Embedded Success

Introduction to ROS and ROS Blockset

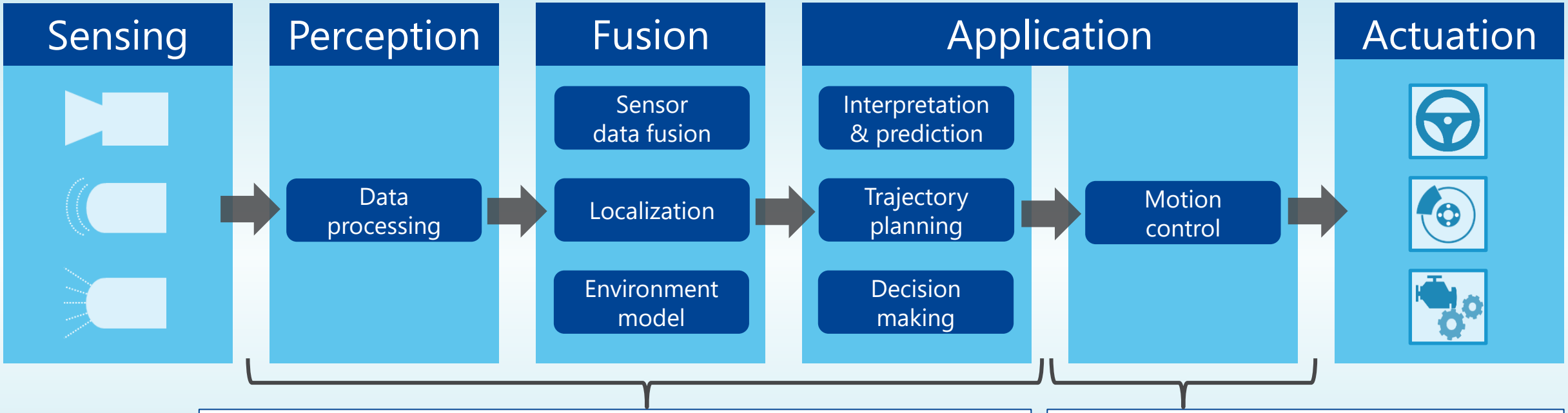
Kensuke Araki

2021.9.16

Agenda

1. Introduction to ROS/ROS2
2. Differences to RTMaps
3. ROS Blockset

ADAS/AD Software Stack



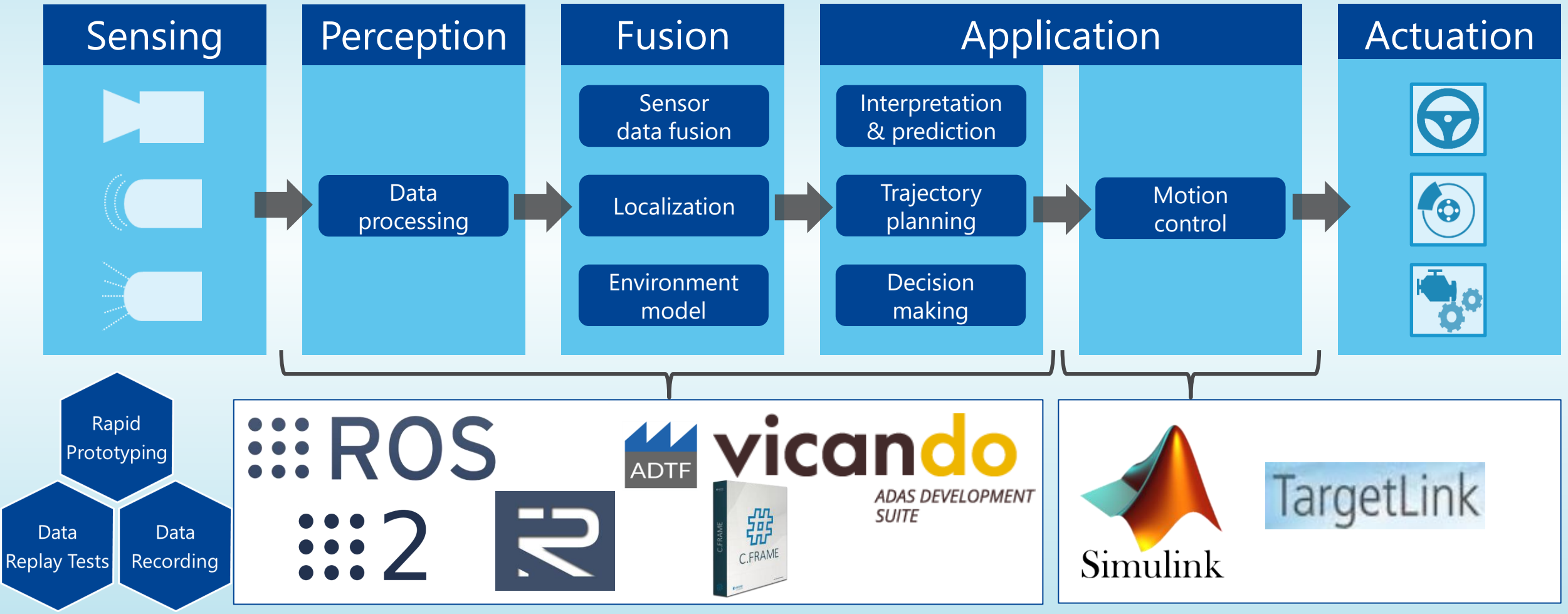
High performance data processing

