INTERNATIONAL STANDARD

ISO/IEC 20237

First edition
2023-10

Information technology — Sparkplug® version 3.0

Reference number
ISO/IEC 20237:2023(E)

COPYRIGHT PROTECTED DOCUMENT

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

ISO and IEC draw attention to the possibility that the implementation of this document may involve the use of (a) patent(s). ISO and IEC take no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, ISO and IEC had not received notice of (a) patent(s) which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at www.iso.org/patents and https://patents.iec.ch. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by OASIS (as Sparkplug 3.0.0, Sparkplug Specification) and drafted in accordance with its editorial rules. It was adopted, under the JTC 1 PAS procedure, by Joint Technical Committee ISO/IEC JTC 1, *Information technology*.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

Table of Contents

Revision Number	Date	Author	Description
1.0	5/26/16	Cirrus Link	Initial Release
2.1	12/10/16	Cirrus Link	Payload B Addition
2.2	10/11/19	Cirrus Link	Re-branding for Eclipse foundation added TM to Sparkplug
3.0.0	11/16/22	Eclipse Sparkplug Specification Project Team	Reorganized to be in AsciiDoc format and to include normative and non-normative statements

Sparkplug®, Sparkplug Compatible, and the Sparkplug Logo are trademarks of the Eclipse Foundation.

1. Introduction

1.1. Rationale and Use Case

Eclipse Sparkplug provides an open and freely available specification for how Edge of Network Gateways (Sparkplug Edge Nodes) or native MQTT enabled end devices and Sparkplug Host Applications communicate bi-directionally within an MQTT Infrastructure. This document details the structure and implementation requirements for Sparkplug compliant MQTT Client implementations on both Edge Nodes and Host Applications.

It is recognized that MQTT is used across a wide spectrum of application solution use-cases, and an almost indefinable variation of network topologies. To that end the Sparkplug Specification strives to accomplish the three following goals.

1.1.1. Define an MQTT Topic Namespace

As noted many times in this document one of the many attractive features of MQTT is that it does not specify any required MQTT Topic Namespace within its implementation. This fact has meant that MQTT has taken a dominant position across a wide spectrum of IoT solutions. The intent of the Sparkplug Specification is to identify and document a Topic Namespace that is well thought out and optimized for the SCADA/IIoT solution sector. In addition, Sparkplug defines a Topic Namespace in such a way that it provides semantics which allow for automatic discovery and bi-directional communication between MQTT clients in a system.

1.1.2. Define MQTT State Management

One of the unique aspects of MQTT is that it was originally designed for real time SCADA systems to help reduce data latency over bandwidth limited and outage prone network infrastructures. These can include cellular, satellite, and other radio based networks. In many implementations the full benefit of this “Continuous Session Awareness” is not well understood, or not even implemented. The intent of the Sparkplug Specification is to take full advantage of MQTT’s native Continuous Session Awareness capability as it applies to real time SCADA/IIoT solutions.

It is important to note that reducing bandwidth usage and being resilient to network drops is advantageous on more reliable and high bandwidth networks as well. By reducing the bandwidth usage, Sparkplug is able to move more data through the network because of its efficiency. This in turn can reduce network costs.

1.1.3. Define the MQTT Payload

Just as the MQTT Specification does not dictate any particular Topic Namespace, it also does not dictate any particular payload data encoding. The intent of the Sparkplug Specification is to define payload encoding mechanisms that remain true to the original, lightweight, bandwidth efficient, low latency features of MQTT while adding modern encoding schemes targeting the SCADA/IIoT solution space.

Sparkplug has defined an approach where the Topic Namespace can aid in the determination of the encoding scheme of any particular payload. Historically there have been two Sparkplug defined encoding schemes. The first one was the 'Sparkplug A' and the second is 'Sparkplug B'. Each of these uses a 'first topic token identifier' so Sparkplug Edge Nodes can declare the payload encoding scheme they are using. These first topic tokens are:

```
spAv1.0
spBv1.0
```

Each token is divided up into three distinct components. These are:

- Sparkplug Identifier
  - Always 'sp'
- Payload Encoding Scheme
  - Currently 'A' or 'B' but there could be future versions
- Payload Encoding Scheme Version
  - Currently v1.0 but denoted in the event that future versions are released

The original 'Sparkplug A' encoding scheme was based on the Eclipse Kura™ open source Google Protocol Buffer definition. 'Sparkplug B' was released shortly after the release of Sparkplug A and addressed a number of issues that were present in the A version of the payload encoding scheme. Due to lack of adoption and the fact that 'Sparkplug B' was made available shortly after the release of 'A', the Sparkplug A definition has been omitted from this document and is no longer supported.

The 'Sparkplug B' encoding scheme was created with a richer data model developed with the feedback of many system integrators and end user customers using MQTT. These additions included metric timestamp support, complex datatype support, metadata, and other improvements.

