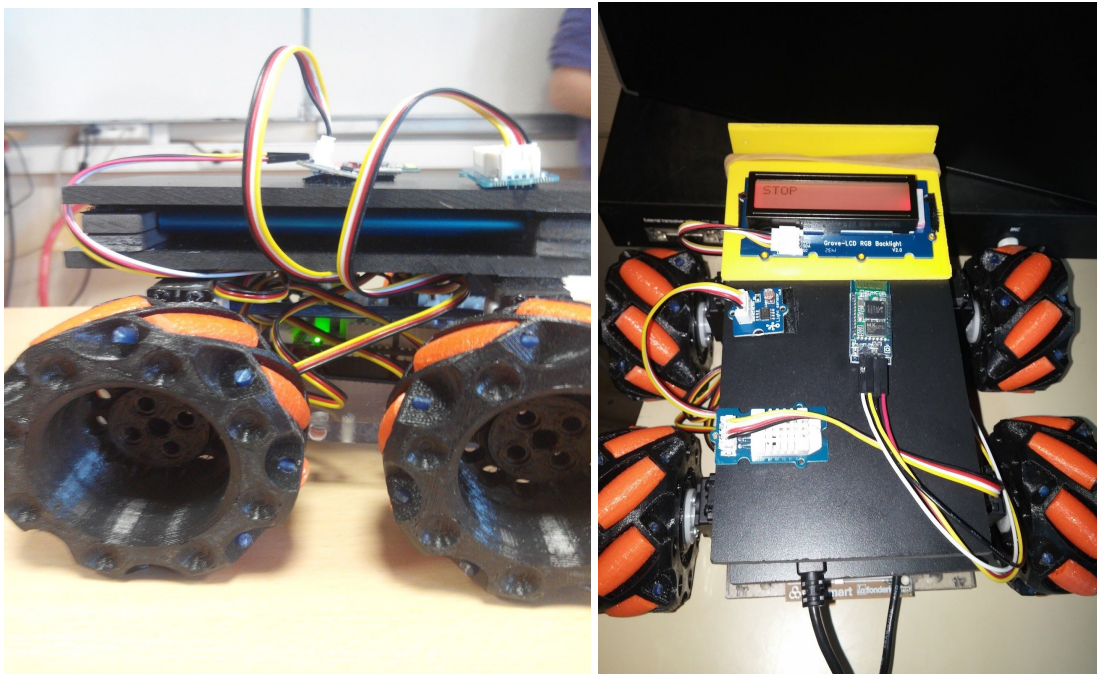


EDUCADUINO MECANUM

Projet de développement d'un robot tout terrain capable de relever des données en zone à risque



Concours Educaduinov 2016

Participants au projet :

Toute la classe de 4^eC et 6 élèves de 3^eE et 3^eB

Collège Victor Duruy

Sommaire :

- Présentation du projet
- Contraintes et contexte de réalisation
- Ressources mises à disposition des élèves (logiciels / matériels / documents)
- Diagrammes de cas d'utilisation
- Cahier des charges
- Diagramme de déploiement
- Coût des matériaux
- Algorigrammes de tests des différents modules et des modes envisagées
- Présentation de l'application
- Maquette
- Modélisation du projet sous sketchup
- Planification (Gantt)
- Améliorations possibles
- Conclusion

Concours :

<http://www.educaduino.fr/concours-educaduinov-eurosmart/>

Présentation :

Ce projet est mis en oeuvre par une classe de 4^e plus 6 élèves d'autres classe de 3^e, le tout géré par Mr. Karl THOMAS, professeur de technologie au collège Victor Duruy.

Le but au final est de pouvoir présenter un robot tout terrain qui pourra, à terme, prendre des informations sur son environnement grâce aux capteurs qu'il embarquera.

Ces environnements pourront être, par exemple, des volcans en activité ou encore dans des sites archéologiques, etc... Globalement des zones à risques.

Contraintes et contextes de réalisation :

Les contraintes de ce projet ont été le matériel manquant et inapproprié, les outils et machines déficientes. Ce projet a été très complexe à réaliser et le contexte très difficile car le fait d'être dans une salle de classe et que le professeur devait s'occuper des autres élèves et aussi par le manque d'heures qui était de deux heures par semaines.

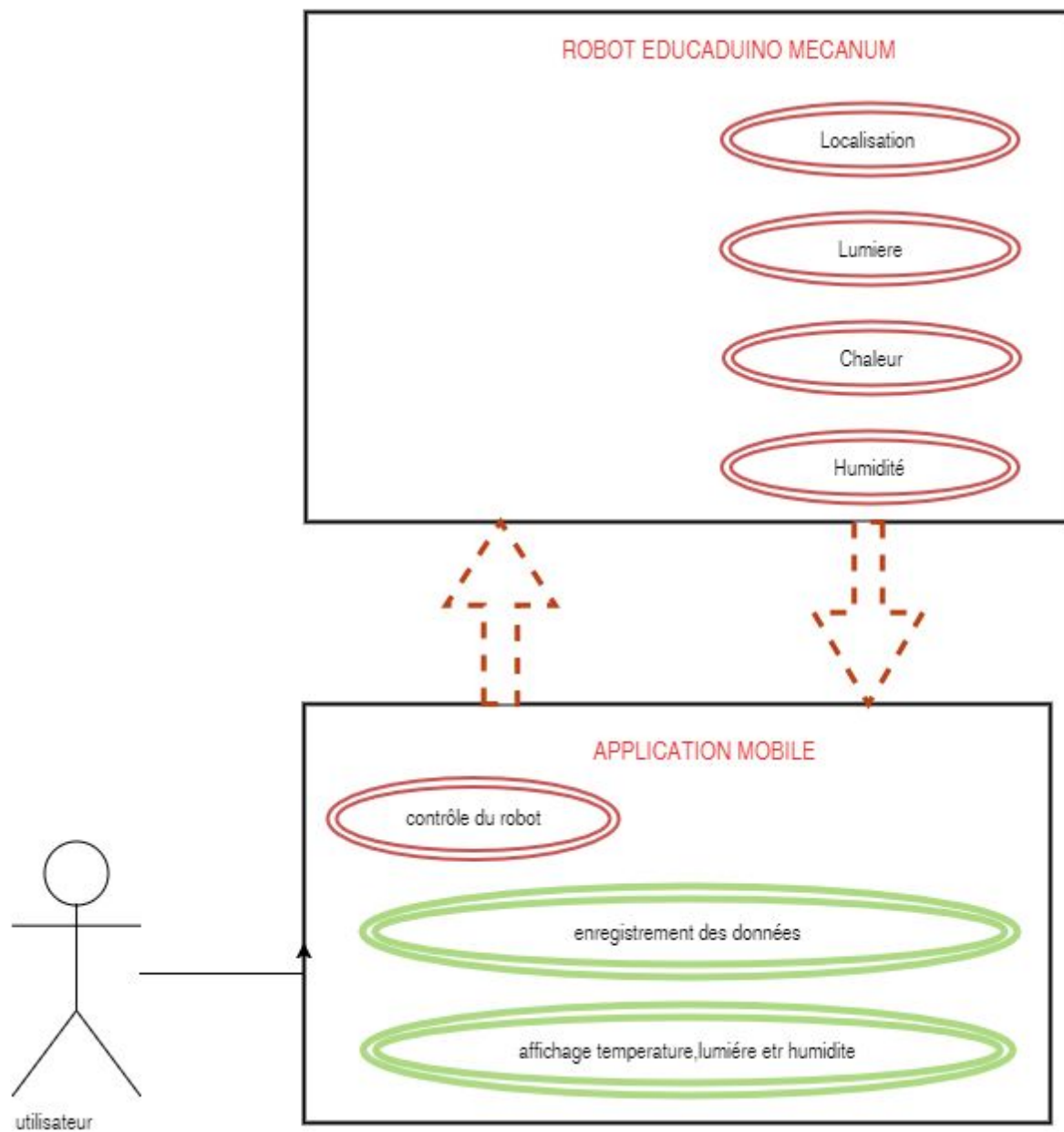
Mais aussi à cause d'un incident dans la classe ce projet a été retardé ce qui a causé une réelle perte de temps qui aurait pu être évitée.

