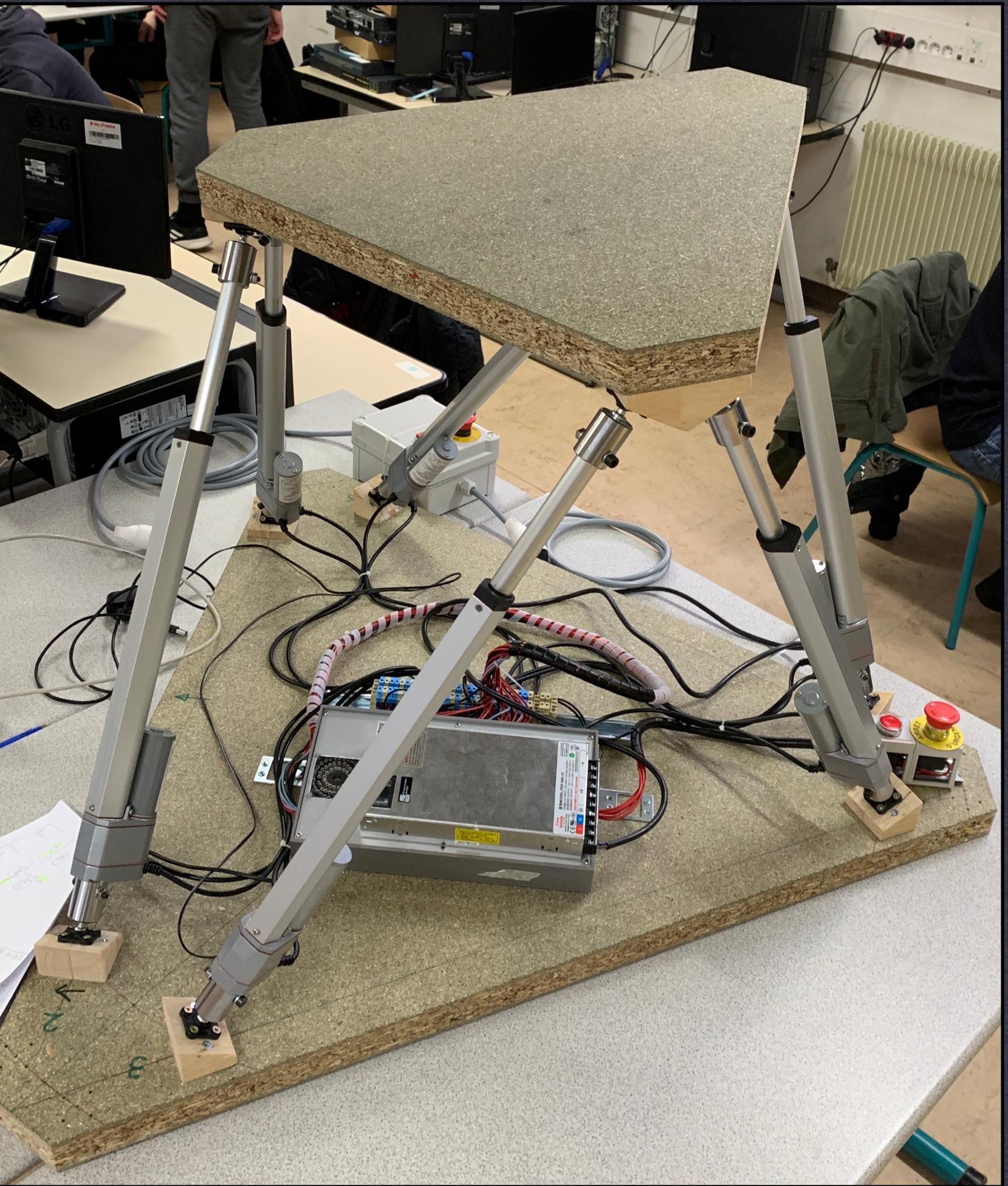




XPLANE ANÉMATIQUE D'UN SIMULATEUR HEXAPOD

Killian Lopes
Flavian Laxenaire
Yohann Rimbault
Aurélien Ferreira Novo



2021-2022

SOMMAIRE

- ◆ PRÉSENTATION
- ◆ CAHIER DES CHARGES
- ◆ RESSOURCES
- ◆ LES DIFFÉRENTES TÂCHES
- ◆ DIAGRAMMES
- ◆ PARTIE PERSONNELLE

PRÉSENTATION

- SDIS 77
- Simulateur d'intervention
- Centre de Formation de la Sécurité Civile

