



### Enoncé du besoin initial du MINI-PROJET complet :

Aujourd'hui, les transports publics traditionnels (bus, tramway et métro) fonctionnent de manière optimale dans les centres-villes denses et les proches banlieues. Pour autant les quartiers à plus faible densité sont souvent moins bien desservis, ce qui contraint l'usager à marcher sur de longues distances ou à avoir recours à son véhicule personnel pour rejoindre la destination de son choix ou accéder à la station du **Réseau de Transports en Commun (RTC)** la plus proche.

Cette desserte insuffisante du premier ou dernier kilomètre a donc un impact direct sur le choix du mode de déplacement du citoyen.



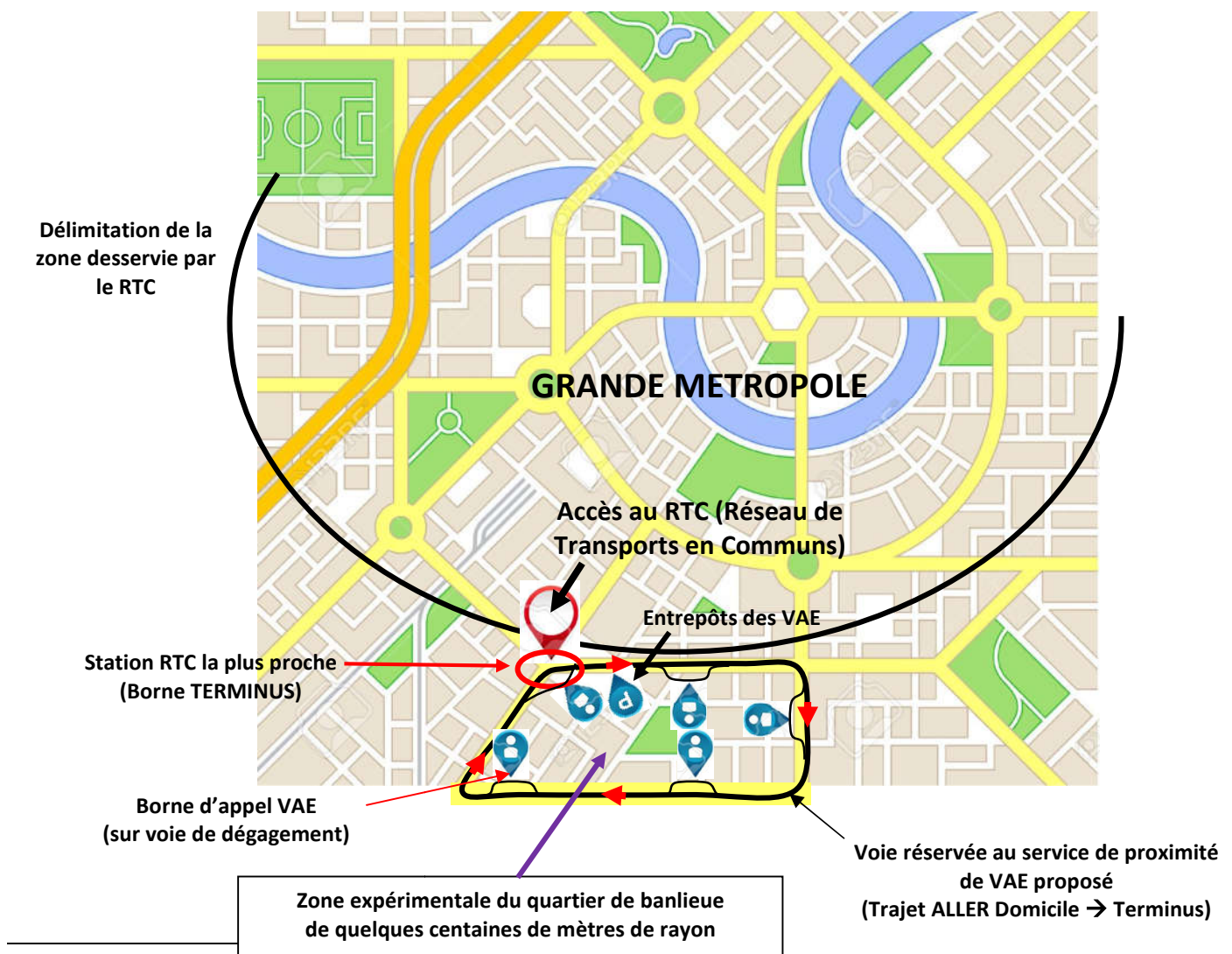
Les intentions sont de développer une mobilité multimodale et décarbonée en complément des services de transports en commun déjà existants. Les enjeux sont doubles :

- **Proposer une mobilité sans contrainte, garantissant :**
  - une continuité totale des déplacements quelque soit la multimodalité<sup>1</sup> ou l'intermodalité<sup>2</sup> ;
  - une diminution significative de l'usage de la voiture individuelle.
- **Favoriser, de fait, la réduction de l'impact environnemental et physique de la mobilité avec :**
  - la maîtrise des consommations d'énergie et des émissions de polluant ;
  - la fluidité d'écoulement de la circulation routière.

