

P2022 : GrovExAO

Dossier technique du projet - partie commune

Table des matières

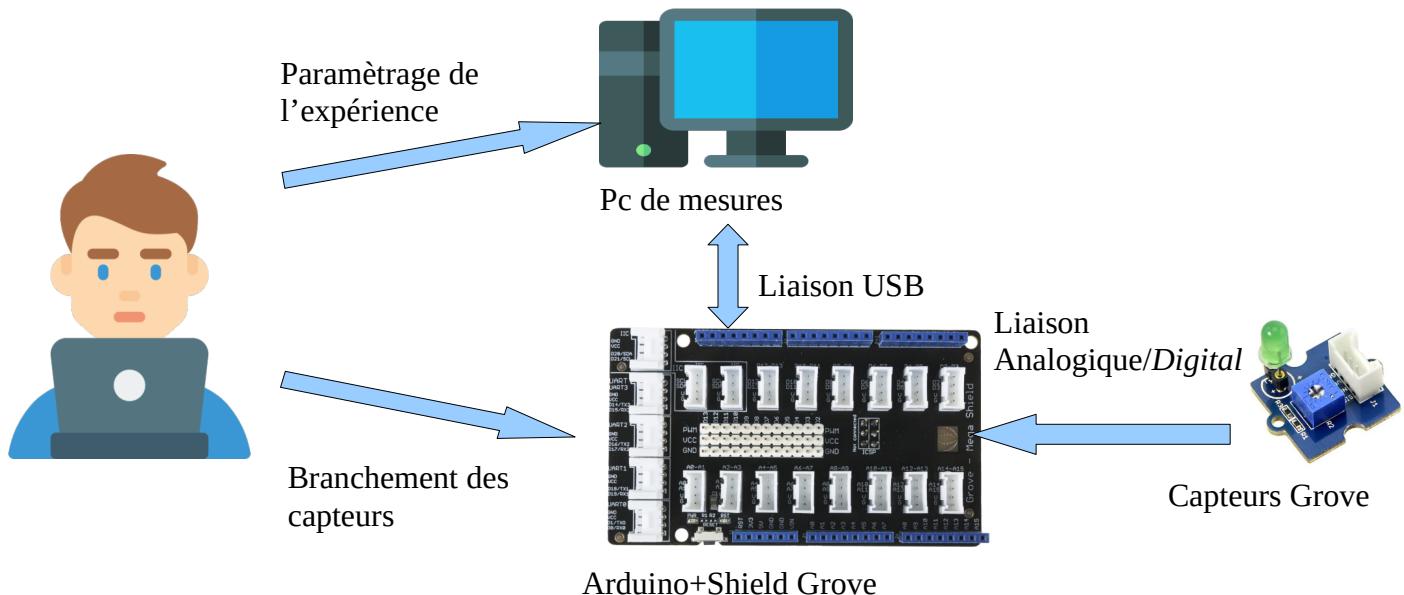
1 - INTRODUCTION.....	2
1.1) SITUATION DU PROJET DANS SON CONTEXTE INDUSTRIEL.....	2
1.1.1 - Synoptique général du système.....	2
1.1.2 - Fonctions du système.....	2
1.1.3 - Diagramme de déploiement d'exploitation.....	3
1.2) CONTRAINTES DIVERSES EXPRIMÉES PAR LE DEMANDEUR.....	3
1.2.1 - Contraintes du demandeur.....	3
1.2.2 - Contraintes matérielles.....	6
1.2.3 - Contraintes logicielles.....	6
2 - SPÉCIFICATIONS FONCTIONNELLES.....	6
2.1) CATALOGUE DES ACTEURS.....	6
2.2) DIAGRAMME DES CAS D'UTILISATION.....	6
2.3) DIAGRAMME DE SÉQUENCE « DÉMARRAGE DU LOGICIEL ».....	7
2.4) CAS D'UTILISATION « CHOISIR LES CAPTEURS À UTILISER POUR L'EXPÉRIENCE».....	7
2.4.1 - Description du cas d'utilisation.....	8
2.4.2 - IHM associée au cas d'utilisation : Choisir les capteurs à utiliser pour l'expérience :	8
2.4.3 - Diagramme de séquence du scénario nominal, alternatif A.....	9
2.5) CAS D'UTILISATION « CHARGER EXPÉRIENCE».....	9
2.5.1 - Description du cas d'utilisation.....	10
2.6) CAS D'UTILISATION « VISUALISER LES VALEURS».....	10
2.6.1 - Description du cas d'utilisation.....	11
2.6.2 - Fenêtre MDI.....	12
2.6.3 - IHM associée au cas d'utilisation :	12
2.6.4 - Diagramme de séquence des séquences.....	13
2.7) CAS D'UTILISATION « ACQUISITION DES VALEURS ».....	14
2.7.1 - Diagramme de séquence.....	15
2.7.2 - Description du cas d'utilisation.....	15
3 - ÉTUDE PRÉLIMINAIRE.....	17
3.1) LES LIBRAIRIES UTILISÉES.....	17
3.2) DIAGRAMME DE CLASSE D'APPLICATION.....	18
3.3) Outils utilisée.....	20
4 - RECETTE.....	23
5 - CONCLUSION.....	24

1 - Introduction

1.1) Situation du projet dans son contexte industriel

Notre système est un système de mesure de grandeur physiques facile d'accès se servant d'une carte Arduino et d'un shield Grove. Les mesures seront faites grâce à des capteurs puis recueillies par le PC de mesure afin qu'elles soient affichées. Le projet devra convenir à une fin éducative du collège jusqu'aux études supérieures que ce soit dans des matières technologiques ou physiques.

1.1.1 - Synoptique général du système



Le système se compose de différents éléments. Le premier élément est un PC de mesure où l'élève configura l'expérience, cela commence par le choix des capteurs puis des paramètres de l'acquisition. Ensuite il y a la carte Arduino connectée à un shield Grove qui sera relié à des capteurs qui mesureront les valeurs.

1.1.2 - Fonctions du système

La fonction principale du projet est donc de recueillir la valeur mesurée d'un capteur, de la traiter et finalement l'afficher. Cette demande de valeur se produit après avoir commencé une expérience. Une expérience pour avoir deux modes différents :

- manuel : L'expérience se lance automatiquement
- automatique : L'expérience se lance en différé, c'est-à-dire que l'expérience se lance à moment prédéfinis pas l'utilisateur

Il y enfin un choix qui est donné qui est de choisir si on veut voir l'historique des valeurs mesurées ou non, cet historique sera visible et on pourra également exporter ces données au format CSV (Comma Separated Value) afin qu'elles soient exploitable par un tableur.

Pour la praticité du projet, il y a la possibilité de sauvegarder la configuration physique d'une mesure (Type de capteur et port de branchement) ainsi que de charger cette configuration avant une expérience pour pouvoir facilement reproduire une expérience avec les mêmes paramètres.

Pour assurer la communication avec le PC de mesure, la carte Arduino utilise un interpréteur de commande « shell_mega28 » proposé par TechnoZone 51, entreprise créatrice du système précédemment utilisé par le collège client, pour pouvoir dialoguer avec la carte Arduino avec des commandes précises. Grâce à cet interpréteur on peut facilement obtenir la valeur renvoyée par un capteur, que ce soit un port analogique et numérique. On peut également modifier le nombre de bit sur lequel la valeur va être renvoyée, soit 8(256 valeurs) ou 10(1024 valeurs) bits.

Par exemple :

Pour lire la valeur d'un capteur analogique qui est branché sur le 00 la commande sera :

- sur 8 bits : **a00**. La valeur de retour sera « **Value = 112** »
- sur 10 bits : **A00**. La valeur de retour sera « **Value = 446** »

Pour lire une valeur numérique qui est branche sur le port 04 la commande sera :

- **R04**. La valeur retournée sera « **Value = 1** »

1.1.3 - Diagramme de déploiement d'exploitation

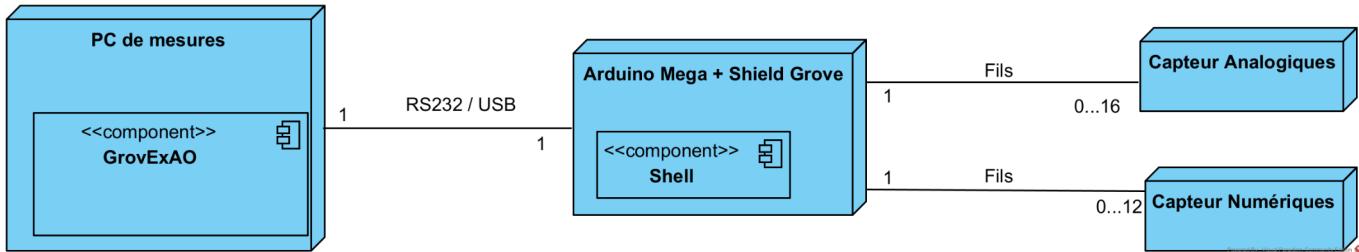


Figure 1: Diagramme de déploiement d'exploitation

L'ordinateur qu'utilise l'élève est relié par un câble USB avec la carte Arduino MEGA en utilisant un bus de données RS232. Ce format de données correspond à la liaison série entre le PC de mesures et la carte Arduino, c'est une méthode de communication qui permet de transmettre un seul bit de donnée à la fois.

L'Arduino quant à lui possède des broches pour se connecter aux capteurs utilisant des broches analogiques et numériques pour les capteurs adaptés. Un capteur analogique est un capteur qui peut délivrer en théorie, sur une plage de tension prédéfini, une infinité de valeurs, c'est de cette façon que va marche par exemple les potentiomètres ou les capteurs de température. Tandis qu'un capteur numérique ne renvoie qu'un 1 ou un 0, c'est par exemple le cas du détecteur de couleur.

