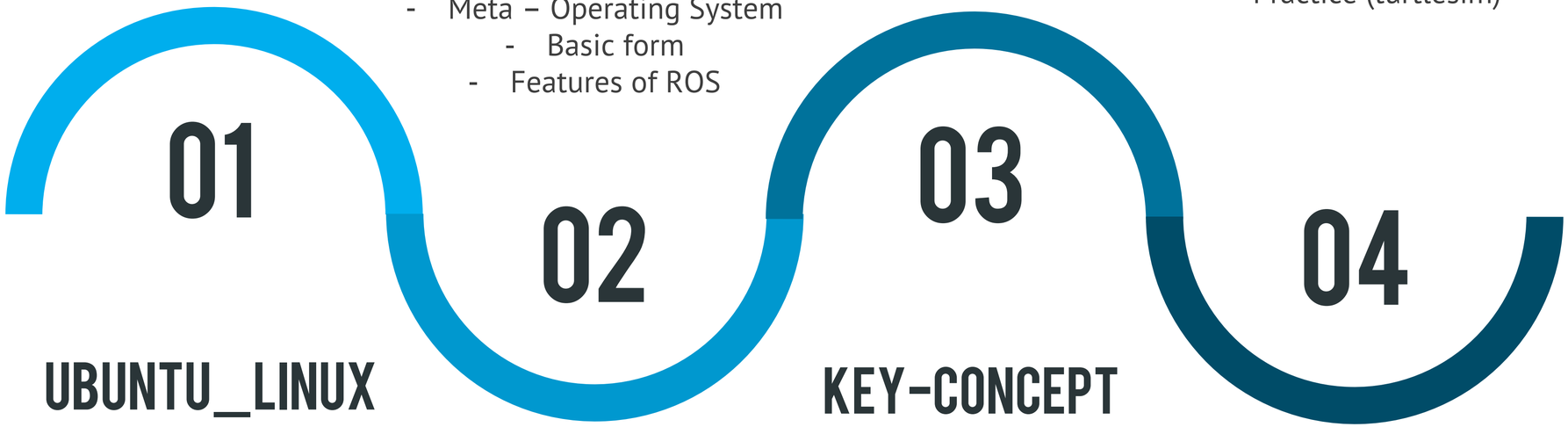
Spring, 2021

지능로봇

UBUNTU_LINUX
ROS AND IT'S CONCEPT
PRACTICE

메카트로닉스공학과

한국산업기술대학교



01

UBUNTU_LINUX

- Terminal
- Command

02

ROS

- What is ROS?
- Why ROS?
- Software framework
- Meta – Operating System
- Basic form
- Features of ROS

03

KEY-CONCEPT

- ROS term
- Message of ROS

04

PRACTICE

- Practice (turtlesim)

01. UBUNTU_LINUX

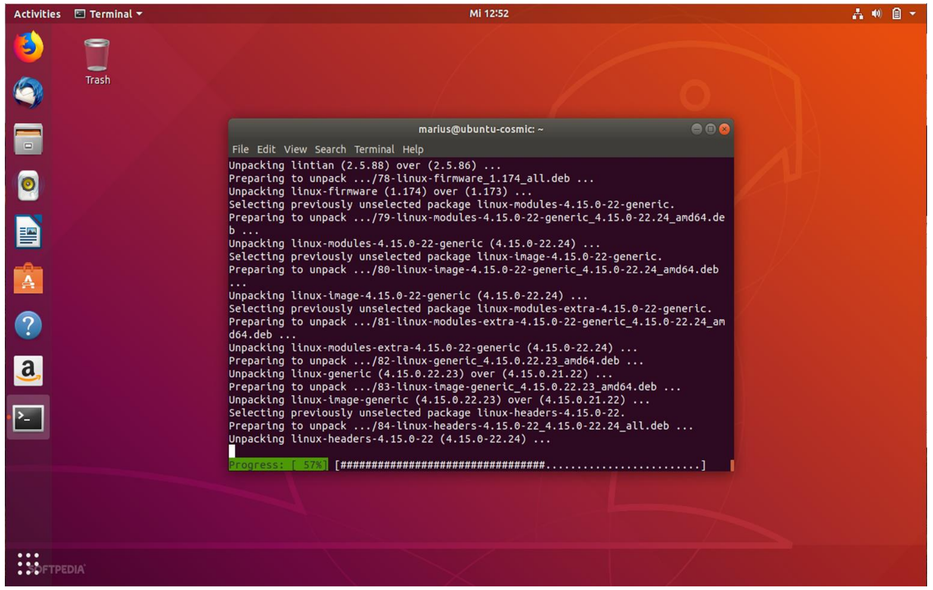
- OS AND KERNAL

OS(Operating System, 운영체제)는 **사용자가 프로그램을 통해 입출력 장치를 편리하게 다룰 수 있도록 해주는 중간역할**

Kernal은 **OS에서 없어선 안될 중요한 존재**로써 **프로세스와 메모리 관리, 하드웨어 입출력 등을 전반적으로 관리, 컴퓨터의 한정된 자원을 효율적으로 사용할 수 있도록 하는 역할**

- LINUX

LINUX는 **Kernal의 일종인 Linux Kernal을 사용하는 OS**를 가리키는 말. **소스코드가 오픈되어 있는 대표적인 오픈 소프트웨어**. 컴퓨터 역사상 가장 많은 사람이 들어간 오픈 소스 프로젝트 로써, 안드로이드 또한 LINUX를 기반으로 함



01. UBUNTU_LINUX

- WINDOWS VS LINUX

OS	Windows	Linux
비용적 측면	- **마이크로소프트 사에서 만든 상용 OS** - **가격이 고가**이며 기본 애플리케이션을 이용할 경우 **추가 비용이 발생**함. 리눅스에서는 무상으로 제공되는 소프트웨어를 윈도우의 경우 **별도 비용을 내고 구입**해야 함	- **무료로 사용할 수 있는 공개 OS** - 무료 라이선스로 사용 제한이 없기 때문에 구축하고자 하는 모든 시스템에 **추가 비용없이 설치가 가능** - 애플리케이션과 문서들 또한 **무상 혹은 저렴하게 구입할 수가 있음**
안정성 및 신뢰도	- **윈도우의 안정성과 신뢰도는 리눅스, 맥OS 등의 타 OS보다 낮은 편** - 윈도우에서 자주 접하게 되는 블루 스크린이 그 대표적인 예	- 리눅스는 **안정성과 신뢰도를 높이는 운영체제**로 인정받으면서 널리 사용되고 있음 - 리눅스는 윈도우에 비해 신뢰도가 높은 편이지만 **디스크 입출력이 비동기화 방식이기 때문에 시스템 충돌, 전원문제 등이 발생할 경우 파일시스템이 깨질 수 있는 우려**도 있음
성능	- 윈도우 개발언어(ASP: Active Server Pages)로 개발된 서비스 페이지의 경우 윈도우OS가 설치된 서버에서 최고 성능을 발휘함	- 대부분의 응용프로그램에 대해 **좋은 성능을 발휘**하며 **낮은 성능의 서버로 고성능**을 낼 수 있음

01. UBUNTU_LINUX

- WINDOWS VS LINUX

OS	Windows	Linux
보안성	- 리눅스에 비해 시스템 버그나 보안 취약점 발견 시 **패치가 나오는데 상당한 시일이 걸림**	- 리눅스는 다중 사용자 체제이므로 관리자 권한(root)으로 로그인하지 않으면 모든 사용자는 보호모드에서 작동하므로 **윈도우에 비해 바이러스가 매우 적고 보안성이 높은 편** - 리눅스의 모든 소스는 인터넷상에 공개되어 있기 때문에 **보안이슈 발생 시 발빠른 대처가 가능**
응용 프로그램	- 윈도우용 무료 프로그램의 수는 매우 적으며 소스코드 없이 제공되기 때문에 **사용자에 의해 수정 혹은 개선이 불가능**	- 리눅스는 다양한 무료 오픈소스 프로그램들이 존재하며, **직접 소스 수정 및 제작을 하여 배포할 수 있음**
상용 애플리케이션	- 윈도우는 다른 어떤 운영체제 보다 많은 애플리케이션을 보유하고 있다는 것이 최대 장점, WMA 스트리밍 서비스 같이 ASP로 개발된 페이지를 구동해야 하는 경우가 그 대표적인 예	- 리눅스를 지원하는 상용 애플리케이션들은 점차 늘고 있는 추세
운영관리 측면	- 텍스트 입력 방식의 리눅스보다 **GUI 기반의 윈도우가 더 편리**함	- 기술 지원 측면에서 리눅스가 **전문적이고 신속하게 이뤄진다고 할 수 있음** - 리눅스 개발 업체들 외에도 지원을 받을만한 뉴스그룹, 메일링 리스트와 같은 무상 질의/답변 포럼이 많으며 사용자가 소스를 수정하여 **직접 문제를 해결**할 수도 있음

