ROS

Communication infrastructure in ROS

Session 03

Kalana Ratnayake

31/10/2020

Session Plan

Session 01

Robotics and ROS

- Introduction to basic concepts of Robotics
- Introduction to ROS
- When and How to use ROS in robotics

Session 02

Communication infrastructure in ROS

- Getting started with ROS
- Publisher Subscriber (C++)
- Publisher Subscriber (Python)

Session Plan (cont..)

Session 03

Communication infrastructure in ROS (Part 2)

- Standard and Custom message, service and action definitions
- Client Server(C++)
- Client Server (Python)
- Action client Action server (C++)
- Action client Action server (Python)

Session 04

Robot specific infrastructure of ROS

- Introduction to Robot Geometry library
- Introduction to Robot Description language
- Introduction to Gazebo

planned based on slides titled 240AR060 Master's degree in Automatic Control and Robotics – Introduction to ROS by Jan Rosell / Carlos Rosales

Session Plan (cont..)

Session 05

Development tools available in ROS

- rosrun, roslaunch
- rostopic, rosservice
- rqt_graph
- rqt_tf_tree
- Catkin build system

planned based on slides titled 240AR060 Master's degree in Automatic Control and Robotics – Introduction to ROS by Jan Rosell / Carlos Rosales

Communication infrastructure in ROS

Publisher & Subscriber

Publisher & Subscriber

- Create a workspace for this workshop

```
mkdir -p ~/ros_workshop/src
cd ~/ros_workshop/
catkin build
```

- Finally

```
source devel/setup.bash
```

- Create a package

```
catkin_create_pkg session3_pubsub std_msgs rospy roscpp
```

Publisher & Subscriber

- Create a msg folder inside session3_pubsub

  session3_pubsub/msg

- Create a custom.mgs inside

  String first_name

  String last_name

  uint8 age

- Modify CmakeLists.txt

Publisher (C++)

- Create a file in -> session3_pubsub/src as publisher.cpp
- Open the -> /Pub-Sub/C++/publisher2.txt
- And copy the content
- Change

  ros::init(argc, argv, "talker");

  TO

  ros::init(argc, argv, “publisher");

Subscriber (C++)

- Create a file in -> session3_pubsub/src as subscriber.cpp
- Open the -> /Pub-Sub/C++/subscriber2.txt
- And copy the content
- Change

  ros::init(argc, argv, “listener");

  TO

  ros::init(argc, argv, “subscriber");

Building the nodes

- Open session3_pubsub/CMakeLists.txt:

```
add_executable(publisher src/ publisher.cpp)
target_link_libraries(publisher ${catkin_LIBRARIES})
add_dependencies(publisher session3_pubsub_generate_messages_cpp)

add_executable(subscriber src/subscriber.cpp)
target_link_libraries(subscriber ${catkin_LIBRARIES})
add_dependencies(subscriber session3_pubsub_generate_messages_cpp)
```

Building the nodes

- Go to root of the workspace

```
cd ~/ros_workshop/
```

```
catkin build
```

Running the nodes

- Open a terminal and run

  ```
  roscore
  ```

- Open a 2nd terminal in the workspace root and run

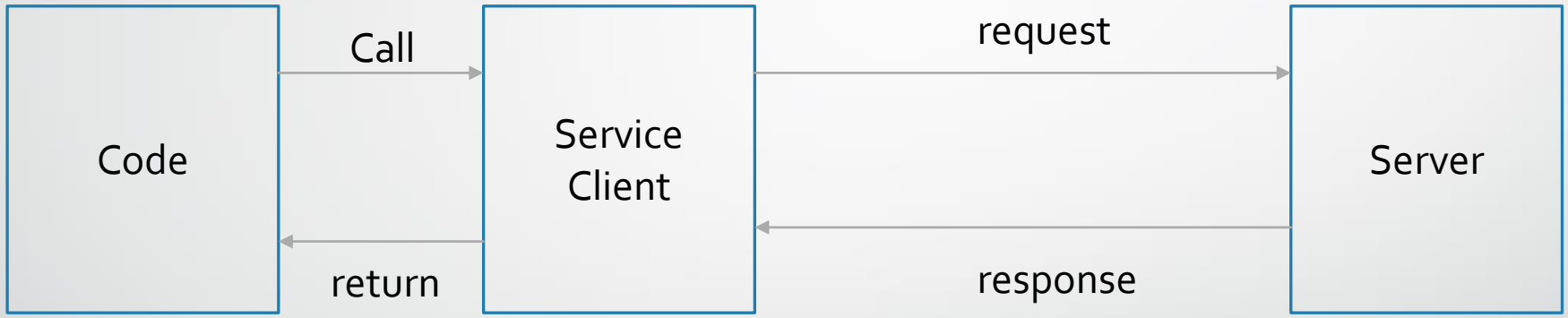
  ```
  source   devel/setup.bash
  rosrun session3_pubsub publisher
  ```

