Transport Certification Australia

INTERCONNECTIVITY OF TELEMATICS DEVICE WITH OTHER SYSTEMS

Functional and Technical Specification Version 1.1

MARCH 2020

www.tca.gov.au

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	Interconnectivity of Telematics Device with Other Systems Functional and Technical Specification
Document Number	TCA-S08
Version	1.1
Version Date	March 2020
Printing Instructions	Colour A4, double-sided

Document History

Version	Date	Description
1.0	April 2007	Final
1.1	March 2020	Replaced 'telematics in-vehicle unit' with 'telematics device', and applied to updated template.

ABOUT US

Transport Certification Australia (TCA) is the Australian entity responsible for providing assurance in the use of telematics and related intelligent technologies.

We manage the National Telematics Framework, which brings producers, providers and consumers together on a common digital business platform.

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.

Contents

FIGURES

TABLES

APPENDICES

1 INTRODUCTION

1.1 Purpose of this Specification

1.1.1 This specification sets out a standardised and interoperable communication interface between a telematics device and another system or device within the vehicle. As these typically provide discrete and complementary functionality, their interconnection allows for the efficient provision of a coherent and complete telematics solution, for example:

- the telematics device provides the capability to track time and position based upon a GNSS reference, and to maintain a mobile data network connection for forwarding of data records to the back office;
- the system or device will typically support a highly-specialised capability to monitor some aspect of vehicle operation, and thereby generate data records; and
- interconnection allows the system or device to synchronise its internal clock to that of the telematics device, to transfer generated data records to the telematics device for the purpose of forwarding to the back office, and to be directed by the telematics device to generate specific data records at specific points in time.

1.1.2 This specification is applicable only to the system-level interconnection of systems and devices with a telematics device; it is not applicable to interconnections between sub-components within a particular system or device.

1.2 Specification Overview

1.2.1 This specification commences with this introduction (Section 1), followed by the:

- background (Section 2);
- references applicable to this specification (Section 3);
- normative requirements of this specification (Section 4); and
- acronyms and definitions for the purpose of this specification (Appendix A).

1.3 Nomenclature

1.3.1 Requirements clauses within this specification that are denoted by:

- ‘shall’ are requirements that must be met;
- ‘should’ are requirements that should desirably be met;
- ‘may’ are requirements that are optional; and
- ‘will’ are obligations that will be met by other parties.

1.3.2 Notes are included by way of clarification and apply to the immediately preceding requirement.

1.3.3 Within this specification integer constants are represented in hexadecimal notation using the prefix ‘0x’. For example, 0x10 represents the decimal number 16. The prefix ‘0x’ is itself not data, and is not encoded within messages.

2 BACKGROUND

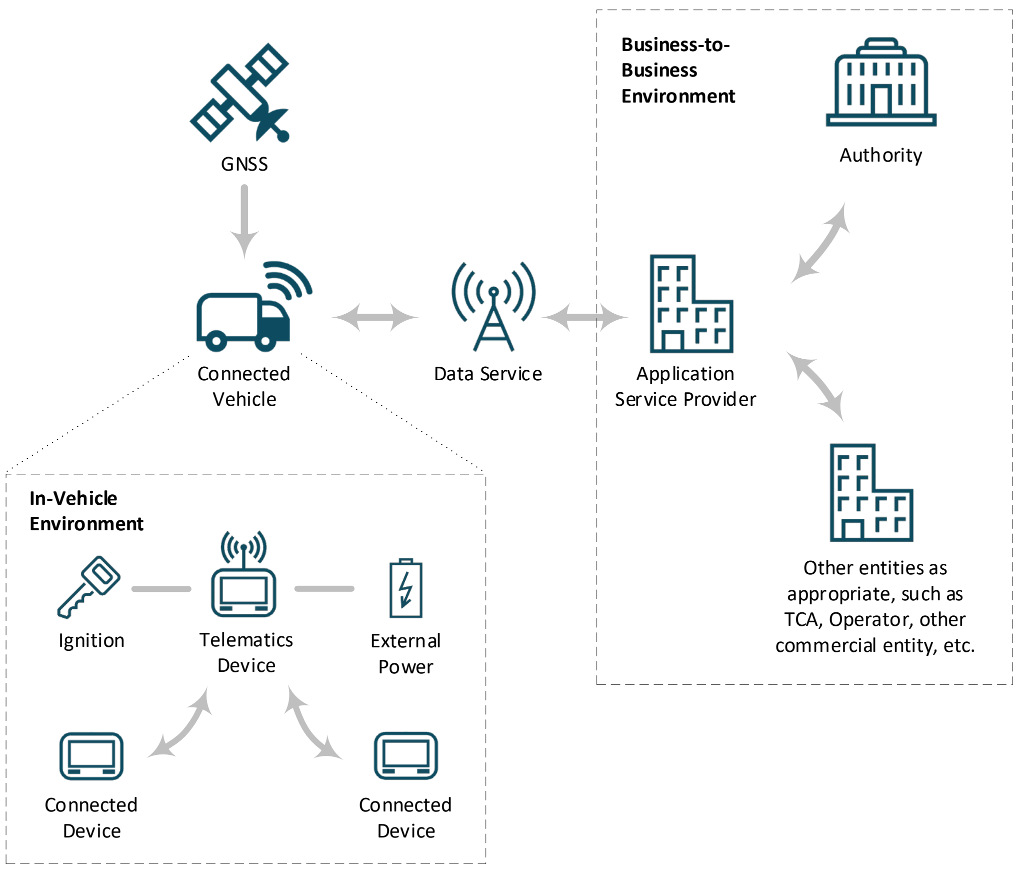
2.1 Context

2.1.1 As the telematics industry evolves, there will be an increasing number and variety of systems or devices available to support telematics applications. Where multiple such devices are installed in close proximity, for example in the same vehicle, it is possible that one device will have capabilities that are useful to other devices. As a notable example, a telematics device encompasses a GNSS-based time and position reference, and also a mobile data network connection to the back office. Both of these capabilities are of potential value to other systems or devices.

2.1.2 The opportunity to leverage the infrastructure capabilities of a telematics device gives rise to the requirement to interconnect systems and devices. Further, having a standard and interoperable specification for such interconnections allows for different combinations of systems or devices to communicate without requiring multiple application-specific communications interfaces to be defined.

2.1.3 The telematics device is the *primary* telematics unit which monitors parameters. For example, Figure 1 shows a telematics device interconnected to two different highly-specialised systems and devices, one of which may be a type-approved on-board mass (OBM) system. Acting as a telematics hub, the telematics device is able to share its GNSS-based time and position reference and mobile data network connection with each of the other systems and devices.