- C/C++, NVIDIA Cuda, Python and DNNs[1)]
- Sensor interfaces and heterogeneous data streams

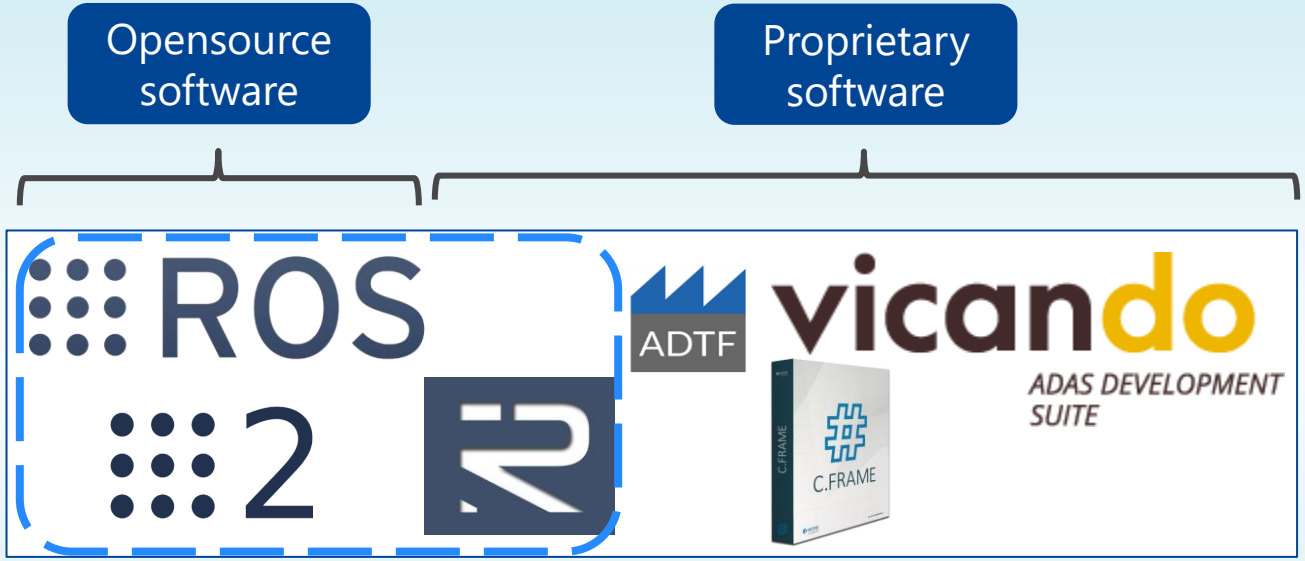
Real-time processing

- Functional safety
- Automotive Ethernet, CAN FD

ADAS/AD Software Stack



Multi-Sensor Software



Multi-Sensor Software

Main :

