SOURCE OF YOUR TECHNOLOGY

VIBEX API DEFINITION

Author	Stefan Smits
Date	19-02-2014
Version	3
Status	Qualified
Ref. no.	VIBeX-002
Ref. name	VIBeX API Definition
Project	VIBeX

Sioux Embedded Systems B.V.
Esp 405
5633 AJ Eindhoven
The Netherlands
Tel: +31 (0)40 2677100
Fax: +31 (0)40 2677101
www.sioux.eu

SOURCE OF YOUR TECHNOLOGY

TABLE OF CONTENTS

SOURCE OF YOUR TECHNOLOGY

CHANGE HISTORY

Version	Date	Author	Description
0.1	2014-10-08	Stefan Smits	Initial Version
0.2	2014-10-15	Stefan Smits	Incorporated Sioux internal review comments
0.3	2014-10-17	Stefan Smits	Incorporated Sioux internal review comments
1.0	2014-11-05	Stefan Smits	Incorporated stakeholder review comments
1.1	2014-11-19	Stefan Smits	Incorporated review comments made during implementation
1.2	2015-01-19	Stefan Smits	Incorporated several comments from TU/e
1.3	2015-03-16	Daan Gerrits	Incorporated comments from Beijer

SOURCE OF YOUR TECHNOLOGY

1 INTRODUCTION

1.1 Purpose

This document describes the VIBeX standard REST API. This interface will serve as a standard for automotive service providers to provide vehicle data to other parties. This will enable makers of applications using vehicle data to abstract the actual retrieval of vehicle data and make their applications portable across VIBeX implementations.

The API is secured so that only authorized entities can retrieve vehicle and only for the vehicles that are allowed to be queried by the entity in question.

1.2 Scope

The available functions and data of the VIBeX API are described within this document. Along with the exposed functions and data, the way in which these should be used is also described.

This document will not detail a specific implementation nor will it put constraints on a possible implementation other than its interface.

Though authentication for VIBeX API use is described in this document, no guarantees for data-privacy can be given as these rely for a large part on the rest of the implementer's platform.

This document is of special interest to implementers and users of a VIBeX API implementation.

1.3 Definitions en abbreviations

1.3.1 Definitions

Culture invariant

string	: A string of characters which represents an object as a string in an unambiguous way. Implementing languages and frameworks may either have a specific setting to parse and output objects in this way. Alternatively an "en-US" locale setting may work or parsing.
Vehicle data	: For the purpose of this document limited to dynamic data specific to a particular vehicle, for which the only way is to get it from the vehicle itself.

1.3.2 Abbreviations

API	: Application Programming Interface, a well-defined interface exposed to other (third party) applications.
DTO	: Data Transfer Object, an object that transfers data between two processes.
JSON	: JavaScript Object Notation, an open standard format that uses human-readable text to transmit data objects consisting of attribute–value pairs. It is used primarily to transmit data between a server and web application, as an alternative to XML.
OBU	: On Board Unit, the device extracting vehicle data from the vehicle to the telematics provider database.

SOURCE OF YOUR TECHNOLOGY

REST	: REpresentational State Transfer, an architectural style which puts several constraints on the way in which services can be offered via the architecture of the World Wide Web,
SW	: Software.
URI	: Uniform Resource Identifier, a string of characters used to uniquely identify a resource.
VIN	: Vehicle Identification Number, a unique identifier for motor vehicles. Though the meaning of certain characters differs across standards, world wide a 17 character wide string is used.

1.4 Referenced documents

1.4.1 Controlling documents

<>

1.4.2 Controlled documents

<>

1.4.3 Background information

The following documents are relevant to the context of the document but do not affect the contents in a direct way:

[APISEC] : Pro ASP.NET Web API Security, by B. Lakshmiraghavan. ISBN: 9781430257820.

[ISOTIME] : ISO 8601 Data elements and interchange formats – Information interchange – Representation of dates and times.

[JSON] : ECMA-404 (10-2013) The JSON Data Interchange Standard.

[OAUTH] : RFC 6749 (10-2012) The OAuth 2.0 Authorization Framework.

[REST] : Representational State Transfer. A software architecture style for distributed systems. REST has emerged as a predominant web API design model. See (a.o.): http://en.wikipedia.org/wiki/Representational_state_transfer

[TWITCODES] : Twitter developer documentation: Error Codes & Responses HTTP Status Codes: https://dev.twitter.com/overview/api/response-codes

[TWITRESP] : Twitter developer blogs: Making API responses match the request Content-Type: https://blog.twitter.com/2012/making-api-responses-match-request-content-type

2 OVERVIEW

A growing amount of vehicle data is logged in databases of different automotive service providers. Examples of this are GPS data tracking vehicle location and speed as well as, in more advanced applications, data harvested from the vehicle itself.

The VIBeX API will provide interested parties a uniform way to interface with the platforms the different providers have built to access vehicle data made available by the implementer. The data published on the VIBeX app will be mostly raw data, some filtering may have taken place to remove invalid (or confusing) entries from the data but other than that client applications will need to implement their own algorithm to form more abstract concepts as 'trips' or 'driving style appraisals'.

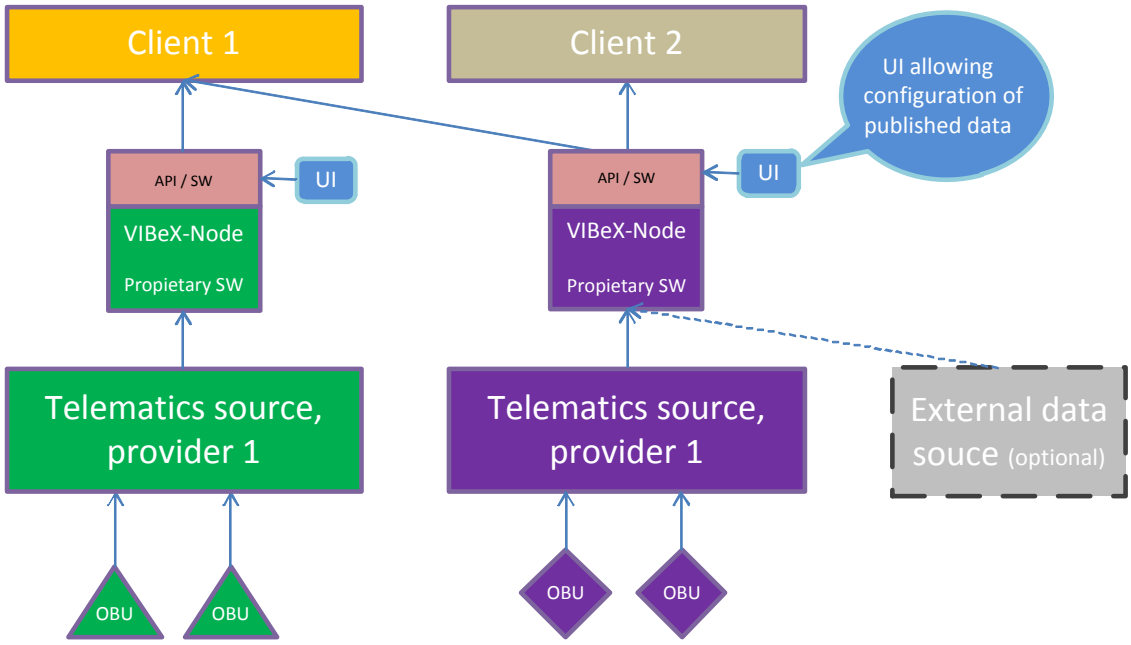

Figure 1 VIBeX Context

3 TECHNOLOGY

3.1 REST

From [APISEC]:

Representation State Transfer (REST) is an architectural style. The term REST was introduced and defined by Roy T. Fielding[1]. A service that conforms to the REST constraints is referred to as being RESTful. To be RESTful, a service has to conform to the following mandatory constraints:

1. **Client-Server constraint**: Separating user interface concerns from data storage concerns. Clients are not concerned with data storage, which is a concern of servers, and servers are not concerned with the user interface or user state, which are concerns of the client.
2. **Stateless constraint**: Each request must be an independent self-contained unit with all the necessary information for the server to service the request without looking at anything else for the context.
3. **Cache constraint**: The server must be able to label a response as cacheable or not, so that the client handles the response appropriately from the point of view of later use.
4. **Layered constraint**: The system is composed into layers, with each layer being able to see and interact with only its immediate neighbour. A layer cannot see through its neighbour. Between the client and server, there could be any number of intermediaries: caches, tunnels, proxies, and so on.
5. **Uniform interface constraint**: The service must provide a uniform interface for identification of resources, manipulation of resources through representations, self-descriptive messages, and providing hypermedia as the engine of application state.

