OBJECT COMPUTING

ROS2 and DDS

Jeremy Adams

Sr. Software Engineer

June 18, 2020

What We Do

We use technology to unlock **business value** for our clients.

Analytics & Machine Learning

Unlock insights, accelerate growth, and strengthen your competitive advantage by transforming data into actionable strategies.

Blockchain Solutions

Achieve supply chain transparency, cryptographic security, and scalable growth through our portfolio of enterprise blockchain solutions.

Software Engineering

Expert software engineering services to meet our clients' complex and evolving business and technical requirements.

IoT & Industrial IoT

Modernize your industrial equipment and enable seamless connectivity across your digital ecosystem with our real-time analytics and predictive maintenance solutions.

Cloud Transformation

Build transformative solutions that ensure interoperability, leverage smart and reusable technologies, and optimize cloud spend.

Systems Integration

Establish full interoperability between devices and applications in real-time, to deliver performance, reliability, scalability, and security.

Who is Object Computing, Inc. (OCI)?

BY THE NUMBERS

27

years in business, serving clients globally

98%

year-over-year client retention

65%

of our leadership team is women & people of color

65%

of our tech team has 15+ years experience

30%

of our tech team has 15+ years tenure at our company

85%

of our leadership is promoted from within

Agenda

1. Why ROS2?
2. Demystify DDS
3. ROS2 using DDS
4. Migrating to ROS2
5. New use cases for ROS2 with DDS

Why ROS2?

ROS2: ROS1 origins

Born from ROS1

- ROS1 limitations
  - Inadequate support for teams of robots
  - Primary support only for Ubuntu and some Windows 10
  - Poor network performance
  - Research oriented
- ROS1 reimagined as ROS2 using
  - Data Distribution Service (DDS)
  - Officially on Windows, Mac, and Linux
    - Community support for other RTOSs
  - Production quality development

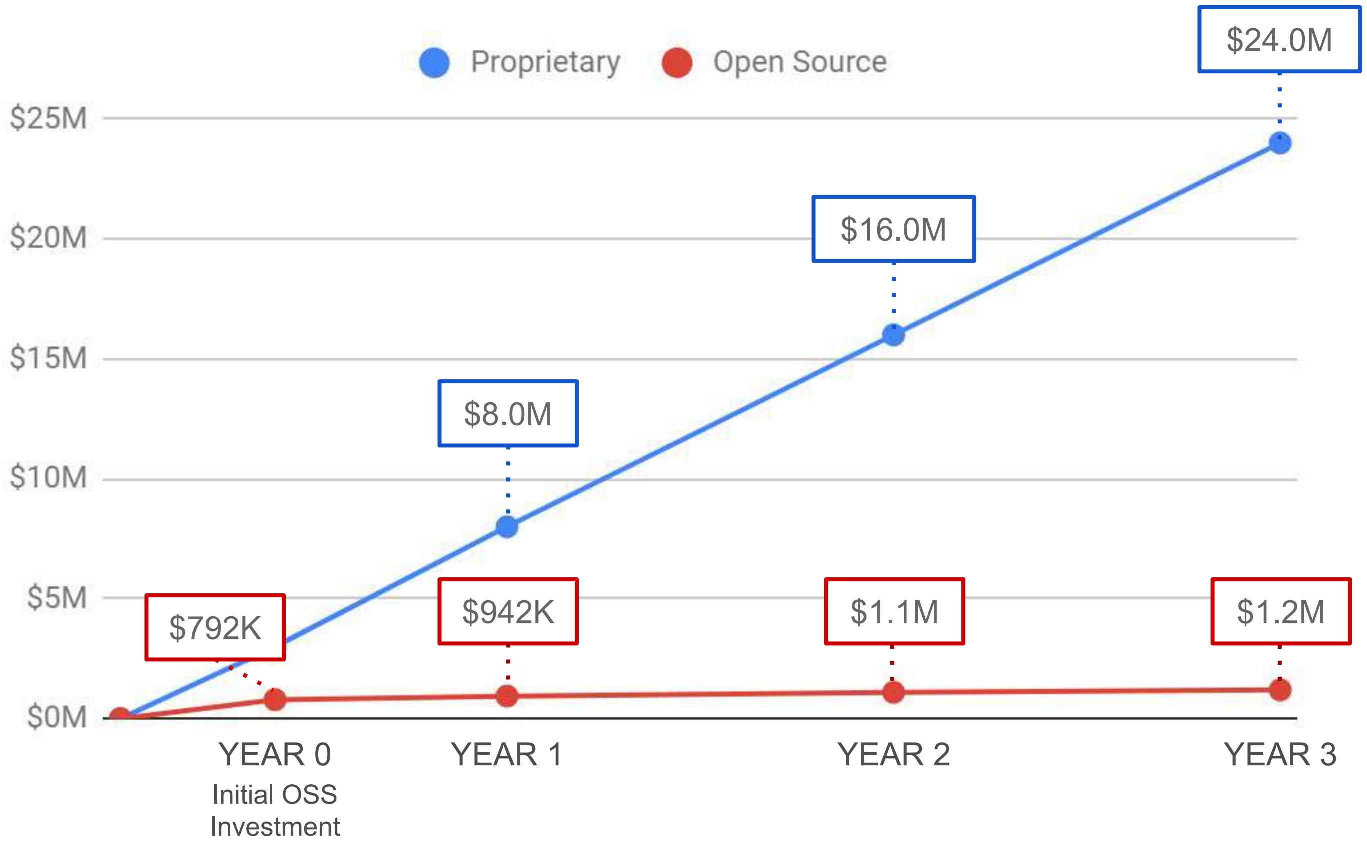
Open Source Software (OSS) cost benefit

Dollars and cents of OSS and open standards

- Lower internal training costs
- Modern design
- Focus your $ on your team’s strengths and market need
- Reduced software tool costs
- Reduced maintenance costs
- Lower recruiting costs
- Reduce cost of rapid prototyping to production

The Business Case for Open Source

Case Study: Cumulative Cost Savings of Open Source Over Proprietary Infrastructure Licensing



Client: A large multinational defense, security, and aerospace company.

Requirement: Scale its geospatial intelligence network from 40 nodes to 400 nodes worldwide.

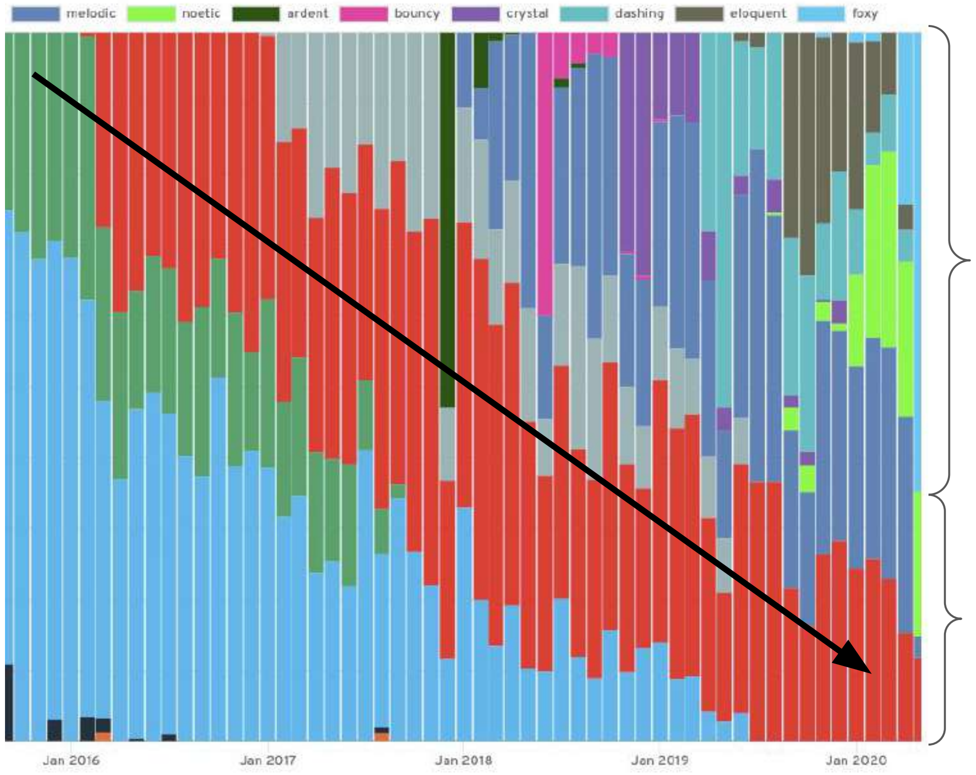
Problem: Proprietary solution for system infrastructure and integration posed significant per-node licensing costs (anticipated to increase from $800K to $8M per year).

Solution: OCI partnered with client to identify and extend an open source solution that would include hardening and resiliency features necessary for Fault Tolerance, Robust Thread Pool Management, and Advanced Service Discovery. OCI software engineers completed the effort in 7 months, at a one-time cost of $792K. Cost of ongoing support estimated at ~$150K per year.

Result: Three-year estimated savings of **$22.8M**.

Why ROS2?: ROS1 is sunsetting

Contributions to ROS1 distribution core packages are decreasing



Most recent ROS2

Most recent ROS1

> "In those almost thirteen years we as a community have made 12 releases happen together, and now we are proud to announce the 13th and last official ROS 1 release: Noetic."

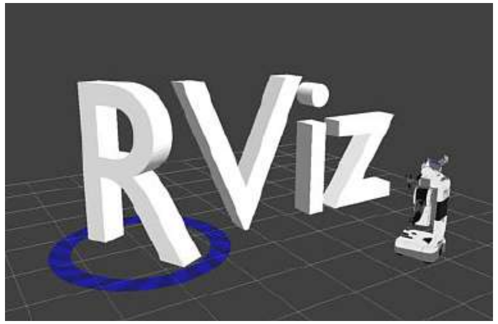
- Open Robotics

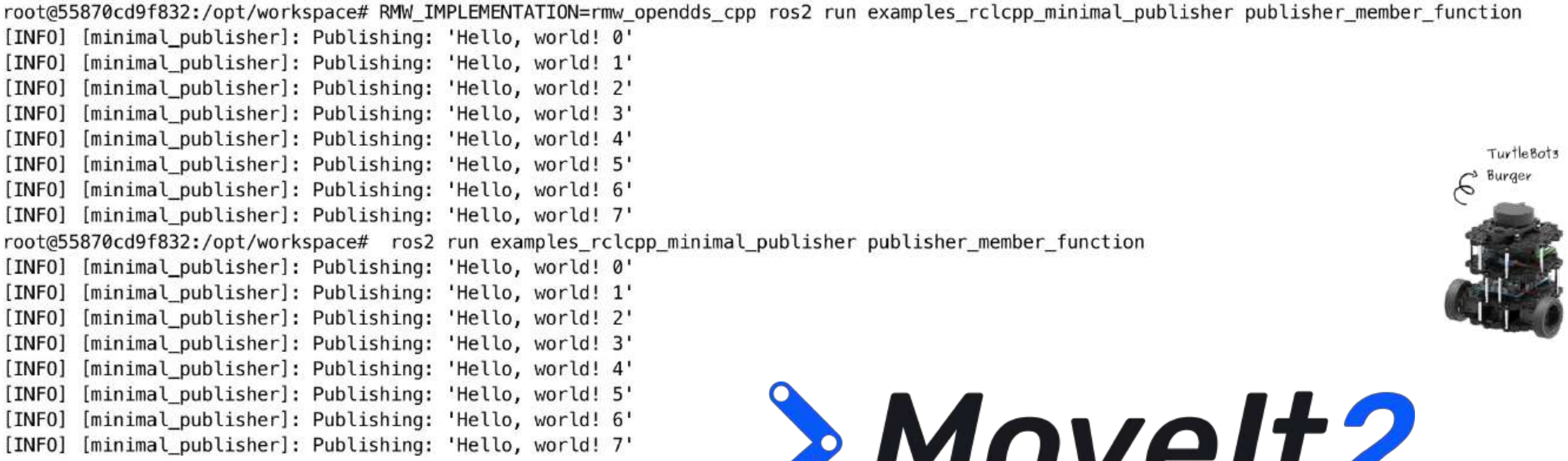
https://www.openrobotics.org/blog/2020/5/23/noetic-ninjemys-the-last-official-ros-1-release

Why ROS2?: ROS1 feature parity

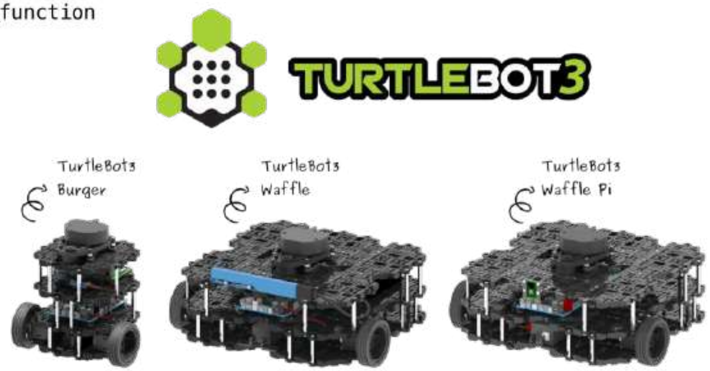
ROS2 has support today for many key features from ROS1

- RViz2
- Navigation stack (aka Nav2)
- Command line tools
- MoveIt (aka MoveIt2)





```
root@55870cd9f832:/opt/workspace# RMW_IMPLEMENTATION=rmw_opendds_cpp ros2 run examples_rclcpp_minimal_publisher publisher_member_function
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 0'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 1'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 2'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 3'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 4'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 5'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 6'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 7'
root@55870cd9f832:/opt/workspace#  ros2 run examples_rclcpp_minimal_publisher publisher_member_function
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 0'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 1'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 2'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 3'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 4'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 5'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 6'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 7'
```

