Inflow Process Specification

Car-Pass Professional Rest JSON Web Services

v1.2 – 3/02/2023

Version History

Date	Version	Changes
27/09/2022	1.0	● Initial Version
02/12/2022	1.1	● Added examples ● Added Manual Correction flow ● Added Waiting For DIV status
03/02/2023	1.2	● Modified examples ● Added detail to correction flow

Context and Objectives

Car-Pass is the nonprofit organization created by **FEBIAC** (Belgian federation of the Car and Two-wheeler Industries), the organizations in charge of the technical inspection and **Traxio** (Federation of car vendors, car service professionals as well as linked sectors), charged by the legislator with a task of public interest i.e. to protect buyers and to promote the fair trade in used vehicles by combating fraud with the odometer.

The reason for the existence of the organization arises from the federal law of 11/06/2004, modified by the law of 28/11/2018. The purpose of this legislation is the prevention and repression of fraud linked to odometer manipulations and provide transparent information about used cars to the buyer. The essential principles are the following:

- Mileage fraud is considered a serious crime sanctioned with severe penalties (up to 1 year imprisonment)
- Creation of a central database containing the odometer readings of all vehicles (cars and vans) registered in Belgium
- All professional car dealers and repair shops are required to transfer VIN number, mileage and date when repairing or maintaining a vehicle or replacing parts (ex. tyres, windscreens,…)
- The seller of a second hand vehicle is obliged by law to deliver a certificate showing the mileage history of the vehicle to the buyer. If he fails to do so the transaction is void

The Car-Pass non-profit was certified by royal decree of 4/5/2006 to manage the database and issue the mileage certificates.

To fulfil its mission, Car-Pass developed an information system covering the following **objectives**:

- Centralize odometer status information originating from multiple data providers located all over the Belgian territory.
- Store the kilometre information about all cars and vans registered in Belgium and gathered at different moments during their 'lifetime'
- Guarantee the quality, security and confidentiality of all centralized data.
- Deliver the Car-Pass document that shows the mileage history and other information foreseen by the law of a used car to any potential buyer of a second hand car.

When offering a registered vehicle for sale, any professional car seller is obliged by law to indicate the kilometre history of the vehicle and the other information foreseen by the law as these are made available by the association referred to in Article 6 at that moment in time.

Needless to say, the accuracy of the transmitted data is very important. We expect that professionals transfer the exact value they read on the dashboard, no approximate values or rounded numbers. They need also to pay attention that the date matches with the value of the odometer reading. Errors, if they remain uncorrected, will eventually appear on the Car-Pass document of the vehicle and may result in a considerable loss of the residual value.

Do note that the Car-Pass legislation only applies for passenger cars and light commercial vehicles (maximum permissible weight up to 3,5 t). Odometer readings of heavy trucks, buses or motorcycles don't need to be sent to Car-Pass and in any case will be rejected.

Car-Pass has drafted a set of guidelines to improve the quality of the transmitted odometer data. If correctly implemented by the software developer, he can apply for the 'Car-Pass approved DMS' certificate (see annex 2)

Do note that our maintenance window for ‘production’ is Thursday 19h – 24h (CET)

Definitions

Term	Description
VIN	Vehicle Identification Number. The unique number for the identification of a vehicle
Odometer Reading	The value of the Odometer as displayed on the dashboard.
Enterprise Organisation (legally responsible)	The enterprise identified by a KBO/BCE number (or VAT number) in the format of 10 digits including leading zero’s eg 04xxxxxxx1
Entered Organisation	The establishment/workshop where the vehicle is being worked, identified by its establishment number from KBO/BCE number. 10 digits, eg 2xxxxxxxx1

Security

Security is basic authentication with the establishment number of the garage as username, for testing purposes other accounts can be set-up. **Most frameworks and tools support the creation of a basic authentication header** based on username and password.

If you do this manually, which is not recommended, refer to Annex 1 for more details.

