

클라우드 로보틱스 기반의 Thin Client 로봇을 위한 ROS2 on Yocto Project 개발

- Yocto Project란?
- Why Yocto Project?
- ROS2 주요특징 & 업계현황
- SW architecture & Functional block diagram
- 청소로봇 주행검증 및 멀티로봇 제어 검증
- Yocto vs. Ubuntu 주행성능 비교
- LiDAR 기반 실내배송로봇 주행 검증
- 결론 및 향후계획

2021년 6월24일

CTO부문 미래기술센터 로봇선행 연구소

김 윤 성, 이 돈 근, 정 성 훈, 문 형 일, 유 창 승, 이 강 영, 최 준 열, 신 원 정

➤ Yocto Project ?

- 하드웨어 아키텍처와 상관없이 리눅스 기반 embedded 시스템을 위한 오픈 소스 협업 시스템 ('2011. 3~)

“ 개발자가 하드웨어 아키텍처에 관계없이 (또는 의존하지 않고) 임베디드 제품용으로 사용할 수 있는 Linux 기반 시스템을 만드는 Open Source Collaboration Project 이다. “

- 여러 오픈 소스의 집합체로, 하드웨어 독립적인 임베디드 제품 설계를 위한 통합 빌드 시스템 제공
- 빌드 환경, 유틸리티, 툴 제공 → 개발자간 작업환경 의존성이 줄어들 지원 Processor (7개)
- ARM, ARM64, x86, x86-64, PowerPC, MIPS, MIPS64

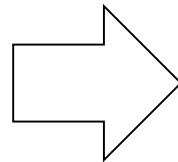
※ Yocto Project Release History 



➤ 기존 Linux 개발 vs. Yocto Project 개발 <https://youtu.be/utZpKM7i5Z4>

As-Is(기존 Linux 개발)

- 공짜인 Linux 배포판 적용
- 제품에 맞는 환경에서 동작하는 Linux 및 tool 개발
- 이슈에 대한 support 필요
→ 비용 및 시간 ↑



To-Be(Yocto Project 개발)

- Linux의 Collaborative Universal Starting Point
- 무료 템플릿, 툴, 메소드, 동작하는 코드 제공
- 컴포넌트를 자유로이 추가/삭제할 수 있음
- 새로운 OS 업데이트 지원 (기존의 최적화된 코드 유지)
→ Faster, Easier, Cheaper !!!

➤ Participants <https://www.yoctoproject.org/ecosystem/participants/>



SIC 업체



제조 업체



SW 업체

Why Yocto Project ?



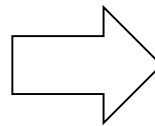
Yocto
(Custom, 맞춤형)

Ubuntu
(Off-The-Rack, 기성품)

- Ubuntu OS는 빠른 프로토타이핑, Proof of Concept 이 장점이나, 다양한 Form factor 로봇 지원, HW 이식성 관점의 멀티 플랫폼, 메모리, SW license 유지보수 및 Production 측면을 고려하여, Bottom-up 방식의 최적화 설계로 ROS2의 장점을 수용한 Yocto Project 기반의 경량 OS 탑재가 필요함

As-Is(Ubuntu 개발)

- 빠른 시스템 구성 및 개발을 통한 프로토타이핑/PoC 에 적합
- Ubuntu를 지원하는 하드웨어 및 대용량 메모리 필요
- HW 이식성, 이력관리를 통한 유지보수 어려움
ex) x86 및 arm64 기반의 동일한 플랫폼 구성 시
→ 별도 개발 필요함
- 불필요한 패키지, 서비스로 인한 메모리 및 부팅시간 증가
- 기본/추가된 SW에 대한 수동 license 추출 필요
- Learning Curve : Linux 데스크탑을 쓰는 것처럼 개발



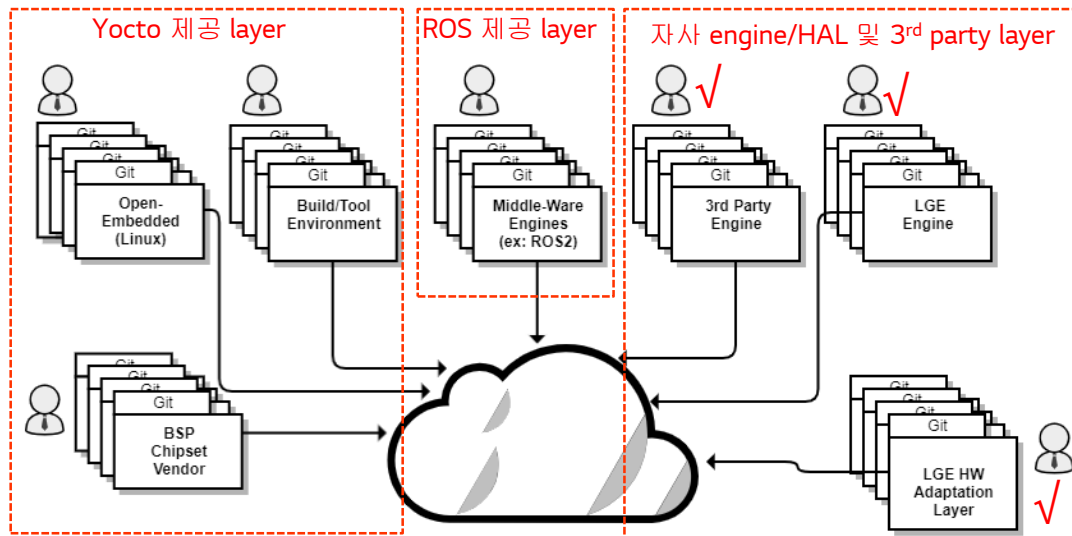
To-Be(Yocto Project 개발)

- 다양한 하드웨어를 지원하는 맞춤형 임베디드 환경에 적합
- 하드웨어 이식성, 유지보수, 이력관리
ex) x86 및 arm64 기반의 동일한 플랫폼 구성 시
→ 부트로더/커널만 변경하면 됨.
- Bottom-up방식 개발을 통한 커스텀화 가능
→ Meta-layer/Recipe를 통한 feature 설정 가능
- 메모리, 부팅시간 최적화 가능
- 빌드 시, 자동으로 license가 포함됨
- Learning curve : 배우기가 어려워 개발자들이 기피

