ROS2 강습회

대한기계학회 IT융합부문
명지대학교 최동일
dongilc@mju.ac.kr

2023. 08. 11

1교시

ROS2 기본 개념 및 소개

ROS2 강습회 : 세부 강의 일정

1교시 (10시~11시)	ROS2 기본 개념 및 소개
	교육의 기본 컨셉 및 의도
	교육의 커버리지
	ROS2의 기본 개념 소개
	ROS기반 개발 사래

2교시 (11시~12시)	리눅스 운영체제 기본 명령어 사용법 실습
	vim3 SBC 소개
	vim3 접속 (PC-VS Code & ssh 프로그램 설치)
	리눅스 운영체제 기본 사용법 강의

점심식사 (12시~1시)

ROS2 강습회 : 세부 강의 일정

3교시 (1시~2시)	파이썬으로 ROS2 토픽 다루기 실습
	Jupyter notebook으로 rclpy 다루기
	Jupyter notebook와 vim3의 ros2 연동
	Jupyter notebook으로 topic 주고 받기
	조원들과 함께 topic 주고 받기

4교시 (2시~3시)	Github와 오픈소스 프로그램 활용 실습
	vim3 이용 github 에서 vim3용 host 프로그램 clone
	host 프로그램 coding using VS Code
	PC에서 로봇제어용 프로그램 github에서 clone
	clone 한 프로그램 동작시켜 로봇 기본 동작확인

ROS2 강습회 : 세부 강의 일정

5교시 (3시~4시)	임베디드 시스템에서의 ROS2 프로그래밍 실습
	상위제어기, 하위제어기, 구조 설명
	Tetrix Arduino 프로그램 굽기
	Tetrix 로봇의 기본 동작 확인
	Tetrix 로봇용 제어 프로그램 github에 제공
	프로그램을 로봇에 적용시켜 지도상에서 동작시키기 실습

ROS2 강습회 : 강사 소개

DRCL (Dynamic Robot Control Lab)		
	이름	최동일(Dongil Choi)
	소속/직위	명지대 기계공학과 교수
	학과/연구실	기계공학과 / 다이나믹로봇제어연구실
	오피스	1공학관 Y221
	연구실	1공학관 Y121, 산학협력관 1층
연락처	사무실	031-324-1427
	핸드폰	010-2731-7234
	이메일	dongilc@mju.ac.kr
박사학위 취득		KAIST 기계공학(2012. 02)

경력사항	
2012.02 ~ 2012.08	Post-doc, Hubo Lab
2012.09 ~ 2013.12	Post-doctoral Fellow, The Robotics Institute(RI), Carnegie Mellon University(CMU)
2014.01 ~ 2016.01	선임 연구원, 한국 항공 우주 연구원(KARI)
2016.01 ~ 2018.08	책임 연구원, Naver Labs
2018.09 ~	명지대학교 기계공학과 부교수

ROS란?

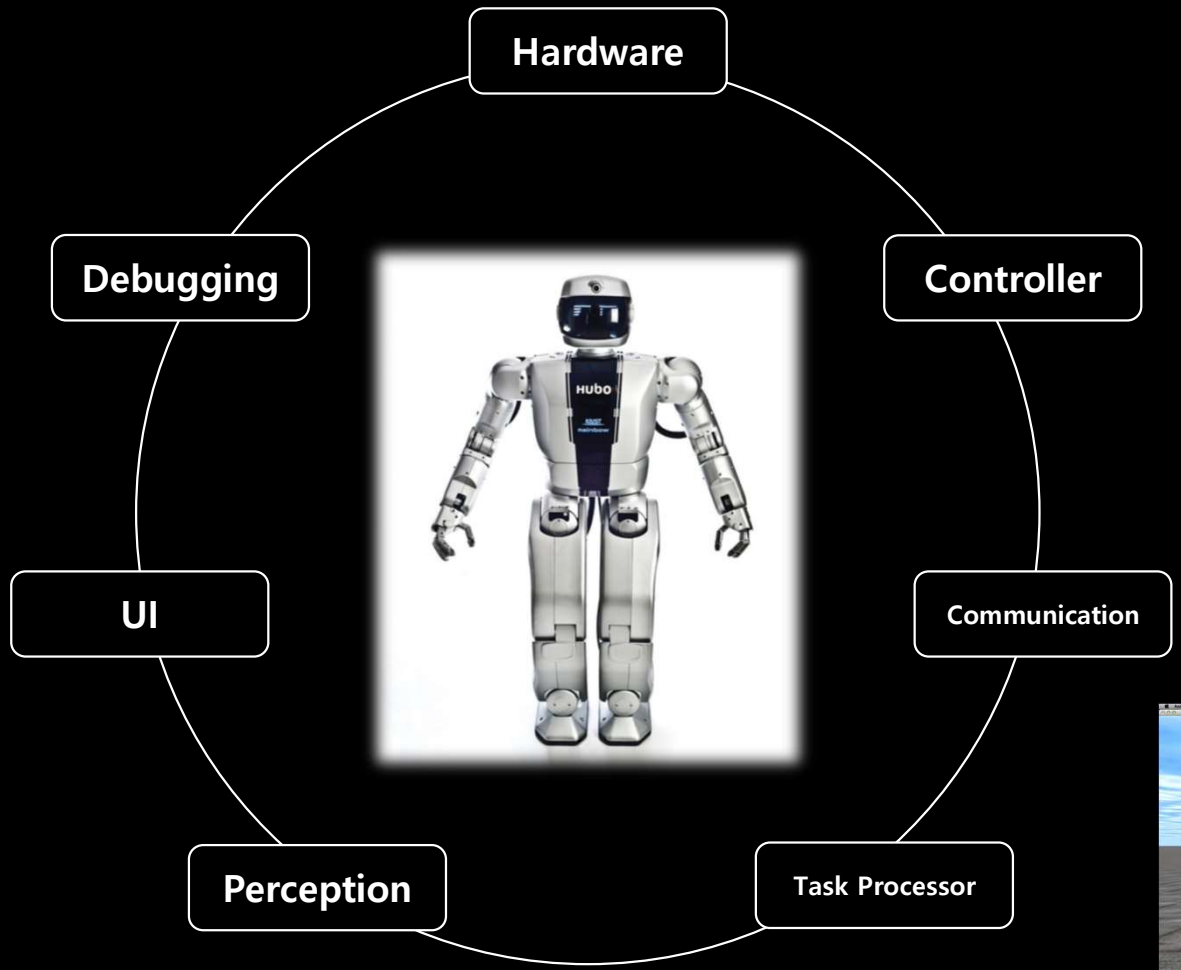
기존에 존재하는 다양한 로봇 'open-source'들을 사용 가능

로봇 개발에 필요한 여러 툴 제공

- **Simulation** – Ignition Gazebo, ISSAC ros, webots
- **Embedded** – rosserial, micro-ROS
- **Visualization, Debug Tools** – RViz, RQt

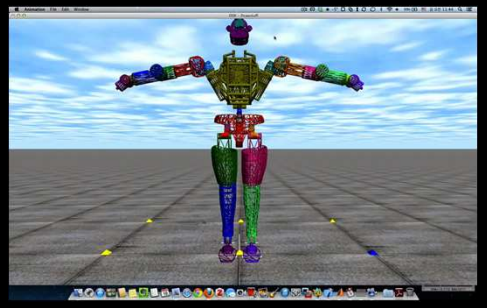
지속적으로 발전하고 유지보수될 수 있는 'Robot OS'

