BMW Group

Interface specification
BMW-Backend – WebAPI

Specification of the interface between the *WebAPI* of BMW Backend-Systems for the RemoteApp

Release:	*< Release der Schnittstelle >*
Version:	*1.0.34*
Datum:	*02.02.2017*
Status:	***in Arbeit** / vorgelegt / freigegeben*
Autor:	*Konrad Hübner, LT-Z-4*
Ablage:	*ConnectedDrive Interface Repository*

Verteiler:

	EI-61	*Florian Wagner, Sascha Paul, Katharina Faustmann*
	FG-711	*Thomas Kupka*
	FG-712	*Albrecht Büchner, Ulrich Frank*
cc:	*FG-662*	*Roman Schlich*

History:

Version	**Date**	**Authors**	**Changes**
0.5	2012-06-13	Hübner, EI-43	Initial creation
0.5.1	2012-06-18	Hübner, EI-43	Removed statistics details, reduced statistics data
0.5.2	2012-06-20	Hübner, EI-43	Feedback of FZ integrated
0.5.3	2012-07-03	Hübner, EI-43	Small updates in API: • RangeSpider may not be available •
0.5.4	2012-07-20	Hübner, EI-43	Minor design changed; feedback of FZ-32 incorportated
0.5.5	2012-07-23	Hübner, EI-43	Data detail corrections
0.5.6	2012-07-25	Büchner, FZ-322	Feedback

0.7.0	2012-08-01	Hübner, EI-43	Added statistics section, reworked charging station search
0.7.1	2012-08-02	Hübner, EI-43	Updated charging / connection status
0.8.0	2012-09-21	Hübner, EI-43	Statistics update, minor fixes
0.8.1	2012-10-22	Hübner, EI-43	Statistics: Efficiency values updated, changed from indicator to percentage-like value
0.8.2	2012-11-12	Hübner, EI-43	Preferred Partner Info for Charging Stations
0.8.3	2012-12-11	Hübner, EI-43	CBS Mapping, Adoptions due to implementation learnings
0.8.4	2012-12-13	Hübner, EI-43	Intermodal Routing, Charging Station Status, Last Destinations
0.8.5	2013-01-07	Hübner, EI-43	HTTP Headers added, intermodal route finished
0.8.6	2013-01-08	Hübner, EI-43	Feedback regarding intermodal routing
0.9	2013-01-25	Hübner, EI-43	Intermodal Routing format update, statistics, minor attribute fixes
0.9.5	2013-02-15	Hübner, EI-43	ChargingStations extended, statistics adjusted
1.0	2013-05-21	Hübner, EI-64	Added changes for combustion cars; fixes resulting from app / backend requirements
1.0.1	2013-06-17	Lucic, FG-662 (ext.)	Chapter “Intermodal Routing”. Added providerIcon URL as a new attribute in the JSON response for the “Intermodal Routing” request.
1.0.2	2013-06-20	Büchner, FG-712	OUT_OF_ORDER removed from chargingstation availability enum
1.0.3	2013-08-08	Hübner, EI-64	Added filter request for charging stations and local search API Removed useAsDestination for sendPoi as not supported by vehicle
1.0.4	2013-08-15	Hübner, EI-64	Added maxresults param to LocalSearch request
1.0.5	2013-08-26	Hübner, EI-64	Added GoogleMapsTile based search Added range estimation information to vehicle resource
1.0.6	2013-09-20	Stäblein, FG-662 (ext.)	Updated Resource ChargingStation (Attribute conditions) Removed GoogleMapsTile search Added searchstatic and dynamicdata to Resource ChargingStation Updated Resource Vehicle Status (Attribute updateReason - Enum values)
1.0.7	2013-09-24	Lucic, FG-662 (Ext.)	Resource vehicle : attribute intermodalRouting: reduced value set to NOT_SUPPORTED and SUPPORTED.
1.0.8	2013-10-23	Faustmann, EI-61	Changes to support combustion cars (license plate, DWA, climate timers, RDU US, local search variant)
1.0.9	2013-10-23	Faustmann, EI-61	Better feedback for remote services, better information about actual execution status. Remote Service status attributes extended.
1.0.10	2013-10-23	Faustmann, EI-61	Updated vehicle status because of LSC for combustion cars and PHEVs. (CR 23976, new status attributes)
1.0.11	2013-11-12	Stäblein, FG-662 (ext.)	Updated Resource ChargingStation (CR 23687, 2 different versions)
1.0.12	2013-12-16	Stäblein, FG-662 (ext.)	Correction of several attributes and JSON Examples. Specified versioning of Method to retrieve RS exection status
1.0.13	2014-01-18	Hübner, EI-64	Updated spec to harmonize requests and data structures as discussed with stakeholders
1.0.14	2014-01-31	Hübner, EI-64	Added PHEV specific values
1.0.15	2014-02-25	Hübner, EI-64	Updated climateControl feature due to vehicle problems with this service

1.0.16	2014-03-21	Hübner, LT-Z-4	Added T2/T3 CR features
1.0.17	2014-04-04	Schlich, FG-662	changed parameter name for chargingStation resrouce dynamicData from chargingStationIds to ids in accordance with Konrad, Sascha and Patrick Groß
1.0.18	2014-05-07	Lucic, FG-662 (ext.)	Resource filter: change attribute acOnly to supported-ChargingModes Resource searchStatic: change attribute acOnly to supportedChargingModes
1.0.19	2014-05-23	Büchner, FG-712	Update getVehicleStatus Logik (Subscription based) Added Plugtypes CEE, TYPE_3
1.0.20	2014-05-27	Büchner, FG-712	Anpassungen WebAPI Efficiencytainment (Statistik)
1.0.21	2014-06-18	Büchner, FG-712	RemainingFuel, MaxFuel wieder als Float
1.0.22	2014-07-02	Büchner, FG-712	TWO_TIMES_TIMER statt DEPARTURE_TIMER für PHEV (Mail v. Katharina) Category Extension for Search POI (->CAB)
1.0.23	2014-07-09	Büchner, FG-712	CEE, TYPE_3 Connector Typen ergänzt, Connector “label” zugefügt
1.0.24	2014-07-23	Büchner, FG-712	TID in SearchPOI CDP Aufruf, IscType/countryCode in Vehicle Resource
1.0.25	2014-10-17	Büchner, FG-712	Erweiterung Charging Station Access PUBLIC, PRIVATE, COMPANY, UNKNOWN ServiceType, Location UNKNOWN dazugefügt
1.0.26	2015-04-28	Manhardt, US-L-1	Added CS29488 Query Autocomplete for Search POI.
1.0.27	2015-05-29	Manhardt, US-L-1	Added CS26114 New chargingStations attribute ‘Operator’ and ChargingStations filter option ‘Operators’. Added HTTP compression. Resort basic interface information. Fixed chapter numeration.
1.0.28	2015-07-09	Manhardt, US-L-1	Remote360
1.0.29	2015-08-06	Manhardt, US-L-1	Removed weekdays from charging overrideTimer Added vehicle image request for every (not mapped) vin.
1.0.30	2015-08-10	Manhardt, US-L-1	Added attribute timestamp in milliseconds in remote360executionStatus Added Smart Charging parameter chargingMode and chargingPreferences
1.0.31	2015-09-01	Manhardt, US-L-1	Added /weather/forecast and /weather/highresolution endpoints (T3/15).
1.0.32	2016-03-22	Manhardt, US-L-1	Add RemoteServiceExecutionHistory (implemented in T1/2016). Renamed /api/v1/ to /webapi/v1/.
1.0.33	2016-05-11	Manhardt, US-L-1	Secret Knowledge for RDU in US.
1.0.34	2017-02-02	Topalovic, ESG – AXE1	Changes from WebAPI T1.5_2017 CR-26259 und NGTP T1.5_2017 CR-26260 Update: 3.3.11.1 Remote360 Status; Add: 3.3.11.2 Remaining play protection time

Content

1 Introduction ... 5
1.1 Current situation ... 5
1.2 Architectural overview ... 5
1.3 Description of the interface ... 5
2 Functional and non-functional requirements ... 7
2.1 Interface data model ... 7
2.2 Communication ... 7
2.3 Interface Behavior ... 7
2.3.1 Timeliness of data ... 7
2.3.2 Completeness of data ... 7
2.3.3 Data validation ... 7
2.3.4 Availability in production environment ... 8
2.3.5 Response time / performance ... 8
2.3.6 Information protection and data privacy ... 8
2.3.7 Security ... 8
3 Description of the physical interface ... 9
3.1 Interface Overview ... 9
3.2 WebAPI Interface ... 9
3.2.1 General information ... 9
3.2.2 Handling of HTTP POST requests ... 10
3.2.3 Versioning ... 10
3.2.4 Language ... 10
3.2.5 User information / Header fields ... 10
3.2.6 HTTP compression ... 10
3.2.7 Communication Protocol ... 11
3.2.8 Access Information ... 11
3.2.9 Test and Quality Assurance ... 11
3.3 Resources ... 11
3.3.1 Resource UserProfile ... 11
3.3.2 Resource UserSettings ... 11
3.3.3 Resource Vehicle ... 12
3.3.4 Resource Vehicle Status ... 35
3.3.5 Resource Charging Profile ... 45
3.3.6 Resource RangeMap ... 48
3.3.7 Resource ChargingStation ... 50
3.3.8 Resource StatisticsData ... 58
3.3.9 Intermodal Routing ... 63
3.3.10 Resource LocalSearch POI ... 67
3.3.11 Remote360 ... 70
3.3.12 Resource Weather ... 74
4 Organizational topics ... 78
4.1 Responsibilities ... 78
4.2 Problem management ... 78
4.3 Change management ... 78
4.4 Dates ... 79
4.5 Open aspects ... 79
4.6 Signatures ... 79

5 Appendix 80
5.1 Additional information 80
5.1.1 Resource ChargingStation variants 80

1 Introduction

This document specifies the WebAPI which is provided by BMW for use by applications through the internet. The main goal is to implement the smartphone apps for BMWi based on this API. The specified feature set defines API version 1. The API shall and will be extended in the future and should become the central access point to BMW data and services over the internet. The API must encapsulate BMW internal systems and data structures and expose them in a format being as self-explaining as possible.

1.1 Current situation

As of today, BMW uses a proprietary library to execute services in the backend (CDxLib). As a result, there is no clear system separation between Apps and backend, as the complete communication is handled by the library and not the app itself. The required use of the library currently restricts the supported platforms to iOS and Android, with WindowsPhone not being supported and completely web-based clients not usable. In addition, the current libraries do not encapsulate system internal data and request structures enough. The interfaces used by external systems should follow the goal to be self-explaining and use-case centered.

1.2 Architectural overview

Providing a simple and clear API to the outside world is essential for the generation of new services and a flexible and decoupled development of smartphone apps.

Latest developments in the area of web APIs has resulted in a predominance of REST based APIs. In addition, the typically used data format is JSON and most of the newly created APIs do only support JSON and no XML.

Authentication and authorization of those web APIs is moving towards OAuth, an open standard for authentication and authorization. It is straight forward and product support for the required infrastructure is improving quickly.

1.3 Description of the interface

The interface is intended to provide a clear and stable communication basis towards BMW backend systems for all external applications. All backend systems shall be encapsulated behind those one interface in order to provide a single point of access. The advantage of a single centralized API access point is that only one authentication and authorization is needed to access all exposed functionality.

Initially, the API shall support required functionality for the BMWi smartphone apps. This includes all current MyBMWRemote App features, as provided today through the CDxLib.

An industry standard based web-interface like the one described in this document is currently not provided by BMW. There are services accessible through http, but the interfaces are not stable and publicly released.

This interface definition only provides the technical description of the interface (methods, data types), but does not define the complete set of functionality, which must be implemented in the underlying backend systems.

2 Functional and non-functional requirements

2.1 Interface data model

All data entities are described along with the corresponding API calls. In general, this API provides access to:

- User profile
- Vehicle base data
- Vehicle status data
- Charging Profile
- Range Spider
- Destinations / Intermodal Route
- Charging Stations
- Statistics Data

2.2 Communication

The interface must be stateless. Any API call can be performed without the need of a specific flow of API calls before. All API calls are synchronous, meaning they return a result immediately. The only exceptions are remote services, which cannot return the execution result as long as the vehicle hasn’t answered. Nevertheless, the API call to trigger a remote service immediately returns data identifying the remote service execution and enabling the client to request status updates.

2.3 Interface Behavior

2.3.1 Timeliness of data

All data provided by the interface must be real-time data from the backend systems. As soon as new data has been received from a vehicle, the application must get this new data as a result upon sending a request to the backend. If data is not updated in real-time, it must contain a timestamp of the last update so that the user can identify outdated / old data.
This does not mean that the data must be real-time from the vehicle. Direct vehicle communication is not intended to happen and the data returned by the API may be not fully up-to-date. Nevertheless, all data received by the vehicle must be processed immediately and presented through the API.

For the version of the WebAPI specified in this document, a timestamp must be attached to statistics data (when has this data been calculated?) and to vehicle status information (when has this data been collected). Regarding vehicle status information, the timestamp is transmitted by the vehicle as part of the LastStateCall.

2.3.2 Completeness of data

The provided data must be as complete as possible. Whenever the vehicle does not send data, it must be replaced with reasonable / valid other values or stay empty. The returned data must always represent a consistent and understandable state of the vehicle to avoid user confusion. If the user cross-checks with the vehicle, he must be able to verify the data returned by the API.

2.3.3 Data validation

The interface must only return data values following the data types and restrictions of this specification. All incoming data must follow the specified data formats as well.

2.3.4 Availability in production environment

The API provides customer functionality. The MCV App and the BMW RemoteApps will not be working without the API. Thus, an availability of 99.9% must be reached. A fallback system needs to be put in place, scheduled maintenance must not interrupt service availability.

2.3.5 Response time / performance

The average response time of the API must not exceed 2s. 95% of all API calls must be finished within 5s. The API provides customer services, performance is directly noticed by customers. Response times are measured from the time a request fully arrived at the BMW backend until the moment the BMW backend starts returning result data to the caller.

2.3.6 Information protection and data privacy

The WebAPI provides access to customer data and customer accounts. Every API method must be classified regarding the data protection requirements. All API calls returning customer data must only return this data for the respective user. No access to any personal data must be possible from other user accounts. A valid authentication and authorization must be required for those API calls. Additional means must be implemented to continuously check the authorization within the backend systems.

2.3.7 Security

The services must be implemented with maximum security in mind. Login and Communication are secured through the OAuth / REST Gateway. Application level security must be provided by the WebAPI implementation.

3 Description of the physical interface

3.1 Interface Overview

The interface described in this document is provided behind one base URL. It must be accessible through https protocol from the Internet without any restrictions, except for the implemented user authorization and authentication mechanisms.
Authentication is done based on the c2b infrastructure; account verification has to be done by the OAuth endpoint at c2b. Accessing single API features must be authorized upon calling them.

3.2 WebAPI Interface

This chapter describes the actual interface and all data structures available through the interface. All URLs to access the interface and resources are described as well as the data structures being returned. The data can contain embedded resources. The API goal is to limit the number of the required calls for all typical use-cases to a minimum while keeping data separation up where required. All available resources / data structures are described in detail and with an example in the following chapters. Authentication will use OAuth and is not explicitly stated for each API call. Authentication and use of OAuth tokens is required for every API call.

Whenever an unsupported method is used with a resource, http status code 405 METHOD NOT ALLOWED must be returned.

3.2.1 General information

Whenever an http 201 CREATED is returned, a header named „Location“ must be included, pointing to the created / updated resource (URL).

In case of all errors (HTTP status codes other than 2xx), the answer must contain a response JSON as the message body describing the reason for the error, encoded as an error code and with an English written description. This may help the client to react appropriate. The error code must be clearly defined and may only be used according to its meaning.

Example:

JSON

```
{
      “error”: {
            “code”: 4711,
            “message”: “Required data is missing”
      }
}
```

Whenever an invalid OAuth access token is used, **ANY** API call must return

- 401 UNAUTHORIZED in case of a **missing or invalid** token.
- 403 FORBIDDEN in case of a **valid** token, but **when accessing a forbidden** resource.

with a corresponding error message that the token is invalid. This is not explicitly stated for each API call but must be implemented. The REST Gateway must provide this functionality.

3.2.2 Handling of HTTP POST requests

Any POST request will encode its payload using the FORM-URLENCODING scheme. The respective content header must be set (content-type application/x-www-form -urlencoded). The JSON payload must always be set as the „data“ parameter. Other parameters may be used to transfer additional information not directly belonging to the REST request (like OAuth token etc).

3.2.3 Versioning

All API calls must contain versioning information. This ensures that in case of API updates services being based upon the old API do not break. API versions must be maintained for a certain period of time, ensuring that consuming services have enough time to migrate.
The version information is included in the URL path as the first dynamic element. An example looks like this:

```
GET /webapi/v1/user HTTP/1.1
```

Whenever fundamental changes to the API are made, the version must be updated. The API should stay stable as long as possible, so this should not happen very often.