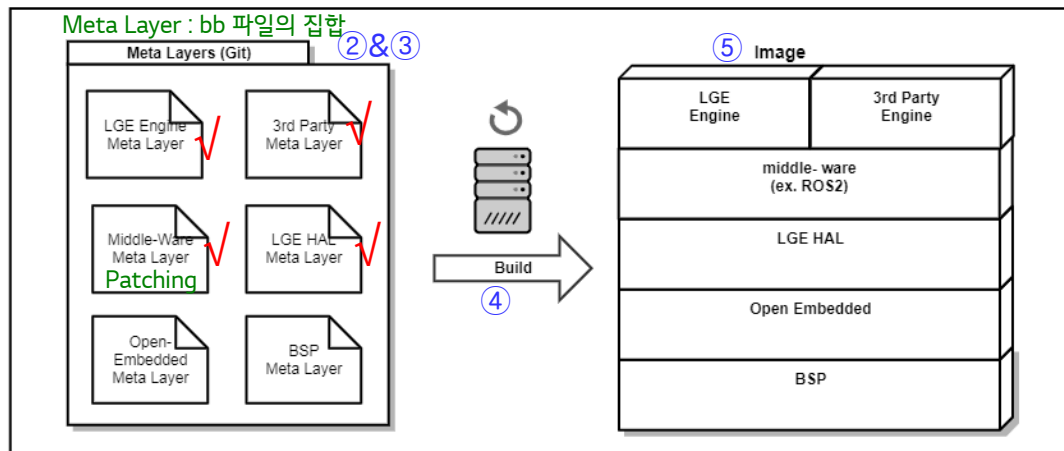
Yocto Project의 Build System

➤ Yocto Build System : BitBake

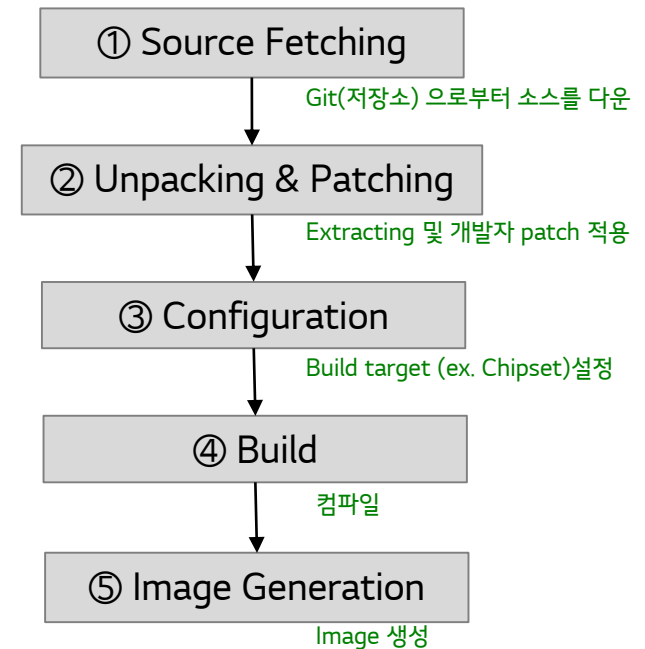
- 독립적인 git 기반(Meta Layer) 의 Bitbake용 파일(bb 파일)로 빌드되며, github 기준으로 소스를 fetch(다운로드)하여 빌드함.
→ Embedded Linux를 위한 cross-compile build 환경 제공
- Bitbake 빌드를 위한 정보는 *.bb 파일에 기술되어 있음 일종의 make file로 git/license 정보, build dependency 정보가 포함됨.
- 자사엔진, 3rd party 엔진, LGE HAL Layer는 모두 bb 파일을 구성해야 함.



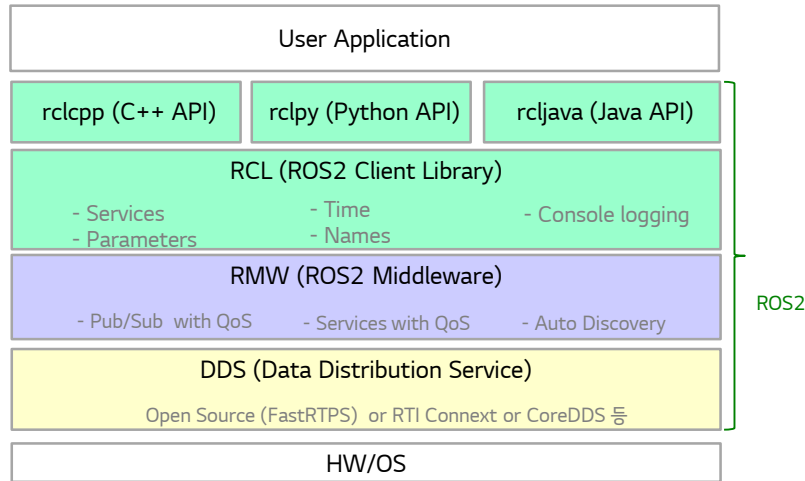
BitBake Build System(빌드 서버) ① Source Download with Git protocol



Build Process



➤ ROS2 Block Diagram



➤ 주요 특징

✓ 앱 개발 도구 강화 (RCL)

- Command line tool, Visualized tool, 시뮬레이터 제공 *rviz, rqt, Gazebo 등*
- ROS2 Client Library 를 통해 개발 언어 지원 및 시스템 자원 제어 기능 제공 *C++, Python, Java, nodejs 등*
- ROS2의 기본 데이터 구조체 제공 및 핸들링 *Services, Time, Parameters 등*

✓ 신뢰성 있는 통신 및 멀티 로봇 지원 강화 (RMW / DDS)

- DDS¹⁾기반의 데이터 중심 분산형 메시지 프로토콜
- RMW 를 통한 DDS 추상화 (DDS vendor들의 솔루션을 선택 사용 가능)
- Master Node 불필요 (Node간 독립성 보장 및 분산 환경 제공)
- 동일 네트워크 내 복수로봇을 동시 제어 (Fleet Control)