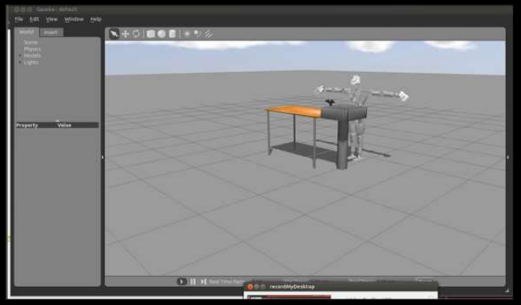
Why ROS



ROS

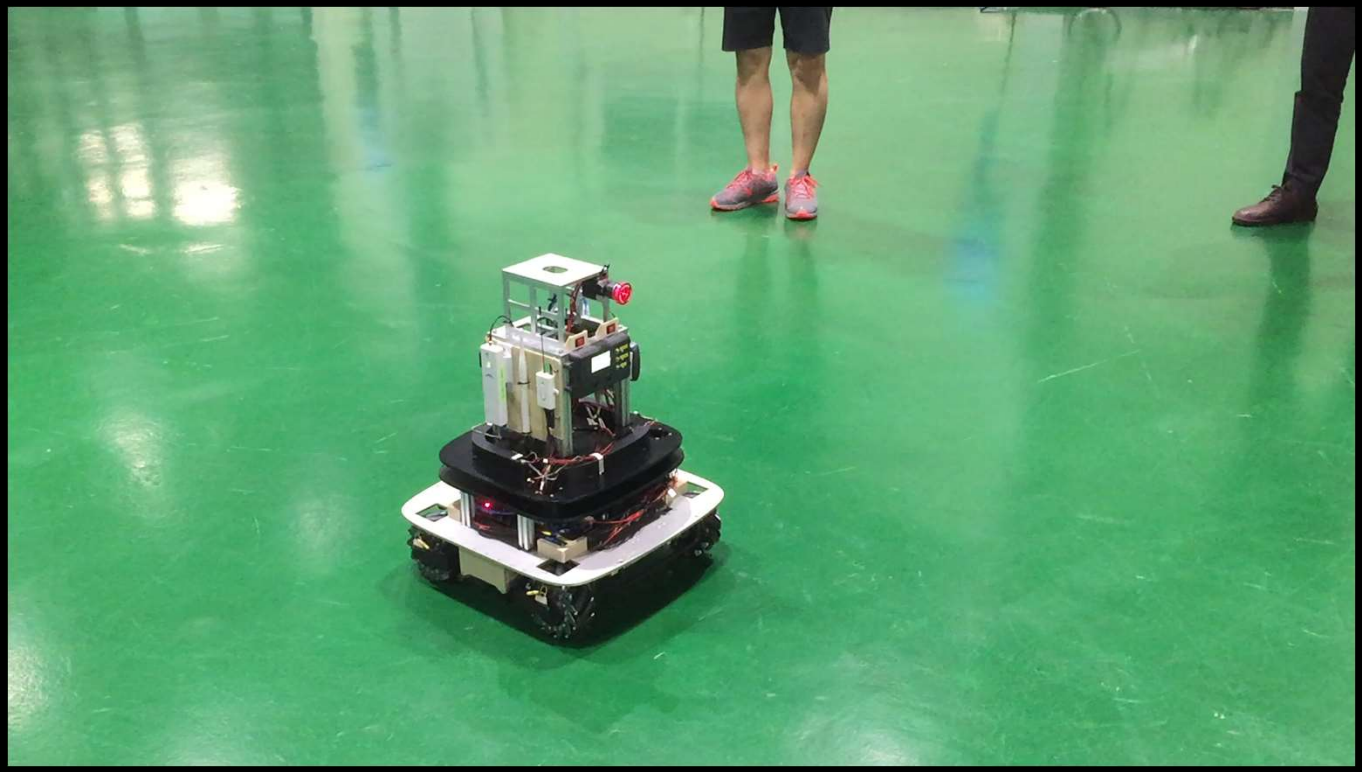
(Robot Operating system)



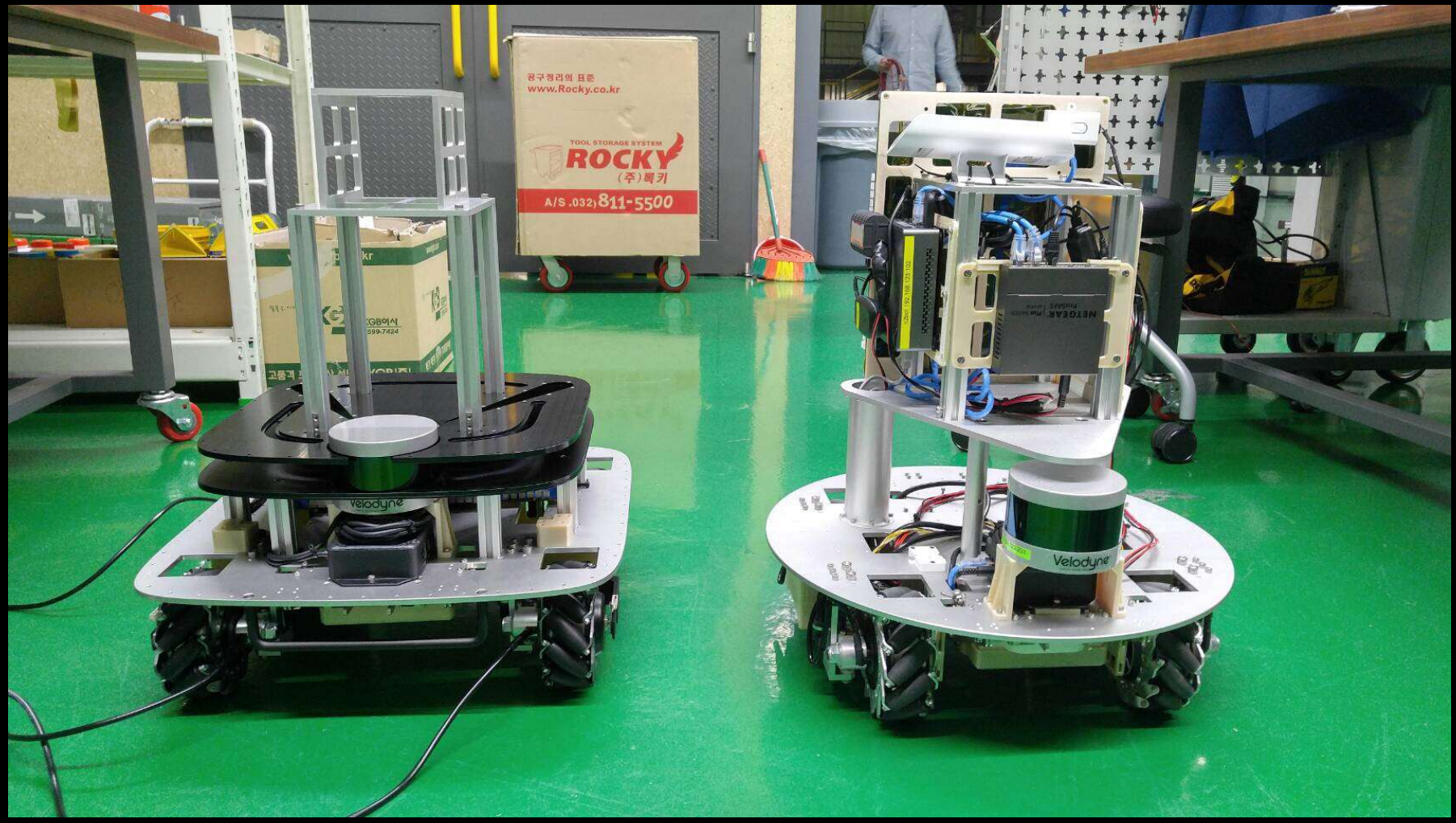


Why ROS

M1, 2016, NaverLabs



Why ROS


www.Rocky.co.kr
TOOL STORAGE SYSTEM
ROCKY
(주)록키
A/S .032) 811-5500
Velodyne
Velodyne

ROS 등장배경

[1983년 최초 상용 핸드폰(?) 모토로라 DynaTAC 8000 와 개발자 Martin Cooper, 점점 발전하는 휴대전화]

휴대폰 OS
- Android
- iOS

→ Robot OS의 필요성 대두

ROS 등장배경

ROS

Open source

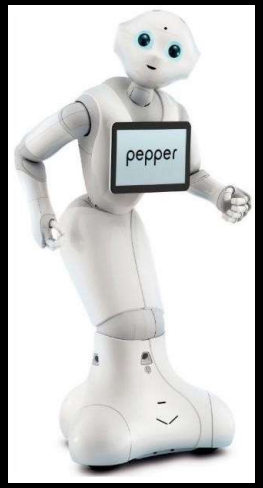
OPRoS

RT MIDDLEWARE

Galapagos

Closed source

NAOqi



fetch robotics

CLEARPATH ROBOTICS

rethink robotics

CRUISE

GaiTech International

hiDOF

Robotnik

SCHUNK

Shadow

SwRI

TOYOTA

YASKAWA

DARPA

QUALCOMM

intel

BOSCH Invented for life

Fraunhofer IPA

YUJIN ROBOT

Google

SoftBank

Sprint

ALDEBARAN Robotics

Alibaba.com

FOXCONN Advancing Through Innovation

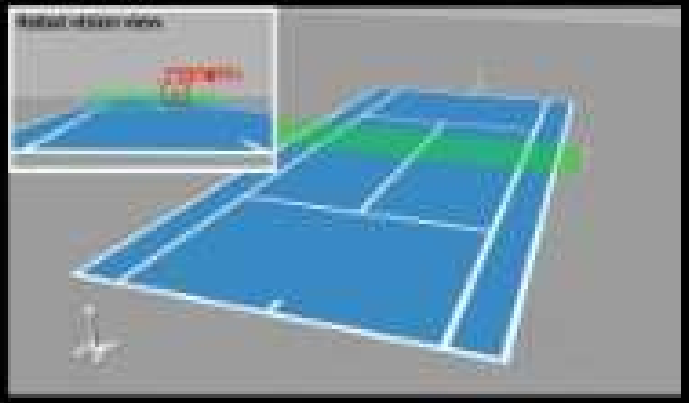
ROS1 특징

- **Platforms** – Linux
- **Real-time** – external frameworks like OROCOS
- **Security** – SROS (security enhancements for ROS)
- **Communication** – XMLRPC + TCPROS
- **Language** - Python2
- **Node master** - ROS Master