<sup>1</sup> La multimodalité désigne la présence de plusieurs modes de transport différents entre deux lieux.

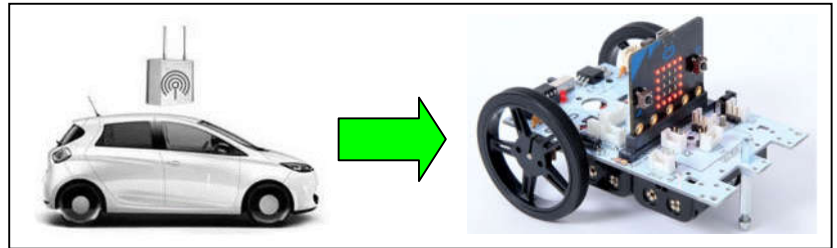
<sup>2</sup> L'intermodalité est l'utilisation de plusieurs modes de transport au cours d'un même déplacement.

La figure qui suit, montre un périmètre d'expérimentation - grandeur nature - qui serait située en zone périurbaine, à proximité d'une grande métropole dotée d'un réseau RTC. Le service de mobilité à la demande doté d'une flotte de **Véhicules Autonomes Electriques (VAE)** circule sur une voie exclusivement réservée à ce type de transport. Elle n'est ici représentée que pour le trajet aller Domicile vers Borne Terminus.

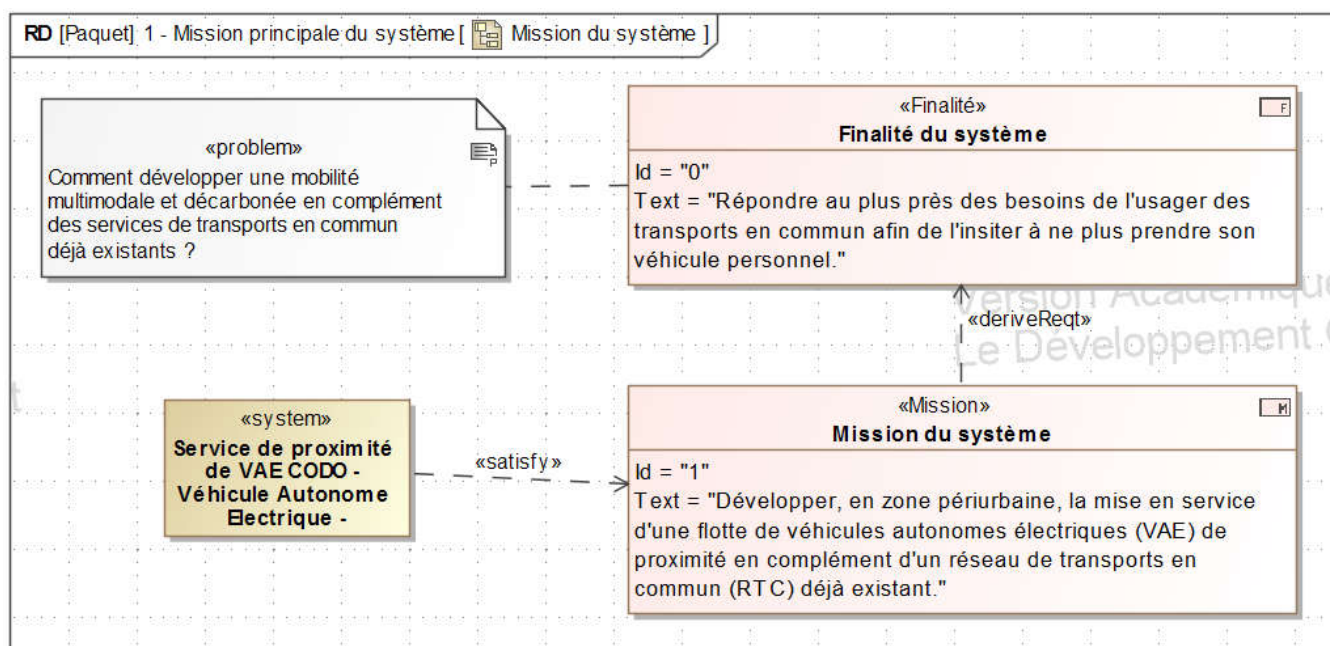


Le VAE doit permettre à tout utilisateur qui en fait la demande, de relier un point d'arrêt à proximité de son domicile à la station du RTC la plus proche dotée d'une borne TERMINUS, le tout en empruntant un circuit bouclé à sens unique de quelques kilomètres. Ce point d'arrêt comporte une borne d'appel d'un VAE mise à la disposition du client. Notons que le chemin du retour (Terminus → Domicile) emprunterait un parcours similaire sur une voie parallèle à l'aide d'une autre flotte de VAE circulant en sens opposé.

Au travers de l'étude du mini-projet dans son ensemble, il s'agit de concevoir et de réaliser à échelle réduite ce service de proximité. La flotte de VAE dédiée au trajet ALLER est fixée de façon arbitraire à 4 véhicules apparentés à 4 Robots CODO ou équivalents. Dans la réalité, le nombre de VAE dépendrait de la fréquence des sollicitations afin de s'assurer que le client n'est pas à s'impatisser plus de quelques minutes.