3.2.4 Language

All API calls must contain language information. The language and region of the user app is set as the Accept-Language header for each request. The header must follow HTTP protocol specification; any allowed header must be accepted. The backend system must try to fulfill the users preferences.

```
Accept-Language: de-DE,de;q=0.8, en;q=0.7
```

In each case where language specific content is sent from the backend, it should respect the transmitted header. Examples are CheckControlMessages and ConditionBasedServices.

3.2.5 User information / Header fields

By definition, the REST gateway will send required user information as HTTP headers internally to the WebAPI. The used headers are

```
X-rgw-gcid:  xyz123abc                         GlobalCustomer-ID
X-rgw-token: 7f1794b09b48caf27aa6f14855b62938  OAuth-Token
X-rgw-tid:   62051                             Lisboa-ID
```

Depending on the market a customer belongs to, either GCID or Lisboa-ID may be present. The access token must always be present. These headers are only internally used and MUST NOT be set by the client application. The headers may also not be part of any response to a requesting client.

3.2.6 HTTP compression

HTTP compression can be used for requests to reduce the payload and improve transfer speed. HTTP data is compressed before it is sent from the server.

```
Accept-Encoding: gzip
```

3.2.7 Communication Protocol

The WebAPI can be reached via a BMW owned internet address through https. The base internet address (domain) must always be reachable to ensure that services built upon the API are kept working.

3.2.8 Access Information

Access to the interface shall be based on OAuth. An OAuth token must be used whenever requests are sent to the API. If the OAuth authentication has not been completed or insufficient login data is sent, the backend must return an http 403 FORBIDDEN status. With the corresponding error information in the return body.

OAuth client secrets can be requested from FG-712.

3.2.9 Test and Quality Assurance

The interface must be continuously tested. Each version of the API must have its own automatic test set, which checks all available functionality. Whenever API changes are made, all existing test cases must continue to run. The test set must also ensure that the transmitted data follows the data format specified. As long as old API versions are officially supported, the test set initially specified for the API version must continue to run without any modifications.

Testreports of the interface must be available at any time:

- Set of executed tests
- Result of the executed tests
- History of testcase execution

Test coverage must include

- all successful API calls
- Erroneous API calls with a check of the expected errors

Tests must be executed with special test accounts to continuously provide the same set of data. No vehicle must be needed. The tests must ensure complete component-testing of the API.

3.3 Resources

3.3.1 Resource UserProfile

The resource of a user profile is not needed in version 1 of the WebAPI. We do not store or request user-specific data. As this might change in the future and all data returned by the WebAPI is always in the context of a user, the user resource will be included in the requests path, even though it cannot be requested specifically. This allows an easy extension with some kind of user profile in the future.

3.3.2 Resource UserSettings

The API provides a mechanism to store generic user settings for a certain user. This mechanism allows each client app to create its own settings data, even though the same API is shared. Settings are stored as a simple key-value mapping. The key is a custom string identifier, the value is a JSON object in its string representation.

REST Methods

Retrieve user settings for a given key

```
GET /webapi/v1/user/setting?name=PHEV_APP HTTP/1.1
```

The call returns the stored JSON object for the key PHEV_APP and returns it with content-type application/json to the caller.

HTTP status codes:

- 200 OK
- 404 NOT FOUND in case there are no settings stored for the given key

Store user settings for a given key

To store settings, a POST request is performed. The following parameters must be provided:

Name	Type	Description	Condition
name	string	Key to store the settings	

data	json	Json object of the specific settings. Example
		{
		“lastScreen” : "STATUS”,
		“units”: “METRIC”
		}	

This data is sent through the following request:

```
POST /webapi/v1/user/setting HTTP/1.1
```

HTTP status codes:

- 200 OK in case the settings have been stored successfuly

3.3.3 Resource Vehicle

This section describes all data fields and operations of a vehicle. In the case of timer related services (charging and climate timers), the appropriate combination of available features is ensured by the underlying backed system. A vehicle is connected to a user. In case of car-sharing usecases, this connection must be switched automatically by the backend.
Availability of services for a vechile might not exist in the granularity this API supports, but the general decision if a vehicle is allowed to perform a service and if the service is active should be transmitted.

F15/F16 vehicles and PL6 cars starting 7/14 use the departure timer for climatization functionality.

Attribute-Specification

Name	Type	Description	Condition
vin	string	Long VIN of the vehicle	readonly
model	string	Model description, like 320d	readonly
bodytype	string	Detailed model code, like F20 or F31	readonly
driveTrain	string	can be: CONV HEV PHEV BEV_REX	Readonly

		BEV	
fuelType	string	Enum: PETROL DIESEL	Readonly; optional Specifies the fuel type in case the vehicle can use fuel
color	string	Textual description of the vehicle color	readonly
colorCode	string	Color code of the vehicle color	readonly
brand	string	[BMW | BMW_I | MINI | RR]	readonly
vehicleFinder	string	Enum: NOT_SUPPORTED NOT_ACTIVATED ACTIVATED DRIVER_DISABLED	readonly; driver_disabled means that this service had been deactivated in the vehicle, even though it could be used
hornBlow	string	Enum: NOT_SUPPORTED NOT_ACTIVATED ACTIVATED	readonly
lightFlash	string	Enum: NOT_SUPPORTED NOT_ACTIVATED ACTIVATED	readonly
doorLock	string	Enum: NOT_SUPPORTED NOT_ACTIVATED ACTIVATED	readonly
doorUnlock	string	Enum: NOT_SUPPORTED NOT_ACTIVATED ACTIVATED CALL_CENTER_ONLY SECRET_KNOWLEDGE	readonly SECRET_KNOWLEDGE is relevant for vehicles in the US and means that the service requires a secret knowledge answer for execution.
climateControl	string	Enum: NOT_SUPPORTED NOT_ACTIVATED START_TIMER DEPARTURE_TIMER	readonly; START_TIMER for ActiveE and combustion vehicles, DEPARTURE_TIMER for F15/F16 vehicles. Electric vehicles do not have dedicated climate control, preconditioning is contained in the charging control In contrast to the standard timer concept the provided times for a departure timer indicate the actual departure time. Climatization will start earlier.
climateNow	string	Enum: NOT_SUPPORTED NOT_ACTIVATED ACTIVATED	readonly
chargingControl	string	Enum: NOT_SUPPORTED NOT_ACTIVATED START_TIMER TWO_TIMES_TIMER WEEKLY_PLANNER	Readonly; WEEKLY_PLANNER for BMW I vehicles. TWO_TIMES_TIMER for PHEV, START_TIMER for ActiveE
chargeNow	string	Enum: NOT_SUPPORTED	readonly Availability of “charge now” Remote Service.

		NOT_ACTIVATED ACTIVATED	This Remote Service is only available for ActiveE vehicles.
supportedChargingModes	enum array	Enum valued: AC_LOW AC_HIGH DC	Readonly, Indicates supported charging modes of the vehicle
sendPoi	string	Enum: NOT_SUPPORTED NOT_ACTIVATED ACTIVATED	Readonly
hub	string	Enum: HUB_ECE HUB_US HUB_CN	Hub this vehicle belongs to. This mapping might be extended in the future. A configuration service could contain the actual URLs for each of the hubs.
breakdownNumber	string	Phone number for this' vehicle breakdown call contact (RoadSideAssist)	Optional; Phone number in international format, e.g.: +49 (89) 382-0
dealer	object	Dealer details, format as specified in the following table.	Optional
yearOfConstruction	int	Construction year	
intermodalRouting	string	Enum: AVAILABLE NOT_AVAILABLE	Defines if intermodal Routing is available for the vehicle.
statisticsCommunityEnabled	boolean	Indicates whether statistic features are enabled for this vehicle or not	Optional. May be missing for vehicles which do not have a statistic feature. This flag must be stored in CSI based on the app setting. Default is true.
statisticsAvailable	boolean	Indicates whether statistics community features are available for this vehicle. If not, no community based features should be shown to the user.	Will be “false” for all vehicles except those with brand “BMW_I”.
rangeMap	string	Enum: NOT_SUPPORTED RANGE_CIRCLE RANGE_POLYGON	Indicates whether a range estimation is available for the vehicle or not and if so, which mode to use
lastDestinations	string	Enum: NOT_SUPPORTED SUPPORTED	Defines if the vehicle sends last destination data to the backend
licensePlate	string	License plate of the vehicle	Optional, may not be available
climateFunction	string	Enum: NOT_SUPPORTED VENTILATION PARK_HEATING AIRCONDITIONING	Indicates which climate control function is available in the car. BMW_I or PHEV -> AIRCONDITIONING SA 534 / SA 4NB -> VENTILATION SA 536 -> PARK_HEATING
onlineSearchMode	string	Enum: NOT_SUPPORTED LIST MAP	Depending on the country the results of the online search may not be displayed on a map. The logic how to set the correct Enum value is specified in

			Resource LocalSearch POI (REST Method Search POIs)
callCenterNumber	string	Phone number of the corresponding country specific call center	Is used in markets when a remote service has CALL_CENTER_ONLY mode
hasAlarmSystem	boolean	Indicates whether the car has an alarm system.	Based upon SA 302
vehicleStatus	string	Enum: NOT_SUPPORTED SUPPORTED	Indicates whether the vehicle provides extended status information through an LSC SUPPORTED if the vehicle has one subservice out of I_LSC_IMM, LSC_BASIS, LSC_PHEV OR is an ActiveE vehicle NOT_SUPPORTED otherwise
countryCode	string	DE, UK, …	Optional, ReadOnly Aus IDS NSC
lscType	string	Enum : NOT_SUPPORTED LSC_PHEV LSC_BASIS I_LSC_IMM	Optional, ReadOnly Aus IDS LSC Subscription
remote360	String	Enum: • NOT_SUPPORTED • NOT_ACTIVATED • ACTIVATED • DRIVER_DISABLED	Readonly;
egoVehiclePath	String	e.g. G30 G30_V2	Full download URL is a combination of HUB URL + egoVehiclePath. Vorerst nur modelRange durchreichen.
asymEncryptionMethod	String	Enum: • RSA-2048 • ECC-512 • NONE	RSA-2048 vorerst einziger Rückgabewert (512 Byte maximal). ECC-512 voraussichtlich ab 19/2018 in atm2. Logik muss über WebAPI Anhand der ComUnit + ggf. PU Step ermittelt werden.
remote360packageEncryption	String	Enum: • ZIP • NONE	ZIP vorerst einziger Rückgabewert.
smartSolution	String	Enum: • NOT_SUPPORTED • NOT_ACTIVATED • ACTIVATED	Interpretation of SMARTCHARGE_CAPABLE and SMARTCHARGE_SERVICE

Dealer Information Attributes:

Name	**Type**	**Description**	**Condition**
name	string	Name of the dealer	mandatory

street	string	Street name of the dealer	
streetNumber			
postalCode	string	Postal code of the dealer	
city	string	City of the dealer	
country	string	Country of the dealer	
lat	double	Latitude of the dealer location	
lon	double	Longitude of the dealer location	
phone	string	Phone number of the dealer	

The drivetrain must be based on IDS attribute drivetrain.
The fuelType must be based on IDS attribute propulsionType.

Example responses for vehicles:

JSON (I01)

```
{
      vehicle: {
            "vin": "0000000000H007745",
            "model": "I3 REX",
            "bodytype": "I01",
            "driveTrain": "BEV_REX",
            "color": "BLACK 2",
            "colorCode": "668",
            "brand": "BMW_I",
            "vehicleFinder": "ACTIVATED",
            "hornBlow": "ACTIVATED",
            "lightFlash": "ACTIVATED",
            "doorLock": "ACTIVATED",
            "doorUnlock": "ACTIVATED",
            "climateControl": "NOT_SUPPORTED",
            "climateNow": "ACTIVATED",
            "chargingControl": "WEEKLY_PLANNER",
            "chargeNow": "ACTIVATED",
            "sendPoi": "ACTIVATED",
            "rangeMap": "RANGE_POLYGON",
            "lastDestinations": "SUPPORTED",
            "hub": "HUB_ECE",
            "breakdownNumber": "+4989358957103",
            "intermodalRouting": "AVAILABLE",
            "statisticsCommunityEnabled": false,
            "statisticsAvailable": true,
            "supportedChargingModes": ["AC_LOW", "DC"],
            "dealer": {
                  “name”:”BMW Niederlassung München”,
                  “street”:”Frankfurter Ring”,
                  “postalCode”:”80807”,
                  “city”:”München”,
                  “country”:”Deutschland”,
                  “phone”:”+49 (89) 353510”
            },
            "licensePlate": "M-DJ 1234”,
            "climateFunction": “PARK_HEATING”,
            "onlineSearchMode": “LIST”,
            "callCenterNumber": “+49 (89) 382-123”,
            "hasAlarmSystem": true
      }
}
```

JSON (I12)

```
{
      vehicle: {
            "vin": "0000000000H007746",
            "model": "I8",
            "bodytype": "I12",
            "driveTrain": "PHEV_PETROL",
            "brand": "BMW_I",
            "vehicleFinder": "DRIVER_DISABLED",
            "hornBlow": "ACTIVATED",
            "lightFlash": "ACTIVATED",
            "doorLock": "ACTIVATED",
            "doorUnlock": "ACTIVATED",
            "climateControl": "NOT_SUPPORTED",
            "climateNow": "ACTIVATED",
            "chargingControl": "WEEKLY_PLANNER",
            "chargeNow": "ACTIVATED",
            "sendPoi": "ACTIVATED",
            "rangeMap": "NOT_SUPPORTED",
            "lastDestinations": "NOT_SUPPORTED",
            "hub": "HUB_ECE",
            "breakdownNumber": "+4989358957103",
            "intermodalRouting": "AVAILABLE",
            "statisticsCommunityEnabled": false,
            "statisticsAvailable": true,
            "supportedChargingModes": ["AC_LOW", "AC_HIGH"],
            "hasAlarmSystem": true
      }
}
```

JSON (BMW Combustion Vehicle)

```
{
      vehicle: {
            "vin": "0000000000H007600",
            "model": "5er Touring",
            "bodytype": "F11",
            "driveTrain": "CONV",
            "brand": "BMW",
            "vehicleFinder": "ACTIVATED",
            "hornBlow": "ACTIVATED",
            "lightFlash": "ACTIVATED",
            "doorLock": "ACTIVATED",
            "doorUnlock": "ACTIVATED",
            "climateControl": "START_TIMER",
            "climateNow": "ACTIVATED",
            "chargingControl": "NOT_SUPPORTED",
            "chargeNow": "NOT_SUPPORTED",
            "sendPoi": "ACTIVATED",
            "rangeMap": "NOT_SUPPORTED",
            "lastDestinations": "NOT_SUPPORTED",
            "hub": "HUB_ECE",
            "breakdownNumber": "+4989358957103",
            "intermodalRouting": "NOT_SUPPORTED",
            "statisticsCommunityEnabled": false,
            "statisticsAvailable": false,
                "hasAlarmSystem": true
        }
}
```

JSON (Remote360)

```
{
        "vehicles": [
                {
                        "vin”: "WBA...H004546",
                        "bodytype": "G30",
                        "color": "CARBON BLACK METALLIC",
                        ...,


                        "remote360": "ACTIVATED",


                        "egoVehiclePath": "/ego/G30",


                        "asymEncryptionMethod": "RSA-2048",


                        "remote360packageEncryption": "ZIP"
                },
                {
                        ...
                },
                ...
        ]
}
```

REST Methods

List all vehicles of current user

```
GET /webapi/v1/user/vehicles HTTP/1.1
```

Response contains an array of vehicles. May be empty. Must not be null. This is the full set of vehicles with all attributes. **An API call to this method must be as quick as possible. Each vehicle in the account shall not add more than 2s execution time to the request. This call is executed on almost every app startup to synchronize the vehicle list.**
The vehicle list must manipulate all vehicles which are CDMA vehicles. They can be identified through the deviceType PSIM_CDMA. Those do not support any remote services. All services must be set to NOT_SUPPORTED, without any exception.

BMW i Vehicles:
All BMW i vehicles (as of today the i3 and i8) are equipped with the WEEKLY_PLANNER.

Combustion Vehicles:
For all combustion vehicles the statistics features will be both “false”, and no intermodal routing is supported.

The climateControl attribute is based on the RCC subservice. Details about how to set the respective values can be found later in this document in the corresponding section.

Vehicle Finder control of old vehicles only supports ACTIVATED / NOT_ACTIVATED / NOT_SUPPORTED values. With CR#19415 it is requested to support combustion cars with the vehicle tracking feature as available for BMW i vehicles.