1.1.4. Background

MQTT was originally designed as a message transport for real-time SCADA systems. The MQTT Specification does not specify the Topic Namespace nor does it define the Payload representation of the data being published and/or subscribed to. In addition to this, since the original use-case for MQTT was targeting real-time SCADA, there are mechanisms defined to provide the state of an MQTT session such that SCADA/Control Human-Machine Interface (HMI) application can monitor the current state of any MQTT enabled device in the infrastructure. As with the Topic Namespace and Payload the way state information is implemented and managed within the MQTT infrastructure is not defined. All of this was intentional within the original MQTT Specification to provide maximum flexibility across any solution sector that might choose to use MQTT infrastructures.

But at some point, for MQTT based solutions to be interoperable within a given market sector, the Topic Namespace, Payload representation, and session state must be defined. The intent and purpose of the Sparkplug Specification is to define an MQTT Topic Namespace, payload, and session state management that can be applied generically to the overall IIoT market sector, but specifically meets the requirements of real-time SCADA/Control HMI solutions. Meeting the operational requirements for these systems will enable MQTT based infrastructures to provide more valuable real-time information to Line of Business and MES solution requirements as well.

The purpose of the Sparkplug Specification is to remain true to the original notion of keeping the Topic Namespace and message sizes to a minimum while still making the overall message

transactions and session state management between MQTT enabled devices and MQTT SCADA/IIoT applications simple, efficient, easy to understand, and implement.

1.2. Intellectual Property Rights

1.2.1. Disclaimers

1.3. Organization of the Sparkplug Specification

This specification is split into the following chapters and appendices:

1.4. Terminology

1.4.1. Infrastructure Components

This section details the infrastructure components implemented.