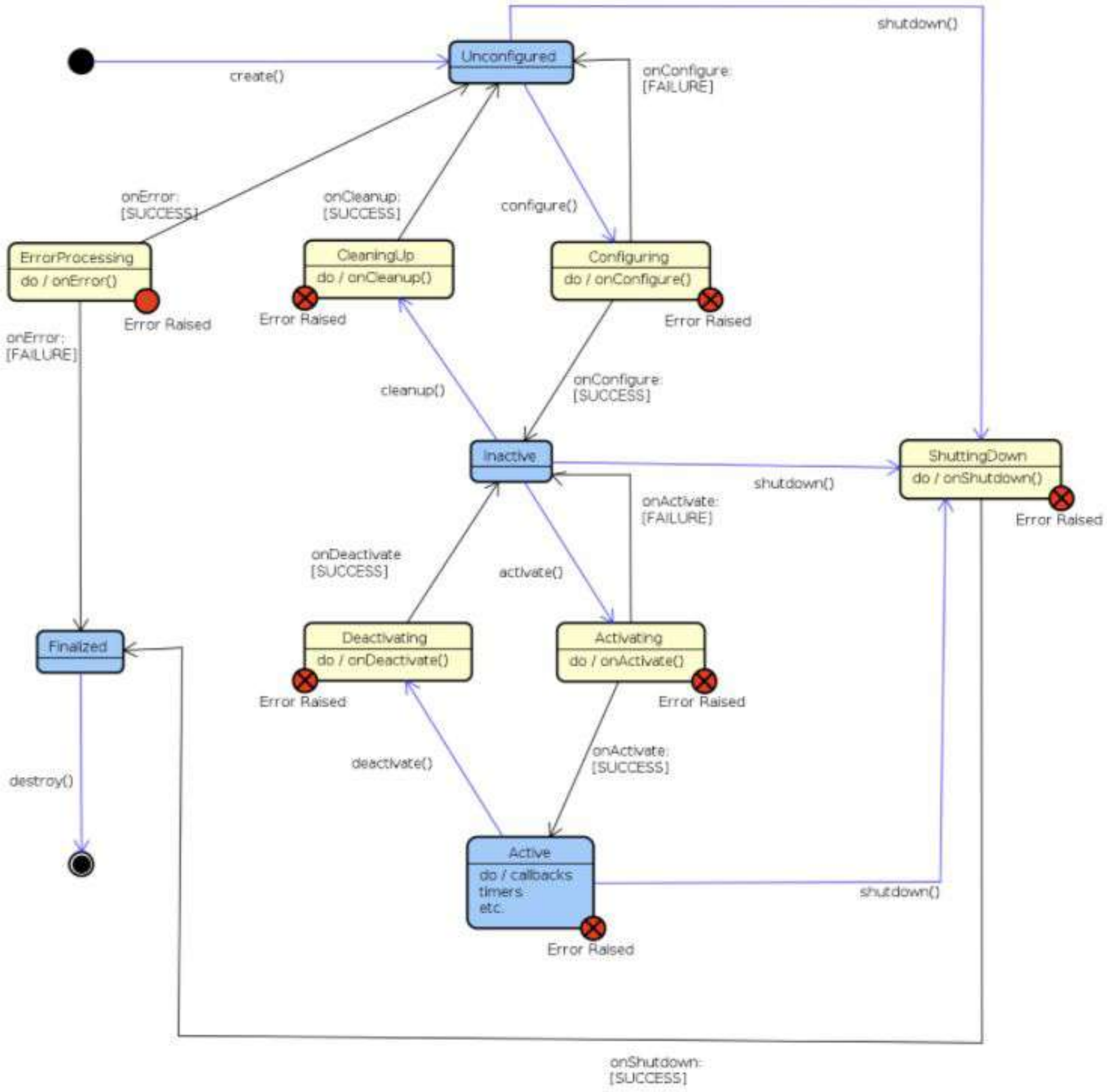


MoveIt2

Why ROS2?: Best new features

Some of the more compelling new features only in ROS2

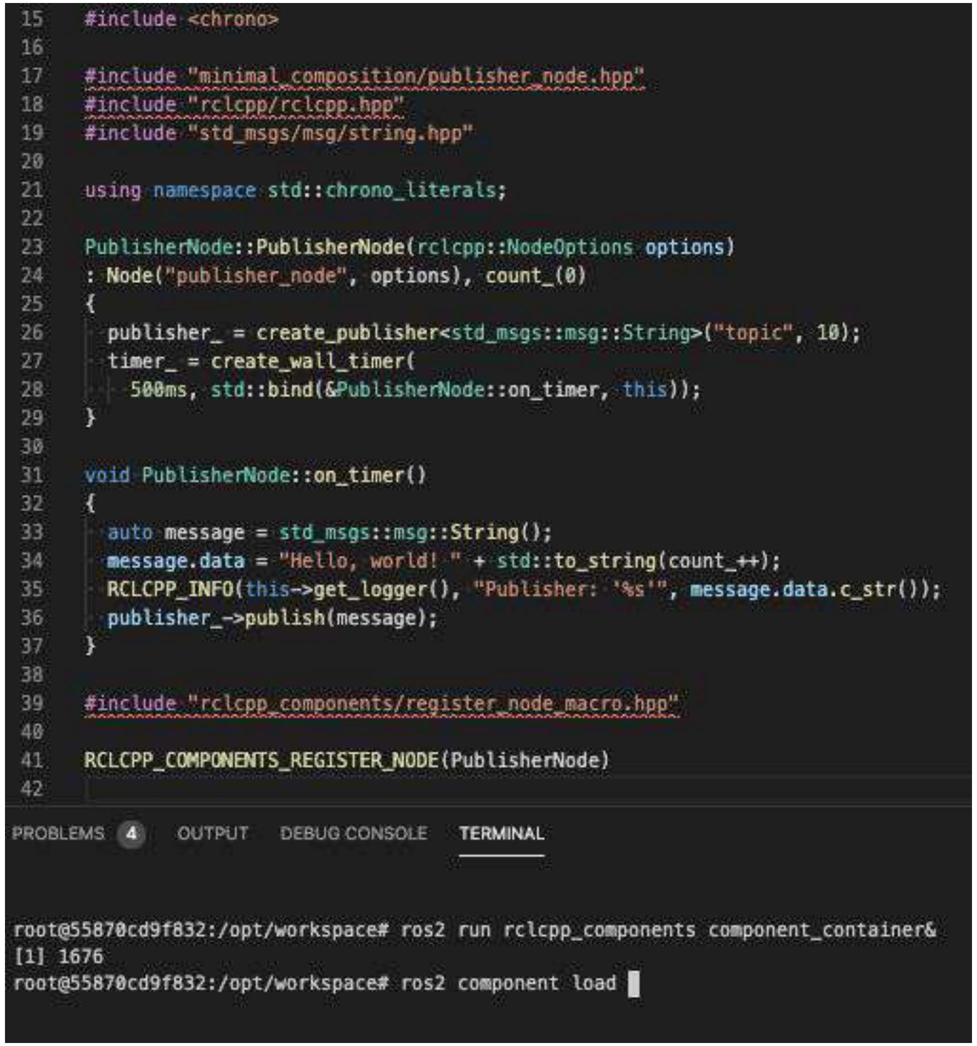
- **Lifecycle nodes** - control the lifecycle/state flow of the node.
- **Node composition** - separation of Nodes and Processes for different runtime configurations
- **Launch** - now has the option for more complicated scenarios (e.g. looping) by promoting Python3 as an option for launch files
- **Embedded/Realtime** - Micro-ROS and VxWorks
- DDS
    - **Discovery** - no centralized message broker required (no ROScore) from RTPS standardized discovery
    - Security
    - QoS



Why ROS2?: Best new features

Some of the more compelling new features only in ROS2

- Lifecycle nodes - control the lifecycle/state flow of the node.
- Node composition - separation of Nodes and Processes for different runtime configurations
- Launch - now has the option for more complicated scenarios (e.g. looping) by promoting Python3 as an option for launch files
- Embedded/Realtime - Micro-ROS and VxWorks
- DDS
  - Discovery - no centralized message broker required (no ROScore) from RTPS standardized discovery
  - Security
  - QoS



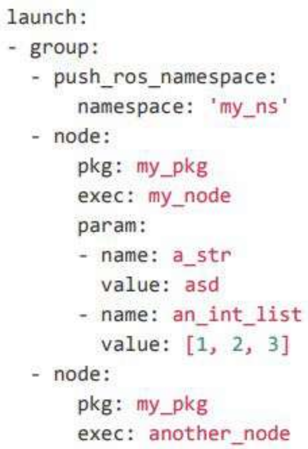
```
#include <chrono>

#include "minimal_composition/publisher_node.hpp"
#include "rclcpp/rclcpp.hpp"
#include "std_msgs/msg/string.hpp"

using namespace std::chrono_literals;

PublisherNode::PublisherNode(rclcpp::NodeOptions options)
: Node("publisher_node", options), count_(0)
{
  publisher_ = create_publisher<std_msgs::msg::String>("topic", 10);
  timer_ = create_wall_timer(
    500ms, std::bind(&PublisherNode::on_timer, this));
}

void PublisherNode::on_timer()
{
  auto message = std_msgs::msg::String();
  message.data = "Hello, world! " + std::to_string(count_++);
  RCLCPP_INFO(this->get_logger(), "Publisher: '%s'", message.data.c_str());
  publisher_->publish(message);
}

#include "rclcpp_components/register_node_macro.hpp"

RCLCPP_COMPONENTS_REGISTER_NODE(PublisherNode)

```

```
root@55870cd9f832:/opt/workspace# ros2 run rclcpp_components component_container&
[1] 1676
root@55870cd9f832:/opt/workspace# ros2 component load 
```

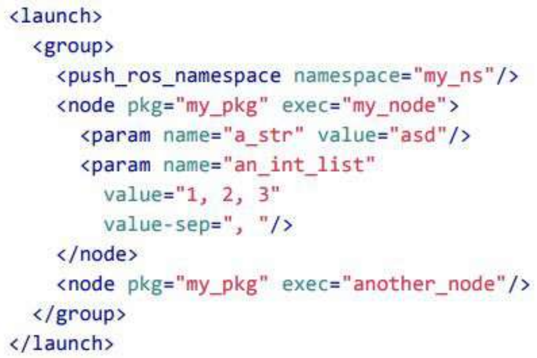
Why ROS2?: Best new features

Some of the more compelling new features only in ROS2

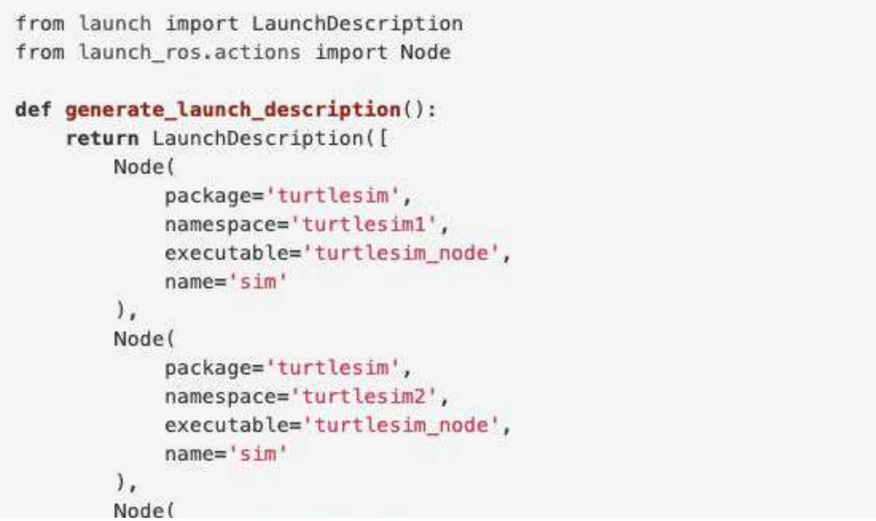
- Lifecycle nodes - control the lifecycle/state flow of the node.
- Node composition - separation of Nodes and Processes for different runtime configurations
- Launch - now has the option for more complicated scenarios (e.g. looping) by promoting Python3 as an option for launch files
- Embedded/Realtime - Micro-ROS and VxWorks
- DDS
  - Discovery - no centralized message broker required (no ROScore) from RTPS standardized discovery
  - Security
  - QoS



```
launch:
- group:
  - push_ros_namespace:
      namespace: 'my_ns'
  - node:
      pkg: my_pkg
      exec: my_node
      param:
      - name: a_str
        value: asd
      - name: an_int_list
        value: [1, 2, 3]
  - node:
      pkg: my_pkg
      exec: another_node
```



```
<launch>
  <group>
    <push_ros_namespace namespace="my_ns"/>
    <node pkg="my_pkg" exec="my_node">
      <param name="a_str" value="asd"/>
      <param name="an_int_list"
        value="1, 2, 3"
        value-sep=", "/>
    </node>
    <node pkg="my_pkg" exec="another_node"/>
  </group>
</launch>
```

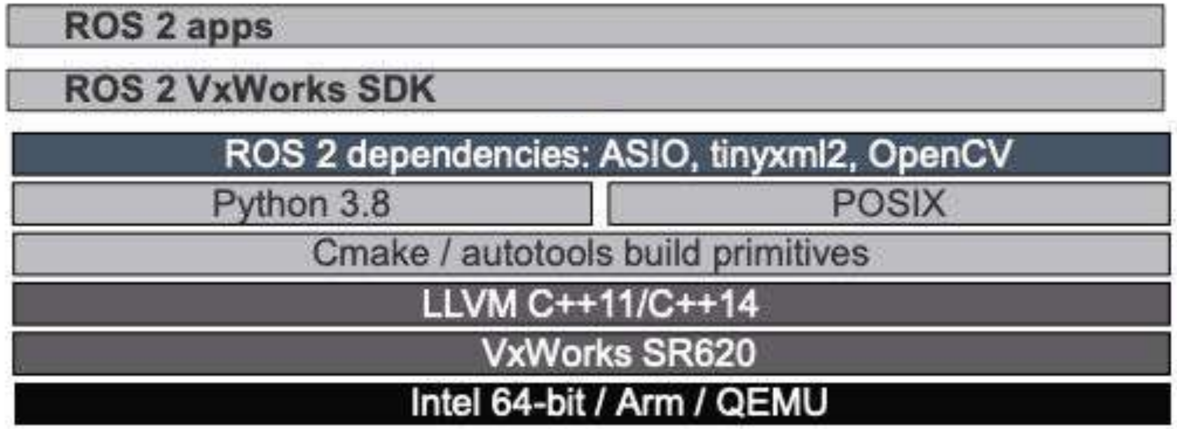


```
from launch import LaunchDescription
from launch_ros.actions import Node

def generate_launch_description():
    return LaunchDescription([
        Node(
            package='turtlesim',
            namespace='turtlesim1',
            executable='turtlesim_node',
            name='sim'
        ),
        Node(
            package='turtlesim',
            namespace='turtlesim2',
            executable='turtlesim_node',
            name='sim'
        ),
        Node(
```

Why ROS2?: Best new features

Some of the more compelling new features only in ROS2

- Lifecycle nodes - control the lifecycle/state flow of the node.
- Node composition - separation of Nodes and Processes for different runtime configurations
- Launch - now has the option for more complicated scenarios (e.g. looping) by promoting Python3 as an option for launch files
- Embedded/Realtime - Micro-ROS and VxWorks
- DDS
  - Discovery - no centralized message broker required (no ROScore) from RTPS standardized discovery
  - Security
  - QoS



ROS 2 apps

ROS 2 VxWorks SDK

ROS 2 dependencies: ASIO, tinyxml2, OpenCV

Python 3.8 | POSIX

Cmake / autotools build primitives

LLVM C++11/C++14

VxWorks SR620

Intel 64-bit / Arm / QEMU

Object Computing, Inc. is a Wind River partner.

Why ROS2?: Best new features

Some of the more compelling new features <u>only</u> in ROS2

- Lifecycle nodes - control the lifecycle/state flow of the node.
- Node composition - separation of Nodes and Processes for different runtime configurations
- Launch - now has the option for more complicated scenarios (e.g. looping) by promoting Python3 as an option for launch files
- Embedded/Realtime - Micro-ROS and VxWorks
- DDS
  - Discovery - no centralized message broker required (no ROScore) from RTPS standardized discovery
  - Security
  - QoS (Quality of Service)

Demystify DDS

You’re a ROS2 developer … but also a DDS user

Questions from the ROS community

Is DDS open source?
What’s difference between DDS and RTPS
What is DDS?
Does hardware matter in DDS spec?
Can DDS connect over the Internet?
Is there any way to use TCP/IP in DDS?
DDS networking config with NAT
Can DDS connect to cloud instances?
Is it possible to use shared memory configuration with DDS?
How does DDS discover other publishers and subscribers?

Why DDS?

Data Distribution Service (DDS) Specification Section 1.2

"Many **real-time applications** have a requirement to model some of their communication patterns as a **pure data-centric exchange**, where applications publish (supply or stream) "data" which is then available to the remote applications that are interested in it. Relevant real-time applications can be found in **C4I, industrial automation, distributed control and simulation, telecom equipment control, sensor networks, and network management systems**.

More generally, any application **requiring (selective) information dissemination is a candidate for a data-driven network architecture**."

……

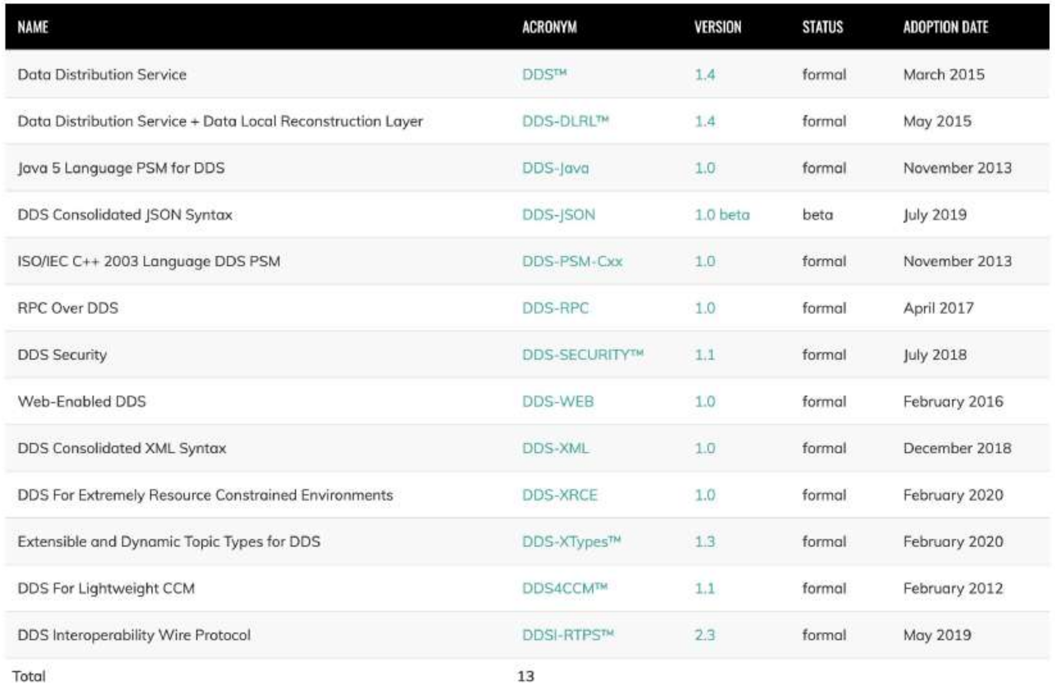
"The purpose of the DDS specification is to define the standardized interfaces and behaviors that enable application portability."