- Open a 3rd terminal in the workspace root and run

  ```
  source   devel/setup.bash
  rosrun session3_pubsub subscriber
  ```

Communication infrastructure in ROS

Server & Client


Code
Call
return
Service
Client
request
response
Server

Server & Client

- Move to ros workspace

```
cd ~/ros_workshop/
catkin build
```

- Finally

```
source devel/setup.bash
```

- Create a package

```
catkin_create_pkg session3_cliserver std_msgs rospy roscpp
```

Server & Client

- Create a srv folder inside session3_cliserver

  session3_cliserver/srv

- Create a custom.srv inside

  uint8 a

  uint8 b

  ---

  uint16 result

- Modify CmakeLists.txt

Server (C++)

- Create a file in -> session3_cliserver/src as server.cpp
- Open the -> Client-Server/C++/server.txt
- And copy the content
- Change

  ros::init(argc, argv, “add_two_ints_server");

  TO

  ros::init(argc, argv, “server");

Client (C++)

- Create a file in -> session3_cliserver/src as client.cpp
- Open the -> Client-Server/C++/client.txt
- And copy the content
- Change

  ros::init(argc, argv, “add_two_ints_client");

  TO

  ros::init(argc, argv, “client");

Building the nodes

- Open session3_cliserver/CMakeLists.txt:

```
add_executable(server src/ server.cpp)
target_link_libraries(server ${catkin_LIBRARIES})
add_dependencies(server session2_server_client_gencpp)

add_executable(client src/client.cpp)
target_link_libraries(client ${catkin_LIBRARIES})
add_dependencies(client session2_ server_client _ gencpp)
```

Building the nodes

- Go to root of the workspace

```
cd ~/ros_workshop/
```

```
catkin build
```

Running the nodes

- Open a terminal and run

  ```
  roscore
  ```

- Open a 2nd terminal in the workspace root and run

  ```
  source   devel/setup.bash
  rosrun session3_cliserver server
  ```

- Open a 3rd terminal in the workspace root and run

  ```
  source   devel/setup.bash
  rosrun session3_cliserver client
  ```

Server (python)

- Create a file in -> session3_cliserver/scripts as server.py
- Open the -> Client-Server/python/server.txt
- And copy the content
- Change

  rospy.init_node(add_two_ints_server', anonymous=True)

  TO

  rospy.init_node('server')
- Open a terminal and Run

  chmod +x server.py

Client (python)

- Create a file in -> session3_cliserver/scripts as client.py
- Open the -> Client-Server /python/client.txt
- And copy the content

- Open a terminal and Run

  chmod +x client.py

Building the nodes

- Open session3_cliserver/CMakeLists.txt:

```
catkin_install_python(PROGRAMS scripts/server.py scripts/client.py
  DESTINATION ${CATKIN_PACKAGE_BIN_DESTINATION}
)
```

- Go to root of the workspace

```
cd  ~/ros_workshop/
catkin  build
```

Running the nodes

- Open a terminal and run

  ```
  roscore
  ```

- Open a 2nd terminal in the workspace root and run

  ```
  source   devel/setup.bash
  rosrun session3_cliserver client.py
  ```

- Open a 3rd terminal in the workspace root and run

  ```
  source   devel/setup.bash
  rosrun session3_cliserver server.py
  ```

Thank you