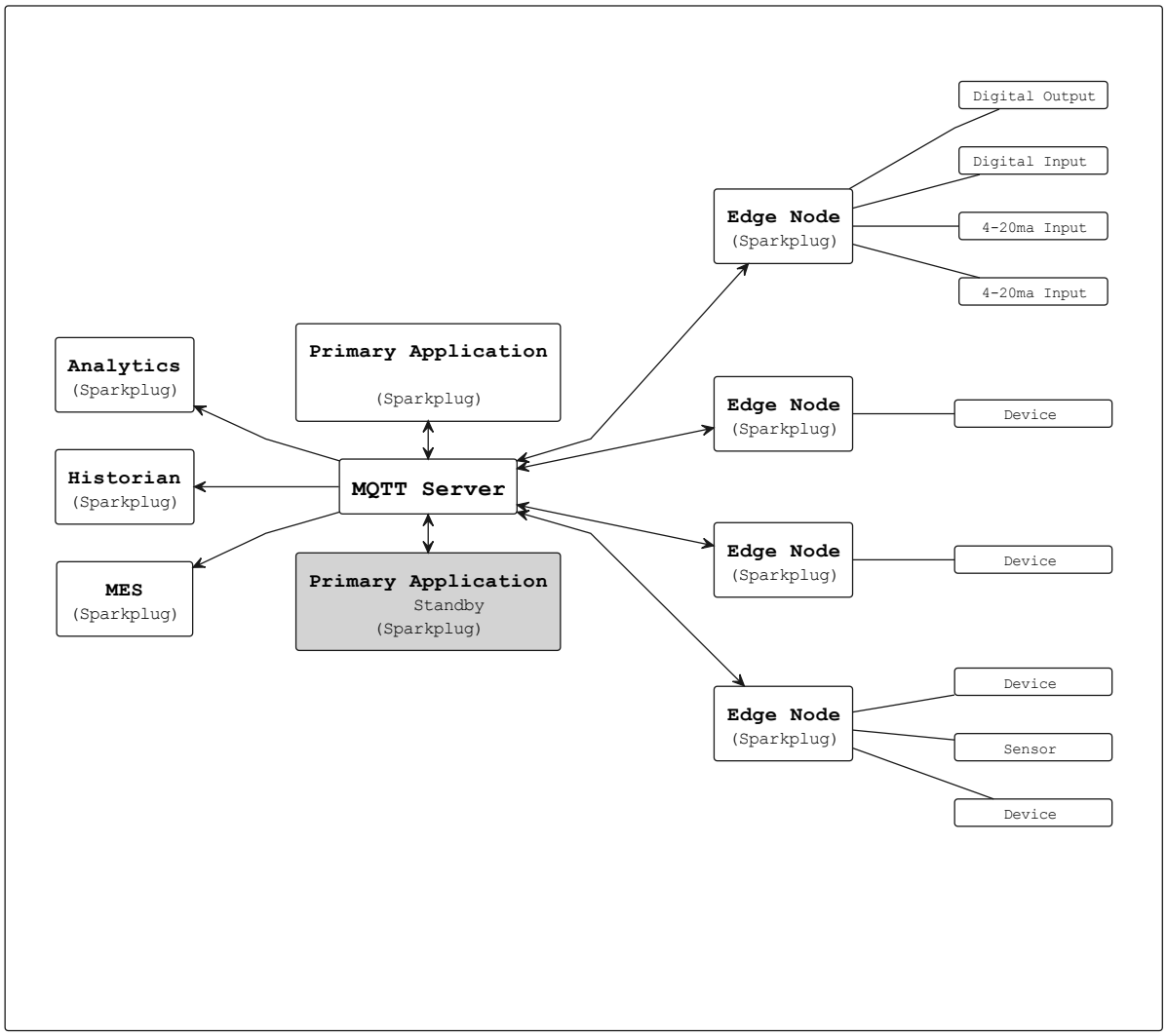

Figure 1 - MQTT SCADA Infrastructure

MQTT Server(s)

Program or device that acts as an intermediary between Clients which publish Application Messages and Clients which have made Subscriptions[MQTTV5-1.2]. MQTT enabled infrastructure requires that one or more MQTT Servers are present in the infrastructure. An MQTT Server must be compatible with the requirements outlined in the Conformance Section. In addition, it must be sized to properly manage all MQTT message traffic.

One can implement the use (if required) of multiple MQTT servers for redundancy, high availability, and scalability within any given infrastructure.

Sparkplug Group

Logical or physical group of Edge Nodes that makes sense in the context of a distributed Sparkplug application. Groups can represent physical groups of Edge Nodes. For example, a

Sparkplug Group could represent a set of Edge Nodes at a particular location, facility, or along a specific oil pipeline. Alternatively, a Sparkplug Group could represent group of similar types of Edge Nodes. For example, it could represent a particular set of like make and models of embedded gateways. The groups are meant to be defined by the system architects as appropriate for their particular application.

Sparkplug Edge Node

Any v3.1.1 or v5.0 compliant MQTT Client application that manages an MQTT Session and provides the physical and/or logical gateway functions required to participate in the Topic Namespace and Payload definitions described in this document. The Edge Node is responsible for any local protocol interface to existing devices (PLCs, RTUs, Flow Computers, Sensors, etc.) and/or any local discrete I/O, and/or any logical internal process variables (PVs).

Sparkplug Device

Physical or logical device that makes sense in the context of a distributed Sparkplug application. Often times a Sparkplug Device will be a physical PLC, RTU, Flow Computer, Sensor, etc. However, a Sparkplug device could also represent a logical grouping of data points as makes sense for the specific Sparkplug Application being developed. For example, it could represent a set of data points across multiple PLCs that make up a logical device that makes sense within the context of that application.

MQTT/Sparkplug Enabled Device

Any device, sensor, or hardware that directly connects to MQTT infrastructure using a compliant MQTT v3.1.1 or v5.0 connection with the payload and topic notation as outlined in this Sparkplug Specification. With MQTT/Sparkplug enabled directly in the device this could bypass the use of a Sparkplug Edge Node in the infrastructure. In this case, the physical device or sensor is the Edge Node. It is up to the developer of the application to decide if the concept of a 'Sparkplug Device' is to be used within their application.

Host Applications

Application that consumes data from Sparkplug Edge Nodes. Depending on the nature of the Host Application it may consume Edge Node data and display it in a dashboard, it may historize the data in a database, or it may analyze the data in some way. SCADA/IIoT Hosts, MES, Historians, and Analytics applications are all examples of potential Sparkplug Host Applications. A Host Application may perform many different functions in handling the data. In addition, Host Applications may also send Sparkplug NCMD or DCMD messages to Edge Nodes.

A Sparkplug Edge Node may specify one Host Application as its 'Primary Host Application'. This is handled by the Edge Node waiting to publish its NBIRTH and DBIRTH messages until the Host Application that the Edge Node has designated as its Primary Host application has come online. Sparkplug does not support the notion of multiple Primary Host Applications. This does not preclude any number of additional Host Applications participating in the infrastructure that are in either a pure monitoring mode, or in the role of a hot standby should the Edge Node's Primary Host Application go offline or become unavailable within the infrastructure.

[tck-id-intro-sparkplug-host-state] Sparkplug Host Applications MUST publish STATE messages denoting their online and offline status.

Primary Host Application

Most important consumer of Sparkplug Edge Node data. The Primary Host Application must be online to keep operations running.

A Primary Host Application may be defined by an Edge Node. The Edge Node's behavior may change based on the status of its configured Primary Host. It is not required that an Edge Node must have a Primary Host configured but it may be useful in certain applications. This allows Edge Nodes to make decisions based on whether or not the Primary Host Application is online or not. For example, an Edge Node may store data at the edge until a Primary Host Application comes back online. When the Primary Host Application publishes a new STATE message denoting it is online, the Edge Node can resume publishing data and also flush any historical data that it may have stored while offline.

