

Mise en oeuvre de ControlBuild

Franck Gérossier
IUT de Montluçon
Université Blaise Pascal



Plan

- Description du contexte
- Création d'un TP (niveau Lic-Pro)
- Test du TP
- Conclusion

Pourquoi ControlBuild ?

- Recherche d'un logiciel de conception d'automatismes « non-constructeurs » qui :
 - Respecte la norme IEC 61131-3.
 - Puisse être utilisé avec tous les automates.
 - Offre une architecture de développement claire.
 - Doit être novateur.
- Prêt du logiciel pendant 1 an afin de le tester.
 - Accord passé entre TNI-Software et certains départements GEl.

Dans quelle formation tester ControlBuild ?

➤ En DUT : NON

- Début de développement : octobre 2005, un peu tard pour une utilisation dans la formation DUT.
- Délicat de mettre en place une formation autour d'un logiciel que l'on n'est pas certain de garder.

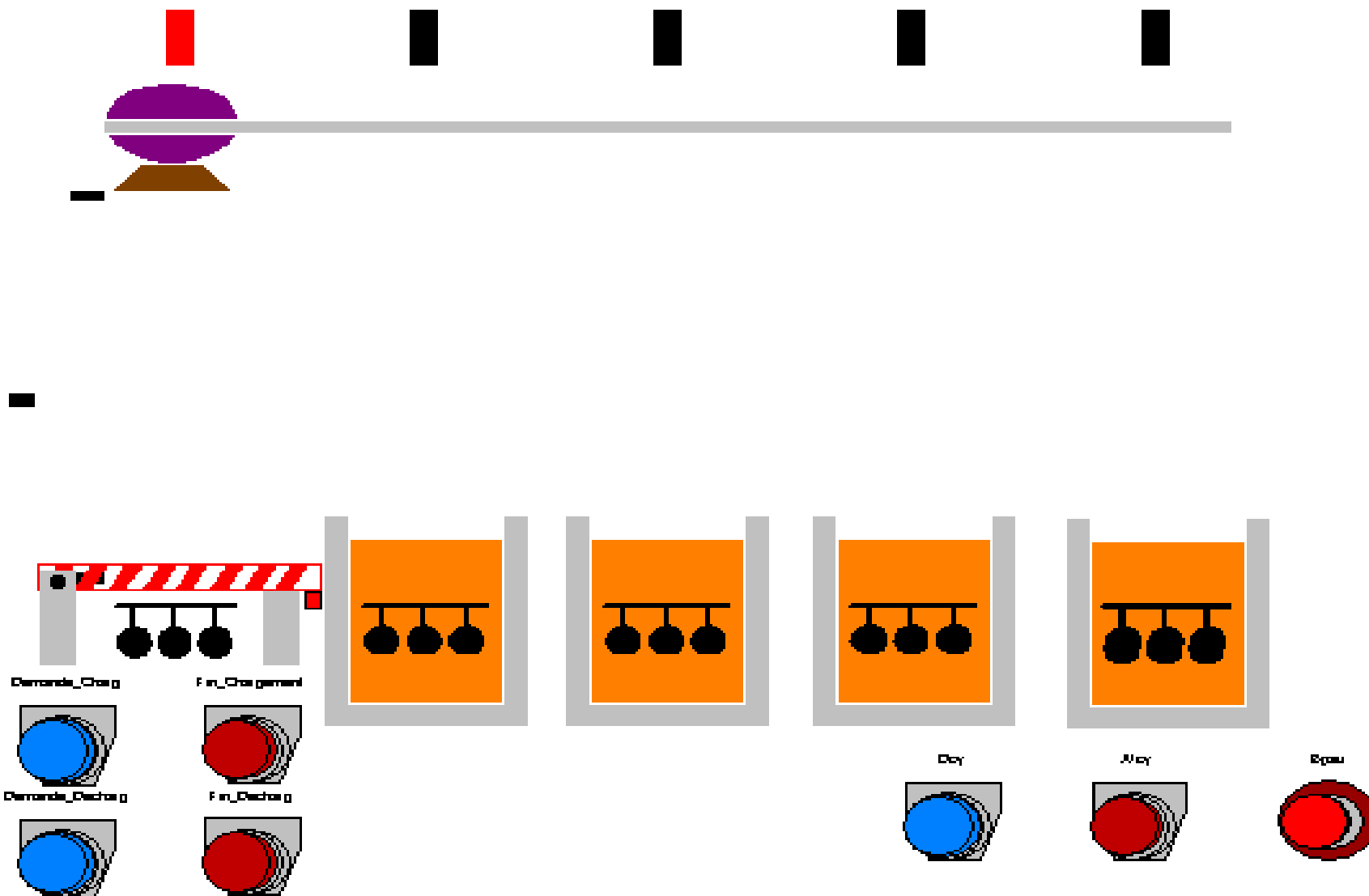
➤ En Licence Professionnelle : OUI

- Les TPs, sur ce thème, se déroulent fin décembre.
- Le profil des étudiants concernés.
- Création d'un TP de 9 heures.
- Mise en oeuvre d'un TP développé à l'origine sur un autre logiciel « d'automatisme »

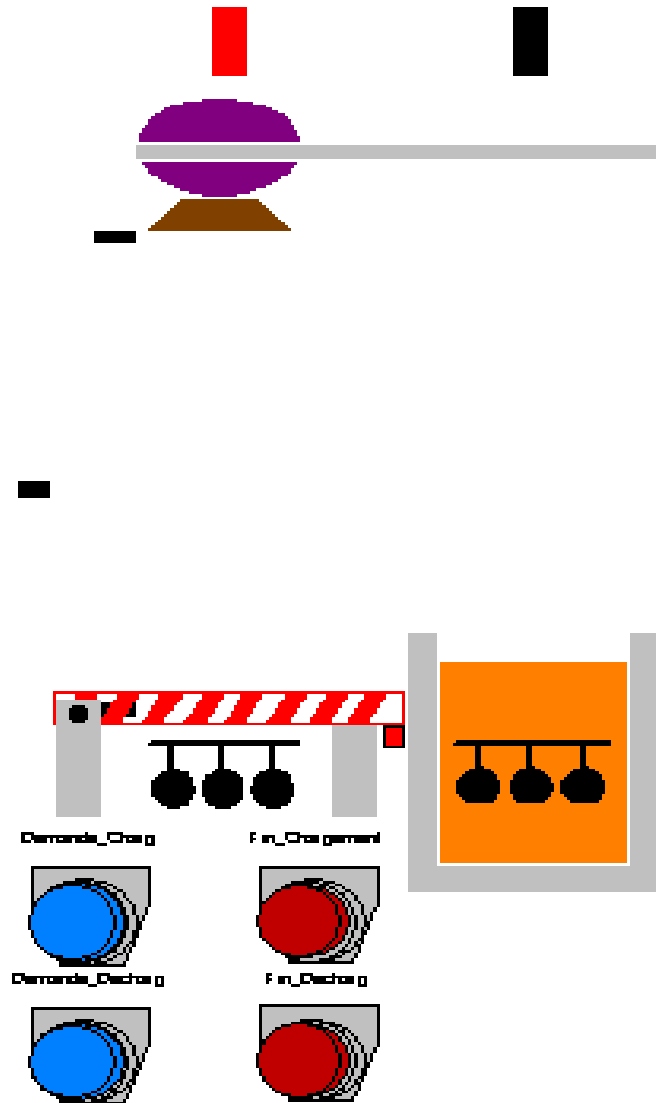
ControlBuild : utilisation en Licence Professionnelle

- Objectifs « conventionnels »
 - Mise en oeuvre des langages de la norme CEI 61131 : compléments sur le langage ST.
 - Retour sur la programmation objet : différenciation entre fonctions / blocs fonctionnels.
- Objectifs « nouveaux »
 - Conception de l'application via un outil type « SADT ».
 - Création et utilisation d'une partie opérative « virtuelle » permettant de valider toutes les étapes de la conception.