Gurcy-le-Châtel



CAHIER DES CHARGES

- Trois aspects (une vue de l'extérieur, la sonorisation et la sensation de mouvement)
- Hexapod sur 6 axes (R_x , R_y , R_z , T_x , T_y , T_z)
- Reproduire les mouvements de l'hexapod virtuellement/physiquement

RESSOURCES

- QT, Matlab, X-Plane, Arduino, Libs GLam/Qam



The Qt
Company



LES DIFFÉRENTES TÂCHES

Étudiant 1 : FlightSimMotionControl - Liaison UDP : X-Plane/Serveur MODBUS

Étudiant 2 : FlightSimMotionControl - Serveur MODBUS/Implémentation MGI

Étudiant 3 : Hexapod - Client MODBUS/MATLAB/ARDUINO -> Vérins physique

Étudiant 4 : FlightSimMockup - Modèle 3D -> Siège/Console instruments

DIAGRAMMES

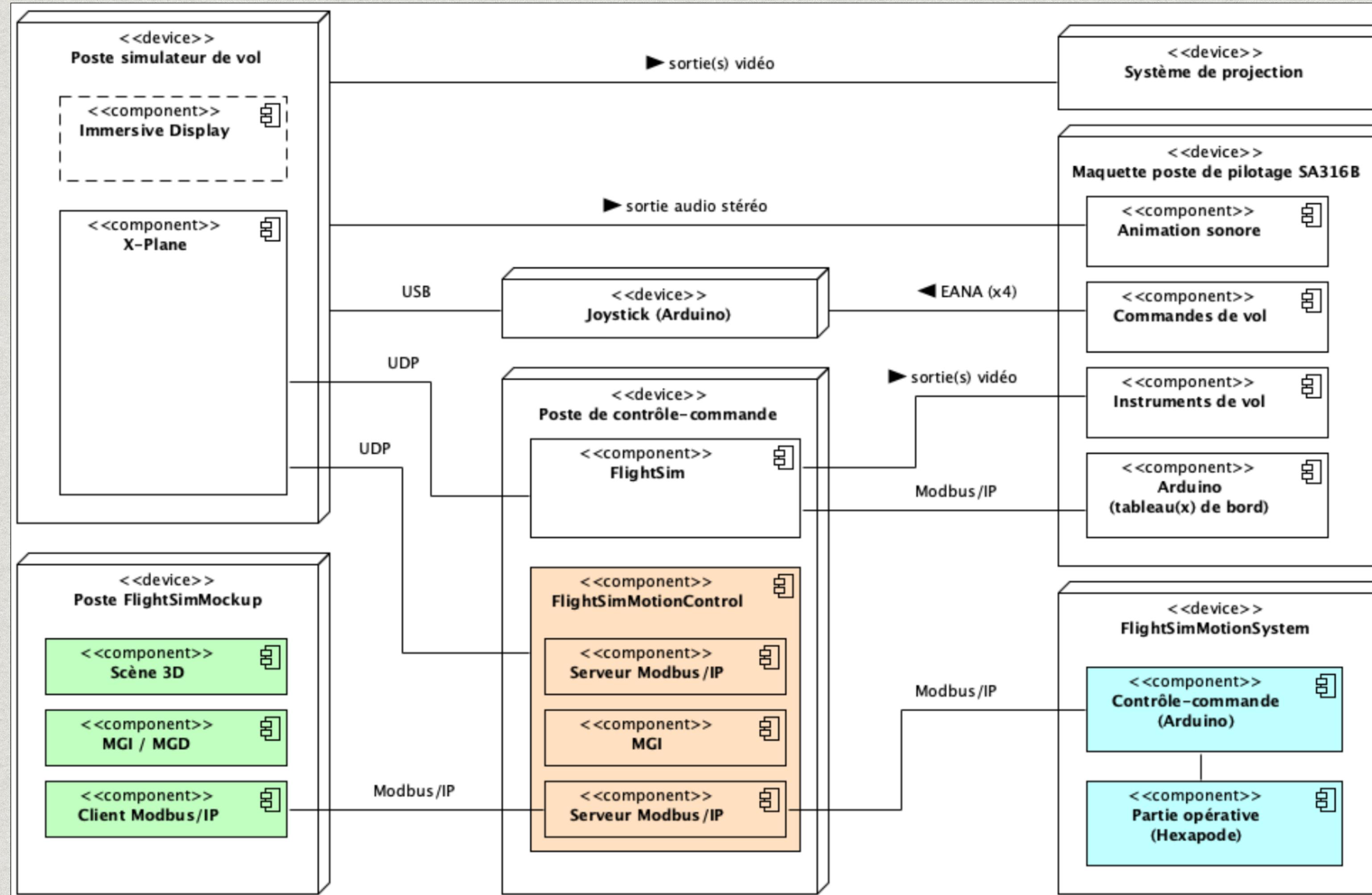


Diagramme de déploiement : PARTIE MISE EN MOUVEMENT

DIAGRAMMES

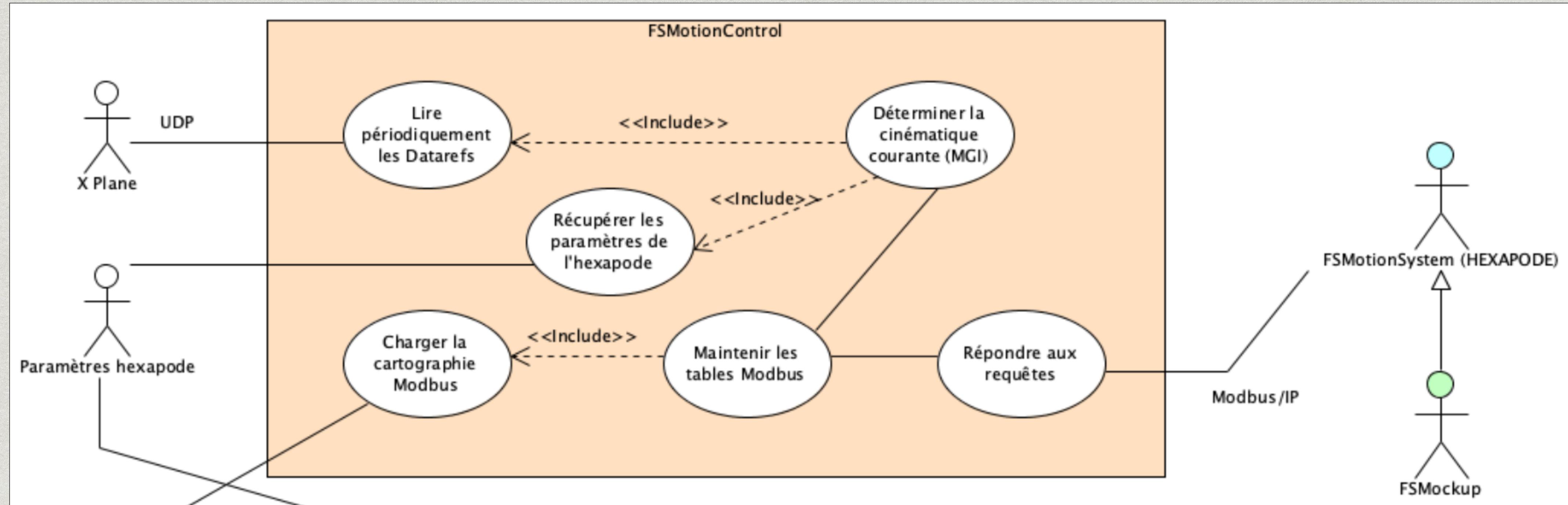


Diagramme de cas d'utilisation 1 : FlightSimMotionControl

DIAGRAMMES

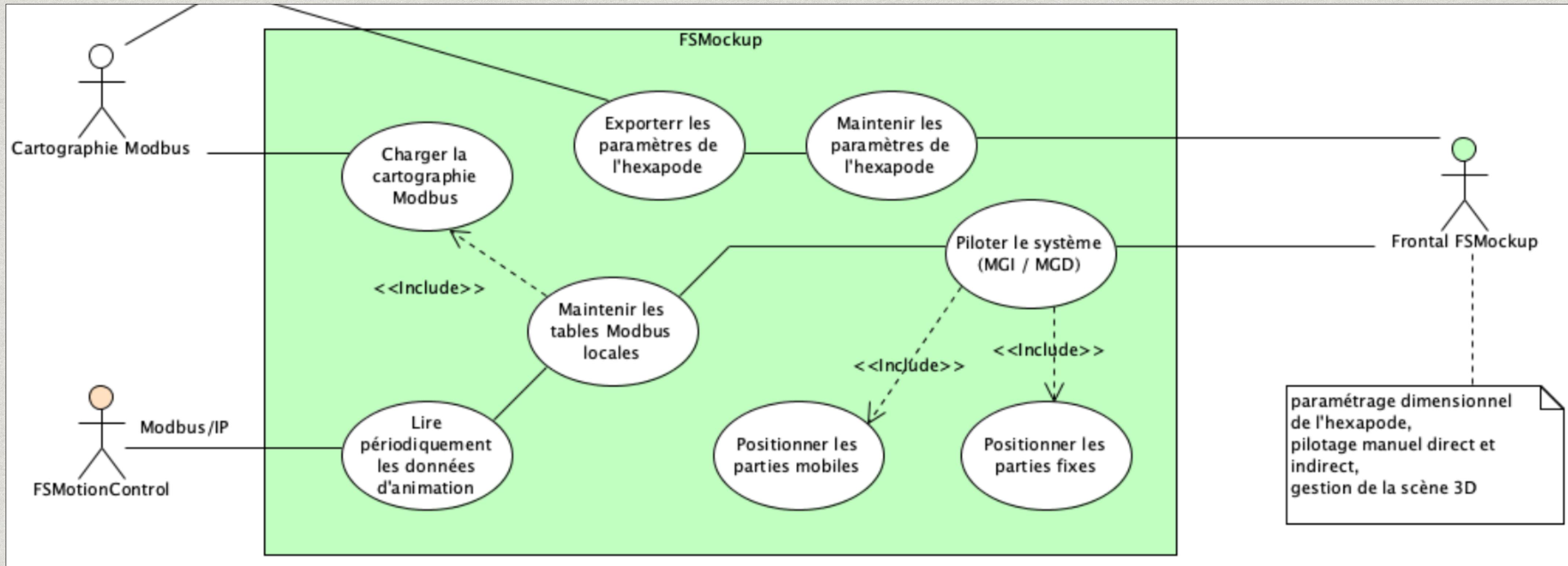


Diagramme de cas d'utilisation 2 : FlightSimMockup

PARTIE PERSONNELLE

- Serveur Modbus TCP / IP
- Implémenter le MGI (Modèles Géométriques Indirect)
- Fusionner avec la partie liaison UDP
- Mettre les angles localement dans le serveur

