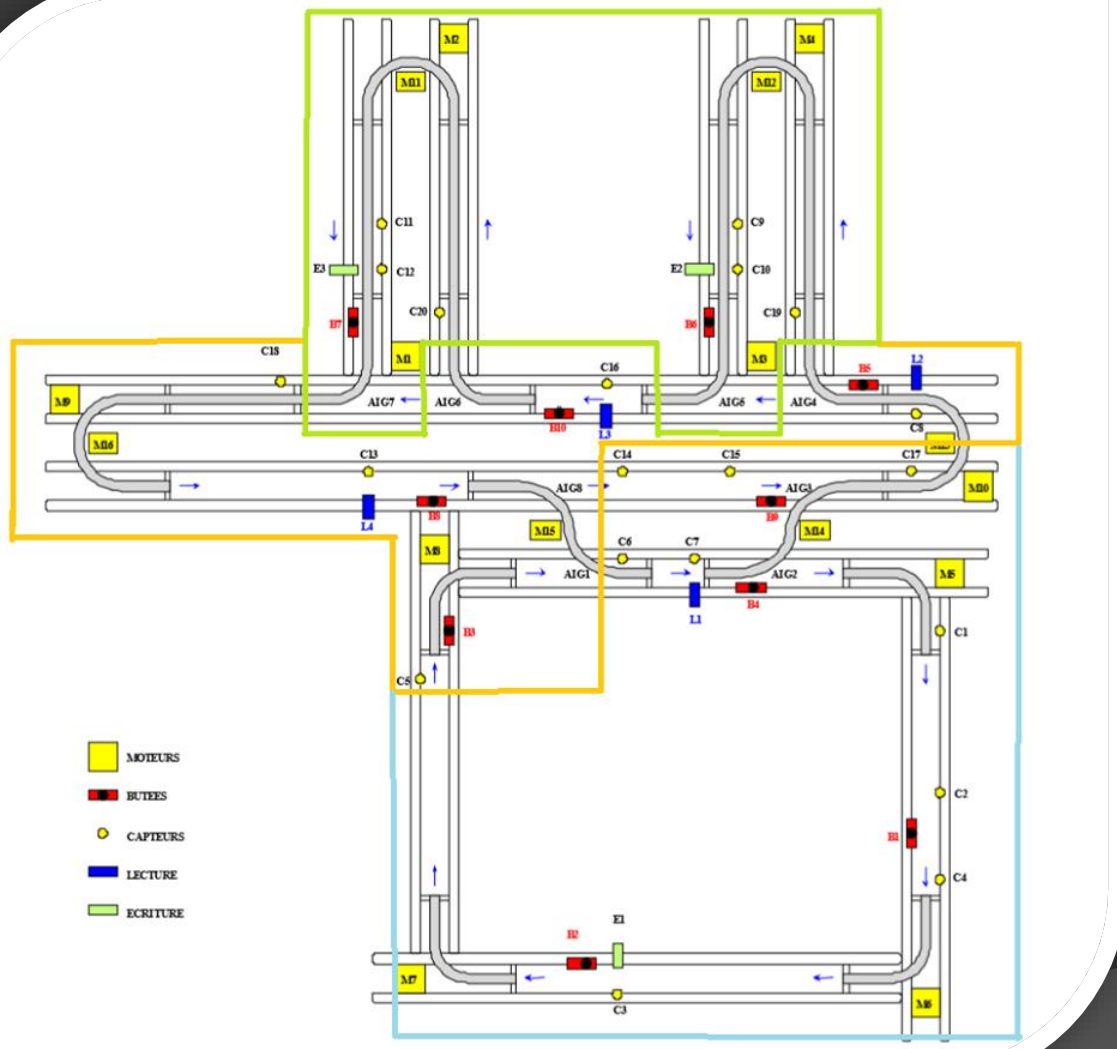
ROBOTISATION & SUPERVISION DE LA LIGNE TRANSITIQUE

Alex Le Saux, Ali Bekkouche, Baptiste Marhadour

Sommaire

La maquette



Station « épis »

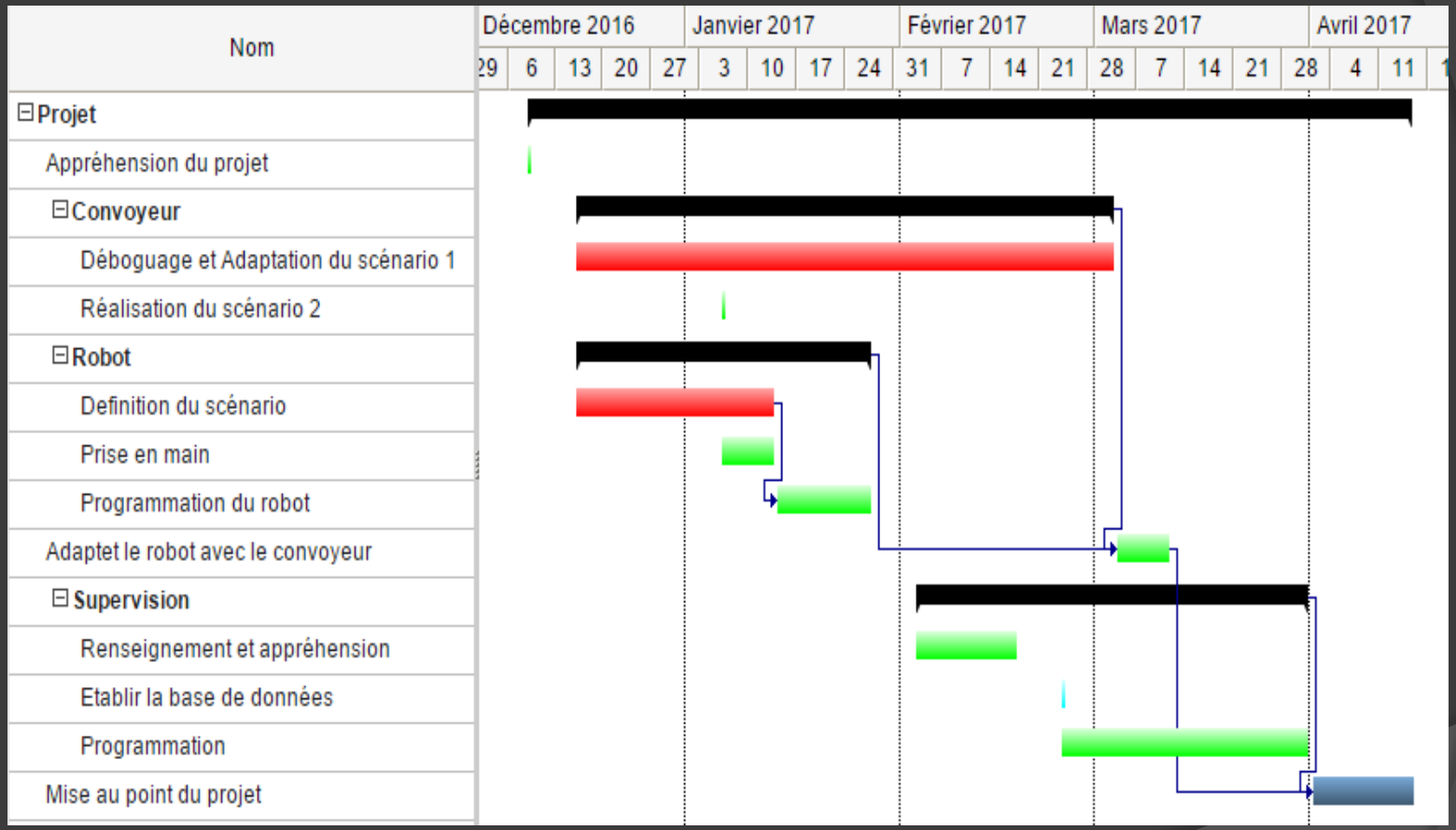
Station « hippodrome »

