The Future of the Robot Operating System

ROS 2.0

Contents

Contents

ROSCon2015

Hamburg, Germany October 3-4, 2015

ROSCon2015

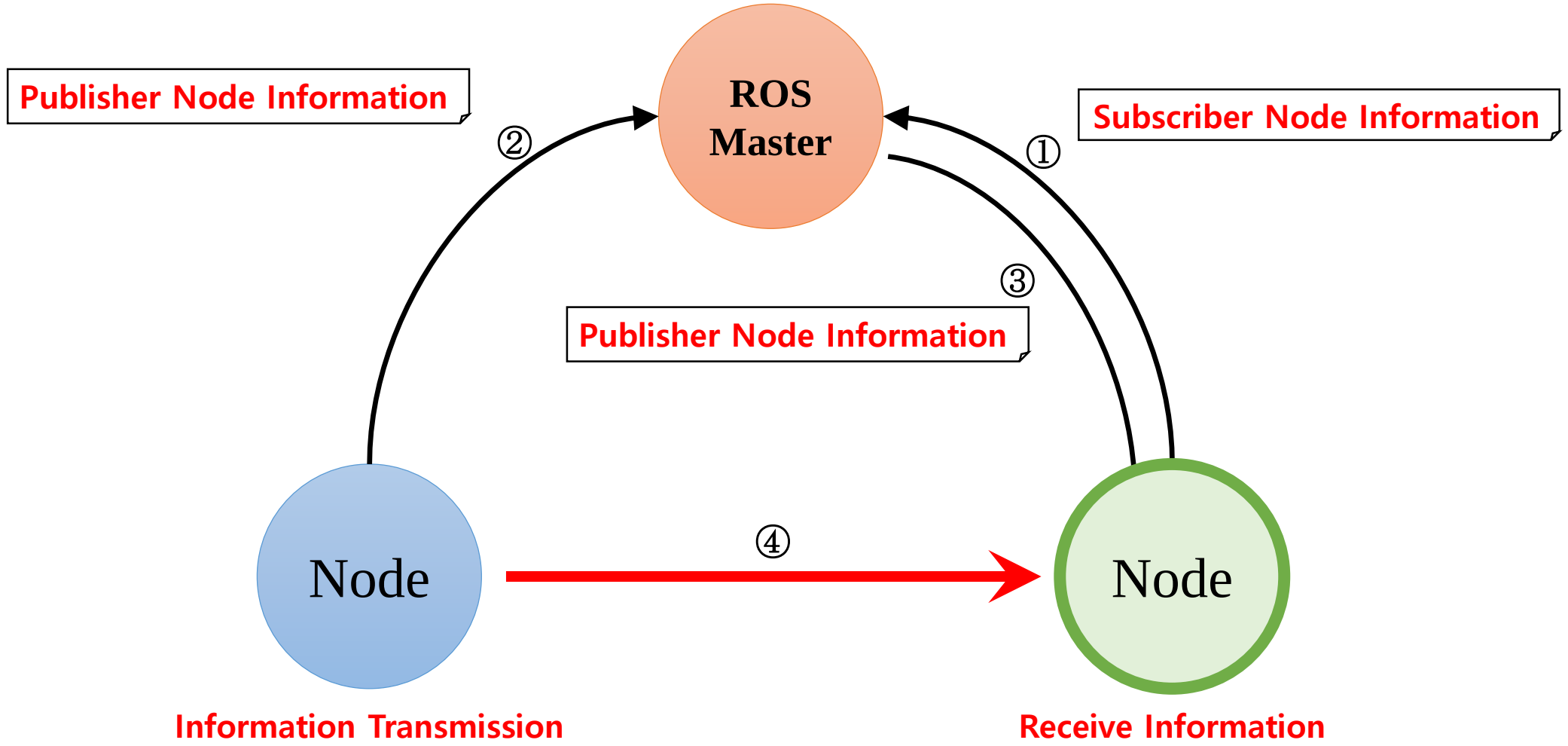
Hamburg, Germany October 3-4, 2015

Why ROS 2?

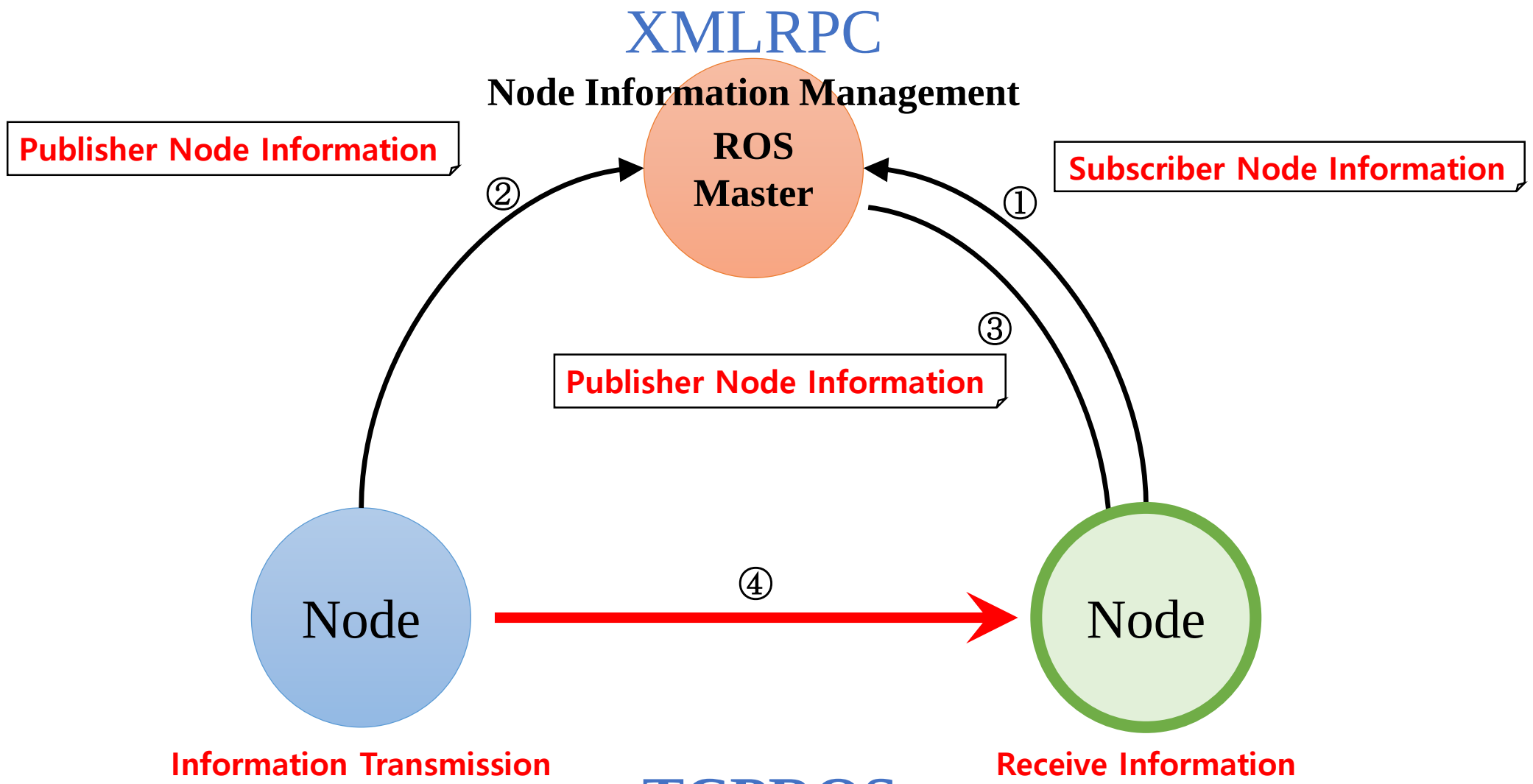
- Multiple robots
- Small embedded platforms
- Real-time systems
- Non-ideal networks
- Production environments
- Prescribed patterns for building and structuring systems
- New technologies: Zeroconf, Protocol Buffers, ZeroMQ, Redis, WebSockets, DDS (Data Distribution Service)
- API changes
- Risk associated with changing the current ROS system

Contents

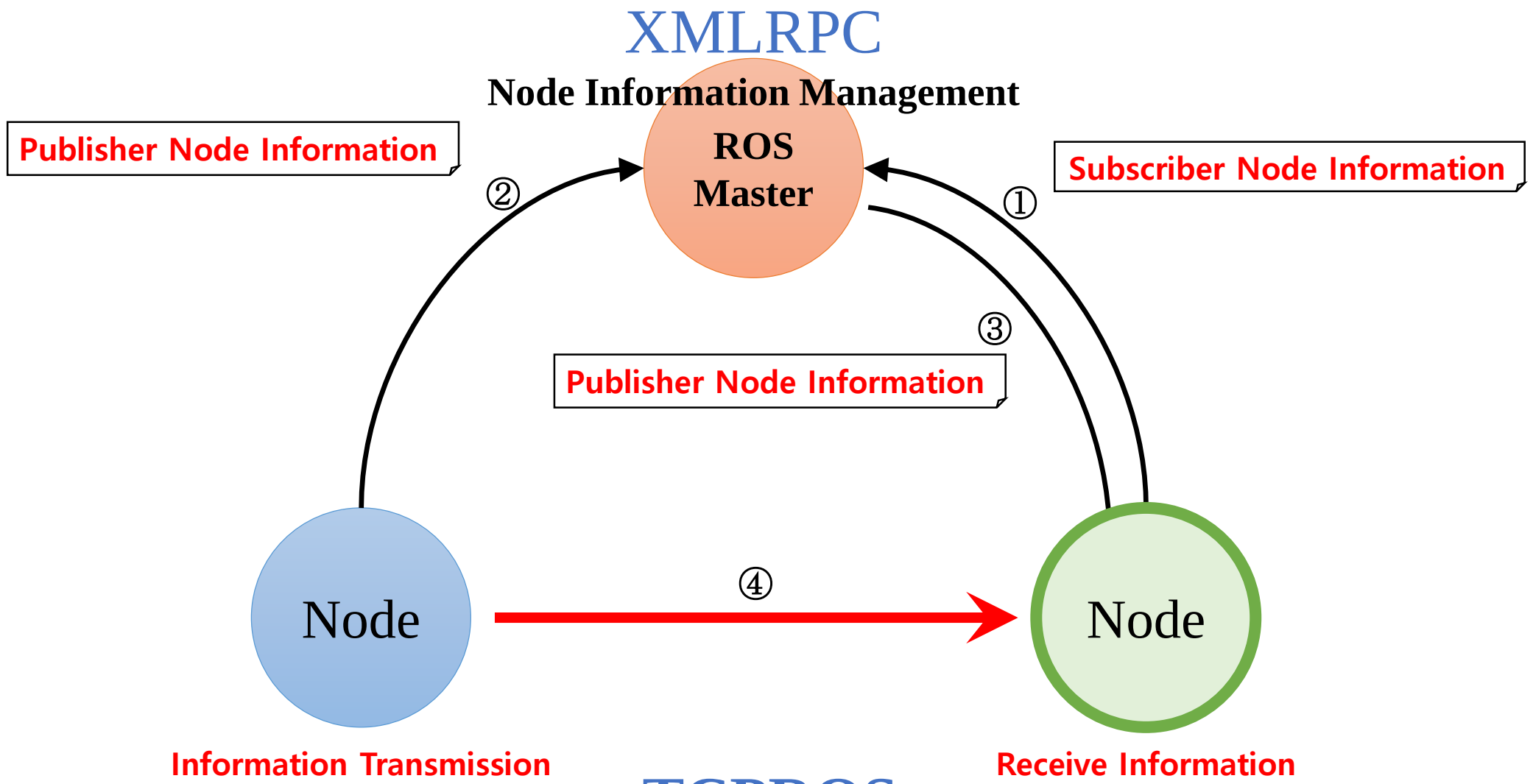
ROS 1.x (XMLRPC + TCPROS)



