INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES TOULOUSE

RAPPORT

Bureau d'études de C++

Moniteur de plante autonome

Julie MOREAU
Madeline RONTARD
4 AE-SE
Groupe 3

Sommaire

I. INTRODUCTION

Pour ce projet de C++, nous sommes parties du constat que s'occuper d'une plante n'est pas aisé, surtout pour des novices. En effet, un préjugé plutôt répandu dit qu'il suffit juste d'arroser une plante pour la faire vivre. Cependant, nombreux sont ceux qui oublient d'arroser leurs plantes ou bien qui, pensant bien faire, l'arrose en trop grande quantité. De plus, pour de nombreuses espèces de végétaux, l'humidité dans l'air ainsi que la luminosité ambiante sont aussi importantes qu'un arrosage adéquat. Nous avons donc choisi de créer un moniteur de plante capable de contrôler l'environnement d'une plante en termes d'humidité dans l'air, dans la terre, ainsi que son exposition à la lumière, en fonction de paramètres extérieurs.

II. LE MONITEUR DE PLANTE AUTONOME

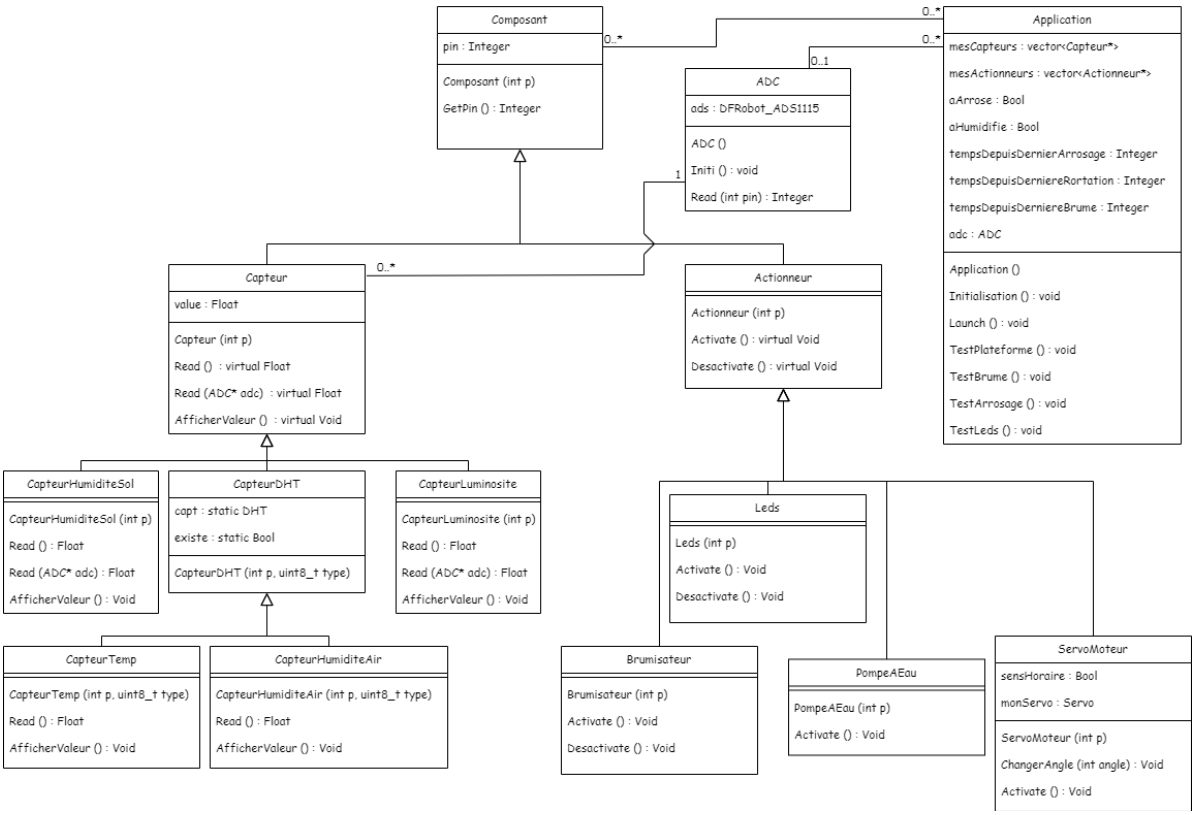
A/ Les fonctionnalités dont nous avons besoin

