Towards simple robotics with the Robot Operating System (ROS)

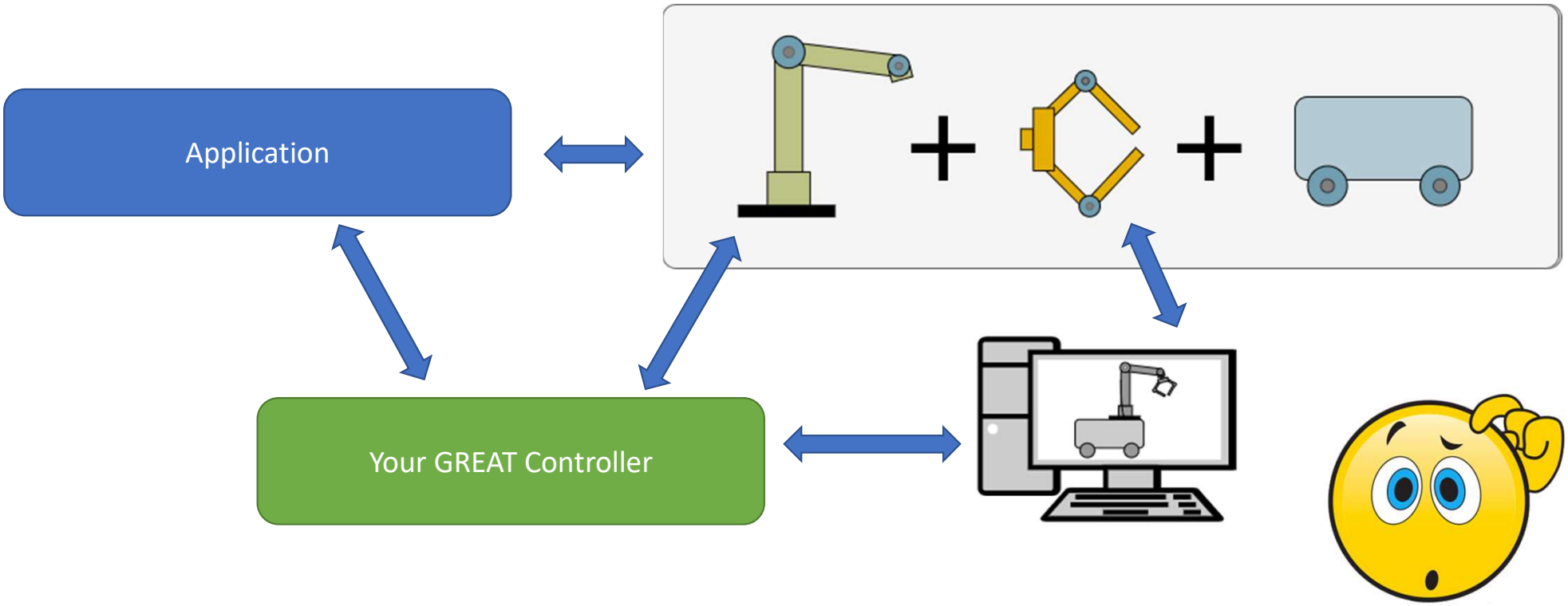
Maciej Bednarczyk, Ph.D
Robotics Research Engineer @ ICube

m.bednarczyk@unistra.fr

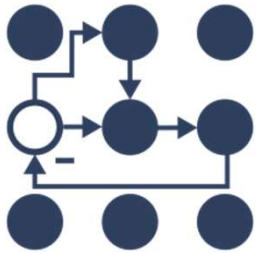
École d'ingénieurs
Télécom Physique
Université de Strasbourg

