Eclipse eCAL (enhanced communication abstraction layer)

SDV Contribution Day – June 2022

www.continental.com
About myself

› 2022 – present
  › Head of SDF Development Platform, Innovation Line Driverless, ADAS

› 2017 – 2022
  › Team Lead, Base Software Development and Integration, R&D SW Engineering

› 1997 – 2017
  › Middleware development for AD systems (eCAL)
  › Rapid prototyping HMI development
  › Anti-lock braking system for Electro-Hydraulic-Brake systems
  › Various other research projects ..

› 1997
  › Diploma Electrical Engineering, Technical University Dresden / Germany
eCAL (enhanced Communication Abstraction Layer) is a fast publish-subscribe middleware that can manage inter-process data exchange as well as inter-host communication.

https://continental.github.io/ecal/
Motivation
Autonomous Driving challenged us ..

- high-performance computer systems needed
- new sensor technologies introduced
- large quantities of data must be transmitted extremely fast
- software components may run on different processor cores
- software components may run on different domain controllers
- software components may run on different operating systems
- all data flows needs to be monitored, recorded and finally analyzed
What about existing solutions?
## What about existing solutions?

<table>
<thead>
<tr>
<th>2015 and earlier</th>
<th>Today</th>
</tr>
</thead>
<tbody>
<tr>
<td>Robotic Operation System ROS 1</td>
<td>Robotic Operation System ROS 2</td>
</tr>
<tr>
<td>- bad overall performance for AD systems</td>
<td>- simple, nice API as ROS 1</td>
</tr>
<tr>
<td>- no Windows support</td>
<td>- powerful, flexible RMW concept</td>
</tr>
<tr>
<td>Data Distribution Service (DDS) implementations</td>
<td>integrates DDS implementations</td>
</tr>
<tr>
<td>- slow inter-process communication</td>
<td>- shared memory support</td>
</tr>
<tr>
<td>- high costs</td>
<td>- open-source alternatives</td>
</tr>
<tr>
<td>- complex build / configuration / API</td>
<td>- Windows supported</td>
</tr>
<tr>
<td>- no Windows support</td>
<td></td>
</tr>
</tbody>
</table>
What distinguishes eCAL from ROS 2?
Customized for autonomous driving

› eCAL is message protocol agnostic
  › different protocols, different use cases
  › message schema evolution well supported

› eCAL is a library
  › minimalistic API
  › easy to build / extend / configure

› eCAL has powerful tools
  › live data monitoring based on dynamic protocol reflection + plugin concept for 2D / 3D
  › distributed recording concept – unique selling point 😊
Architecture overview
Architecture overview

› supports **POSIX** as well as **Windows** operating systems

› supports different transport protocols
  
  › inter-process communication: **shared memory**
  
  › inter-host communication: **udp multicast / tcp**

› supports **different serialization formats**:
  
  › google::protobuf
  
  › capnproto
  
  › google::flatbuffers, messagepack, json ..

› supports **publish / subscribe** and **client / server** pattern

› **brokerless dynamic design**
Architecture overview

- language bindings for C, C++, C#, Python, Rust, Go, M-Script, Simulink

- shipped with **eco system tools** for
  - **live monitoring** of all software component interfaces
  - orchestrated, **distributed message recording**
  - **message replay** real-time or stepwise
  - automated software **component start, stop** and supervising
  - all tools realized as command line and GUI application

- **open sourced** by Continental under Apache 2 license since 2019

[GitHub Link](https://github.com/continental/ecal)
Typical use cases
Autonomous Vehicle communication stack

HMI
AD Lane Detection
AD Vehicle Localization
AD Driving Functions

Camera / Radar Interfaces
CAN Gateways
Backend Interface

eCAL Live Data Monitor

Recording Srv.
Recording Srv.
Recording Service

eCAL runtime

Windows 10
QNX
Ubuntu 18.04
Ubuntu 18.04
Ubuntu 20.04

Dev. Control PC
Speedgoat
NVIDIA Tegra A
NVIDIA Tegra B
MultiCore HPC
Software component validation

- AD Lane Detection
- AD Vehicle Localization
- AD Driving Functions

- eCAL Player
- eCAL Monitor
- Image + Lane

- Windows 10
- Ubuntu 20.04

- Dev. Control PC
- MultiCore HPC
Summary

› eCAL is designed for autonomous driving applications
› eCAL combines modern communication patterns with state-of-the-art message protocols
› eCAL has powerful tools for rapid prototyping
› eCAL is open source since 2019 and looking forward to be part of the Eclipse family 😊

Thank you for your attention
Backup
Architecture overview (technical)

User Land
- C++
- C
- Python
- Simulink
- Rust
- Go

eCAL API + Tools
- Communication Pattern, Discovery, Language Bindings
- Monitor, Record, Replay, Automate

Message Layer
- Google Protobuf, Google Flatbuffers, Cap’nProto, JSON...

Transport Layer
- UDP Multicast / TCP / Shared Memory

OS Layer
- Windows / Linux / QNX / macOS

HW Layer
- X86 / AMD64 / ARMv8/9
eCAL and friends

› OSS projects using eCAL
  › ROS2 middleware plugin RMW_ECAL – https://github.com/continental/rmw_ecal
  › Mathworks Simulink toolbox – https://github.com/mathworks/ecal-toolbox
  › Agtonomy Trellis hybrid autonomy agriculture vehicles – https://github.com/agtonomy/trellis
  › Generic Foxglove Studio Visualization interface (part of next eCAL OSS release) – https://foxglove.dev/

› OSS projects used by eCAL (the bigger ones)
  › google protobuf – https://developers.google.com/protocol-buffers
  › hdf5 hierarchical data format – https://www.hdfgroup.org/solutions/hdf5/
  › asio c++ – https://think-async.com/Asio/
  › fineftp-server – https://github.com/continental/fineftp-server
  › tcp_pubsub – https://github.com/continental/tcp_pubsub
  › iceoryx (optional shm layer) – https://github.com/eclipse-iceoryx/iceoryx
Local IPC Performance (Q2/2022)