DIAGRAMMES DE CLASSE

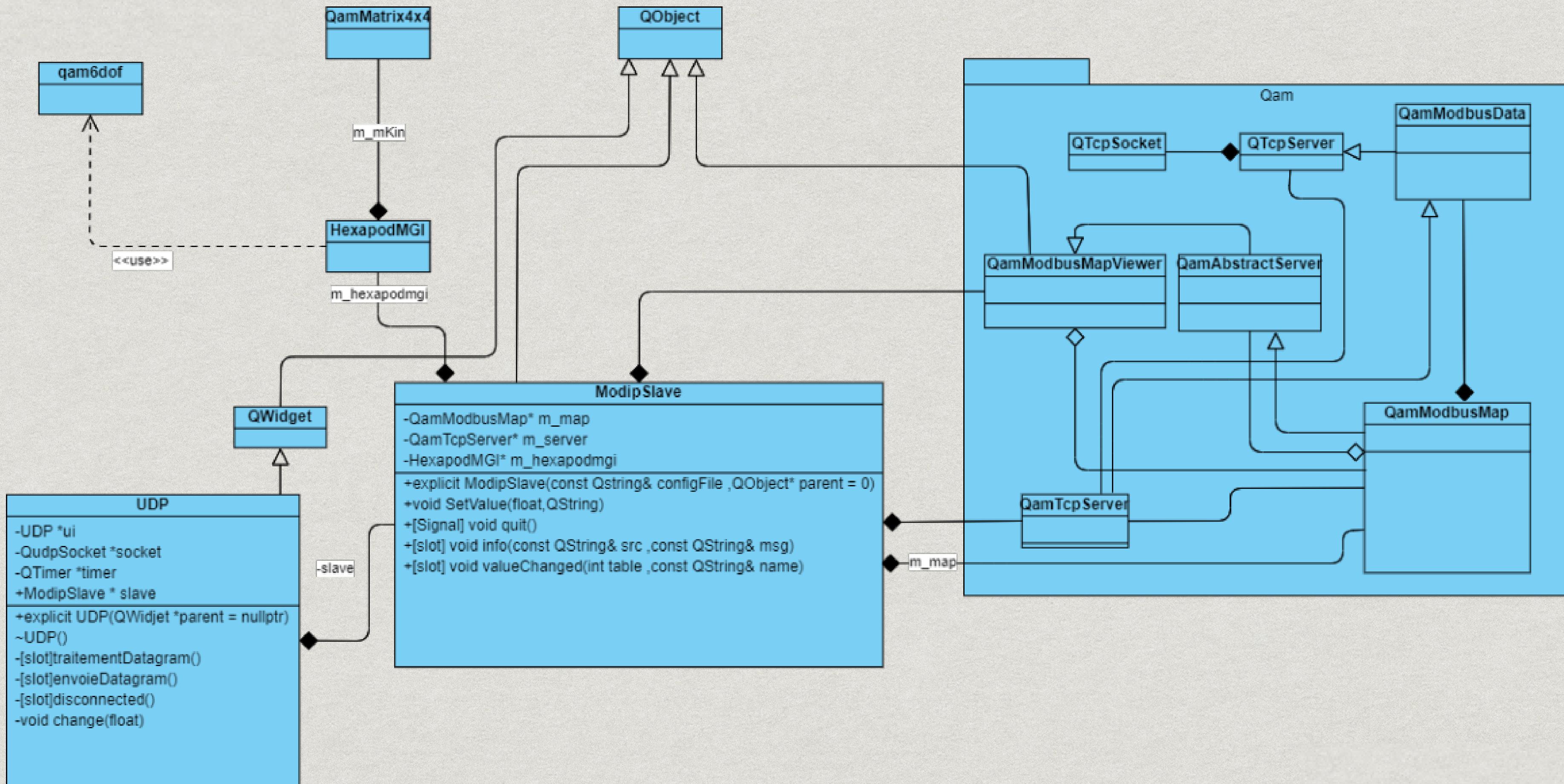