Les ressources mises à disposition des élèves :

Les logiciels qui nous ont aidés à nous organiser sont:

- MIT App Inventor qui permet de programmer une application.
- Arduino (ardublock) pour programmer le robot lui-même.
- Kanbanchi pour tout planifier.
- Draw.io pour les diagrammes.

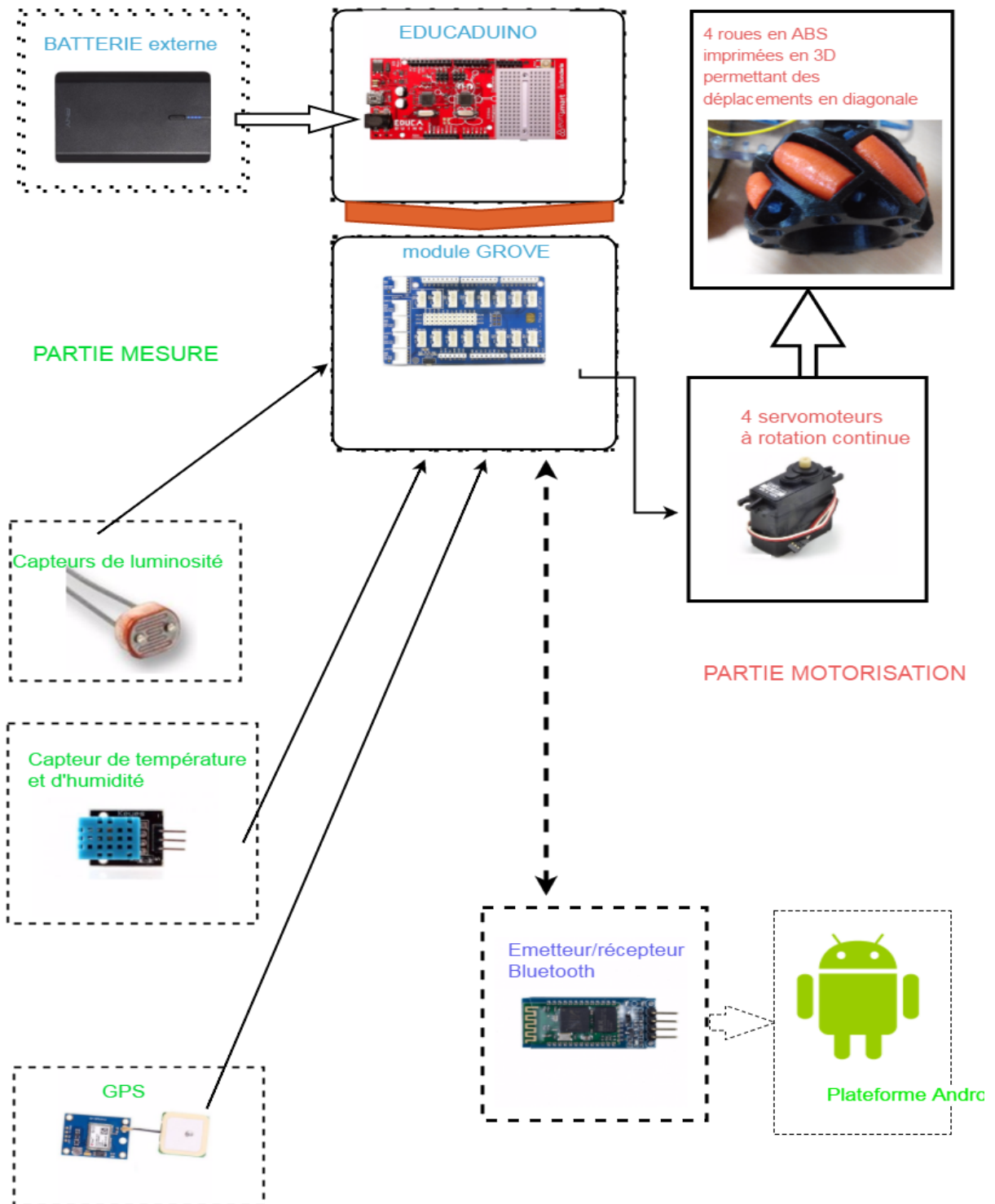
Diagramme de cas d'utilisation :



Cahier des charges :

Fonction/contraintes	Critères d'appréciation	Niveaux de performance et flexibilité	
FP1	Pouvoir s'aventurer dans des zones a risque	Type de robot	Robot comportant 4 roues motrices <u>mecanum</u>
		Moyen de déplacement tout-terrain	
FC1	Pouvoir récupérer des informations dans des zones a risques	Capteurs d'informations	Thermique lumière humidité GPS
FC2	Types de matériaux	Avoir des matériaux économiques	Plexiglas plastique d'impression 3D <u>filaflex</u> et ABS plaque PVC
FC3	Avoir des matériaux de qualité	Résistance en milieu extérieur	Arrachement, choc , chute(25cm max), radioactivité, chaleur extrême
FC4	Être alimenté en énergie	Type d'alimentation	Embarquée, rechargeable (batterie)
FC5	Être ergonomique	Prise en main	facile
FC6	Ne pas dépasser le budget prévu	Prix	Environ 250 €
FC7	Être esthétique	Forme	Simple, rectangulaire
		Couleurs	aléatoires
FC8	Respecter les normes de sécurité	normes	Éléments non coupants, non toxiques, etc
FC9	Pouvoir accéder facilement aux composants du robot.	Éléments détachables	Velcro (scratch)
FC10	Doit intégrer un système de pilotage	Carte électronique	<u>arduino</u>
		Système de connexion sans fil	Bluetooth
		Système de guidage	smartphone

Diagramme de déploiement :



Coût des matériaux :