### Mission principale du MINI-PROJET « SERVICE DE PROXIMITE VAE CODO »



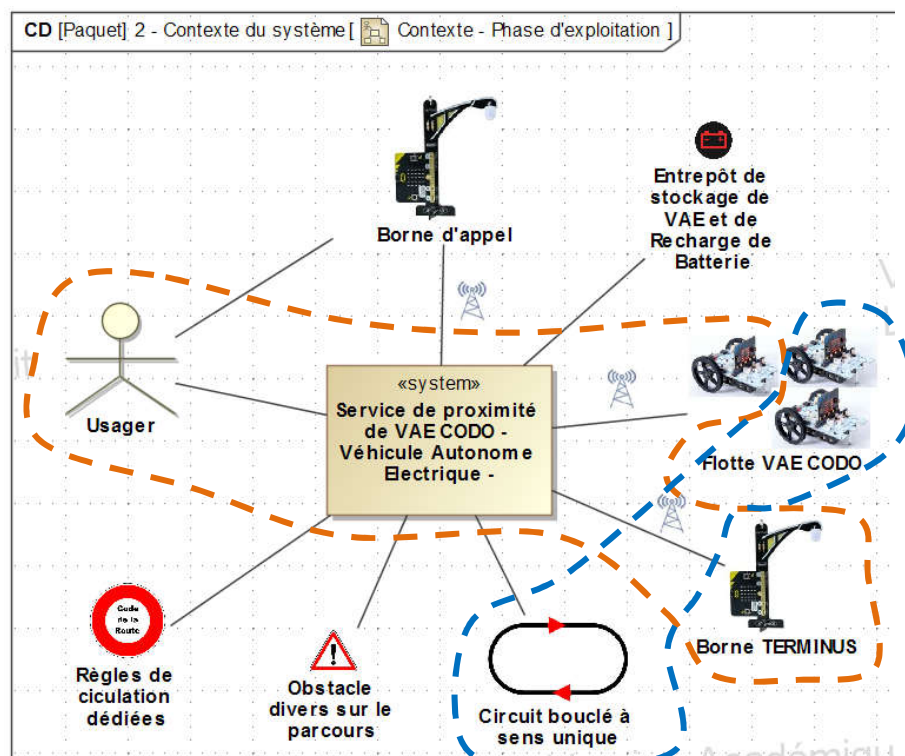
### Diagramme de contexte du MINI-PROJET

Ce diagramme présente les acteurs et les éléments environnant du véhicule autonome en phase d'utilisation.

A l'échelle de ce TP, nous nous limiterons à la réalisation d'une seule exigence par groupe de travail :

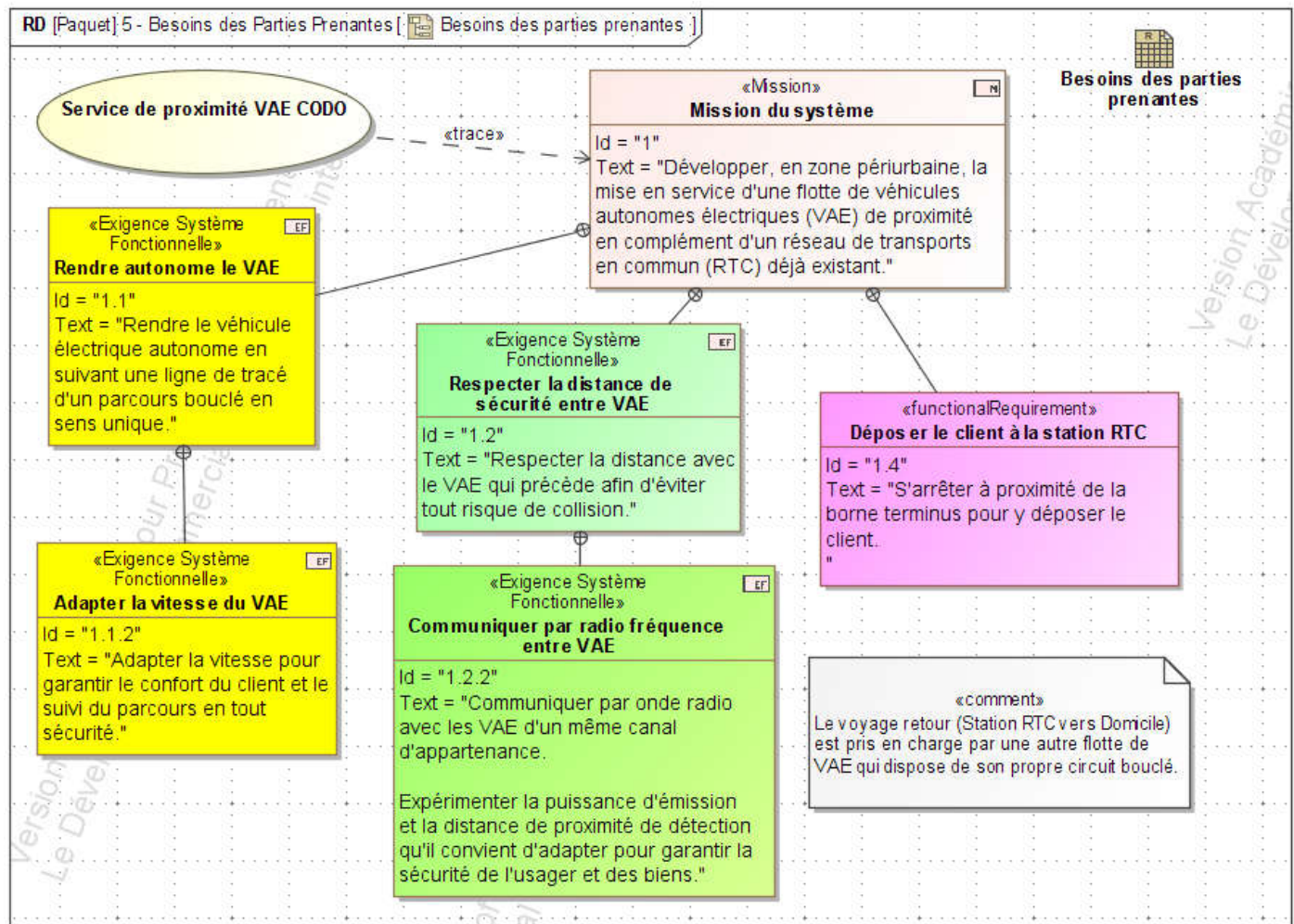
- Soit l'exigence de sécurité entre VAE en lien avec l'anticollision.
- Soit l'exigence de service en lien avec l'arrêt à la borne Terminus

