Bunny

Table des matières

1 Introduction

Implémentation du serveur Bunny.

L'arborescence ci-dessous montre comment le code du LAPP (les feuilles) est appelé à partir de la mise en marche du serveur.

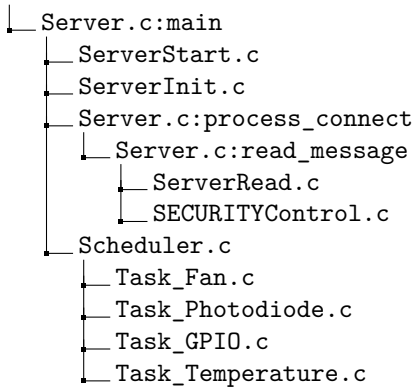

FIGURE 1 : *lancement du serveur* BUNNY

A l'initialisation, bunny met en route les ventilateurs de la façon suivante (selon le pilot) :

- `26 7 9f`
- `26 8 9f`
- `26 9 9f`
- `26 a 97`
- `26 0 ff`
- `26 1 ff`

2 Répertoires

- *Server* : Main
- *Control* : Analyseurs Grammaticaux
- *Scheduler* : Threads Panazol + Wiener

- *Util* : Misc + parser Panazol
- *Board* : ioctls Panazol

3 Comportement

Les 4 thread sont lancés périodiquement selon les fréquences inscrites dans les fichier *include*.

Le controleur corba est bloqué durant les connexion tant que celle-ci ne se lancent pas dans un thread. Ce comportement est hérité de Big.

4 Remontées des messages d'erreur

A priori les erreurs atteignent le contrôleur camera. Comment le contrôleur sécurité peut-il s'inscrire afin que Bunny lui envoit ses erreurs ?

```
Il faudrait que le contrôleur sécurité ait une fonction "CameraMessage" calculée sur
celle du CameraControleur.
Voir dans Camera2.idl:

    // Callback - Messages from the CPU
    void CameraMessage(in Dash::BlockAcceptor::Block data);

Ensuite, il faut envoyer à Bunny l'adresse corba. Dans CameraInterface_BigD.py:

    ##
    ## Send the controller node to the camera
    ## @param name CORBA address of Controller
    def SendControllerOp(self,name):
        self.Print("Send the controller node for %s"%(name))
        try:
            command = "BEGIN\nNODE CONTROLLER %s\nENDMSG"%(self.create_nodes_op(name,"CameraMessage"
            self.send_command(command)
        except Exception,x:
            traceback.print_exc()
            raise Camera2.CameraError("While contacting camera : %s"%str(x))
        return
```