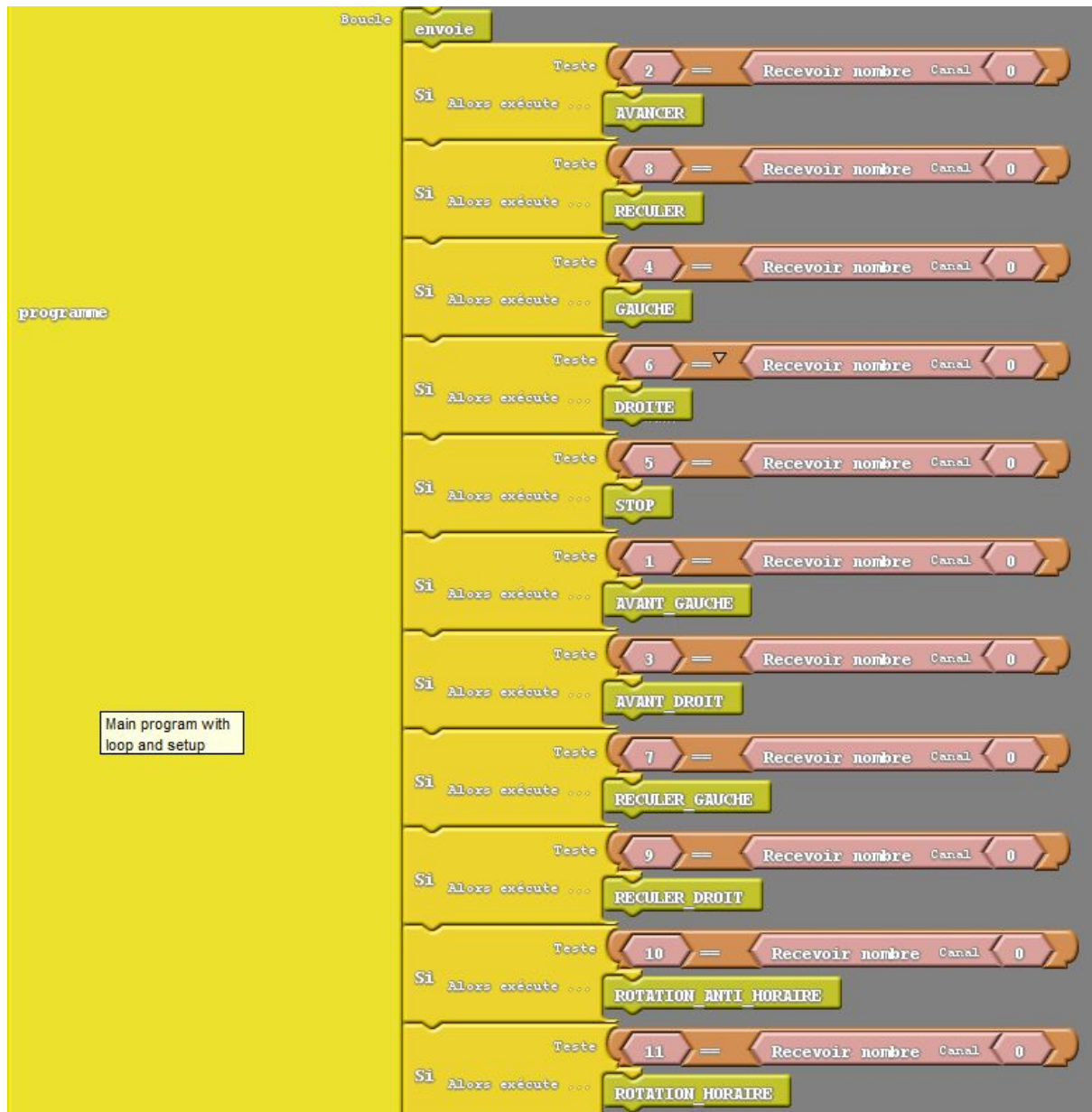
1 Batterie portable	34,98 €
Capteur de température et d'humidité	7,08 €
Capteur de luminosité "grove"	3.48
Carte "educaduino plus "	33,00 €
4 servos moteurs	18,50 € x 4
Plaque de plexiglas	9,00 €
Shield Grove Mega	13,88 €

Algorigrammes :

Voyons comment fonctionne le programme du côté de la carte educaduino v :

Cette partie du programme permet d'initialiser dans le setup les servos moteurs pour qu'ils ne bougent pas.





Cette partie du programme permet la réception de données.

Si on reçoit 1 comme valeur le robot se déplace en diagonale gauche vers l'avant.

Si on reçoit 2 comme valeur le robot avance.

Si on reçoit 8 comme valeur le robot recule.

Etc...



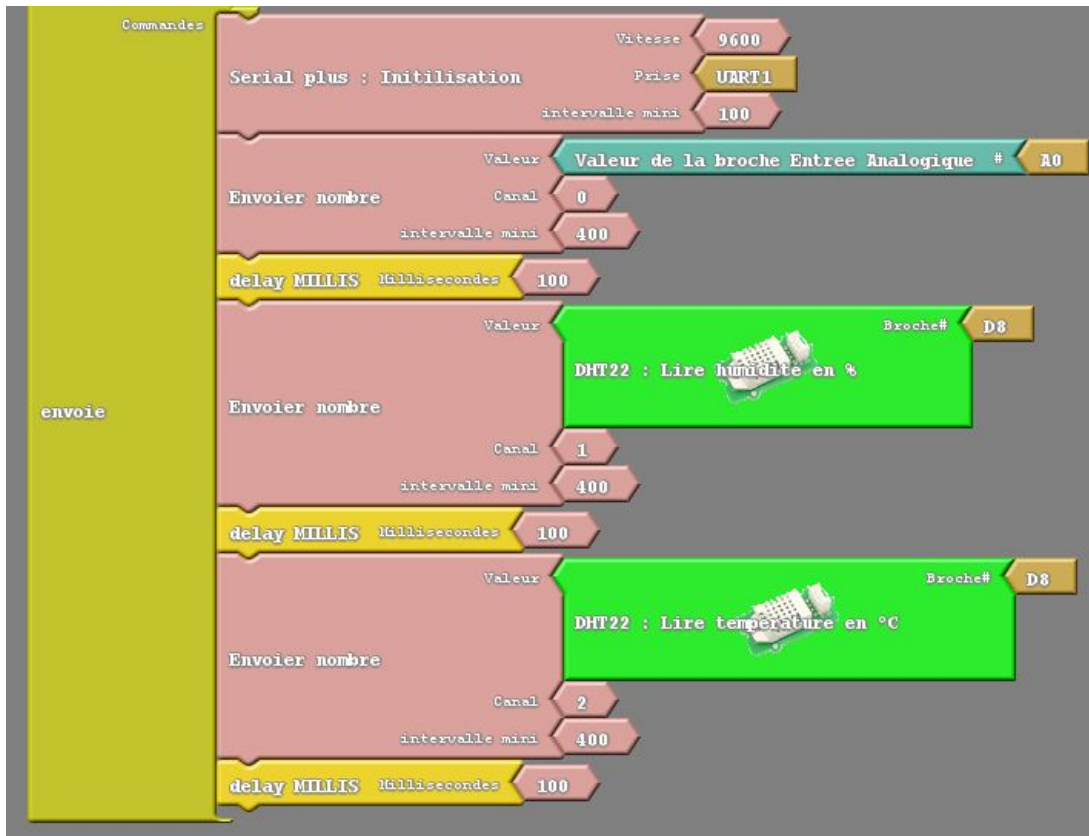
La valeur pour permettre aux servomoteurs moteurs de tourner dans le sens antihoraire est 170.