The VIBeX API will satisfy these constraints as follows:
Client-Server constraint: The VIBeX API responds with the requested data, without bothering about client state or how data will be presented to the end user.
Stateless constraint: The VIBeX API doesn't maintain something like a session state between subsequent invocations. Each request is independent from previous requests.
Cache constraint: The VIBeX API will indicate whether the response is cacheable by the client or not.
Layered constraint: The VIBeX API describes only a single layer of the system implementing it.
Uniform interface constraint: Includes four constraints:

- Identification of resources: A resource is any data that a web API sends to its clients. A resource is identified by a [URI]. A car with a VIN "LJCPCBLCX11000237" will be represented by `https://[IMPLEMENTER]/VIBeX/x.y/Vehicles/LJCPCBLCX11000237`
- Manipulation of resources through representations. All manipulations on/with resources must be performed by using the HTTP verbs GET, PUT, POST, or DELETE. The information that is exchanged is formatted in (= represented in) JSON. Specific constraints apply for these verbs[2]:
    - GET: guaranteed not to cause any side effect and is said to be nullipotent: nothing happens to the system's state, even when called multiple times, or not called at all.
    - PUT: updating an existing resource and is said to be idempotent: the effect to the system state will be the same as that of the first call, even when called

[1] Fielding, Roy Thomas. "Architectural Styles and the Design of Network-based Software Architectures." Doctoral dissertation, University of California, Irvine, 2000.
[2] These contraints must be satisfied by the VIBeX API. We need to keep this in mind.

multiple times subsequent to the first call. (Means: performing the same update multiple times will results in no error and no difference in system state.)
    - DELETE: deleting an existing resource. Is also idempotent! (Means: Deleting a previously deleted resource should not give an error.)
    - POST: creating a new resource.
- Self-descriptive messages: Both the request message and the response message must have a self-description of the representation so that it can be parsed and handled correctly without any other knowledge. E.g. a query POST (= create) for engine on status data may look like (JSON format):
  `{[{“Signal”:2, “Time”:”2014-10-08T13:37:42”, “Location”:“51.55, 5.7”, “Value”:true}, {“Signal”:2, “Time”:”2014-10-08T14:06:42”, “Location”:{ “51.4333, 5.4833”, “Value”:false}]}`
  The format that is used/requested can be indicated in the HTTP header by using the attribute ‘Content-Type’ and/or ‘Accept’.
- Hypermedia as the engine of application state: This constraint requires that a client enters a RESTful service through a fixed URL. From that point onward, any future action a client takes will be based on what the client gets to discover within the resource representation returned by the service. This means that a service not only returns data but also links to related data (e.g. link to URL to request a more details of a driver). The links available will be based on what the client is authorized to do.

4 ARCHITECTURE

4.1 Versioning

To allow for future extensions and changes the API will be versioned. This will be done by following the VIBeX API location in the calls with a version number to form the API endpoint. The result will look as follows:

```
/[IMPLEMENTER]/VIBeX/x.y/…
```

Where:

- [IMPLEMENTER] is the URL for this specific implementation.
- ‘x’ is the major VIBeX API version.
- ‘y’ is the minor version.

Any changes to the VIBeX API that would result in a breaking change from the current API version will need an updated major version number. Changes that are backward compatible need only an increment in minor version number.

Versioning will start at 1.0, the API version described in this document is version 1.0.

4.2 Protocol Conventions

(Based on http://www.mattjcowan.com/funcoding/2013/03/10/rest-api-with-llblgen-and-servicestack/)

4.2.1 Output Format

The output format can be indicated in two ways. You can either use the “accept” header attribute or use the format parameter in the URI. Below an example of a request HTTP header with output format override:

```
Content-Type: application/json
Accept: application/json
```

If present, the format parameter will overrule the “accept” header attribute. If the format specified by the format parameter or the “accept” header attribute is not supported, the output will be formatted according to the format used for the request message.

Using the format parameter is done by appending it to the end of the request URI. Example:

```
{baseUri}/entities?format=json
```

The VIBeX API only supports JSON formatting.

The fieldnames in the JSON format equal to the attribute names as defined in the DTO’s (see section 6.3). An example of a JSON formatted response is shown below:

```
GET https://[IMPLEMENTER]/VIBeX/1.0/ApiSupportedSignals -->
200 OK
{
    ApiSupportedSignals : [
        {
            “SignalId” : “2”
            “Enabled” : true
        },
```

```
            {
                "SignalId" : "3"
                "Enabled" : false
            },
            …
        ]
    }
```

Note: JSON-LD has been mentioned as a possible way to make the API messages self-descriptive. At the time of specifying API v1.0, JSON-LD does not seem to be widely supported by libraries. Because of this including it in API v1.0 would incur additional overhead to implementers of both VIBeX servers and clients, therefore JSON-LD will not be part of this standard. It will remain under consideration for future API versions.

4.2.2 Security

The API can't be used without proper authentication. The authentication process is described in chapter 5.

After authentication all or a subset, depending on user rights, of functionality described within this document becomes available.

The data visible through the API also depends on the credentials. Authentication will only expose data from a single fleet or a subset of a single fleet.

4.2.3 Discoverable Entities

The entity API allows a user or application to discover the entities available in the API (i.e. which data, not the data itself).

Uri	Parameters	Protocol
{baseUri}/entities {baseUri}/entities?format=json	<none>	GET

The API gives you an "HREF" property back for each entity that you can store and use to navigate to each particular entity

4.2.4 Querying Entities

You can browse a specific entity type using the plural form of an entity name.

Querying all instances of an entity type

Uri	Parameters	Protocol
{baseUri}/{pluralizedEntityName}	See 'advanced queries' section	
{baseUri}/vehicles {baseUri}/vehicles/meta {baseUri}/vehicles?format=json	sort={sort} pageSize={pageSize} pageNumber={pageNumber}	GET

Returns a list of the requested entity with a maximum of 200 entities per request. Pagination must be used to retrieve more entities. The returned entities are also restricted to entities for which the requestor is authorized to access.

Querying a specific instance of an entity type using a primary key.

If the entity has multiple primary keys, just add them one after the other according to their index into the Uri.

Uri	Parameters	Protocol
{baseUri}/{pluralizedEntityName}/{pk1}/{pk2}		

{baseUri}/vehicles/LJCPCBLCX11000237		GET

Querying a specific instance of an entity type using unique constraints:
If the entity has unique constraints, they are discoverable in the field properties using the "vehicles/meta" convention.

Unique constraints can be composed of multiple fields, just append each value for each field in the constraint to the URL. You can also use the filter API to achieve the same thing.