ROS1 을 이용한 개발사례

Task-space position control of
Mecanum mobile manipulator

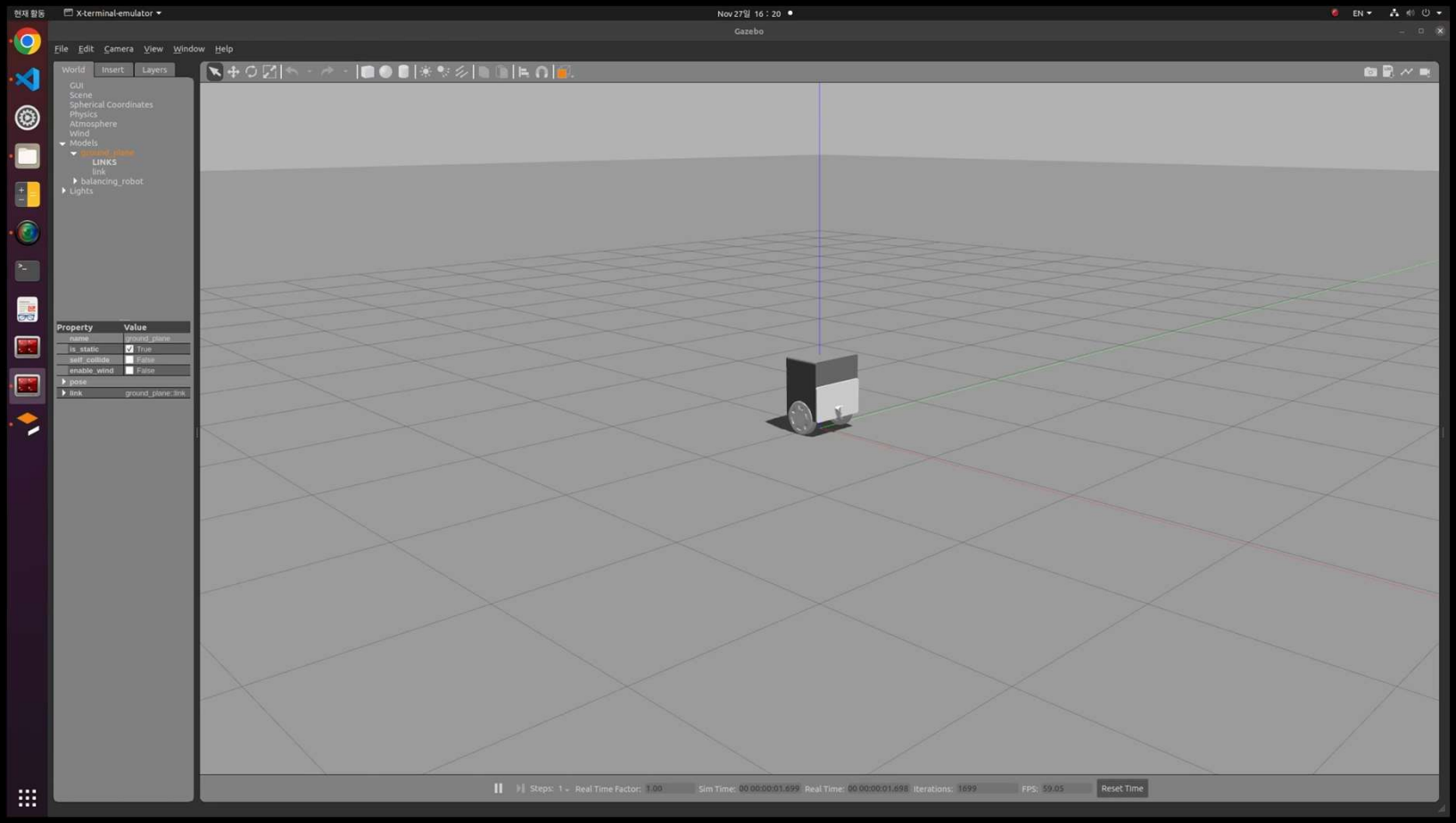
Gazebo Simulation of Autonomous Delivery Robot
using MPC for High-speed Mobility



Ball tracking and trajectory prediction
system for tennis robots

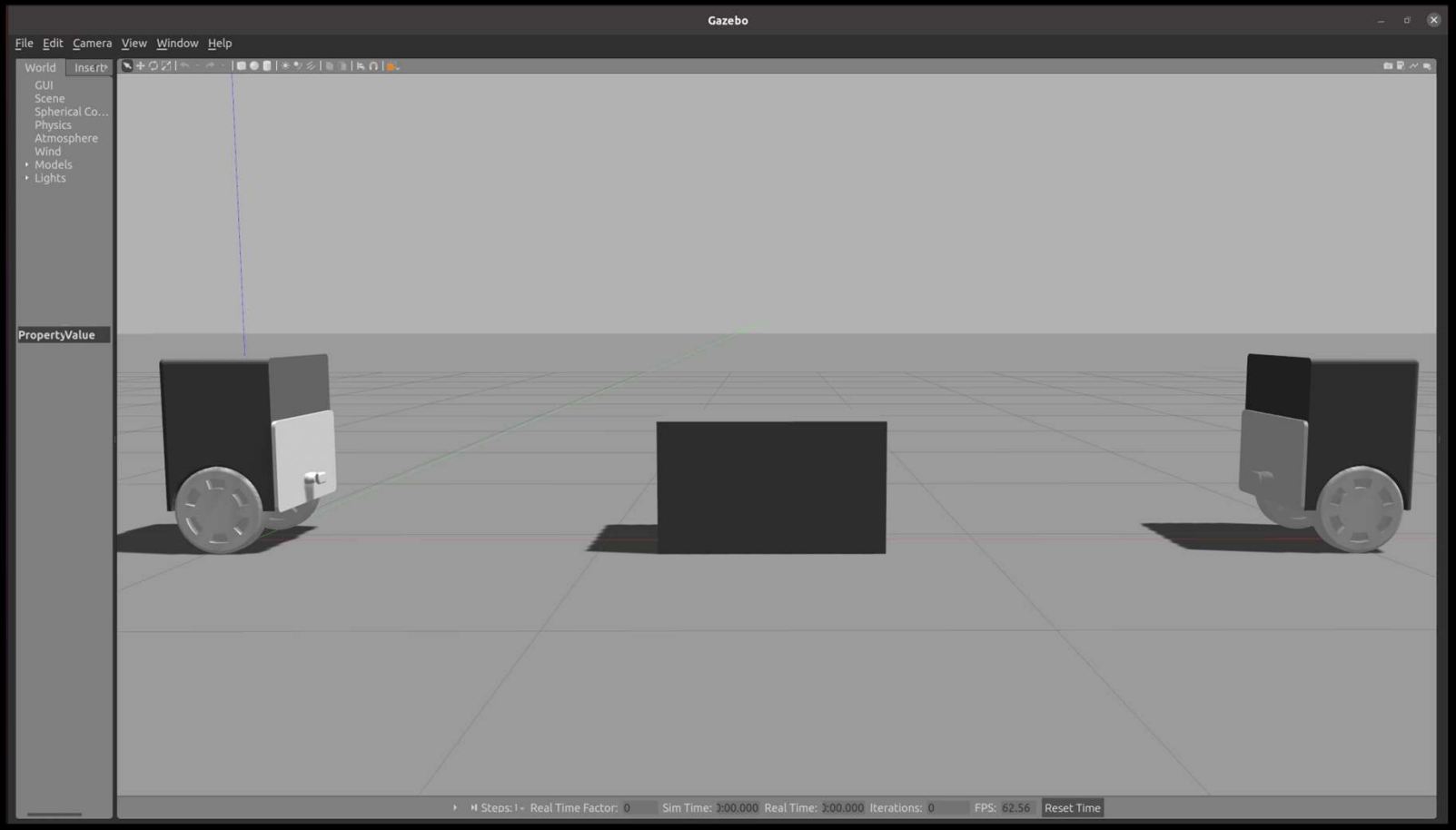
Development and Decoupled Optimal
Control of CMG Unicycle-Legged Robot