Station « magasin »

Le cahier des charges

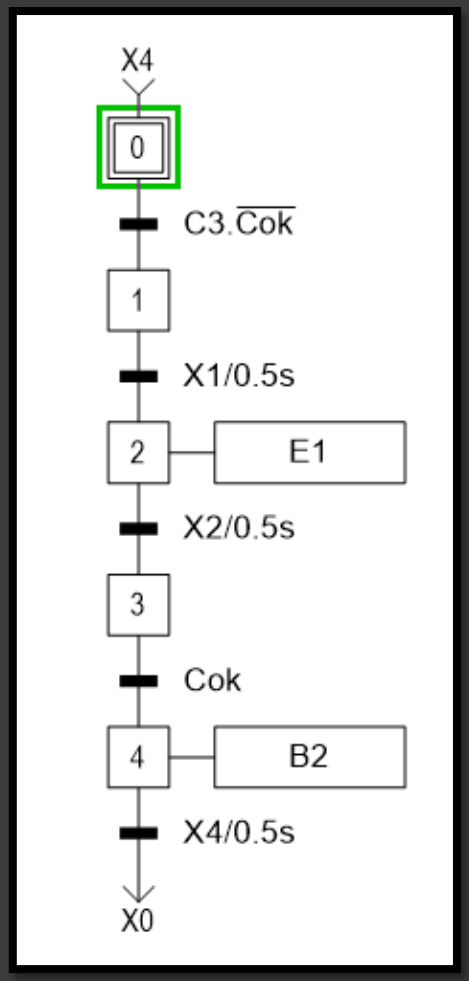
- ***Programmation des automates***
- ***Programmation du robot***
- ***Supervision***