- Gestion de l'humidité dans l'air :
    - Nous souhaitons la détecter à l'aide d'un capteur d'humidité et en ajouter à l'aide d'un brumisateur lorsque celle-ci descend en dessous de 40% jusqu'à atteindre 70% environ. Ces pourcentages seront en fait modulés par la température (mesurée avec un capteur de température) pour qu'une température plus élevée qu'habituellement entraîne une plus haute humidité dans l'air. Ces seuils sont évidents adaptables en interne pour pouvoir être en accord avec les besoins du type de plante (plante tropicale, plante fleurie, verte…).
- Gestion de l'humidité dans la terre :
    - Nous souhaitons la mesurer à l'aide d'un capteur capacitif d'humidité planté dans la terre de la plante et arroser la plante à l'aide d'une mini pompe à eau reliée à une réserve d'eau. Cette pompe se déclenchera pendant 1 seconde toutes les 4 secondes lorsque l'humidité du sol atteint 40 % jusqu'à atteindre 80% d'humidité. Ce fonctionnement, toutefois étonnant, permet de laisser le temps à l'eau de se diffuser uniformément dans la terre et ainsi de mesurer une humidité représentative de ce que la plante reçoit sur ces racines. Ce mécanisme permet également d'éviter de noyer la plante par un arrosage abondant.
- Gestion de l'exposition à la lumière :
    - Nous souhaitons pouvoir gérer l'exposition à la lumière naturelle et à la lumière artificielle afin que notre système soit autonome en période de vacances et en période classique. Ainsi, nous mesurerons la lumière ambiante avec un capteur de luminosité et nous allumerons un bandeau LED placé au-dessus de la plante lorsque la lumière descend en dessous 20%. Dans les faits, nous n'avons pas de bandeau LED. Celui-ci sera donc représenté par la LED intégrée sur la carte ESP8266. D'autre part, nous placeront la plante sur une plateforme, capable de tourner grâce à un servo moteur. Cette plateforme tournera de manière régulière (toutes les 10 secondes pour pouvoir faire une démonstration sur quelques minutes) afin

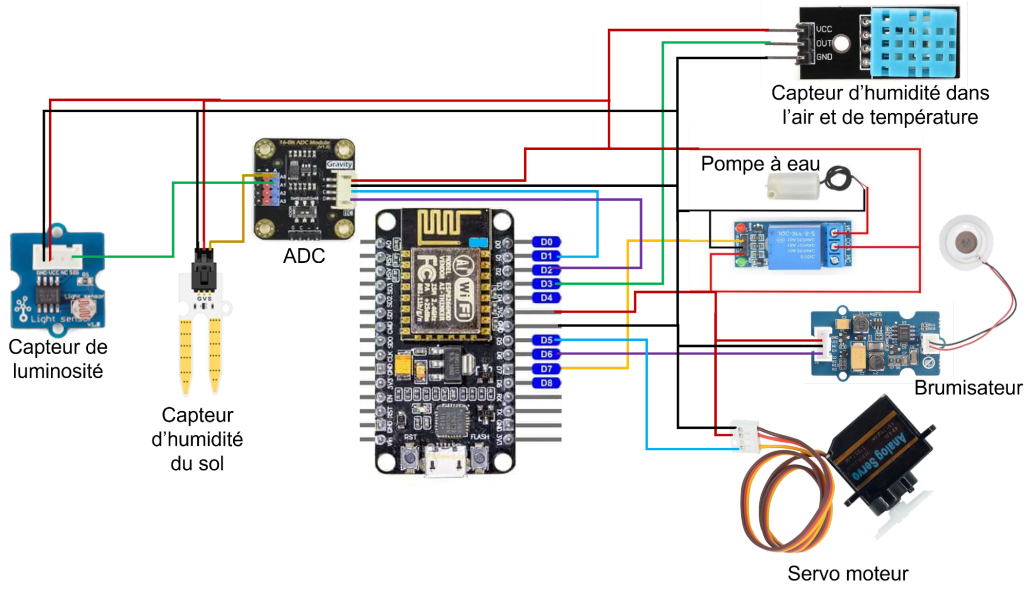
que la plante, si celle-ci est exposée au soleil direct à travers une fenêtre, ne soit pas brûlée par celui-ci et qu'elle puisse grandir de façon uniforme.

