INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES TOULOUSE

4AE-SE

Application du C++ au domaine des objets connectés

Rapport Bureau d'étude

Conception d'un système de gestion dynamique de feux de signalisation pour voiture/piéton

SALHI Abdelmajid
ROUMANOS Emmanuel
PICARD Christophe

Année universitaire 2023-2024

Sommaire

Présentation de l'idée du projet.....1
Capteurs et actionneurs reliés à la carte ESP.....1
Diagramme de Cas d'Utilisation.....2
Diagramme de Classe.....3
Séquence de Fonctionnement.....3
Séquence 1 : Fonctionnement normal sans intervention des piétons.....3
Séquence 2 : Intervention des piétons avec le bouton.....4
Conclusion.....5
Bilan du Projet.....5
Problèmes rencontrés.....5
Améliorations Possibles.....6

Introduction

L'objectif de ce bureau d'études est de réaliser un système innovant composé de capteurs et d'actionneurs. Ce projet, réalisé en langage C++, met en évidence les principes de programmation orientée objet, notamment l'héritage de classe et le polymorphisme. La programmation et l'intégration des composants permettent d'illustrer ces concepts étudiés en cours.

Le projet repose principalement sur une carte ESP, fournie par les enseignants, qui est compatible avec les bibliothèques Arduino pour simplifier le développement et l'intégration matérielle.

Nous remercions Mr GOURVES pour son aide tout au long du projet.
L'équipe de travail est composée de Salhi Abdelmajid , Roumanos Emmanuel et Christophe Picard, ensemble nous avons collaboré pour mener au mieux ce projet.

Présentation de l’idée du projet

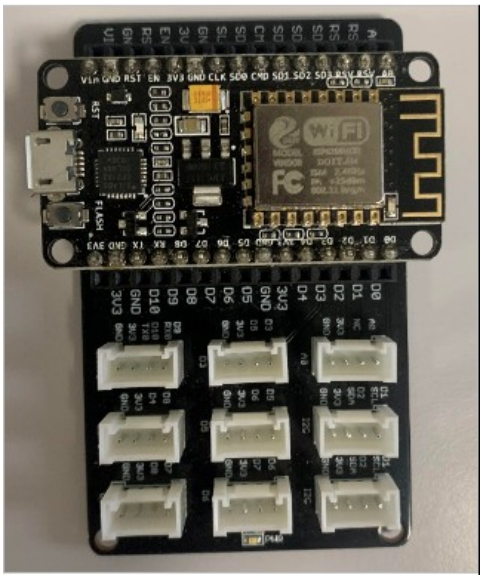
Ce projet consiste en la conception d’un système de gestion dynamique de feux de signalisation pour voitures et piétons. L’objectif est de garantir une fluidité optimale du trafic tout en assurant la sécurité des piétons. Le système réagit dynamiquement grâce à un bouton-poussoir utilisé par les piétons pour demander à traverser. De plus, nous avons intégré une option visant les personnes mal voyantes. En effet, lorsqu’ils appuient sur le bouton-poussoir et que le feu des véhicules est vert, un buzzer intervient pour les prévenir qu’ils ne doivent pas encore passer jusqu’à ce que le feu devienne rouge.

Capteurs et actionneurs reliés à la carte ESP

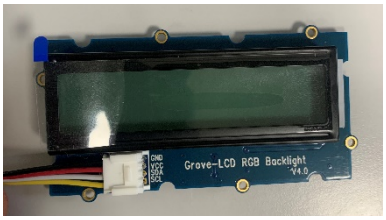
- **LCD RGB Backlight** : Affiche l’état du feu voiture (rouge, jaune, vert).
- **LED pour piétons** : Indique rouge ou éteint pour guider les piétons.
- **Bouton poussoir** : Permet aux piétons de demander l’arrêt du trafic.
- **Buzzer** : Émet un signal sonore pour alerter les piétons lorsque leur demande est prise en compte et que le feu véhicule n’est pas encore rouge.

Bouton poussoir

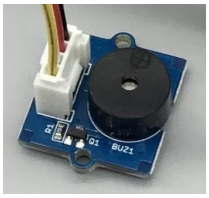
LED piéton



Carte microcontrôleur ESP



LCD RGB Backlight



Buzzer

Diagramme de Cas d'Utilisation

L'interaction entre le système et son environnement repose sur deux types d'utilisateurs principaux :

- Les **voitures** : Circulent lorsque le feu est vert et doivent s'arrêter lorsque le feu est rouge.

- Les **piétons** : Utilisent le bouton pour demander à traverser, ce qui déclenche une modification de l'état des feux.