Uri	Parameters	Protocol
{baseUri}/{pluralizedEntityName}/uc/{constraintName}/{value1}/{value2}		
{baseUri}/vehicles/uc/licenseplate/{licplate}		GET

4.2.5 Exceptions

In case of an error, the error message will also be returned in the requested format instead of the HTML often seen with 400 and 500 errors [TWITRESP].

```
{
    "responseStatus" : { "errorCode" : "NullReferenceException",
        "errors" : [  ],
        "message" : "Object reference not set to an instance of an
object."
    }
}
```

4.2.6 Advanced Queries

4.2.6.1 PAGING parameters: paging data

Paging mechanism is provided because we want to reduce the amount of data transferred via the API as much as possible.[3]

Unless otherwise specified, by default, paging is set to 10 items at a time. In this example we'll query for the 2nd page of customer data using a page size of 5.

```
/vehicles?pageSize=5&pageNumber=2
```

One thing you'll notice is the "Paging" object that is returned as part of the response. The paging object has everything that's needed to construct good paging capabilities on the client-side.
Paging JSON Object

```
{ …,
    "paging" : { "firstItemOnPage" : 6,
    "hasNextPage" : true,
    "hasPreviousPage" : true,
    "isFirstPage" : false,
    "isLastPage" : false,
    "lastItemOnPage" : 10,
```

[3]The mechanism described here is based on an existing implementation. Taken over to reduce effort. See link in chapter Protocol Conventions. Furthermore, this mechanism is a more elaborated implementation compared to what is default offered by Service Stack.

```
    "pageCount" : 19,
    "pageNumber" : 2,
    "pageSize" : 5,
    "totalCount" : 91
  }
}
```

4.2.6.2 SORT parameter: sorting on specific fields

You can easily sort your data, and sort on multiple fields as well.

In this example, the list of vehicles will be sorted according to VIN in descending order. If you don't specify a sort operator for a field, it's assumed to be 'ascending'.

```
/vehicles?sort=vin:desc
```

4.3 Return codes and errors

4.3.1 HTTP Codes

The API attempts to return appropriate HTTP status codes for every request. See table.

Code	Text	Description
200	OK	Success.
304	Not Modified	There was no new data to return.
400	Bad request	The request was invalid. An accompanying error message will explain why. A request without authentication is considered invalid and you will get this response.
401	Unauthorized	Authentication credentials were missing or incorrect.
403	Forbidden	The request is understood, but it has been refused or access is not allowed. An accompanying error message will explain why. This code is used when requests are being denied due to update limits
404	Not Found	The URI requested is invalid or the resource requested, such as a user, does not exists. Also returned when the requested format is not supported by the requested method.
406	Not Acceptable	Returned by the Search API when an invalid format is specified in the request.
410	Gone	This resource is gone. Used to indicate that an API endpoint has been turned off. For example: "This VIBeX REST API v1.0 implementation will soon stop functioning. Please migrate to API v1.1."
429	Too Many Requests	Returned when a request cannot be served due to the application's rate limit having been exhausted for the resource.
500	Internal Server Error	Something is broken. Please contact the implementer.
502	Bad Gateway	The VIBeX interface or the implementing service is down or being upgraded.
503	Service Unavailable	The servers hosting this VIBeX implementation are up, but overloaded with requests. Try again later.
504	Gateway timeout	The servers hosting this VIBeX implementation are up, but the request couldn't be serviced due to some failure within our stack. Try again later.

Based on [TWITCODES]

4.3.2 Error Messages

When the API returns error messages, it does so in your requested format. For example, an error from a JSON method might look like this:

```
{
    "errors": [
        {
          "message": "Sorry, that page does not exist",
          "code": 34
        }
    ]
}
```

4.3.3 Error Codes

In addition to descriptive error text, error messages contain machine-parseable codes. While the text for an error message may change, the codes will stay the same. The following table describes the codes which may appear when working with the API:

Code	Text	Description
10	Could not authenticate you	Your call could not be completed as dialled.
11	Sorry, that page does not exist	Corresponds with an HTTP 404 - the specified resource was not found.
12	Rate limit exceeded	The request limit for this resource has been reached for the current rate limit window.
13	Invalid or expired token	The access token used in the request is incorrect or has expired.
14	Your account is suspended and is not permitted to access this feature	Corresponds with an HTTP 403 - the access token being used belongs to a suspended user and they can't complete the action you're trying to take
15	Over capacity	Corresponds with an HTTP 503 – The VIBeX implementation is temporarily over capacity.
16	Internal error	Corresponds with an HTTP 500 - An unknown internal error occurred.
17	Could not authenticate you	Corresponds with a HTTP 401 - it means that your oauth_timestamp is either ahead or behind our acceptable range
18	Bad authentication data	Typically sent with HTTP code 400. The method requires authentication but it was not presented or was wholly invalid.
19	User must verify login	Returned as a challenge in the authorization service when the user has login verification enabled on their account and needs to be directed to the implementing service to generate a temporary password.

Based on [TWITCODES]

5 SECURITY

The VIBeX API requires authentication which follows the OAuth 2.0 standard. Specifically VIBeX is configured to work according to the “Resource Owner Password Credentials Grant” flow described in the standard [OAUTH].

A view of this flow in the context of a VIBeX implementation is shown in Figure 2.

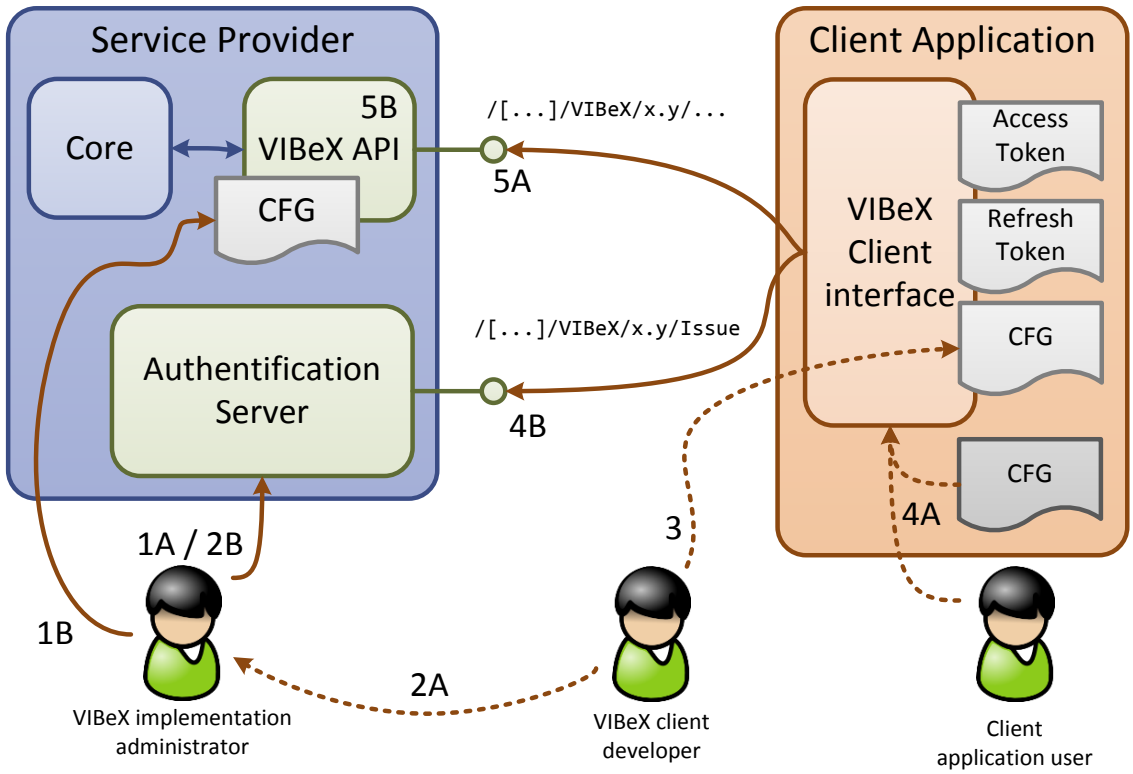

Figure 2 Authentication

1)
   a) The VIBeX implementer ensures a OAuth2 authentication server is configured. This server needs to have the ‘Resource owner flow enabled’ and needs to be accessible from the endpoint `https://[implementation]/VIBeX/x.y/security`.
   b) The administrator registers the signing thumbprint of the Identity Server in the configuration of the API.
2)
   a) A VIBeX client application developer requests access to the VIBeX API for ‘client_id’ and ‘client_secret’ tokens.
   b) The authentication server administrator supplies a ‘client_id’ and ‘client_secret’ token to the developer so the client application can authenticate itself to the provider’s server.
   Note: The developer of the client application needs to ensure that the ‘client_id’ and ‘client_secret’ are kept confidential.
3) Client’s developer registers the client_id and client_secret somewhere in the configuration of the client application.
4)

