

The Future of the Robot Operating System

ROS 2.0

ROBOTIS

Open Source Team

Yoonseok Pyo

Contents

I. ROS 2

II. ROS 2의 대표적인 3가지 기능

1. DDS (Data Distribution Service)
2. Real-time Computing
3. Embedded System

III. ROS 2의 개발 현황

IV. 미래의 로봇 운영체제가 갖추어야 할 것

Contents

I. ROS 2

II. ROS 2의 대표적인 3가지 기능

1. DDS (Data Distribution Service)
2. Real-time Computing
3. Embedded System

III. ROS 2의 개발 현황

IV. 미래의 로봇 운영체제가 갖추어야 할 것

ROSCon2015



Hamburg, Germany October 3-4, 2015

A grayscale photograph of a large audience seated in a hall, facing a stage with a large projection screen. The text "Hot Issue : ROS 2.0" is overlaid in red on the image.

Hot Issue : ROS 2.0

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

I. ROS 2

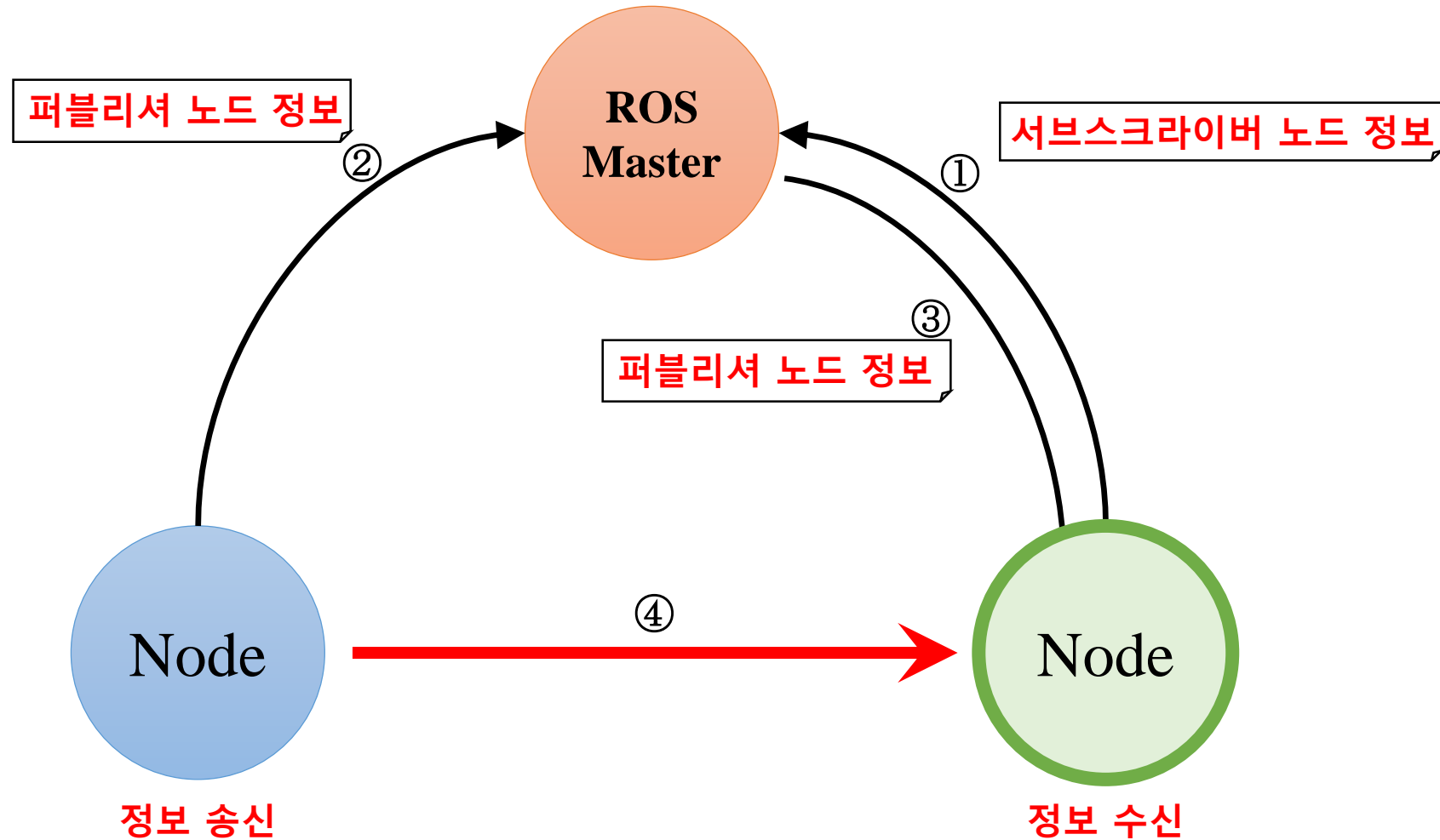
II. ROS 2의 대표적인 3가지 기능

1. DDS (Data Distribution Service)
2. Real-time Computing
3. Embedded System

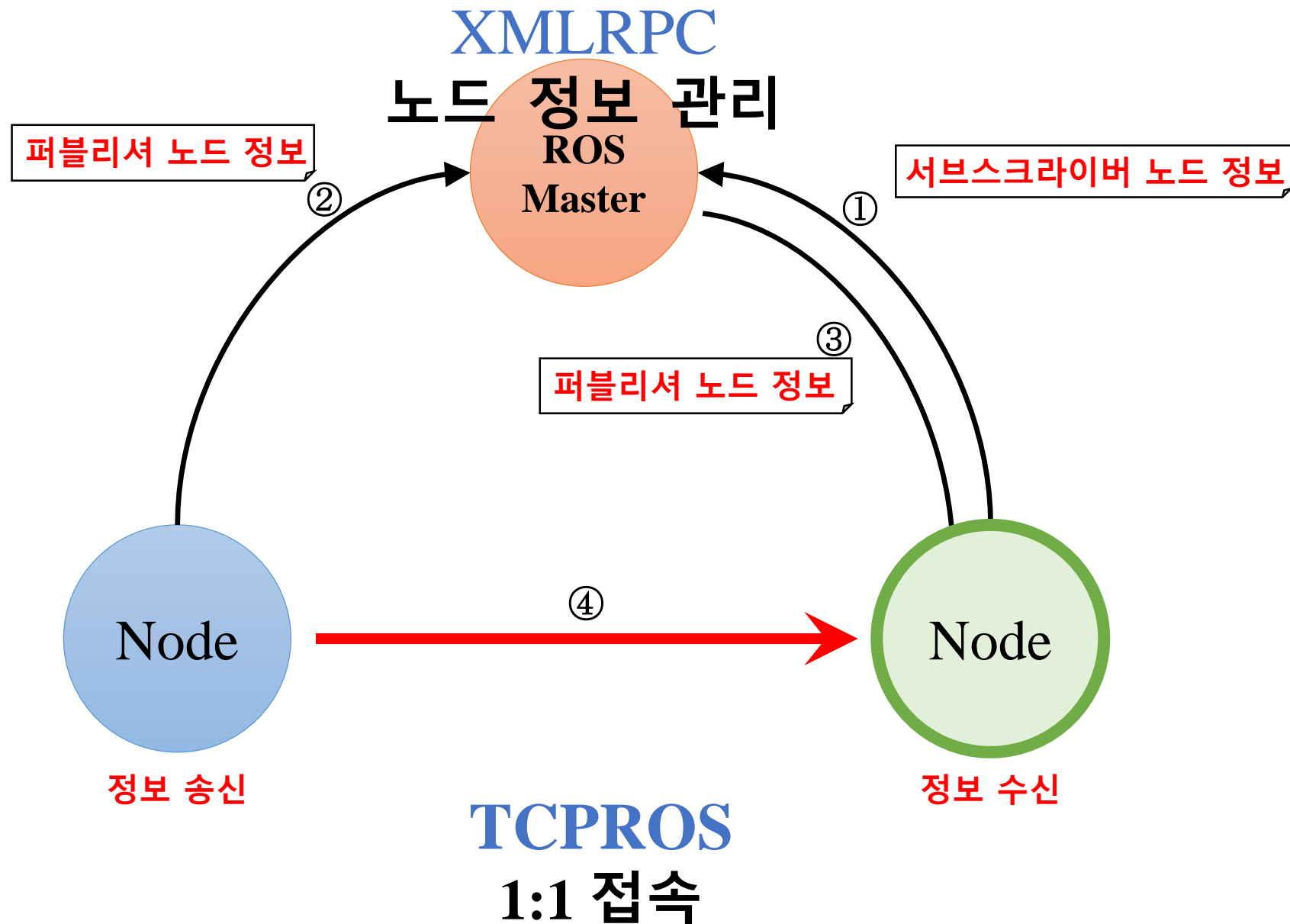
III. ROS 2의 개발 현황

IV. 미래의 로봇 운영체제가 갖추어야 할 것

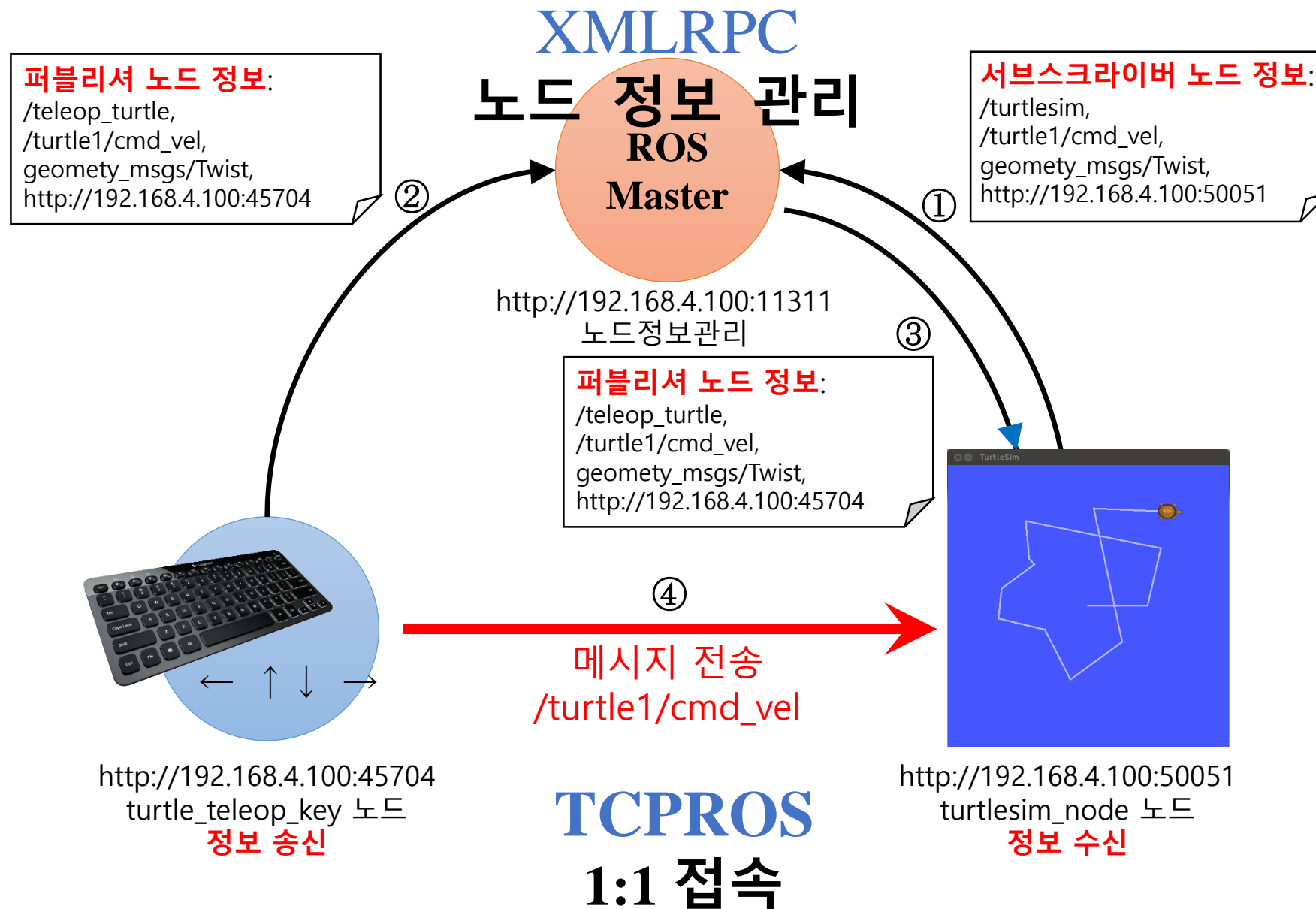
ROS 1.x (XMLRPC + TCPROS)