Figure 1: Example of Telematics Device Interconnected to Multiple Systems and Devices

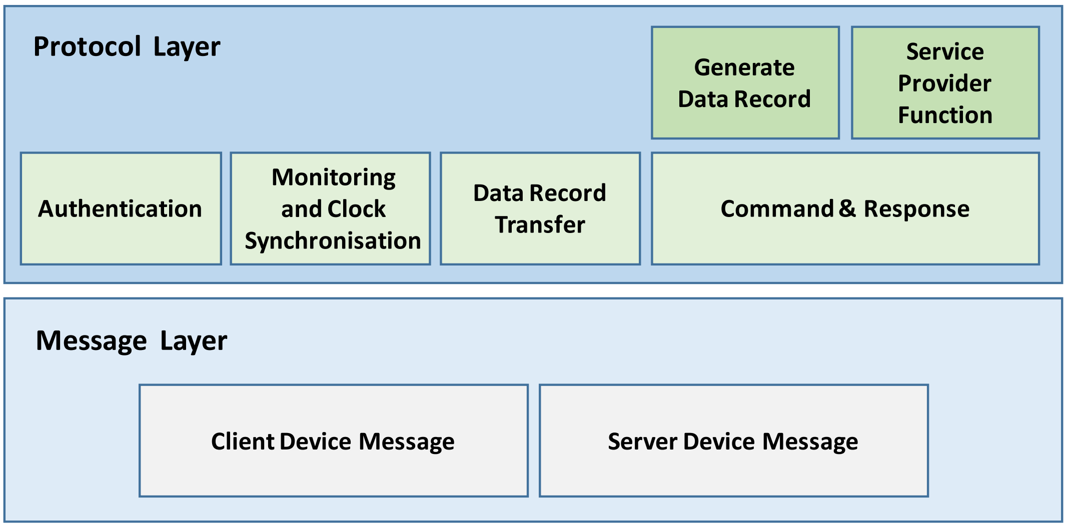


GNSS
Connected Vehicle
Data Service
Business-to-Business Environment
Authority
Application Service Provider
Other entities as appropriate, such as TCA, Operator, other commercial entity, etc.
In-Vehicle Environment
Ignition
Telematics Device
External Power
Connected Device
Connected Device

2.2 Approach

2.2.1 It is within this context that this specification exists, to define a communications interface for the interconnection of a system or device with a telematics device. The logical structure of this communications interface is show in **Figure 2**.

Figure 2: Structure of the Communications Interface



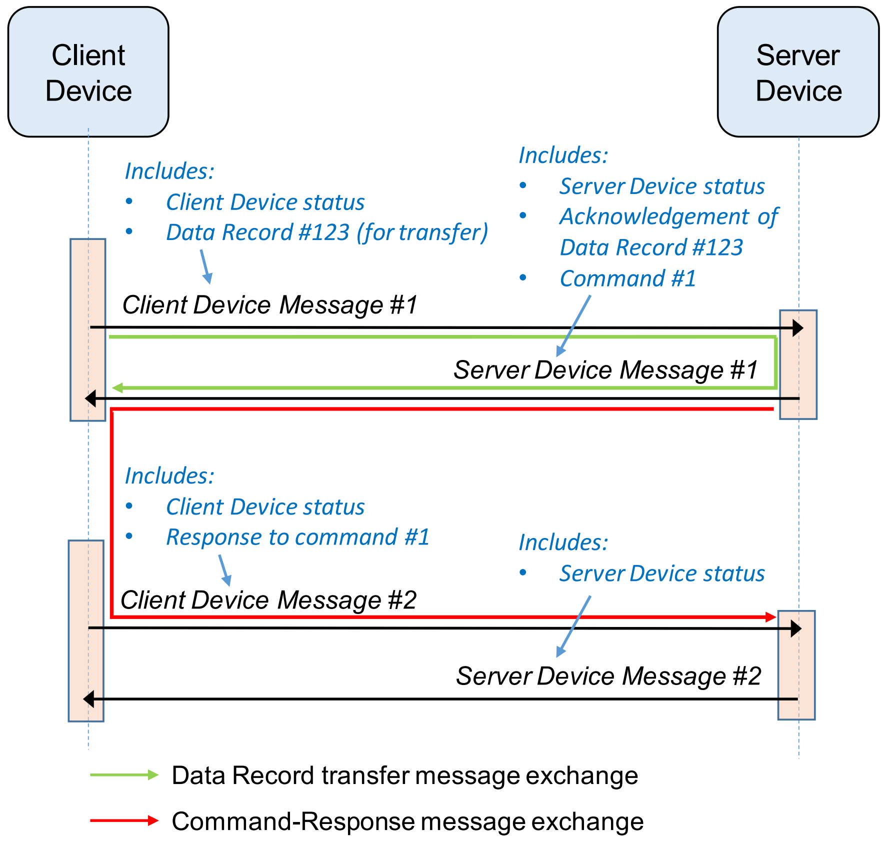
2.2.2 Underpinning the communications interface is the Message Layer. It is from this layer that the system or device, and the telematics device, adopt the roles of the client device and the server device, respectively:

- The client device initiates all Message Layer communication by transmitting a client device message to the server device; and
- The server device (i.e. the telematics device) then replies to the client device message with its own server device message. The server device is not permitted to initiate Message Layer communication. Where the server device has a requirement to communicate (e.g. to issue a command to the client device), it must wait until it next receives a client device message before it is permitted to transmit its server device message in reply.

2.2.3 The Protocol Layer builds upon the foundation of the Message Layer to provide higher-level capabilities, notably including data record transfer, and a Command and Response mechanism. Each message exchanged between the client device and the server device at the Message Layer provides a transport for Protocol Layer interactions. For example, the Message Layer exchange shown in **Figure 3** supports two independent and simultaneous Protocol Layer interactions:

- Data Record Transfer – the client device transfers a data record to the server device by encoding that data record within a client device message. The server device then encodes its acknowledgment of receipt of that data record within its reply server device message; and
- Command and Response – the server device issues a command to the client device by encoding that command within a server device message, but noting that the server device message can only be sent in reply to a received client device message.

Figure 3: Protocol Layer Interactions using Message Layer Exchange



2.2.4 An example of a command issued by the server device might be for the client device to generate a specific data record. The client device encodes its response to that server device command within its next client device message.

2.2.5 Each exchange within the Message Layer is initiated by the client device transmitting a client device message. By contrast, the Command and Response mechanism within the Protocol Layer is initiated by the server device issuing a command. The opposing direction of these Message Layer and Protocol Layer flows is resolved by each server device command being encoded (or 'piggy-backed') within a server device message (reply) message. Thus, while the server device can issue Protocol Layer commands, it cannot initiate Message Layer communication.