01. UBUNTU_LINUX

- UBUNTU

Ubuntu는 **Linux Kernal**을 기반으로 한 Linux 배포판 가운데 하나

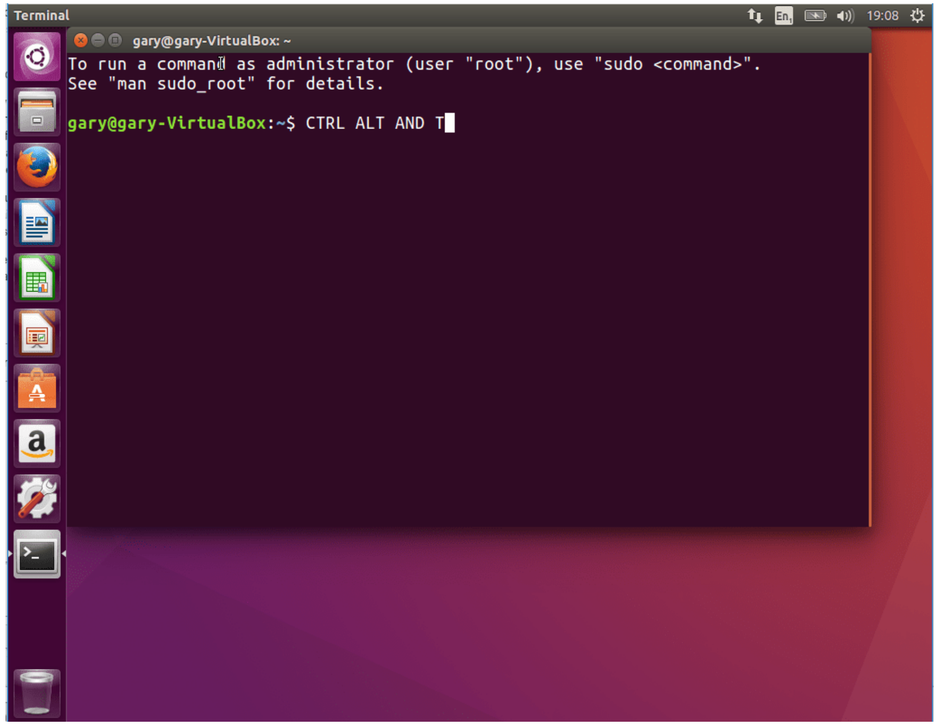
Ubuntu는 배포판을 수정하거나 수정한 것을 재배포할 수 있는 **자유 소프트웨어**로, 지금까지 수많은 변형 배포판들이 나왔음

Ubuntu의 기본적인 철학, 즉 **전 세계의 사람 누구나 어렵지 않게 리눅스를 사용하자**는 표어에서 알 수 있듯이, 기본적으로 세계의 **다양한 언어를 지원**하고 그다지 **높은 사양의 컴퓨터가 필요하지 않음**

01. UBUNTU_LINUX

- TERMINAL

- **Terminal**은 **입출력장치와 컴퓨터 간의 소통을 가능하게 해주는 인터페이스**와 같은 역할
Ubuntu에서는 대부분의 작업을 **터미널을 통해서 할 수 있음**
- **Terminal**에서 가장 기본적으로 사용되는 프로그램은 **Shell**로서, Enter를 누를 때마다 **문자 기반 명령어를 컴퓨터 언어로 변환하여 컴퓨터에게 전달**
- Windows에선 cmd.exe를 기본적으로 사용하며, **Ubuntu**에선 **bash**를 사용
- 대표적인 명령어는 다음과 같음
• ls : 현재 경로 내의 폴더 및 파일 리스트 출력
• cd : 해당 경로로 이동
• chmod : 권한 변경
• ssh a@b : a에서 b로 통신 연결
• grep : 파일들에서 패턴 찾기
• sudo : 관리자 권한 부여
• apt install/remove/update/upgrade : 패키지 설치, 제거 및 업데이트



01. UBUNTU_LINUX

- COMMAND

명령어	사용법
passwd	패스워드 변경
ls	파일 리스트 보기
cd	디렉토리 변경
cp	파일 복사
mv	파일이름 / 위치 변경
mkdir	디렉토리 생성
rm	파일 삭제

명령어	사용법
rmdir	디렉토리 삭제
pwd	현재의 디렉토리 경로 보여주기
chmod	파일 Permission 변경
who	현재 시스템에 login하고 있는 사용자의 리스트를 보여줌
vi, touch, cat, nano	새로운 파일을 만드는 방법
cat, head, tail	파일 내용만 보기

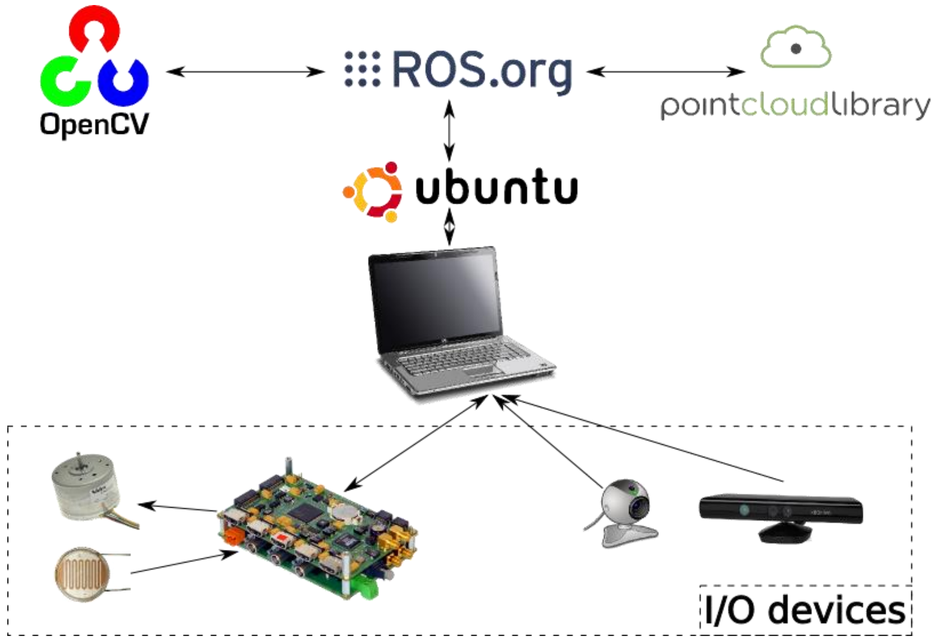
02. ROS (ROBOT OPERATING SYSTEM)

- WHAT IS ROS?

ROS는 로봇 소프트웨어를 작성하기 위한 **프레임워크**로써, **각양각색의 로봇 플랫폼**에서 로봇 행동을 만들어 내는 작업을 단순화시키는 목적의 도구, 라이브러리, 그리고 규약들을 모은 것이다.

- WHY ROS?

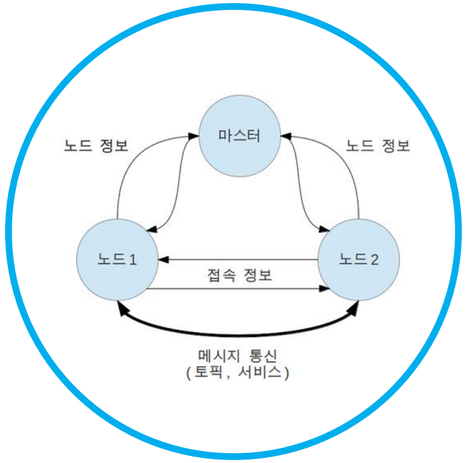
신뢰성 있는 범용 로봇 소프트웨어를 만드는 것은 **매우 어려움**. 하나의 개인 혹은 기관에서 밑바닥부터 완전한 시스템을 만들 수는 없음.
ROS는 **밑바닥부터 협력적인 로봇 소프트웨어** 개발을 장려하도록 만들어짐.