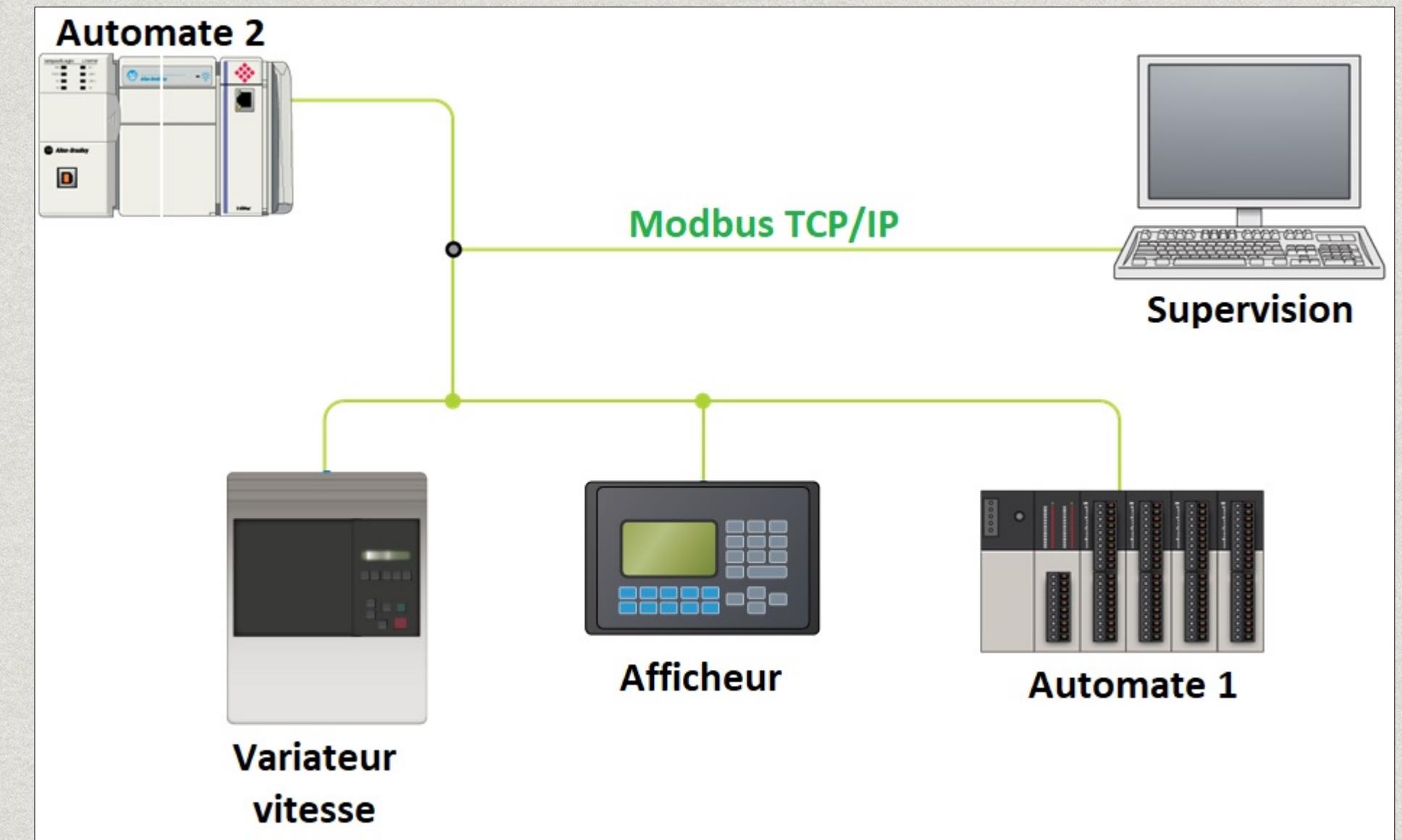