- for BMW I vehicles and vehicles with SubService NO_LOCATING_VF the mobileSupport CSI attribute must be evaluated and vehicle position returned according to this attribute. No distance check must be performed
- For all other vehicles, the distance check must be performed and device lat/lon must be provided

Current market-specific limitations of remote services, independent of vehicle:

- US/CA: no RDU -> Call Center Use Case
  Vehicle attribute “doorUnlock” = “CALL_CENTER_ONLY”
- US: no RCC (subscription should not be there)
- UK: no Horn Blow
- JP: no Horn Blow
- CH: no Horn Blow
- IE: no Horn Blow
- BR: no RDU (CallCenter)
- AU: no Horn Blow

Array-Element-Name: vehicles
Example JSON:

JSON

```
{
      vehicles: [
            {
            "vin": "0000000000H007745",
            "model": "I3 REX",
            "bodytype": "I01",
            "driveTrain": "BEV_REX",
            "color": "BLACK 2",
            "colorCode": "668",
            "brand": "BMW_I",
            "vehicleFinder": "ACTIVATED",
            "hornBlow": "ACTIVATED",
            "lightFlash": "ACTIVATED",
            "doorLock": "ACTIVATED",
            "doorUnlock": "ACTIVATED",
            "climateControl": "NOT_SUPPORTED",
            "climateNow": "ACTIVATED",
            "chargingControl": "WEEKLY_PLANNER",
            "chargeNow": "ACTIVATED",
            "sendPoi": "ACTIVATED",
            "rangeMap": "RANGE_POLYGON",
            "lastDestinations": "SUPPORTED",
            "hub": "HUB_ECE",
            "breakdownNumber": "+4989358957103",
            "intermodalRouting": "AVAILABLE",
            "statisticsCommunityEnabled": false,
            "statisticsAvailable": true,
            "supportedChargingModes": ["AC_LOW", "AC_HIGH", "DC"]
            “dealer”: {
                  “name”:”BMW Niederlassung München”,
                  “street”:”Frankfurter Ring”,
                  “postalCode”:”80807”,
                  “city”:”München”,
                  “country”:”Deutschland”,
                  “phone”:”+49 (89) 353510”
            },
            "licensePlate": "M-DJ 1234",
            "climateFunction": "AIRCONDITIONING",
            "onlineSearchMode": “LIST”,
            "callCenterNumber": “+49 (89) 382-123”,
            "hasAlarmSystem": true
            },
                  ...
            {
                  "vin": "0000000000H007746",
                  "model": "I8",
                  ...
            }
      ]
}
```

HTTP status codes:

- 200 OK

The result array may be empty if the user does not have any vehicle attached to his account.

Get specific vehicle of current user

```
GET /webapi/v1/user/vehicles/<vehicle_id> HTTP/1.1
```

Response contains a single vehicle object if vehicle is found in the user context. The vehicle_id used as a parameter is the long VIN of the vehicle. A response must only be sent if this VIN is connected to the current user.

HTTP status codes:

- 200 OK
- 404 NOT FOUND

Update vehicle attributes for specific vehicle

```
PUT /webapi/v1/user/vehicles/<vehicle_id> HTTP/1.1
```

Name	Type	Description	Condition

data	json	Json object of the vehicle, identical to the response of a single vehicle. This means, data will be in the form
		{
		“vehicle” : {
		……
		}
		}	

This request only allows to change the “statisticsEnabled” call. All other attributes will remain untouched and will be ignored by the server.

Response contains a single vehicle object if vehicle is found in the user context. The vehicle_id used as a parameter is the long VIN of the vehicle. A response must only be sent if this VIN is connected to the current user.

HTTP status codes:

- 200 OK
- 404 NOT FOUND

Response contains the updated vehicle json, identical to get GET call.

Get vehicle image for specific vehicle

The following request is only allowed to for vehicles from the user's vehicle list. Access to other VINs will be denied.

```
GET /webapi/v1/user/vehicles/<vehicle_id>/image HTTP/1.1
```

The following request was introduced in T3/2015 to get images for any VIN, even if the vehicle is not in the user's vehicle list or does not support Remote Services.

```
GET /webapi/v1/vehicles/<vehicle_id>/image HTTP/1.1
```

Response contains binary data of the requested vehicle image. The following parameters are supported:

Name	**Type**	**Description**	**Condition**
width	int	Width of the image	
height	int	Height of the image	
view	string	Enum: • FRONTSIDE • FRONT • REARSIDE • REAR • SIDE • DASHBOARD • DRIVERDOOR • REARBIRDSEYE	Must not be used together with angle parameter
angle	int	Angle of the image to render	May be a positive or negative integer. Must not be used together with view parameter. The value is used as the ANGLE parameter for Cosy

This request should be used once, the resulting image should be cached due to performance implications on the image rendering system. Background will always be transparent.

HTTP status codes:

- 200 OK
- 404 NOT FOUND if no vehicle with the given VIN exists for the current user or no image is available

Example Requests:

Request with view port:
`/v1/user/vehicles/<<vin>>/image?width=400&height=300&view=FRONTSIDE`

Request with view angle:
`/v1/user/vehicles/<<vin>>/image?width=400&height=300&angle=75`

This allows to display best matching real vehicle images to the user.

Send POI to user vehicle

```
POST /webapi/v1/user/vehicles/<vehicle_id>/sendpoi    HTTP/1.1
```

Name	Type	Description	Condition
name	string	Name / description of the POI	optional
category	string		optional
additionalInfo	string		optional
organization	string		optional
street	string		optional
city	string		optional
postalCode	string		optional
country	string		optional
region	string	County of the POI	optional
website	string		optional
email	string		optional
lat	double		mandatory
lon	double		mandatory
phoneNumbers	object	Can contain up to 4 phone numbers	optional
number	string		optional
type	string	Enum: • UNKNOWN • HOME • WORK • CELL	optional

The POI data must be JSON encoded within the content of the request. The content length header must be set. Data structure follows the POI specification.

POI data must be encapsulated in a JSON object “poi” and set as the “data” parameter of the POST request. Following is an example of a POI with all available data fields set.

JSON (POI)

```
{
      “poi”: {
            “name”:”BMW Welt”,
            “category“:“Restaurant“,
            “additionalInfo”:”Hier wollen wir uns treffen”,
            “organization”:”BMW”,
            “street”:”Am Olympiapark 1”,
            “city”:”München”,
            “postalCode”:”80709”,
            “country”:”Deutschland”,
            “region”:”Bayern”,
            “website”:”http://www.bmw-welt.com”,
            “email”:”info@bmw-welt.com”,
            “lat”:48.177,
            “lon”:11.557,
            “phoneNumbers”: [
                  {
                        “number”:”+49 89 382 12345”,
                        “type”:”WORK”
                  },
                  {
                        “number”:”+49 89 382 54321”,
                        “type”:”HOME”
                  }
            ]
      }
}
```

Response contains no data.

HTTP status codes:

- 204 NO CONTENT in case of a success (no ACK for delivery)
- 403 FORBIDDEN if sending POIs to this vehicle is not allowed
- 404 NOT FOUND in case the vehicle does not exist
- 400 BAD REQUEST if the data does not contain all required fields
    - The error description should contain the fields missing

Execute remote service

```
POST /webapi/v1/user/vehicles/<vehicle_id>/executeService HTTP/1.1
```

Call starts execution of a remote service for this vehicle. The following parameters are supported (use form-encoded parameter:

Name	Type	Description	Condition
serviceType	string	the service to execute	can be one of: • VEHICLE_FINDER • HORN_BLOW • LIGHT_FLASH • DOOR_LOCK • DOOR_UNLOCK • CLIMATE_CONTROL • CLIMATE_NOW • CHARGING_CONTROL • CHARGE_NOW • START_CHARGING • STOP_CHARGING • START_PRECONDITIONING • REMOTE360
count	int	number of times a horn blow or light flash shall be executed	optional, only used for horn_blow and light_flash; if not specified, default value 2 will be used
data	String	POST body payload with climate or charge schedule (json)	optional. Depends upon the executed service type. Only for charging_control and climate_control. The payload depends on the supported climatization time scheme as specified below.

extended-StatusUpdates	boolean	Defines if extended status feedback (including push notifications) shall be used or the standard one	Optional. If not provided, value "false" is anticipated. Extended feedback is only provided for door and climate services.
bmwSkAnswer	String	Customer's Secret Knowledge Answer	Mandatory if remote service capability in vehicles list is SECRET_KNOWLEDGE.
bmwSkQuestionId	String		Optional. Not implemented yet.

There are several aspects to be checked when executing remote services:

Vehicle Finder for Combustion vehicles and Pre-NBT vehicles:

- Allow service execution for VEHICLE_FINDER for all combustion cars which support it
- Ensure that all BMW_I brand vehicles still get the vehicle finder service returned as "NOT_SUPPORTED". They receive location via LSC as of today.
- Process the result of the remote service similar to the RCP remote service
  - Extract vehicle position
  - Store vehicle position and response timestamp into CSI
  - Store the status of the attributes moving, ignitionOn and engineOn in CSI
  - Mark the service execution as successful if the service returned with an ACK
- Extend the /user/vehicles/<<vin>>/status call to support location retrieval for combustion vehicles
  - Extend the possible status types for the location with the items TOO_FAR_AWAY and VEHICLE_ACTIVE. No location may be set in this case

Remote Door Unlock in US requires the secret knowledge answer that the customer must have set in the ConnectedDrive customer portal to be executed.

To determine the set secret knowledge question id, GCDM provides the following service:
GET /gcdm/protected/{clientId}/{clientVariantId}/customers/userAccount

For a list of secret knowledge question ids including translations, the following call can be used:
GET /gcdm/public/{clientId}/{clientVariantId}/configurations/secretKnowledgeQuestions

GCDM API Documentation:
https://smuc3825.muc:8443/display/GASIS/GCDM+API+Gateway+Interface+Specification

ClimateControl Detailed Specification:

Start Timer / Departure Timer data specification

The "climatFunction" attribute indicates which function is available in the car:

- F15/F16: departure time
- All other BMW brand cars until today: start time

Vehicles with START_TIMER as their climateControl feature can send one timer in the vehicle as the start time for climatization.

Vehicles with DEPARTURE_TIMER as their climateControl feature (note: This is different to chargingControl) can send one timer, which is interpreted as a departure time.

Name	**Type**	**Description**	**Condition**
timerEnabled	boolean		mandatory

time	String	time specification of the timer	format: hh:mm – time is specified in vehicle local time, no timezone conversion is made
position	string	Enum: • TIMER1 • TIMER2	**Optional**; identifies the timer to actually set. If not provided, will default to TIMER1.

Request data for a start / departure timer which default to timer 1. This is the only method which is officially supported

JSON (for start and departure timer)

```
{
      "timer": {
            "timerEnabled": "true",
            "time": "17:35"
      }
}
```

Request data for timer with explicit selection of timer position (not officially supported)

JSON (explicit timer, not supported)

```
{
      "timer": {
            "timerEnabled": "true",
            "time": "17:35",
            "position": "TIMER2"
      }
}
```

This sets timer 2 to the selected time, timer1 is set to NOACTION in the NGTP message.

Due to vehicle restrictions it is not possible to set both climate timer at the same time. It is also not possible to set timer 2 for some vehicles. As a consequence the default behavior is to set only timer 1 for both start and departure timer vehicles. Timer 2 can be used for testing, but is not officially supported.

NBT 07/12 – 03/13 ClimateNow Issue:
In case a vehicle has deviceType TCB_NBT and the iStep PU is 07/12, 11/12 or 03/13, a remote service of type CLIMATE_NOW does not work. Thus, the service must be set to NOT_SUPPORTED for these vehicles.

ActiveE special treatment:

The ActiveE provides support for ClimateControl, ChargeControl, ClimateNow and ChargeNow services. These are implemented with a “hack” based on the two climate timers of combustion vehicles. The following paragraph defines how to implement this with the old climate timers of combustion cars.

ActiveE Climate Timer

The ActiveE vehicle only supports a single climate timer. In order to execute a climate timer, the backend must perform the following conversion:

- Use the climate timer provided in START_TIMER format to fill a standard RCC timer structure and fill the **second** timer of the NGTP message.
- Set the **first** timer to **NOACTION**
- Send the timer structure to the vehicle

Executing a remote service of service type CLIMATE_NOW must be converted by the WebAPI into a RCC remote service with the **second** timer set to „activate“ but no time given. This triggers immediate climatization in the respective vehicle.

ActiveE Charging Timer

The ActiveE vehicle only supports a single charging timer. In order to execute a charging timer request, the backend must perform the following conversion:

- Use the charging timer provided in SINGLE_TIMER format to fill a standard RCC climate timer structure and fill the **first** timer.
- Set the **second** timer to **NOACTION**
- Send the timer structure to the vehicle

ActiveE start charging / stop charging

The ActiveE is currently the only vehicle which supports the “chargeNow” remote service. The remote service will be triggered by the service types START_CHARGING and STOP_CHARGING. These are executed through a CDP interface and TSSB.

Decision to enable chargeNow and chargingControl for the ActiveE vehicle must ONLY be based on the TSBG_E82E service subscription. As this vehicle is a special case, its base data differs from typcial electric vehicles.

In case the service call succeeds, the vehicle status must be updated accordingly in CSI. The timestamp of the preconditioning activation must be stored. As soon as a new SOC entry is retrieved from the vehicle, CSI must be updated accordingly. The information if the service was successful or not can only be based on an updated timestamp of the SOC data and a comparison of status attributes. When querying the status of a START_CHARGING or STOP_CHARGING service call, the getSOC() call must internally be performed and the timestamp must be compared to the trigger time of the service execution.

HTTP status codes:

- 200 OK
- 403 FORBIDDEN if executing the requested remote service is not allowed
- 404 NOT FOUND in case the vehicle does not exist
- 400 BAD REQUEST if not all required data fields are set
- 409 CONFLICT in case another remote service is currently executing

The result data contains a unique event id which can be used to query the status of the remote service request.

Remote360 Detailed Specification:

data – detailed description for remote360			
Name	**Type**	**Description**	**Condition**
publicKey	String	PublicKey der ATM	Mandatory except when action=atmUploadPrivateKey [HEX]
encryptedPackageKey	String	Symmetrischer One-Time Schlüssel wird von der App zum Ausführungszeitpunkt generiert und mittels publicKey verschlüsselt.	Mandatory except when action=atmUploadPrivateKey [HEX]
imageCompression	int	Higher value means better quality.	Mandatory except when action=atmUploadPrivateKey [0..255]

action	string	“atmUploadPrivateKey”: trigger ATM to upload the last generated public key again.	Optional

Request example JSON (start remote360 remote service)

```
{
        "remote360": {
                "publicKey": "xxxx",
                "encryptedPackageKey": "xxxx",
                "imageCompression": 80,
                "pushToken": "xxxx"
        }
}
```

Request example JSON (trigger ATM to upload public key)

```
{
        "remote360": {
                "action": "atmUploadPrivateKey"
        }
}
```

Terminate Remote Service

Currently it is only possible to terminate an ongoing Remote360 execution.

```
GET /webapi/v1/user/vehicles/<vehicle_id>/terminateService HTTP/1.1
```

Attribute-Specification:

Input Parameters			
Name	**Type**	**Description**	**Condition**
eventId	String	The event ID of the remote service execution	mandatory

HTTP status codes:

- 200 OK
- 404 NOT FOUND if no service with the given eventide exists (anymore) for the current user.