Dans le cas d'un effectif important ou d'une dotation limitée en matériel, les 2 exigences peuvent être abordées et réparties indifféremment. L'une comme l'autre requiert les compétences nécessaires pour l'activité de modulation démodulation numérique qui suivra,





## Diagramme partiel des exigences du MINI-PROJET se limitant à l'exigence traitée dans ce TP.



## Répartition des tâches du TP traitant de l'exigence d'anticollision entre VAE CODO

Activité	Chronologie	Travail		Innover	Expérimenter	Répartition /éq. ou /binôme	INVENTAIRE DES ACTIVITES ET PROGRESSION sur la base de 4 équipes de 4 élèves		Matériel pour 4 équipes
		Collaboratif	Coopératif				Intitulé de l'activité	Problème technique à résoudre	4 CODOS OU 4 BORNES d'APPEL (avec carte µBit + Batterie) + 8 BALISES (carte µBIT alimentées en USB)
ACTIVATION (H1)							ETRE CAPABLE D'EMPRUNTER LA VOIE PRINCIPALE DE FACON AUTONOME ET D'EVITER LES COLLISIONS		
1	H2	X			X	Bin 1 / eq	VAE CODO Suiveur de ligne	Comment suivre un parcours avec la meilleure fluidité de déplacement qui soit ? (Suivre un parcours en autonomie et Adapter la vitesse de déplacement)	1 CODO avec batterie / équipe
2	H2	X			X	Bin 2 /Eq	Communication radio unidirectionnelle entre 2 cartes (simplex ou UNICAST)	Comment programmer une communication unidirectionnelle (simplex ou UNICAST) entre 2 balises ? Comment paramétrer correctement, dans le contexte d'étude, la balise émettrice (TX) et la balise réceptrice (RX) ? (Adapter au mieux la valeur de puissance d'émission et la distance de détection de l'émetteur)	2 balises soit 2 carte µBIT alimentée en USB / équipe
3	H3		X		X	Eq1 &Eq2  Eq3 &Eq4	Respect de la distance de sécurité entre 2 VAE CODO	Comment respecter, en mode simplex ou UNICAST, la distance d'anticollision entre un CODO1 TX et un CODO2 RX ? (Hypothèse simplificatrice : Le VAE CODO 1 reste toujours devant le VAE CODO 2 Valider les paramètres de l'activité 2)	1 CODO TX avec batterie pour équipe 1 (idem eq3) 1 CODO RX avec batterie pour équipe 2 (idem pour eq4)

## Répartition des tâches du TP traitant de l'exigence d'arrêt du VAE CODO à la borne Terminus

<b>ACTIVATION (H1)</b>							<b>ETRE CAPABLE DE S'ARRETER A LA BORNE TERMINUS POUR Y DEPOSER LE CLIENT</b>		
1 OU 4	H2 H3	X			X	/ Bin	Arrêt Terminus	Comment arrêter la course du VAE CODO une fois la borne terminus atteinte ?	1 BORNE D'APPEL avec batterie + 1 une simple carte µBIT en remplacement du CODO / binôme

## HORS TEMPS SCOLAIRE

HORS TEMPS SCOLAIRE							Restitution des travaux par groupe de travail – Présentation type POWER POINT...		
---------------------	--	--	--	--	--	--	--	--	--