1.2) Contraintes diverses exprimées par le demandeur

1.2.1 - Contraintes du demandeur

- Le système devra fonctionner peu importe le système d'exploitation et n'inclure aucune librairie propre à un système d'exploitation.
- Interface graphique adaptée à un collégien et suffisamment complète pour des étudiants supérieurs.
- Langue de toutes les interfaces en français et en anglais
- Le système d'exploitation de développement sera Windows car c'est le système d'exploitation d'utilisation majoritairement utilisé dans les collèges
- Application portable avec aucune installation de logiciel à faire.
- Librairie libre de droit
- Branchement en USB

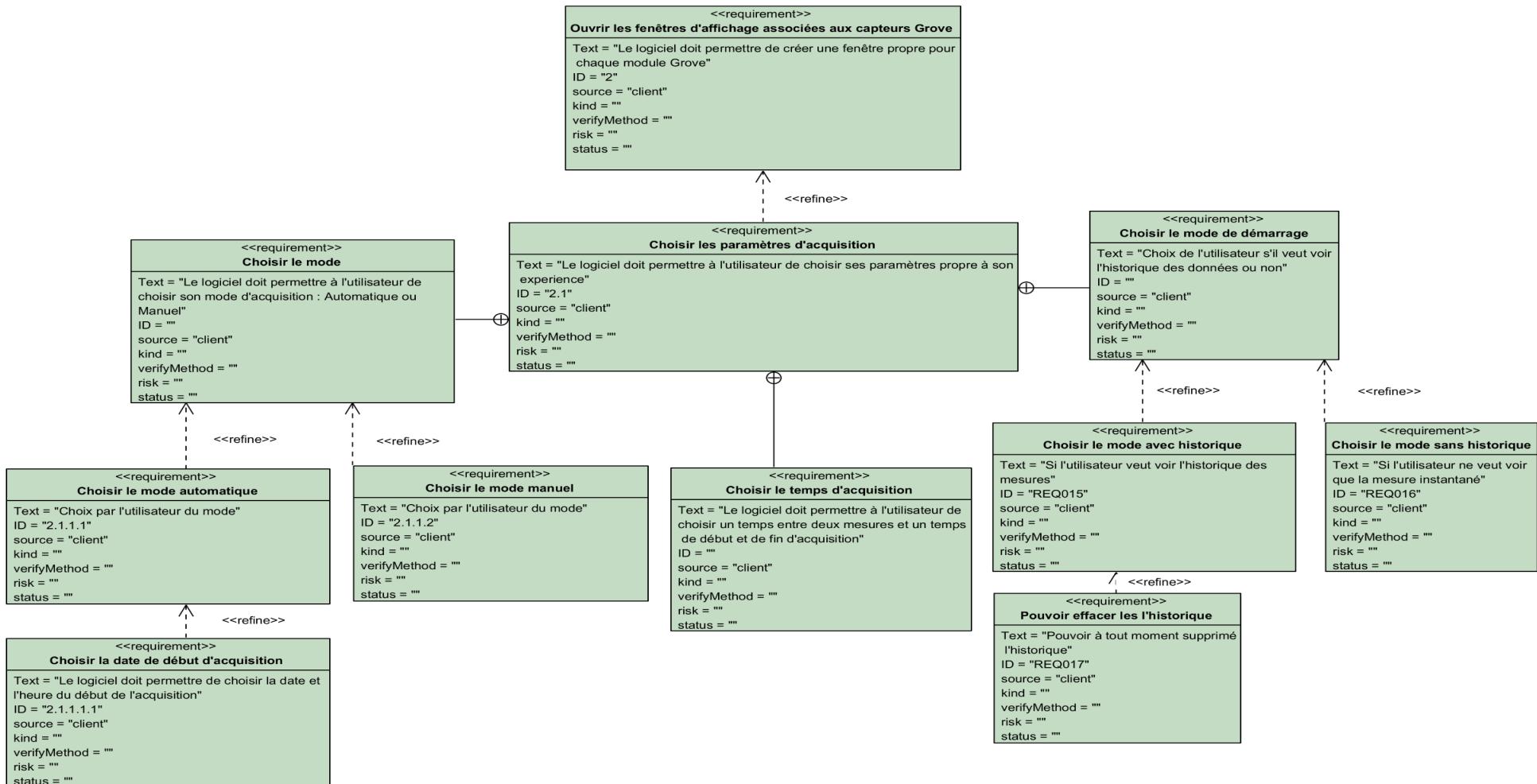


Figure 2: Diagramme d'exigence : Acquisition et visualisation des valeurs

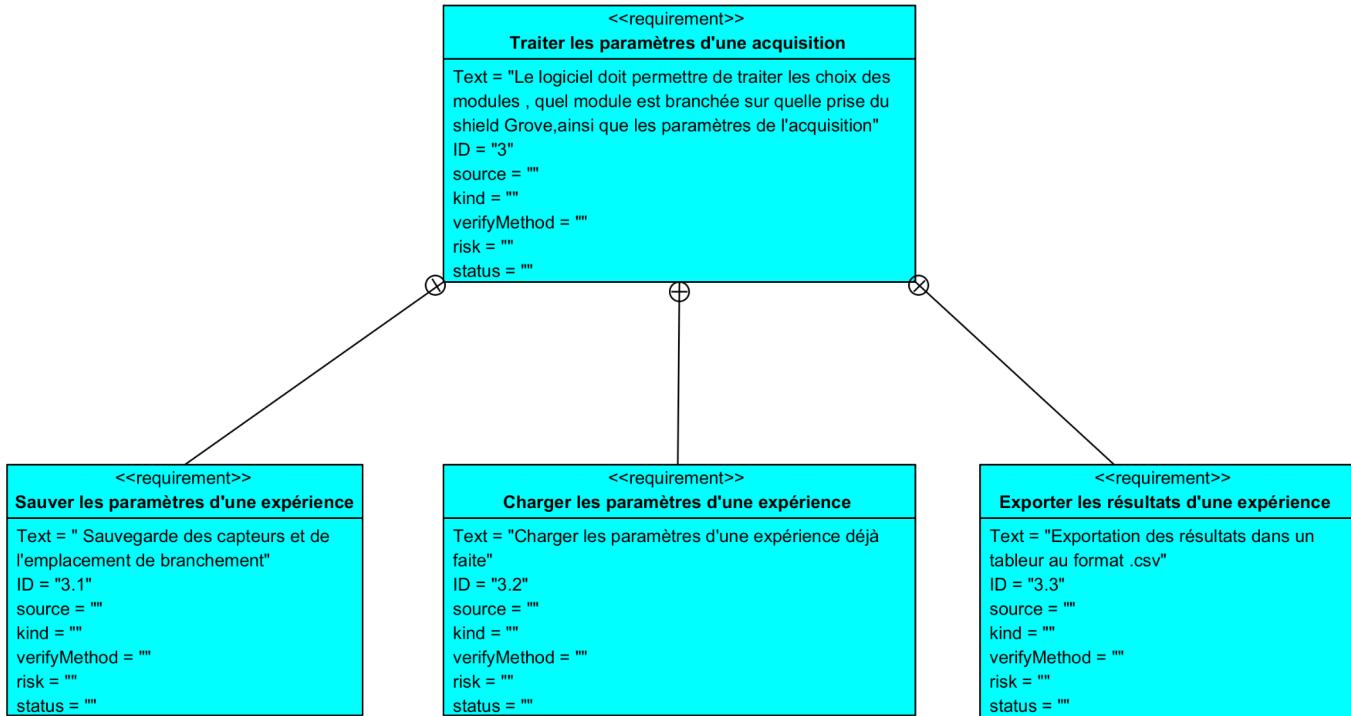


Figure 3: Diagramme d'exigence : Traitement des paramètres d'une acquisition

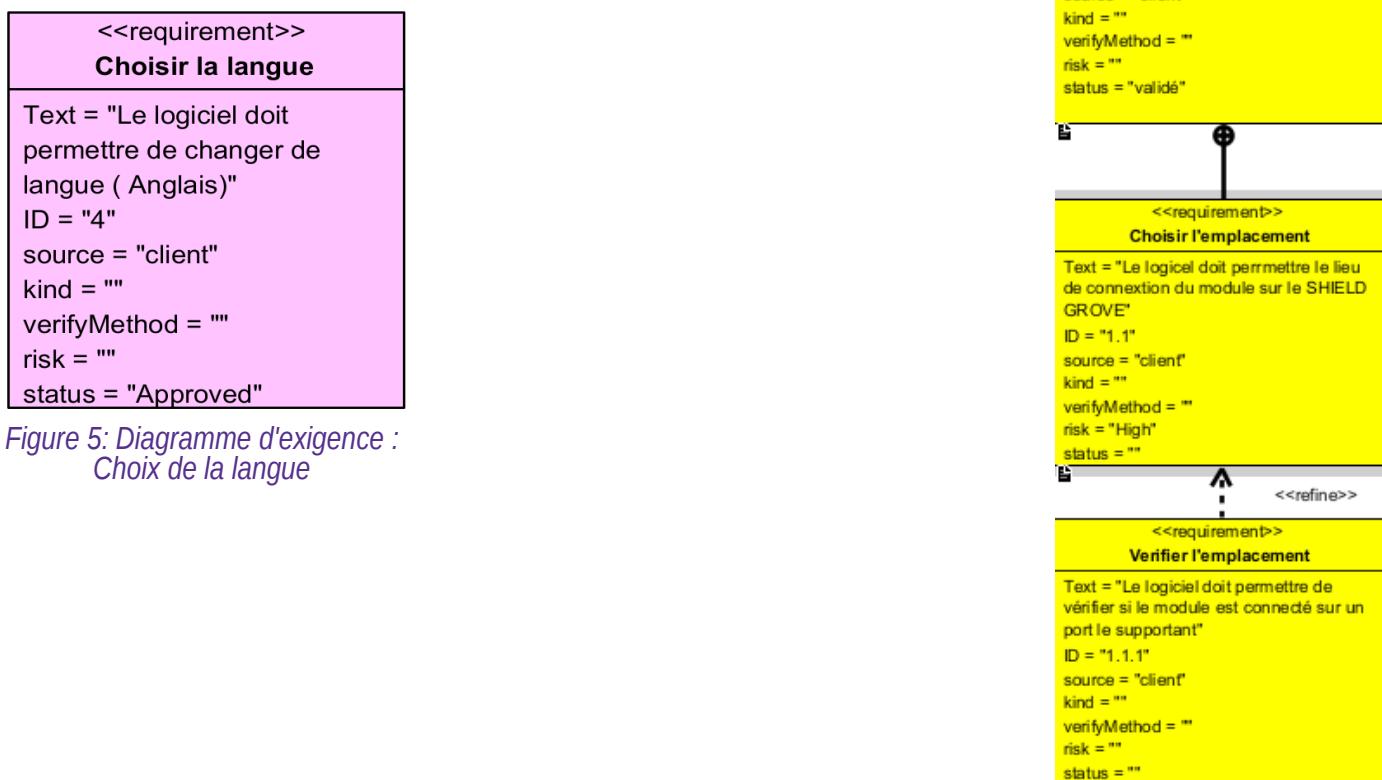


Figure 5: Diagramme d'exigence : Choix de la langue

Figure 4: Diagramme d'exigence : Choix des capteurs

1.2.2 - Contraintes matérielles

- Carte Arduino Mega
- Shield Grove
- Tous les capteurs nécessaires compatibles Grove
- Ordinateur avec un port USB