Thème du TP : traitement de surface



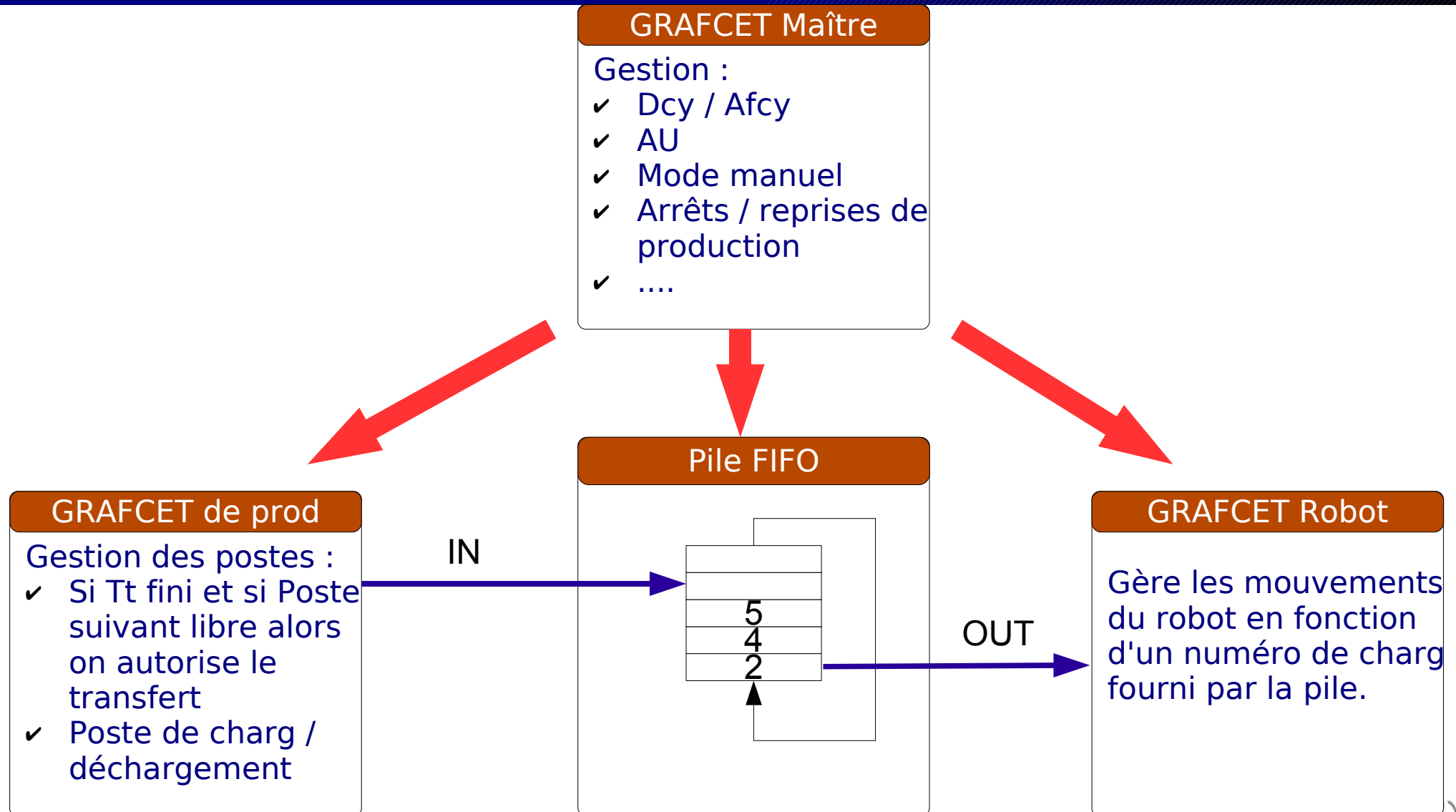
Thème du TP : traitement de surface



Cahier des charges

- ✓ Quatre bains de traitement
- ✓ Un poste de charg/déchargement
- ✓ Lots de pièces accrochés sur des barres de transport
- ✓ Un robot manipulateur
- ✓ Un système automatique d'accrochage des pièces.
- ✓ Un charg/déchargement des pièces effectué par un opérateur
- ✓ Chaque pièce doit subir dans l'ordre les 4 traitements.
- ✓ Les temps de traitements sont identiques et très supérieurs aux temps de transferts.

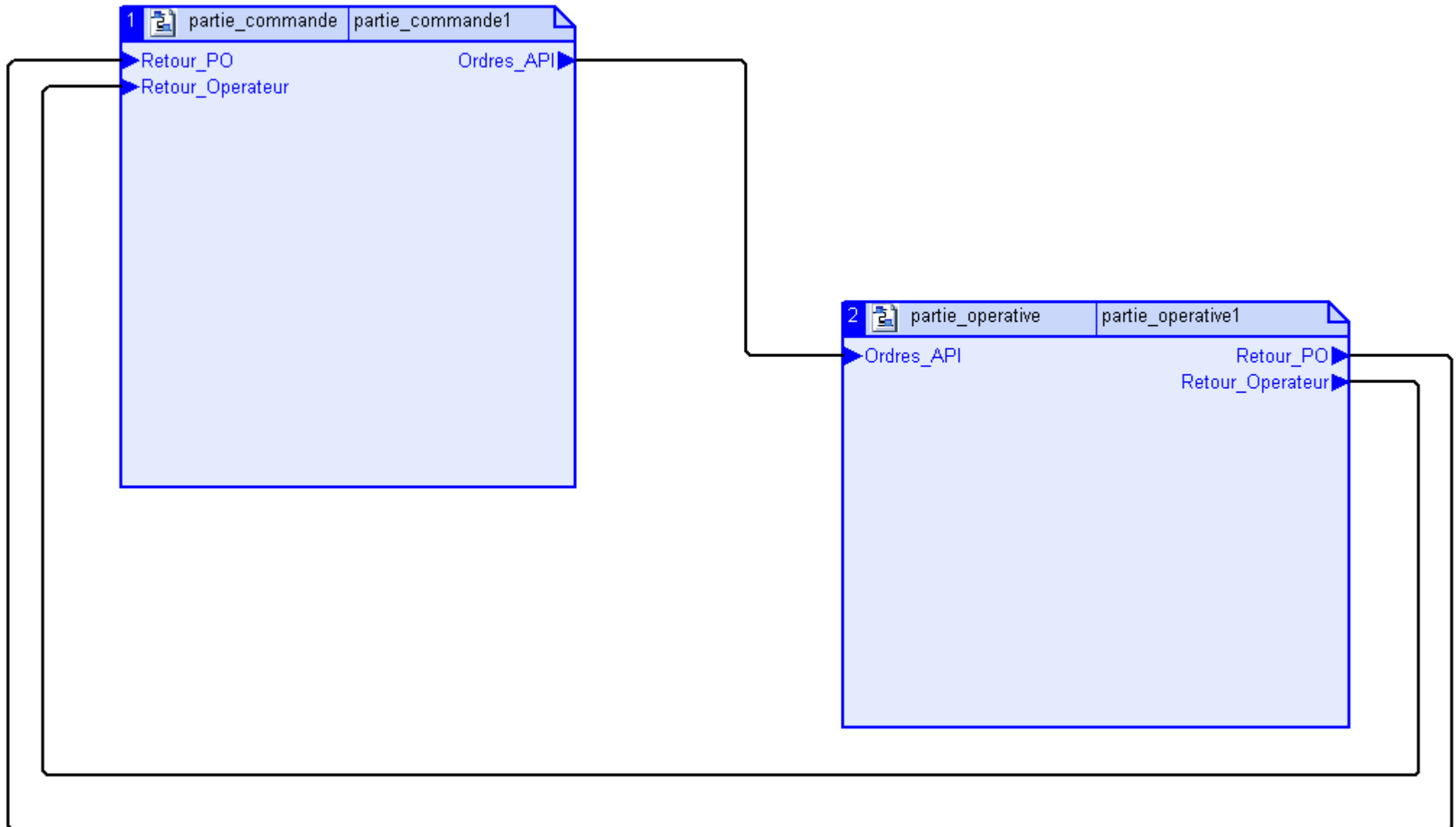
Mise en forme du TP (version antérieure)

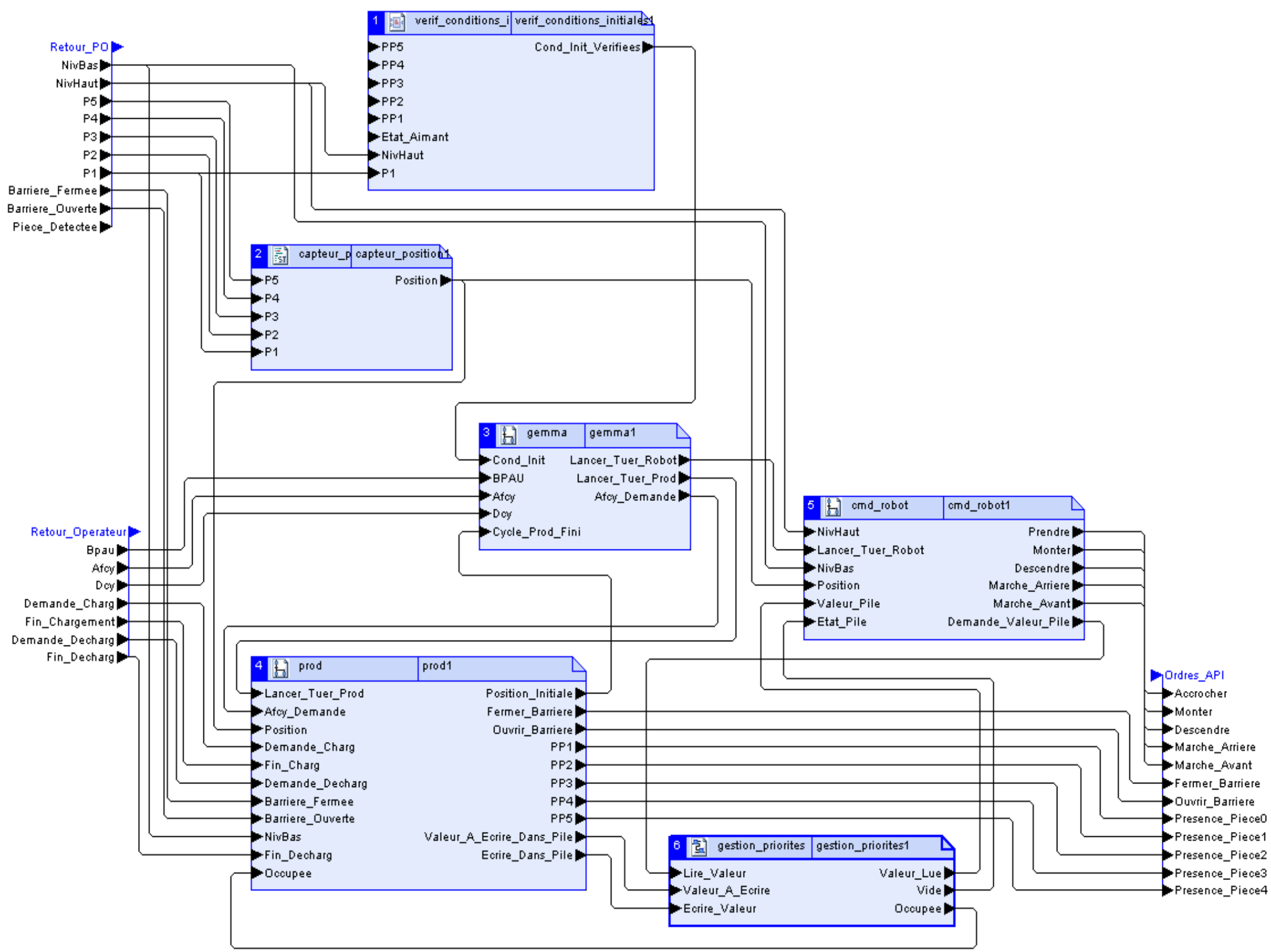


Points abordés par les étudiants (version antérieure)