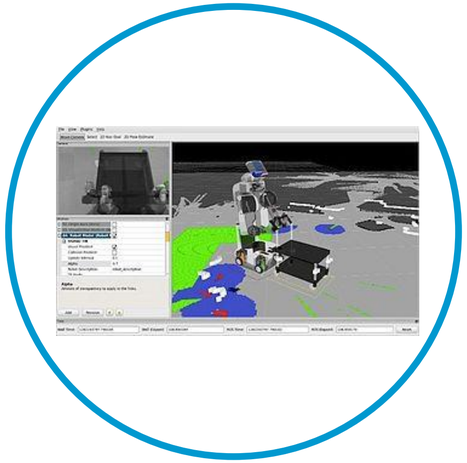
02. ROS (ROBOT OPERATING SYSTEM)

- SOFTWARE FRAMEWORK

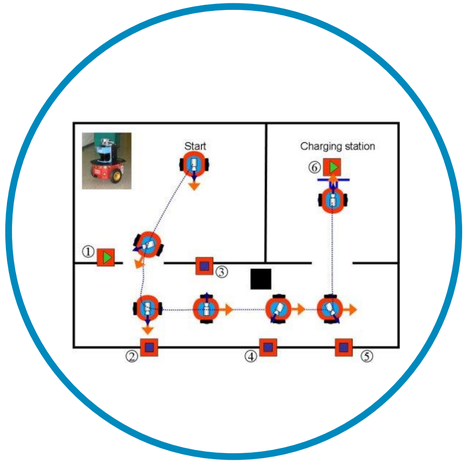
프레임워크 : 어떠한 목적을 달성하기 위해 복잡하게 얽혀 있는 문제를 해결하기 위한 구조.
ROS는 **로봇 구동 및 협업**을 위해 통신환경 / 도구 / 기능 / 생태계 를 제공하는 **프레임워크**.



노드간에 메시지 교환
방법으로 복잡한
프로그램을 잘게 나눠
공동 개발이 가능



명령어 도구 / 시각화
도구(Rviz) / GUI 도구(rqt) /
3차원 시뮬레이터(Gazebo)
등 **각종 유용한 도구 제공**



로보틱스에서 많이
사용되는 모델링, 센싱,
인식, 내비게이션,
매니퓰레이션 **기능 지원**

세계를 대상으로
로보틱스 소프트웨어
공동 개발이 가능한
생태계 생성이 가능
.

02. ROS (ROBOT OPERATING SYSTEM)

- META – OPERATING SYSTEM

Meta - Operating System은 딱히 정의된 용어는 아니지만, 어플리케이션 분산 컴퓨팅 자원간의 가상화 레이어로 **분산 컴퓨팅 자원을 활용하여, 스케쥴링 및 로드, 감시, 에러 처리 등을 실행하는 시스템**이라고 볼 수 있음

다수의 이기종 하드웨어간의 데이터 송수신, 스케쥴링, 에러 처리 등 로봇 응용 소프트웨어 개발을 위한 필수 기능들을 라이브러리 형태로 제공

02. ROS (ROBOT OPERATING SYSTEM)

- BASIC FORM OF ROS

- MASTER

노드와 **노드** 사이의 **연결 및 메시지 통신**을 위한 네임 서버.
마스터가 없으면 **ROS노드** 간에 **통신을 할 수 없다.**

- NODE

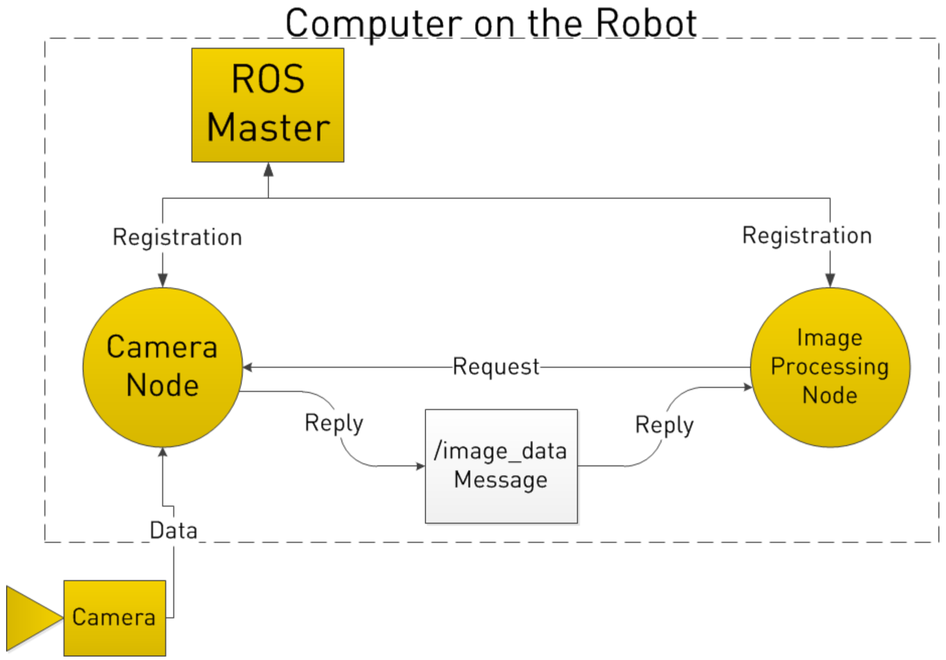
최소 단위의 실행 가능한 프로세서를 가리키는 용어로써 **하나의 실행 가능한 프로그램**으로 생각하면 된다. 각 **노드**는 **메시지 통신**으로 데이터를 주고 받는다.

- PACKAGE

하나 이상의 노드, 노드 실행을 위한 정보 등을 묶어 놓은 것.

- MASSAGE

메시지를 통해 **노드간의 데이터**를 주고받게 된다. **메시지**는 interger, floating, point, boolean 과 같은 변수형태이다.



02. ROS (ROBOT OPERATING SYSTEM)

- FEATURES OF ROS

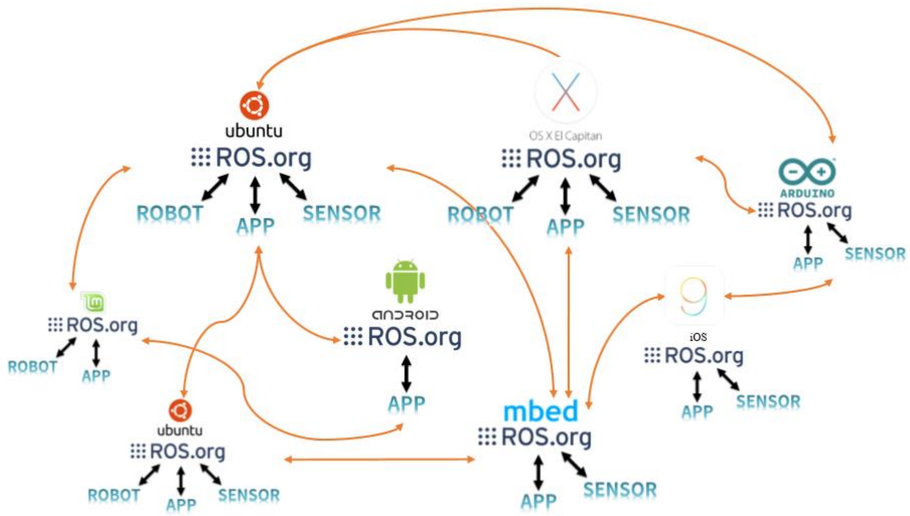
COMMUNICATION INFRA / VARIOUS CAPABILITIES / VARIOUS TOOLS

1
- **메시지 파싱 기능**
  - 로봇 개발에 빈번히 사용되는 **통신 시스템 제공**
  - 코드 재사용을 촉진하는 노드들 간의 **메시지 전달 인터페이스**

2
- **메시지의 기록 및 재생**
  - 노드 간 송/수신되는 데이터인 메시지를 **저장**하고 필요시에 **재사용** 가능
  - 저장된 메시지를 기반으로 **반복적인 실험** 가능, **알고리즘 개발**에 용이함

3
- **메시지 사용으로 인한 다양한 프로그래밍 언어 사용 가능**
  - 노드 간의 데이터 교환이 **메시지**를 사용하기 때문에
  각 노드는 **서로 다른 언어**로 작성이 가능함

4
- **분산 매개 변수 시스템**
  - 시스템에서 사용되는 변수를 글로벌 키 값으로 작성하여
  공유 및 수정이 실시간으로 반영됨

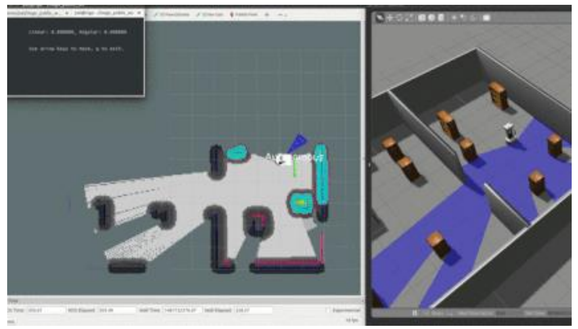


02. ROS (ROBOT OPERATING SYSTEM)

- FEATURES OF ROS

COMMUNICATION INFRA / **VARIOUS CAPABILITIES** / VARIOUS TOOLS

1
- **로봇에 대한 표준 메시지 정의**
  - 각종 센서의 **표준 메시지를 정의**하여 모듈화하고 **협업 작업을 유도**하여 효율성을 향상시킴



