

ROSCon
Madrid 2023



Introducción a `ros2_control`

Sai Kishor Kothakota
Madrid, 2023

Index

- PAL Robotics Company
- ROS2 Control
 - Hardware Components
 - Resource Manager
 - Controller Manager
 - Controllers
 - Semantic Components
- Conclusion

Company

-  Founded in 2004
-  Located in Barcelona
-  +20 nationalities
-  ~100 people
-  80% Engineers | 10% Ph.D.
-  Robots sales +35 countries

Our robots



Business units

Intralogistics

INDUSTRY | RETAIL | HEALTHCARE

Platforms for automating transportation of goods, inventory robots.



Legged

RESEARCH | UNIVERSITIES

Humanoid service Platforms for state-of-the-art research



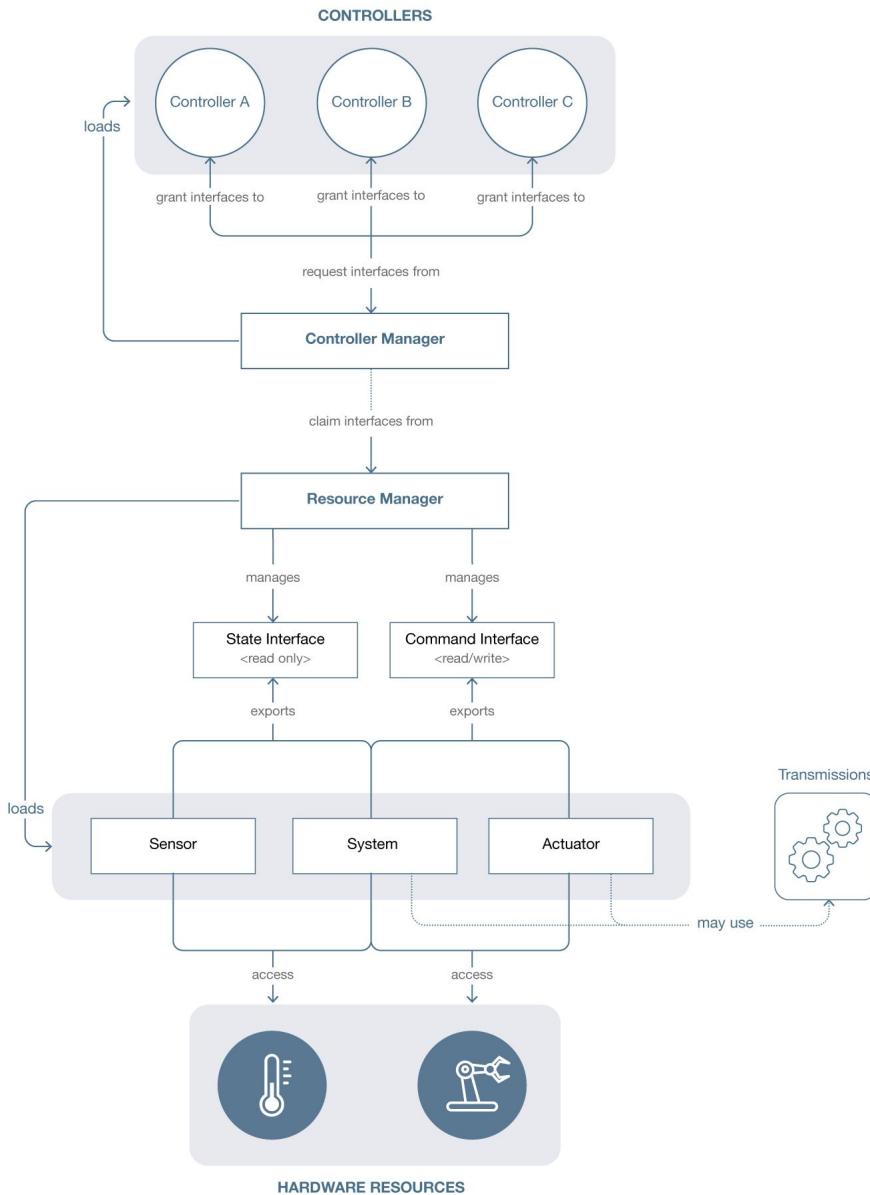
Mobile Interaction

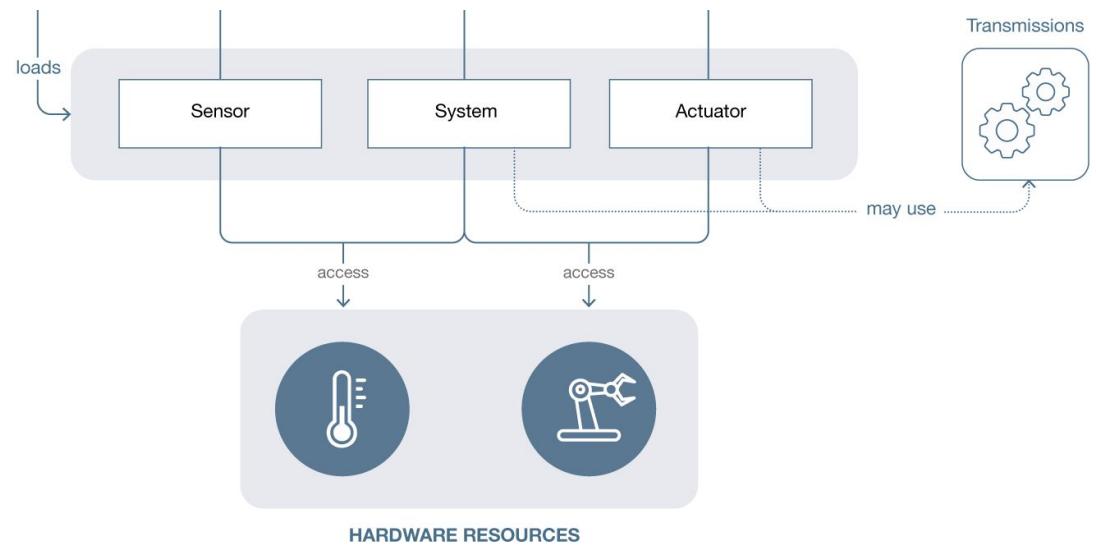
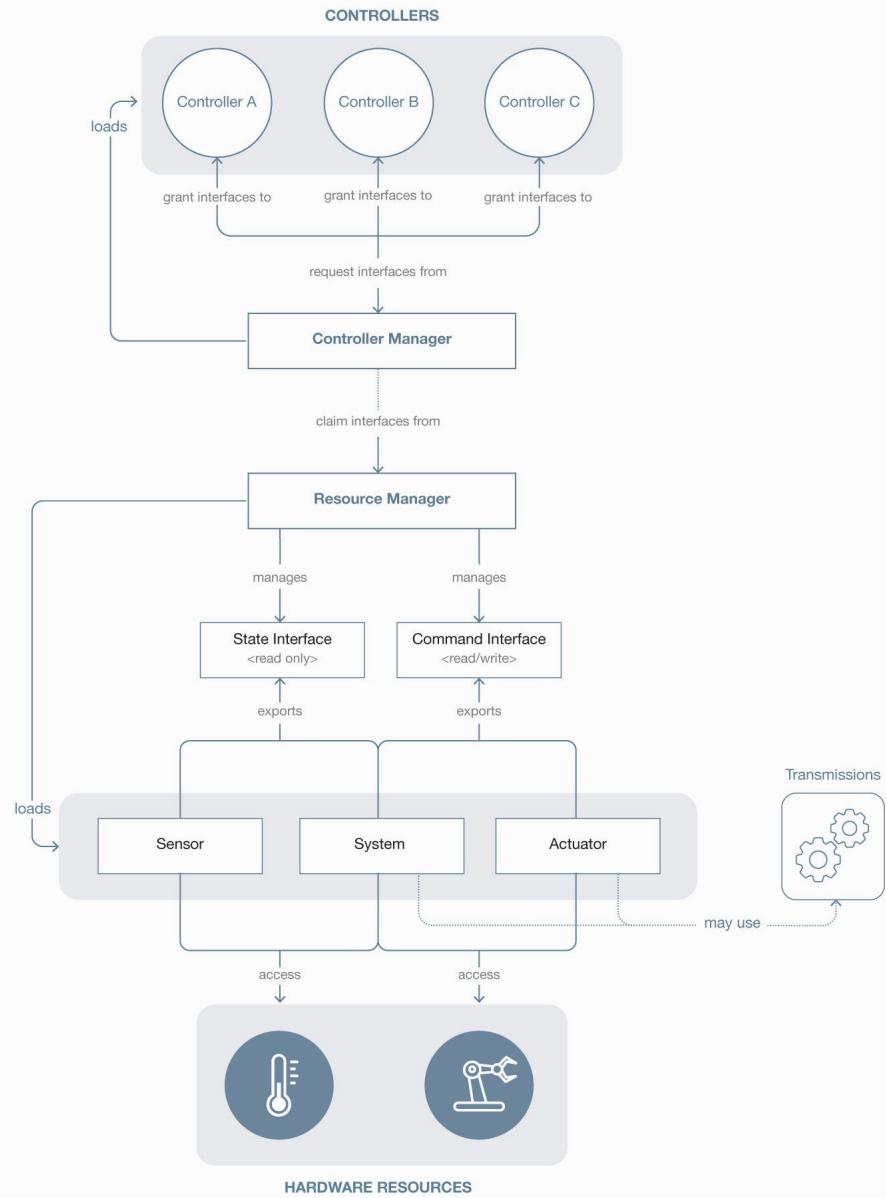
RESEARCH | INDUSTRY | HEALTHCARE

ARI & TIAGO products and services for industry & research.