Gestion de projet



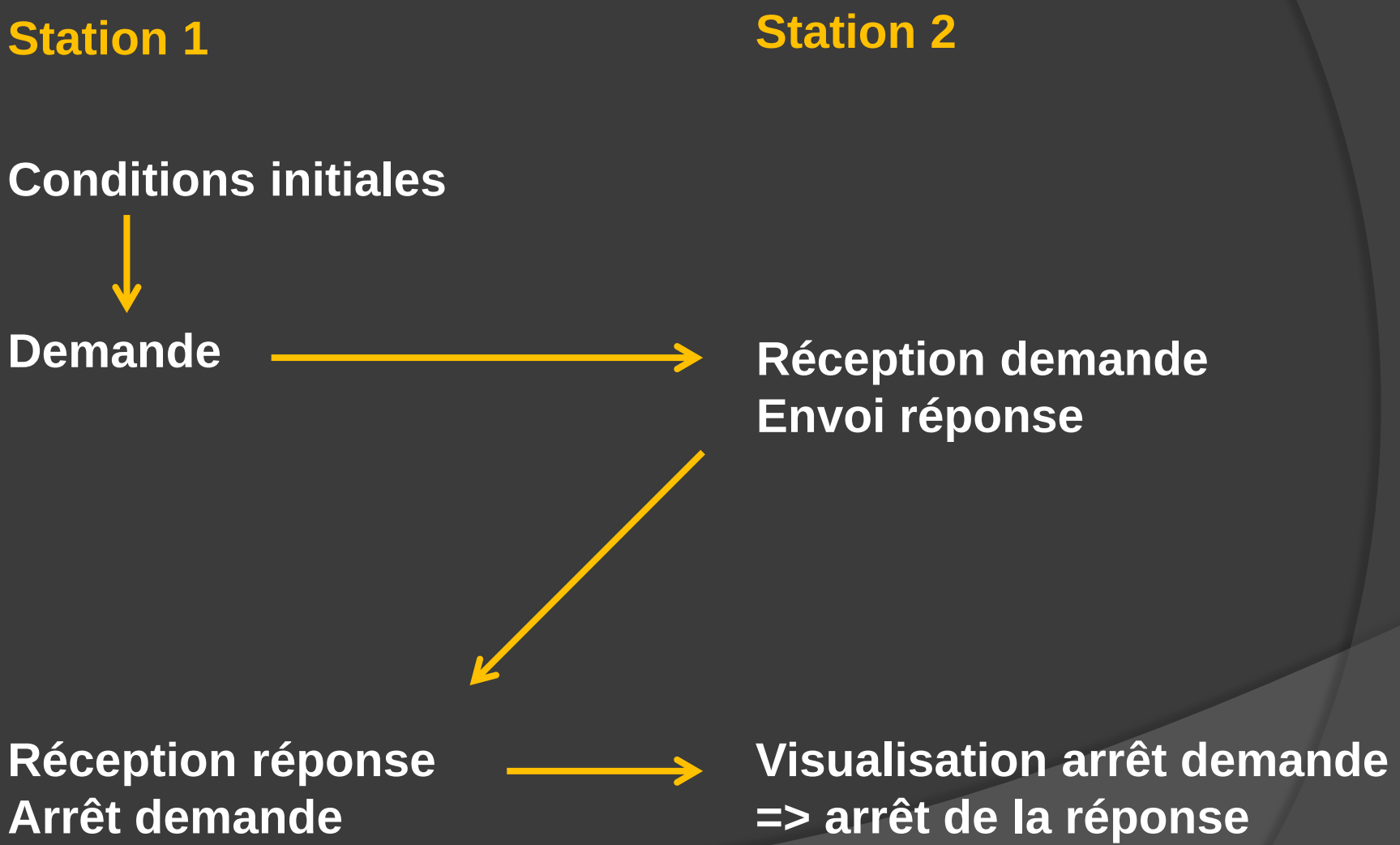
Programmation automate

Du GRAFCET au LIST

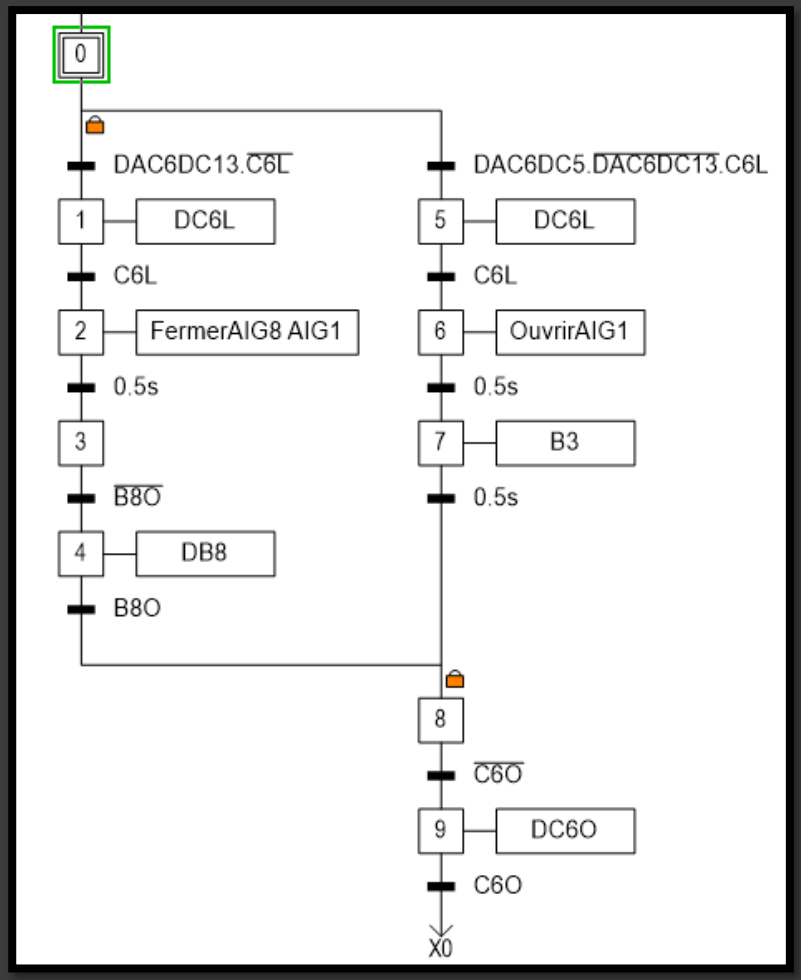


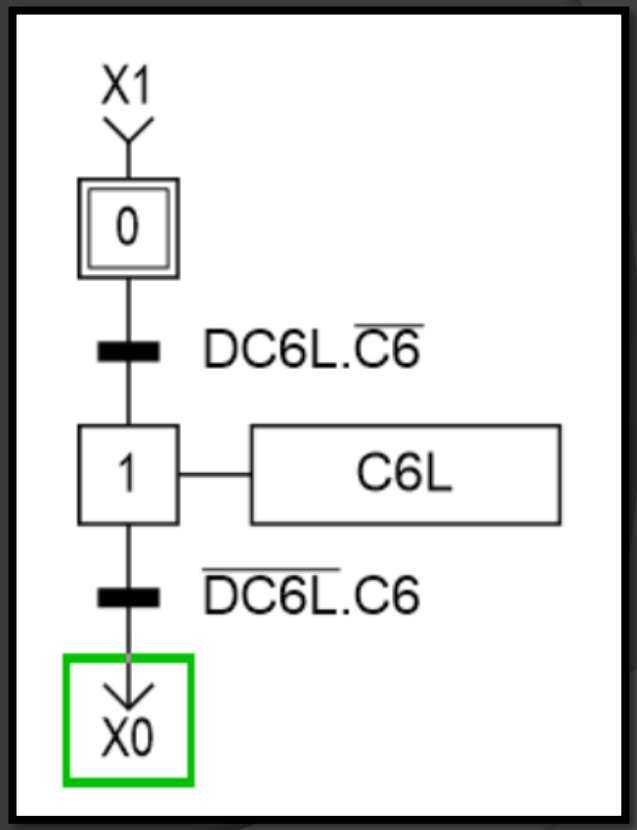

```
U     M     10.6
U     "C3"                    E0.3
UN    "Cok"                   M131.5
R     M     10.6
S     M     10.3

U     M     10.3
U     T     12
R     M     10.3
S     M     10.4

U     M     10.4
U     T     13
R     M     10.4
S     M     10.5

U     M     10.5
U     "Cok"                   M131.5
R     M     10.5
S     M     10.7

U     M     10.7
U     T     14
R     M     10.7
S     M     10.6

BE
```

Sémaphores


Station 1
Station 2
Conditions initiales
Demande
Réception demande
Envoi réponse
Réception réponse
Arrêt demande
Visualisation arrêt demande
=> arrêt de la réponse

Sémaphores

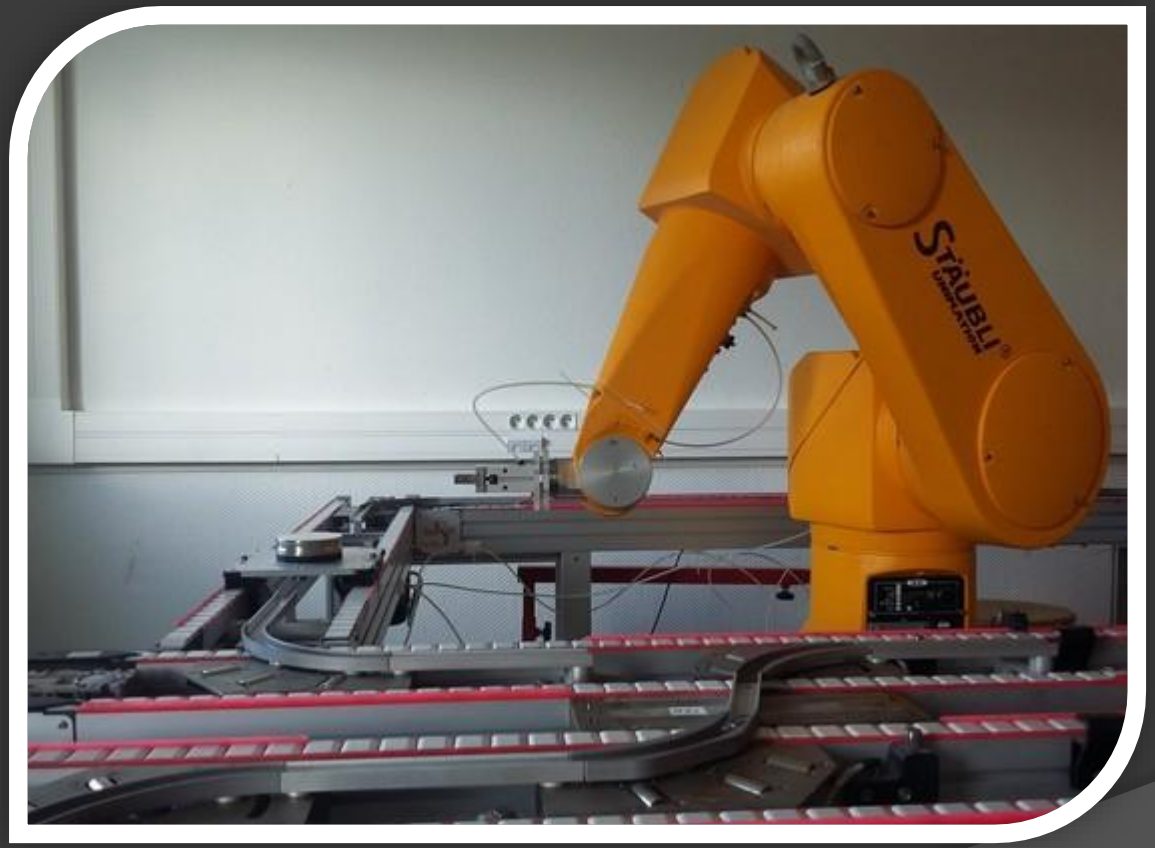

0
DAC6DC13.C6L
1
DC6L
C6L
2
FermerAIG8 AIG1
0.5s
3
B8O
4
DB8
B8O
DAC6DC5.DAC6DC13.C6L
5
DC6L
C6L
6
OuvrirAIG1
0.5s
7
B3
0.5s
8
C6O
9
DC6O
C6O
X0