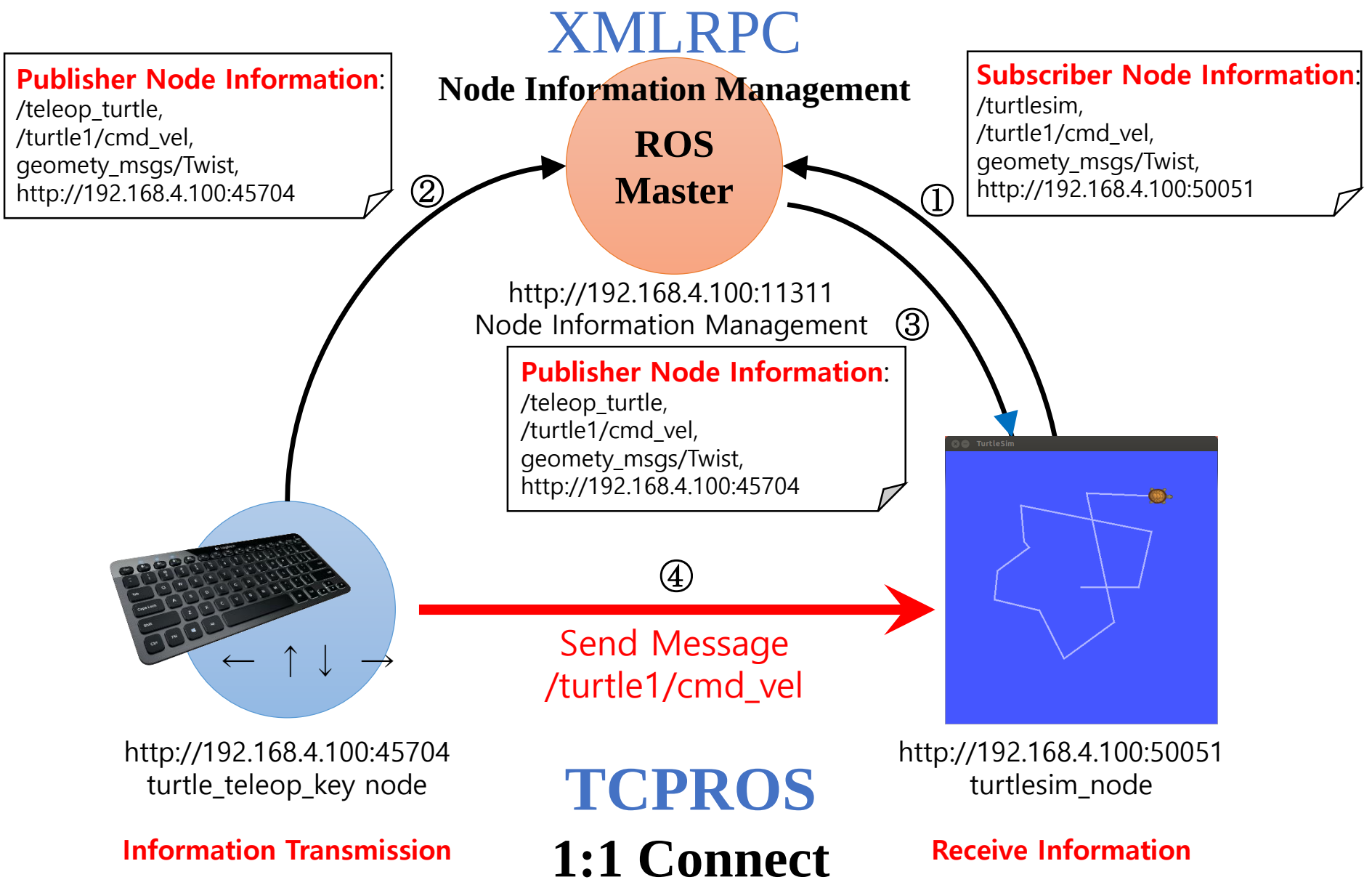
ROS 1.x (XMLRPC + TCPROS)



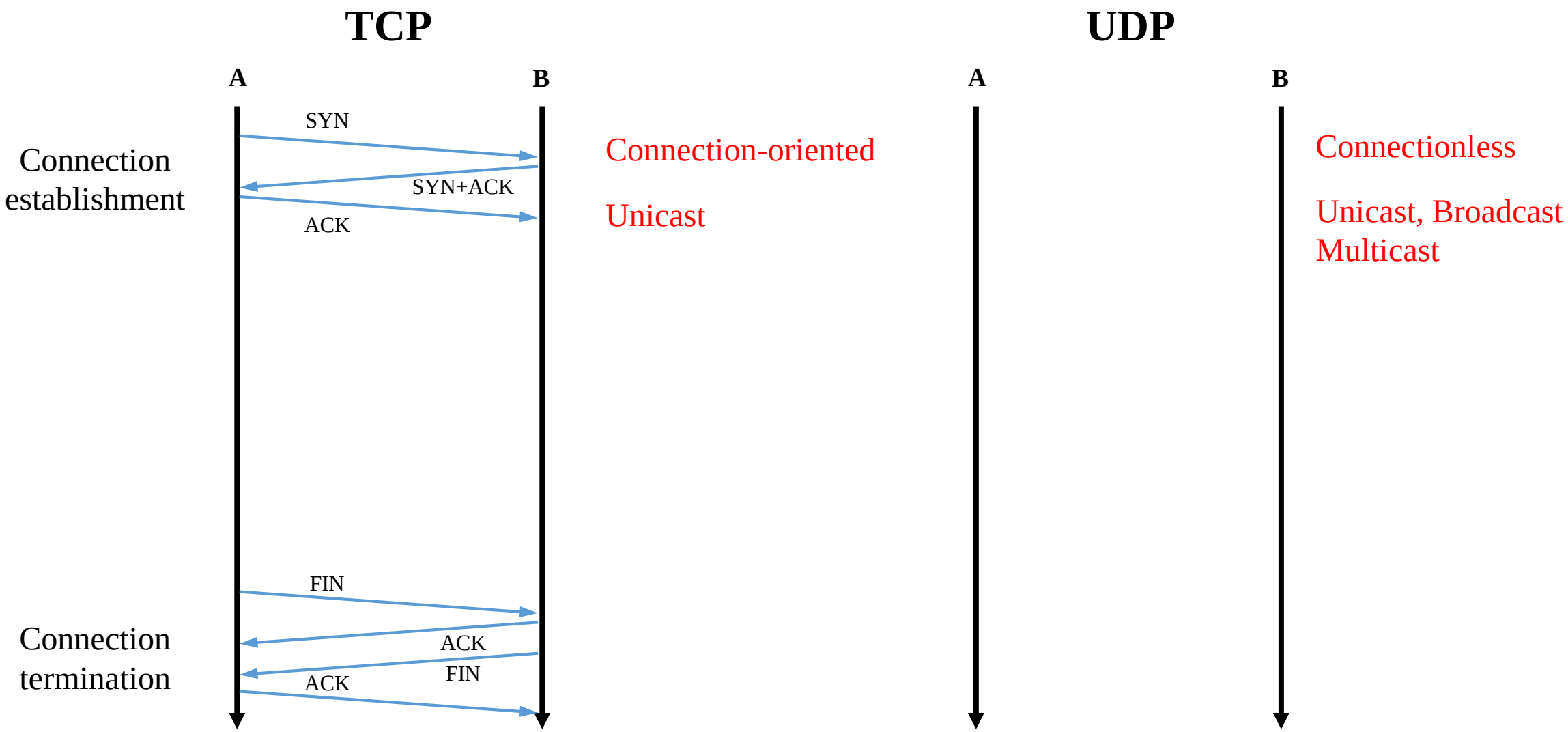
ROS 1.x (XMLRPC + TCPROS)



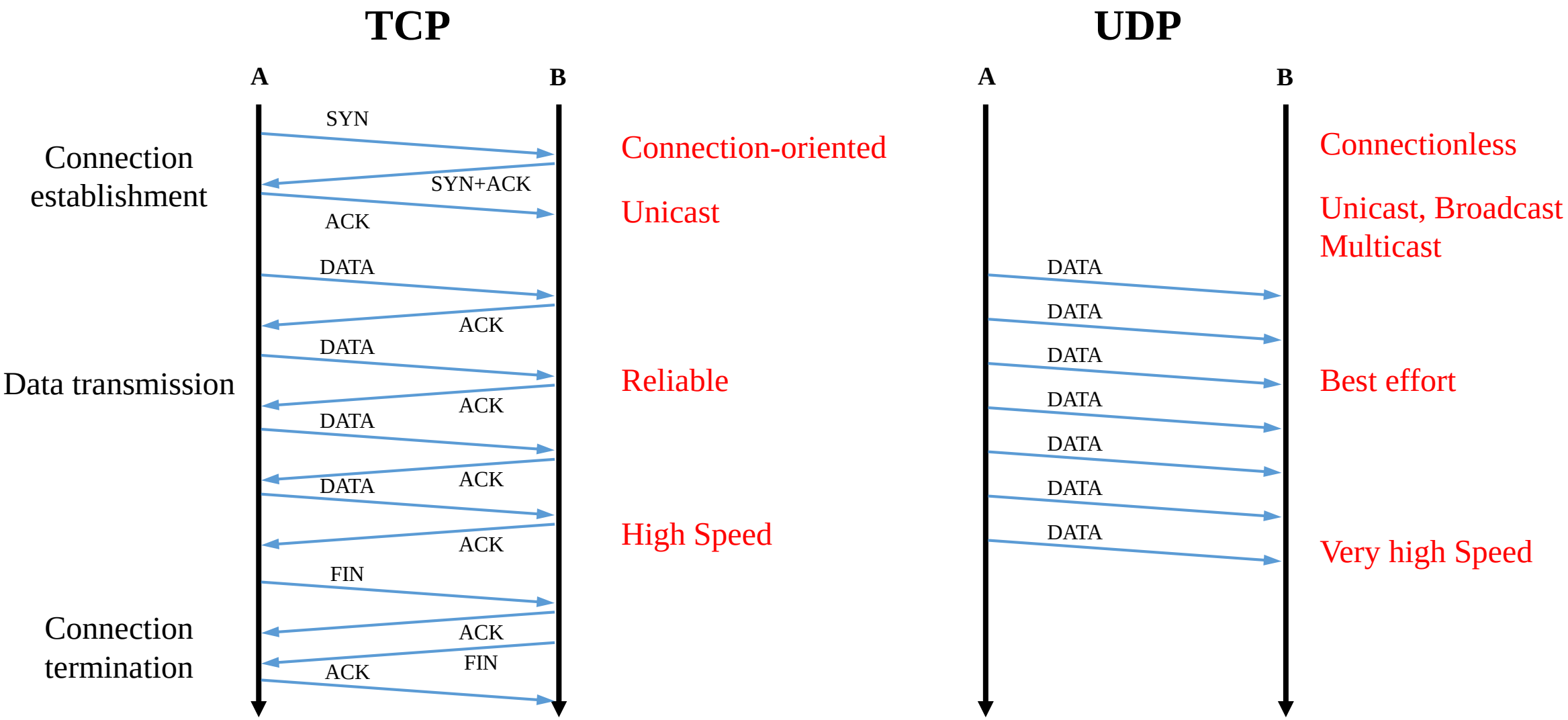
ROS 1.x (XMLRPC + TCPROS)