ROS1 을 이용한 개발사례



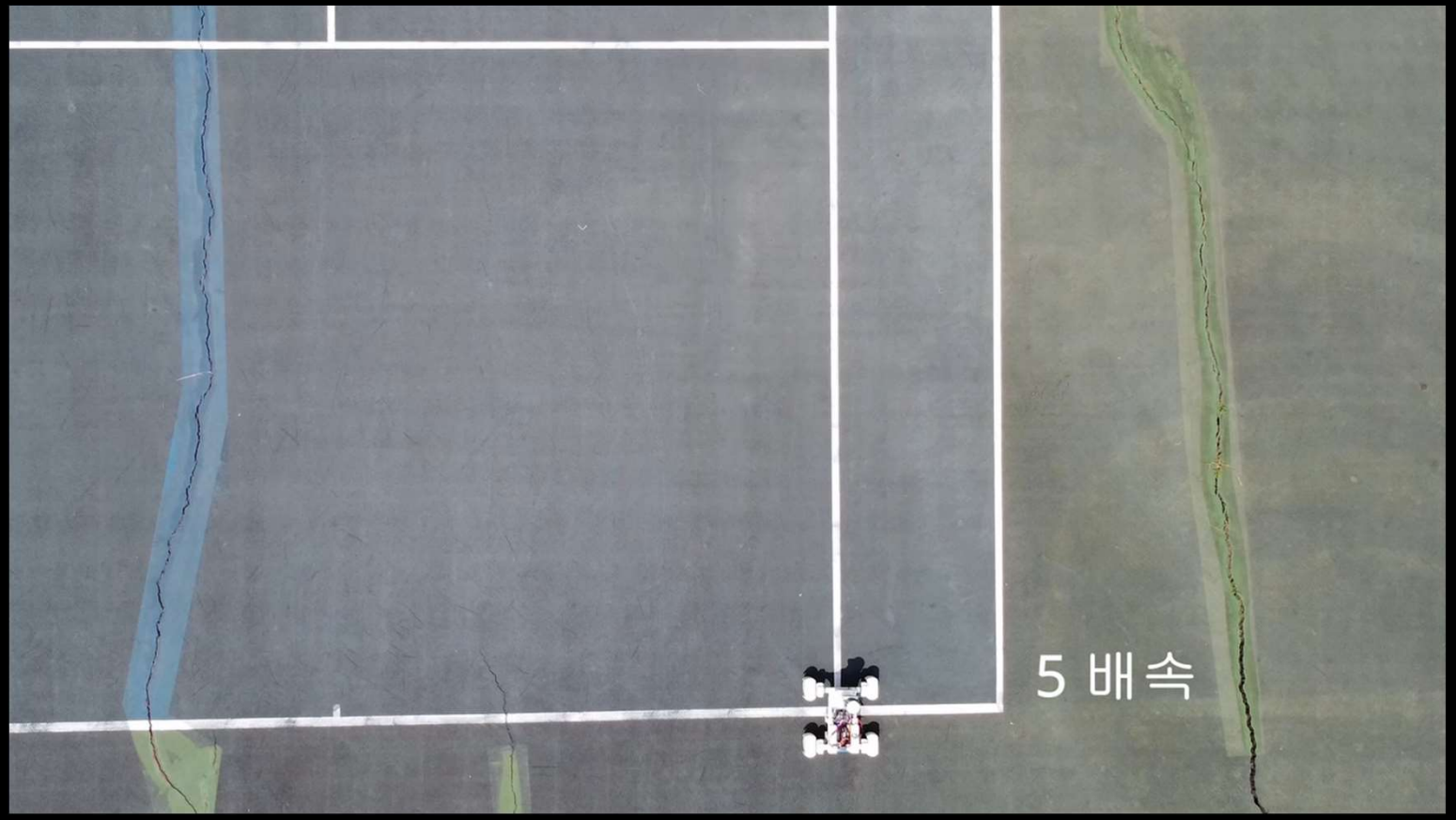
분리합체 및 주행제어 시뮬레이션

ROS1 을 이용한 개발사례



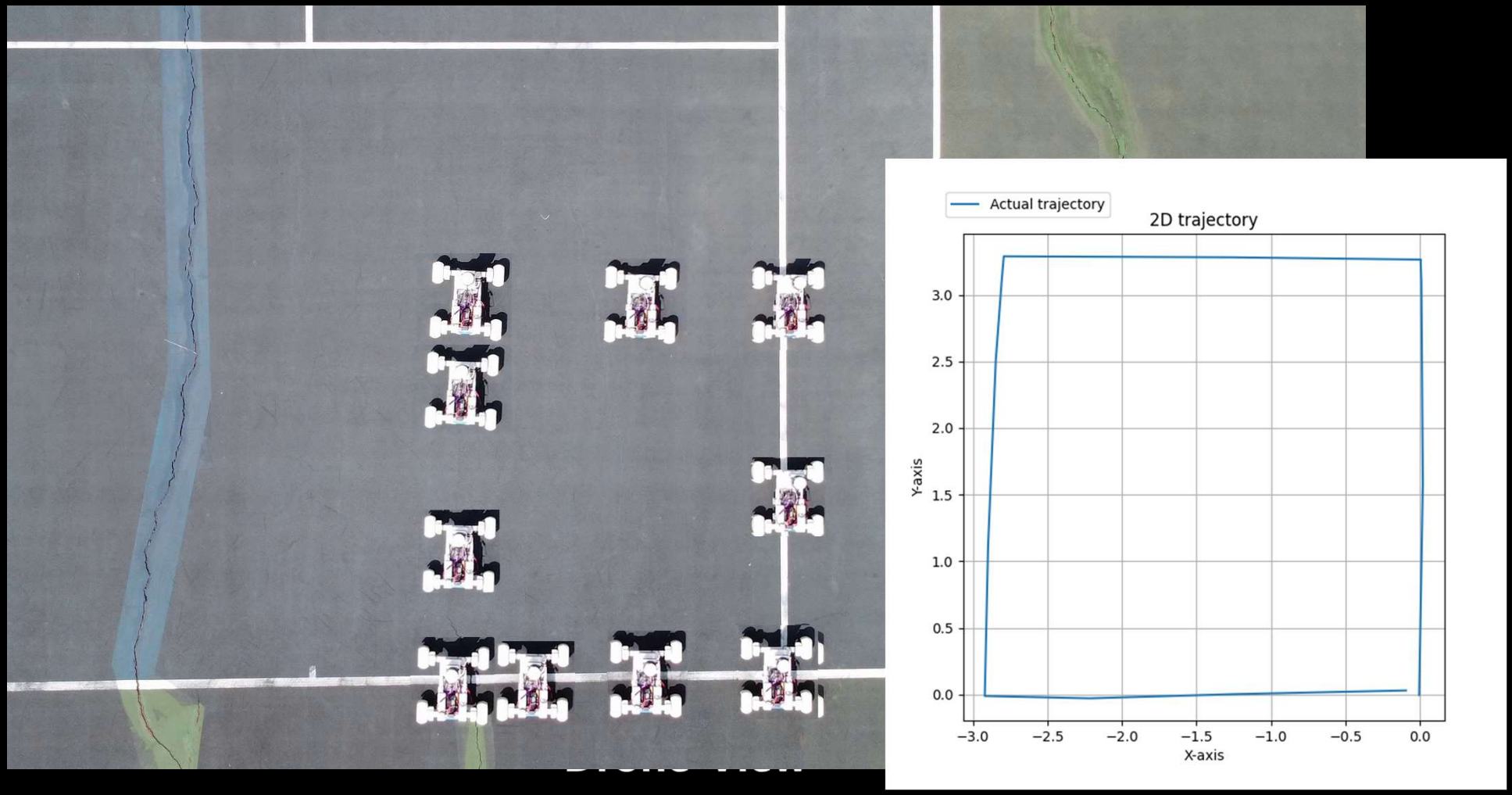
분리합체 및 주행제어 시뮬레이션

ROS1 을 이용한 개발사례



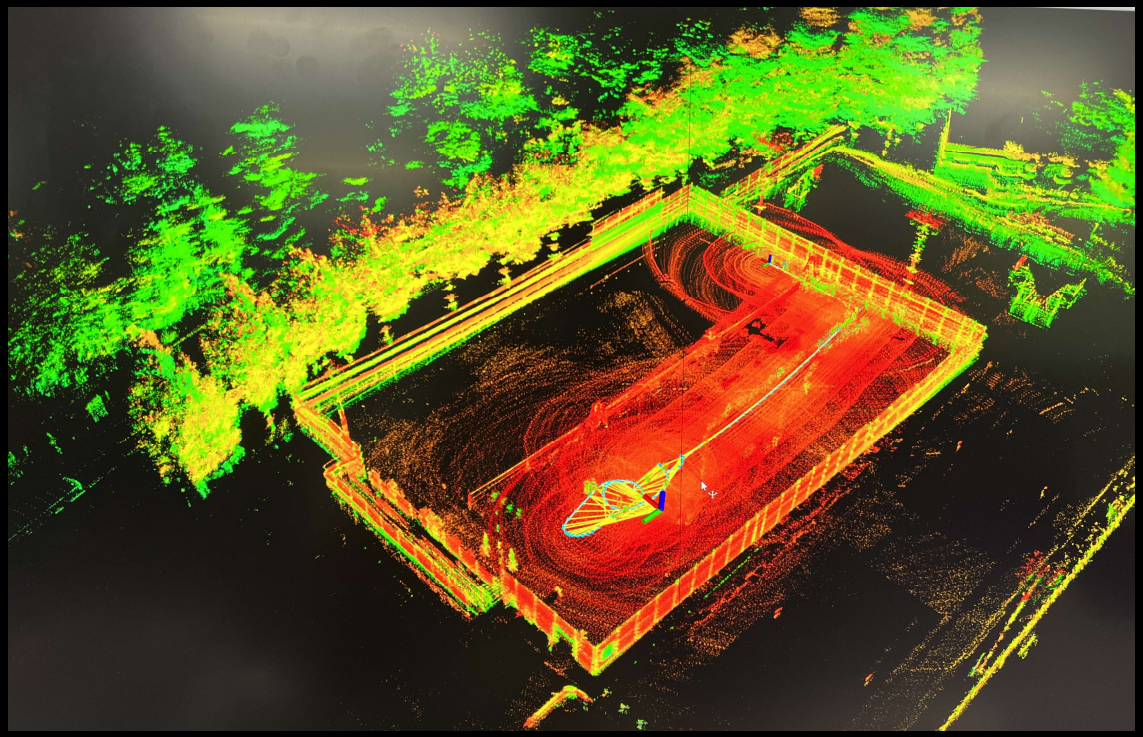
Drone View

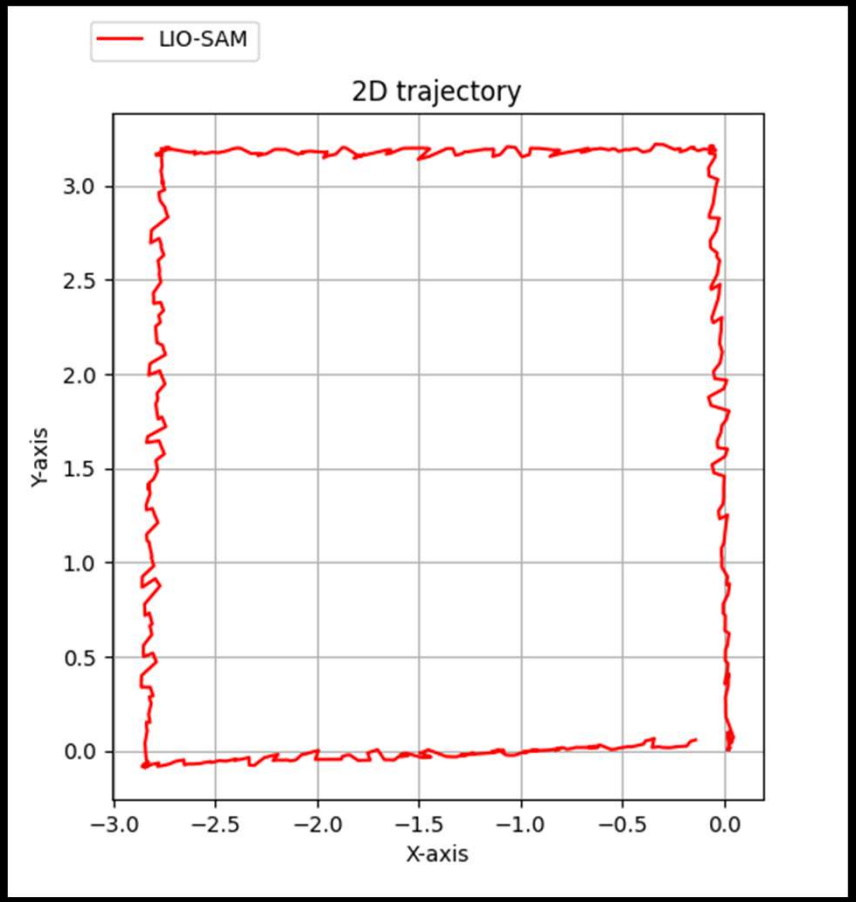
ROS1 을 이용한 개발사례



ROS1 을 이용한 개발사례

LIO SAM





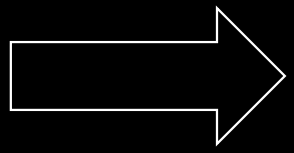
ROS2의 등장

ROS1은 실제 상용화에는 부족하다는 문제점들이 제기되었고, 개발자들은 기존 버전의 업데이트보다 새로운 ROS를 만들기로 결정하였다.

RTOS 사용불가

TCPROS사용으로 실시간성 저해

Python2 사용



ROS2

ROS2 특징