2.2.6 The client device and the server device are able to monitor and influence each other's behaviour through the encoding of status information within their messages. For example, the client device status information includes the readiness of the client device to accept specific commands from the server device. Similarly, the server device status information includes the readiness of the server device to accept the transfer of data records from the client device.

2.3 Authentication

2.3.1 Any client device and any server device that is connected to the other in accordance with this specification may freely exchange messages, and basic status information within those messages. However, most Protocol Layer functionality defined by this specification is restricted, and only accessible where the client device and server device are considered to be authenticated to each other:

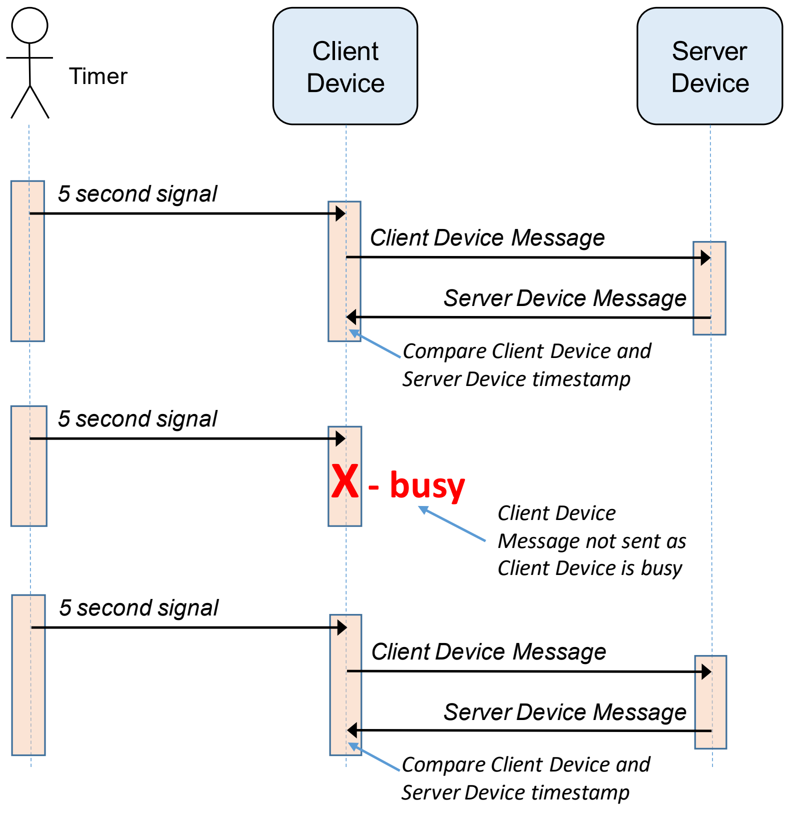
- the client device considers the server device to be authenticated where messages received from the server device contain an identifier that has been entered into the client device by an appropriately authorised operator; and
- the server device considers the client device to be authenticated where messages received from the client device contain an identifier that has been entered into the server device by an appropriately authorised operator.

2.3.2 Each message transmitted by the client device and the server device includes a flag within the status information to indicate whether the sender of the message currently considers the recipient of the message to be authenticated.

2.4 Monitoring and Clock Synchronisation

2.4.1 The client device is required to send a client device message to the server device routinely every 5 seconds. At busy times the client device may skip sending the client device message, but must send at least one client device message every 60 seconds. This interaction is shown in **Figure 4**.

Figure 4: Status Monitoring and Clock Synchronisation Interaction



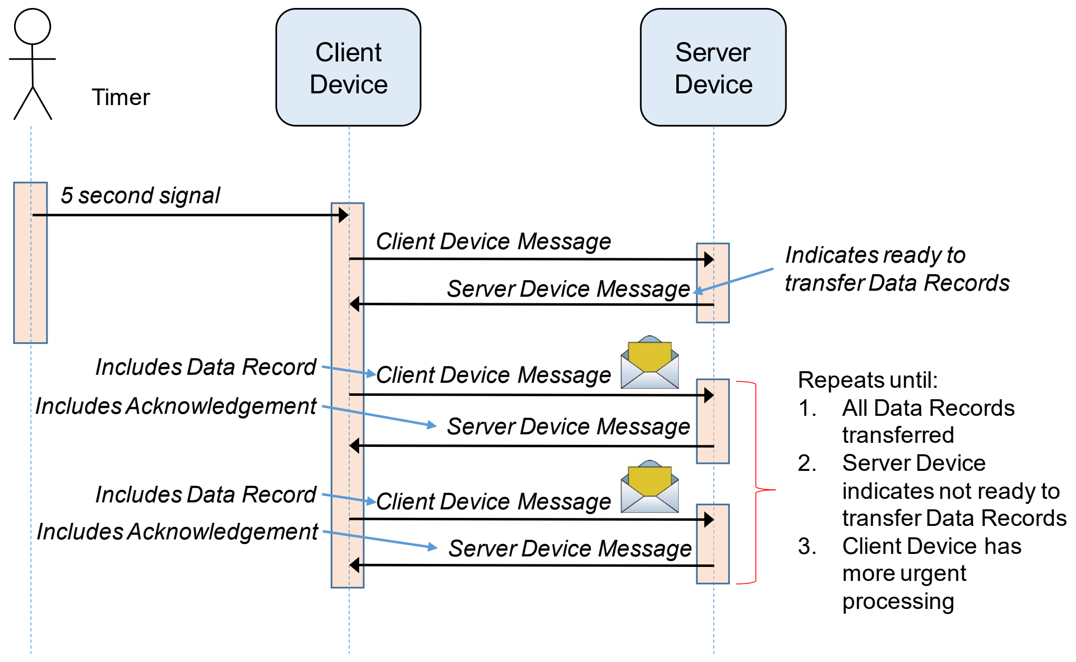
2.4.2 The regular client device message acts as a 'heartbeat', and provides the server device with an assurance that the client device is operational, and also client device status information. In turn, the server device message transmitted in reply to the client device message provides the client device with an assurance that the server device is operational, and also server device status information.

2.4.3 Upon receipt of each server device message, the client device is required to inspect the timestamp within the message, and to ensure that its internal clock is within a specified tolerance of this value. This ensures that the date and time within data records generated by the client device is consistent with that of other systems and devices installed in the vehicle.

2.5 Data Record Transfer

2.5.1 Data records generated by the client device are transferred to the server device, nominally for the purposes of forwarding to the back office via the server device's mobile data network connection. As each data record is acknowledged by the server device (as having been successfully transferred), the client device is able to remove that data record from its internal storage. The data record transfer interaction is shown in **Figure 5**.

