J1939 Research Document

Minor - Signals and Embedded Systems

Kenneth Asamoah Boakye-Buah

November 21, 2025

Revision History

Revision	Date	Author(s)	Description
1.0	2025-11-18	Kenneth	Draft.

Contents

1 Abstract

This research document investigates the SAE J1939 protocol as a higher layer built on top of the CAN 2.0B standard for heavy-duty vehicles and industrial automation. Guided by the DOT ICT framework (Library, Field, Workshop), the report first explains what J1939 is, why it is needed in automation, and how its standards and message structures are defined. Next, it analyses how J1939 is implemented in current industry systems, including the rules that must be followed in message protocols and how the protocol supports communication between critical ECUs such as engine, steering, and brake controllers. Finally, the document explores how J1939 can be used on CAN 2.0 hardware and software in a workshop setting by simulating different ECUs and their message exchanges. Practical aspects such as the 9-pin Deutsch connector, PGNs, SPNs, message formats, and Request/Transport Protocol mechanisms are described to support future automation projects.

2 Introduction

When a driver presses the brake pedal in a bus, dozens of electronic systems must instantly coordinate to ensure safety. Behind this seamless interaction lies SAE J1939, a standardized higher layer protocol built on top of the CAN bus communication standard that makes automation reliable in industries where failure is not an option. J1939 has become the backbone of automation in heavy-duty vehicles and related industrial systems. As automation projects grow more complex, understanding how J1939 operates is essential for engineers and researchers alike.

This paper uses the DOT ICT framework, moving from literature study (Library) to industry practice (Field) and finally workshop implementation (Workshop), with the focus on critical ECUs such as the engine, steering and brake systems of an automated vehicle.

3 Research Questions

The following research questions guide this study. Each question is aligned with the DOT ICT framework to ensure a structured approach from theoretical foundations to industry practice and practical implementation.

- **What is J1939 CAN?** (Library – Literature Study)
  - Why is it used in automation? (Library + Field)
  - What are the J1939 standards and message structures? (Library)
- **How is J1939 implemented in current industry automation systems?** (Field)

  - What rules must be followed, and how are they applied in message protocols? (Library + Workshop)
  - How does J1939 support communication between ECUs such as Engine, Steering, and Brake? (Field + Workshop)

- **How can J1939 be built on CAN 2.0 for an automation project?** (Workshop)

3.1 Research Methodology

This section explains how each methodology will be used in each research question.

- **Library:**
  - Literature study is conducted using the SAE J1939 standards and their message structure, online documentation, blogs and academic papers.
- **Field:**
  - Industry case studies of ECU communication (Engine, Steering, Brakes) and automation systems are analysed to see how J1939 is applied in practice.
- **Workshop:**
  - Simulations of different ECUs are designed using the J1939 message protocol on CAN 2.0 hardware/software to test implementation concepts.

4 What is J1939 CAN?

4.1 Definition and Need for J1939

SAE J1939 is a family of standards that defines how Electronic Control Units (ECUs) communicate over the CAN bus in heavy-duty vehicles such as trucks, buses and agricultural machinery. CAN on its own only provides basic framing, arbitration and error handling. J1939 builds on this as a higher layer protocol that standardizes the meaning of messages and signals across different manufacturers (Kvaser, 2025; Electronics, 2025).

In simple terms, J1939 offers a standardized method of communication between ECUs and across manufacturers. Passenger cars often use proprietary protocols; knowing how to read engine speed from a Ford will not automatically help with a BMW. With J1939, the same standardized way of defining engine speed can be applied across all compliant devices, as long as the basic rules of the protocol are followed (AutoPi.Io, n.d.; Electronics, 2025).

4.2 Historical Background

The origin of J1939 can be traced back to the 1990s. In 1994 the first standard documents were released, such as J1939-11, J1939-21 and J1939-31 (Electronics, 2025). In 2000 J1939 formalized the use of CAN as the physical layer. Around 2001 J1939 began replacing older standards like J1708 and J1587, and from 2002 onward J1939 became the dominant protocol for heavy-duty vehicles. More recently, J1939-22 has been introduced to incorporate the next generation of CAN bus: CAN FD (Flexible Data Rate).