1. Data Distribution Service (DDS)
- communication pipeline interface, can have security configurations.

2. Nodes
- Executed code utilizing ROS, can have configurable parameters.

3. Publishers & Subscribers
- Nodes can publish data for subscribers to process.

4. Services
- Nodes request data, and other nodes send a single response.

5. Actions
- Node sets a goal, and receives feedback until result is given.

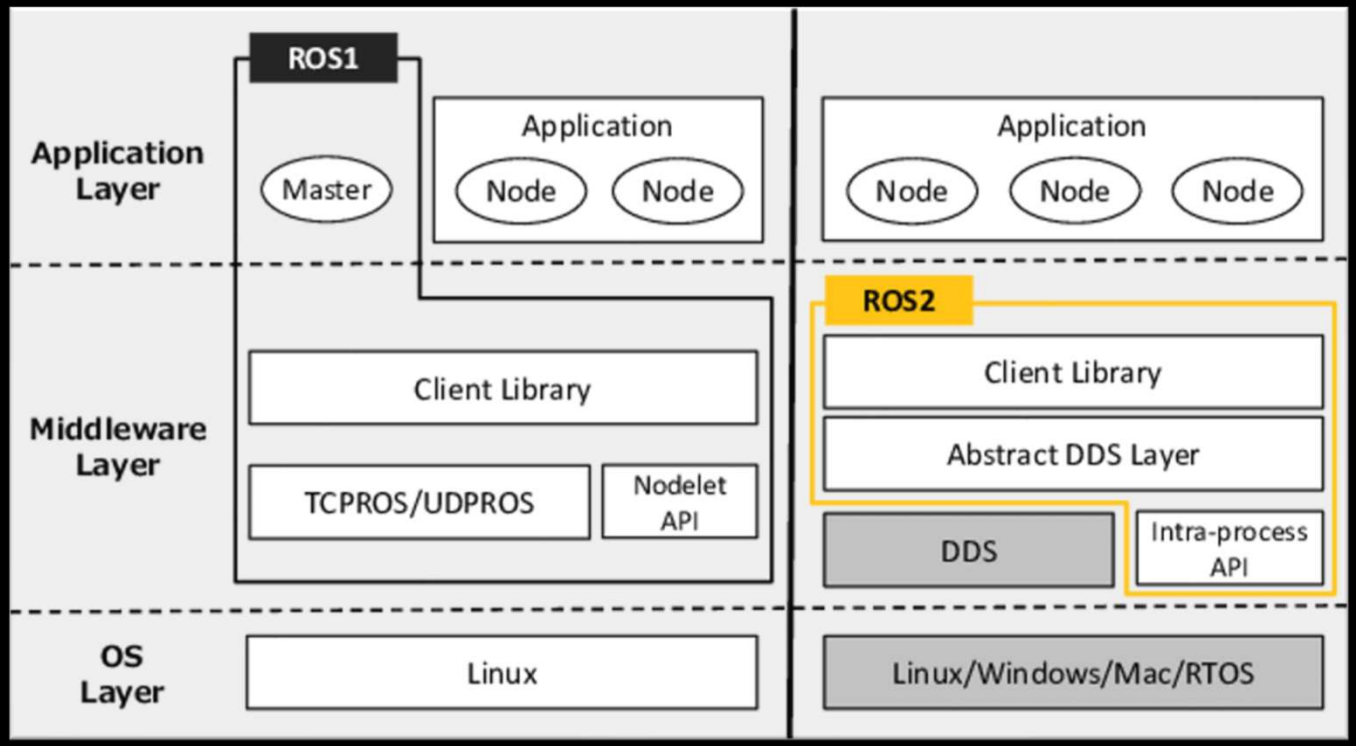
6. Bag Files
- Save and playback data being published.