TLS versions

Only TLSv1.2 and TLSv1.3 is supported

Ciphers:

```
ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:
ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:
ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:
DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384
```

Vehicle History Report

Professional dealers offering second-hand vehicles for sale are legally obliged to display the information that appears on the Car Pass in the advertisement and showroom. Therefore they have to request a vehicle history report via web services or the Car-Pass website.

To request a Vehicle History Report (VHR), the user has to provide the VIN and the Odometer Reading of the day. It is not allowed to include another date in the VHR request than the date of the request. It goes without saying that the odometer reading must be accurate and match with the day of the request because Car-Pass will store this data and they will appear and all upcoming VHR and Car-Pass documents.

In fact a request for a VHR is treated as an ordinary Observation. Therefore the same web service method is used to provide a new Observation. The distinction is made based on the provided Observation Type.

Important Notice:

If the provided data caused an issue, an issue feedback is returned . This issue has to be solved before the user can retrieve the Vehicle History Report.

Issues can be corrected in the same way as a normal correction. When the correction is valid and the data is accepted by Car-Pass the Vehicle History Report will be generated. The professional can retrieve the Vehicle History report by performing a status request. The VHR contains a unique hyperlink. This hyperlink needs to be included in the advertisement for the vehicle, to allow potential buyers to consult the Car-Pass data for that particular vehicle. It is also possible to receive the vehicle history as a pdf, that can be printed and apposed in or close to the vehicle in the showroom.

While creating a **vehicle history** *we consult several external databases*. These sometimes respond very slowly or not at all. For this, our timeout is set to 1 minute. When we return an 'ONGOING', it means that we are waiting for a response. Please write your application so that it recalls the method GET /observation{request with a maximum of 1 request in 10 seconds until you get a valid response back that is different from 'ONGOING' and this for a duration of minimum 90 seconds.

Technical documentation

A swagger page can be found on: https://ws-professionals.beta.car-pass.be/api/swagger-ui.html

Base urls

Beta	https://ws-professionals.beta.car-pass.be/api
Production	https://ws-professionals.car-pass.be/api

Dates are formatted dd/mm/yyyy
Supported languages are NL, FR, DE. EN only for the Vehicle History Report pdf