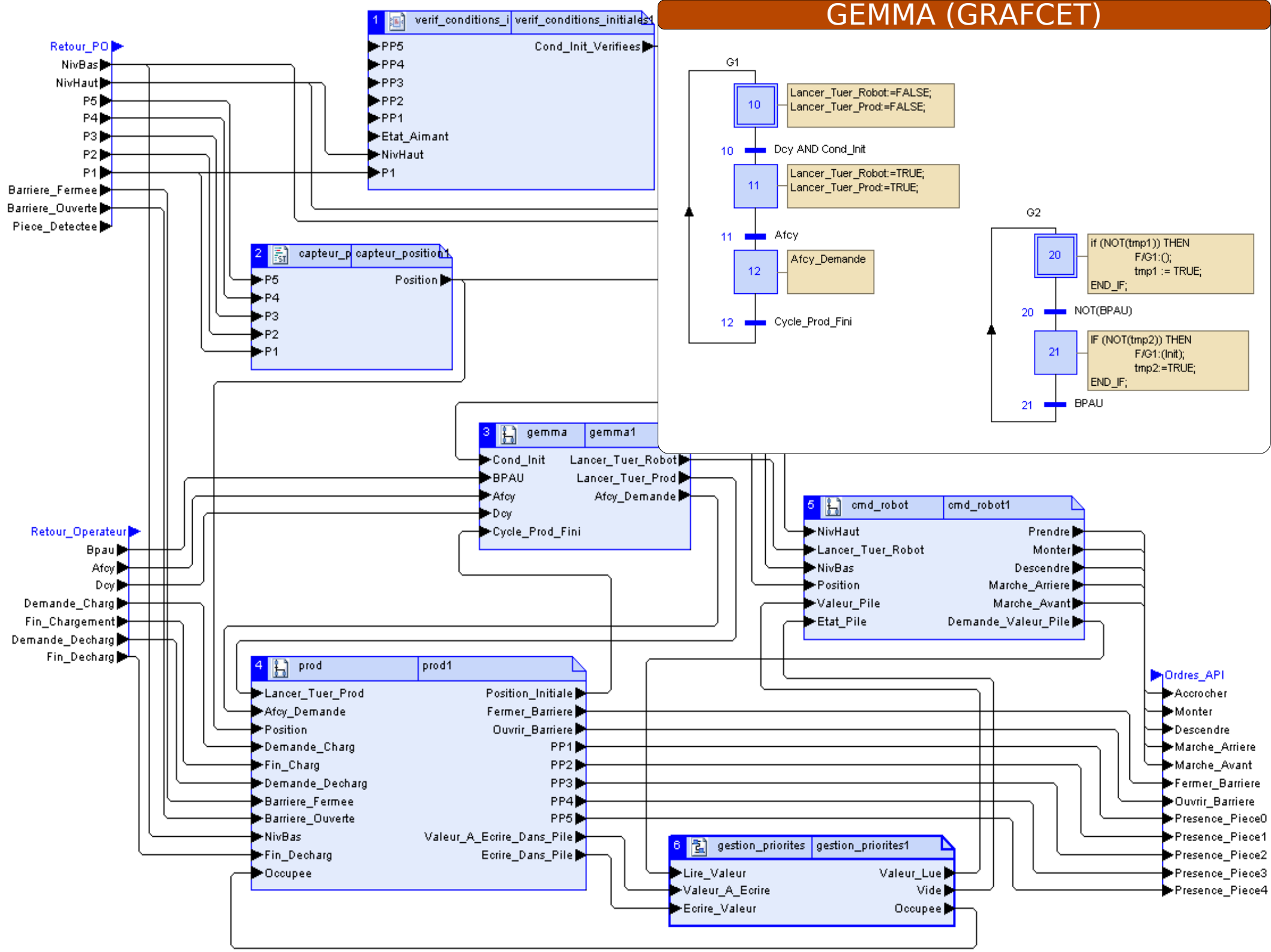
- **Éléments fournis**
 - GRAFCET robot
 - GRAFCET de production simplifié
 - Simulation de la partie opérative (ISAGRAF)
- **Éléments réalisés par les étudiants**
 - Bloc fonctionnel FIFO + tests (langage ST)
 - GRAFCET de production avec gestion du poste charg / Déchargement (SFC + ...)
 - GRAFCET maître (SFC + ...)

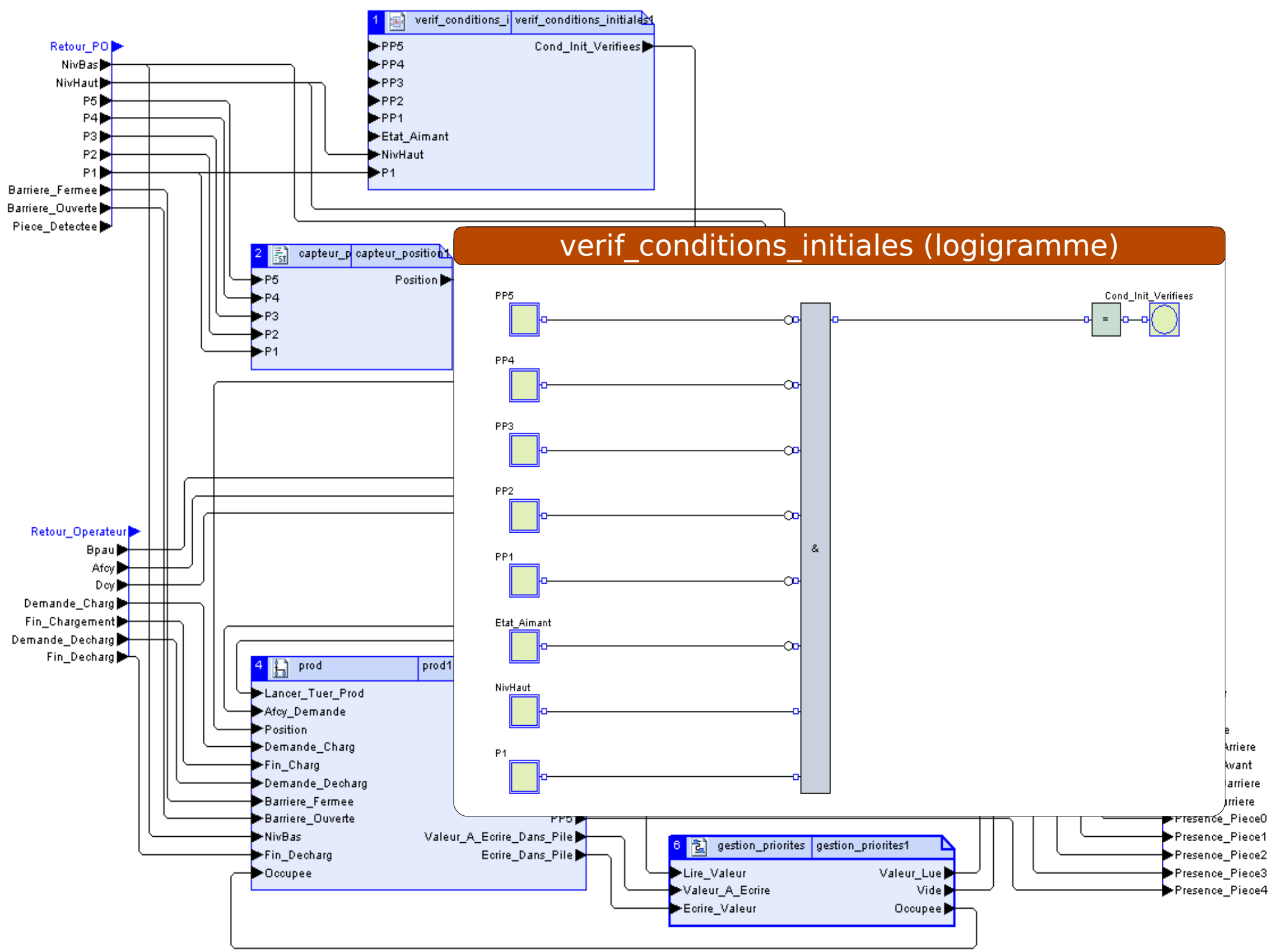
Mise en forme du TP sous ControlBuild

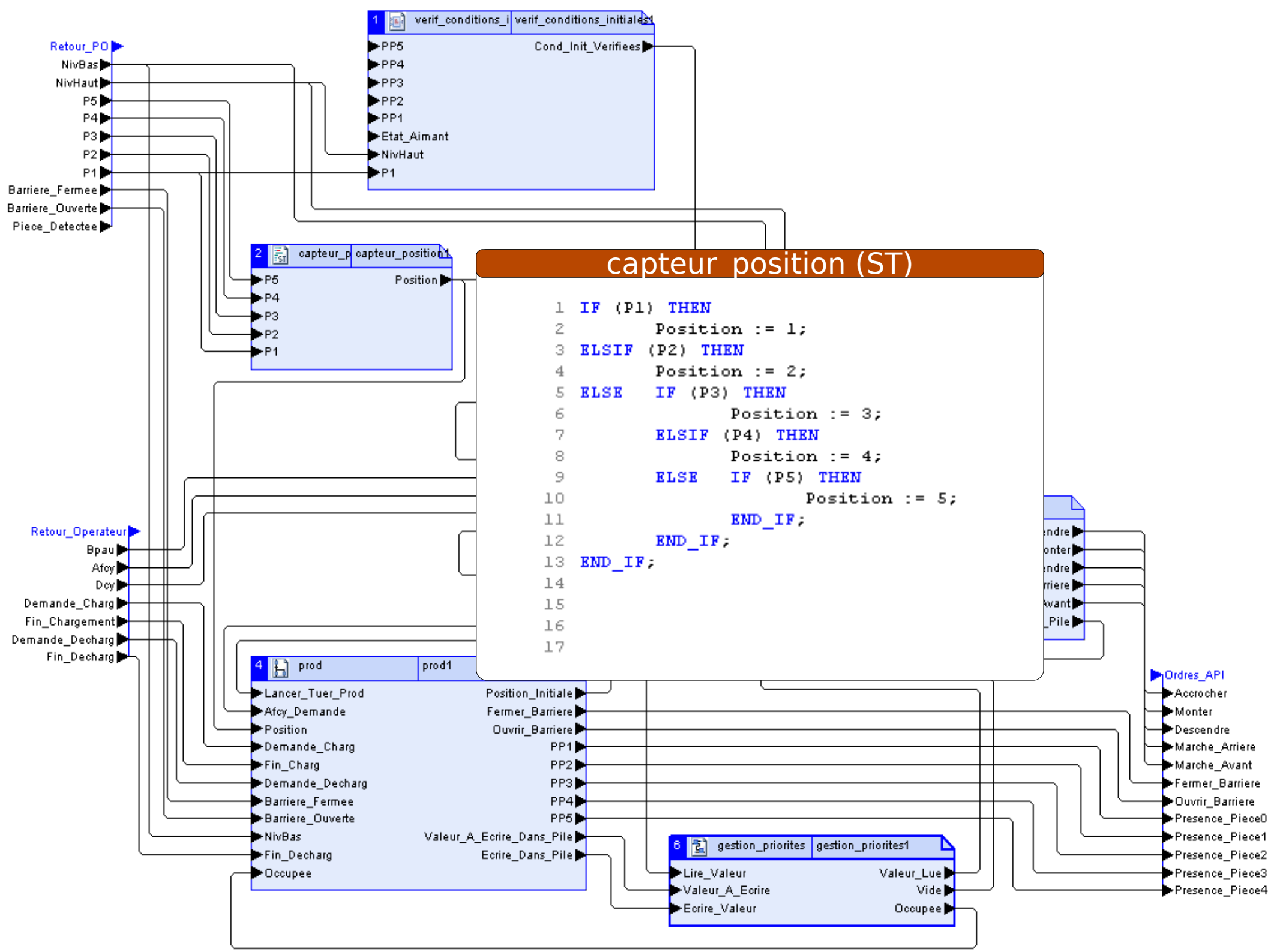




GEMMA (GRAF CET)