La valeur pour permettre aux servomoteurs moteurs de s'arrêter est 95.

Voici la partie du programme qui permet d'envoyer les différentes valeurs sur différents canaux

Pour avoir toutes les informations séparément, nous envoyons les valeurs des capteurs de lumière, d'humidité et de lumière tous les 100ms.



- Ecran LCD :



Dans l'image ci-dessus, L'écran LCD permet d'afficher un texte lors d'une action précise. Si on fait reculer le robot l'écran LCD affichera le texte "reculer" et avec le deuxième bloc nous pouvons choisir la couleur de l'affichage du texte.

Nous faisons ceci pour chaque déplacement du robot.

Présentation de l'application :

Ce robot est équipé d'une carte "educduino plus", qui gère le pilotage et les données obtenues par différents capteurs qui sont situés sur le robot.

Le smartphone est connecté par bluetooth à la carte "educduino plus". Le smartphone affiche les données enregistré par les capteurs, et avec quelques petites modifications, elle pourrait être remplacée par une liaison d'un autre type (ex: radio).

Du côté du smartphone, le pilotage est interfacé par une application créée grâce au site de développement d'applications "MIT App Inventor 2" disponible sur toutes les plateformes.

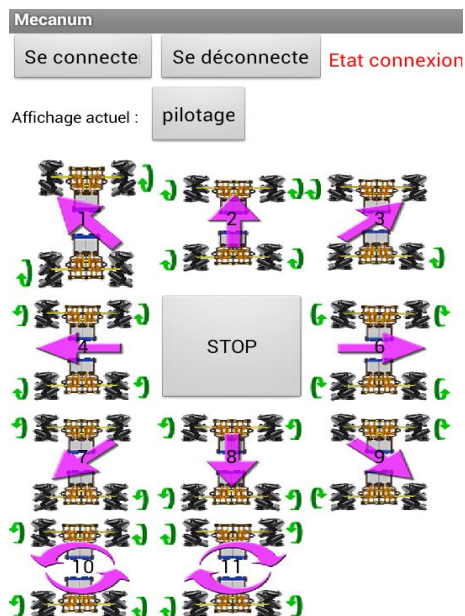
L'application dispose de deux modes de pilotage ; un mode où la direction est assurée par des flèches directionnelles, l'autre par l'accéléromètre du téléphone gérant les 2 axes X et Y.

Les valeurs des capteurs sur le robot sont renvoyées vers le smartphone et les valeurs sont tracés sur le smartphone.

Grâce à deux smartphones nous pouvons aussi voir ce que voie le robot en live en lui attachant un smartphone et le connectant à un ou plusieurs autres smartphones grâce à l'application "Alfred".

Modes de pilotage :

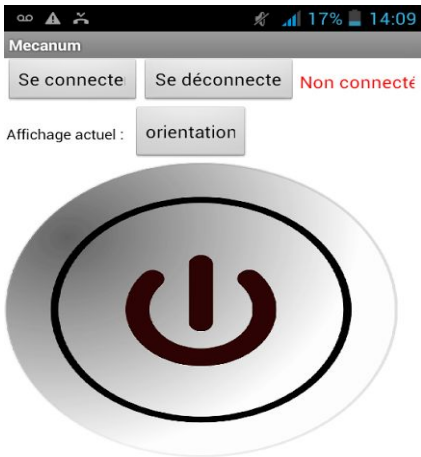
Les flèches directionnelles :



La direction est gérée par des flèches directionnelles : Celles pur la droite, gauche avant et arrière, mais aussi celles gérant les 4 diagonales et les deux demi-tours.

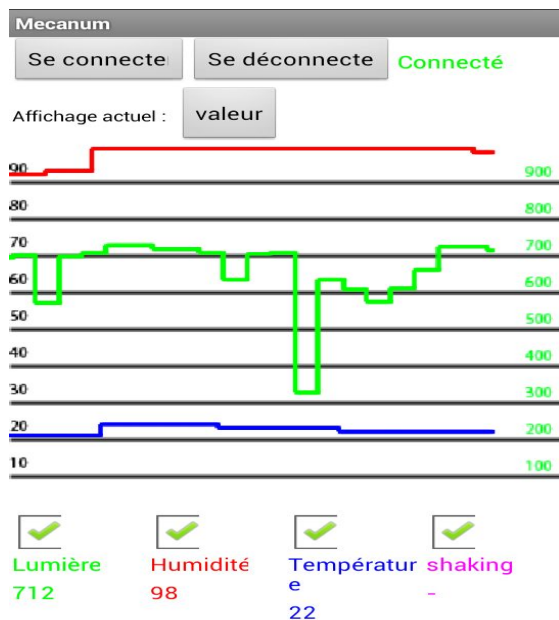
Orientation avec accéléromètre :

Le bouton permet d'activer le pilotage du robot avec l'inclinaison du smartphone.



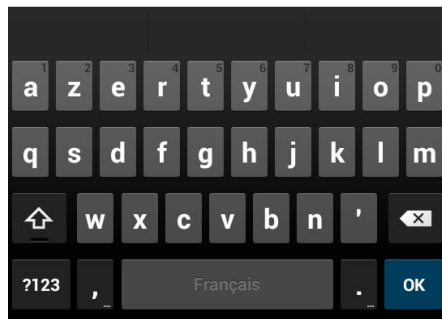
L'affichage et le Traçage :

Ceci est une représentation graphique des valeurs de l'humidité, de la température et de la lumière. Elles sont envoyée directement de la carte "educaduino plus" au smartphone grâce au bluetooth.



L'enregistrement des données des capteurs :

Ci-dessous l'écran qui permet d'enregistrer les valeurs sous la forme d'un texte. Dans le rectangle orange, on nomme le document qui va être créé comme dans l'exemple: réception.txt.



le fichier qui vient d'être enregistré, avec les valeurs pour :

l'humidité, la lumière, la température.