➤ ROS2 배포이력



'17. 12/8



'18. 7/2

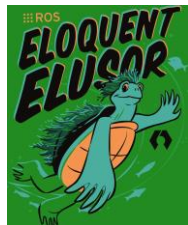


'18. 12/14



'19. 5/31

1st LTS(Long Term Support)
: 2년간 지원

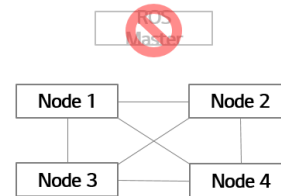


'19. 11/23



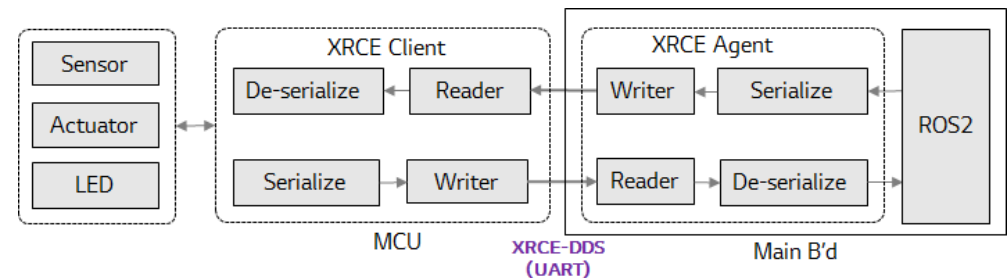
'20. 6/5

2nd LTS(Long Term Support)
: 3년간 지원



✓ MCU등 Embedded 시스템에 대한 고려 (Micro ROS)

- 저사양 embedded 시스템과의 Serial 통신 표준 인터페이스 제공



1) OMG 표준의 분산통신을 위한 메시지 프로토콜로서, 군용, 선박, 철도등에서 주로 사용되고 있으며, ROS2 에서 신뢰성있는 시스템 구성을 통해 양산가능한 플랫폼을 지향하면서 채택하게 됨

오픈소스(ROS)를 적용한 업계/부품/패키지들이 꾸준히 증가하고 있으며, OS, 클라우드 및 칩셋 공급업체의 지원이 확대되고 있음.

Microsoft Amazon, Google Intel, Qualcomm

업계 및 ROS 현황 Review

Implication

업계현황

❑ iRobot



iRobot Create2 : Roomba 교육용 버전을 ROS로 제공

자체 운영체제에서 ROS2와 같은 범용 운영체제의 변경 및 중요성 시사

"iRobot devices still run on a proprietary operating system, but will ultimately move to something more common like the emerging ROS2 robot operating system. As the industry grows, Angle is betting that there will be a common operating system layer." - IROBOT CEO Colin Angel



❑ Xiaomi

Navigation : ROS에서 제공하는 TurtleBot 과 유사

오픈소스 기반의 Player Project 를 적용한 로봇청소기 개발

※ Player Project (2000~2011)

- ROS의 전신이며, 센서 및 모터 기반 모바일 로봇을 위한 오픈소스 제공

❑ Amazon



RoboMaker & DeepRacer



ROS 기반 로봇서비스 제공 : 개발환경, 시뮬레이션, 테스트 및 배포

❑ Microsoft

ROS를 윈도우 10과 클라우드 플랫폼 애저(Azure)에 도입

❑ Intel, Qualcomm

ROS 기반 Robot Platform 배포 (Intel : Openvino, Qualcomm RB3/RB4)

❑ 자사 SVL(Silicon Valley Lab)



WebOS를 적용한 소형주행로봇 선행 개발 (Raspberry-Pi 기반)

❑ ROS 사용현황

20.7 기준

- 사용자 수(명) : 714K, 패키지 다운로드 횟수: 39M

- 공식 패키지 수 : 86,128 개

ROS 월간 페이지 뷰와 패키지 다운로드 및 등록/로봇 수가 꾸준히 증가

ROS를 지원하는 부품의 증가 (120여 개의 다양한 센서/모터 지원)

❑ ROS의 장점

분산 프로세스(Node) 기반의 독립적인 동작 방식

다양한 개발도구(Tools) 지원

→ Rosbag, Rviz, Rqt, Gazebo 등
로깅 3D 시각화 시뮬레이터

ROS 현황 및 장점

▪ 오픈 플랫폼 (ROS2) 도입 필요

- 로봇 생태계의 다수 개발자 및 오픈 패키지를 접목한 로봇 개발 가능
- 비핵심 기능은 오픈 소스 활용, 핵심기능에 역량 집중 가능
- 오픈소스와 자사 알고리즘의 성능비교/벤치마킹을 통한 기술력 증가

ROS was built from the ground up to encourage collaborative robotics software development. The primary goal of ROS is to support code reuse in robotics research and development. SW 재사용



"Nearly 55 percent of total commercial robots shipped in 2024 (over 915,000 units) will have at least one Robot Operating System package installed" -Robotics & Automation News



ROS 적용 제품의 증가 예상 (약 55% 로봇제품, 2024년)