- Data-Centric Publish and Subscribe Approach
- Real-Time Application Domains are a major target
- Applications needing Data-Driven Network Architecture will benefit

https://www.omg.org/spec/category/data-distribution-service/

DDS is a Collection of Specifications

Formally Released Specifications for DDS



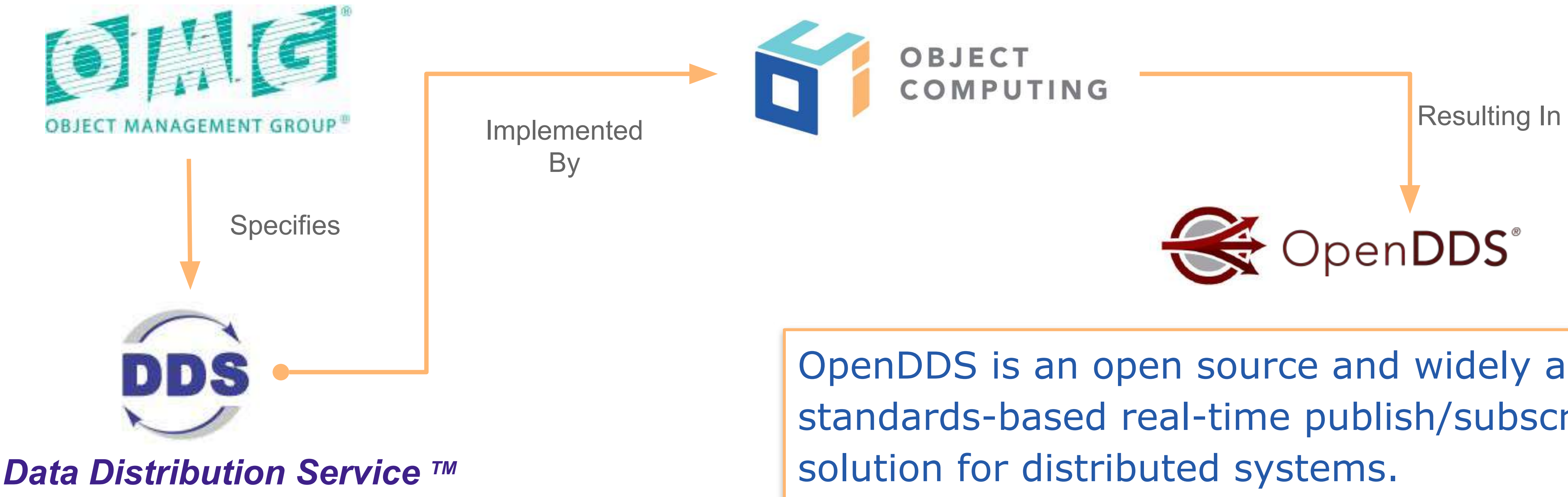
NAME	ACRONYM	VERSION	STATUS	ADOPTION DATE
Data Distribution Service	DDS™	1.4	formal	March 2015
Data Distribution Service + Data Local Reconstruction Layer	DDS-DLRL™	1.4	formal	May 2015
Java 5 Language PSM for DDS	DDS-Java	1.0	formal	November 2013
DDS Consolidated JSON Syntax	DDS-JSON	1.0 beta	beta	July 2019
ISO/IEC C++ 2003 Language DDS PSM	DDS-PSM-Cxx	1.0	formal	November 2013
RPC Over DDS	DDS-RPC	1.0	formal	April 2017
DDS Security	DDS-SECURITY™	1.1	formal	July 2018
Web-Enabled DDS	DDS-WEB	1.0	formal	February 2016
DDS Consolidated XML Syntax	DDS-XML	1.0	formal	December 2018
DDS For Extremely Resource Constrained Environments	DDS-XRCE	1.0	formal	February 2020
Extensible and Dynamic Topic Types for DDS	DDS-XTypes™	1.3	formal	February 2020
DDS For Lightweight CCM	DDS4CCM™	1.1	formal	February 2012
DDS Interoperability Wire Protocol	DDSI-RTPS™	2.3	formal	May 2019
Total	13			

https://www.omg.org/spec/category/data-distribution-service/

Specs most relevant to ROS2 to various degrees:

- Data Distribution Service (Core spec)
- DDS Security
- DDS Interoperability Wire Protocol (aka RTPS)
- RPC (Remote Procedure Call) Over DDS
- Interface Definition Language (IDL)
- DDS For Extremely Resource Constrained Environments (XRCE)
- Extensible and Dynamic Topic Types for DDS (XTypes)

What is OpenDDS?



OBJECT MANAGEMENT GROUP

Specifies

DDS

Data Distribution Service ™

Implemented By

OBJECT COMPUTING

Resulting In

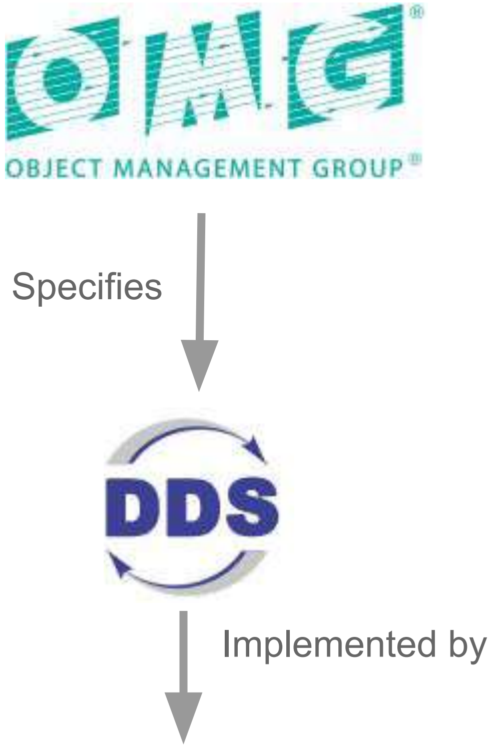
OpenDDS

https://www.omg.org/spec/category/data-distribution-service/

OpenDDS is an open source and widely adopted standards-based real-time publish/subscribe solution for distributed systems.

- opendds.org
- https://github.com/objectcomputing/OpenDDS

What makes OCI unique among DDS providers?



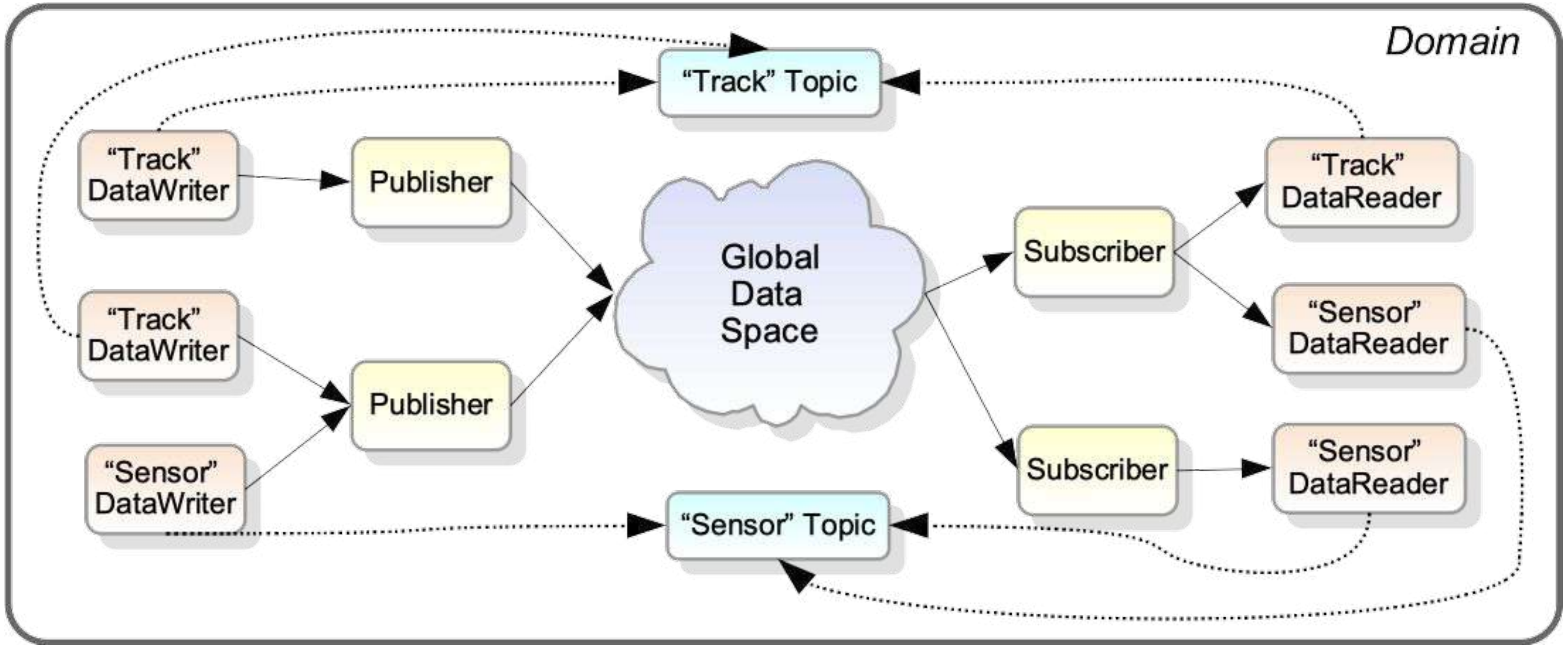
DDS/RTPS Implementations

- OCI - OpenDDS
- Other DDS/RTPS vendors

Core and DDS Services	**Other OCI technology services**
Full Lifecycle Engineering Services +	AI/ML +
DDS knowledge +	Cloud +
Completely Open Source DDS Product +	Microservices +
Complete DDS Implementation including security +	Blockchain

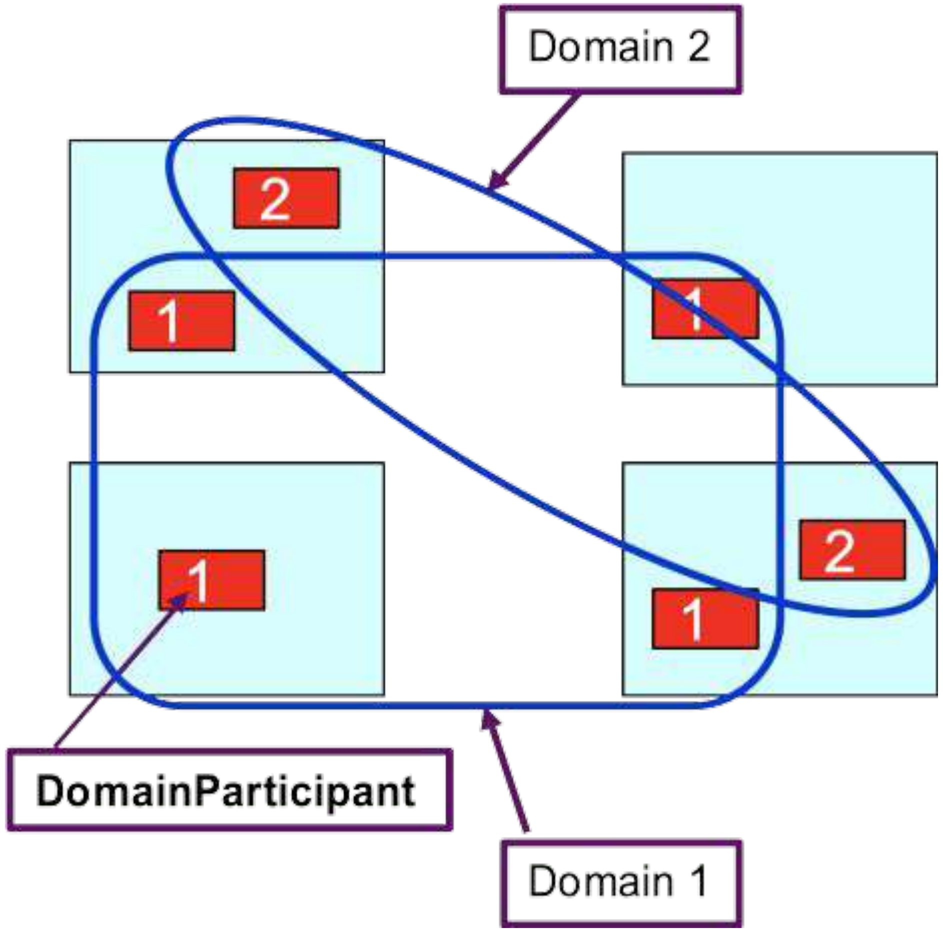
Basic Components of a DDS System

- DataWriter - creates Samples of a single application data type
- DataReader - receives Samples of a single application data type
- Publisher - applies control and restrictions to the flow of data from DataWriters
- Subscriber - applies control and restrictions to the flow of data from DataReaders
- Topic - is associated with a single data type and the distribution and availability of samples



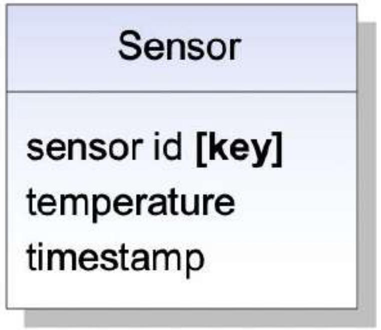
Some more components of a DDS System

- Domain
  - Independent global data space
  - Identified by numeric value called Domain ID
- Domain Participant
  - Only participants in the same domain can communicate



Samples and Instances

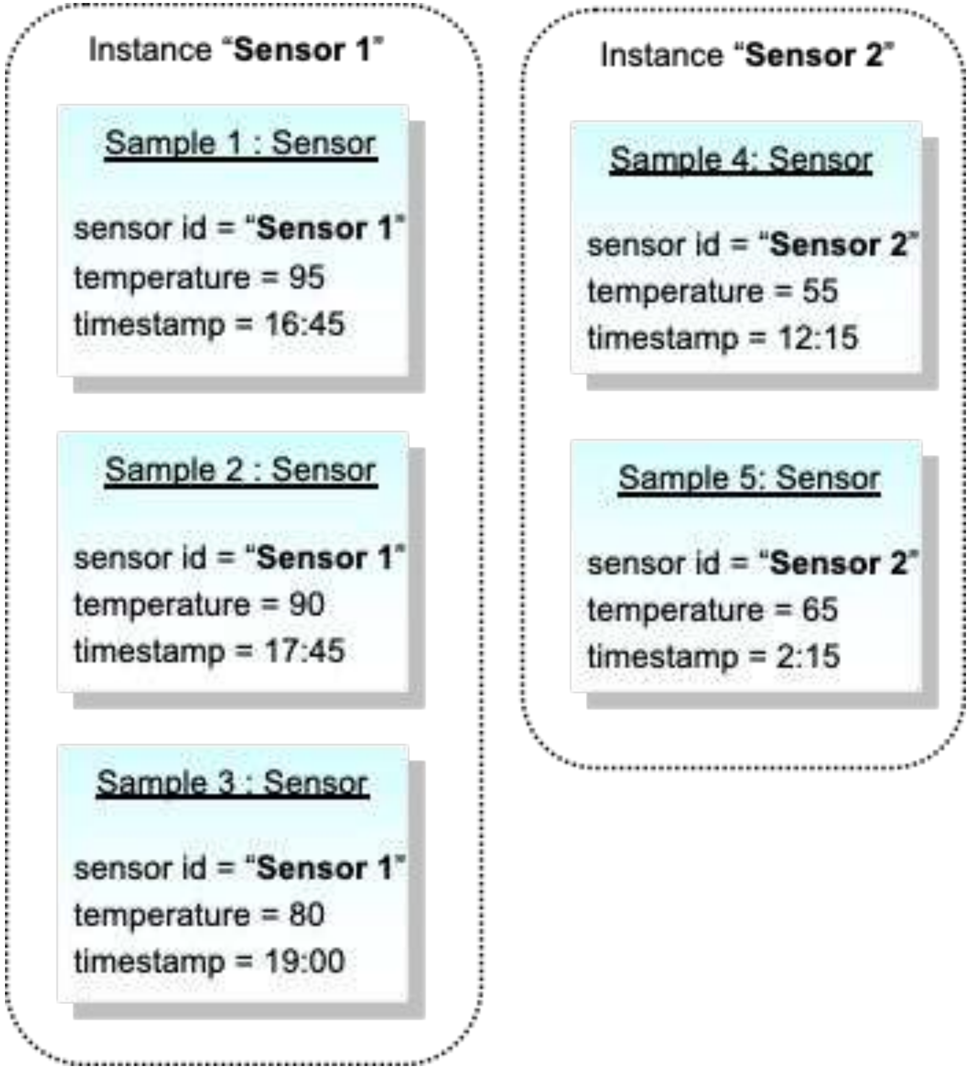
- Sample
  - Individual data element
  - All samples published have the same type



