RAPPORT E6.2

BTS SYSTÈMES NUMÉRIQUES

2021-2022

ENZO LOUAIL

INTRODUCTION

Plan

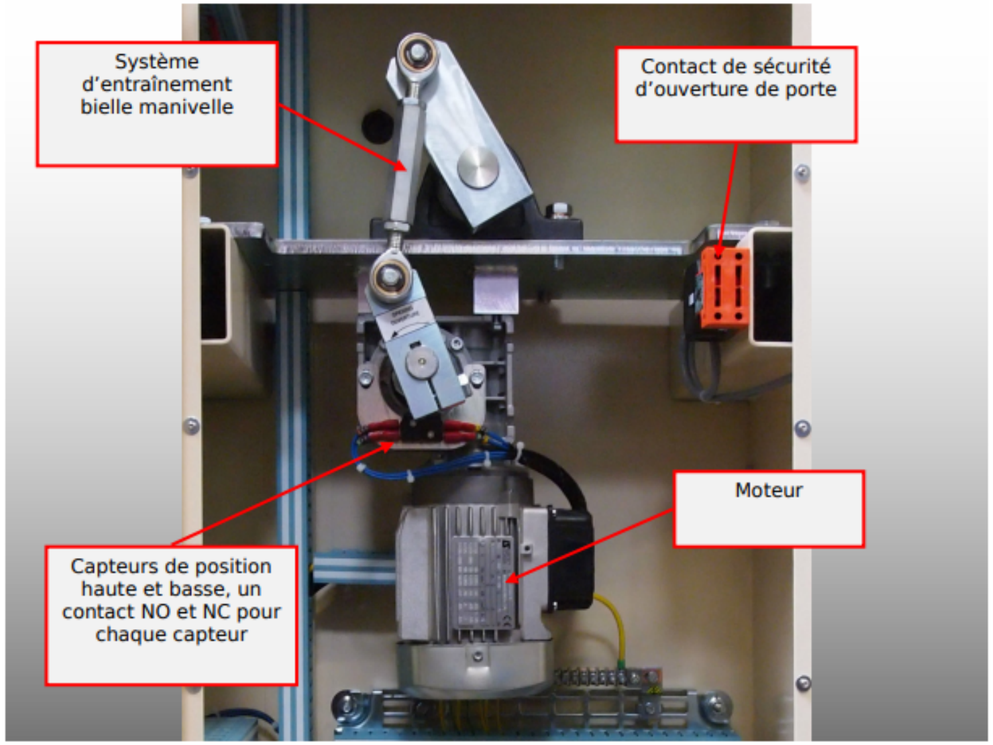
1. Présentation de la barrière de parking
2. Limitation du logiciel et modifications apportées

1. PRÉSENTATION DE LA BARRIERE DE PARKING

Système :