a) The client retrieves user name and password directly from a user or from client application configuration data.
   Note: To deal with multiple fleets within a database, one could concatenate the company name and user name to form a token unique across the entire database.
b) The client requests an accesstoken and refreshtoken from the Identity Server. Together with this request the company name, user name and password are passed. If credentials are correct, the identity Sever will respond with an access token and refresh token.
c) The access token and refresh token token must be stored in the client implementation. The access token can be used to perform subsequent calls to the VIBeX API, the refreshtoken can be used to obtain a new access token if the current one has expired. If both tokens don't work, the client must request new tokens.
   Note: The developer of the client application needs to ensure that the tokens are kept confidential. This might mean that on a mobile platform, software should not be allowed on jailbroken/rooted devices.

5)

a) The client can perform API requests, passing the access token with each request.
b) The VIBeX API will decrypt the access token using the signing key of the Identity Server. If this fails, the access token is rejected. If the access token is expired the token is also rejected.

6 DATA TYPES

6.1 Primitive Types

JSON defines only a limited set of data types that can be used. Below the definition of the types as they appear in the ECMA standard [JSON].

Type	Description	Examples
String	A string is a sequence of zero or more Unicode characters, wrapped in double quotes, using backslash escapes. A character is represented as a single character string. A string is very much like a C or Java string.	“TG-BX-40” “speed” “-3.14”
Number	A number is very much like a C or Java number, except that the octal and hexadecimal formats are not used.	-3.14 42 7.125e9
Boolean	Either of the values true or false	true false

Object	An object is an unordered set of name/value pairs. An object begins with { (left brace) and ends with } (right brace). Each name is followed by : (colon) and the name/value pairs are separated by , (comma).	{
		“name” : “speed”,
		“format” : “number”,
		“unit” : “Km/H”
		}

Array	An array is an ordered collection of values. An array begins with [(left bracket) and ends with] (right bracket). Values are separated by , (comma).	[“brake”, “clutch”, “cruise_active”, “speed”]
Null	An empty value, using the word null	null

6.2 Derived Types

Apart from the basic types, there are several types that derive from the base type by putting constraints on the values these can take. These derived types are described in the table below:

Type	Base	Description	Examples
Coordinate	String	Represents a GPS coordinate according to the formatting used by, among others, google. It consists of 2 doubles within quotes, separated by a comma.	“51.4333, 5.4833” “52.3562, 4.8795”
DateTime	String	Represents a date and / or time formatted in accordance with ISO 8601 [ISOTIME]. If any further requirements exist at the point where a date time is used in either a request or a reply, this must be specified.	“2014-10-08T 13:37:42+02:00” “20130907T123641 +02” “2012W011T113540Z” “2011-08-06” “2011-08-06T10”
Registration	String	Identifies a motor vehicle by its license registration. The license registration itself can assumed to be case insensitive, but there are no constraints	“TG-BX-40” “tgbx40” “AA-86”

		put on the case in the VIBeX API. As of VIBeX 1.0 .	
VIN	String	Uniquely identifies a motor vehicle. A VIN is a 17 character string consisting of the characters ‘0’-‘9’, ‘A’-‘Z’ and ‘a’-‘z’, excluding ‘I’, ’i’, ‘O’, ‘o’, ‘Q’ and ‘q’. The VIN itself is case insensitive, but there are no constraints put on the case in the VIBeX API.	“LJCPCBLCX11000237”

6.3 DTO’s

6.3.1 Vehicle

Class	Vehicle
Description	Represents a vehicle registered in the database of this VIBeX implementation to which the authenticated user has access.
Remark	-
Restricted use	Read: OK; Create/Update/Delete: not allowed.

Attribute	Type	R/W	Description
Vin	VIN	R	The VIN for the vehicle.
Registration	Registration	R	The license registration of the vehicle.

6.3.2 API Supported Signal

Class	SignalImplemented
Description	An object used to represent whether a particular signal is present in the VIBeX implementation and whether it is enabled for access.
Remark	-
Restricted use	Read: OK/Update; Create /Delete: not allowed.

Attribute	Type	R/W	Description
SignalId	number	R	The name of the signal (as defined in chapter 8).
Enabled	boolean	R/W	Whether the current configuration of the VIBeX instance allows querying of this signal type for the currently authenticated user. The accessibility of signals is set at fleet level.

6.3.3 Vehicle Supported Signal

Class	SignalAvailable
Description	An object used to represent whether a particular signal is available for a car being queried.
Remark	-
Restricted use	Read: OK; Create/Update/Delete: not allowed.

Attribute	Type	R/W	Description
SignalId	number	R	The name of the signal (as defined in chapter 8).
Available	boolean	R	Whether the signal is available on the queried vehicle.
Occurs	boolean	R	Whether the signal has occurred at least once on the queried vehicle. This can be used to check whether the specific configuration of this vehicle may not be able to report this

			signal (e.g. a KIA Cee'd without steering wheel buttons will not have an 'occurs' for these even though the signals are marked as available because they are defined for other Cee'ds).

6.3.4 Signal Value

Class	`Signal`
Description	A single entry in an array of signal entries returned as a query.
Remark	-
Restricted use	Read: OK; Create/Update/Delete: not allowed.

Attribute	Type	R/W	Description
`SignalId`	number	R	The Signal ID
`Time`	DateTime	R	The time at which the signal was logged.
`Location`	Coordinate	R	The coordinate of the vehicle at the time the signal was logged.
`Value`	Object	R	The value of the signal in an appropriate object, formatted so that it can be unambiguously transformed. The type of the content of the string is dependent on the definition of the signal, as in chapter 8.

7 INTERFACE

Image 3 shows the domain model for the VIBeX API.

Conceptually there is a set of implementation data. Currently this implementation data consists only of a Horizon, the oldest date and time for which data can be guaranteed to be present, and a list of signals available on the queried instance of the VIBeX interface with their enabled state.

Secondly there is a list of vehicles that can be queried. Every vehicle is identified by VIN and license registration. For every vehicle the list of signals available for that vehicle can be queried. Another call allows retrieval of data for the available signals.