X1
0
DC6L.C6
1
C6L
DC6L.C6
X0

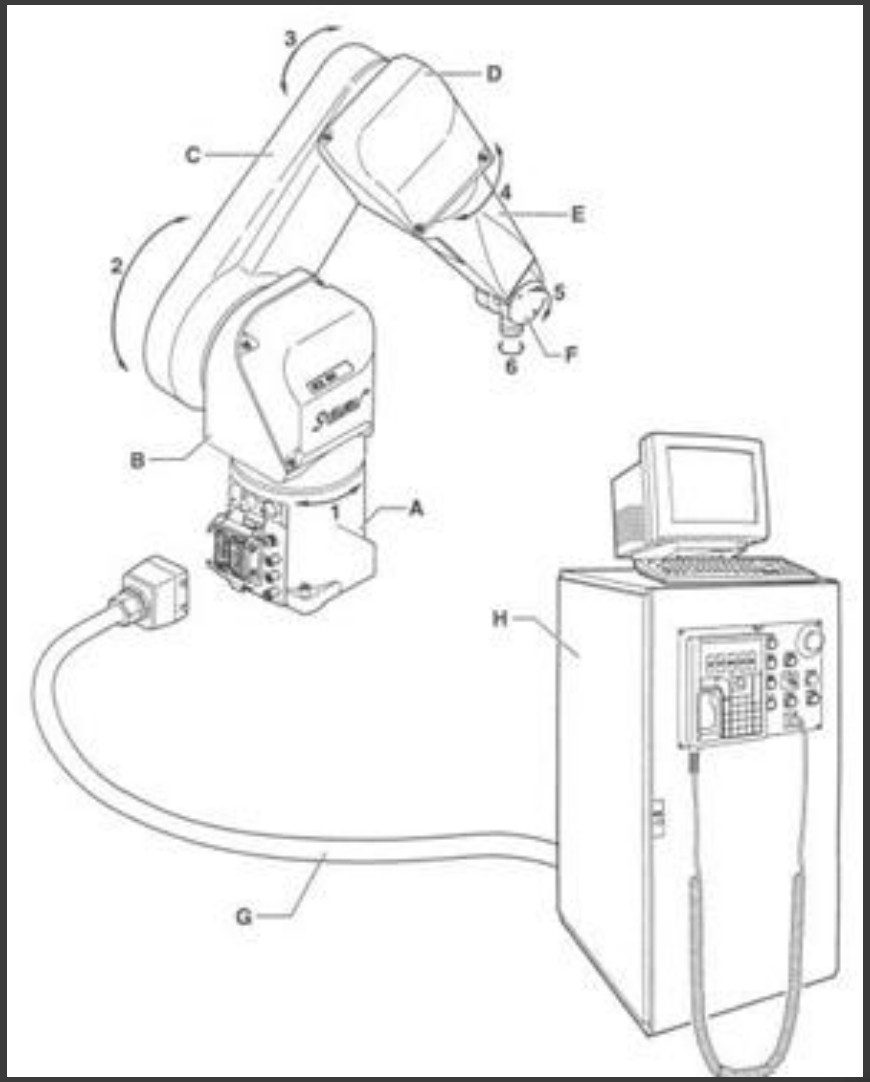
Gestion des défauts

- Trois défauts programmés, visualisés par un voyant. Démonstration.

Le robot



Présentation du robot



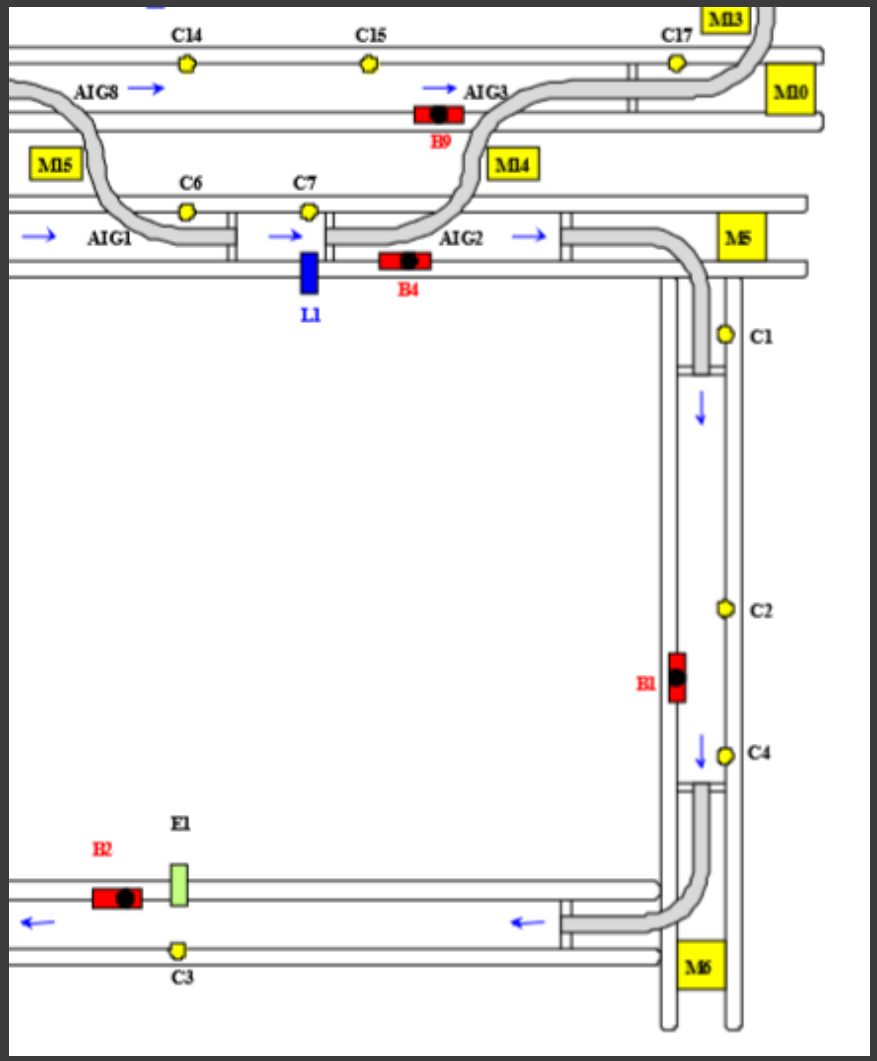
Stäubli RX 90

6 degrés de liberté

- A : Pied
- B : Epaule
- C : Bras
- D : Coude
- E : Avant-bras
- F : Poignet
- G : Câble de liaison
- H : Baie de commande