Query Service Execution Status

```
GET /webapi/v1/user/vehicles/<vehicle_id>/serviceExecutionStatus HTTP/1.1
```

Call to query service execution status. The following parameters are supported (use form-encoded parameter:

Name	**Type**	**Description**	**Condition**

serviceType	string	the service to query the last status for	can be one of: • VEHICLE_FINDER • HORN_BLOW • LIGHT_FLASH • DOOR_LOCK • DOOR_UNLOCK • CLIMATE_CONTROL • CLIMATE_NOW • CHARGING_CONTROL • CHARGE_NOW • START_CHARGING • STOP_CHARGING • START_PRECONDITIONING
eventId	string	The event ID of the remote service execution	Optional; If an eventId is specified, the service execution status for this event must be returned. If not, the latest status for this service type will be returned.
dlat	double	Device position latitude	Optional; Required for vehicles without the option to disable vehicle tracking. Used to calculate distance to vehicle
dlon	double	Device position longitude	Optional; Required for vehicles without the option to disable vehicle tracking. Used to calculate distance to vehicle

Response Data:
Response contains service execution metadata and an optional result information:

Name	Type	Description	Condition
serviceType	string	The service type to check the status for	Optional; can be one of: • VEHICLE_FINDER • HORN_BLOW • LIGHT_FLASH • DOOR_LOCK • DOOR_UNLOCK • CLIMATE_CONTROL • CLIMATE_NOW • CHARGING_CONTROL • CHARGE_NOW
status	string	Enum • INITIATED (backend has sent the SMS to the vehicle) • PENDING (as long as no answer has been received) • DELIVERED (as soon as the NGTP message has been downloaded from the queue; only for NGTP vehicles) • EXECUTED (vehicle has answered with ACK) • NOT_EXECUTED (vehicle has answered but with NACK) • TIMED_OUT	
eventId	string	The event ID of the remote service execution	Optional; If an eventId is specified, the service execution status

			for this event must be returned. If not, the latest status for this service type will be returned.
result	string	Enum: • STATUS_CHANGED (vehicle answered with ACK and updated status could be verified) • STATUS_NOT_CHANGED (vehicle answered with a ACK but because of the state of the car command was not executed.) • STATUS_MAY_HAVE_CHANGED (vehicle answered with a ACK but we don't have enough information to be sure if the function was executed.) • STATUS_UNKNOWN (vehicle answered with a ACK but values in NGTP uplink are unknown)	
initiatedAt	date	Provides the timestamp when this remote service has been initiated	Format: yyyy-MM-dd'T'HH:mm:ss+hh:mm

The following rules need to be applied for the extended status feedback. The different status attributes that may occur when the car answers with an ACK have the following rules:

Remote Service Typ	**Status attribute**	**Rule**
Remote Door Lock	STATUS_NOT_CHANGED	If oldDoorStatus == newDoorStatus
Remote Door Lock	STATUS_UNKNOWN	If newDoorStatus == unknown
Remote Door Lock	STATUS_CHANGED	If oldDoorStatus != newDoorStatus
Remote Door Unlock	STATUS_NOT_CHANGED	If oldDoorStatus == newDoorStatus
Remote Door Unlock	STATUS_UNKNOWN	If newDoorStatus == unknown
Remote Door Unlock	STATUS_CHANGED	If oldDoorStatus != newDoorStatus
Remote Climate Control - Climate Now	STATUS_NOT_CHANGED	If SA 534 && ignitionOnStatus == true
Remote Climate Control - Climate Now	STATUS_MAY_HAVE_CHANGED	If SA 536 && ignitionOnStatus == true
Remote Climate Control - Climate Now	STATUS_UNKNOWN	If ignitionStatus == unknown
Remote Climate Control - Climate Now	STATUS_CHANGED	If ACK && ignitionOnStatus == false
Remote Light Flash	STATUS_NOT_CHANGED	If ignitionOnStatus == true
Remote Light Flash	STATUS_UNKNOWN	If ignitionStatus == unknown
Remote Light Flash	STATUS_CHANGED	If ACK && ignitionOnStatus == false
Remote Horn Blow	STATUS_CHANGED	If ACK
Remote Climate Control - Timer	STATUS_CHANGED	If ACK
Vehicle Finder	STATUS_CHANGED	If ACK

JSON

```
{
      "executionStatus": {
            "serviceType": "DOOR_LOCK",
            "status": "EXECUTED",
            "result": "STATUS_CHANGED",
            "eventId": abc123cba@bmw.de,
            "initiatedAt": "2014-03-21T09:25:30+01:00”
      }
}
```

Query Remote Service Execution History

```
GET /webapi/v1/user/vehicles/<vehicle_id>/serviceExecutionHistory
HTTP/1.1
```

Call to query service execution history. The following parameters are supported (use form-encoded parameter:

Request Parameter Description:

Name	Type	Description	Condition
maxAge	int	Maximum age of the log entry in seconds	Optional
limit	int	Maximum number of events to return	Optional
offset	int	Item to start from. Default=0;	Optional

Response Attribute Description:

Name	Type	Description	Condition
serviceType	String	The service type to check the status for	Mandatory; can be one of: • VEHICLE_FINDER • HORN_BLOW • LIGHT_FLASH • DOOR_LOCK • DOOR_UNLOCK • CLIMATE_CONTROL • CLIMATE_NOW • CHARGING_CONTROL CHARGE_NOW
status	String	Enum • INITIATED (backend has sent the SMS to the vehicle) • PENDING (as long as no answer has been received) • DELIVERED (as soon as the NGTP message has been downloaded from the queue; only for NGTP vehicles) • EXECUTED (vehicle has answered with ACK) • NOT_EXECUTED (vehicle has answered but with NACK) • TIMED_OUT	Mandatory
result	String	Enum:	Optional

		• STATUS_CHANGED (vehicle answered with ACK and updated status could be verified) • STATUS_NOT_CHANGED (vehicle answered with a ACK but because of the state of the car command was not executed.) • STATUS_MAY_HAVE_CHANGED (vehicle answered with a ACK but we don't have enough information to be sure if the function was executed.) STATUS_UNKNOWN (vehicle answered with a ACK but values in NGTP uplink are unknown)	
eventId	String	The event ID of the remote service	Mandatory
client	String	The client from where the remote service was initiated (e.g. CallCenterClient, Webportal, Remote App)	Mandatory
initiatedAt	String	Provides the timestamp when this remote service has been initiated	Mandatory Format: yyyy-MM-dd'T'HH:mm:ss+hh:mm
data	Object	POST body payload from executed service with climate or charge schedule.	Optional

JSON Sample Response

```
{
      "serviceExecutionHistory": [
            {
                  "serviceType": "DOOR_LOCK",
                  "status": "EXECUTED",
                  "result": "STATUS_CHANGED",
                  "eventId": "abc123cba@bmw.de",
                  "client": "Webportal",
                  "initiatedAt": "2014-03-21T09:25:30+01:00"
            },
            {
                  "serviceType": "CLIMATE_CONTROL",
                  "status": "EXECUTED",
                  "result": "STATUS_CHANGED",
                  "eventId": "abc123cbb@bmw.de",
                  "client": "Remote App",
                  "initiatedAt": "2014-03-21T09:28:30+01:00"
                  "data":
                        {
                              "timer": {
                                    "timerEnabled": "true",
                                    "time": "17:35"
                              }
                        }
            }
      ]
}
```

HTTP status codes:

- 200 OK
- 404 NOT FOUND if the eventId does not exist for this user

Hide Remote Service Execution History

```
GET /webapi/v1/user/vehicles/<vehicle_id>/hideServiceExecutionHistory
HTTP/1.1
```

Call to set a marker in the SPGW to hide the remote service history entries older and equal than the current timestamp. The history will not be deleted in the backend, but is not visible in any customer frontend anymore.

Resend Remote Service from Execution History

```
GET /webapi/v1/user/vehicles/<vehicle_id>/resendRemoteService HTTP/1.1
```

Resend a formerly initiated Remote Service.
Initiates a new Remote Service Execution with same parameters.
Request Parameter Description:

Name	**Type**	**Description**	**Condition**
eventId	String	eventId of a formerly initiated Remote Service.	Mandatory

Response and HTTP status codes equally to `/webapi/v1/user/vehicles/<vehicle_id>/executeService`

Update vehicle push information

```
POST /webapi/v1/user/vehicles/<vehicle_id>/updatePush HTTP/1.1
```

Request updates push information for this vehicle. Body contains the push information specification. The data update is bound to the current device. If set to “OFF”, the push information will be removed from the backend.

Attribute-Specification

Name	**Type**	**Description**	**Condition**
uuid	string	installation UUID to identify this device	required
deviceType	string	specifies the device type this token is valid for Enum: • IOS • ANDROID	required
pushToken	string	the push token to use	optional, but required when push_type is different from “off”. Only then it may be missing
pushType	string	defines how push shall be handled for this device. Also used to disable push notifications	Required

		(delete push token in the backend) Enum: • OFF • SILENT • ALWAYS	
appId	string		The app ID for the client app. This will be the bundle identifier for iOS apps and the registration ID for Android devices.

JSON

```
{
      pushInformation: {
            “uuid”: ”cba-34-bvd-aa”,
            “deviceType”: ”IPHONE”,
            “pushToken”: ”abc123-24cde-321acbc”,
            “pushType”: ”ALWAYS”,
            “appId”: “de.bmw.iRemote”
      }
}
```

HTTP status codes:

- 200 OK
- 404 NOT FOUND if the vehicle does not exist for this user

Note: This call must also be used whenever a vehicle is changed within the app or the user logs out actively.

Push messages are sent to the device as a result of the following triggers:

- Remote Service Execution ACK
    - Silent push (no message) to trigger status update if app is running
- Remote Service Execution NACK
    - Push message with matching error information
- Remote Service Timeout
    - Push message with failure notice
- LSC data update
    - Silent push (no message) to trigger data update in the app

Get last destinations

```
GET /webapi/v1/user/vehicles/<vehicle_id>/destinations HTTP/1.1
```

Request last destinations for this vehicle. This is based on the list of destinations sent by the vehicle. The array may be empty in case no destinations are available in the backend.

Attribute-Specification

Name	Type	Description	Condition

lat	double	latitude	required
lon	double	longitude	required
street	string	Street address	
streetNumber	string	Street number of address	
city	string	city	
country	string	Country of the address	
type	string	Enum: • INTERMODAL • DESTINATION	
createdAt	date		Date, formatted as yyyy-MM-dd’T’HH:mm:ssZ

JSON

```
{
      destinations: [
            {
                  “lat”: 48.11,
                  “lon”: 11.5,
                  “street”:”Teststraße”,
                  “streetNumber”:”122a”,
                  “city”:”Teststadt”,
                  “country”:”Deutschland”,
                  “type”: ”DESTINATION”,
                  “createdAt”: “2013-03-11T14:33:15Z”
            },
            {
                  “lat”: 47.21,
                  “lon”: 10.75,
                  ...
            },
            {
                  “lat”: 48.36,
                  “lon”: 11.42,
                  ...
            }
      ]
}
```

HTTP status codes:

- 200 OK
- 404 NOT FOUND if the vehicle does not exist for this user

Other calls

All calls to the vehicle path with methods PUT / POST / DELETE will be answered with a HTTP 405 METHOD NOT ALLOWED status code.

3.3.4 Resource Vehicle Status

This section describes all data fields and operations of a vehicle status. The vehicle status contains all the dynamic / changing information that has been stored in the backend by the vehicle. Depending on the vehicle, the available status information differs:

- BMW i vehicles have the most complete status information. The LSC data contains almost all the attributes available
- BMW PHEV vehicles also return much information through the LSC
- BMW vehicles with LSC support return the information available in the vehicle key
- ActiveE vehicles have a special status set for the battery, but no detailed vehicle information
- BMW vehicles without LSC support do not return any data except for their position after an executed vehicle finder remote service

After the description of all data fields, example JSON for each of the vehicle types is available.

Attribute-Specification

Name	**Type**	**Description**	**Condition**
vin	string	The long vin of the vehicle	
doorDriverFront	string	Enum: • OPEN • CLOSED • INVALID	
doorDriverRear	string	Enum: • OPEN • CLOSED • INVALID	optional (only if door exists)
doorPassengerFront	string	Enum: • OPEN • CLOSED • INVALID	
doorPassengerRear	string	Enum: • OPEN • CLOSED • INVALID	optional (only if door exists)
hood	string	Enum: • OPEN • CLOSED • INVALID	
trunk	string	Enum: • OPEN • CLOSED • INVALID	
windowDriverFront	string	Enum: • CLOSED • INTERMEDIATE • OPEN • INVALID	
windowDriverRear	string	Enum: • CLOSED • INTERMEDIATE • OPEN • INVALID	optional (only if door exists)
windowPassengerFront	string	Enum: • CLOSED • INTERMEDIATE • OPEN • INVALID	

windowPassengerRear	string	Enum: • CLOSED • INTERMEDIATE • OPEN • INVALID	optional (only if door exists)
sunroof		Enum: • CLOSED • INTERMEDIATE • OPEN • INTERMEDIATE_TILT • OPEN_TILT • INVALID	optional (only if sunroof exists)
doorLockState	string	Enum: • UNLOCKED • SELECTIVE_LOCKED • LOCKED • SECURED • INVALID	
parkingLight	string	Enum: • OFF • LEFT • RIGHT • INVALID	Status of parking lights
positionLight	string	Enum: • OFF • ON • INVALID	Status of position lights
position	object	position of vehicle	
lat	double	latitude of vehicle	optional (only if display allowed)
lon	double	longitude of vehicle	optional (only if display allowed)
heading	int	heading of vehicle	optional (only if display allowed); unit is degrees
status	string	vehicle position status Enum: • TOO_FAR_AWAY • INVALID • OK • DRIVER_DISABLED • VEHICLE_ACTIVE	Vehicle active is set based upon response data of remote service execution
cbsData	object	contains cbs entries	
cbsType	string	Enum: • BRAKE_PADS_FRONT • BRAKE_PADS_REAR • BRAKE_FLUID • VEHICLE_CHECK • VEHICLE_TUV • EMISSION_CHECK • OIL	
cbsState	string	Enum: • OK • PENDING • OVERDUE	This maps to green, yellow, red status

cbsRemainingMileage	int	range until cbs is due in kilometer	may be missing if service is only time-dependent
cbsDueDate	string	date when service is due	Format: yyyy-MM may be missing
cbsDescription	string	Description of the CBS value, to be shown to the user.	Must be in the correct language, based on the Language http header
checkControlMessages	array		
ccmId	int	identifier of the Check Control message	
ccmMileage	int	mileage when this CCM was recorded, in kilometers	
ccmDescriptionShort	string	Description of the CCM	Must be in the correct language, based on the Language http header
ccmDescriptionLong	string	Description of the CCM	Must be in the correct language, based on the Language http header
mileage	int	mileage of the vehicle in kilometer	
remainingFuel	float	remaining fuel in liters	optional, only for vehicles having a gas tank
maxFuel	float	Maximum fuel capacity in liter	Optional, only if vehicle has fuel
remainingRangeFuel	int	remaining range in kilometer for fuel drive	optional, only for vehicles which run (partly) with gas
remainingRangeFuelMls	int	remaining range in miles for fuel drive	optional, only for vehicles which run (partly) with gas
maxRangeFuel	int	Maximum range in kilometer for fuel drive	optional, only for vehicles which run (partly) with gas
maxRangeFuelMls	int	Maximum range in miles for fuel drive	optional, only for vehicles which run (partly) with gas
remainingRangeElectric	int	remaining range in kilometer for electric drive	optional, only for vehicles running with electric drive
remainingRangeElectricMls	int	remaining range in miles for electric drive	optional, only for vehicles running with electric drive
maxRangeElectric	int	Maximum predicted range if battery is fully loaded; in kilometers	Optional, only for vehicles with batteries
maxRangeElectricMls	int	Maximum predicted range if battery is fully loaded; in miles	Optional, only for vehicles with batteries

connectionStatus	string	Enum: • DISCONNECTED • CONNECTED • ERROR	optional, only for electric vehicles
chargingStatus	string	Enum: • CHARGING • WAITING_FOR_CHARGING • FINISHED_FULLY_CHARGED • NOT_CHARGING • FINISHED_NOT_FULL • ERROR • INVALID	optional, only for electric vehicles; INVALID in case the vehicle is not connected
chargingTimeRemaining	int	remaining charging time in minutes	optional, only for electric vehicles
chargingLevelHv	int	charging level high-voltage in percent	optional, only for electric vehicles
lastChargingEndReason	string	Enum: • UNKNOWN • CHARGING_GOAL_REACHED • END_REQUESTED_BY_DRIVER • CONNECTOR_REMOVED • POWERGRID_FAILED • HV_SYSTEM_FAILURE • CHARGING_STATION_FAILURE • PARKING_LOCK_FAILED • NO_PARKING_LOCK	optional, only for electric vehicles, only valid if chargingStatus is NOT_CHARGING. May be missing for electric vehicles if chargingStatus is different to NOT_CHARGING
lastChargingEndResult	string	Enum: • SUCCESS • FAILED • UNKNOWN	
updateTime	Date	Time this data set has been retrieved by the vehicle	Format: yyyy-MM-dd’T’HH:mm:ssTZD
updateReason	string	Enum • CHARGING_STARTED • PREDICTION_UPDATE • TEMPORARY_POWER_SUPPLY_FAILURE • CHARGING_DONE • CHARGING_INTERRUPTED • CHARGING_PAUSED • VEHICLE_SHUTDOWN • ON_DEMAND • NO_LSC_TRIGGER • NO_CYCLIC_RECHARGING • CYCLIC_RECHARGING • DOOR_STATE_CHANGED • VEHICLE_SECURED	Enum contains all lsc trigger types known within Q4/13 release (CR 23053)

		• VEHICLE_SHUTDOWN_SECURED • VEHICLE_MOVING • VEHICLE_UNSECURED • UNKNOWN	
remainingEcoRangeElectric			
batteryTemperature	Int	Temperature in °C	
insideTemperature	Int	Temperature in °C	
outsideTemperature	int	Temperature in °C	
rearWindow	string	Enum: • OPEN • CLOSED • INVALID	optional (only if rearWindow exists)
convertibleRoofState	string	Enum: • CLOSEDSECURED • INTERMEDIATE • OPENSECURED • LOADINGPOSITION • HARDTOPMOUNTED • CLOSED • OPEN • EMERGENCYLOCKED • LOADINGPOSITIONINTERMEDIATE • INVALID	optional (only if convertible exists)
publicKey	String	PublicKey from ATM. Required to encrypt the user's key for a Remote360 Remote Service execution (encryptedPackageKey)	Optional; [HEX]

Example:

JSON (I01)

```
{
      "vehicleStatus": {
            "vin": "0000000000H007745",
            "mileage": 15553,
            "updateReason": "VEHICLE_SHUTDOWN",
            "updateTime": "2013-05-21T11:20:01+0200",
            "doorDriverFront": "CLOSED",
            "doorDriverRear": "CLOSED",
            "doorPassengerFront": "CLOSED",
            "doorPassengerRear": "CLOSED",
            "windowDriverFront": "CLOSED",
            "windowDriverRear": "CLOSED",
            "windowPassengerFront": "CLOSED",
            "windowPassengerRear": "CLOSED",
            "sunroof": "INTERMEDIATE_TILT",
            "trunk": "CLOSED",
            "hood": "CLOSED",
            "doorLockState": "LOCKED",
            "parkingLight": "OFF",
            "positionLight": "OFF",
            "remainingFuel": 8,
            "remainingRangeElectric": 95,
            "remainingRangeElectricMls": 59,
            "remainingRangeFuel": 97,
            "remainingRangeFuelMls": 60,
            "maxRangeElectric": 118,
            "maxRangeElectricMls": 73,
            "maxFuel": 9,
            "connectionStatus": "CONNECTED",
            "chargingStatus": "FINISHED_NOT_FULL",
            "chargingLevelHv": 81,
            “chargingTimeRemaining”:168,
            "lastChargingEndReason": "CHARGING_GOAL_REACHED",
            "lastChargingEndResult": "SUCCESS",
            "position":    {
                  "lat": 48.177376,
                  "lon": 11.557335,
                  "heading": 180,
                  "status": "OK"
            },
            "checkControlMessages": [
                  {
                        “ccmId”:1,
                        “ccmMileage”:12345,


                        “ccmDescriptionShort”: “Bitte Bremsen prüfen”,

                        “ccmDescriptionLong”: ”Bitte Bremsen prüfen…”
                  },
                  ...,
                  {
                        “ccmId”:23,
                        “ccmMileage”:32343,
                        “ccmDescriptionShort”:”Lenkung defekt”
      }
            ],
,
            "cbsData": [   {
                  "cbsType": "VEHICLE_CHECK",
                  "cbsState": "PENDING",
                  "cbsRemainingMileage": 5000,
                  "cbsDueDate": "2013-8",
                  "cbsDescription": "Sichtprüfung demnächst fällig…"
            }]
      }
}
```

JSON (I12 / PHEV)

```
{
      "vehicleStatus": {
            "vin": "0000000000H007746",
            "mileage": 31286,
            "updateReason": "CHARGING_DONE",
            "updateTime": "2013-12-02T13:45:15+0100",
            "doorDriverFront": "OPEN",
            "doorDriverRear": "CLOSED",
            "doorPassengerFront": "CLOSED",
            "doorPassengerRear": "CLOSED",
            "windowDriverFront": "OPEN",
            "windowDriverRear": "CLOSED",
            "windowPassengerFront": "CLOSED",
            "windowPassengerRear": "CLOSED",
            "sunroof": "INVALID",
            "trunk": "CLOSED",
            "hood": "CLOSED",
            "doorLockState": "UNLOCKED",
            "parkingLight": "OFF",
            "positionLight": "OFF",
            "remainingFuel": 25,
            "remainingRangeElectric": 24,
            "remainingRangeElectricMls": 15,
            "remainingRangeFuel": 350,
            "remainingRangeFuelMls": 217,
            "maxRangeElectric": 81,
            "maxRangeElectricMls": 50,
            "maxFuel": 30,
            "connectionStatus": "CONNECTED",
            "chargingStatus": "FINISHED_FULLY_CHARGED",
            "chargingLevelHv": 100,
            “lastChargingEndResult": "FAILED",
            "position":    {
                  "lat": 48.177376,
                  "lon": 11.557335,
                  "heading": 89,
                  "status": "OK"
            },
            "checkControlMessages":    [
                  {
                  "ccmId": 586,
                  "ccmMileage": 31280,


                  "ccmDescriptionLong": "Motor nicht abstellen. Nach …",


                  "ccmDescriptionShort": "Antrieb. Kein Neustart"


                  },


                  {


                  "ccmId": 49,


                  "ccmMileage": 31272,

                  "ccmDescriptionLong": "Fahren weiterhin möglich. Von
…",
                  "ccmDescriptionShort": "Antrieb. Demnächst prüfen"
                  }
            ],
            “cbsData":    [
            {
                  "cbsType": "VEHICLE_TUV",
                  "cbsState": "OK",
                  "cbsDueDate": "2021-11",
                  "cbsDescription": "Nächste gesetzliche Fahrzeug…"
            },
            {
                  "cbsType": "BRAKE_FLUID",
                  "cbsState": "OVERDUE",
                  "cbsDueDate": "2001-12",
                  "cbsDescription": "Wechseltermin überschritten! Bitte…"
            }
            ]
      }
}
```

JSON (Remote360)

```
{
      "vehicleStatus": {
            "vin”: "WBA...H004546",
            "mileage": 15553,
            "updateTime": "2014-11-15T14:33:20+0000",
            "updateReason": "VEHICLE_SHUTDOWN",
            "position": {
                  "lat": 48.1234,
                  "lon": 12.14654,
                  "heading": 180,
                  "status": "OK",
            },
            ...,
            "publicKey": "XXXXX123432AAAAA",
      }
}
```

CBS values are retrieved from the vehicle based on their ID. Not all CBS messages are of relevance for the customer. The following mapping provides the information about which CBS ID is mapped to which CBS ENUM type. All other IDs shall be dropped.

CBS ID Mapping:

CBD-ID (int)	CBS-Enum Value
2	BRAKE_PADS_FRONT
6	BRAKE_PADS_REAR
3	BRAKE_FLUID
100	VEHICLE_CHECK
32	VEHICLE_TUEV
33	EMISSION_CHECK
1	OIL
17	VEHICLE_CHECK

JSON (Combustion Vehicle Status with LSC)

```
{
      vehicleStatus: {
            “updateTime”:”2012-06-15T14:33:20+0000”,
            “updateReason”:”VEHICLE_SHUTDOWN”,
            “vin”:” WBA3A51020FZ99561”,
            "mileage": 31286,
            "doorDriverFront": "OPEN",
            "doorDriverRear": "CLOSED",
            "doorPassengerFront": "CLOSED",
            "doorPassengerRear": "CLOSED",
            "windowDriverFront": "OPEN",
            "windowDriverRear": "CLOSED",
            "windowPassengerFront": "CLOSED",
            "windowPassengerRear": "CLOSED",
            "sunroof": "CLOSED",
            "trunk": "CLOSED",
            "hood": "CLOSED",
            "doorLockState": "SECURED",
            "parkingLight": "OFF",
            "positionLight": "OFF",
            "remainingFuel": 25,
            "maxFuel": 30,
            "position":
            {
                  "lat": 48.177376,
                  "lon": 11.557335,
                  "heading": 89,
                  "status": "OK"
            }
      }
}
```

JSON (Legacy Combustion Vehicle Status for vehicle within 1.5km range or tracking enabled)

```
{
      vehicleStatus: {
            “updateTime”:”2012-06-15T14:33:20+0000”,
            “updateReason”:”VEHICLE_FINDER”,
            “vin”:” WBA3A51020FZ99561”,
            “position”: {
                  “lat”:48.1234,
                  “lon”:12.14654,
                  “status”:”OK”,
            },
      }
}
```

JSON (Legacy Combustion Vehicle Status for vehicle outside 1.5km range)

```
{
      vehicleStatus: {
            “updateTime”:”2012-06-15T14:33:20+0000”,
            “updateReason”:”VEHICLE_FINDER”,
            “vin”:” WBA3A51020FZ99561”,
            “position”: {
                  “status”:”TOO_FAR_AWAY”,
            },
      }
}
```

JSON (ActiveE Vehicles)

```
{
      vehicleStatus: {
            “updateTime”:”2012-06-15T14:33:20+0000”,
            “vin”:”WBA...H004546”,
            “position”: {
                  “status”:”VEHICLE_ACTIVE”,
            },
            “mileage”:8534,
            “remainingRangeElectric”:114,
            “remainingEcoRangeElectric”:114,
            “maxRangeElectric”:160,
            “connectionStatus”:”CONNECTED”,
            “chargingStatus”:”CHARGING”,
            “chargingTimeRemaining”:168,
            “chargingLevelHv”:82,
            “batteryTemperature”:20,
            “insideTemperature”:22,
            “outsideTemperature”:15
      }
}
```

REST Methods

Get vehicle status

```
GET /webapi/v1/user/vehicles/<vehicle_id>/status HTTP/1.1
```

This request returns the current vehicle status as stored in the backend. Device location must be transmitted for the distance check (whether location may be displayed or not). If the vehicle does not require the check (because service can be disabled within the vehicle, like for MCV), parameters can be neglected.

Parameter-Specification for the vehicle status request

Name	Type	Description	Condition
dlat	double	device latitude	Optional; required for combustion vehicles and ActiveE, if vehicle tracking cannot be disabled in the vehicle
dlon	double	device longitude	Optional; required for combustion vehicles and ActiveE, if vehicle tracking cannot be disabled in the vehicle
deviceTime	string	Time information for the device	Optional; Required for BMWi vehicles with charging / range prognosis. No timezone information given as local device time is needed. Format: yyyy-MM-dd’T’HH:mm:ss

Response contains the status data as JSON encoded.

HTTP status codes:

- 200 OK
- 400 BAD REQUEST if mandatory parameters are missing or invalid

- 404 NOT FOUND in case no status data is available for this vehicle. This is the case if no LSC has ever been sent to the backend (new vehicle) or for combustion cars, no vehicle finder has ever been executed

Depending upon the vehicle, the status query must perform different checks and call different systems. The following description shall help to understand the API behavior:

- Vehicle is of brand BMW_I
  - Do **not** check device lat/lon
  - Call CCG to retrieve status data
  - Use vehicle position from CSI
- Vehicle is of brand BMW
  - Vehicle has subservice LSC_PHEV or LSC_BASIS
    - Do not check device lat/lon
    - Call CCG to retrieve status data
    - Use vehicle position from CSI
    - Check the mobileSupport flag in CSI
  - Vehicle has PU newer than 07/14 AND t.b.d.
    - Do **not** check device lat/lon
    - Check mobileSupport flag in CSI
    - Use vehicle position from RemoteService status store

Update T2/2014 :

```
IF   ( Vehicle has subservice LSC_PHEV or LSC_BASIS or I_LSC_IMM )
{
     Do not check device lat/lon
     Call CCG to retrieve status data (if updateTime !=NULL)
     Use vehicle position from CSI
}
ELSE
{
wie bisher
}
```

3.3.5 Resource Charging Profile

This section describes all data fields and operations of a charging profile. The data must be filled following the supported charging control type (single timer, two times timer, weekly planner), as specified in the vehicle.

Attribute-Specification

Name	Type	Description	Condition
climatizationEnabled	boolean	flag to enable / disable preconditioning for all set timers	global flag for all timers
timer	object	a schedule timer	
timer1	object	Timer 1	optional.
timer2	object	Timer 2	optional.
timer3	object	Timer 3	optional.
overrideTimer	object	The override Timer	optional, does not support weekdays
timerEnabled	boolean	Indicates whether this timer is currently set active	true/false
departureTime	string	selected departure time	hh:mm in 24 hour format
weekdays	array	Enum • MONDAY • TUESDAY	String Array of weekdays as defined in the Enum. Each weekday

		- WEDNESDAY - THURSDAY - FRIDAY - SATURDAY - SUNDAY	contained counts as active for this timer, all others as inactive
preferredChargingWindow	object	defines a time window where charging should be preferred for cost-efficient charging	
enabled	boolean		defines if the preferred charging window should be used or not
startTime	string	defines the start time of the charging window	hh:mm in 24 hour format
endTime	string	defines the end time of the charging window	hh:mm in 24 hour format
chargingMode	string	Enum: - IMMEDIATE_CHARGING - DELAYED_CHARGING	Optional since T3/15 for Smart Charging: DELAYED_CHARGING enables the selected “chargingPreferences”.
chargingPreferences	string	Enum: - CHARGING_WINDOW - SMART_CHARGING - NO_PRESELECTION	Optional since T3/15 for Smart Charging in combination with chargingMode: Selected chargingPreferences is active if chargingMode=DELAYED_CHARGING. NO_PRESELECTION is an initial delivery condition which means neither CHARGING_WINDOW nor SMART_CHARGING is selected in the HMI.

Examples:

BMW I Vehicles support the weekly planner which allows a detailed description of the departure behavior, including weekdays. This profile can be sent whenever the chargingControl attribute of the vehicle is set to WEEKLY_PLANNER.

JSON (weekly planner)

```
{
      "weeklyPlanner": {
            "chargingMode":"DELAYED_CHARGING",
            "chargingPreferences":"CHARGING_WINDOW",
            "climatizationEnabled":true,
            "timer1": {
                  "departureTime":"15:33",
                  "timerEnabled":true,
                  "weekdays": ["MONDAY", "TUESDAY", "SUNDAY"]
            },
            "timer2": {
                  "departureTime":"07:15",
                  "timerEnabled":true,
                  "weekdays": ["MONDAY", "TUESDAY", "FRIDAY"]
            },
            "timer3": {
                  "departureTime":"12:00",
                  "timerEnabled":false,
                  "weekdays": []
            },
            "CHARGING_WINDOWoverrideTimer": {
                  "departureTime":"15:45",
                  "timerEnabled":true
            },
            "preferredChargingWindow": {
                  "enabled":true,
                  "startTime":"22:00",
                  "endTime":"04:00"
            }
      }
}
```

BMW PHEV vehicles support the departure timer concept, but with a reduced function set for the timers. Only two timers can be sent and no weekdays can be set. This structure can be sent whenever a vehicle has the value “TWO_TIMES_TIMER” set as the chargingControl attribute.

JSON (PHEV DepartureTimer)

```
{
      “departureTimer”: {
            “timer1”: {
                  “departureTime”:”15:33”,
                  “timerEnabled”:true,
            },
            “timer2”: {
                  “departureTime”:”07:15”,
                  “timerEnabled”:true,
            },
            “climatizationEnabled”:true,
            “preferredChargingWindow”: {
                  “enabled”:true,
                  “startTime”:”22:00”,
                  “endTime”:”04:00”
            }
      }
}
```

REST Methods

Get charging profile

```
GET /webapi/v1/user/vehicles/<vehicle_id>/chargingprofile HTTP/1.1
```

This request returns the currently set charging profile of the vehicle. This call may not be available for all vehicles.

Response contains the charging profile data JSON encoded.

HTTP status codes:

- 200 OK
- 404 NOT FOUND in case no charging profile is set for the vehicle

Create or Update charging profile

As the charging profile must be sent to the vehicle, a remote service must be used. This is executed asynchronously. Consequently, the profile cannot be updated with a POST or PUT request, because no instant feedback can be given if the update succeeded or not.

In case of a transient state (new charging profile sent to the backend, but not acknowledged), the behaviour must be specified. For now, the app has to keep track of the status, store the transient profile, disable further service execution and wait for the backend result. In case of an ACK it has to reload the charging profile from the backend.

3.3.6 Resource RangeMap

This section describes all data fields of a range spider

Attribute-Specification

Name	Type	Description	Condition
lat	double	latitude	
lon	double	longitude	
center	object	center coordinate for this range spider	
quality	string	Enum: NO_SERVICE BAD AVERAGE GOOD	In case of NO_SERVICE, the range spider calculation is not available for the requested location
rangemaps	array	container for different range maps	
rangemap	object	containing the range spider coordinates	
type	string	Enum: • ECO_PRO_PLUS • ECO_PRO • COMFORT • SPORT	the driving mode this range spider is valid for
polyline	array	contains the sorted list of coordinates for the range map	Optional. If we have detailed information, a polyline is returned.
radius	double	The radius for a reduced range spider	Optional. If the correct range spider cannot be calculated because LSC update is needed, a circle is returned.

Example:

JSON (detailed data)

```
{
      “rangemap”: {
            “quality”: ”GOOD”,
            “center”: {
                  “lat”: 48.12345,
                  “lon”: 11.43432
            },
            “rangemaps”: [
                  {
                        “type”: ”ECO_PRO_PLUS”,
                        “polyline”: [
                              {
                                    “lat”: 48.123,
                                    “lon”: 11.123,
                              },
                              ...
                              {
                                    “lat”: 48.012,
                                    “lon”: 10.935,
                              },
                        ],
                  },
                  {
                        “type”: ”COMFORT”,
                        “polyline”: [
                              ...
                        ],
                  }
            ]
      }
}
```

JSON (radius only)

```
{
      “rangemap”: {
            “quality”:”BAD”,
            “center”: {
                  “lat”:48.12345,
                  “lon”:11.43432
            },
            “rangemaps”: [
                  {
                        “type”:”ECO_PRO_PLUS”,
                        “radius”:5.75
                  },
                  {
                        “type”:”COMFORT”,
                        “radius”:4.86
                  }
            ]
      }
}
```

REST Methods

Get vehicle range spider

```
GET /webapi/v1/user/vehicles/<vehicle_id>/rangemap HTTP/1.1
```

This request results in generation and delivery of a range spider for the current vehicle status. This includes driving mode, charging state, location

Parameter-Specification for the vehicle range spider request:

Name	Type	Description	Condition

deviceTime	string	Time information for the device	Optional; Required for BMWi vehicles with charging prognosis. Format: yyyy-MM-ddTHH:mm:ss Example: 2013-01-17T16:00:00 Please note: no time-zone information is given

Example request:
https://b2vapi.bmwgroup.us/webapi/v1/user/vehicles/0000000000H006657/rangemap?deviceTime=2013-01-17T16:00:00

Response contains the range spider data JSON encoded.