4.3 Why J1939 is Important in Automation

Besides its use in heavy-duty vehicles, J1939 is widely used for vehicle **telematics** systems which gather and transmit data on vehicle usage, maintenance needs and performance (AutoPi.Io, n.d.). This helps fleet managers monitor and optimize their operations, resulting in cost savings and improved efficiency.

J1939 also plays a crucial role in:

- **Vehicle diagnostics:** providing standardized diagnostic messages and codes that technicians use to find and fix issues.
- **Engine and powertrain control:** modern engines use J1939 to communicate between different control units, ensuring smooth and efficient operation.
- **Interoperability:** allowing components from different manufacturers to work together seamlessly by using common PGNs and SPNs.

4.4 Key Properties and Characteristics of J1939

Key properties of the J1939 standard include (Electronics, 2025):

- Default baud rate of 250 kbit/s, with newer variants also supporting 500 kbit/s.
- Use of the extended 29-bit CAN identifier (CAN 2.0B).
- Most data is broadcast on the bus, but some data is only available by explicitly requesting it.
- J1939 messages are identified by 18-bit **Parameter Group Numbers (PGNs)**, while individual signals are called **Suspect Parameter Numbers (SPNs)**.
- Multi-byte variables are sent least significant byte (LSB) first, and PGNs up to 1785 bytes are supported via the J1939 Transport Protocol.

4.5 J1939 in the OSI Model

J1939 can be mapped onto the OSI model as follows (Electronics, 2025):

- **Physical Layer:** Standards J1939/11–16 describe 250 kbit/s and 500 kbit/s communication, the 9-pin connector, wiring and shielding. The physical layer is based on ISO 11898 (CAN).
- **Data Link & Transport Layer:** J1939/21 defines the frame structure, address claiming, transport protocols, request messages and acknowledgments.
- **Network Layer:** J1939/31 describes communication between segments via repeaters, bridges, forwarders and routers.
- **Application Layer:** J1939/71 and J1939/73 define the PGNs, SPNs, Diagnostic Messages (DM) and Diagnostic Trouble Codes (DTCs).

4.6 J1939 Connector and Pinout

The J1939 9-pin Deutsch connector (J1939-13) is the standard off-board diagnostic connector used to interface with the J1939 network in most heavy-duty vehicles. The connector exists in two main variants:

- **Type 1 (black stick type):** older version, supports 250 kbit/s. It does not include an internal termination resistor, so an external 120 Ω resistor is needed on the network.
- **Type 2 (green stick type):** newer version, supports up to 500 kbit/s and includes a built-in 120 Ω terminating resistor between CAN_H and CAN_L.

The pinout is used for troubleshooting and logging J1939 networks (Electronics, 2025; AutoPi.Io, n.d.-b):

- A – Ground: provides ground connection.
- B – +12V: supplies 12 V power, often used to power tools or loggers.
- C – CAN1 / J1939 High: high line of the first CAN/J1939 network.
- D – CAN1 / J1939 Low: low line of the first CAN/J1939 network.
- E – CAN shield.
- F – CAN2 High or J1708/J1587 High: high line of the second network, depending on the application.
- G – CAN2 Low or J1708/J1587 Low: low line of the second network.
- H – OEM specific: reserved for manufacturer-specific functions.
- J – OEM specific: additional manufacturer-specific pin.