In a traditional SCADA system the SCADA Host would be the Primary Host Application. With this same concept in mind, there can only be one Primary Host Application configured in an Edge Node as a result.

Sparkplug Identifiers

Sparkplug defines identifiers or IDs for different physical or logical components within the infrastructure. There are three primary IDs and one that is a composite ID. These are defined as the following.

- Group ID
  - **[tck-id-intro-group-id-string] The Group ID MUST be a UTF-8 string and used as part of the Sparkplug topics as defined in the Topics Section.**
  - **[tck-id-intro-group-id-chars] Because the Group ID is used in MQTT topic strings the Group ID MUST only contain characters allowed for MQTT topics per the MQTT Specification.**
    - Non-normative comment: The Group ID represents a general grouping of Edge Nodes that makes sense within the context of the Sparkplug application and use-case.
- Edge Node ID
  - **[tck-id-intro-edge-node-id-string] The Edge Node ID MUST be a UTF-8 string and used as part of the Sparkplug topics as defined in the Topics Section.**
  - **[tck-id-intro-edge-node-id-chars] Because the Edge Node ID is used in MQTT topic strings the Edge Node ID MUST only contain characters allowed for MQTT topics per the MQTT Specification.**
    - Non-normative comment: The Edge Node ID represents a unique identifier for an Edge Node within the context of the Group ID under which it exists.
- Device ID
  - **[tck-id-intro-device-id-string] The Device ID MUST be a UTF-8 string and used as part of the Sparkplug topics as defined in the Topics Section.**

  - **[tck-id-intro-device-id-chars] Because the Device ID is used in MQTT topic strings the Device ID MUST only contain characters allowed for MQTT topics per the MQTT Specification.**
    - Non-normative comment: The Device ID represents a unique identifier for a Device within the context of the Edge Node ID under which it exists.

- Edge Node Descriptor (composite ID)
  - The Edge Node Descriptor is the combination of the Group ID and Edge Node ID.
  - **[tck-id-intro-edge-node-id-uniqueness] The Edge Node Descriptor MUST be unique within the context of all of other Edge Nodes within the Sparkplug infrastructure.**
    - In other words, no two Edge Nodes within a Sparkplug environment can have the same Group ID and same Edge Node ID.
    - Non-normative comment: The Device ID represents a unique identifier for a Device within the context of the Edge Node ID under which it exists.

Sparkplug Metric

Identifies a single 'tag change event' in the Sparkplug Payload. It represents an event that occurred at the Edge Node or Device such as a value or quality of a data point changing. For example, it could represent the value of an analog or boolean changing at a Sparkplug Device. A Sparkplug Metric typically includes a name, value, and timestamp. Sparkplug Metrics are also used in NCMD and DCMD messages to send messages to Edge Nodes and Devices to change values at the Edge.

Data Types

There are different uses of the term 'datatype' in the specification. Sparkplug encodes the payloads using Google Protocol Buffers. Google Protocol Buffers has its own scalar value types here: https://developers.google.com/protocol-buffers/docs/proto#scalar

The Google Protocol Buffer datatypes define what actually travels over the TCP/IP socket in the MQTT payload. For ease of programming, Google Protobuf includes a compiler tool that generates code in multiple different languages. These Protobuf datatypes are then represented by their proper native programming language datatypes. This is done on a per language basis after the Google Protobuf file is used to generate the code for each specific language.

In addition to Protobuf datatypes and native programming language datatypes there are also 'Sparkplug datatypes'. These are defined in the Sparkplug Protobuf Schema. These datatypes are those that are used for Sparkplug Metrics. Every Metric must include a Sparkplug Datatype in the NBIRTH or DBIRTH message depending on whether the Metric is a 'Node level' or 'Device level' metric. Each of the Sparkplug Datatypes is then represented by a Google Protobuf datatype.

1.5. Normative References

- [BCP14] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997. Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, May 2017.

- [MQTTV50] MQTT Version 5.0. Edited by Andrew Banks, Ed Briggs, Ken Borgendale, and Rahul Gupta. 07 March 2019. OASIS Standard. https://docs.oasis-open.org/mqtt/mqtt/v5.0/os/mqtt-v5.0-os.html. Latest version: https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html.
- [MQTTV311] MQTT Version 3.1.1 Plus Errata 01. Edited by Andrew Banks and Rahul Gupta. 10 December 2015. OASIS Standard Incorporating Approved Errata 01. http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/errata01/os/mqtt-v3.1.1-errata01-os-complete.html. Latest version: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html.
- [ISO/IEC 20922:2016] Information technology — Message Queuing Telemetry Transport (MQTT) v3.1.1