1.2.3 - Contraintes logicielles

- LibreOffice, car c'est une multi-plateforme et libre, pour la conception des documentations et des revues.
- Utilisation de Visual Paradigm pour la création des diagrammes UML
- Trello pour l'organisation du travail
- Gitlab afin de mettre en commun le travail effectué
- Fichier .ini pour connaître les paramètres de chaque capteur
- Qt 6.2 Long Term Support Release, pour être sur le logiciel soit mise à jour
- Doxygen pour la documentation du code source

2 - Spécifications fonctionnelles

2.1) Catalogue des acteurs

Acteur	Rôle
Élève	Utilisateur devant faire une ou plusieurs acquisitions de grandeurs physiques.
Professeur	Maîtrise du système et l'ajout de nouveaux capteurs

2.2) Diagramme des cas d'utilisation

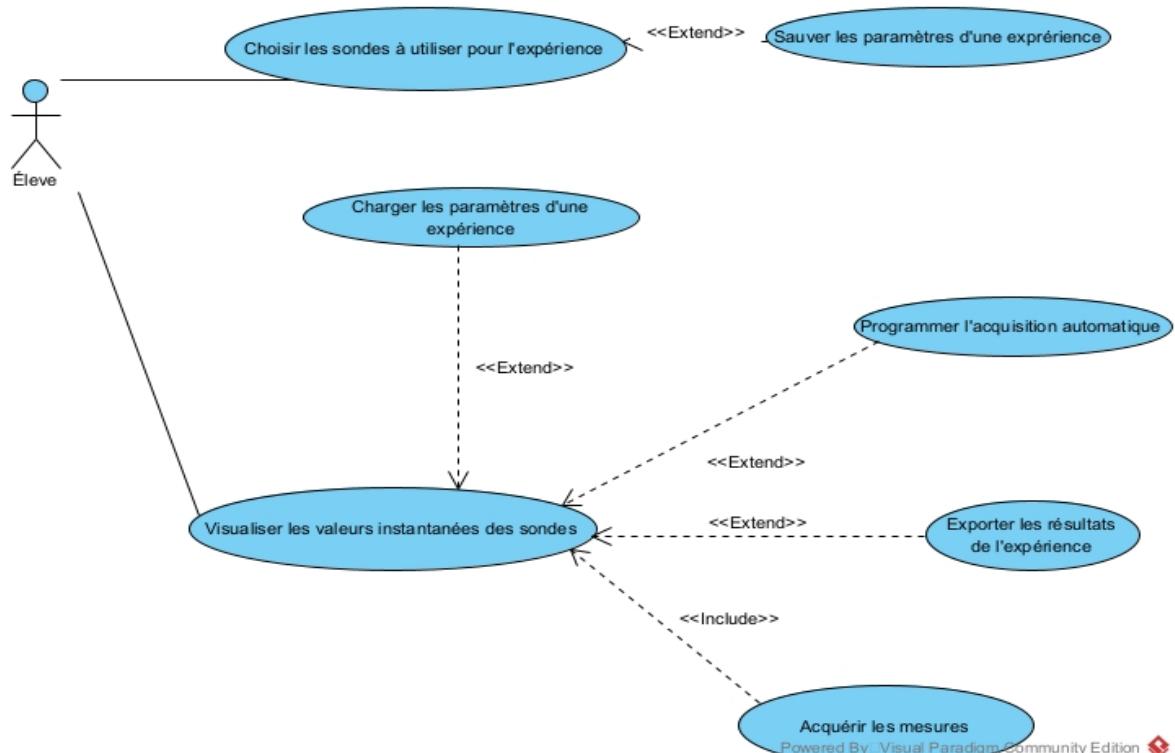


Figure 6: Diagramme des cas d'utilisations des systèmes

2.3) Diagramme de séquence « démarrage du logiciel »

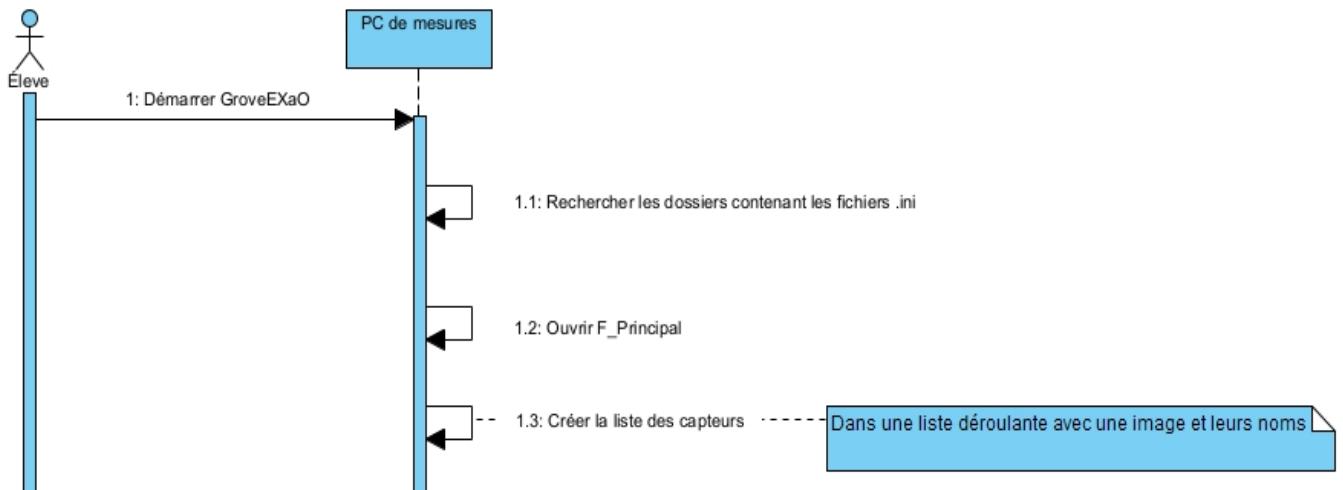


Figure 7: Diagramme de séquence : Démarrage du logiciel

Quand l'utilisateur lance le logiciel, celui-ci doit rechercher les dossiers contenant les fichiers de configuration INI à des fins de création d'une liste contenant le nom des capteurs, leurs configurations (fichier INI associé) et leur images associées. Ensuite GroveEXaO charge la liste des capteurs. Finalement après avoir créé la fenêtre principale où le choix des capteurs se fait, le logiciel affiche cette liste avec les images associées aux capteurs.

2.4) Cas d'utilisation « Choisir les capteurs à utiliser pour l'expérience»

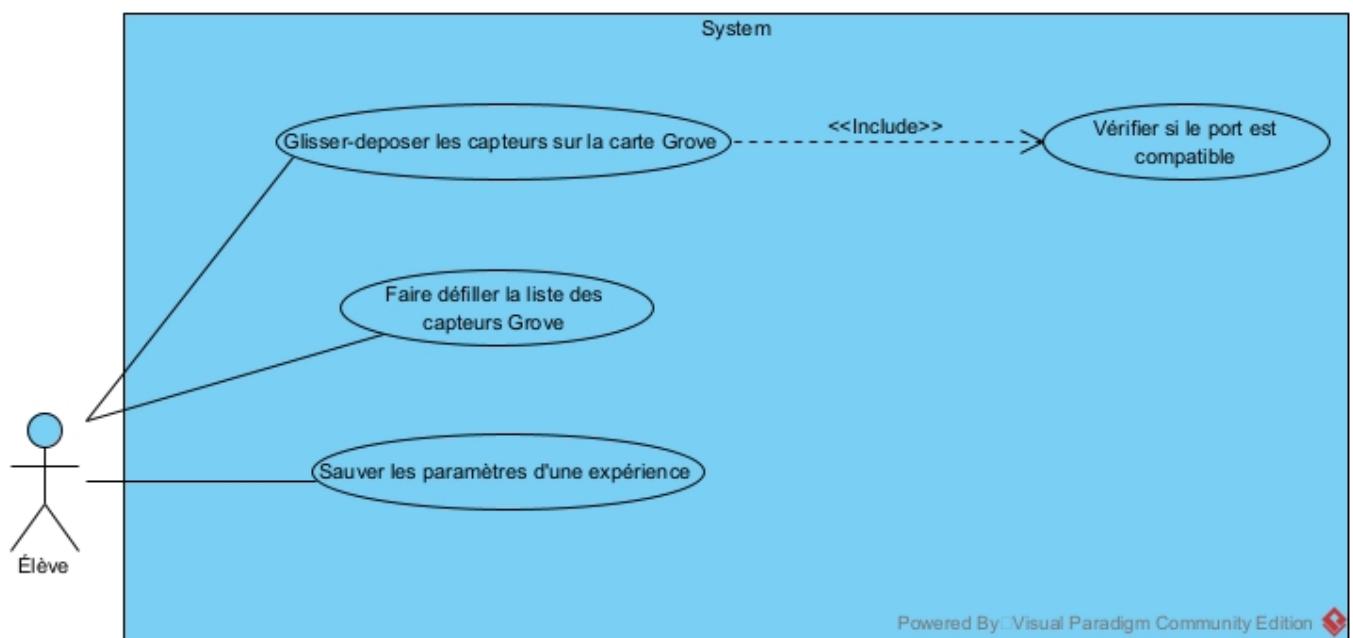


Figure 8: Choisir des capteurs

2.4.1 - Description du cas d'utilisation

Nom CU: Choisir des capteurs	Référence : CU1	Nom : Kanaïe Atrian
Pré-conditions	1. Le logiciel doit être lancé et opérationnel. 2. Carte Arduino avec les capteurs correctement branchés. 3. Avoir une expérience en mémoire déjà sauvegardée.	
Scénario nominal	1. Glisser et déposer les capteurs sur les bons ports du shield Grove. 2. Défiler la liste des capteurs 3. Valider le plan de câblage.	
Scénario alternatif A	Condition: A1. Glisser et déposer les capteurs sur les bons ports du shield Grove. A2. Capteurs non compatibles avec le port sélectionné A3. Affichage d'un message d'erreur	
Scénario alternatif B	Condition B1. Glisser et déposer les capteurs sur les bons ports du shield Grove. B2. L'utilisateur annule le plan de câblage. B3. Les capteurs disparaissent du schéma	