Travail coopératif # Typologie d'activités plutôt complémentaires

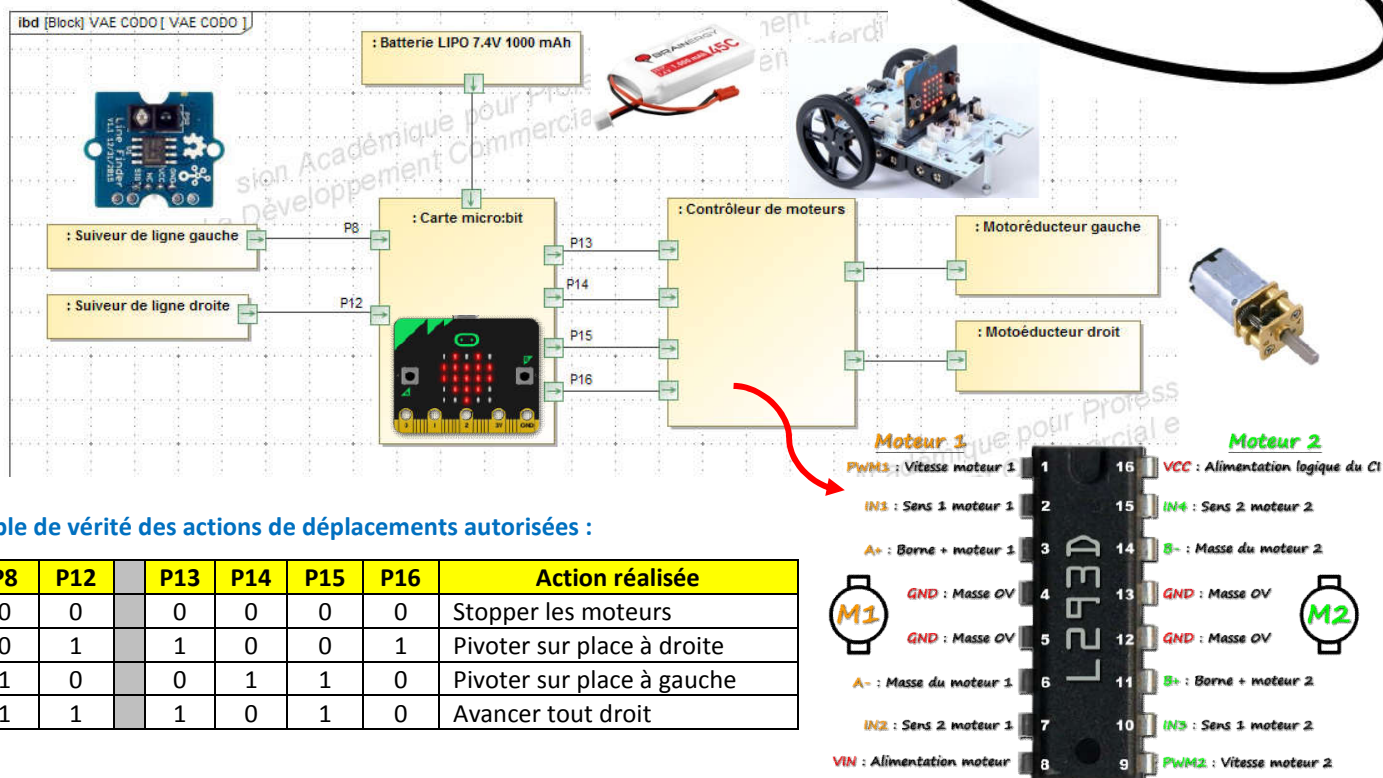
Travail collaboratif # Typologie d'activités plutôt communes

## ETRE CAPABLE D'EMPRUNTER LA VOIE PRINCIPALE DE FACON AUTONOME

## Activité 1 - Gestion du VAE CODO en suiveur de ligne

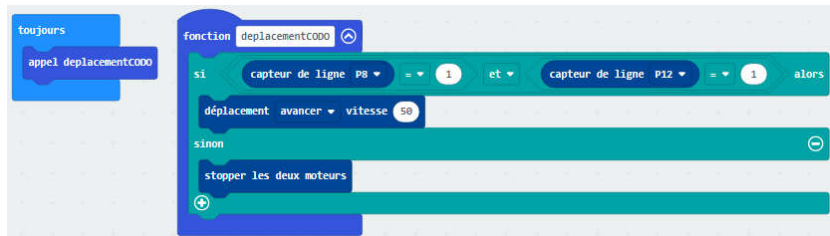


- 1- Mettre au point le programme de la carte micro :bit permettant le suivi de ligne d'un parcours bouclé à une vitesse de mobile favorisant la fluidité du déplacement.



## Exemple de programmation sous MAKECODE et son code PYTHON associé

VAE CODO en marche avant tant que la ligne est détectée.



NB : Ce programme fait appel au sous-programme « deplacementCODO »

```

def deplacementCODO():
    if CODO.detectline(DigitalPin.P8) == 1 and CODO.detectline(
        DigitalPin.P12) == 1:
        CODO.robot_move(CODO.RobotDirection.FORWARD, 50)
    else:
        CODO.motor_stop()

def on_forever():
    deplacementCODO()
    basic.forever(on_forever)

```

**Attention !!! Si vous utilisez MAKECODE, penser à télécharger l'extension « CODO »**



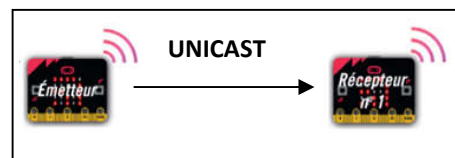
- 2- Préciser le nom et la valeur du paramètre de réglage retenu.

