Transport Certification Australia

Telematics Device to Drowsiness Device

Communications Protocol

April 2019

This document has been published by
Transport Certification Australia Limited.

Transport Certification Australia Ltd
T +61 3 8601 4600
F +61 3 8601 4611
E tca@tca.gov.au
W www.tca.gov.au

ABN 83 113 379 936

Document Details

Title	Telematics Device to Drowsiness Device Communications Protocol
Document Number	TCA-G46
Version	1.0
Version Date	April 2019
Printing Instructions	Double sided, colour

Document History

Version	Date	Description
0.1	January 2019	First version for working group comment
0.2	March 2019	Updated draft based on working group feedback
1.0	April 2019	First release

Transport Certification Australia Limited believes this publication to be correct at time of printing and does not accept responsibility for any consequences arising from the use of information herein. Readers should rely on their own skills and judgment to apply information to particular issues.

NATIONAL TELEMATICS FRAMEWORK

The National Telematics Framework is a digital business platform consisting of infrastructure and rules that support an open marketplace of telematics and related intelligent technology providers.

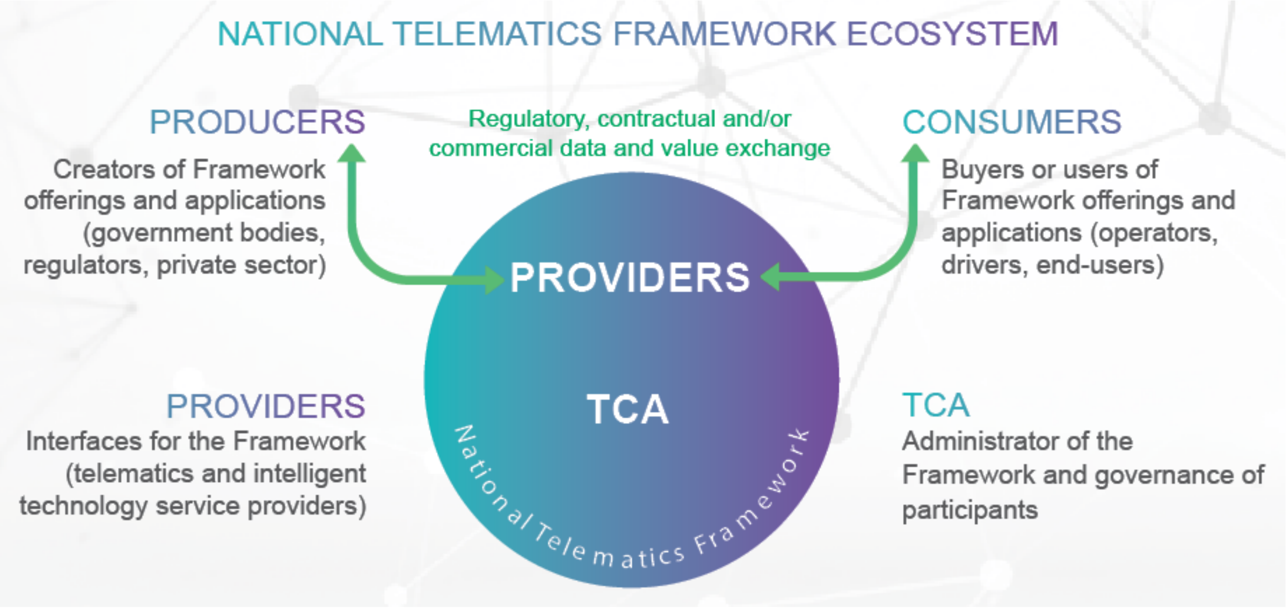
The National Telematics Framework:

- Provides a national platform for the use of telematics and related intelligent technologies
- Supports different applications across regulatory, contractual and commercial needs
- Supports different levels of assurance
- Is outcome focussed and encourages innovation.

The adoption of the National Telematics Framework for the delivery of offerings and applications both for public policy and private decision making is a world first. It has positioned Australia as the leader in the delivery of such services through the advent of the digital economy.

The National Telematics Framework was established following a series of decisions made by Responsible Ministers between 2003 and 2008, and was globally recognised as an International Standard (ISO 15638) in 2012.