- Partie opérative
- Automate WAGO
- Logiciel C++

PARTIE OPÉRATIVE



BARRIÈRE

Montée / descente : inversion du sens du moteur

BALISE ORANGE

Allumer / éteindre : Clignotement

MOTEUR

Accélérations et décélérations progressives

AUTOMATE WAGO

05

FAIRE LE LIENS ENTRE PARTIE OPÉRATIVE ET LOGICIEL

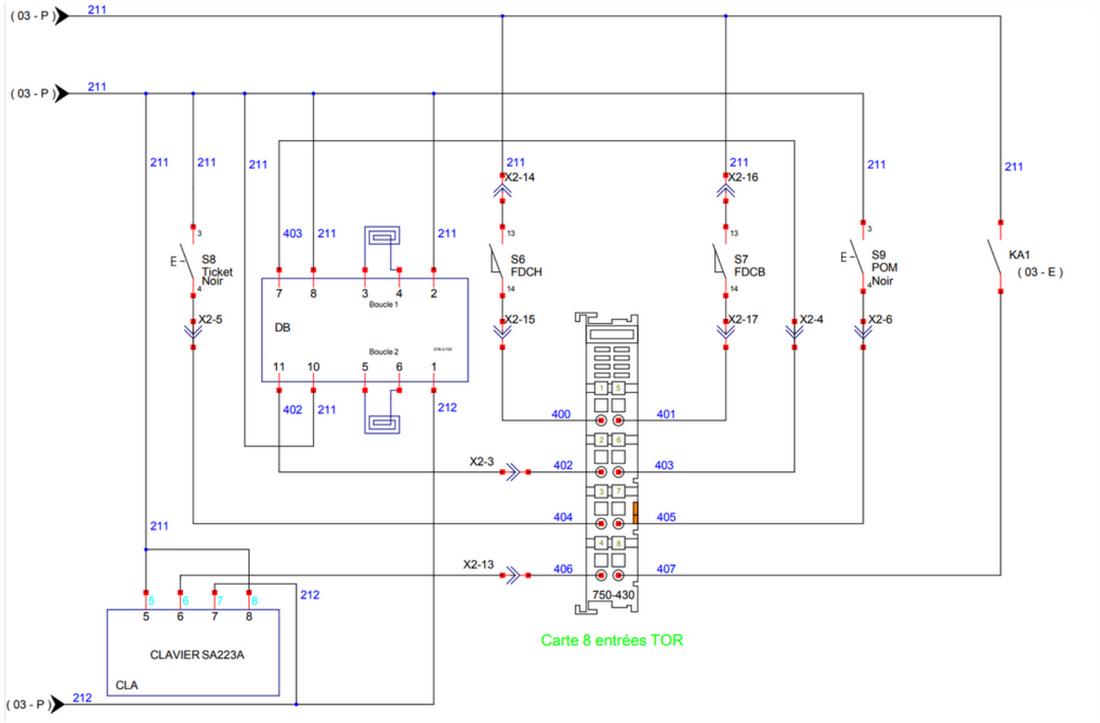
Via Ethernet
Adrresse IP

CONTRÔLE DES ACTIONNEURS ET CAPTEURS

contacteur KA1
Clavier
Bouton poussoir POM et Ticket
2 capteurs boucle
capteurs de fin de course haut et bas