ROS 2

ROS2 Control - Motivation



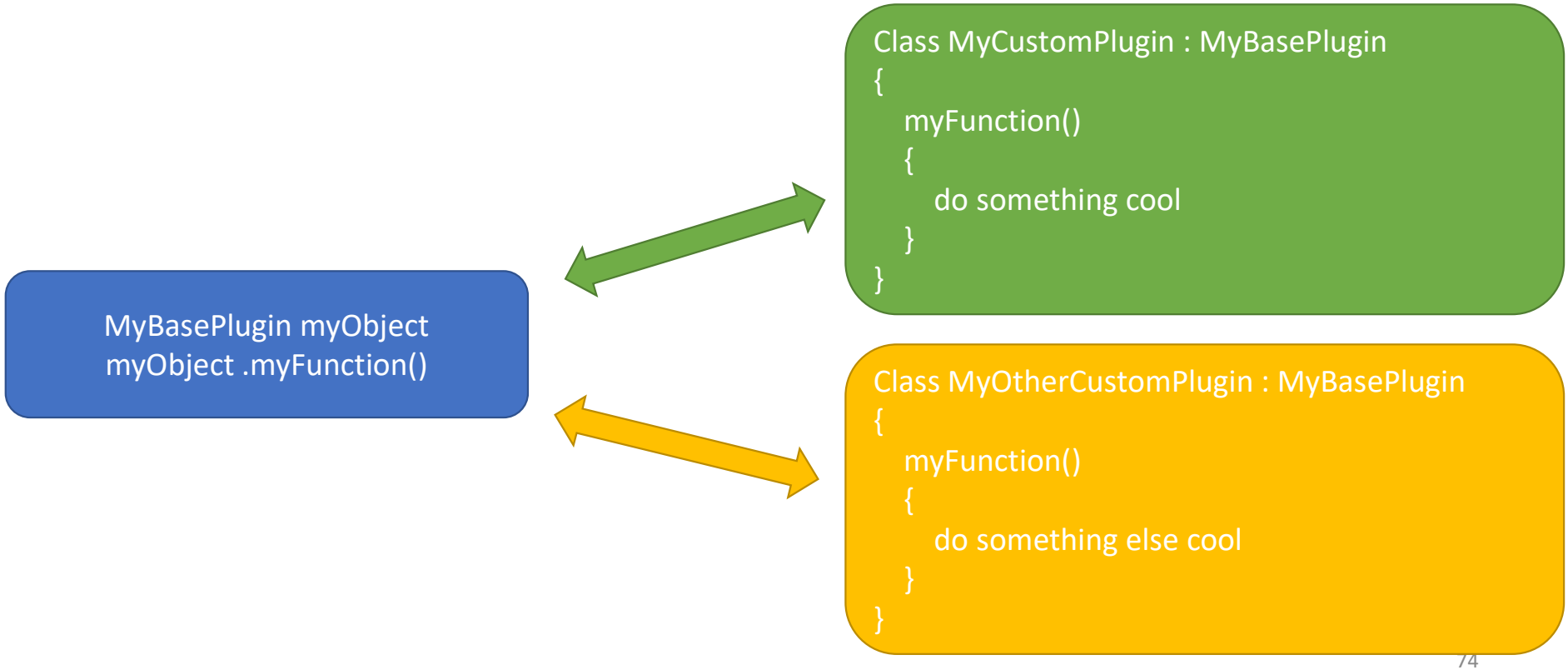
ROS2 Control



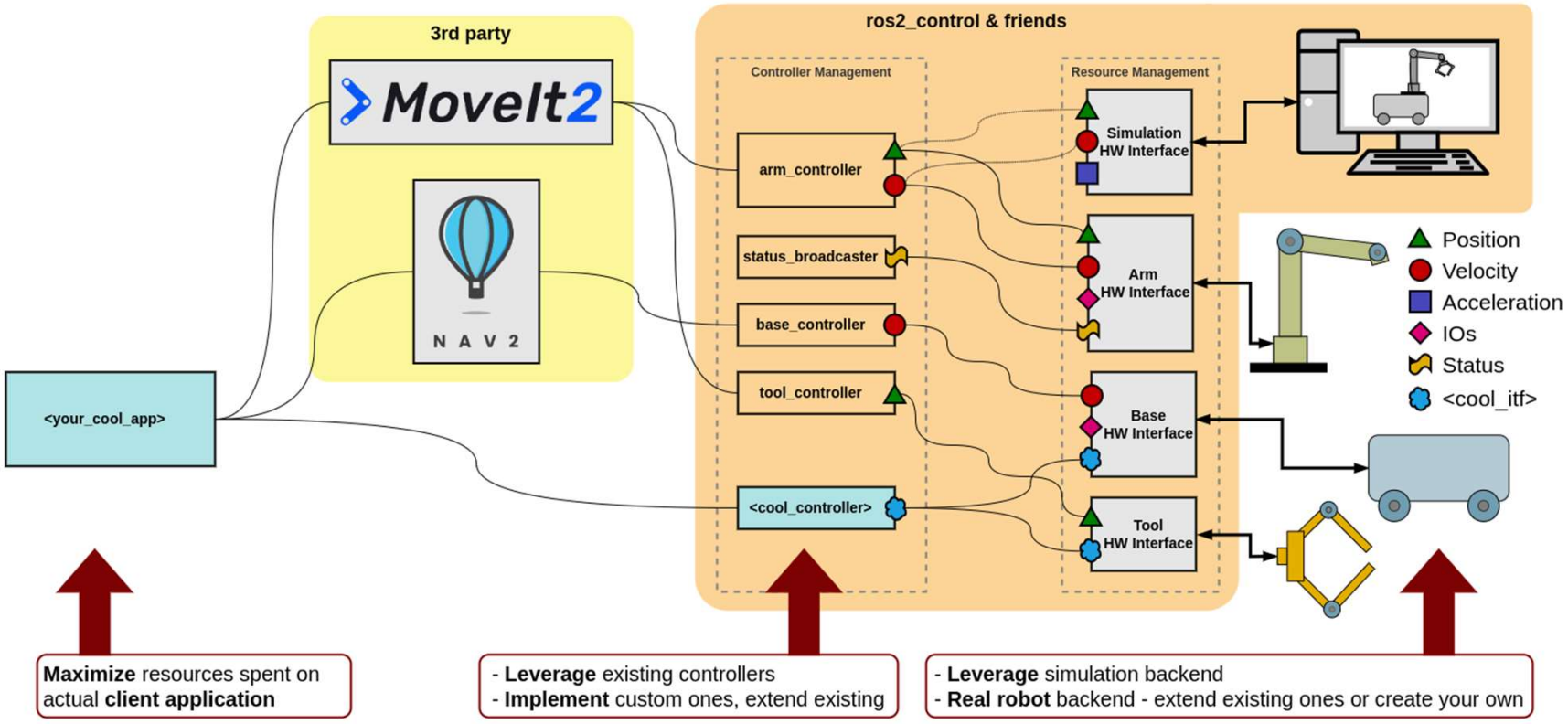
- Framework for real-time robot control
- Designed for general robotics applications
- Abstraction Layer for simple integration of hardware and controllers
- Standard Interfaces and Robot Agnostic design
- Modular design and easy configuration using description files
- Focus on applications and reuse available hardware drivers and controllers
- Easy switch between real robot and simulation
- Use of standard interfaces to integrate 3rd party ROS2 packages

