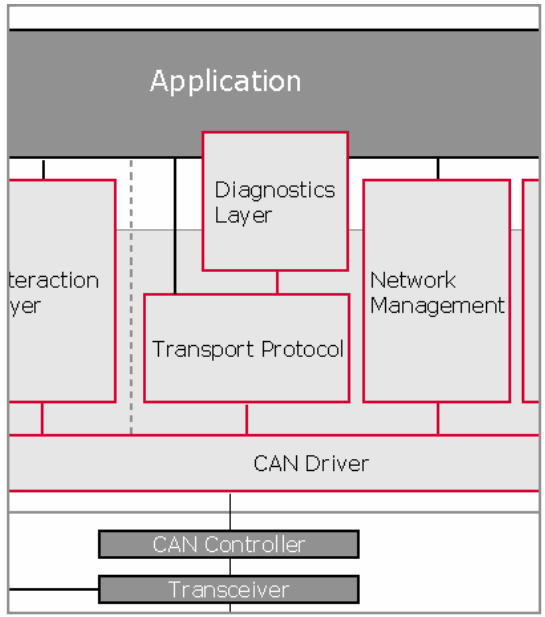
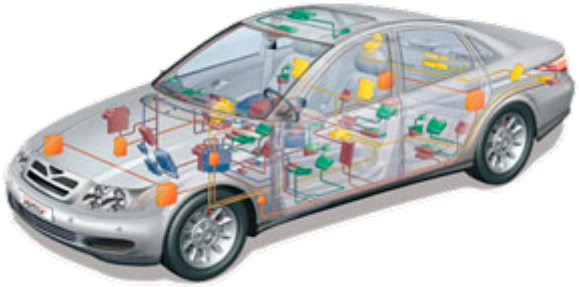


CANbedded

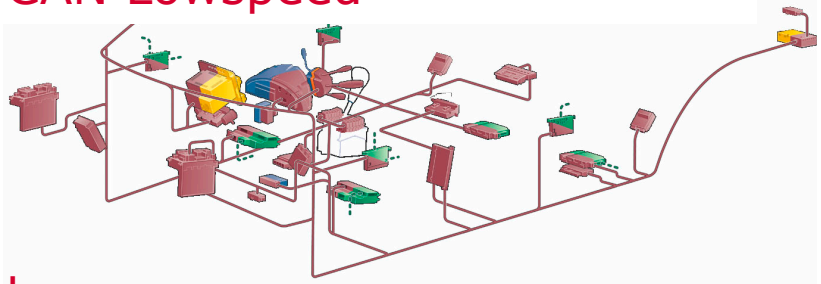
Embedded Software for Automotive Applications

Vehicle with different bus systems

(CAN Highspeed, CAN Lowspeed, LIN, FlexRay, **MOST** …)



e.g. CAN Lowspeed:
Many ECUs participate in the CAN Lowspeed



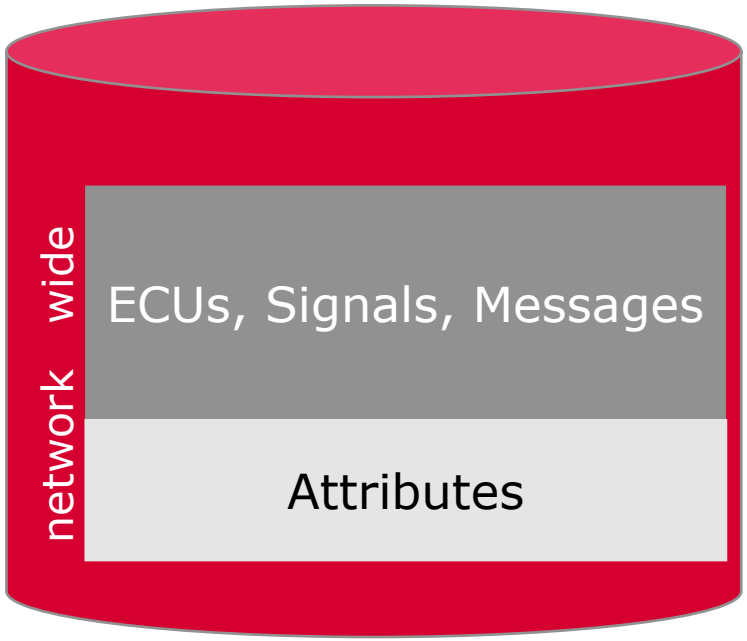
For communication the ECUs need:

Physical connection >> bus system

Logical connection >> data base file **>>**

Introduction

The data base file is the basic element for ECU communication



fixed
defined by the OEM

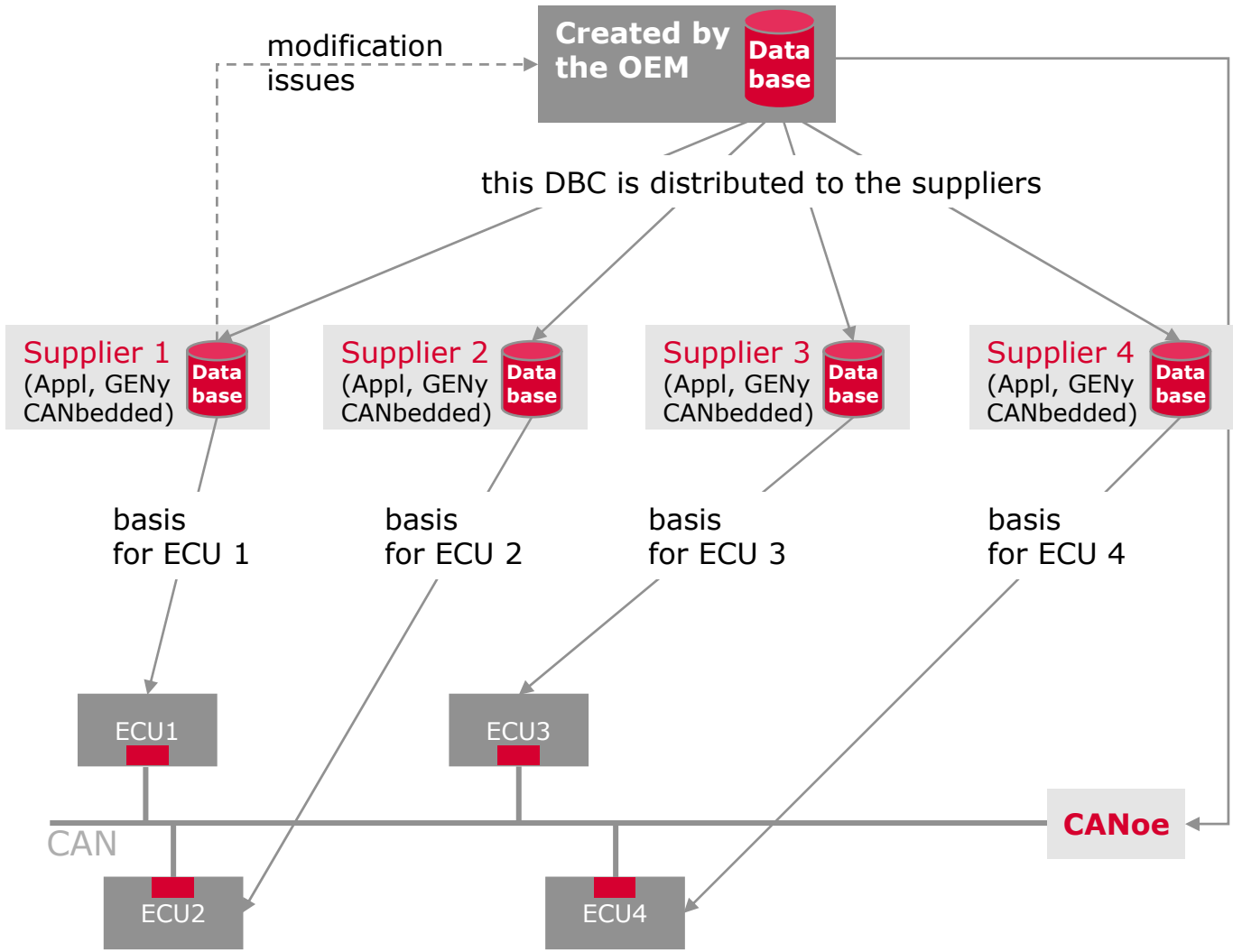
e.g. DBC, FIBEX, LDF

flexible
defined by the OEM
and Vector

ECU-specific
settings done by suppliers
with configuration tool
GENy/CANgen

ECU1	ECU2	ECU3
Assertions, Polling, Flags, Functions...	Assertions, Polling, Flags, Functions...	Assertions, Polling, Flags, Functions...

