Assinatura de Tópicos no ROS 2

Walter Fetter Lages

fetter@ece.ufrgs.br

Universidade Federal do Rio Grande do Sul

Escola de Engenharia

Departamento de Sistemas Elétricos de Automação e Energia

ENG10052 Laboratório de Robótica

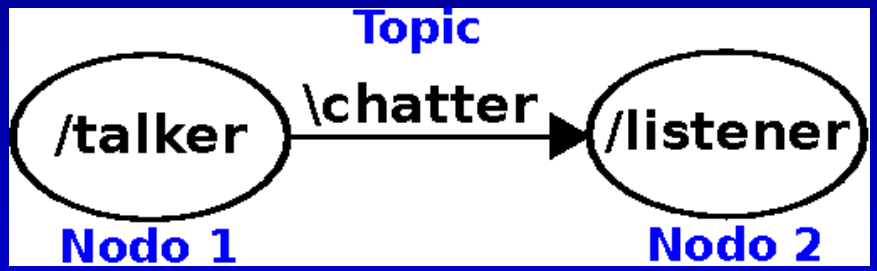
Introdução

Nodo: processo do S.O. hospedeiro

Tópico: mecanismo de comunicação entre nodos do tipo *publisher/subscriber*

Mensagem: dados publicados nos tópicos

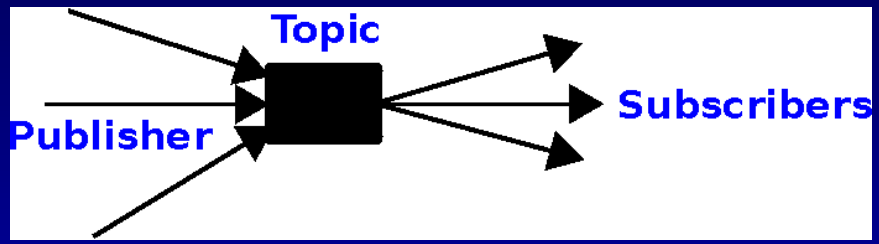
Gráfico de computação: Representa a comunicação entre os nodos através de tópicos



UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Tópicos

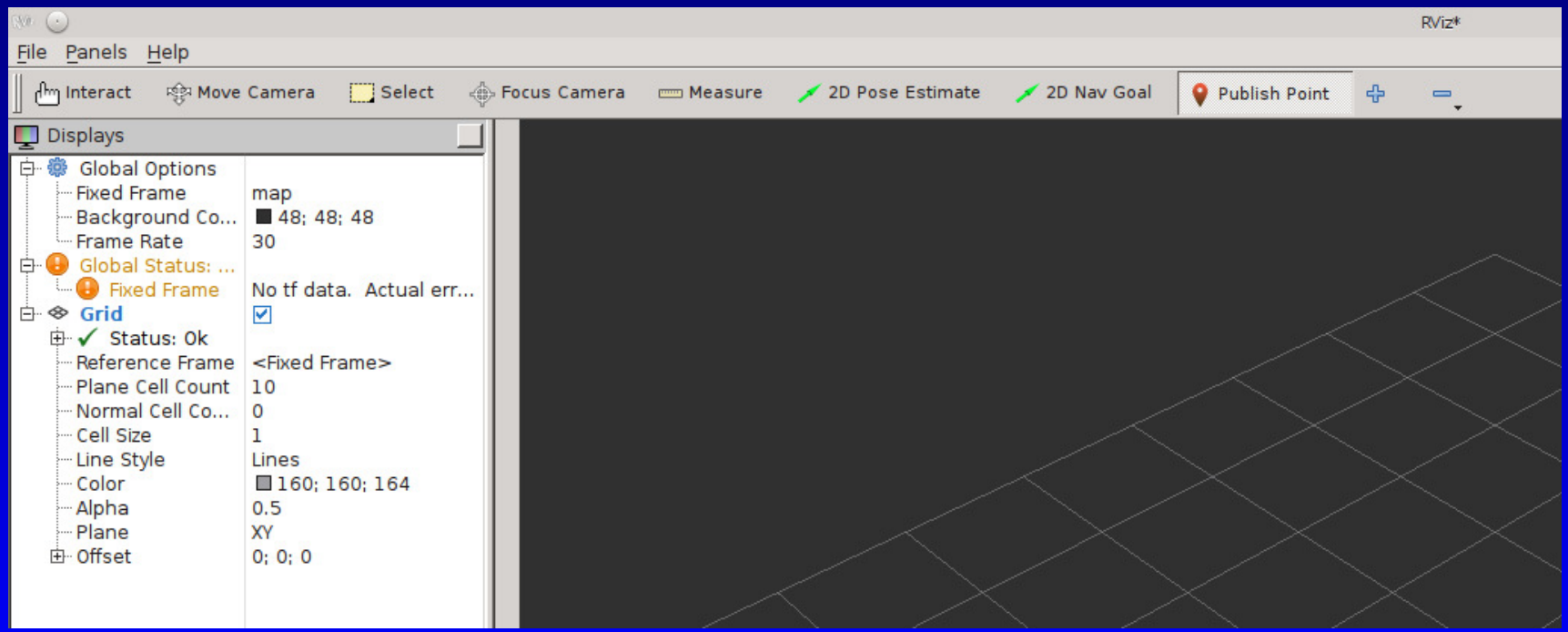
- Nodos podem publicar mensagens em tópicos
- Cada tópico pode ter vários publicadores e assinantes