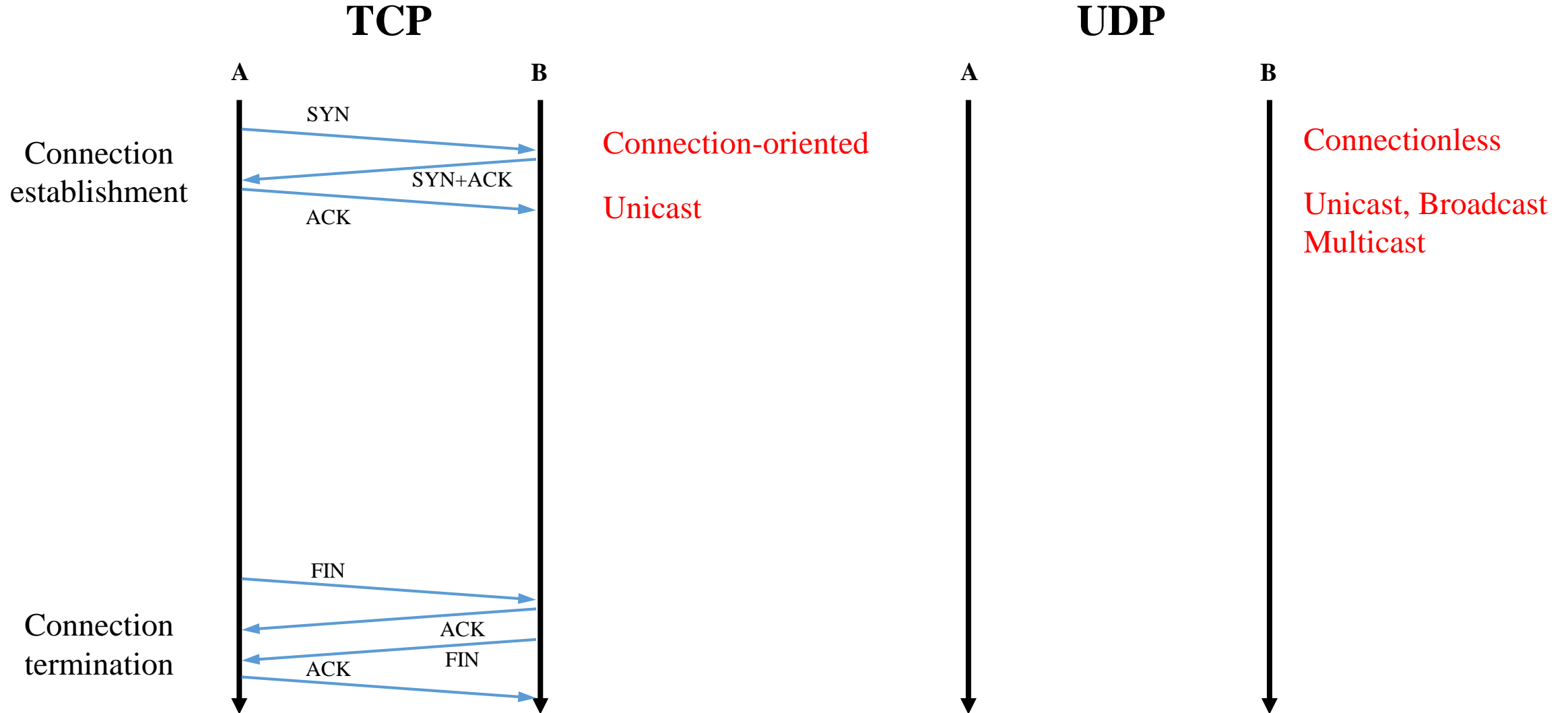
ROS 1.x (XMLRPC + TCPROS)



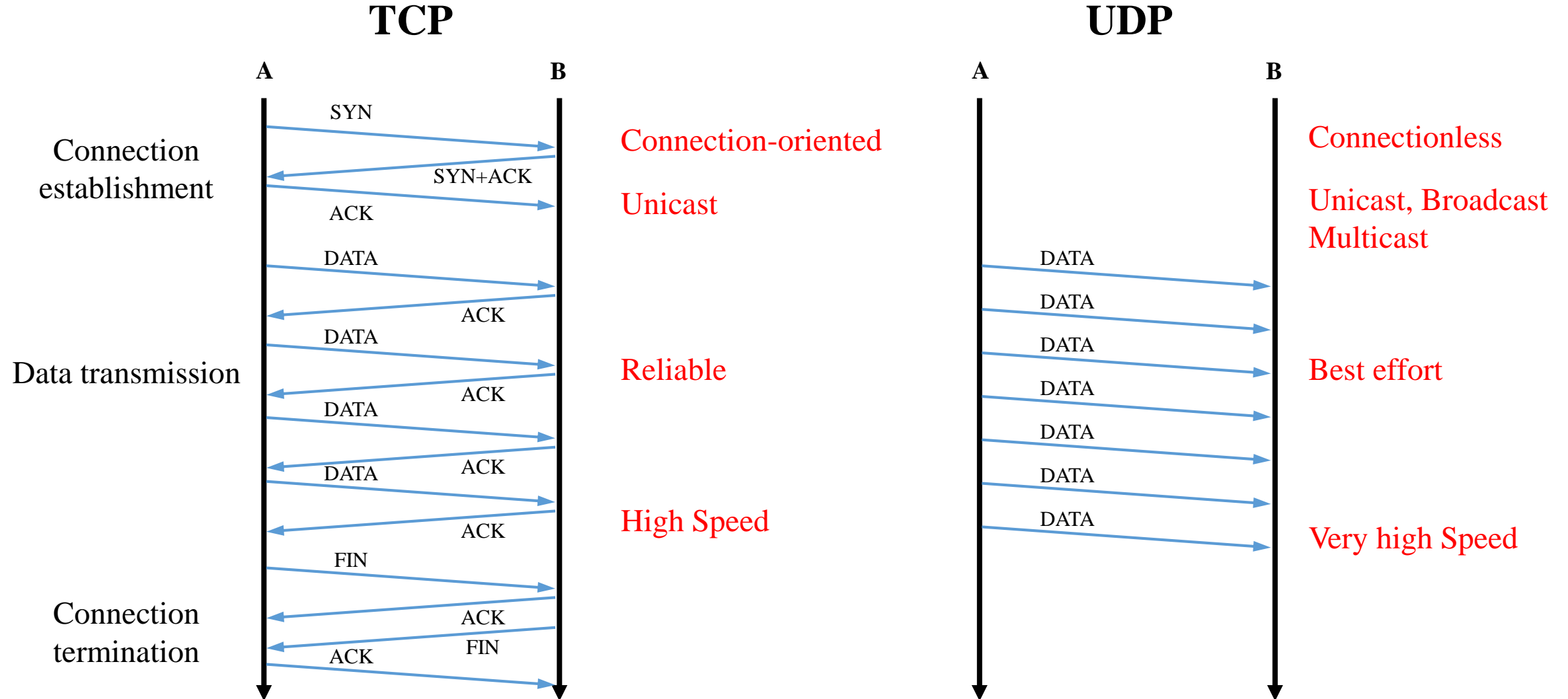
ROS 1.x (XMLRPC + TCPROS)



Transport: TCP vs UDP

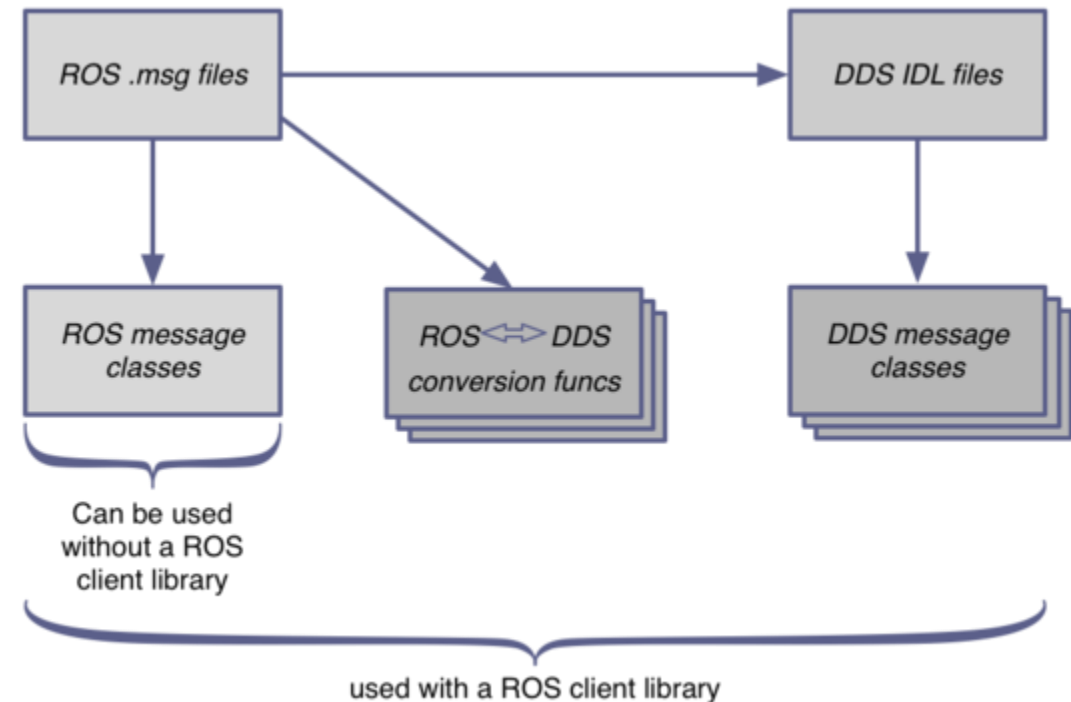
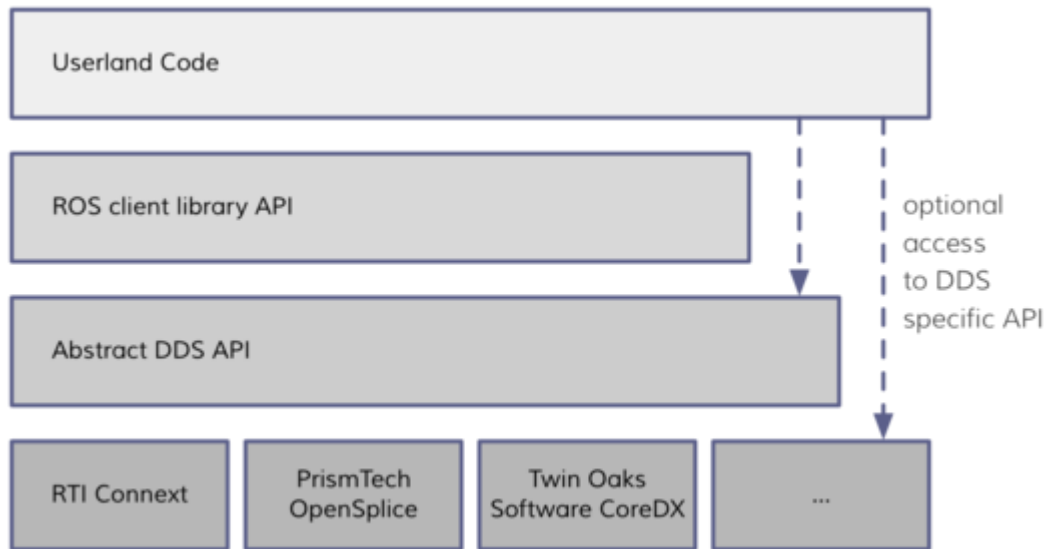


