Herman elektronika
Brněnská 993
664 42 Modřice
WWW.HERMAN.CZ
Ing. Ivo Herman, CSc.
tel.: +420 545 214 226
fax.: +420 545 214 268
trade@herman.cz

Document title	# OBU services for public transport priority
Version	1.05A
Additional information	
Project	Public transport priority control from OBU

Created by:	Ivo Herman jr.	Date:	11.07.2023	Document Nr:
Approved by:		Date:		

List of revisions and appendices:

Revision	Author	Date
1.01	Ivo Herman jr.	28.4.2022
1.02	Ivo Herman jr.	7.6.2022
1.03	Ivo Herman jr.	26. 10. 2022
1.04	Ivo Herman jr.	25. 1. 2023
1.05	Ivo Herman jr.	31. 5. 2023
1.05A	Ivo Herman jr.	11. 7. 2023

Details of the revision	
1.01	Added figure of stop sequence, added stationList
1.02	Added time information to priority status, clarified use of HTTP, removed Websocket.
1.03	Websocket added, more precise description of priority status, added request type to the priority status
1.04	Added element with embarkation in progress to service 3250
1.05	Improved readability of the document, no functional change in the messages. Only added requestType to the example XML for service 3251 (the element was present in the description).
1.05A	Removed service 3251 for the purpose of use in Třebíč. This service is not necessary there.

Appendices	

Table of contents

1 Document purpose

This document provides a basic description of two services which are necessary for public transport priority control using V2X. These two services are provided by the OBU unit, manufactured the company Herman Electronics – unit UCU 5.0V.

This document should be used in a situation where the OBU unit by Herman is installed in a vehicle with a 3rd party board computer. The board computer can then implement these two simple services to enable V2X public transport priority and to gain benefits which V2X communication offers.

The priority control is done in the OBU unit, the board computer only provides the trip-related data to the OBU. That is, the board computer provides to the OBU necessary data (current line, connection, destination, delay, door status and list of stops), based on which the OBU controls sending of the requests for priority at signalized intersections. Without such data from the board computer, the OBU is not able to send the right request to the traffic light controller.

On the contrary, the OBU informs the board computer about the prioritization request and status. This is used for instance to inform the driver that the public transport vehicle should leave the stop to pass the intersection on green and without stopping.

NOTE: SERVICE 3251 WAS REMOVED FROM THIS DOCUMENT, AS IT WILL NOT BE USED IN TŘEBÍČ.

2 Communication in general

There are two directions of communication:

1) The information from the board computer to the UCU unit, mainly the information about the current trip of the public transport vehicle – service 3250, section 3.1.
2) The information about the priority request from UCU to the board computer. In addition, this message also might contain recommended behavior of the vehicle/driver. Service 3251, section **Chyba! Nenalezen zdroj odkazů.**. - NOT PART OF THIS DOCUMENT, AS ONLY THE FIRST SERVICE IS NECESSARY.

2.1 Interfaces

The OBU unit supports the following interfaces for IP communication:

- Ethernet
- LTE
- Wi-Fi (client or AP)

Typically, in a board network, the Ethernet bus will be used.

2.2 Communication protocols

For a working priority, it is necessary to send data from the board computer to the OBU unit and from the unit to the board computer as well as.

The following communication protocols are supported by the OBU:

- UDP – bidirectional communication,
- HTTP – in this case, the board computer acts as a server and OBU as a client. The OBU unit then periodically queries (GET) the board computer for the service 3250 (data about the trip) and sends information about priority status (POST) to the board computer.
- Websocket - in this case, the board computer acts as a server and OBU as a client. The OBU unit then opens typically two websocket connections (one for each service), or both services

can run over the same websocket connection. The board computer must send regularly the data about the current trip (service 3250) and in the second websocket connection the OBU sends status of the priority requests.

The precise specification of the protocols are available upon request. **We recommend using the HTTP protocol.**

3 Services for public transport priority control

Both services are available as XML or JSON, depends on the requirements – only XML is shown, JSON has a corresponding structure.

The services are part of a complete API. For brevity, the rest of the API is not shown here.

