

Conceitos do ROS

Grafo de computação:

- **O ROS é uma rede de processamento peer-to-peer**
- **Conceitos Básicos:**
 - **Master** - Nodo principal que gerencia a rede.
 - **Nodo** - Processo em execução.
 - **Serviço de Parâmetros distribuídos** - Parâmetros acessíveis por todos os nodos.
 - **Serviços** - Comunicação direta entre dois processos
 - **Tópicos** - Comunicação multicast entre vários processos.
 - **Bags** - Sistema de log de mensagens

Conceitos do ROS

Grafo de computação:

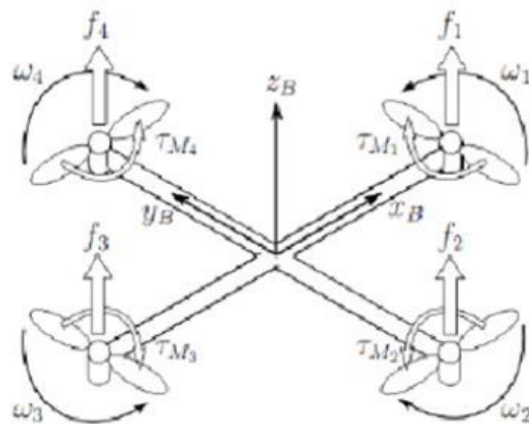
- **O ROS é uma rede de processamento peer-to-peer**
- **Conceitos Básicos:**
 - **Master** - Nodo principal que gerencia a rede.
 - **Nodo** - Processo em execução.
 - **Serviço de Parâmetros distribuídos** - Parâmetros acessíveis por todos os nodos.
 - **Serviços** - Comunicação direta entre dois processos
 - **Tópicos** - Comunicação multicast entre vários processos.
 - **Bags** - Sistema de log de mensagens

Tópico

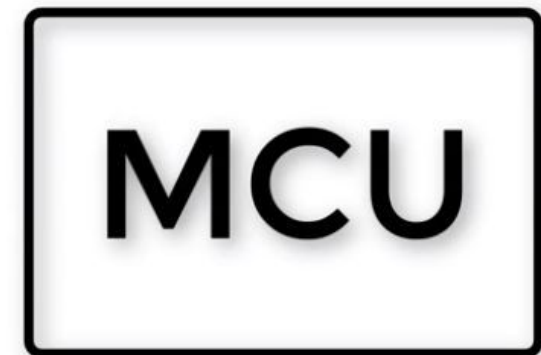
- Funciona como **um barramento**, onde os nodos trocam informações.
- Utiliza a semântica Anuncia/Pública/Se escreve(**Advertise/Publish/Subscribe**)
- Mensagens **fortemente tipadas** conforme arquivos de descrição de mensagem do ROS.
- **Comunicação não bloqueante.**
- Pode haver múltiplos nodes publicando e se inscrevendo em um tópico.



Tópico



`rpm_rotors_x`



Tópico

```
score http://ros-VirtualBox:11311/
re service [/rosout]


s@ros-VirtualBox: ~
heta=[0,000000]

@ros-VirtualBox: ~
-----
keys to move the turtle.

s@ros-VirtualBox: ~
278259
141144
62605822086
ocity: 1.0
locity: 0.40000000596

409027
486954
69005811214
ocity: 1.0
locity: 0.40000000596

VirtualBox:~$
[1.0, 0.0, 0.0] [0.0, 0.0, 0.4]
^CUnhandled exception in thread started by
sys.excepthook is missing
lost sys.stderr
ros@ros-VirtualBox:~$ rostopic pub -r 5 turtle1/cmd_vel geometry_msgs/Twist -- '
[1.0, 0.0, 0.0]' '[0.0, 0.0, 0.4]'
□
```



DICA: Use o TAB para completar os comandos

Trabalhando com tópicos:

- Iniciar o ROS: (abra um terminal)
 - `$ roscore`
- Verificar nodos em execução: (abra um novo terminal de testes)
 - `$ rosnode list`
- Executar simulador de tartaruga (abra um novo terminal)
 - `$ rosrun turtlesim turtlesim_node`
- Verificar nodos em execução (no terminal de testes)
 - `$ rosnode list`
- Executar controle de tartaruga (abra um novo terminal)
 - `$ rosrun turtlesim turtle_teleop_key`
- Verificando arquitetura do ROS (no terminal de testes)
 - `$ rosrun rqt_graph rqt_graph`

DICA: Use o TAB para completar os comandos

Trabalhando com tópicos:

- Verifique o que está sendo publicado: (no terminal de testes)
 - `$ rostopic echo /turtle1/cmd_vel`
- Atualize a interface gráfica do rqt_graph
 - Clicando no botão de update.
- Descobrindo o tipo de mensagem publicado em um tópico
 - `$ rostopic type /turtle1/cmd_vel`
- Descobrindo os tipos de dados de uma mensagem
 - `$ rosmmsg show geometry_msgs/Twist`
- Publicando msgs (`rostopic pub [topic] [msg_type] [args]`)
 - `$ rostopic pub -1 /turtle1/cmd_vel geometry_msgs/Twist -- '[2.0, 0.0, 0.0]' '[0.0, 0.0, 1.8]'`
- Publicando várias msgs repetidamente
 - `$ rostopic pub /turtle1/cmd_vel geometry_msgs/Twist -r 1 -- '[2.0, 0.0, 0.0]' '[0.0, 0.0, -1.8]'`

DICA: Use o TAB para completar os comandos

Trabalhando com tópicos:

- Atualize a interface gráfica do rqt_graph
 - Clicando no botão refresh no canto superior esquerdo.
- Medindo a frequência de publicação de msgs (terminal de testes)
 - `$ rostopic hz /turtle1/pose`
- Plotando os dados dos tópicos
 - `$ rosrn rqt_plot rqt_plot`