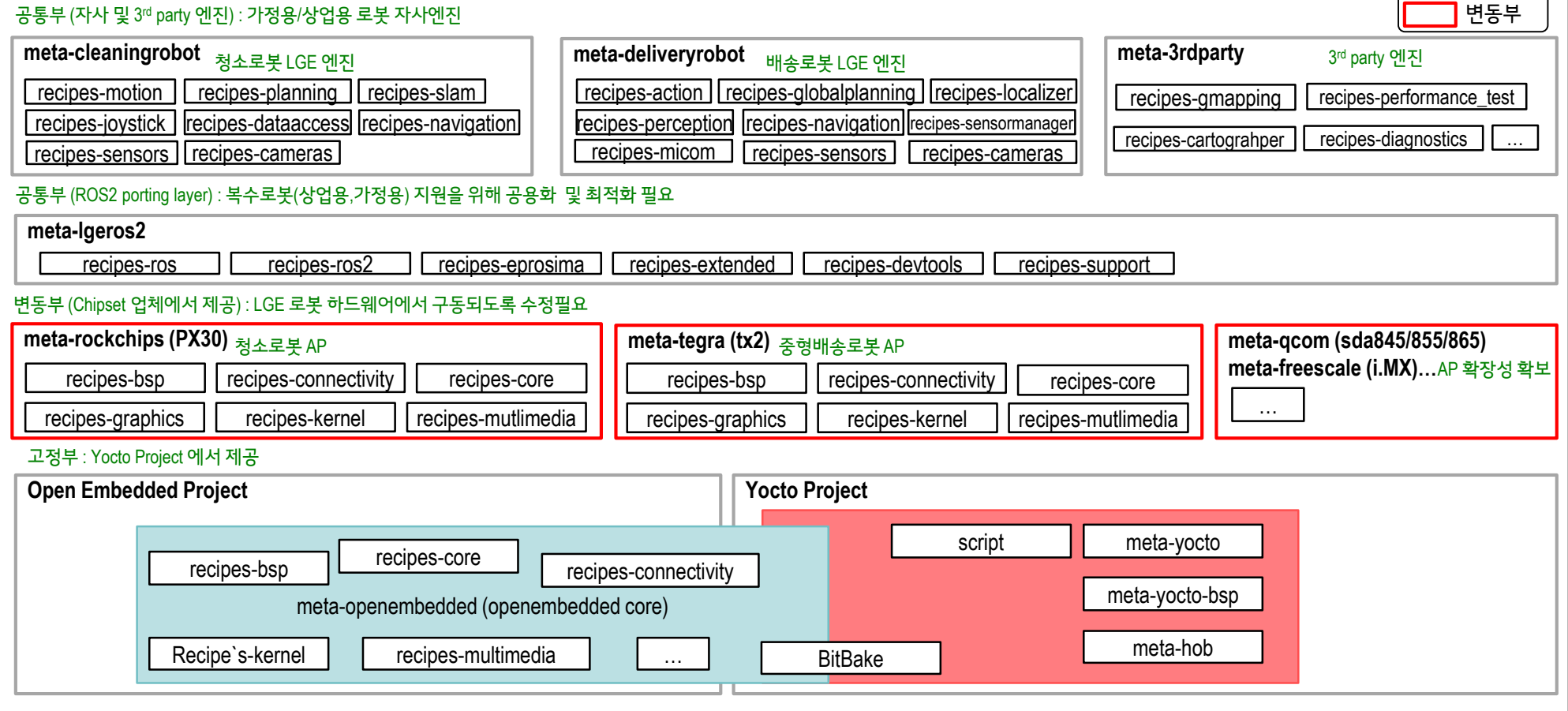
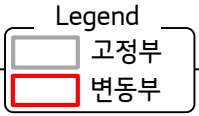
▪ ROS 생태계의 산출물을 활용한 신속한 개발 대응력 확보

오픈 패키지 적용을 통한 신속한 부품 bring-up 및 검증 진행 가능
빠른 프로토타이핑을 통한 신속한 검증 및 평가가능

▪ 개발효율성 극대화(모듈러 기반 공동개발, tool 재사용)

복잡한 SW를 잘게 나누어 독립적인 공동 개발 가능, SW 이식성 용이
별도의 tool 개발 불필요

개발기간/인원, 실패비용 개선 가능

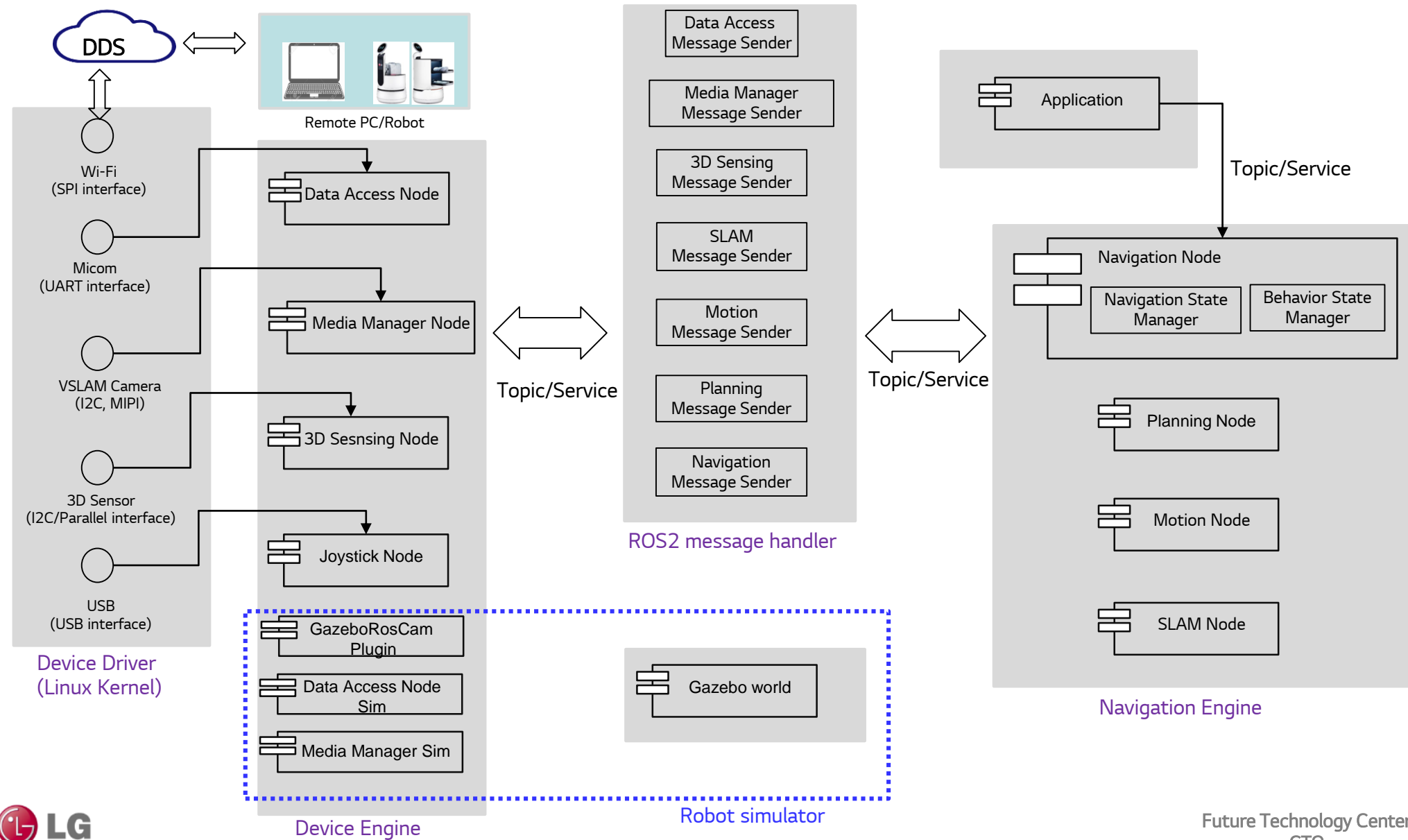


➤ Configuration of meta-layer

- Meta-lge : LGE 로봇 engine을 탑재를 위한 meta-layer
- Meta-3rd party : 오픈 소스패키지 탑재를 위한 meta Layer (SLAM, 음성인식, diagnostics 등의 오픈 패키지의 빠른 적용 가능)
- Meta-lgeros2 : ros2 지원을 위한 meta layer, LGE 로봇 기반으로 memory 최적화 필요
- Meta-rockchips, Meta-tegra : Chipset 업체에서 제공, LGE 로봇 하드웨어 기준으로 수정 및 최적화 필요
- Yocto Project(w/ Open Embedded) : Yocto Project에서 제공

