DOI: 10.12700/AIS.2025.021

Design and transmission of data structures for industrial robots in digitized industrial systems

Zolta´n Szila´gyi
Doctoral School of Applied Informatics and Applied Mathematics, Óbuda University
Budapest, Hungary
szilagyi.zoltan@amk.uni-obuda.hu

Csaba Hajdu
Departement of Informatics Sze´chenyi University
Gyo˝r, Hungary
hajdu.csaba@sze.hu

Ka´roly Sze´ll
Technical Institute Alba Regia Faculty, Óbuda University
Sze´kesfehe´rva´r, Hungary
szell.karoly@amk.uni-obuda.hu

Pe´ter Galambos
Antal Bejczy Center for Intelligent Robotics, Óbuda University
Budapest, Hungary
peter.galambos@irob.uni-obuda.hu

Abstract—The increasing integration of industrial robots into digitized manufacturing systems requires unified, lightweight, and vendor-independent data models. Existing frameworks such as the OPC UA Companion Specification for Robotics offer rich semantics and interoperability but are computationally intensive, while MQTT-based approaches provide simplicity and speed at the cost of structure and meaning. This paper proposes the Sparkplug Robotics Profile v1, a lightweight extension to the Sparkplug B standard that bridges this gap by introducing robot-specific semantics and state management within an event-driven MQTT architecture. The proposed model defines structured telemetry (RobotData), command (RobotCmd), and lifecycle (RobotBirth) message types using Google Protocol Buffers for efficient transmission and consistent interpretation across platforms. This approach enables real-time data exchange, scalable multi-robot coordination, and native compatibility with XR-based digital-twin environments. The results highlight that Sparkplug Robotics Profile v1 preserves the simplicity of MQTT while providing semantic depth comparable to OPC UA, facilitating interoperable and resource-efficient robot connectivity from edge to cloud.

Index Terms—Industrial robotics, data models, OPC UA, MQTT, Sparkplug B, digital twin, XR systems.

I. Introduction

The evolution of Industry 4.0 has transformed industrial robots from isolated automation units into fully integrated cyber-physical systems that continuously exchange information with manufacturing execution, planning, and visualization platforms [1]. As industrial environments become increasingly data-driven, the need for standardized, real-time, and semantically rich data exchange between robots and supervisory systems has intensified.

Traditional robot communication frameworks are highly vendor-specific, relying on proprietary APIs or fieldbus extensions that hinder interoperability. To address this, the OPC UA Companion Specification for Robotics provides a comprehensive object-oriented model for describing robot kinematics, states, and control entities [2]. While this standard enables strong interoperability and alignment with higher-level industrial information models, its complexity and resource demands make it less suitable for lightweight edge or extended-reality (XR) applications [3].

At the opposite end of the spectrum, MQTT-based communication—particularly the Eclipse Sparkplug B specification—offers a simple, event-driven architecture that efficiently transmits process data in near real time. However, it lacks the domain semantics required to describe robotic systems meaningfully. Consequently, an architectural gap persists between semantic richness and communication efficiency.

This research introduces a new, intermediate approach that combines the strengths of both paradigms: the Sparkplug Robotics Profile v1. The proposed profile defines a robot-specific data structure within the Sparkplug B framework, enabling real-time telemetry, command execution, and lifecycle monitoring in a unified namespace. The objective is to support scalable, manufacturer-independent robot integration across industrial, edge, and XR contexts while maintaining low latency and minimal computational overhead.

The OE-RH-423/2024 project has been implemented with the support of doctoral research work carried out at company-university cooperation - relevant to domestic R&D programs – provided by ”O´buda University Cooperative PhD Student Scholarship Program”, from the ”Talent Care Fund” of O´buda University. Pe´ter Galambos is a Bolyai Fellow of the Hungarian Academy of Sciences. Pe´ter Galambos is supported by the UNKP-22-5 (Bolyai+) New National Excellence Program of the Ministry for Innovation and Technology from the source of the National Research, Development and Innovation Fund.

II. OPC UA Companion Specification for Robotics

The OPC Unified Architecture (OPC UA) is one of the most important industry standards for vendor-independent data communication, based on an object-oriented model and hierarchical address space [4]. The Companion Specification for Robotics, published by the OPC Foundation and the VDMA in 2019, aims to provide a unified description of the states, kinematics, and control parameters of robots, in particular industrial arm robots.In the model, each robot appears as a component, a RobotType object derived from a DeviceType.