Introduction

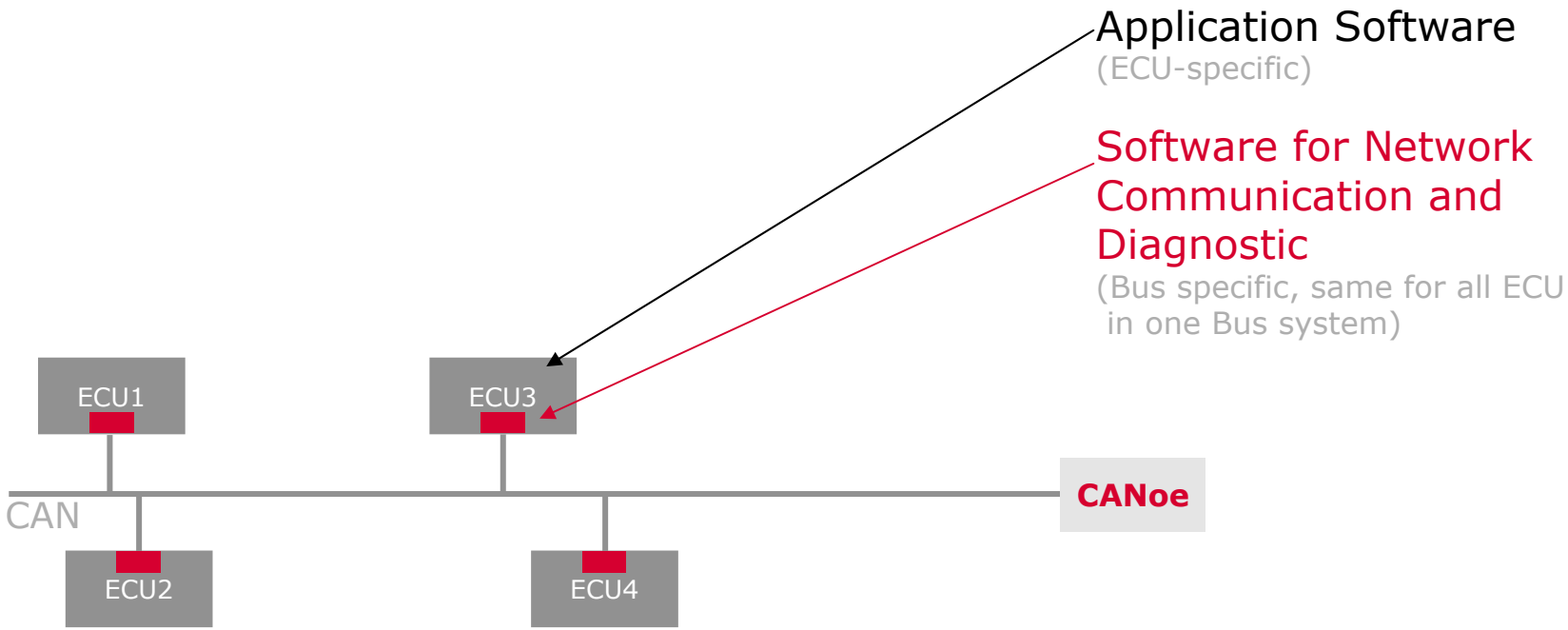


Introduction

Almost the same communication / diagnostic tasks for all ECUs

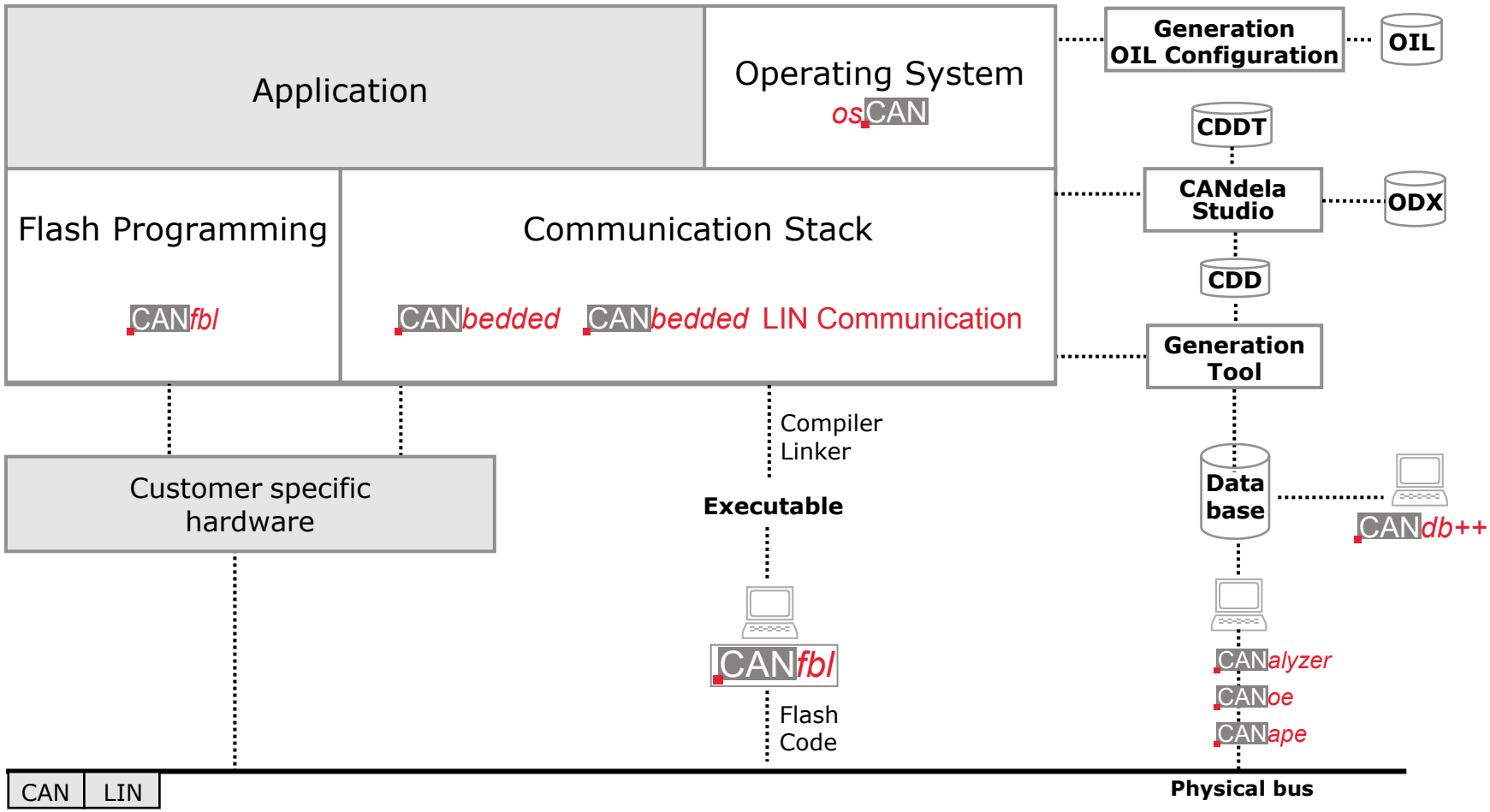
- Save precious development time for your core application
- Avoid developing already existing solutions