HTTP status codes:

- 200 OK
- 404 NOT FOUND if the vehicle does not exist
- 424 FAILED_DEPENDENCY in case vehicle tracking is off and/or no vehicle position exists in the backend. The corresponding error must be returned in more detailed error codes as a JSON response

In case of errors, a corresponding error object must be returned explaining why the range spider could not be created. If the content provider fails to return the range spider, the fallback version with a radius only shall be returned, based upon the vehicle location and the remaining mileage. If this is not possible, an error must be returned.
For vehicles where a range calculation is not possible (combustion cars), a 404 NOT FOUND shall be returned.

3.3.7 Resource ChargingStation

This section describes all data fields of a ChargingStation POI. It is an extension to the POI data to make sure it can be used as a POI.

Resource ChargingStation is returned by the following methods specified in this document:

- chargingstation/search
- chargingstation/<chargingstation_id>
- chargingstation/dynamicdata
- chargingstation/searchstatic

The attribute authenticationMethods was changed for the BMW i Remote App V 1.3 to provide the functionality to directly switch into the Hubject App (if available). The authenticationMethod ID is needed for this Use Case.

However the existing methods

- chargingstation/search
- chargingstation/<chargingstation_id>

(up to App Vesion V 1.2) are not affected by this change and will still return a String array.

This approach is working as the logic how the BMW i Remote App requests is changed from V 1.2 to V 1.3. From V 1.3 upwards the App is using the methods

- chargingstation/dynamicdata
- chargingstation/searchstatic

instead to request charging station information.

Attribute-Specification

Name	**Type**	**Description**	**Condition**
id	string	Unique ID of this POI	mandatory
name	string	Name of POI	mandatory
street	string	Street of POI including streetnumber	optional
city	string	City of POI	optional
postalCode	string	Postal code of POI	optional
county	string	County of POI	optional
country	string	Country of POI	optional
phone	string	Phone of POI	optional
email	string	contact email	optional, valid email-address
website	string	web address	optional, valid URL
lat	double	POI latitude	mandatory
lon	double	POI longitude	mandatory
connectors	object		optional
type	string	Enum: • HOME_PLUG • CHADEMO • TYPE_1 • TYPE_2 • TYPE_3 • COMBO_1 • COMBO_2 • TYPE_CN • CEE	optional
totalCount	int	Number of available connectors of a certain type	optional
availableCount	Int	Number of available connectors of a certain type	optional
availability	string	Enum: • UNKNOWN • AVAILABLE • CRITICAL • OCCUPIED	optional critical availability is identical to the definition of status “yellow” for onboard charging stations.
operator	string	The operator of this charger	optional
provider	string	The data provider, e.g. Inrix	Optional
openingHours	string	Opening hours of this charger. Localized based on the locales in Accept-Language header	optional
totalConnectors	int	The number of connectors for this charging station	optional
availableConnectors	int	Number of free connectors	optional
access	string	Enum • PUBLIC • PRIVATE • COMPANY	optional

		• UNKNOWN	
additionalInfo	string	Free text info field for a charger	Optional
preferredPartner	boolean	Indicator if the charging station is a preferred station of not	optional
preferredPartnerUrl	string	URL where more information about the preferred partner program can be retrieved	optional
location	string	Enum • INDOOR • OUTDOOR • UNKNOWN	Optional
freeCharge	boolean	Indicates if free charge is available	Optional
open24h	boolean	Indicates if the charging station is open 24/7	Optional
paymentMethods	String array	List of payment methods supported	Optional
authenticationMethods	String array	**Up to version 1.2 of the BMW i Remote App:** List of authentication methods supported (Only translations) Returned by methods: - chargingstation/search - chargingstation/<chargingstation_id>	Optional
	Object array	**BMW i Remote App 1.3 and all other Apps:** List of authentication methods supported (IDs and translations) Returned by methods: - chargingstation/dynamicdata - chargingstation/searchstatic	
greenEnergy	boolean	If the charging station provides green energy, this value shall be set to true	Optional
serviceType	string	Enum: • SELF_SERVICE • VALET • UNKNOWN	Optional

Example JSON

```
{
      “chargingstation”: {
            “id”:”charger_123”,
            “name”:”BMW FIZ”,
            “street”:”BMW Allee 2”,
            “postalCode”:”80807”,
            “city”:”München”,
            “county”:”Bayern”,
            “country”:”Deutschland”,
            “email”:”strom@swm.de”,
            “website”:”http://www.swm.de”,
            “lat”:48.193,
            “lon”:11.570,
            “operator”:”SWM”,
            “provider”: “INRIX”,
            “additionalInfo”: “This is how you have to charge...”,
            “availability”:”AVAILABLE”,
            “totalConnectors”:9,
            “availableConnectors”:3,
            “open24h”: false,
            “openingHours”: ”Mo-Fr: 08:00 - 18:00, Sa-So: 10:00 – 15:00”,
            “freeCharge”: false,
            “access”: ”PUBLIC”,
            “location”: “OUTDOOR”,
            “preferredPartner”: true,
            “serviceType”: “SELF_SERVICE”,
            “paymentMethods”: [“American Express”, “VISA”],

//Up to BMW i Remote App V 1.2:
            “authenticationMethods”: [“RFID”, “ChargeNow”],

//From BMW i Remote App V 1.3 upwards:
            “authenticationMethods”: [
                  {id:"RFID", label:"RFID"},
                  {id:"CHARGE_NOW", label: "ChargeNow"}],


            “greenEnergy”: true,
            “connectors”: [
                  {
                        “type”: ”TYPE_2”, “label”:”xyz”


                        “totalCount”: 10,

                        “availableCount”: 5
                  },
                  {
                        “type”:”COMBO_1”,


                        “totalCount”: 10,

                        “availableCount”: 5
                  }

            ],
      }
}
```

REST Methods

Search ChargingStations

```
GET /webapi/v1/chargingstations/search HTTP/1.1
```

This request queries the backend for POIs following the specified search criteria. The search is based upon lat/lon with radius (all three must be provided) and an optional limit

Request-Parameter-Specification

Name	Type	Description	Condition
lat	double	Search center latitude	optional
lon	double	Search center longitude	optional

radius	double	Search radius in km	Required for lat/lon search
limit	int	Maximum number of results to be returned	Optional. If not specified, the server will impose a limit
freeCharge	boolean	Filter on charging stations which are free of charge	Optional true == only free false == all (free and paid)
open24h	boolean	Filter on charging stations which are open 24/7	Optional true == opened 24/7 false == all chargers
location	string	Enum: • INDOOR • OUTDOOR	Optional, may contain any combination of the allowed values, comma separated
access	string	Enum: • PUBLIC • RESTRICTED	Optional
preferredPartner	boolean		Optional true == only preferred partner stations false == all chargers
operator	string	The operator to search for. Must be retrieved by a separate filter data request	Optional
provider	string	The data provider, e.g. Inrix	Optional
plugType	string	Enum: • HOME_PLUG • CHADEMO • TYPE_1 • TYPE_2 • TYPE_3 • COMBO_1 • COMBO_2 • TYPE_CN • CEE	Optional. Values should be limited to the ones returned by the separate filter data request in order to simplify the user decision
authenticationMethod	string	The authentication methods to filter for, as pipe separated list. Search will be OR based. Values must be retrieved by a separate filter data request	optional
paymentMethod	string	The payment methods to filter for, as a pipe separated list. Search will be OR based. Values must be retrieved by a separate filter data request	Optional

Response contains a list of JSON encoded ChargingStation POIs. List may be empty. Example:

JSON

```
{
      “chargingstations”: [
            {
```

```
            “id”:”charger_123”,
            “name”:”BMW FIZ”,
            ...
        },
        {
            ...
        }

    ]
}
```

HTTP status codes:

- 200 OK

Search ChargingStations - static data only

```
GET /webapi/v1/chargingstations/searchstatic HTTP/1.1
```

Request-Parameter-Specification

Name	**Type**	**Description**	**Condition**
Lat	double	Search center latitude	optional
Lon	double	Search center longitude	optional
Radius	double	Search radius in km	Required for lat/lon search
Limit	int	Maximum number of results to be returned	Optional. If not specified, the server will impose a limit
freeCharge	boolean	Filter on charging stations which are free of charge	Optional true == only free false == all (free and paid)
open24h	boolean	Filter on charging stations which are open 24/7	Optional true == opened 24/7 false == all chargers
Location	string	Enum: • INDOOR • OUTDOOR	Optional, may contain any combination of the allowed values, comma separated
Access	string	Enum: • PUBLIC • RESTRICTED	Optional
preferredPartner	boolean		Optional true == only preferred partner stations

			false == all chargers
operator	string	The operator to search for. Must be retrieved by a separate filter data request	Optional
provider	string	The data provider, e.g. Inrix	Optional
supportedChargingModes	enum array	List containing subset of the values "AC_LOW", "AC_HIGH", "DC"	optional
plugType	string	Enum: • HOME_PLUG • CHADEMO • TYPE_1 • TYPE_2 • COMBO_1 • COMBO_2 • TYPE_CN • TYPE_3 • CEE	Optional. Values should be limited to the ones returned by the separate filter data request in order to simplify the user decision
authenticationMethod	string	The authentication methods to filter for, as pipe separated list. Search will be OR based. Values must be retrieved by a separate filter data request	optional
paymentMethod	string	The payment methods to filter for, as a pipe separated list. Search will be OR based. Values must be retrieved by a separate filter data request	optional

Response contains a list of JSON encoded ChargingStation POIs. List may be empty. Example:

JSON

```
{
      “chargingstations”: [
            {
                  “id”:”charger_123”,
                  “name”:”BMW FIZ”,
                  “street”:”BMW Allee 2”,
                  “postalCode”:”80807”,
                  “city”:”München”,
                  “lat”:48.193,
                  “lon”:11.570
            },
            {
                  ...
            }

      ]
}
```

HTTP status codes:

- 200 OK

Get ChargingStation Status

```
GET /webapi/v1/chargingstations/<chargingstation_id> HTTP/1.1
```

This request queries the backend to return a single charging station. This call is used to get the dynamic status for a single, dedicated station (out of the favorites).

Response data:

JSON

```
{
      “chargingstation”: {
            “id”:”charger_123”,
            “name”:”BMW FIZ”,
            ...
      }
}
```

HTTP status codes:

- 200 OK
- 404 NOT FOUND in case the charging station ID is unknown

Get ChargingStation Filter Data

```
GET /webapi/v1/chargingstations/filter HTTP/1.1
```

This request queries the backend to return filter data which can be used to fill the request parameters of a charging station search request.

Parameter-Specification for the charging station filter request:

Name	Type	Description	Condition
Lat	double	Search center latitude	optional
Lon	double	Search center longitude	optional
Radius	double	Search radius in km	Required for lat/lon search
supported-ChargingModes	enum array	List containing subset of the values "AC_LOW", "AC_HIGH", "DC"	optional

Response data:

JSON

```
{
      “chargingstationFilter”: {
            “operators”:[“BMW”, “e.on Energie”, “SWM”],
            “plugTypes”: [”HOME_PLUG”, “TYPE_2”],
            “authenticationMethods”: [“ChargeNow”, “RFID”, “Phone”],
            “paymentMethods”: [“ChargeNow”, “Account”],
            “providers”: [“Hubject”, “Inrix”]
      }
}
```

HTTP status codes:

- 200 OK

Get ChargingStations Status (List)

```
GET /webapi/v1/chargingstations/dynamicdata HTTP/1.1
```

This request queries the backend to return multiple charging stations. This call is used to get the dynamic status for charging stations (multiple, when zooming in on the map / single ones if you choose a charging station from favourites).

Request-Parameter-Specification

Name	Type	Description	Condition
ids	String	List of Unique charging station IDs (comma separated)	Mandatory

Response data:

JSON

```
{
      “chargingstations”: [
            {
                  “id”:”charger_123”,
                  “name”:”BMW FIZ”,
                  ...
            },
            {
                  ...
            }

      ]
}
```

HTTP status codes:

- 200 OK (Result list can be empty)

3.3.8 Resource StatisticsData

This section describes all data fields and operations of efficiency data. Only the last trip and aggregated data is available in the backend. More detailed data cannot be retrieved through the API.

Attribute-Specification Last Trip

Name	Type	Description	Condition

lastTrip	object	Contains the trip information about the last trip	
totalDistance	double	Trip distance in km	
electricDistance	double	Part of the total distance which has been driven with electric power in km	
avgElectricConsumption	double	Energy used for the last trip in kWh/100km	
avgCombinedConsumption	double	Combined consumption value in l/100km	
avgRecuperation	double	Energy recuperated during the last trip in kWh/100km	
efficiencyValue	double	Synthetic efficiency value	Between 0 and 1, identifying the efficiency in percent
totalConsumptionValue	double	Efficiency rating for electric consumption	Between 0 and 1
auxiliaryConsumptionValue	double	Efficiency rating for auxiliary power usage	Between 0 and 1
drivingModeValue	double	Efficiency rating for driving mode	Between 0 and 1
accelerationValue	double	Efficiency rating for acceleration	Between 0 and 1
anticipationValue	double	Efficiency rating for anticipation	Between 0 and 1
Date	string	Date when the last trip data has been collected	Date format: 2012-07-31T15:13:33+0000
electricDistanceRatio	double	Ratio of electric distance out of total distance	Between 0 and 1; NOT a “star” value; PHEV only;
savedFuel	double	Saved fuel due to electric driven distance compared to fuel-only	NOT YET AVAILABLE (T3 2014)
Duration	int	Duration of the last trip in minutes	T3 2014

The underlying backend system uses different fields depending on the vehicle type (I01 or I12). Nevertheless, regarding the API, no different value names are needed. The mapping is as follows:

CCGW value	**WebAPI**
Value A (Electrical overall consumption I01)	totalConsumptionValue
Value B (Auxiliary consumption I01)	auxiliaryConsumptionValue
Value C (driving mode I01)	drivingModeValue
Value D (acceleration value I01 & I12)	accelerationValue
Value E (anticipation value I01 & I12)	anticipationValue
Value F (combined consumption value I12)	totalConsumptionValue
Value G (Auxiliary consumption I12)	auxiliaryConsumptionValue
Value H (driving mode I12)	drivingModeValue
Value I (PHEV charging behavior)	chargingBehaviorValue
Value J (share of electric driven distance)	electricDistanceShareValue
???	electricDistanceRatio (T3 2014)

Attribute-Specification All Trips

Name	Type	Description	Condition
allTrips	object	Contains aggregated data for all trips	
avgElectricConsumption	object	Container element for aggregated consumption data	I01 only
avgRecuperation	object	Container element for aggregated recuperation data	
chargecycleRange	object	Container element for aggregated charge cycle data	I01 only
totalElectricDistance	object	Container element for aggregated distance data	
avgCombinedConsumption	object	Container element for aggregated consumption data	I12 only
communityLow	double	Community low value for respective statistic element	
communityAverage	double	Community avg value for respective statistic element	
communityHigh	double	Community high value for respective statistic element	
userAverage	double	User average for the respective statistic element	
userHigh	double	User high value for respective element	
userCurrentChargeCycle	double	Distance the user did drive during his current charging cycle so far	As this value contains several trips, it is an aggregated value already
savedCO2	double	Value in kg	Saved CO2 if standard energy has been used
savedCO2greenEnergy	double	Value in kg	Saved CO2 if green energy has been used
userTotal	double	Accumulated total of the user	
electricDistanceRatio	object	Container element for aggregated values	Single values between 0 and 1
totalDistance	object	Container element for total driven distance	
totalSavedFuel	object	Container element for total saved fuel	NOT YET AVAILABLE (T3 2014)
resetDate	string	Date of last Reset	NOT YET AVAILABLE (T3 2014) (date format: 2012-07-31T15:13:33+0000)

REST Methods

Get Last Trip Statistics Data

```
GET /webapi/v1/user/vehicles/<vehicle_id>/statistics/lastTrip HTTP/1.1
```

This request returns the statistics overview of the given vehicle. It contains information about the last trip and the overall lifetime performance as JSON.