- **RTMaps from Intempora/dSPACE**
- **ROS from Open Source Robotics Foundation**
- **EB Assist ADTF from Elektrobit/Continental, Germany**

Other :

- Vicando from Zuragon, Sweden
- C.FRAME from CMORE, Germany
- vADASdeveloper from Vector Informatik, Germany
- Polysync from Harbrick, USA, and Automated Driving Systems Toolbox for Matlab from Mathworks

ROS- Robot Operating System

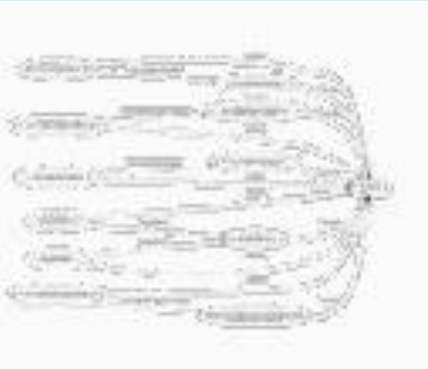
- Is it an operating system ?
- Is it free ?
- What about the market share ?
- Is ROS2 the new hype ?

ROS- Robot Operating System

- A free open source framework for algorithm development
- Contains multiple components sharing the same clock
- Data triggered processing of heterogeneous data streams

