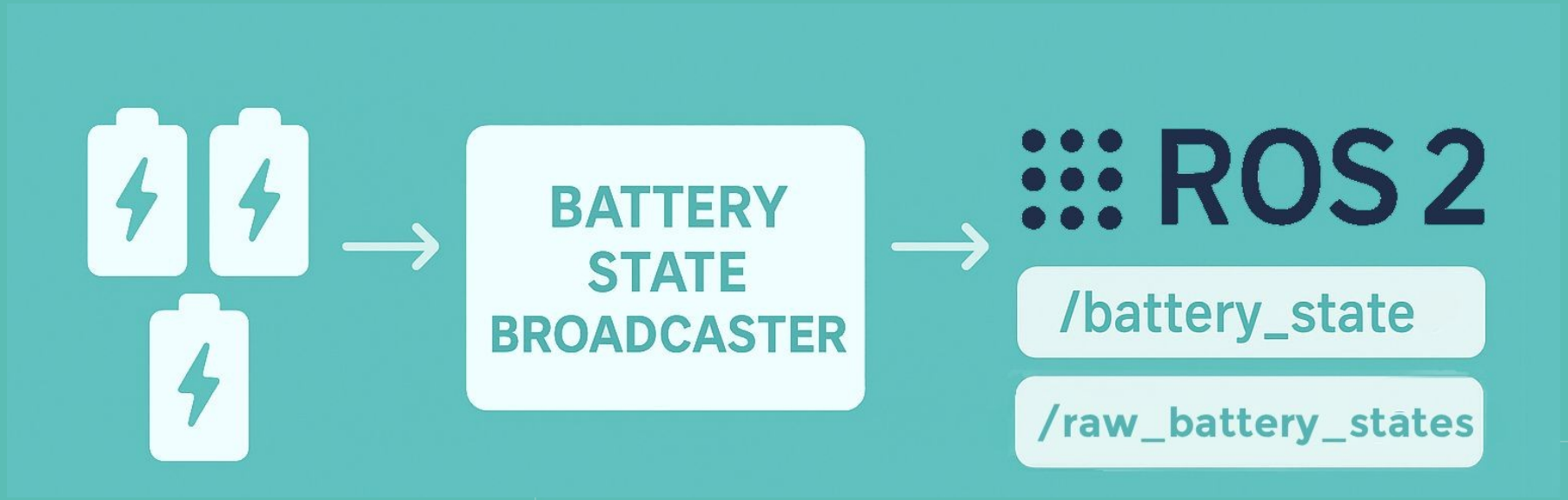
Keeping Robots Safe and Charged

» New ROS 2 Broadcasters

Yara Shahin and Dr. Ing. Denis Stogl

b»robotized

Battery State Broadcaster



The Gap

- The Missing Bridge
  - ✅ Hardware interfaces expose battery data
  - ✅ `sensor_msgs/BatteryState` Interface exists
  - ❌ No standard broadcaster connecting them.

- No System-Level Intelligence
  - ❌ Multiple batteries = scattered data
  - ❌ No aggregate health overview

Every team re-invents the wheel! 🛞

Why This Matters

- 🤖 Mobile Fleet Operations
  - Multiple robots with 2-3 battery packs each
  - Need centralized health dashboards
  - Mission planning requires accurate charge estimates

- 🚁 Drone Swarms
  - Critical safety: return-to-home on low battery
  - Multi-cell monitoring for balanced charging

Can you answer "Is my robot ready?" at a glance 📊

Solution: `battery_state_broadcaster`

- 🎯 **Multi-battery support**
  - Aggregated view: System-level battery health in **`BatteryState`**
  - Raw data: Per-joint (battery) details in **BatteryStateArray.**

- ⚙️ **Smart Battery analytics**
  - Weighted averaging for system-level metrics
  - Presence detection for hot-swappable batteries

» Solution: `battery_state_broadcaster`

- 🔧 **Flexible parametrization for Interface Support**
  - Mandatory Interfaces: `battery_voltage`
  - Optional interfaces: current, temperature, charge, percentage, status, ...
  - Parameters for battery properties
  - and more!

🔋 Full Coverage for `sensor_msgs::msg::BatteryState`!

How to use?

- Step 1️⃣: Hardware Interfaces (or mockHw!)
  - Expose the relevant interfaces as double. Only `battery_voltage` is mandatory!
- Step 2️⃣: URDF

```
<ros2_control name="my_mobile_robot" type="system">
  <hardware>
    <plugin>my_mobile_robot/MobileRobotSystem</plugin>
  </hardware>
  <joint name="${prefix}left_leg">
    <state_interface name="battery_voltage"/>
    <state_interface name="battery_current"/>
    <state_interface name="battery_power_supply_status"/>
    <state_interface name="battery_present"/>
  </joint>
  <joint name="${prefix}right_leg">
    <state_interface name="battery_voltage"/>
    <state_interface name="battery_temperature"/>
    <state_interface name="battery_power_supply_health"/>
  </joint>
</ros2_control>
```

» How to use?