Figure 5: Data Record Transfer Interaction



2.5.2 In summary, this Protocol Layer interaction comprises the following sequence of events:

- At some time prior to Data Record Transfer occurring, the client device generates one or more data records, and stores them to its internal storage;
- The client device receives a server device message that indicates the server device is ready for the transfer of data records. By necessity, this server device message will have been sent in reply to a client device message as the server device cannot initiate Message Layer communication;

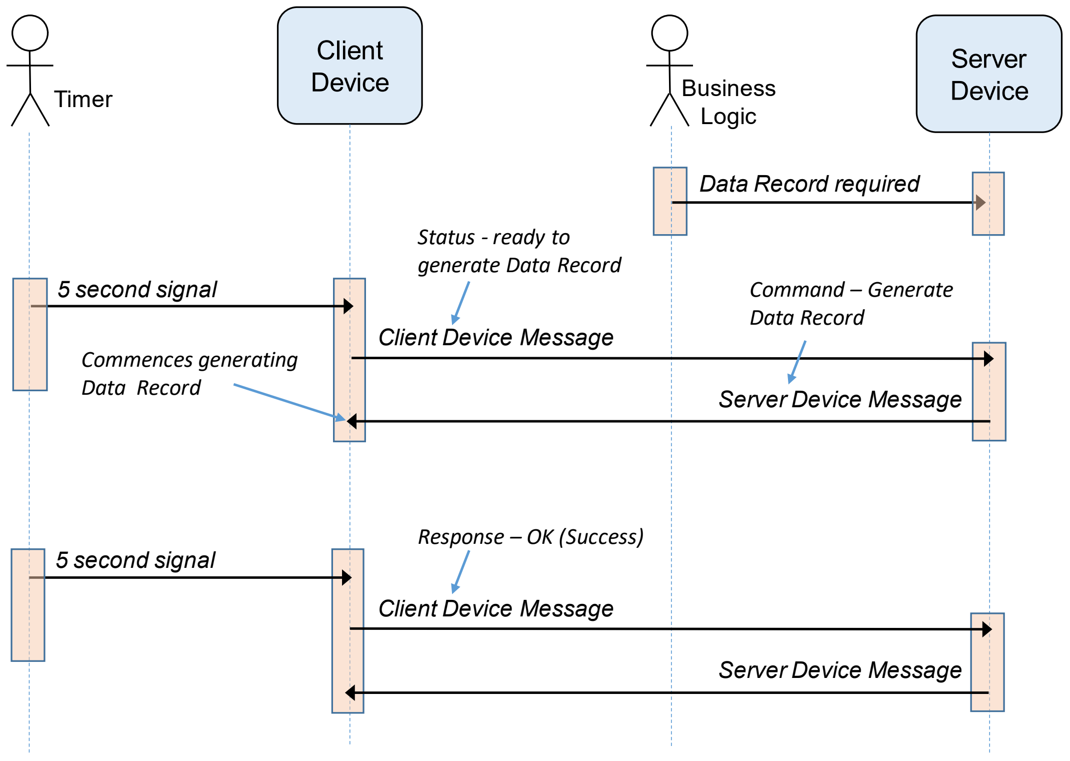
- The client device transmits a client device message to the server device, and within this message includes a data record. This client device message can be transmitted at any time suitable to the client device, and is not required to coincide with the every 5-second schedule for 'heartbeat' messages;
- The server device stores the data record to its own internal storage, and replies to the client device with a server device message that acknowledges receipt of the data record, and that indicates whether the server device is ready for the transfer of further data records; and
- The interaction repeats until such time that all stored data records are transferred, until the server device indicates that it is not ready for the transfer of further data records, or until the client device has more urgent processing to perform.

2.5.3 This interaction allows the client device control over the timing of transfer of data records to the server device. However, the server device is also allowed an ability to regulate the flow of data record transfer in accordance with the device's processing capacity and priorities.

2.6 Data Record Generation

2.6.1 While the client device is able to generate data records in accordance with its own internal business rules, the server device is also able to request that the client device generate data records using the Protocol Layer Command and Response mechanism introduced in 0.0.2098159504. The data record generation interaction is shown in **Figure 6**.

Figure 6: Data Record Generation Interaction



2.6.2 In summary, this Protocol Layer interaction comprises the following sequence of events:

- Business logic associated with the server device identifies a requirement for a specific data record to be generated by the client device. Potentially this includes allowing the driver to request data record generation;
- The server device receives a client device message (e.g. the every 5-second 'heartbeat' client device message), and inspects the status information encoded within that message to determine that the client device is ready to accept a Generate Data Record command;
- The server device replies to the client device message with a server device message, and within that server device message the server device encodes the Generate Data Record command;
- The client device receives that server device message, and validates the Generate Data Record command. Where the command is valid, the client device initiates generation of the required data record; and
- In the next client device message transmitted, the client device provides a response to the Generate Data Record command. The client device may transmit a client device message immediately specifically to respond to the Generate Data Record command, or may elect to delay its response until the next scheduled ('heartbeat') client device message is transmitted.

2.6.3 This interaction does result in some latency between the server device identifying the requirement for data record generation, and being able to transmit the Generate Data Record command. Typically the delay will be not more than 5 seconds, but in the worst case could be 60 seconds.

2.7 Service Provider Function

2.7.1 The Service Provider Function is a mechanism that allows the communications protocol defined by this specification to be used for purposes beyond the scope of this specification. For example, the Service Provider Function could be used to exchange provider-specific sensor readings or status information. Use of the Service Provider Function requires collaboration between providers of client device and server device equipment. The use of the Service Provider Function is subject to any type-approved functionality and behaviour of each device not being hindered or degraded.

2.8 Extension Profiles

2.8.1 This specification is generic in that it is not specific to any given application or capability. However, it is structured to allow inclusion of any number of extension profiles, where each extension profile defines how the specification applies to a given application or capability. This version of this specification supports the OBM system extension profile.

2.8.2 Each client device is able to support at most one extension profile, and this will relate to the specific capabilities of that device. The server device is able to simultaneously support multiple extension profiles, allowing it to simultaneously participate in multiple independent interconnections in accordance with this specification.

3 REFERENCES

3.1.1 Documents referenced in this specification are listed below:

- Interface Between Data Terminal Equipment and Data Circuit-Terminating Equipment Employing Serial Binary Data Interchange, ANSI/TIA/EIA-232-F-1997, TIA 1997.
- 7-bit Coded Character Set for Information Interchange, ISO/IEC 646:1991, ISO 1991.
- Telecommunications and Information Exchange between Systems - High-Level Data Link Control (HDLC) Procedures, ISO/IEC 13239:2002, ISO 2002.
- Transport Certification Australia (TCA) 2018, *On-Board Mass System Functional and Technical Specification*, Transport Certification Australia. Melbourne.