- Cada nodo pode publicar ou assinar vários tópicos
- Publicadores e assinantes não sabem da existência um dos outros
- A ordem de execução não é garantida
- Comunicação assíncrona

Exemplo

- O RViz2 publica no tópico `/clicked_point` as coordenadas do *click* do mouse
- Ferramenta Publish Point
  - Só funciona com pontos do *grid*
  - Reduzir o tamanho da célula para 0.001
  - Aumentar o número de células para 10000



UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

Tópico /clicked_point

- Verificar o tipo da mensagem

rviz2 &

ros2 topic info /clicked_point

- Verificar a descrição da mensagem

ros2 interface show geometry_msgs/msg/PointStamped

UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

geometry_msgs/msg/PointStamped

```
# This represents a Point with reference coordinate frame and timestamp

std_msgs/Header header
        builtin_interfaces/Time stamp
                int32 sec
                uint32 nanosec
        string frame_id
Point point
        float64 x
        float64 y
        float64 z
```

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Verificação com `ros2 topic`

- Pode-se verificar o que é publicado nos tópicos com o comando

---

ros2 topic echo /clicked_point

- Obviamente, nesse caso, só será publicado algo se for clicado em algum ponto
- Será criado um pacote para assinar o tópico `/clicked_point` e exibir os dados na tela

Criação do Pacote

- Criar o pacote:

```
cd ~/colcon_ws/src
ros2 pkg create --build-type ament_cmake --dependencies
    rclcpp geometry_msgs --node-name click_subscriber
    eng10026_subscriber
```

- `package.xml` deve ser editado para configurar os detalhes de documentação e incluir dependências
- Editar `CMakeLists.txt` para descomentar e ajustar as *tags* para compilação e instalação do pacote

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

CMakeLists.txt

- Editar `CMakeLists.txt` para descomentar e ajustar as *tags*:

```
add_executable(click_subscriber
  src/click_subscriber.cpp)

ament_target_dependencies(click_subscriber
  rclcpp geometry_msgs)

install(TARGETS click_subscriber
  DESTINATION lib/${PROJECT_NAME})

install(DIRECTORY launch config
  DESTINATION share/${PROJECT_NAME})
```

Código Fonte do Nodo

- Editar o arquivo `click_subscriber.cpp` com o código fonte no diretório `src`
- Compilar com os comandos:

---

cd ~/colcon_ws

colcon build –-symlink–install

- O executável estará em `install/eng10026_subscriber/lib/eng10026_subscriber/click_subscriber`

Código Fonte do Nodo

```
#include <rclcpp/rclcpp.hpp>
#include <geometry_msgs/msg/point_stamped.hpp>

class ClickSubscriber: public rclcpp::Node
{
  public:
  ClickSubscriber(void);
  void show(void);

  private:
  rclcpp::Subscription<geometry_msgs::msg::PointStamped>::
      SharedPtr clickSub_;
  double point_[3];
  void clickCB(const geometry_msgs::msg::PointStamped::SharedPtr
      click);
};
```

Código Fonte do Nodo

```
ClickSubscriber::ClickSubscriber(void): Node("click_subscriber")
{
  using std::placeholders::_1;

  clickSub_=create_subscription<geometry_msgs::msg::PointStamped
    >("clicked_point",100,std::bind(&ClickSubscriber::clickCB,this,
    _1));
}
```

Código Fonte do Nodo

```
void ClickSubscriber::clickCB(const geometry_msgs::msg::
        PointStamped::SharedPtr click)
{
   point_[0]=click->point.x;
   point_[1]=click->point.y;
   point_[2]=click->point.z;

   show();
}

void ClickSubscriber::show(void)
{
        std::cout << "point=";
        for(int i=0;i < 3;i++) std::cout << point_[i] << " ";
        std::cout << std::endl;
}
```

UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

Código Fonte do Nodo

```
int main(int argc,char* argv[])
{
  rclcpp::init(argc,argv);
  rclcpp::spin(std::make_shared<ClickSubscriber>());
  rclcpp::shutdown();
  return 0;
}
```

Execução do Nodo

- Chamando diretamente o executável:

```
install/eng10026_subscriber/lib/eng10026_subscriber/
    click_subscriber
```

- Usando o comando `ros2 run`:

```
ros2 run eng10026_subscriber click_subscriber
```

- Usando um arquivo de *launch*:

```
ros2 launch eng10026_subscriber click.launch.xml
```

- *Launch* com Rviz:

```
ros2 launch eng10026_subscriber display.launch.xml
```

UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

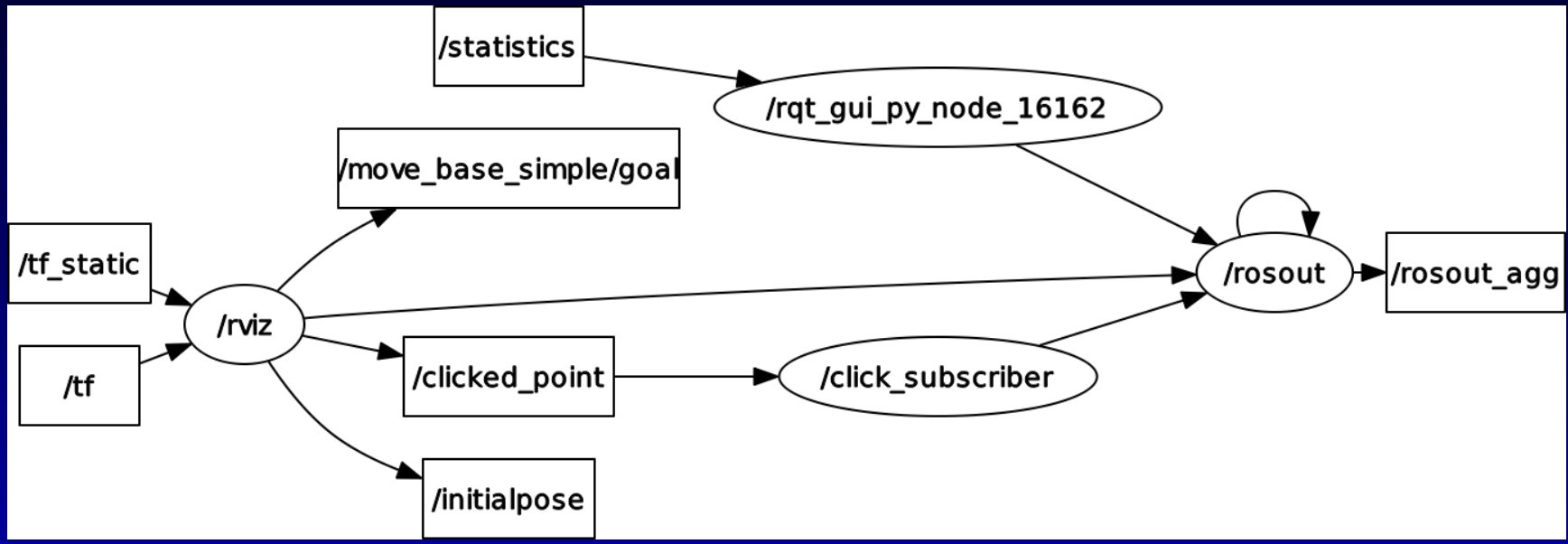
click.launch.xml

```
<launch>
  <node name="click_subscriber" pkg="eng10026_subscriber" exec=
    "click_subscriber"
    output="screen" />
</launch>
```

display.launch.xml

```
<launch>
    <include file="$(find-pkg-share eng10026_subscriber)/launch/
    click.launch.xml" />
    <node name="rviz" pkg="rviz2" exec="rviz2" args="-d $(find-
    pkg-share eng10026_subscriber)/config/display.rviz" />
</launch>
```

Gráfico de Computação



Instalação do Pacote

```
cd ~/colcon_ws/src
git clone –b $ROS_DISTRO http://git.ece.ufrgs.br/eng10026/
      eng10026_subscriber
cd ~/colcon_ws
colcon build ––symlink–install
source ~/colcon_ws/install/setup.bash
```

Pacote eng10026_subscriber

```
colcon_ws/
└─src/
  └─eng10026_subscriber/
    ├─CMakeLists.txt
    ├─config/
    │ └─click.rviz
    ├─launch/
    │ ├─click.launch.xml
    │ └─display.launch.xml
    ├─package.xml
    └─src/
      └─click_subscriber.cpp
```