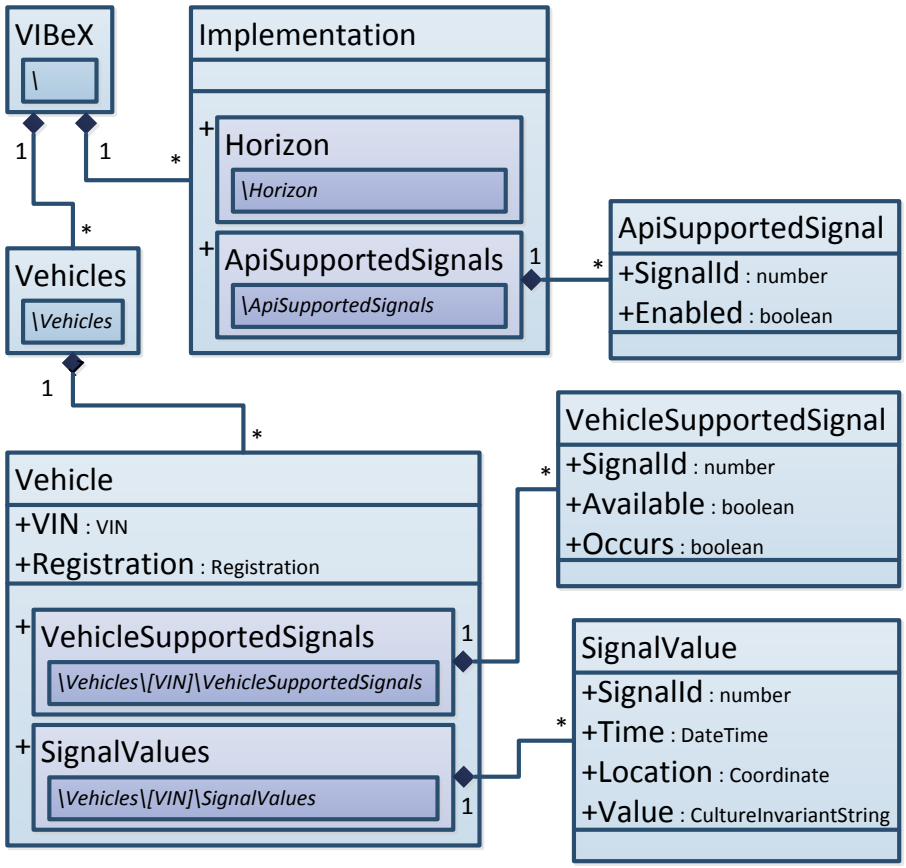

Figure 3 VIBeX Domain

7.1 Security

The security interface is responsible for passing authentication tokens to the user of the interface.

The interface will have the endpoint `https://[IMPLEMENTER]/VIBeX/x.y/Issue`.

7.1.1 POST /Issue/OAuth2/Token

Returns an OAuth2 accesstoken and a refreshtoken for authentication to the service.

Parameter	Type	Optional	Description
GrantType	string	No	'grant' or 'refresh' to either receive a new set of tokens or refresh a currently used token set.
Username	string	Yes	The name of the user of the client implementation. *Note: Must be passed on 'GrantType=grant.*
Password	string	Yes	The password. *Note: Must be passed on 'GrantType=grant.*
RefreshToken	string	Yes	The refresh token. *Note: Must be passed on 'GrantType=refresh'.*

Example:
POST
`https://[IMPLEMENTER]/VIBeX/x.y/Issue/OAuth2/Token.json?GrantType=grant&username=JanDoe&password=1234`
➔
```
{
    "Accesstoken" : "[ACCESSTOKEN]",
    "ExpiresIn" : 36000,
    "restricted_to" : [],
    "token_type": "bearer",
    "RefreshToken" : "[REFRESHTOKEN]"
}
```

7.2 Implementation

The implementation interface will allow the user of the interface to query implementation details about the VIBeX interface as well as configuration of the instance.

7.2.1 GET /Horizon

As some VIBeX implementers will log a great deal of vehicle data which may lose its value as soon as it is interpreted, or after a very short period of time after logging, it is quite likely that vehicle data will have a limited lifetime.

The GET /horizon call will return a DateTime which represents the oldest date and time for which it can be guaranteed that data will still be present for querying for the fleet for which the user is currently authorized.

Example:
GET `https://[IMPLEMENTER]/VIBeX/x.y/Horizon`
➔
```
{
    "Time": "2014-10-30T14:00:00"
}
```

7.2.2 GET /ApiSupportedSignals

A GET call on /signals will return the list of signals, in the form of 'ApiSupportedSignal' DTO's known to this implementation of the VIBeX interface and available for the currently authenticated customer. The returned entities will also show whether or not the signal is currently enabled.

Example:
GET `https://[IMPLEMENTER]/VIBeX/x.y/ApiSupportedSignals`

➔

```
{
    “Result” :
    [
        {
            “SignalId” : 2,
            “Enabled” : true
        },
        {
            “SignalId” : 3,
            “Enabled” : false
        },
        …
    ]
}
```


7.2.3 GET /ApiSupportedSignals/[SIGNAL ID]

A GET call on a particular signal can be used to ascertain whether it is implemented or not and if its implemented whether it is enabled or not.

If the signal passed by [SIGNAL ID] is not implemented the returned result will be empty (i.e. ‘{}’). Otherwise the return will be a single SignalImplemented DTO.

Parameter	Type	Optional	Description
[SIGNAL ID]	number	No	The signal for which the enabled status should be retrieved.

Example:
GET `https://[IMPLEMENTER]/VIBeX/x.y/ApiSupportedSignals/2`
➔

```
{
    “SignalId” : 2,
    “Enabled” : true
}
```


7.2.4 PUT /ApiSupportedSignals/[SIGNAL ID]

If the user is authorized to enable or disable which of the implemented signals is available for querying, the PUT /ApiSupportedSignals/[SIGNAL ID] can be used to change the availability of a signal. The ApiSupportedSignals addressed is echoed as a reply with its new value.

Parameter	Type	Optional	Description
[SIGNAL ID]	number	No	The ID for the signal that should be enabled or disabled.
Enabled	boolean	No	True if the signal should be enabled, false if it should be disabled.

Example:
PUT `https://[IMPLEMENTER]/VIBeX/x.y/ApiSupportedSignals/2?enabled=false`
➔

```
{
    “SignalId” : 2,
    “Enabled” : false
}
```

7.3 Vehicles

The vehicle interface will allow querying of logged vehicle data per individual vehicle.

7.3.1 GET /Vehicles

The GET /vehicles query can be used to retrieve a list of vehicles available for querying. The list consists of vehicle objects, containing a VIN and a license registration.

Example:
GET https://[IMPLEMENTER]/VIBeX/x.y/Vehicles/
➔

```
{
    “Result” :
    [
        {
            “Vin” : “1M8GDM9AXKP042788”,
            “Registration” : “31-RJ-JK”
        },
        {
            “Vin” : “JH4TB2H26CC000000”,
            “Registration” : “62-KKB-5”
        },
        …
    ]
}
```

7.3.2 GET /Vehicles/[VIN]

The availability of a vehicle in the VIBeX interface provider’s database can be checked by querying the vehicles list with the VIN of a single vehicle. If the vehicle exists a single ‘Vehicle’ object with the VIN and license registration of the vehicle is returned, otherwise the returned result will be empty (i.e. ‘{}’).

Parameter	Type	Optional	Description
[VIN]	VIN	No	The VIN number for the vehicle to query.

Example:
GET https://[IMPLEMENTER]/VIBeX/x.y/Vehicles/1M8GDM9AXKP042788
➔

```
{
    “Vin” : “1M8GDM9AXKP042788”,
    “Registration” : “31-RJ-JK”
}
```

7.3.3 GET /Vehicles/[VIN]/VehicleAvailableSignals

Per vehicle different signals may be available. To check which signals are available for a particular vehicle, the GET / VehicleAvailableSignals /[VIN]/signals call can be used. This will return a list with ‘SignalAvailable’ objects denoting the availability for each of the enabled signals for this particular vehicle.

Parameter	Type	Optional	Description
[VIN]	VIN	No	The VIN number for the vehicle to query.