Transport: TCP vs UDP



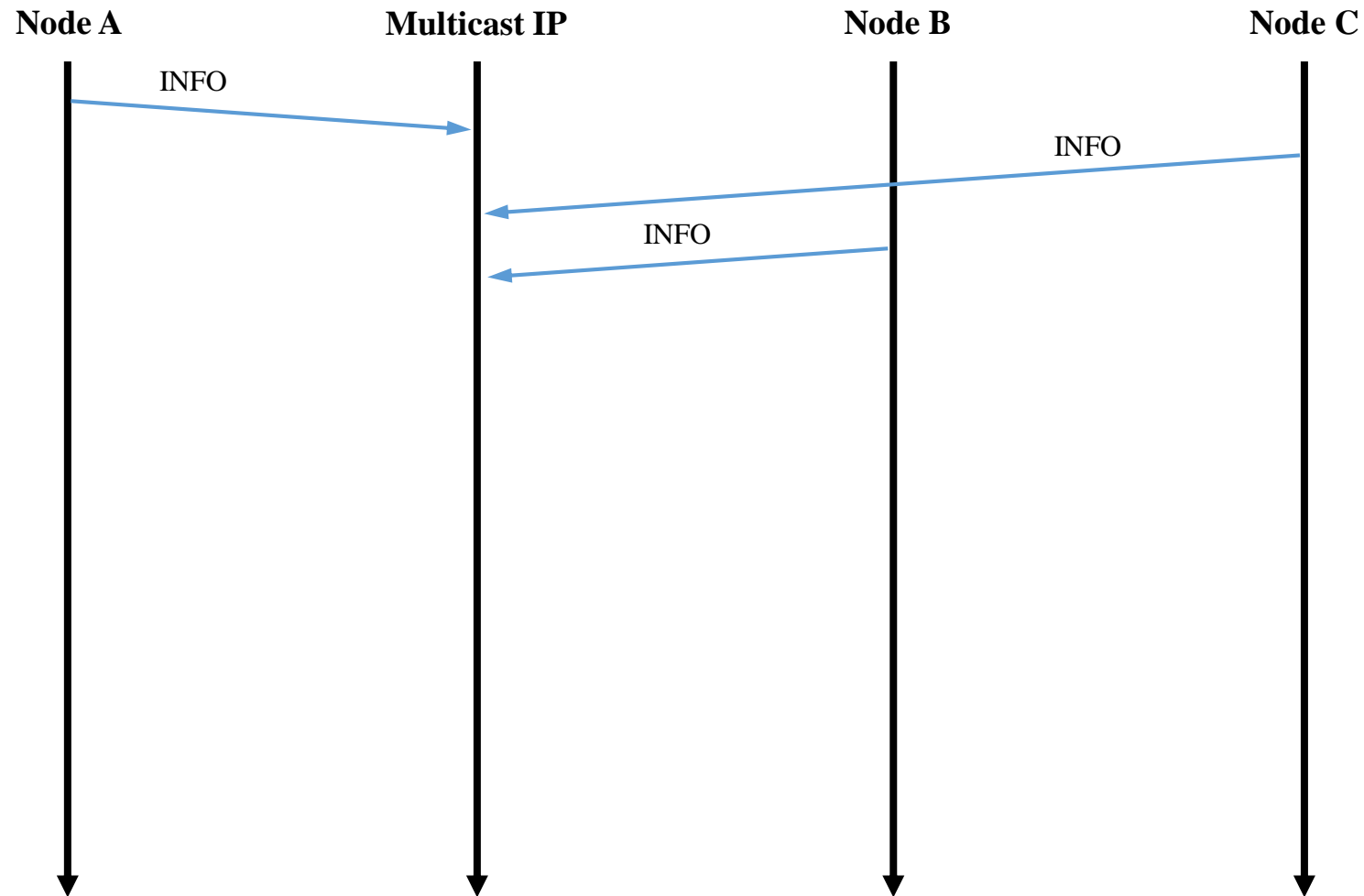
DDS (Data Distribution Service)

- DDS: 데이터 중심의 분산 시스템의 출판/구독형 미들웨어
Object Management Group(OMG)가 관리
- 통신 프로토콜: RTPS (Real Time Publish Subscribe)
UDP기반이지만, TCP의 장점을 가지고 있다.



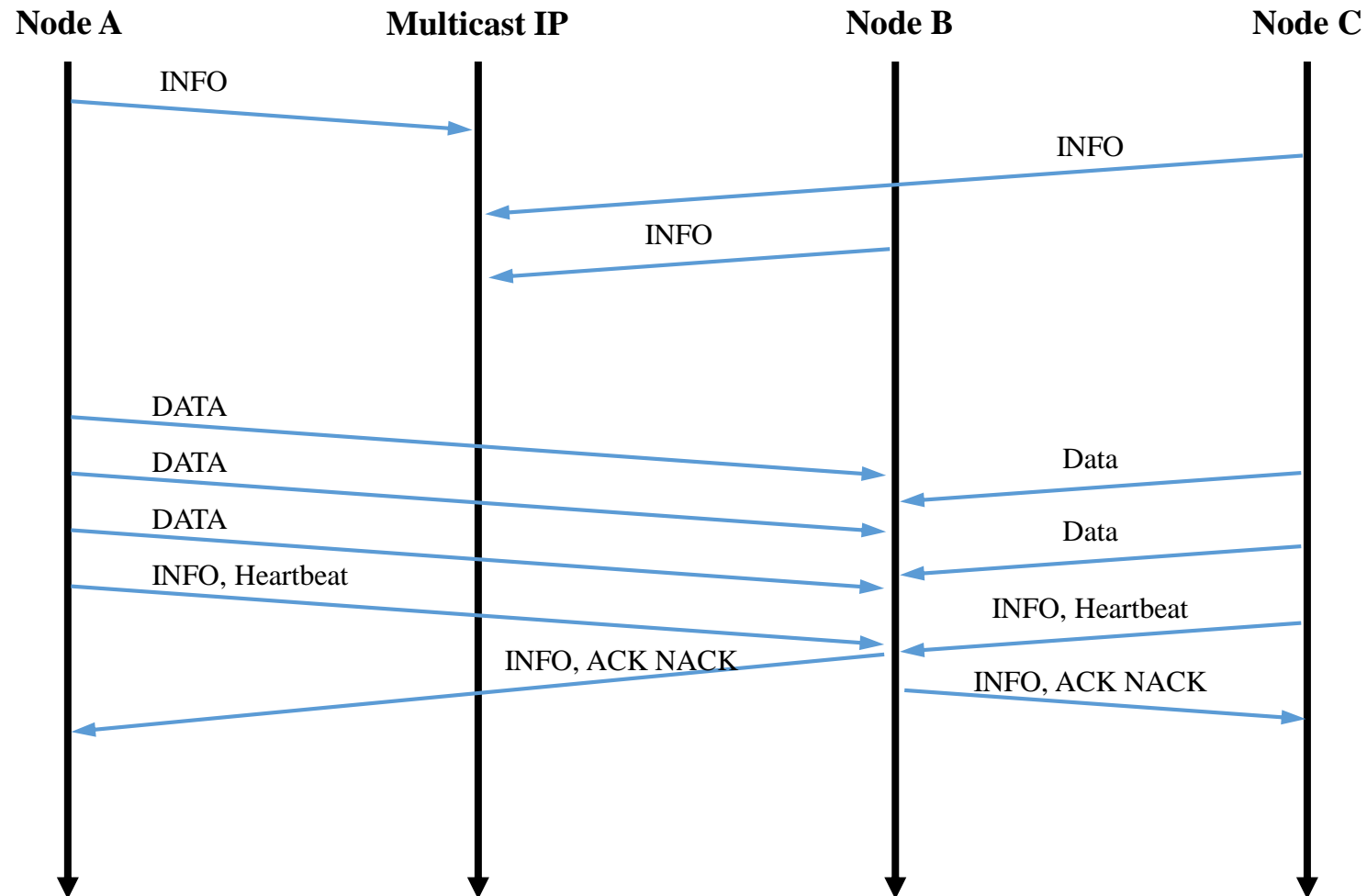
DDS (Data Distribution Service)

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



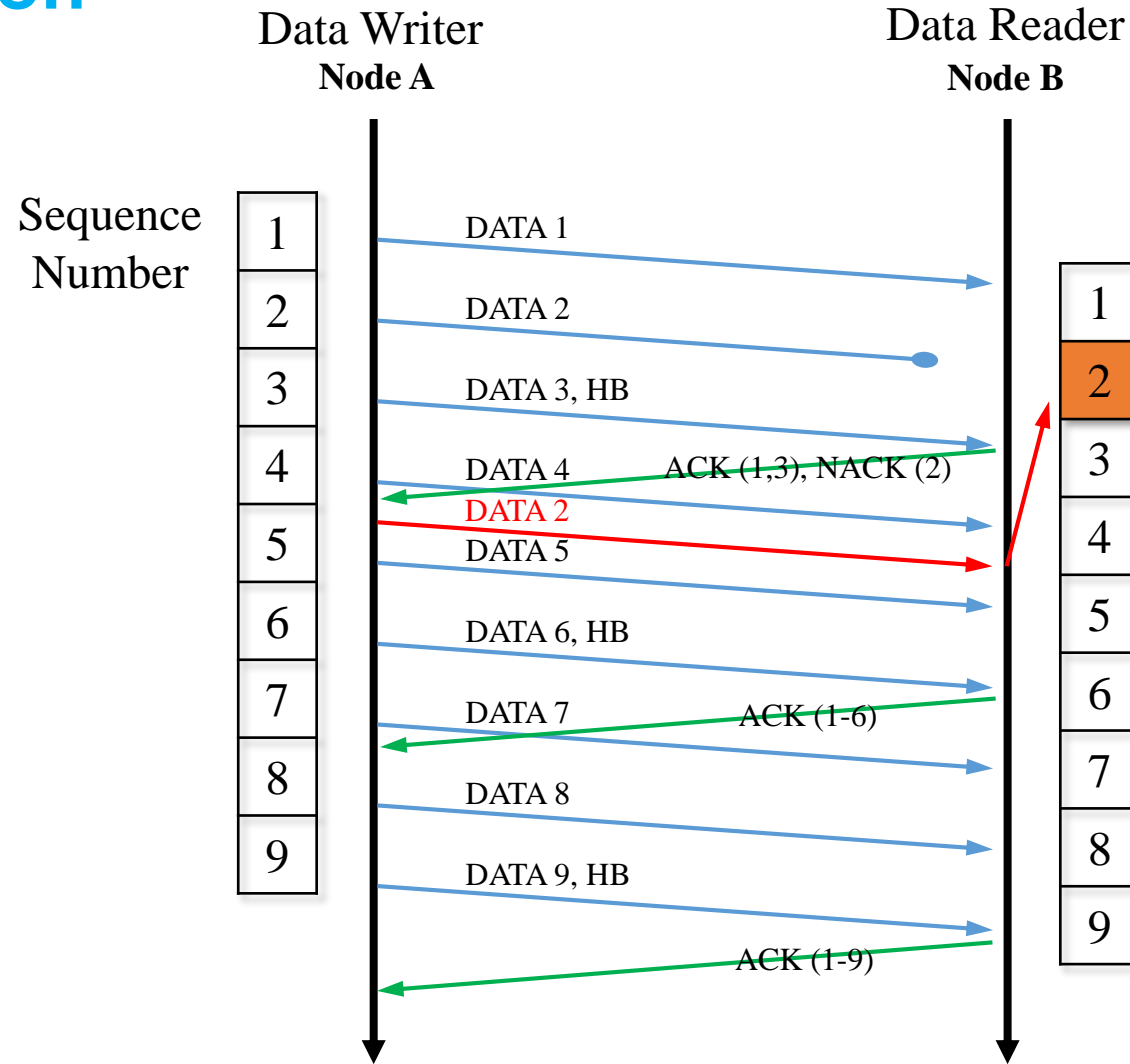
DDS (Data Distribution Service)