>> use Standard Software Components

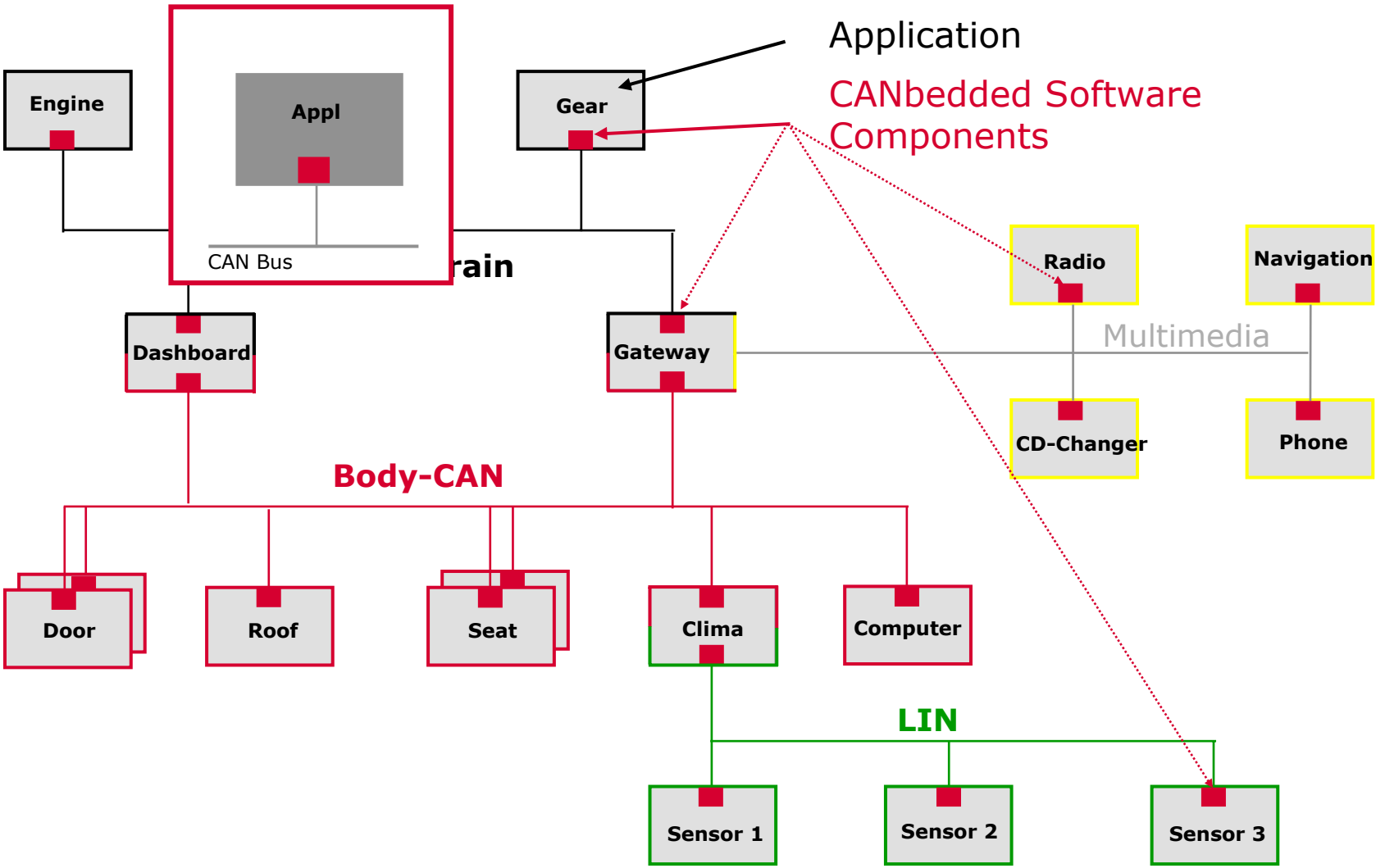


Tool Chain – Software Components

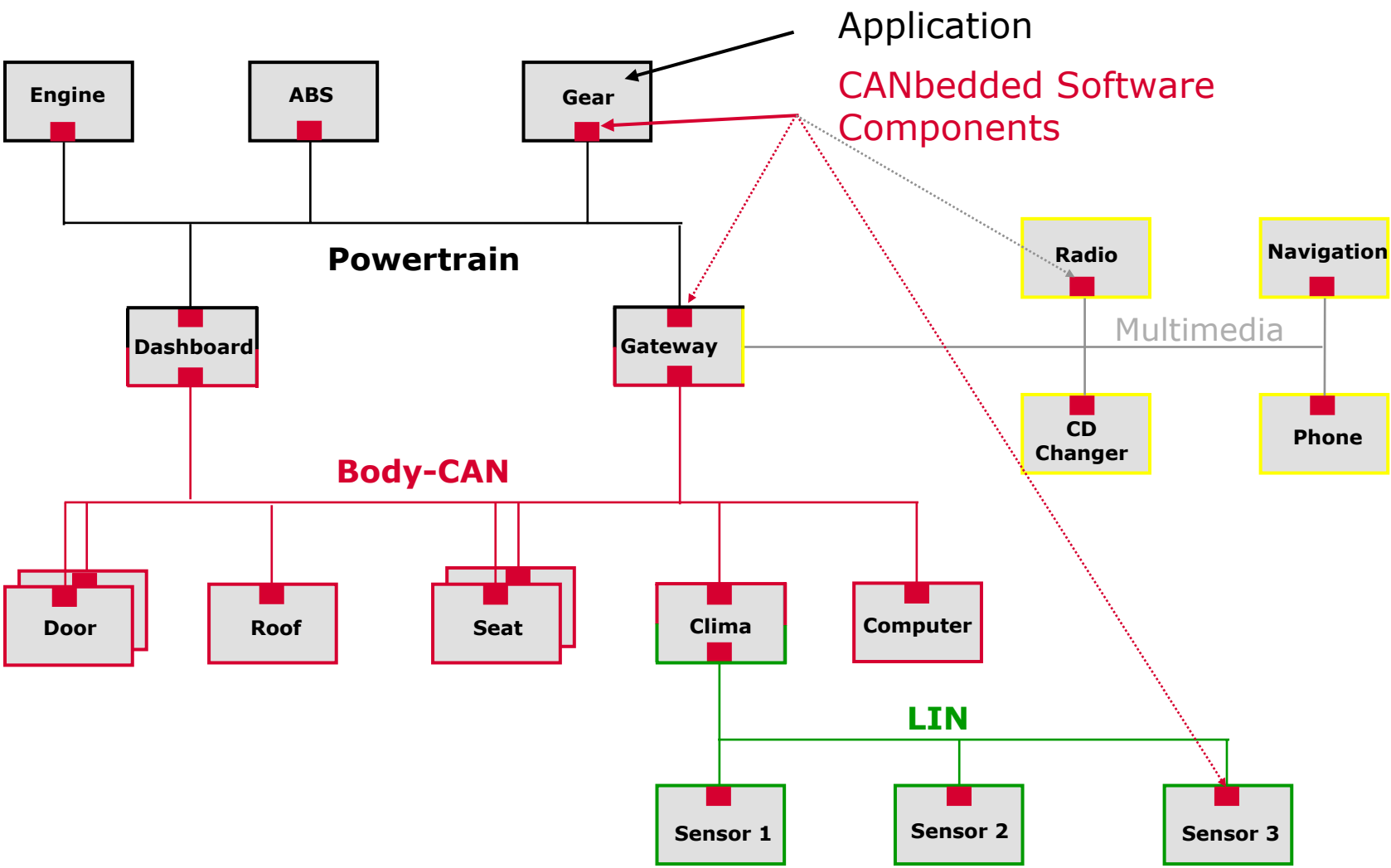
Your Task - Vector's Solutions



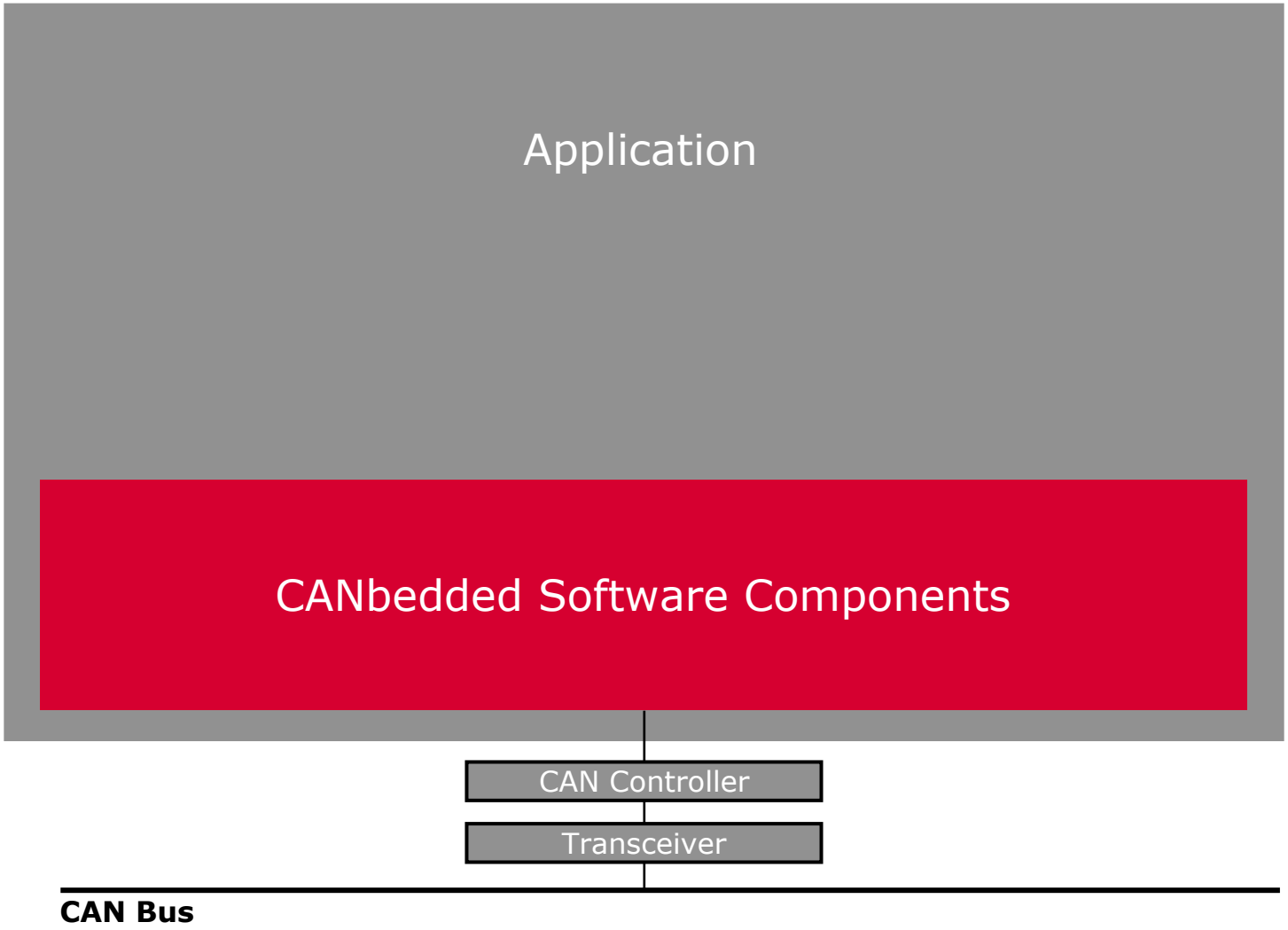
Any ECU needs Communication Components



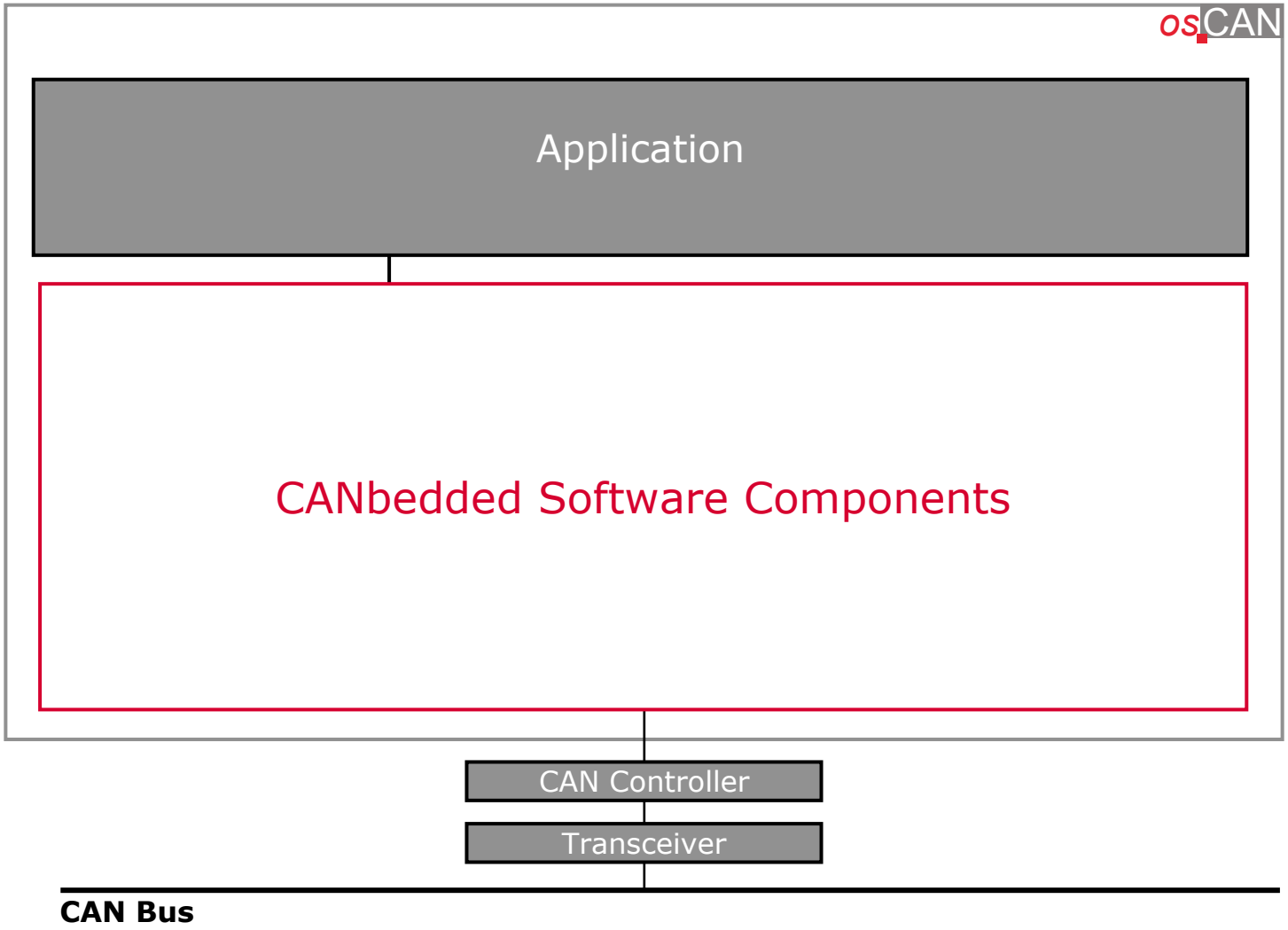
Any ECU needs Communication Components



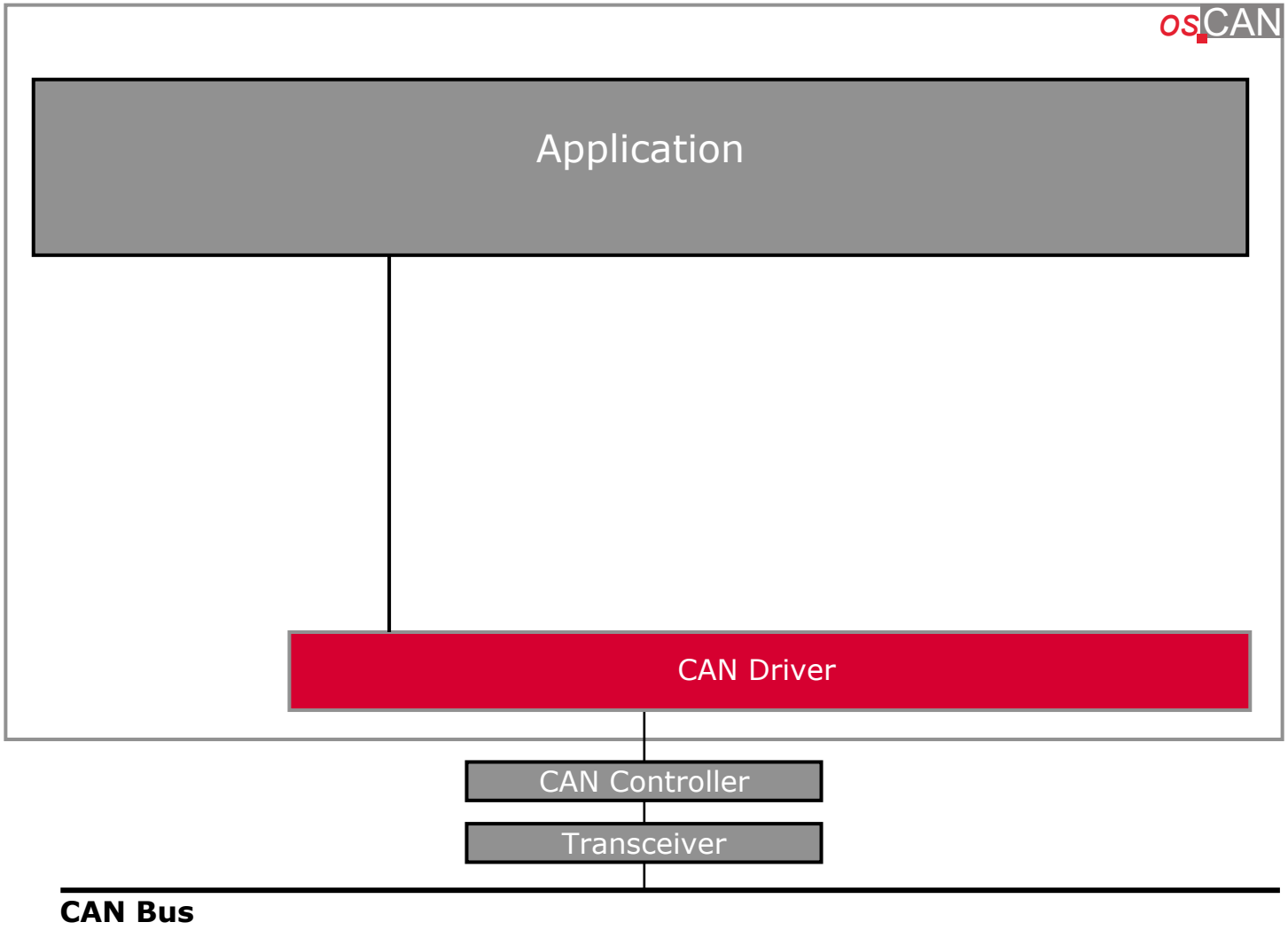
Inside The ECU



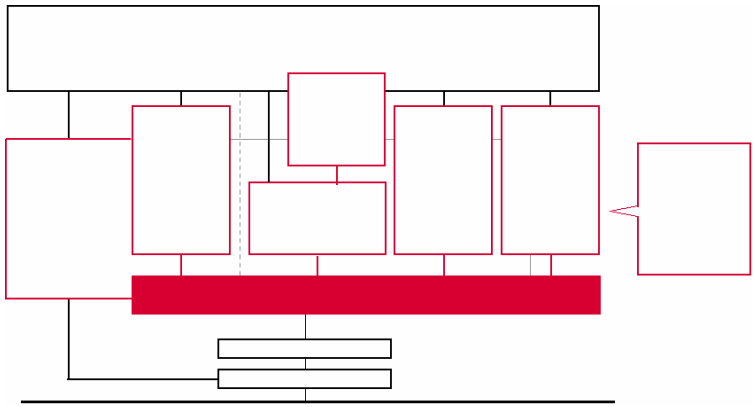
Inside The ECU