Introducción a ros2_control





Hardware Components

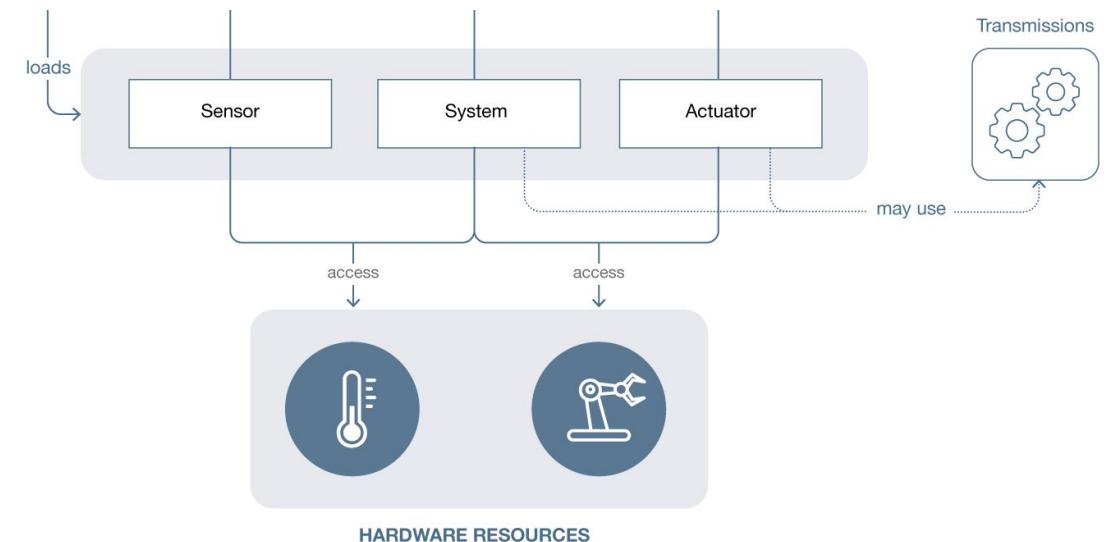
- Provides an abstraction from ros2_control to realize the communication with the physical hardware
- The components are exported as plugins
- Resource manager is responsible for loading them and maintaining their lifecycle

Types

Sensor

Actuator

System

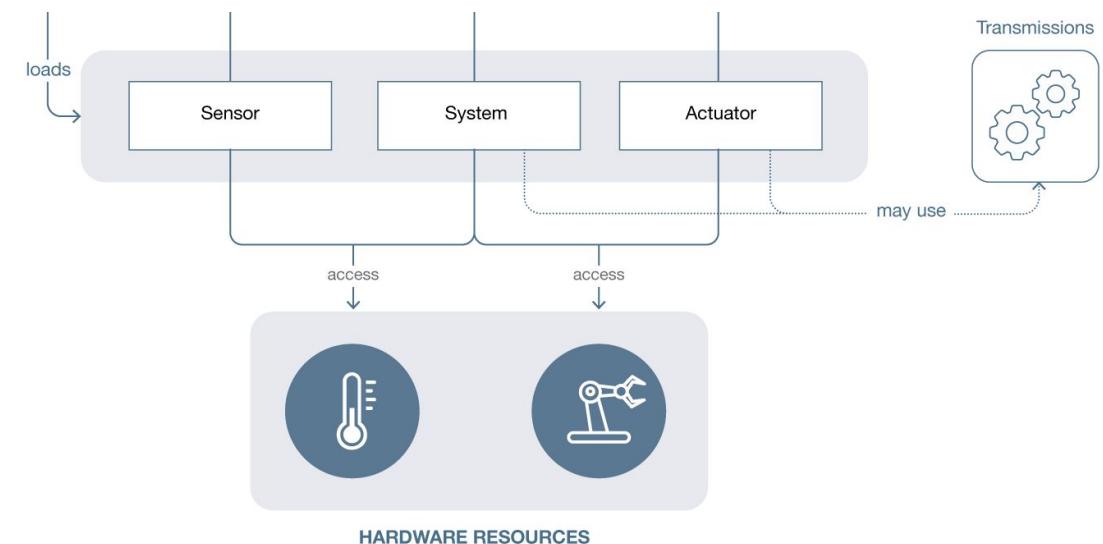


Sensor

Actuator

System

```
<ros2_control name="ForceSensor" type="sensor">
<hardware>
  <plugin>vendor_specific/ForceSensorHardware</plugin>
  <param name="foo">0.43</param>
</hardware>
<sensor name="my_force_sensor">
  <state_interface name="force"/>
  <param name="frame_id">rrbot_tcp</param>
  <param name="force_limit">100</param>
</sensor>
</ros2_control>
```

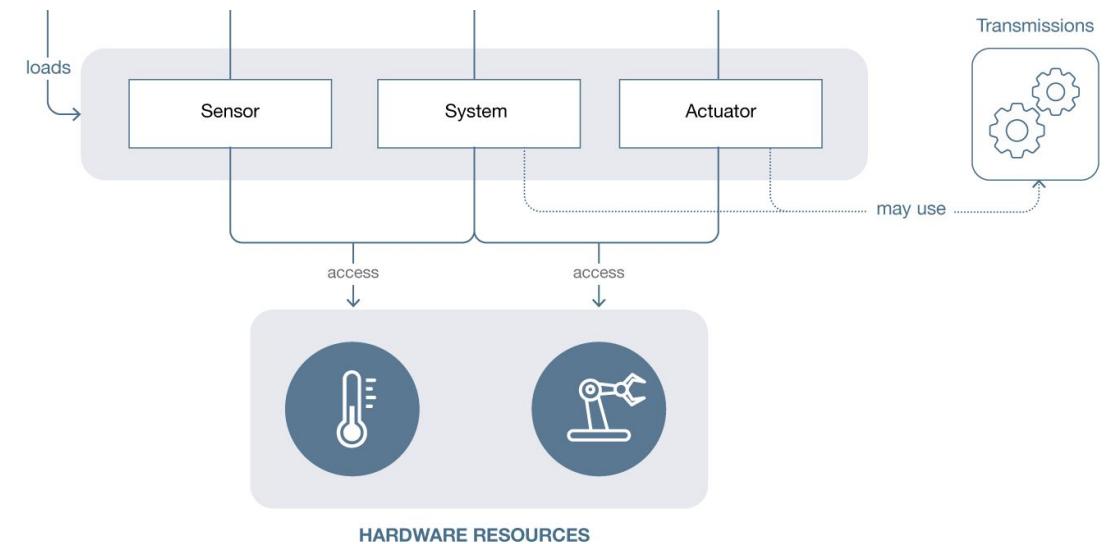


Sensor

Actuator

System

```
<ros2_control name="Gripper" type="actuator">
  <hardware>
    <plugin>vendor_specific/PositionActuatorHardware</plugin>
    <param name="foo">1.23</param>
    <param name="bar">3</param>
  </hardware>
  <joint name="gripper_joint">
    <command_interface name="position">
      <param name="min">0</param>
      <param name="max">50</param>
    </command_interface>
    <state_interface name="position"/>
    <state_interface name="velocity"/>
  </joint>
</ros2_control>
```