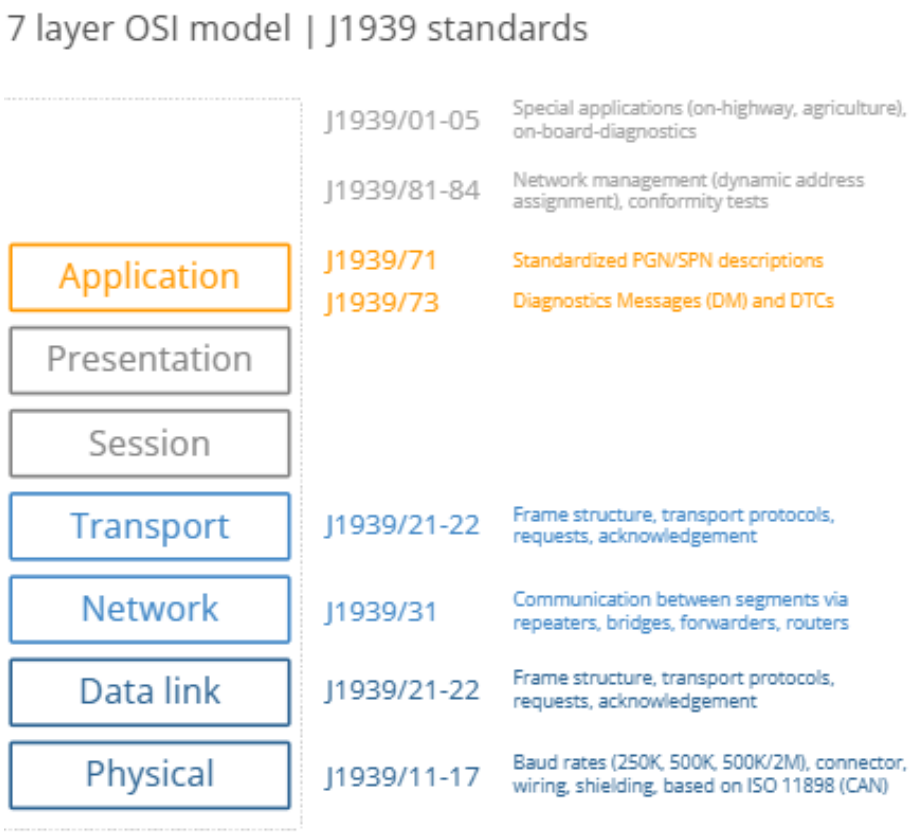

Figure 1: J1939 OSI Layer

4.7 J1939 Frame Format and Message Structure

The J1939 message format is based on the CAN data frame and consists of: Start of Frame (SOF), 29-bit Identifier, control bits (RTR, IDE, DLC), Data Field, CRC and End of Frame (EOF) ("SAE J1939 Frame Format and Message Structure," n.d.).

- **Start of Frame (SOF):** a single dominant bit that marks the beginning of a message.
- **Remote Transmission Request (RTR):** in classical CAN this bit distinguishes data frames (0) from remote frames (1). J1939 forbids the use of remote frames and requires RTR=0. All J1939 requests are done via the Request PGN and not by setting RTR to 1. If RTR is set to 1 the frame violates the J1939 specification and most compliant ECUs will ignore it or treat it as a non-J1939 CAN frame.
- **Data Field:** contains the actual information being transmitted, from 0 to 8 bytes for classic CAN frames.
- **Checksum (CRC):** the 15-bit CAN cyclic redundancy check ensures data

integrity and detects bit errors during transmission. If an error is detected, a CAN Error Frame is generated and all other nodes discard the faulty frame.

- **End of Frame (EOF):** a sequence of recessive bits that marks the end of the CAN frame.

4.8 PGNs and SPNs

In J1939, the extended 29-bit CAN identifier contains the J1939 Parameter Group Number (PGN) and the Source Address. The PGN is an 18-bit subset of the 29-bit ID and serves as a unique frame identifier within the J1939 standard (Electronics, 2025; J1939 Protocol, n.d.).

- The 29-bit identifier is composed of a 3-bit Priority, the 18-bit PGN and an 8-bit Source Address.
- The PGN itself can be split into: 1-bit Reserved, 1-bit Data Page, 8-bit PDU Format (PF) and 8-bit PDU Specific (PS).

