OMNICOMM

Omnicomm Online

Integration Manual
02.04.2020

Contents

Omnicomm Online

Omnicomm Online allows the user to control the operation of vehicles and drivers by means of reports being part of it. To access Omnicomm Online only a personal computer is required, connected to the Internet.

Omnicomm Online has built-in special-purpose tools to collect the processed data and use them in the accounting documents and in the fleet monitoring systems.

This manual describes the built-in tools operation and the integration with third-party systems.

General Information

The integration with Omnicomm Online is used to expand the functionality of third-party systems and to automate the input of data obtained from Omnicomm Online into the accounting systems.

The following integration methods are available:

- REST API
- using the SOAP protocol

> We recommend using REST API for integration. It is not possible to expand integration capabilities using the SOAP protocol.

The integration module for Omnicomm system should be developed and deployed by the third-party accounting system implementers.

Omnicomm does not undertake to delegate specialists or to develop the integration module for third-party systems.

Main Features

REST API

REST API facilitates the automation of:

- user management
- vehicle profile management

- geofence management
- receiving reports
- receiving notifications
- using the OVMS (video service) subsystem

REST API makes it possible to quickly start collecting, processing, and monitoring data from vehicles.

SOAP

Data, downloaded from the Omnicomm Online system, allows, at the initiation of appropriate functionality of third-party systems, the following tasks to be performed automatically:

- Entering data on mileage and fuel consumption in the waybills
- Accounting for mileage and usage time for calculating the amount of work, driver’s salary, etc.
- Accounting for attachable equipment to calculate scope of performed work
- Accounting for mileage, engine hours and auxiliary equipment for maintenance planning
- Write-off of fuel consumed
- Comparison of documented refills with the actual ones to prevent theft of fuel
- Generation of violation reports
- Plan vs. actual analysis on work done, fuel used, etc.
- Use of the current location to select suitable vehicle to fulfill the order
- Track visualization on the map in a third-party software

Various reports can be generated using the vehicle parameters downloaded from Omnicomm Online.

Data from Omnicomm Online enable quick estimation of the accepted planning system correctness and effective dealing with all the possible machinations of fleet employees, associated with theft of petroleum products and unauthorized use of vehicles.

The main targets for the integration of the systems are accounting

systems, ERP-systems, and branch accounting systems.

> SOAP protocol does not support the use of OVMS (video service) subsystem. Use REST API for quick integration with the OVMS subsystem.

Web services provide for retrieving the following parameters:

- Mileage
- Fuel consumption
- Engine operation time
- Auxiliary equipment operation time
- Fuel volume graph
- Mileage with exceeded speed*
- Movement duration*
- Engine operation time in motion*
- Engine operation without motion*
- “Engine Off” time*
- Fuel level at a given moment*
- Fuel consumption in motion*
- Fuel consumption without motion*
- Fuel consumption rate per one hour of engine operation*
- Track*
- Actual location*

* – Parameters marked are available from version 2.4.2.

In addition to these parameters, it is possible to receive all the events displayed in the “Events” report in Omnicomm Online.

Connection

To obtain the connection address for web services, the client should contact Omnicomm’s Technological Support Team.

Connection to the web-service is performed by the Technical Support

Team Ph. 8 800 100-24-42, ext.5

Demo access to the web-services

REST API

Use the following for demo access:

address: http://stage.omnicomm.ru

username: demodealer

password: demodealer1

SOAP

If necessary, to test the connection to web-services (to verify the application without using a real Omnicomm Online account, or if there is doubt about the network settings), one can use the web-services demo server of Omnicomm.

Web services connection address:

http://demo.omnicomm.ru:8000/AnalyticalServer/v2/ws?wsdl

Login: rudemoru Password: rudemo123456

REST API

You can find the API method specification at https://developers.omnicomm-world.com

Obtaining the rights to use REST API

Please contact the Omnicomm technical support by emailing to support@omnicomm-world.com to obtain the rights to use REST API.

Authorization

A JWT token must be indicated in the Authorization header when accessing the REST API methods (except for authorization methods). This JWT token grants the right to use REST API.

JWT format: JWT<space><the JWT received from the authorization method>

Example:

Authorization: JWT eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MTU3MDM1OTEsImxvZ2luIjo
4ifQ.0I0CXcwWWxZWARE0eUEPOAvKd0prW_Uf0jbOMLnd5SI

The JWT expiry day is indicated in the payload-attribute exp in the Unix Time Stamp format in UTC. After the expiry, when accessing the method the Error 401 Unauthorized will appear.

The JWT can be obtained using the method POST /auth/login?jwt=1 or, upon expiry, post /auth/refresh by indicating in the Authorization refresh header the JWT obtained during the authorization from post /auth/login?jwt=1.