UTILISATION DU PROTOCOLE MODBUS / TPC

SCHÉMAS ÉLECTRIQUES



Carte 8 entrées TOR

DIFFÉRENTS CAPTEURS DISPONIBLES EN ENTRÉES

Contacteur KA1

Capteurs de fin de courses haut et bas

Bouton poussoirs POM et Ticket

Plaques Boucles de courants

Clavier

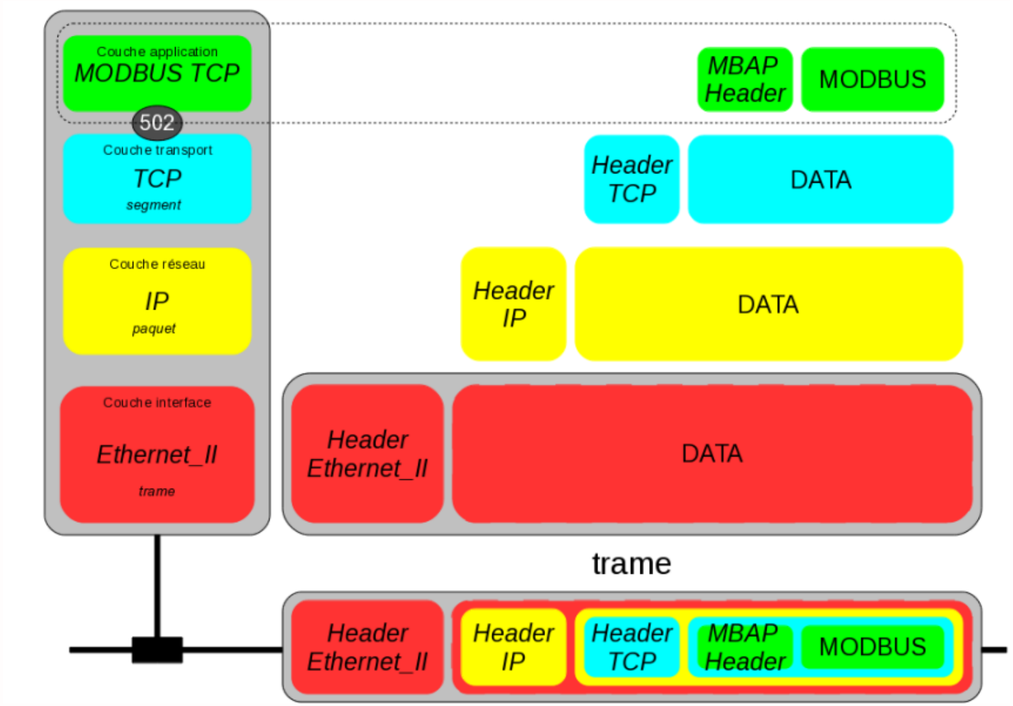
DIFFÉRENTS CATEURS EN SORTIES

Capteurs de fin de courses haut et bas

Balise orange

07

Protocole Modbus / TCP



MODBUS / TCP

Protocole de communication

Automate programmable

Mode Maitre / Esclave

Architecture Client / serveur

Module serveur sur écoute : port 502

Encapsuler dans une trame TCP

TRAMES

2 Modes : RTU ou ASCII

Format Questions réponses

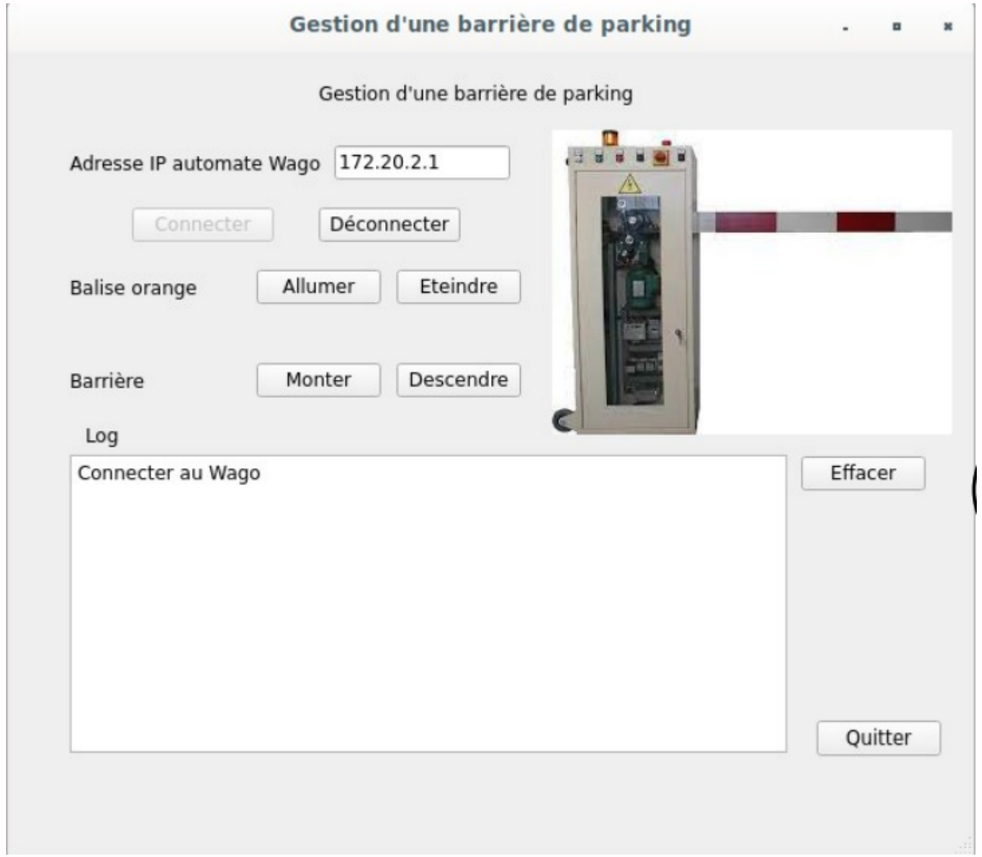
Numéro station esclave : 1 octet

Code fonction + bit erreur : 1octet

Information spécifique / données transmises : n octet

Mot de controle : 2 octet

2. LIMITATION DU LOGICIEL ET MODIFICATIONS APPORTÉES



Logiciel :