Functional Block Diagram for LGE Cleaning Robot

➤ PX30 기반 청소로봇시스템 (ROS2 on Yocto Linux) Yocto Project 기반 ROS2 를 탑재한 청소로봇시스템 개발(제어로봇시스템학회 논문지 제 27권 제1호, 54-60)



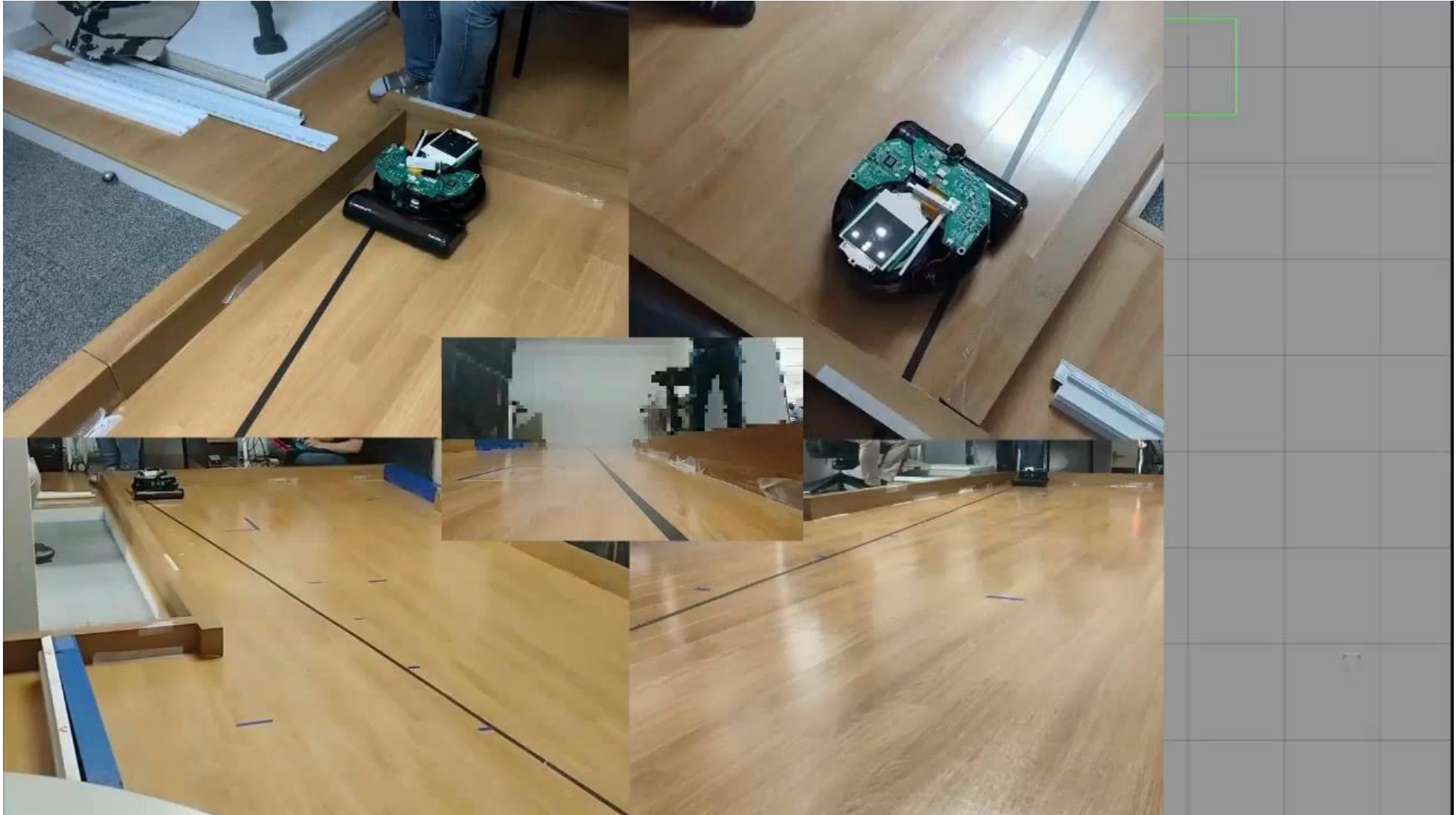
Simulator based on ROS2 for LGE Cleaning Robot

- ROS2 기반 청소로봇 Simulator의 개발환경 구성
- 모델링 적용 및 Gazebo Plugin 개발환경 적용완료
- Open Visual Odometry View 제공으로 다양한 debugging 기능제공
- SLAM/Navigation/Planning/Motion node는 Simulator로 개발하여 빠른 디버깅 및 구현 진행

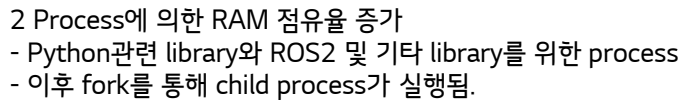
Gazebo Indoor World

Zigzag 주행 및 Simulator 검증

➤ 지그재그 주행 및 이에 따른 SLAM 동작을 Simulator로 검증함.



Python 기반 launch 방식



Python Process 제거로 1 Process 동작에 의한 RAM 점유율 절감
- ROS2 및 기타 library를 위한 process 만 동작함

	Total	Used	Free	Shared	Buff/cache	Available
Memory(KB)	986416	304268	606604	264	75544	671308
Swap	65532	0	65532			

Zigzag 주행 중 Memory 최적화 및 CPU 점유율

➤ Flash memory 사용량

- ROS2, OpenCV 최적화를 통한 code size 확보
 - ROS2 : 불필요한 package 삭제
 - OpenCV : python, binary, 불필요한 library 삭제
- 512MB Flash memory 기준 : available size 154MB 수준

➤ CPU 점유율

- Zigzag 동작 시 직선주행
SLAM : 11%
Navigation : 9%
 - CPU 점유율 User 33%, system 5%, Idle : 59%
- Zigzag 동작 시 회전상태
SLAM : 22%
Navigation : 11%
 - User 47%, system 9%, Idle : 42%

➤ Yocto 기반 ROS2 탑재에 따른 임베디스 환경에서의 성능평가 결과

Memory, CPU 측면에서 동작상 특이사항 없으며, 충분한 margin을 확보함.