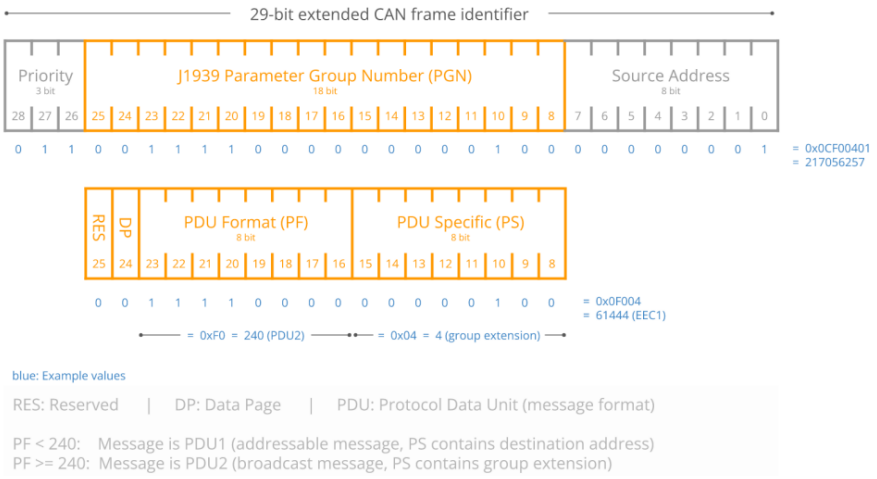

Figure 2: J1939 frame format.

The PDU Format determines the message type:

- **PDU1 (PF Less Than 240):** addressable, peer-to-peer messages. Here the PDU Specific field contains a destination address.
- **PDU2 (PF Greater Or Equal 240):** broadcast messages. In this case the PDU Specific field is used as a group extension and there is no specific destination address in the PGN.

The Data Page and Extended Data Page bits act as counters to expand the total number of available PGNs.

4.8.1 SPNs as Signals

SPNs (Suspect Parameter Numbers) represent the individual CAN signals within a J1939 frame. SPNs are grouped by PGNs and are described by their start bit, bit length, scale and offset. This information is needed to extract and decode J1939 parameters to human-readable form (Electronics, 2025).

For example, consider a raw J1939 frame:

- CAN ID: 0CF00400
- Data bytes: FF FF FF 68 13 FF FF FF

From the 29-bit ID we extract PGN 0F004 (decimal 61444), which corresponds to *Electronic Engine Controller 1.* One of the SPNs in this PGN is **engine speed**, SPN 190. The signal has:

- start bit 24,
- length 16 bits,
- little-endian byte order,
- scale 0.125 and offset 0.

Re-ordering bytes 68 13 to 13 68 (little endian) and converting to decimal gives 4968. Applying the standard conversion: physical value = $0.125 \times 4968 + 0 = 621$ RPM. In practice, tools handle these calculations automatically using DBC files and J1939 databases.

5 How is J1939 implemented in current industry automation systems?

5.1 Rules in Message Protocols

In industry, J1939 is implemented by strictly following the rules for frame identifiers, addressing and timing.

- **Priority:** lower numeric priority (0) means higher priority on the bus; 7 is the lowest priority. Safety-critical messages (e.g., brake control) typically use higher priority.
- **Source Address (SA):** each ECU has a unique SA (0–253). No two devices on the same J1939 network are allowed to share the same address.
- **Broadcast vs. peer-to-peer:** PGNs with PDU2 format are broadcast, while PDU1 PGNs carry a destination address and are sent peer-to-peer.

- **Cyclic and on-request messages:** many PGNs are transmitted periodically (e.g., engine speed every 100 ms), while others—especially diagnostic data—are only sent when explicitly requested (DM2, DM14, etc.).

Address management, request handling and acknowledgement behaviour are all defined in J1939/21 and J1939/81 and must be followed for interoperability between different vendors.

5.2 Message Types in J1939

The J1939 document describes five message types (SAE J1939 Protocol: Introduction and Packet Analysis, 2025):

- **Commands** – instruct an ECU to perform an action (e.g., torque command to the engine).
- **Request** – one ECU asks another to transmit a specific PGN.
- **Broadcasts/Responses** – periodic data (engine speed, wheel speed, etc.) and responses to requests.
- **Acknowledgment** – positive/negative/busy responses using PGN 59392.
- **Group Functions** – protocol services such as Request, Acknowledgment and Transport Protocol PGNs (RTS/CTS/BAM).

5.3 Request Messages

Most J1939 messages are transmitted at a fixed periodic rate, but some are only transmitted when requested. A typical example is diagnostic message DM2, which contains previously active Diagnostic Trouble Codes (DTCs).

J1939 uses a special PGN for all such requests (J1939 Protocol, n.d.):

- **PGN 59904 (0xEA00)** – Request.
- Data length is 3 bytes: the requested PGN, sent LSB first.