2
- **로봇 기하학 라이브러리**
  - 로봇, 센서 등의 상대적 좌표를 트리화 시키는 **TF**(TransForm) 제공

3
- **로봇 기술 언어**
  - 로봇의 **물리적 특성**을 설명하는 XML 문서 기술

4
- **진단 시스템**
  - 로봇의 상태를 한 눈에 파악할 수 있는 **진단 시스템** 제공

5
- **센싱/인식**
  - 센서 드라이버, 센싱/인식 레벨의 라이브러리 제공

6
- **내비게이션**
  - 로봇의 **위치/자세** 추정, 지도내의 **자기 위치** 추정 제공
  - 지도 작성에 필요한 **SLAM**, 작성된 지도 내에서 목적지를 찾아가는 **Navigation** 라이브러리를 제공

7
- **매니퓰레이션**
  - 로봇 암에 사용되는 **IK**, **FK**, 응용단의 **Pick and Place**를 지원하는 다양한 매니퓰레이션 라이브러리 제공
  - GUI 형태의 매니퓰레이션 Tools 제공 (Moveit!)

02. ROS (ROBOT OPERATING SYSTEM)

- FEATURES OF ROS

COMMUNICATION INFRA / VARIOUS CAPABILITIES / **VARIOUS TOOLS**

1

- **명령어 도구, Terminal**
  - **GUI 없이** ROS에서 제공되는 명령어로만 로봇 access 및 거의 모든 ROS 기능 소화

2

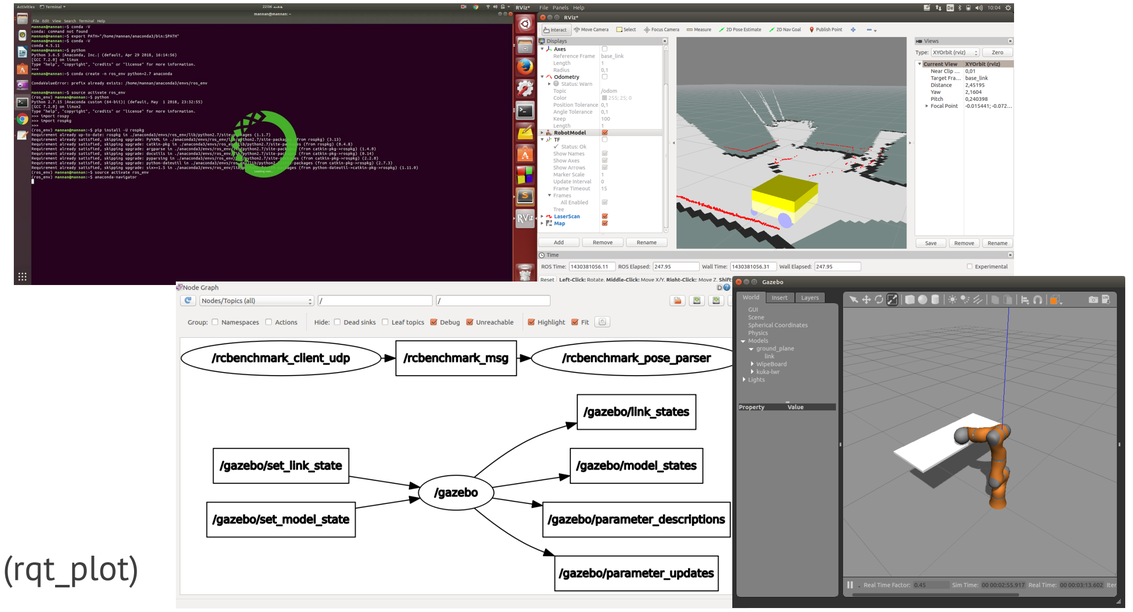
- **시각화 도구, Rviz**
  - 강력한 **3D 시각화 도구** 제공
  - 레이저, 카메라 등의 **센서 데이터를 시각화**
  - **로봇 외형**과 **계획된 동작**을 표현

3

- **GUI 도구, RQT**
  - 그래픽 인터페이스 개발을 위한 Qt 기반 **프레임워크** 제공
  - 노드와 그들 사이의 **연결 정보** 표시 (rqt_graph)
  - 인코더, 전압, 또는 시간이 지남에 따라 변화하는 숫자를 **플로팅** (rqt_plot)
  - 데이터를 메시지 형태로 **기록**하고 **재생**(rqt_bag)

4

- **Simulation 도구, Gazebo**
  - 물리 엔진을 탑재하여 로봇, 센서, 환경 모델 등을 지원하는 **3차원 시뮬레이터**
  - ROS와 **높은 호환성**을 지님



03. KEY-CONCEPT OF ROS

- ROS TERM

TOPIC

- **토픽은 노드 메시지의 이름**과 같다.
- **토픽이라는 메시지 공간**을 만들고 수신자와 발신자가 그를 통해 메시지를 주고 받을 수 있다.
- 지속적으로 메시지를 송신 해야 하는 **센서 데이터에 적합**

PUBLISHER

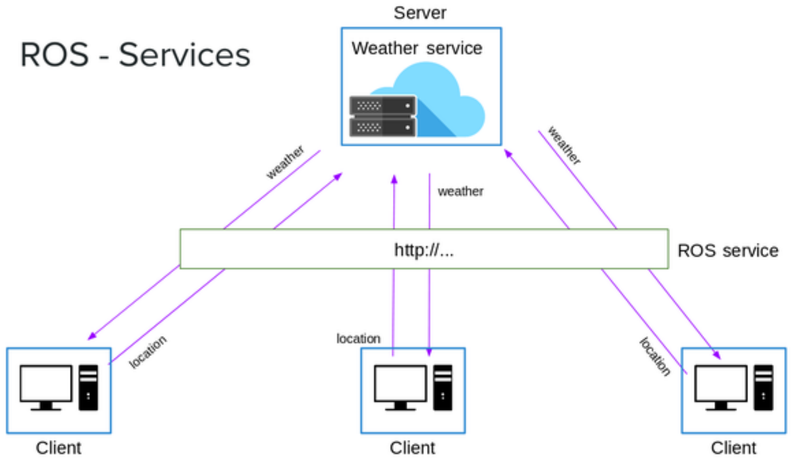
- 토픽의 내용에 해당하는 **메시지 형태의 데이터를 송신하는 것**을 의미한다.
- 토픽을 포함한 자신의 정보를 마스터에 등록하고 **구독을 원하는 서브스크라이버 노드에게 메시지를 보낸다.**
- 복수개로 등록이 가능하다.

SUBSCRIBER

- 토픽의 내용에 해당되는 **메시지 형태의 데이터를 수신하는 것**을 말한다.
- 토픽을 포함한 자신의 정보를 마스터에 등록하고 **구독하고자 하는 퍼블리셔 노드로부터 메시지를 받는다.**
- 복수개로 구독이 가능하다.

03. KEY-CONCEPT OF ROS

- ROS TERM



SERVICE

- 요청과 응답이 함께 사용되는 **동기 방식의 메시지 교환 방식**
- 요청이 있을 경우 응답하는 **서비스 서버**와 응답을 받는 **서비스 클라이언트**로 나뉘어 있음
- **1회성**으로 수행됨

SERVICE SERVER

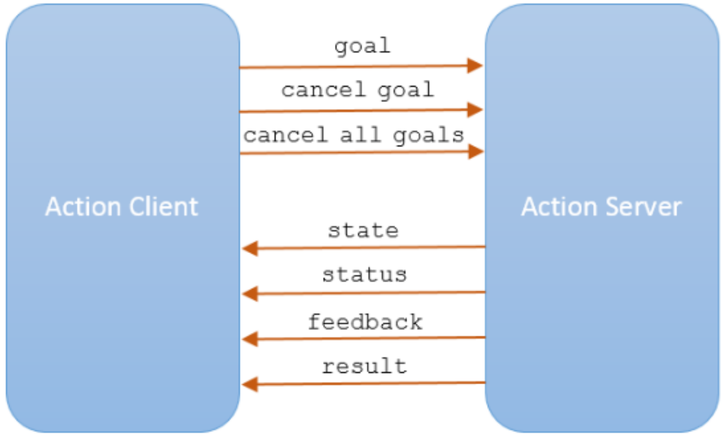
- 요청을 입력으로 받고, 응답을 출력으로 하는 서비스 메시지 통신의 서버 역할
- **서비스 요청에 의하여 주어진 서비스를 수행** 후에 **응답을 서비스 클라이언트에게 전달**
- 클라이언트에 메시지 형태로 응답