Figure 1: National Telematics Framework Ecosystem



NATIONAL TELEMATICS FRAMEWORK ECOSYSTEM

PRODUCERS
Creators of Framework offerings and applications (government bodies, regulators, private sector)

Regulatory, contractual and/or commercial data and value exchange

CONSUMERS
Buyers or users of Framework offerings and applications (operators, drivers, end-users)

PROVIDERS
Interfaces for the Framework (telematics and intelligent technology service providers)

TCA
Administrator of the Framework and governance of participants

Contents

FIGURES

TABLES

APPENDICES

1 INTRODUCTION

1.1 Overview

Drowsiness monitoring technologies capable of measuring one or more physiological signs of drowsiness in drivers are now widely available.

Transport Certification Australia (TCA) has worked with telematics providers and suppliers of fatigue management devices to develop a protocol that:

- Enables standardisation and interoperability
- Removes barriers to the use of fatigue management devices
- Provides easy adoption by technology providers.

The protocol describes the arrangements for transferring information between telematics devices (the primary telematics unit which monitors vehicle parameters) and drowsiness monitoring devices (a smart device measuring the drowsiness of a driver).

Forming part of the National Telematics Framework the protocol promotes standardisation and interoperability in the provision of technology and services by providers.

1.2 Document Structure

This document contains the following sections:

- About This Document
- Technology Stack – describes the high-level technology used to deliver the protocol
- Message Design – describes the messaging between devices
- Message Specification – describes the content of each message
- Appendix A – provides a JSON example and schema for the protocol.

1.3 About This Document

This document is targeted primarily at providers (refer to Figure 1), but can also be referred to by producers and consumers seeking standardisation and interoperability between telematics devices and drowsiness monitoring devices.

Please note that this document incorporates technical subject matter which assumes a level of knowledge expected of providers of the National Telematics Framework, but which may not be familiar to other stakeholder groups.

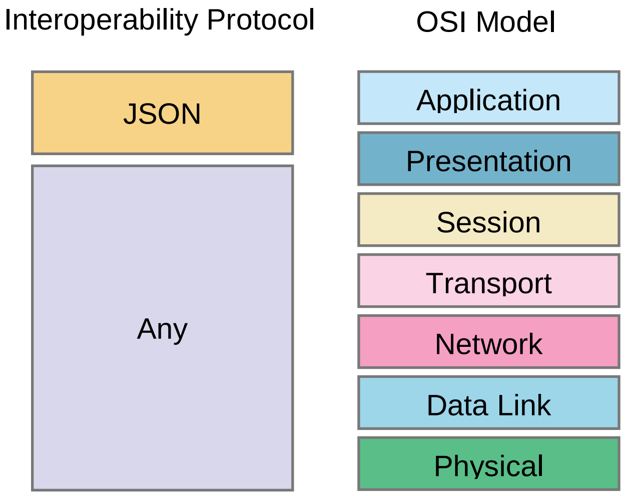
For background knowledge or further information on this subject, send an email to **tca@tca.gov.au**.

2 TECHNOLOGY STACK

This section describes the technology stack for the protocol.

The protocol uses JSON for upper layer communication, with any lower layer technology stack as selected by the provider. Figure 2 shows a comparison of the protocol with Open Systems Interconnection (OSI) model as a reference for developers.

Figure 2: National Telematics Framework Ecosystem



Accommodating the use of any lower layer stack allows individual providers (and devices) to support their own communication mechanisms.

Defining messages in JSON allows different devices to share a common language once lower layer communications are established.

JSON use is defined in RFC 8259 and implemented using UTF-8 text encoding.

Data types within messages can be: Number, Integer, String, Boolean, Array, Object, null.

A JSON schema is provided in Appendix A, Section A.3, to formally define message format, and can be used to validate implementations.

Note: Security is not specified in this protocol. Developers are encouraged to implement appropriate security through lower layer protocols.

3 MESSAGE DESIGN

This section describes message design and usage.

The protocol incorporates bi-directional, asynchronous messaging between two connected devices.