» Step 3️⃣: Configure

» Step 4️⃣: Launch 🔋

```yaml
battery_state_broadcaster:
  ros__parameters:
    state_joints: ["left_leg", "right_leg"]
    interfaces:
      left_leg:
        battery_current: true
        battery_power_supply_status: true
        battery_present: true
      right_leg:
        battery_temperature: true
        battery_power_supply_health: true
    left_leg:
      minimum_voltage: 0.0
      maximum_voltage: 10.0
      capacity: 12000.0
      design_capacity: 13000.0
      power_supply_technology: 3
      location: "left_slot"
```

The Aggregation Magic

Field	Aggregated	Raw
voltage, temperature, current, percentage	Mean across reporting joints	Per-joint interface value or nan
charge, capacity, design_capacity	Sum across all joints	Per-joint interface value or nan
Power supply status and health	Highest enum (most critical status)	Per-joint interface or 0 (unknown)
location, serial_number	All joints appended	Per-joint parameter or empty
present	Always true (system level)	Per-joint interface value or inferred from voltage

VDA5050 Safety State Broadcaster

Industry-Standard Safety for `ros2_control`

The Gap

- **What is VDA5050?**
  - 📋 The Industry Standard for AGV/AMR Communication.
  - 🌍 Vendor-agnostic fleet management.
- **The Missing Bridge**
  - ✅ SafetyState Interface defined by VDA5050.
  - ✅ VDA5050 connectors exist.
  - ❌ No standard way to expose safety state from `ros2_control` to `VDA5050`.

Every team re-invents the wheel! 🛞

Why SafetyState Matters

- 🚨 **Fleet-Level Emergency Response**
  - 10+ AMRs operating in same facility.
  - E-stop on Robot A → Fleet manager needs to know immediately.
  - Reroute nearby Robot B to avoid collision zone.

Can you answer "Is my fleet safe to operate?" in real-time ⚠️

Solution: `vda5050_safety_state_broadcaster`

- 🎯 **VDA5050-Compliant Output**
  - Publishes `control_msgs::msg::VDA5050SafetyState`
- 🔧 **Intelligent E-Stop Aggregation for multiple interfaces**
  - Multiple Interface monitoring for each state.
  - Priority-based selection **(manual > remote > auto-ack)**
- 🚨 **Field Violation Monitoring**
  - Multiple Interface monitoring.

Solution: `vda5050_safety_state_broadcaster`

- 🔧 **Flexible parametrization for Interface Support**
  - Freely configure lists of interface names for `fieldViolation` and `E-stop` types.

✅ **Full Coverage for VDA5050::SafetyState Schema**

How to use?

» Step 1️⃣: Hardware Interfaces (or mockHw!)

» Expose the relevant interfaces as double.

» Step 2️⃣: URDF

```
<ros2_control name="my_fleet_system" type="system">
    <hardware>
        <plugin>my_fleet_system/FleetSystem</plugin>
    </hardware>

    <sensor name="MQTT_sensor1">
        <state_interface name="fieldViolation"/>
        <state_interface name="eStopManual"/>
        <state_interface name="eStopRemote"/>
        <state_interface name="eStopAutoack"/>
    </sensor>

    <sensor name="MQTT_sensor2">
        <state_interface name="fieldViolation"/>
        <state_interface name="eStopRemote"/>
    </sensor>

</ros2_control>
```

How to use?

» Step 3️⃣: Configure

» Step 4️⃣: Launch 🔋

```yaml
vda5050_safety_state_broadcaster:
  ros__parameters:
    fieldViolation_interfaces:
      - MQTT_sensor1/fieldViolation
      - MQTT_sensor2/fieldViolation
    eStop_manual_interfaces:
      - MQTT_sensor1/eStopManual
    eStop_remote_interfaces:
      - MQTT_sensor1/eStopRemote
      - MQTT_sensor2/eStopRemote
    eStop_autoack_interfaces:
      - MQTT_sensor1/eStopAutoack
```

Integration with ros2_controllers

- ✅ Consistency with `ros2_controllers` best practices.
- ✅ Uses `generate_parameter_library`
- ✅ Standard lifecycle management
- ✅ Extensive documentation
- ✅ Unit Tests

Open source: Community-maintained!

Thank You:)

- **Special Thanks:**
  - @ottojo for the support with `battery_state_broadcaster`.
  - ros-controls maintainers for guidance.
  - b»robotized for supporting open source!
- **Questions?**
  - 📧 yara.shahin@b-robotized.com
  - 🐱 github.com/YaraShahin

Join the conversation