User Management

User management covers adding, deleting, and blocking users in Omnicomm Online, as well as getting a list of all Omnicomm Online users.

An authorized user may access the vehicle data.

An external system can carry out tasks on a user's behalf after logging in under their account (after obtaining the JWT with the user's right).

Description of user management methods: https://developers.omnicomm-world.com/#/Users/

Vehicle Management

It is necessary to add a vehicle profile to enable the processing of vehicle data in Omnicomm Online.

A vehicle is identified by a unique string identifier (UUID), which assigned when a new vehicle profile is added to Omnicomm Online, or by the terminal identification number.

Vehicles can be added to groups and the same vehicle can belong to more than one group. It is also possible to configure access to vehicle groups for users. Vehicle groups are created when a user is added as well as in the Omnicomm Online interface.

Description of vehicle management methods:
https://developers.omnicomm-world.com/#/Vehicles/

Geofence Management

Geofences are virtual areas on the map created by users in Omnicomm Online. When creating a geofence, specify its shape (a polygon, a circle, or a line) and its geographic coordinates.

Geofences are used to monitor vehicle location (entering/exiting a geofence) and other operation parameters, such as the vehicle's speed.

Description of geofence management methods:
https://developers.omnicomm-world.com/#/Geozones/

Notification Handling

Notifications are used to promptly notify users about recorded events.

Description of methods for handling notifications:
https://developers.omnicomm-world.com/#/Notifications/

Reports

Reports are used to obtain various information about vehicle operation.

Description of methods for obtaining reports:
https://developers.omnicomm-world.com/#/Reports/

Omnicomm Video Service

The Omnicomm video service covers the video terminal management and provides video material to the user.

Main features of the service:

- receiving, storing, modifying, and providing data of video terminal profiles
- receiving, storing, modifying, and providing task parameters for video file download
- executing video file download tasks

Description of methods for managing the video service: https://developers.omnicomm-world.com/#/VideoService/

Obtaining a video fragment

This section describes how to use the video service in a typical scenario.

To obtain a video fragment:

1. Log in to the account of a dealer or a user who has rights to the vehicle and to use the video service: POST /auth/login?jwt=1

2. Get the video profile from the vehicle terminal ID: GET /service/ovms/api/profiles

3. Get the video file:

- Create a task to obtain the video fragment based on the received video profile: POST /service/ovms/api/tasks
- Periodically check the task's status: GET /service/ovms/api/tasks/{task_id}
- After receiving the “done” status for the task, request the video file: GET /service/ovms/api/tasks/file/{task_id}

Restrictions

There are restrictions on the intensity of requests that may be sent to the Onmicomm Online REST API to protect it against DoS-attacks and errors of third-party systems.

When these restrictions are exceeded, any requests from the corresponding IP address or user to the REST API will be blocked.

Unsuccessful authorization attempts

Not more than 10 in 1 minute from the same IP-address.

Authorized calls

Not more than 180 in 1 minute for each user.

Unauthorized calls

Not more than 60 in one minute from the same IP-address.

SOAP

SOAP Methods List

For date and time data transmitting the UNIXTIME (in seconds) format is used. Units of other parameters are listed below.

signIn – authorization

Input Values
String login – user name in the system **String password** – password in the system

Returned Values
Boolean **status** – true/false true in case of successful authorization String **sessionId** – in case of successful authentication, the session identifier (minimum 16 characters) Unixtimestamp **dateTimeEnd** – in case of successful authorization, time of the session termination (the time after which you must log in again) String **error** – error message in case of improper authorization (incorrectly entered username and password, or incorrect data format)

getObjectSet – list of objects

SOAP

Input Values
String **sessionId** – session ID obtained during authorization

Returned Values
Boolean **status** – true/false true in case of successful operation. String **error** – error message in case an error occur Dataset **objects** – list of vehicles available to the user — owner of the registered session (meaning only those objects which the user has the right to view): Integer **id** – object identifier (the same as the identifier of the unit) String **objectName** – name of vehicle String **objectType** – type of vehicle String **GarageNumber** – garage number

getSmoothedFuel – smoothed fuel level for the period

Input Values
String **sessionId** – session identifier obtaied during authorization Integer **objectId** – identifier of vehicle / unit Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval

SOAP

Returned Values
Boolean **status** – true/false true in case of successful operation Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval Dataset **fuel** – data set (all the data from the archive for the selected period): Unixtimestamp **timeStamp** – time of the registered fuel level Double **smothedFuel** – smoothed value of the fuel, liters, accuracy up to 0.1 l String **error** – error message in case an error occur

SOAP

getFuelConsumption – fuel consumption for the period

Input Values
String **sessionId** - session identifier obtained during authorization Integer **objectId** - identifier of vehicle / unit Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) - end time of the interval