Diagramme de classes - FlightSimMotionControl

INTRODUCTION MODBUS

- Modbus protocole de communication
- Relation Maître / Esclave
- Modbus TCP/IP → RTU + ethernet
- Interactions simultané de plusieurs maîtres avec un seul esclave



INTRODUCTION MODBUS

- Coils : sorties binaires on/of
- Input bits : entrées binaires (lecture seule).
- Input registers : entrées analogiques (lecture seule).
- Holding registers : paramètres analogiques modifiables.

```
7 HOST;127.0.0.1
8 PORT;502
9 INFO;FLIGHTSIM HEXAPOD
10
11 # 4xxxx - holding registers
12
13 # hexapod current position
14
15 40001;2;FFFF; Tx; Translation 0x (longitudinal), en mm; Float; 0
16 40003;2;FFFF; Ty; Translation 0y (latéral), en mm; Float; 0
17 40005;2;FFFF; Tz; Translation 0z (altitude), en mm; Float; 200
18 40007;1;FFFF; Rx; Rotation 0x (roulis), en degrés; Int; 0
19 40009;1;FFFF; Ry; Rotation 0y (tangage), en degrés; Int; 0
20 40011;1;FFFF; Rz; Rotation 0z (lacet), en degrés; Int; 0
21
22 # hexapod actuator lengths
23
24 40101;1;FFFF; L1; Longueur vérin n°1, em mm ; Int; 0
25 40102;1;FFFF; L2; Longueur vérin n°2, em mm ; Int; 0
26 40103;1;FFFF; L3; Longueur vérin n°3, em mm ; Int; 0
27 40104;1;FFFF; L4; Longueur vérin n°4, em mm ; Int; 0
28 40105;1;FFFF; L5; Longueur vérin n°5, em mm ; Int; 0
29 40106;1;FFFF; L6; Longueur vérin n°6, em mm ; Int; 0
30
31 # hexapod template
32
33 40201;1;FFFF; baseRadius; Rayon ancrages base, em mm ; Int; 500
34 40202;1;FFFF; baseGap; 1/2 écart ancrages base, em mm ; Int; 86
35 40203;1;FFFF; topRadius; Rayon ancrages platine, em mm ; Int; 300
36 40204;1;FFFF; topGap; 1/2 écart ancrages platine, em mm ; Int; 35
37 40205;1;FFFF; minLen; longueur min. (vérin rentré), em mm ; Int; 560
38 40206;1;FFFF; maxLen; longueur max. (vérin sorti), em mm ; Int; 860
39 40207;1;FFFF; maxAngle; angle absolu max. Rx,Ry,Rz, en degrés ; Int; 15
40 40208;1;FFFF; maxTrans; déplacement abs. max. Tx,Ty, en mm ; Int; 40
41 40209;1;FFFF; maxSpeed; vitesse linéaire max., em mm/s ; Int; 30
```

Fichier configuration pour QamModbusMap - fsmockup.csv

SERVEUR MODBUS

```
int main(int argc, char *argv[])
{
    QApplication app(argc, argv) ;
    ModipSlave* slave = new ModipSlave(":/fsmockup.csv", &app);
    UDP w;
    w.show();
```

- Création d'un esclave
- Fichier .csv en argument

```
ModipSlave::ModipSlave(const QString& configFile, QObject* parent )
: QObject(parent)
{
    // cartographie Modbus

    m_map     = new QamModbusMap( QamModbusMap::ServerMode, this ) ;

    m_map->setVerbose( false ) ;

    connect( m_map, SIGNAL(info(QString,QString)), this, SLOT(info(QString,QString)) ) ;
    connect( m_map, SIGNAL(valueChanged(int,QString)), this, SLOT(valueChanged(int,QString)) ) ;

    m_map->loadMap( configFile ) ;

    // serveur TCP

    m_server = new QamTcpServer( m_map, this ) ;
    m_hexapodmgi = new HexapodMGI();
    m_server->start( m_map->port() ) ;
```