B/ Le diagramme de classe

Le diagramme de classe de notre système se compose de nombreuses classes :



C/ Le schéma électronique du système



D/ Fonctionnement logiciel

Notre fichier principal crée un objet de type Application A. Dans le setup, il lance l'initialisation de cette application A. Les actionneurs sont alors créés. Ils sont stockés dans le vecteur mesActionneurs de l'application. De la même manière, les capteurs sont créés et stockés dans le vecteur mesCapteurs de l'application. On initialise également un adc qui servira à obtenir les valeurs des capteurs analogiques car nous n'avons accès qu'à une entrée analogique sur la carte.

Puis la fonction loop du fichier principal va appeler de manière répétitive la méthode Launch() de l'application. Launch() exécute successivement les méthodes TestPlateforme(), TestBrume(), TestArrosage() et TestLeds().
La méthode TestPlateforme() consiste à activer la plateforme sur laquelle est posée le pot (donc la faire tourner d'un angle donné) après au moins un certain temps d'attente.
La méthode TestBrume() consiste à activer le brumisateur suivant un cycle d'hystérésis. On attend de mesurer un certain niveau d'humidité bas de l'air (modulé par la valeur de la température) pour activer la brume. Celle-ci sera désactivée quand le taux d'humidité dans l'air sera à un certain niveau d'humidité haut d'air (également modulé par la valeur de la température). L'intérêt de moduler notre seuil par la valeur de la température est qu'une plante à 10°C a besoin de moins d'humidité dans l'air qu'une plante à 30°C.
La méthode TestArrosage() consiste à déclencher la pompe à eau suivant un cycle d'hystérésis. On attend de mesurer un certain niveau d'humidité bas pour activer une première fois la pompe, puis elle est activée régulièrement jusqu'à obtenir un niveau d'humidité haut.
La méthode TestLeds() consiste à allumer la LED que si la lumière ambiante est en dessous d'un certain seuil. Cela signifie que celle-ci n'est pas suffisante pour subvenir aux besoins de la plante. La fonctionnalité de s'éteindre la nuit n'a pas été développée car il faudrait pouvoir récupérer l'heure réelle.

III. Conclusion

Les fonctionnalités visées ont pu être réalisées. Une démonstration dans des conditions réelles est plutôt compliquée à montrer. En effet, les durées entre deux arrosages sont normalement de l'ordre de quelques jours, la plateforme n'a pas d'utilité à tourner toutes les 10 secondes, surtout dans un intérêt d'économie d'énergie, et l'ajout d'humidité dans l'air avec un brumisateur est un processus long. D'autre part, les valeurs de seuils choisies pour tous les actionneurs ont été données de façon approximative et sans chercher à s'adapter à un type de plante en particulier. Pour un développement industriel, il faudrait pouvoir adapter les seuils avec des boutons et pouvoir surveiller les différents capteurs à l'aide d'un écran LCD. De plus, un bouton marche/arrêt et un bouton reset n'auraient pas été de trop, mais par manque de temps nous n'avons pas pu les inclure. En effet, nous avons perdu beaucoup de temps à essayer de faire marcher certains capteurs parfois capricieux, et certains moteurs que nous n'avons finalement pas inclus dans notre système. Nous n'avons obtenu un ADC, un capteur d'humidité et un capteur de température qu'à la dernière séance ce qui nous a laissé peu de temps pour réaliser certaines fonctionnalités. Malgré tout, nous sommes très satisfaites du résultat. Notre moniteur de plante autonome est un prototype actuellement utilisable pour une utilisation personnelle.