Simulation du chargement et déchargement de palettes

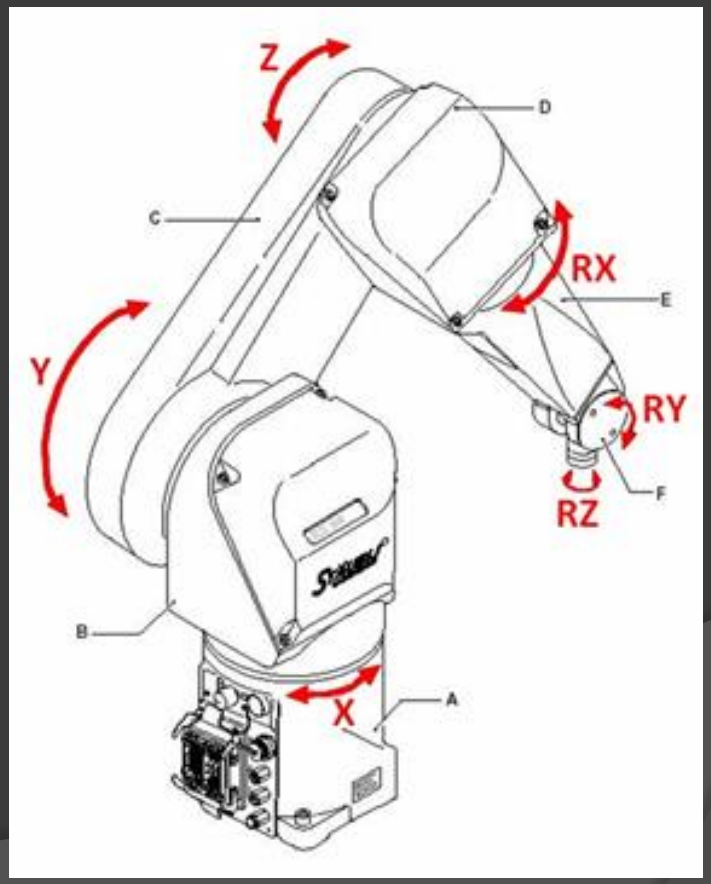
Présentation du robot



Simulation du chargement et déchargement de palettes

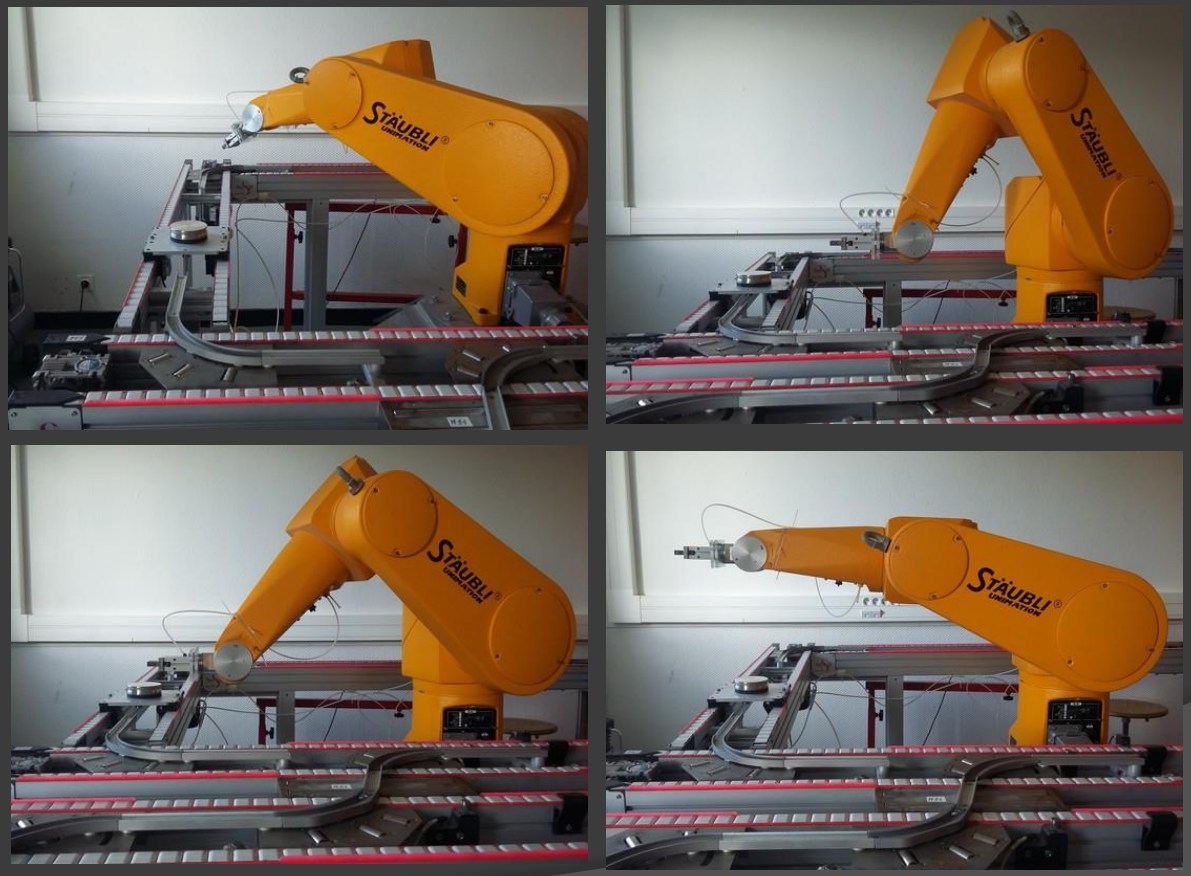
Programmation du robot

- Sauvegarde de positions



Programmation du robot

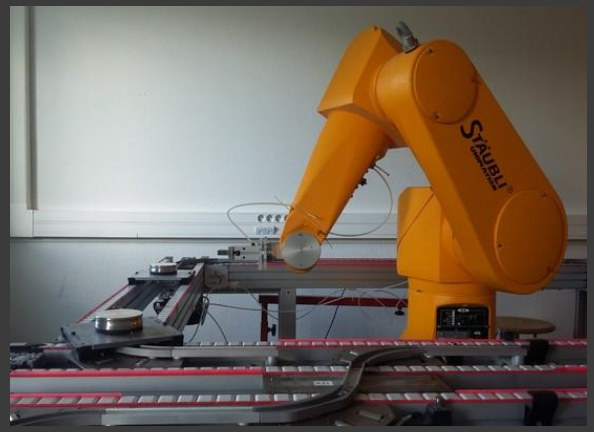
- Sauvegarde de positions



Programmation du robot

- Programme

Langage V+



```
.PROGRAM decharger.pg()
        MOVE #bh
        BREAK
        MOVE #bb
        BREAK
        CLOSE I
        BREAK
        SIGNAL 1
        DELAY 1
        SIGNAL -1
        MOVE #bd
        BREAK
        OPEN I
        BREAK
        MOVE #bh
        BREAK
```