ROS2 and Plugins

Plugins: dynamically loadable classes loaded from a runtime library



ROS2 Control – Overview



ROS2 Control – Overview

Hardware Abstraction Components

System

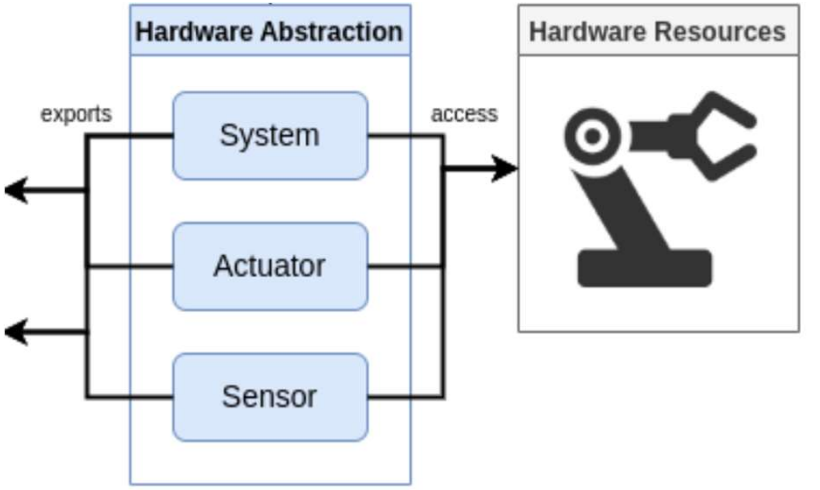
- Complex multi-DoF hardware with sensors
- Used for single communication channel (driver, SDK, …)
- Read and Write component

Sensor

- Read only component

Actuator

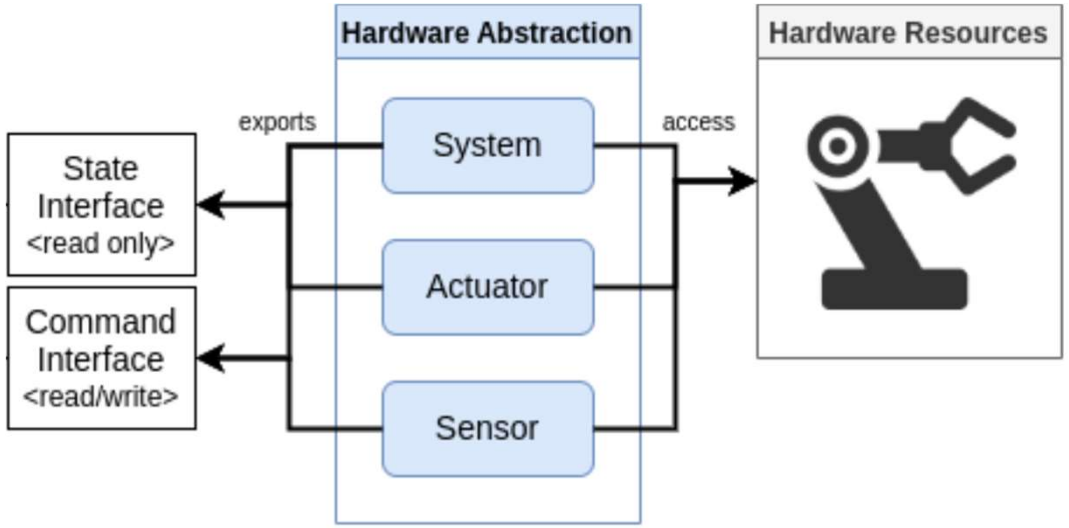
- Simple 1 DoF hardware
- Associated to a single joint
- Read (optional) and Write component