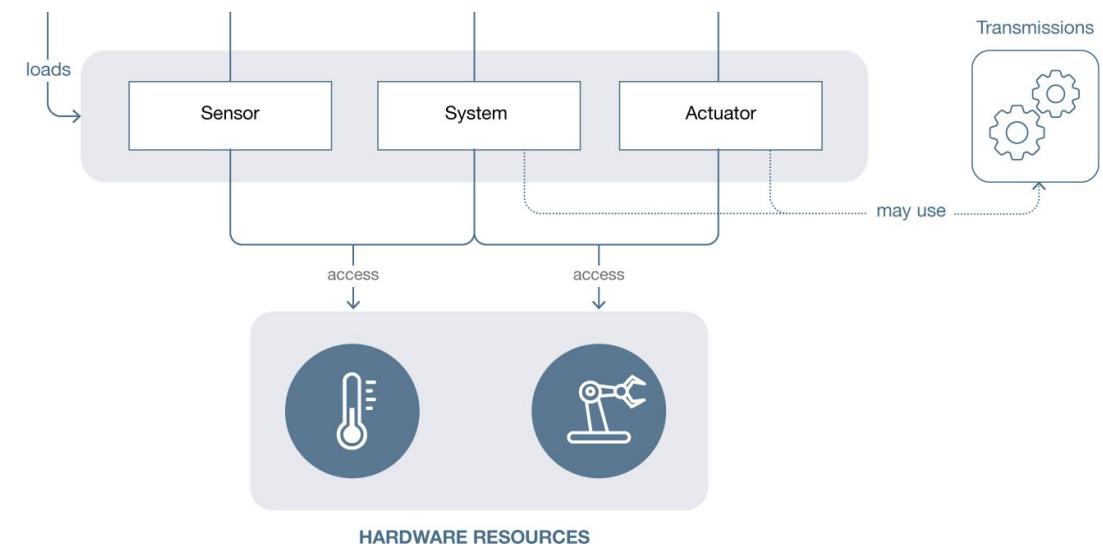


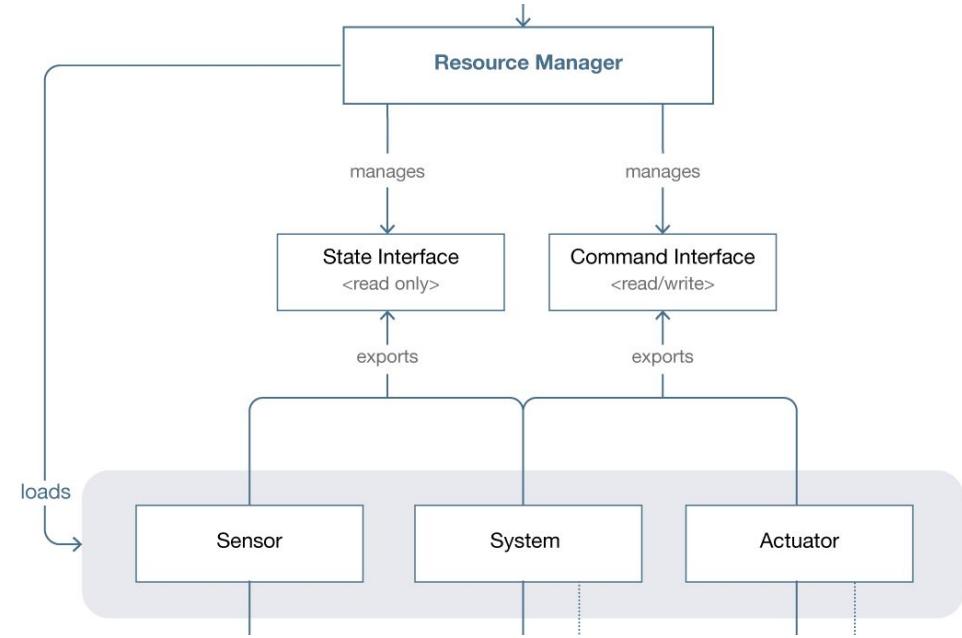
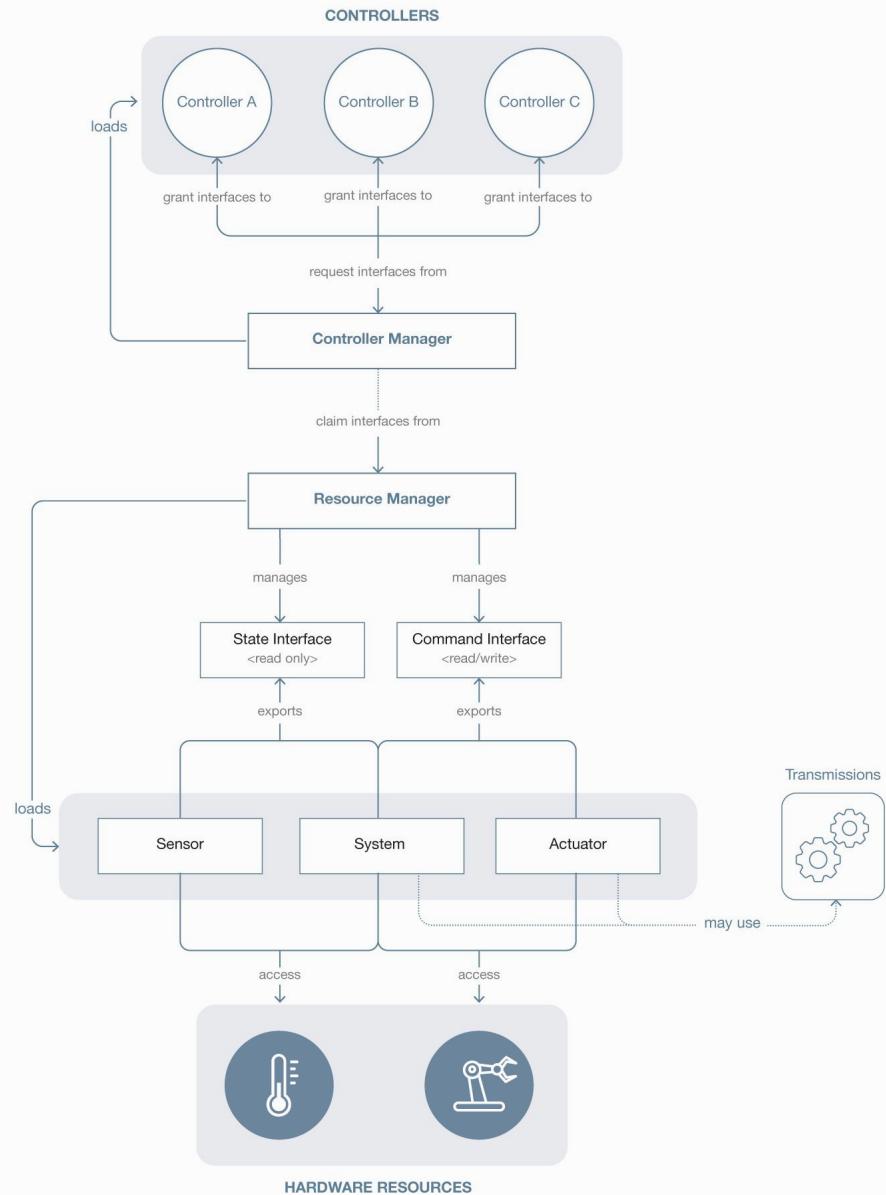
Sensor

Actuator

System

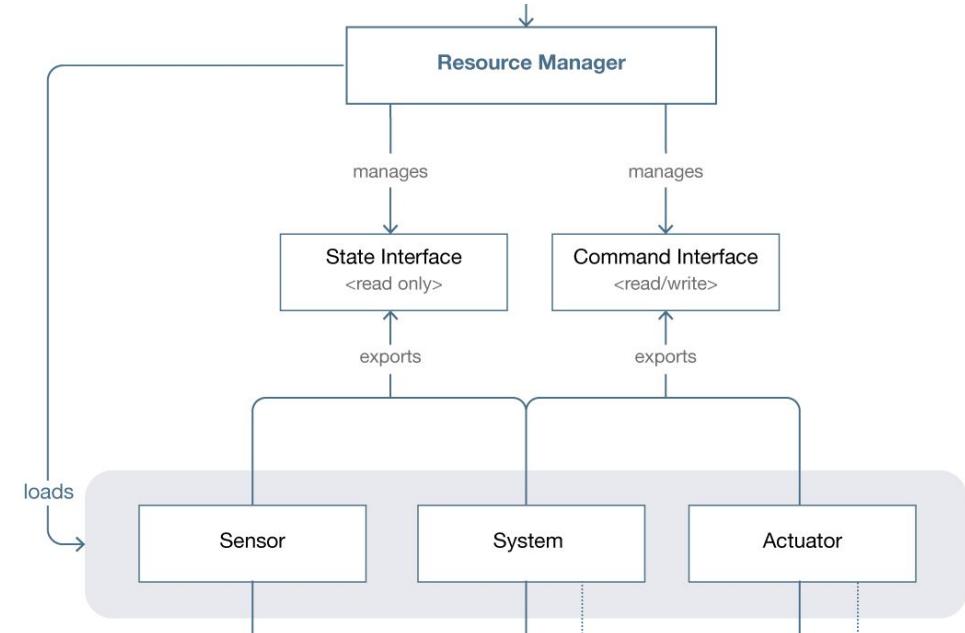
```
<ros2_control name="MyRobotArm" type="system">
  <hardware>
    <plugin>vendor_specific/PositionOnlyHardware</plugin>
    <param name="foo">2</param>
    <param name="bar">2</param>
  </hardware>
  <joint name="joint1">
    <command_interface name="position" />
    <state_interface name="position" />
  </joint>
  <joint name="joint2">
    <command_interface name="position" />
    <state_interface name="position" />
  </joint>
</ros2_control>
```





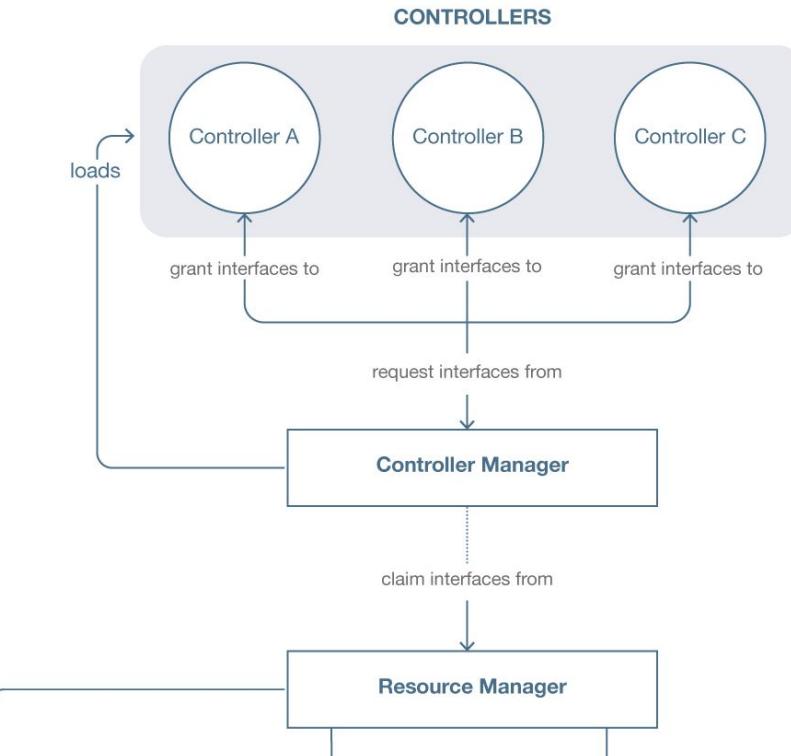
Resource Manager

- Abstracts physical hardware and its drivers (called hardware components)
- Loads components as plugin
- Responsible for loading them, maintaining their lifecycle, and components' state and command interfaces
- Why this level of abstraction?
 - Reuse of implemented hardware components
 - Flexible hardware applications for both state and command interfaces