## ETRE CAPABLE D'EVITER LES COLLISIONS ENTRE LES VAE CODO (travail préliminaire)

### Activité 2 – Communication radio entre 2 cartes micro:bit



- 3- Programmer une communication radio unidirectionnelle (simplex ou unicast) entre une carte émettrice (TX) et une carte réceptrice (RX) afin de détecter sur la carte RX la proximité de la carte TX.



Voir la fiche d'information sur la communication radio de la carte micro:bit en annexe.



- 4- Adapter la valeur des paramètres de puissance d'émission et de distance de détection au plus près des besoins du mini-projet. Compléter le tableau de mesure suivant et préciser le couple de valeurs à retenir.

Puissance de transmission (Variable Force de 0 à 9)	Variable Distance (de 0 à 9)	Estimation de la distance de détection en cm
1	3	
1	2	
1	1	
0	5	
0	4	
0	3	

## ETRE CAPABLE D'EVITER LES COLLISIONS ENTRE 2 VAE CODO

### Activité 3 - Gestion du respect des distances de sécurité entre 2 VAE CODO



- 5- Adapter les programmes précédents sur les VAE CODO (CODO1 en TX et CODO2 en RX) afin de valider le couple de paramètres retenus. S'assurer d'un maximum de fluidité de parcours sans risque de collision mutuelle.



**Règle de circulation :** Le CODO1 reste toujours devant le CODO2.



- 6- Préciser les noms et les valeurs des paramètres de réglages retenus.



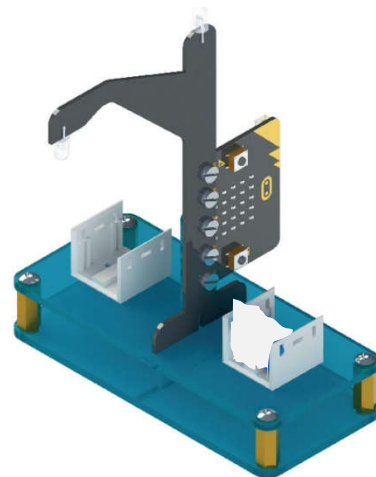
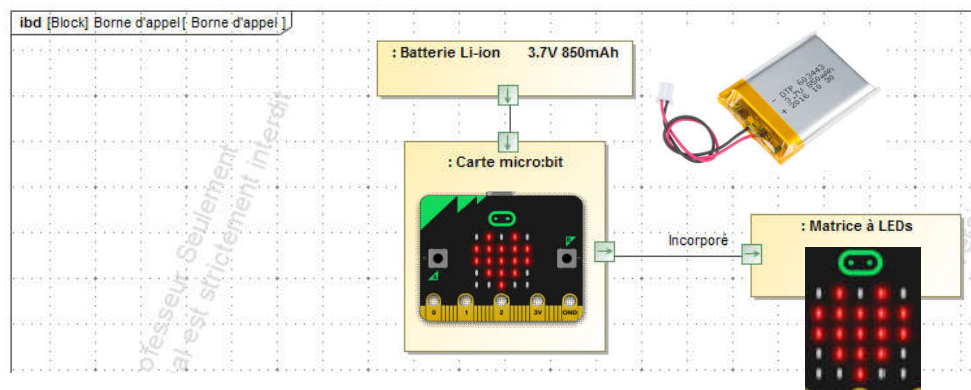
- 7- L'exigence d'anticollision est-elle satisfaite ?



## ETRE CAPABLE DE S'ARRETER A HAUTEUR DE LA BORNE TERMINUS POUR Y DEPOSER LE CLIENT

## Activité 1 - Arrêt TERMINUS

## DIAGRAMME INTERNE DE BLOC DE LA BORNE TERMINUS EMETTRICE (TX)

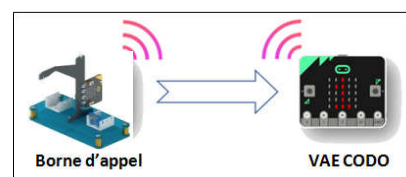


Voir la fiche d'information sur la communication radio de la carte micro:bit en annexe.



1- A l'aide de la borne Terminus et d'une autre carte micro:bit (en remplacement du VAE CODO), concevoir les deux programmes (émetteur –TX- et récepteur -RX-) qui permettent :

- **D'émettre un signal radio périodique** depuis la borne Terminus ;
- **De réceptionner cet appel** depuis la carte micro:bit de substitution ;
- **De signaler la prise en compte de l'arrêt par le VAE.**



**CONSEIL :** Il semble judicieux de s'aider de la matrice à LED de chaque carte et d'user de l'affichage de divers icônes aisément programmables pour élaborer et mettre au point ces 2 programmes.



2- Tester la proposition et s'assurer de son bon fonctionnement.

3- Compléter le tableau de mesure suivant.

Adapter la valeur des paramètres de puissance d'émission et de distance de détection au plus près des besoins du cahier des charges sachant que ce dernier impose une distance d'arrêt inférieure ou égale à une quinzaine de centimètres.