ROS- Robot Operating System

Communication

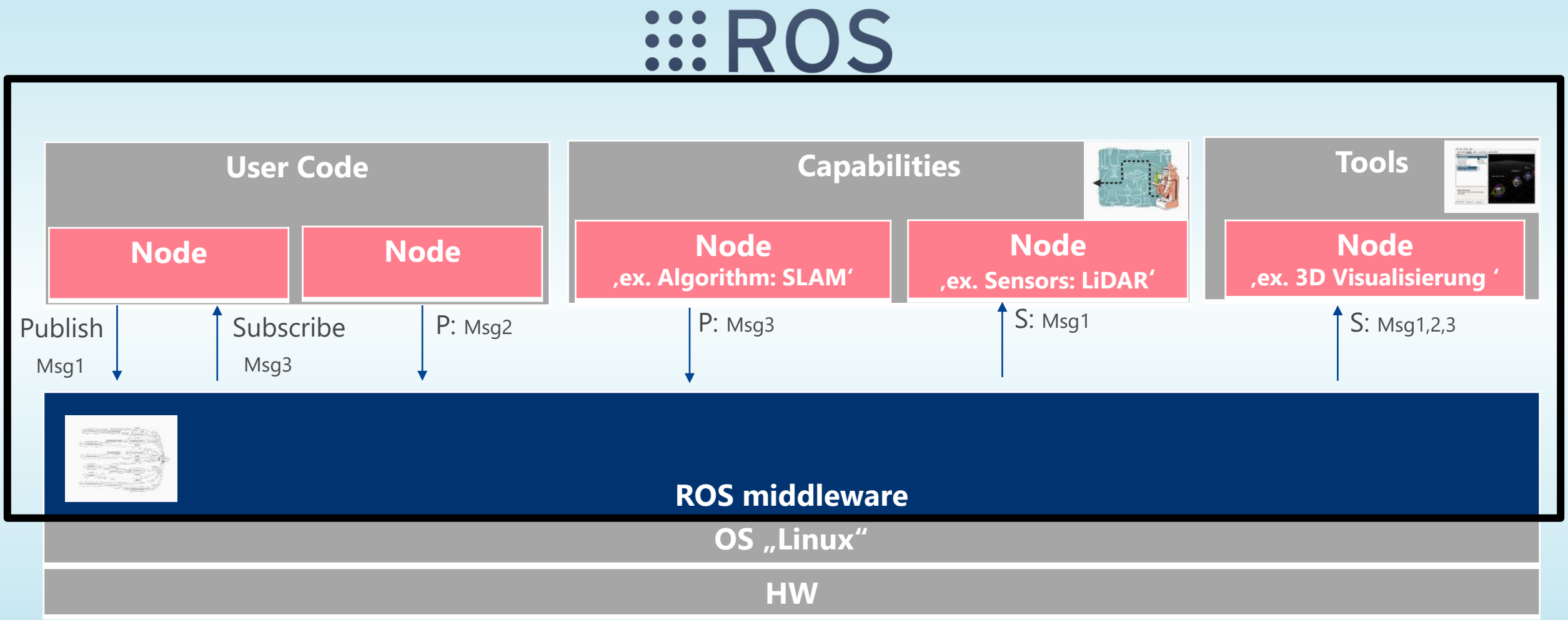


Tools

Capabilities

Ecosystem

ROS- Robot Operating System



ROS Eco-System

- High market share
  - More than a decade in use “2007”
  - Quasi-standard in universities and labs
  - Widely used in ADAS/AD prototyping