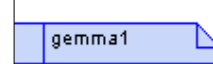
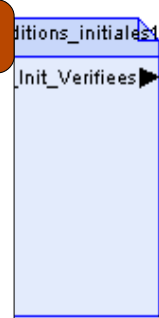




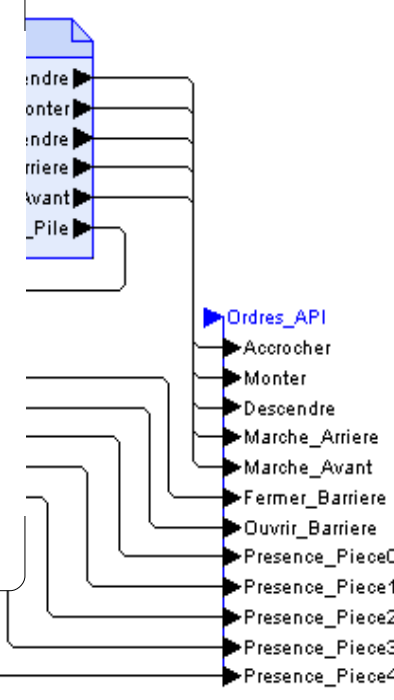
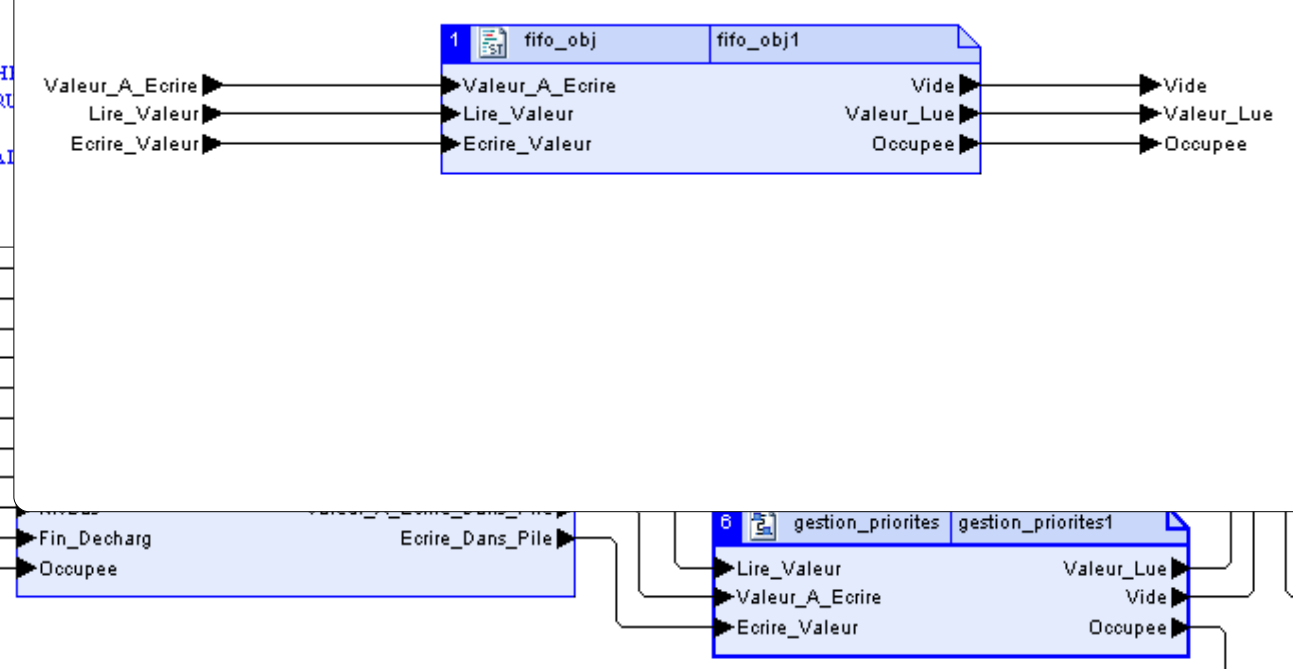


fifo_obj (ST)

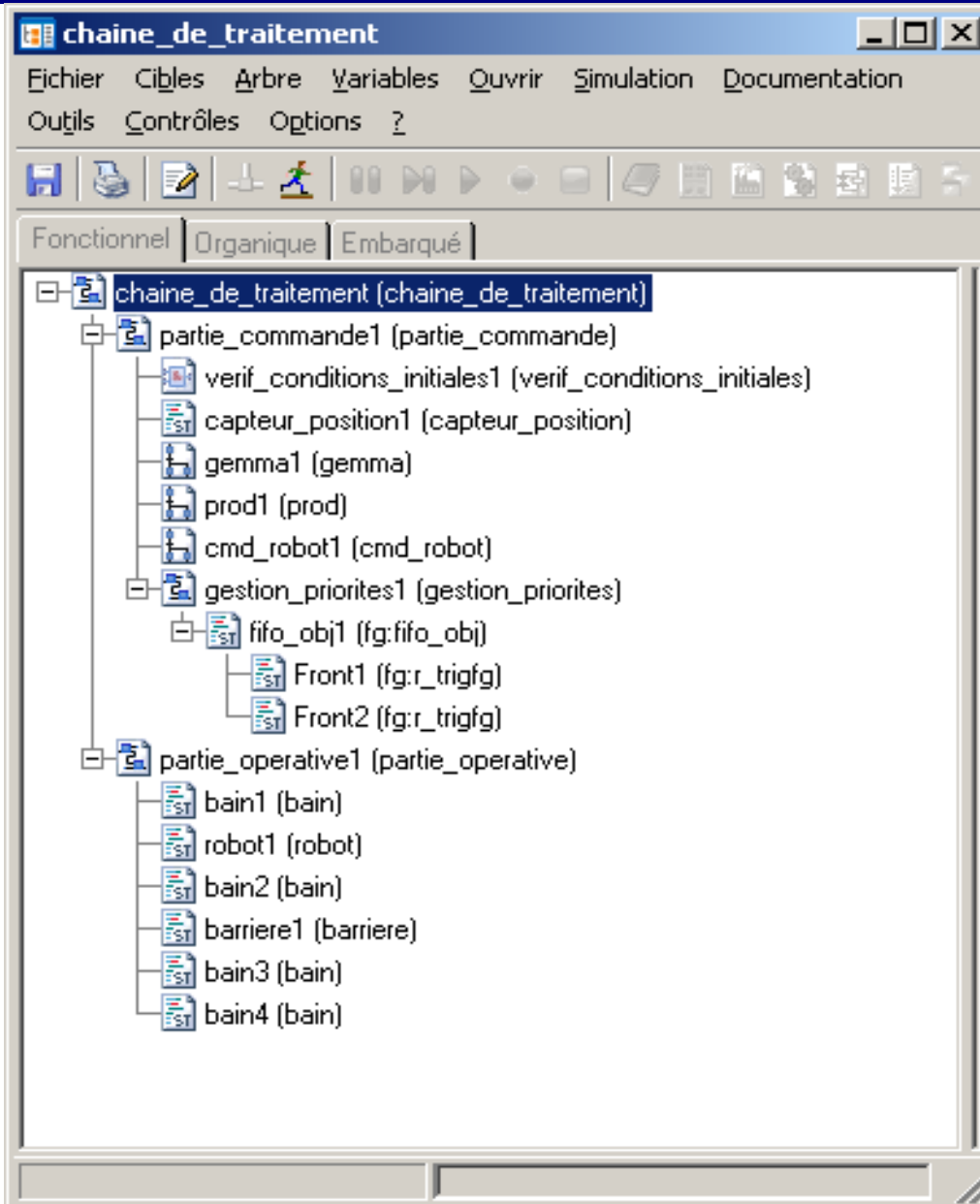
```
1 Front1(Ecrire_Valeur);
2 Front2(Lire_Valeur);
3 Front_Ecrire := Front1.Q;
4 Front_Lire := Front2.Q;
5
6
7 (*Front_Ecrire := r_trig(Ecrire_Valeur , tmp1);
8 Front_Lire := r_trig(Lire_Valeur , tmp2);
9 *)
10 IF (Front_Ecrire) THEN
11     Pile[IndIn] := Valeur_A_Ecrire;
12     IndIn := IndIn + 1;
13     IF (IndIn = 6) THEN
14         IndIn := 1;
15     END_IF;
16 END_IF;
17
18 IF (Front_Lire AND (IndOut <> IndIn)) THEN
19     Valeur_Lue := Pile[IndOut];
20     IndOut := IndOut + 1;
21     IF (IndOut = 6) THEN
22         IndOut := 1;
23     END_IF;
24 END_IF;
25
26 IF (IndIn = IndOut) THEN
27     Vide := TRUE;
28 ELSE
29     Vide := FALSE;
30 END_IF;
31
32 IF (Ecrire_Valeur) THEN
33     Occupee := TRUE;
34 ELSE
35     Occupee := FALSE;
36 END_IF;
37
38 ..
```