Préciser les valeurs du couple retenu

Puissance de transmission (Variable Force de 0 à 9)	Variable Distance (de 0 à 9)	Estimation de la distance de détection en cm
1	3	
1	2	
1	1	
0	5	
0	4	
0	3	



4- L'exigence d'arrêt à hauteur de la borne terminus est-elle satisfaite ?



**ANNEXE - Fiche d'information sur la communication radio avec la carte micro:bit**

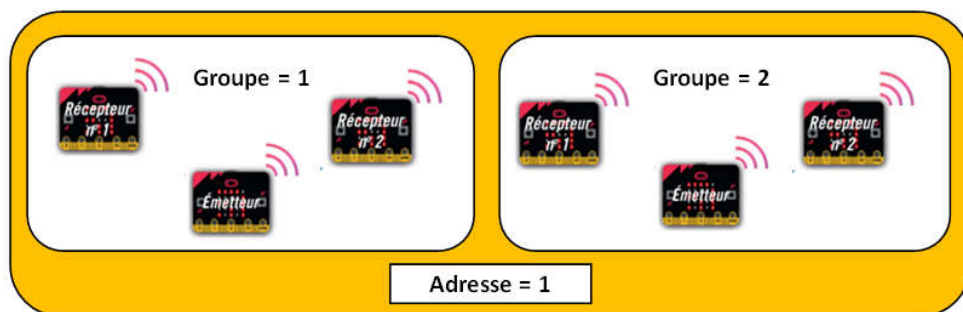
L'unité centrale de traitement (CPU) sur le micro: bit est un Nordic Semiconductor [nRF51822](#). En plus d'être un processeur informatique à usage général, cette puce contient un module émetteur radio 2,4 GHz et un module récepteur radio 2,4 GHz qui utilisent le protocole propriétaire Gazell de Nordic pour pouvoir échanger les données entre cartes micro:bit.

Lorsqu'une donnée est envoyée par un module émetteur, elle est diffusée à l'ensemble des cartes réceptrices qui sont dans le champ de portée de réception du signal radio.

L'envoi de données n'étant pas ciblé vers un destinataire, toutes les cartes décodent la même information en réception tel une échange *broadcast*. Pour pouvoir cibler le destinataire, il est nécessaire d'identifier ce dernier sur le réseau. Le système d'identification fonctionne comme suit : l'émetteur doit spécifier une adresse et un groupe d'appartenance.

Une carte micro:bit est donc identifiée par son adresse et son groupe.

L'exemple de la figure ci-contre propose 2 groupes à la même adresse. Les cartes du groupe 1 peuvent communiquer entre elles (idem pour les cartes du groupe 2) cependant les cartes du groupe 1 ne peuvent pas communiquer avec celle du groupe 2 même si elles ont la même adresse.



**Le paramètre adresse** (par défaut = 0x75626974) est un nom exprimé sous forme d'adresse 32 bits. Il est utilisé pour filtrer les messages entrants au niveau matériel.

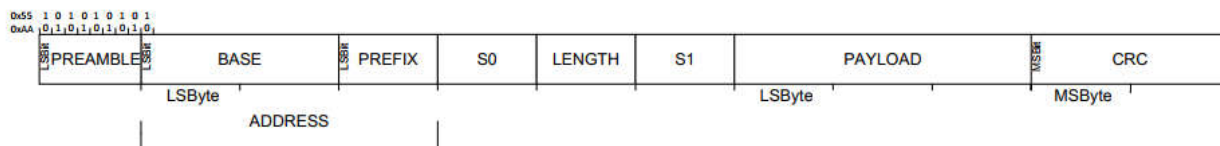
**Le paramètre groupe** (par défaut = 0) est un entier compris entre 0 et 255. Comme l'adresse, il est utilisé lors du filtrage des messages.

**Autres paramètres radio configurables** → Plusieurs paramètres peuvent être réglés pour ce type de communication à distance :

- **Le réglage de la puissance d'émission** (de 0 à 7). La valeur 0 correspond à une faible portée et une faible consommation tandis que la valeur 7 correspond à une portée maximale de plusieurs dizaines de mètres. Ces valeurs correspondent à une puissance émise en dBm avec respectivement -30, -20, -16, -12, -8, -4, 0 et 4 dBm. Pour 0 dBm, la portée est d'environ 20 mètres.
- **Le réglage du débit binaire** est également possible avec les commandes `radio.RATE_250KBIT`, `radio.RATE_1MBIT` et `radio.RATE_2MBIT` qui permettent d'avoir respectivement des vitesses de transmission de 256Kbit/s, 1Mbit/s et 2 Mbit/s.
- **Le réglage du canal** avec la commande `radio.config(chanel=7)` par défaut. Un canal compris entre les valeurs 0 et 83 correspond en fait à une fréquence donnée autour de 2.4 GHz par pas de 1 MHz. Un changement de canal permet de créer un autre réseau au même titre qu'une adresse et un groupe.
- **La longueur du message** (Length = nombre de caractères du message) est définie à 32 par défaut. Mais on peut modifier ce nombre à l'aide de la commande `radio.config(length=57)` la valeur maximum pouvant alors atteindre 251.
- **Le nombre de messages dans la file d'attente** : `radio.config(queue=3)`. Au delà de 3 messages en attente, ils seront supprimés.