7. Packages
- Distribution method of ROS code.

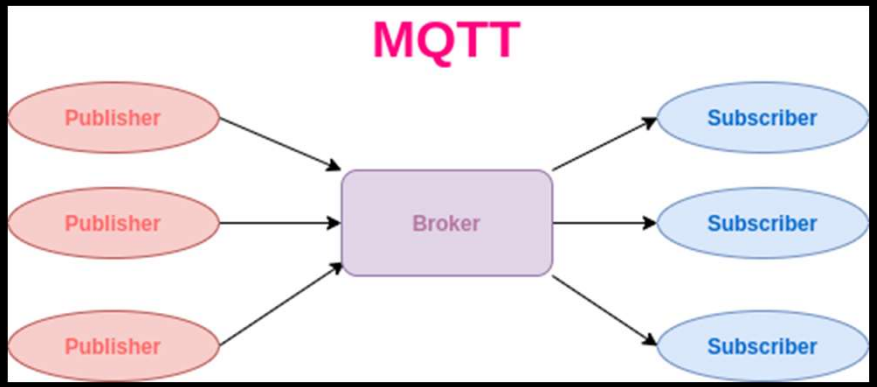
8. Cross Platform/ Version Support
- Works across most operating systems, and can interface with ROS1 systems.

ROS1 vs ROS2

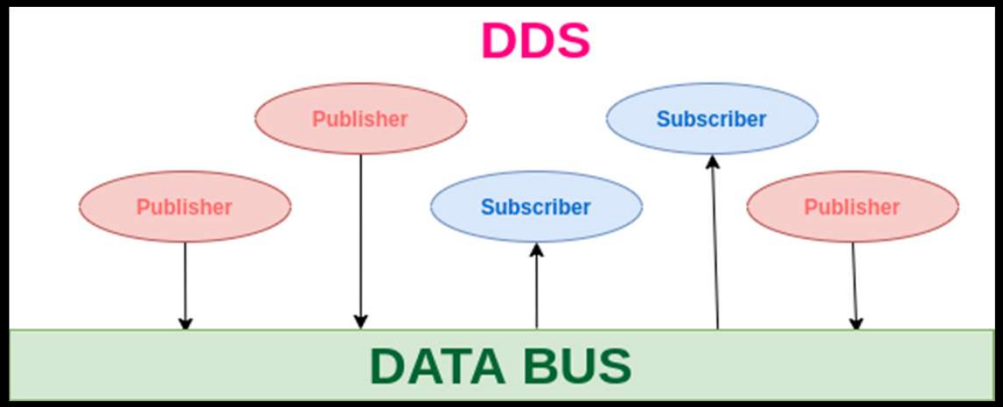
- ROS1은 Master Node가 필요하지만 ROS2는 필요하지 않다.
- ROS1은 Linux에서만, ROS2는 Linux/Window/Mac/RTOS.
- ROS1은 Python2를 사용하지만, ROS2는 Python3.



ROS1 vs ROS2



ROS1 통신 방식



ROS2 통신 방식

- ROS1에서 사용했던 방식인 MQTT는 TCP 기반으로 브로커를 통해 메시지를 전달
- ROS2의 DDS는 UDP 기반으로 중간 매개체가 없음
- UDP 기반의 통신이기때문에 기본적으로는 신뢰성이 없지만 QoS(Quality of Service)를 설정하여 신뢰성을 확보

DDS란?

- DDS는 분산 객체에 대한 기술 표준을 제정하기 위해 1989년에 설립된 비영리 단체인 OMG(Object Management Group) 관리 하에 산업 표준

- 리눅스, 윈도우, macOS, 안드로이드, VxWorks 등 다양한 운영체제를 지원

- DDS는 미들웨어이기에 그 상위 레벨이라고 볼 수 있는 사용자 코드 레벨에서 DDS를 사용하기 위해 기존에 시용하던 프로그래밍 언어를 변경할 필요가 없음

- ROS2에서도 이 특징을 충분히 살려 다음 그림과 같이 DDS를 RMW(ROS middleware)로 추상화하였으며 벤더 별로 RMW를 지원

- UDP 기반의 신뢰성 있는 멀티캐스트(Reliable multicast)를 구현하여 시스템이 최신 네트워킹 인프라의 이점을 효율적으로 활용

DDS란?

- UDP의 멀티캐스트는 브로드캐스트처럼 여러 목적지로 동시에 데이터를 보낼 수 있지만, 불특정 목적지가 아닌 특정된 도메인 그룹에 대해서만 데이터를 전송

- ROS1에서는 노드 사이의 연결을 위해 네임 서비스를 마스터에서 실행했었어야 했고, 이 ROS Master가 연결이 끊기거나 죽는 경우 모든 시스템이 마비되는 단점이 있었음

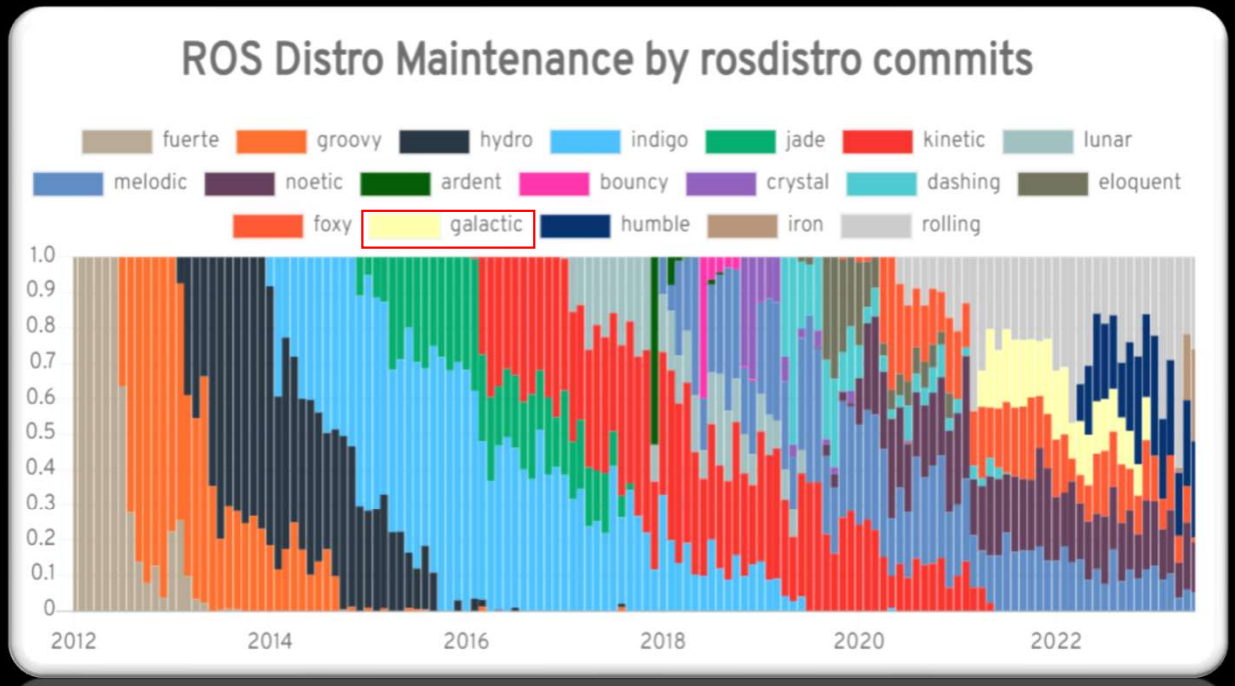
- DDS는 동적 검색(Dynamic Discovery)을 통하여 어떤 토픽이 지정 도메인 영역에 있으며 어떤 노드가 이를 발신하고 수신하는지 알 수 있음

- ROS2에서는 ROS Master가 없어지고 DDS의 동적 검색 기능을 시용함에 따라 노드를 DDS의 Participant로 취급하게 되었으며, 동적 검색 기능을 이용하여 이를 연결

DDS란?

- QoS (Quality of Services) in ROS2
  - Reliability
    - Reliable: TCP처럼 데이터 손실을 방지함으로써 신뢰도를 우선시하여 사용
    - Best effort: UDP처럼 통신 속도를 최우선시하여 사용
  - History: 통신 상태에 따라 정해진 크기만큼의 데이터를 보관
  - Durability: 데이터를 수신하는 서브스크라이버가 생성되기 전의 데이터를 사용할지 폐기할지에 대한 설정
  - Deadline: 정해진 주기 안에 데이터가 발신 및 수신되지 않을 경우 이벤트 함수를 실행
  - Lifespan: 정해진 주기 안에서 수신되는 데이터만 유효 판정하고 그렇지 않은 데이터는 삭제
  - Liveliness: 정해진 주기 안에서 노드 혹은 토픽의 생사를 확인
  - ROS2에서는 DDS-Security라는 DDS 보안 사양을 ROS2에 적용하여 보안에 대한 이슈를 통신단부터 해결

ROS2



galactic

위 그래프를 통해 알 수 있듯이 현재 ROS2를 많이 사용하고 있는 추세이고, **'galactic'**을 사용하여 실습할 예정입니다.