It includes:

- identification and metadata objects (Identification, Vendor, FirmwareVersion)
- robot kinematic structure (Axes, Joint, ToolFrame, BaseFrame)
- State and diagnostic information (StateMachine, MotionDeviceSystemState, SafetyState)
- control and programming entities (Program, Task, MotionJob, ProgramState)

The Companion Specification aims to ensure that the data link between the robot and the controller works in accordance with the concepts of Device Integration and Machine Information Model, allowing interoperability with MES/SCADA systems and other OPC UA devices. The descriptive model supports multi-axis kinematics, coordinate transformations, safety state retrieval, and unified handling of fault code objects. Although the Companion Specification is very detailed and industry-wide, its implementation complexity is significant. An entire OPC UA Robotics server stack contains hundreds of NodeId, type definitions and references. Mapping the entire hierarchy of the model is computationally and memory intensive, especially on edge devices or XR-based digital twins where latency and processing capacity are limited [5]. The benefits of the standard - transparency, formalised structure, functional safety compliance - are mainly for factory-level integrations, but it is proving too cumbersome for lightweight edge and real-time monitoring tasks.

III. The need for simplification and the architectural gap

In robotic data modelling today, there is a dichotomy: OPC UA-based systems offer rich semantics but are complex and resource-intensive; in contrast, MQTT-based solutions such as Sparkplug B are lightweight, fast but poor semantically. Currently, there is no overlap between the two paradigms, which hinders unified edge-to-cloud communication. In OPC UA Robotics, all robot, controller, program, axis relations are represented by explicit node references, and communication is client/server. This is deterministic, but difficult to scale when hundreds of robots need to transmit data to a cloud or XR system simultaneously. MQTT, on the other hand, uses a publish/subscribe architecture that natively supports asynchronous data flow, but has no common data model - messages are just topics and payloads. Developments towards digital twins, predictive maintenance and XR visualisation require robot data to be available in real time, with low latency and in a manufacturer-independent way. This cannot be achieved by the classical OPC UA model due to node-level query logic, especially when data flows through a central broker or distributed edge network.

The architectural gap is therefore between semantics and speed:

- OPC UA provides meaning but loses lightness,
- MQTT provides speed, but loses reporting,

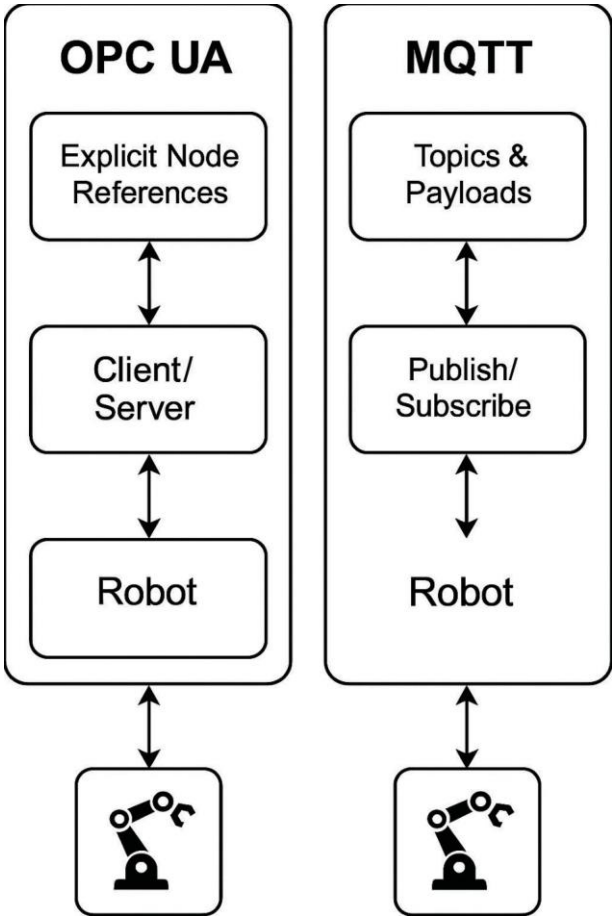

Fig. 1. OPCUA and MQTT protocols

This creates the need for a new, intermediate approach, a lightweight data model that preserves Sparkplug's event-driven, broker-based architecture, but in-

cludes robot-specific metrics and states - the basis of the Sparkplug Robotics Profile v1 concept.

