The State of ROS: From the Iron Age to the Jazzy Age

Chris Lalancette - ROS 2 Technical Lead
Yadunund Vijay - Iron ROS Boss

October 18, 2023
Who are we?
Outline

- ROS distribution update
- The community
- New core features in Iron Irwini (May 2023)
- Features in development for Jazzy Jalisco (May 2024)
- Long-term outlook for core features
ROS Distribution update
ROS Distros (REP-2000)

<table>
<thead>
<tr>
<th>Distro EOL Dates</th>
<th>Kinetic</th>
<th>Dashing</th>
<th>Galactic</th>
<th>Melodic</th>
<th>Foxy</th>
<th>Iron</th>
<th>Noetic</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Distro Release Dates</th>
<th>Jazzy</th>
<th>Jalisco (LTS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>6/2020</td>
<td></td>
<td>5/2024</td>
</tr>
</tbody>
</table>

ROS 2 ROLLING RIDLEY
ROS Distros (REP-2000)

**Distro EOL Dates**
- Kinetic: 4/2021
- Dashing: 5/2021
- Galactic: 11/2022
- Melodic: 5/2023
- Foxy: 5/2023
- Iron: 11/2024
- Noetic: 5/2025

**Distro Release Dates**
- Jazzy: 6/2020
- Jalisco (LTS): 5/2021
- Humble Hawsbill: 5/2022
- Irwin: 5/2023

**ROS 2 ROLLING RIDLEY**
- Jazzy Jalisco (LTS): 5/2024
ROS Distros (REP-2000)

- ROS 1: **1.5 years to transition!**
- ROS 2: Humble, Iron, or Rolling (Jazzy)

**Distro EOL Dates**
- Kinetic: 4/2021
- Dashing: 5/2021
- Galactic: 11/2022
- Melodic: 5/2023
- Foxy: 5/2023

**Distro Release Dates**
- 6/2020
- 5/2021
- 11/2024

**Jazzy Jalisco (LTS)**: 5/2024

**ROS 2 ROLLING RIDLEY**
How are releases produced?
# Packages available (September 2023)

![NOETIC NINJEMYS](image1)
1987

![HUMBLE HAWKSBILL](image2)
1323

![IRON IRWINI](image3)
1156

![ROLLING RIDLEY](image4)
1073
45.1%  
Percentage ROS downloads that were ROS 2 in April 2023!

43,447,402  
ROS package downloads in April 2023 alone!

17.14%  
Growth in ROS downloads between 4/2022 and 4/2023
Package downloads

Download Percentage from packages.ros.org April 2022 vs April 2023

- In 2022:
  - kinetic: 4.01%
  - melodic: 10.1%
  - noetic: 31%
  - foxy: 27.3%
  - galactic: 10.5%
  - humble: 12.1%
  - rolling: 2.71%
- In 2023:
  - kinetic: 1.03%
  - melodic: 9.98%
  - noetic: 18.9%
  - foxy: 5.22%
  - galactic: 14.6%
  - humble: 11.7%
  - rolling: 4.12%
  - All Other Distros: 32.9%
The Community
## ROS Bosses

- What is a ROS Boss?
- Who are the ROS Bosses?

### ROS 1
- Noetic: Shane Loretz
- Melodic: Chris Lalancette
- Lunar: Mikael Arguedas / Chris Lalancette
- Kinetic: Tully Foote
- Jade: Tully Foote / William Woodall
- Indigo: Tully Foote
- Hydro: Tully Foote
- Groovy: Tully Foote
- Fuerte: Ken Conley
- Electric: Ken Conley
- Diamondback: Ken Conley
- C Turtle: Ken Conley
- Box Turtle: Ken Conley

### ROS 2
- Rolling: Steven! Ragnarök
- Jazzy: Marco A. Gutiérrez
- Iron: Yadunund Vijay
- Humble: Audrow Nash
- Galactic: Scott K Logan
- Foxy: Jacob Perron / Dharini Dutia
- Eloquent: Michael Carroll
- Dashing: Steven! Ragnarök
- Crystal: Steven! Ragnarök
- Bouncy: Mikael Arguedas / Steven! Ragnarök
- Ardent: Steven! Ragnarök

### ROS 3
- Noetic: Shane Loretz
- Melodic: Chris Lalancette
- Lunar: Mikael Arguedas / Chris Lalancette
- Kinetic: Tully Foote
- Jade: Tully Foote / William Woodall
- Indigo: Tully Foote
- Hydro: Tully Foote
- Groovy: Tully Foote
- Fuerte: Ken Conley
- Electric: Ken Conley
- Diamondback: Ken Conley
- C Turtle: Ken Conley
- Box Turtle: Ken Conley
TSC (Technical Steering Committee)
Resources