3.1 Service 3250 – 3rdPartyBoardComputerData

With this service the OBU unit, installed in a public transport vehicle, receives the necessary information about the current ride (trip/service) of the vehicle. This service is used when the board computer in the vehicle is a third-party device (not made by Herman). The board computer sends this message periodically to the OBU – at least every 10 seconds (or more often) and when an important parameter changed, e.g. a presence at stop, or the OBU can periodically poll for the data (in HTTP). The data contained in this message are used for proper V2X communication, OBU settings and mainly for controlling public transport priority requests. However, the priority requests are generated autonomously by OBU, based on its own configuration data.

The minimal set of data required for a working public transport prioritization:

- vehicle ID,
- vehicle traction (type) – bus, trolleybus, tram,
- line number,
- connection number,
- vehicle destination,
- vehicle course,
- last, current and next stop,
- door status,
- indication of the current presence in given stop,
- current delay.

If any of the data is not available in the given public transport priority system (for instance, course is not used), the value should be -1 or empty string.

In the example message below, the optional data is marked gray.

Request from Station (Board computer) -> UCU

HTTP Endpoint: /boardComputerTripData (recommended; may be changed by the board computer vendor)

Example DTO: XML **ucu3rdPartyBoardComputerData**

```
<ucu3rdPartyBoardComputerData dt="2018-08-26T11:00:12Z">
  <vhc id="503" tract="bus" lineNum="8" lineTxt="8x" course="101" connId="3"/>
  <vhcState mov="0" mode="0" routePhase="1"/>
  <destin code="936" name="Dest name"/>
  <stationLast stationId="123" stationName="Central stop" rpGeo="1"/>
  <stationCurrent stationId="567" stationName="Small stop" rpGeo="0"/>
```

```
    <stationFollowing stationId="568" stationName="Small stop2"/>
    <delay value="63" valid="1"/>
    <door open="0"/>
    <embarkation enabled="0"/>
    <apc enabled="0" count="25"/>
    <stationList>
          <station stationId="121" stationName="First stop"/>
          <station stationId="122" stationName="Second stop"/>
          …
          <station stationId="129" stationName="Last stop"/>
    </stationList>
</ucu3rdPartyBoardComputerData>
```

Element	Attribute	Type	Description
ucu3rdPartyBoardComputerData	dt	DT	Current date and time when the message was generated at the board computer.
vhc			Information about a vehicle and its current ride
vhc	id	int	Vehicle number – must be unique within the fleet of the public transport operator
vhc	tract	string	Traction (vehicle type). Can be selected from • Bus • Tram • Trolleybus
vhc	lineNum	int	The line number, at which the vehicle currently operates. Set to 0 if not used. The line number is invalid (set to zero), when at vhcState/mode is 0 – not at service
vhc	lineTxt	string	Textual line name. If no line is set, use an epmty string. Line name is invalid (set to empty) when at vhcState/mode is 0 – not at service
vhc	course	int	Course number. If not set, use 0. Is invalid (set to zero), when at vhcState/mode is 0 – not at service
vhc	connId	int	ID of the connection. Is invalid (set to zero), when at vhc/mode is 0 – not at service
vhcState	mov	int	0 …… vehicle is stationary at stop (doors open) 1 …… vehicle rides between stops (doors closed)
vhcState	mode	int	0 …… not at service 1 …… service selected 2 ……. ride based on sequence of stops

			3 ……. Ride based on the destination
vhcState	routePhase	int	Phases of a selected ride 0 …… nothing selected 1 …… before the ride starts 2 ……. Ride in progress 3…….. ride end – arrival at terminus + waiting at the terminus
destin	code	int	Destination code
destin	name	string	Destination name
stationLast	stationId	int	Number (ID) of the last passed (visited) stop. It is a unique ID of the stop (e.g. from a national register of stops).
stationLast	stationName	string	Name of the last passed (visited) stop
stationLast	rpGeo	bool	0 …… vehicle is not present in the geographical area of the last stop 1 …… vehicle is present in the geographical area of the last stop
stationCurrent	stationId	int	Number (ID) of the following (not yet visited) stop. It is a unique ID of the stop (e.g. from a national register of stops).
stationCurrent	stationName	string	Name of the following (not yet visited) stop .
stationCurrent	rpGeo	bool	0 …… vehicle is not present in the geographical area of the following stop 1 …… vehicle is present in the geographical area of the last stop
stationFollowing	stationId	int	Number (ID) of the next but one stop (stop after the not yet visited). It is a unique ID of the stop (e.g. from a national register of stops).
stationFollowing	stationName	string	Name of the next but one stop (stop after the not yet visited).
delay	valid	bool	Indicates if the delay is available.
delay	value	Int	Delay of the vehicle in seconds. Positive value means delay, negative value means traveling ahead of schedule.
door	open	bool	Indicates door status – open=1 or closed=0
embarkation	enabled	bool	Indicates if an embarkation of the passengers is enabled or in progress – the doors can be or are open