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



DDS (Data Distribution Service)

- Retransmission



Contents

I. ROS 2

II. ROS 2의 대표적인 3가지 기능

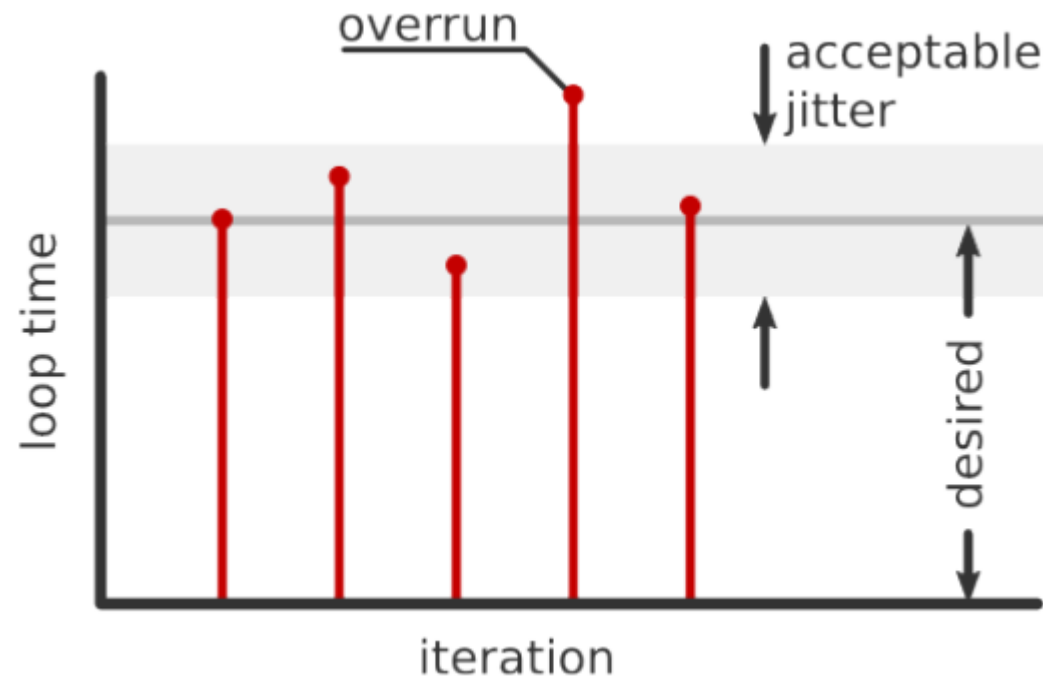
1. DDS (Data Distribution Service)
2. Real-time Computing
3. Embedded System

III. ROS 2의 개발 현황

IV. 미래의 로봇 운영체제가 갖추어야 할 것

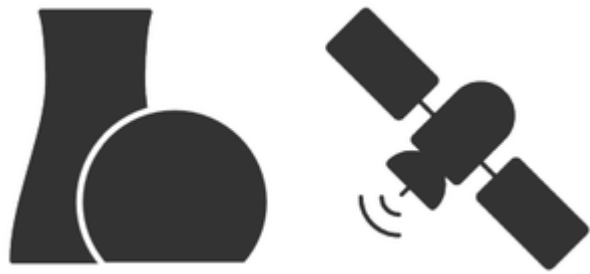
Real-time Computing

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



Real-time Computing

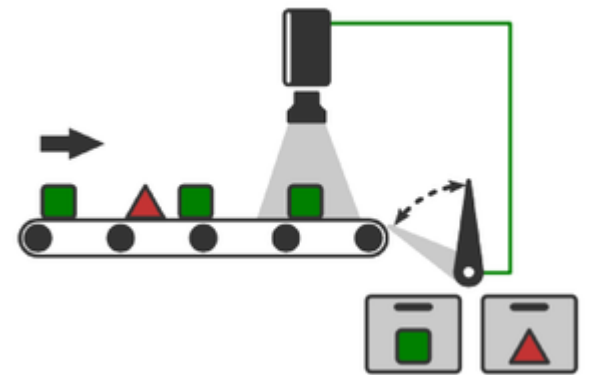
- Hard real-time systems
- Soft real-time systems
- Firm real-time systems



reactor, aircraft, spacecraft control



audio / video streaming



financial forecasting,
robot assembly lines

Real-time Computing

- Use an OS able to deliver the required determinism

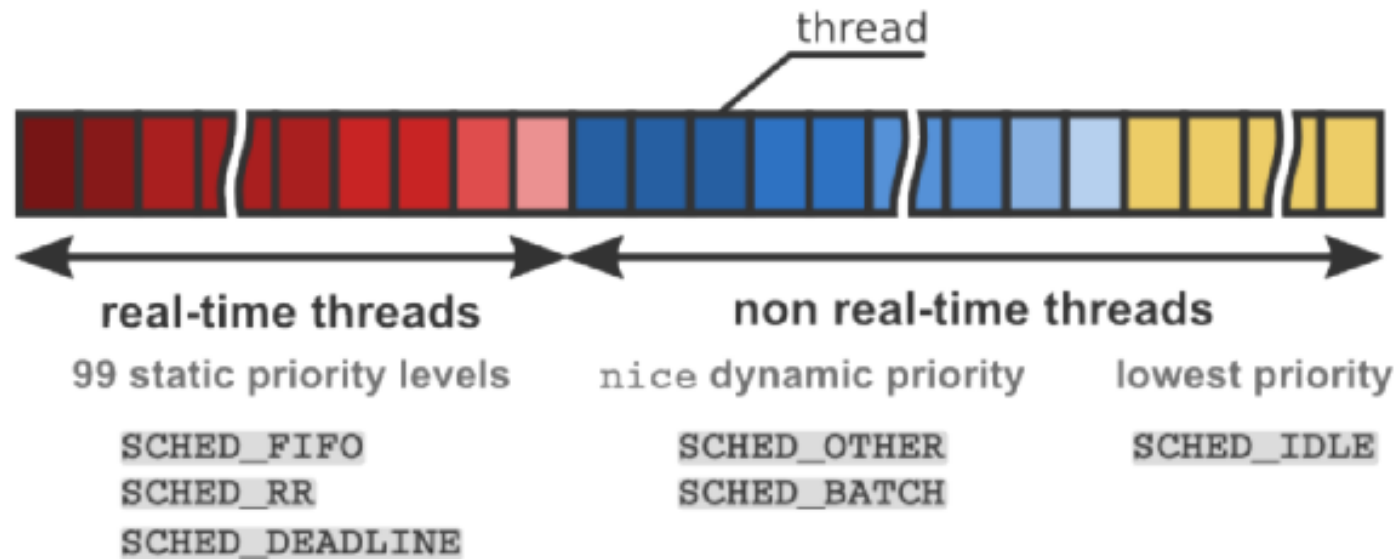
OS	real-time	max latency (μ s)
Linux	no	10^4
RT PREEMPT	soft	10^1 - 10^2
Xenomai	hard	10^1