To request a PGN from another ECU:

1. The requester sends PGN 59904 with:
    - PDU1 format, where the PDU Specific field is the destination address (or 255 for global request).
    - Data bytes 1–3 containing the PGN that is being requested (LSB–MSB).
2. The target ECU checks whether it supports the requested PGN.

3. If supported, it transmits the requested PGN either to the requester or as a broadcast, depending on the PGN definition.

4. If not supported or temporarily impossible, the ECU may respond with an Acknowledgment (PGN 59392) stating ACK, NACK, busy or access denied.

Thus, requests are handled at the J1939 application level, not via the CAN RTR bit.

5.4 Transport Protocol Messages

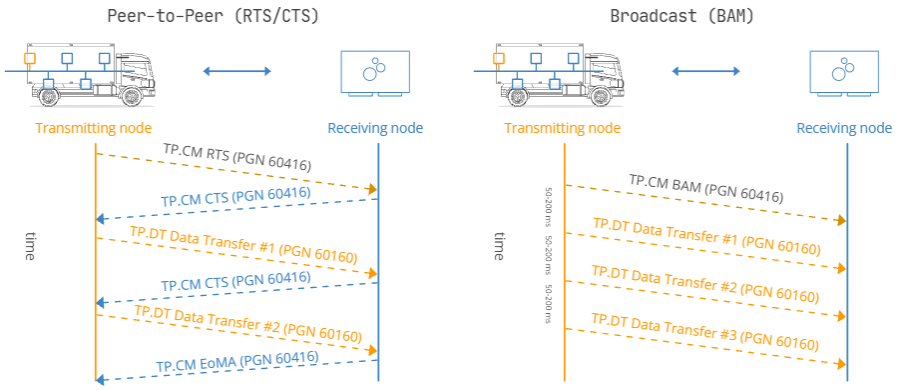

Figure 3: J1939 Transport Protocol

Some J1939 messages are longer than the 8-byte CAN data field. For these cases J1939 defines a Transport Protocol (TP) that splits large messages into multiple CAN frames (SAE J1939 Protocol: Introduction and Packet Analysis, 2025; J1939 Protocol, n.d.).

Two PGNs are used:

- **TP.CM (Connection Management) – PGN 60416 (0xEC00)** Used to announce and control multi-packet transfers.
- **TP.DT (Data Transfer) – PGN 60160 (0xEB00)** Carries the actual data bytes in packets of 7 bytes per frame.

There are two main modes:

Broadcast Announce Message (BAM)

BAM is used when a large PGN is broadcast to all nodes without flow control.

1. Sender calculates message size N and number of packets $k = \lceil N/7 \rceil$.

2. Sender transmits a TP.CM frame with control byte = 32 (BAM), message size, packet count and the PGN of the large message.

3. After a short delay, the sender transmits k TP.DT packets. Each packet has:

   - Byte 1 – sequence number (1, 2, 3, ...),
   - Bytes 2–8 – 7 bytes of payload.

4. Any node interested in that PGN listens to the BAM announcement, then collects and reassembles the TP.DT packets.

RTS/CTS Connection Mode

RTS/CTS is used for large messages sent to a specific ECU with flow control.

1. **RTS (Request To Send):** The sender transmits TP.CM with control byte 16 (RTS), total message size, total packet count, proposed packets per block and the PGN.

2. **CTS (Clear To Send):** The receiver replies with TP.CM control byte 17 (CTS), indicating how many packets it is ready to receive and from which sequence number.

3. **TP.DT packets:** The sender transmits the indicated number of TP.DT packets containing the data.

4. Steps 2 and 3 repeat until all packets are transmitted.

5. **EndOfMsgACK:** After receiving all packets, the receiver sends TP.CM control byte 19 to acknowledge successful reception. If an error occurs, either side can send TP.CM with control byte 255 (Abort) and a reason code.

These Transport Protocol messages belong to the *Group Function* message type and implement multi-packet transfer on top of the basic CAN frames.

5.5 Communication Between Engine, Steering and Brake ECUs

In the field, all ECUs share the same J1939 CAN bus. Each ECU periodically broadcasts its data using standardized PGNs and SPNs:

- The **engine ECU** broadcasts engine-related PGNs such as engine speed, torque, fuel rate and temperature.

- The **brake ECU** broadcasts wheel speeds, brake pressure and ABS/EBS status.