Ci-dessous le texte des valeurs envoyé dans la mémoire du smartphone.

```
""Humidite"" ""0"" ""0"" ""0"" ""0"" ""0"" ""0"" ""
0"" ""0"" ""0"" ""0"" ""0"" ""0"" ""0"" ""0""
""0"" ""0"" ""0"" ""0"" ""0"" ""58"" ""58"" ""58""
58"" ""58"" ""58"" ""58"" ""58"" ""58"" ""58""
""85"" ""85"" ""85"" ""83"" ""83"" ""83"" ""83""
9"" ""69"" ""69"" ""69"" ""66"" ""66"" ""64""
""60"" ""60"" ""60"" ""60"" ""60"" ""60"" ""60"" ""6
"" ""57"" ""57"" ""57"" ""57"" ""57"" ""57""
""56"" ""56"" ""56"" ""56"" ""56"" ""56"" ""56"" ""56
"" ""56"" ""56"" ""56"" ""56"" ""56"" ""56"" ""56""
56"" ""56"" ""56"" ""56"" ""56"" ""56"" ""56"" ""56
"" ""56"" ""56"" ""56"" ""56"" ""56"" ""56"" ""56""
6"" ""56"" ""56"" ""56"" ""56"" ""56"" ""56"" ""56""
""56"" ""56"" ""56"" ""56"" ""56"" ""56"" ""56"" ""55
"" ""55"" ""55"" ""55"" ""55"" ""55"" ""55"" ""55""
""79"" ""80"" ""80"" ""80"" ""80"" ""78"" ""78"" ""7
""Lumiere"" ""0"" ""0"" ""0"" ""0"" ""0"" ""0"" ""
0"" ""0"" ""0"" ""0"" ""0"" ""0"" ""0"" ""0""
""0"" ""0"" ""0"" ""0"" ""0"" ""0"" ""0"" ""0""
0"" ""0"" ""245"" ""245"" ""237"" ""236"" ""2
239"" ""250"" ""250"" ""250"" ""250"" ""250""
9"" ""259"" ""261"" ""261"" ""245"" ""247"" ""
""256"" ""250"" ""250"" ""250"" ""249"" ""249"" ""26
248"" ""248"" ""259"" ""256"" ""256"" ""259""
"" ""259"" ""259"" ""259"" ""259"" ""259"" ""
""254"" ""252"" ""252"" ""259"" ""259"" ""259
58"" ""258"" ""252"" ""252"" ""255"" ""248""
"" ""250"" ""250"" ""250"" ""250"" ""259"" ""2
260"" ""260"" ""261"" ""261"" ""259"" ""259""
0"" ""260"" ""254"" ""254"" ""265"" ""264"" ""
""259"" ""259"" ""248"" ""248"" ""248"" ""24
248"" ""248"" ""248"" ""248"" ""248"" ""248""
```

Parlons à présent de la partie programmation de l'application mobile grâce à "MIT App Inventor".

Il y a différents éléments que nous allons vous présenter :

Au début du programme, les valeurs sont initialisées à 0 par les blocs suivants :

```

initialize global reception_temperature to 0
initialize global reception_humidité to 0
initialize global reception_lumière to 0
initialize global reception to " "

```

La fonction recevoir consiste à récupérer les données des différents capteurs grâce à une horloge (clock_recevoir dans notre exemple) qui se réactualise toutes les 15 millisecondes. Chaque valeur est reliée à un canal différent.

Si le robot est connecté le texte connecté apparaît en vert.

Le smartphone reçoit des informations du bluetooth. On supprime les > et on stocke dans la variable réception. Si la variable contient <0= on le stocke dans la variable lumière et affiche dans le label lumière. Si la variable contient <1= on le stocke dans la variable humidité et affiche dans le label humidité. Si la variable contient <2= on le stocke dans température et affiche dans le label température.

```

when clock_recevoir.Timer
do
  if BluetoothClient1.isConnected
  then
    set info_connect.TextColor to green
    set info_connect.Text to "Connecté"
    set global reception to replace all text call BluetoothClient1.ReceiveText
    segment ">"
    replacement ""
    if contains text get global reception
    piece "<0="
    then
      set global reception_lumière to replace all text get global reception
      segment "<0="
      replacement ""
      set lumière.Text to replace all text get global reception
      segment "<0="
      replacement ""
    if contains text get global reception
    piece "<1="
    then
      set global reception_humidité to replace all text get global reception
      segment "<1="
      replacement ""
      set humidité.Text to replace all text get global reception
      segment "<1="
      replacement ""
    if contains text get global reception
    piece "<2="

```


Cette partie du programme sert à afficher les différents arrangements de pilotage, enregistrement et valeurs.

L'application ne peut pas contenir autant d'arrangement par manque de place donc nous en mettons certains invisible et d'autres visible à certain moment.

```
when Button_pilotage .Click
do
  if Button_pilotage .Text = " valeur "
  then
    set Button_pilotage .Text to " pilotage "
    set arrangementgraphX .Visible to false
    set Arrangement_PILOTAGE_BOUTON .Visible to true
  else if Button_pilotage .Text = " pilotage "
  then
    set Button_pilotage .Text to " enregistrement "
    set Arrangement_PILOTAGE_BOUTON .Visible to false
    set liste .Visible to true
  else
    set Button_pilotage .Text to " valeur "
    set liste .Visible to false
    set arrangementgraphX .Visible to true
```


Quand la clock tracer marque le temps, l'axe se trace en fonction des valeurs reçu par la clock recevoir :

```
when clock_tracer.Timer
do
  set ShakingLabel.Text to "-"
  set clock_connecter_et_envoyer.TimerEnabled to false
  if get global xAxisIndex >= 320
  then
    call Canvas1.Clear
    set global xAxisIndex to 0
  if xAccelCheckBox.Checked
  then
    initialize local XAccel to get global reception_lumière
    in
      call DrawLineSegment2
      LineColor green
      CurrentAccel get XAccel
      PrevAccel get global PrevXAccel
      set global PrevXAccel to get XAccel
  if yAccelCheckBox.Checked
  then
    initialize local YAccel to get global reception_humidité
    in
      call DrawLineSegment
      LineColor red
      CurrentAccel get YAccel
      PrevAccel get global PrevYAccel
      set global PrevYAccel to get YAccel
  if zAccelCheckBox.Checked
  then
    initialize local ZAccel to get global reception_temperature
    in
      call DrawLineSegment
      LineColor blue
```

Et un segment se crée en fonction de l'axe et de la couleur reçu précédemment :

```
to DrawLineSegment LineColor CurrentAccel PrevAccel
do
  set Canvas1 . LineWidth to 3
  set Canvas1 . PaintColor to get LineColor
  call Canvas1 . DrawPoint
    x get global xAxisIndex
    y call AdjustY value get PrevAccel
  call Canvas1 . DrawLine
    x1 get global xAxisIndex
    y1 call AdjustY value get PrevAccel
    x2 get global xAxisIndex
    y2 call AdjustY value get CurrentAccel
```

Ces petites lignes permettent d'ajuster dans l'image les traits créés par les valeurs pour qu'il soit au bon niveau dans le tableau.

```
to AdjustY value
result round 300 + ? -3 × ? get value
```

```
to AdjustY2 value
result round 300 + ? -0.3 × ? get value
```

Tout ce qui est ici est nécessaire pour créer une liste en initialisant tout d'abord les valeurs de la liste puis créer un document texte qui enregistre toutes les valeurs quand on appuie sur le bouton enregistrer.

