MMITSS HMI Data Interface

Author: Larry Head

Date: November 18, 2019

# Purpose

This document describes the mmitss-him-controller data acquisition needs and hmi-data format.

# Background

MMITSS version 1.0 used a web page to display vehicle (OBU) status and operations. Figure 1 shows a picture of the original web page. This picture does not show the table of remove vehicle BSM data, EVA, Incident Ahead, and School Zone graphics that were in the upper left had corner.

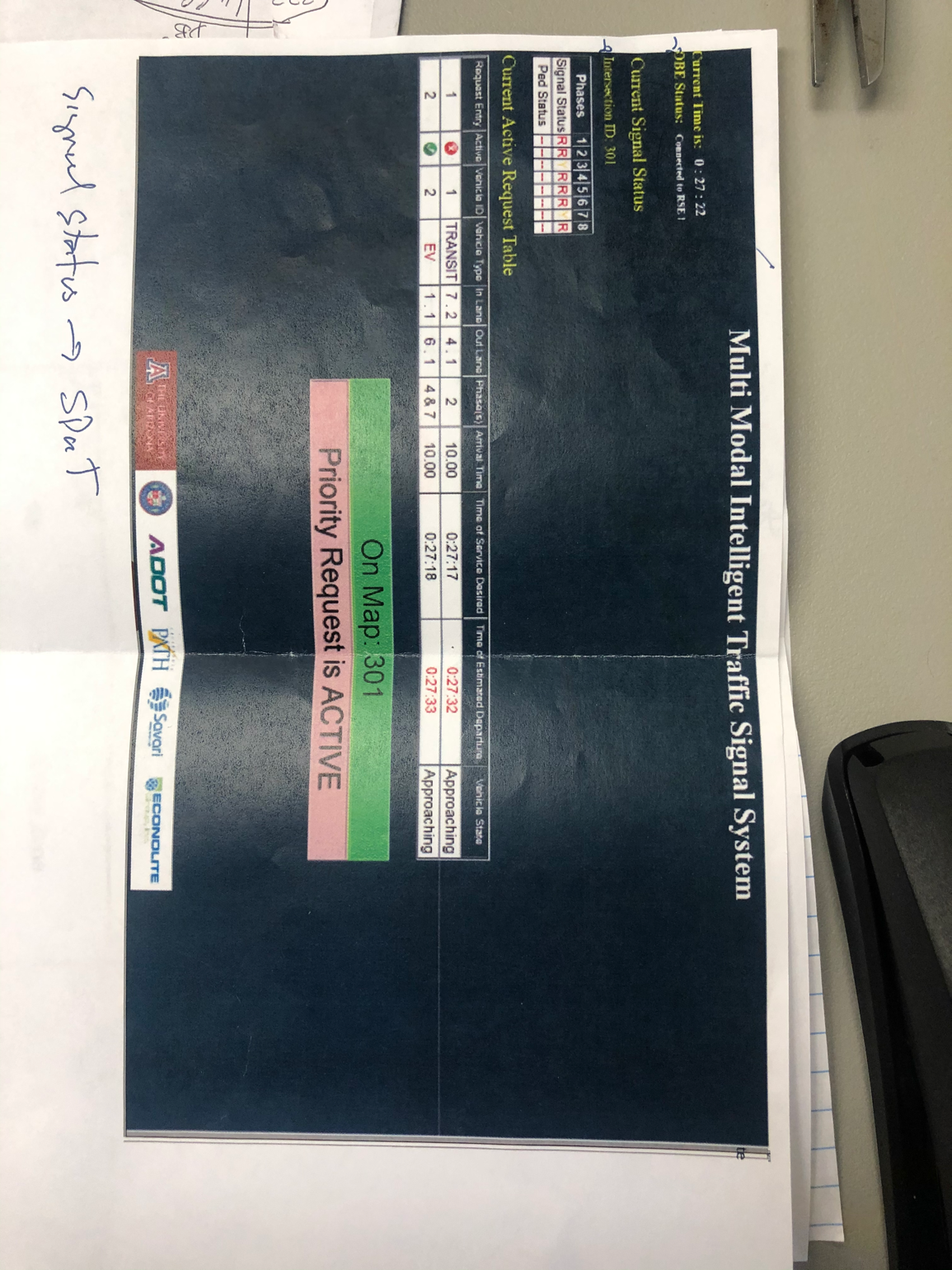


Figure 1. MMITSS 1.0 Vehicle HMI

The strengths of this web based hmi were that it showed:

* the “On Map XXX” message once the vehicle OBU had received a MAP message and determined that the vehicle was located within the boundaries of the MAP
* A “Priority Request is ACTIVE” message once the vehicle OBU had broadcast a SRM message
* A table showing all active requests with te data from the Signal Status Message (aka the Priority Status Message (PSM) in the mmitss 1.0 code.
* The status of all 8-phases of a traffic signal controller with phase interval (R, Y, G) and pedestrian interval (W, PC, DW)
* (not shown) an Emergency Vehicle Alert (fire truck)
* (not shown) a table of BSM data from other vehicles
* (not shown) school zone alert
* (not shown) incident ahead alert

interfaceJsonString = json.dumps({

"mmitss\_hmi\_interface":

{

"hostVehicle" :

{

"heading\_Degree" : hv\_heading\_Degree,

"position" :

{

"elevation\_Meter" : hv\_elevation\_Meter,

"latitude\_DecimalDegree" : hv\_latitude\_DecimalDegree,

"longitude\_DecimalDegree" : hv\_longitude\_DecimalDegree

},

"secMark\_Second" : secMark,

"speed\_MeterPerSecond" : hv\_speed\_Meterpersecond,

"temporaryID" : hv\_tempID,

"vehicleType" : hv\_vehicleType,

"lane": 1, # this is desireable data

"speed\_mph": hv\_speed\_mph,

"priority" : {"OnMAP" : onMAP, "requestSent" : requestSent}

},

"remoteVehicles" :

remoteVehicles,

"infrastructure":

{

"availableMaps": availableMaps,

"currentPhase" : current\_phase\_status, # data for signal head, min, and max

"phaseStates" : phase\_table, #data for 8-phase display table

"activeRequestTable" : activeRequestTable

},

}

})

s.sendto(interfaceJsonString.encode(),hmi)

# MMITSS HMI Controller Design

Figure 2. MMITSS HMI interface json.

It is probably easiest to understand the design by looking at the json that is created to send to the hmi. Figure 2 shows the json. Currently, it is assumed that this json is sent to the hmi every 0.1 seconds (but this can be discussed). The primary key is “mmitss\_hmi\_interface”.

The first section contains information about the host vehicle (hv) including:

Table 1. Host vehicle (hv) data

|  |  |  |
| --- | --- | --- |
| Data Field | Description | Units and Range |
| secMark\_Second | Time stamp of host vehicle data | 0 to 60999 in milliseconds of the minute |
| tempID | Temporary vehicle ID | 0-9999 (4 octets) |
| vehicleType | Vehicle Type | {unknown – passenger vehicles, ev, transit, truck} |
| latitude\_DecimalDegree | latitude | Decimal degree (8 digits of precision to the right of the decimal) |
| Longitude\_DecimalDegree | longitude | Decimal degree (8 digits of precision to the right of the decimal) |
| Elevation\_Meter | elevation | Meters (xxxx.x) |
| heading\_degree | Compass heading | 0 to 359.9875 (4 decimal places are provided) |
| speed\_mph | speed | mph (integer) |
| lane | Current lane (derived from MAP) | Integer |
| signalGroup | Current signal group (derived from MAP | integer |
| onMAP | Boolean indicating if the vehicle is on the MAP | True/False |
| requestSent | Boolean indication if a priority request has been sent (as opposed to sent and acknowledged) | True/False |

The vehicle lane and signalGroup should be dislayed with the host vehicle data.

The OnMAP and requestSent status should be displayed as indicators (text boxes?) as shown in Figure 1. Both of these indicators change color based on their status (e.g. dark when not onMAP and no request sent and green for OnMAP and yellow for requestSent True).