Returned Values
Boolean **status** - true/false true in case of successful operation. Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval Double **fuelConsumption** – fuel consumption for the period, liters, accuracy up to 0.1 l String **error** – error message in case an error occur

getEvents – list of events

Input Values
String sessionId – session identifier obtained during authorization **Integer objectId** – identifier of vehicle/unit. Optional parameter, if the identifier is not present, it returns the data for all vehicles. **Integer type** – type of event, required. **Unixtimestamp timeBegi**n (seconds) – start time of the interval **Unixtimestamp timeEnd** (seconds) - end time of the interval

SOAP

Returned Values
Boolean **status** – true/false true in case of successful operation. String **error** - error message in case an error occur Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval Dataset **objectEvents** – data set for each event: Unixtimestamp **timeStamp** – date and time of the event Integer **objectId** – identifier of vehicle/ unit String **type** – type of event String **parameters** – parameters of the event String **eventAddress** – address of the event, if available String **iButton** – iButton code in HEX. Only for events such as ‘Driver’ String **name** – name of geofence. Only for entry and exit to or from Geofence events

SOAP

getMileage – mileage for the period

Input Values
String **sessionId** – session identifier during authorization Integer **objectId** – identifier of vehicle/unit Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) - end time of the interval

Returned Values
Boolean **status** - – true/false true in case of successful operation. Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval Double **Mileage** – mileage in km for the specified interval, accuracy 0.1 km String **error** – error message in case an error occur

getEngineOnTime – engine running time for the period

Input Values
String **sessionId** - session identifier obtained during authorization Integer **objectId** - identifier of vehicle/unit Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) - end time of the interval

SOAP

Returned Values
Boolean **status** - true/false true in case of successful operation Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval Double **engineOnTime** – total running time of the engine, in seconds String **error** – error message in case an error occur

getVehiclesState – VH current status

Input Values
String **sessionId** – session identifier obtained during authorization VehiclesType **vehicles** – list of VH IDs

Returned Values
Boolean **status** – operation status String **error** – error message in case an error occur vehicleStatesType **states** – list of parameters describing each VH status

SOAP

getMileageSpeedExcess – mileage with excess speeding for the period

Input Values
String **sessionId** – session identifier obtained during authorization Integer **objectId** – identifier of vehicle/unit Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval

Returned Values
Boolean **status** - true/false true in case of successful operation. Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval Double **mileageSpeedExcess** – mileage with speeding in km for the specified interval, accuracy 0.1 km String **error** – error message in case an error occur

getMovementTime – movement time for the period

Input Values
String **sessionId** – session identifier obtained during authorization Integer **objectId** – the identifier of vehicle/unit Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval

SOAP

Returned Values
Boolean **status** - true/false true in case of successful operation. Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval Double **movementTime** – movement time, seconds String **error** – error message in case an error occur

getEngineOnTimeInMovement – engine running time in movement for the period

Input Values
String **sessionId** – session identifier obtained during authorization Integer **objectId** – identifier of vehicle/unit Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd (seconds)** – end time of the interval

Returned Values
Boolean **status** - true/false true in case of successful operation Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval Double **engineOnTimeInMovement** – engine running time during movement, seconds String **error** – error message in case an error occur

getEngineOnTimeWithoutMovement – engine running time without movement for the period

SOAP

Input Values
String **sessionId** – session identifier obtained during authorization Integer **objectId** – identifier of vehicle/unit Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval

Returned Values
Boolean **status** – true/false true in case of successful operation Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval Double **engineOnTimeWithoutMovement** – engine running time without movement, seconds String **error** – error message in case an error occur

SOAP

getEngineOffTime – engine shutdown time for the period

Input Values
String **sessionId** – session identifier obtained during authorization Integer **objectId** – identifier of vehicle/unit Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval

Returned Values
Boolean **status** - true/false true in case of successful operation Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval Double **engineOffTime** – engine shutdown time for the period, seconds String **error** – error message in case an error occur

getFuelConsumptionInMovement – fuel consumption during movement for the period

Input Values
String **sessionId** – session identifier obtained during authorization Integer **objectId** – identifier of vehicle/unit Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval

SOAP

Returned Values
Boolean **status** - true/false true in case of successful operation Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval Double **fuelConsumptionInMovement** – fuel consumption during movement for the period, liters, accuracy 0.1 liters String **error** – error message when an error occurs

getFuelConsumptionWithoutMovement – fuel consumption without movement for the period

Input Values
String **sessionId** – session identifier obtained during authorization Integer **objectId** – identifier of vehicle/unit Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval

Returned Values
Boolean **status** – true/false true in case of successful operation Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval Double **fuelConsumptionWithout Movement** – fuel consumption without movement for the period, litres, accuracy 0.1 l String **error** – error message when an error occurs