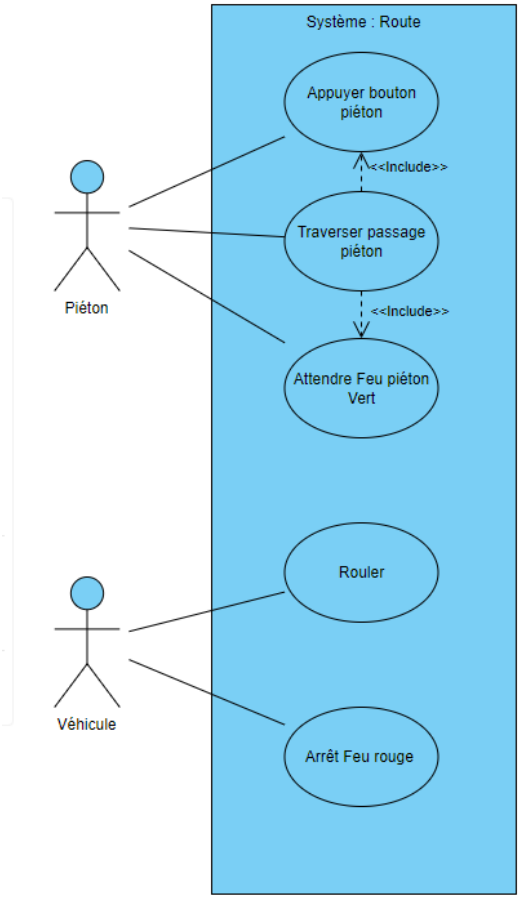
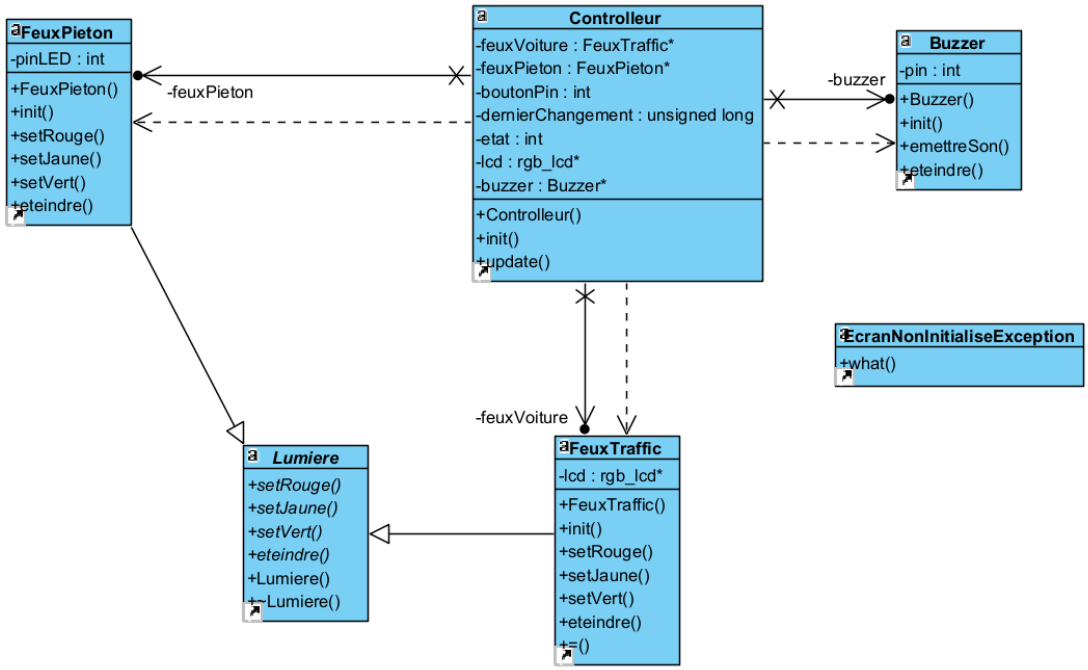


Diagramme de Classe

Pour mener à bien notre projet, nous avons réalisé un diagramme de classe comportant *5 classes* afin de contrôler tous les composants mis en jeu. Pour avoi une idée globale de toutes les méthodes et attributs déclarés au sein de nos différentes classes, nous vous proposons le diagramme de classe suivant :



Nous avons tout d'abord déclaré une classe mère Lumiere afin de l'utiliser pour ses deux classes filles FeuxTraffic et FeuxPieton. A travers ces deux classes, nous redéclarons les méthodes de la classe Lumiere (polymorphisme) pour pouvoir utiliser au mieux leurs composants désignés : la LED pour le feu piéton et le LCD pour le feu du traffic.