- C++
- Framework QT
- Contrôle Balise / Barrière
- Logs

Diagramme de classe

2 Classes Majeurs :

- EsclaveModbusTCP

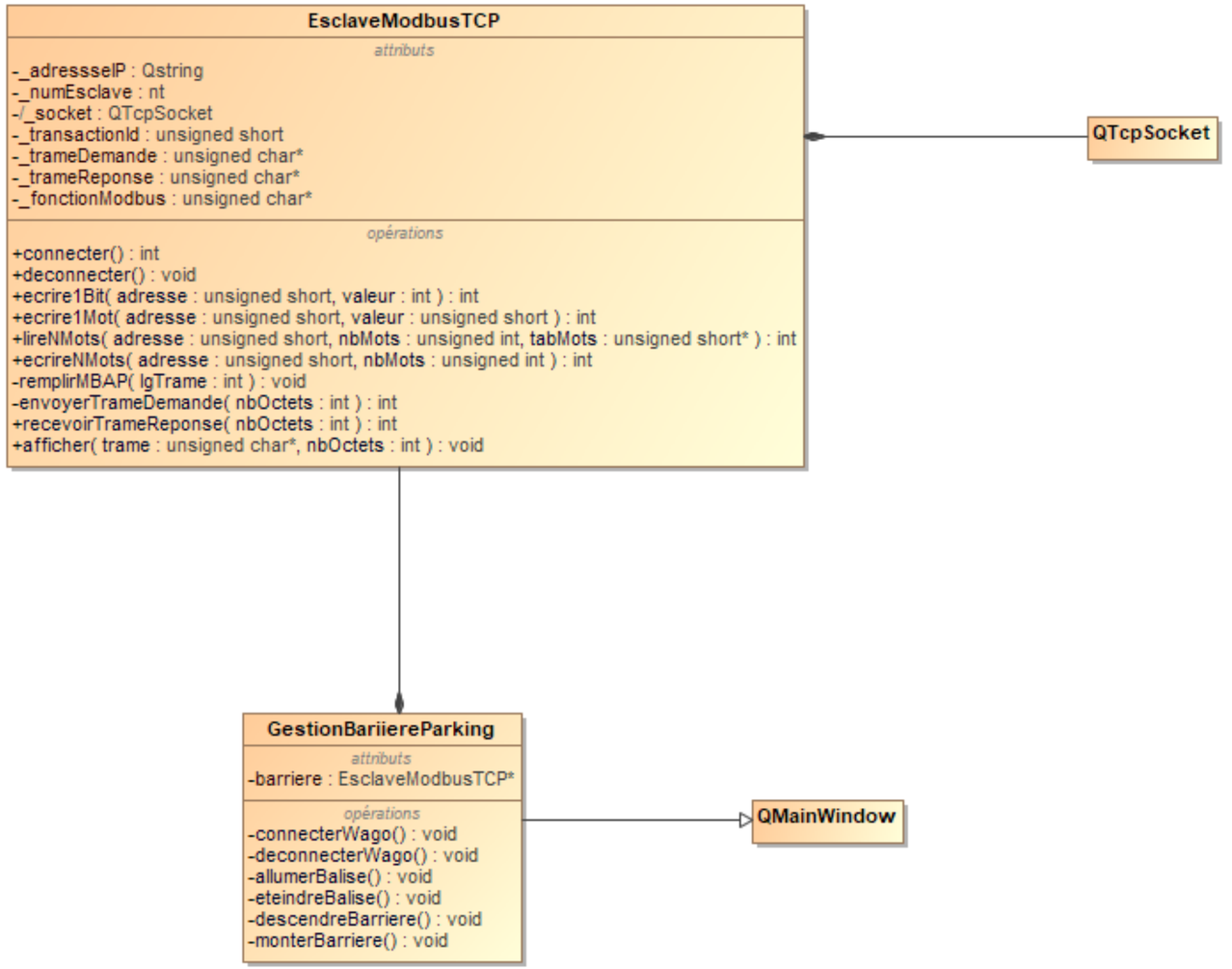
- GestionBarriereParking

EsclaveModbusTCP :

- Fonctions Modbus TCP
- Connexion TCP avec le Wago

GestionBarriereParking :

- Interface Homme Machine
- Classe Fonctionnelle de la barrière



CRÉATION DE LA CLASSE BARRIÈRE

- Séparations de l'interface de la partie fonctionnelle
- Meilleure lisibilité au niveau du code
- Facilité d'ajout de nouveau contenu

```h
#ifndef BARRIERE_H
#define BARRIERE_H

#include <QString>
#include "esclavemodbustcp.h"

class Barriere
{
public:


    Barriere();
    ~Barriere();
    void connexionWago(QString adresseIP);
    void deconnectionWago();
    void allumer();
    void eteindre();
    void monter();
    void descendre();
    QString lectureCapteurs();
    int getStatutBarriere();

private:

    EsclaveModbusTCP* connexionTCP = nullptr;
    int tabCapteurs[8] = {0,0,0,0,0,0,0,0};

    const QString labelEntrees[8] = {"FDCHaut", "FDCBas", "Boucle 1", "Boucle 2", "Ticket 4", "POM", "Clavier", "'KA1"};
    const unsigned short ADRESSE_CAPTEURS = 0x000;
    const unsigned short ADRESSE_MONTER = 0x0200;
    const unsigned short ADRESSE_DESCENDRE = 0x0201;
    const unsigned short ADRESSE_BALISE = 0x0202;

};
#endif //BARRIERE_H