맵 공유를 통한 멀티 청소 로봇 제어 : 동작구조 및 적용기술

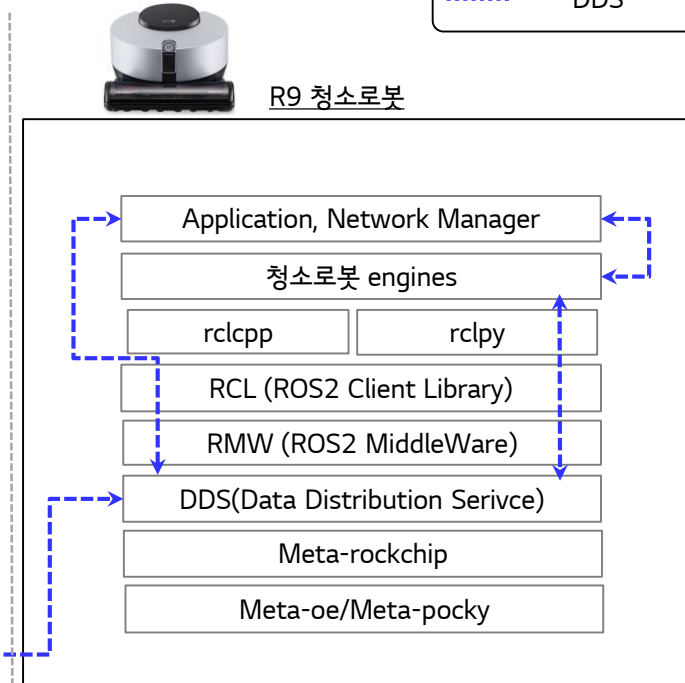
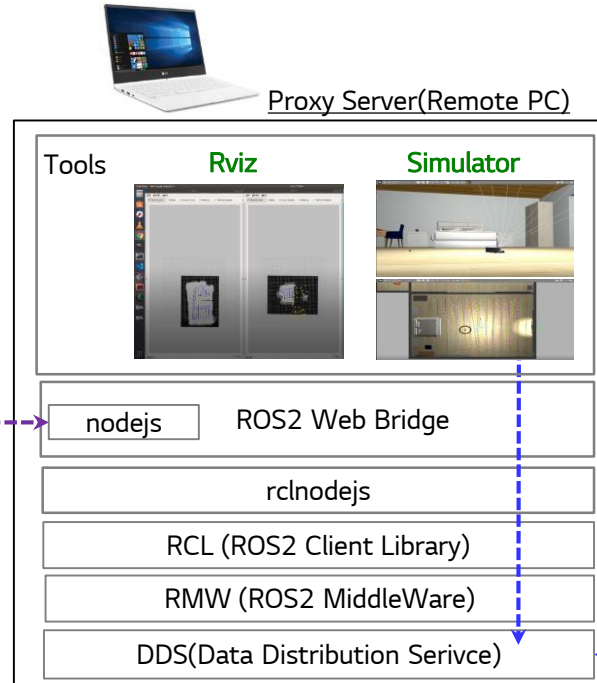
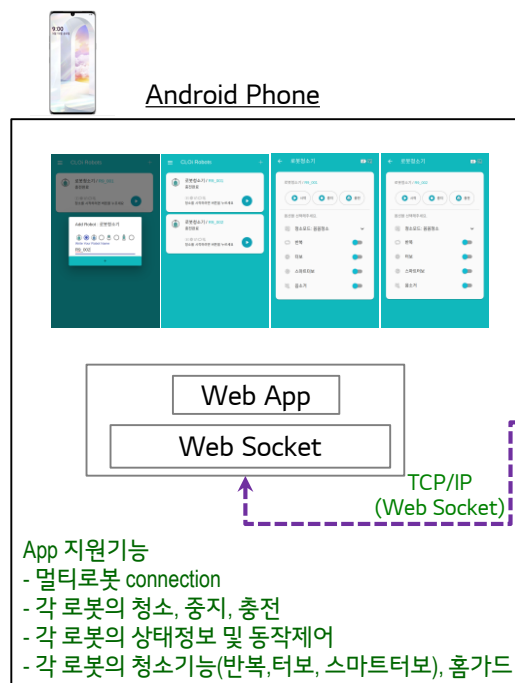
➤ Android용 Web app 기반의 map sharing을 통한 멀티로봇 검증

ROS2 기반 Map Sharing을 통한 멀티로봇 제어 (2021 제26회 제어로봇시스템학회 학술대회)

..... Socket 통신
..... DDS

동작
구조

적용
기술



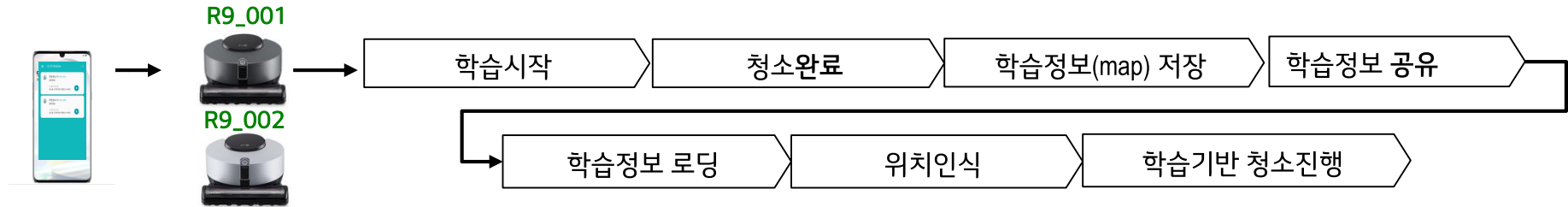
- Android용 Web App
 - Web socket 통신 기반으로 ROS2 환경설치 불필요.
 - Web App의 높은 OS 이식성 활용 가능
 - ThinQ App 리소스 활용