4 REQUIREMENTS FOR INTERCONNECTIVITY OF TELEMATICS DEVICE WITH OTHER SYSTEMS

4.1 Overview

4.1.1 This section contains the requirements for the interconnection of a telematics device and a system or device, as described in the following sections:

a. Part A which defines a core capability that is to be implemented by the telematics device and all systems and devices, and is not specific to any given application or capability; and

b. Part B which defines an OBM system extension profile that describes functionality specific to OBM systems.

4.1.2 Additional parts may be added to this specification as required to accommodate additional extension profiles pertaining to other applications or capabilities.

4.1.3 This section characterises the system or device as the client device, and the telematics device as the server device:

a. The client device is so-called because it initiates all Message Layer communication; and

b. The server device, being the telematics device, is so-called because it receives and replies to Message Layer communication from the client device. The server device cannot initiate Message Layer communication.

4.1.4 The terminology client device and server device does not imply any responsibilities or capabilities beyond those defined in this specification.

PART A CORE CAPABILITY

PHYSICAL CHARACTERISTICS

A.1 Applicability

A.1.1 The client device and the server device shall each meet the requirements of Part A of this specification.

A.2 Extension Profiles

A.2.1 In addition to the requirements of A.1.1, the client device and server device may meet the requirements of the OBM system extension profile (see Part B).

A.2.2 The client device shall not implement more than one of the extension profiles specified in A.2.1.

A.2.3 The server device may implement any combination of the extension profiles specified in A.2.1.

A.3 Communications Interface General Requirements

A.3.1 The client device and server device shall be connected by a communications interface that supports bidirectional point-to-point transmission of an ordered sequence of bytes.

Note: The communications interface between the client device and the server device may be wired or wireless.

A.3.2 In meeting the requirements of A.3.1, any wired connection between the client device and server device shall be robust and shall remain connected under normal operating conditions of the equipment on which it is installed, including under the environmental conditions applicable for type-approval of both the client device and server device.

A.3.3 The communications interface between the client device and server device shall support a minimum data rate of 19,200 bits-per-second.

A.3.4 The communications interface between the client device and server device shall meet the requirements of the type-approval of each device.

A.3.5 In meeting the requirements of A.3.1, any connection between the client device and server device that is other than a point-to-point wired connection shall have an associated mechanism to ensure the authenticity of data transmitted and received by each device.

Note: For example, a wired or wireless network connection might be augmented with Transport Layer Security (TLS), or an equivalent technology, to provide assurance of data authenticity. For direct, wired connections (e.g. RS-232), assurance of data authenticity is vested in the physical connection between the client device and server device.

A.4 Communications Interface as RS-232

A.4.1 In meeting the requirements of A.3, the client device and server device may be connected via an ANSI/TIA/EIA-232-F (RS-232) interface.

Note: RS-232 is explicitly included within this specification because of its prevalence and simplicity. Notwithstanding this, RS-232 is neither required or preferred as a communications interface.

A.4.2 In the case of A.4.1, the RS-232 cable shall:

a. comprise Transmitted Data (TxD), Received Data (RxD) and Common Ground (GND) signal connections only;

b. be limited in total capacitance and overall length such that the RS-232 communications interface is able to operate reliably at the configured data rate;

c. use insulated, stranded copper (optionally tinned) 20 AWG wire for each signal connection; and

d. have its shield connected to chassis GND at the client device end.

Note: Where applicable, Transmitted Data (TxD) and Received Data (RxD) may be crossed between the client device and server device (i.e. to provide a null modem).

A.4.3 In the case of A.4.1, the RS-232 interface shall be configured to operate with 8 data bits, 1 stop bit, and no parity bits.

A.4.4 In the case of A.4.1, the RS-232 interface shall not use any flow control mechanism.

Note: This requirement precludes the use of software flow control (e.g. using start-of-text [STX] and end-of-text [ETX] characters), and also the use of 'transparency mode'.

MESSAGE LAYER

A.5 Message Flow

A.5.1 The client device and server device shall communicate through the exchange of messages, the:

a. client device shall initiate communication by transmitting a client device message to the server device; and

b. server device shall process the client device message, and thereafter in reply, transmit a server device message to the client device.

A.5.2 The client device shall transmit a client device message to the server device:

a. nominally once every 5 seconds;

b. at least once every 60 seconds; and

c. more frequently where required by this specification.

Note: The every 5-second message is termed a 'heartbeat' message.

While the client device should transmit a message each 5 seconds, this specification acknowledges that reasonably this schedule could be disrupted by factors such as resource contention, system load, or higher priority processing within the client device.

The only two valid requirements to transmit a client device message more frequently than every 5 seconds are to respond to a command from the server device (see A.12.4), or to transfer a data record (see A.13.3).

A.5.3 Where the client device has transmitted a client device message to the server device, the client device shall not transmit a subsequent client device message until at least:

a. a valid server device message has been received in reply from the server device; or

b. a message flow timeout event has been identified in accordance with A.5.5.

Note: This requirement ensures that the client device has no more than one outstanding request message at any time.

A server device message is in reply to a client device message where the two messages have the same message sequence number (see A.7.5).

A.5.4 The server device shall not transmit a server device message except in reply to a client device message received from the client device.

A.5.5 The client device shall identify a message flow timeout event where a valid server device message is not received from the server device within 1 second of successfully transmitting a client device message.

Note: Following a message flow timeout event, the client device should not transmit another client device message prior to the next every 5-second 'heartbeat' message.

The client device should ignore a server device message received after a message flow timeout event has been identified (i.e. a late message).

A.6 Message Encoding

A.6.1 Each message transmitted by the client device or the server device shall comprise multiple fields, with each field encoded using a data type shown in **Table 1**.

Table 1: Message Data Encoding Data Types

Data Type	Abbreviation	Values
Text	TXT	See A.6.2
Integer	INT (signed) UINT (unsigned)	See A.6.3
Binary	BIN	An arbitrarily long string of bytes

A.6.2 Message data fields of type Text shall:

a. be encoded using ISO/IEC 646:1991 (ASCII) printable characters in the range 32 to 126 inclusive; and

b. where the actual field length is shorter than the allocated field length, be padded to the right with spaces.

A.6.3 Message data fields of type Integer shall be stored and transmitted from most significant byte (first) to least significant byte (last).

Note: This is commonly referred to as 'big-endian' format.

A.7 Message Structure

A.7.1 Each message transmitted by the client device or the server device shall have the structure shown in **Table 2**.