CAN Driver

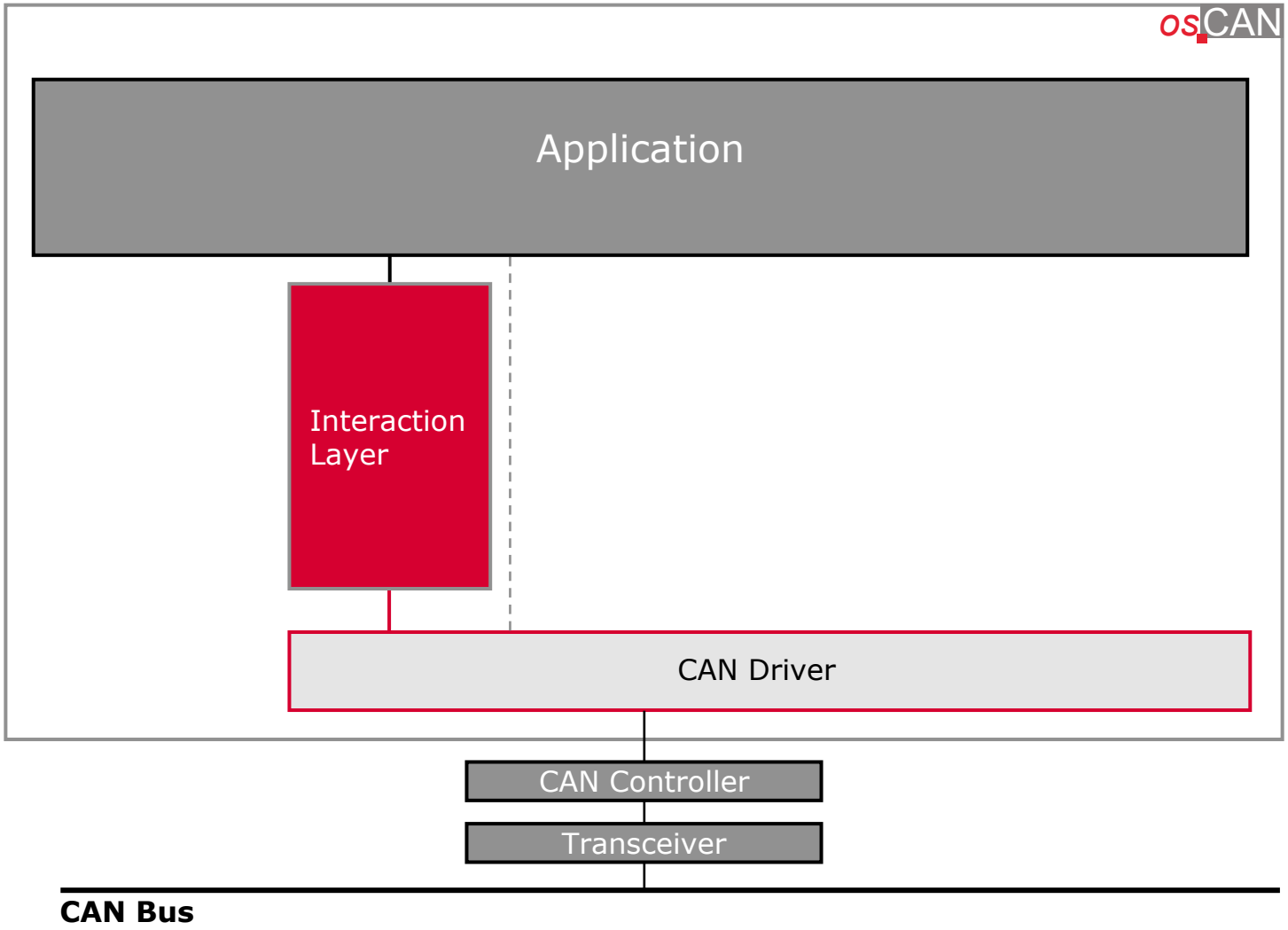


Handling of Hardware Specific CAN Chip Characteristics and Provision of a Standardised Application Interface

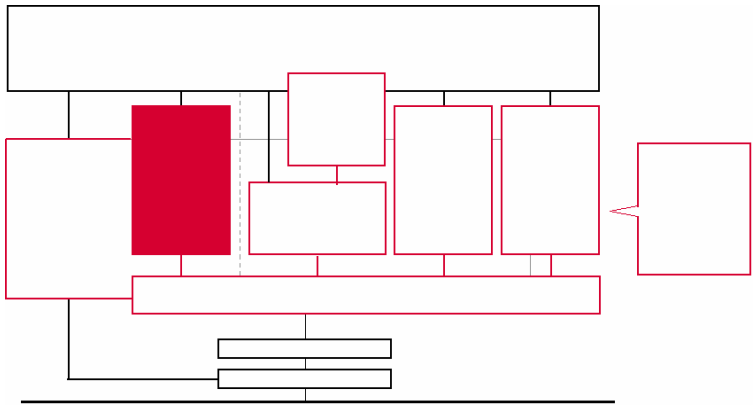


- Initialisation
- Transmission and Reception of Messages with Data- and Functional Interface
- Data- and Functional Notification
  - Indication (Rx)
  - Confirmation (Tx)
- Overrun and Error Handling
- Wakeup Detection
- Efficient Search Algorithms for Software Acceptance Filtering