IV. MQTT Sparkplug - a lightweight, event-driven industrial data model

The Message Queuing Telemetry Transport (MQTT) protocol is the de facto standard for lightweight, reliable and asynchronous data transmission in industrial IoT systems [6]. The basic principle of MQTT is message-centric, publish/subscribe communication, which uses a central broker to implement loose coupling between devices. Traditional MQTT applications, however, do not define a uniform data structure or state management; each vendor uses its own topic names and JSON payloads, leading to interoperability problems in the long run [7].

This shortcoming is addressed by the Eclipse Sparkplug standard, developed as part of the Eclipse Foundation Tahu Project and now captured in the Sparkplug Specification v2.2 standard document.

- 1. Unified payload format - transmits raw data in a binary Google Protocol Buffers (Protobuf) format, where each metric is associated with a type, quality flag and timestamp.
- 2. Event-driven state machine - the lifecycle of devices is described by Birth and Death messages and the MQTT Last Will and Testament mechanism. This allows the system to detect when a device has failed, rebooted or lost connection.

Sparkplug's message structure is based on three main hierarchical levels:

```
spBv1.0/<Group_ID>/
  <Message_Type>/
  <Edge_Node_ID>/
  <Device_ID>
```

This enables the creation of a Unified Namespace in which messages are organised thematically and logically, and data can be interpreted independently of the producer.The specification does not define domain-specific data models: a "metric" can be any name-value pair, so it lacks the semantic layer needed for robotics, for example.This gap justifies the introduction of a Sparkplug extension that specifically unifies the data structure and command system for industrial robots.

V. Sparkplug Robotics Profile v1 - proposed extension

The Sparkplug Robotics Profile v1 aims to formulate a lightweight, vendor-independent data model for industrial robots that fits the Sparkplug B framework but has extensive robot-specific semantics. The proposed profile is not intended to replace the OPC UA Robotics specification, but to provide an edge-level alternative optimized for real-time telemetry and XR/digital twin applications.

The profile defines three main message types:

- RobotBirth – publishing metadata and schema versions (model, manufacturer, axle number, frame definitions)
- RobotData – continuous telemetry data (positions, speeds, statuses)
- RobotCmd – non real-time commands and configuration instructions

```
message   RobotData   {
  uint64 timestamp = 1;
  repeated double joint_pos = 2;
  repeated double joint_vel = 3;
  Pose tcp_pose = 4;
  string mode = 5; // AUTO, MANUAL, TEACH
  string run_state = 6; // RUN, FAULT
  string safety_state = 7; // NORMAL, STOP
  string fault_code = 8;
  double energy_Wh = 9;
  double cycle_time_ms = 10;
}
```

The Pose structure gives the TCP position in quaternions (x,y,z,qw,qx,qy,qz), which facilitates direct spatial mapping of XR systems.
The profile follows the original Sparkplug B pattern, but extends it at the device-level:

```
spBv1.0/RobotCell/NBIRTH/Edge1/RobotA
spBv1.0/RobotCell/DDATA/Edge1/RobotA
spBv1.0/RobotCell/DCMD/Edge1/RobotA
```

This ensures that multiple robots (e.g. RobotA, RobotB) and multiple edge nodes (e.g. Edge1, Edge2) can be coherently managed in the same namespace.

Typical examples of RobotCmd messages:

```
load_job(name)
start_job()
stop_job()
ack_fault()
set_mode(AUTO|MAN)
set_speed_override(percent)
```

The introduction of Sparkplug Robotics Profile v1 does not require firmware modifications to existing robot controllers. A middleware layer (gateway) interfaces the manufacturer-specific APIs (FANUC, ABB, KUKA, UR) to the Sparkplug message structure [8].

VI. Results and future work

One of the key benefits of the concept is that Sparkplug-based communication is easier to implement