ROS2 Control – Overview

State Interface

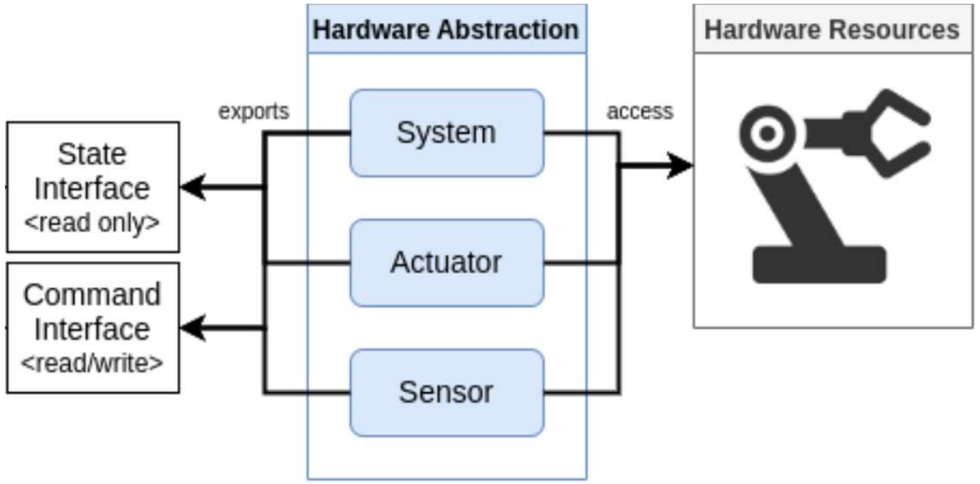
- System state data
- Hardware Abstraction
- Pointer to **READ-ONLY** data
- Configuration using description files



ROS2 Control – Overview

Command Interface

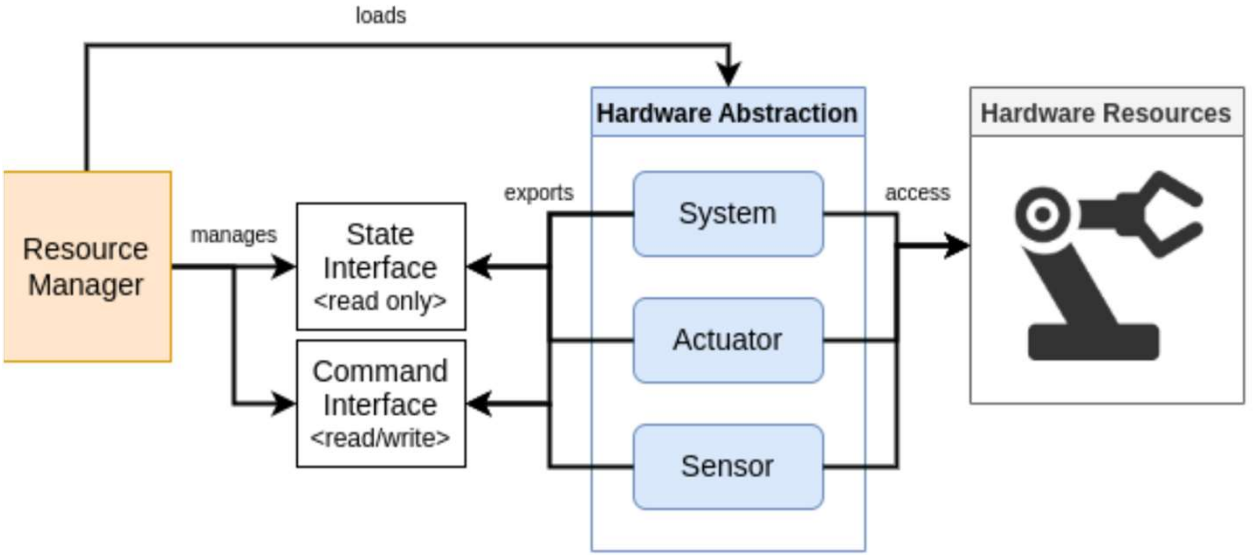
- Command data
- Hardware Abstraction
- Pointer to **READ** and **WRITE** data
- **EXCLUSIF** write access
- Configuration using description file



ROS2 Control – Overview

Resource Manager

- Hardware Interface Management
- Hardware Abstraction Management



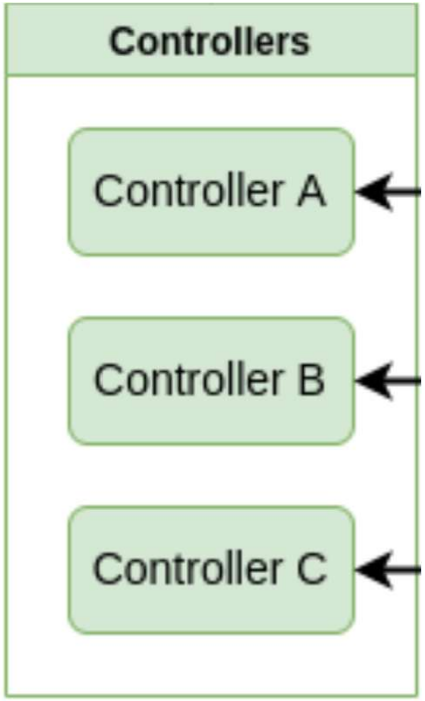
ROS2 Control – Overview

Controllers

- Data processing and/or system control
- Loaded as Plugins
- State Machine:
  - Unconfigured
  - Inactive
  - Active
  - Finalized