Interaction Layer

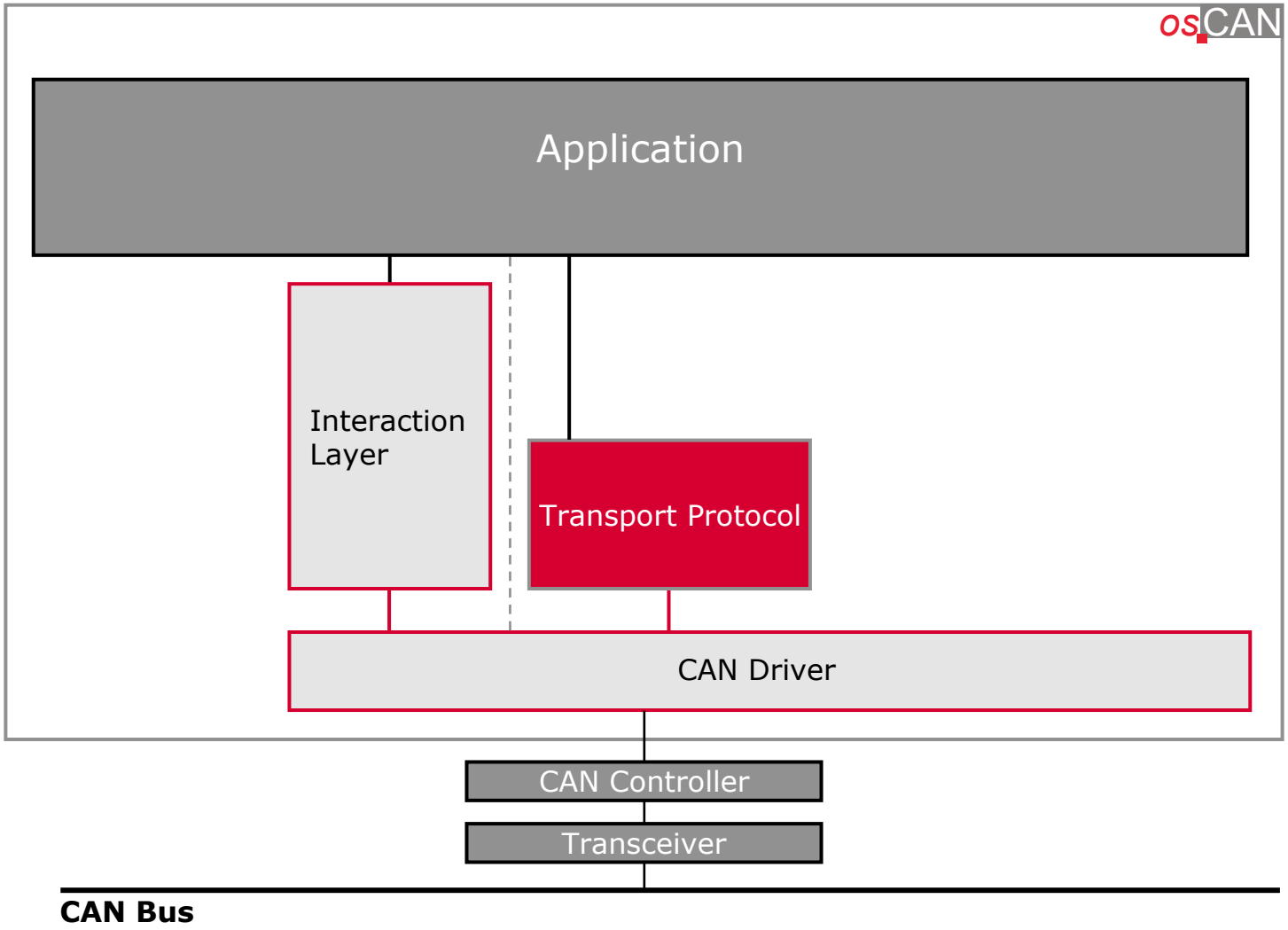


Interaction Layer with Signal Interface

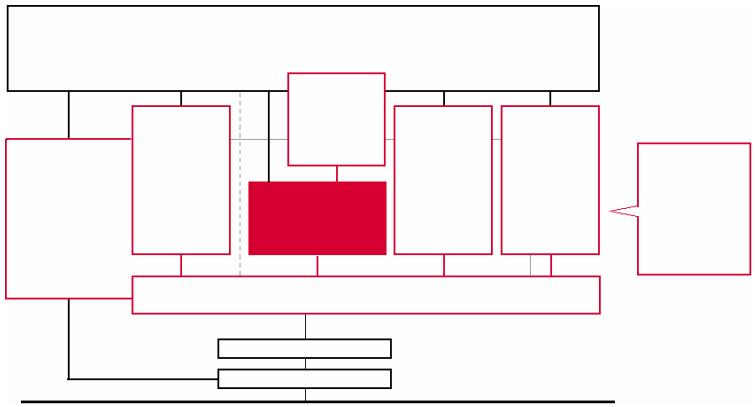


- Sending of Messages According to the Specified Transmission Types
- Checking of Minimum Distances Between Transmit Messages
- Monitoring of Receive Messages
- Setting of Default Values
- Ensuring of Data Consistency
- Signal Oriented Application Interface for Data Exchange and Notification