Sensor

sensor id **[key]**
temperature
timestamp

- Instance
  - Set of samples identified by the same key value



Instance "**Sensor 1**"

Sample 1 : Sensor
sensor id = "**Sensor 1**"
temperature = 95
timestamp = 16:45

Sample 2 : Sensor
sensor id = "**Sensor 1**"
temperature = 90
timestamp = 17:45

Sample 3 : Sensor
sensor id = "**Sensor 1**"
temperature = 80
timestamp = 19:00

Instance "**Sensor 2**"

Sample 4: Sensor
sensor id = "**Sensor 2**"
temperature = 55
timestamp = 12:15

Sample 5: Sensor
sensor id = "**Sensor 2**"
temperature = 65
timestamp = 2:15

Demystify DDS

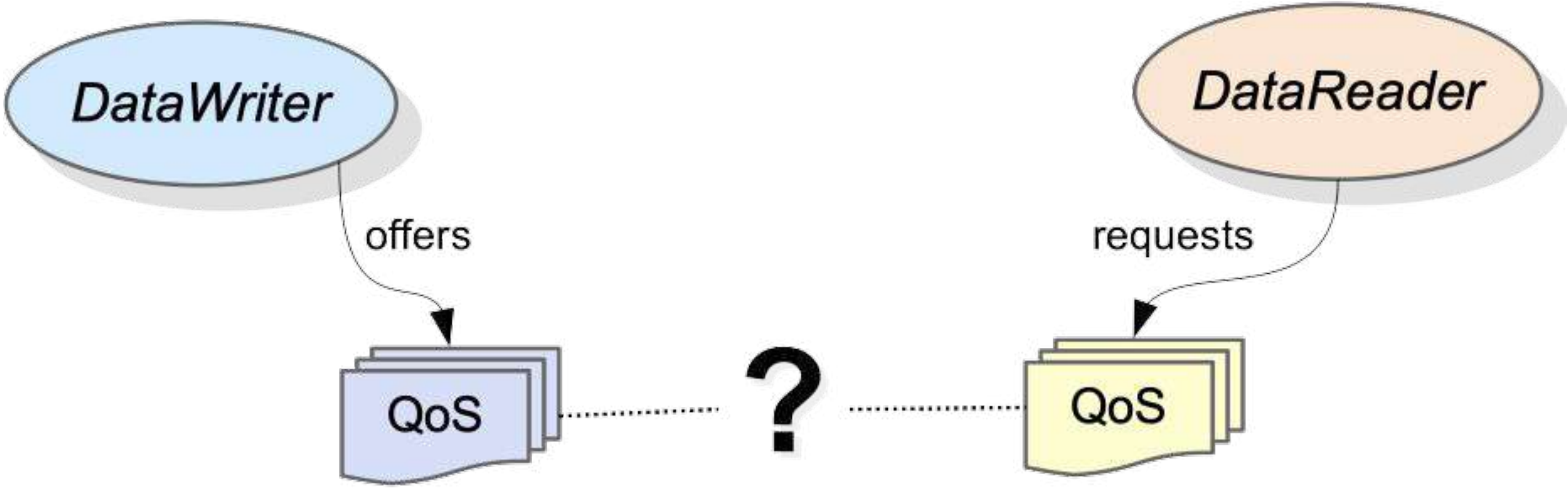
DDS QoS

- DDS specification defines 20+ QoS policies
- DDS QoS policies can be applied to entities
  - Topic
  - DataWriter
  - DataReader
  - Publisher
  - Subscriber
  - DomainParticipant
- Each entity type supports a subset of the policies

QoS Policy	QoS Policy
DURABILITY	USER DATA
HISTORY	TOPIC DATA
LIFESPAN	GROUP DATA
WRITER DATA LIFECYCLE	PARTITION
READER DATA LIFECYCLE	PRESENTATION
ENTITY FACTORY	DESTINATION ORDER
RESOURCE LIMITS	OWNERSHIP
RELIABILITY	OWNERSHIP STRENGTH
TIME BASED FILTER	LIVELINESS
DEADLINE	LATENCY BUDGET
TRANSPORT PRIORITY	

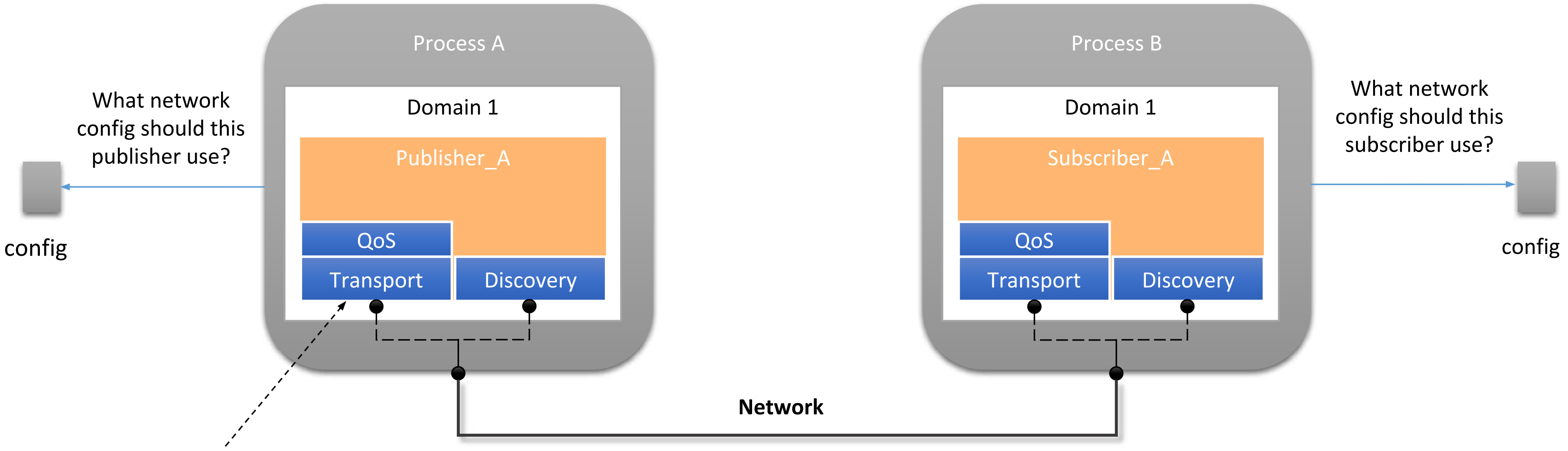
DDS QoS

- DataWriter offers its QoS policies to DataReaders
- DataReader requests the QoS policies it needs
- If the relevant requested policies are not compatible with those offered, communication is not established



Demystify DDS

Pluggable Transport in OpenDDS



Pluggable Transport
TCP, UDP, Multicast,
Shared Memory,
RTPS_UDP

DDS Security Specification

Object Management Group's specification for DDS Security includes:

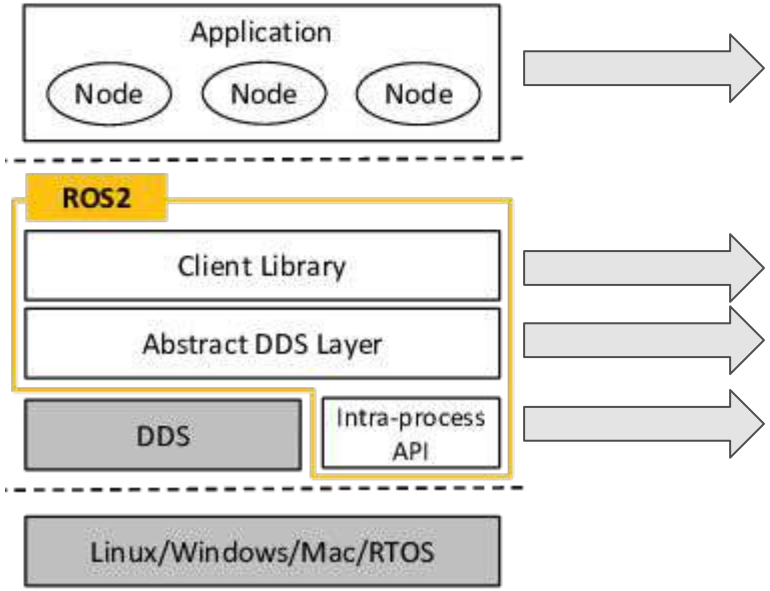
- Authentication of Participating Applications
    - Application identities determined by certificates signed by a common CA
- Access Control by Topic
    - Configuration files (signed by CA) determine which applications have access (read/write/both) to which topics
- Data Protection via Encryption and/or Message Authentication
    - Topic-by-topic configuration determines whether to encrypt or only sign network messages
    - Scope of data protection is also configurable: payload only or including headers

ROS2 using DDS

Questions from the ROS community

How is ROS2 using DDS?
Are ROS2 and DDS ready for product quality deployments?
Can ROS communicate between machines if they are running different DDS implementations?
What's the performance difference between ROS1's TCPROS and UDPROS and DDS/RTPS?
How does ROS2 interface DDS? Is it a separate process or an API call?
What is DDS XRCE and how does it relate to ROS2?
How does ROS2 select the default QoS profile ?
Is it possible to have ROS2 with DDS communicate to applications only using DDS?
How are ROS2 .msg files converted into .idl files for DDS?
How does ROS2 handle services and actions using DDS?
How do I change the DDS implementation?

ROS2 employs a generic middleware interface



Your ROS2 Nodes

RCLCPP/RCLPY/RCL

RMW (ROS Middleware)

DDS

How is ROS2 using DDS?

How does ROS2 interface DDS? Is it a separate process or an API call?

How do I change the DDS implementation?

ROS2 using DDS

RMW (ROS MiddleWare)

- C API http://docs.ros2.org/eloquent/api/rmw/
- The many RMW vendor implementations are usually comprised of RMW and ROSIDL Typesupport packages
- DDS standards continuously evolving and so is the RMW
    - match standards
    - fix bugs
    - Improve performance
- DDS topics are used for
    - ROS2 pub/sub (also topics)
    - client/service
    - action

ROS2 Implements a subset of the QoS in DDS

- History - how many samples (messages) to keep
- Reliability - best effort vs. reliable
- Durability - late joining subscribers can see previously published messages
- Deadline - publisher minimum rate and subscriber minimum wait time
- Lifespan - how long the message is valid
- Liveliness - Subscriber can request that Publisher indicates that it is alive at a configurable interval

ROS2 using DDS

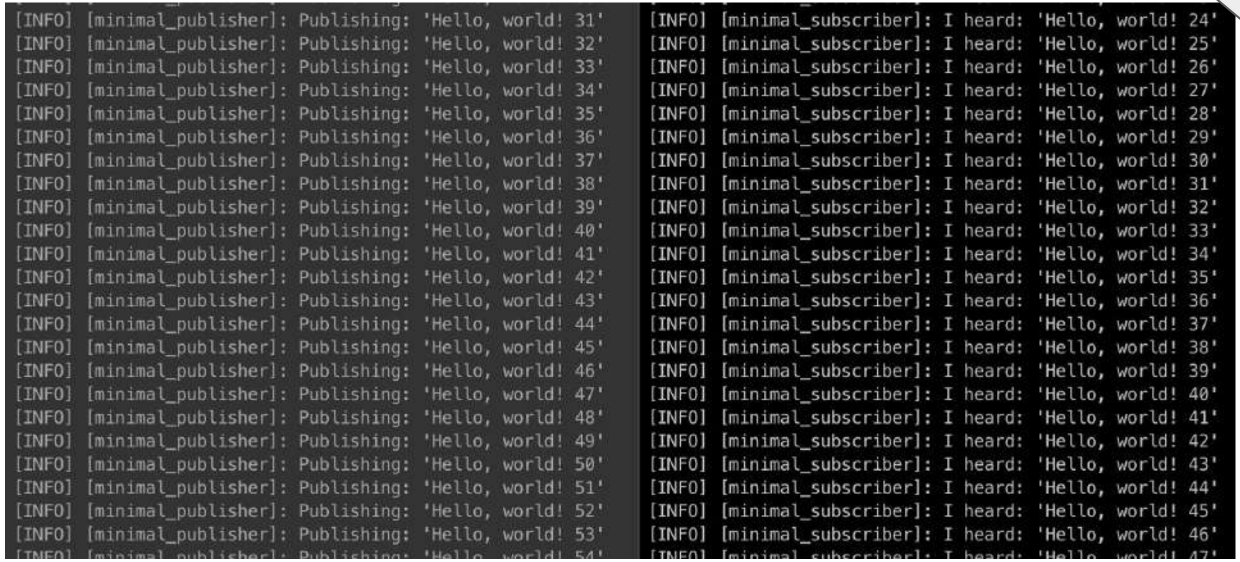
Communication troubleshooting

- Nodes not discovered?
- Nodes QoS not compatible?
- Comms failing between two computers?

Wireshark RTPS built-in filters

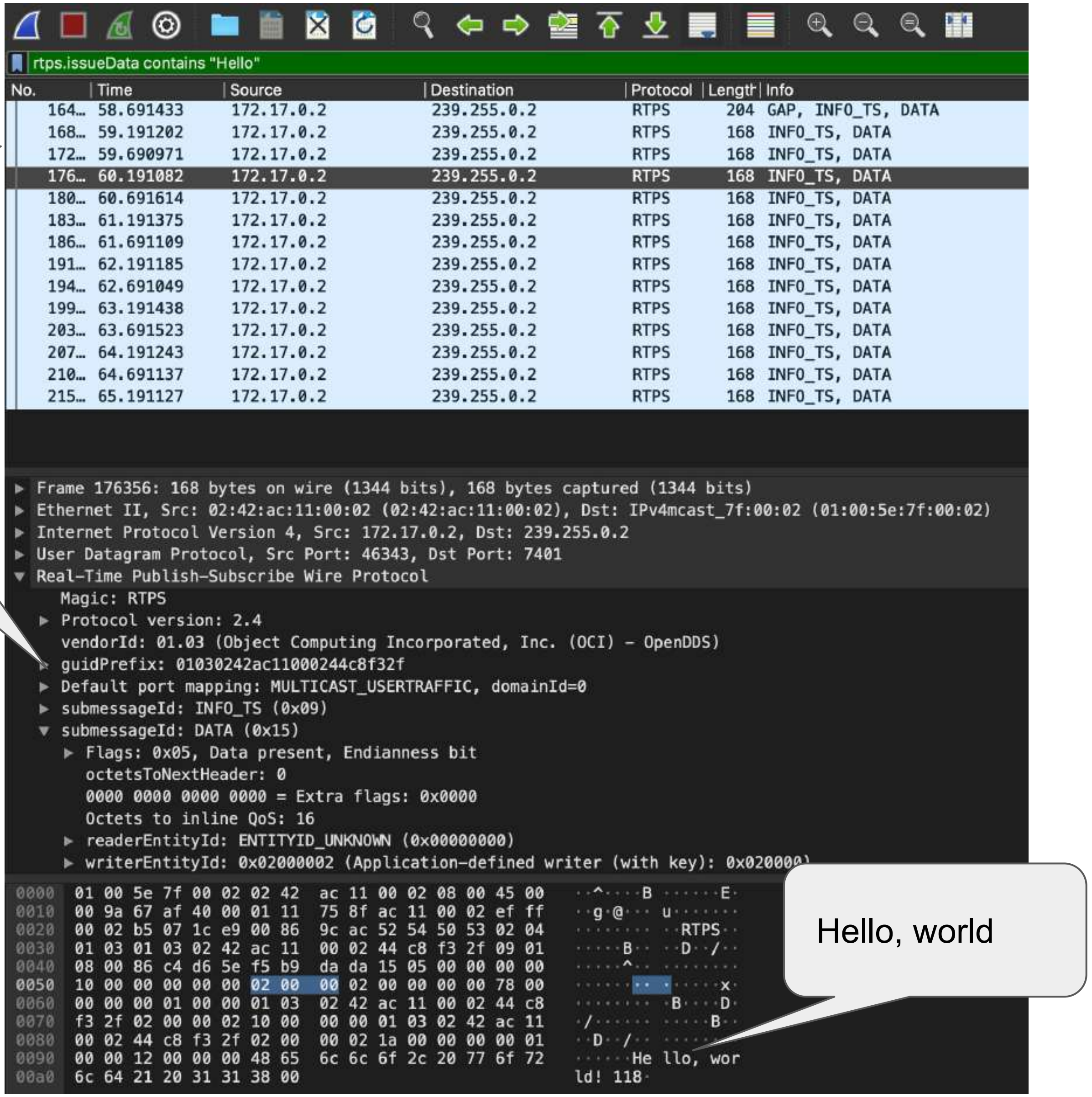
RTPS Vendor

ROS minimal pub/sub



```
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 31'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 32'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 33'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 34'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 35'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 36'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 37'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 38'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 39'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 40'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 41'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 42'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 43'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 44'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 45'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 46'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 47'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 48'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 49'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 50'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 51'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 52'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 53'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 54'
```

```
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 24'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 25'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 26'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 27'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 28'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 29'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 30'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 31'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 32'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 33'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 34'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 35'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 36'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 37'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 38'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 39'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 40'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 41'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 42'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 43'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 44'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 45'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 46'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 47'
```



rtps.issueData contains "Hello"