Table 2: Message Structure and Header Fields

Field	Data Type	Bytes	Value
Start Sentinel	UINT	2	Value: 0xCAFE. See A.7.2.
Protocol Version Code	UINT	1	See A.7.3
Integrity Code Algorithm Code	UINT	1	See A.7.4
Message Length	UINT	2	Length of the message in bytes (including the Integrity Code)
Message Sequence Number	UINT	2	See A.7.5
Message Date Time	TXT	14	UTC date and time from the message sender's internal clock, in the format YYYYMMDDHHMMSS
Message Sender	TXT	10	Identity of the message sender. See A.10.1.
Status Flags	UINT	2	See A.7.6 and A.7.7
Extension Profile Code	UINT	1	See A.7.8 and A.7.9
Extension Profile Status Flags	UINT	1	See A.7.10
Fields specific to client device record (see A.7.11) or server device record (see 0)			
Integrity Code	BIN	0 .. N	See A.8

A.7.2 Each message shall commence with the Start Sentinel value 0xCAFE, and shall end with a valid Integrity Code (see 0).

Note: The pattern 0xCAFE may legitimately appear within a message (and not be considered the start of a new message) where that message has a valid integrity code.

A.7.3 Each message shall have the Protocol Version Code field set to the value shown in **Table 3** that represents the current version of this specification.

Table 3: Protocol Version Code Values

Protocol Version Code	Specification Version and Date
1	1.0
2	1.1 (current version)

A.7.4 Each message shall have the Integrity Code Algorithm Code field set to a value shown in **Table 4** to indicate the length of the Integrity Code field, and the algorithm by which it is calculated.

Table 4: Integrity Code Algorithm Code Values

Code	Integrity Code Algorithm	Integrity Code Length
1	CRC-32. (See A.8.3)	4 bytes

A.7.5 Each message shall have the Message Sequence Number field set to a number in the range 1 to 65535 inclusive:

a. each client device message shall be assigned sequential and incrementing numbers by the client device, with the value 1 being used after the value 65535; and

b. each server device message shall have the message sequence number of the client device message that it is in reply to.

Note: Client device messages are never retransmitted, and thus every client device message will have a unique message sequence number (notwithstanding the wraparound of this sequence).

The server device's echoing of the Message Sequence Number field allows the client device message and the server device message to be correlated.

A.7.6 Each client device message shall have the Status Flags field set in accordance with the values shown in **Table 5**.

Table 5: Client Device Status Flags

Bit Mask	Status Flag	Values
0x01	Server device authenticated	See A.10. Values: • 0 = not authenticated • 1 = authenticated
0x02	Ready to receive Generate Data Record command	See A.14. Values: • 0 = not ready (or not authenticated) • 1 = ready
0x04	Ready to receive Service Provider Function command	See A.15. Values: • 0 = not ready (or not authenticated) • 1 = ready

A.7.7 Each server device message shall have the Status Flags field set in accordance with the values shown in Table 6.

Table 6: Server Device Status Flags

Bit Mask	Status Flag	Values
0x01	Client device authenticated	See A.10. Values: • 0 = not authenticated • 1 = authenticated
0x02	Ready to transfer data record	See A.13. Values: • 0 = not ready (or not authenticated) • 1 = ready

A.7.8 Each client device message shall have the Extension Profile Code field set:

a. in the case that the client device supports a single extension profile, to the value shown in **Table 7** that corresponds to that extension profile; or

b. otherwise (i.e. where the client device does not support an extension profile), to the value zero (0).

Table 7: Profile Code Values

Profile Code	Profile Description	Reference
0	None (no applicable extension profile)	Not applicable
1	OBM system extension profile	See Part B

A.7.9 Each server device message shall have the Extension Profile Code field set:

a. in the case that the client device message that the server device message is in reply to has a non-zero value for the Extension Profile Code field, and where the server device also supports the extension profile that corresponds to that value; or

b. otherwise, to the value zero (0).

Note: Some functionality defined within this specification may be unavailable in the case that the client device does not implement any extension profile. For example, the Generate Data Record command relies upon a supported extension profile to define data record types.

A.7.10 Each message shall have the Extension Profile Status Flags field set:

a. in the case that the Extension Profile Code field in that message is set to a non-zero value, in accordance with the requirements of the extension profile that corresponds to that value; or

b. otherwise, to the value zero (0).

A.7.11 Each client device message shall have the structure shown in **Table 8**.

Table 8: Client Device Message Structure and Header Fields

Field	Data Type	Bytes	Value
Common fields (see to A.7.1)			
Command Sequence Number	UINT	2	See A.12
Response Code	UINT	1	See A.12
Data Record Count	UINT	2	See A.13.1
Data Record Number	UINT	2	See A.13.2
Data Record	BIN	0…N	See A.13.2
Integrity Code (see A.7.1)			

A.7.12 Each server device message shall have the structure shown in **Table 9**.

Table 9: Server Device Message Structure and Header Fields

Field	Data Type	Bytes	Value
Common fields (see A.7.1)			
Acknowledged Data Record Number	UINT	2	See A.13.4
Command Code	UINT	1	See A.12
Command Data	BIN	0…N	See A.12
Integrity Code (see A.7.1)			

A.8 Message Authenticity and Integrity

A.8.1 Each message shall include an Integrity Code.

Note: The Integrity Code provides the receiver with a level of assurance regarding the integrity of the message, and (for some algorithms) the authenticity of the message.

A.8.2 The Integrity Code shall be formed as a function of the message data (excluding the Integrity Code itself).

A.8.3 Except where otherwise required by this specification, the Integrity Code shall be a 4-byte integer value calculated using the CRC-32 algorithm (as per ISO/IEC 13239:2002) with polynomial 0x04C11DB7.

Note: This specification is structured to allow for the addition of alternate Integrity Code algorithms.

A.8.4 On receipt of a message, the receiving device shall discard the message where the expected (calculated) value and the received value of the Integrity Code are not identical.

Note: Where a message is discarded due to an invalid Integrity Code, the message data (bytes) should be considered as potentially containing some fragment of a subsequent and valid message.

PROTOCOL LAYER

A.9 Protocol Support

A.9.1 The client device and the server device shall each support the following functionality:

a. Authentication (see A.10);

b. Clock Synchronisation (see A.11);

c. Command and Response Mechanism (see A.12);

d. Data Record Transfer (see A.13);

e. Generate Data Record (see A.14); and

f. Service Provider Function (see A.15).

A.9.2 In addition to the requirements of A.9.1, the client device and the server device shall each support the functionality associated with each supported extension profile (see A.2.2 and A.12.3).

A.10 Authentication

A.10.1 The client device and the server device shall each consider the interconnected device authenticated, where:

a. an operator has pre-entered the identifier of the interconnected device in accordance with A.10.5 and A.10.6; and

b. the most recent and prior message received from the interconnected device in the previous 2 minutes has a Message Sender field value that exactly matches that pre-entered identifier.