The image shows two Scratch code blocks. The first block is a 'when recevoir_liste .Timer' block. It contains three 'add items to list' blocks. Each 'add items to list' block has a 'list' input and an 'item' input. The 'item' inputs are 'global list_temp', 'global reception_temperature', and 'global list_humidité'. Above this block are three 'initialize global list' blocks: 'initialize global list to create empty list', 'initialize global list_temp to make a list Temperature', and 'initialize global list_humidité to make a list Humidite'. The second block is a 'when Button_enregistrer .Click' block. It contains a 'call File1 .AppendToFile' block with 'text' set to 'list to csv table list' and 'fileName' set to 'TextBox1 .Text'. To the right of this block are three 'list to csv row list' blocks, each connected to a 'get global list' block: 'get global list_humidité', 'get global list_lumiere', and 'get global list_temp'. Above this second block is an 'initialize global list_lumiere to make a list Lumiere' block.

Voici la partie du programme qui nous informe de l'état de la connexion bluetooth entre l'application et la carte educaduinov :

The image shows a Scratch code block: 'when clock_connecter_et_envoyer .Timer'. It contains an 'if' block. The 'if' block has a condition 'BluetoothClient1 . IsConnected'. If the condition is true, it contains two 'set' blocks: 'set info_connect . TextColor to green' and 'set info_connect . Text to Connecté'. If the condition is false, it contains two 'set' blocks: 'set info_connect . TextColor to red' and 'set info_connect . Text to Non connecté'.

Les différents bloc servent à envoyer chaque instruction au robot, il y en a 11 tel que : avancer, reculer, aller sur les cotés, en diagonale, se stopper.

```
when Bouton_11 .Click
do
  if BluetoothClient1 .IsConnected
  then
    call BluetoothClient1 .SendText
    text join "<0="
           11
           ">"
```

```
when Bouton6 .Click
do
  if BluetoothClient1 .IsConnected
  then
    call BluetoothClient1 .SendText
    text join "<0="
           6
           ">"
```

```
when Bouton7 .Click
do
  if BluetoothClient1 .IsConnected
  then
    call BluetoothClient1 .SendText
    text join "<0="
           7
           ">"
```

Ci-dessous les blocs permettent de piloter le robot en fonction de l'inclinaison du smartphone.

Quand le "bouton_start" est enfoncé il met la variable à 1.

A chaque clock, si le Bluetooth est connecté, si le bouton pilotage texte est égale à orientation et que global Xavance est égale à 1 alors, on va envoyer des valeurs pour piloter le robot en fonction de l'inclinaison du smart-phone.

Par exemple si l'inclinaison est vers l'avant alors $-3 < XAccel < 3$ et $YAccel > 3$ on envoie un ordre qui permet d'avancer le robot vers l'avant qui est la valeur 2. Ainsi de suite pour les différentes inclinaisons.

```
when bouton_start.TouchDown
do set global Xavance to 1
```

```
when bouton_start.TouchUp
do set global Xavance to 0
```

```
when Clock1.Timer
do if BluetoothClient1.IsConnected
then if Button_pilotage.Text == "orientation"
and get global Xavance == 1
then if AccelerometerSensor1.XAccel > -3
and AccelerometerSensor1.XAccel < 3
and AccelerometerSensor1.YAccel > 3
then call BluetoothClient1.SendText
text join "<0=" 2 ">"
set Label5.Text to "Avant"
if AccelerometerSensor1.XAccel < -3
and AccelerometerSensor1.YAccel > 3
then call BluetoothClient1.SendText
text join "<0=" 3 ">"
set Label5.Text to "Avant-Droit"
if AccelerometerSensor1.XAccel > 3
and AccelerometerSensor1.YAccel < -3
and AccelerometerSensor1.YAccel > -3
then call BluetoothClient1.SendText
text join "<0=" 4 ">"
set Label5.Text to "gauche"
if AccelerometerSensor1.XAccel > -3
and AccelerometerSensor1.XAccel < 3
```



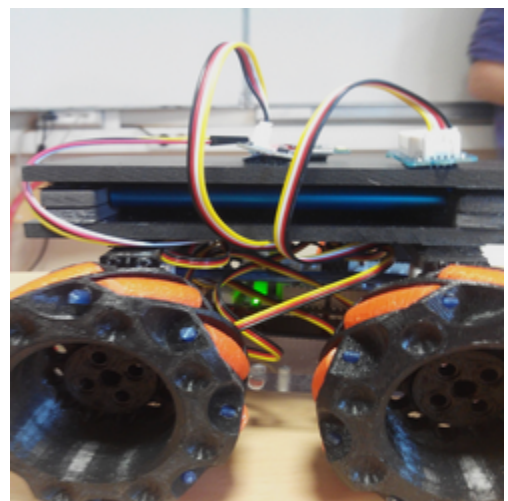
Maquette :



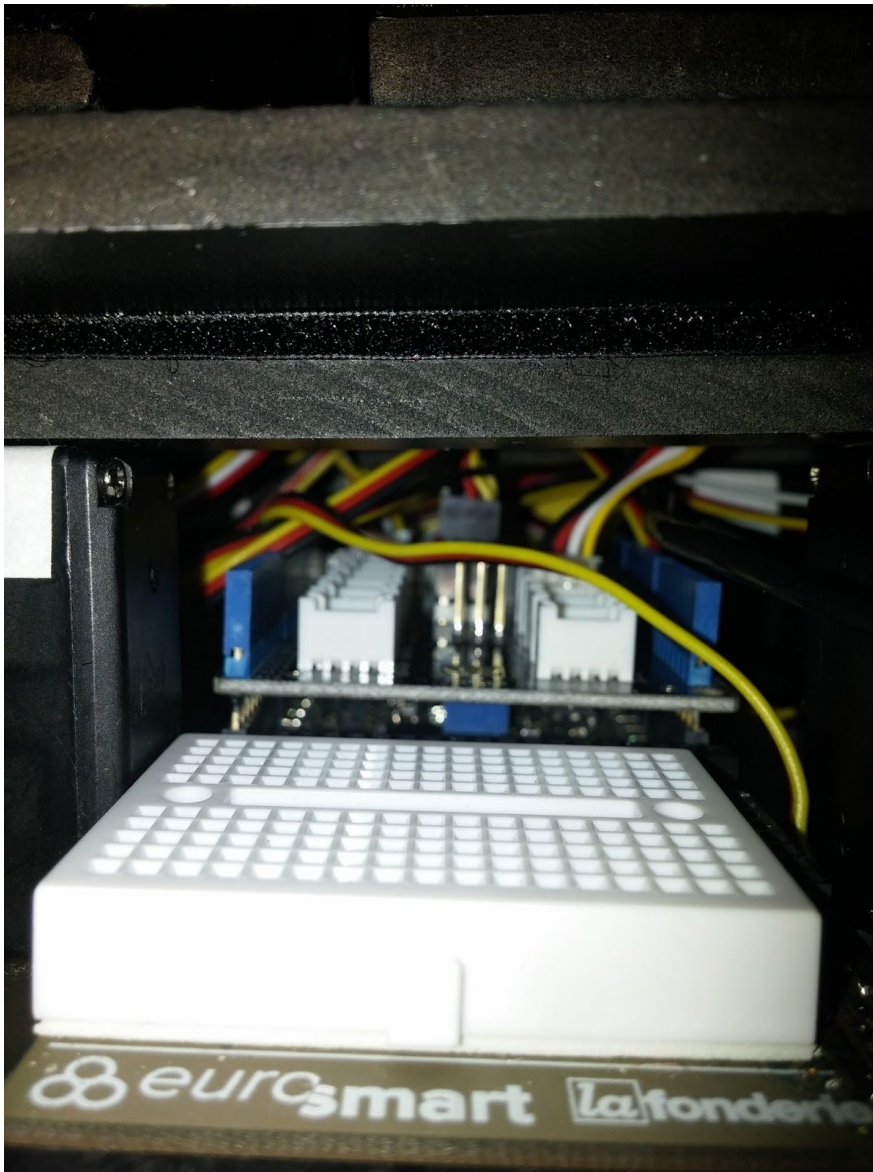
Vu du dessus :



Vu de côté :



Sur cette photo on peut remarquer la plaque d'essai ainsi que la carte Arduino avec son shield grove vu de derrière.