Example:
GET
https://[IMPLEMENTER]/VIBeX/x.y/Vehicles/1M8GDM9AXKP042788/VehicleAvailableSignals
➔

```
{
    “Result” :
    [
        {
            “SignalId” : 2,
            “Available” : true,
            “Occurs” : true
        },
        {
            “SignalId” : 8,
            “Available” : true,
            “Occurs” : false
        },
        {
            “SignalId” : 308,
            “Available” : false,
            “Occurs” : false
        },
        …
    ]
}
```


7.3.4 GET /Vehicles/[VIN]/VehicleAvailableSignals/[SIGNAL ID]

The availability of a single signal for a vehicle, identified by VIN, may be checked by this call.

Parameter	Type	Optional	Description
[VIN]	VIN	No	The VIN number for the vehicle to query.
[SIGNAL ID]	number	No	The ID for the signal for which to retrieve data.

Example:
GET
https://[IMPLEMENTER]/VIBeX/x.y/Vehicles/1M8GDM9AXKP042788/VehicleAvailableSignals/2
➔

```
{
    “SignalId” : 2,
    “Available” : true,
    “Occurs” : true
}
```


7.3.5 GET /Vehicles/[VIN]/SignalValues

A set of available signal values for a single vehicle, identified by VIN, may be returned by this call. Without optional filter string all available signals are returned, if a filter string with comma separated signal ID’s is passed only the available signal data for those signals will be returned. To limit which data is retrieved, the ‘StartDateTime’ and ‘StopDateTime’ parameters can be used to limit the retrieved data to a specific interval.

Parameter	Type	Optional	Description

[VIN]	VIN	No	The VIN number for the vehicle to query.
filter	string	Yes	A filter string with comma separated signal ID's. The available signal values for the signal ID's passed into the call are returned.
StartDateTime	DateTime	Yes	The lowest date/time for which to retrieve data (inclusive)
StopDateTime	DateTime	Yes	The highest date/time for which to retrieve data (exclusive)

Example:
GET
https://[IMPLEMENTER]/VIBeX/x.y/Vehicles/1M8GDM9AXKP042788/SignalValues?Filter=2,8,191&StartDateTime=2014-04-18T13:00:12+2:00&StopDateTime=2014-04-18T13:00:14+2:00
➔

```
{
    “Result” :
    [
        {
            “SignalId” : 2,
            “Time” : “2014-04-18T13:00.123:12+2:00”
            “Location” : “51.4333, 5.4833”,
            “Value” : true
        },
        {
            “SignalId” : 191,
            “Time” : “2014-04-18T13:00.544:12+2:00”
            “Location” : “51.4333, 5.4833”,
            “Value” : 31.55
        },
        …
    ]
}
```

7.3.6 GET /Vehicles/[VIN]/SignalValues/[SIGNAL ID]

To retrieve signal data from a vehicle, this call can be used. By supplying a VIN and a signal ID all available signal data can be retrieved. To limit which data is retrieved, the 'StartDateTime' and 'StopDateTime' parameters can be used to limit the retrieved data to a specific interval.

Remark: As a special case the [SIGNAL ID] for GPS can be passed. This will result in only the timestamp and location of the signal objects in the list returned being filled in.

Parameter	Type	Optional	Description
[VIN]	VIN	No	The VIN number for the vehicle to query.
[SIGNAL ID]	number	No	The ID for the signal for which to retrieve data.
StartDateTime	DateTime	Yes	The lowest date/time for which to retrieve data (inclusive)
StopDateTime	DateTime	Yes	The highest date/time for which to retrieve data (exclusive)

Example:
GET
https://[IMPLEMENTER]/VIBeX/x.y/Vehicles/1M8GDM9AXKP042788/191?StartDateTime=2014-04-18T13:00:12+2:00&StopDateTime=2014-04-18T13:00:14+2:00

➔

```
{
    “Result” :
    [
        {
            “SignalId” : 191,
            “Time” : “2014-04-18T13:00.544:12+2:00”
            “Location” : “51.4333, 5.4833”,
            “Value” : 31.55
        },
        {
            “SignalId” : 191,
            “Time” : “2014-04-18T13:00.853:12+2:00”
            “Location” : “51.4333, 5.4833”,
            “Value” : 34.78
        },

        …
    ]
}
```

SOURCE OF YOUR TECHNOLOGY

8 DEFINED SIGNALS

ID	Name	Type	Unit	Description
3	Accessory	boolean	On/Off	Key is in accessory position
4	Actual Engine Percentage Torque	number	%	Engine torque
5	Adaptive Front Lighting System (AFS)	boolean	On/Off	Adaptive Front lighting System is enabled
6	Aftertreatment SCR Catalyst Tank Level	number	%	Level of the aftertreatment fluid
7	Aftertreatment SCR Catalyst Tank Temperature	number	°C	Temperature of the aftertreatment fluid
8	Airconditioning	boolean	On/Off	Air conditioning is enabled
9	Airconditioning Temperature	number	°C	Temperature of the airconditioning
10	Alarm	boolean	On/Off	Vehicle alarm was triggered
11	Alarm Set	boolean	On/Off	Alarm is set
12	Alarm Siren	boolean	On/Off	Alarm siren sounds
13	Ambient Air Temperature	number	°C	Temperature outside the vehicle
14	Any Door	boolean	On/Off	Any door is open
15	Any Fog Light	boolean	On/Off	Any (front or rear) fog light is on
16	Any Front Door	boolean	On/Off	Any front door is open
17	Any Gear	boolean	On/Off	Any gear is selected
18	Any Passenger Door	boolean	On/Off	Any door (except the driver door) is open
19	Any Rear Door	boolean	On/Off	Any rear door is open
20	Auto Lights	boolean	On/Off	Light switch is in automatic position
21	Axle Weight	number	kg	The weight acting on an axle
22	Barometric Pressure	number	kPa	Barometric pressure
23	Battery Current	number	A	Current from the battery pack (in electric and/or hybrid vehicles)
24	Battery Voltage	number	V	Voltage of the battery pack (in electric and/or hybrid vehicles)
25	Brake	boolean	On/Off	Brake pedal is pressed
27	Brake Front	boolean	On/Off	Front brake of a motor bike is operated
28	Brake Gear	boolean	On/Off	Brake gear of the automatic gearbox is selected

26	Brake Pressure	number	%	Pressure acting on the braking system
29	Brake Rear	boolean	On/Off	Rear brake of a motor bike is operated
268	BusActive	boolean	On/Off	Signal for determine bus activity
35	Button Diamond	boolean	On/Off	Diamond button on the steering wheel pressed
36	Button Microphone	boolean	On/Off	Microphone button on the steering wheel pressed
37	Button Mute	boolean	On/Off	Mute button on the steering wheel pressed
38	Button Phone	boolean	On/Off	Phone button on the steering wheel pressed
39	Button Phone End Call	boolean	On/Off	End call button on the steering wheel pressed
40	Button Star	boolean	On/Off	Star button on the steering wheel pressed
41	Button Talk	boolean	On/Off	Talk button on the steering wheel pressed
42	Button Volume Down	boolean	On/Off	Volume down button on the steering wheel pressed
43	Button Volume Up	boolean	On/Off	Volume up button on the steering wheel pressed
44	Cabin Interior Temperature	number	°C	Temperature of the cabin
45	Calculated Load	number	%	Calculated load of the engine
46	Calculated Torque	number	Nm	Calculated amount of torque delivered by the engine
49	Charge Current	number	A	The electric current which is going into the battery
50	Charger Connected	boolean	On/Off	Battery charger is connected (electrical vehicles)
299	Charger Connected Lock	boolean	On/Off	The charger for the electric vehicle is locked
51	Charging	boolean	On/Off	Battery is charging (electric/hybrid vechicles)
52	Climate Control	boolean	On/Off	Climate control
53	Clutch	boolean	On/Off	Clutch pedal is pressed
55	CNGLevel	number	%	Level of the CNG fuel tank
56	Cruise Active	boolean	On/Off	Cruise control is controlling the speed
57	Cruise Cancel	boolean	On/Off	Cruise cancel button is pressed
58	Cruise Control	boolean	On/Off	Cruise control system is enabled
218	Cruise Control Setpoint	number	km/h	The set point of the cruise control
60	Cruise Min	boolean	On/Off	Cruise decrease speed button is pressed
62	Cruise Off	boolean	On/Off	Cruise off button is pressed