Two message types are defined in this protocol:

Heartbeat message: *Used to frequently communicate device status and sensor data[1].*

No set messaging frequency is defined. While the heartbeat contains useful status information, it is not required to keep the devices 'active'. The decision for heartbeat frequency may be informed by the lower layer communications used (e.g. to avoid timeouts and sessions ending). Nominally, the heartbeat message is expected approximately every 30 seconds. The heartbeat message contains the message header and zero or more message containers, depending on the information to be conveyed.

Event message: *Used to communicate specific events that occur and are reported.*

The event message contains the message header and the event message container. The event message container supplies the event type and event data. Due to device operation not being standardised, events are defined at a high level and event data is not standardised. Instead, the device populates the event data in its own format for later interpretation. The threshold to trigger each event is not defined. Each of these may be further defined in future versions of the protocol.

Although a JSON scheme is provided to validate implementations, schema validation is unlikely to be used at runtime and it is possible errors will occur within messages due to sensor failures, unexpected bugs, etc.

When a message is received that cannot be parsed, it is expected that either the whole message is discarded, or the offending field(s) are discarded. In either case, a 'MESSAGE_ERROR' response should be sent.

Note: Messageacknowledgement is not built into the messaging but may be implemented at lower layers.

[1] Here, sensor data refers to any information collected by or available to the device including positioning data, vehicle data, driver data and drowsiness data.

4 MESSAGE SPECIFICATION

This section describes the contents, fields, data formats and use of each message and message containers.

4.1 Message Header

Use the header in Table 1 for each message (i.e. both heartbeat and event messages).

Table 1: Message Header Format

Element Name	Type	Units / enum	Description
protocolVersion	String		Current version of the protocol – 1.0
messageNumber	Integer		A sequential number used in each message sent by the device using this protocol, used to determine message order and missing messages
messageDateTime	String	yyyy-mm-ddTnn:nn:nn.nnnZ	Date and time of the sent message formatted as per ISO 8601
messageType	String	HEARTBEAT; EVENT	Name of message type being sent
deviceId	String		Identifier of the device sending the message

deviceStatus	Array	OK;
		LOW_BATTERY;
		MESSAGE_ERROR;
		DEVICE_FAULT;
		UNSUPPORTED_MESSAGE;
		VALIDATION_ERROR
			Report of device status and various issues. Either report 'OK' only, or one or more other statuses
			See Table 2 for explanation of status meanings.

Table 2 provides detail for the use of enumeration of device status.

Table 2: Device Status Enumeration

Enumeration	Use
OK	No issues / normal operation
LOW_BATTERY	Internal battery is low
MESSAGE_ERROR	The last-received message contains an error and cannot be read
DEVICE_FAULT	Some fault is detected on the device and may affect messaging
UNSUPPORTED_MESSAGE	The last-received message type is not supported
VALIDATION_ERROR	The last-received message failed validation/security credentials

4.2 Heartbeat Message

This section describes the heartbeat message of the data profile.

It is used to communicate frequently between the drowsiness and telematics device containing basic information about the status of the device and what it is measuring.

A heartbeat message contains a mandatory message header and zero or more message containers. This section contains tables detailing the contents of each message container.

Information relevant to a device type is organised in different containers that can be used optionally. The currently supported heartbeat message containers are provided in Table 3.

Table 3: Heartbeat Message Container Types

Container Type	Description
Positioning container	Used to communicate a device's measure of current position
Vehicle container	Used to communicate the current vehicle in use and its attributes
Driver container	Used to communicate the driver and Hours of Service status
Drowsiness container	Used to communicate a device's measure of driver drowsiness

The contents of each message container are provided in the following tables. Tables 4-7 describe the content and usage of the positioning, vehicle, driver and drowsiness container respectively.

Table 4: Positioning Container

Element Name	Type	Units / enum	Description
satelliteCount	Number		Number of satellites used to determine position. Where no position can be established, populate as zero satellites and omit the remaining fields of the positioning container. If there are non-zero satellites, the latitude, longitude and HDOP fields of the positioning container are mandatory.
latitude	Number	decimal degrees	Measured in decimal degrees; resolved to at least 5 decimal places
longitude	Number	decimal degrees	Measured in decimal degrees; resolved to at least 5 decimal places
hdop	Number		HDOP value for measured position. *Note: Omitted when zero satellites are used.*
altitude	Number	metres	Elevation as per mean sea level (MSL) in metres. *Note: Accuracy in good conditions is in the range of +/- 25 m*
vdop	Number		VDOP value for a measured position where altitude is used. *Note: Omitted when zero satellites are used.*
gpsSpeed	Number	km/h	GPS-derived speed measurement, such as Doppler