The second section contains the remote vehicle (rv) data including:

Table 2. Remote vehicle data

|  |  |  |
| --- | --- | --- |
| Data Field | Description | Units and Range |
| secMark\_Second | Time stamp of host vehicle data | 0 to 60999 in milliseconds of the minute |
| tempID | Temporary vehicle ID | 0-9999 (4 octets) |
| vehicleType | Vehicle Type | {unknown – passenger vehicles, ev, transit, truck} |
| latitude\_DecimalDegree | latitude | Decimal degree (8 digits of precision to the right of the decimal) |
| Longitude\_DecimalDegree | longitude | Decimal degree (8 digits of precision to the right of the decimal) |
| Elevation\_Meter | elevation | Meters (xxxx.x) |
| heading\_degree | Compass heading | 0 to 359.9875 (4 decimal places are provided) |
| speed\_mph | speed | mph (integer) |

A list of remote vehicles is sent to the hmi as:

"remoteVehicles" :

remoteVehicles,

where remote vehicles is a list of basicVehicle dictionaries.

A sample from the debugger is shown in Figure 3.

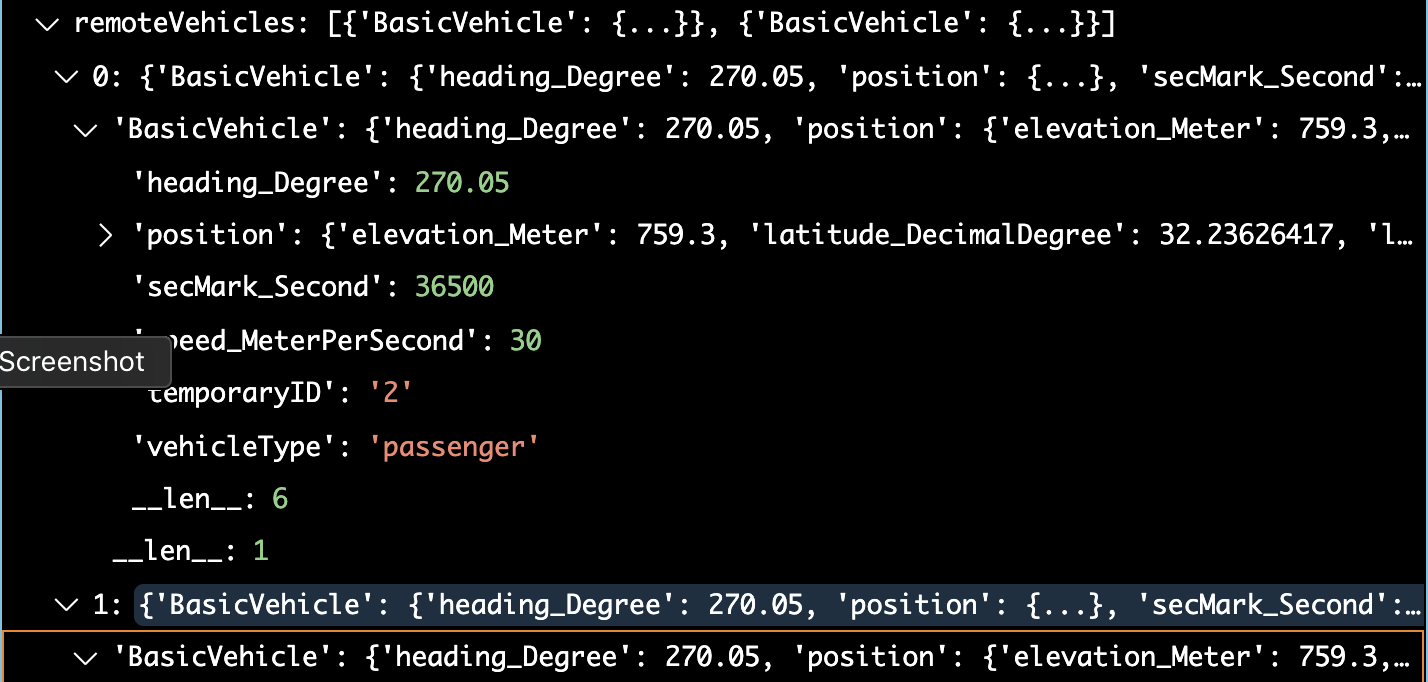


Figure 3. Sample list of remoteVehicles

The next section contains the MAP message data that has been received by the OBU.

