

Enseignant : C.A. ABID	Année universitaire : 2025-2026
TP 2	
Module : Programmation robotique avec ROS2	

### Exercice 1.

- Créer un nouveau package appelé **pubsub**  
 - Développer un éditeur (**xpm\_pub.cpp**) qui envoie la valeur RPM (de type float) d'une roue au topic **xpm**, et un abonné (**speed\_calc.cpp**) qui à la réception de la valeur RPM, il calcule la vitesse en m/sec et affiche le résultat sur la sortie standard.

On suppose que la valeur RPM est fixée dans l'éditeur par une constante :

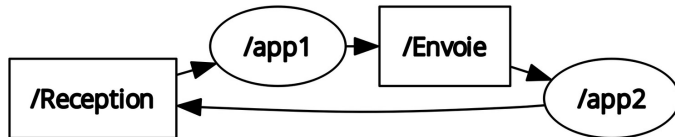
```
constexpr float RPM_VALUE = 10.0;
```

On suppose aussi que le rayon de la roue est fixé dans l'abonné par la constante suivante :

```
constexpr float WHEEL_RADIUS = 12.5 / 100; /// centimeters to meters
```

### Exercice 2.

On se propose de développer deux nœuds ROS permettant d'échanger des messages textuels (programme de type chat), comme illustré par le graphe de calcul suivant.



1) Écrire une classe **ChatNode** dont le constructeur prends trois paramètres :

- Nom du nœud
- Le nom du topic pour la réception des messages
- Le nom du topic pour l'envoi des messages

Cette classe doit permettre :

- L'abonnement au topic de réception afin d'afficher les messages reçus
- La publication de messages sur le topic d'envoi
- La saisie de messages depuis l'entrée standard (clavier) afin de les transmettre
- L'identification de l'émetteur dans chaque message affiché

2) Écrire deux programmes exécutables utilisant la classe **ChatNode** et permettant l'échange bidirectionnel de messages entre eux.

3) Lancer les deux nœuds avec la configuration QoS suivante :

- Fiabilité : **Best Effort**, durabilité **Volatile** (puis **Transient Local**) et Historique : **Keep Last** (profondeur = 1, puis profondeur 10)
- Fiabilité : **Reliable**, durabilité **Volatile** (puis **Transient Local**) et Historique : **Keep Last** (profondeur = 1, puis profondeur 10)

Pour chacune des configurations, essayer d'effectuer :

- un envoi de messages très rapide,
- la suspension puis la reprise d'un nœud,
- le démarrage tardif d'un nœud après l'envoi de messages.

### Exercice 3.

On se propose de développer un nœud permettant de contrôler le déplacement de la tortue dans **turtlesim** à partir du clavier.

Écrire une classe **KeyboardTeLeopNode** dont le constructeur prend un paramètre :

- Le nom du nœud

Cette classe doit permettre :

- La publication de messages de type **geometry\_msgs/msg/Twist** sur le topic **/turtle1/cmd\_vel**
- La lecture des commandes clavier pour contrôler le déplacement de la tortue
- L'association des touches clavier aux mouvements suivants :
  - Avancer
  - Reculer
  - Tourner à gauche
  - Tourner à droite
  - Arrêter le mouvement

1) Écrire un programme exécutable permettant :

- Le lancement du nœud **KeyboardTeLeopNode**
- La récupération des entrées clavier en temps réel
- L'envoi des commandes de vitesse correspondantes vers la tortue

2) Tester le programme en lançant successivement :

- Le nœud **turtlesim\_node**
- Le nœud développé précédemment