Transport: TCP vs UDP

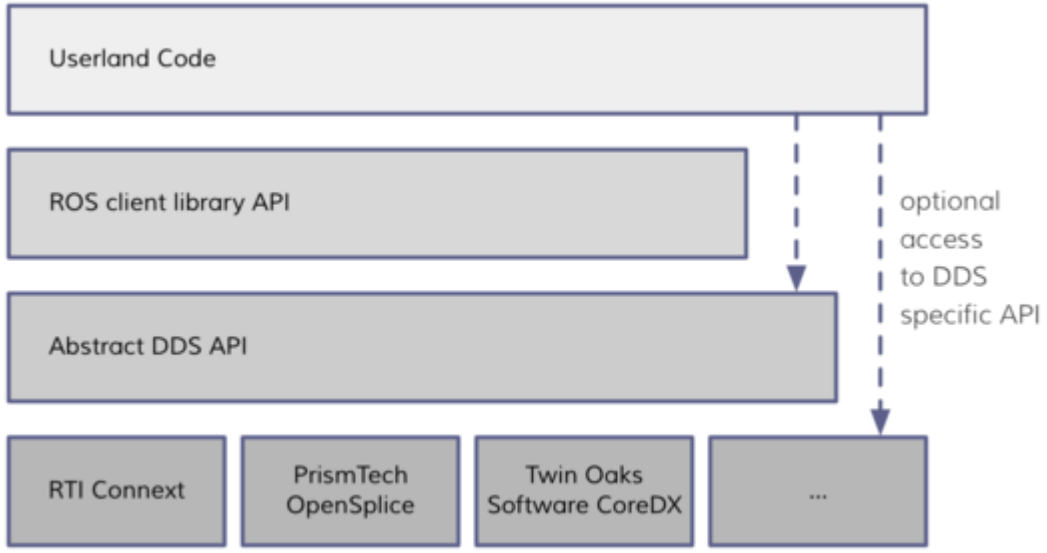


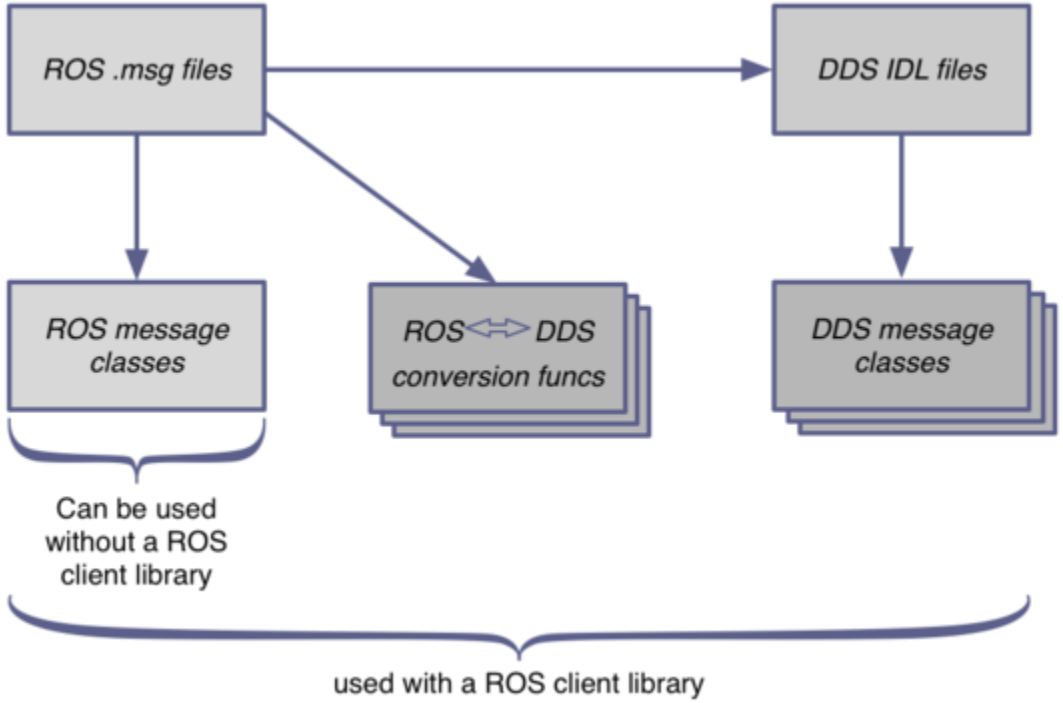
Transport: TCP vs UDP



DDS (Data Distribution Service)

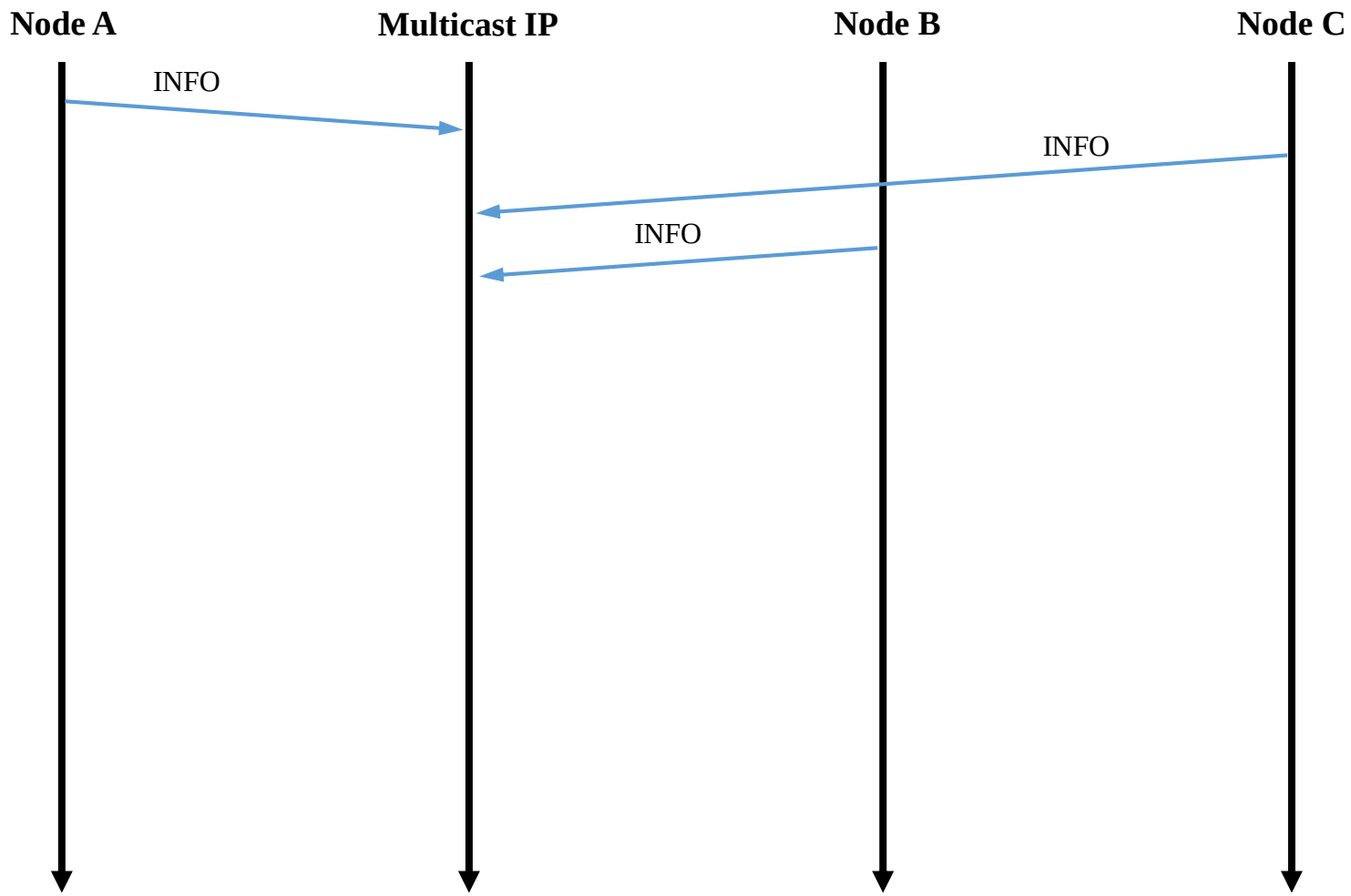
- DDS: Publish/subscribe middle ware for data-centric distributed systems, Managed by Object Management Group(OMG)
- Communication Protocol: RTPS (Real Time Publish Subscribe) It is based on UDP, but has the advantage of TCP





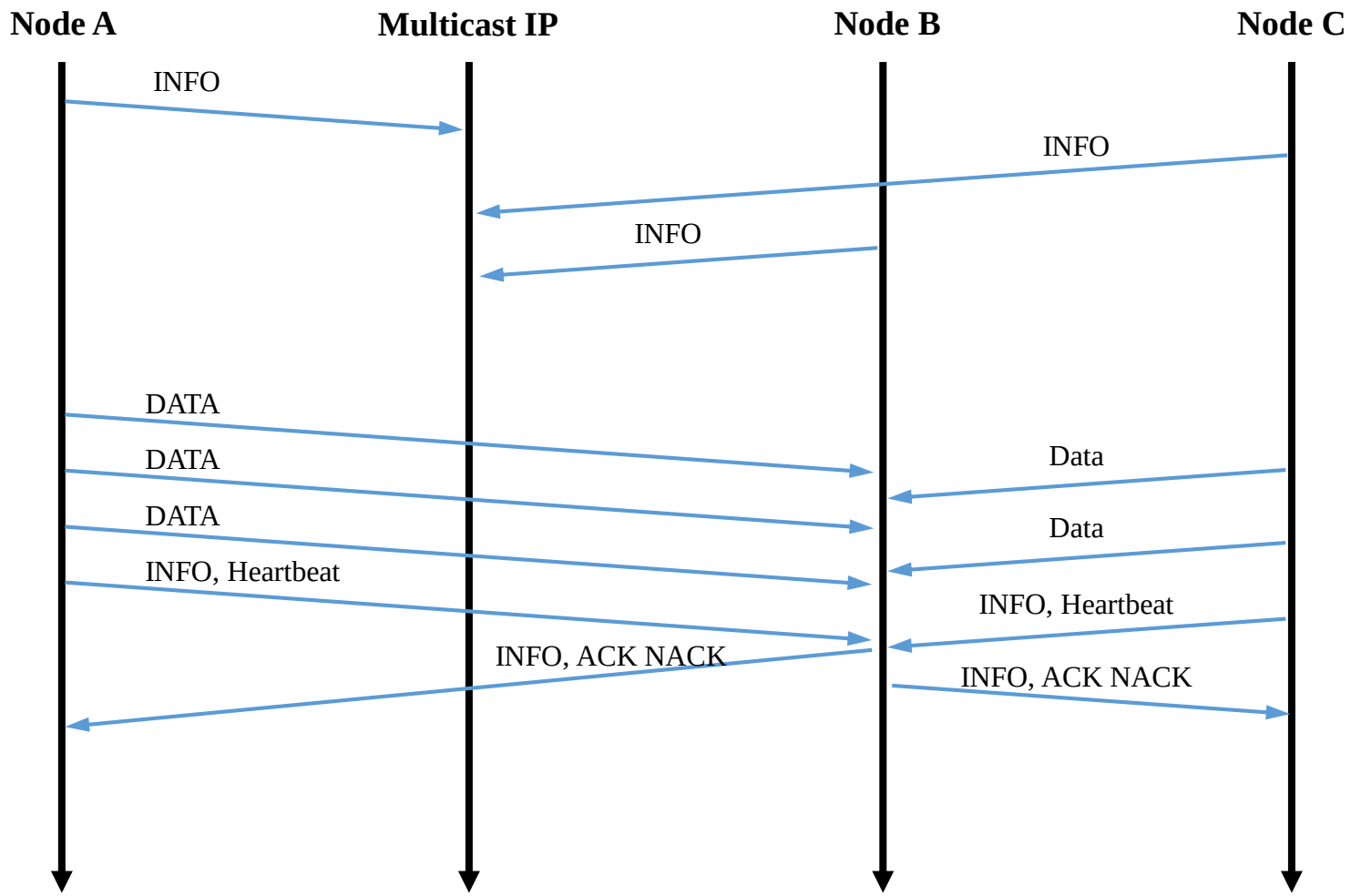
DDS (Data Distribution Service)