Real-time Computing

- Use an OS able to deliver the required determinism

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

- Prioritize real-time threads

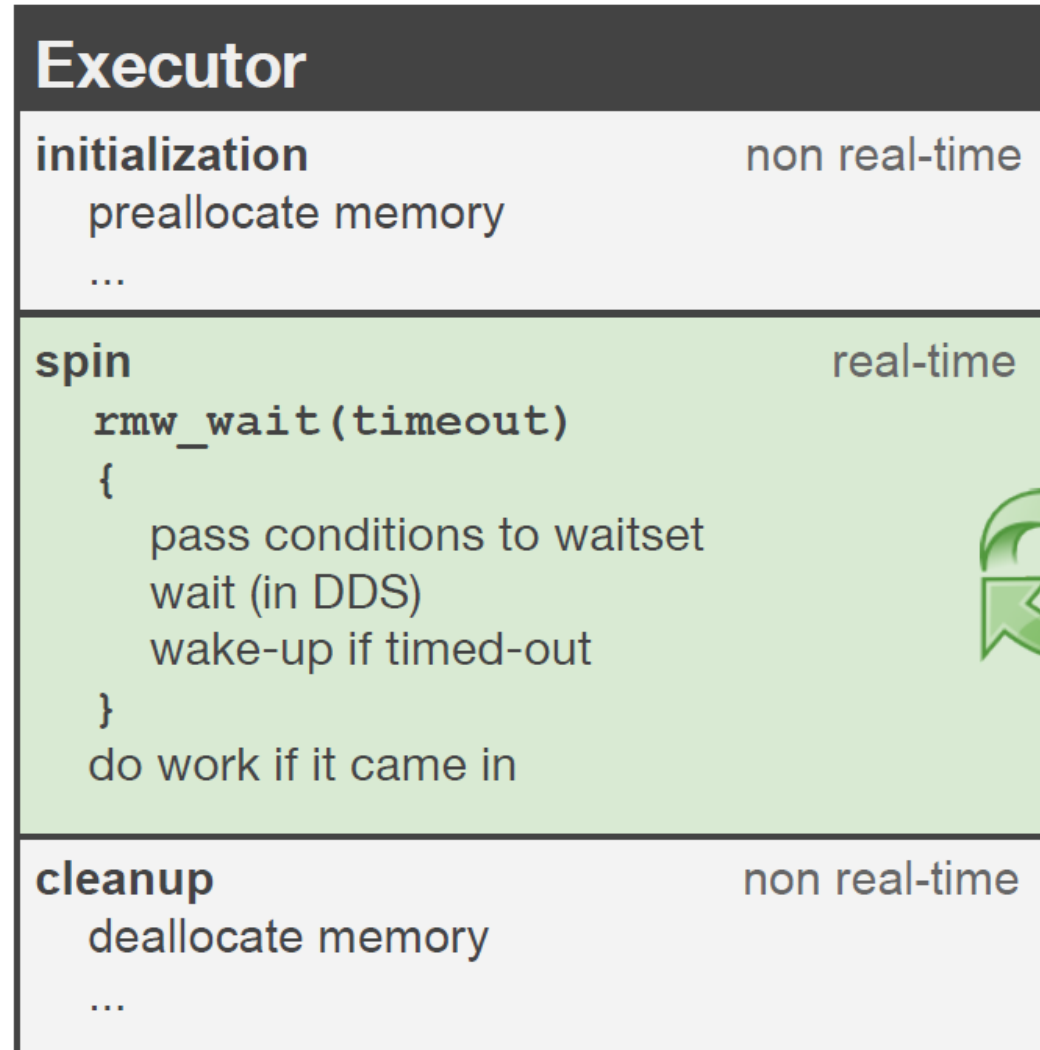


Real-time Computing

- 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

Real-time Computing

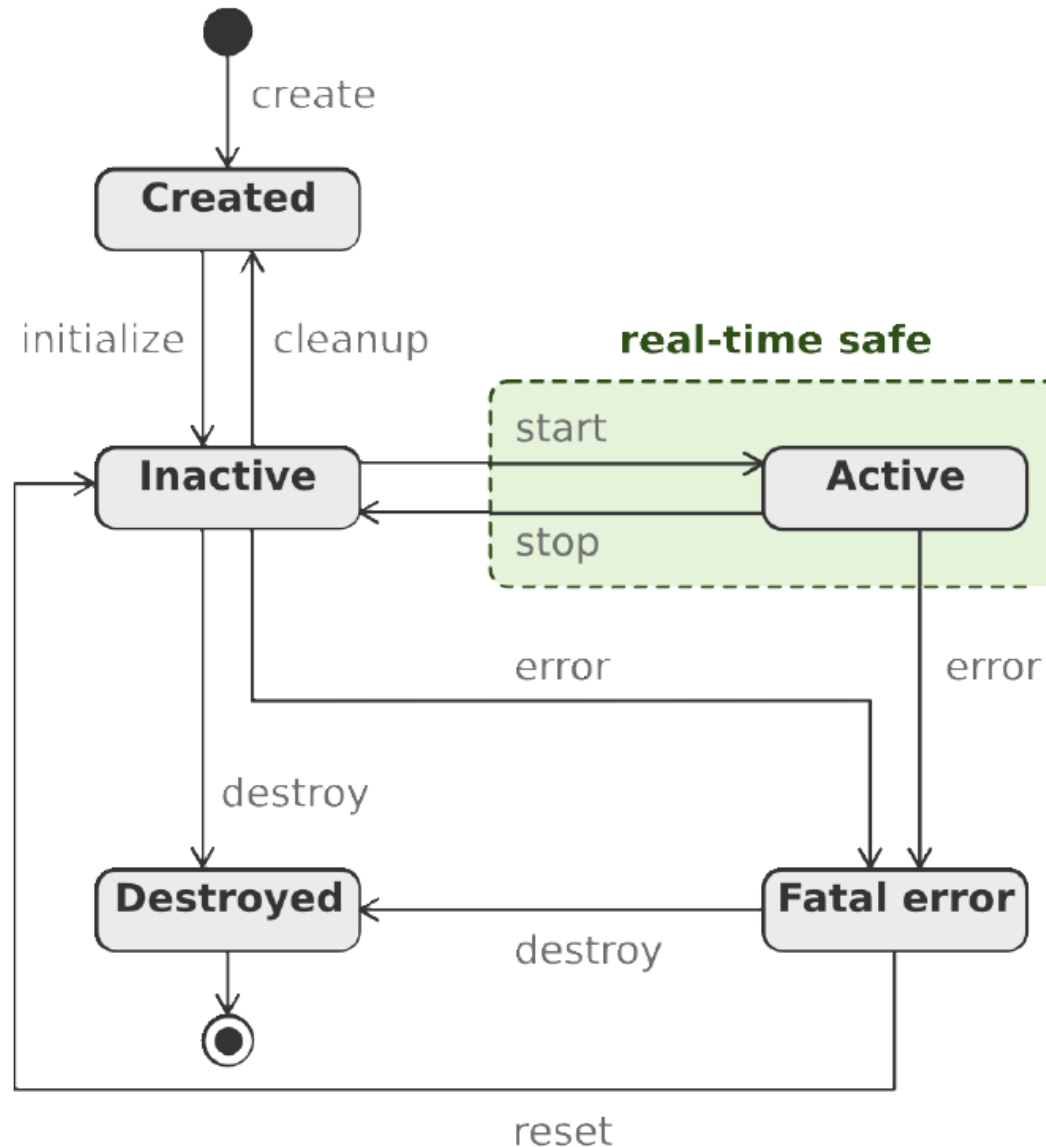
- Real-time code



loop until interrupted

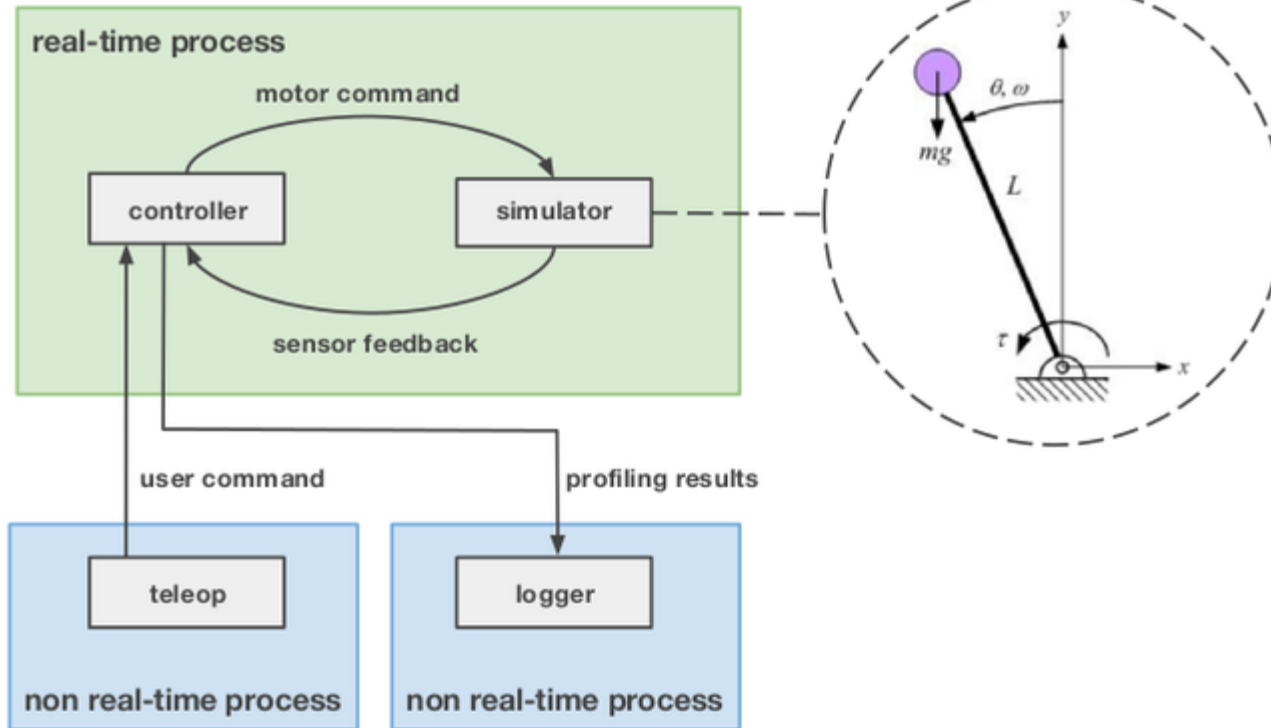
Real-time Computing

- Node lifecycle



Real-time Computing

- Real-time Benchmarking

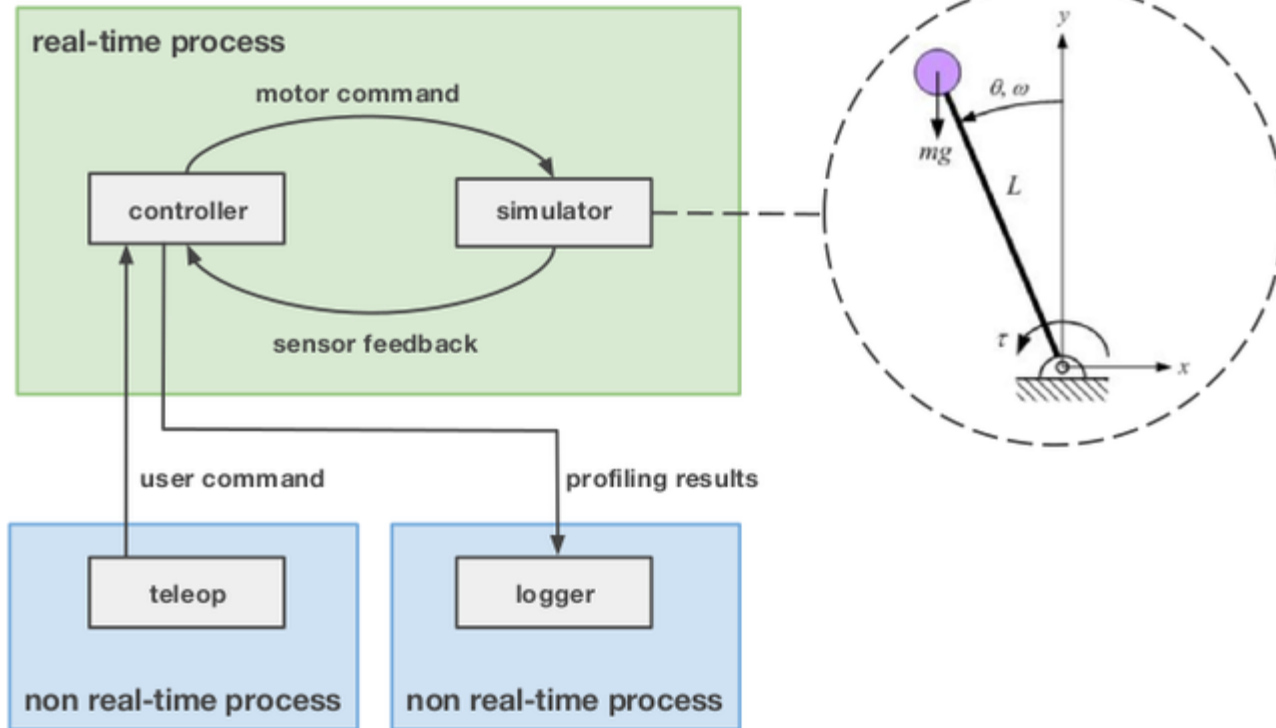


Goal

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

Real-time Computing

- Real-time Benchmarking



Goal

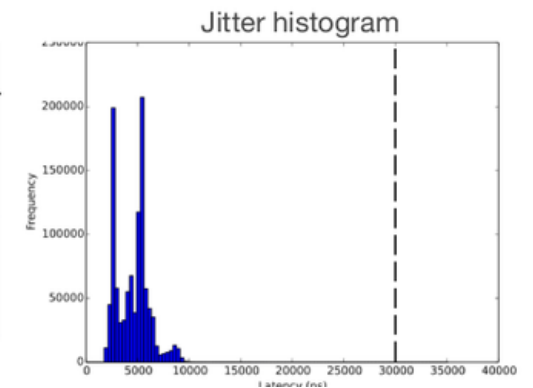
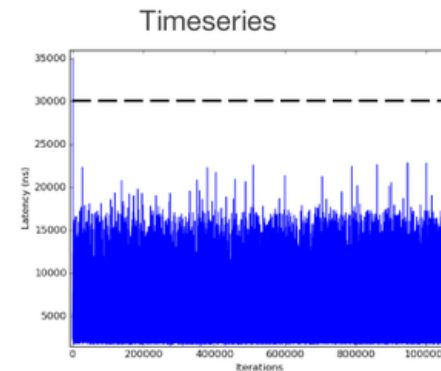
- **1 KHz** update loop (1 ms period)
- Less than **3%** jitter (30 μ s)