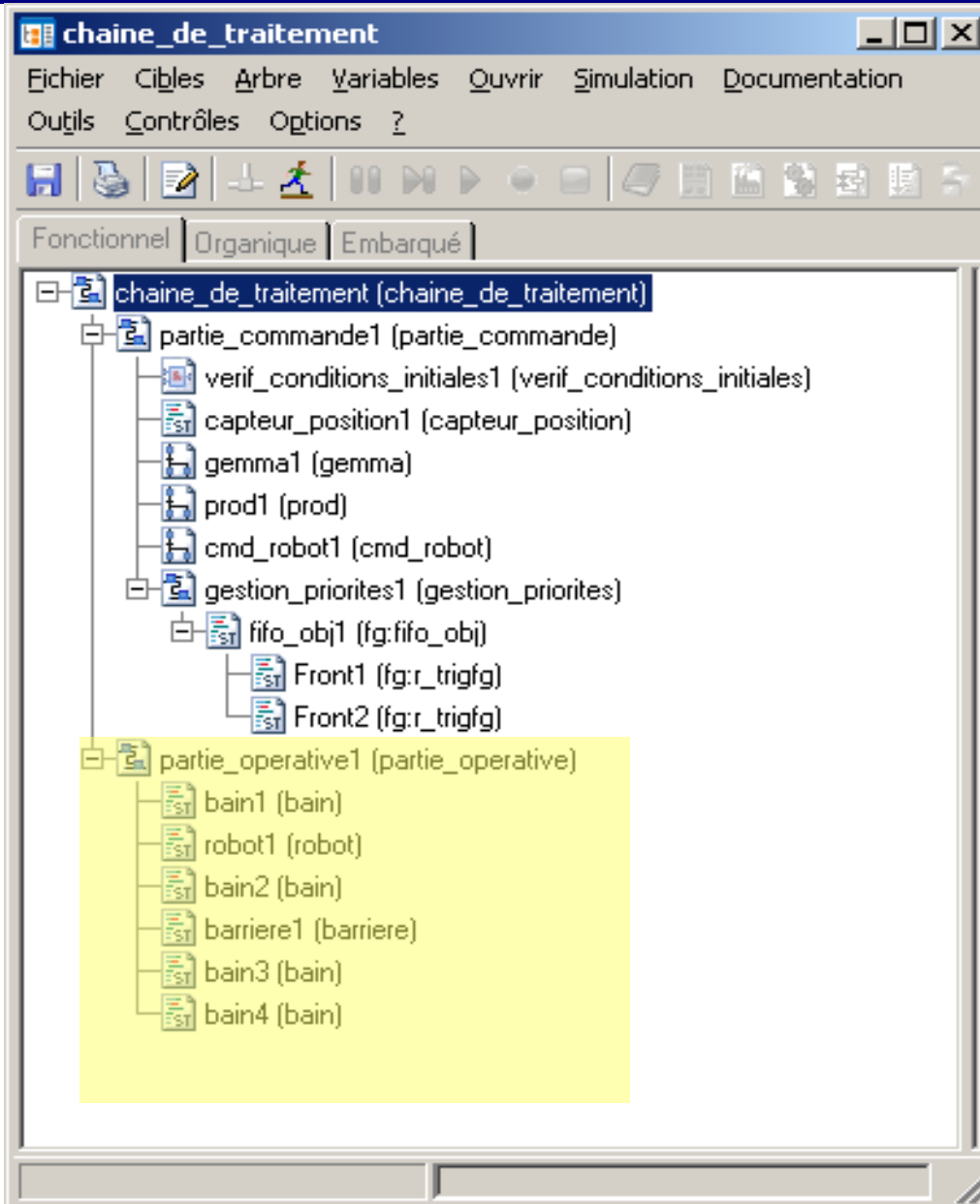
Gestion priorités (MAC)



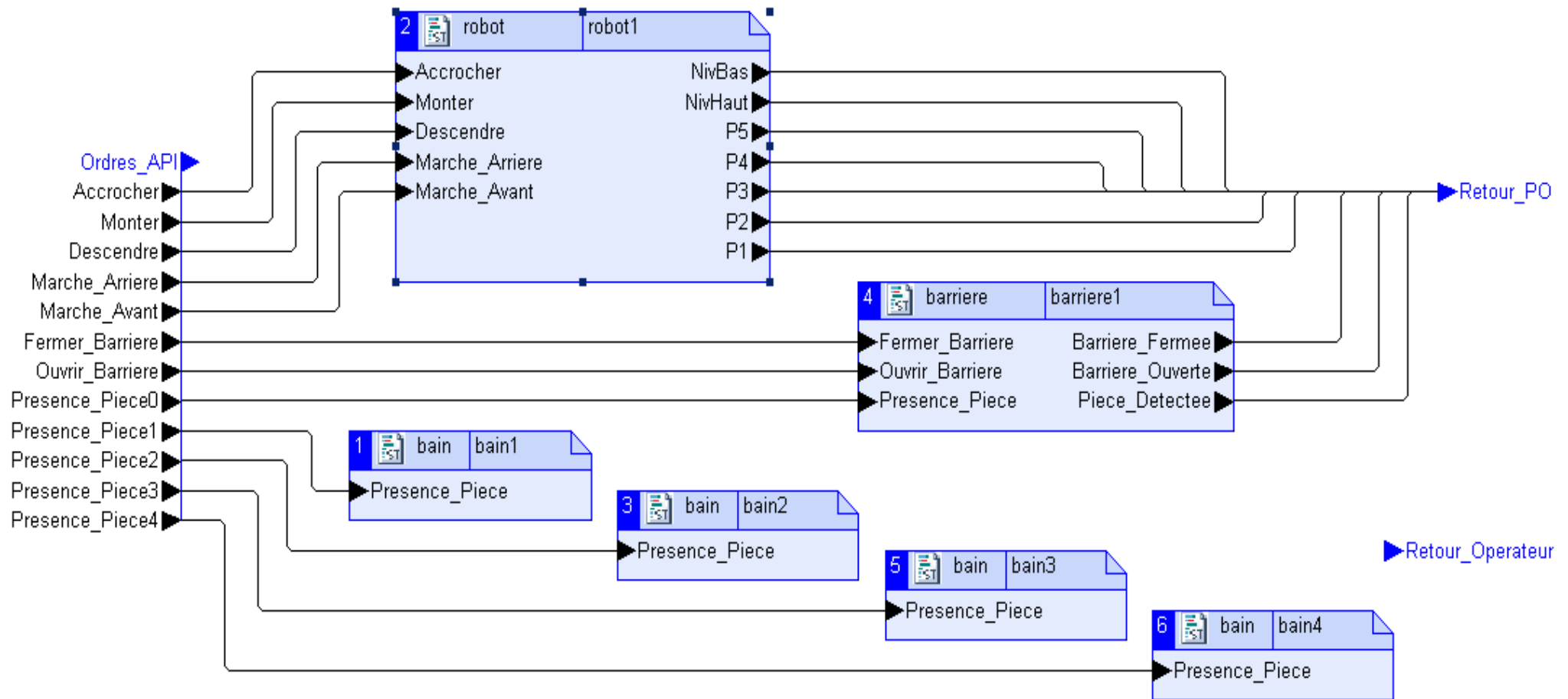
Mise en forme du TP sous ControlBuild : Arbre



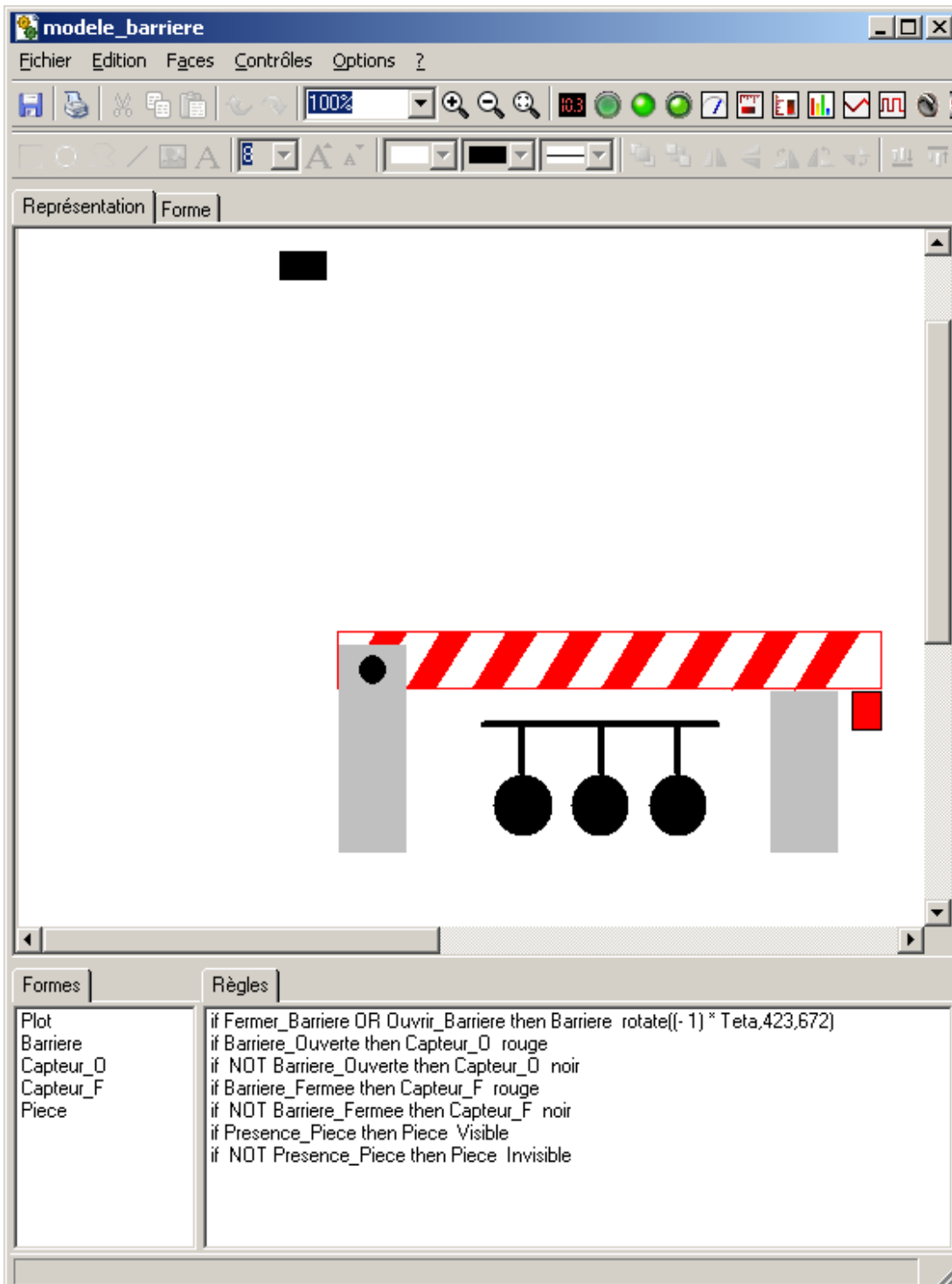
Mise en forme du TP sous ControlBuild : Arbre