- **Auto-Discovery**: No need to know the IP Address and Port number



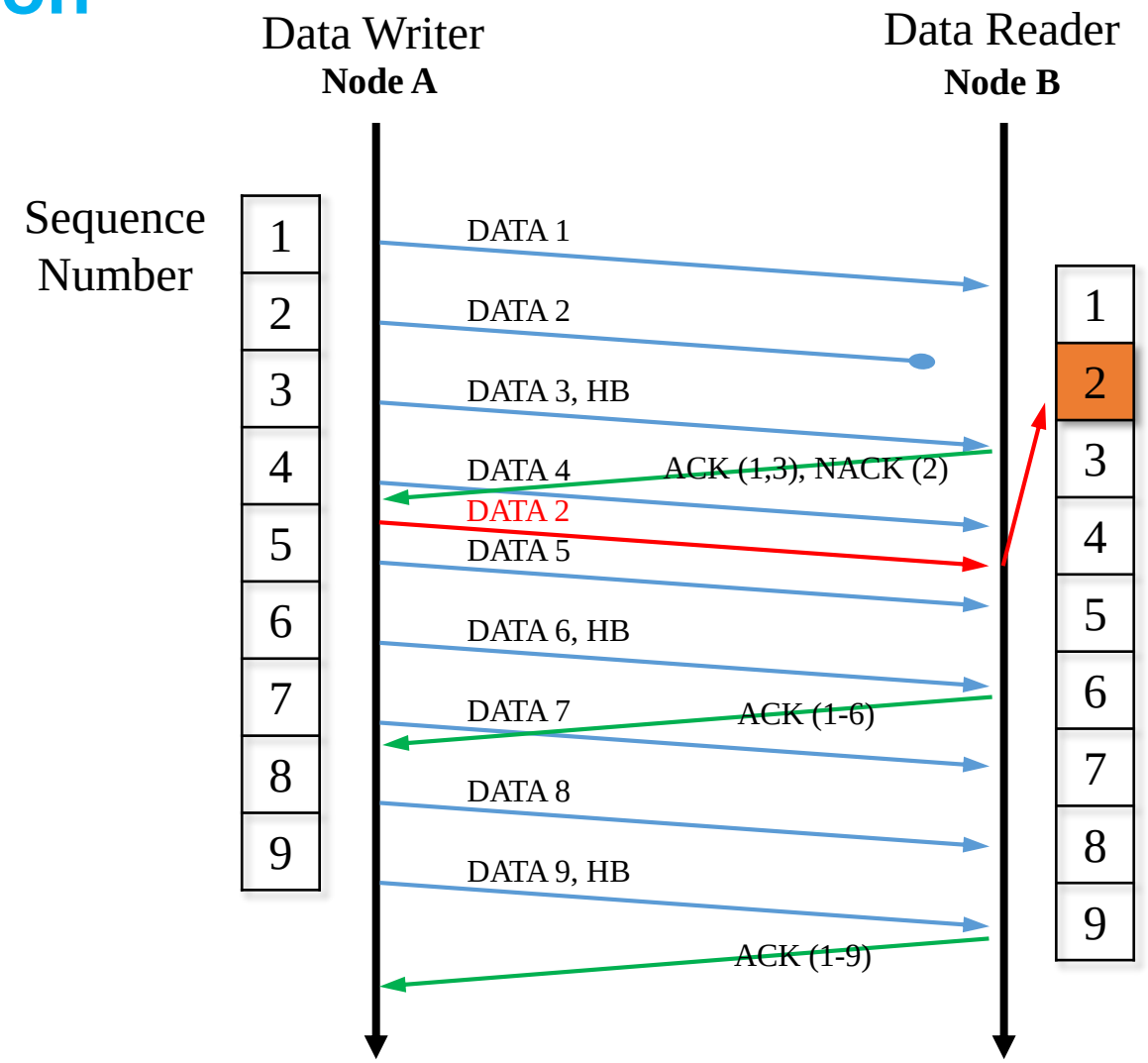
DDS (Data Distribution Service)

- **Auto-Discovery**: No need to know the IP Address and Port number



DDS (Data Distribution Service)

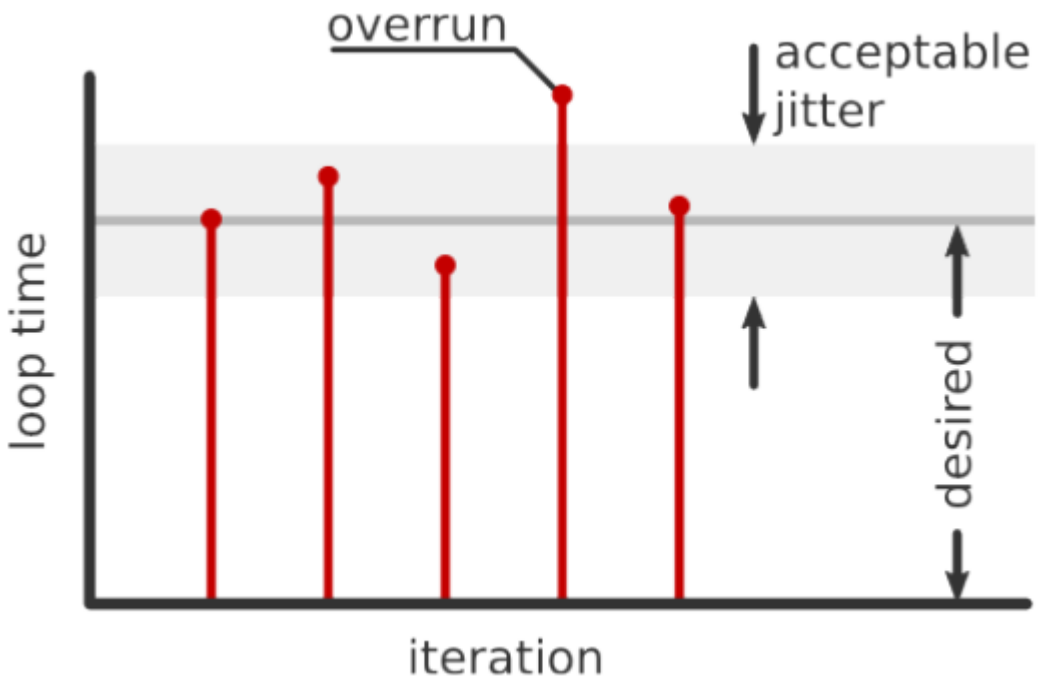
- **Retransmission**



Contents

Real-time Computing

- Real-time: It's about **determinism**, not performance!



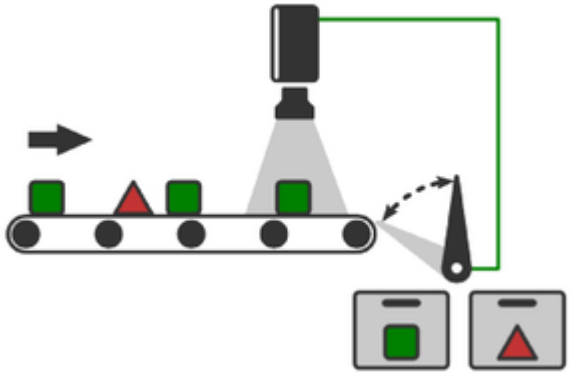
Jackie Kay (OSRF) et. al. "Real-time Performance in ROS 2", ROSCon2015

Real-time Computing

- Hard real-time systems
- Soft real-time systems
- Firm real-time systems

reactor, aircraft, spacecraft control

audio / video streaming



financial forecasting,
robot assembly lines

Jackie Kay (OSRF) et. al. "Real-time Performance in ROS 2", ROSCon2015

Real-time Computing

- Use an OS able to deliver the required determinism

OS	real-time	max latency (µs)
Linux	no	10^4
RT PREEMPT	soft	10^1-10^2
Xenomai	hard	10^1

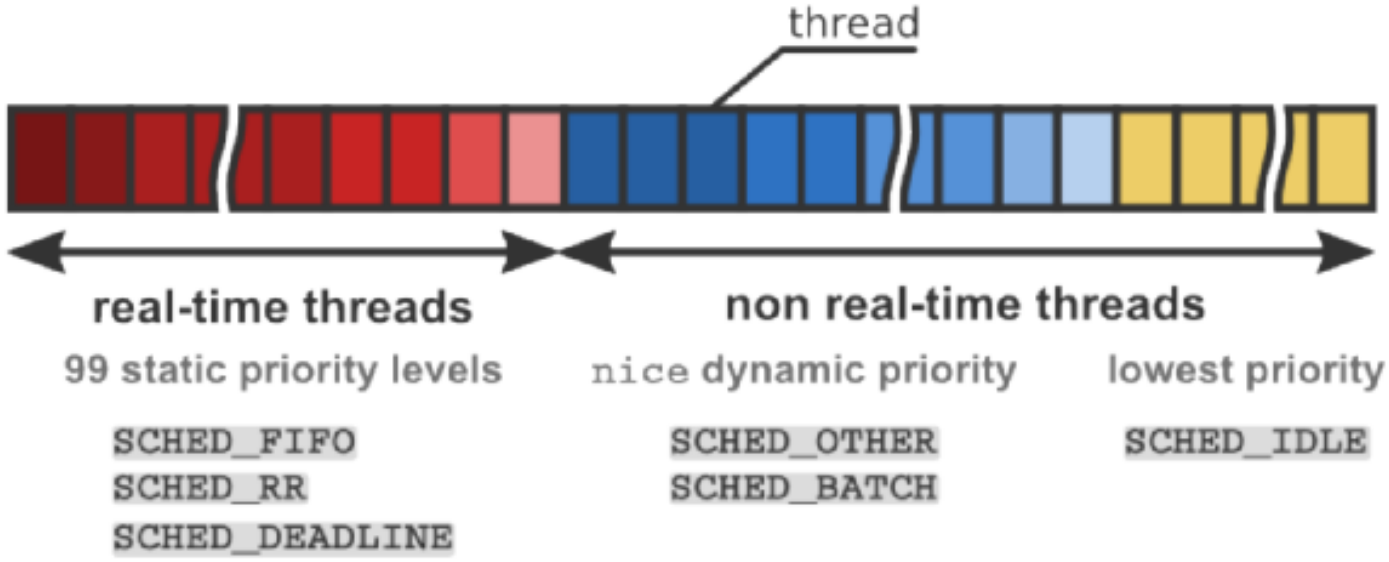
Jackie Kay (OSRF) et. al. "Real-time Performance in ROS 2", ROSCon2015

Real-time Computing

- Use an OS able to deliver the required determinism

OS	real-time	max latency (μs)
Linux	no	10^4
RT PREEMPT	soft	10^1-10^2
Xenomai	hard	10^1