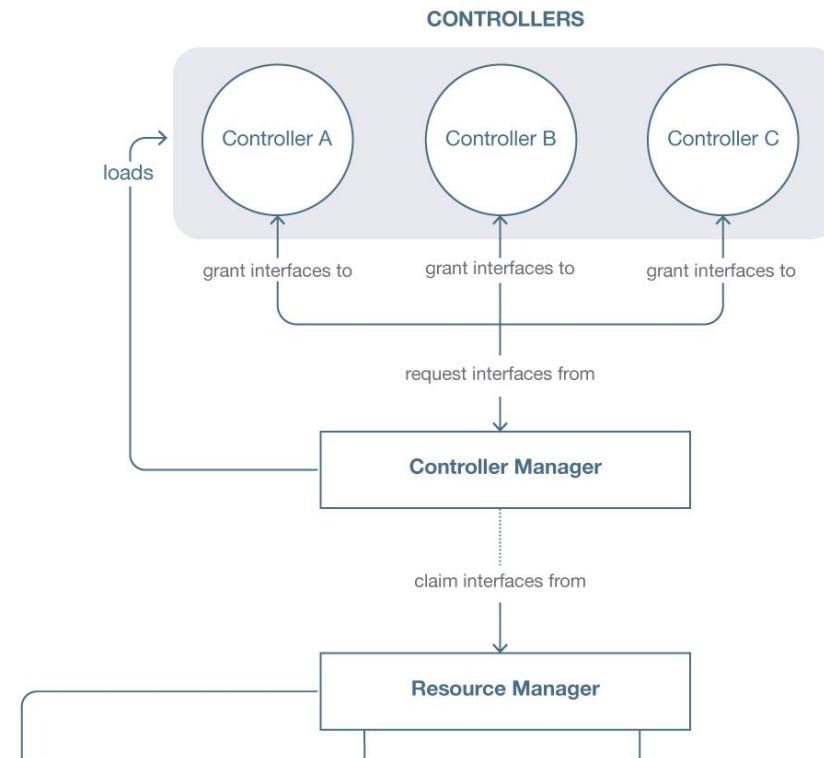
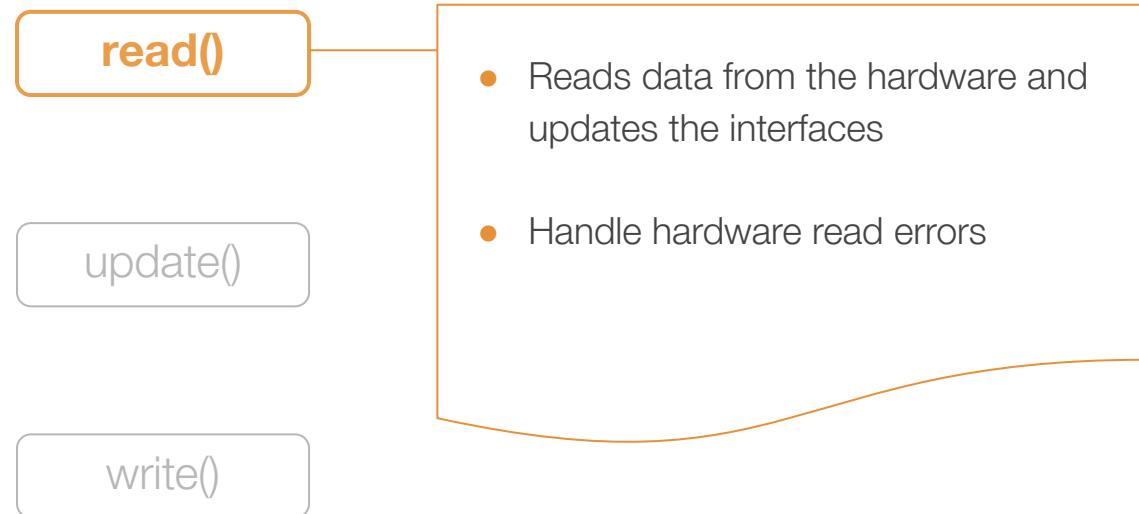


Controller Manager

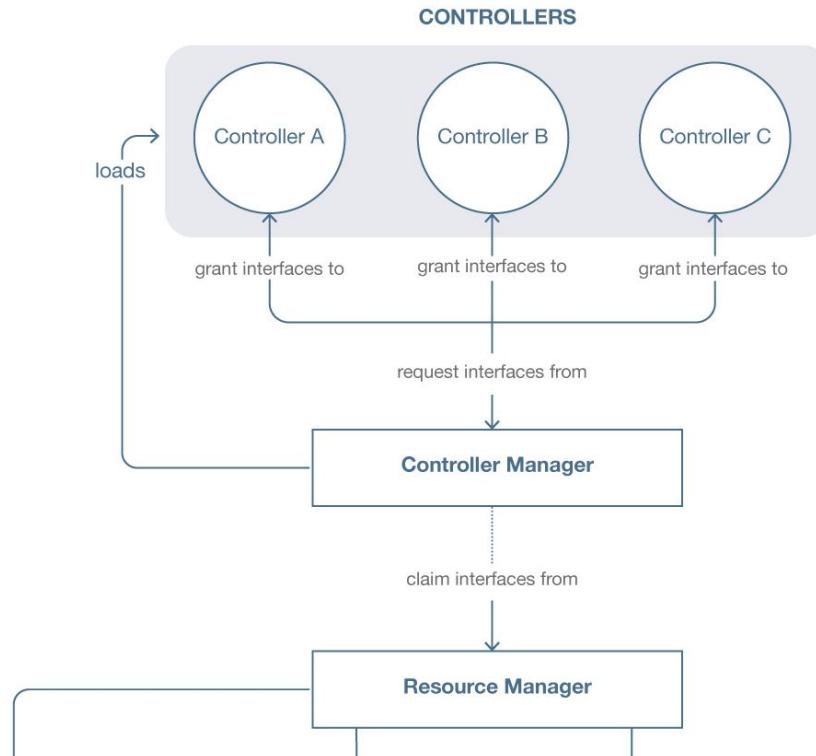
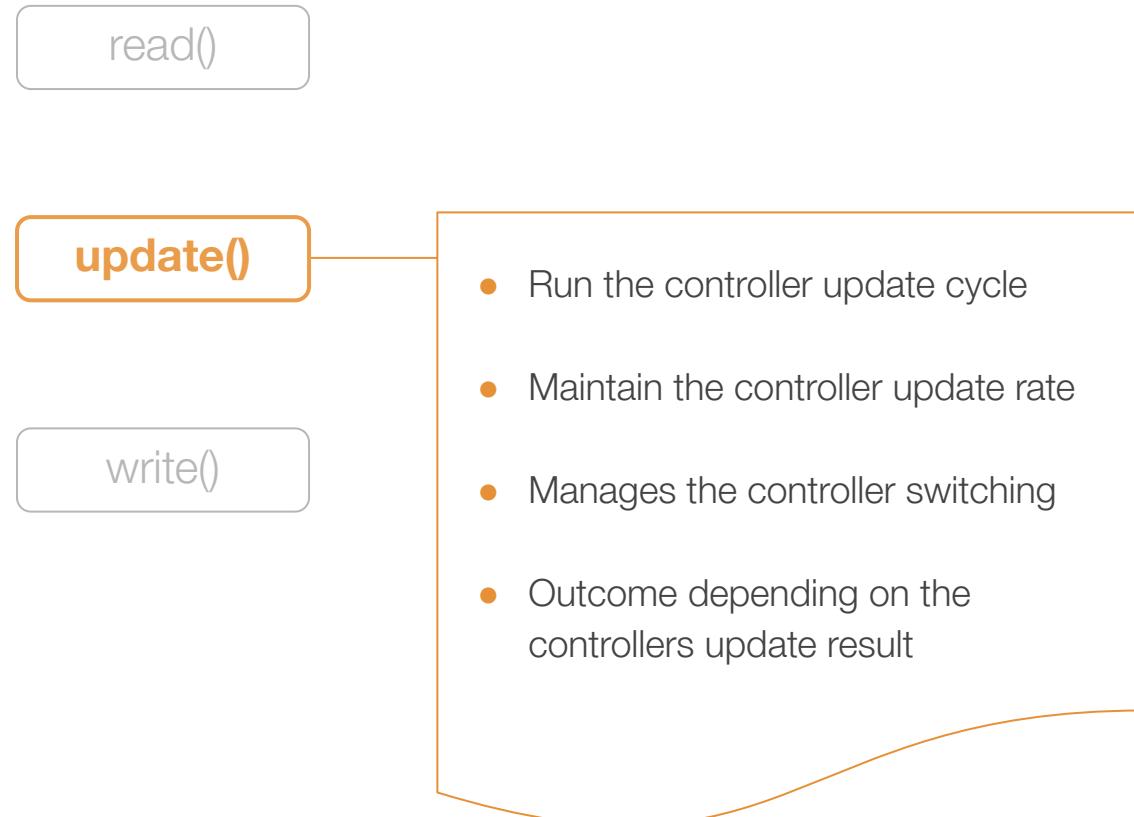
- An entry point for users via ROS services
- A node without an executor
- Connects controllers and the hardware-abstraction layer
- Manages Loading, Configuring, Activation, Deactivation, and Unloading of the controllers.
- Responsible for granting controllers access to the hardware via interfaces when enabled
- Manages the access to the hardware interfaces