- 시뮬레이터 연동 검증
 - Gazebo 기반 ROS2 시뮬레이터 개발
 - 다양한 센서, 디버깅 기능 지원
- Tool 확대개발 (Rviz Visualizer)
 - ROS2 지원 tool의 확대개발
 - Gazebo 대비 저사양 PC에서 개발 가능
- Web Binder
 - Web app interface 지원하기 위한 ROS2 Web bridge 적용

- R9 청소로봇 자율주행
 - 주행모드에 따른 자율주행 **지그재그, 꼼꼼, 집중청소**
 - 장애물/사람인식에 따른 회피주행
- 성능개선
 - 저사양 memory, CPU 지원
 - Shared memory 적용으로 Network 성능확보
- 멀티로봇 제어
 - DDS 통신 기반 Map sharing 구현 및 검증
 - 협업청소를 통한 멀티로봇 제어 검증


App 기반 맵 공유를 통한 멀티 청소 로봇 제어

➤ 두 대의 가정용 청소로봇을 사용하여, 학습 및 맵 공유를 통한 협업청소 검증



TX2 배송로봇 Navigation2 주행검증

- Yocto 기반 배송로봇에 slam toolbox 및 Navigation2 (ROS에서 공개한 Open Navigation)를 적용하고, 주행검증을 완료함.
 - Slam toolbox로 Map 생성 후, Navigation2로 주행검증을 완료함.



ROS2 on Yocto
Navigation2 and Slam-toolbox

➤ 성능 (CPU, Memory)

- CPU : 최초 Localization 시 user 24% / idle 69%, 이후 주행 시 user 약 7~9% / idle 81~84% 수준
- RAM : used 634MB, free 7GB 수준
- CPU, RAM 측면에서 충분한 margin이 확보됨.

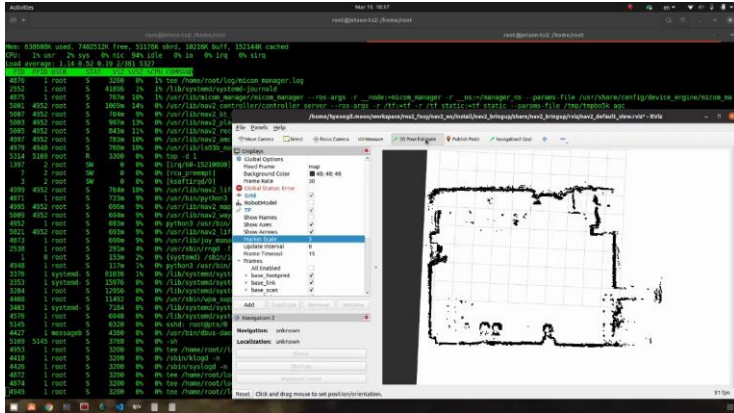
Yocto vs. Ubuntu Navigation2 주행 성능비교

NAV2 성능비교

- Yocto 및 Ubuntu 기반의 TX2 배송로봇으로 Navigation2 주행 시 성능(CPU, Memory)을 비교



Yocto 기반 TX2 배송로봇



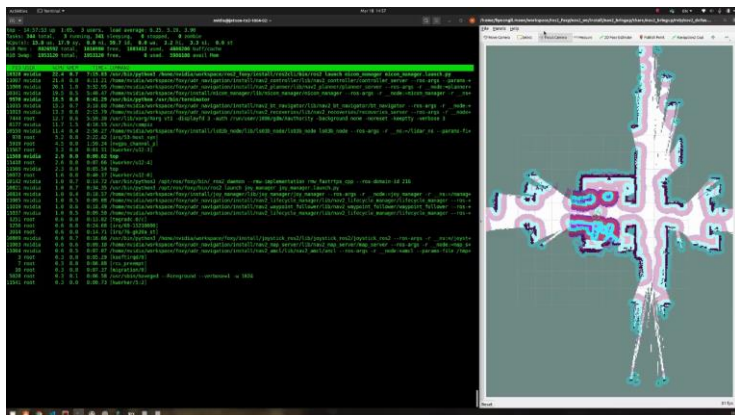
CPU 점유율

	User	System	etc	Idle
Ubuntu	20~24%	20~23%	6~7%%	46~54%
Yocto	7~10%	5~7%	2~3%	81~85%