A.10.2 Each client device message shall have the Status Flags field set in accordance with A.7.6 to indicate if the server device is currently considered authenticated to the client device.

A.10.3 Each server device message shall have the Status Flags field set in accordance with A.7.7 to indicate if the client device is currently considered authenticated to the server device.

A.10.4 The client device and the server device shall continue to exchange messages irrespective of their respective authentication status, but noting the following exclusions of functionality where either device is not considered authenticated:

a. Data Record Transfer (see A.13);

b. Data Record Generation (see A.14); and

c. Service Provider Function (see A.15).

A.10.5 The client device and server device shall each restrict the pre-entry of the interconnected device identifier in accordance with A.10.1a to authorised operators.

A.10.6 The client device and server device shall each store a record of the pre-entry of the interconnected device identifier in accordance with A.10.1a, including the identity of the authenticated and authorised operator, and the date and time of the event.

A.11 Clock Synchronisation

A.11.1 Upon receiving a server device message with a Message Date Time field value that differs from the current client device internal clock by 4 seconds or more, the client device shall adjust its internal clock to that value.

Note: This ensures that the client device internal clock is within approximately +/- 5 seconds of the server device internal clock.

It is permissible for the client device to simultaneously synchronise its internal clock to a more accurate reference source than the server device (e.g. its GNSS signal) providing that it remains within the tolerances of this specification.

A.12 Command and Response Mechanism

A.12.1 The client device and the server device shall implement a command and response mechanism whereby the:

a. server device issues a command to the client device by setting the Command Code and Command Data fields within a server device message in accordance with A.12.2 and A.12.3; and

b. client device receives that command, (where relevant) initiates any associated processing, and thereafter responds by setting the Command Sequence Number and Response Code fields in subsequent client device messages in accordance with A.12.4.

A.12.2 Each server device message shall have the Command Code field set:

a. in the case that the server device has an identified requirement to issue a command to the client device, and meets the pre-conditions associated with that command as defined by this specification, to a non-zero value corresponding to that command; or

b. otherwise, to the value zero (0).

Note: By specifying a non-zero Command Code, the server device is able to 'piggy-back' a command in its reply to a client device message. The value of zero is used where the server device does not issue a command to the client device.

A.12.3 Each server device message shall have the Command Data field populated with variable length data in accordance with the requirements associated with any non-zero value of the Command Code field, or omitted in the case that the Command Code field has a zero (0) value.

Note: The optional and variable-length Command Data field is used to convey data associated with any command issued to the client device by the server device.

A.12.4 Each client device message shall have the Command Sequence Number and Response Code fields set in response to the most recent and prior server device message received by the client device that contains a non-zero Command Code value:

a. in the case that there is such a server device message, the:

   i. Command Sequence Number field shall be set to the Message Sequence Number field of that server device message; and

   ii. Response Code field shall be set to a value that indicates that results of processing that Command Code in accordance with A.12.6; or

b. otherwise, to the value zero (0).

Note: The client device repeats the Command Sequence Number and Response Code values in each and every subsequent client device message until such time that another Command Code is received; this provides robustness in the case of client device messages being lost.

In the case that the client device has received a non-zero Command Code in a server device message, the client device may reply immediately with a client device message, or may wait until the next client device message is transmitted for another reason (e.g. every 5-second heartbeat, data record transfer).

A.12.5 The Command Code field within each server device message and the Response Code field within each client device message shall have a value as shown in **Table 10**, or another value as shown in an extension profile.

Table 10: Command Code and Response Code Values

Message Type	**Command Code**	**Response Code**	**Reference**
Blank / Void	0x00	0x00	A.12.4
Generate Data Record	0x02	-	A.14
Service Provider Function	0x04	-	A.15
OK (Success)	-	0x01	A.12.6
Not Authenticated (Error)	-	0x41	A.12.6
Command Not Known (Error)	-	0x43	A.12.6
Command Not Supported (Error)	-	0x45	A.12.6
Command Temporarily Unavailable (Warning)	-	0x47	A.12.6
Command Not Valid (Error)	-	0x49	A.12.6

Note: By convention, the most significant bit indicates a Core Capability Message Type (0x00-0x7f) or an extension profile-specific message type (0x80-0xff), and the least significant bit indicates a Command (even number) or a Response (odd number).

A.12.6 In meeting the requirements of A.12.1b, the client device shall set the Response Code field to the value listed in A.12.5 according to the following ordered criteria:

a. Command Not Known (Error) – in the case that the value of the Command Code field is not recognised;

b. Not Authenticated (Error) – in the case that the value of the Command Code field is recognised, but the associated functionality requires the server device to be considered authenticated to the client device, and this has not yet occurred;

c. Command Not Supported (Error) – in the case that the value of the Command Code field is recognised, but the associated functionality is not supported;

d. Command Not Valid (Error) – in the case that the value of the Command Code field is recognised and supported, but where any associated Command Data is invalid;

e. Command Temporarily Unavailable (Warning) – in the case that the value of the Command Code field is recognised and supported, but that the associated functionality is temporarily unavailable;

f. another value as shown in the extension profile associated with a non-zero value of the Extension Profile Code field; or

g. otherwise, to the value OK (Success).

Note: Command Temporarily Unavailable (Warning) might be returned where the function is not available due to a hardware malfunction, or because the client device is 'busy' and does not have the capacity to handle the request.

The client device can subsequently use the Status Flags within the client device message to regulate the timing of any subsequent use of that Command Code by the server device (i.e. the client device can prevent the server device from immediately retrying if it has a requirement to do so).

A.13 Data Record Transfer

A.13.1 Each client device message shall have the Data Record Count field set to the number of data records awaiting transfer to the server device.

Note: This count is inclusive of any data record currently being transferred (as that data record cannot yet have been acknowledged by the server device).

A.13.2 The client device shall transfer a data record to the server device by transmitting a client device message in which the:

a. Data Record Number field is set to the record number of that data record; and

b. Data Record field contains that data record in its entirety.

A.13.3 The client device shall transfer a data record to the server device when all of the following conditions are met:

a. the client device has at least one data record awaiting transfer;

b. the client device is ready and able to transfer that data record;

c. the server device is considered to be authenticated to the client device; and

d. the most recent and prior server device message received during the prior 5 seconds has a value for the Status Flag field that indicates the:

i. client device is considered to be authenticated to the server device; and

ii. server device is ready to transfer data records.

Note: This requirement allows the client device to continue transferring data records in quick succession, and without the need to wait until the next 5-second 'heartbeat' client device message is transmitted. This requirement also allows the server device to regulate the transfer of data records.

A.13.4 The client device shall transfer data records to the server device in the order of data record generation.