Transport Protocol

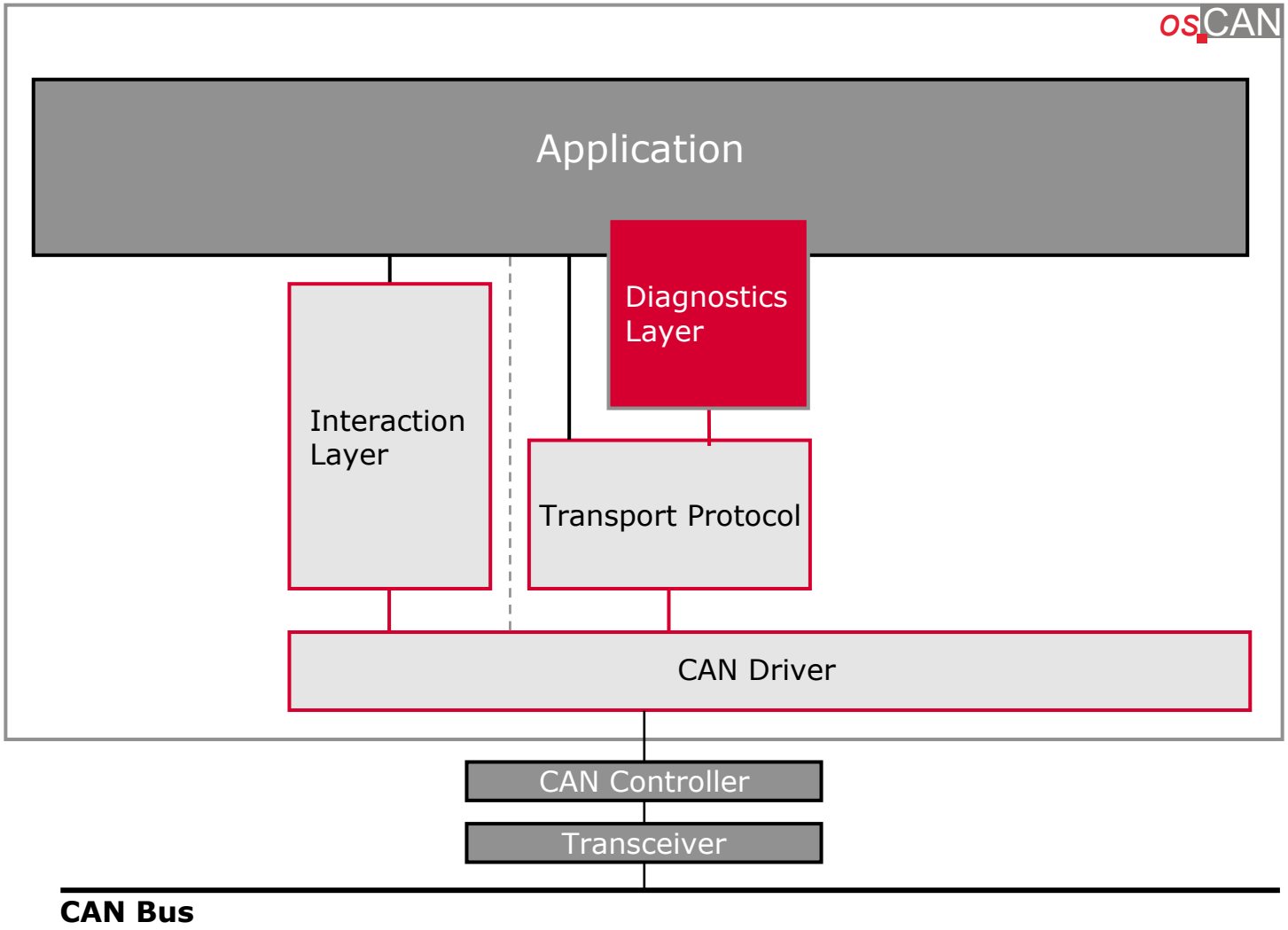


Transport Protocol for Data Exchange of Data Link Layer Independent Information

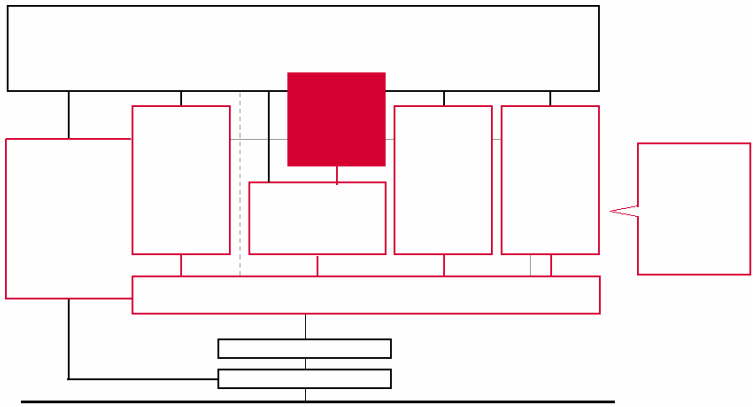


- Segmentation of Data in Transmit Direction
- Collection of Data in Receive Direction
- Exchange of Communication Parameters
- Control of Data Flow with Synchronisation of Transmission and Reception
- Detection of Errors
  - Message Loss
  - Message Doubling
  - Message Sequence
- Additional Addressing Information (Normal, Extended)

Diagnostics Layer

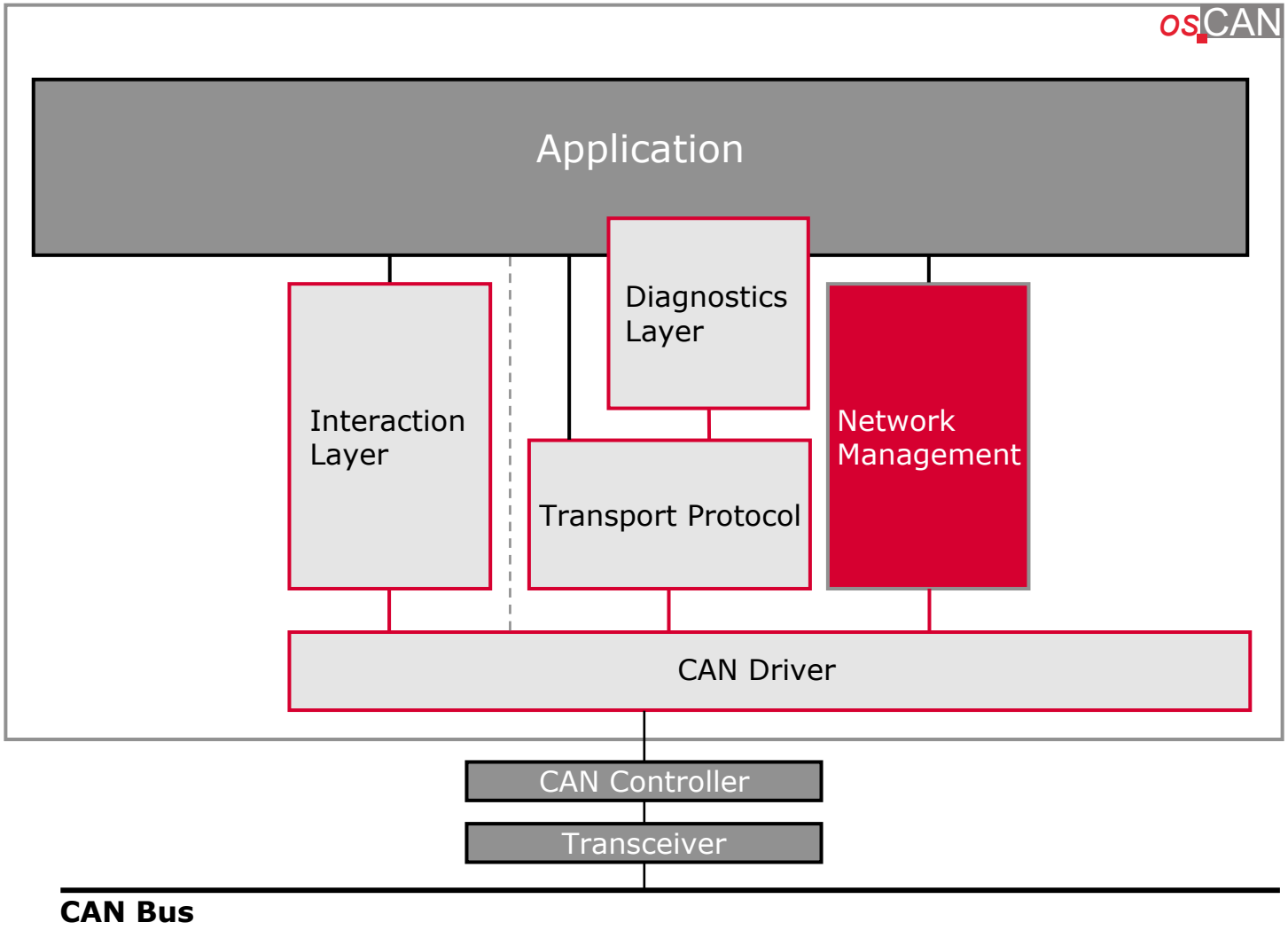


Diagnostics Layer According to ISO14229 / ISO14230 (Keyword Protocol 2000)

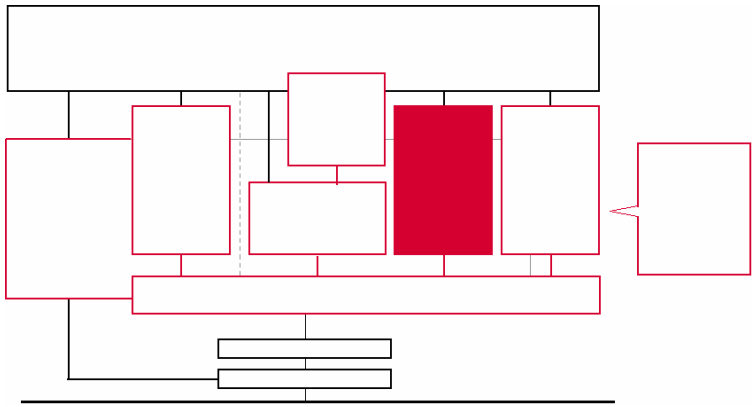


- Functional Interface for Diagnostic Services
- Direct Processing of CAN Specific Diagnostic Requests (Enable/Disable Normal Message Transmission)
- Negative Responses (e.g. Service not Available)
- Exception Handling (e.g. Busy, Request Pending)
- Address Handling (Detection of Response Service Identification)