No.	Time	Source	Destination	Protocol	Length	Info
164…	58.691433	172.17.0.2	239.255.0.2	RTPS	204	GAP, INFO_TS, DATA
168…	59.191202	172.17.0.2	239.255.0.2	RTPS	168	INFO_TS, DATA
172…	59.690971	172.17.0.2	239.255.0.2	RTPS	168	INFO_TS, DATA
176…	60.191082	172.17.0.2	239.255.0.2	RTPS	168	INFO_TS, DATA
180…	60.691614	172.17.0.2	239.255.0.2	RTPS	168	INFO_TS, DATA
183…	61.191375	172.17.0.2	239.255.0.2	RTPS	168	INFO_TS, DATA
186…	61.691109	172.17.0.2	239.255.0.2	RTPS	168	INFO_TS, DATA
191…	62.191185	172.17.0.2	239.255.0.2	RTPS	168	INFO_TS, DATA
194…	62.691049	172.17.0.2	239.255.0.2	RTPS	168	INFO_TS, DATA
199…	63.191438	172.17.0.2	239.255.0.2	RTPS	168	INFO_TS, DATA
203…	63.691523	172.17.0.2	239.255.0.2	RTPS	168	INFO_TS, DATA
207…	64.191243	172.17.0.2	239.255.0.2	RTPS	168	INFO_TS, DATA
210…	64.691137	172.17.0.2	239.255.0.2	RTPS	168	INFO_TS, DATA
215…	65.191127	172.17.0.2	239.255.0.2	RTPS	168	INFO_TS, DATA

```
▶ Frame 176356: 168 bytes on wire (1344 bits), 168 bytes captured (1344 bits)
▶ Ethernet II, Src: 02:42:ac:11:00:02 (02:42:ac:11:00:02), Dst: IPv4mcast_7f:00:02 (01:00:5e:7f:00:02)
▶ Internet Protocol Version 4, Src: 172.17.0.2, Dst: 239.255.0.2
▶ User Datagram Protocol, Src Port: 46343, Dst Port: 7401
▼ Real-Time Publish-Subscribe Wire Protocol
    Magic: RTPS
  ▶ Protocol version: 2.4
    vendorId: 01.03 (Object Computing Incorporated, Inc. (OCI) - OpenDDS)
  ▶ guidPrefix: 01030242ac11000244c8f32f
  ▶ Default port mapping: MULTICAST_USERTRAFFIC, domainId=0
  ▶ submessageId: INFO_TS (0x09)
  ▼ submessageId: DATA (0x15)
    ▶ Flags: 0x05, Data present, Endianness bit
      octetsToNextHeader: 0
      0000 0000 0000 0000 = Extra flags: 0x0000
      Octets to inline QoS: 16
    ▶ readerEntityId: ENTITYID_UNKNOWN (0x00000000)
    ▶ writerEntityId: 0x02000002 (Application-defined writer (with key): 0x020000)
```

```
0000  01 00 5e 7f 00 02 02 42  ac 11 00 02 08 00 45 00   ··^····B ······E·
0010  00 9a 67 af 40 00 01 11  75 8f ac 11 00 02 ef ff   ··g·@··· u·······
0020  00 02 b5 07 1c e9 00 86  9c ac 52 54 50 53 02 04   ········ ··RTPS··
0030  01 03 01 03 02 42 ac 11  00 02 44 c8 f3 2f 09 01   ·····B·· ··D··/··
0040  08 00 86 c4 d6 5e f5 b9  da da 15 05 00 00 00 00   ·····^·· ········
0050  10 00 00 00 00 00 02 00  00 02 00 00 00 00 78 00   ········ ······x·
0060  00 00 00 01 00 00 01 03  02 42 ac 11 00 02 44 c8   ········ ·B····D·
0070  f3 2f 02 00 00 02 10 00  00 00 01 03 02 42 ac 11   ·/······ ·····B··
0080  00 02 44 c8 f3 2f 02 00  00 02 1a 00 00 00 00 01   ··D··/·· ········
0090  00 00 12 00 00 00 48 65  6c 6c 6f 2c 20 77 6f 72   ······He llo, wor
00a0  6c 64 21 20 31 31 38 00                            ld! 118·
```

Hello, world

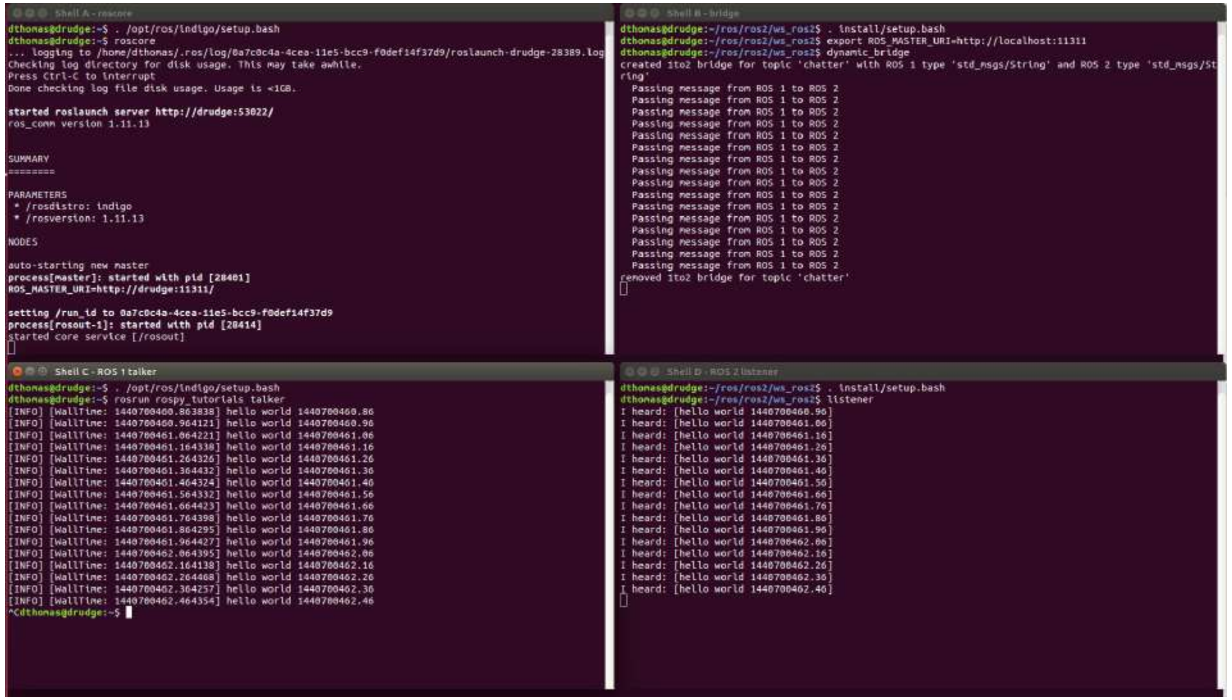
Migrating to ROS2

But is it ready?

Migrating to ROS2

Ways to migrate

- Use ROS1_bridge
  - Connects ROS1 messaging to ROS2 in an automatic fashion
  - Must be rebuilt for custom messages
- Mixed ROS1/ROS2 node
  - APIs between ROS1 and ROS2 intentionally do not conflict
  - More convenient to add ROS2 functionality over time
- Start fresh with a bundled set of packages such as:
  - Nav2
  - Autoware.Auto
  - MoveIt2

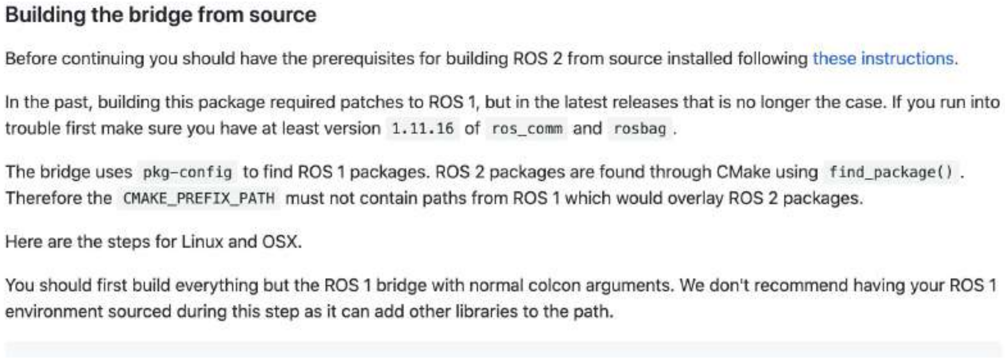


```
dthomas@drudge:~$ . /opt/ros/indigo/setup.bash
dthomas@drudge:~$ roscore
... logging to /home/dthomas/.ros/log/0a7c0c4a-4cea-11e5-bcc9-f0def14f37d9/roslaunch-drudge-28389.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://drudge:53022/
ros_comm version 1.11.13


SUMMARY
========

PARAMETERS
 * /rosdistro: indigo
 * /rosversion: 1.11.13

NODES

auto-starting new master
process[master]: started with pid [28401]
ROS_MASTER_URI=http://drudge:11311/

setting /run_id to 0a7c0c4a-4cea-11e5-bcc9-f0def14f37d9
process[rosout-1]: started with pid [28414]
started core service [/rosout]
```

```
dthomas@drudge:~/ros/ros2/ws_ros2$ . install/setup.bash
dthomas@drudge:~/ros/ros2/ws_ros2$ export ROS_MASTER_URI=http://localhost:11311
dthomas@drudge:~/ros/ros2/ws_ros2$ dynamic_bridge
created 1to2 bridge for topic 'chatter' with ROS 1 type 'std_msgs/String' and ROS 2 type 'std_msgs/St
ring'
  Passing message from ROS 1 to ROS 2
  Passing message from ROS 1 to ROS 2
  Passing message from ROS 1 to ROS 2
  Passing message from ROS 1 to ROS 2
  Passing message from ROS 1 to ROS 2
  Passing message from ROS 1 to ROS 2
  Passing message from ROS 1 to ROS 2
  Passing message from ROS 1 to ROS 2
  Passing message from ROS 1 to ROS 2
  Passing message from ROS 1 to ROS 2
  Passing message from ROS 1 to ROS 2
  Passing message from ROS 1 to ROS 2
  Passing message from ROS 1 to ROS 2
  Passing message from ROS 1 to ROS 2
  Passing message from ROS 1 to ROS 2
  Passing message from ROS 1 to ROS 2
removed 1to2 bridge for topic 'chatter'
```

```
dthomas@drudge:~$ . /opt/ros/indigo/setup.bash
dthomas@drudge:~$ rosrun rospy_tutorials talker
[INFO] [WallTime: 1440700460.863838] hello world 1440700460.86
[INFO] [WallTime: 1440700460.964121] hello world 1440700460.96
[INFO] [WallTime: 1440700461.064221] hello world 1440700461.06
[INFO] [WallTime: 1440700461.164338] hello world 1440700461.16
[INFO] [WallTime: 1440700461.264326] hello world 1440700461.26
[INFO] [WallTime: 1440700461.364432] hello world 1440700461.36
[INFO] [WallTime: 1440700461.464324] hello world 1440700461.46
[INFO] [WallTime: 1440700461.564332] hello world 1440700461.56
[INFO] [WallTime: 1440700461.664423] hello world 1440700461.66
[INFO] [WallTime: 1440700461.764398] hello world 1440700461.76
[INFO] [WallTime: 1440700461.864295] hello world 1440700461.86
[INFO] [WallTime: 1440700461.964427] hello world 1440700461.96
[INFO] [WallTime: 1440700462.064395] hello world 1440700462.06
[INFO] [WallTime: 1440700462.164138] hello world 1440700462.16
[INFO] [WallTime: 1440700462.264468] hello world 1440700462.26
[INFO] [WallTime: 1440700462.364257] hello world 1440700462.36
[INFO] [WallTime: 1440700462.464354] hello world 1440700462.46
^Cdthomas@drudge:~$
```

```
dthomas@drudge:~/ros/ros2/ws_ros2$ . install/setup.bash
dthomas@drudge:~/ros/ros2/ws_ros2$ listener
I heard: [hello world 1440700460.96]
I heard: [hello world 1440700461.06]
I heard: [hello world 1440700461.16]
I heard: [hello world 1440700461.26]
I heard: [hello world 1440700461.36]
I heard: [hello world 1440700461.46]
I heard: [hello world 1440700461.56]
I heard: [hello world 1440700461.66]
I heard: [hello world 1440700461.76]
I heard: [hello world 1440700461.86]
I heard: [hello world 1440700461.96]
I heard: [hello world 1440700462.06]
I heard: [hello world 1440700462.16]
I heard: [hello world 1440700462.26]
I heard: [hello world 1440700462.36]
I heard: [hello world 1440700462.46]
```



Building the bridge from source

Before continuing you should have the prerequisites for building ROS 2 from source installed following these instructions.

In the past, building this package required patches to ROS 1, but in the latest releases that is no longer the case. If you run into trouble first make sure you have at least version `1.11.16` of `ros_comm` and `rosbag`.

The bridge uses `pkg-config` to find ROS 1 packages. ROS 2 packages are found through CMake using `find_package()`. Therefore the `CMAKE_PREFIX_PATH` must not contain paths from ROS 1 which would overlay ROS 2 packages.

Here are the steps for Linux and OSX.

