiser la séance
ERROR	Indiquer une erreur
IMPACT	Indiquer un impact dans une zone de la table
STAT	Transmettre la zone impactée à l'écran
BILAN	Indiquer le nombre de balles en fin de séance
INFO	Transmettre l'erreur du robot à l'écran
FINISH	Indiquer que le robot a envoyé toutes les balles

# Diagramme états-transitions



# Scénario de fin de séance





Choix d'un j



BEAUMONT Jérôme

CHILA Patrick

ELOI Damien

LEBESSON Emmanuel

LEGOUT Christophe

MARTINEZ Michel

MATTENET Adrien

Youssef Hammouma

# Choix d'une

Séance balles liftés zone 4

Nom de la séance

Nombre de balles/minutes

20

Nombre de balles

10

Balle coupée

Puissance

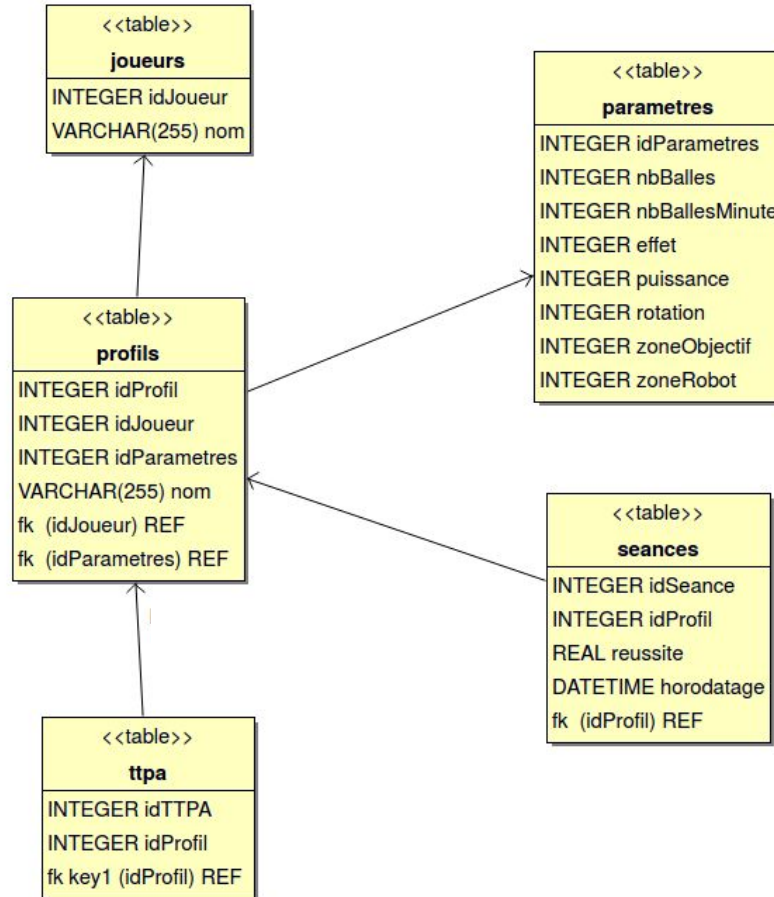
0%

☐ Rotation

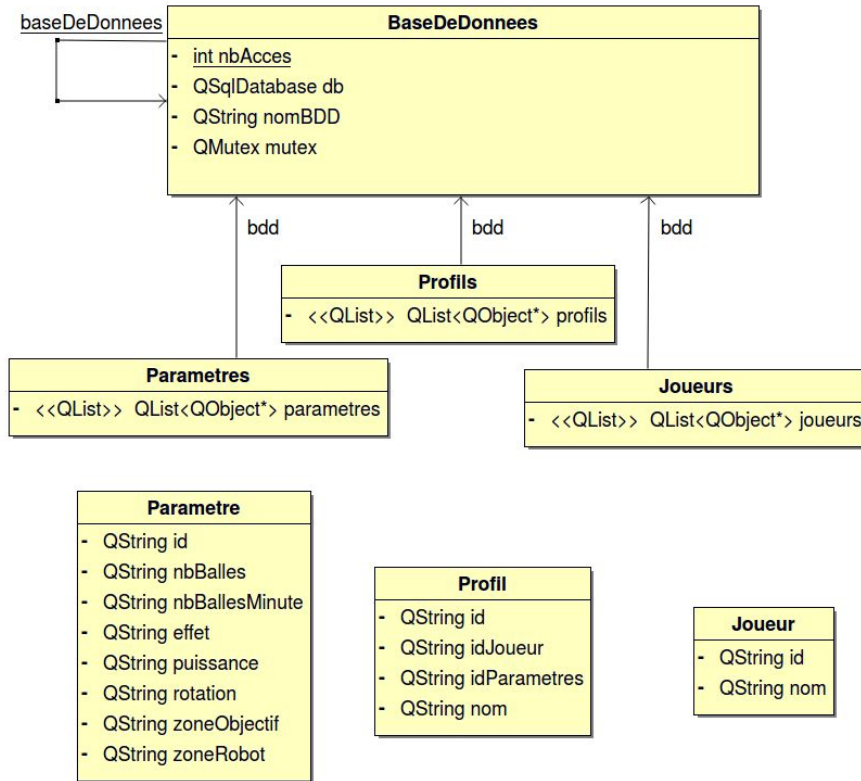
CHOISIR LA POSITION DU ROBOT



Base de do:



# I





# Tests de validation

Description	OUI	NON
La création d'un joueur est possible	X	
Le paramétrage d'une séance d'entraînement est possible	X	
La réalisation d'une séance d'entraînement est possible	X	
La liaison Bluetooth est opérationnelle et les connexions des modules sont affichées	X	
Les informations d'une séance sont transmises aux modules	X	
L'application mobile est déployée	X	

# Conclusion

## État de l'avancement

### Reste à faire :

- Visualisation des statistiques en fin de séance

## Améliorations possibles

- Système de points en fonction des statistiques du joueur
- Déconnexion des appareils TTPA depuis l'application