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%



Contents

I. ROS 2

II. ROS 2의 대표적인 3가지 기능

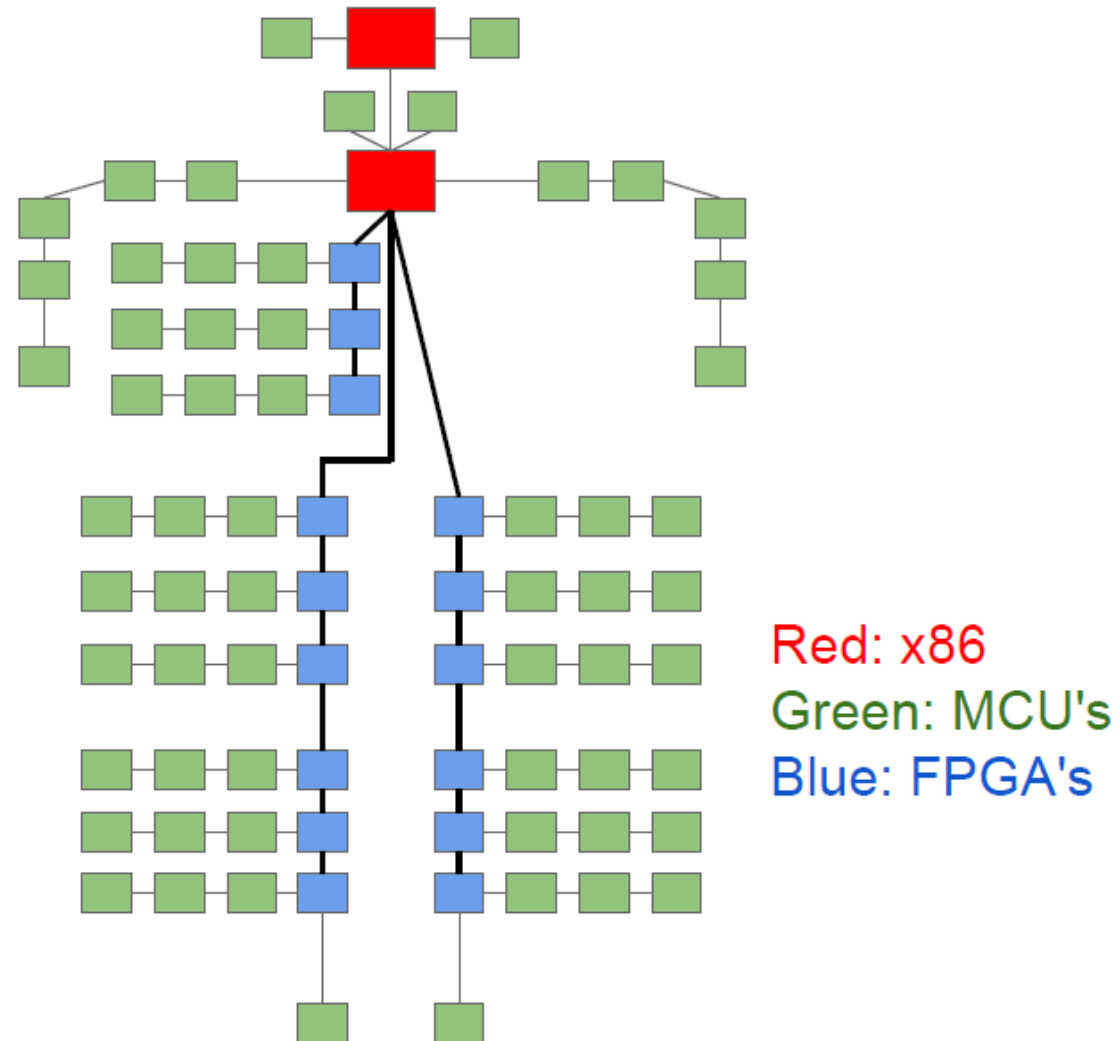
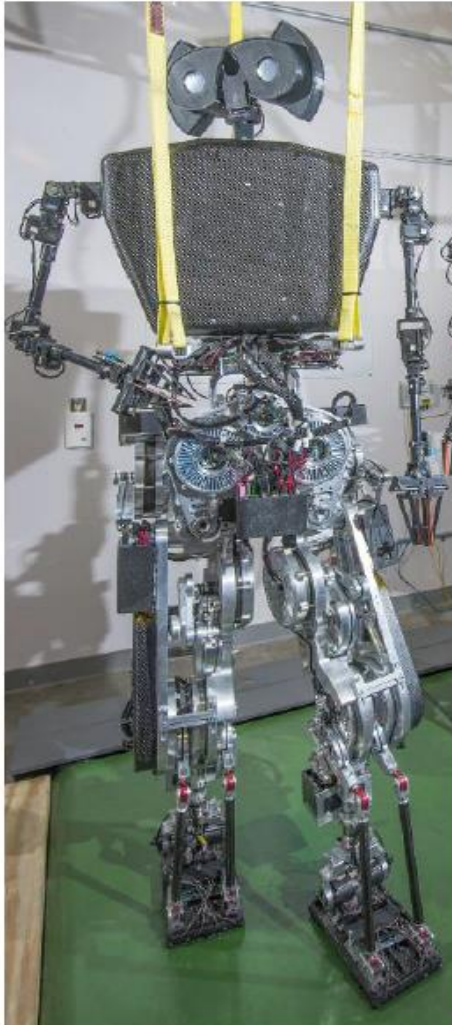
1. DDS (Data Distribution Service)
2. Real-time Computing
3. Embedded System

III. ROS 2의 개발 현황

IV. 미래의 로봇 운영체제가 갖추어야 할 것

Embedded Systems

- Small embedded systems are everywhere!



Scope

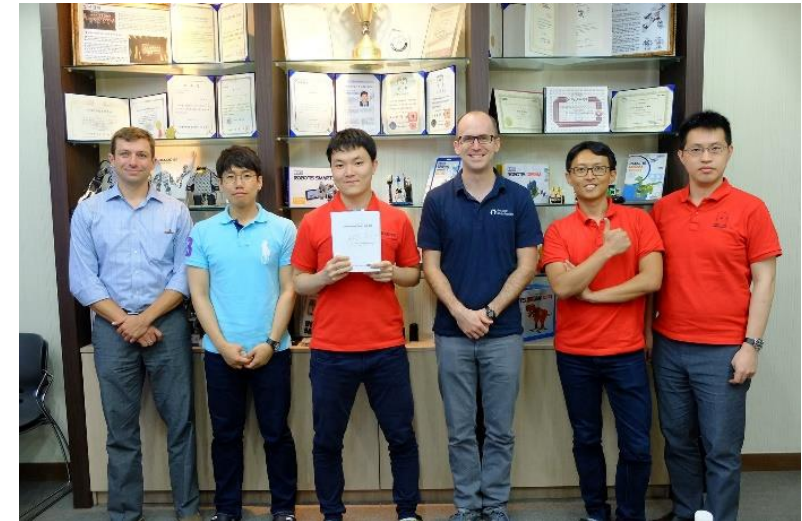
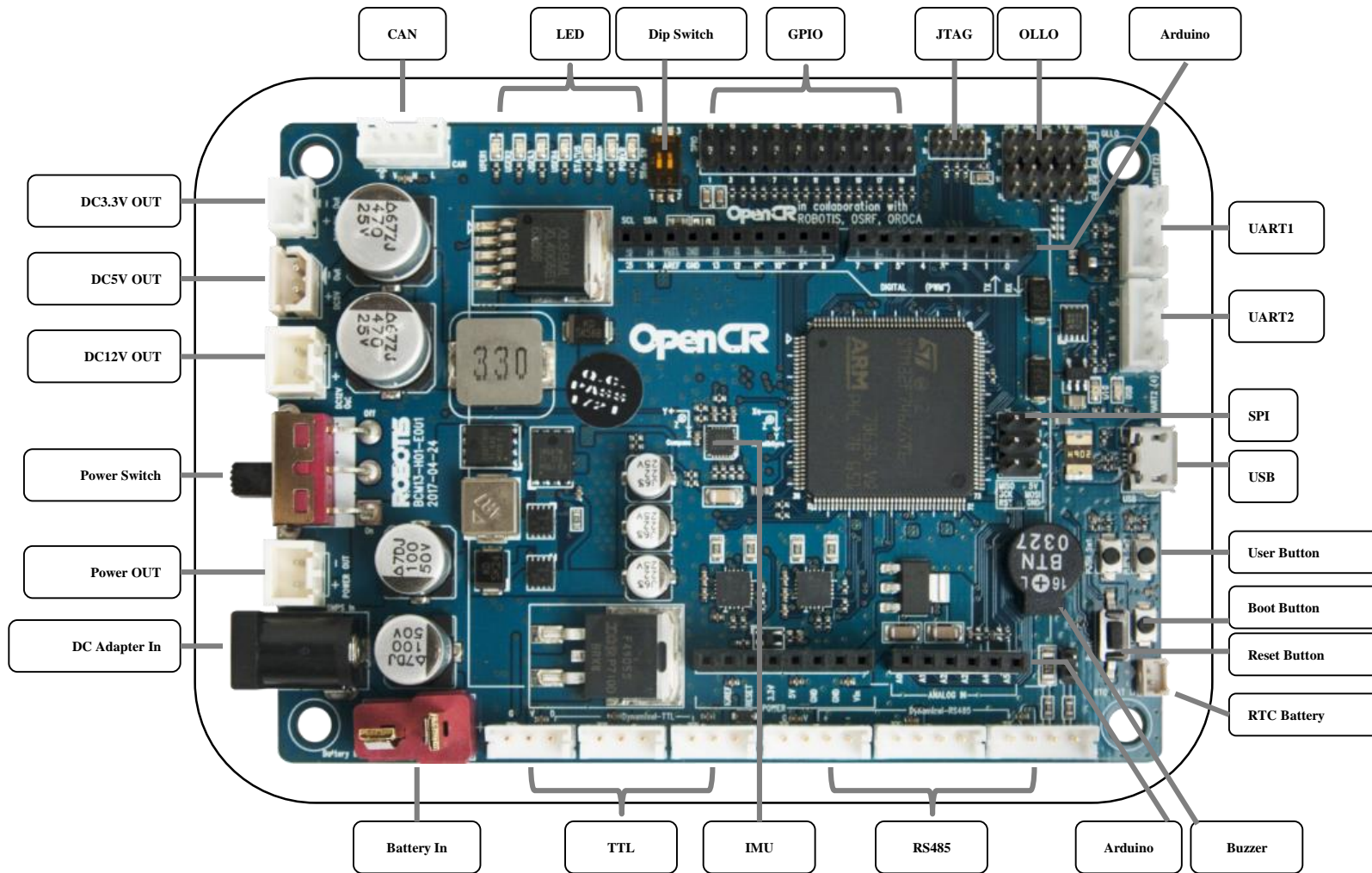