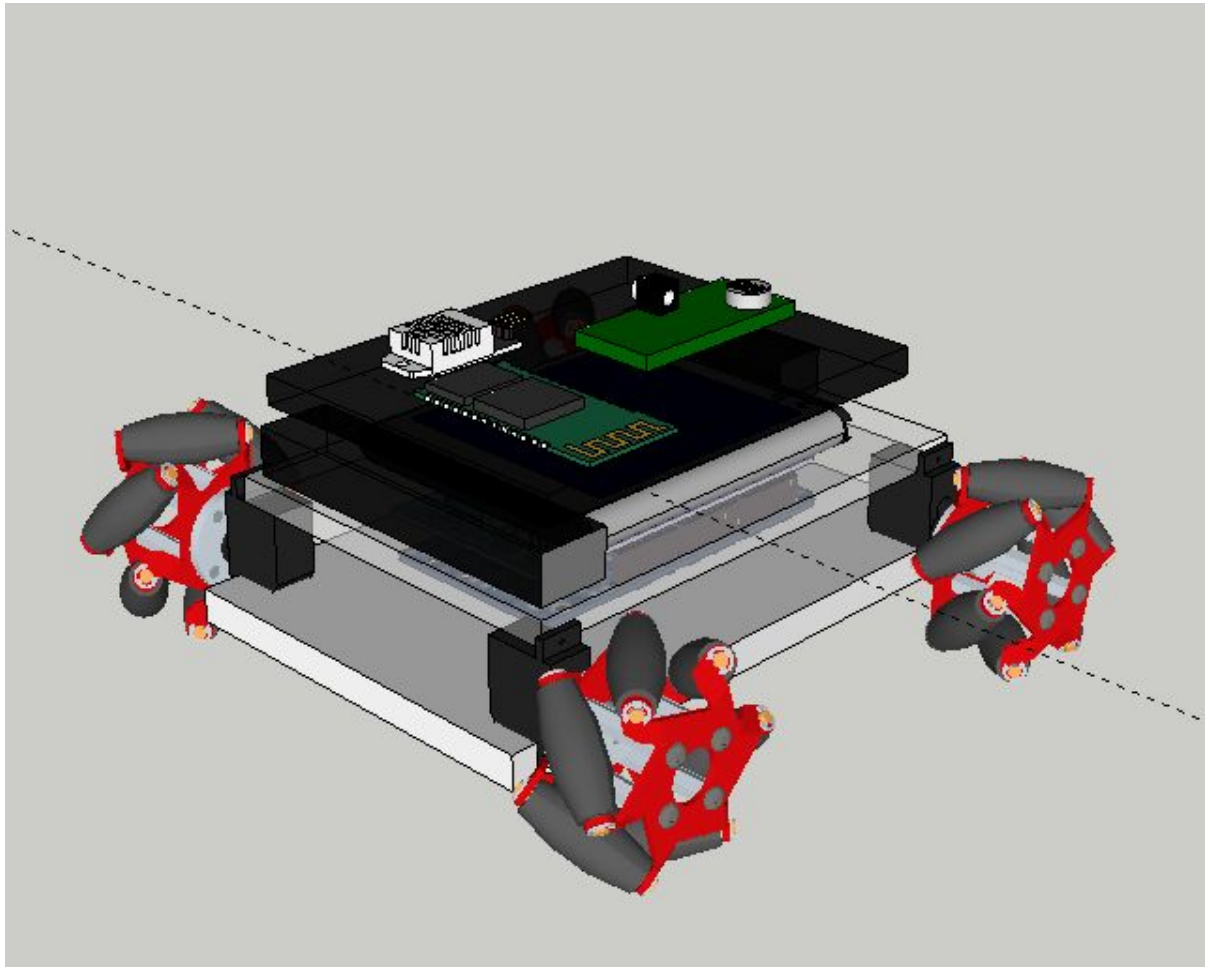




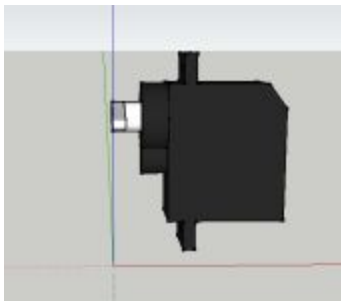
Sur cette photo on peut voir un afficheur LCD ainsi que les capteurs : d'humidité, de lumière et de température .

Modélisation du projet sous sketchup :

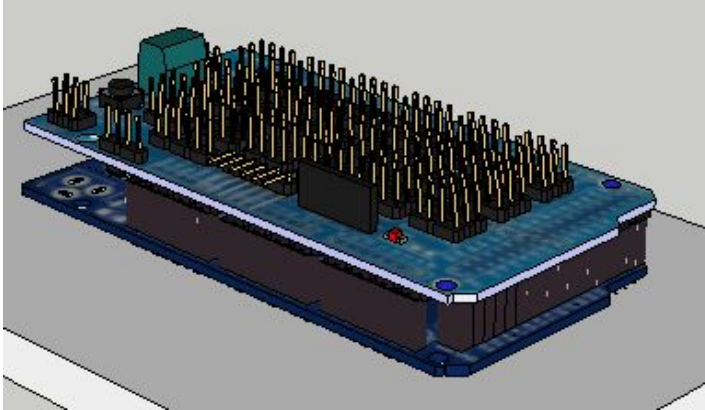
Voici le plan de base, le squelette du robot Mecanum. On voit le squelette du robot :



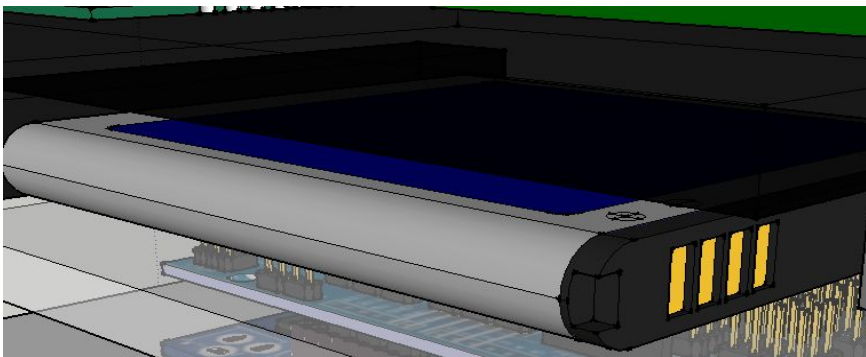
- Les servomoteurs:
Ils servent a faire tourner les roues.