**Configuration des paquets radios**

La radio envoie les différents champs du paquet dans l'ordre suivant : PREAMBULE, ADRESSE, S0, LENGTH, S1, PAYLOAD et CRC comme illustré ci-dessous de gauche à droite.



**Le champ PREAMBULE** est toujours long d'un octet. Son LSB est envoyé en premier à l'antenne. Si le premier bit de l'ADRESSE est 0 alors le préambule sera mis à 0xAA sinon le PREAMBULE sera mis à 0x55. **Le champ ADRESSE** de l'onde radio est composé de 2 parties : la base (réglable de 0 à 7) et le préfixe. Seules les données en provenance d'une carte présentant la même adresse seront pris en compte.

**Les champs S0, LENGTH, S1, PAYLOAD** concernent les données applicatives. La longueur combinée de ses données ne peut pas dépasser 254 octets (3 octets pour S0, LENGTH, S1 et 251 octets pour le message).

**Le champ CRC** de 16 bits (Contrôle de Redondance Cyclique) qui débute par l'octet le plus significatif permet de détecter d'éventuelles erreurs de transmission.

Les messages reçus sont stockés dans une file d'attente de la mémoire vive (RAM) de la carte micro:bit type FIFO. Par défaut, le nombre maximum de messages est de 3 mais il est possible d'augmenter la taille de la file d'attente. Notons que plus elle est grande et plus la mémoire utilisée est importante. La lecture d'un message efface ce dernier de la file d'attente, mais la mémoire est pleine, les nouveaux messages sont ignorés. Les messages sont classés par adresse et par groupe.





## Programmation de la communication radio

L'outil **Radio** sur le logiciel MakeCode (par exemple) permet la communication entre plusieurs cartes micro:bit. Il est nécessaire de créer deux programmes, un pour chaque carte micro:bit. Elles doivent être programmées selon le même « groupe radio » pour que les programmes fonctionnent

Voici l'essentiel des instructions et fonctions utilisées pour la communication radio :

Instruction ou fonction	Bloc MAKECODE	Code PYTHON
Définir l'identifiant du groupe (0 à 255)	radio définir groupe 1	<code>def radio.set_group(id: int32): None</code>
Modifier le niveau de puissance en sortie d'émetteur (0 à 7)	radio définir puissance de transmission 7	<code>def radio.set_transmit_power(power: int32): None</code>
Retourner les propriétés du dernier paquet radio reçu	paquet reçu force du signal	<code>def radio.received_packet(type: number): number</code>
Envoyer une information à l'aide de paquets radio (un nombre ou un nom associé à un nombre ou une chaîne de caractères)	envoyer le nombre 0 par radio envoyer la valeur "name" = 0 par radio envoyer la chaîne "" par radio	<code>def radio.send_number(value: number): None</code> <code>def radio.send_value(name: str, value: number): None</code> <code>def radio.send_string(value: str): None</code>
Réceptionner une information radio (un nombre ou un nom associé à un nombre ou une chaîne de caractères)	quand une donnée est reçue par radio receivedNumber quand une donnée est reçue par radio name value quand une donnée est reçue par radio receivedString	<code>def radio.on_received_number(cb: (number) -&gt; None): None</code> <code>def radio.on_received_value(cb: (str, number) -&gt; None): None</code> <code>def radio.on_received_string(cb: (str) -&gt; None): None</code>
Changer la bande de transmission d'émission et de réception au canal donné (0 à 83)	radio régler la bande de fréquence 0	<code>def radio.set_frequency_band(band: int32): None</code>
Envoyer un évènement par radio aux périphériques environnants	radio déclencher l'évènement de la source MICROBIT_ID_BUTTON_A avec la valeur MICROBIT_EVT_ANY	<code>def radio.raise_event(src: int32, value: int32): None</code>

### Exemples de programme radio (sous MAKECODE et le code PYTHON associé)

Communication unidirectionnelle (simplex ou UNICAST) entre 2 cartes micro:bit avec affichage du nombre reçu sur le récepteur une fois la détection à une certaine distance de l'émetteur effectuée		
<b>Côté émetteur (TX) :</b>		<pre>radio.set_group(1) radio.set_transmit_power(7)  def on_forever():     radio.send_number(1)     basic.forever(on_forever)</pre>
<b>Côté récepteur (RX) :</b>	 <p>NB : <b>Projeter...</b> → ré-assignation d'un nombre au sein d'une plage (entre -98 et -45) à une nouvelle plage (entre 9 et 0) La force du signal peut se situer entre -98 (plus faible valeur) et -45 (plus forte valeur).</p>	<pre>def on_received_number(receivedNumber):     global Force, Distance     Force = radio.received_packet(RadioPacketProperty.SIGNAL_STRENGTH)     Distance = Math.map(Force, -98, -45, 9, 0)     if Distance &lt;= 4:         basic.show_number(receivedNumber)     else:         basic.clear_screen()     radio.on_received_number(on_received_number)  Distance = 0 Force = 0 radio.set_group(1)</pre>