	8/16-bit MCU	32-bit MCU		ARM A-class (smartphone without screen)	x86 (laptop without screen)
		"small" 32-bit MCU	"big" 32-bit MCU		
Example Chip	Atmel AVR	ARM Cortex-M0	ARM Cortex-M7	Samsung Exynos	Intel Core i5
Example System	Arduino Leonardo	Arduino M0 Pro	SAM V71	ODROID	Intel NUC
MIPS	10's	100's	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



OpenCR

- Open-source Control module for ROS (OpenCR, [Link](#))
- '로스콘 2016'에서 'ROS 임베디드 보드' 발표 ([Link](#))

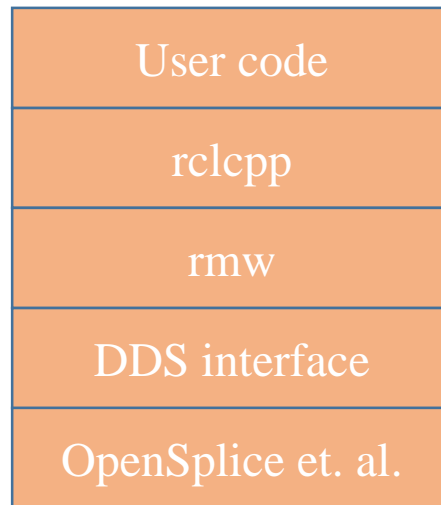


FreeRTPS

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

FreeRTPS User API	
Portable discovery, serialization, etc.	
Minimalist UDPv4	POSIX UDPv4
Vender Ethernet	Ethernet

**Flexible library stack,
elegant API via C++**



"Normal" ROS2

**Minimalist library stack,
ugly API, C-only**



ROS2 using FreeRTPS

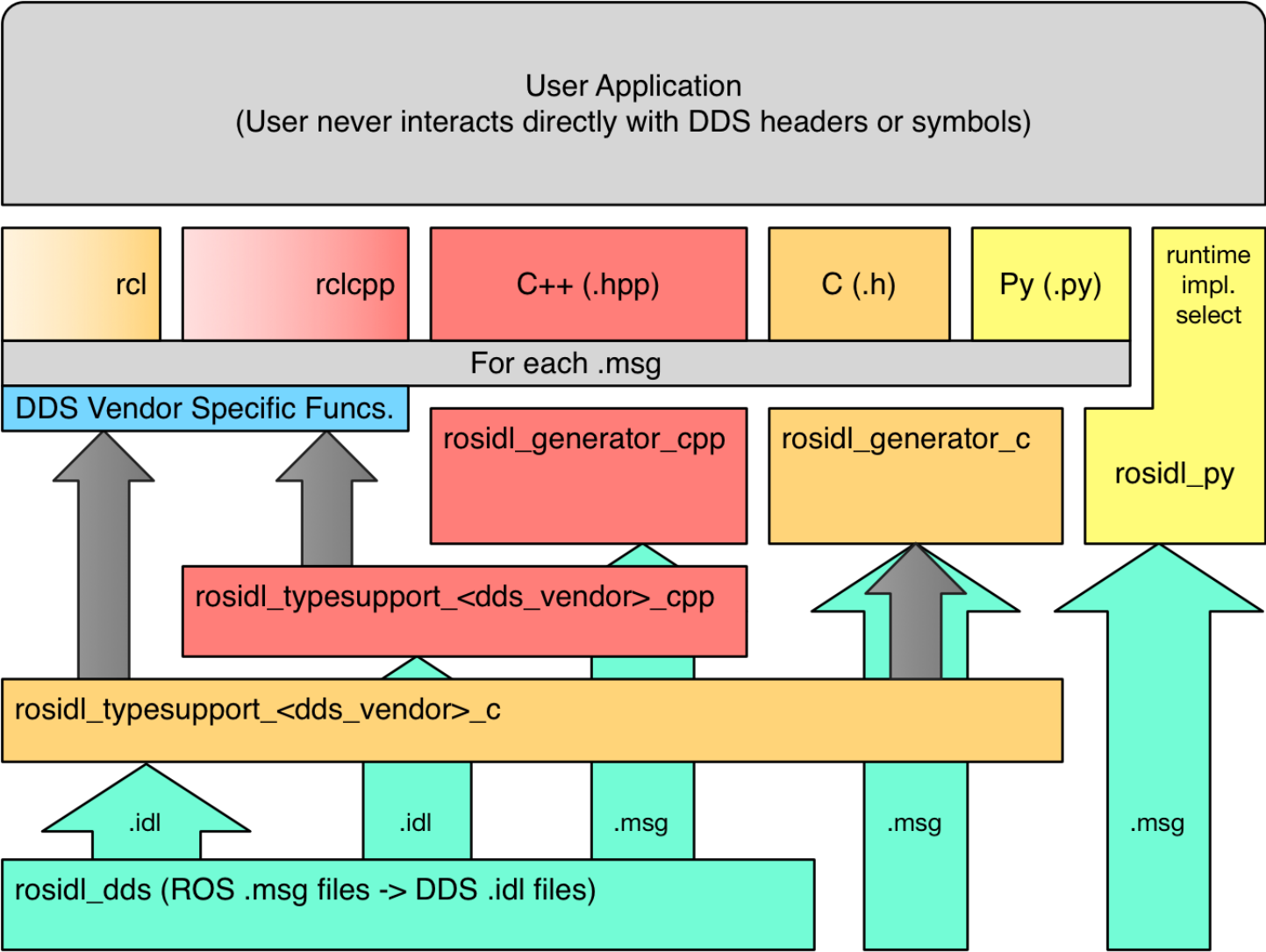


ROS 2 features for embedded systems

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

ROS 1	ROS 2
<ul style="list-style-type: none">• <i>startup sequencing</i>	<ul style="list-style-type: none">• <i>no master</i>
<ul style="list-style-type: none">• <i>XML-RPC discovery</i><ul style="list-style-type: none">▫ <i>parse XML trees</i>	<ul style="list-style-type: none">• <i>multicast UDP discovery</i><ul style="list-style-type: none">▫ <i>parse parameter lists</i>
<ul style="list-style-type: none">• <i>TCP data streams</i>	<ul style="list-style-type: none">• <i>RTPS/UDP data streams</i>
<ul style="list-style-type: none">• <i>UDPROS not complete</i>	<ul style="list-style-type: none">• <i>extensive QoS on UDP</i>

State of ROS 2: Architecture



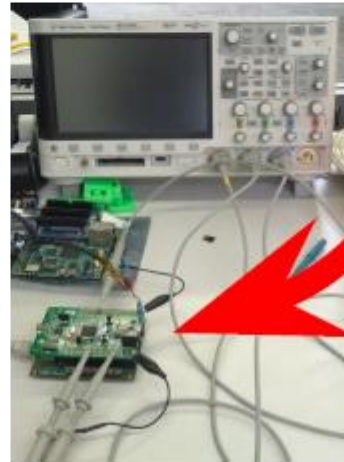
* replace <dds_vendor> with a package for each DDS vendor

Performance measurements: IMU demo

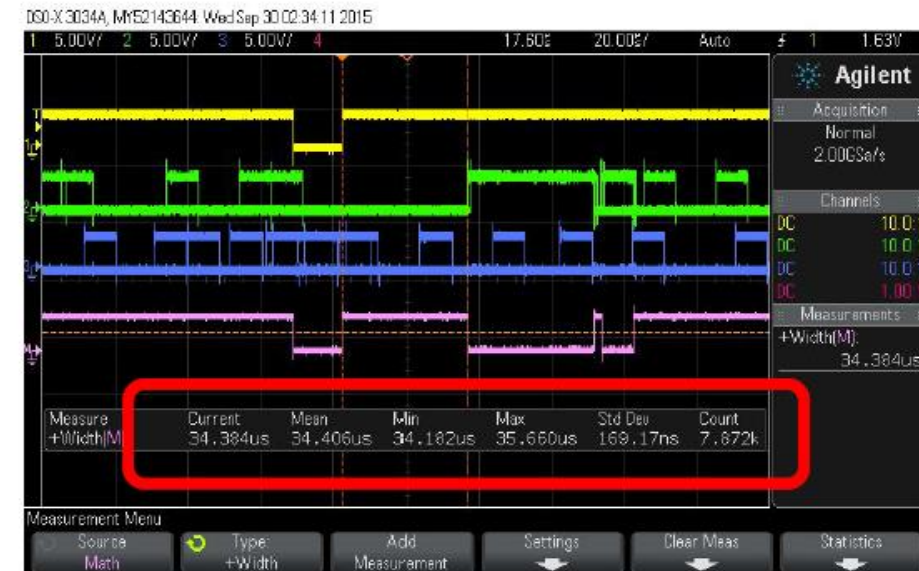
STM32F4-Discovery stack: \$55
Slightly modified to use both
Ethernet PHY and IMU
Goal: measure FreeRTOS jitter



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



IMU CS
Ethernet TXEN
Ethernet RXDV
time calculation



Contents

I. ROS 2

II. ROS 2의 대표적인 3가지 기능

1. DDS (Data Distribution Service)
2. Real-time Computing
3. Embedded System

III. ROS 2의 개발 현황

IV. 미래의 로봇 운영체제가 갖추어야 할 것

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

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, macOS	Ubuntu, macOS, Windows
Communication	XMLRPC + TCPROS	DDS (RTPS)
Languages	C++03, Python 2	C++14, Python 3 (3.5+)
Build system	roscpp → catkin	ament, colcon
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	roscpp (UART)	FreeRTPS, microRTPS, microROS
Node manager	ROS Master	No

State of ROS 2: Schedule

- 2018.07.02 - ROS 2 2nd release (Bouncy Bolson)
- 2017.12.08 - ROS 2 1st release (Ardent Apalone)
- 2017.09.13 - ROS 2 Beta3 release (code name R2B3)
- 2017.07.05 - ROS 2 Beta2 release (code name R2B2)
- 2016.12.19 - ROS 2 Beta1 release (code name Asphalt)
- 2016.10.04 - ROS 2 Alpha8 release (code name Hook.and.Loop)
- 2016.07.14 - ROS 2 Alpha7 release (code name Glue Gun)
- 2016.06.02 - ROS 2 Alpha6 release (code name Fastener)
- 2016.04.06 - ROS 2 Alpha5 release (code name Epoxy)
- 2016.02.17 - ROS 2 Alpha4 release (code name Duct tape)
- 2015.12.18 - ROS 2 Alpha3 release (code name Cement)
- 2015.11.03 - ROS 2 Alpha2 release (code name Baling wire)
- 2015.08.31 - ROS 2 Alpha1 release (code name Anchor)

Contents

I. ROS 2

II. ROS 2의 대표적인 3가지 기능

1. DDS (Data Distribution Service)
2. Real-time Computing
3. Embedded System

III. ROS 2의 개발 현황

IV. 미래의 로봇 운영체제가 갖추어야 할 것

미래의 로봇 운영체제가 갖추어야 할 것

- 기능
- 커뮤니티 파워
- 산업용 로봇 벤더들과의 협력
- 하드웨어를 잡아라
- 상업성 확보, 완전한 생태계 구축

미래의 로봇 운영체제가 갖추어야 할 것

● 기능

- 대동소이 [大同小異]
- 처음에는 각각의 특징이 있었으나 닮아가고 서로 보완하고 있다.
- 기능 갖춤은 필수이지 차별 점이 아니다.