HTTP status codes:

- 200 OK
- 404 NOT FOUND if the vehicle does not exist

Response contains a JSON encoded statistics object for the last trip. In case no last trip exists, all values will be missing. The container element “lastTrip” must still exist. Example:

JSON (I01)

```
{
      “lastTrip”: {
            “efficiencyValue”:0.68,
            “totalDistance”:55,
            “electricDistance”:45,
            “avgElectricConsumption”: 11.5,
            “avgRecuperation”: 1.3,
            “totalConsumptionValue”:0.27,
            “auxiliaryConsumptionValue”:0.48,
            “drivingModeValue”:0.56,
            “accelerationValue”:0.88,
            “anticipationValue”:0.64,
            "duration": 42,
            “date”:”2012-07-31T13:21:00+0000”
      }
}
```

JSON (I12)

```
{
      “lastTrip”: {
            “efficiencyValue”:0.68,
            “totalDistance”:55,
            “electricDistance”:45,
            “avgElectricConsumption”: 11.5,
            “avgCombinedConsumption”: 3.7,
            “avgRecuperation”: 1.3,
            “totalConsumptionValue”:0.27,
            “auxiliaryConsumptionValue”:0.48,
            “drivingModeValue”:0.56,
            “accelerationValue”:0.88,
            “anticipationValue”:0.64,
            “date”:”2012-07-31T13:21:00+0000”
      }
}
```

Get Community Statistics Data

```
GET /webapi/v1/user/vehicles/<vehicle_id>/statistics/allTrips HTTP/1.1
```

This request returns the statistics overview of the given vehicle. It contains information about the overall lifetime performance as JSON. In case no aggregated values exists, all values will be missing. The container elements must still exist. After a user has reset his values, only the user related items may miss. Community values are always transmitted, if available.

HTTP status codes:

- 200 OK
- 404 NOT FOUND if the vehicle does not exist

Response contains a JSON encoded statistics overview object. Example:

JSON (I01)

```
{
   "allTrips": {
      "avgElectricConsumption":    {
            "communityLow": 18.88,
            "communityAverage": 23.84,
            "communityHigh": 30.61,
            "userAverage": 27.83
      },
      "avgRecuperation":    {
            "communityLow": 2.25,
            "communityAverage": 5.34,
            "communityHigh": 10.75,
            "userAverage": 4.5
      },
      "chargecycleRange":    {
            "communityAverage": 89.53,
            "communityHigh": 170,
            "userAverage": 1,
            "userHigh": 1,
            "userCurrentChargeCycle": 78
      },
      "totalElectricDistance":    {
            "communityLow": 32,
            "communityAverage": 1403.81,
            "communityHigh": 5338.62,
            "userTotal": 64.06
      },
      "savedCO2": 1.776,
      "savedCO2greenEnergy": 10.448
   }
}
```

JSON (I12)

```
{
   "allTrips": {
      "avgRecuperation":    {
            "communityLow": 2.25,
            "communityAverage": 5.34,
            "communityHigh": 10.75,
            "userAverage": 4.5
      },
      "totalElectricDistance":    {
            "communityLow": 32,
            "communityAverage": 1403.81,
            "communityHigh": 5338.62,
            "userTotal": 64.06
      },
      "avgCombinedConsumption":    {
            "communityLow": 0,
            "communityAverage": 1.01,
            "communityHigh": 2.87,
            "userAverage": 20.58
      },
      "savedCO2": 1.776,
      "savedCO2greenEnergy": 10.448
   }
}
```

Reset Personal Statistics Data

```
GET /webapi/v1/user/vehicles/<vehicle_id>/statistics/reset HTTP/1.1
```

This request resets the personal statistics data. It results in all personal values being not available any more, until new data has been collected. The “last trip” call will result in a 404 afterwards. All user specific values for “all trips” shall be missing (not transmitted to the client through the api).

HTTP status codes:

- 204 NO CONTENT in case the request was successful
- 404 NOT FOUND if the vehicle does not exist

3.3.9 Intermodal Routing

This section describes the data format and requests for intermodal route planning. This is used in combination with the last destinations / last mile use-case.

Attribute-Specification Intermodal Route

Name	Type	Description	Condition
Routes	object	Array containing the calculated routes	
Type	string	Enum: • WALK • PUBLIC	Type of route / leg
providerIcon	string	Contains the URL to a provider icon to be displayed within the app (.png)	
Duration	int	Duration of the route in minutes	
Distance	int	Distance of the route in meter	Only available for walking routes
startLocation	object	Start location of a route	Object description see table “location” below
endLocation	object	End location of a route	Object description see table “location” below
Legs	object	Container for all legs of a route	Object description see table “leg” below

Attribute-Specification object “location”

Lat	double	Latitute	

Lon	double	Longitude	
street	string	Street name of an address	
Streetnumber	string	Street number of an address	
City	string	City of an address	
Country	string	Country of an address	
Name	string	The name of a start or end location or leg	
Type	string	Enum: • PARK_AND_RIDE • STATION • ADDRESS	Only to be used for start / end location
Time	string	Time of departure / arrival at the respective location	Format: yyyy-MM-ddTHH:mm:ss

Attribute-Specification object "leg"

Type	string	Enum: • TRAIN • URBAN • SUBWAY • TRAM • BUS • WALK • WALK_LINK	Only with respect to a leg. Indicates the transportation means. WALK_LINK is a walking section between two public transport legs. WALK and WALK_LINK will never follow directly to each other
Duration	int	Duration of the leg in minutes	
Headsign	string	Direction of this leg.	Optional; typically used for public transport legs, indicating the destination stop of the respective line.
startLocation	object	Start location of a leg	Object description see table "location" above
endLocation	object	End location of a leg	Object description see table "location" above
Color	string	Color of the respective leg	As HEX RGB code (HTML style with # upfront)
Line	string	Name of the pubic transport line to be used	
Polyline	object	Contains the polyline for walking routes	
Lat	double	Latitute of a polyline entry	
Lon	double	Longitude of a polyline entry	

Get Intermodal Route

```
GET /webapi/v1/user/vehicles/<vin>/routes/calculate HTTP/1.1
```

This request triggers calculation of an intermodal route from given start to destination.

Request-Parameter-Specification

Name	**Type**	**Description**	**Condition**
startLat	double	Start latitude	mandatory
startLon	double	Start longitude	mandatory
destLat	double	Destination latitude	mandatory
destLon	double	Destination longitude	mandatory
departureTime	string	Time at which departure is planned	Format: yyyy-MM-dd’T’HH:mm:ss Timezone information not needed, as this is always local time. Typicalls, this is filled with the current device time of the client
destStreet	string		optional
destStreetnumber	string		optional
destCity	string		optional
destCountry	string		optional

The requesting address is sent to the backend in order to allow the routing algorithm to provide the user selected destination as its destination. Reverse geocoding based on the GPS position will often result in different output.

HTTP status codes:

- 200 OK in case a route is returned
- 400 BAD PARAMETER in case the request contained missing or invalid parameters
- 412 PRECONDITION FAILED in case intermodal routing is not available for this vehicle / location

If no intermodal routing is available for the given location, the result must contain respective error information in addition to the http error code.

Respone contains a JSON with the intermodal route in the following format:

JSON

```
{
      “routes”: [ {
            “type”:”PUBLIC”,
            “providerIcon”: ”https://csp.bmwgroup.com/bmwi/Provider-
            LogoApp
/X.png”,
            “duration”: 25,
            “startLocation”: {
                  “lat”: 48.123,
                  “lon”: 11.543,
            },
            “endLocation”: {
                  “lat”: 48.1369,
                  “lon”: 11.5754,
                  “street”: “Marienplatz”,
                  “streetnumber”: “23”,
                  “city”: “München”,
                  “country”: “Deutschland”
            },
            “legs”: [
                  {
                        “type”: “SUBWAY”,
                        “line”: “U6”,
                        “color”: “#a31254”,
                        “headsign”: “Harras”,
                        “duration”: 15,
                        “startLocation”: {
                              “name”: “P+R irgendwo”,
                              “type”: “PARK_AND_RIDE”,
                              “lat”: 48.123,
                              “lon”: 11.543,
                              “time”: “2013-01-07T09:32:00”
                        },
                        “endLocation”: {
                              “name”: “Bahnhof B”,
                              “type”: “STATION”,
                              “lat”: 48.123,
                              “lon”: 11.543
                              “time”: “2013-01-07T09:38:00”
                        }
                  },
                  {
                        “type”: “WALK”,
                        “color”: “#ffffff”,
                        “duration”: 5,
                        “startLocation”: {
                              “name”: “Bahnhof B”,
                              “type”: “STATION”,
                              “lat”: 48.123,
                              “lon”: 11.543
                              “time”: “2013-01-07T09:38:00”
                        },
                        “endLocation”: {
                              “name”: “Zielstraße 1”,
                              “type”: “ADDRESS”,
                              “lat”: 48.123,
                              “lon”: 11.543
                              “time”: “2013-01-07T09:38:00”
                        },
                        “polyline”: [
                              {
                                    “lat”:48.123,
                                    “lon”:11.123,
                              },
                              ...
                              {
                                    “lat”:48.012,
                                    “lon”:10.935,
                              }
                        ]
                  }
            ]
      },
      {
            “type”:”WALK”,
            “duration”: 55,
            “distance”: 1020,
            ...
      }]
```

```
}
```

In case the original request contained a destination address, this address shall be identical in the destination section of the JSON response. If no destination address is given, the destination shall be identified by reverse geocoding.

3.3.10 Resource LocalSearch POI

This section describes all data fields of a Search POI. It is an extension to the POI data to make sure it can be used as input for the send to car feature. The content attribution is not necessarily part of each POI, typically it is an attribute valid for the entire result list.

Attribute-Specification

Name	Type	Description	Condition
name	string	Name of POI	mandatory
street	string	Street of POI including streetnumber	optional
city	string	City of POI	optional
postalCode	string	Postal code of POI	optional
county	string	County of POI	optional
country	string	Country of POI	optional
phoneNumbers	object	Phone numbers of POI	optional
email	string	contact email	optional, valid email-address
website	string	web address	optional, valid URL
lat	double	POI latitude	mandatory
lon	double	POI longitude	mandatory
openingHours	string	Opening hours of this charger. Localized based on the locales in Accept-Language header	optional
rating	double	Rating of this POI	optional
attribution	string	The content attribution for the POI result list	Optional; If provided, it must be displayed
formattedAddress	string	A formatted address string containing combined address components as a comma separated string. This reflects local address formatting rules like positioning of street name and street number or postal code.	optional

Example JSON

```
{
      "poi": {
            "name": "BMW Welt",
            "street": "Am Olympiapark 1",
            "city": ”München”,
            “postalCode”: ”80709”,
            “country”: ”Deutschland”,
            “region”: ”Bayern”,
            "formattedAddress": "Am Olympiapark 1, 80709 München, 
Deutschland",
            “website”: ”http://www.bmw-welt.com”,
            “email”: ”info@bmw-welt.com”,
            “lat”: 48.177,
            “lon”: 11.557,
            “rating”: 3.6,
            “phoneNumbers”: [
                  {
                        “number”:”+49 89 12345”,
                        “type”:”UNKNOWN”
                  }
            ]
      }
}
```

REST Methods

Search POIs

```
GET /webapi/v1/pois/search HTTP/1.1
```

This request queries the backend for POIs following the specified search criteria. Search uses a bounding box.

Request-Parameter-Specification

Name	Type	Description	Condition
id	string	List of comma separated POI IDs for direct request from query autosuggestion.	optional
lat	double	Search center latitude	mandatory
lon	double	Search center longitude	mandatory
radius	double	Search radius in km; the radius is used to provide a preferred region, it is not treated as a hard boundary.	mandatory
query	string	Search term	optional (either ‘id’ or ‘query’ must be given)
maxresults	integer	Max number of results returned	Optional; defaults to 20 if not provided
category	string	Search Provider	Optional, if empty CDP selects the provider ‘bikes’ will enforce CallABike (DB)

Response contains a list of JSON encoded POIs. List may be empty. Example:

JSON

```
{
        "pois": [
                {
                        "id":"refkey",
                        "name":"BMW FIZ",
                        ...
                },
                {
                        ...
                }

        ],
        "attribution": "Powered by Google"
}
```

HTTP status codes:

- 200 OK

The online search is until today always powered by Google. With the market Turkey this will change for the first time. For this purpose the TID (UserID/LisboaID) is transferred via HTTP Header Variable 'BMW-User' to CDP.
Because of legal issues we cannot display POIs on a Google Map that are not searched by Google. Instead these POIs will be displayed in a list.
The following logic needs to be implemented:

- No SA 6AK: "onlineSearch" = NOT_SUPPORTED
- SA 6AK && Market Turkey: "onlineSearch" = LIST
- (SA 6AK || SA 612) && All other ConnectedDrive Markets: "onlineSearch" = MAP

RETURN FORMAT FOR CAB RESULTS TBD

Query Autosuggest

This section describes all data fields of a POI Query Autocomplete. It is an extension to the POI data to offer search suggests while the user is entering a search query in the Remote App.

REST Method

```
GET /webapi/v1/pois/queryAutosuggest HTTP/1.1
```

This request queries the backend for query autosuggest following the specified search criteria.

Request-Parameter-Specification:

Name	Type	Description	Condition
lat	double	Search center latitude	Mandatory
lon	double	Search center longitude	Mandatory
radius	double	Search radius in km; the radius is used to provide a preferred region, it is not treated as a hard boundary.	Optional
query	string	Search term for autocomplete	Mandatory

Attribute-Specification:

Name		Type	Description	Condition
meta		object		mandatory
	provider	string	Provider name. e.g. Google or Here	mandatory

	status	string	ok or error status	optional
suggestions		object		optional
	name	string	Suggested keyword or poi	mandatory
	id	string	If the suggestion is a POI, the ID identifies the POI to request the details as described in section 'Search POI'.	optional

Example JSON

```
{
      "meta": {
            "provider":"Google",
            "status":"ok"
      },
      "suggestions": [
      {
            "name": "Hotel in der Nähe von München, Deutschland"
      },
      {
            "name": "Hotel Augusta, Kesselmarkt, Augsburg, Deutschland",
            "id":"ChIJW58tr1G9nkcR9YNC9es0zJw"
      },
      {
            "name": "Hotel Amba, Arnulfstraße, München, Deutschland",
            "id":"ChIJx1ak-_t1nkcR-VR8kUhcRIU"
      }
}
```

HTTP status codes:

- 200 OK

The language and region of the user app is set as the Accept-Language header for each request. The backend system must try to fulfill the users preferences for the Query Autocomplete.

```
Accept-Language: de-DE,de;q=0.8, en;q=0.7
```

In each case where language specific content is sent from the backend, it should respect the transmitted header.

3.3.11 Remote360

3.3.11.1 Remote360 Status

```
GET /webapi/v1/user/vehicles/<vehicle_id>/remote360executionStatus HTTP/1.1
```

Attribute-Specification:

Input Parameters			
Name	**Type**	**Description**	**Condition**
eventId	string	The event ID of the remote service execution	mandatory

startId	int	Do prevent the app to download the whole history every time, you can set an id to start from. If startId is higher than last statusId, the last available statusId will be delivered.	Optional (Potentieller Streichkandidat)

HTTP status codes:

- 200 OK
- 404 NOT FOUND if no vehicle with the given VIN exists for the current user or sessionId is unknown.

Output Parameters		
Name	**Type**	**Description**
eventId	String	The event ID of the remote service execution
executionStatus	String	Enum: IN_PROGRESS, ERROR, OK TERMINATED (See chapter 'Terminate Remote Service') Quelle: NGTP getEventData
availablePackages	Object[]	Array of already available data packages to download
dataPackageId	int	Unique package id Quelle: NGTP getEventData
filenamePackage	String	Filename of the package
statusHistory	Object[]	
statusId	int	Aufsteigender Zähler
timestamp	int	Unix-Timestamp in milliseconds
statusCode	int	Status- / Fehler-Codes laut nachfolgender Tabelle Quelle: NGTP getEventData
mobileNetworkTechnology	String	Enum: UMTS, GRPS, EDGE, LTE Quelle: NGTP getEventData
mobileNetworkRssi	int	Signalqualität (größer = besser) Quelle: NGTP getEventData
playProtectionLockTime	int	Restzeit [in Sekunden] wann Spielschutz für neue Ausführung freigegeben ist. „0", wenn Spielschutz-Limit nicht erreicht.
newPublicKey	String	Base64 Neuer PublicKey, wenn sich der publicKey verändert hat. Mögliche Alternative: Verzicht auf Anzeige im remote360executionStation. Die App müsste dann aber stattdessen im Fehlerfall den vehicleStatus erneut prüfen.