- Prioritize real-time threads



Jackie Kay (OSRF) et. al. "Real-time Performance in ROS 2", ROSCon2015

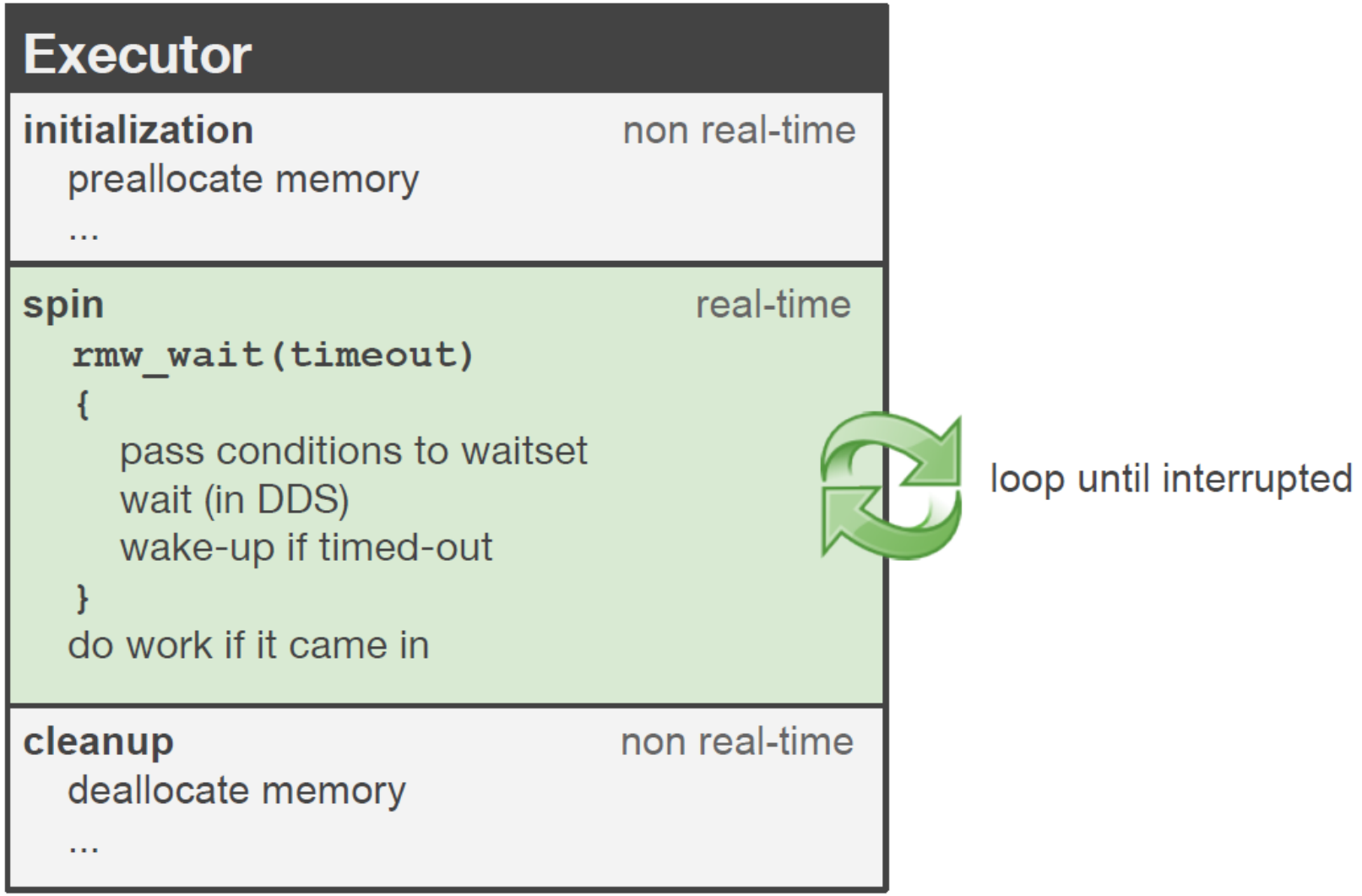
Real-time Computing

- Avoid sources of non-determinism in real-time code
  - Memory allocation and management (malloc, new)
  - Blocking synchronization primitives (mutex)
  - Printing, logging (printf, cout)
  - Network access, especially TCP/IP
  - Non real-time device drivers
  - Accessing the hard disk
  - Page faults