RTMaps versus ROS – Simple example: Edge detection

Image capture

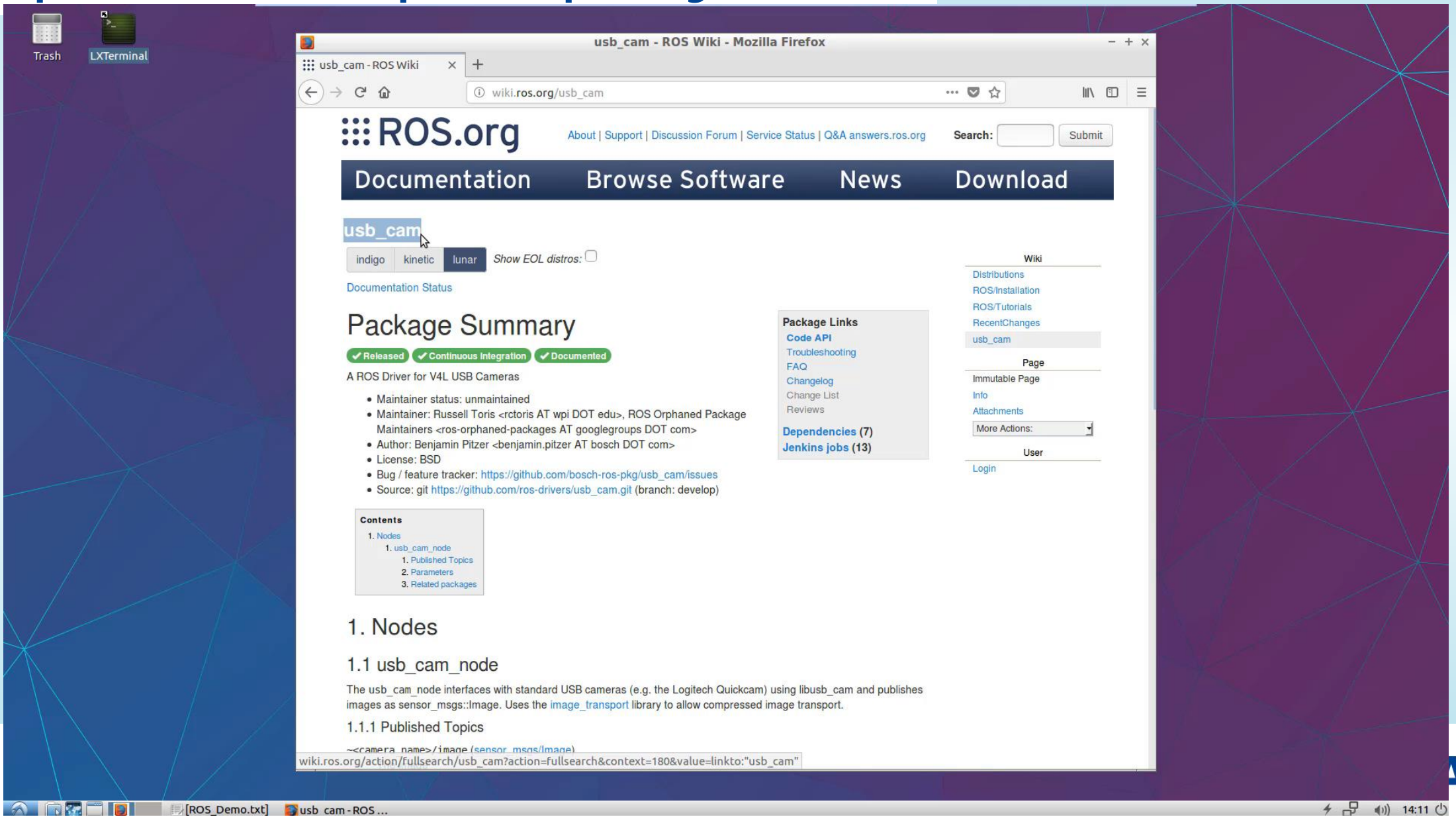
Data conversion
+ edge detection

OpenCV

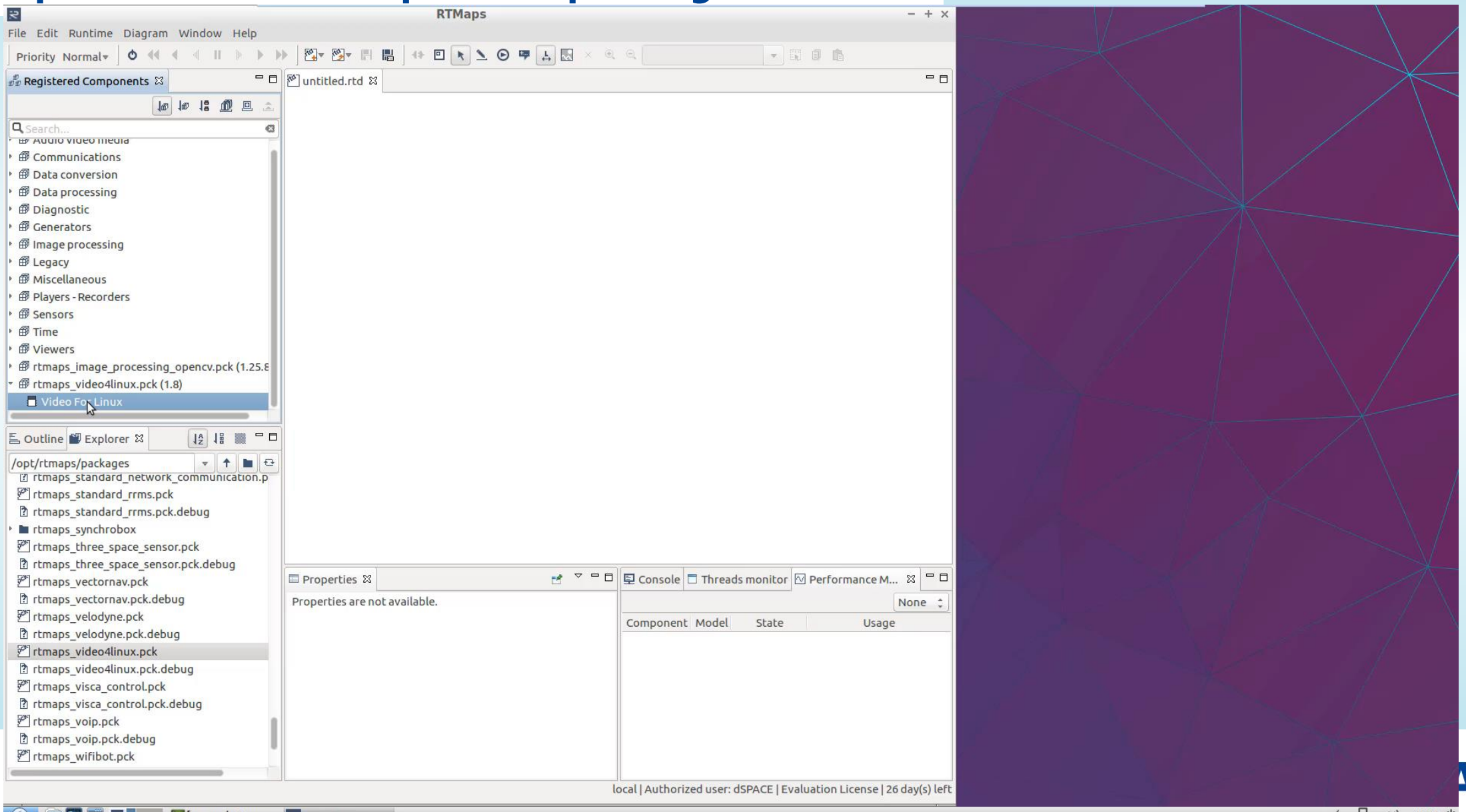
Image view (source)