apc	enabled	bool	Indicates if an automatic passenger counting system is available at the vehicle
apc	count	int	Number of passengers in the vehicle
stationList			An ordered list of all stations on this route, starting from the origin station and ending with the terminus station. Of course, last, current and next stations above must be listed in this list.
station	stationId	int	Number of the stop in the list.
station	stationName	string	Name of the stop in the list

Response UCU -> Station (Board computer)

No data, just acknowledgement.

3.1.1 Logic of stop switching

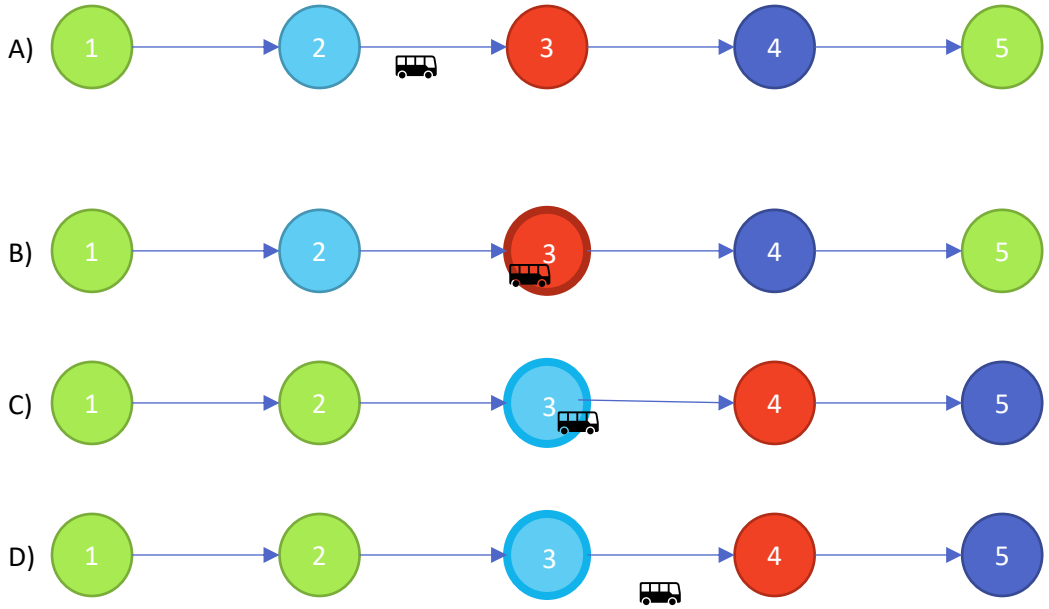

Figure 1 – Switching of the stops. The icon of a bus indicates the vehicle position between stops. Colors: orange: last station, green: current station, blue: following station. The thick circle around the stop indicates that the vehicle is currently present at the stop.

The switching is depicted in Fig. 1.

Case A): The vehicle travels between two stops, Nr. 2 and Nr. 3. The last stop (in which the vehicle is not present) is stop Nr. 2, the current stop is stop Nr. 3 (not yet reached, so rpGeo=false) and the following stop is stop number 4.

Case B): The vehicle has reached stop 3, in area of which it is now present. The last stop is still Nr. 2, the current is stop Nr. 3 (the vehicle is present in the stop area, so rpGeo=true), the following stop is Nr. 4.

Case C): The vehicle has just closed the door (or by any mean evaluated that it has left the stop). The last stop has been changed to stop Nr. 3 (the vehicle can still be in the area of the stop – rpGeo=true), the current stop is stop Nr. 4 and the following stop is stop Nr. 5.

Case D): The vehicle travels between two stops, 3 and 4. The last stop (in which the vehicle is not present) is stop Nr. 3 (rpGeo=false), the current stop is stop Nr. 4 (not yet reached = rpGeo=false) and the following stop is stop Nr. 5.

The change of the stops (e.g., current -> last) must occur in the same message as the change of door or stop presence status.