SOAP

getFuelConsumptionInMotohour – fuel consumption per motor hour

Input Values
String **sessionId** – session identifier obtaind during authorization Integer **objectId** – identifier of vehicle/unit Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval

Returned Values
Boolean **status** - true/false true in case of successful operation Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval Double **fuelConsumptionIn Motohour** – average fuel consumption for the engine hour for the period, liters, accuracy 0.1 l String **error** – error message when an error occurs

getFuelAtTime – fuel level at a given moment

Input Values
String **sessionId** – session identifier obtained during authorization Integer **objectId** – identifier of vehicle/unit Unixtimestamp **time** (seconds) – moment of time

SOAP

Returned Values
Boolean **status** - true/false true in case of successful operation Unixtimestamp **time** (seconds) – moment of time Double **fuelAtTime** – fuel level at a given moment, litres, accuracy 0.1 l String **error** – error message in case an error occur

SOAP

getUserNotificationsByPeriod – user notifications by the period

Input Values
String **sessionId** – session identifier obtained during authorization Unixtimestamp **timeBegin** – start time of the interval (UTC), seconds Unixtimestamp **timeEnd** – end time of the interval (UTC), seconds. If not defined, end time = system time of the request execution by the server Integer **page** – requested page number. If not defined, first page containing perPage records is returned Integer **perPage** – quantity of records per page, if not defined, quantity is not limited

Returned Values
Boolean **status** – operation status. True in case of successful operation String **error** – error message in case an error occur Unixtimestamp **timeBegin** – start time of the interval (UTC) , seconds Unixtimestamp **timeEnd** – end time of the interval (UTC), seconds userNotificationsType **userNotifications** – array of returning parameters sets Integer **notificationsCount** – total number of notifications for all pages. If no notification found, error code 10 is returned

SOAP

getVisitedGeozonesByPeriod – geofences visited by the period

Input Values
String **sessionId** – session identifier obtained during authorization Unixtimestamp **timeBegin** – start time of the interval (UTC), seconds Unixtimestamp **timeEnd** – end time of the interval (UTC), seconds vehicleIdsType **vehicleId** – list of the VH IDs. If absent, all available vehicles are used for request

Returned Values
Boolean **status** – operation status String **error** – error message in case en error occur Unixtimestamp **timeBegin** – start time of the interval (UTC), seconds Unixtimestamp **timeEnd** – end time of the interval (UTC), seconds visitedGeozone **geozoneVisits** – block of arrays of visited geofences parameters

getVehiclesParams – list of parameters available to the user

Input Values
String **sessionId** – session identifier obtained during authorization

SOAP

Returned Values
Boolean **status** – operation status String **error** – error message in case an error occur Vehicle **vehicles** – VH parameters that the current user is entitled to view

SOAP

signOut – session termination

Input Values
String **sessionId** – session identifier obtained during authorization

Returned Values
Boolean **status** – operation status String **error** – error message in case an error occur String **sessionId** – terminated session ID

getActiveNotificationRules – profiles of active notifications

Input Values
String **sessionId** – session identifier obtained during authorization

Returned Values
Boolean **status** – operation status String **error** – error message in case an error occur rulesType **rules** – parameters for each notification profile

SOAP

setDeviceIdToNotificationRules – assignment of notification profiles to VH

Input Values
String **sessionId** – session ID obtained during authorization String **deviceId** – device ID String **deviceTypeId** – device type identifier rulesType **rules** – identifiers of notification profiles to which it is necessary to assign VH

Returned Values
Boolean **status** – operation status String **error** – error message in case an error occur

SOAP

getFuelLevelsByTimeMoment – fuel level at a certain time

Input Values
String **sessionId** – session identifier obtained during authorization Unixtimestamp **timeMoment** – moment of time (UTC), seconds vehicleAndTankIdsType **vehicleAndTankIds** – list of vehicles and fuel tanks IDs. In the absence of the list, the request will be executed for all vehicles and tanks available to the user

Returned Values
Boolean **status** – operation status String **error** – error message in case an error occur Unixtimestamp **timeMoment** – moment of time (UTC), seconds fuelData **fuelDataSet** – list of parameters for each VH: int **vehicleId** – vehicle identifier; int **tankNumber** – fuel tank number; fuelLevelsType **fuelLevels** – fuel level data; activityPeriodsType **activityPeriods** – engine operation data; ignitionOffListType **ignitionOffList** – ignition- OFF data; ignitionOnListType **ignitionOnList** – ignition-ON data; llsFailurePeriodsType **llsFailurePeriods** – data on fuel level sensor failures