- The **steering ECU** can broadcast steering angle and steering status.

Because these signals are defined in the J1939-71 application layer, any compliant ECU can interpret them. For example:

- The brake controller can use engine speed and torque information to coordinate retarder or engine brake.
- The engine controller can receive brake or stability control requests and reduce torque.
- A steering or stability ECU can combine steering angle with wheel speed data to detect under- or over-steer and request braking or torque reductions.

In the workshop, a diagnostic tool connects through the 9-pin connector and acts as another node on the J1939 network. It listens to Diagnostic Messages (DM1 for active DTCs, DM2 for previously active DTCs, etc.) and can send Request messages for additional diagnostic data. Technicians can also clear DTCs by sending the appropriate J1939 diagnostic PGNs.

6 How can J1939 be built on CAN 2.0 for an automation project?

From a workshop perspective, J1939 can be implemented on standard CAN 2.0B hardware and used in student or prototype automation projects.

6.1 Hardware and Physical Setup

- Use CAN 2.0B capable microcontrollers or development boards (e.g., with integrated CAN controllers and transceivers).
- Configure the bus for 250 kbit/s or 500 kbit/s, with two 120 Ω termination resistors between CAN_H and CAN_L at the ends of the bus (or one built-in on the Type 2 Deutsch connector plus an external one).
- Optionally expose a 9-pin J1939 Deutsch connector so that standard tools and loggers can be connected.

6.2 Software Design

- Implement sending and receiving of 29-bit CAN identifiers according to J1939.
- For each simulated ECU (e.g. Engine, Steering, Brake), assign a valid J1939 Source Address.

- Define which PGNs and SPNs each ECU will transmit or consume, based on examples from J1939-71 (e.g., engine speed SPN 190).
- Implement periodic broadcast of key PGNs using timers.
- Implement Request handling for PGN 59904 and, if needed, the Transport Protocol PGNs for messages longer than 8 bytes.

6.3 Logging and Analysis

Using a J1939 compatible CAN logger and tools like `asammdf`, recorded data can be decoded using DBC files. The decoded SPNs can then be visualised in scripts or dashboards (for example using Grafana) to analyse ECU behaviour over time.

6.4 Link to DOT ICT Framework

- **Library:** understanding J1939 standards, PGNs, SPNs and frame formats.
- **Field:** analysing how real vehicles use J1939 for engine, steering and brake communication and diagnostics.
- **Workshop:** building a small-scale J1939 network on CAN 2.0 hardware to simulate ECUs and test automation concepts in practice.

7 Conclusion

This report showed that SAE J1939 is a higher layer protocol on top of CAN 2.0B that standardizes communication between ECUs using 29-bit identifiers, PGNs and SPNs. Literature and field examples demonstrated how J1939 is used in heavy-duty vehicles for engine, steering and brake coordination, as well as diagnostics via the 9-pin Deutsch connector. From a workshop perspective, the study described how J1939 messages, Request PGNs and the Transport Protocol (RTS/CTS and BAM) can be implemented on standard CAN hardware. Overall, J1939 provides a robust and interoperable communication backbone for automation projects involving multiple ECUs.

References

1. AutoPi.Io. *(n.d.). J1939 explained (2025): PGNs, SPNs and heavy-duty diagnostics.* Retrieved from https://www.autopi.io/blog/j1939-explained/#basic-components-of-j1
2. SAE J1939 Frame Format and Message Structure. *(n.d.).* RF Wireless World. Retrieved from https://www.rfwireless-world.com/terminology/sae-j1939-frame-format-messag

3. Electronics, C. *(2025, January 23).* J1939 Explained - A simple Intro [2025]. CSS Electronics. Retrieved from https://www.csselectronics.com/pages/j1939-explained-simple what-is-j1939

4. Kvaser. *(2025, February 5).* J1939 Introduction. Retrieved from https:// kvaser.com/about-can/higher-layer-protocols/j1939-introduction/

5. J1939 Protocol. *(n.d.).* (C) Copyright 2025. Retrieved from https://www. typhoon-hil.com/documentation/typhoon-hil-software-manual/References/j1939_ protocol.html

6. SAE J1939 Protocol: Introduction and Packet Analysis. *(2025).* [Online tutorial / article].