Response example JSON

```
{
"remote360Status": {
      "eventId": "463339333938371BF3667200@bmw.de",
      "executionStatus": "IN_PROGRESS", // ERROR, OK
      "mobileNetworkTechnology": "3G",
      "networkQuality": 5,
      "playProtectionLockTime": "60",
      "newPublicKey": "XXX12345XXX",
      "availablePackages": [
            {"dataPackageId": 51324, "filenamePackage": "left.mp4"},
            {"dataPackageId": 51325, "filenamePackage": "right.mp4"},
            {"dataPackageId": 51326, "filenamePackage": "cdata.xml"}
            ]
            "statusHistory": [
                  {
                        "statusId": 1,
                        "timestamp": 1438934899789,
                        "statusCode": 3004,
                  },
                  {
                        "statusId": 2,
                        "timestamp": 1438934897972,
                        "statusCode": 3012,
                  },
                  ...
            ]
      }
}
```

statusCodes:

Status Code	Nr.	Beschreibung	ENUM Wert ASN.1 Codec	Sender	Notification
1001	1	iCAM nicht aufnahmebereit- interner Fehler.	icamInternalError	iCAM	Alert
1002	2	Die Ausführung der Funktion Remote 360 durch andere Funktion verhindert.	otherServiceRunning	iCAM	Alert
1003	3	Bilder aufgenommen und die Kameras sind verschmutzt	picturesTaken + camPolluted	iCAM	Silent
1004	4	Bilder aufgenommen und die Außenspiegel konnten nicht ausgeklappt werden	picturesTaken + mirrorError	iCAM	Silent
1005	5	Bilder aufgenommen und die Kameras sind verschmutzt und die Außenspiegel konnten nicht ausgeklappt werden	picturesTaken + camPolluted + mirrorError	iCAM	Silent
1006	6	Bilder aufgenommen.	picturesTaken	iCAM	Silent
2001	7	No data connection available (DataConnection is not available/could not be opened)	noIpConnection	ATM	Alert
2002	8	Communication with Backend causes HTTP Timeout or HTTP Error (Error in HTTP Communication with Backend)	communicationError	ATM	Alert
2003	9	Backend returned invalid data (Error in Data returned from Backend)	invalidData	ATM	Alert
2004	10	Bilder aufgenommen und Upload wird gestartet.	startUpload	ATM	Silent
2005	11	Fehler bei der Entschlüsselung des sym. Schlüssel	decryptionError	ATM	Alert
2006	12	Fehler bei der Verschlüsselung eines Datenpakets	encryptionError	ATM	Alert
2007	13	Kameras nicht verfügbar. Keine Antwort von iCAM.	icamNotAvailable	ATM	Alert
3001	15	GPS Ortung im Fahrzeug deaktiviert		Backend	Alert
3002	16	Funktion wird gerade von einem anderen User ausgeführt. Bitte versuchen Sie später.		Backend	Alert
3003	17	Schlüsselpaar ungleich.		Backend	SIlent
3004	18	Spielschutz i.O. („Spielschutzregelung eingehalten“)		Backend	Silent

3005	19	Spielschutz nicht i.O. („Spielschutzregelung nicht eingehalten. Verbliebene Zeit bis zum Entsperren").		Backend	Alert
3006	20	Status Aktivierung Fahrzeug i.O. (R360 im Fahrzeug aktiviert)		Backend	
3007	21	Status Aktivierung Fahrzeug nicht .I.O. (R360 im Fahrzeug nicht aktiviert)		Backend	Alert
3008	22	Mobilfunk-Verbindung mit dem Fahrzeug aufgebaut (SMS angekommen)		Backend	Silent
3009	23	Datenverbindung-Verbindung mit dem Fahrzeug aufgebaut (nach Abholen der DQL Remote 360)		Backend	Silent
3010	24	Länderspezifische Freigabe i.O. (kein Grenzübergang)		Backend	Silent
3011	25	Länderspezifische Freigabe nicht i.O. (Fahrzeug im Ausland)		Backend	Push
3012	26	Paket im Backend hochgeladen (steht zum Download bereit)		Backend	Silent/ Alert
3013	27	Mobilfunk-Verbindung mit dem Fahrzeug nicht verfügbar bzw. abgebrochen (SMS nicht angekommen)		Backend	Alert
3014	28	Keine Datenverbindung-Verbindung mit dem Fahrzeug nicht verfügbar bzw. abgebrochen (IP-Verbindung)		Backend	Alert
3015	29	Verhindern gleichzeitigen Aufruf		Backend	Alert
3016	30	Verhindern mehr als 3 Bilder-Sätze auf dem Backend vorliegen		Backend	Alert
3017		Fehler bei der Entschlüsselung. Sym. Schlüssel nicht gleich.		Backend	Silent
	31	Keine Mobilfunk-Verbindung mit dem Backend		App	
	32	Keine Daten-Verbindung mit dem Backend		App	
	33	End-Daten erhalten keine Bilder (Es ist ein Initialisierungsfehler aufgetreten.Neue Initialisierung nötig.		App	

3.3.11.2 Remaining play protection time

```
GET /webapi/v1/user/vehicles/<vehicle_id>/ remainingPlayProtectionTime
HTTP/1.1
```

Attribute-Specification:

Input Parameters			
Name	**Type**	**Description**	**Condition**
vin	string	The vin of the remote service execution	mandatory

HTTP status codes:

- 200 OK
- 404 NOT FOUND if no vehicle with the given VIN exists for the current user

Output Parameters		
Name	**Type**	**Description**
playProtectionLockTime	int	Restzeit [in Sekunden] wann Spielschutz für neue Ausführung freigegeben ist.

		„0“, wenn Spielschutz-Limit nicht erreicht.

Response example JSON

```
{
"remote360Status": {
      "playProtectionLockTime": "60"
      }
}
```

3.3.11.3 Download remote360 data packages

Download eines einzelnen Datenpakets als Binary.
Das Datenpaket enthält entweder Bilddaten oder Kalibrierungsdaten. Die Zuordnung des Bildes und die Kalibrirungsdaten erfolgt anhand des Dateinamens (z.B. LEFT.mp4). Der Dateiname ist über den Parameter filenamePackage abzulesen

```
GET /webapi/v1/user/vehicles/<vehicle_id>/remote360downloadPackage
HTTP/1.1
```

Attribute-Specification:

Input Parameters			
Name	**Type**	**Description**	**Condition**
eventId	string	The event ID of the remote service execution	mandatory
dataPackageId	int	ID (or name) of the package. vin_creation-time_DDMMYYHH24MISSMS.blob	mandatory

HTTP status codes:

- 200 OK
- 404 NOT FOUND if no vehicle with the given VIN exists for the current user or no image is available

3.3.12 Resource Weather

3.3.12.1 Forecast

```
GET /webapi/v1/weather/forecast HTTP/1.1
```

Returns the weather forecast for the given geo coordinates. The request parameters 'lat' and 'lon' denote the geo coordinates for the weather forecast.

Request-Parameter-Specification

Name	**Type**	**Description**	**Condition**
lat	double	Search center latitude	mandatory
lon	double	Search center longitude	mandatory
tempscale	string	Temperature unit Celsius or Fahrenheit.	optional, default: “C”.

		Enum: - C - F	
days	int	Number of days [1-6] for forecast.	optional, Range: 1-6

Response contains a list of JSON encoded weather forecast for the given location. List may be empty. Example:

JSON

```
{
  "location": {
    "latitude": 48.191469,
    "longitude": 11.56699,
    "city": "MÃ¼nchen"
  },
  "temperatureScale": "C",
  "forecastCount": 3,
  "forecasts": {
    "forecast": [


      {


        "date": "2015-08-31T22:00:00.000+0000",

        "forecastDescription": "Morgens Temperatur 19, nachmittags 30 und 
abends 21 Grad. Fast klar, abends Schauer.",
        "overallDayForecast": {
          "weatherState": "CLOUDY",
          "minTemperature": 17,
          "maxTemperature": 31,
          "avgTemperature": 24,
          "windSpeed": 4,
          "description": "Morgens Temperatur 19, nachmittags 30 und 
abends 21 Grad. Fast klar, abends Schauer."
        },
        "morningForecast": {
          "weatherState": "CLOUDY",
          "humidity": 63,
          "minTemperature": 19,
          "maxTemperature": 19,
          "windSpeed": 1,
          "windDirection": "SOUTHWEST",
          "chanceOfRain": 1
        },
        "noonForecast": {
          "weatherState": "CLOUDY",
          "humidity": 29,
          "minTemperature": 30,
          "maxTemperature": 30,
          "windSpeed": 1,
          "windDirection": "SOUTHWEST",
          "chanceOfRain": 2
        },
        "eveningForecast": {
          "weatherState": "MODERATE_RAIN",
          "humidity": 69,
          "minTemperature": 21,
          "maxTemperature": 21,
          "windSpeed": 4,
          "windDirection": "WEST",
          "chanceOfRain": 83
        }
      }
    ]
  }
}
```

HTTP status codes:

- 200 OK

3.3.12.2 Highresolution Weather

```
GET /webapi/v1/weather/highresolution HTTP/1.1
```

Returns the weather details for the given coordinates. The request parameters 'lat' and 'lon' denote the geo coordinates for the weather forecast.
Request-Parameter-Specification

Name	**Type**	**Description**	**Condition**
lat	double	Search center latitude	mandatory
lon	double	Search center longitude	mandatory
tempscale	string	Temperature unit Enum: - C F	optional, default: “C”.
resolution	int	Resolution in minutes [1, 10, 30, 60, 120, 240]	optional
limit	int	Limit of measure points [1-30]	optional

Response contains a list of JSON encoded weather states for the given location. List may be empty.
Example:

JSON

```
{
  "provider": "Foreca",
  "resolution": 120,
  "unit": "C",
  "location": "München",
  "forecasts": {
    "forecast": [
      {
        "weatherState": "LIGHT_RAIN",
        "humidity": 32,
        "temperature": 29,
        "windSpeed": 1,
        "windDirection": "WEST",
        "chanceOfRain": 10,
        "avgRainfall": 0.1,
        "time": "2015-09-01T14:00:00.000+0000"
      },
      {
        "weatherState": "LIGHT_RAIN",
        "humidity": 38,
        "temperature": 27,
        "windSpeed": 4,
        "windDirection": "NORTH",
        "chanceOfRain": 24,
        "avgRainfall": 0.1,
        "time": "2015-09-01T16:00:00.000+0000"
      }
    }
  }
}
```

HTTP status codes:

- 200 OK

4 Organizational topics

4.1 Responsibilities

Functional responsibility

The functional responsibility for the WebAPI belongs to EI-6. In case of interface changes, it is an EI-6 task to inform all interface users. Any change affecting the functional or nun-functional behavior of the interface must be approved by EI-6 before being implemented.
The backend IT groups have to provide one contact person consolidating all backend requirements and making sure that requested changes are implemented in the respective systems and in the expected manner. This interface contract is the only valid contract for the WebAPI, no sub-contracts with underlying systems shall exist.

Technical responsibility

The interface is provided to EI-6 by the ConnectedDrive backend IT groups (FG-71 / FG-632). The responsibility for operations, continuous availability and monitoring relies upon FG-420. In case of any technical or functional issues, EI-6 must be informed immediately.

Contact persons for productive operations

Topic	Responsible / Department / Phone

Decision board

For using the interface an approval is required. This includes providing necessary access keys. EI-6 has to define a decision board for this approval process. Without written approval, no access keys may be given to any interested party.

4.2 Problem management

Approach

In case of technical problems, FG operations must be contacted to investigate the problems.
In case of all functional problems, FG-72 must provide support in order to analyze and solve the problem.

4.3 Change management

In case of interface changes, the EI-4 change request process must be used. All required decision groups are already in place. API changes must be approved by the InterfaceManagement group (EI-64) to ensure a technical and conceptual consistency. Without that approval, no change request may be implemented.

The API must be versioned. Changes which are not backward compatible (bigger change than just adding new attributes in returned data) must be checked if they lead to a new API version. The old interface version must be continued for at least one year, except in cases where urgent requirements

enforce a change of an existing interface. As the interface is used by customer systems, ensuring functional correctness and ongoing correct operation is the number one priority.
Creating a new API may only be decided by EI-6

4.4 Dates

Milestones	**Responsible**	**Date**
Availability of current RemoteServices following the new interface on a test environment		2012-10-31
Interface provided on INT with a subset of the final features (RemoteServices and VehicleStatus)		2012-11-15
Completion of interface on INT environment		2012-11-30
Completion of interface on PROD environment		2013-03-31

4.5 Open aspects

Aspect	**Responsible**	**Date**
SLA – Response times and availability should be specified and agreed upon		

4.6 Signatures

- The interface specification has been accepted by the signing persons at TT.MM.JJJJ and is binding for the implementation.
- Changes regarding content and timeline require written agreement of all signing persons.

Projekt / Anwendung	Verantwortlicher	Kurzz.	Telefon	Datum / Unterschrift
B2V	Albrecht Büchner	FG-712		
BMWi	Christian Clauss	FG-662		
App	Florian Wagner	EI-61		
EI-6 BE-Team	Konrad Hübner	EI-64		

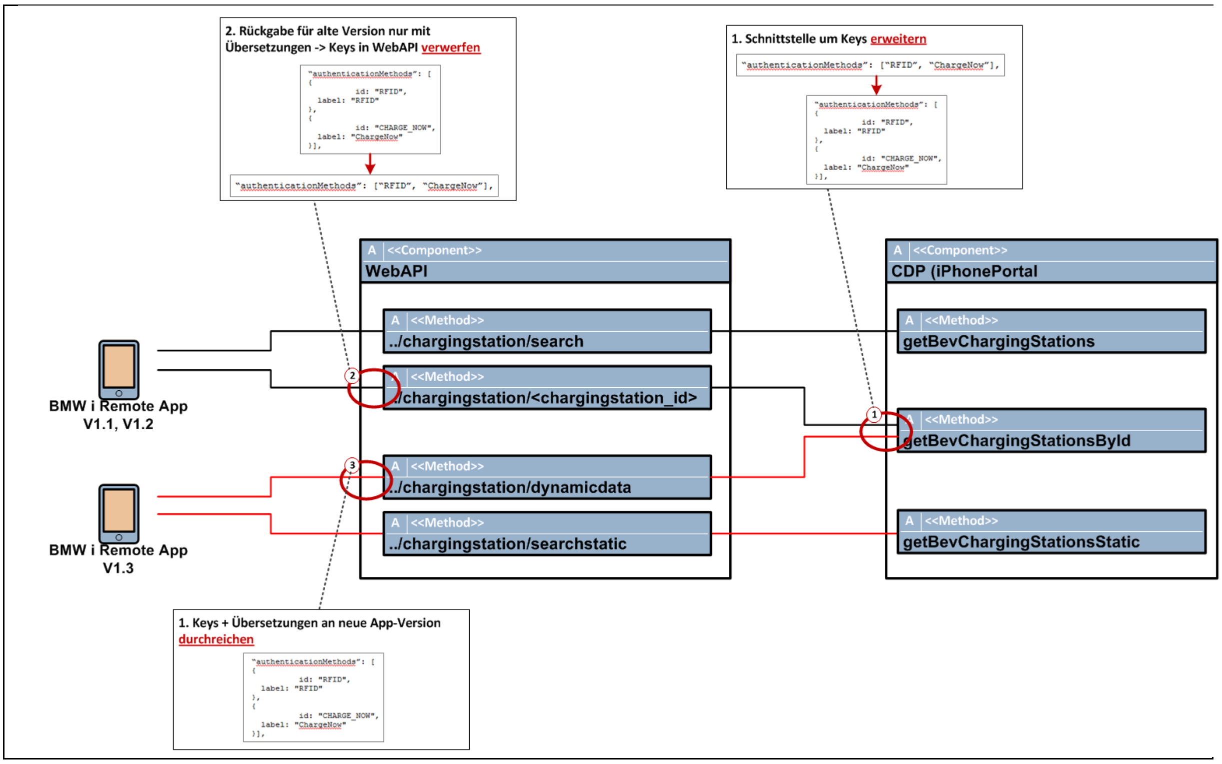
5 Appendix

5.1 Additional information

5.1.1 Resource ChargingStation variants

Depending on the methods called by the BMW i Remote App, the ChargingStation Resource that is returned varies in the attribute authenticationMethods (String-Array vs. Key-Value Object-List).

The following figure shows the changes and access paths that are relevant for this variation based on the actual IT Architecture:


2. Rückgabe für alte Version nur mit Übersetzungen -> Keys in WebAPI verwerfen
1. Schnittstelle um Keys erweitern
A <<Component>>
WebAPI
A <<Method>>
../chargingstation/search
A <<Method>>
../chargingstation/<chargingstation_id>
A <<Method>>
../chargingstation/dynamicdata
A <<Method>>
../chargingstation/searchstatic
A <<Component>>
CDP (iPhonePortal
A <<Method>>
getBevChargingStations
A <<Method>>
getBevChargingStationsById
A <<Method>>
getBevChargingStationsStatic
1
2
3
BMW i Remote App V1.1, V1.2
BMW i Remote App V1.3
1. Keys + Übersetzungen an neue App-Version durchreichen

```
"authenticationMethods": [
{
            id: "RFID",
  label: "RFID"
},
{
            id: "CHARGE_NOW",
  label: "ChargeNow"
}],
```

```
"authenticationMethods": ["RFID", "ChargeNow"],
```

```
"authenticationMethods": ["RFID", "ChargeNow"],
```

```
"authenticationMethods": [
{
            id: "RFID",
  label: "RFID"
},
{
            id: "CHARGE_NOW",
  label: "ChargeNow"
}],
```

```
"authenticationMethods": [
{
            id: "RFID",
  label: "RFID"
},
{
            id: "CHARGE_NOW",
  label: "ChargeNow"
}],
```

We enhanced the interface of the CDP (1) and thus needed to change interface of the WebAPI (2) to transform the new result into the String-Array to provide the old ChargingStation Resource towards BMW i Remote App Versions 1.1. and 1.2.
BMW i Remote App Verion 1.3 and upwards will use another method which hast he Key-Value Object list as Charging Station Resource response.
[CR 23687]