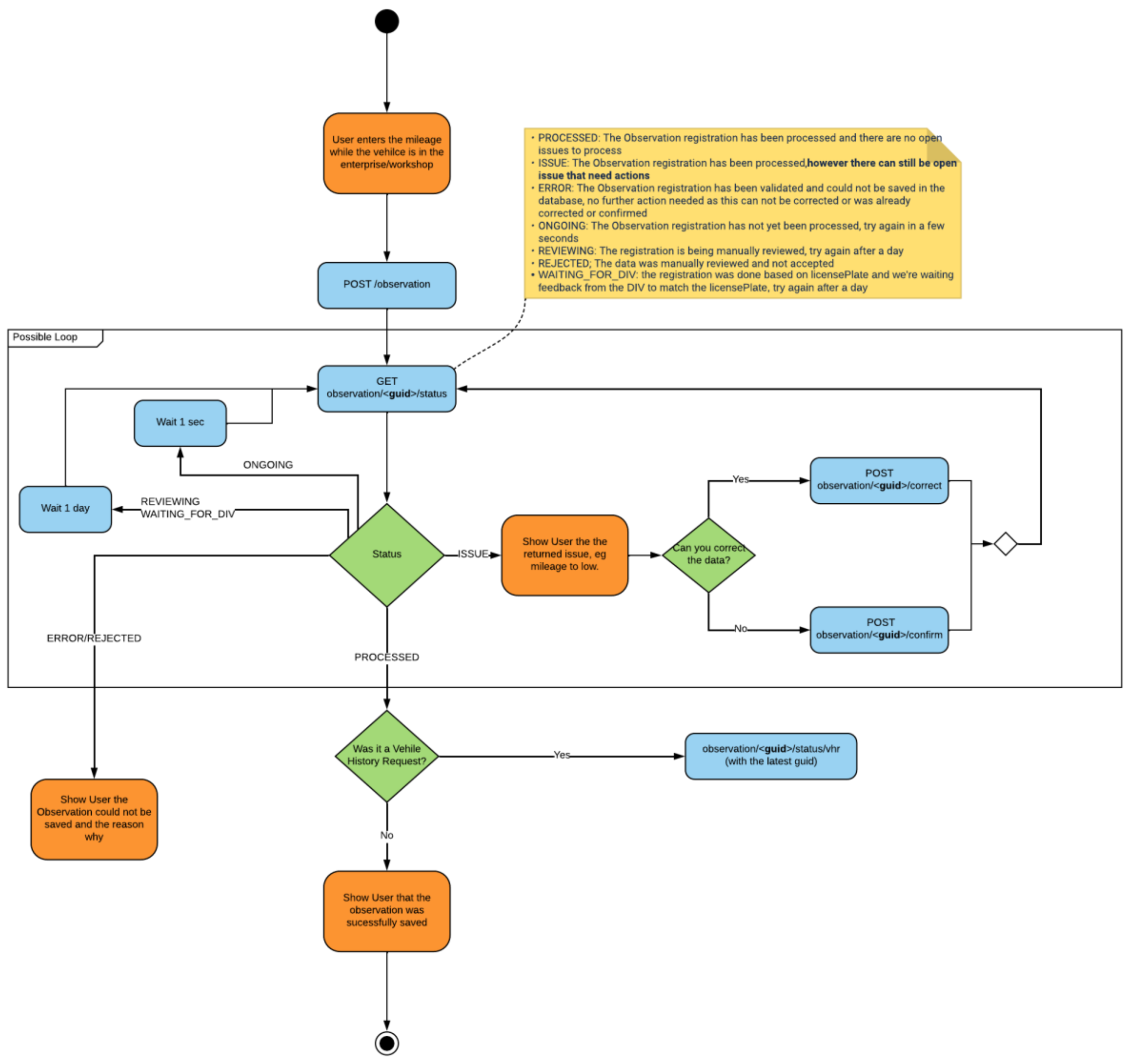
Services

Workflow

Important to know is that processing of requests happens asynchronously. You will make a request to submit data, you will receive a guid that you'll need to store and use in the following requests. The below requests contain more information on how and when to use these.

You'll start with POST OBSERVATION then use GET observation{requestguid}/status to check if registration was successful or further action is needed.
If the status is ISSUE, further action is needed either POST observation{requestguid}/correction or POST observation{requestguid}/confirm

Observation flow


User enters the mileage while the vehilce is in the enterprise/workshop
POST /observation
Possible Loop
GET observation/<guid>/status
Wait 1 sec
Wait 1 day
ONGOING
REVIEWING
WAITING_FOR_DIV
Status
ISSUE
Show User the the returned issue, eg mileage to low.
Can you correct the data?
Yes
No
POST observation/<guid>/correct
POST observation/<guid>/confirm
ERROR/REJECTED
PROCESSED
Was it a Vehile History Request?
Yes
observation/<guid>/status/vhr (with the latest guid)
No
Show User the Observation could not be saved and the reason why
Show User that the observation was sucessfully saved

- PROCESSED: The Observation registration has been processed and there are no open issues to process
- ISSUE: The Observation registration has been processed,**however there can still be open issue that need actions**
- ERROR: The Observation registration has been validated and could not be saved in the database, no further action needed as this can not be corrected or was already corrected or confirmed
- ONGOING: The Observation registration has not yet been processed, try again in a few seconds
- REVIEWING: The registration is being manually reviewed, try again after a day
- REJECTED; The data was manually reviewed and not accepted
- WAITING_FOR_DIV: the registration was done based on licensePlate and we're waiting feedback from the DIV to match the licensePlate, try again after a day

Ongoing

ONGOING means the Car-Pass system is still gathering data and validating the registration.