Une classe Buzzer est également déclarée pour pouvoir utiliser le buzzer aux moments où nous le souhaitons.
La classe *EcranNonInitialiseExeception* est héritée de la classe Exception basique pour pouvoir gérer l'erreur qui pourrait se produire si l'écran n'est pas initialisé.
Et enfin, nous avons décidé d'utiliser une classe Controlleur qui va permettre de gérer tout le fonctionnement de notre projet afin d'allégir notre main principal. Grâce à cette décision, nous pouvons directement modifier le fonctionnement à partir de cette classe sans modifier le main principal.

Afin d'illustre le comportement générale du système , on propose les différentes séquence de fonctionnement .

Séquence 1 : Fonctionnement normal sans intervention des piétons

1. **Initialisation :**
    - Les feux sont configurés au démarrage :

      - Voitures : Feu vert.

      - Piétons : Feu rouge.
2. **Feu vert pour les voitures (État 0) :**

   - Les voitures circulent normalement pendant **20 secondes**.

   - Le bouton piéton est surveillé pendant cette phase.

3. **Transition jaune pour les voitures (État 2) :**

   - Les feux des voitures passent au jaune pendant **2 secondes** pour avertir les conducteurs.

4. **Feu rouge pour les voitures et vert pour les piétons (État 1) :**

   - Les piétons peuvent traverser pendant **10 secondes**.

   - Les voitures sont à l'arrêt.

5. **Retour au feu vert pour les voitures (État 0) :**

   - Une fois la traversée terminée, les feux des voitures passent au vert et ceux des piétons au rouge.

Séquence 2 : Intervention des piétons avec le bouton

1. **Situation initiale :**

   - Le système est en **État 0** (feu vert pour les voitures, feu rouge pour les piétons).

2. **Bouton pressé par un piéton :**

   - Le système détecte l'appui sur le bouton et émet un signal sonore pendant **2 secondes**.

   - Le système bascule à **l'État 2** (transition jaune).

3. **Transition jaune pour les voitures (État 2) :**

   - Les feux des voitures passent au jaune pendant **2 secondes**.

4. **Feu rouge pour les voitures et vert pour les piétons (État 1) :**

   - Les piétons traversent pendant **10 secondes**.

5. **Retour au cycle normal (État 0) :**

   - Les feux reviennent au vert pour les voitures et au rouge pour les piétons.

Conclusion

Bilan du Projet

Le projet respecte les critères du cahier des charges en mettant en œuvre les principes de la programmation orientée objet. Le code est structuré de manière modulaire, avec une séparation claire des responsabilités entre les différentes classes.

En ce qui concerne l'utilisation du C++, plusieurs éléments clés ont été intégrés :

-La création de plusieurs classes pour modéliser les différents composants du système.

- L'utilisation du mécanisme d'héritage pour organiser au mieux les relations entre les classes.

-La redéfinition de l'opérateur = que l'on à définit mais que l'on a jugé pas pertinent de l'utiliser dans le projet, c'est-à-dire qu'il n'y a pas une situation où il serait judicieux de l'utiliser

-L'utilisation de la Library (STL) pour gérer les collections d'objets dynamiques afin de gérer l'état des feux de signalisation. Grâce à ce vecteur, nous pouvons insérer de nouveaux feux à l'avenir

- La gestion des erreurs à l'aide d'exceptions spécifiques pour rendre le système plus robuste.

Problèmes rencontrés

Au cours de ce BE, nous avons rencontré des problèmes au niveau du fonctionnement des différents composants mis à notre disposition. Mais au long des différentes recherches et de l'utilisation des ressources mises par Arduino, nous avons compris petit à petit comment les utiliser au mieux.

Au niveau du codage en lui-même, nous n'avons pas vraiment rencontré de problèmes. Le problème était plutôt sur la logique de programmation de nos différentes structures, c'est-à-dire, comment et où les utiliser , également des problèmes au niveau de l'utilisation des différentes boucles(for,while) par rapport à l'insertion sur une carte embarquée , mais également des problème de synchronisation temporel pour la gestion des feux qui ont par la suite était résolus.

Améliorations Possibles

- Utilisation d'un Capteur de proximité : Nous pourrions ajouter des capteurs pour détecter les voitures et ainsi adapter dynamiquement les durées des feux.

- Nous aurions pu faire un usage plus judicieux de la librairie Vecteur et non l'utiliser comme nous l'avons utilisé ici.

- Nous aurions pu également faire une situation concrète où l'exception serait mise en valeur afin que vous puissiez constater qu'elle fonctionne bien, tout comme pour l'utilisation de l'opérateur = que nous avons redéfini.