Contrôle d'une cellule flexible de production robotisée via vidéo projecteur interactif

Notice d'utilisation

Date	Version	Commentaire	Auteur
23/06/15	0.1	Création	Alexandra GORRY POLLET

Table des matières

I) Prérequis

Pour reprendre le projet, il faut s'assurer d'avoir l'espace de travail ROS qui contient tous les fichiers nécessaires à son fonctionnement. Il est situé dans le compte utilisateur *alexandra* (connexion avec le mot de passe aip-2015) lorsque l'ordinateur est démarré sous Ubuntu.

Il s'agit du dossier nommé *workspace* situé dans *Home*. Ce dernier regroupe tous les fichiers qui seront compilés pour lancer les différents noeuds ROS.

D'autres documents peuvent être utiles pour connaître le projet, tels que le cahier des charges ou le plan de développement (amené à être modifié). Ils sont situés dans le dossier *Cellule Flexible*. C'est aussi dans ce dossier que seront rangés les fichiers à lancer sur PL7 et contenant le code à charger sur les automates. Ils se nomment *scenario1.stx* pour l'automate 1 et *scenario2.stx* pour l'automate 2. Ils contiennent déjà toute la configuration. De plus, une représentation de la machine à états codée nommée *MAE.jpg* est disponible dans ce même dossier.

II) Contenu du dossier *workspace*

Sont présents à l'ouverture du dossier, 4 sous-dossiers : *build*, *devel*, *install* et *src*. Celui qui nous intéresse ici et qui contient tout le code est le dernier. C'est là qu'est l'ensemble du premier scénario : le dossier *scenario1*, qui est notre package et qui contient lui-même 4 sous-dossiers : *include*, *msg*, *src*, *srv* ; ainsi que les fichiers *CmakeLists.txt* et *package.xml*.

C'est ici que se trouve le coeur du projet. Les aspects importants sont développés par la suite.

1) *CmakeLists.txt*

C'est dans ce fichier qu'il faut rajouter les lignes nécessaires à l'insertion d'un nouveau type de message ou de service. Il faudra rajouter le nom des fichiers créés dans les parties *add_message_files* et *add_service_files*.

C'est aussi ici qu'il faudra rajouter les lignes nécessaires suite à la création d'un nouveau noeud pour que celui-ci soit compilé. Ces lignes sont situées à la fin du document.

On trouve par exemple :

```
add_executable(code_serv src/code_serv.cpp)
add_dependencies(code_serv scenario1_gencpp)
target_link_libraries(code_serv ${catkin_LIBRARIES} modbus)
```

Pour chaque nouveau noeud, il faudra suivre ce modèle en précisant :

- dans première ligne : le nom du noeud et le chemin du fichier dans lequel son code se situe ;
- dans la deuxième ligne : le nom du noeud suivi du nom du package avec l'extension précisant que l'ont code en C++ (gencpp) ;
- dans la troisième ligne : ajouter la librairie au noeud (ici modbus).

NB : Il est à noter que, plus haut dans le *CmakeLists.txt*, on retrouve les lignes :

```
include_directories(
     ${catkin_INCLUDE_DIRS}
     /usr/local/include/
)
```

Elles servent à indiquer le chemin d'où se situe la librairie à utiliser et sont très importantes. Il ne faudra toutefois pas oublier de faire un *#include* dans le code des noeuds concernés.

2) *msg* et *srv*

C'est dans ces dossiers que seront ajoutés respectivement :

- les nouveaux types de messages avec l'extension .msg ;
- les nouveaux types de services avec l'extension .srv.

3) *src*

Le dossier *src* contient le code des noeuds. Tout le code est commenté et expliqué dans les fichiers.

Les fichiers *code_serv.cpp* et *publisher_zone1.cpp* sont les fichiers utiles pour le projet. Ils sont, comme leurs noms l'indiquent, le service DemandeTrajet et le publisher du topic EtatZone.

Le fichier *code_ client.cpp* sert à simuler le noeud qui envoie des demandes au service. Ce noeud est normalement situé du côté du vidéo projecteur interactif mais il peut être utile de le simuler.

Le fichier *publisher_zone2.cpp* est le début du développement du deuxième scénario (qui est à rajouter au premier) mais devra être repris. Il reste cependant une base intéressante bien qu'il ressemble fortement à celui de la zone 1.

III) Compiler et lancer le projet

Pour compiler le projet, il faut sourcer le bon dossier dans le terminal, c'est-à-dire le *workspace*. Pour cela, utiliser la commande :

```
cd ~/workspace/
```

Pour lancer la compilation, taper :

```
catkin_make
```

S'il n'y a pas d'erreur, on peut lancer le coeur ROS avec :

```
roscore
```

L'étape suivante consiste à lancer les noeuds. Il faudra autant de nouvelles fenêtres dans le terminal que de noeuds à lancer (le client et le publisher pour le scenario 1, par exemple). Pour chacun, exécuter :

```
rosrun scenario1 publisher_zone1
```

Il faut bien sûr adapter cette ligne avec les bons noms de package et de noeud.

IV) Test

Lorsque tous les noeuds sont lancés, il faut charger les machines à états sur les automates. Pour le scénario 1, seul l'automate 1 est requis. Il faut ensuite faire une demande de trajet via un noeud ROS :

- soit du côté du tableau interactif ;
- soit avec le client qui sert de simulation.

De plus, pour pouvoir charger un programme sur un automate via le réseau, il faut que le Driver XIP soit en marche. Et, dans PL7, ouvrir le volet "AP", aller dans "Définir l'adresse de l'automate...", choisir de driver "XIP01" et rentrer l'adresse XWAY de l'automate entre accolades suivi de "SYS" dans le champ prévu à cet effet (exemple : {1.3}SYS pour l'automate1).