SERCIVE CLIENT

- 요청을 출력으로 하고, 응답을 입력으로 받는 서비스 메시지 통신의 클라이언트 역할
- **요청을 서비스 서버에 전달**하고 그 **응답을 서비스 서버로부터 받음**
- 서버에 메시지 형태로 요청

03. KEY-CONCEPT OF ROS

- ROS TERM



ACTION

- 액션 서버와 액션 클라이언트 간에 중간중간 **피드백이 가능**한 프로토콜
- 길고 복잡한 작업 수행 시 사용

ACTION SERVER

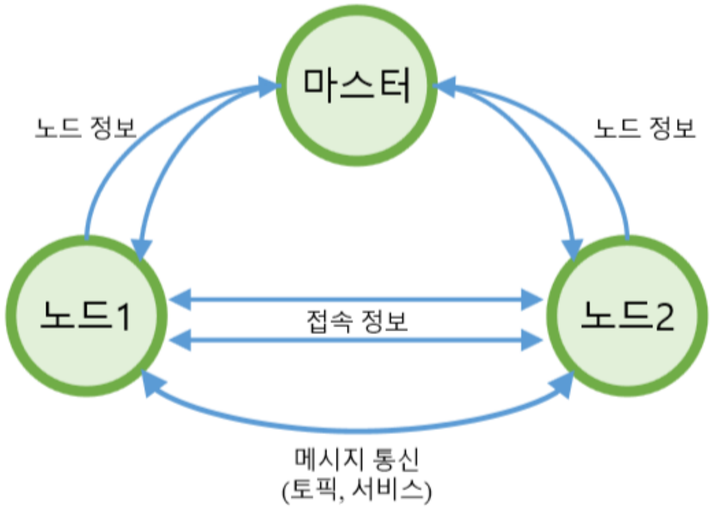
- 액션 클라이언트로부터 받은 **복수의 목표를 수행**
- 작업수행 간에 **서버 상태**와 **작업수행 상황에 대한 피드백**을 액션 클라이언트로 지속적으로 보내고 **모든 작업이 끝나면 결과를 전달** 함

ACTION CLIENT

- 액션 서버로 **복수의 목표를 전달**
- 작업수행 간에 액션 서버가 개별 혹은 전체의 목표를 취소하도록 할 수 있음
- 액션 서버로부터 **서버의 상태**와 **작업수행 상황에 대한 피드백**을 받고 **모든 작업이 끝나면 결과를 전달 받음**

03. KEY-CONCEPT OF ROS

- MESSAGE OF ROS



ROS에서 가장 기본이 되는 기술적 포인트는
노드 간의 메시지 통신

03. KEY-CONCEPT OF ROS

- MESSAGE OF ROS

1. 마스터 구동

XMLRPC
(XML-Remote Procedure Call)
- RPC 프로토콜의 일종으로서, 인코딩 형식에서는 XML을 채택하고, 전송 방식에서는 HTTP 프로토콜을 사용하고 있음.

- 노드의 정보가 XMLRPC 파일에 담기게 됨.

http://ROS_MASTER_URI:11311
- 마스터의 경우 ROS_MASTER_URI라는 호스트 네임과 11311이라는 포트번호를 가짐

명령어
- $roscore

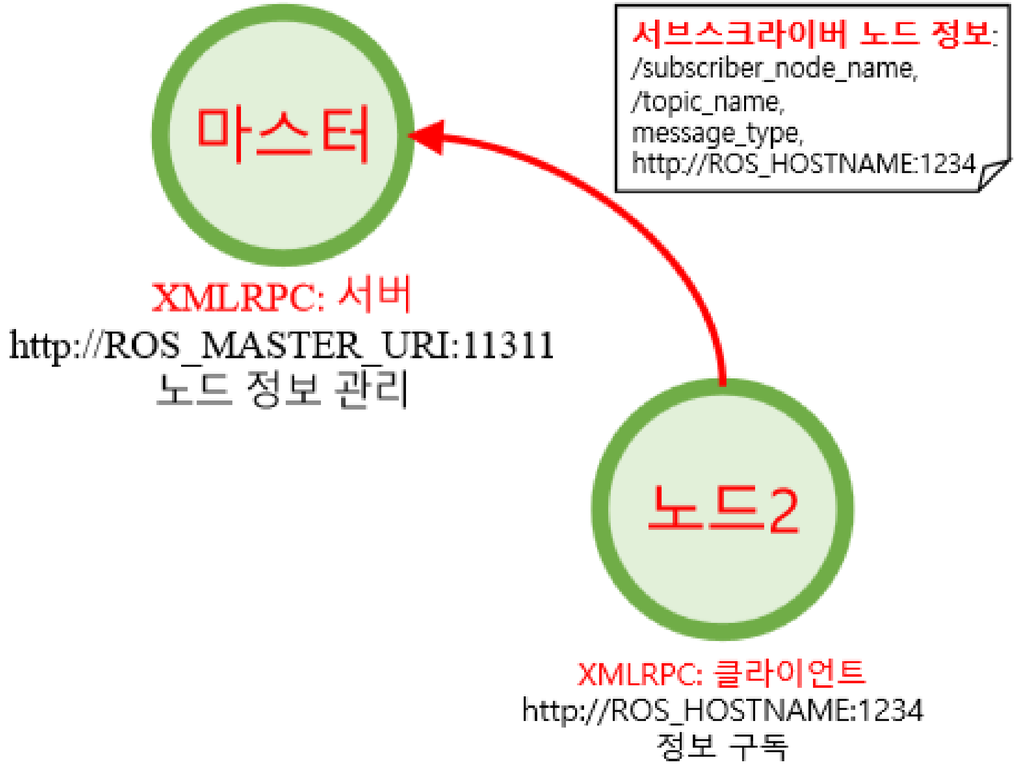
03. KEY-CONCEPT OF ROS

- MESSAGE OF ROS

2. 서브스크라이버 노드 구동

서브스크라이버 노드 정보
- /subscriber_node_name : 노드의 이름
- /topic_name : 토픽의 이름
- message_type : 어떤 메시지 형태인지
  ex) odometry, camera etc…
- http://ROS_HOSTNAME:1234
  ROS_HOSTNAME : 호스트 네임
  1234 : 포트번호