TP sous ControlBuild : partie opérative

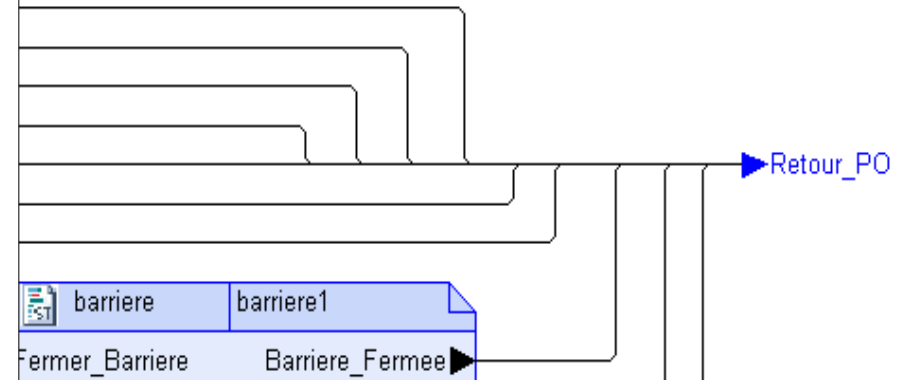


Modèle d'animation de barriere



Description du contexte
Création du TP
Test du TP

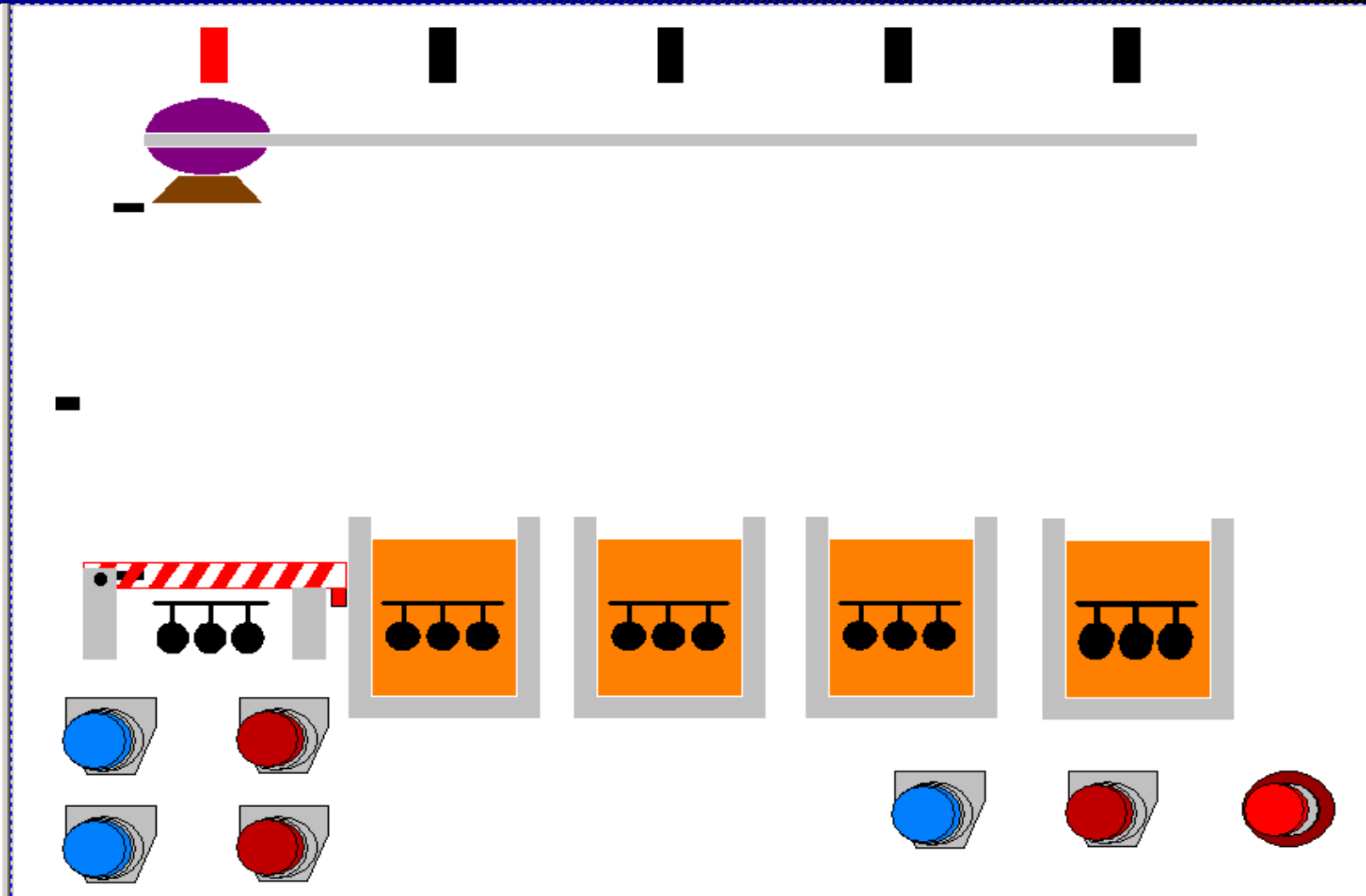
relative



barriere (ST)

```
1 IF (Fermer_Barriere) THEN
2     IF (Teta <> 0) THEN
3         Teta := Teta - IncR;
4         Barriere_Ouverte := FALSE;
5     END_IF;
6
7 ELSIF (Ouvrir_Barriere) THEN
8     IF (Teta <> 90) THEN
9         Teta := Teta + IncR;
10        Barriere_Fermee := FALSE;
11    END_IF;
12 END_IF;
13
14 IF (Teta = 0) THEN
15     Barriere_Fermee := TRUE;
16 ELSIF (Teta = 90) THEN
17     Barriere_Ouverte := TRUE;
18 END_IF;
19
20 IF (Presence_Piece) THEN
21     Piece_Detectee := TRUE;
22 ELSE
23     Piece_Detectee := FALSE;
24 END_IF;
25
```

Partie opérative : Synoptique



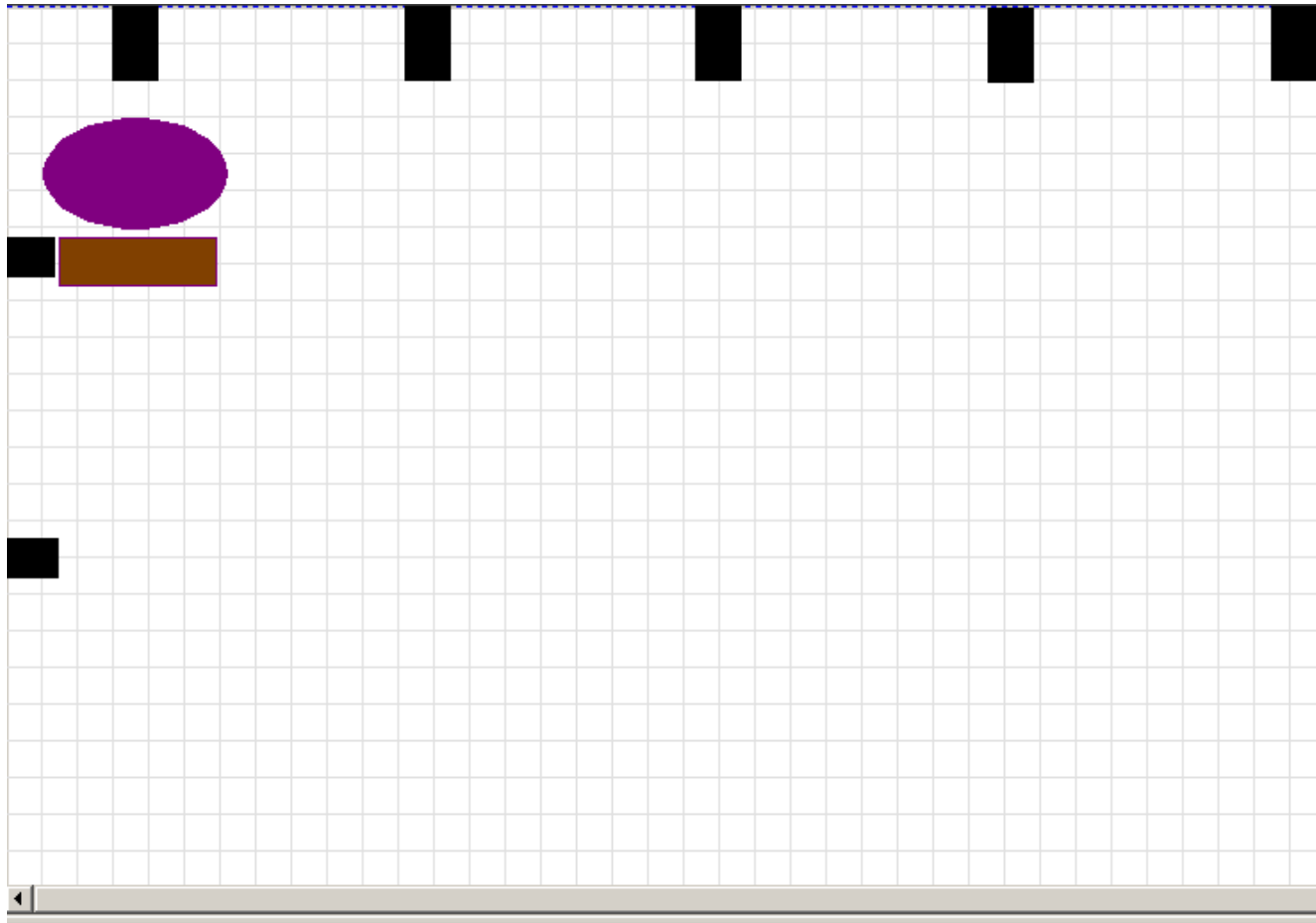
modele_bain (comportement : (bain) - bain1)
modele_bain (comportement : (bain) - bain2)
modele_bain (comportement : (bain) - bain3)
modele_bain (comportement : (bain) - bain4)
modele_robot (comportement : (robot) - robot1)
modele_barriere (comportement : (barriere) - barriere1)

