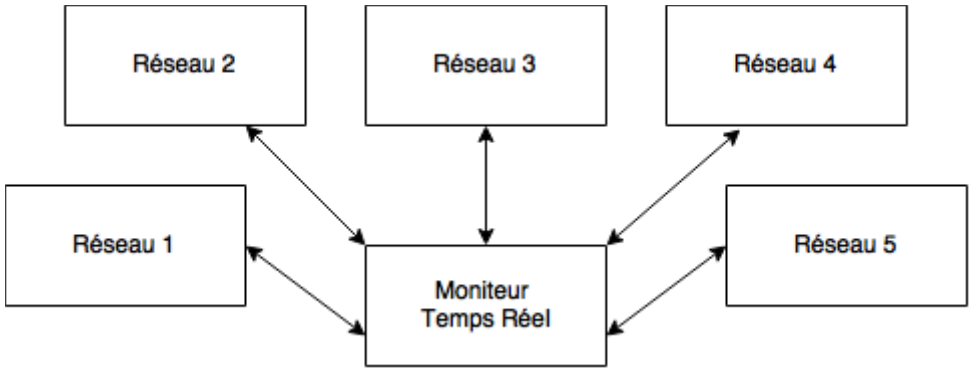
Livrable Evolution Réseau de Petri

Comme le projet livré n'intègre pas la notion de réseau de pétri en couche haute, nous vous livrons ici quelques idées que nous avons pour développer cette fonctionnalité.

Tout d'abord, le but est d'intégrer ROS à cette notion de réseau de pétri. Comme nous avons besoin d'un moniteur temps réel pour faire évoluer le réseau, nous pensons que l'architecture devrait être la suivante :



Chaque réseau correspondrait donc à un nœud ROS, pour lequel les étudiants devront établir la connexion avec le Moniteur Temps Réel. Pour créer des réseaux de Petri, chaque nœud doit pouvoir être endormi ou réveillé pas le moniteur temps réel.

Les réseaux doivent donc avoir une fonction WAIT qui permet d'attendre un jeton et de le consommer lorsqu'il est disponible, une fonction Wait2Let qui permet d'attendre un jeton sans le consommer, et une fonction EAT qui permet comme son nom l'indique de consommer le jeton.
Une fonction SEND permet de libérer un jeton.

Pour utiliser au mieux l'environnement ROS, nous pensons que le mieux serait de créer trois services dans le moniteur Temps Réel, correspondant aux fonctions Wait2Let, Wait et Eat. Pour la fonction SEND, un simple message asynchrone suffit.

Nous utiliserons donc un topic pour faire transiter le message. L'avantage des services est que ceux ci sont des messages synchrones, et sont donc définis par une demande et un acquittement. La fonction EAT est la plus simple à réaliser. Le service doit s'assurer que le jeton concerné est bien disponible, et dans ce cas, changer sa disponibilité et envoyer un acquittement comme quoi tout s'est déroulé sans erreur.

Les fonctions restantes (wait et wait2let) sont les plus difficiles à élaborer. En effet, elles doivent normalement permettre au moniteur Temps réel « d'endormir » les procédures concernées jusqu'à la libération du jeton demandé.

Nous n'avons pas pu tester si les services sont bloquants, c'est à dire envoyer l'acquittement seulement quand le jeton est disponible, et donc voir si le client était bien inactif en attendant la demande. Si c'est le cas, la solution est trouvée.