For each map, the following data is sent to the hmi:

Table 3. Available MAP data fields

|  |  |  |
| --- | --- | --- |
| Data Field | Description | Units and Range |
| Intersection ID | Intersection id number | Integer (0…65535) (0 to 255 are reserved for testing only |
| Descriptive Name | Name fo the intersection | 1….63 characters (IA5 string) |
| map\_active | Boolean value that is True if the vehicle is on this MAP or False if it is not. | True or False |
| map\_age | Length of time since the last reception of this MAP from an RSU | 0-300 seconds |
|  |  |  |

The map information is sent to the hmi using a list of available maps:

"availableMaps": availableMaps,

Where availableMaps is a list of dictionary entries for each map based on the data in Table 1 above. An example of the availableMaps list is shown in Figure 4.

Figure 4. Example availableMaps list showing 5 available maps.

[{'IntersectionID': 101, 'DescriptiveName': 'Daisy Mountain and Gavilan Peak', 'active': True, 'age': 0}, {'IntersectionID': 102, 'DescriptiveName': 'Daisy Mountain and Dedication', 'active': False, 'age': 1}, {'IntersectionID': 103, 'DescriptiveName': 'Gavilan Peak and Boulder Creek High School', 'active': False, 'age': 201}, {'IntersectionID': 104, 'DescriptiveName': 'Gavilan Peak and Memorial Way', 'active': False, 'age': 101}, {'IntersectionID': 105, 'DescriptiveName': 'Daisy Mountain and Hastings', 'active': False, 'age': 34}]

## Signal information

The signal information is derived from the SPaT data. It is assumed that there is 8 phases of signal phase SPaT data an 8 pedestrian phases of SPaT data. The hmi display will show a signal head with three balls (red, yellow, and green) as well as minimum end time and maximum end time. In addition, a list of all 8 phases will be show that contains the status of each phase and each ped. Figure 4has a simple drawing of the signal display.