Programmation du robot

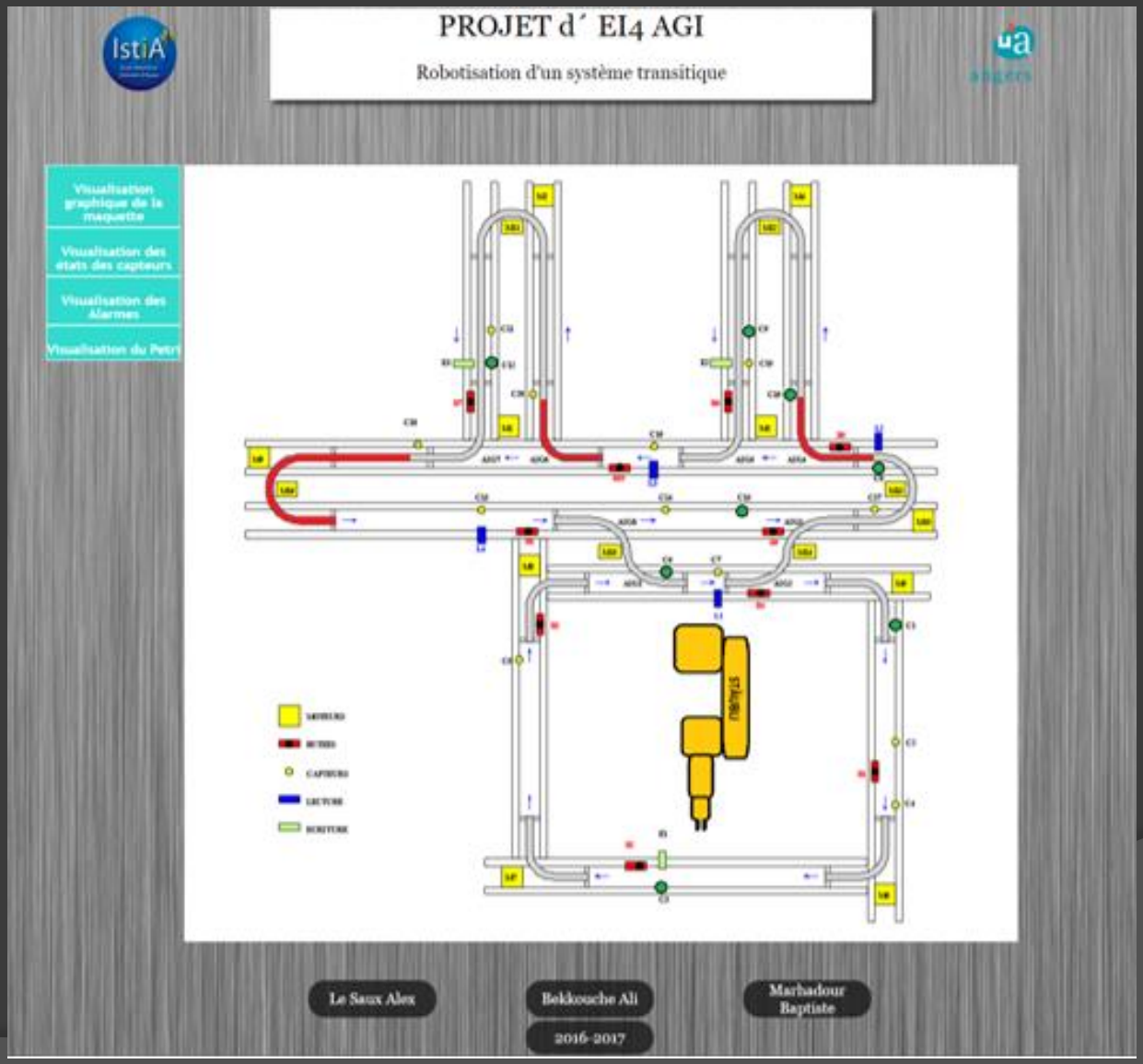
- Programme principal

Structure:

Initialisation (mise en position de repos, initialisation des variables)
Faire en boucle

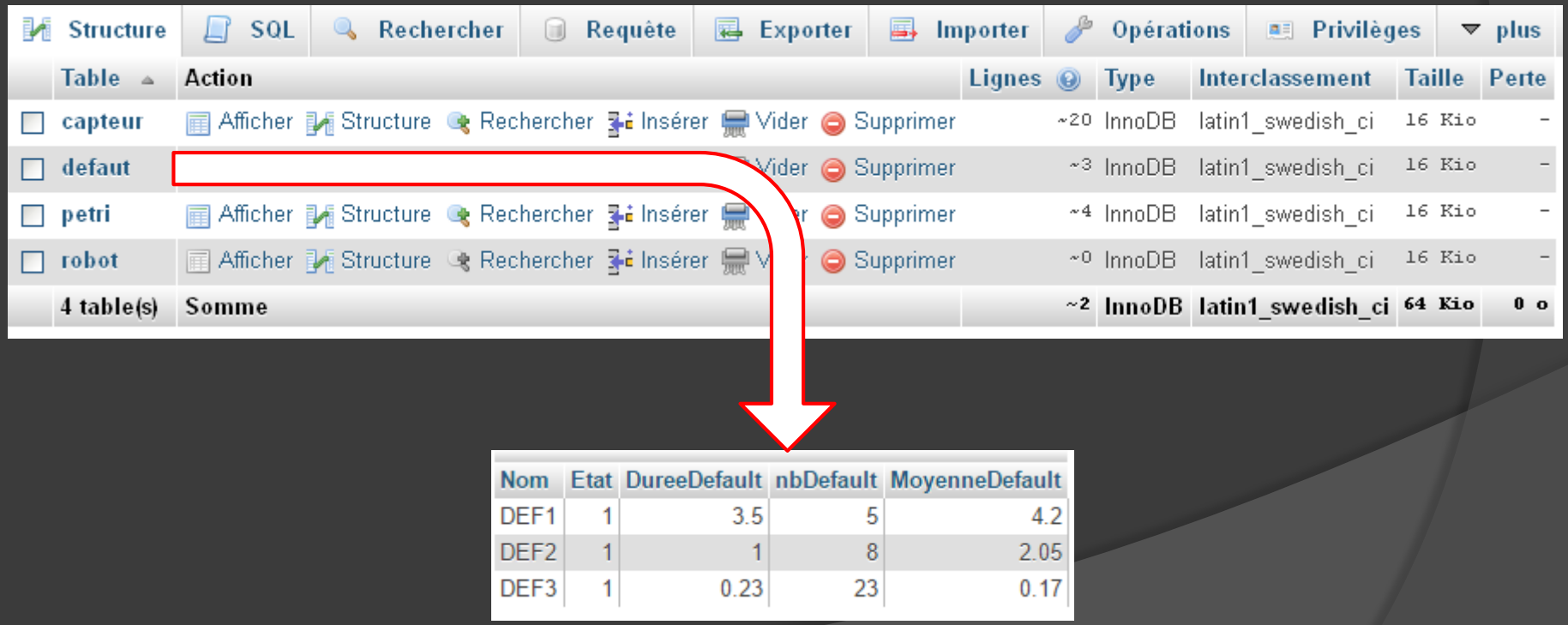
- Si demande de chargement, vérifier position du robot
  - Si robot hors de la zone de chargement, passer par la position de repos
  - Exécuter le programme de chargement,
  - Mettre à jour les variables d'état
  - Après une tempo de 1,5s, si plus de demande, retourner au repos
  - Mettre à jour les variable d'état
- Si demande de déchargement, même procédure