2.4.2 - IHM associée au cas d'utilisation : Choisir les capteurs à utiliser pour l'expérience :

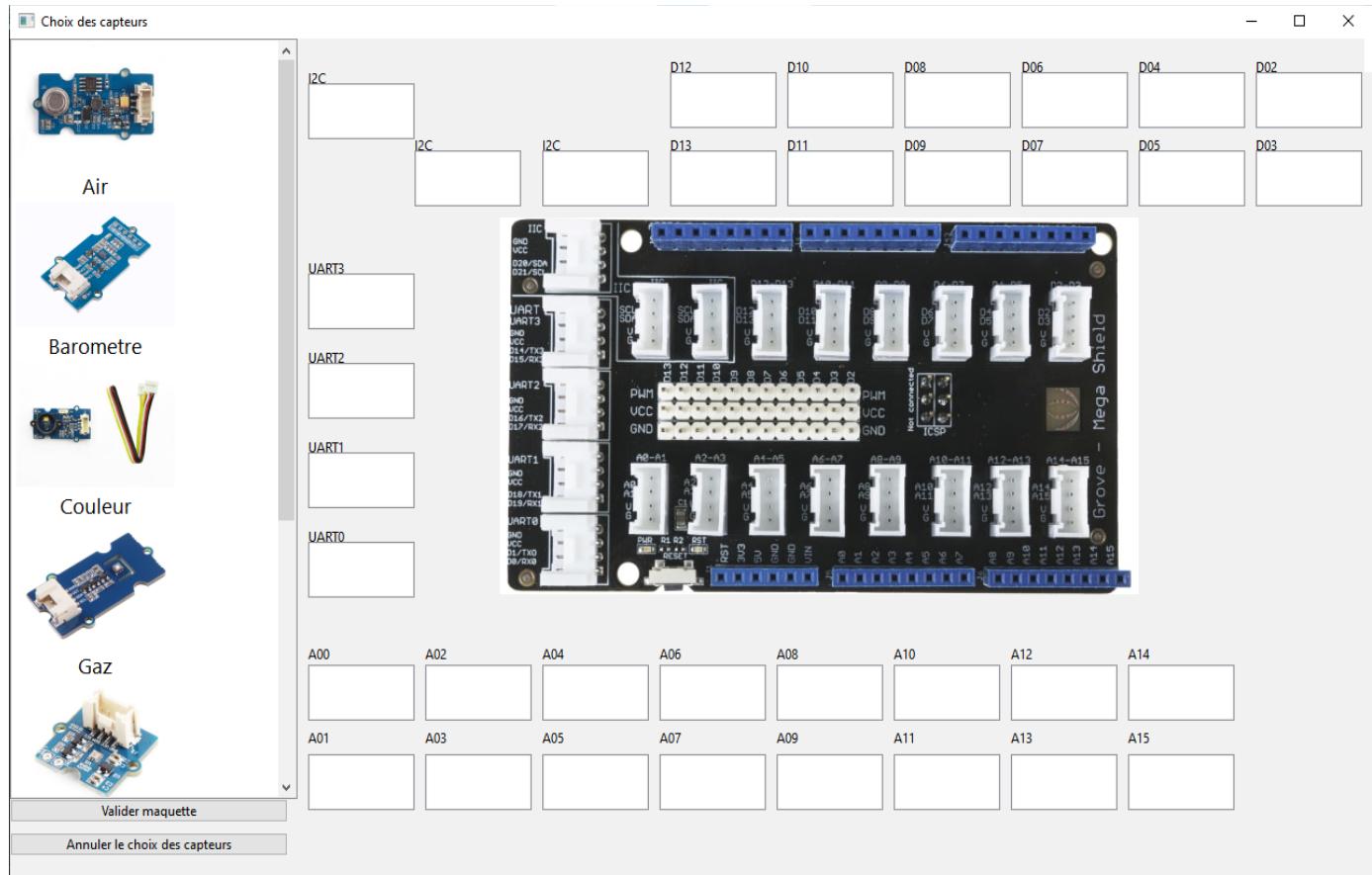
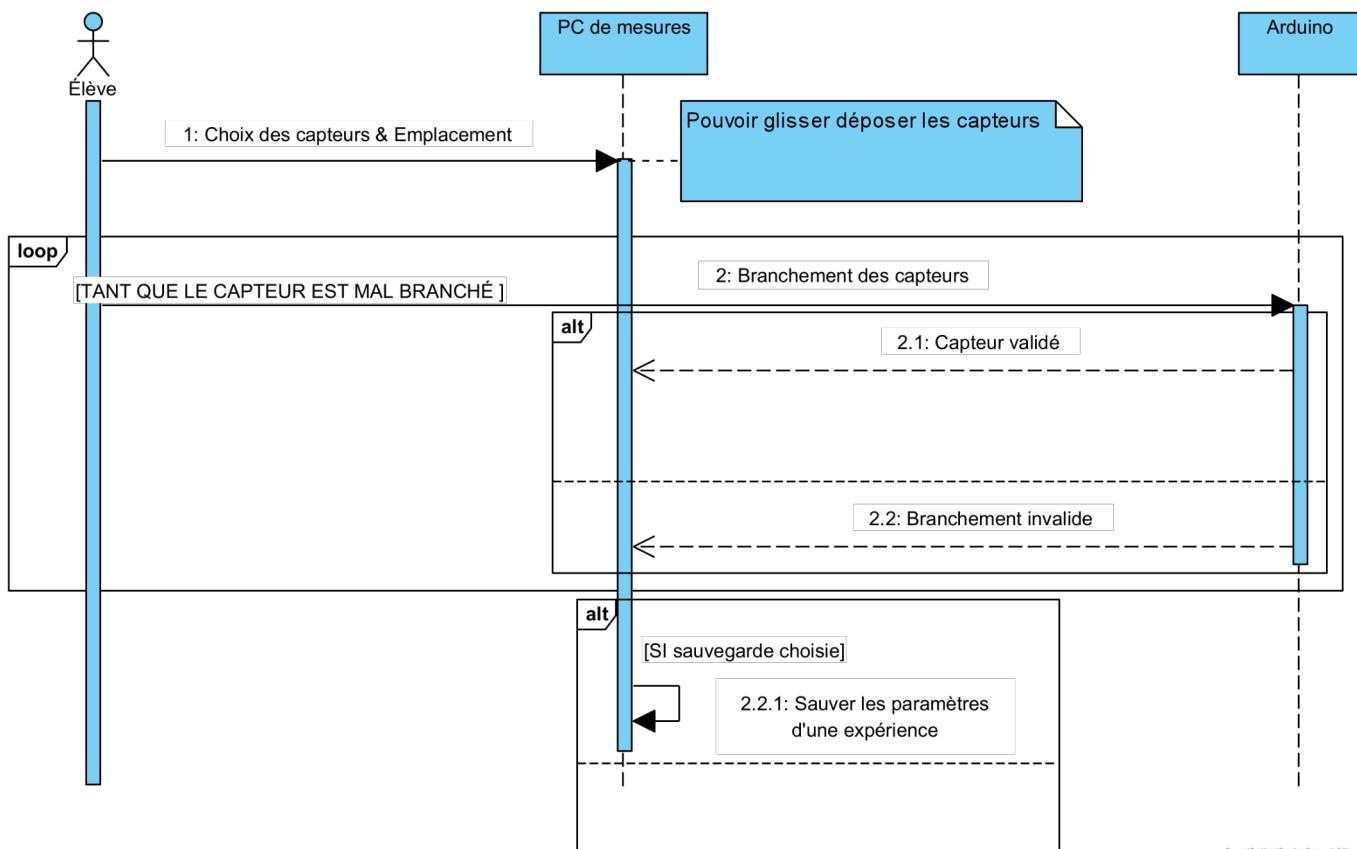
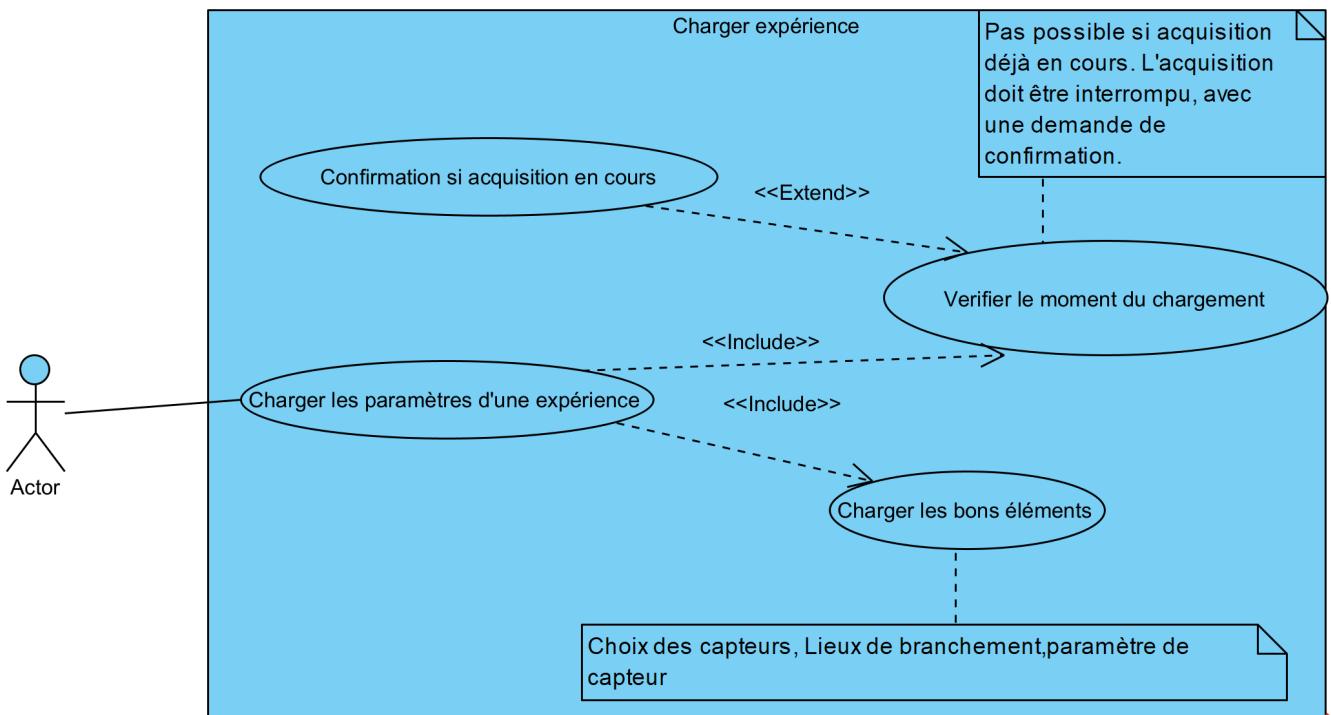