ROS2 기본적인 명령어

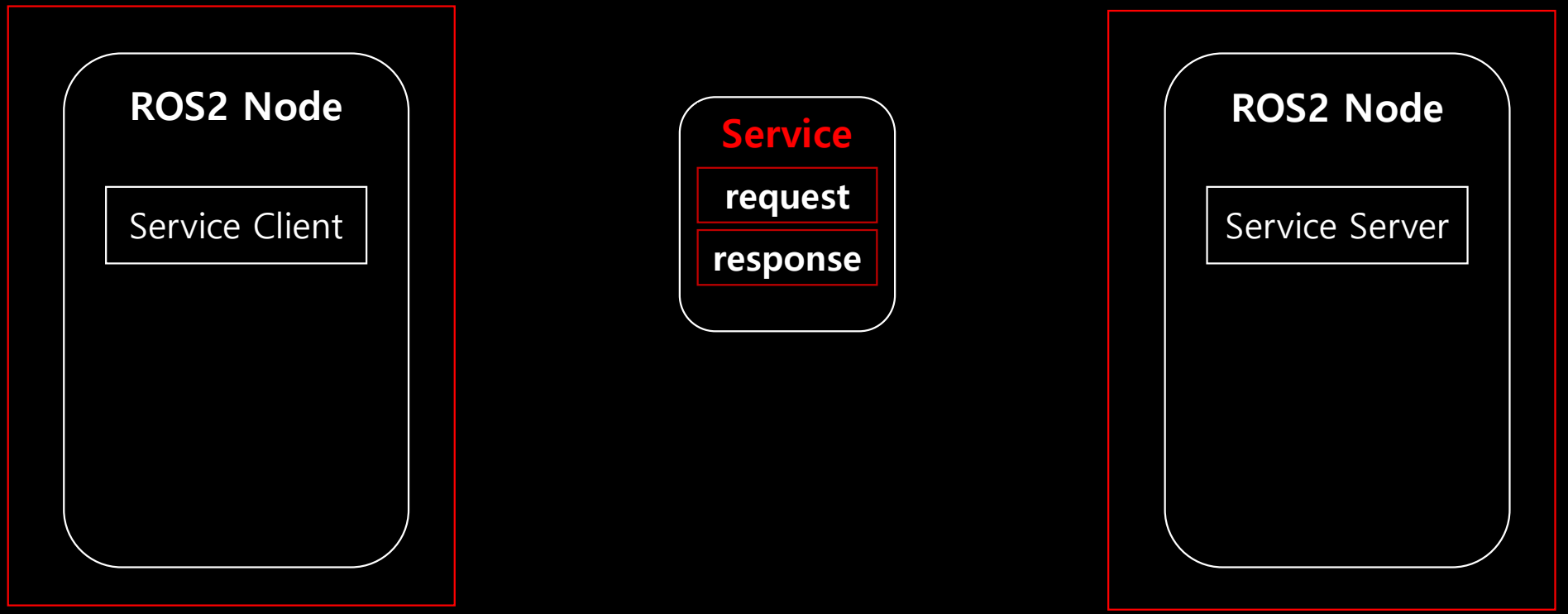
Service ⇨ 동기식 양방향 메시지 송수신 방식

Topic ⇨ 비동기식 단방향 메시지 송수신 방식

Action ⇨ 비동기식 양방향 메시지 송수신 방식

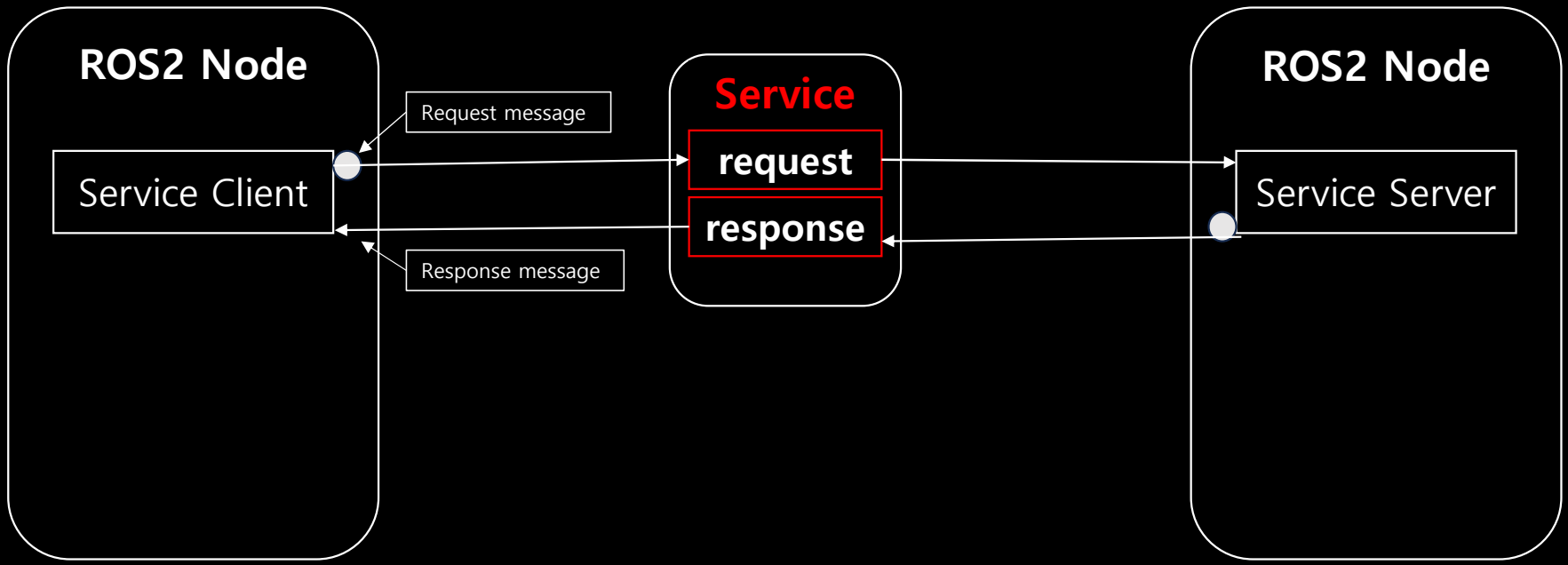
ROS2 기본적인 명령어 Node

- 최소 단위의 실행 가능한 프로세스



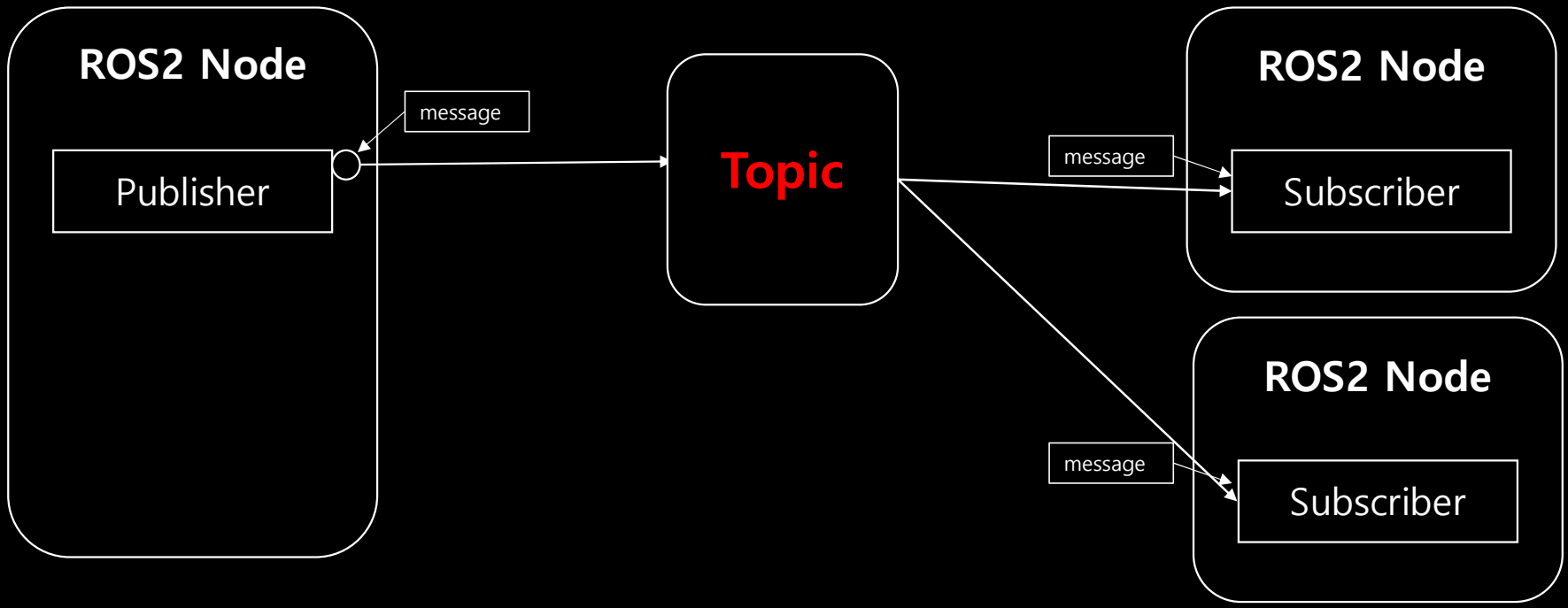
ROS2 기본적인 명령어 Service

- 두 노드(node)가 데이터를 주고받는 방식 중에 클라이언트(client)가 서버에게 요청(request)하면 응답(response)을 받을 수 있는 방식.
- 이때 입력 혹은 출력 데이터는 있을 수도 있고 없을 수도 있다.