La supervision



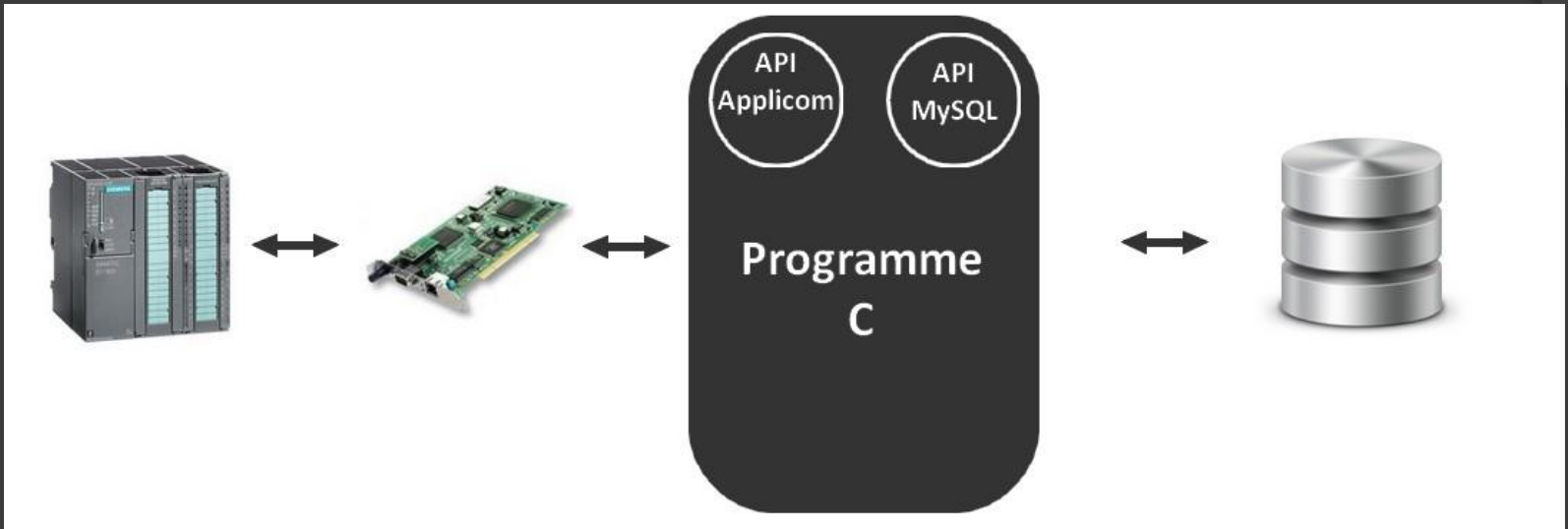
La base de données

- Utilisation de Wamp
  - phpMyAdmin



Nom	Etat	DureeDefault	nbDefault	MoyenneDefault
DEF1	1	3.5	5	4.2
DEF2	1	1	8	2.05
DEF3	1	0.23	23	0.17

Le programme C



Le programme C

#intégration des librairies et APIs

int main () {
déclaration des variables;

/***Initialisation de la BDD***/
Ouverture de la connexion à la BDD {

- Mise à zéro des états des capteurs et des défauts, des compteurs de files du réseau de Petri, et de la position du robot
- Récupération des données relatives aux défauts (historique)
- Affecter ces données aux variables du programme

}

/*** Boucle de traitement ***/
Si (connexion applicom établie) {
Faire en boucle
{

- Lecture des données de la station magasin
- Lecture des données de la station hippodrome
- Lecture des données de la station épis
- Mise à jour de la base de donnée (changement d'état des capteurs, compteurs des files d'attente du réseau de Petri, états des défauts, calcul des durées et temps moyens des défauts)

} //Fin boucle de traitement
} //Fin du main + fonctions additionnelles

Le programme C

- Utilisation des fonctions de l'API Applicom
  - **readpackibit**(&nchan,&neq,&nb,&adr,tabl,&status)
  - **transwordbit**(&nb,tabl,tablbit,&status)

```
etat_capteurs_actuel[14]=tablbit[8+4];    //lecture état C14 - E1.4
```

Le programme C

- Utilisation des fonctions de l'API MySQL

```
char requete[512];
MYSQL mysql;
mysql_init(&mysql);
mysql_options(&mysql,MYSQL_READ_DEFAULT_GROUP,"option");
if(mysql_real_connect(&mysql,"127.0.0.1","root","","projetaba",0,NULL,0))
{
    mysql_query(&mysql,"UPDATE `capteur` SET `Etat` =0");
    mysql_query(&mysql,"UPDATE `defaut` SET `Etat` =0");
    mysql_query(&mysql,"UPDATE `petri` SET `Valeur` =0");
    mysql_query(&mysql,"UPDATE `robot` SET `position` =1");

    sprintf(requete,"UPDATE `capteur` SET `Etat` =%d
WHERE          `Nom`='C%d'",etat_capteurs_actuel[1], 1);
    mysql_query(&mysql,requete);
 }
```

L’interface web

- Connection à la base de données

```php
<?php
    // Connexion à la base de données
    try {
        $bdd = new PDO('mysql:host=localhost;dbname=projetABA', 'root', '');
    }
    catch(Exception $e) {
        die('Erreur : '.$e->getMessage());
    }
?>
```

- Requête SQL préparée

```php
$reqCapteurs = $bdd->query('SELECT * FROM capteur');
```

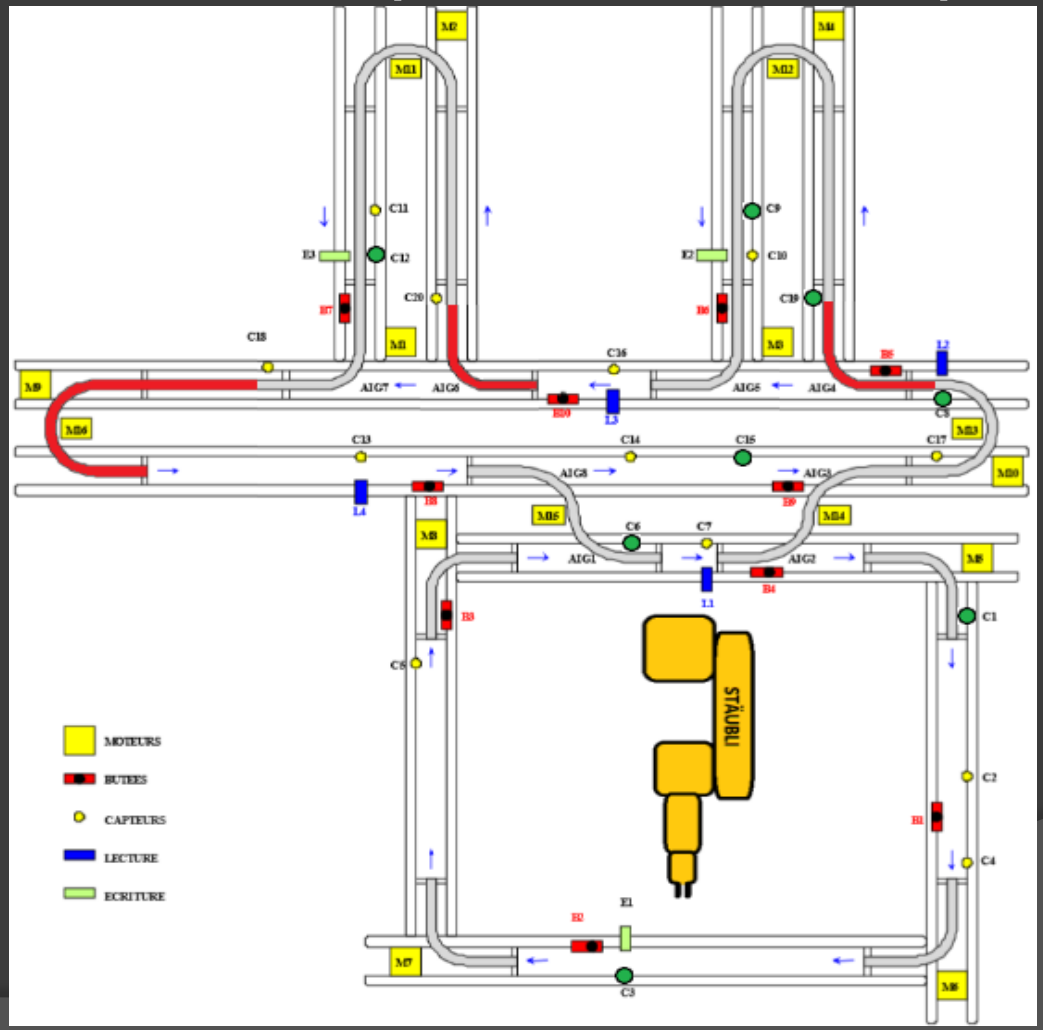
L’interface web

- Placement des capteurs sur la supervision

```php
$i_C = 0;       // indice pour les capteurs
// Tableau avec les valeur gauche et haute des capteurs
$Pos_image_left_C = [837,837,559,837,391,565,626,817,663,663,358,358,345,564,655,550,812,272,712,408];
$Pos_image_top_C =  [540,678,851,753,581,477,477,353,191,228,191,228,404,404,404,328,403,328,266,266];

while($donneeCapteurs = $reqCapteurs->fetch())  // boucle qui parcours la table tant qu'il y a des élements
{
    $LEFT = $Pos_image_left_C[$i_C];
    $TOP = $Pos_image_top_C[$i_C];
    if($donneeCapteurs["Etat"]=="1") //vérifie l'etat du capteur
        echo '<DIV style="position:absolute; left:'.$LEFT.'px; top:'.$TOP.'px;  z-index:2">
        <IMG src="Capteur.png"> </DIV>';    //Affichage d'un point vert pour le capteur actif
    $i_C+=1;
}
```