Table 5: Vehicle Container

Element Name	Type	Units / enum	Description
vehicleId	String		Registration number or another identifier for the vehicle
vehicleSpeed	Number	km/h	Any non-GPS based determination of speed. Reverse gears should be represented as a negative number.
steeringAngle	Number	decimal degrees	Steering angle expressed as positive (right turn), negative (left turn) or zero (straight), value range of -180 to 180.
vehicleType	String		Vehicle type free text used to describe vehicle class. Suggested enumeration: Light; Heavy; Bus; Taxi
vehicleConfiguration	String		Vehicle configuration free text used to describe vehicle combination

Table 6: Driver Container

Element Name	Type	Units / enum	Description
driverId	String		Persistent identifier for the current driver
drivingDuration	String	PT[n]H[n]M[n]S	Driving time since last break, formatted as per ISO 8601. Omit Y, M and D elements
serviceDuration	String	PT[n]H[n]M[n]S	Total number of work hours for the shift, formatted as per ISO 8601. Omit Y, M and D elements
shiftStartTime	String	yyyy-mm-ddTnn:nn:nn.nnnZ	Start date/time of the shift used to determine ‘serviceDuration’. Formatted as per ISO 8601.
drivingArrangement	String		Driving arrangement as either SOLO or TWO_UP. Additional values may be supported in future updates to this specification.

Table 7: Drowsiness Container

Element Name	Type	Units / enum	Description
drowsinessValue	Number		Proprietary value for the drowsiness measurement
drowsinessMetric	String		Free text name of measurement used
drowsinessDisplay	String		Proprietary description of the current drowsiness state. Can be used where 'drowsinessValue' is not appropriate, or to send a human-readable message on the drowsiness state.
technologyType	String		Free text name of the technology used to determine measurement. Used where the same drowsiness metric could be measured with two different technologies.

4.3 Event Message

This section describes the event message of the data profile.

Event messages are used to send notifications that are not otherwise communicated in the heartbeat. They are designed to be triggered rather than status-based data.

Each event message contains an event type and event data. Event data is flexible and may vary between systems, depending on the thresholds for triggering each event type.

Event triggers and event data are not defined and are left to developers to implement. These may be further defined in future versions of the protocol.

Table 8: Event Message Container

Element Name	Type	Units / enum	Description

eventType	String	HARSH_BRAKING;
		HARSH_STEERING;
		HARSH_ACCELERATION;
		SPEEDING_EVENT;
		LANE_DEPARTURE;
		DROWSINESS_EVENT;
		REST_BREAK_REQUIRED
			An enumerated string describing the type of event that occurred

eventData	String		Free text string to describe the event.

The proposed protocol incorporates a placeholder list of event definitions is provided to assist in generating and interpreting event messages.

Table 9: Event Type Definitions

Event Type	Description
HARSH_BRAKING	Vehicle deceleration exceeds threshold
HARSH_STEERING	Rate of change of steering angle exceeds threshold
HARSH_ACCELERATION	Vehicle acceleration exceeds threshold
SPEEDING_EVENT	Exceeding a speed threshold or speed limit detected
LANE_DEPARTURE	Lane departure warning activated
DROWSINESS_EVENT	Driver drowsiness exceeds threshold or changes risk category
REST_BREAK_REQUIRED	A rest break is recommended based on hours of service / hours of driving

