Continental
The Future in Motion

eCAL

enhanced Communication Abstraction Layer

https://github.com/continental/ecal

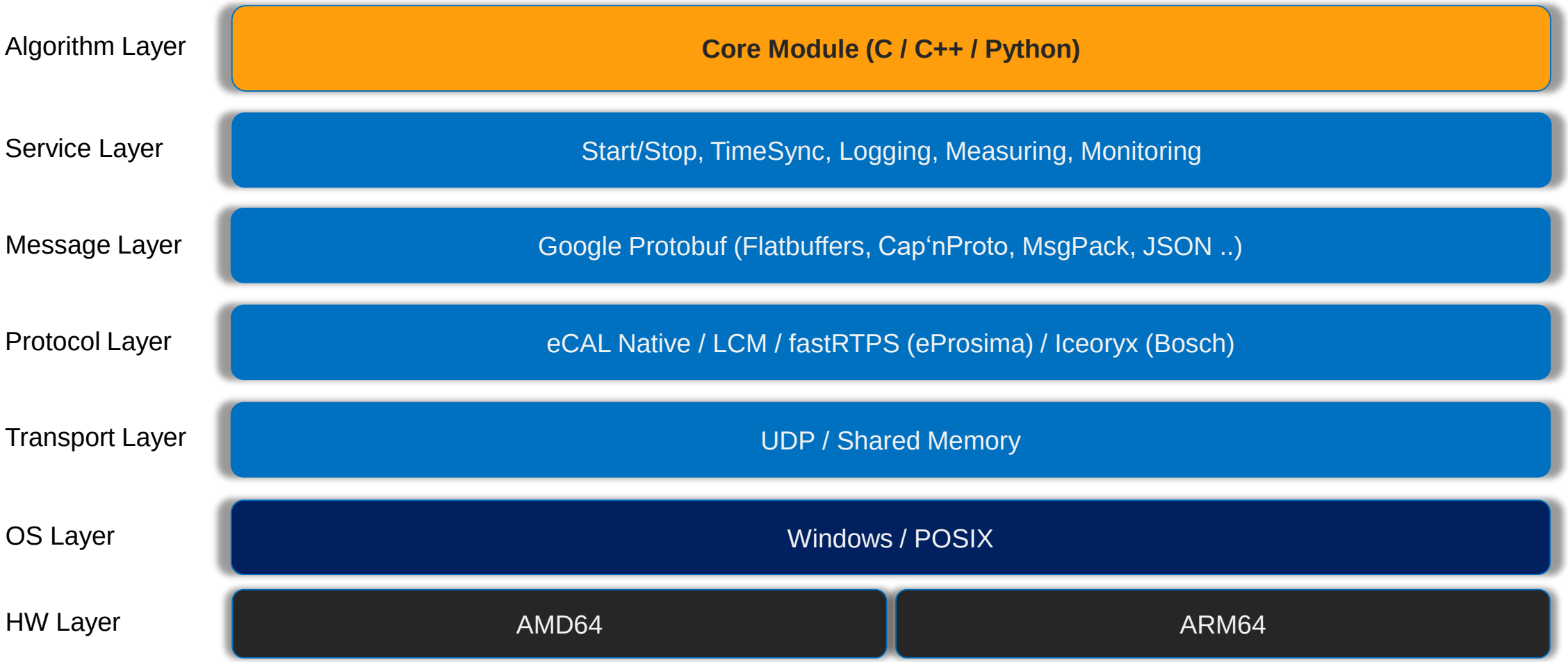
Overview

- eCAL == enhanced Communication Abstraction Layer
- middleware for high performance and scalable inter-process communication on single computational nodes or in heterogeneous networks
- designed for minimal latency and high data throughput
- lightweight API for message transport only
- operates on a wide range of hardware platforms from high end server machines to ARM based embedded hardware
- easy integration in different computing languages and frameworks

Features

- loose connection of all components via publish / subscribe pattern
- all participants synchronize all information automatically, no central demon instance
- different transport layers (inner-process, shared memory, udp unicast/multicast, lcm, rtps)
- support a subset of quality of services (depends from transport layer mode)
- native support of different serialization formats:
  - google::protobuf (monitor reflection supported)
  - capnproto (monitor reflection supported)
  - google::flatbuffers, messagepack, json ..
- application eco system:
  - eCALMon: monitoring interface for real-time diagnostic and message debugging
  - eCALRec: recording distributed in an eCAL network or on a central host
  - eCALPLay: message replay with modern user interface or via command line

Architecture



Transport Layers

- inner process
  - ultra fast, reliable, single threaded, single process
- shared memory
  - ultra fast, none reliable, highest throughput for 1 to n scenarios, multi threaded, multi process
- udp multicast (can use multiple multicast groups for data transport)
  - performance depends on ethernet stack, none reliable, single threaded, multi process / hosts
- google:lcm (http://lcm-proj.github.io/)
  - udp multicast based (single multicast group), none reliable, single threaded, multi process / host
- fastRTPS (http://www.eprosima.com)
  - dds standard - supports QOS, multi process / host (currently multicast transport only)
- Iceoryx (https://github.com/eclipse/iceoryx)
  - Bosch zero copy shared memory transport layer (ipc only)

Serialization support

- Binary
  - `eCAL::CPublisher`
- String
  - `eCAL::CStringPublisher<std::string>`
- Google:Protobuf https://developers.google.com/protocol-buffers/
  - `eCAL::CProtoPublisher<GoogleProtobufType>`
- Google:Flatbuffers https://google.github.io/flatbuffers/
  - `eCAL::CFlatPublisher<flatbuffers::FlatBufferBuilder>`
- CapnProto https://capnproto.org/index.html
  - `eCAL::CCapnpPublisher<capnp::MallocMessageBuilder>`
- Message Pack http://msgpack.org/
  - `eCAL::CMsgPackPublisher<CAddress>`
- JSON
  - `eCAL::CProtoDynJSONSubscriber`

Applications

eCALMon

monitors all eCAL entities

monitors the internal state of the whole eCAL cloud

central logging target for all eCAL participants

live data preview for
- raw payload
- string payload
- protobuf messages
- CapnProto messages

plugin interface for customer data visualization

Qt

eCALRec

record local and cloud messages

record local and cloud CAN messages

HDF5 measurement format includes time stamps, data clock, data payload and data description

scenario tagging

reliable rpc interface based on ASIO service

Qt coming soon ☺

eCALPlay

measurement replay

stepwise / intervall replay

scenario playlist

replay with / without frame dropping

command line application with interactive mode

Qt

Thank you
for your attention!