Points abordés par les étudiants sous ControlBuild

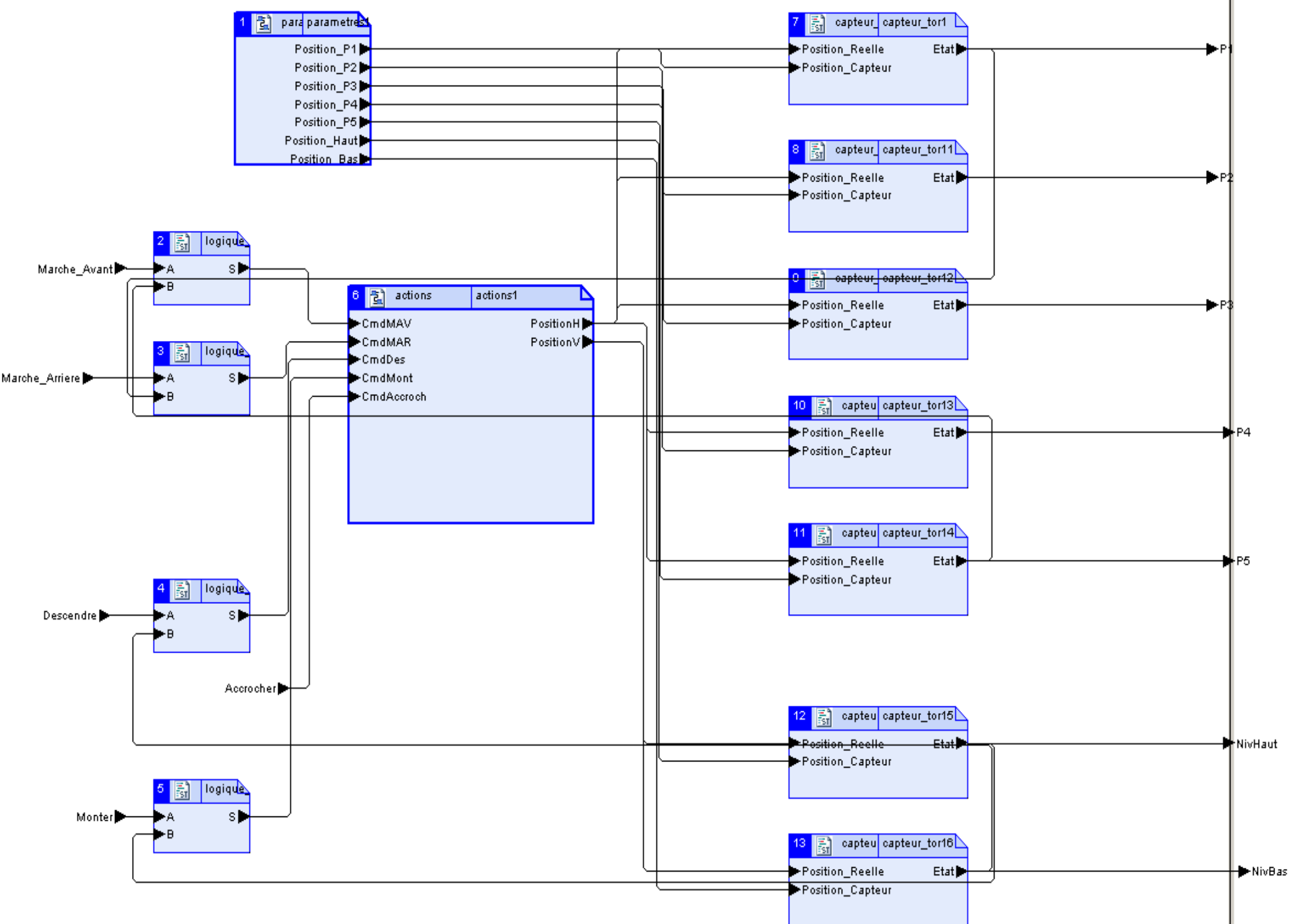
- Éléments fournis (1ère partie)
 - Tout, sauf la pile Fifo (on fournit une version bêta)
- Éléments réalisés par les étudiants (1ère partie)
 - Étude du logiciel : description MAC.
 - Création et test de la pile Fifo (ST)
- Éléments fournis (2ème partie)
 - Rien.
- Éléments réalisés par les étudiants (2ème partie)
 - Redéfinir le robot (partie commande et partie opérative)

Robot +

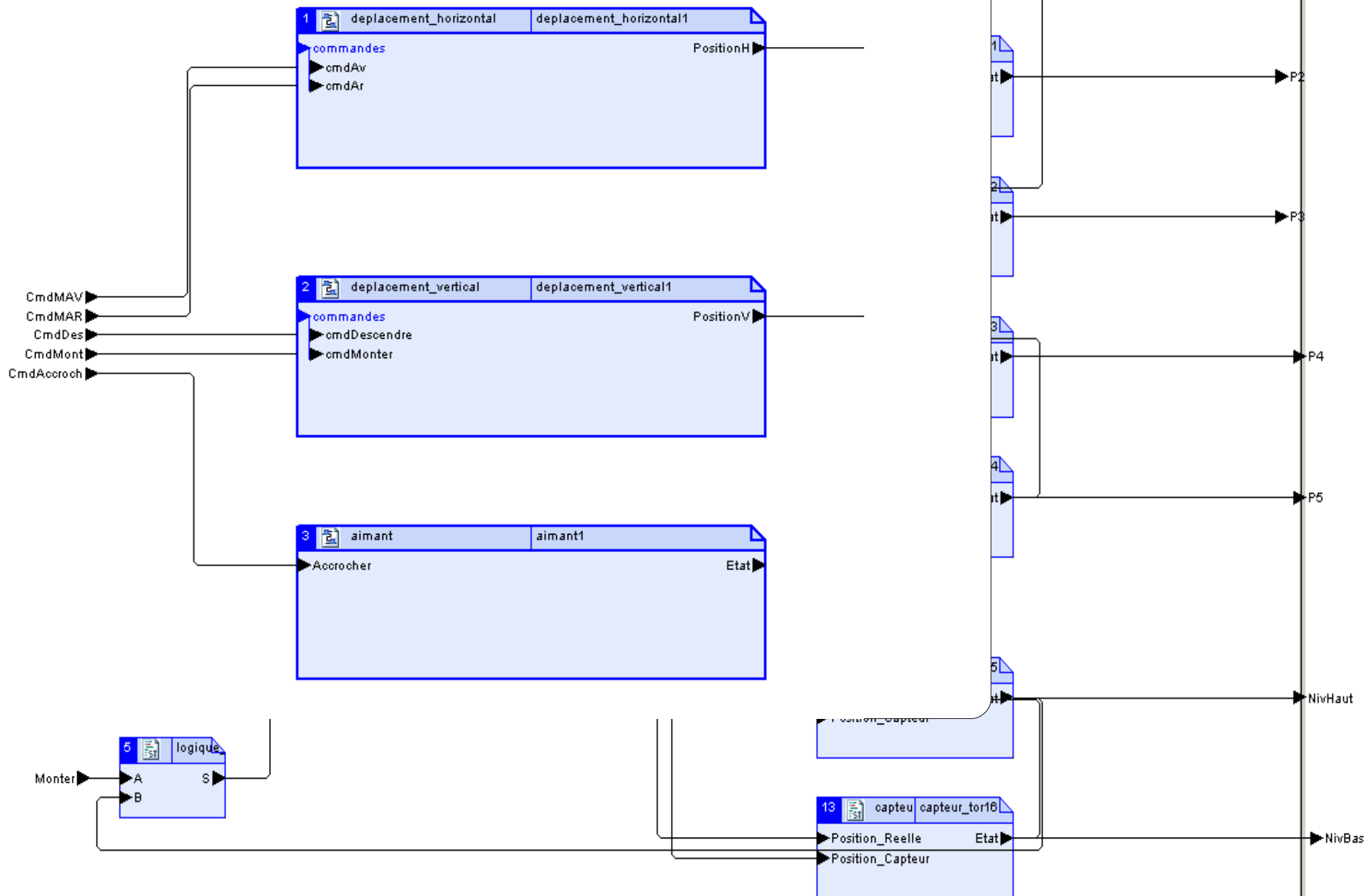
Synoptique Robot



```
model_actions (comportement : (actions) - actions1)
model_captor (comportement : (capteur_tor) - capteur_tor1)
model_captor (comportement : (capteur_tor) - capteur_tor11)
model_captor (comportement : (capteur_tor) - capteur_tor12)
model_captor (comportement : (capteur_tor) - capteur_tor13)
model_captor (comportement : (capteur_tor) - capteur_tor14)
model_captor (comportement : (capteur_tor) - capteur_tor15)
model_captor (comportement : (capteur_tor) - capteur_tor16)
```