Figure 9: Maquette de l'interface du choix des capteurs

2.4.3 - Diagramme de séquence du scénario nominal, alternatif A



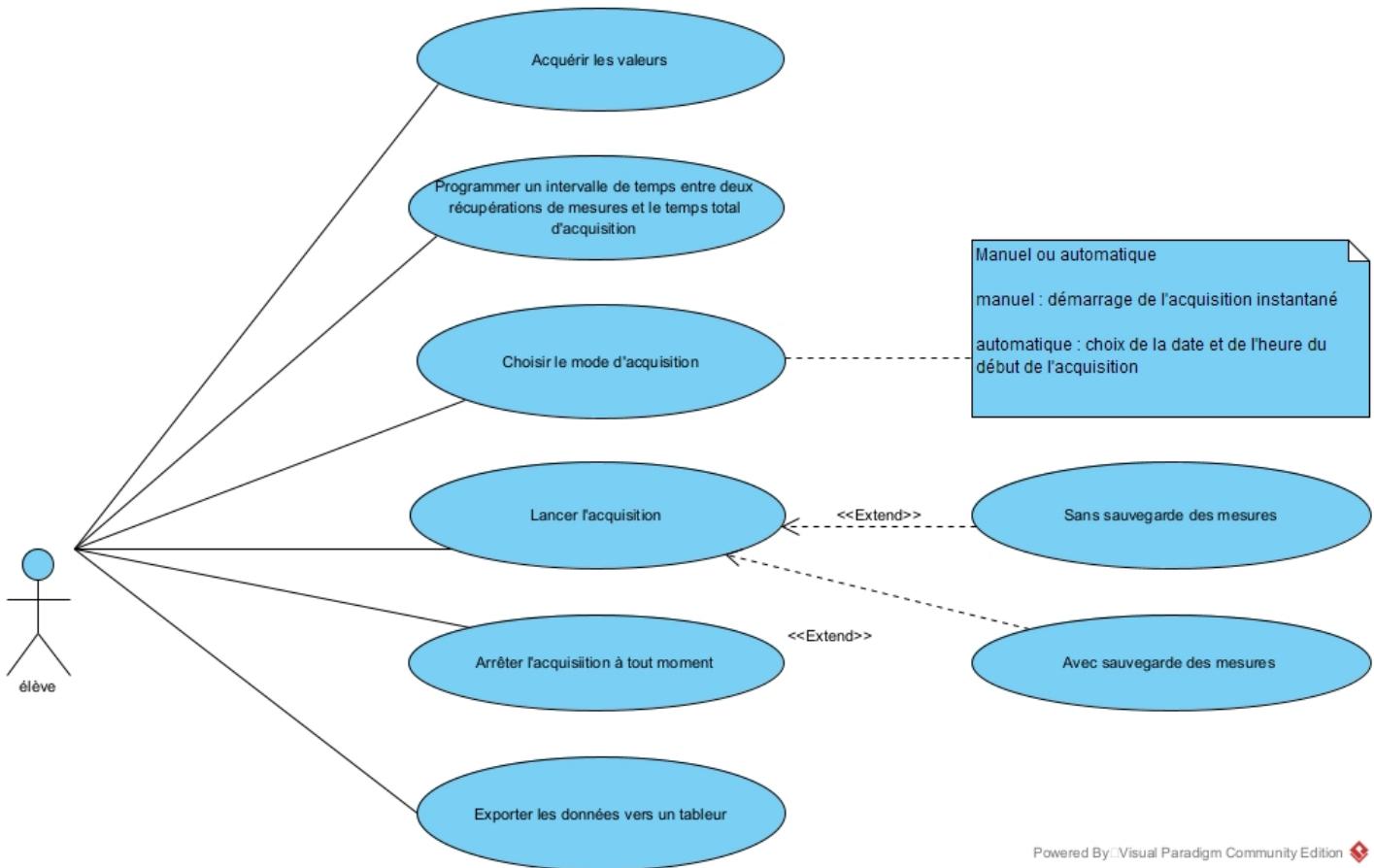
2.5) Cas d'utilisation « Charger expérience »



2.5.1 - Description du cas d'utilisation

Nom CU: Charger expérience	Référence : CU2	Nom : Kanaïe Atrian
Pré-conditions	1. Le système doit être lancé et opérationnel. 2. Carte arduino avec les capteurs correctement branchés. 3. Avoir une expérience en mémoire.	
Scénario nominal	1. L'utilisateur doit pouvoir charger une ancienne expérience avec les paramètres associés. Si une acquisition est en cours une demande de confirmation sera faite.	

2.6) Cas d'utilisation «Visualiser les valeurs»



Powered By: Visual Paradigm Community Edition

2.6.1 - Description du cas d'utilisation

Nom CU: Visualiser les valeurs	Référence : CU3	Nom : CHARRON & CARIOU
Pré-conditions	1. Brancher la carte Arduino MEGA au PC de mesures avec le câble USB 2. Sélectionner les capteurs et leurs emplacements avec le drag & drop ou par chargement de paramètres de l'expérience	
Scénario nominal : (en mode manuel avec sauvegarde des données)	1. L'utilisateur prépare les paramètres de l'expérience (mode de mesure, temps total, temps entre deux acquisitions, sauvegarde des données) 2. L'utilisateur lance l'acquisition en mode manuel 3. Le logiciel demande les valeurs mesurées par les capteurs de la carte Arduino MEGA 4. Le logiciel affiche les valeurs dans la zone 1 (voir IHM page 12)	
Scénario alternatif A (en mode manuel avec pas d'historique)	Condition : Choix du mode manuel avec pas d'historique A1. L'utilisateur prépare les paramètres de l'expérience (mode de mesure, temps total, temps entre deux acquisitions, sauvegarde des données) A2. L'utilisateur lance l'acquisition en mode manuel A3. Le logiciel demande les valeurs mesurées par les capteurs de la carte Arduino MEGA A4. Le logiciel affiche les valeurs dans la zone 1 et dans le tableau zone 2 (voir IHM) A5 : Au choix : Exportation possible des valeurs vers un tableau en format .csv	
Scénario alternatif B (en mode automatique avec pas d'historique)	Condition : Choix du mode automatique avec pas d' historique B1. L'utilisateur prépare les paramètres de l'expérience (mode de mesure, temps total, temps entre deux acquisitions, sauvegarde des données) B2. L'acquisition se lance à la date et l'heure précisée par l'utilisateur B3. Le logiciel demande les valeurs mesurées par les capteurs de la carte Arduino MEGA B4. Le logiciel affiche les valeurs dans la zone 1 (voir IHM)	
Scénario alternatif C (en mode automatique avec historique)	Condition : Choix du mode automatique avec historique C1. L'utilisateur prépare les paramètres de l'expérience (mode de mesure, temps total, temps entre deux acquisitions, sauvegarde des données) C2. L'acquisition se lance à la date et l'heure précisée par l'utilisateur C3. Le logiciel demande les valeurs mesurées par les capteurs de la carte Arduino MEGA C4. Le logiciel affiche les valeurs dans la zone 1 et dans le tableau zone 2 (voir IHM) C5. Au choix : Exportation possible des valeurs vers un tableau en format .csv	
Scénario alternatif D (Si arrêt en cours d'acquisition)	Condition : Choix d'arrêter l'acquisition en cours D1. L'utilisateur prépare les paramètres de l'expérience (mode de mesure, temps total, temps entre deux acquisitions, sauvegarde des données) D2. L'acquisition se lance D3. Le logiciel demande les valeurs mesurées par les capteurs de la carte Arduino MEGA D4. L'utilisateur arrête de force l'acquisition D5. Tous les traitements de valeurs restent possibles	
Post-conditions	1. L'acquisition s'arrête au bout du temps décidé. 2. La carte Arduino MEGA arrête de mesurer les données physiques 3. Le PC de mesures attend le traitement des mesures ou de la prochaine acquisition.	

2.6.2 - Fenêtre MDI

Une fenêtre de visualisation des mesures sera ouverte par capteur utilisé. Ces fenêtres s'afficheront dans une zone MDI présente dans la fenêtre principale. Une zone MDI est une zone dans laquelle on peut ouvrir plusieurs autres fenêtres.

Dans cet exemple, la fenêtre est « MainWindow » dans laquelle est située la zone MDI. Ici est affiché 6 fois la fenêtre d'affichage des données.