You should first build everything but the ROS 1 bridge with normal colcon arguments. We don't recommend having your ROS 1 environment sourced during this step as it can add other libraries to the path.

https://github.com/ipa-hsd/action_bridge

New use cases for ROS2 with DDS

Questions from the ROS community

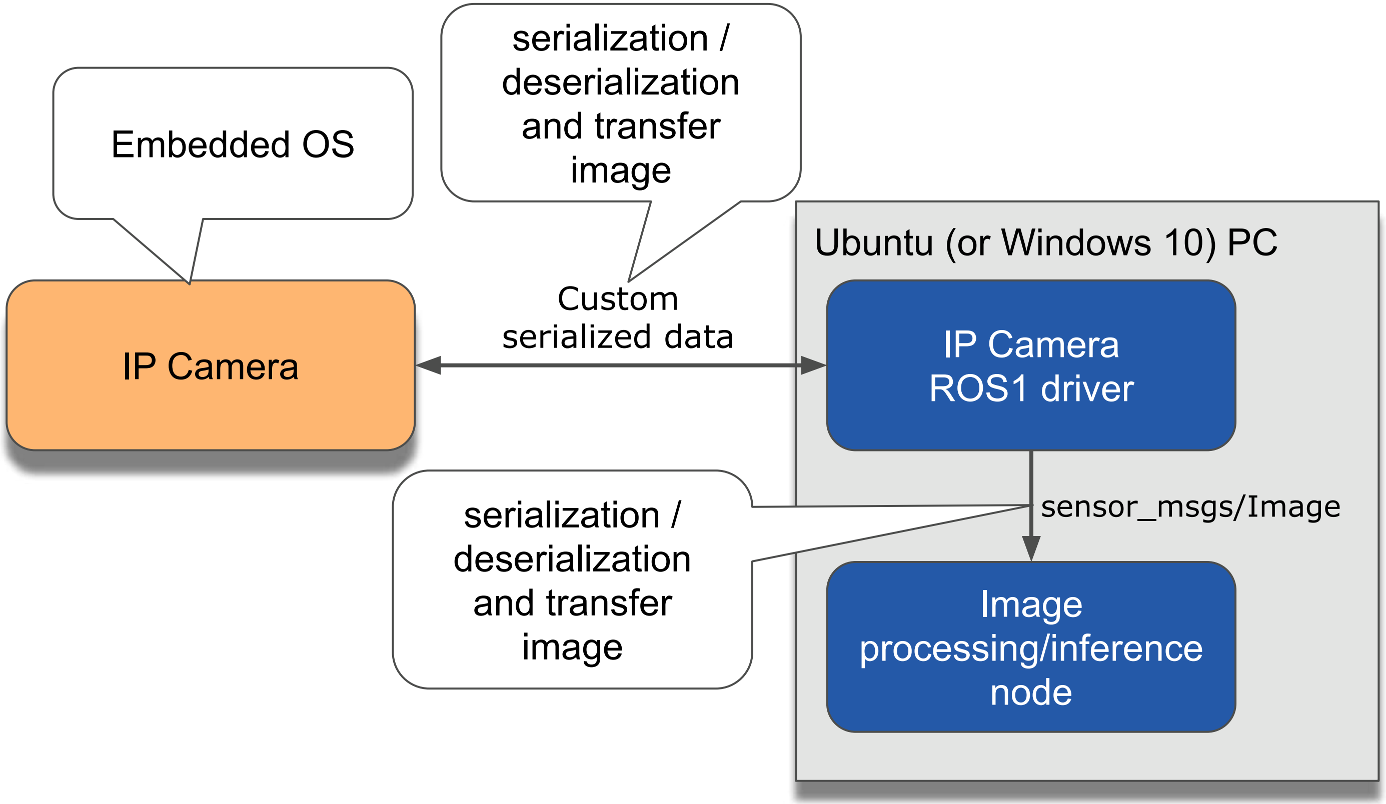
How can I connect ROS2 nodes between different subnets?
Can my ROS2 node connect to the cloud?
Can I connect nodes over the Internet?
Can my ROS2 nodes communicate between machines on different networks with DDS?
Is there any way to use TCP/IP in ROS2 DDS communication?
Does ROS2 with DDS work between NAT firewalls?
Can ROS2 nodes connect to cloud instances remotely?

OpenDDS Case Studies

- *Embedded DDS networks*
- *Edge-to-Cloud Communications*
- *Ground Vehicle Integration*

Embedded Edge processing Use Case

What is typical for ROS1?



Embedded OS

serialization / deserialization and transfer image

Custom serialized data

IP Camera

Ubuntu (or Windows 10) PC

IP Camera ROS1 driver

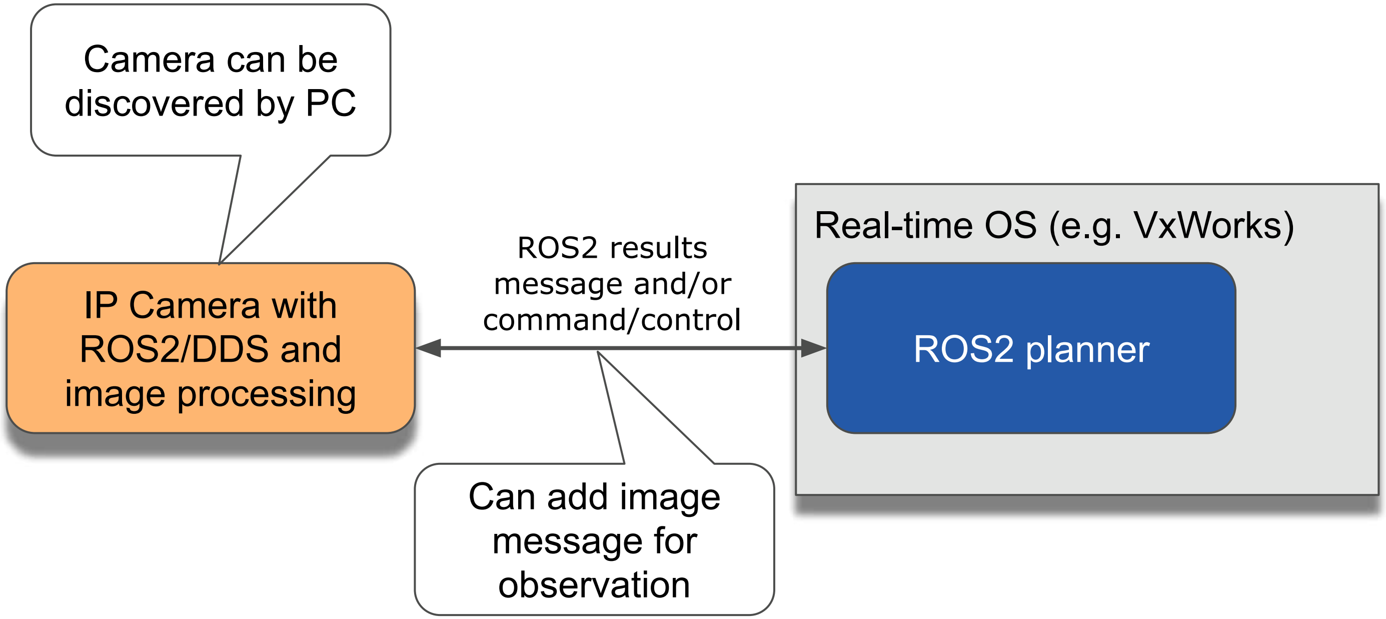
sensor_msgs/Image

serialization / deserialization and transfer image

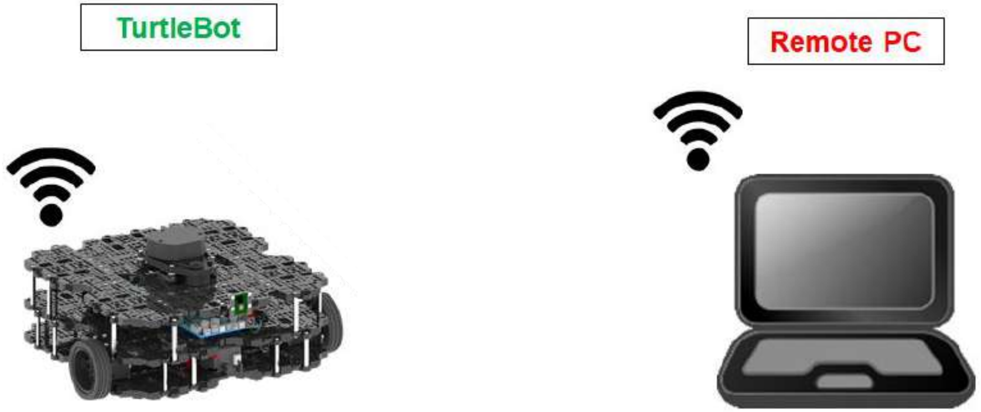
Image processing/inference node

Embedded Edge processing Use Case

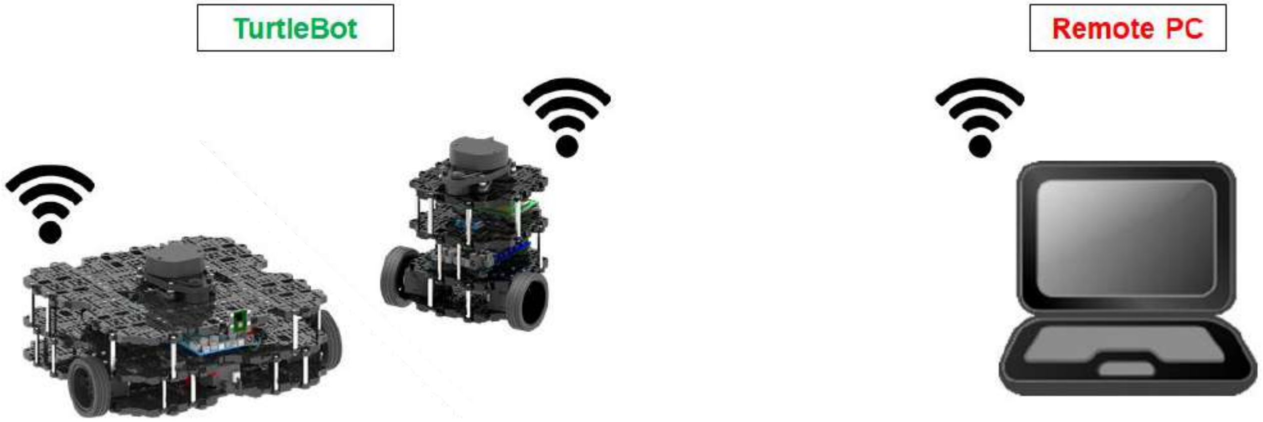
What could we do in ROS2?


Camera can be discovered by PC
IP Camera with ROS2/DDS and image processing
ROS2 results message and/or command/control
Can add image message for observation
Real-time OS (e.g. VxWorks)
ROS2 planner

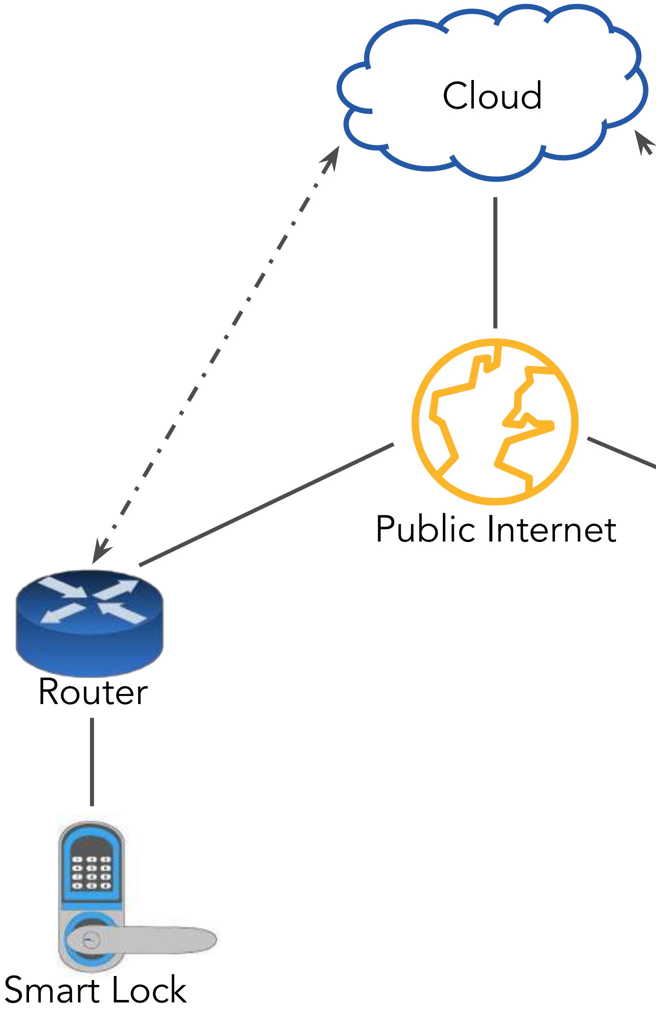
In ROS1, we’re used to this...



And sometimes this ...



Example: IoT Smart Lock



Cloud services provide monitoring, control, user and device management, firmware updates, etc.

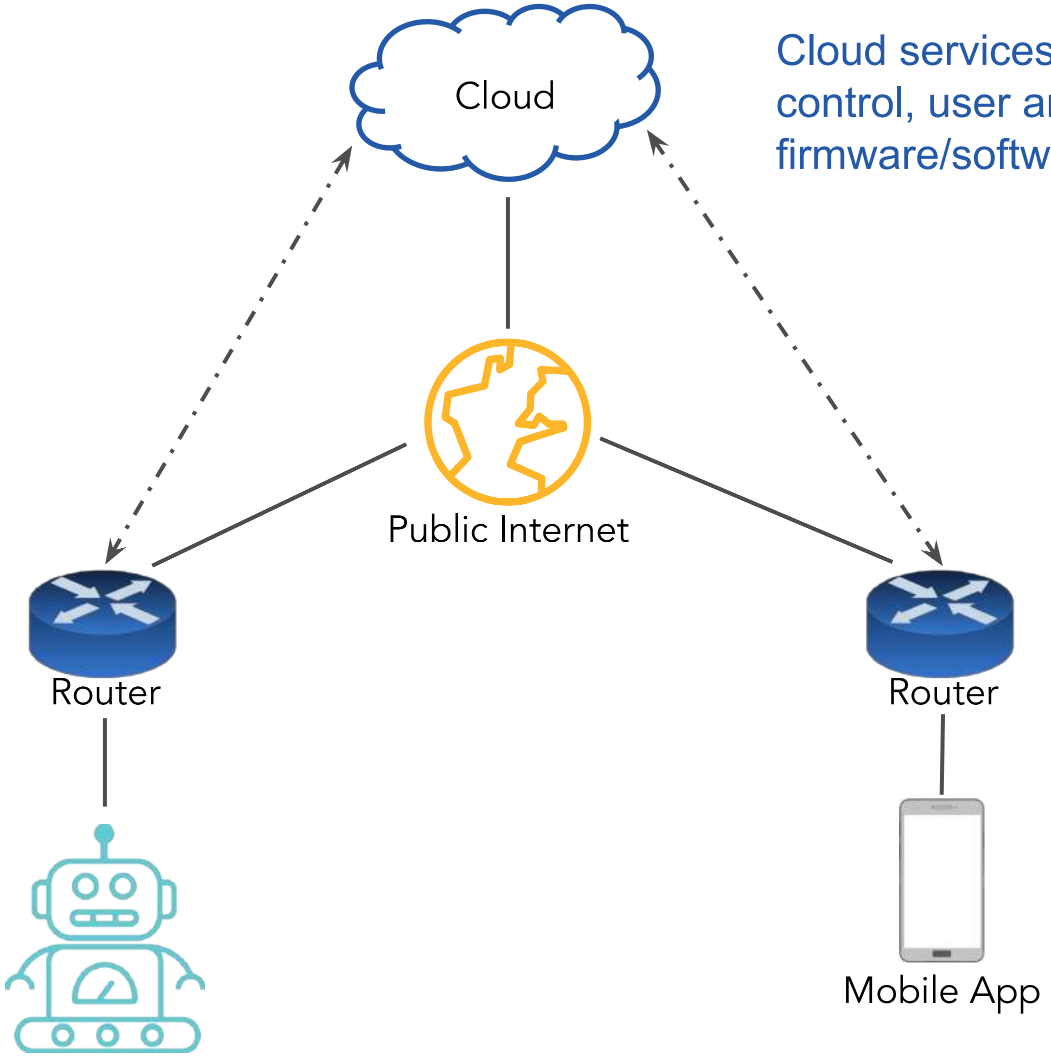
Smart Lock reports its state (locked/unlocked) and responds to control/configuration from authorized users

Mobile app communicates with one or more Smart Locks/robots and cloud services to monitor and manage devices

Example: IoT S ck

ROBOT

Cloud services provide monitoring, control, user and device management, firmware/software updates, etc.