  → Ensure correct setup before execution

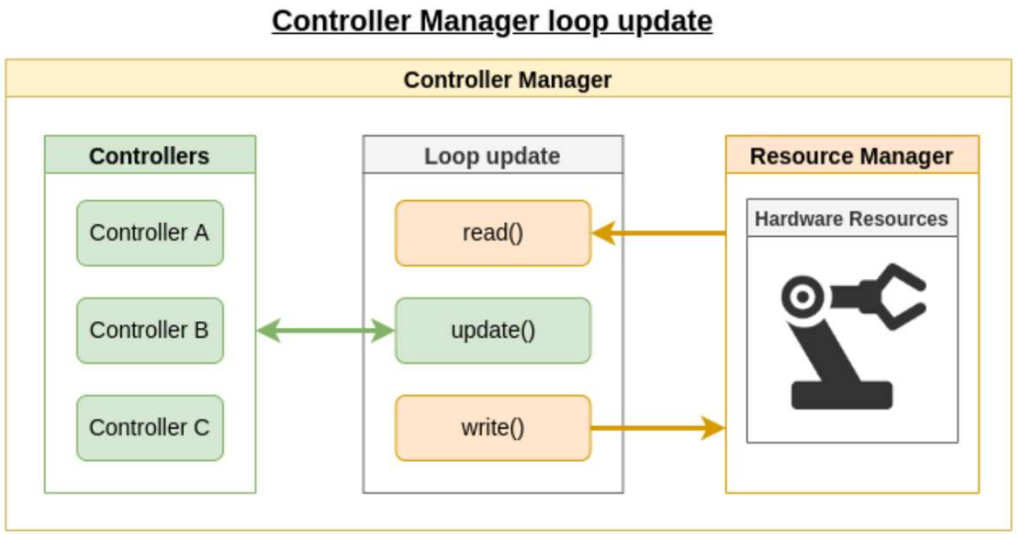
  → Load, restart and switch controllers



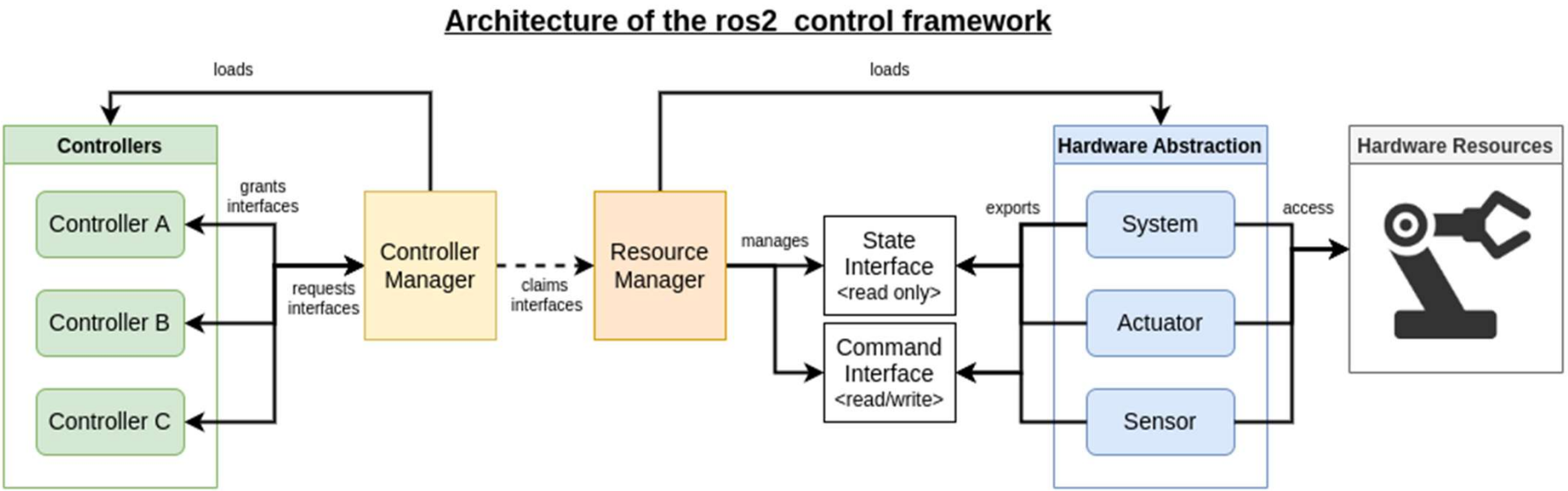
ROS2 Control – Overview

Controller Manager

- Main component of *ros2_control*
- Controller loading
- Interface management
- Execution
- User entry-point
- Node without loop to facilitate integration into 3rd party systems