For a Vehicle History request this can mean that external sources are being called for data. In case of a Vehicle History request we ask special consideration that it can take up to 1,5 minutes for car-pass to gather data from all external sources.

In other cases it can mean there is a high load and a backlog in processing data.

In both scenarios you have to reschedule the status call, but we ask to do this with exponential backoff procedure. Eg. After 1 second, 2 seconds, 4 seconds,…
This will limit the load for your and the Car-Pass system

Reviewing & Rejected

These are special statuses that require consideration

The REVIEWING status can be triggered through

- The manual correction API, see below in the operations
- By doing multiple corrections and sending a correction on 129 issue

To prevent fraud, Car-Pass limits how many corrections can be made. If a correction is done, but the data is still not logical and there is again an issue the issueType 129 will be shown. This indicates that any following correction (not confirmation) will have to be reviewed by Car-Pass. If the user does such a correction, the next status request will indicate status REVIEWING. This means this review is pending, and will be processed in 1-2 business days.

Do not keep repeating a status call for this. The review will be done between 1-2 business days, afterwards the status can be retrieved.

If the data is accepted, the status will be PROCESSED, no further action needed. If the data was not accepted the status will be REJECTED. In both cases the user has received feedback on the email address known at Car-Pass. In case of rejected, the original submitted data can still have an open issue and a correction can be tried again

Waiting For DIV

This is only relevant when submitting data based on a **license plate**. When submitting a license plate Car-Pass will check this plate with DIV. If DIV is not available the validation of the submitted data can not happen, in this case the status is temporarily set as WAITING_FOR_DIV. Car-Pass will automatically retry this data when DIV is available again, afterwards the final status of this registration becomes available which can be PROCESSED or ISSUE if there was something wrong. (See status and correction operations)
As such, when encountering this status, **do not** keep repeating status calls but schedule this for example try again in 6 hours or the next day.

Operations

POST /observation

- **Take note of the odometer reading on the dashboard**: this operation is performed by a staff member in the workshop and for obvious reasons requires the presence of the vehicle.
- Car-Pass expects to receive the exact odometer reader indicated on the dashboard, no approximate values or rounded numbers.
- Workshops have to send the data they hold to Car-Pass immediately, when the vehicle is still in the workshop.
- Car-Pass doesn't accept that the same user sends an odometer for the same vehicle twice with the same odometerReadingDate. The second reading will generate an error and will be rejected.

Input

Field	Type	Mandatory	Description
Vin	String, max 17	N	Either vin or licensePlate must be provided. Both are not accepted together.
licensePlate	String, max 9	N	Either vin or licensePlate must be provided. Both are not accepted together. licensePlate cannot be used with observationType VEHICLE_HISTORY_DOCUMENT
Unifier	Integer	N	The DIV unifier, used to identify vins shorter than 17 characters
firstUseDate	Date	N	The firstUseDate of the Vehicle, used to identify vin's shorter than 17 characters
odometerReadingDate	Date	Y	The date the odometer was read (we expect this information on the day it was read, so most of the time should be equal to the current date)
odometerReading	Integer	Y	The exact reading as read from the dashboard
observationType	Enum	Y	● REPAIR_MAINTENANCE_SERVICE: a normal registration, to be used in most situations ● ODOMETER_REPLACED: indicate that the odometer counter has been replaced during the work ● VEHICLE_HISTORY_DOCUMENT: a registration that request a vehicle history document to be generated ● ROAD_SIDE_ASSISTANCE: to be used for registration done during roadside assistance
enterpriseOrganisation	String	Y	The enterprise number of the enteredOrganisation

enteredOrganisation	String	Y	The KBO number of the establishment where odometer was read