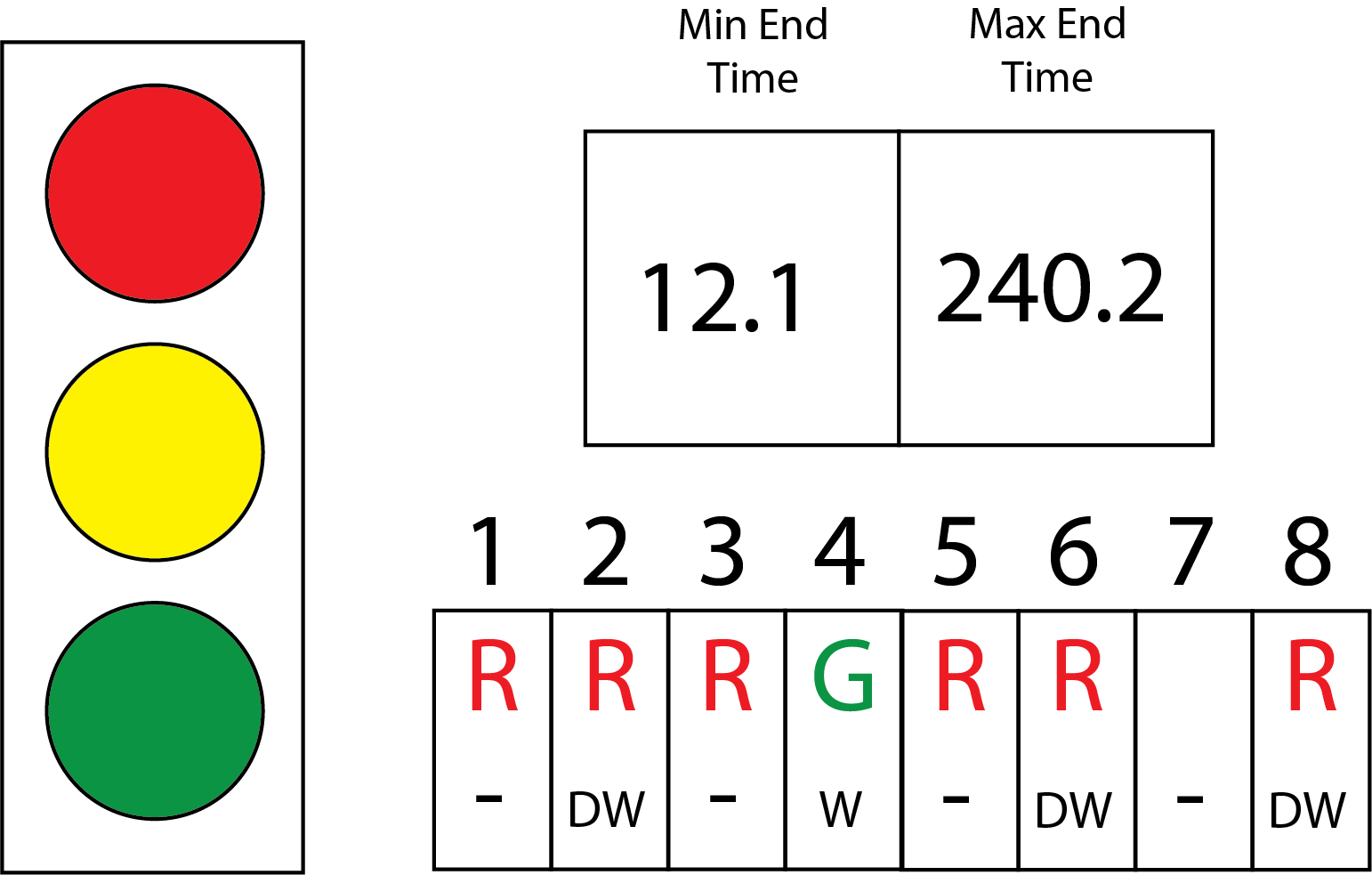


Figure 5. Signal Status Display

The signal head (left) shows the SPaT data from the vehicles current lane and current signal group. The data is contained in the infrastructure section (key) under the currentPhase key as shown in Figure 5.

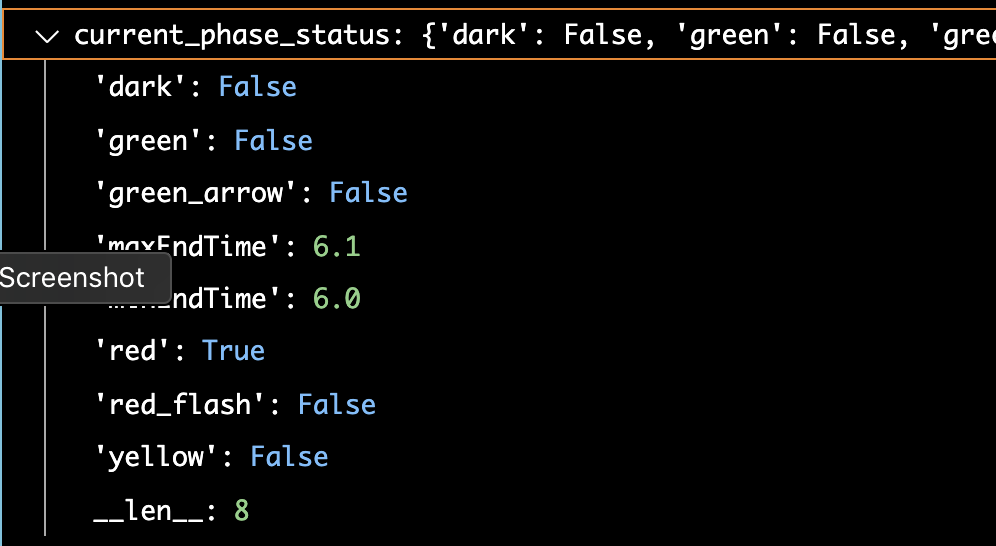


Figure 6. Current signal head data.

If the signal is “dark” all indications are dark (black). Green, red, and yellow are active if their values are True. green\_arrow is not used at this time. red\_flash may be best handled in the controller (open for discussion). maxEndTime and minEndTime are in seconds and provided for display as shown in Figure 4.

The 8-phase (and ped) status display data is referenced by the phaseStates key. The data is shown in Figure 7.