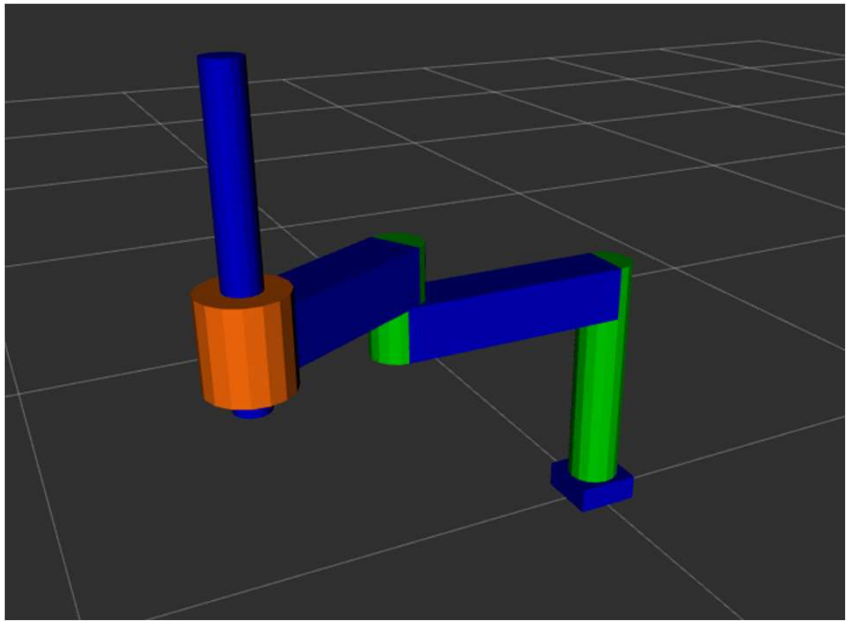
ROS2 Control – Overview



Scara tutorial *ros2_control*

Content of the tutorial :

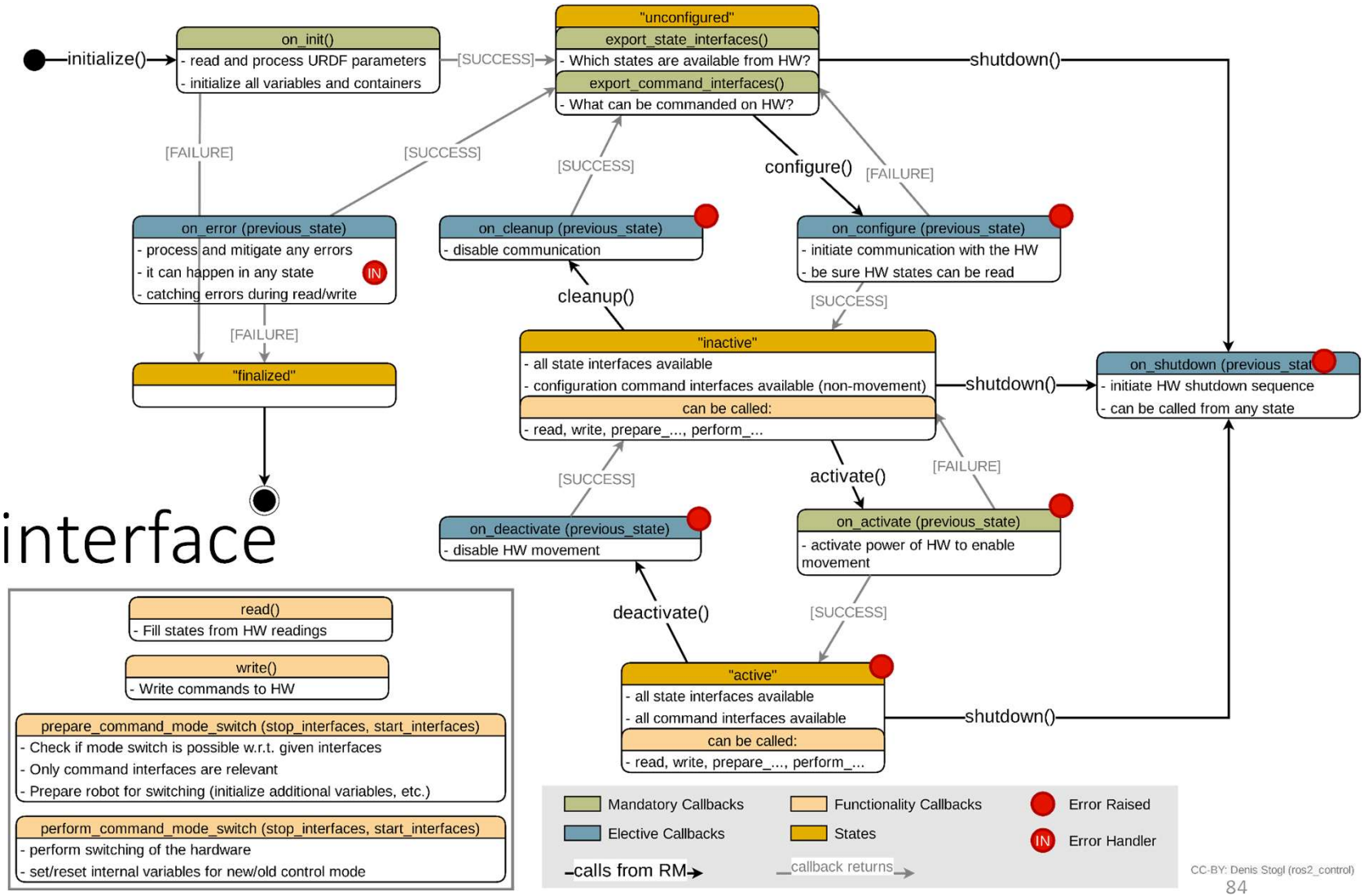
- Write a URDF description of a simple SCARA manipulator
- Launch and interact with the SCARA robot
- Write a custom hardware interface for the SCARA robot
- Write a custom controller for the SCARA robot
- Set up the SCARA manipulator to run with ros2_control and Gazebo