Appendix A JSON Specification and Examples

A.1 Example Heartbeat Message

```
{
  "deviceId": "device id",
  "deviceStatus": ["OK"],
  "messageNumber": 5,
  "protocolVersion": "1.0",
  "messageDateTime": "2018-12-12T12:13:13Z",
  "messageType": "HEARTBEAT",
  "content": {

    "positioningStatus": {
      "satelliteCount": 2,
      "hdop": 2.3,
      "latitude": -36.31234,
      "longitude": 136.00,
              "altitude": 100,
       "vdop": 2.1,
      "gpsSpeed": 23.3
    },

    "vehicleStatus": {
      "vehicleId": "1234",
      "steeringAngle": -23.3,
      "vehicleType": "big thing",
      "vehicleConfiguration": "P2-44"
    },

    "driverStatus": {
      "driverId": "1q234",
      "drivingDuration": "PT1H",
      "serviceDuration": "PT3H30M",
              "shiftStartTime": "2018-12-12T09:09:02Z",
      "drivingArrangement": "SOLO"
    },

    "drowsinessStatus": {
      "drowsinessValue": 123,
      "drowsinessMetric": "d",
      "drowsinessDisply": "ok",
       "technologyType": "whizbang"
    }

  }

}
```

A.2 Example Event Message

```json
{
  "deviceId": "device id",
  "deviceStatus": ["OK"],
  "messageNumber": 5,
  "protocolVersion": "1.0",
  "messageDateTime": "2018-12-12T12:13:13Z",
  "messageType": "EVENT",
  "content": {
    "genericEvent": {
      "eventType": "HARSH_BRAKING",
      "eventData": "asdfasdf"
    }
  }
}
```