Controller Manager



Controller Manager



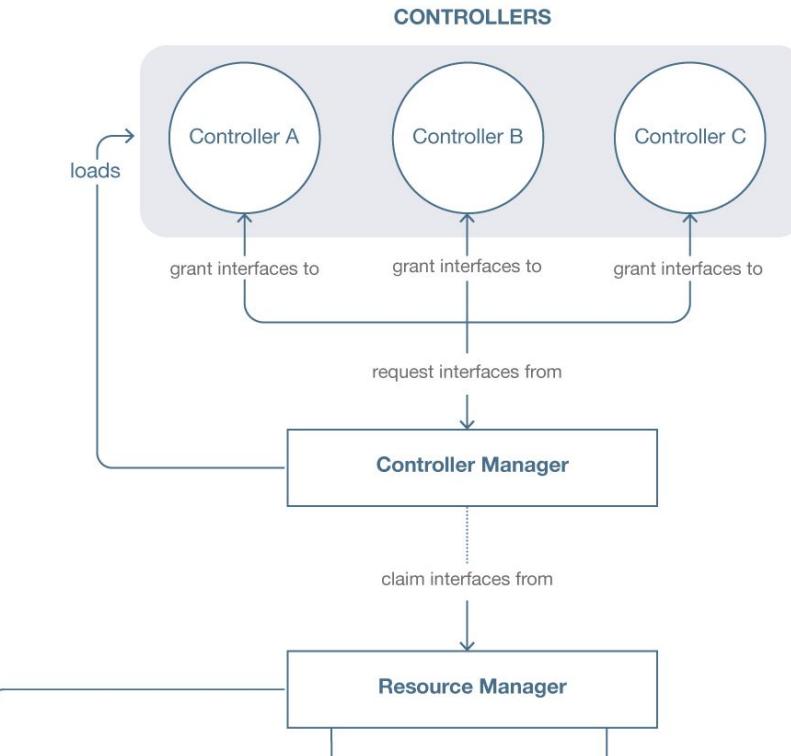
Controller Manager

read()

update()

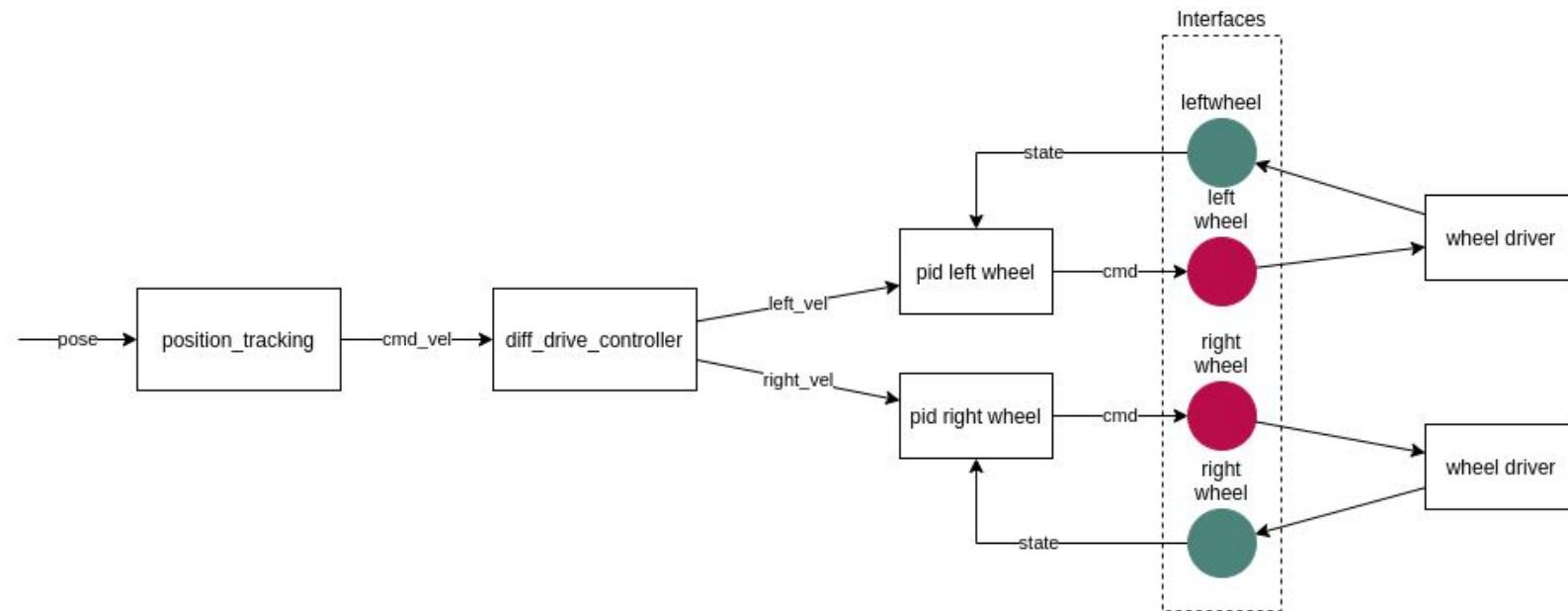
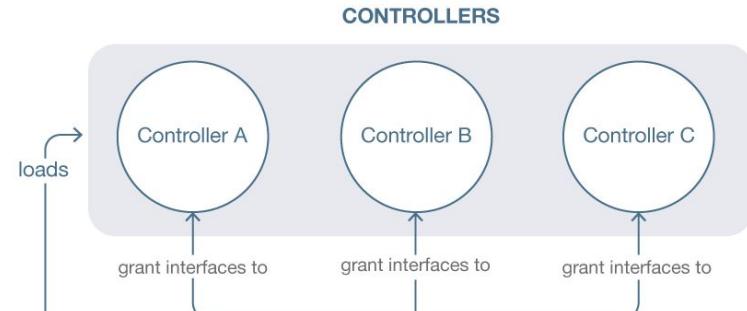
write()

- Writes data to the hardware from the interfaces updated by the controller
- Handle hardware write errors



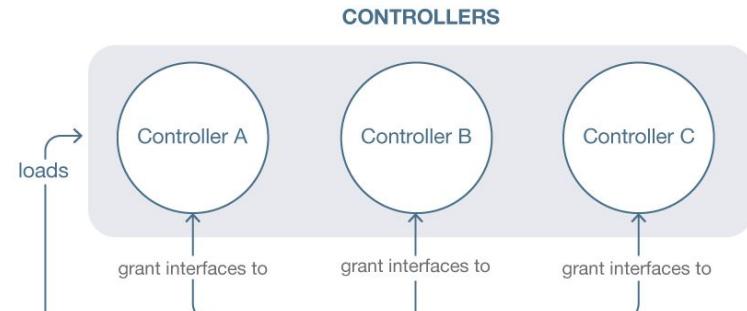
ROS2 Controllers

- Similar job of a ROS Controller but better with lifecycle
- Loosing of state and command interfaces
- Ability to chain with other controllers
- Ability to update synchronously and asynchronously
- Ability to have different update rates w.r.t controller manager



ROS2 Controllers

- Similar job of a ROS Controller but better with lifecycle
- Loosing of state and command interfaces
- Ability to chain with other controllers
- Ability to update synchronously and asynchronously
- Ability to have different update rates w.r.t controller manager