https://github.com/ICube-Robotics/scara_tutorial_ros2

SCAN ME

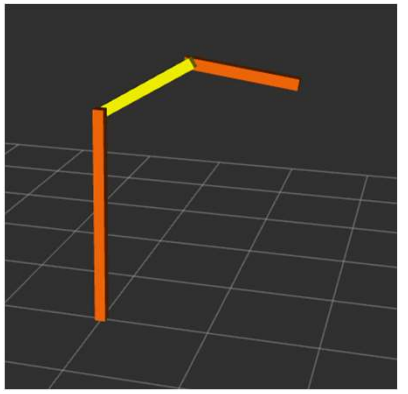
Hardware interface



More *ros2_control* examples

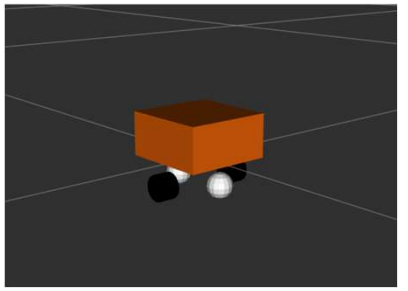
ros2_control_demos

```
$ cd ros2_ws
$ git clone https://github.com/ros-controls/ros2_control_demos
$ colcon build
$ source install/setup.bash
```



rrbot

```
$ ros2 launch ros2_control_demo_example_1 rrbot.launch.py
```

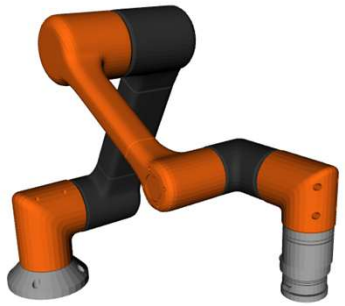


diffbot

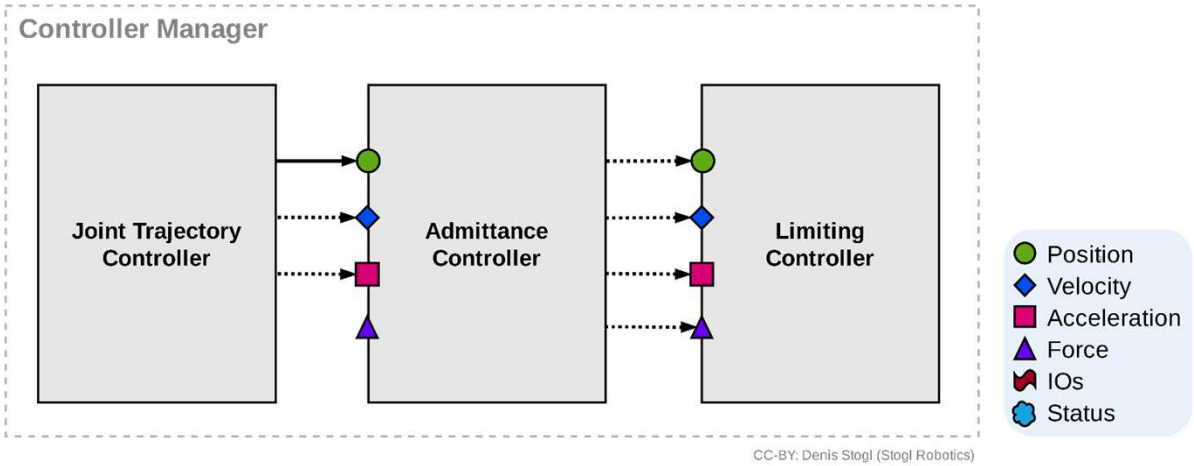
```
$ ros2 launch ros2_control_demo_example_2 diffbot.launch.py
```