Jackie Kay (OSRF) et. al. "Real-time Performance in ROS 2", ROSCon2015

Real-time Computing

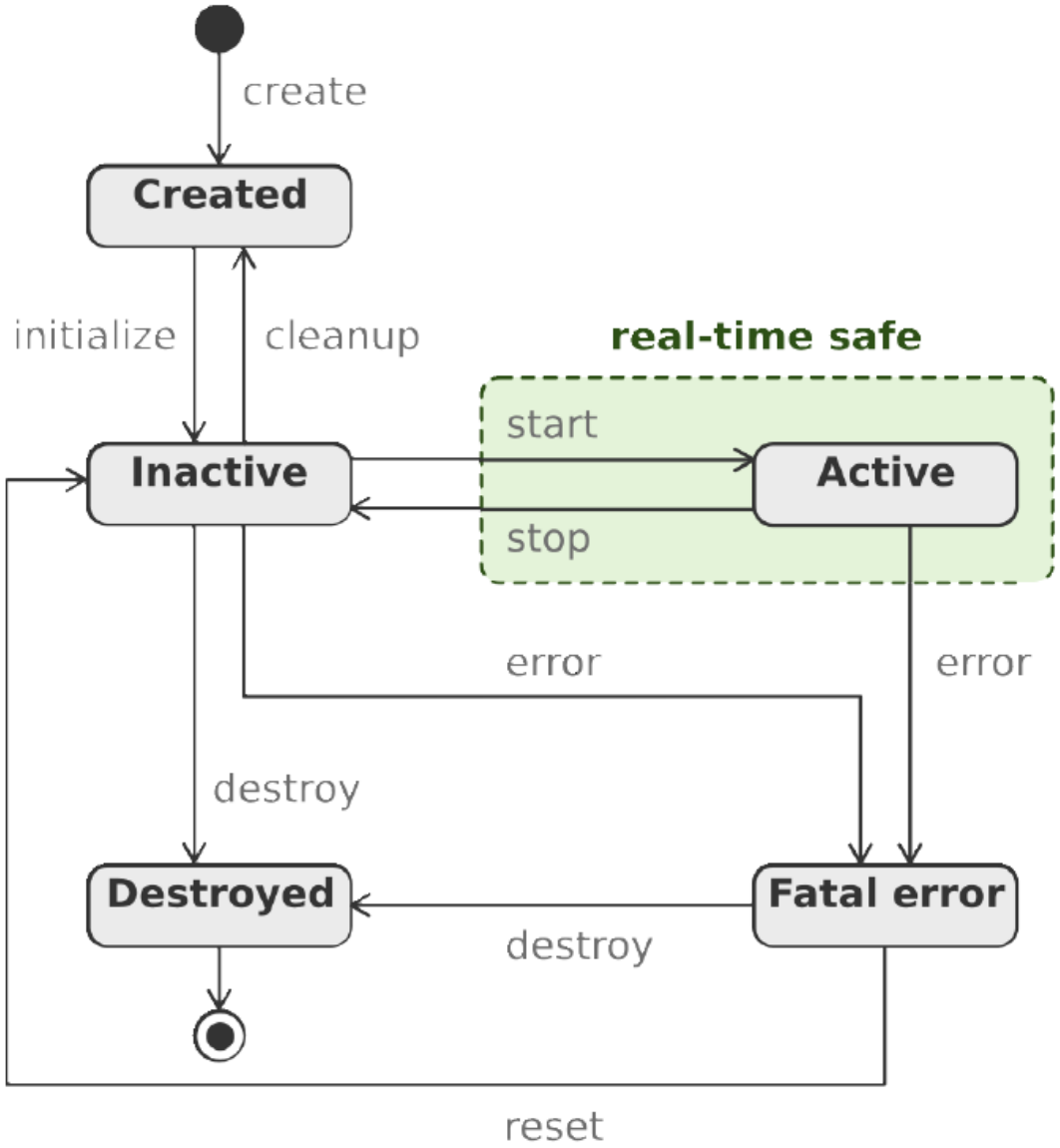
- Real-time code



Jackie Kay (OSRF) et. al. "Real-time Performance in ROS 2", ROSCon2015

Real-time Computing

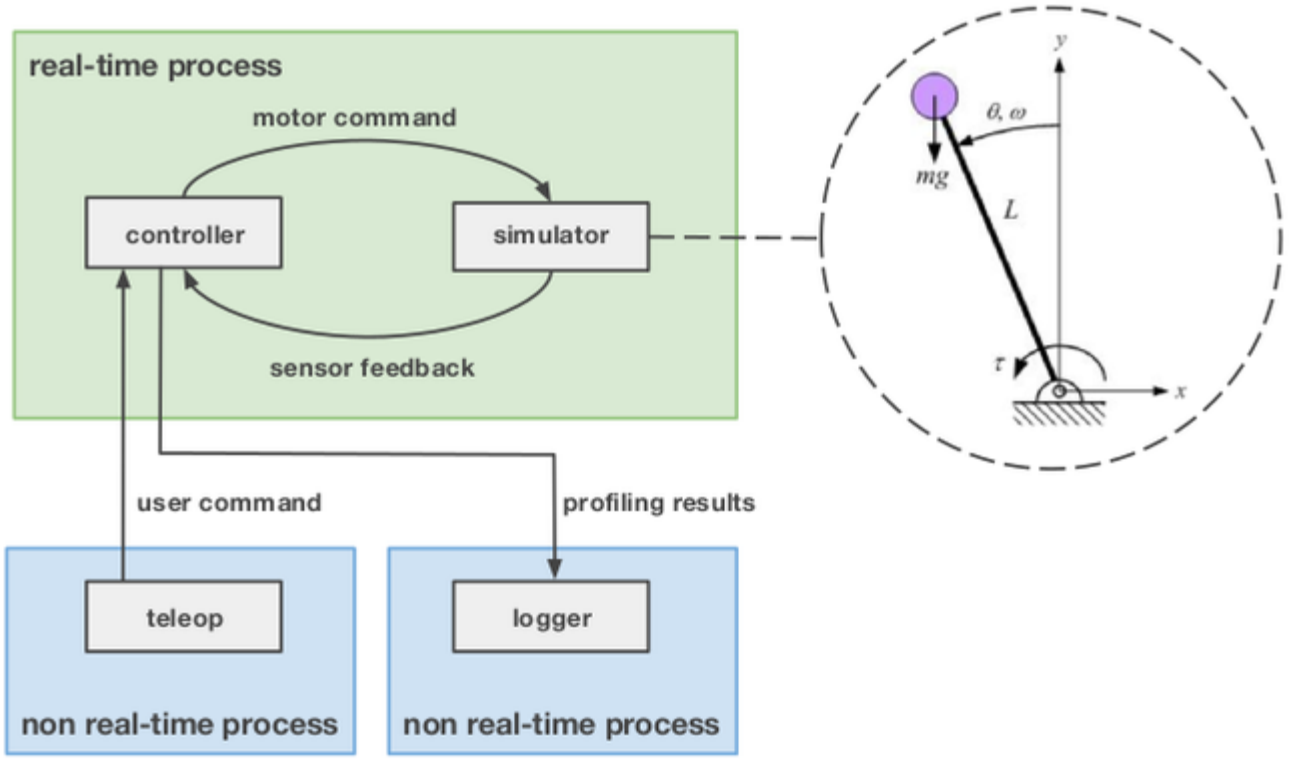
- Node lifecycle



Jackie Kay (OSRF) et. al. "Real-time Performance in ROS 2", ROSCon2015

Real-time Computing

- Real-time Benchmarking



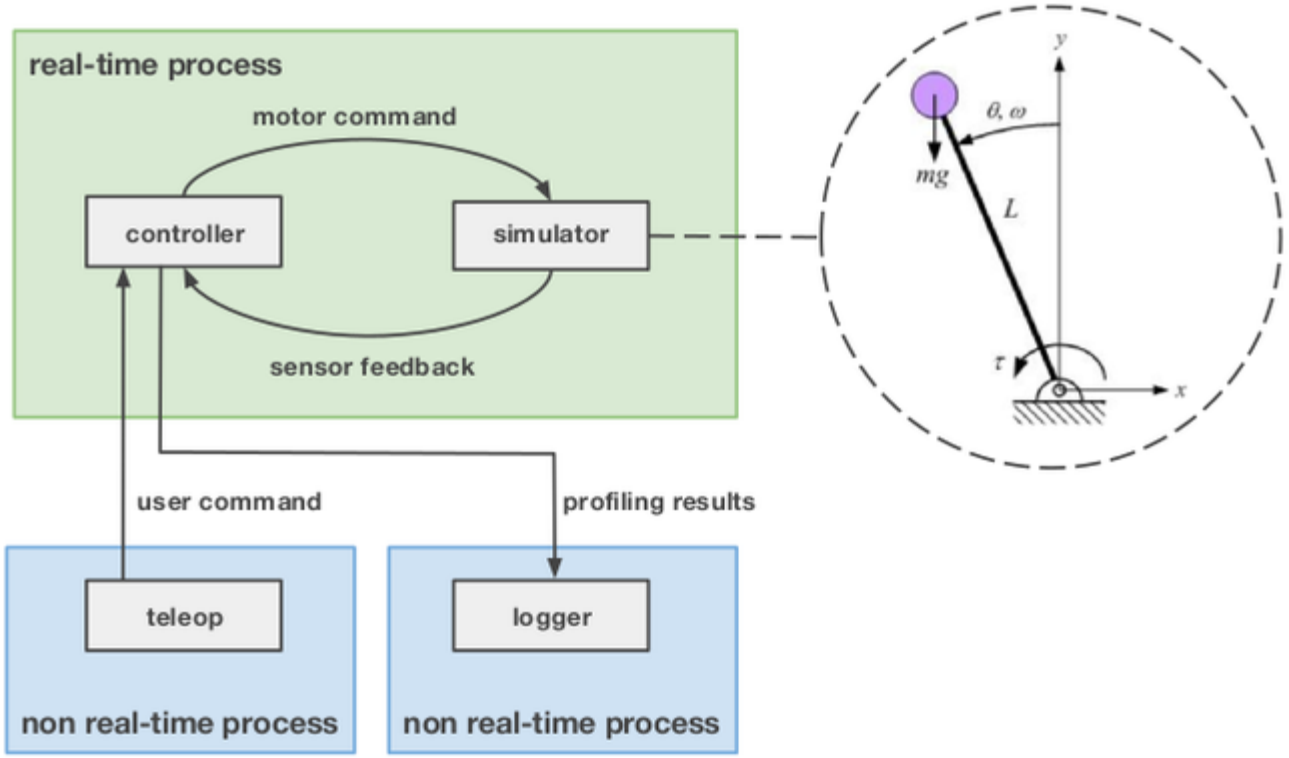
Goal

- **1 KHz** update loop (1 ms period)
- Less than **3%** jitter (30 μs)

Jackie Kay (OSRF) et. al. "Real-time Performance in ROS 2", ROSCon2015

Real-time Computing

- Real-time Benchmarking



Goal

- **1 KHz** update loop (1 ms period)
- Less than **3%** jitter (30 μs)

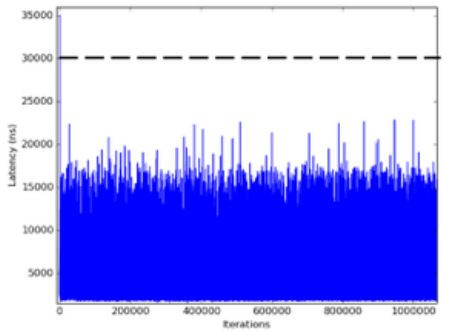
ROS 2 Real-time Benchmarking: Results

No stress

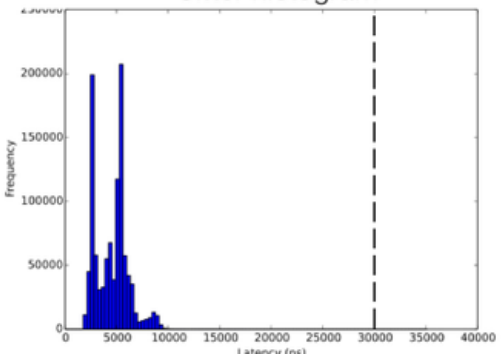
1,070,650 cycles observed

	Latency (ns)	% of update rate
Min	1620	0.16%
Max	35094	3.51%
Mean	4567	0.46%

Timeseries



Jitter histogram

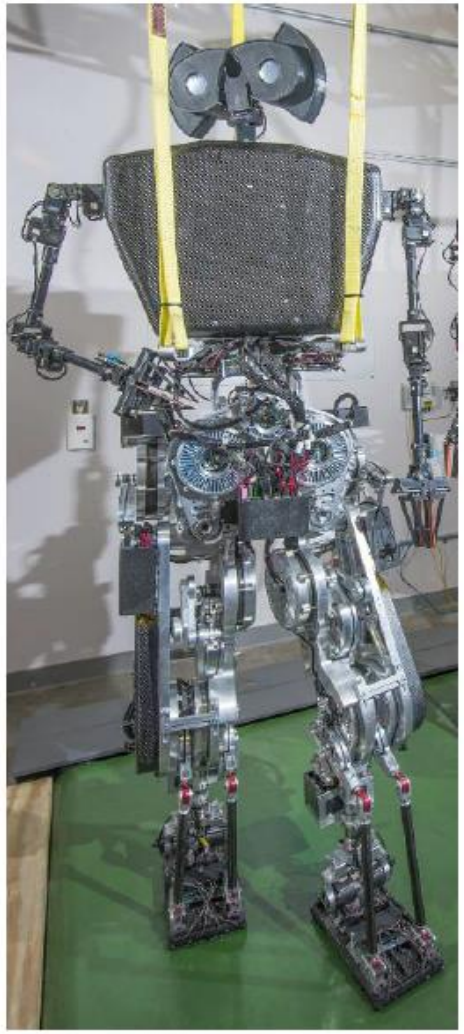


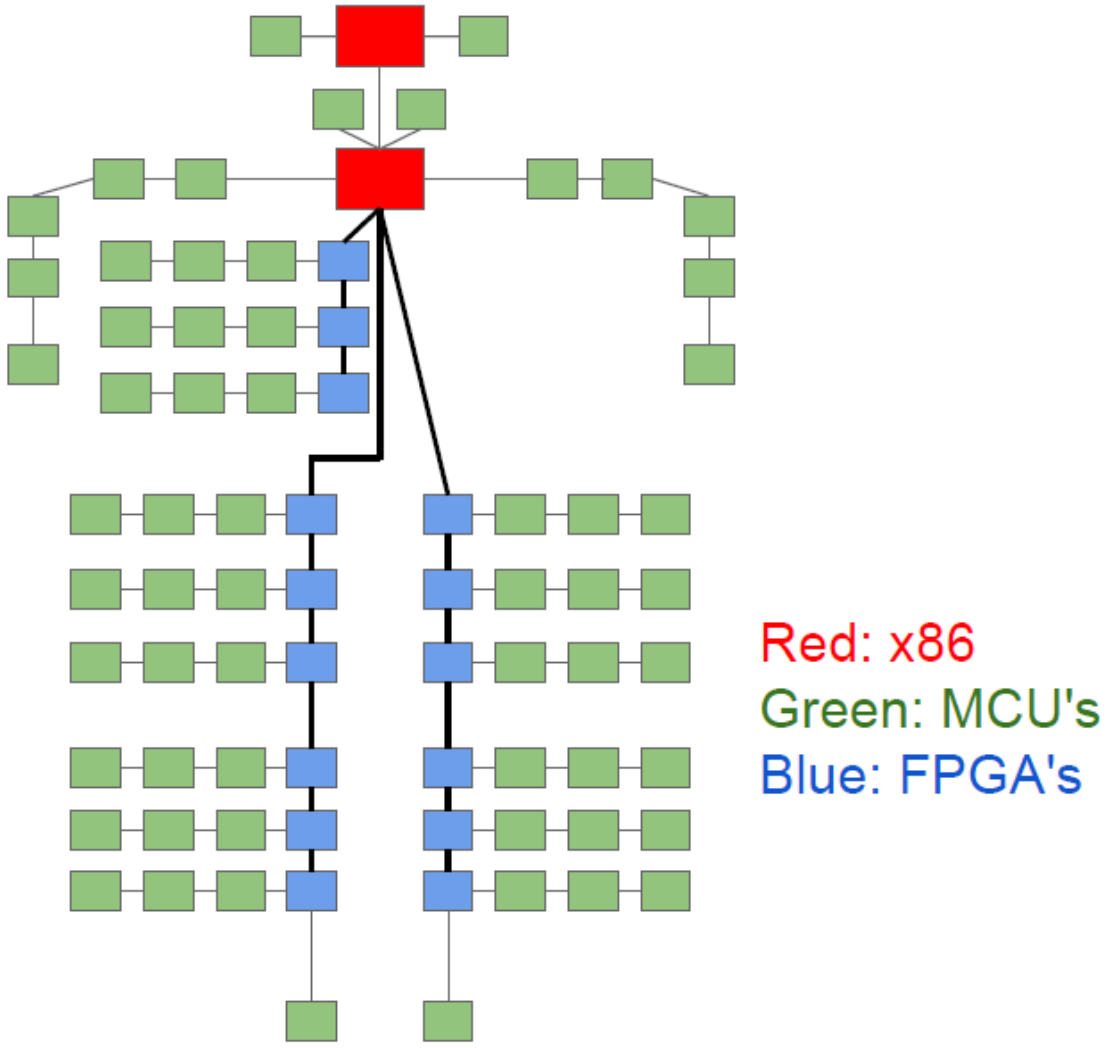
Jackie Kay (OSRF) et. al. "Real-time Performance in ROS 2", ROSCon2015

Contents

Embedded Systems

- Small embedded systems are everywhere!