- Core documentation: https://docs.ros.org
- Package index: https://index.ros.org
- Discourse for discussion, announcements, or release: https://discourse.ros.org
- Discord server for real-time conversation: https://www.ros.org/blog/discord/
- ROS Robotics stack exchange: https://robotics.stackexchange.com/questions/tagged/ros
- Working groups: https://docs.ros.org/en/rolling/The-ROS2-Project/Governance/Working-Groups.html
- GitHub:
  - https://github.com/ros2
  - https://github.com/ament
  - https://github.com/ros
New Core Features in Iron Irwini

Released May 23rd 2023

Changelog
Python API Documentation

**Primary Contributors:** Abrar Rahman Protyasha & Yadunund Vijay

- [https://docs.ros.org](https://docs.ros.org) contains API level documentation for all released packages
- C++ has existed for awhile
- Python API docs are new
- rclpy: [https://docs.ros.org/en/rolling/p/rclpy](https://docs.ros.org/en/rolling/p/rclpy)
REP-2012 Service Introspection

Primary Contributors: Brian Chen & Jacob Perron & Chris Lalancette

What?
Ever want to dig into a ROS service and see who called it, the request, and the results?

Now you can!

Why?
Debugging services is hard. We wanted a simple way to watch a service in a running ROS system. You can now see service events in a separate topic e.g.

/myservice/_service_event

How?
Service introspection is not enabled by default. It must be enabled for the Client and Server respectively.

It is a ROS service debug mode!
Pre & Post Parameter Callbacks

Primary Contributors: Deepanshu Bansal & Jacob Perron

What?
Node parameter callbacks used to be “all or nothing,” Every param had to be correct to take effect. Now you can modify parameter lists on the fly! The post parameter is where you want to change state.

Why?
We wanted to give users more flexibility without breaking API. This approach allows users to add validation and sanitization to complex parameter lists.

How?
Check out the
add_pre_set_parameters_callback
add_post_set_parameters_callback
methods in your ROS node.
Matched Events

Primary Contributors: Barry Xu & Tomoya Fujita

What?
Want to know when a publisher/subscriber pair establish or drop a connection? Matched event callbacks let you do just this! This can also be triggered based on compatible QoS.

Why?
ROS node computations can be expensive, and some users don’t want to publish unless the right nodes are listening. Matched events allow this feature.

How?
Matched callbacks can be set within PublisherEventCallbacks and SubscriptionEventCallbacks which are passed as options to the publisher or subscription resp.
External Logger Configuration!

Primary Contributors: Lei Liu & Barry Xu & Tomoya Fujita

What?
ROS logging is normally configured at the system level. Have you ever wanted to set just one node to verbose mode at runtime?

Now you can!

Why?
ROS systems can be huge! Users needed a granular way to set node level logging from the CLI.

How?
Enable enable_logger_service at node creation which will create two new services:

1. mynode/set_logger_levels
2. mynode/get_logger_levels
New Default Bag Format: MCAP

**Primary Contributors:** James Smith & Emerson Knapp & Michael Orlov

**Standards Evolution**
Influenced by ROS 1 bags, and fixing many of the issues in SQLite3, MCAP is the new default bagging format for ROS 2 that’s optimized for read and write!

**Baked in Message Defs**
Message definitions are baked into MCAP, making the format portable! Third party tools like PlotJuggler and Foxglove can now render any MCAP bag file.

**Enhanced Performance**
Preliminary performance evaluation indicates a ~2-5x increase in message throughput over SQLite on ROS bag benchmarks (YMMV).
Experimental rclcpp executor

Primary Contributors: Mauro Passerino & Lenny Story & Alberto Soragna

What?
The executor is responsible for calling your callbacks. The default one uses a fixed priority order. The new EventsExecutor maintains a queue of events and executes them as they arrive.

Why?
The EventsExecutor can be more fair to events and can have better performance than the default executor.

How?
When writing your nodes, first explicitly create the rclcpp::experimental::executors::EventsExecutor. Then add your node to the executor and spin it!
Features in development for Jazzy Jalisco
rmw_zenoh

- DDS is here to stay but we think an alternative middleware protocol is beneficial to the community.

- Motivations and community discussions: https://discourse.ros.org/t/investigation-into-alternative-middleware-solutions/32642/43

rospkg2 service/action record and play
Long term outlook for core features
Use of “conda” for Windows
Performance improvements
Ability to release Rust packages
Evolving messages over time
Automatically detect “best” data delivery
Interested in helping?

http://docs.ros.org/en/rolling/The-ROS2-Project/Contributing.html
Gazebo
Gazebo Harmonic features

- Python bindings for systems
- Automatically compute Moment of Inertia of meshes
- Exert forces and torques by dragging objects in the scene
- Simulate virtual mass of displaced volume of objects in fluids
- Generate custom Gazebo messages
- Lens Flare support
ROS is healthy and growing, thanks to you!