63	Cruise Plus	boolean	On/Off	Cruise increase speed button is pressed
64	Cruise Resume	boolean	On/Off	Cruise resume button is pressed
65	Cruise Set	boolean	On/Off	Set cruise speed button is pressed
66	Dashboard Dimmer	number	%	Dashboard luminance
67	Direction Indicator	boolean	On/Off	Vehicle going forward or backward
276	Distance travelled	number	m	Distance travelled during trip
72	Door Front Left	boolean	On/Off	Front left door is opened
155	Door Front Right	boolean	On/Off	Front right door is open
132	Door Rear Left	boolean	On/Off	Rear left door is opened
182	Door Right Rear	boolean	On/Off	Rear right door is opened
68	Doors Locked	boolean	On/Off	Vehicle doors are locked
69	Doors Unlocked	boolean	On/Off	Vehicle doors are unlocked
70	Drive	boolean	On/Off	Automatic gearbox is in drive gear
71	Drive Gear	boolean	On/Off	Automatic gearbox is in drive gear
74	Driver Seatbelt	boolean	On/Off	Driver is wearing a seatbelt
75	Driver Ventilation	boolean	On/Off	Driver ventilation mode is selected
73	Drivers Demand Engine Percentage Torque	number	%	Demanded percentage of the engine torque
283	Electric Range	number	km	Range until next recharge
77	Engine Air Filter Difference Pressure	number	kPa	Pressure difference over the air filter
78	Engine Air Intake Pressure	number	kPa	Pressure of the air intake
79	Engine Coolant Level	number	%	Level of the engine coolant fluid
80	Engine Coolant Temperature (ECT)	number	°C	Temperature of the engines coolant fluid
81	Engine Current	number	A	The electric current going through the drive
82	Engine Fuel Delivery Pressure	number	kPa	Fuel pressure
83	Engine Hours	number	h	Amount of hours the engine has run
86	Engine Load	number	%	Load on the engine
87	Engine Oil Pressure	number	kPa	Pressure of the engine oil
88	Engine Oil Temperature (EOT)	number	°C	The temperature of the engine oil

89	Engine Percentage Load At Current Speed	number	%	Load on the engine
90	Engine Run	boolean	On/Off	Engine is running
91	Engine Start	boolean	On/Off	Engine starter motor active
92	Engine Torque	number	Nm	Torque delivered by the engine
94	First Gear	boolean	On/Off	First gear of the gearbox is selected
96	Forward	boolean	On/Off	Vehicle is going in forward direction
97	Four Wheel Drive	boolean	On/Off	Four wheel drive is active
300	Four Wheel Drive Lock	boolean	On/Off	The four wheel drive is locked
98	Four Wheel Drive4L	boolean	On/Off	Four wheel drive 4L is active
99	Front Doors	boolean	On/Off	One of the front doors is opened
100	Front Fog Light	boolean	On/Off	Front fog lights are on
101	Fuel CNG	boolean	On/Off	The engine runs on CNG
102	Fuel Consumption	number	mL	Amount of fuel consumed by the engine
103	Fuel Feed Pump	boolean	On/Off	The fuel feed pump is running
104	Fuel Level	number	L	Amount of fuel available in the fuel tank
282	Fuel Range	number	km	Range until next refuel
107	Fuel Rate	number	L/h	Amount of fuel consumed by engine per unit of time
0	GPS	null	-	The GPS coordinates of the vehicle
109	Hazard Lights	boolean	On/Off	The hazard lights are enabled
110	Hazard Lights Active	boolean	On/Off	Hazard lights of the vehicle are active
111	Hazard Lights Pulse	boolean	On/Off	Actuator signal that makes the hazard lights blink
112	Hazard Lights Switch	boolean	On/Off	Hazard lights switch is enabled
114	Heating	boolean	On/Off	Heating is enabled
115	High Beam	boolean	On/Off	High beam is switched on
116	High Beam Flash	boolean	On/Off	Driver is flashing the high beam
117	High Beam Lights	boolean	On/Off	High beam lights are active
298	Hill Hold	boolean	On/Off	The hill hold system is enabled
118	Hood	boolean	On/Off	Hood of the car is opened

119	Horn	boolean	On/Off	Horn is operated
120	Hydro Motor	boolean	On/Off	Hydromotor of the forklift is active
85	Intake Air Temperature (IAT)	number	°C	The temperature of the air going into the engine
123	Jaws Locked	boolean	On/Off	Jaws of the truck are locked
124	Key Out	boolean	On/Off	Key is pulled out of the ignition slot
125	Key Out Engine Run	boolean	On/Off	Key is pulled out of the ignition slot with the engine still running
126	Key Present	boolean	On/Off	Key is in the ignition slot
127	Kick Down	boolean	On/Off	Kick down of the accelerator pedal
266	LDI	boolean	On/Off	Left direction indicator
128	LDI Active	boolean	On/Off	Left direction indicator is currently controlled
129	LDI Pulse	boolean	On/Off	Actuator signal for the left direction indicator light
130	LDI Switch	boolean	On/Off	Direction indicator switch is in the left position
133	Lights	boolean	On/Off	Lights are on (Low beam and/or parking light)
135	Low Beam	boolean	On/Off	Low beam lights are enabled
84	Manifold Absolute Pressure (MAP)	number	kPa	The absolute pressure in the manifold
136	Max Pack Temperature	number	°C	Maximum battery pack temperature
137	Message Counter	number	-	Incremental message counter
307	MIL ABS	boolean	On/Off	A malfunction indication light is lit for the ABS
310	MIL Airbag	boolean	On/Off	A malfunction indication light is lit for one of the airbags
309	MIL Braking System	boolean	On/Off	A malfunction indication light is lit for the braking system
308	MIL Charge Circuit	boolean	On/Off	A malfunction indication light is lit for the charge circuit
311	MIL Diesel Engine	boolean	On/Off	A malfunction indication light is lit for the diesel engine
312	MIL Electronic Power Control (EPC)	boolean	On/Off	A malfunction indication light is lit for electronic power control
313	MIL Electronic Stability Program (ESP)	boolean	On/Off	A malfunction indication light is lit for the electronic stability program
138	MIL Emission Control System	boolean	On/Off	A malfunction indication light is lit for the emission control system
306	MIL Engine Cooling	boolean	On/Off	A malfunction indication light is lit for the engine cooling system
319	MIL Lane Assist	boolean	On/Off	A malfunction indication light is lit for the lane assist
317	MIL Power Steering	boolean	On/Off	A malfunction indication light is lit for the power steering