A.3 JSON Schema

```
{
   "$id":"http://www.tca.gov.au/schemas/fatigue/protocol/2018-12",
   "$schema": "http://json-schema.org/draft-07/schema#",
   "description": "Telematics Drowsiness Device Communications Protocol",
   "type": "object",
   "required": ["protocolVersion", "messageNumber", "messageDateTime", "deviceId", "deviceStatus"],
   "properties": {
      "protocolVersion": {
         "type": "string",
         "const": "1.0"
      },
      "messageNumber": {
         "type": "integer",
         "minimum": 1
      },
      "messageDateTime": {"$ref": "#typeDateTime" },
      "messageType": {"$ref": "#enumMessageType" },
      "deviceId": {
         "type": "string",
         "minLength": 1
      },
      "deviceStatus":  {
         "type": "array",
         "minItems": 1,
         "if": {
            "description": "test if length of device status is one ...",
            "maxItems": 1
         },
         "then": {
            "description": "... in which case 'OK' is an allowable value",
            "items": {
               "anyOf": [
                  { "const": "OK" },
                  { "$ref": "#enumDeviceStatus" }
               ]
            }
         },
         "else": {
            "description": "... otherwise 'OK' is not an allowable value",
            "items": {"$ref": "#enumDeviceStatus" }
         }
      },
      "content": {
         "description": "payload container where content is inserted - if no content then omit this
            property",
         "type": "object",
         "minProperties": 0,
         "properties" : {
            "positioningStatus": {"$ref": "#statusPositioning"},
            "vehicleStatus": {"$ref": "#statusVehicle"},
            "driverStatus": {"$ref": "#statusDriver"},
            "drowsinessStatus": {"$ref": "#statusDrowsiness"},
            "genericEvent": {"$ref": "#eventGeneric"}
         }
      }
   },
   "if" : {
      "description": "test if this is a HEARTBEAT message",
      "properties": { "messageType": { "const": "HEARTBEAT" } }
   },
   "then" : {
      "description": "for HEARTBEAT messages all content property names must end with 'Status'",
      "properties": {
         "content": {
            "propertyNames": {"pattern": "^.*Status$"}
        }
      }
   },
   "else" : { "if" : {
      "description": "test if this is an EVENT message",
      "properties": { "messageType": { "const": "EVENT" } }
   },
   "then" : {
      "description": "for EVENT messages all content property names must end with 'Event', and only
         one content property is allowed",
      "properties": {
         "content": {
            "propertyNames": {"pattern": "^.*Event$"},
            "maxProperties": 1
         }
      }
   } },

   "definitions": {

      "type.iso8601.dateTime" : {
         "id": "#typeDateTime",
         "type": "string",
         "pattern": "^[0-9]{4}-[0-9]{2}-[0-9]{2}T[0-9]{2}:[0-9]{2}:[0-9]{2}Z$"
      },

      "type.iso8601.duration" : {
         "id": "#typeDuration",
         "type": "string",
         "pattern": "^PT([0-9]+H)?([0-9]+M)?([0-9]+S)?$"
      },

      "enum.messageType": {
         "id": "#enumMessageType",
         "type": "string",
         "enum": ["HEARTBEAT", "EVENT"]
      },

      "enum.deviceStatus": {
         "id": "#enumDeviceStatus",
         "type": "string",
         "enum": ["LOW_BATTERY", "DEVICE_FAULT", "MESSAGE_ERROR", "UNSUPPORTED_MESSAGE",
            "VALIDATION_ERROR"]
      },

      "enum.eventType": {
         "id": "#enumEventType",
         "type": "string",
         "enum": ["HARSH_BRAKING", "HARSH_STEERING", "HARSH_ACCELERATION", "SPEEDING_EVENT",
            "LANE_DEPARTURE", "DROWSINESS_EVENT", "REST_BREAK_REQUIRED"]
      },

      "enum.drivingArrangement": {
         "id": "#enumDrivingArrangement",
         "type": "string",
         "enum": ["SOLO", "TWO_UP"]
      },
      "content.status.positioning" : {
         "id": "#statusPositioning",
         "description": "content type for POSITIONING container (HEARTBEAT message)",
         "type": "object",
         "required": ["satelliteCount"],
         "properties": {
            "latitude" : {
               "type": "number",
               "minimum": -90.0,
               "maximum": 90.0
            },
            "longitude" : {
               "type": "number",
               "minimum": -180.0,
               "maximum": 180.0
            },
            "satelliteCount" : {
               "type": "integer",
               "minimum": 0
            },
            "hdop" : {
               "type": "number",
               "minimum": 0.0,
               "maximum": 99.9
            },
            "gpsSpeed" : {
               "type": "number",
               "minimum": 0.0
            },
              "vdop" : {
               "type": "number",
               "minimum": 0.0,
               "maximum": 99.9
            },
            "altitude" : {
               "type": "number"
            }
         },
         "if" : {
            "description": "test if satellite count is greater than zero (indicator of signal
               available)",
            "properties": { "satelliteCount": { "minimum": 1 } }
         },
         "then" : {
            "description": "these values are expected if satellite count is greater than zero",
            "required": ["latitude", "longitude", "hdop"]
         }
      },
      "content.status,vehicle" : {
         "id": "#statusVehicle",
         "description": "content type for VEHICLE container (HEARTBEAT message)",
         "type": "object",
         "required": ["vehicleId"],
         "properties": {
            "vehicleId" : {
               "type": "string",
               "minLength": 1
            },
            "vehicleSpeed" : {
               "type": "number"
            },
            "steeringAngle" : {
               "type": "number",
               "minimum": -180.0,
               "maximum": 180.0
            },
            "vehicleType" : {
               "type": "string",
               "minLength": 1
            },
            "vehicleConfiguration" : {
               "type": "string",
               "minLength": 1
            }
         }
      },

      "content.status.driver" : {
         "id": "#statusDriver",
         "description": "content type for DRIVER container (HEARTBEAT message)",
         "type": "object",
         "required": ["driverId"],
         "properties": {
            "driverId" : {
               "type": "string",
               "minLength": 1
            },
            "drivingDuration" : {"$ref": "#typeDuration"},
            "serviceDuration" : {"$ref": "#typeDuration"},
            "shiftStartTime" : {"$ref": "#typeDateTime"},
            "drivingArrangement" : {"$ref": "#enumDrivingArrangement" }
         }
      },

      "content.status.drowsiness" : {
         "id": "#statusDrowsiness",
         "description": "content type for DROWSINESS container (HEARTBEAT message)",
         "type": "object",
         "required": ["drowsinessValue"],
         "properties": {
            "drowsinessValue" : {
               "type": "number"
            },
            "drowsinessDisplay" : {
```

```
               "type": "string",
               "minLength": 1
            },
            "drowsinessMetric" : {
               "type": "string",
               "minLength": 1
            },
            "technologyType" : {
               "type": "string",
               "minLength": 1
            }
         }
      },

      "content.event.generic" : {
         "id": "#eventGeneric",
         "description": "content type for EVENT container (EVENT message)",
         "type": "object",
         "required": ["eventType"],
         "properties": {
            "eventType": {"$ref": "#enumEventType" },
            "eventData": {
               "type": "string",
               "minLength": 1
            }
         }
      }

   }
}
```

Transport Certification Australia