● 커뮤니티 파워

● 산업용 로봇 벤더들과의 협력

● 하드웨어를 잡아라

● 상업성 확보, 완전한 생태계 구축

미래의 로봇 운영체제가 갖추어야 할 것

● 기능

- 대동소이 [大同小異]
- 처음에는 각각의 특징이 있었으나 닮아가고 서로 보완하고 있다.
- 기능 갖춤은 필수이지 차별 점이 아니다.

● 커뮤니티 파워

- 기능 보강, 빠른 피드백, 콘텐츠 생성, 보급, 문서화, 표준화 등에 큰 도움이 된다.
- 개발자/사용자 커뮤니티 없는 로봇 운영체제에는 미래란 있을 수 없다.

● 산업용 로봇 벤더들과의 협력

● 하드웨어를 잡아라

● 상업성 확보, 완전한 생태계 구축

미래의 로봇 운영체제가 갖추어야 할 것

● 기능

- 대동소이 [大同小異]
- 처음에는 각각의 특징이 있었으나 닮아가고 서로 보완하고 있다.
- 기능 갖춤은 필수이지 차별 점이 아니다.

● 커뮤니티 파워

- 기능 보강, 빠른 피드백, 콘텐츠 생성, 보급, 문서화, 표준화 등에 큰 도움이 된다.
- 개발자/사용자 커뮤니티 없는 로봇 운영체제에는 미래란 있을 수 없다.

● 산업용 로봇 벤더들과의 협력

- 산업용 로봇의 벤더들은 보유 기술은 공개할 수 없지만 산학협력은 해야 한다.
- 오월동주 [吳越同舟] (ROS-Industrial)

● 하드웨어를 잡아라

● 상업성 확보, 완전한 생태계 구축

미래의 로봇 운영체제가 갖추어야 할 것

● 기능

- 대동소이 [大同小異]
- 처음에는 각각의 특징이 있었으나 닮아가고 서로 보완하고 있다.
- 기능 갖춤은 필수이지 차별 점이 아니다.

● 커뮤니티 파워

- 기능 보강, 빠른 피드백, 콘텐츠 생성, 보급, 문서화, 표준화 등에 큰 도움이 된다.
- 개발자/사용자 커뮤니티 없는 로봇 운영체제에는 미래란 있을 수 없다.

● 산업용 로봇 벤더들과의 협력

- 산업용 로봇의 벤더들은 보유 기술은 공개할 수 없지만 산학협력은 해야 한다.
- 오월동주 [吳越同舟] (ROS-Industrial)

● 하드웨어를 잡아라

- 하드웨어 추상화 플랫폼 실현, 표준/규격 제안, 모듈화 하드웨어 유도

● 상업성 확보, 완전한 생태계 구축

미래의 로봇 운영체제가 갖추어야 할 것

● 기능

- 대동소이 [大同小異]
- 처음에는 각각의 특징이 있었으나 닮아가고 서로 보완하고 있다.
- 기능 갖춤은 필수이지 차별 점이 아니다.

● 커뮤니티 파워

- 기능 보강, 빠른 피드백, 콘텐츠 생성, 보급, 문서화, 표준화 등에 큰 도움이 된다.
- 개발자/사용자 커뮤니티 없는 로봇 운영체제에는 미래란 있을 수 없다.

● 산업용 로봇 벤더들과의 협력

- 산업용 로봇의 벤더들은 보유 기술은 공개할 수 없지만 산학협력은 해야 한다.
- 오월동주 [吳越同舟] (ROS-Industrial)

● 하드웨어를 잡아라

- 하드웨어 추상화 플랫폼 실현, 표준/규격 제안, 모듈화 하드웨어 유도

보이지 않는 생태계 속의 분업

하드웨어 모듈 + 운영체제 + 앱 + 유저

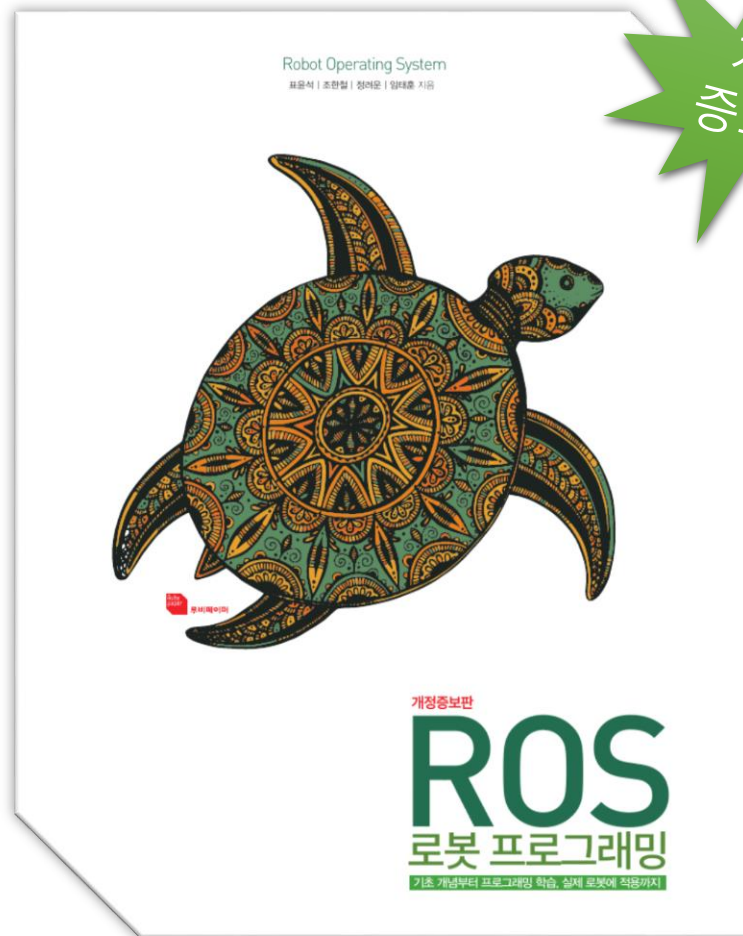
● 상업성 확보, 완전한 생태계 구축

- ROS는 아카데미적인 면에서 영향력을 발휘했지만 수익 면에서는 NAOqi 처럼 개발자의 수익도 염두 해둔 APP시장을 키워야 더욱 풍부한 응용 애플리케이션 개발을 이끌 수 있다.
- 생태계 (ECO system)

질문 대환영!

* 온라인 상의 질문이라면
오로카 및 로열모를 이용해주세요!

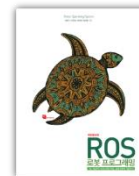
여기서! 광고 하나 나가요~



개정
증보판

✓ 한국어판 구매 링크

✓ 4개 언어로 출판!



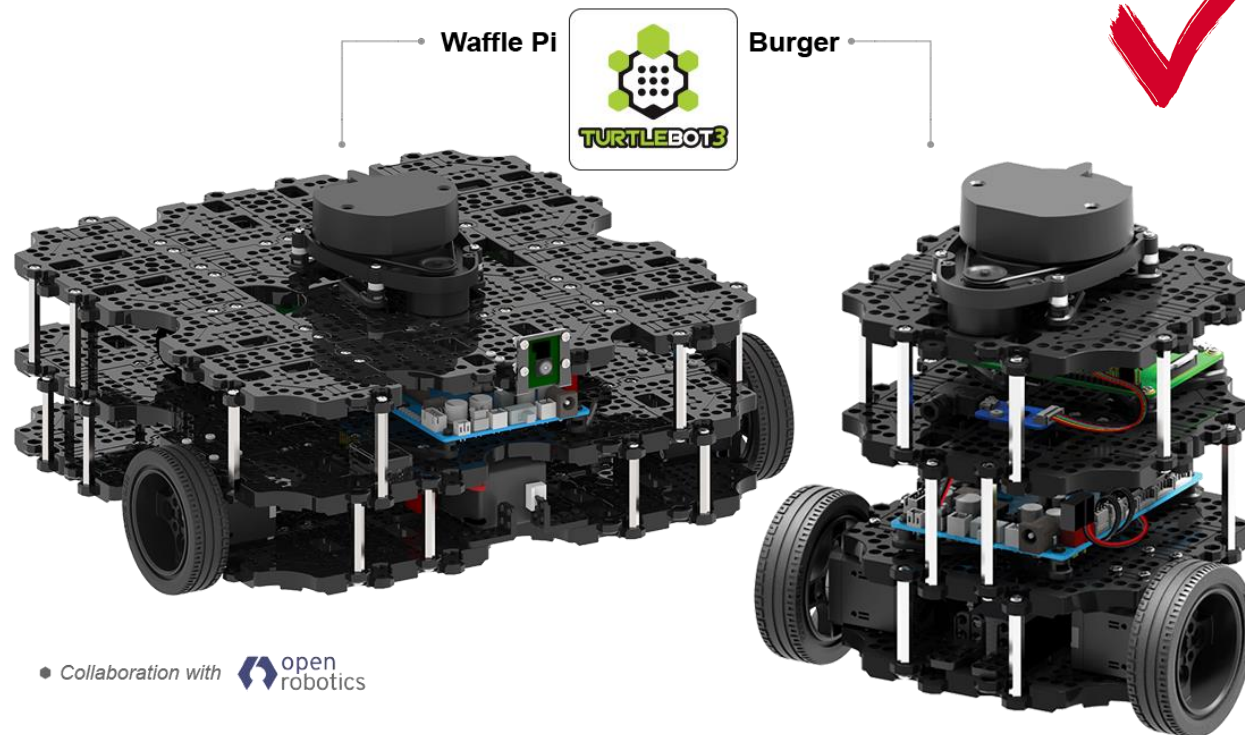
국내 유일! 최초! ROS 참고서!
ROS 공식 플랫폼 **TurtleBot3** 개발팀이
직접 저술한 바이블급 ROS 책

여기서! 광고 둘 나가요~

TURTLEBOT3

인공지능(AI) 연구의 시작,
ROS 교육용 공식 로봇 플랫폼

터틀봇3는 ROS기반의 저가형 모바일 로봇으로
교육, 연구, 제품개발, 취미 등 다양한 분야에서
활용할 수 있습니다.



✓ [Direct Link](#)

여기서! 광고 셋 나가요~



- 오로카
- www.oroqa.org
- 오픈 로보틱스 지향
- 풀뿌리 로봇공학의 저변 활성화
- 공개 강좌, 세미나, 프로젝트 진행

- 로봇공학을 위한 열린 모임 (KOS-ROBOT)
- www.facebook.com/groups/KoreanRobotics
- 로봇공학 통합 커뮤니티 지향
- 일반인과 전문가가 어울러지는 한마당
- 로봇공학 정보 공유
- 연구자 간의 협력

- RobotSource
- www.robotsource.org
- 글로벌 로보틱스 커뮤니티 지향
- 로봇공학 정보 공유
- 자신의 로봇 프로젝트 공유
- DIY 로봇 프로젝트 진행

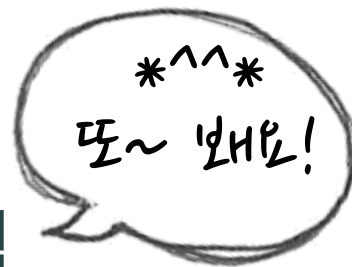
혼자 하기엔 답답하시다고요?

커뮤니티에서 함께 해요~

끝.

표윤석

Yoonseok Pyo
pyo@robotis.com
www.robotpilot.net



www.facebook.com/yoonseok.pyo