2.6.3 - IHM associée au cas d'utilisation :

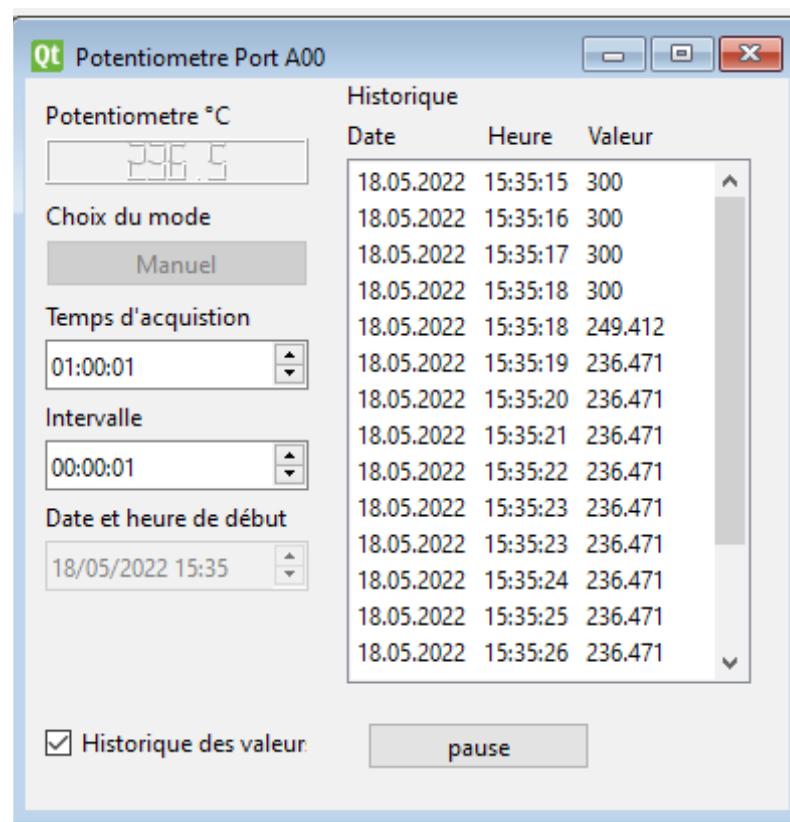
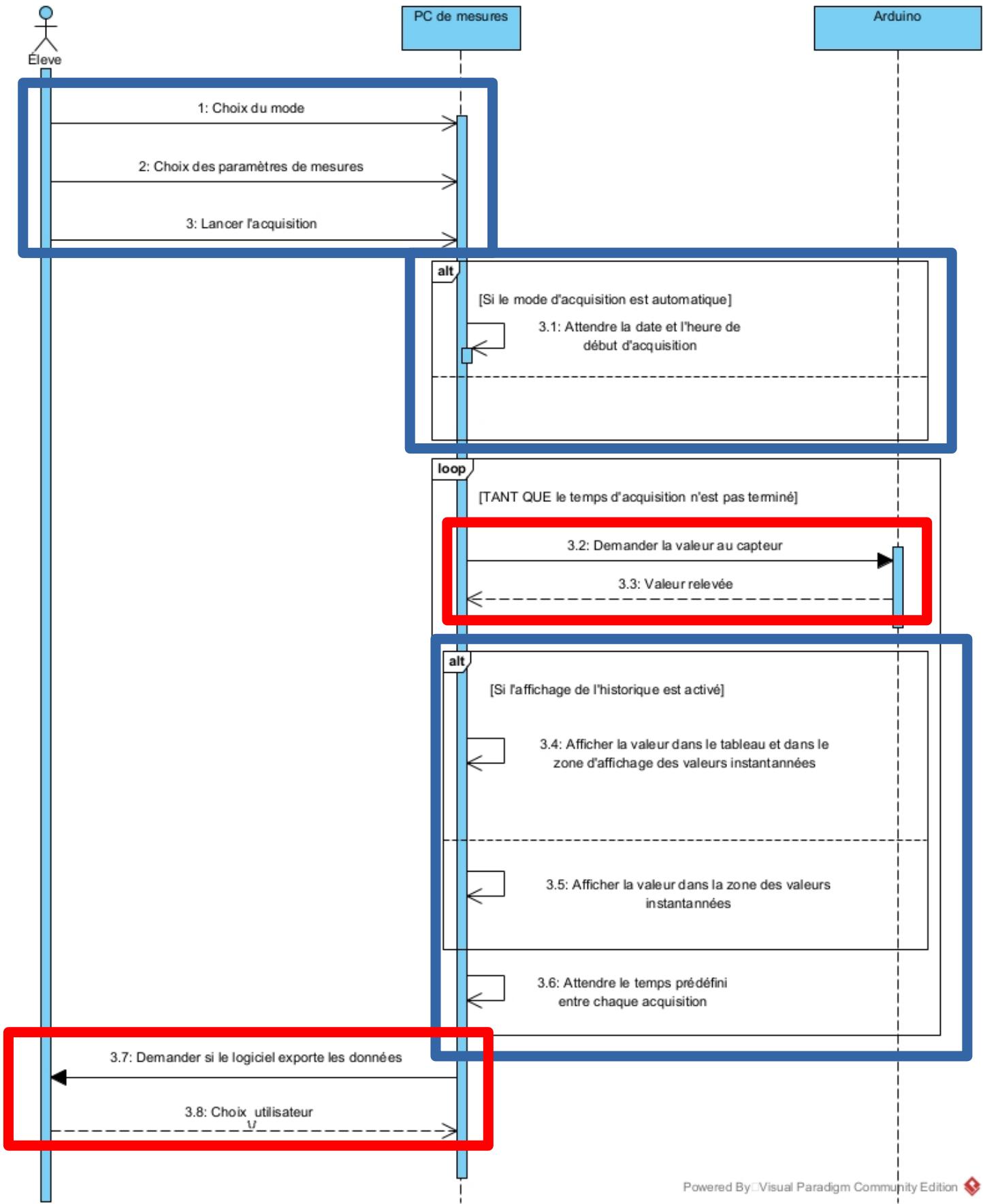


Figure 10: Interface de la visualisation des valeurs

2.6.4 - Diagramme de séquence des séquences

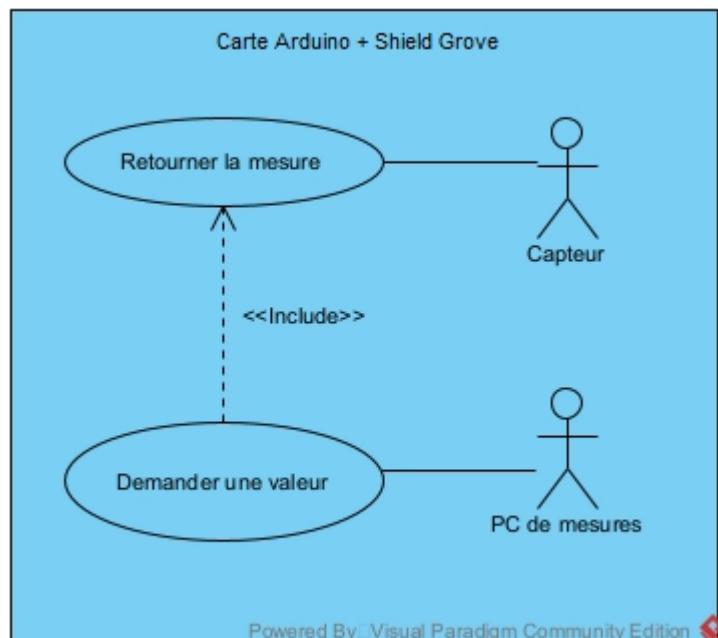


Cette partie du développement est partagée entre deux personnes. Celle de Dylan CHARRON (zone bleue) et celle de Vasco CARIOU (zone rouge).

Le premier cadre correspond au choix des différents paramètres d'acquisitions. Dans un premier temps, l'utilisateur choisira le mode d'acquisition (soit automatique ou manuel). Ensuite, il choisira le temps d'acquisition total, l'intervalle de temps entre 2 mesures et si le mode automatique a été choisi il définira la date et l'heure du début de l'acquisition. Pour terminer le paramétrage, l'utilisateur choisira s'il veut sauvegarder les mesures ou non, et enfin, il pourra lancer l'acquisition. Si l'utilisateur a choisi le mode automatique, le programme attendra la date et l'heure définie précédemment pour lancer l'acquisition. Une fois l'acquisition lancée, le programme affichera la valeur mesurée dans la zone des valeurs instantanées puis il attendra l'intervalle entre 2 mesures. Si l'utilisateur a choisi d'afficher l'historique des mesures, les valeurs s'afficheront dans le tableau d'historique des valeurs.

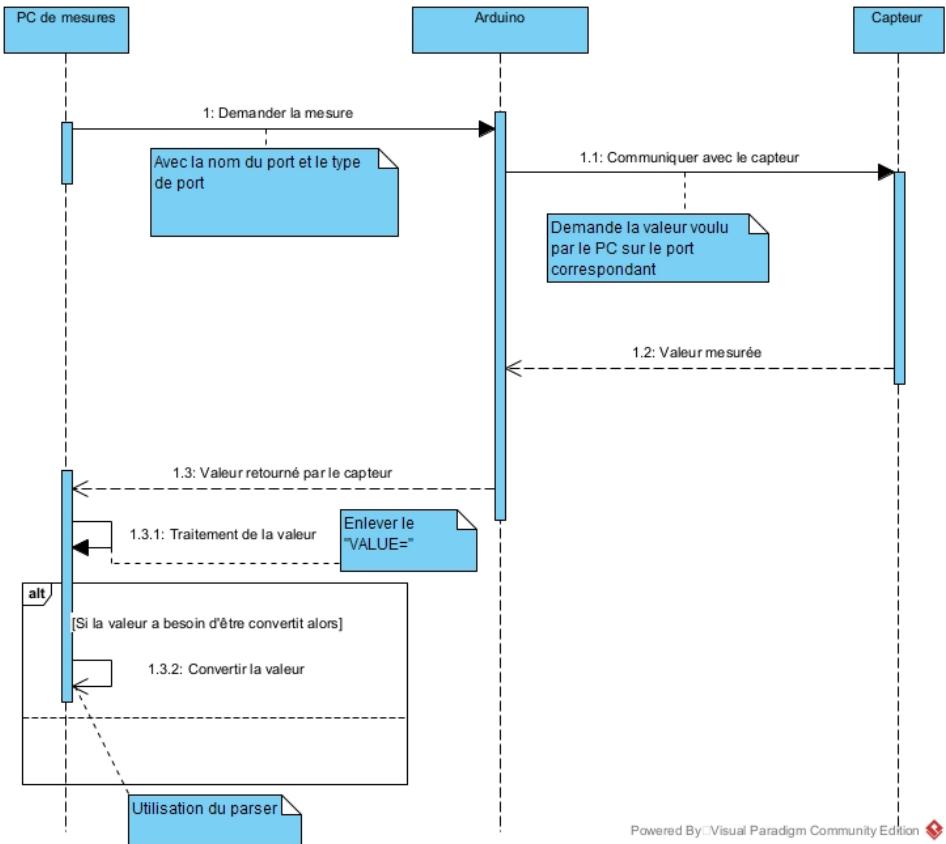
La partie de Vasco consiste quant à elle à l'acquisition des valeurs mesurées par les capteurs et au renvoi vers la fenêtre d'acquisition qui correspond au capteur d'où la mesure provient. À la fin de l'acquisition il sera également possible de faire une exportation des valeurs mesurées de tous les capteurs au format CSV.