320	MIL Transmission	boolean	On/Off	A malfunction indication light is lit for the transmission
139	Mileage To Service	number	km	Distance to go before the vehicle needs service
140	Neutral Gear	boolean	On/Off	(automatic) gearbox is in neutral gear
141	Odometer	number	km	Travelled distance since vehicle was released from the factory
145	Operate Fork	boolean	On/Off	Fork of the forklift is currently used
146	Pack Current	number	A	Current delivered by the battery pack (in electric and/or hybrid vehicles)
147	Pack Voltage	number	V	Voltage of the battery pack (in electric and/or hybrid vehicles)
284	Park Distance Control (PDC)	boolean	On/Off	Park Distance Control is enabled
148	Parking Assistant	boolean	On/Off	Parking assistant enabled
149	Parking Brake	boolean	On/Off	Parking brake is applied
150	Parking Brake Motor	boolean	On/Off	The parking motor is enabled
151	Parking Gear	boolean	On/Off	(automatic) gearbox is in parking gear
152	Parking Light	boolean	On/Off	Parking lights are enabled
153	Parking Light Left	boolean	On/Off	The left parking light is enabled
154	Parking Light Right	boolean	On/Off	The right parking light is enabled
157	Passenger Seat Contact	boolean	On/Off	There is a passenger on the passenger seat
158	Passenger Seat Insecure	boolean	On/Off	A passenger is not wearing a seat belt
160	Passenger Seat Secure	boolean	On/Off	Either the passenger seat is empty or the passenger is wearing a seat belt
156	Passenger Seatbelt	boolean	On/Off	Passenger wears a seatbelt
161	Power Steering	boolean	On/Off	Power steering is enabled
162	PTOState	boolean	On/Off	State of the power take-off
265	RDI	boolean	On/Off	Right direction indicator
165	RDI Active	boolean	On/Off	Right direction indicator is currently active
166	RDI Pulse	boolean	On/Off	Actuator signal for the right direction indicator light
167	RDI Switch	boolean	On/Off	The direction indicator switch is activating the right direction indicator
168	Rear Center Seatbelt	boolean	On/Off	Passenger in the center rear seat is wearing a seat belt
169	Rear Doors	boolean	On/Off	One of the rear doors is open
170	Rear Fog Light	boolean	On/Off	Rear fog light is enabled

171	Rear Left Seatbelt	boolean	On/Off	Passenger in the left rear seat is wearing a seat belt
172	Rear Right Seatbelt	boolean	On/Off	Passenger in the right rear seat is wearing a seat belt
173	Rear Window Heater	boolean	On/Off	Rear window heater is enabled
174	Remote Any Key	boolean	On/Off	Any of the remote keys is pressed
175	Remote Lock	boolean	On/Off	Close key on the remote is pressed
176	Remote Trunk	boolean	On/Off	Trunk key on the remote is pressed
177	Remote Trunk Idle	boolean	On/Off	Trunk key on the remote is pressed with engine running
178	Remote Unlock	boolean	On/Off	Open key on the remote is pressed
180	Reverse Gear	boolean	On/Off	Reverse gear is selected
183	RPM	number	RPM	Engine speed
95	Second Gear	boolean	On/Off	Second gear of the gearbox is selected
188	Side Stand	boolean	On/Off	Side stand of a motor bike is being used
189	Slide Door	boolean	On/Off	Slide door of the vehicle is open
190	Snow Switch	boolean	On/Off	Snow engine map enabled
191	Speed	number	km/h	Vehicle speed
204	Speed FL	number	km/h	Speed of the front left wheel
205	Speed FR	number	km/h	Speed of the front right wheel
134	Speed Limiter	boolean	On/Off	Speed limiter is enabled
314	Speed Limiter Active	boolean	On/Off	Speed limiter is actively limiting the speed
216	Speed Limiter Setpoint	number	km/h	The set point of the speed limiter
207	Speed RL	number	km/h	Speed of the rear left wheel
208	Speed RR	number	km/h	Speed of the rear right wheel
209	Spread Width Left	number	dm	Distance the salt spreader spreads on the left side
210	Spread Width Right	number	dm	Distance the salt spreader spreads on the right side
211	Stability Program	boolean	On/Off	Stability program is active
212	Start Stop	boolean	On/Off	Start/stop system is active
213	State Of Charge	number	%	State of charge of the battery pack
214	State Of Health	number	%	State of health of the battery pack

2	Switched power	boolean	On/Off	Power supply is active
226	Throttle	number	%	Accelerator pedal position
227	Throttle Switch	boolean	On/Off	Accelerator pedal pressed
229	Time Hours	number	h	Current hour of the day
230	Time Minutes	number	min	Current minute of the hour of the day
231	Time Seconds	number	s	Current second of the hour of the day
232	Total Fuel Used	number	l	Total amount of fuel used
281	Total Range	number	km	Range until next refuel or recharge
233	Tow Bar Connected	boolean	On/Off	Tow bar is connected
234	Traction Control	boolean	On/Off	Traction control system is enabled
235	Trunk	boolean	On/Off	Trunk is open
239	Unlock All Doors	boolean	On/Off	Unlock all doors key on the remote is pressed
240	Unlock Doors	boolean	On/Off	Unlock doors key on the remote is pressed
241	Unlock Slide Door	boolean	On/Off	Unlock slide door key on the remote is pressed
246	Vehicle Locked	boolean	On/Off	Vehicle is locked
247	Vehicle Motion	boolean	On/Off	Vehicle motion detected
248	Vehicle Overspeed	boolean	On/Off	Overspeed detected
304	Warning Brake Fluid Level Low	boolean	On/Off	The brake fluid level is low
321	Warning Brake Pad	boolean	On/Off	The brake pads are worn
305	Warning Bulb Defective	boolean	On/Off	One of the lighting bulbs is defective
303	Warning Coolant Level Low	boolean	On/Off	The engine coolant level is low
301	Warning Diesel Particulate Filter (DPF)	boolean	On/Off	There is a problem with the diesel particulate filter
318	Warning ESP Disabled	boolean	On/Off	ESP is manually disabled
105	Warning Fuel Level Low	boolean	On/Off	Fuel level is low
106	Warning Fuel Lid	boolean	On/Off	The Fuel lid is not closed
302	Warning Oil Level Low	boolean	On/Off	The engine oil level is low
144	Warning Oil Pressure Low	boolean	On/Off	Engine oil pressure is low
316	Warning Power Steering	boolean	On/Off	There is a minor problem with the power steering

315	Warning Tyre Pressure Low	boolean	On/Off	The pressure of one of the tyres is low
322	Warning Windscreen Washer Fluid Level Low	boolean	On/Off	The windscreen washer fluid level is low
277	Wheel FL distance travelled	number	m	Distance travelled by the front left wheel
278	Wheel FR distance travelled	number	m	Distance travelled by the front right wheel
279	Wheel RL distance travelled	number	m	Distance travelled by the rear left wheel
280	Wheel RR distance travelled	number	m	Distance travelled by the rear right wheel
289	Window Close FL	boolean	On/Off	The front left window is being closed
291	Window Close FR	boolean	On/Off	The front right window is being closed
293	Window Close RL	boolean	On/Off	The rear left window is being closed
295	Window Close RR	boolean	On/Off	The rear right window is being closed
290	Window Open FL	boolean	On/Off	The front left window is being opened
292	Window Open FR	boolean	On/Off	The front right window is being opened
294	Window Open RL	boolean	On/Off	The rear left window is being opened
296	Window Open RR	boolean	On/Off	The rear right window is being opened
285	Window Position FL	number	%	Position of the front left window (0 = closed)
286	Window Position FR	number	%	Position of the front right window (0 = closed)
287	Window Position RL	number	%	Position of the rear left window (0 = closed)
288	Window Position RR	number	%	Position of the rear right window (0 = closed)
251	Wiper	boolean	On/Off	Wipers are activated
297	Wiper Automatic	boolean	On/Off	The wipers are operated automatically
253	Wiper Fast	boolean	On/Off	Wiper is operating in the highest possible speed
254	Wiper Interval	boolean	On/Off	Wiper is in interval mode
257	Wiper Motor	boolean	On/Off	Wipermotor is enabled
258	Wiper Once	boolean	On/Off	Wiper is performing a single sweep
261	Wiper Rear	boolean	On/Off	Rear window wiper
262	Wiper Slow	boolean	On/Off	Wiper is in slow mode