명령어
- $rosrun 패키지이름 노드이름



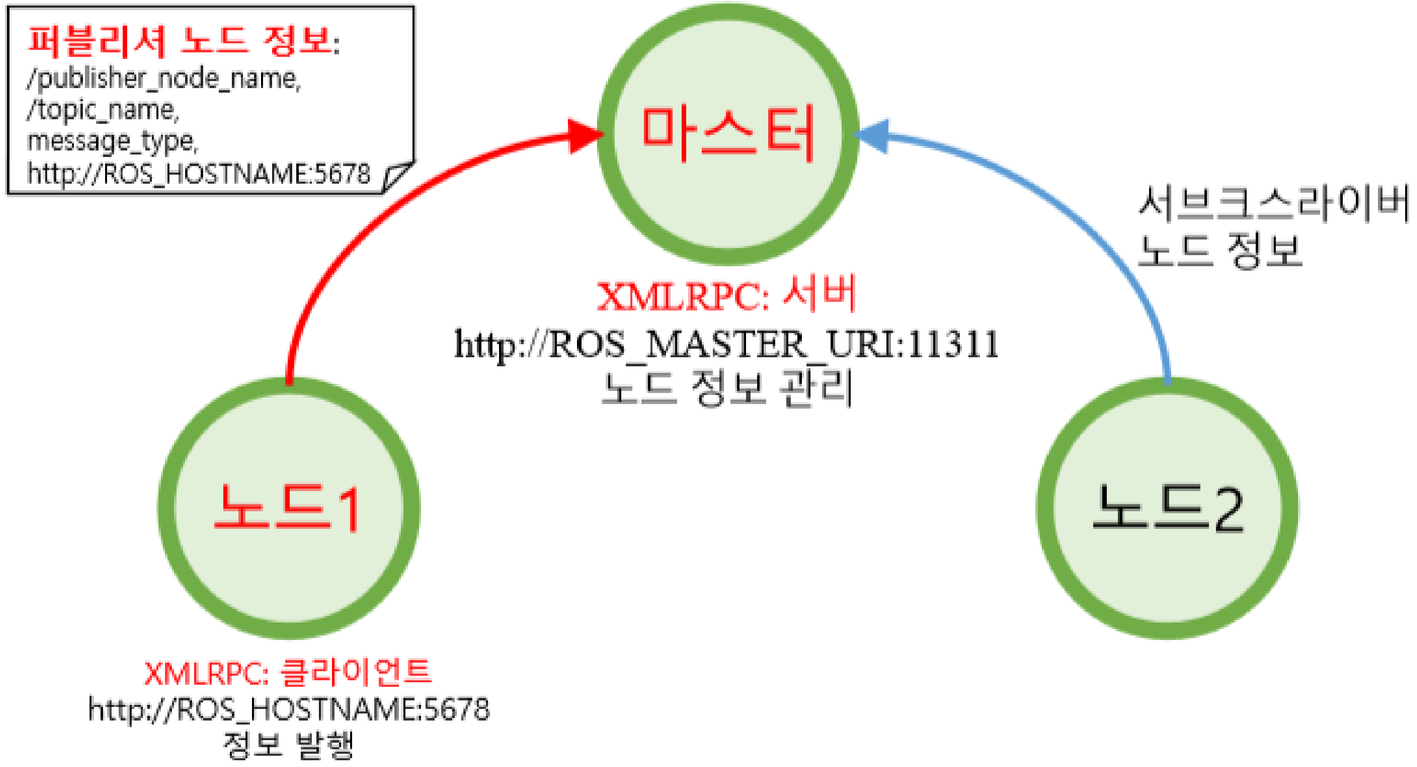
03. KEY-CONCEPT OF ROS

- MESSAGE OF ROS

3. 퍼블리셔 노드 구동

퍼블리셔 노드 정보
- /publisher_node_name : 노드의 이름
- /topic_name : 토픽의 이름
- message_type : 어떤 메시지 형태인지
  ex) odometry, camera etc…
- http://ROS_HOSTNAME:5678
  ROS_HOSTNAME : 호스트 네임
  5678 : 포트번호