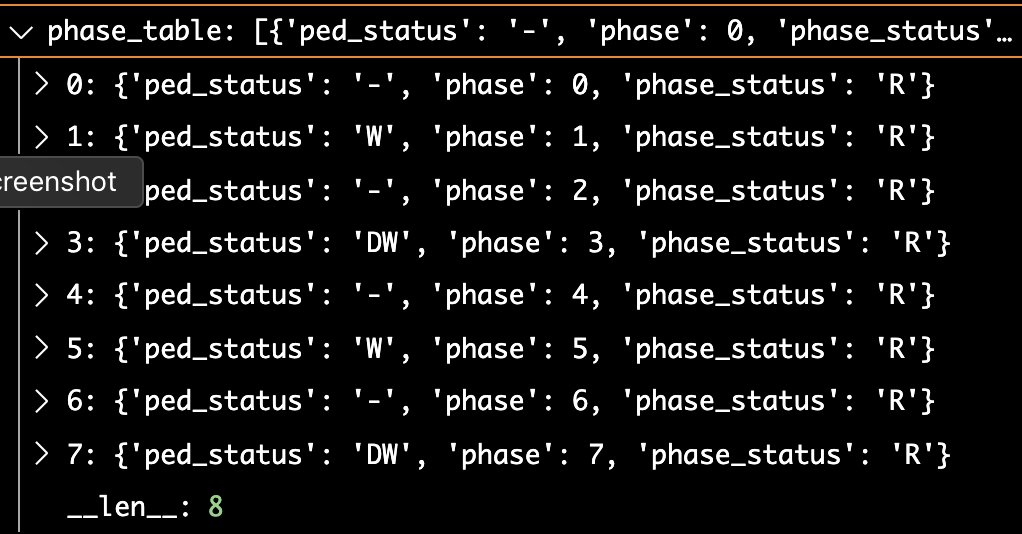


Figure 7. 8-phase status table data

## ActiveRequestTable

The active request table data is stored in the activeRequestTable key. The data for each active request is shown in Figure 8. Figure X shows a list with 5 active requests.



Figure 8. Active priority request data.

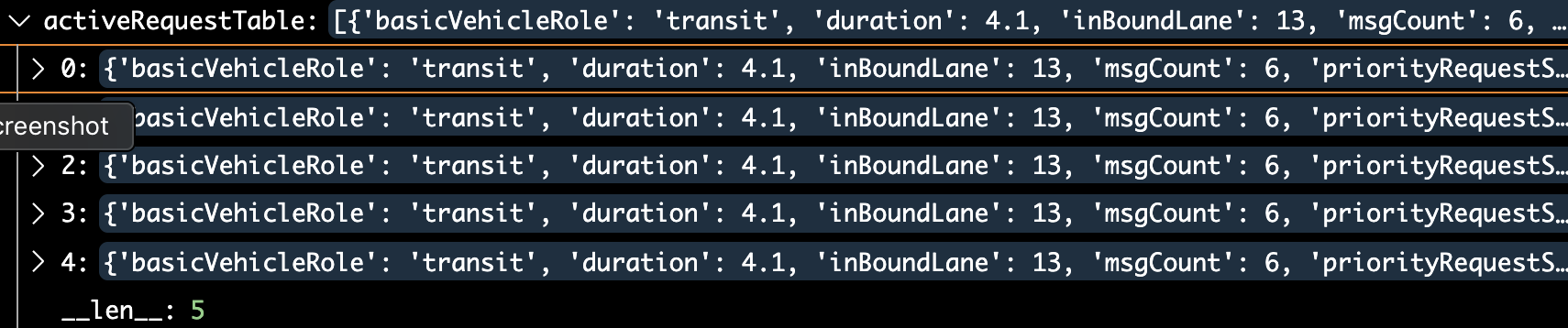


Figure 9. Active request table with 5 active priority requests.

The request table. Should be structured in the following order:

1. RequestID
2. VehicleID
3. basicVehicleRole
4. priorityRequestStatus
5. messageCount
6. inBoundLane
7. vehicleETA
8. vehicleDuration

Note: vehicle role as shown, doesn’t match the J2735 standard at this time. We need to address this issue.