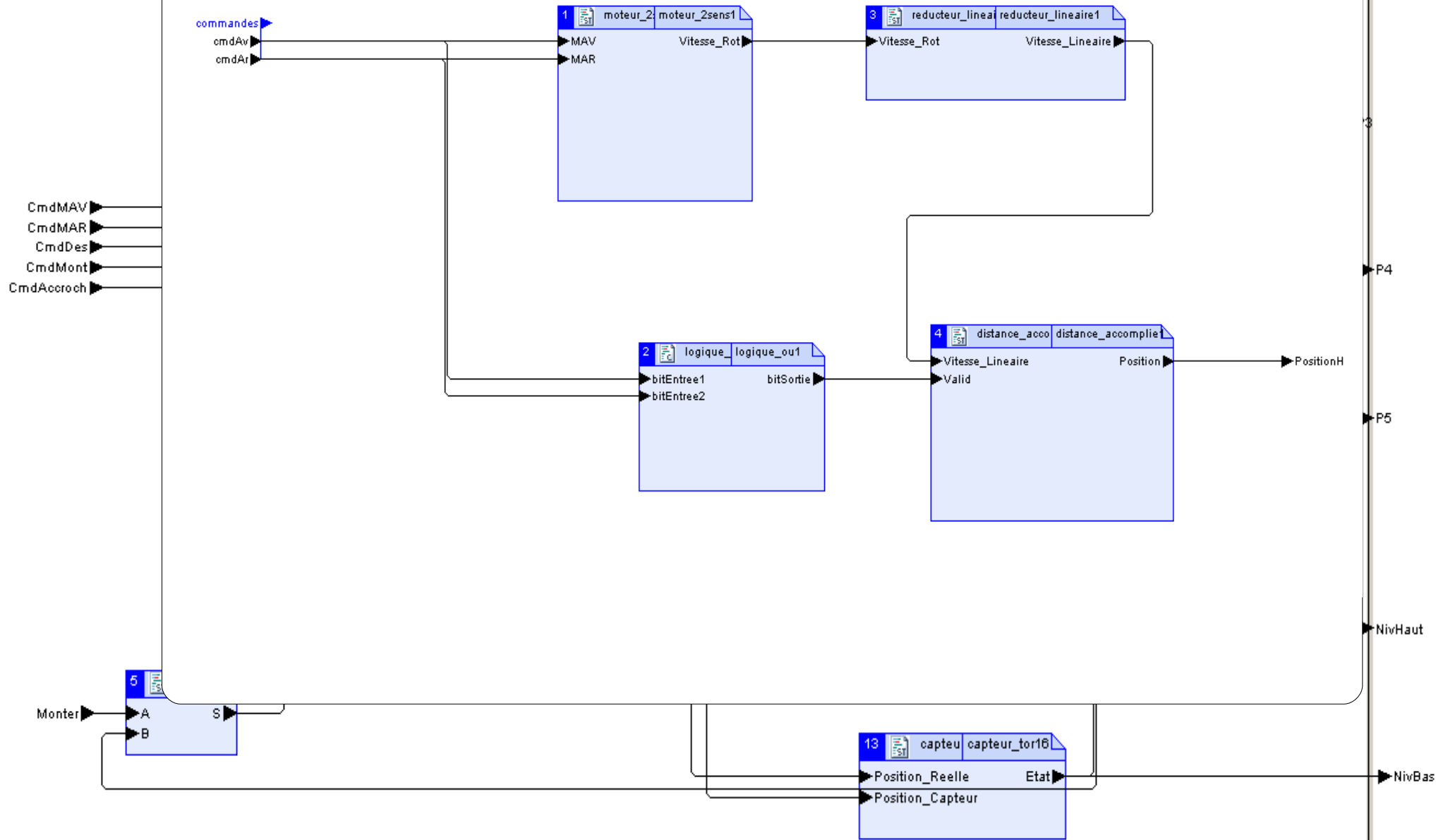


Actions (MAC)



Actions (MAC)

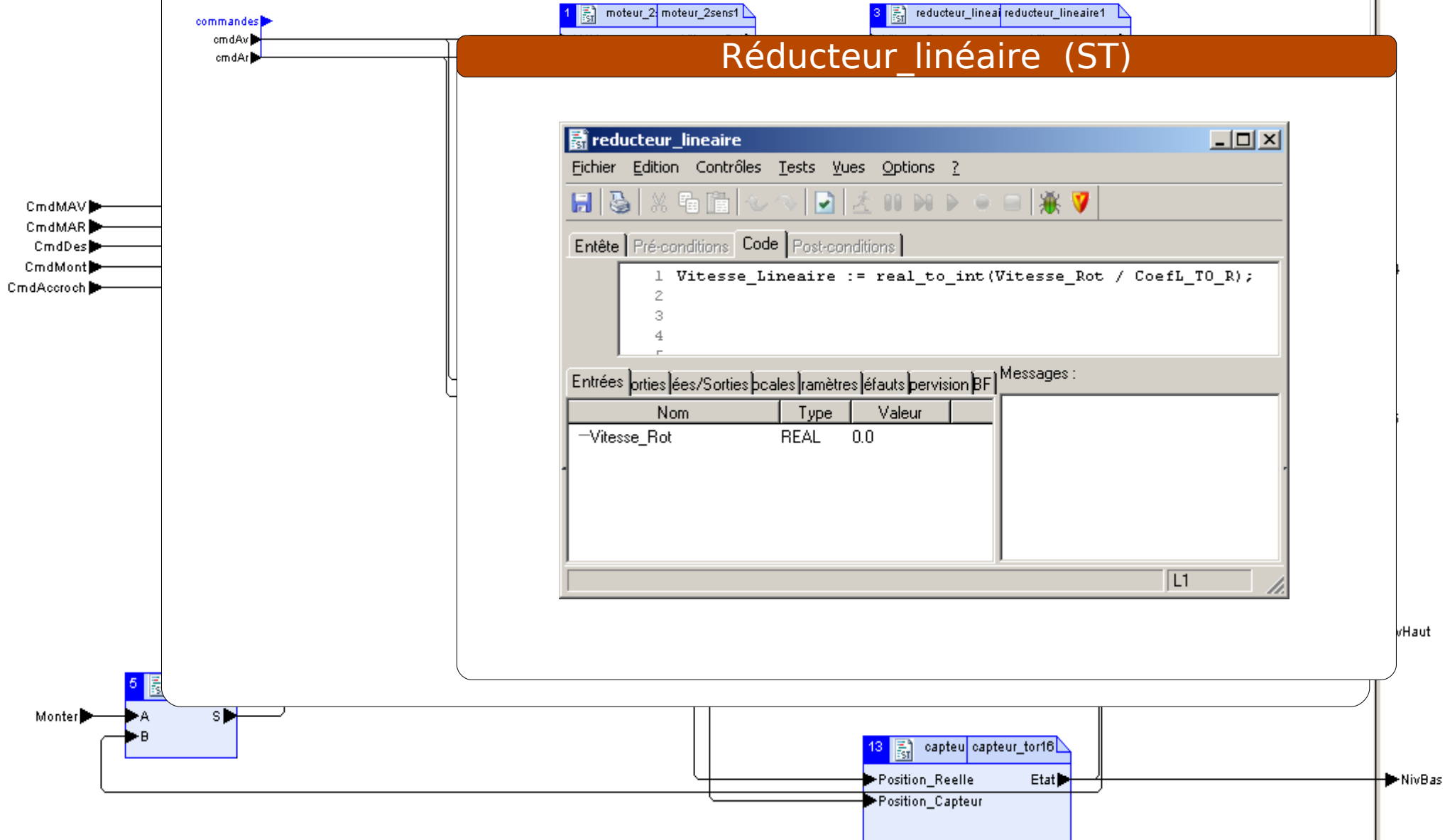
Déplacement_horizontal (MAC)



Actions (MAC)

Déplacement_horizontal (MAC)

Réducteur_lineaire (ST)



Utilisation de ControlBuild par les étudiants

- Séance de 9 heures découpée en 2*4,5 heures
- Travaillent en binôme
- Possèdent comme bases :
 - Leurs connaissances sur les langages.
 - Un cours sur : le GEMMA, le SADT, Les langages.
 - Un sujet de TP qui au départ se présente comme un didacticiel :
 - Comment ouvrir, utiliser et modifier une application
 - Comment utiliser la description en MAC
 - Comment créer des « génériques » et les modules associés (animation, synoptique ...)

1ère partie : création d'une pile FIFO

- Les étudiants sont désorientés par l'ergonomie du logiciel. Une fois cette difficulté vaincue, ils découvrent son fonctionnement.
- L'approche SADT des MAC est bien accueillie. Elle concrétise les concepts développés en cours.
- Les étudiants, issus en majorité d'un BTS, MAI ou électrotechnique, n'ont jamais suivis de cours d'informatique. Ils éprouvent beaucoup de difficultés pour discerner le « générique » des objets (notion d'instance).
- En partant d'une application complète, l'étudiant progresse plus vite.
- La modification d'un bloc fonctionnel FIFO ne pose pas de problèmes avec cette approche.

2ème partie : création d'un modèle de simulation du robot

- L'étudiant doit concevoir le projet
- Le découpage en modules fonctionnels du robot pose des problèmes. L'étudiant comprend le principe, mais le découpage du robot en fonctionnalités reste délicat (autonomie des étudiants dans l'analyse).
- Une fois l'étudiant orienté (présentation collective des fonctionnalités), la conception du robot par une modélisation type « SADT » est rapide.
- La création des modèles et des synoptiques, après explication, a bien fonctionné.
- Nous avons volontairement insisté sur la partie « analyse fonctionnelle descendante ». L'intégration des notions physiques dans les modèles de simulation n'a pas été demandé (inertie des moteurs, frottements ...).

Conclusion

- Les TPs se sont déroulés correctement. L'approche, type « SADT » du logiciel, reste au départ déroutante pour les étudiants.
- Cela oblige l'étudiant à formaliser son analyse, pour un résultat final beaucoup plus « réfléchi » que lors de l'utilisation de logiciels « traditionnels ».
- Il reste à la société TNI-Software à corriger certains éléments de conception : suivi de la norme 61131, gestion des instances « multiples » ...

Conclusion

- Les évolutions présentes et futures de ce logiciel :
 - Amélioration de la lisibilité des MAC
 - Programmation multi-automates (Schneider, Siemens ..)
 - Gestion des réseaux
 - Découpage de l'application pour le partage sur les cibles (automates), avec mise en oeuvre de la communication inter-automate.
- Se sont, à priori, des choix d'évolution judicieux.
- Bilan positif qui a poussé l'IUT de Montluçon à l'intégrer parmi ses outils de formation.