명령어
- $rosrun 패키지이름 노드이름

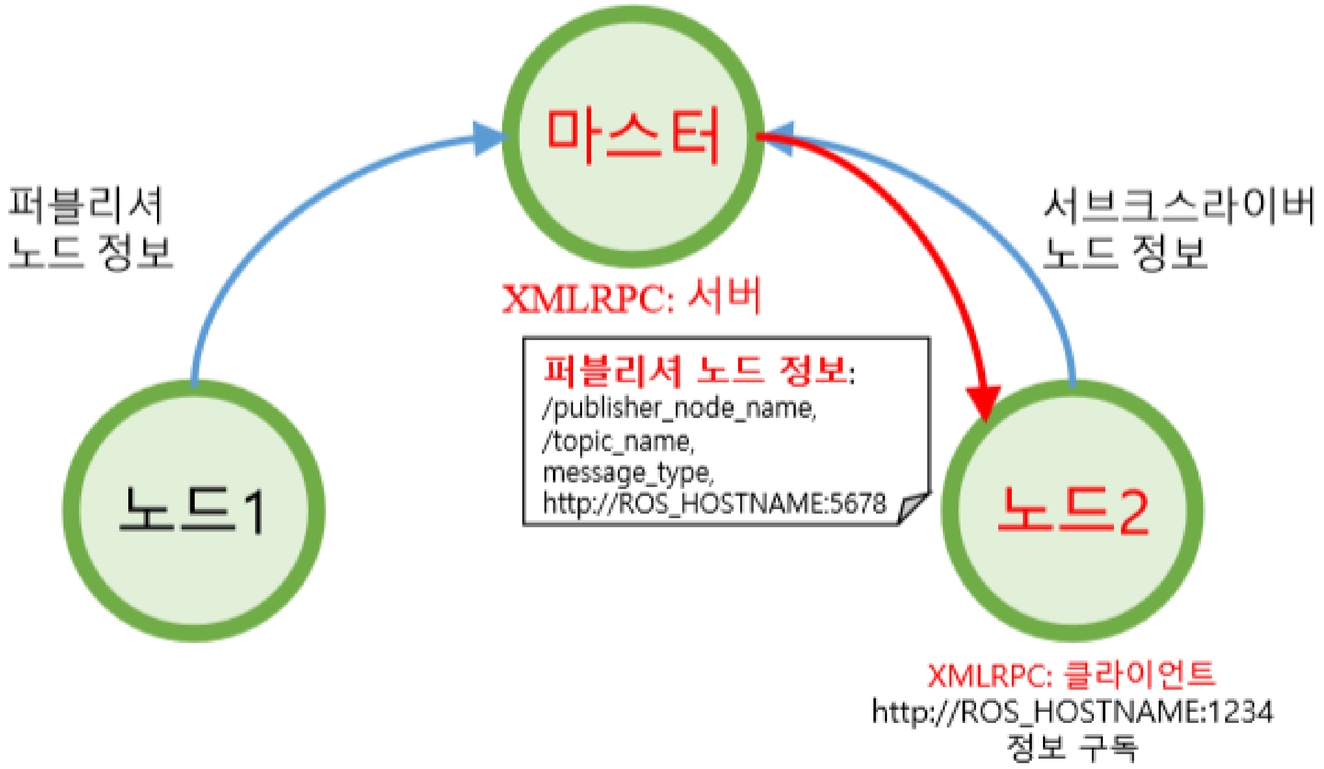


03. KEY-CONCEPT OF ROS

- MESSAGE OF ROS

4. 퍼블리셔 정보 알림

마스터는 서브스크라이버 노드에게 새로운 퍼블리셔 정보를 알림



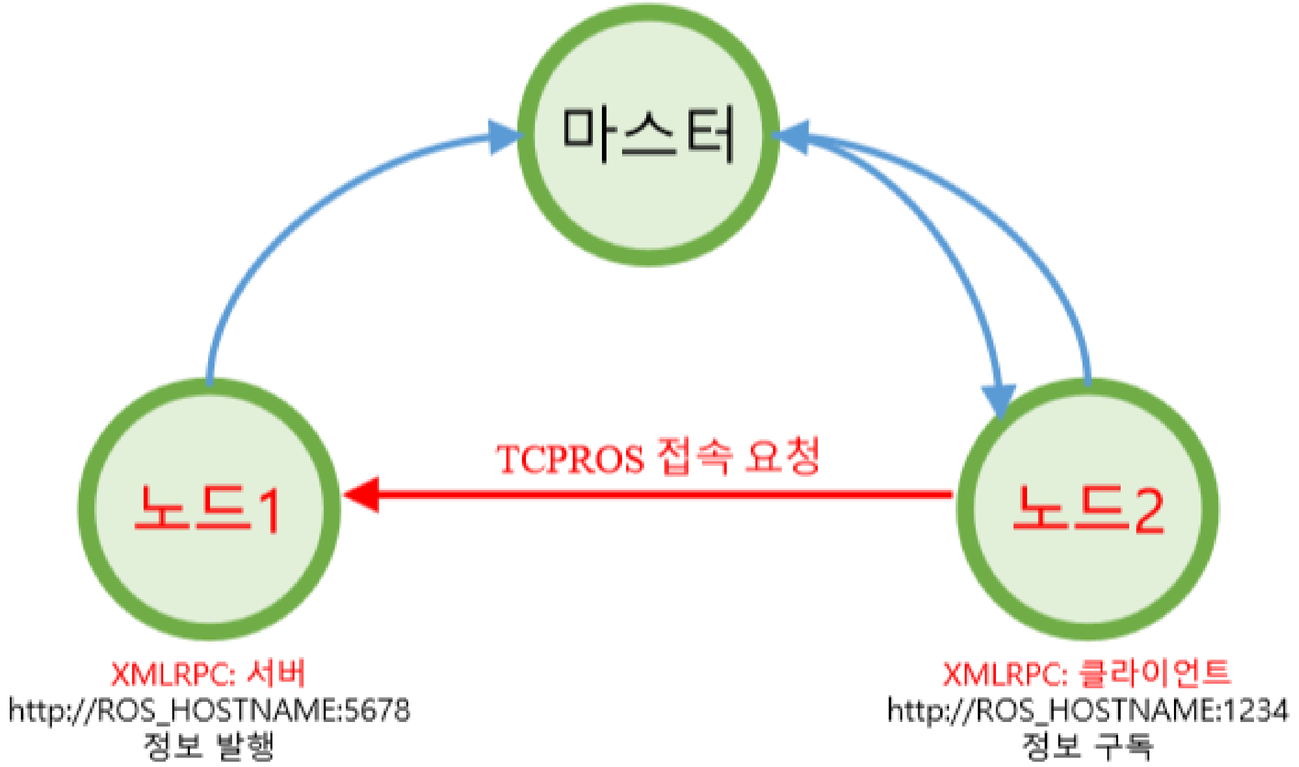
03. KEY-CONCEPT OF ROS

- MESSAGE OF ROS

5. 퍼블리셔 노드에 접속 요청

마스터로부터 받은
퍼블리셔 정보를 이용하여
TCPROS 접속을 요청

TCPROS
- 메시지 및 서비스에서 사용되는 TCP/IP 기반의 메시지 방식

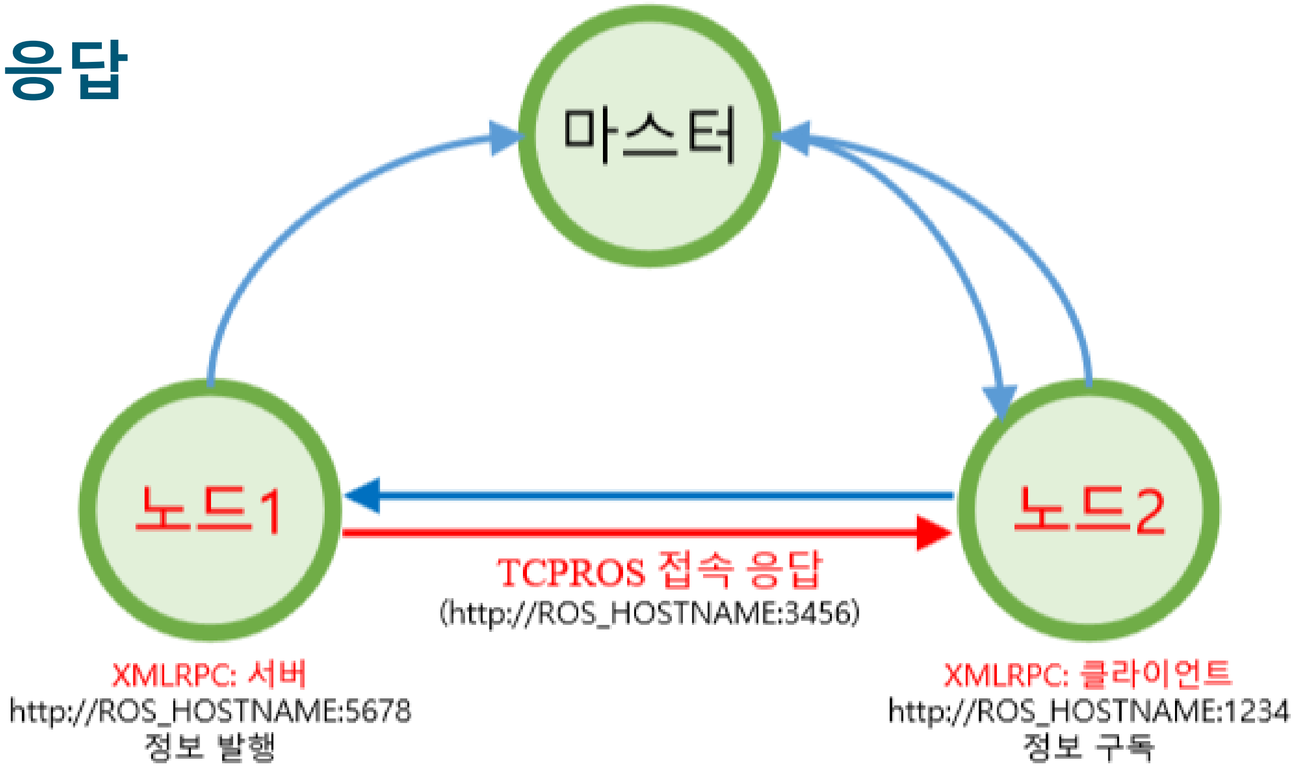


03. KEY-CONCEPT OF ROS

- MESSAGE OF ROS

6. 서브스크라이버 노드에 접속 응답

접속 응답에 해당되는
자신의 TCP URI 주소와
포트번호를 전송

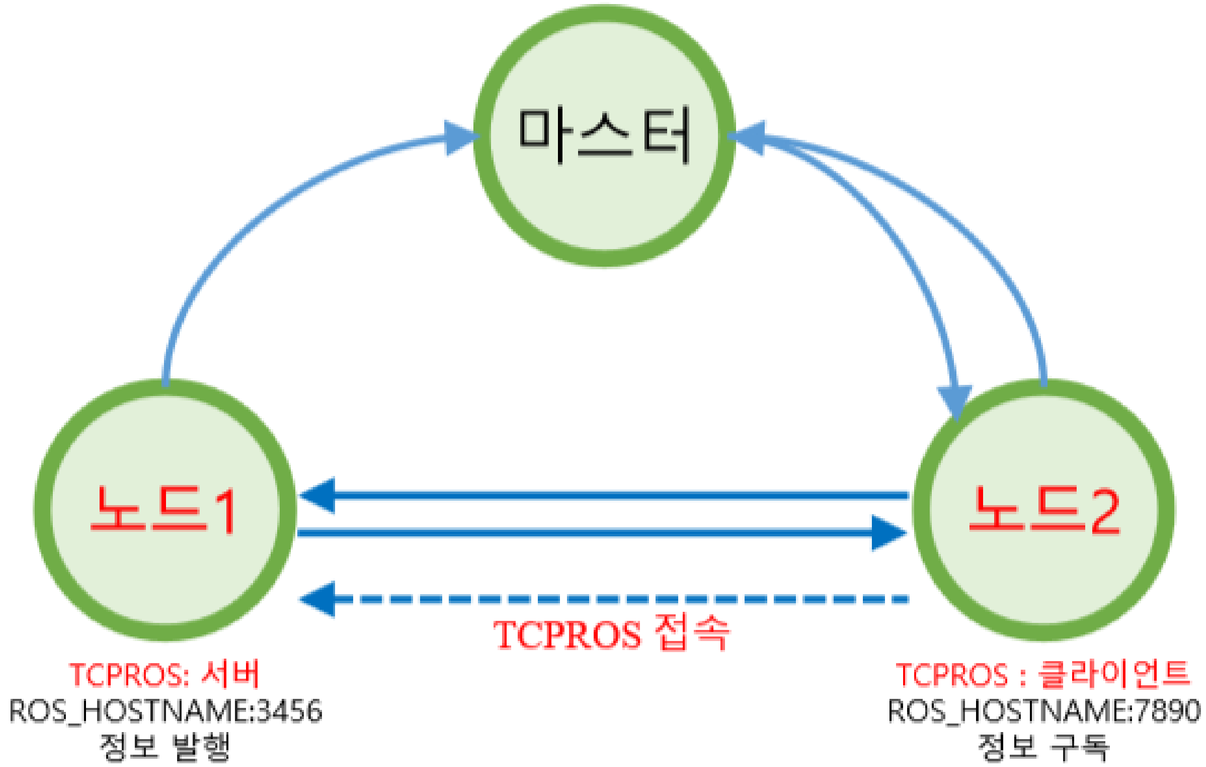


03. KEY-CONCEPT OF ROS

- MESSAGE OF ROS

7. TCP 접속

TCPROS를 이용하여 퍼블리셔 노드와 직접 연결한다.