RAM 사용량

	Used	Free	Buffer/Cache
Ubuntu	약 1.7GB	약 1.6GB	약 4GB
Yocto	약 600MB	약 7GB	약 170MB

Ubuntu 기반 TX2 배송로봇



✓CPU 점유율

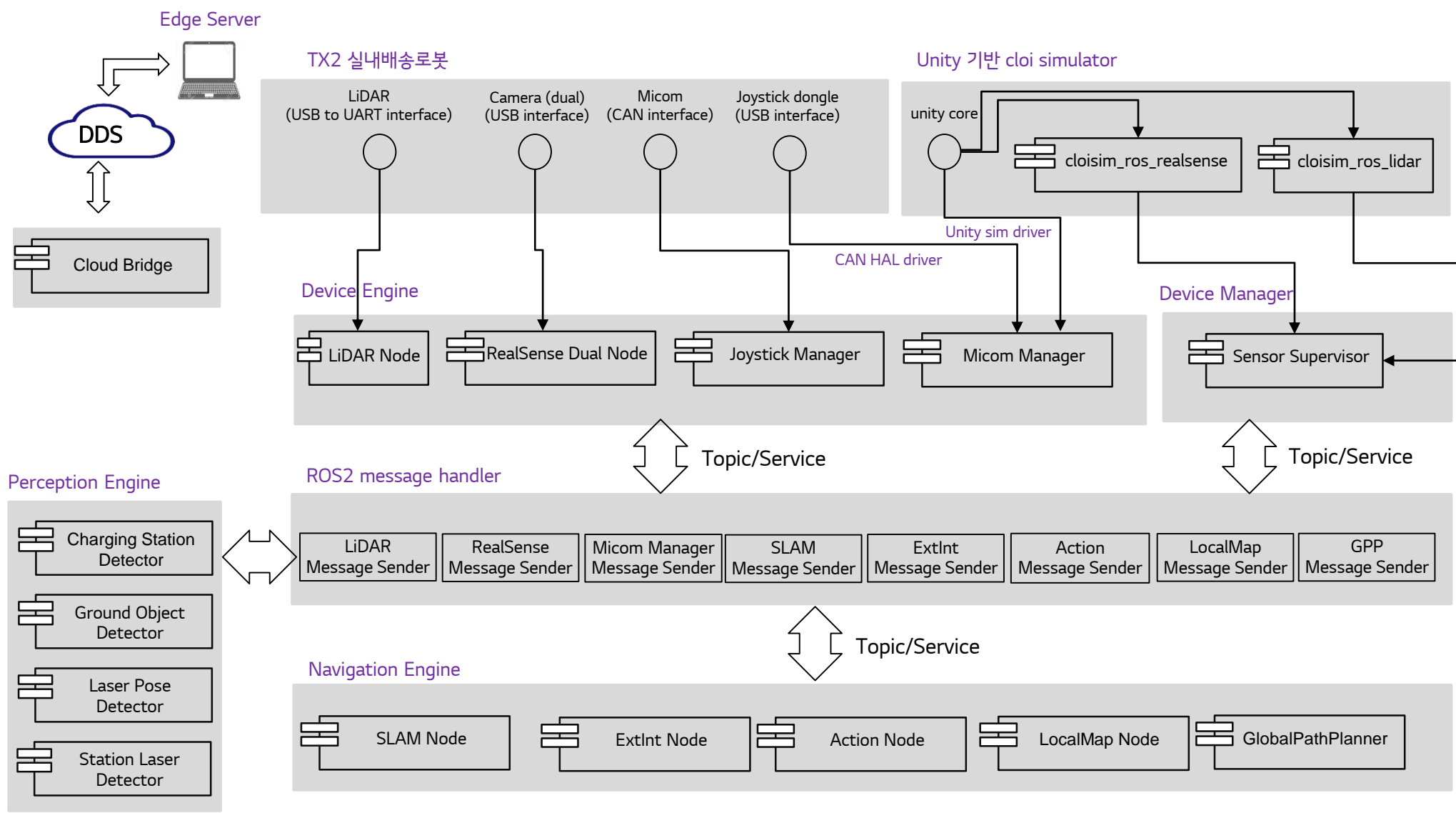
Yocto 적용 시, Ubuntu 대비 약 35% 이상 성능개선 확인

✓RAM 사용량

Yocto 적용 시, Ubuntu 대비 약 1.1GB 절감 (8GB RAM 기준)

TX2기반 ROS2 Navigation Engine 구성

➤ Functional Block Diagram for Delivery Robot



LiDAR기반 실내배송로봇 주행검증

➤ 실내배송로봇의 실기검증 진행

- LiDAR 기반 LGE Navigation의 Mapping 및 Move to Goal을 검증함
- CPU 점유율 : Mapping 시 25~28%, 주행 시 30~45% 수준(평균 약 40%)
- RAM 사용량 : 1.05GB 수준 (8GB RAM 기준)

ROS2 on Yocto

LG Navigation Mapping and Move to Goal

결론 및 향후 계획

➤ 결론

- Yocto Project 기반 ROS2 탑재 및 최적화 진행 (청소로봇, 배송로봇)
- Yocto 적용에 따른 HW 확장성 및 SW 이식성 확보
- ROS2 최적화를 통한 저사양 Flash Memory, RAM 적용 가능성 검증완료
→ 저사양의 임베디드 시스템 상용화 가능성 검증완료 (최소사양 Flash memory 512MB, RAM 512MB)
- Navigation2 주행 검증을 통한 Ubuntu vs. Yocto의 성능비교
- LGE 배송로봇의 LiDAR 기반 주행엔진 구현 및 검증

➤ 향후 계획

- 실내 배송로봇 실증, 성능 및 신뢰성 확보
- 클라우드 연동을 통한 멀티로봇 제어 구현 및 검증
- 로봇엔지 off-loading을 통한 클라우드 기반 Thin Client 로봇제품 개발
- ROS2에서 제공하는 Open package 적용을 통한 benchmark test 진행
- ROS2 Middleware(RCL¹⁾, RMW²⁾, DDS³⁾) Patch 검토 및 반영을 통한 성능(CPU, Memory, Latency) 개선 지속 진행

1) ROS Client Library : 다양한 프로그래밍 언어(c, c++, python, java 등)를 지원하는 클라이언트 라이브러리, 2) ROS MiddleWare : 여러 DDS vendor를 지원하기 위한 추상화 인터페이스 제공,

3) Data Distribution Service : ROS에서 채택한 분산환경을 위한 데이터 중심의 통신표준

➤ YOCTO PROJECT LONG TERM SUPPORT ANNOUNCED <https://www.yoctoproject.org/yocto-project-long-term-support-announced/>

- 릴리스에 대한 지원을 확장하는 계획 발표
- 새로운 지원계획은 초기 2년 동안 적용되며 Yocto Project 3.1(Dunfell)부터 지원됨. (기존 1년)
- Dunfell의 주요 변경점
 - ① Kernel version 5.4로 변경 (기존 4.14)
 - ② Python2 제거
 - ③ Improvements in the build equivalence mechanism

➤ Release History

<https://wiki.yoctoproject.org/wiki/Releases>

Codename	Yocto Project Version	Release Date	Current Version	Support Level	Poky Version	BitBake branch
Kirkstone	3.5	April 2022		Future	27.0	
Honister	3.4	October 2021		Planning	26.0	
Hardknott	3.3	April 2021		Stable	25.0	1.50
Gatesgarth	3.2	Oct 2020	3.2.1	EOL	24.0	1.48
✓ Dunfell	3.1	April 2020	3.1.3	Long Term Support	23.0	1.46
Zeus	3.0	October 2019	3.0.4	Community	22.0.3	1.44
Warrior	2.7	April 2019	2.7.4	EOL	21.0	1.42
Thud	2.6	Nov 2018	2.6.4	EOL	20.0	1.40
Sumo	2.5	April 2018	2.5.3	EOL	19.0	1.38
Rocko	2.4	Oct 2017	2.4.4	EOL	18.0	1.36
Pyro	2.3	May 2017	2.3.4	EOL	17.0	1.34
Morty	2.2	Nov 2016	2.2.4	EOL	16.0	1.32
Krogoth	2.1	Apr 2016	2.1.3	EOL	15.0	1.30
Jethro	2.0	Nov 2015	2.0.3	EOL	14.0	1.28
Fido	1.8	Apr 2015	1.8.2	EOL	13.0	1.26
Dizzy	1.7	Oct 2014	1.7.3	EOL	12.0	1.24