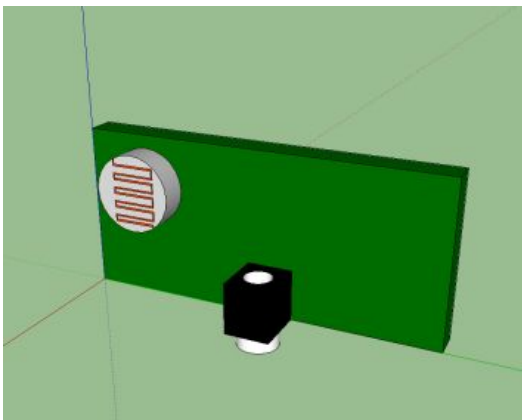
- La carte Educaduino plus avec son shield grove :
Elle sert à traiter toutes les informations venant des capteurs et du smartphone. Elle envoie au smartphone les valeurs des capteurs.



- La batterie :
Elle sert à alimenter constamment le robot en énergie.

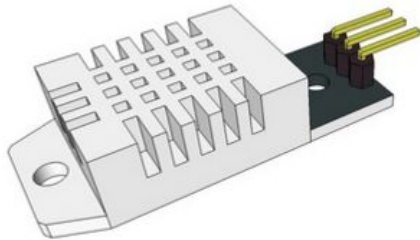


- Le capteur de lumière :
Il sert à capter la lumière. Il envoie les informations à la carte educaduino qui les transmet au bluetooth.



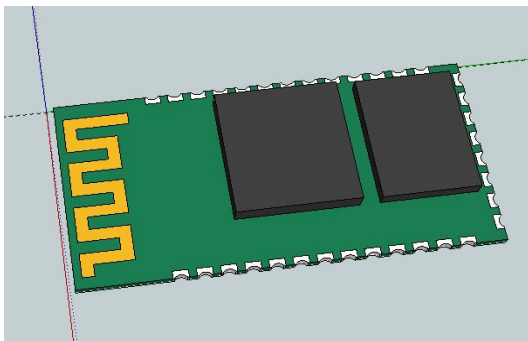
- Le capteur d'humidité et de température :

Il sert à capter l'humidité et la température . Il envoie les informations à la carte educaduino qui les transmet au bluetooth.

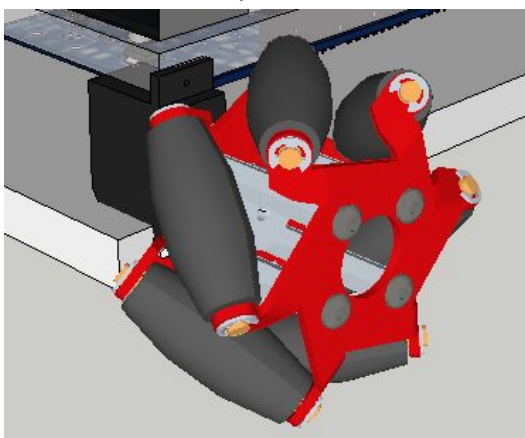


- Le Bluetooth :

Le bluetooth connecte le téléphone et la carte educaduino. Il reçoit et envoie des informations.



- Les roues : Elles servent à se déplacer de façon différente. Nous voulions quelque chose d'innovant. Grâce aux roues le robot peut se déplacer en crabe ce qui facilite certains déplacements difficiles.



Planification :

Semaine 8	Semaine 9	Semaine 10	Semaine 11	Semaine 12	Semaine 13	Semaine 14	Semaine 15	Semaine 16	Semaine 17
22/02/16	29/02/16	07/03/16	14/03/16	21/03/16	28/03/16	04/04/16	11/04/16	18/04/16	25/04/16
		Contrainte et contexte, ressource							
		{groupe 1 lucie, manon, akina}			[15/03/16 - 25/03/16]				
							Cahier des Charges		
	{Groupe 2 Leo, Nicolas et Zacharie}							[08/03/16 - 15/04/16]	
		{Groupe 3 Zoe Ameline Victoria Jeanne}					Chef de projet		
								[15/03/16 - 15/04/16]	
							Modelisation sur Sketchup		
		{Groupe 4 Hugo et Lena}						[15/03/16 - 15/04/16]	
							Modélisation du robot		
		{Groupe 5 Antoine}						[15/03/16 - 15/04/16]	
				2 diagrammes					
		{Groupe 6 Eluan Oscar Manon}			[15/03/16 - 25/03/16]				
			Resume du reglement et numerotation des cartes						
		{groupe 7 Emma Tessa Pauline}			[15/03/16 - 25/03/16]				
			Presentation et situation du projet dans son environnement						
		{Groupe 8 Matteo Michael Noam}			[15/03/16 - 29/03/16]				
							App inventor		
		{Groupe 9 Niels et Paul}						[15/03/16 - 15/04/16]	
							Explication du programme app inventor		
						{Groupe 10 Pablo Lubin}		[08/04/16 - 15/04/16]	

Améliorations possibles :

Le robot se déplace bien mais certains déplacements sont plus difficiles du fait de ses roues imprimées. Nous pourrions remplacer les roues par des roues industrielles.

Les données inscrites sur le diagramme sont espacées dans le temps. Nous pourrions réduire le délai entre les données pour un meilleur affichage sur le canvas.

Il n'y a pas d'affichage du temps pendant la réception des données ni pendant leur enregistrement. Nous pourrions régler ce problème en affichant l'heure de la réception des données.

Nous pourrions rajouter un capteur GPS ainsi nous connaîtrions la position exacte du robot quand il enregistre les données.

Conclusion :

Sans la carte Educaduino plus nous n'aurions pas pu faire notre projet.

Nous avons fabriqué un robot tout terrain qui prend des informations sur son environnement grâce aux capteurs de température, d'humidité et de lumière.

Le robot se déplace dans toutes les directions. Il arrive à enregistrer les données qui sont le taux d'humidité, la température en degrés Celsius et la quantité de lumière. On peut aussi voir en direct depuis le smartphone via un autre smartphone fixé sur le robot grâce à l'application Alfred.

Grâce à ce projet nous avons appris à programmer une application avec APP Inventor, à modéliser sous SketchUp un objet réel et également un objet 3D. Nous avons aussi pu tout programmer : une carte Educaduino et un smartphone. Nous avons planifié avec Gantt et construit une maquette de taille réelle. Nous avons fait très attention avec les matériaux et les gadgets qui nous entourent pour le robot car tout coûte très cher et il fallait faire très attention. Nous avons appris à travailler en projet en partageant les travaux avec Google Drive pour centraliser les informations pour faciliter le travail en équipe ce qui nous a permis de mener ce projet très instructif et intéressant à son terme.