Note: The order of data record generation is ascending order of record number, but noting that record number values wrap-around periodically.

Where the client device has previously attempted to transfer a data record, but has not had the transfer of that data record acknowledged by the server device, it should reattempt transfer of that data record.

A.13.5 Each server device message shall have the Acknowledged Data Record Number field set to the record number of the data record mostly recently transferred successfully from the client device.

Note: The server device is required to repeat the Acknowledged Data Record Number field value in each subsequent server device message until another data record has been transferred and requires acknowledgement; this allows for robustness in the event that server device messages are lost.

The server device does not have any facility to reject a transferred data record. However, in the event that the server device fails to successfully receive the transferred data record (e.g. its internal storage is full), it should reflect this by not updating the Acknowledged Data Record Number field in its reply server device message. In addition, the server device should set the Status Flag field to indicate that it is not ready to accept transfer of data records until such time that is has sufficient internal storage capacity.

A.13.6 Upon receiving a server device message with a non-zero value of the Acknowledged Data Record Number field, the client device shall:

a. in the case that the corresponding data record is awaiting transfer to the server device, mark the data record with that record number as transferred, and decrement the Data Record Count value (see A.13.1); or

b. otherwise, take no action.

Note: The client device may delete data records from its internal storage after they are acknowledged as transferred by the server device.

A.14 Generate Data Record Command

A.14.1 In accordance with the requirements of A.12, the server device may issue the Generate Data Record command to request that the client device generate a specified type of data record.

Note: Generation of a data record by the client device will typically also involve collection of relevant data.

A.14.2 The server device shall only issue the Generate Data Record command when all of the following conditions are met:

a. The server device has an identified requirement to generate the specified type of data record in accordance with its type-approval;

b. The client device is considered to be authenticated to the server device; and

c. The server device is replying to a client device message in which the:

i. Status Flags field has a value (see A.7.6) that indicates that the client device is ready to receive the Generate Data Record command, and that the server device is considered to be authenticated to the client device; and

ii. Extension Profile Code field specifies an extension profile that is also supported by the server device, and that is associated with the record type that is required to be generated.

A.14.3 Where the server device issues the Generate Data Record command, the Command Data field shall have the format shown in **Table 11**.

Table 11: Generate Data Record Command Data Format

Field	**Data Type**	**Bytes**	**Value**
Record Type	UINT	1	See A.14.4

A.14.4 Where the server device issues the Generate Data Record command, the server device shall include within the Command Data field the record type to be generated.

Note: The Record Type value shall be as defined by the relevant extension profile (e.g. OBM system extension profile – see Part B).

A.14.5 Where the server device has issued the Generate Data Record command in accordance with A.14.1, in setting the Response Code field in accordance with A.12.6 the client device shall use the value:

a. Command Not Valid (Error) – in the case that it does not recognise or cannot collect the specified record type;

b. OK (Success) – in the case that it is currently (already) collecting the specified record type;

c. Command Temporarily Unavailable (Warning) – in the case that it cannot collect the specified record type at present; or

d. otherwise, the value OK (Success).

A.15 Service Provider Function

A.15.1 In accordance with the requirements of A.12, the server device may issue the Service Provider Function command to access non-type approved and service provider-specific functionality within the client device.

Note: The Service Provider Function is subject to the agreement and collaboration of the providers of the client device and the server device.

A.15.2 The implementation of the Service Provider Function command shall be such that the performance of the client device and the server device is not hindered or degraded below the requirements of their respective type-approval.

A.15.3 The server device shall only issue the Service Provider Function command when all of the following conditions are met:

a. the client device and the server device have a shared understanding and support for the functionality uniquely identified by the client device's identifier and the Service Provider Message Type (see A.15.5a);

b. the client device is considered to be authenticated to the server device; and

c. the server device is replying to a client device message in which the Status Flags field has a value that indicates that the client device is ready to accept the Service Provider Function command (see A.7.6), and that the server device is considered to be authenticated to the client device.

Note: The identifier of each type-approved device has a three-character prefix assigned by TCA, and that prefix identifies the service provider for that device.

A.15.4 Where the server device issues the Service Provider Function command, the Command Data field shall have the format shown in **Table 12**.

Table 12: Service Provider Command Data Format

Field	**Data Type**	**Bytes**	**Value**
Service Provider Message Type	UINT	1	See A.15.5
Service Provider Payload	BIN	0 .. N	See A.15.5

A.15.5 Where the server device issues the Service Provider Function command, the server device shall include within the Command Data field:

a. A Service Provider Message Type as understood by both the client device and the server device; and

b. A variable length payload as pertains to the Service Provider Message Type, and as recognised by both the client device and the server device.

PART B ON-BOARD MASS SYSTEM EXTENSION PROFILE

GENERAL

B.1 Extension Applicability

B.1.1 The OBM system extension profile shall apply to the electronic control unit (ECU) of a type-approved OBM system meeting the requirements of the client device.

B.2 Data Record Transfer

B.2.1 Where OBM system data records are transferred in accordance with A.13, the data records shall be formatted in accordance with the *On-Board Mass System Functional and Technical Specification*, and shall use binary encoding.

B.3 Generation of Data Records

B.3.1 In populating the Generate Data Record command data in accordance with A.14.4, the record type shall be defined in accordance with the *On-Board Mass System Functional and Technical Specification*.

B.3.2 In the case that the server device specifies the generation of the OBM system quality record in accordance with A.14.4 and B.3.1, the client device shall interpret this as the generation of the OBM system quality record for each connected mass sensor unit (MSU).

Appendix A Acronyms and Definitions

Acronyms

Acronym	Definition
CRC	cyclic redundancy check
ECU	electronic control unit
ETX	end-of-text
GND	Common Ground
GNSS	Global Navigation Satellite System
MSU	mass sensor unit
OBM	on-board mass
RxD	Received Data
STX	start-of-text
TxD	Transmitted Data
UINT	unsigned integer
UTC	Coordinated Universal Time

Definitions

Term	Definition
core capability	That part of this specification that is applicable to the telematics device and all systems and devices.
data record	A discrete and defined set of data elements, including a (unique) record number, and record date time (of data record generation).
data record generation	The process of creating a data record, including assignment of a record number and record date time, but excluding collection of any constituent data.
extension profile	That part of this specification that is applicable to only those systems or devices associated with a specific application or capability.
integrity code	A value derived from the data within a message, and used for the purposes of detecting errors that may be introduced during the transmission of that message.
telematics device	The primary telematics unit which monitors parameters.

CONTACT

Transport Certification Australia
Level 6, 333 Queen Street
Melbourne VIC 3000

Phone:	+ 61 3 8601 4600
Email:	tca@tca.gov.au
Website:	www.tca.gov.au