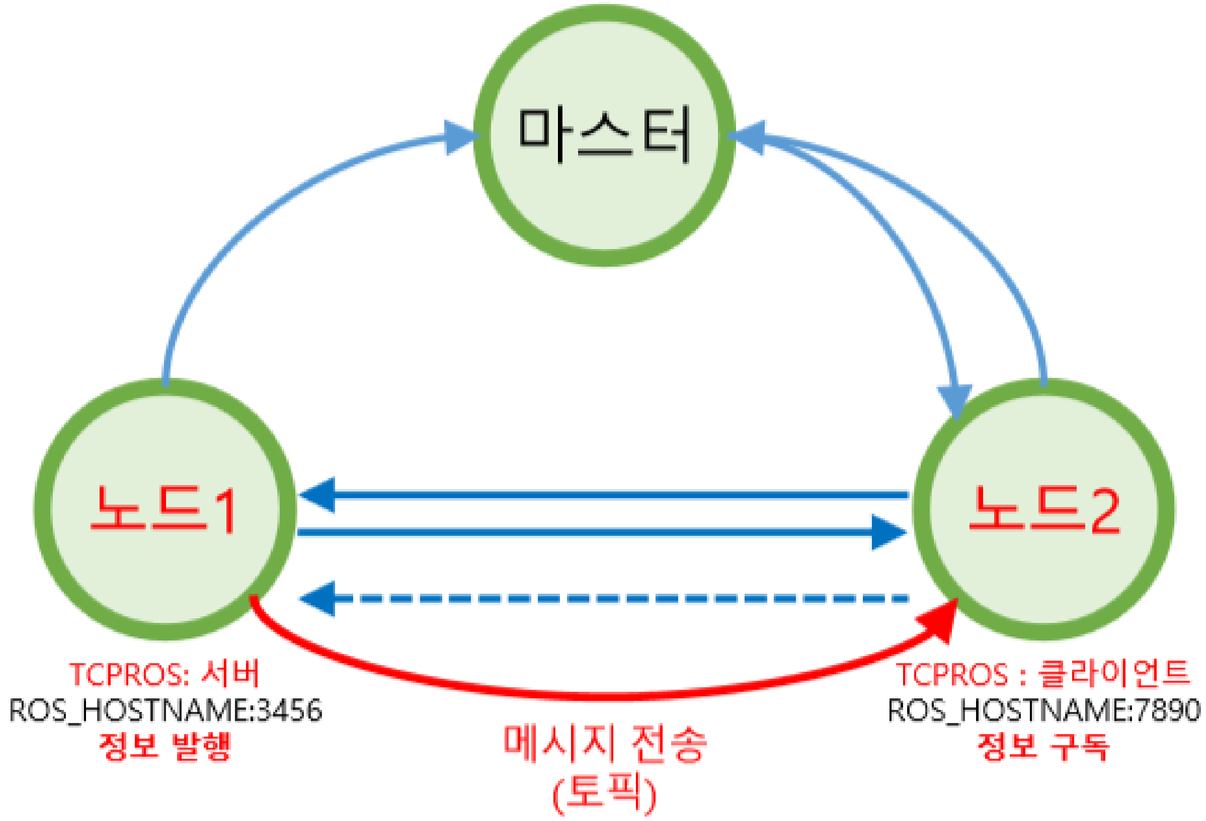
03. KEY-CONCEPT OF ROS

- MESSAGE OF ROS

8. 토픽 메시지 전송

퍼블리셔 노드는
서브스크라이버 노드에게
메시지를 전송

토픽방식에서는 접속을
끊지 않는 이상
지속적으로 메시지를 전송

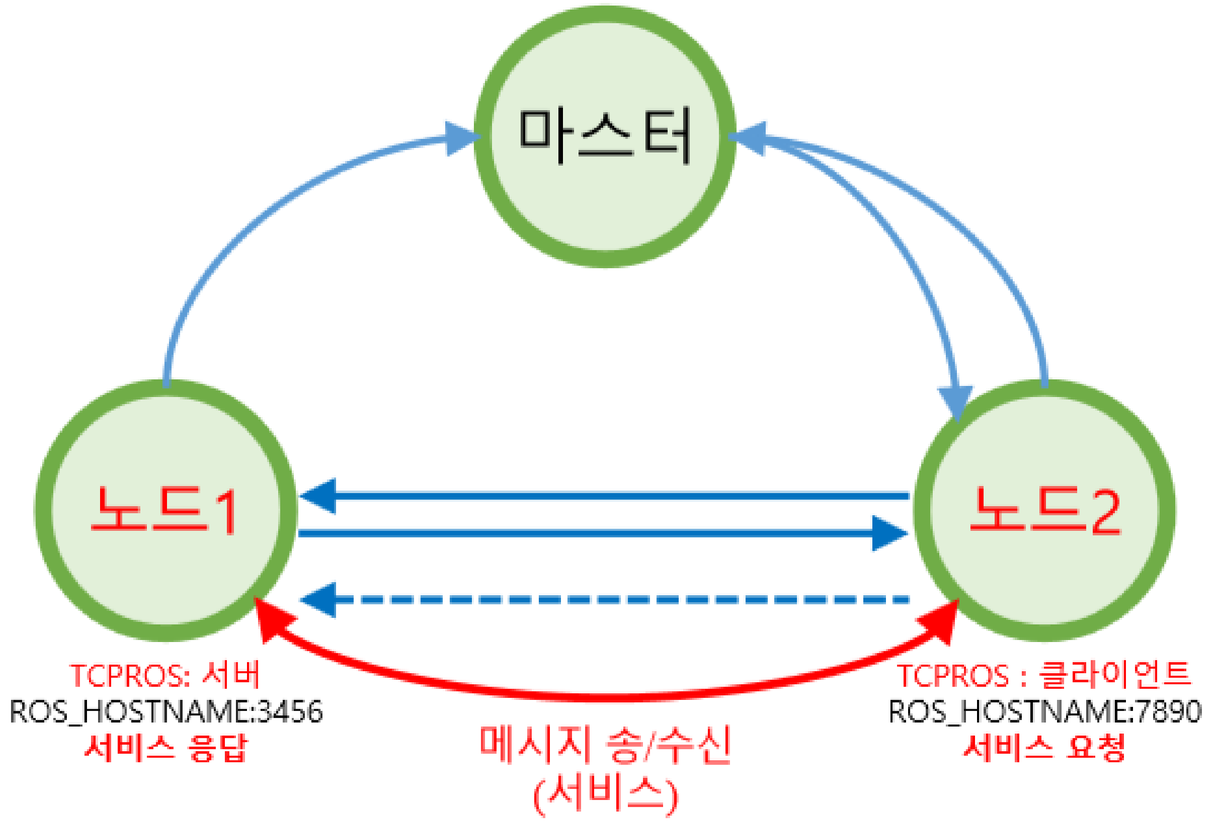


03. KEY-CONCEPT OF ROS

- MESSAGE OF ROS

9. 서비스 요청 및 응답

1회에 한해 접속, 서비스 요청 및 서비스 응답이 수행되고 서로 간의 접속을 끊는다.



04. PRACTICE

- PRACTICE (TURTLESIM)

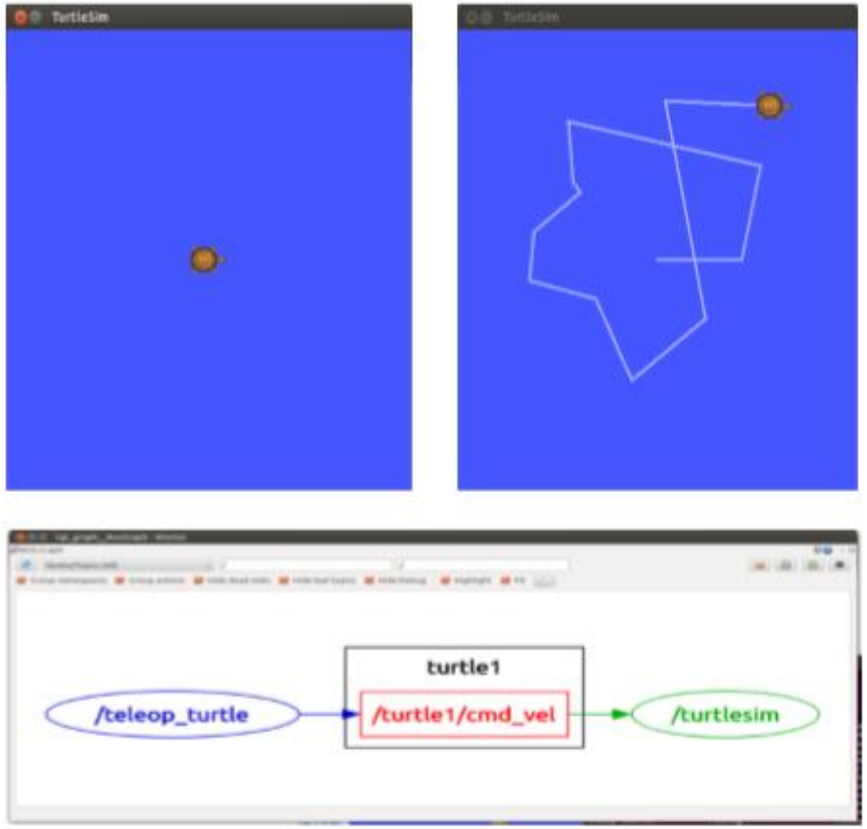
각각 다른 터미널에 작동시켜야 함

$ roscore
: ROS 마스터 실행 (노드간의 통신을 위함)

$ rosrun turtlesim turtlesim_node
: turtlesim_node 실행 (키보드의 입력을 받아 움직이는 노드)

$ rosrun turtlesim turtle_teleop_key
: turtle_teleop_key 실행 (키보드의 입력을 turtlesim_node로 전달하는 노드)

$ rosrun rqt_graph rqt_graph
: 현재 활성화된 노드와 토픽 메시지를 확인 가능



THANK YOU

SOURCES

[1] ROBOTIS ROS SEMINAR, Pyoyunseok , 2018.1.4

[2] Programming Robots with ROS, Morgan Quigley / Brian Gerkey / William D. Smart, 2017.3.31

[3] ROS용어 정리, Monster's wastebasket, http://jungmonster.tistory.com/154