SOAP

getFuelLevelsByPeriod – fuel level by the period of time

Input Values
String **sessionId** – session identifier obtained during authorization Unixtimestamp **timeBegin** – start time of the interval (UTC), seconds Unixtimestamp **timeEnd** – end time of the interval (UTC), seconds. If the end time is not indicated, the end time of the period = the system time of the beginning of the Server-side request processing VehicleAndTankIdsType **vehicleAndTankIds** – list of vehicles and fuel tanks IDs. In the absence of the list, the request will be executed for all vehicles and tanks available to the user Int **reduce** – thinning: 0 = not required 1 = required

Returned Values
Boolean **status** – operation status String **error** – error message in case an error occur Unixtimestamp **timeBegin** – start time of the period (UTC), seconds Unixtimestamp **timeEnd** – end time of the period (UTC), seconds fuelData **fuelDataSet** – list of parameters for each vehicle: int **vehicleId** – VH identifier; int **tankNumber** – fuel tank number; fuelLevelsType **fuelLevels** – fuel level data; activityPeriodsType **activityPeriods** – engine operation data; ignitionOffListType **ignitionOffList** – ignition- OFF data; ignitionOnListType **ignitionOnList** – ignition-ON data; llsFailurePeriodsType **llsFailurePeriods** – data on fuel level sensor failures

SOAP

getSmoothedFuelLevelsByPeriod – smoothed fuel levels by the period of time

Input Values
String **sessionId** – session identifier obtained during authorization Integer **objectId** – object identifier Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval vehicleAndTankIdsType **vehicleAndTankIds** – list of vehicles and fuel tanks IDs. In the absence of the list, the request will be executed for all vehicles and tanks available to the user Int **reduce** – thinning: 0 = not required 1 = required

Returned Values
Boolean **status** – operation status String **error** – error message in case an error occur Unixtimestamp **timeBegin** – start time of the period (UTC), seconds Unixtimestamp **timeEnd** – end time of the period (UTC), seconds fuelData **fuelDataSet** – list of parameters for each VH: int **vehicleId** – VH identifier; int **tankNumber** – fuel tank number; fuelLevelsType **fuelLevels** – fuel level data; activityPeriodsType **activityPeriods** – engine operation data; ignitionOffListType **ignitionOffList** – ignition- OFF data; ignitionOnListType **ignitionOnList** – ignition-ON data; llsFailurePeriodsType **llsFailurePeriods** – data on fuel level sensor failures

SOAP

getRefuelingsAndDrainsByPeriod – fuel draining/refueling operations by the period of time

Input Values
String **sessionId** – session identifier obtained during authorization Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval vehicleAndTankIdsType **vehicleAndTankIds** – list of vehicles and fuel tanks IDs. In the absence of the list, the request will be executed for all vehicles and tanks available to the user Integer **page** – number of the requested page with data Integer **perPage** – number of entries per page; if it is not preset, it will be taken as unlimited one String **sortname** – field by which it is necessary to sort out the return parameters String **sortorder** – sort order: **asc** – ascending **desc** – descending

Returned Values
Boolean **status** – operation status String **error** – error mesage in case an error occur Unixtimestamp **timeBegin** – start time of the interval (UTC), seconds Unixtimestamp **timeEnd** – end time of the interval (UTC), seconds Integer **entriesCounter** – total number of entries by the period of time RefuelingsAndDrainsType **RefuelingsAndDrains** – list of parameters for each VH

getVehiclesProfiles – VH profiles matching the VH identifiers

SOAP

Input Values
String **sessionId** – session identifier obtained during authorization VehiclesType **vehicles** – list of VH IDs

Returned Values
Boolean **status** – operation status String **error** – error message in case an error occur **Vehicles** – list of parameters for each VH

getCurrentObjectState – vehicle current status

Input Values
String **sessionId** – session ID obtained during authorization Integer **objectId** – object identifier

SOAP

Returned Values
Boolean **status** – true/false. True in case of successful operation. String **error** - error message in case an error occur String **lastGPS** – latest valid coordinates. Contains latitude and longitude values separated by a semi-colon Integer **lastGPSDir** – movement direction, degrees from 0 to 359 Double **currentSpeed** – current speed at the given moment, in kph, accuracy 0.1 kph Double **currentFuel** – current fuel level, in litres, accuracy 0.1 l Boolean **currentIgn** – ignition status. True if the ignition is ON Boolean **speedExceed** – speed threshold exceed. True in case of speed threshold exceeded Integer **lastGPSSat** – number of satellites with the last valid coordinates Double **currentInputValue** – actual value of universal input. Attributes: Integer number – UI number, String name – UI name

getReportData – report for auxiliary equipment over the period, TPMS, IQFreeze

Input Values
String **sessionId** – session identifier obtained during authorization Integer **objectId** – object identifier Long **timeBegin** – start time of the interval (UTC), seconds Long **timeEnd** – end time of the interval (UTC), seconds String **reportTemplateID** – identifier of report template in Omnicomm Online. Possible values: addEquipment, TPMS, refState, refWork