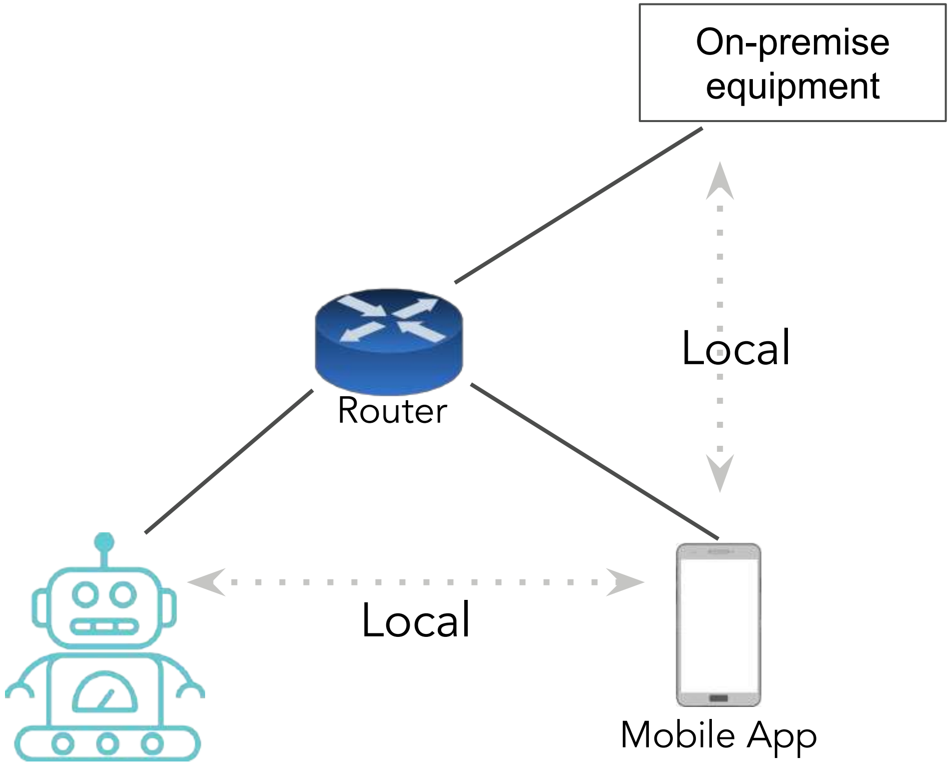
Robot reports its state and responds to control/configuration from authorized users

Mobile app communicates with one or more robots and cloud services to monitor and manage devices

Use Case #1: Leveraging DDS for Direct Connectivity

Robot and mobile app on same local network

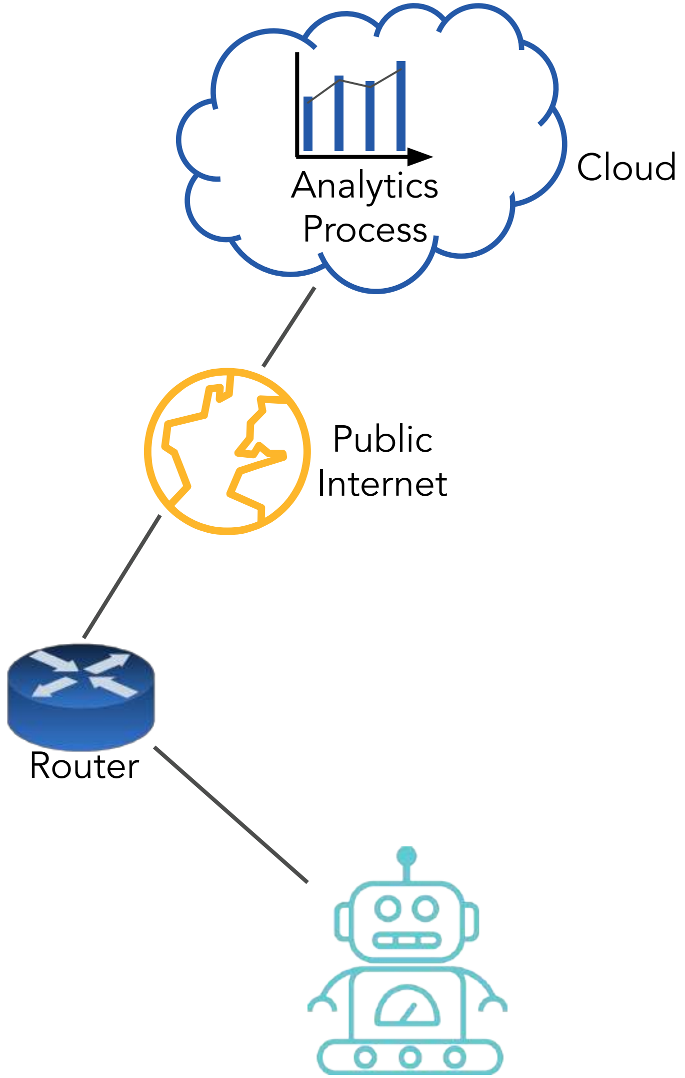
- Ensures secure, direct interactions between device and app
- Device and app discover each other via standard DDS discovery
- No cloud interaction or Internet traversal required



Use Case #2: Leveraging DDS for Cloud Analytics

Data collection, analytics process, etc. are all DDS participants

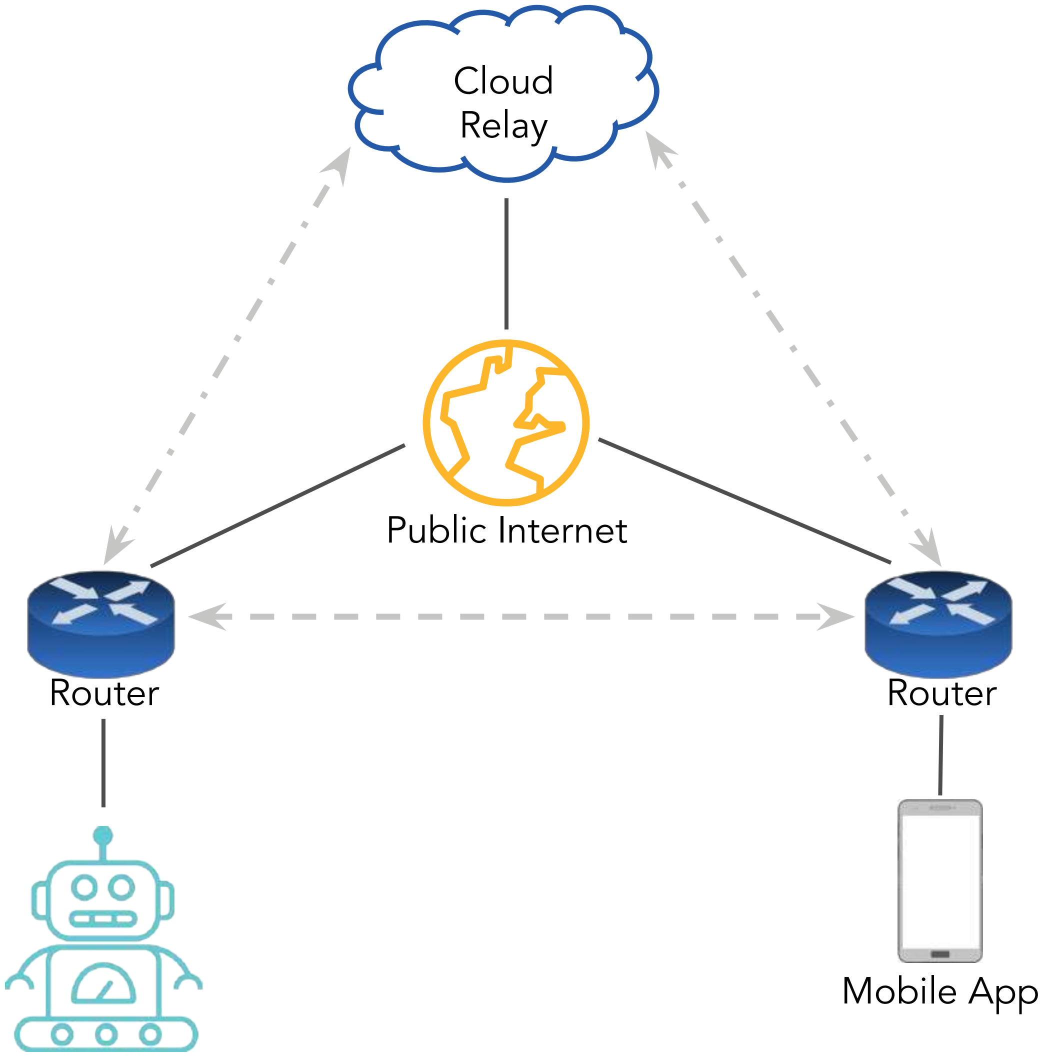
- End-to-end security
- Data collection governed by DDS QoS
- Interoperability achieved with common data model across tiers



Use Case #3: Leveraging DDS for Remote Connectivity

Robot and mobile app on different networks

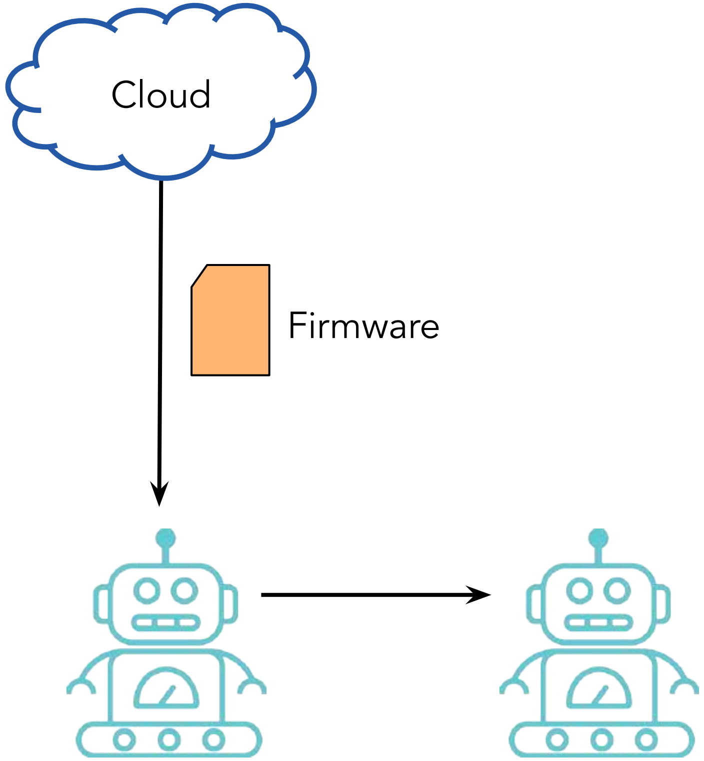
- Leverage capabilities of edge devices to offload cloud processing
- Discovery and connectivity require minimal cloud interaction for cost savings and scalability
- Ensures secure interactions between device and app, even across public Internet



Use Case #4: Firmware Distribution

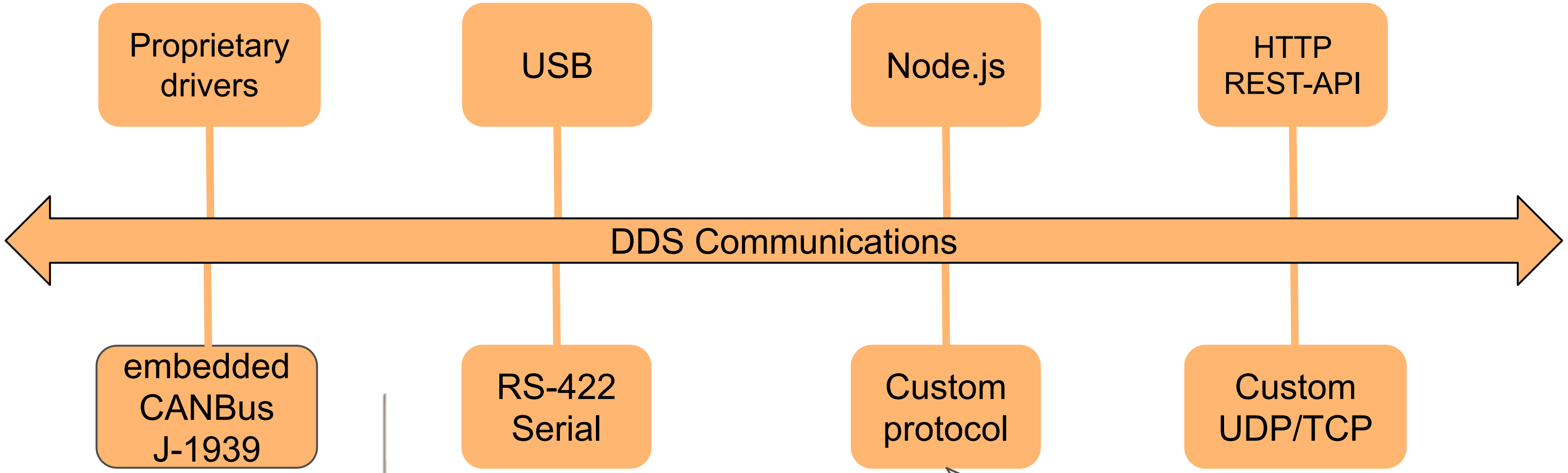
Distribute new firmware among robots

- Minimize costly downloads from cloud (1M devices * 64MB * 4/yr)
- Utilize extra storage on devices
- Peering controlled by PARTITION QoS
- Each device shares firmware with its peers



OpenDDS for Vehicle Communications

DDS as a multiprotocol bus



Proprietary drivers

USB

Node.js

HTTP REST-API

DDS Communications

embedded CANBus J-1939

RS-422 Serial

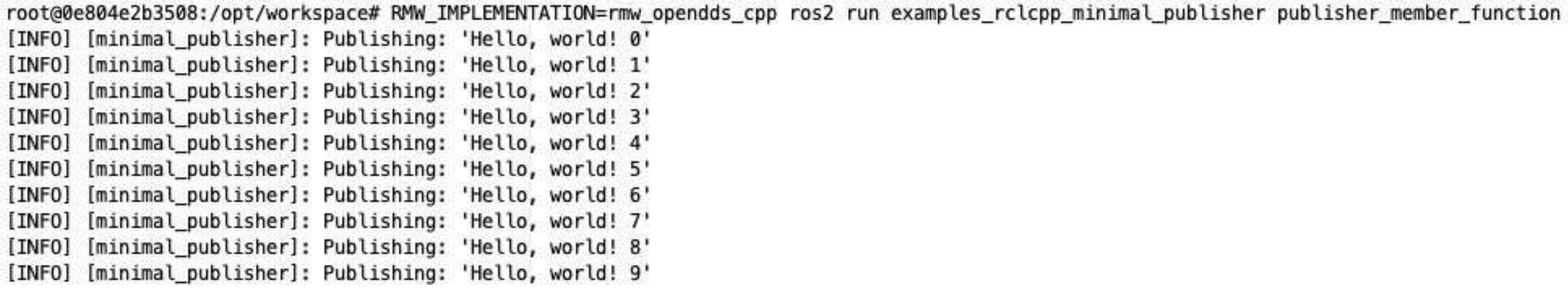
Custom protocol

Custom UDP/TCP

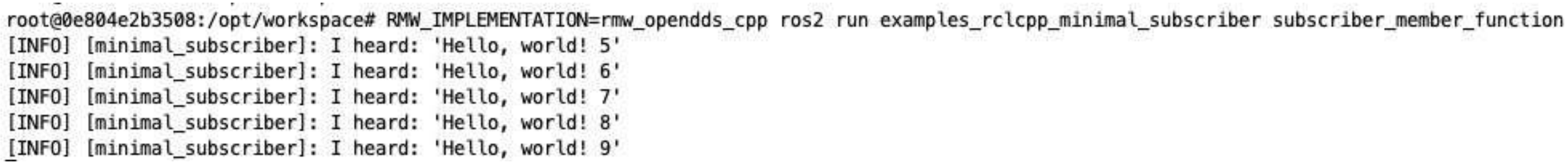
Could be an Industrial automation protocol like OPC-UA, Modbus, EtherCAT, Ethernet/IP, etc

OpenDDS RMW

Publish/Subscribe

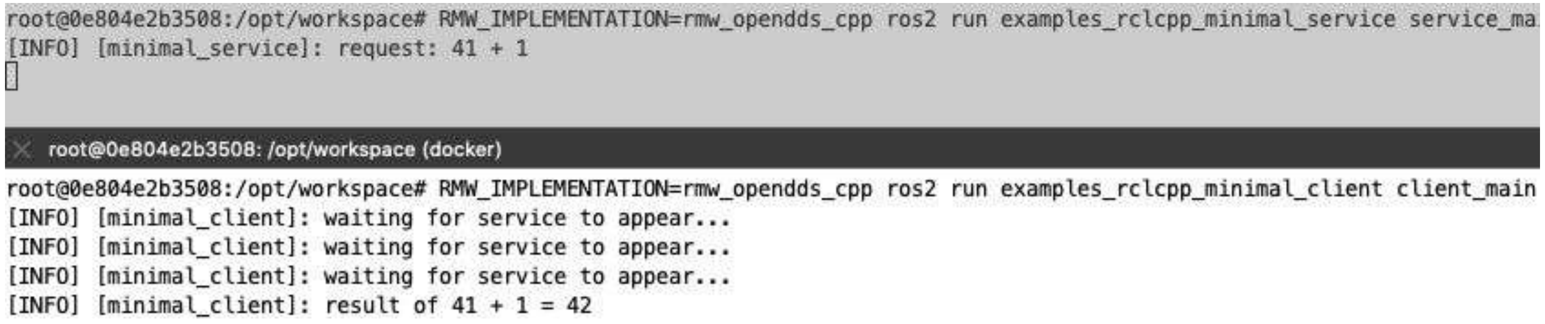


```
root@0e804e2b3508:/opt/workspace# RMW_IMPLEMENTATION=rmw_opendds_cpp ros2 run examples_rclcpp_minimal_publisher publisher_member_function
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 0'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 1'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 2'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 3'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 4'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 5'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 6'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 7'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 8'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 9'
```



```
root@0e804e2b3508:/opt/workspace# RMW_IMPLEMENTATION=rmw_opendds_cpp ros2 run examples_rclcpp_minimal_subscriber subscriber_member_function
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 5'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 6'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 7'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 8'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 9'
```