Morgan Quigley (OSRF) "ROS 2 on ″small″ embedded systems", ROSCon2015

Scope



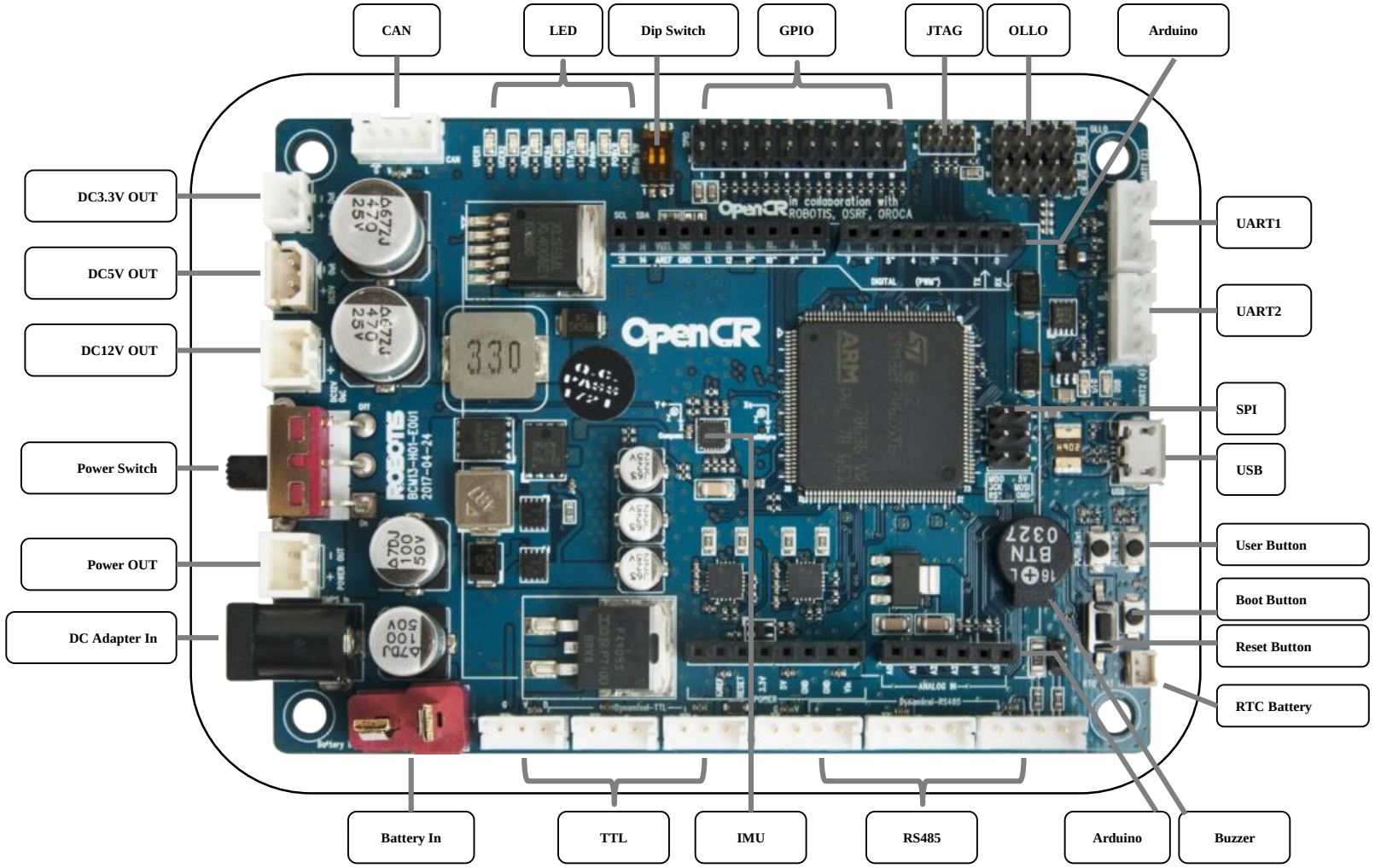

	8/16-bit MCU	32-bit MCU		ARM A-class (smartphone without screen)	x86 (laptop without screen)
		"small" 32-bit MCU	"big" 32-bit MCU		
Example Chip	Atmel AVR	ARM Cortex-M0	ARM Cortex-M7	Samsung Exynos	Intel Core i5
Example System	Arduino Leonardo	Arduino M0 Pro	SAM V71	ODROID	Intel NUC
MIPS	10's	100's	100's	1000's	10000's
RAM	1-32 KB	32 KB	384 KB	a few GB (off-chip)	2-16 GB (SODIMM)
Max power	10's of mW	100's of mW	100's of mW	1000's of mW	10000's of mW
Peripherals	UART, USB FS, ...	USB FS	Ethernet, USB HS	Gigabit Ethernet	USB SS, PCIe

Future work **Target MCU** **"Normal" ROS2**

Morgan Quigley (OSRF) "ROS 2 on ″small″ embedded systems", ROSCon2015

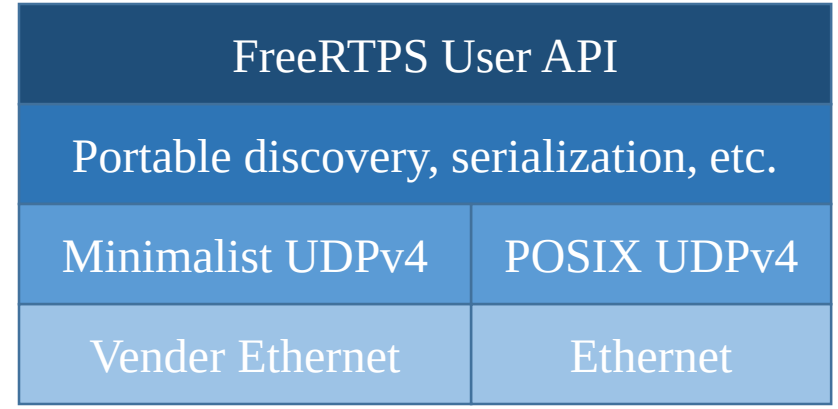
OpenCR

- Open-source Control module for ROS (OpenCR, Link)
- Present 'ROS Embedded Board' at 'ROSCon 2016' (Link)

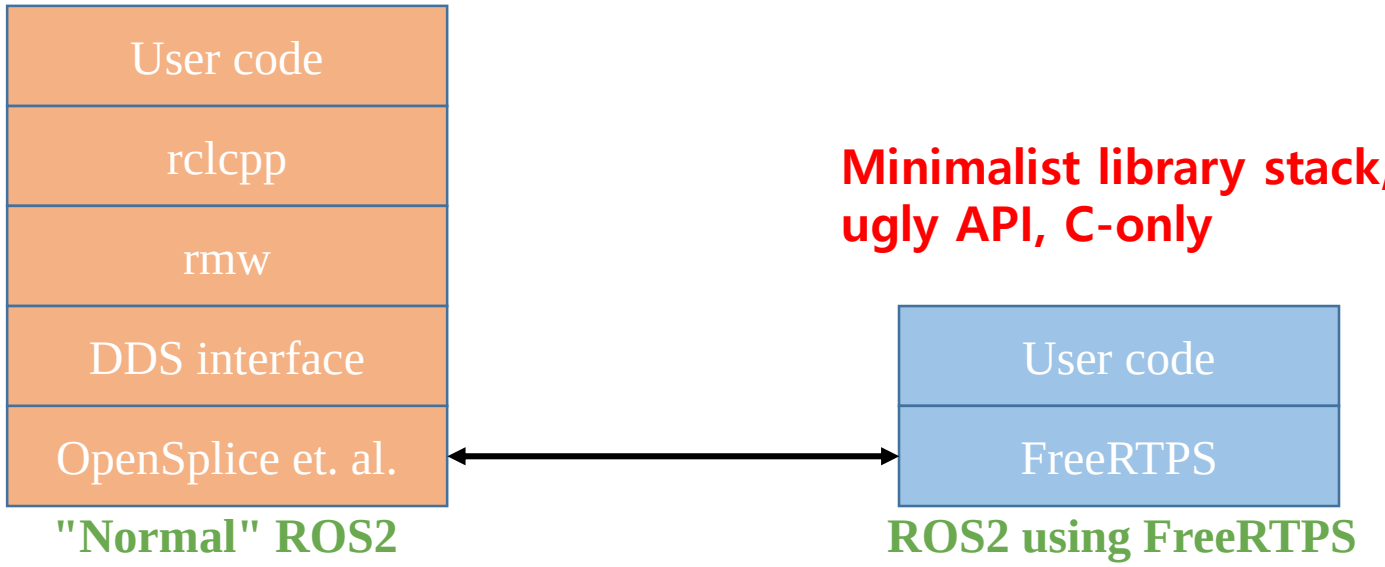


FreeRTPS

- RTPS for embedded systems
- Apache2 License
- https://github.com/ros2/freertps
- It can use on MCU and on Linux.



Flexible library stack, elegant API via C++



Morgan Quigley (OSRF) "ROS 2 on ″small″ embedded systems", ROSCon2015

ROS 2 features for embedded systems

- ROS2 / DDS / RTPS is much more embedded friendly than the ROS1 protocols

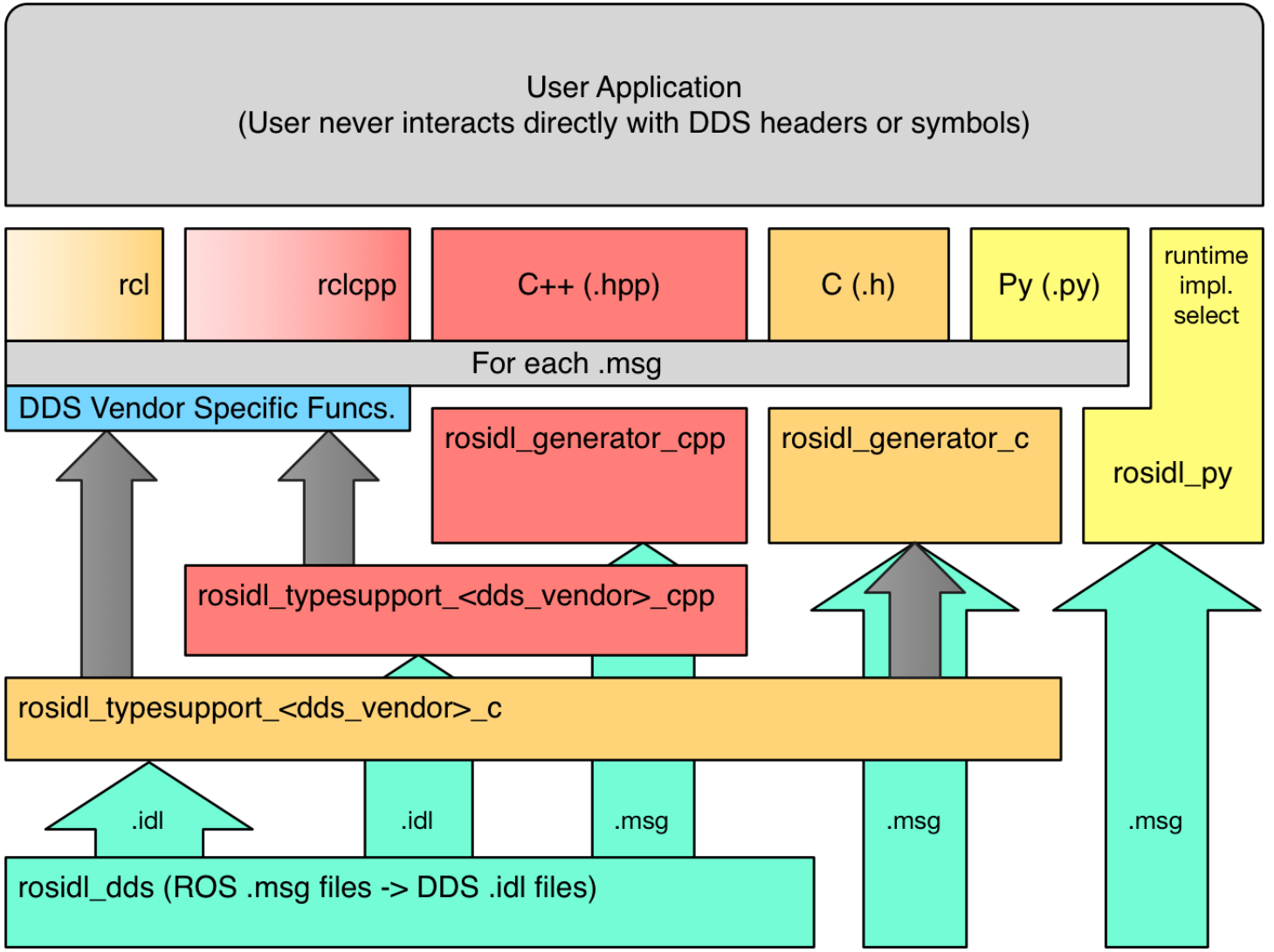
ROS 1	ROS 2
• *startup sequencing*	• *no master*

• *XML-RPC discovery*
  ▫ *parse XML trees*
	• *multicast UDP discovery*
	  ▫ *parse parameter lists*

• *TCP data streams*	• *RTPS/UDP data streams*
• *UDPROS not complete*	• *extensive QoS on UDP*

Morgan Quigley (OSRF) "ROS 2 on ″small″ embedded systems", ROSCon2015

State of ROS 2: Architecture

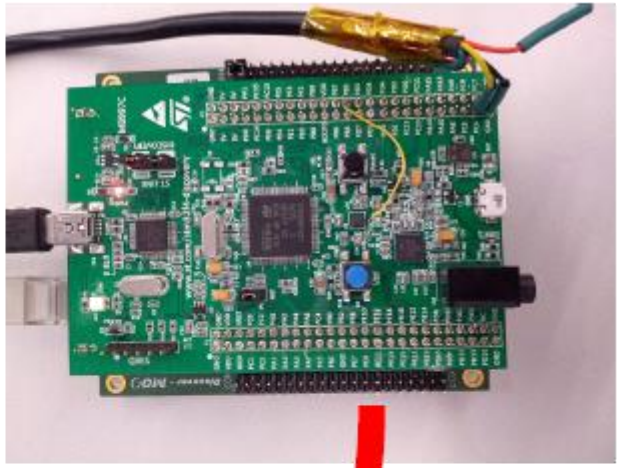


* replace <dds_vendor> with a package for each DDS vendor

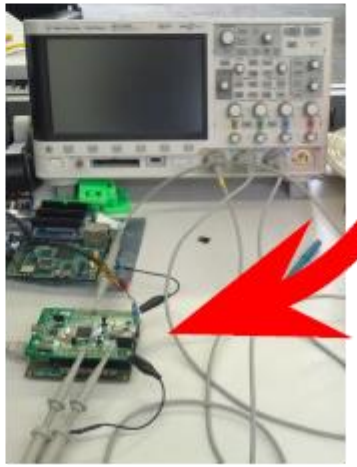
https://github.com/ros2/ros_core_documentation/