Image view (processed)

RTMaps versus ROS – Simple example: Edge detection



RTMaps versus ROS – Simple example: Edge detection

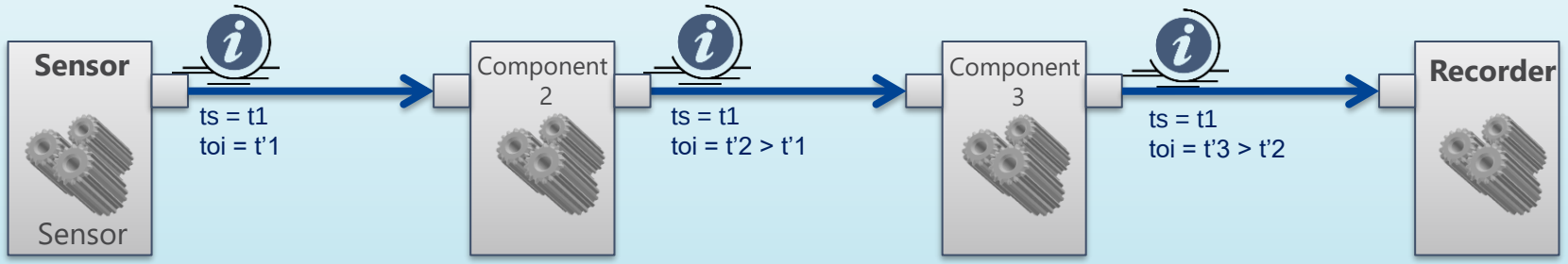


RTMaps versus ROS – Summary of main points

Criterion	RTMaps	ROS
Performance	**Very good** • Inter-thread communication via shared memory • Pre-allocated buffers, no memory copies • No serialization/deserialization • **Up to 30% performance gain**	**Poor** • Communication between nodes via IP stack (UDP/TCP) • Dynamic memory allocation and memory copies • Serialization/Deserialization
Ease of use	**Very good** Graphical interface, component (model) based development	**Good for Linux developers, poor for others** No graphical interface for design but « launch files » (XML files…)
Debugging of application	**Easy** Step by step debugging, pausing execution at break points etc. possible as entire application is typically in one process	**Difficult due multiple processes** Pausing execution of entire application with multiple processes (ROS nodes) difficult
Quality	**Good**	**Good for ROS core, but not guaranteed for other ROS components** Consequence: For productive projects the code of ROS components would have to be reviewed -> expertise required, difficult under project pressure

RTMaps versus ROS – Summary of main points

Criterion	RTMaps	ROS
Data recording and accuracy of sensor data time stamps	**Very good** Two "times" are recorded: 1) Time-stamp ts when sensor data is captured 2) Time-of-issue toi when output data of a component is written	**Moderate** Time stamp is taken whenever the recorder node receives data (not when data was captured or fed to the recorder). Due to serialization/deserialization and IP communication between nodes there is an intrinsic delay.
Replay of big data	**Very good** High performance "Random access".	**Moderate** - Low performance "no random access". - Bags are divided into chunks. Retrieving those chunks' addresses for replay can be time consuming.
Price	Commercial tool, high price	Free of charge (open-source)