Network Management

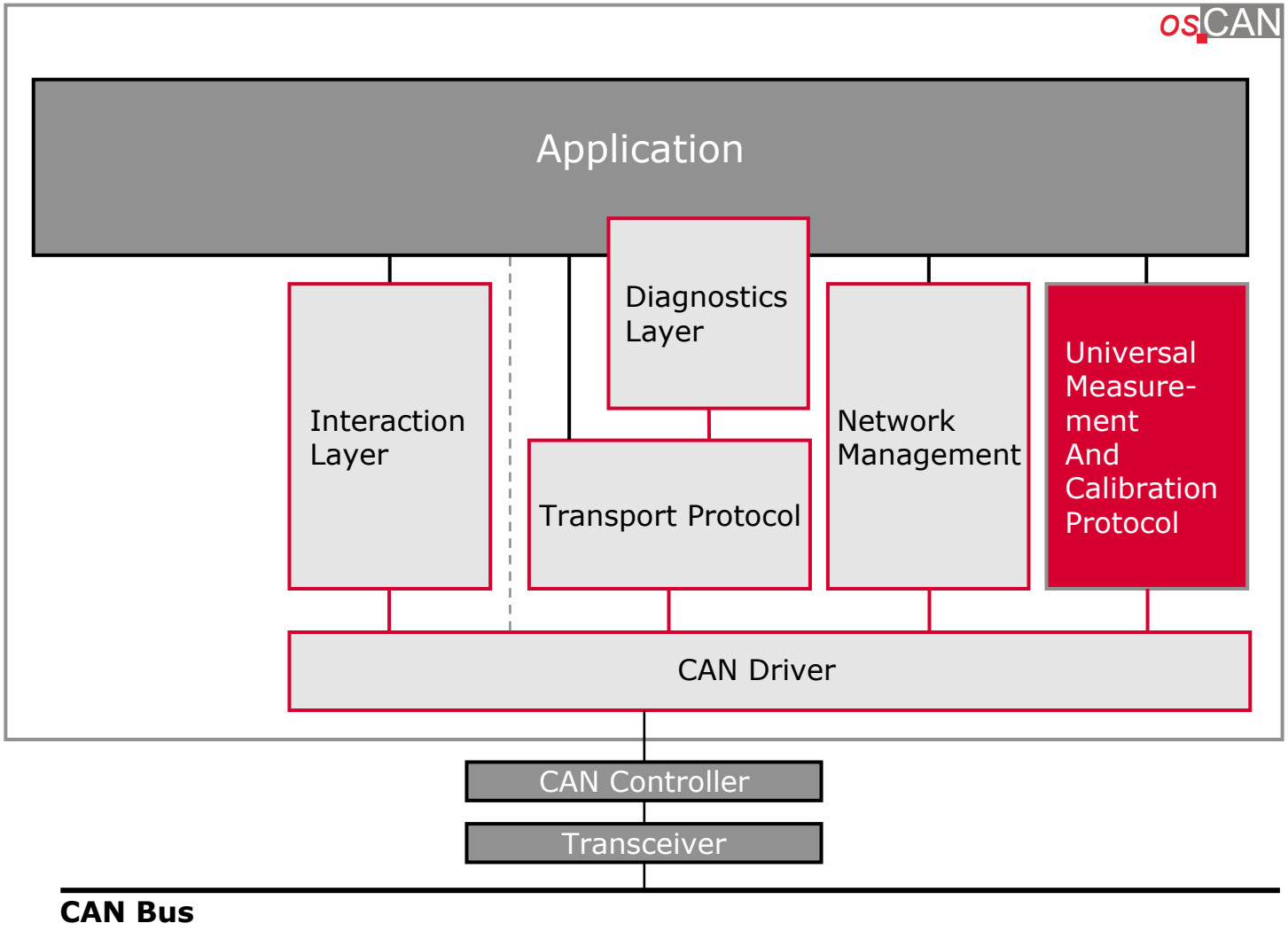


Network Management to Control the CAN Bus

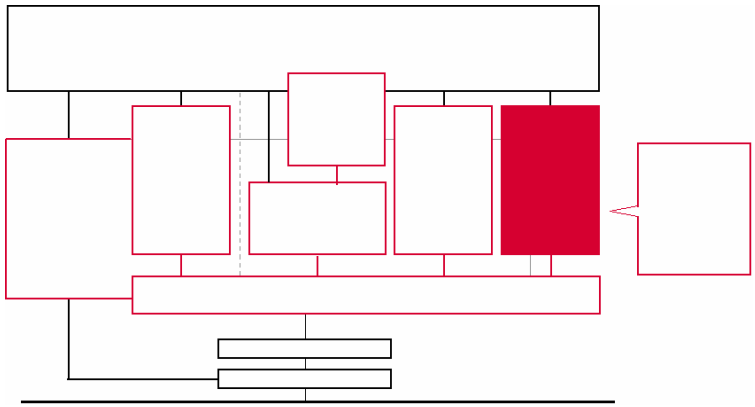


- Synchronized Transition to Bus Sleep
- Determination of Net Configuration at Startup
- Monitoring of Net Configuration During Operation
- Error Recovery after Bus-Off
- Provision of Network Status Information

Measurement and Calibration Protocol - XCP

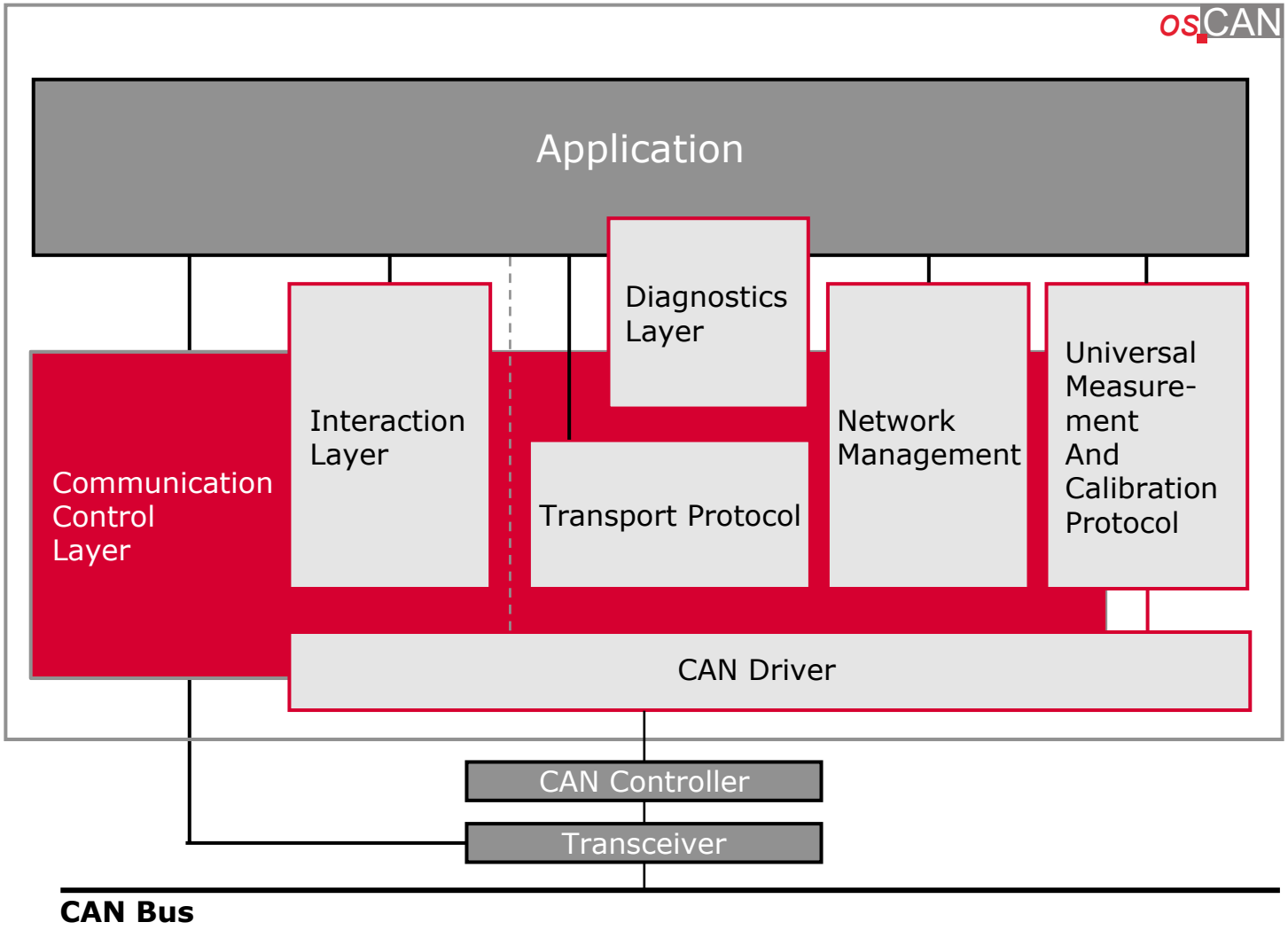


Universal Measurement and Calibration Protocol for Measurement and Calibration on various bus systems

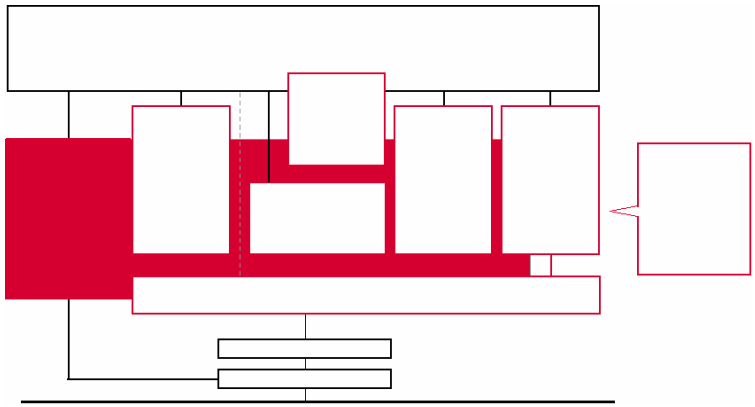


- Read and Write Access to Various Memory Locations
- Different Data Access Methods (Polling, Cyclic and Event-Triggered)
- Flash Programming
- Simultaneous Handling of Several Controls