```

CRÉATION DE LA GESTION D'EXCEPTION

- Existence des erreurs des fonctions Modbus TCP
- Visibilité des erreurs par l'utilisateur

CRÉATION DE LA CLASSE TCPEXCEPTION

```
64  class TcpException {
65  private:
66      int code;
67      QString message;
68
69  public:
70      TcpException(int code, QString message);
71      QString obtenirErreur(TcpException &e);
72      int getCode();
73      QString getMessage();
74  };
75
76  #endif // ESCLAVEMODBUSTCP_H
77
```

- Implémentation dans les fichier esclavemodbus.h et esclavemodbus.cpp
- 3 méthodes : 2 accesseurs, obtenirErreur

```
String TcpException::obtenirErreur(TcpException &e) {
    QString codeErreur = QString::number(e.getCode());
    QString erreur = "Numéro de l'erreur : " + codeErreur + " \nCause  : " + e.getMessage(
    return erreur;
```

Création d'une méthode pour lire les capteurs

```
90
91  QString Barriere::lectureCapteurs() {
92      unsigned short capteurs;
93      int resultat;
94      QString message;
95      QString lesCapteurs;
96       resultat = connexionTCP->lireNMots(ADRESSE_CAPTEURS, 1, &capteurs);
97      if (resultat == (TRANSMISSION_ERREUR | TRANSACTION_TIMEOUT_ERREUR)) {
98          if (resultat == TRANSMISSION_ERREUR) {
99              message = "Erreur lors de la transmission";
100         }
101         else {
102             message = "Erreur transaction timeout";
103         }
104         throw TcpException(resultat, message);
105     } else {
106         for (int i = 0; i < 8; i++) {
107             if ((capteurs & (1<<i)) != 0) {
108                 lesCapteurs = lesCapteurs + labelEntrees[i] + " - ";
109                 tabCapteurs[i] = 1;
110             }
111         }
112     }
113     return lesCapteurs;
114 }
```

Utilités

- Visuel dans les logs
- Pour les utilisateurs
- Vérification pour une meilleure sécurité

Lecture

- A la connexion : Test KA1
- Au lancement du changement d'état de la barrière : Test KA1 et capteurs de fin de course

AJOUT RAPIDE UTILITAIRE

```
38
39  void IHM::connecterWago()
40  {
41      ui->textEdit_log->append("Demande de connection au Wago");
42
43      QString adresseIP = ui->lineEdit_adresseIPWago->text();
44      laBarriere = new Barriere();
45      try {
46          laBarriere->connexionWago(adresseIP);
47
48          ui->pushButton_connecter->setEnabled(false);
49
50          ui->textEdit_log->append("Connection au Wago établie");
51
52          ui->pushButton_deconnecter->setEnabled(true);
53          ui->pushButton_allumerBalise->setEnabled(true);
54
55          laBarriere->lectureCapteurs();
56
57          if (laBarriere->getStatutBarriere() == 0)
58              ui->pushButton_descendreBarriere->setEnabled(true);
59          else
60              ui->pushButton_monterBarriere->setEnabled(true);
61      } catch (TcpException &e) {
62          ui->textEdit_log->append("Echec de la connection au Wago");
63          ui->textEdit_log->append(e.obtenirErreur(e));
64      }
65      try {
66      ui->textEdit_log->append(laBarriere->lectureCapteurs());
67      } catch (TcpException &e) {
68          ui->textEdit_log->append("Echec lors de la lecture des capteurs");
69          ui->textEdit_log->append(e.obtenirErreur(e));
70      }
71  }
```

AJOUT DE L'ACTUALISATION DES BOUTONS

- Grace à la méthode de la lecture des capteurs
- Meilleure sécurité
- Logiciel plus interactif

AJOUT D'UNE FONCTIONS RETOURNANT LE STATUT DE LA BARRIÈRE

Vérification avant la monter ou de descendre la barrière

UTILISATIONDE LA BALISE

- Allumage pendant les actions monter et descente
- Indication du placement de la barrière

CONCLUSION

- Ajout de la classe Barrière
- Ajout de la gestion d'exception
- Ajout d'une méthode pour lire les capteurs
- Ajout d'une méthode qui retourne l'état de la barrière
- Ajout de l'utilisation de la balise
- Ajout de l'actualisation des boutons