Performance measurements: IMU demo

STM32F4-Discovery stack: $55
Slightly modified to use both Ethernet PHY and IMU
Goal: measure FreeRTPS jitter

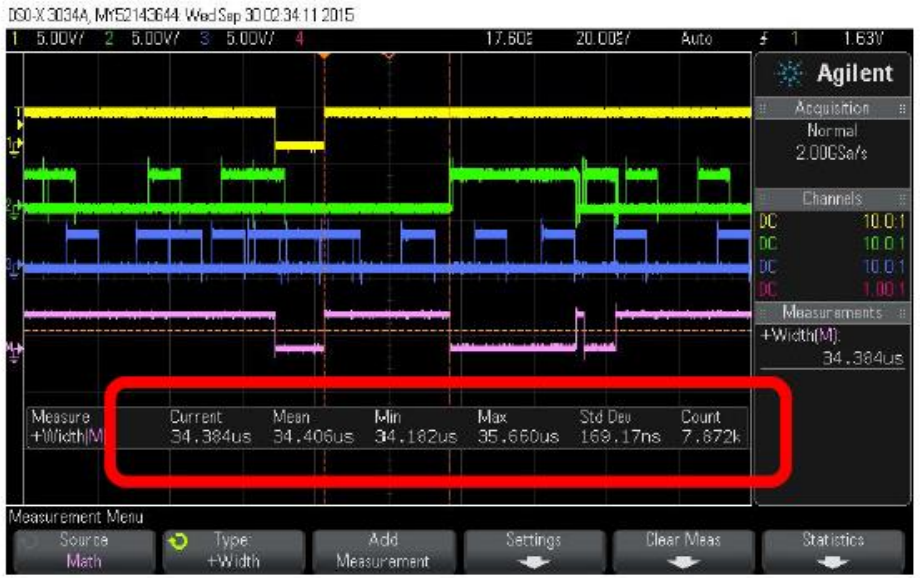


Accelerometer CS and Ethernet TXEN signals to Agilent DSO-X 3034A



IMU CS
Ethernet TXEN
Ethernet RXDV

time calculation



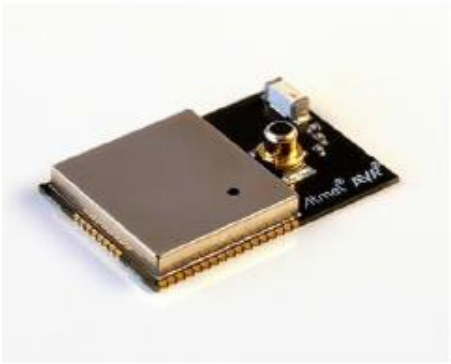
Contents

State of ROS 2

- Goals

Support multi-robot systems involving unreliable networks

Remove the gap between prototyping and final products



"Bare-metal" micro controller

Support for real-time control

Cross-platform support

Dirk Thomas (OSRF) et. al. "State of ROS 2 - demos and the technology behind", ROSCon2015

State of ROS 2: TODO list

- New APIs
- ROS on DDS
- Mapping between ROS interface types and DDS IDL types
- RPC API design in ROS
- Parameter API design in ROS
- ROS 2 middleware interface
- Real-time Systems
- Build system ‘ament’

State of ROS 2: Changes between ROS 1 and ROS 2

	ROS 1	ROS 2
Platforms	Ubuntu, OS X	**Ubuntu, OS X, Windows**
Communication	XMLRPC + ROSTPC	**DDS**
Languages	C++03, Python 2	**C++11, Python 3 (3.5+)**
Build system	rosbuild → catkin	**ament**
Messages, Services	*.msg, *.srv	**new *.msg, *.srv, *.msg.idl, *.srv.idl**
roslaunch	XML	**written in Python**
multiple nodes	one node in a process	**multiple nodes in a process**
real-time	external frameworks like Orocos	**real-time nodes when using a proper RTOS with carefully written user code**
Graph API	remapping at startup time only	**remapping at runtime**
Embedded systems	rosserial (UART)	**FreeRTPS (UART, Ethernet, WiFi, et. al.)**
		No non-isolated build
		No devel space

State of ROS 2: Schedule

- 2015-08-31: ROS 2 Alpha1 release (code name Anchor)
- 2015-11-03: ROS 2 Alpha2 release (code name Baling wire)
- 2015-12-18: ROS 2 Alpha3 release (code name Cement)
- 2016-02-17: ROS 2 Alpha4 release (code name Duct tape)
- 2016-04-06: ROS 2 Alpha5 release (code name Epoxy)
- 2016-06-02: ROS 2 Alpha6 release (code name Fastener)
- 2016-07-14: ROS 2 Alpha7 release (code name Glue Gun)
- 2016-10-04: ROS 2 Alpha8 release (code name Hook-and-Loop)

- 2016-12-19: ROS 2 Beta1 release (code name Asphalt)
- 2017-07-05: ROS 2 Beta2 release (code name R2B2)
- 2017-09-13: ROS 2 Beta3 release (code name R2B3)

- **2017-12-08: ROS 2 release (Ardent Apalone)**

Contents

What the Future Robot Operating System should Have?

- Function
- Community power
- Cooperation with industrial robot vendors
- Catch hardware
- Securing commercial viability, building a complete ecosystem

What the Future Robot Operating System should Have?

- **Function**
  - [大同小異]
  - At first, there were features of each, but they are becoming similar while complementing each other
  - Funcionality is essential, not discrimination
- **Community power**
- **Cooperation with industrial robot vendors**
- **Catch hardware**
- **Securing commercial viability, building a complete ecosystem**

What the Future Robot Operating System should Have?

- **Function**
  - [大同小異]
  - At first, there were features of each, but they are becoming similar while complementing each other
  - Funcionality is essential, not discrimination
- **Community power**
  - It is very helpful to enhance functions, quick feedback, content creation, dissemination, documentation and standardization
  - There is no future for robotic operating systems without a developer/user community
- **Cooperation with industrial robot vendors**
- **Catch hardware**
- **Securing commercial viability, building a complete ecosystem**

What the Future Robot Operating System should Have?

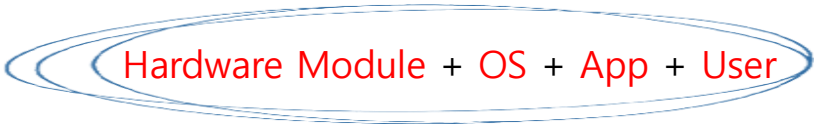
- **Function**
  - [大同小異]
  - At first, there were features of each, but they are becoming similar while complementing each other
  - Funcionality is essential, not discrimination
- **Community power**
  - It is very helpful to enhance functions, quick feedback, content creation, dissemination, documentation and standardization
  - There is no future for robotic operating systems without a developer/user community
- **Cooperation with industrial robot vendors**
  - [吳越同舟] (ROS-Industrial)
  - Industrial robot vendors can not disclose their technology, but industry-academia cooperation should be done
- **Catch hardware**
- **Securing commercial viability, building a complete ecosystem**

What the Future Robot Operating System should Have?

- **Function**
  - [大同小異]
  - At first, there were features of each, but they are becoming similar while complementing each other
  - Funcionality is essential, not discrimination
- **Community power**
  - It is very helpful to enhance functions, quick feedback, content creation, dissemination, documentation and standardization
  - There is no future for robotic operating systems without a developer/user community
- **Cooperation with industrial robot vendors**
  - [吳越同舟] (ROS-Industrial)
  - Industrial robot vendors can not disclose their technology, but industry-academia cooperation should be done
- **Catch hardware**
  - Hardware abstraction platform realization, standard/specification proposal, modularization hardware induction
- **Securing commercial viability, building a complete ecosystem**

What the Future Robot Operating System should Have?

- **Function**
  - [大同小異]
  - At first, there were features of each, but they are becoming similar while complementing each other
  - Funcionality is essential, not discrimination
- **Community power**
  - It is very helpful to enhance functions, quick feedback, content creation, dissemination, documentation and standardization
  - There is no future for robotic operating systems without a developer/user community
- **Cooperation with industrial robot vendors**
  - [吳越同舟] (ROS-Industrial)
  - Industrial robot vendors can not disclose their technology, but industry-academia cooperation should be done
- **Catch hardware**
  - Hardware abstraction platform realization, standard/specification proposal, modularization hardware induction
- **Securing commercial viability, building a complete ecosystem**



  - ECO system
  - The ROS was academically influential. However, in terms of profits, it is necessary to develop the APP market with the developer's profit, like NAOqi, to lead to the development of rich advanced applications.

Question Time!

Advertisement #1

✓ Download link

✓ Language:

English, Chinese, Japanese, Korean

“ROS Robot Programming”

A Handbook is written by TurtleBot3 Developers

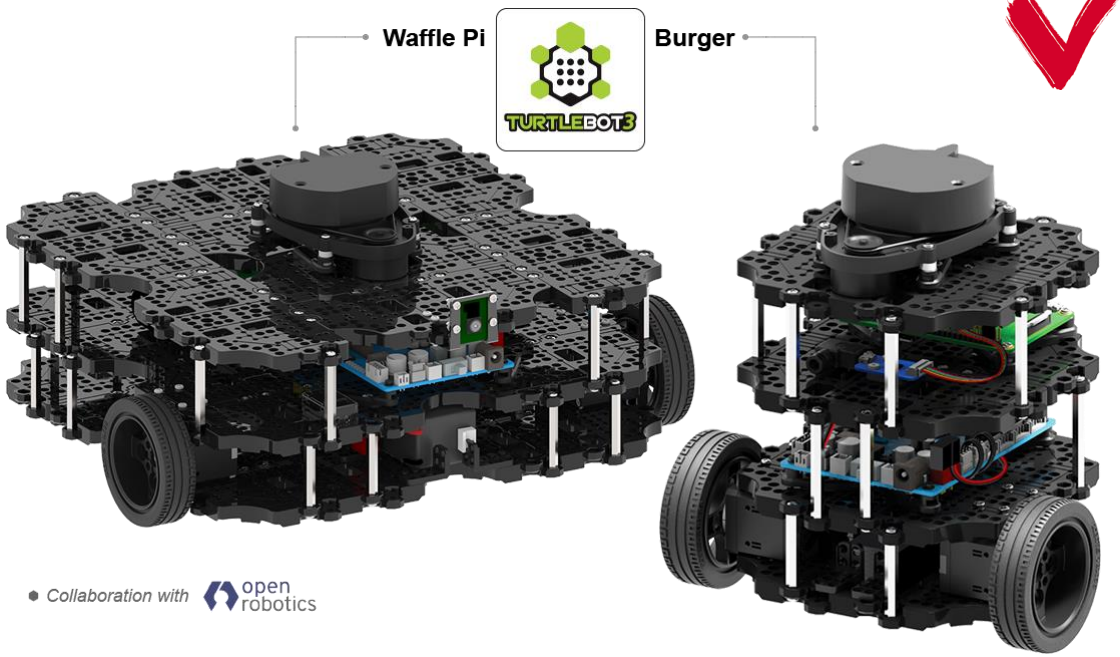
Advertisement #2

TURTLEBOT3

AI Research Starts Here
ROS Official Platform

TurtleBot3 is a new generation mobile robot that’s modular, compact and customizable. Let's explore ROS and create exciting applications for education, research and product development.

Direct Link



Advertisement #3

www.robotsource.org

The ‘RobotSource’ community is the space for people making robots.

We hope to be a community where we can share knowledge about robots, share robot development information and experiences, help each other and collaborate together. Through this community, we want to realize open robotics without disti nguishing between students, universities, research institutes and companies.

Join us in the Robot community ~

END.