2.7) Cas d'utilisation « Acquisition des valeurs »



Ce diagramme se divise en deux cas d'utilisations. En premier, il y a le cas **Retourner Valeur** qui gère la demande de la valeur mesuré par le capteur et son retour. Cette valeur est obtenue à l'aide de l'interpréteur de commande et des obtenu par la fenêtre d'acquisition des données. Ensuite lorsque la valeur mesurée est renvoyée le cas **Demande une Valeur** qui s'occupe lui d'envoyer la valeur mesurée par le capteur vers la fenêtre d'affichage des valeurs.

2.7.1 - Diagramme de séquence



2.7.2 - Description du cas d'utilisation

Nom CU: Acquistion des valeurs	Référence : CU4	Nom : CARIOU
Pré-conditions	1. Brancher la carte Arduino MEGA au PC de mesures avec le câble USB 2. Avoir une liaison ouverte et fonctionnelle 3. Logiciel opérationnel	
Scénario nominal :	1. Le PC de mesures demande la valeur d'un capteur à la carte Arduino avec en paramètre le port et le nom du capteur. 2. La carte Arduino demande au capteur ciblé la valeur 3. La carte retourne la valeur au PC de mesures 4. Le PC traite la valeur mesuré, il enlève le VALUE= 5. Le PC affiche la valeur	
Scénario alternatif A	Condition : Le capteur à besoins d'avoir sa valeur convertit A1. Le PC de mesures demande la valeur d'un capteur à la carte Arduino avec en paramètre le port et le nom du capteur. A2. La carte Arduino demande au capteur ciblé la valeur A3. La carte retourne la valeur au PC de mesures A4. Le PC traite la valeur du mesuré, il enleve le VALUE= puis converti la valeur à avec le parser 5. Le PC affiche la valeur	
Post-conditions	1. La valeur mesurée s'affiche qui correspond au capteur depuis laquelle elle vient.	

3 - Étude préliminaire

3.1) Les librairies utilisées

Pour la réalisation du projet nous utilisons différentes bibliothèque de Qt (cf 3.3). Celles utilisées sont :

- QSerialPort pour la communication série avec la carte Arduino
- QMdiArea qui permettra d'afficher plusieurs fenêtres dans une zone prédéfinie
- QLinguist pour traduire en anglais l'intégralité des interfaces utilisateurs

Les fichiers INI sont des fichiers de configuration, il intègre les paramètres et configurations rencontrées. L'écriture n'a pas codifié, c'est le développeur qui créer lui-même ses paramètres. La seule partie récurrente de ces fichiers est l'encadrage des noms des différentes sections du fichier, chaque nom de section est encadré par deux « <nom_section> »

```
1 [TITRE]
2 Nom=Potentiomètre
3 [PORT]
4 Type=Analog
5 Interface=A
6 // A = Les sorties analogique paires
7 [SUP]
8 Expression=(a*5/1023)*300)/5
9 Unit="°"
```

Figure 11: Fichier ini du capteur potentiomètre

Afin de contenir toutes les informations utiles au bon fonctionnement de l'application. Nous avons créé une structure nommée « Capteur » composée de différents champs qui seront utiles pour la désignation de tous les capteurs utilisés :

- **Nom : QString** : Contiens le nom du capteur
- **Unites:QString** : Contiens l'unité de mesure du capteur associé
- **FormuleMath:QString** : Si besoin, La formule mathématique.
- **Port : Unsigned int** : Contiens le numéro du port
- **Image : pix** : Contiens l'image du capteur associé
- **Type : char**: Contiens le type de valeur retourné (Analogique ou numérique)

3.2) Diagramme de Classe d'application

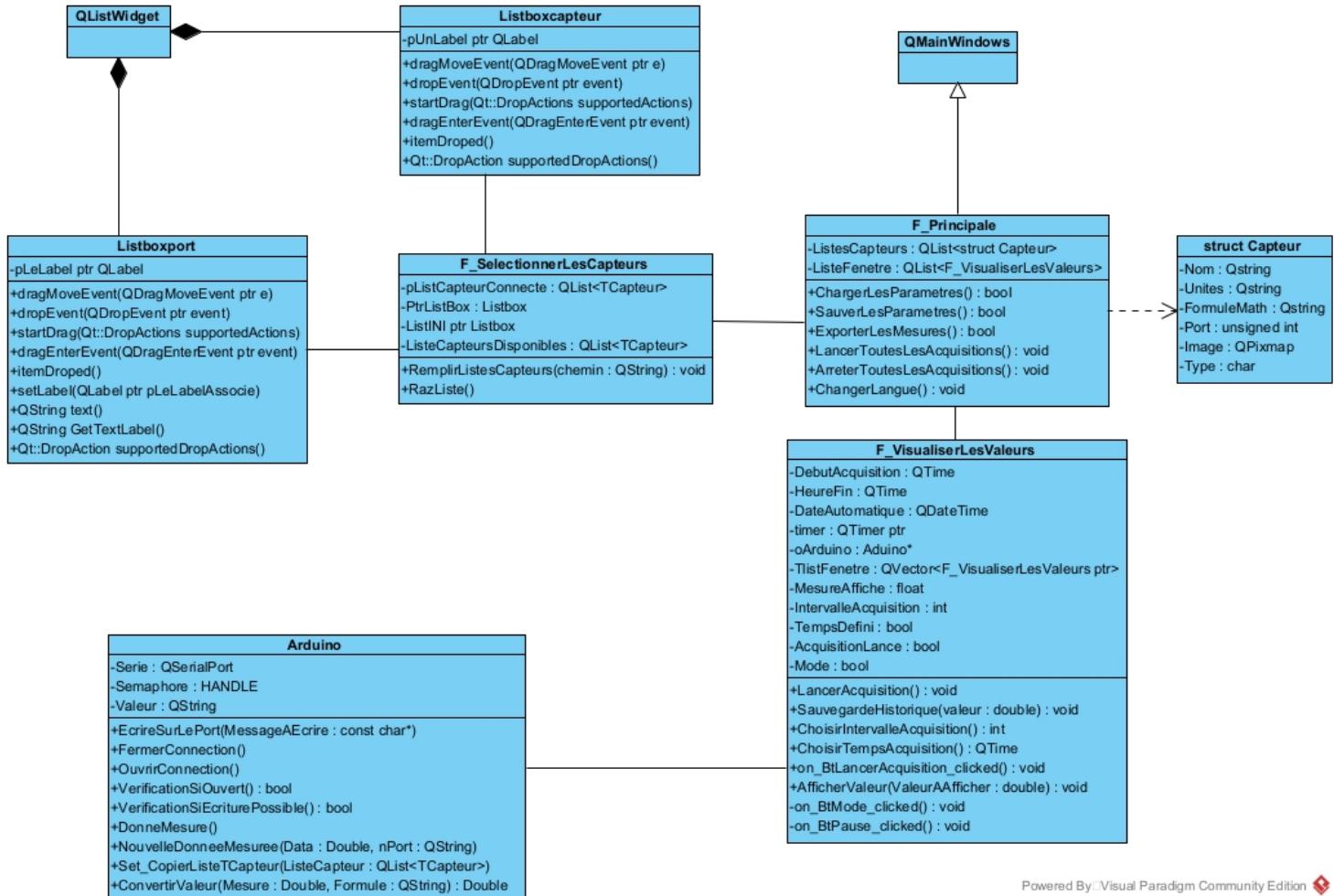


Figure 12: Diagramme de classes

Powered By: Visual Paradigm Community Edition

La classe **F_Principale** décrit l'utilisation de la fenêtre principale avec utilisation d'une liste de capteurs contenant une structure type.

La structure **Capteur** comprend le nom du capteur (Capteur de température par exemple), l'unité correspondante, la formule mathématique s'il y en besoin pour la traduction de la tension électrique en une valeur utilisable. Le port correspond au numéro du port utilisé sur la carte Arduino, cette valeur est utilisable avec le type qui précise si le capteur doit être utilisé grâce à un port Analogique ou Numérique. Dans ces fichiers, il doit y avoir une photographie correspondante aux capteurs à des fins d'illustration dans cette liste.

Explication des attributs :

- **ListCapteurs : QList** : Liste contenant les capteurs et leurs paramètres.
- **ListeFenetre:QList** : Liste contenant les fenêtre d'acquisition de chaque capteur (cf. 2.6.2).

Explicaton des méthodes :

- **ChargerLesParametres() :bool** Méthode servant à charger les paramètres d'une expérience.
- **SauverLesParametres() : bool** Sauvegarder les paramètres d'une expérience.
- **ExporterLesMesures() : bool** Sers à exporter les valeurs mesurées vers un tableau sous le format csv.

- **LancerToutesLesAcquisitions() : void** Méthode pour lancer toutes les acquisitions en même temps (sinon à lancer à la main sur la fenêtre d'acquisition.).
- **ArreterToutesLesAcquisitions() : void** Méthode pour arrêter toutes les acquisitions.
- **ChangerLangue() : bool** Méthode pour changer la langue. Les langues disponibles sont anglaises et françaises.

La classe `F_VisualiserLesValeurs` représente les méthodes et les attributs qui vont servir pour la fenêtre d'acquisition.

Explication des attributs :

- **DureeAquisition : QTime** : Temps total d'acquisition exprimé en seconde.
- **DateDebutAcquisition : QTime** : Si le mode d'acquisition est automatique, détermine le jour et l'heure du début de l'acquisition.
- **IntervalleEntreDeuxAcquisitions : QTime** : Le temps entre chaque acquisition exprimée en seconde

La classe `Capteurs` représente les méthodes et les attributs qui vont servir pour l'utilisation de l'arduino avec les modules groves.

