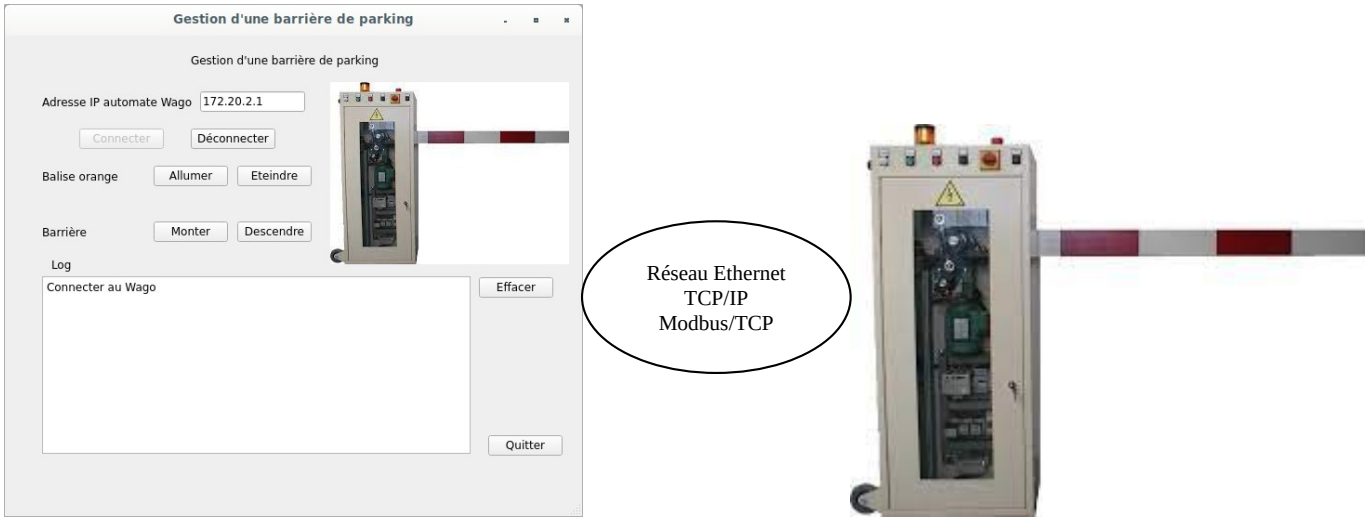
Epreuve E6.2 (Projet) : Candidat CNED

1. Présentation

Le dossier, support de cette épreuve, concerne une maquette didactique représentant une barrière de parking de la marque DEC industrie.
Ce système est composé de :

- Une partie opérative : Barrière, balise orange, 2 plaques boucles de courant
- Un automate WAGO pour piloter les capteurs et actionneurs de la partie opérative. Cet automate comprend une interface Ethernet, utilise le protocole Modbus/TCP. Son adresse IP est 172.20.2.1.
- Un logiciel développé en C++, avec le framework Qt, pour gérer la barrière et sa balise, via le protocole Modbus/TCP.



2. Documentation

- Document de présentation de la barrière de parking DEC Industrie.
- Schémas électriques de la barrière de parking.
- Manuel de l'automate WAGO 750-881.
- Adressage des données des contrôleurs WAGO-I/O-SYSTEM 750.
- Logiciel de gestion de la barrière de parking sous forme d'un projet Qt, écrit en langage C++.

3. Exploitation et mise en œuvre

Les pistes de travail et de réflexion pourront aborder les points listés ci-dessous.
Ces pistes ne sont données qu'à titre indicatif. Le candidat pourra exploiter les points qui lui semblent pertinents et traiter tout autre domaine qu'il juge opportun.

- Etablir les diagrammes UML ou SysML correspondant à l'analyse, la conception et la réalisation du système et de son logiciel de gestion.
- Expliquer les caractéristiques et les fonctions de la partie opérative.
- Expliquer la configuration de l'adressage IP de l'automate WAGO.
- Etudier la communication réseau entre le logiciel de gestion et l'automate WAGO.
  - Présenter le protocole Modbus/TCP, ainsi que le plan d'adressage Modbus de l'automate WAGO en lien avec les éléments de la partie opérative.
- Comprendre et présenter le logiciel de gestion de la barrière, utilisant le framework Qt.
- Critiquer et proposer des améliorations du logiciel de gestion pour
  - Afficher, utiliser les entrées logiques (TOR) reliées à l'automate WAGO (attention, les bits ne sont pas dans le même ordre que le câblage : 7=KA1, 6=clavier, 5=POM, 4=Ticket, 3-2=boucle, 1=FDCBas, 0=FDCHaut),
  - Arrêter les actions "monter" ou "descendre" quand la barrière arrive en butée (utilisation des capteurs fin de course),
  - Prendre en compte les erreurs dans l'utilisation du protocole Modbus/TCP,
  - Créer une classe C++ "Barriere" pour dissocier l'IHM graphique Qt de la classe d'interface pilotant la barrière,
  - Animer un synoptique en fonction de l'état de la partie opérative : Balise orange, barrière, capteurs, etc.