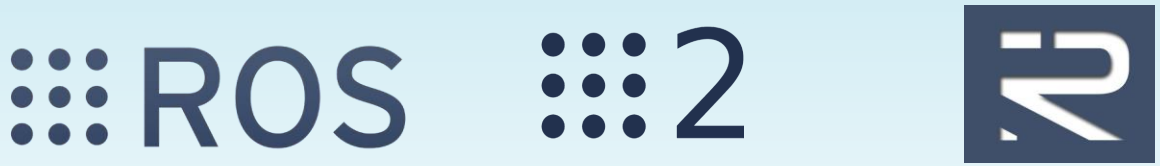
ROS2

- Keeping the good and solving the bad issues
  - Performance upgrade

    - Standardized DDS communication between nodes.
    - Nodes can now be implemented as threads also.
    - Shared memory “unique pointers” communication.
    - Python based launch and not xml based.

- No existing fully stable version till November 2019.

ROS vs ROS2 vs RTMaps



RTMaps is 5-10 times better than ROS2* in terms of

CPU utilization „load“

Loss-less data transmission

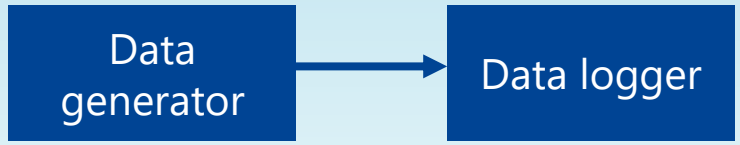
Benchmarked with ROS2 Dashing Diademata

Benchmarking ROS2 vs RTMaps

- **Software**
  - ROS2: Dashing Diademata
  - RTMaps: Version 4.6.0
- **OS**
  - Ubuntu 18.04.3 LTS
- **Hardware**
  - PC: Alienware Area 51 R5
  - Processor: Intel(R) Core(TM) i7 7800X
    - 6 cores / 12 virtual cores, at 4.0GHz
    - 8.25MB Cache
  - RAM: 16 GB (2 x 8 GB), DDR4-Speicher, 2.666 MHz
  - Memory: M.2 PCIe NVME SSD

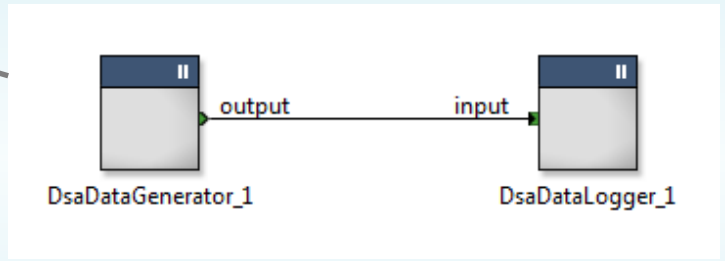
Middleware performance

- **Constant**: Data is generated/sent and logged for a 20 second period.
- **Variable**: Generated vector size and output period.



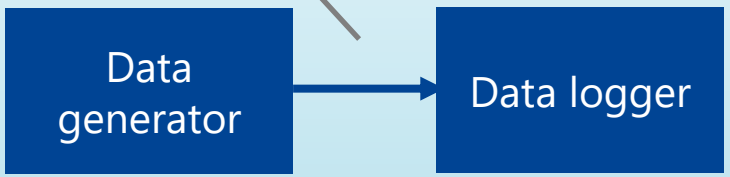
RTMaps

Component to component

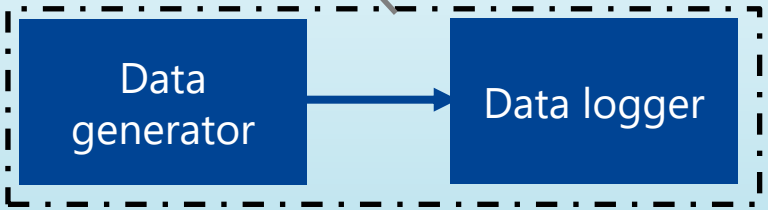


ROS2

Node to node



Composed in a single node



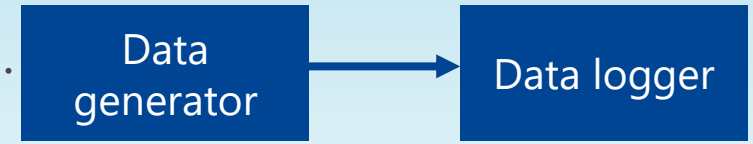
For ROS2:

Default QoS setting is used.

use_intra_processs_comm = 1

Middleware performance

- **Constant**: Data is generated/sent and logged for a 20 second period.
- **Variable**: Generated vector size and output frequency.