- Création d'une configuration
- Chargement fichier de configuration
- Création du serveur
- Lancement du serveur sur le port spécifié



```
QamMatrix6x1 ModipSlave::MGI(float Rx, float Ry, float Rz)
{
    // param = 6-DOF [ Tx,Ty,Tz, Rx,Ry,Rz ] unités mm et degrés

    QamMatrix6x1 kin ;
    // QamModbusMap::PrimaryTable table = QamModbusMap::HoldingRegister ;

    // lecture des registres
    // kin(0) = m_map->value(table, "Tx" ).toFloat() ;
    // kin(1) = m_map->value(table, "Ty" ).toFloat() ;
    // kin(2) = m_map->value(table, "Tz" ).toFloat() ;
    // kin(3) = m_map->value(table, "Rx" ).toFloat() ;
    // kin(4) = m_map->value(table, "Ry" ).toFloat() ;
    // kin(5) = m_map->value(table, "Rz" ).toFloat() ;

    kin(0) = 0;
    kin(1) = 0;
    kin(2) = 0;
    kin(3) = Rx;
    kin(4) = Ry;
    kin(5) = Rz;

    m_hexapodmgi->setMGI(kin);
    //qDebug() << kin;
    return m_hexapodmgi->actuatorLen();
}
```

- Écrit localement les valeurs
- Problème d'écriture de décimale
 - valeur * 100 à l'écriture
 - valeur / 100 à la lecture

- Modèle Géométrique Indirect

Conversion angles (Rx, Ry, Rz)
→ matrice de 6 longueurs de vérins

```
ui->labpitch->setText(ps); //affichage du tangage sur l'application
slave->SetValue( QString::number(round(ps.toFloat()*100)*-1) ,QString("Ry"));

ui->labheading->setText(hs); //affichage du lacet sur l'application
slave->SetValue( QString::number(round(hs.toFloat()*100)) ,QString("Rz"));

ui->labroll->setText(rs); //affichage du roulis sur l'application
slave->SetValue( QString::number(round(rs.toFloat()*100)) ,QString("Rx"));

qDebug() << "=====";
QamMatrix6x1 LengthVerin = slave->MGI(rs.toFloat(),ps.toFloat(),hs.toFloat()); //recuperation longeur verins via le MGI

//qDebug() << round(LengthVerin(0));
slave->SetValue(QString::number(round(LengthVerin(0))),QString("L1"));
slave->SetValue(QString::number(round(LengthVerin(1))),QString("L2"));
slave->SetValue(QString::number(round(LengthVerin(2))),QString("L3"));
slave->SetValue(QString::number(round(LengthVerin(3))),QString("L4"));
slave->SetValue(QString::number(round(LengthVerin(4))),QString("L5"));
slave->SetValue(QString::number(round(LengthVerin(5))),QString("L6"));

|
ui->LL1->setText(QString::number(LengthVerin(0))); //affichage de la longeur des verins
ui->LL2->setText(QString::number(LengthVerin(1)));
ui->LL3->setText(QString::number(LengthVerin(2)));
ui->LL4->setText(QString::number(LengthVerin(3)));
ui->LL5->setText(QString::number(LengthVerin(4)));
ui->LL6->setText(QString::number(LengthVerin(5)));
```

ÉTAT DE CONNEXION UDP

- UDP mode non connecté, pas de retour client
- Problématique → état de connexion ?

```
UDP::UDP(QWidget *parent)
    : QWidget(parent)
    , ui(new Ui::UDP)
{
    ui->setupUi(this);
    socket = new QUdpSocket(this);
    socket->bind(QHostAddress("192.168.0.103"), 49000);

    envoieDatagram();
    timer = new QTimer();
    timer->start(100);

    QTimer::connect(timer, SIGNAL(timeout()), this, SLOT(disconnected()));

    connect(socket, SIGNAL(readyRead()), this, SLOT(traitementDatagram()), Qt::QueuedConnection);
}
```

```
void UDP::traitementDatagram()
{
    timer->start(100);
    ui->statconnection->setText("<font color='lime'>CONNECTED</font>");
}
```