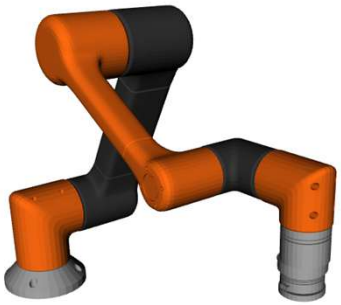
r6bot

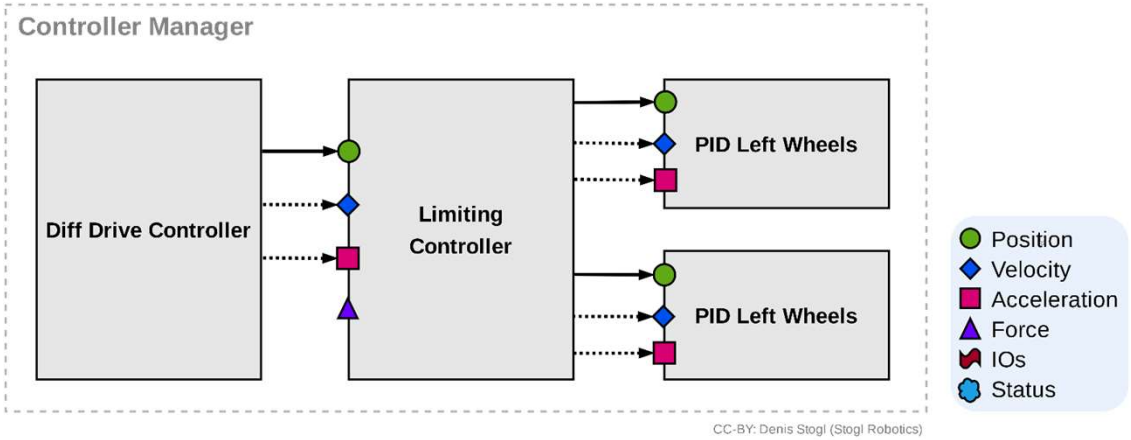
```
$ ros2 launch ros2_control_demo_example_7 r6bot_controller.launch.py.launch.py
```

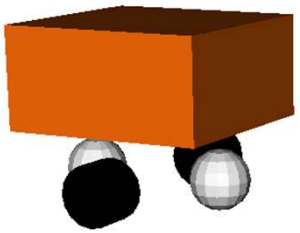


Controller Chaining



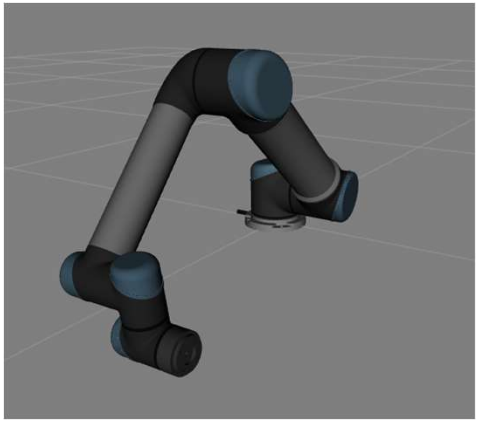


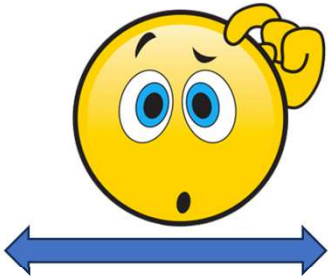




Allergic to C++ or want to use Python only libs ?

ros2_control setup





Robotics Toolbox Python

Allergic to C++ or want to use Python only libs ?

Pytroller project

Install *pytroller*

```
$ cd ros2_ws
$ git clone https://github.com/ICube-Robotics/pytroller
$ colcon build
$ source install/setup.bash
```

Create a *pytroller*

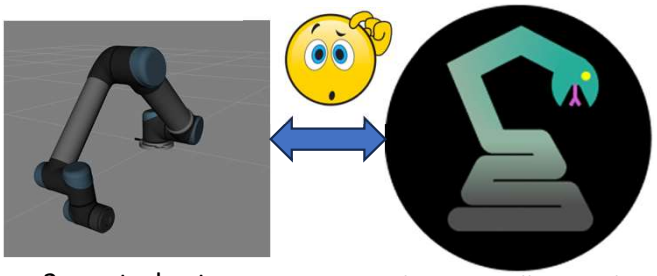
```
$ cd ros2_ws
$ ros2 pytroller create my_pytroller
```

```
# my_pytroller/script/my_pytroller_logic_impl.py

from math import cos, sin
def pytroller_logic_impl(period, states, commands, msg, params):
    commands[b'joint1/effort'] = msg.data[0]
    commands[b'joint2/effort'] = msg.data[1]
    return commands
```

```
$ rosdep install --ignore-src --from-paths . -y -r
$ colcon build
$ source install/setup.bash
```

Allergic to C++ or want to use Python only libs ?



Install the *pytroller* demo config

```
$ cd ros2_ws
$ git clone https://github.com/ICube-Robotics/pytroller_examples
$ rosdep install --ignore-src --from-paths . -y -r
$ pip3 install roboticstoolbox-python==1.0.3
$ colcon build
$ source install/setup.bash
```

```
$ ros2 launch rtb_bringup rtb.launch.py
```

Run *slider_publisher*

```
$ sudo apt install ros-humble-slider-publisher
$ ros2 launch slider_publisher example.launch file:=Twist.yaml
```