SOAP

Returned Values
Boolean **status** – true / false. True in case of successful operation String **error** – error message in case an error occur ReportDataType **reportData** – array including report data

getSEOnTime – auxiliary equipment running time for the period

Input Values
String **sessionId** – session identifier obtained during authorization Integer **objectId** – identifier of vehicle/unit Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) - end time of the interval

Returned Values
Boolean **status** - true/false true in case of successful operation Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval Double **sEOnTime** (seconds) – auxiliary equipment running time for every connected UI. Attributes: Integer number – UI number, String name – UI name

getStatisticsByPeriod – statistics for the period

SOAP

Input Values
String **sessionId** – session identifier obtained during authorization Unixtimestamp **timeBegin** – start time of the interval (UTC), seconds Unixtimestamp **timeEnd** – end time of the interval (UTC), seconds. If not defined, end time = system time of the request execution by the server. int **objectType** – type of object: 0=vehicle; 1=driver; If the type does not not exist, error code 12 is returned. objectIdsType **objectIds** – array of the type objectIdsType, containing list of objectId parameters of int type. In case of non-existance, the query is performed for all the objectId of the corresponding type, available to the user. requiredStatParamsType **requiredStatParams** – list of the required subgroups of the ‘Statistics’ report parameters. If the list is empty, all the subgroups with all parameters are returned

SOAP

Returned Values
movingAndWorkingParamsType **movementAndWorkingParams** – subgroup of VH movement and operation parameters. If the list is empty, all the parameters of the subgroup are returned, otherwise only the enumerated parameters are returned. If the subgroup is not transmitted, the whole subgroup will not be returned. fuelParamsType **fuelParams** – subgroup of parameters for fuel, draining and refuellings. If the list is empty, all the parameters of the sub- group are returned, otherwise only the enumerated parameters are returned. If the subgroup is not transmitted, the whole subgroup will not be returned. optionalEquipmentParamsType **optionalEquipmentParams** – subgroup of auxiliary equipment parameters. If the list is empty, all the parameters of the subgroup are returned, otherwise only the enumerated parameters are returned. If the subgroup is not transmitted, the whole subgroup will not be returned. CANDataParamsType **CANDataParams** – subgroup of CAN parameters. If the list is empty, all the parameters of the subgroup are returned, otherwise only the enumerated parameters are returned. If the subgroup is not transmitted, the whole subgroup will not be returned. addDataParamsType **addDataParams** – subgroup of statistics aditional parameters: TPMS, iQFreeze, etc.

SOAP

getTracksByPeriod – VH track for the period

Input Values
String **sessionId** – session identifier obtained during authorization Unixtimestamp **timeBegin** – start time of the interval (UTC), seconds Unixtimestamp **timeEnd** – end time of the interval (UTC), seconds. If the end time is not indicated, the end time of the period = the system time of the beginning of the Server-side request processing VehiclesType **vehicles** – list of vehicle ID Int **reduce** – thinning: 0 = not required 1 = required

Returned Values
Boolean **status** – operation status String **error** – error message in case an error occur Unixtimestamp **timeBegin** – start time of the interval (UTC), seconds Unixtimestamp **timeEnd** – end time of the interval (UTC), seconds trackDataSetType **trackDataSet** – parameters of the track for every vehicle: **trackPoint** – parameters of the track point by one vehicle unixtimestamp **timestamp** – time of the event when the coordinates have been fixed Integer **latitude** – latitude with accuracy of 0.0000001 degree Integer **longitude** – longitude with accuracy of 0.0000001 degree Integer **direction** – direction, degrees Integer **sattelitesCount** – number of satellites Double **speed** (km/hour) – speed Long **timeStamp** (seconds) – time of event. (UTC)

getTrack – track

SOAP

Input Values
String **sessionId** – session identifier obtained during authorization Integer **objectId** – object identifier Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval

Returned Values
Boolean **status** – true/false true in case of successful operation String **error** – error message in case an error occur Unixtimestamp **timeBegin** (seconds) – start time of the interval Unixtimestamp **timeEnd** (seconds) – end time of the interval Dataset **trackEvents** – array of track points: String **gpsPos** – event coordinates. Contains latitude and longitude values separated by a semicolon Integer **gpsDir** – movement direction, degrees from 0 to 359 Integer **sattelitesCount** – number of satellites Double **speed** – speed, in km/hour with accuracy up to 0.1 km/hour Unixtimestamp **timeStamp** – point date and time

getEngineStatisticsByPeriod – engine operation statistics for a period

Input values
String **sessionId** – session identifier received during the authorization Integer **vehicleId** – vehicle identifier Unixtimestamp **timeBegin** (seconds) – interval start time Unixtimestamp **timeEnd** (seconds) – interval end time