état de connection
DISCONNECTED

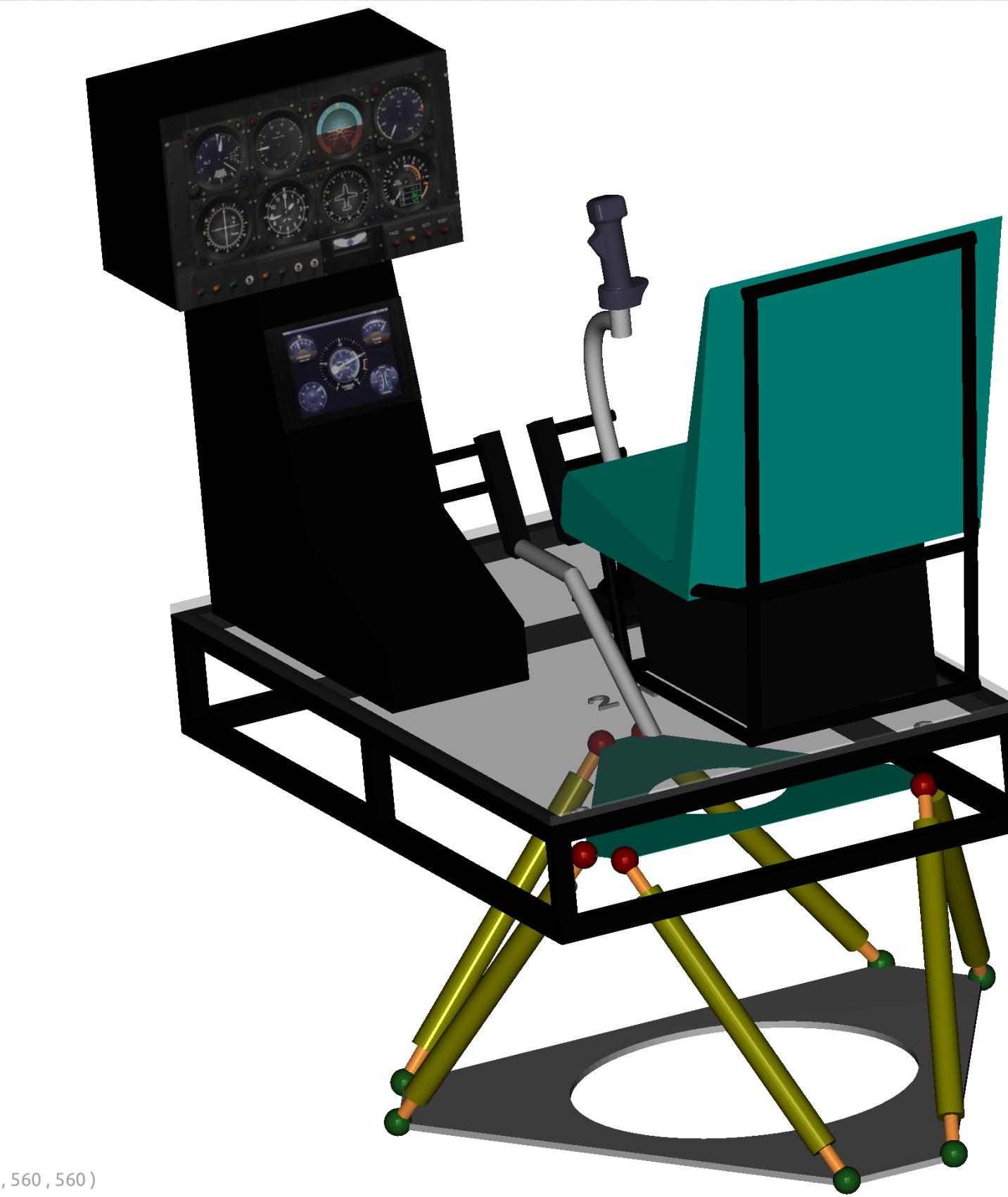
- Utilisation d'un minuteur (QTimer)
- Reset du minuteur si SIGNAL : readyRead
- Sinon appel d'une fonction : disconnected

ÉTAT D'AVANCEMENT

						état de connection DISCONNECTED
roulis / Rx TextLabel	tangage / Ry TextLabel	lacet / Rz TextLabel				
verin n°1 TextLabel	verin n°2 TextLabel	verin n°3 TextLabel	verin n°4 TextLabel	verin n°5 TextLabel	verin n°6 TextLabel	
<hr/> quitter						

Interface Homme / Machine - FlightSimMotionControl

ÉTAT D'AVANCEMENT



server:

sampling: 20 ms

- ✓ Serveur Modbus TCP / IP
- ✓ Fusion avec la partie UDP
- ✓ MG
- ✗ Documentation

MERCI DE VOTRE ATTENTION