Communication Control Layer



Communication Control Layer



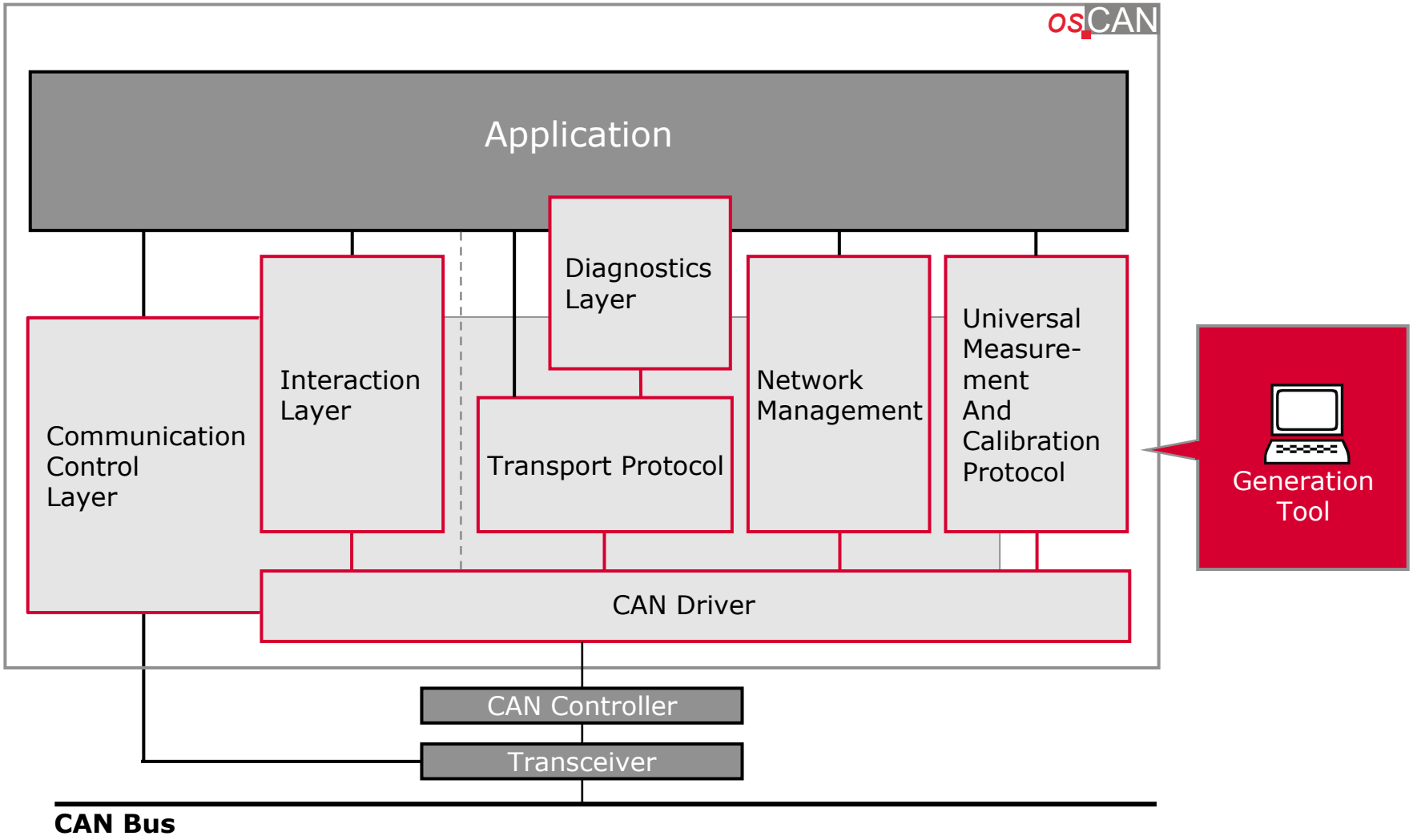
Integration of the Software Components

- CAN Driver,
- Interaction Layer,
- Network Management,
- Transport Protocol
- Diagnostics

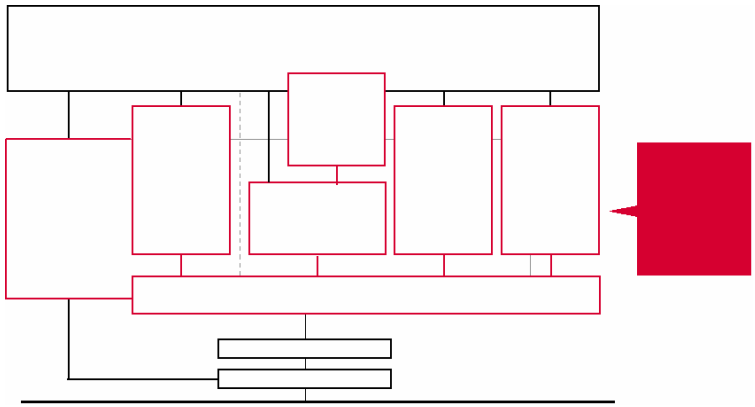
Abstraction for different

- Vehicle manufactureres
- Microcontrollers
- Compiler/linker
- CAN Controllers / Transceivers
- Configured via Generation Tool
- Global Debug Mechanism

Generation Tool

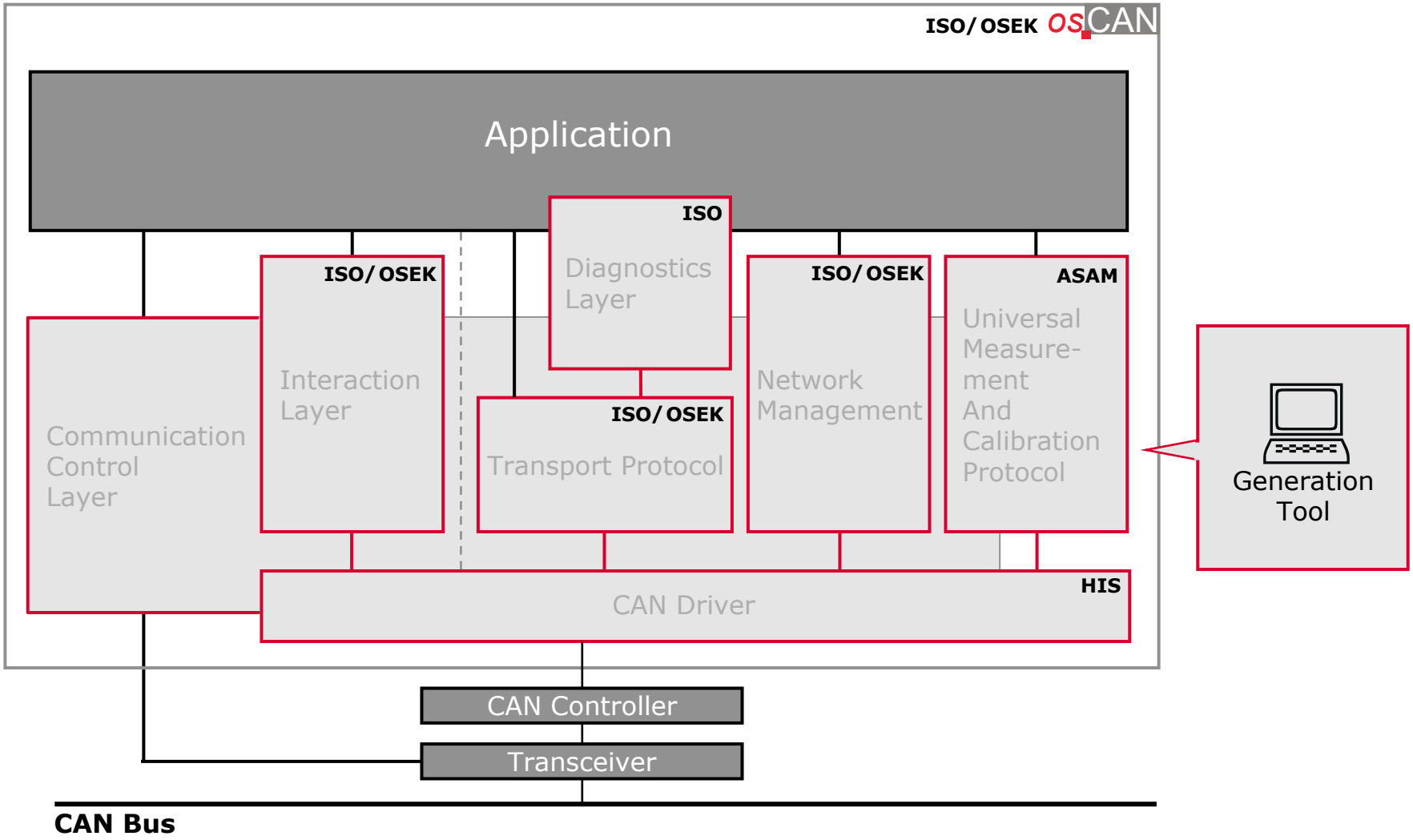


Generation Tool for Parameters and Configuration



- Used for the Complete Set of Vector's CAN Software Components
- Driven by Communication Matrix (Network Database)
- User Specific Settings for Each Node (Application Database)
- Part of Vector's Tool Chain

CANbedded Software Components and Standards



Generation Process