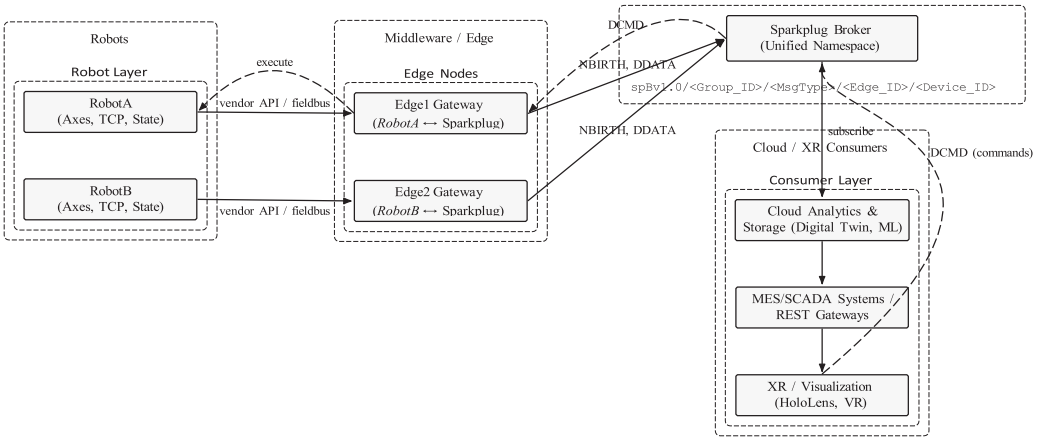

Fig. 2. Sparkplug Robotics Profile data flow: Robot → Middleware → Broker ↓ Cloud/XR. The consumer layer is placed below the broker to maintain full-page width while preserving logical data direction.

on edge devices compared to traditional OPC UA servers, while maintaining industry interoperability. The structure of the profile allowed robots from different manufacturers to appear in a single namespace and to retrieve control and diagnostic data in a common format.

The introduction of Sparkplug Robotics Profile v1 offers new value on several levels:

- At the industrial level: a unified data profile in the MQTT ecosystem, directly usable for robot monitoring, SCADA and digital twin applications.
- At the developer level: a simple, extensible message schema that allows new metrics or commands to be added while maintaining compatibility.
- At the scientific level: the concept provides a transparent, reproducible framework for comparative study and modelling of robotic data transmission.

Further work will focus on three key areas:

- Formal schema standardization: Mapping the current field list into an RFC-like document, and consultation with the Eclipse Sparkplug Working Group community.
- Full interoperability: Testing the profile on robot controllers from different vendors and developing an automatic data mapping between Sparkplug and OPC UA models.
- Digital twin integration: Development of XR and MES-level integrations based on Sparkplug Robotics Profile data, which can be validated in real-time manufacturing environments.

VII. Conclusion

The aim of the research was to bridge the gap between the complex but detailed model of the OPC UA Companion Specification for Robotics and the lightweight event-driven architecture of MQTT Sparkplug. The presented Sparkplug Robotics Profile v1 provides an intermediate solution that provides unified semantics, real-time transmission and vendor independence for robotic data exchange. The solution simplifies the integration of heterogeneous robot fleets from an industrial perspective.

References

[1] J. E. Solanes, A. Mun˜oz, L. Gracia, A. Mart´ı, V. Girbe´s-Juan, and J. Tornero, “Teleoperation of industrial robot manipulators based on augmented reality,” *International Journal of Advanced Manufacturing Technology*, vol. 111, no. 3, pp. 1077 – 1097, type: Article.

[2] A. Busboom, “Automated generation of OPC UA information models — a review and outlook,” *Journal of Industrial Information Integration*, vol. 39, p. 13.

[3] S. K. S. P. G. Zolta´n Szila´gyi, Csaba Hajdu, “Defining user coordinate systems for industrial robots in mixed reality,” in *2024 IEEE 7th International Conference and Workshop O´buda on Electrical and Power Engineering (CANDO-EPE)*. IEEE, pp. 225–230.

[4] OPC Foundation, *OPC Unified Architecture Specification*, OPC Foundation, 2006. [Online]. Available: https://opcfoundation.org

[5] “Data integration between mixed reality software and industrial robot controller via opc ua,” in *AIS 2024 - 19th International Symposium on Applied Informatics and Related Areas*. , author = Szila´gyi Zolta´n, Hajnal E´ va, Cso´ka Imre, Be´res Pe´ter, Sze´ll Ka´roly, Galambos Pe´ter, urldate = 2025, date = 2024,.

[6] OASIS, *MQTT Version 3.1.1*, OASIS, 2014.

[7] *Information technology – The JSON data interchange format*, ISO/IEC Std., 2017.

[8] Z. Szila´gyi, K. Sze´ll, and P. Galambos, “Middleware for mixed reality integration of industrial robot cells,” in *2024 IEEE 6th International Symposium on Logistics and Industrial Informatics (LINDI)*. IEEE, pp. 221–226.