L’interface web

- Placement des capteurs sur la supervision

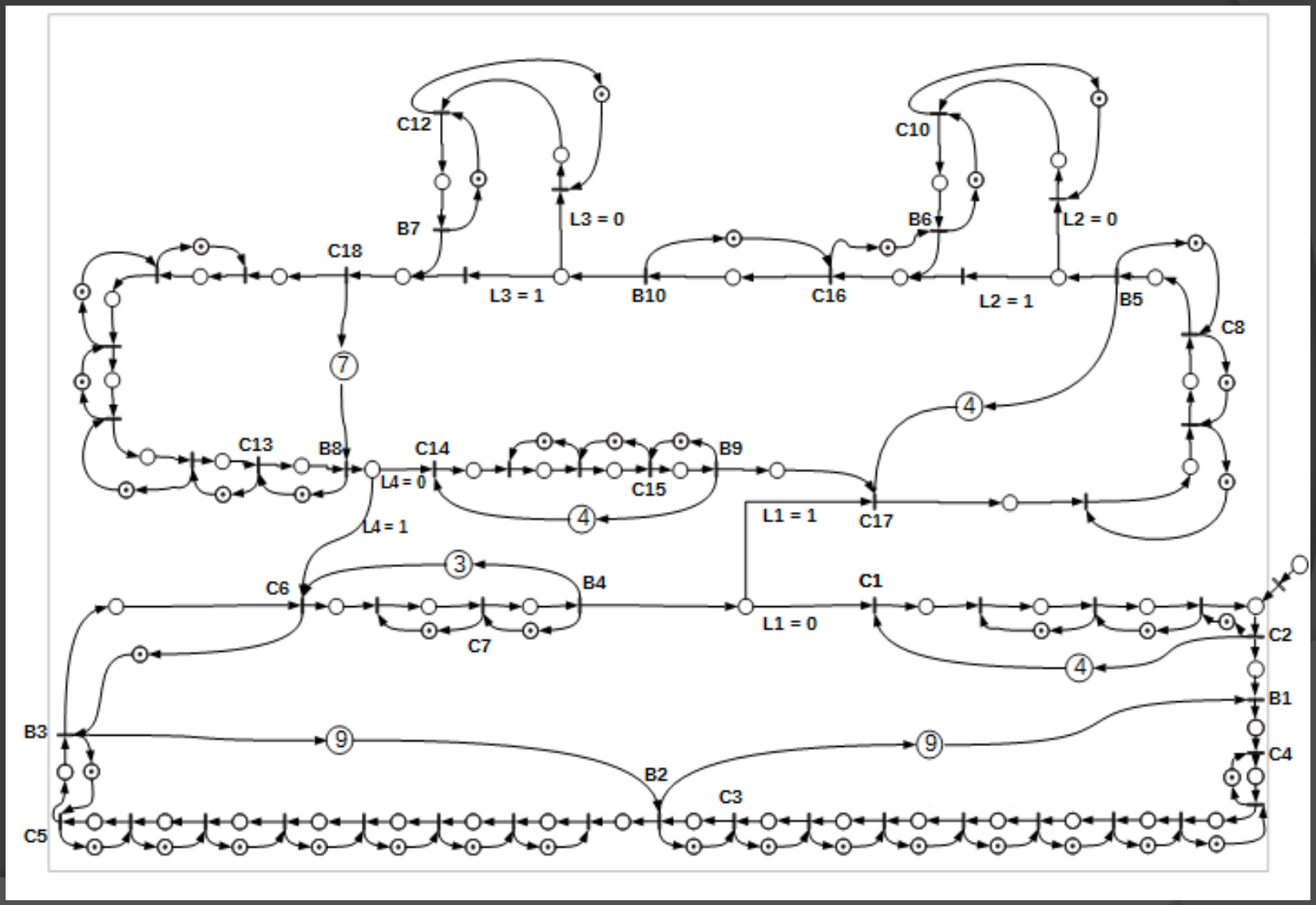


L’interface web

- Rafraichissement automatique de la supervision

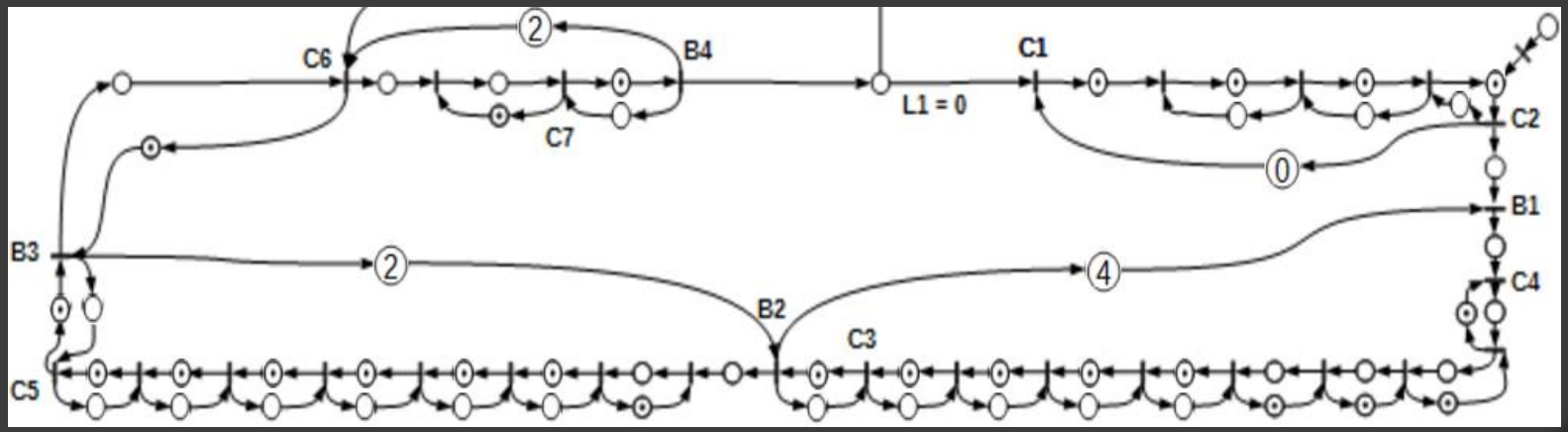
```
<!-- Raffraichissement de la bdd -->
<SCRIPT type="text/javascript">

    $(document).ready(function(){
        refreshTable();
    });


    function refreshTable(){
        $('#table').load('BaseSupervision.php', function(){
            setTimeout(refreshTable, 1000);
        });
    }
</SCRIPT>
```

Réseau de Petri



Réseau de Petri

- Visualisation en temps réel du nombre de palettes présentes dans le magasin



Conclusion