Explication des méthodes:

- **void EcrireSurLePort(const char *MessageAEcrire)** : Ecrit à la carte Arduino le message qu'il y a sur le port puis sauvegarde cette information dans une variable.
- **void FermerConnection()** : Ferme la connection série
- **void OuvrirConnection()** : Ouvre la connection série, si elle est déjà ouverte le message « Port non disponible » est affiché
- **bool VerificationSiOuvert()** : Vérifie que le port est ouvert
- **bool VerificationSiEcriturePossible ()** : Vérifie que l'écriture est ouverte
- **Double MesureAConvertir (Data : double, QString FormuleMathCapteur)** : Méthode qui utilise le parser afin de convertir la valeur d'une valeur analogique à un grandeur physique affichable et exploitable pour l'utilisateur.

Explication mutateur :

- **Set_CopierListeCapteur (QList<TCapteur> ListeCapteurF_Principale)** : Copie la liste des capteur utilisé de `F_Principal` vers Arduino. Ce mutateur sera utiliser dans la classe `F_Principale`.

Explication Slots :

- **void DonneMesure()** : Méthode la plus importante de la classe, elle est appelé dès qu'une nouvelle données est prête à être lu. Elle récupere, avec le numéro du port récupérer dans `EcrireSurLePort`, la formule du capteur associé. Ensuite elle récupere la valeur, fait le premier traitement puis s'il y a une formule) convertis la valeur avec le parser. Finalement elle renvoie cette valeur en faisant une émission qui sera reçue par la classe `F_VisualiserLesValeurs`.

Explication signal :

- **void NouvelleDonneeMesuree(QString sPort, double Data)** : S'occupe de l'émission faite à la fin de `DonneMesure`. Elle contient le numéro du port sur lequel est branché le capteur et permet de différencier tous les capteurs utilisés et la valeur à affichés.

Explications des attributs :

- **QSerialPort Serie** : Objet de type `QSerialPort` qui servira à configurer et utiliser la liaison série

- **HANDLE Semaphore** : Sémaphore qui servira pour protéger la liaison série lorsque plusieurs fenêtres voudront l'utiliser.
- **QString Valeur** : La valeur mesurée

La classe F_SelectionnerLesCapteurs représente les méthodes et les attributs qui vont servir pour la fenêtre d'acquisition

3.3) Outils utilisée

Trello :

- Gestionnaire de projet en ligne
- Adapté pour suivre la méthode agile (SCRUM)
- Nous avons choisi cet outil, car il nous permet de nous organiser via un site web afin de mener à bien notre projet.



Qt :

API orienté objet utilisant le langage C++. Ce logiciel et ce langage nous permettre de concevoir l'intégralité de notre porjet que ce soit le back et le front end.

La version utilisée sera QT6.2 (Long Term Support Release)

Les versions LTSR sont des versions garantis d'être tenu à jour pendant une longue période, ainsi le projet restera fonctionnel pendant tout aussi longtemps



Gitlab :

- Plateforme de développement collaboratif
- Permet de piloter des dépôts de code source et de gérer leurs différentes versions
- Open-source • Nous avons choisi cet outil, car il nous permet de partager nos réalisations de façon très simple avec toute l'équipe de projet, nos professeurs et le client sur un projet Gitlab privé qu'il a créé.



5 5-GrovExAO

Identifiant de projet : 32041214 Quitter le projet

Ajouter aux favoris 1 Créer une divergence 0

~ 112 commits 1 branche 0 étiquette 49 Mo fichiers 49,2 Mo Storage

main 5-GrovExAO / + Historique Rechercher un fichier EDI Web Clone

P2022 Dylan CHARRON authored 3 hours ago 90f4540b

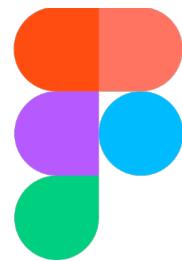
Upload File Configuration de l'intégration et de la livraison continues Ajouter un README Add LICENSE

Ajouter un CHANGELOG Ajouter un CONTRIBUTING Ajouter une grappe de serveurs Kubernetes Configure Integrations

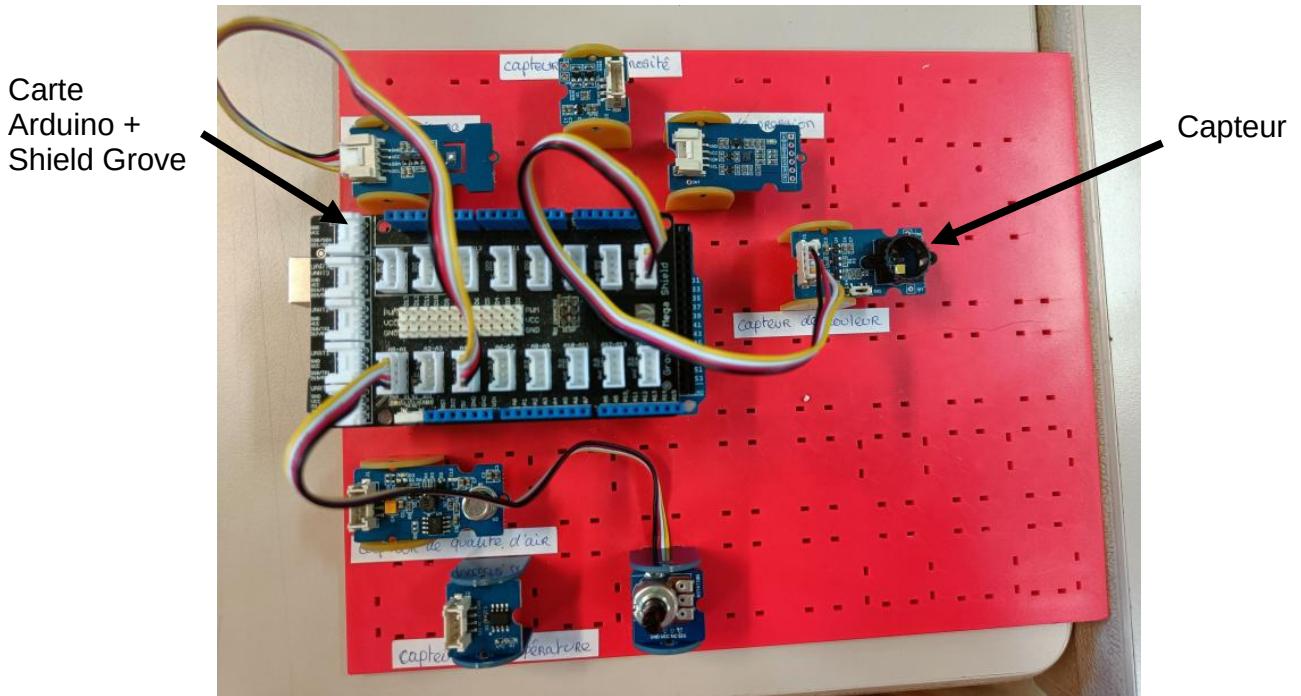
Nom	Dernier commit	Dernière mise à jour
Selection	P2022	1 week ago
backlog	P2022	2 weeks ago
documentation	P2022	4 hours ago
presentations	P2022	4 hours ago
realisations	P2022	3 hours ago
ressources	P2022	2 weeks ago
untitled	P2022	1 week ago
.gitlab-ci.yml	Configure SAST in `.gitlab-ci.yml`, creating th...	1 month ago
_SynchroProjet.bat	P2022	2 weeks ago
ajoutgitlab	Ressources pour les étudiants	1 month ago
ajoutgitlabProjet	P2022	3 weeks ago

Figma :

Figma est un site internet permettant de réaliser des maquettes d'interface graphique.
Les maquettes permettent de visualiser à quoi ressemblera l'interface graphique une fois terminer.
Il permet de créer plusieurs fenêtres avec des zones de texte, des images ou des boutons.
Il est assez simple à prendre en main



4 - Recette



Pour le bon fonctionnement de ce projet, il faudra posséder et mettre en place une carte Arduino MEGA avec un shield GROVE. La carte aura besoin d'être connectée au PC de mesure avec une liaison USB. Pour pouvoir faire les mesures, il faudra prendre tous les capteurs voulu et compatibles avec le shield, il faudra ensuite les brancher au shield et recueillir les données avec notre application.

5 - Conclusion

Notre projet se répartit en 3 grandes parties distinctes :

- Paramétrage matériel d'une acquisition dans l'interface graphique (Choix des capteurs, branchement au port).
- Paramétrage logiciel d'une acquisition dans l'interface graphique (mode, temps d'acquisition, sauvegarde) et visualisation des valeurs mesurées.
- Acquisition des mesures et communication entre la carte Arduino et le PC de mesures.

Avant tout, l'utilisateur doit choisir les capteurs qu'il va utiliser avec le numéro de port associé. Cette sélection se fait avec un drag & drop (Glisser – Déposer). L'intégralité des capteurs disponibles sont présents dans une liste déroulante, avec une photographie d'illustration. Chaque capteur a un fichier INI associé contenant toutes les configurations nécessaires aux paramètres du capteur. Il sera également possible de charger des paramètres d'une expérience antérieure. À ce jour l'interface graphique est fonctionnelle avec la liste des capteurs. Pour l'instant les tâches restantes sont :

- La détection d'erreurs de placement des capteurs

Ensuite, la deuxième partie du projet est la création de la fenêtre principale dans laquelle seront affichées les fenêtres de visualisation des valeurs des capteurs. Le deuxième point de cette partie sera le paramétrage des différentes acquisitions. À ce jour, la fenêtre principale est créée et l'affichage de la fenêtre de visualisation des mesures est faite. **Pour la suite du projet il reste à faire**

Pour finir, la gestion de l'acheminement des données est effectué grâce à une liaison série ainsi qu'un interpréteur de commande. L'utilisation d'un parser mathématique est nécessaire pour certains capteurs, où la donnée renvoyée ne correspond pas à la valeur mesurée. Pour le moment le recueil de donnée fonctionne sur un capteur à la fois. Par la suite il faudra inclure le multitâche pour pouvoir faire plusieurs acquisitions en même temps sur plusieurs capteurs. Pour finir il faudra intégrer les formules mathématiques si nécessaire avant de transmettre la donnée à la fenêtre d'acquisition. Les formules mathématiques si nécessaire avant de transmettre la donnée à la fenêtre d'acquisition.