`joint_trajectory_controller`

`diff_drive_controller`

`gripper_controllers`

`joint_state_broadcaster`

`ackermann_steering_controller`

`force_torque_sensor_broadcaster`

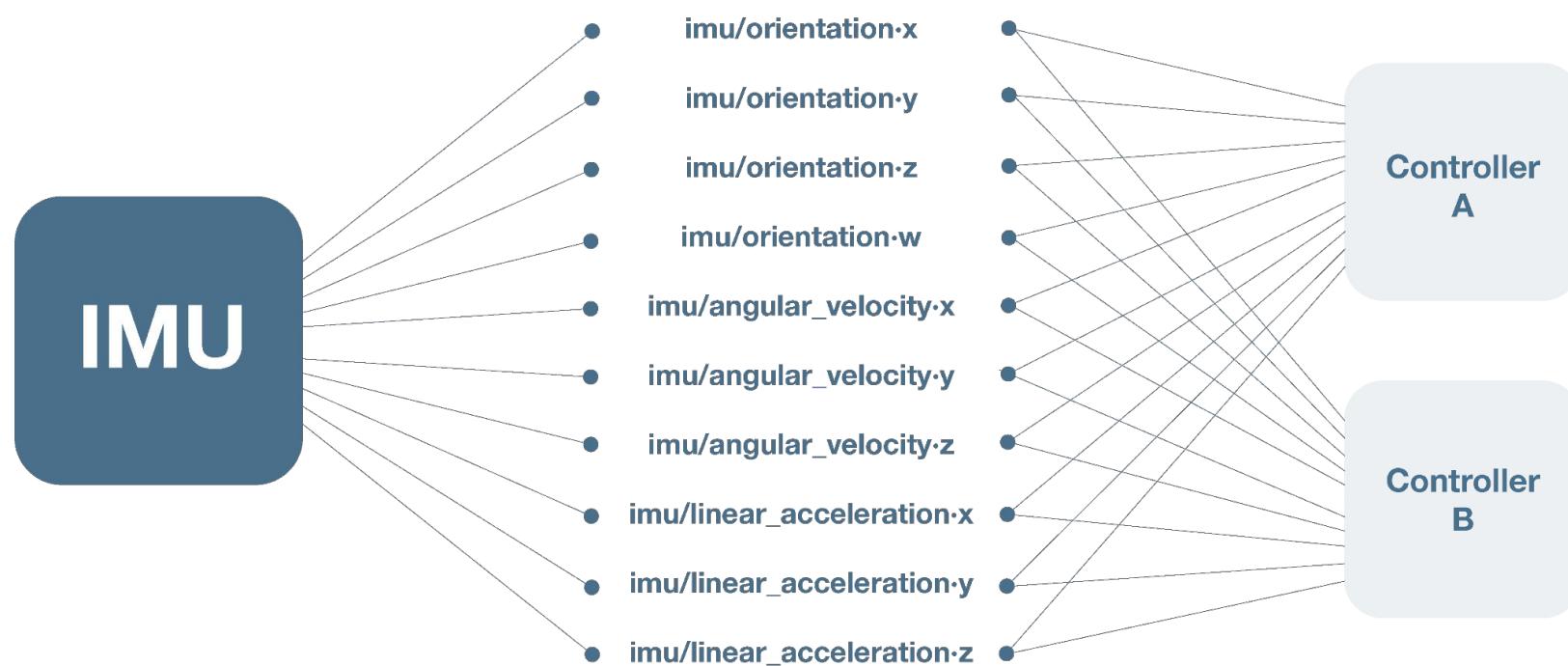
`admittance_controller`

`bicycle_steering_controller`

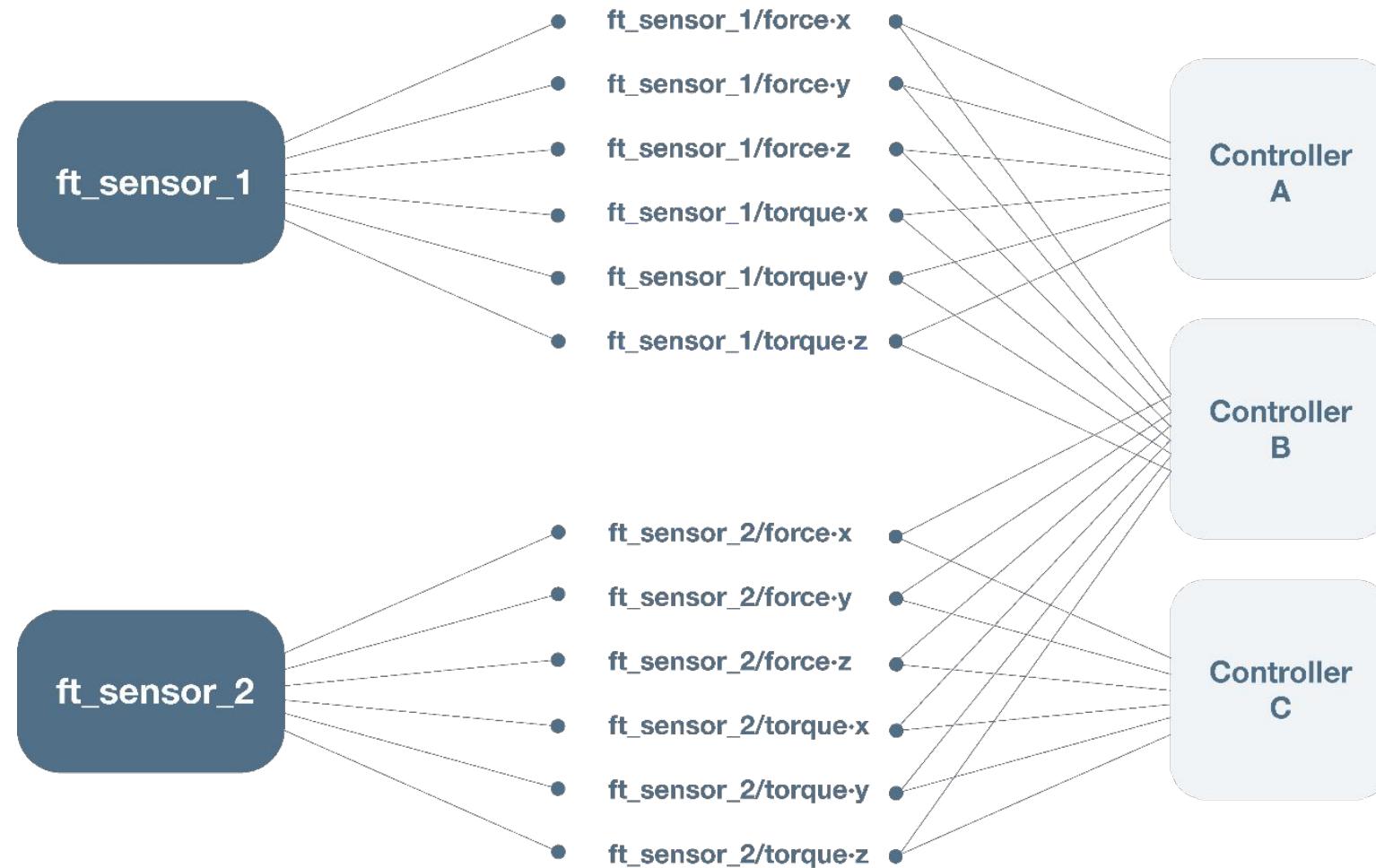
`imu_sensor_broadcaster`

Semantic Components

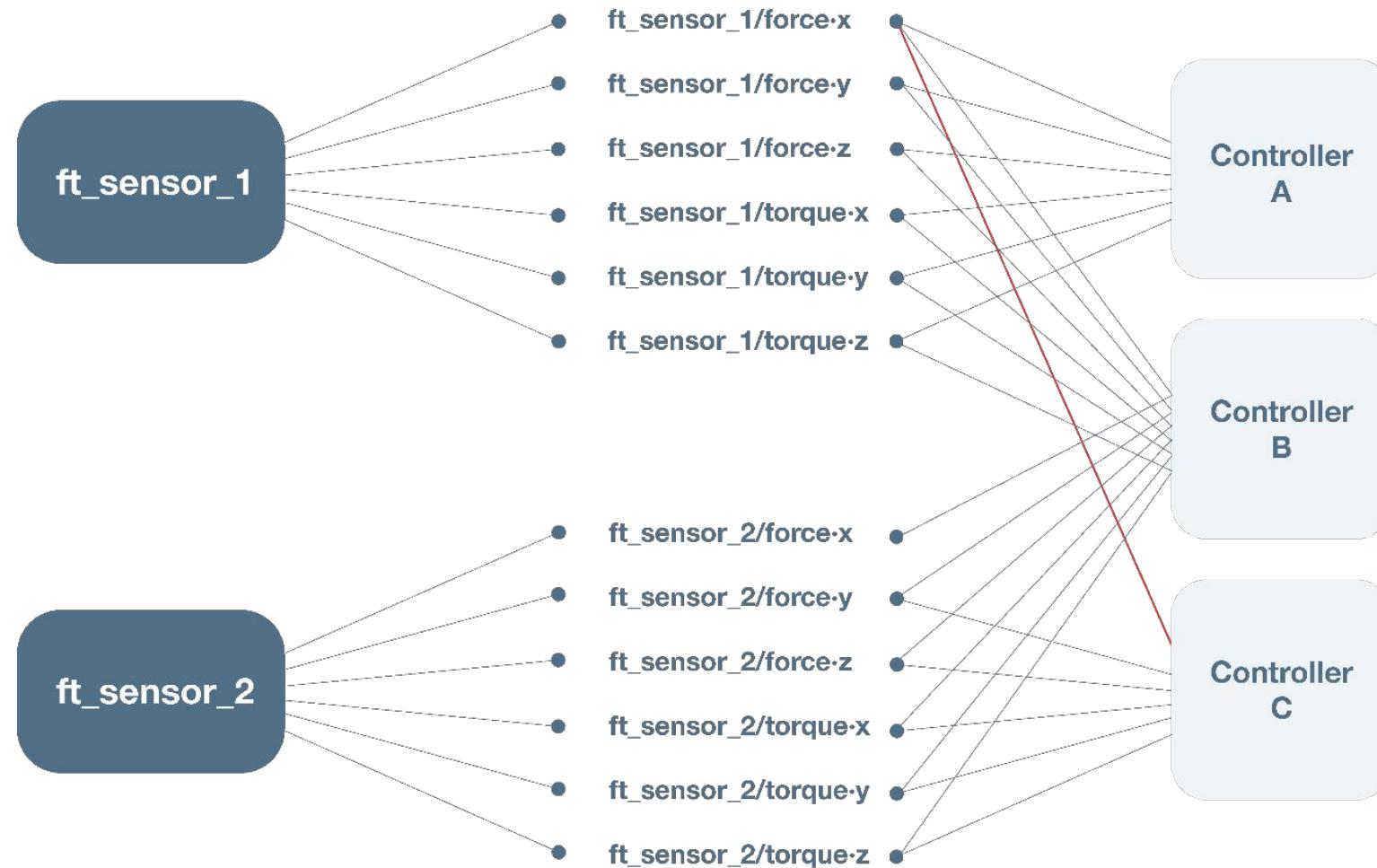
- Why do we need it?