6 MByte vector

20 Byte vector

CPU load „100% = 1 core“

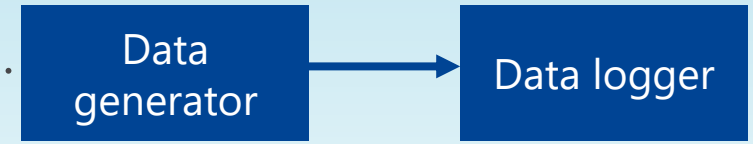
Output frequency	ROS composed	ROS nodes	RTMaps
30 Hz	74 %	84%	4%
66 Hz	91 %	143%	8%
100 Hz	102 %	161%	12%
5 kHz			
10 kHz			
20 kHz			

Data loss = 95%
Delay = 5s

Delay = 7s

Middleware performance

- **Constant**: Data is generated/sent and logged for a 20 second period.
- **Variable**: Generated vector size and output frequency.



Output frequency	6 MByte vector	20 Byte vector		
		CPU load „100% = 1 core“		
		ROS composed	ROS nodes	RTMaps
30 Hz				
66 Hz				
100 Hz				
5 kHz				
10 kHz				
20 kHz				

Delay = 8s

Data loss = 11%

106 %	93%	20%
155 %	231%	29%
193 %	268%	44%

Important Information!

Embedded Success

RTMaps at Indiana University

Indiana University – Purdue University Indianapolis (IUPUI) is researching ways to improve road transport safety for autonomous applications by analyzing the benefits of high-speed sensor data processing. RTMaps Embedded and NXP BlueBox are serving as the core real-time execution platform for embedded computing capabilities.

"RTMaps Embedded is designed to face and win multisensor challenges. It provides us with an efficient and easy-to-use framework for fast and robust developments in areas such as advanced driver assistance systems, automotive vehicles, and robotics."

Professor Mohamed El-Sharkawy, Purdue School of Engineering and Technology

Source: dSPACE Magazine 02/2019

RTMaps at P3

P3 has developed an Autonomous Data and Analytics Platform for Testing (ADAPT) to help customers evaluate the implementation of features for ADAS and autonomous driving. These include vision-based features and features for testing sensors as well as sensor configurations and algorithms. ADAPT leverages RTMaps software to verify and validate ADAS and algorithms for autonomous driving.

P3's ADAPT demo vehicle features a data logging PC with RTMaps software and a set of sensors (lidar, ultrasound, forward-looking camera, forward-looking radar).

"We are very pleased with the capabilities offered by RTMaps. The performance was seamless and logging was reliable."

Modar Horani, Managing Principal of Systems Engineering at P3 North America

Source: dSPACE Magazine 01/2018

RTMaps at NAVYA

World's first driverless production vehicle

The NAVYA ARMA is considered to be the world's first driverless production vehicle for regular traffic. The autonomous shuttle bus can carry up to 15 passengers and reaches a speed of 45 km/h. Fifty electrically operated vehicles have already been deployed for passenger transport worldwide. As of September 2017, they have safely transported 180,000 passengers. NAVYA is a French company specializing in designing electric, autonomous systems. For developing the complex autonomous driving functions, NAVYA relies on the multisensor development environment RTMaps from Intempora.

www.navya.tech

"The performance and ease of use of RTMaps have played a decisive role in prototyping, testing, and benchmarking perceptual and data fusion algorithms for our level-5 autonomous driving system."

Pascal Lecuyot, R&D Director - Perception, NAVYA

Source: dSPACE Magazine 02/2017

RTMaps – Customers