SOAP

Returned Values
Boolean **status** – true/false. Returns 'true' if the operation was successful String **error** – text of the error message that appears when the error occurs Unixtimestamp **timeBegin** (seconds) – interval start time Unixtimestamp **timeEnd** (seconds) – interval end time Unixtimestamp **lastDataTimestamp** (seconds) – – timestamp of the last processed data (UTC) Unixtimestamp **operationStartDate** (seconds) – operation start time for the period (UTC) Unixtimestamp **operationEndDate** (seconds) – operation end time for the period (UTC) Integer **operationTime** (seconds) – operation time for the period Integer **engineOffTime** (seconds) – engine off time for the period Integer Integer **engineOnTime** (seconds) – engine on time for the period Integer **engineIdlingTime** (seconds) – engine idle operation time for the period Integer Integer **engineOperationTimeNormalSpeed** (seconds) – engine operation time under normal load for the period Integer **engineOperationTimeMaxSpeed** (seconds) – engine operation time at ultimate load for the period Integer **engineLoadTime** (seconds) – engine operation time under load for the period Integer **dataAbsenceTime** (seconds) – time of data absence for the period

SOAP

getShiftDataByPeriod – information about the shifts

Input values
String **sessionId** – session identifier received during the authorization Unixtimestamp **timeBegin** (seconds) – scheduled shift start time Unixtimestamp **timeEnd** (seconds) – scheduled shift end time Integer **devBeforeBegin** (seconds) – allowed deviation from the shift start time ahead of schedule Integer **devAfterBegin** (seconds) – allowed deviation from the shift start time behind schedule Integer **devBeforeEnd** (seconds) – allowed deviation from the shift end time ahead of schedule Integer **devAfterEnd** (seconds) – allowed deviation from the shift end time behind schedule Integer **vehicleId** – vehicle identifier

SOAP

Returned Values

Boolean **status** – true/false. Returns 'true' if the operation was successful
String **error** – text of the error message that appears when the error occurs
Integer **vehicleId** – vehicle identifier
Integer **vehicleType** – vehicle type (0 - car, 1 - fuel tanker)
shiftData:

> Unixtimestamp **pointDate** (seconds) – actual time of shift start/end (UTC)
> Integer **engineOperationTime** (seconds) – engine operation time at the moment of actual shift start/end
> Integer **shiftEngineOperationTime** (seconds) – engine operation time for the shift
> Double **mileage** (km) – engine operation time at the moment of actual shift start/end
> Double **shiftMileage** (km) – mileage for the shift
> **fuelData** – fuel parameters group:
>
> > Double **fuelVolume** (l) – fuel volume at the moment of actual shift start/end
> > Double **fuelConsumption** (l) – actual consumption at the moment of actual shift start/end

SOAP

getIntervalsInfo – information divided into intervals

Input values
String **sessionId** – session identifier received during the authorization Integer **vehicleId** – vehicle identifier Unixtimestamp **timeBegin** (seconds) – interval start time Unixtimestamp **timeEnd** (seconds) – interval end time Boolean **partsFlag** – division into parts (true or false) Integer **interval** (min) – interval length Boolean **geocodingFlag** – address resolution based on the coordinates(true or false) Boolean **additionalTankFlag** – returns a data set on the fuel level in the additional tank (true or false)

SOAP

Returned Values

Boolean **status** – true/false. Returns 'true' if the operation was successful
Integer **errorCode** – error code. Possible error codes:
2: Authorization required - authorization is required to access the data
3: Dead session number – the session has expired, re-authorization is required
4: Bad interval – incorrect time interval entered
5: Bad object – there is no vehicle with this identifier
7: Unusable object– the value cannot be calculated for an object with this identifier.
9: Access denied – no access rights to the object
10: Data not found – there is no data for the corresponding input values (no raw data events for the [request period + one event before the period start])
11:Blocked interval – the requested interval contains data blocking periods
13: Invalid format – the format is incorrect
14: Undefined error – the error is unspecified
19: Too many intervals: current N, allowed M - the number of intervals per period (N) exceeds the limit (M)
String **erorrDescription** – optional
Integer **vehicleId** – vehicle identifier
Integer **vehicleType** – vehicle type (0 - car, 1 - fuel tanker)
Integer **tanksNumber** – the number of fuel tanks
Integer **numberOfIntervals** – the number of intervals in the request period
periodData – the data set for the request period
intervalsData – the data for the interval:

> Integer **intervalNumber** – interval sequence number
> Unixtimestamp **startTime** – interval start date
> Unixtimestamp **endTime** – interval end date
> **statData** – statistics data set
> **gpsData** – GPS data set
> **engineData** – data set on engine operation
> **fuelData** – data set on fuel level
> **seData** – data set on auxiliary equipment operation
> **canData** – CAN bus data

Errors

List of returned errors:

0: No errors – there are no errors

1: Signing in failed – incorrect Login / Password entered

2: Authorization required – authorization is required to access the data

3: Dead session number – session has expired, re-authorization is required

4: Bad interval – incorrect time interval entered

5: Bad object – there is no object with this identifier

6: Admin login – someone is trying to log in as Admin User

7: Unusable object – the value cannot be calculated for the object with this identifier

8: Bad event type – there is no event type with this identifier

9: Access denied – no authorization to access the object

10: Data not found – no data for the corresponding input values

11: Blocked interval – the requested interval contains data blocking periods

12: Bad object type – the specified object type does not exist

13: Invalid format – the format is incorrect

14: Undefined error – the error is unspecified

15: 404 – page not found

Work Client Example

Import of Interfaces

wsimport -d bin -s src http://demo.omnicomm.ru:8000/AnalyticalServer/ws?wsdl

Java code:

It is necessary to change strings “user” and “pass” by real values.

```
package ru.omnicomm.test.client;

import ru.omnicomm.analyticalserver.*;

import java.net.MalformedURLException;
```

SOAP

```
import java.net.URL;

import java.util.List;

public class  ExampleClient {
   public static void main(String[] args) throws MalformedURLException {
      AnalyticalServer = new AnalyticalServer(new URL("http://demo.omnicomm.ru:8000/AnalyticalSer

      AnalyticalServerWS port = AnalyticalServer.getAnalyticalServerPort();

      AuthResponseEntry auth = port.signIn("user", "pass");

      String sessionId = auth.getsessionId ();

      System.out.println("auth sessionId: " + sessionId);

      ObjectSetResponseEntry objects = port.getObjectSet(sessionId);
      List<Vehicle> vehicles = objects.getVehicleList();

      for (Vehicle vehicle : vehicles) {
         System.out.printf("vehicle: %d / %s\n", vehicle.getVehicleID(), vehicle.getRegNmb());
      }

   }
}
```

Types of Events

Event type number	Meaning	Parameter values, comments
1	Start of refuelling (for refuelling tankers – fueling)	Value of refuelling, accuracy 0.1 litres
2	End of refuelling (for refuelling tankers – fueling)	Value of refuelling, accuracy 0.1 litres
3	Beginning of draining	Value of draining, accuracy 0.1 litres

SOAP

Event type number	Meaning	Parameter values, comments
4	End of draining	Value of draining, accuracy 0.1 litres
5	Ignition ON	Time from the last ignition switch off, minutes
6	Ignition OFF	
7	External power supply ON	
8	Battery power ON	
9	Driver authorisation	iButton key code in HEX
10	Transition to roaming	
11	Exit from roaming	
12	Instant speeding	Maximum speed value, accuracy 0.1 kph
13	Idle time	
14	Beginning of speeding	
15	Beginning of transaction groups (for refuelling tankers)	Fuel volume before the beginning of transaction groups, accuracy 0.1 litres

SOAP

Event type number	Meaning	Parameter values, comments
16	End of transactions group (for refuelling tankers)	Fuel volume after finishing transactions group, accuracy 0.1 litres
17	Start of fueling during transactions group (for refuelling tankers)	
18	End of fueling during transactions group (for refuelling tankers)	
19	Start of draining during transactions group (for refuelling tankers)	
20	End of draining during transactions group (for refuelling tankers)	
21	Connection established	
22	Additional equipment turned ON	
23	Additional equipment turned OFF	
24	Exceeding the auxiliary equipment max limits	

SOAP

Event type number	Meaning	Parameter values, comments
25	Return to auxiliary equipment normal values	
31	Pressing the panic button	
32	Overload of auxiliary equipment	
33	Power ON	
34	Instant RPM exceeding	
35	Entering the geofence	
36	Exiting the geofence	
38	Power OFF	
42	Beginning of stop	
43	End of stop	
44	Beginning of acceleration	
45	End of acceleration	
46	Digital input ON	
47	Digital input OFF	

SOAP

Event type number	Meaning	Parameter values, comments
48	Instant acceleration	
49	Unknown driver	
52	Device tampering	
53	Driver authorization finished	
54	iButton applied	
55	Deleted driver registration	

OMNICOMM

info@omnicomm-world.com

www.omnicomm-world.com