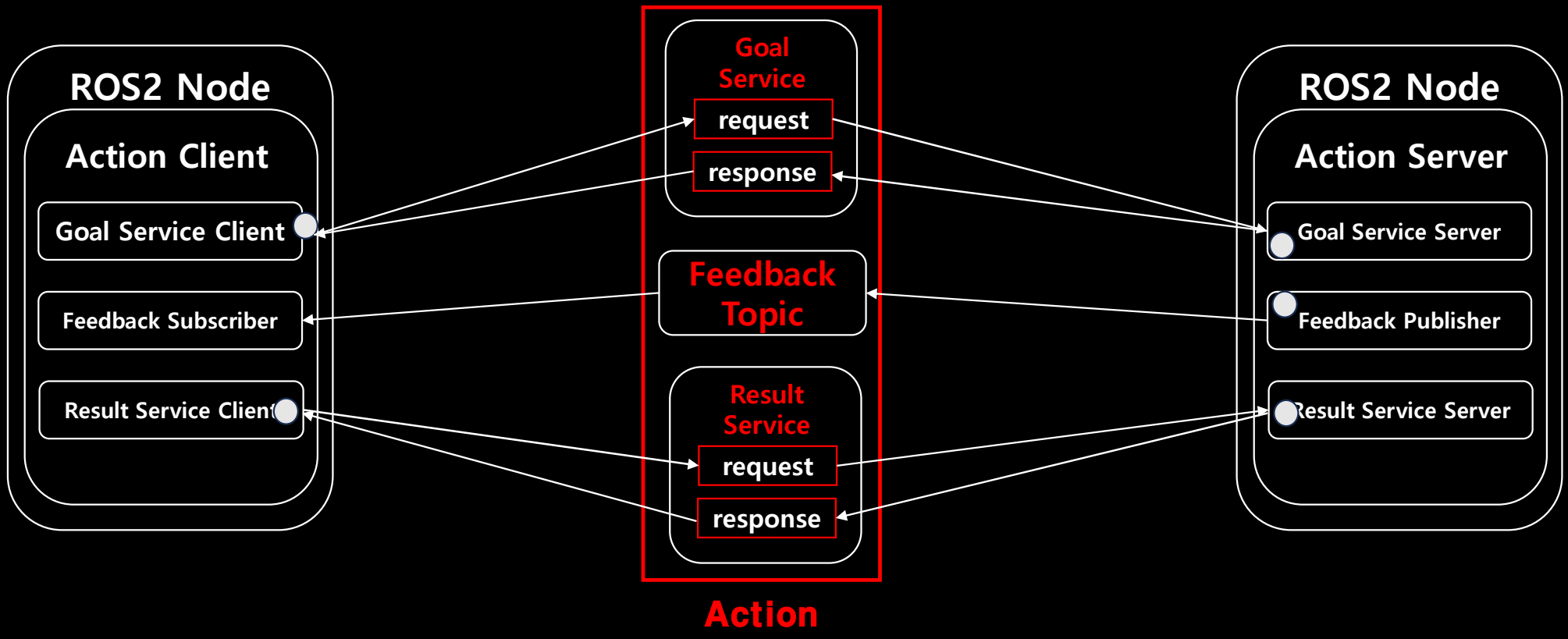
ROS2 기본적인 명령어 Topic

- 토픽(Topic)은 서비스(Service)와는 다르게 발행(publish)과 구독(subscribe)으로 되어있는 개념
- 토픽은 1:1 통신을 기본으로 하지만, 1:N, N:1, N:N 통신도 가능.

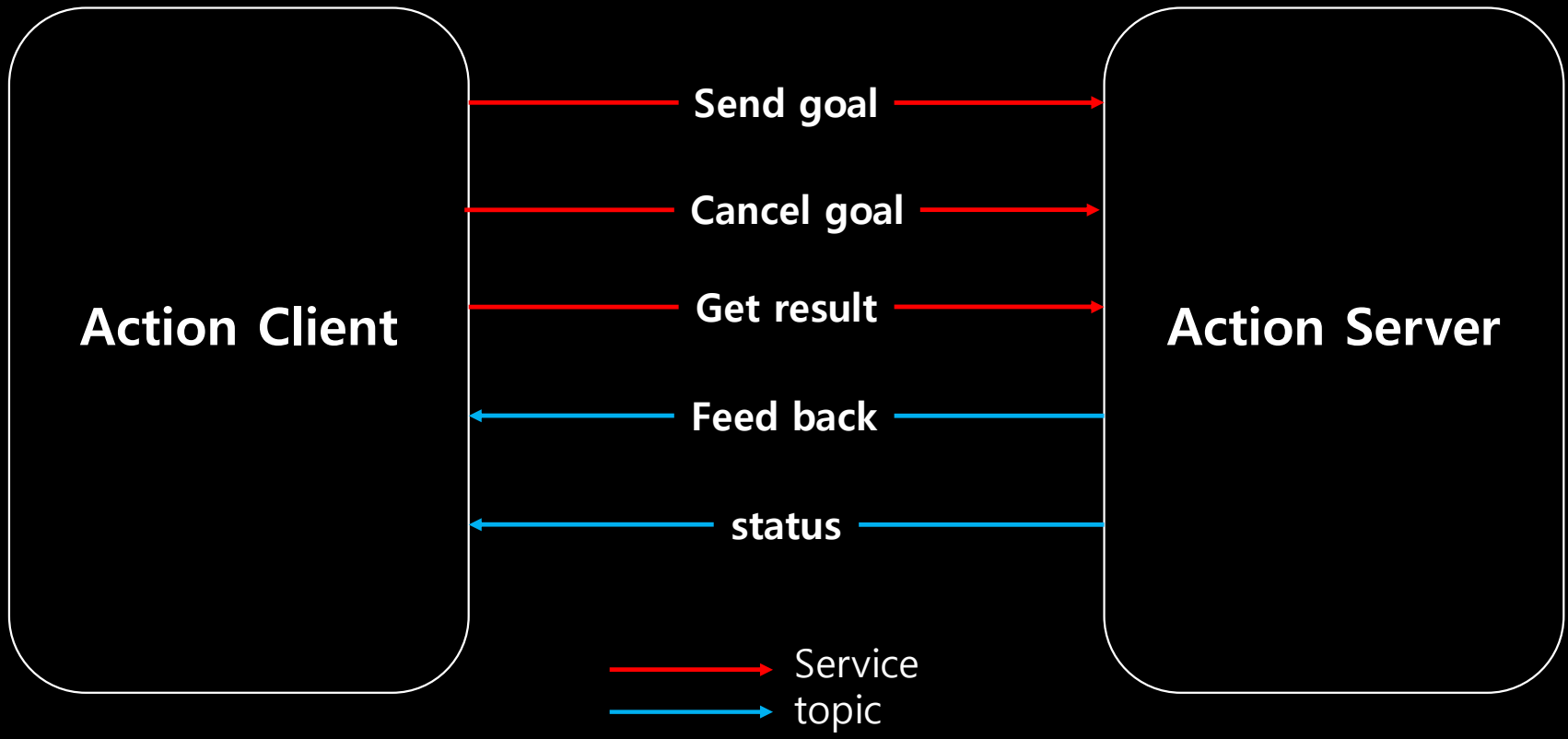


ROS2 기본적인 명령어 Action

- Action server를 구현하는 노드에 클라이언트 노드에서 먼저 서비스로 목표를 요청하면 응답하는 것까지는 service와 같지만, Action은 목표(Goal)를 달성할 때까지 topic으로 feedback을 해준다. Feedback이 끝나면 결과(result) 서비스를 사용한다.



ROS2 기본적인 명령어 Action



ROS Action 액션의 간략화된 개념

감사합니다.