Client/Server

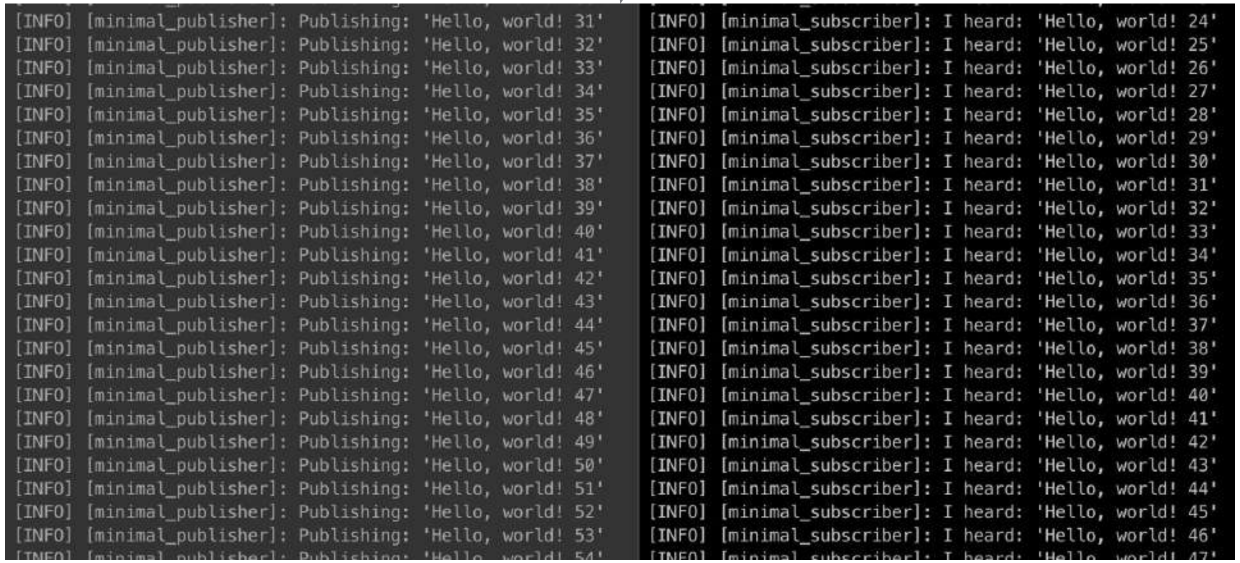


```
root@0e804e2b3508:/opt/workspace# RMW_IMPLEMENTATION=rmw_opendds_cpp ros2 run examples_rclcpp_minimal_service service_ma
[INFO] [minimal_service]: request: 41 + 1
```

```
root@0e804e2b3508:/opt/workspace# RMW_IMPLEMENTATION=rmw_opendds_cpp ros2 run examples_rclcpp_minimal_client client_main
[INFO] [minimal_client]: waiting for service to appear...
[INFO] [minimal_client]: waiting for service to appear...
[INFO] [minimal_client]: waiting for service to appear...
[INFO] [minimal_client]: result of 41 + 1 = 42
```

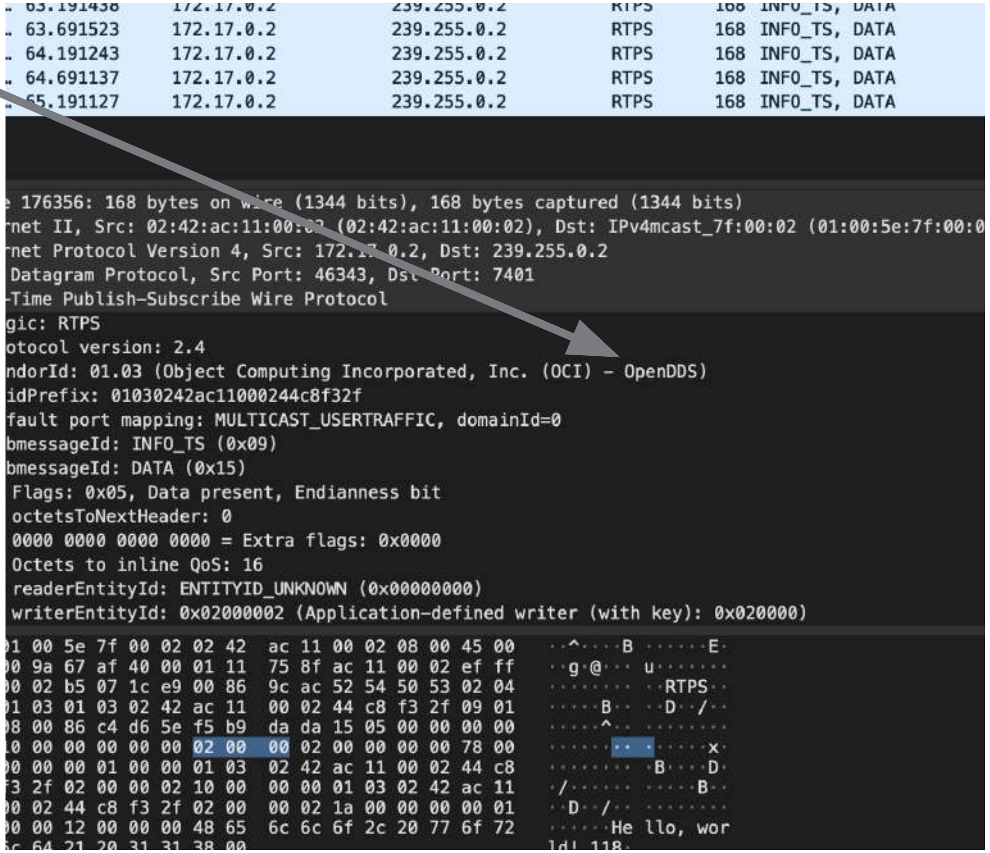
OpenDDS RMW

ROS2 Pub/Sub using OpenDDS



```
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 31'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 32'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 33'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 34'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 35'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 36'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 37'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 38'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 39'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 40'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 41'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 42'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 43'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 44'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 45'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 46'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 47'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 48'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 49'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 50'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 51'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 52'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 53'
[INFO] [minimal_publisher]: Publishing: 'Hello, world! 54'
```

```
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 24'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 25'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 26'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 27'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 28'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 29'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 30'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 31'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 32'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 33'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 34'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 35'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 36'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 37'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 38'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 39'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 40'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 41'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 42'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 43'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 44'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 45'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 46'
[INFO] [minimal_subscriber]: I heard: 'Hello, world! 47'
```



```
. 63.191438   172.17.0.2    239.255.0.2    RTPS    168 INFO_TS, DATA
. 63.691523   172.17.0.2    239.255.0.2    RTPS    168 INFO_TS, DATA
. 64.191243   172.17.0.2    239.255.0.2    RTPS    168 INFO_TS, DATA
. 64.691137   172.17.0.2    239.255.0.2    RTPS    168 INFO_TS, DATA
. 65.191127   172.17.0.2    239.255.0.2    RTPS    168 INFO_TS, DATA

e 176356: 168 bytes on wire (1344 bits), 168 bytes captured (1344 bits)
rnet II, Src: 02:42:ac:11:00:02 (02:42:ac:11:00:02), Dst: IPv4mcast_7f:00:02 (01:00:5e:7f:00:0
rnet Protocol Version 4, Src: 172.17.0.2, Dst: 239.255.0.2
 Datagram Protocol, Src Port: 46343, Dst Port: 7401
-Time Publish-Subscribe Wire Protocol
gic: RTPS
otocol version: 2.4
ndorId: 01.03 (Object Computing Incorporated, Inc. (OCI) - OpenDDS)
idPrefix: 01030242ac11000244c8f32f
fault port mapping: MULTICAST_USERTRAFFIC, domainId=0
bmessageId: INFO_TS (0x09)
bmessageId: DATA (0x15)
 Flags: 0x05, Data present, Endianness bit
 octetsToNextHeader: 0
 0000 0000 0000 0000 = Extra flags: 0x0000
 Octets to inline QoS: 16
 readerEntityId: ENTITYID_UNKNOWN (0x00000000)
 writerEntityId: 0x02000002 (Application-defined writer (with key): 0x020000)

01 00 5e 7f 00 02 02 42  ac 11 00 02 08 00 45 00    ··^····B ······E·
00 9a 67 af 40 00 01 11  75 8f ac 11 00 02 ef ff    ··g·@··· u·······
00 02 b5 07 1c e9 00 86  9c ac 52 54 50 53 02 04    ········ ··RTPS··
01 03 01 03 02 42 ac 11  00 02 44 c8 f3 2f 09 01    ·····B·· ··D··/··
08 00 86 c4 d6 5e f5 b9  da da 15 05 00 00 00 00    ·····^·· ········
10 00 00 00 00 00 02 00  00 02 00 00 00 00 78 00    ········ ······x·
00 00 00 01 00 00 01 03  02 42 ac 11 00 02 44 c8    ········ ·B····D·
f3 2f 02 00 00 02 10 00  00 00 01 03 02 42 ac 11    ·/······ ·····B··
00 02 44 c8 f3 2f 02 00  00 02 1a 00 00 00 00 01    ··D··/·· ········
00 00 12 00 00 00 48 65  6c 6c 6f 2c 20 77 6f 72    ······He llo, wor
6c 64 21 20 31 31 38 00                             ld! 118·
```

OpenDDS RMW

- Two main GitHub repositories
  - https://github.com/oci-labs/rmw_opendds
  - https://github.com/oci-labs/rosidl_typesupport_opendds
- Not deployed to rosdistro (yet)
- RMW development environment
  - Best place to start is https://github.com/oci-labs/rmw_build/blob/master/README.md

OCI is the home of OpenDDS®

COMMUNICATION

Telemetry tracking & control

Software Defined Radio

MINING

Sensor aggregation & intelligence

AEROSPACE & DEFENSE

Battlefield Management Systems

Sensor & Logistics Data Dissemination

CLIMATE CONTROL / HVAC

Device Management

Secure Cloud Communications

BAE SYSTEMS

DEPARTMENT OF DEFENSE UNITED STATES OF AMERICA

iridium

LEONARDO

L3HARRIS™

NAVAIR

- https://opendds.org/
- Liberally licenced for commercial use
- Created as an open-source project 15 years ago
- C++ implementation
- C++, Java, Javascript, Python (in devel), and C# language bindings
- Shared memory and RTPS UDP transports available, among others
- Runs on Mac, Windows, Linux, and VxWorks and more
- Supports the DDS security spec

Your Next Steps

Learn more about ROS2 and OpenDDS at OCI

- OpenDDS project website
  - https://opendds.org/
- Learn more about DDS
  - https://objectcomputing.com/services/training
- Get help with DDS
  - https://objectcomputing.com/products/open-source-support
- Try OpenDDS RMW and contribute issues and/or PRs
  - https://github.com/oci-labs/rmw_build/blob/master/README.md
- Try OpenDDS in a project
  - https://opendds.org/quickstart/

Your Next Steps

Contact Us About How We Can Help You…

- Solution Engineering including ROS2 and DDS
- Cloud-hosted simulation and digital twin support
- Integration with Enterprise IT with Microservices
- Perception-based Machine Learning
- Solution Analysis and Feasibility Studies
- Prototyping
- Custom training targeted to DDS and ROS2
- Improve Software Engineering Practices
- End-to-end Security Design
- Your idea?

LEARN MORE ABOUT OCI EVENTS AND TRAINING

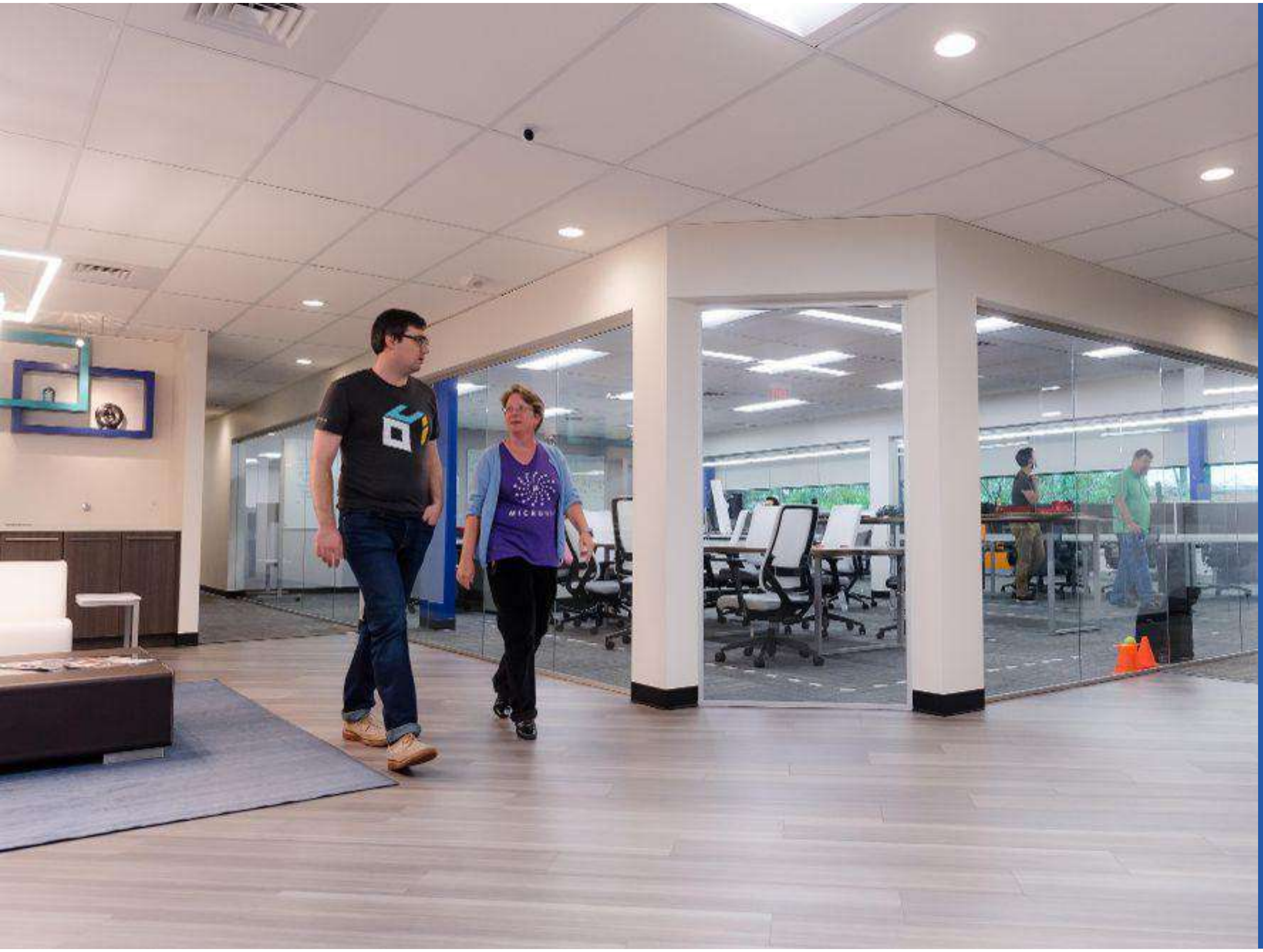
Events:

- objectcomputing.com/events
- Open enrollment online training:
  Introduction to OpenDDS Programming (C++ / Java)
  July 22 – 23, 2020

For More Training:

- objectcomputing.com/training

Or email info@ocitraining.com to schedule a custom training program for your team online, on site, or in our state-of-the-art, Midwest training lab.



OBJECT COMPUTING

CONNECT WITH US

☎ 1+ (314) 579-0066

✉ info@objectcomputing.com

objectcomputing.com