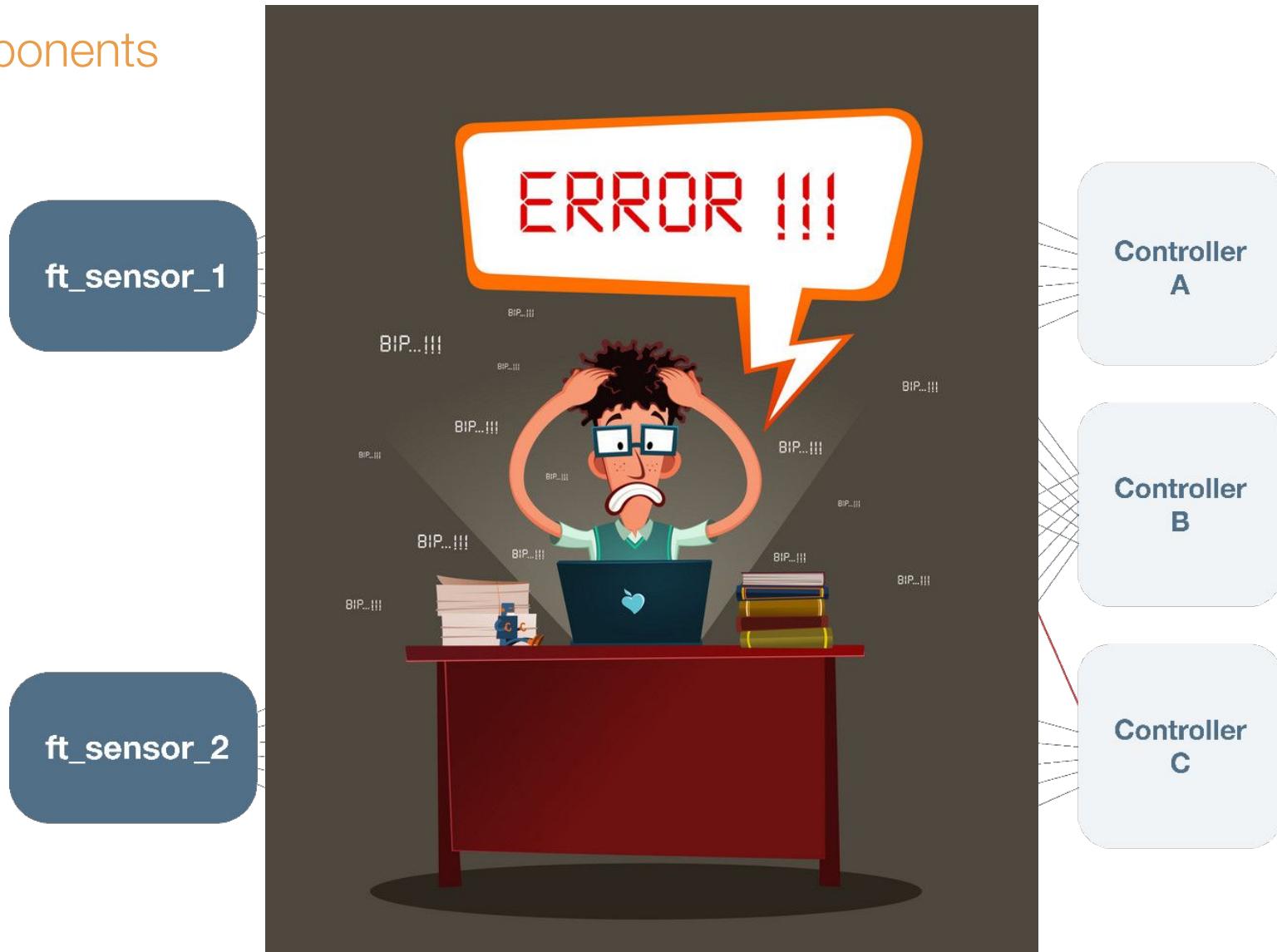
Semantic Components



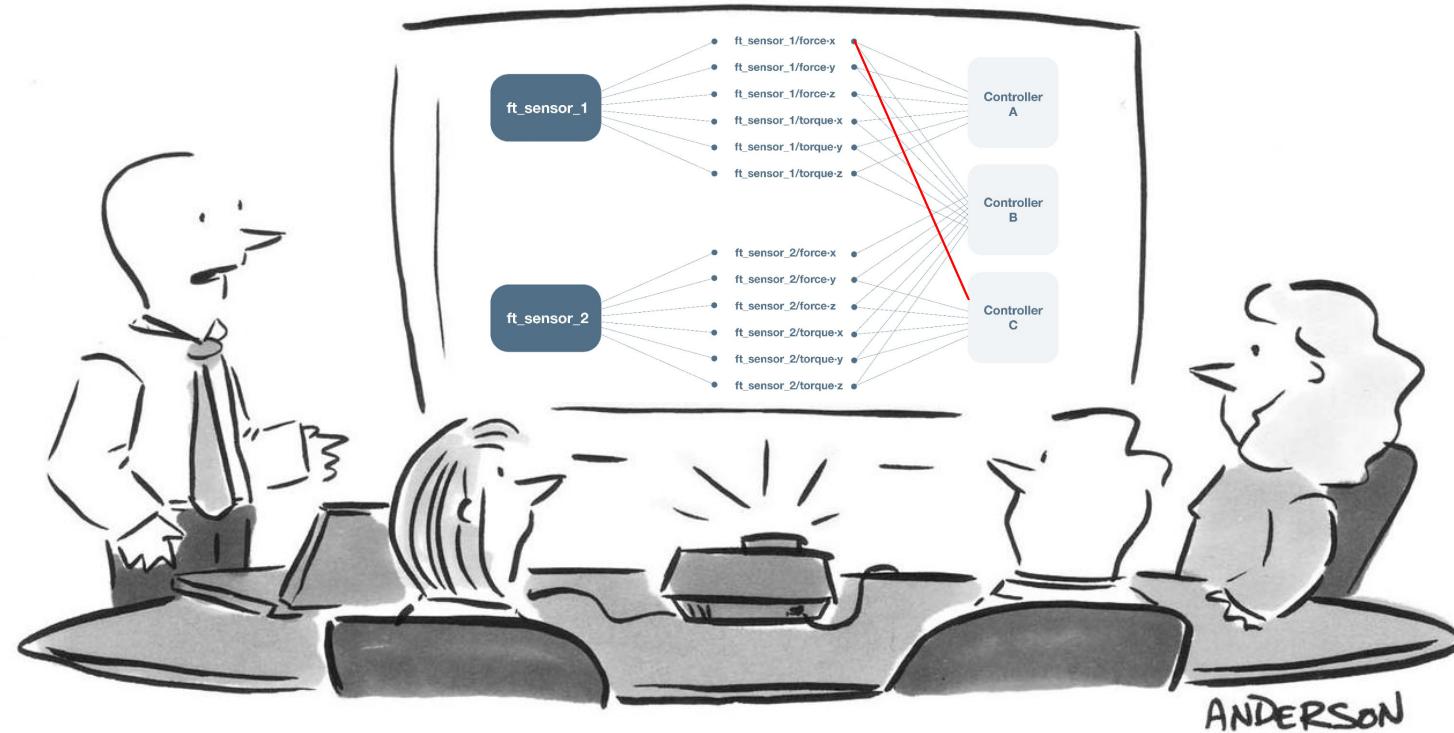
Semantic Components



Semantic Components



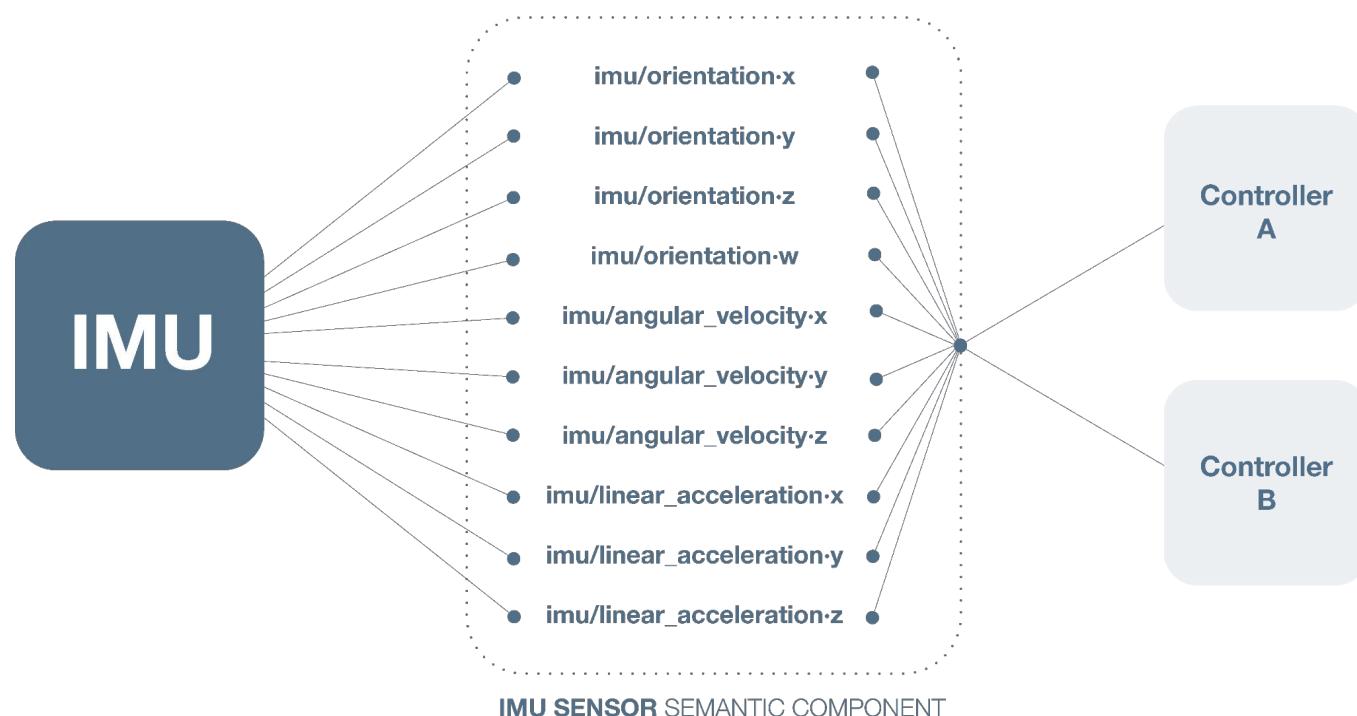
Semantic Components



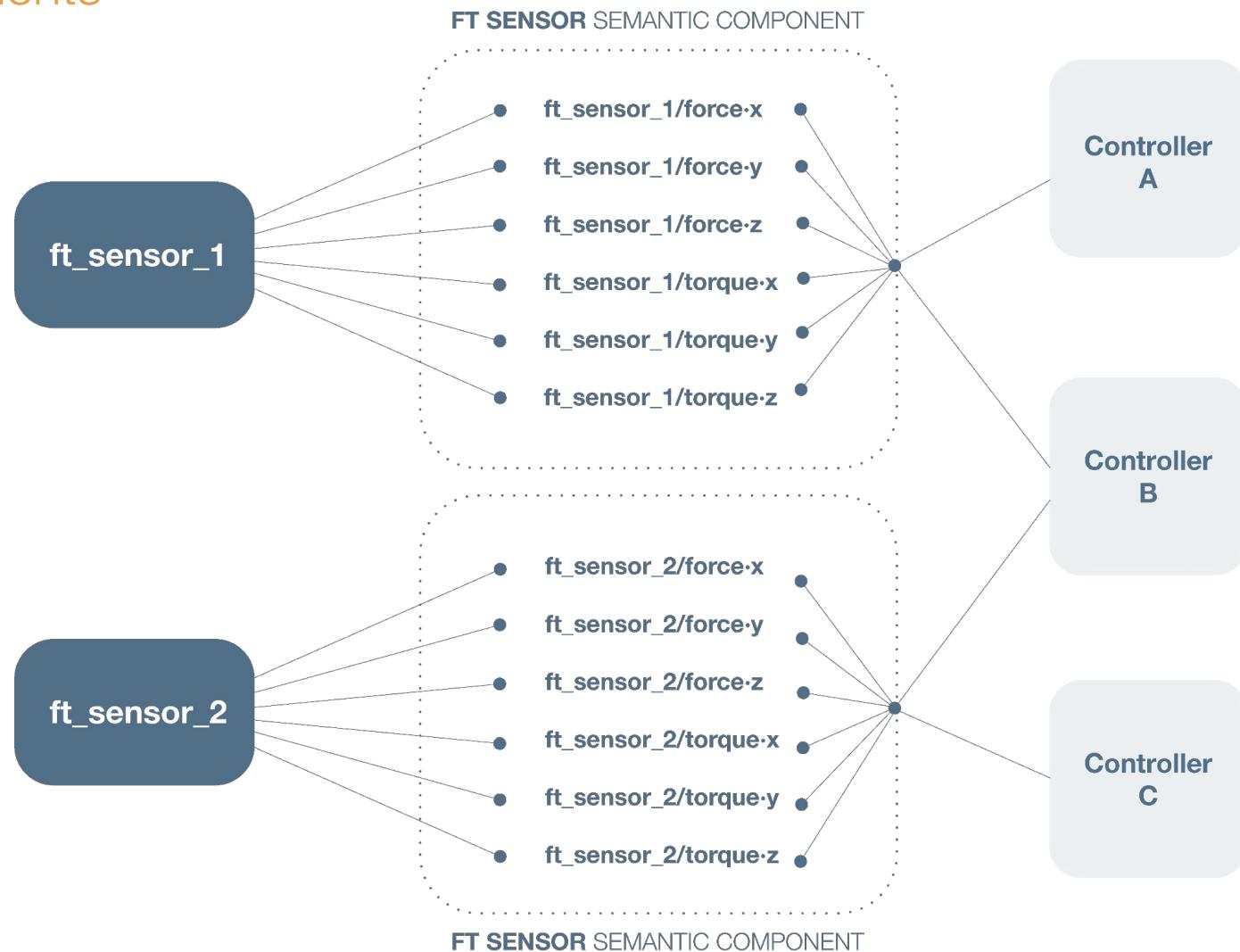
“The good news is we discovered the **typo** and have now hired an expert.”

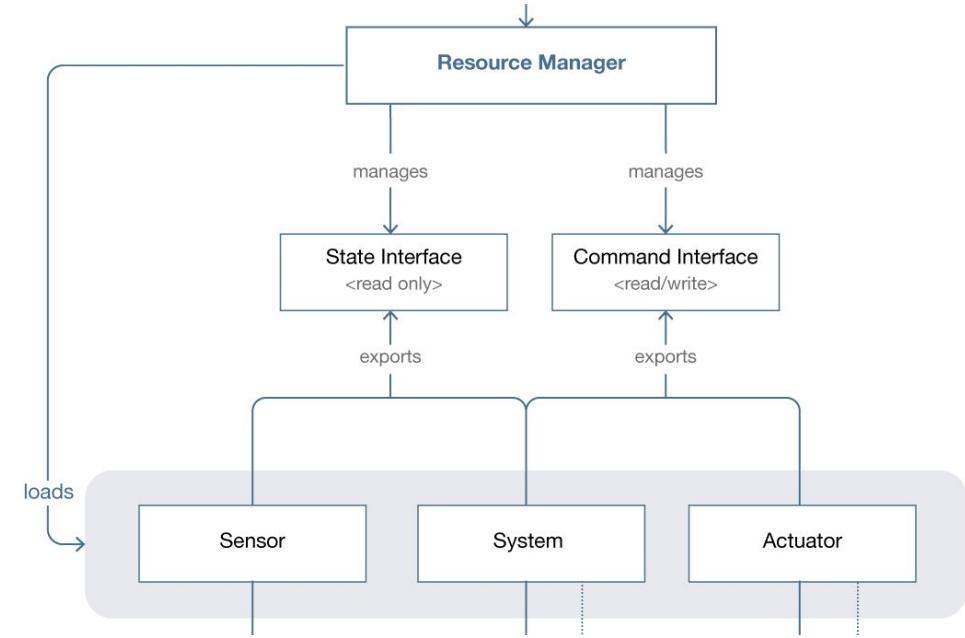
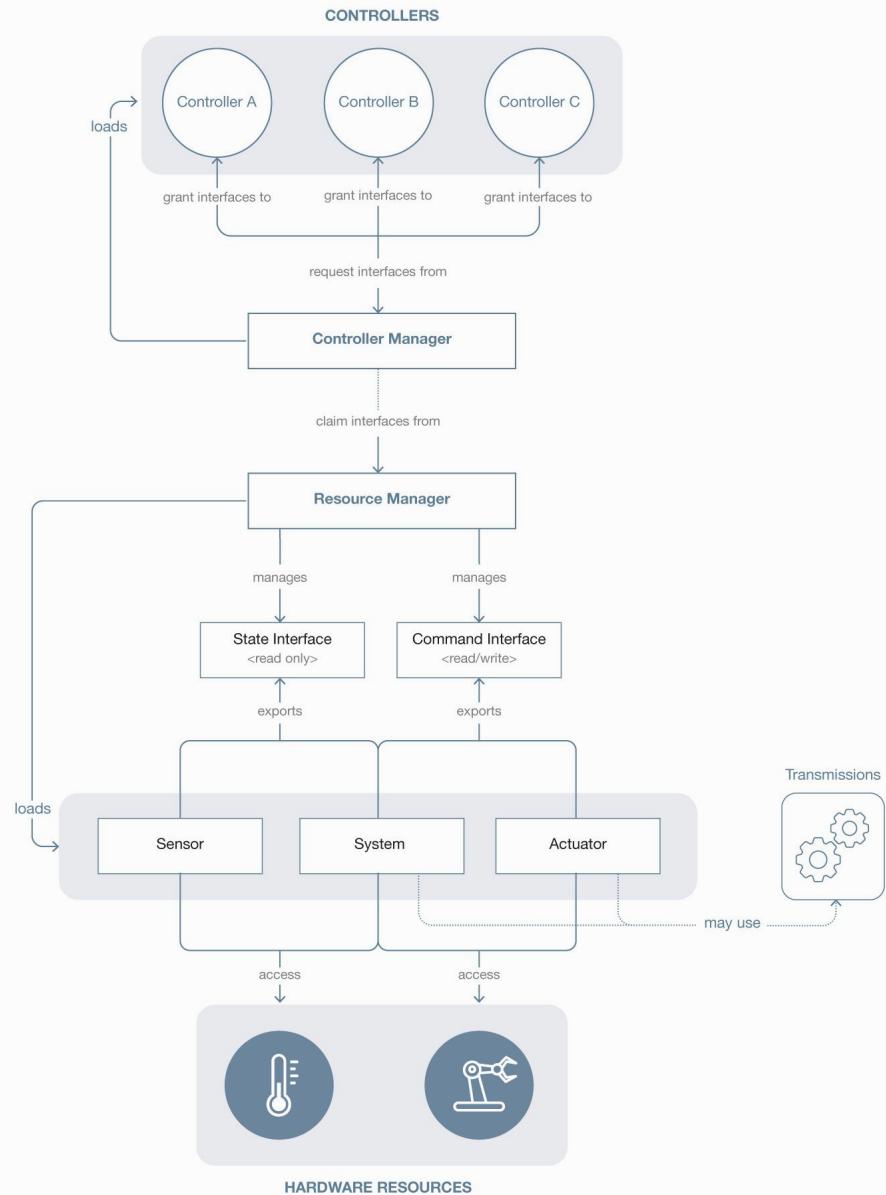
Semantic Components

- Loaning of the state interfaces
- Abstract all interfaces with semantic meaning into one component
- Ability to retrieve data into the ROS2 message types



Semantic Components





Conclusion

Conclusions

- ros2_control surpasses ros_control in flexibility
- Reuse of already existing hardware components
- Controller chaining
- Ability to run controllers with different update rate
- Ability to choose the components to run asynchronously
- Semantic components to wrap data with semantic meaning

Thank you

Sai Kishor Kothakota
sai.kishor@pal-robotics.com

pal-robotics.com