Response

Field	Type	Description
requestGuid	String (UUID)	This is unique identifier that you'll use to follow up your request

Refer to GET /observation/{requestGuid}/status

GET observation/{requestGuid}/status

Usage: based on the request guid in the response of POST /observation or POST /observation{requestGuid}/correct retrieve the status to verify if data has been loaded or further action is needed

Input

- The requestGuid from the POST as URL path parameter

Output

Field	Type	Description
Status	ENUM	• PROCESSED: the observation has been successfully registered, no further action needed • ISSUE: Car-Pass notices a data issue, explanation in the linked issue, a Correction or Confirm action is needed • ERROR: the provided data was not able to be saved in the database, no further action is needed for this request • ONGOING: The provided observation is still being validated by the Car-Pass system • REVIEWING: The provided observation will be manually validated

		by a Car-Pass employee ● REJECTED: the provided data could not be accepted after manual review ● WAITING_FOR_DIV: the registration was done based on licensePlate and we're waiting feedback from the DIV to match the licensePlate
Issues	List	Will be filled in with issues that require attention by the user
Issue.type	String	A unique key for the type of issue
Issue.description	Object	Placeholder object for translations. The description explains the data issue that needs to be Corrected or Confirmed
Issue.description.nl		
Issue.description.fr		
Issue.description.de		

POST /observation/{requestGuid}/correction

Usage: this allows correcting the data from a previous request when the status was **ISSUE**. When the status is not ISSUE a synchronous error is returned.

Example: on the GET /status call an issue was returned that the inputted mileage seems to low. This is shown to the user in the garage, he/she verifies this and sees an input error is made. The user types in another odometerReading value and this is submitted, all other data needs to be filled in and stays the same. (The user could also modify the odometerReadingDate and/or VIN to correct the issue)

Example: on the GET /status call an issue was returned that the VIN is unknown. This is shown to the user in the garage, he/she verifies this and sees an input error is made. The user types in another VIN value and this is submitted, all other data needs to be filled in and stays the same.

Input

- The requestGuid from observation being corrected as URL path parameter

Fields

Refer to POST /observation, this uses the same object

Response

Field	Type	Description
requestGuid	String (UUID)	This is unique identifier that you'll use to follow up your request

Refer to GET /observation/{requestGuid}/status

POST /observation/{requestGuid}/confirm

Usage: this allows **confirming** the data from a previous request when the status was **ISSUE**. The user verifies the information provided in the status call describing the issue, and when the user is either unable to provide a correction or is sure the data as submitted is valid a confirm needs to be sent that will close the issue.

Input

- The requestGuid from observation being corrected as URL path parameter

Request body is not required
Response

Field	Type	Description
requestGuid	String (UUID)	This is unique identifier that you'll use to follow up your request

Refer to GET /observation/{requestGuid}/status

GET observation/{requestGuid}/status/vhr

Usage: when in the POST /observation an observationType VEHICLE_HISTORY_DOCUMENT has been used to request a document to be used in secondhand car sales **and** the GET /observation/{requestGuid}/status has returned status PROCESSED (if there are still issues you won't receive a Vehicle History Document)

Input

- The requestGuid from observation being corrected as URL path parameter

Response

Field	Type	Description
file	String (base64 encoded pdf)	A base64 encoded string that represents a pdf containing the vehicle history document
publicUrl	String (url)	This url is an online version of the vehicle history document and can be used in online ads for a second hand vehicle

POST /observation/{requestGuid}/manual-correction

Usage: this allows to correct data from a previous request that was in status PROCESSED, **when there was no issue**.

Example: the user in the workshop notices a typo after submitting the data to Car-Pass and wants to make a correction.

This can only be done when there were **no issues** on the previous registration, the previous registration is in status PROCESSED, the previous registration was not a CORRECTION

Input

- The requestGuid from observation being corrected as URL path parameter

Fields

Refer to POST /observation, this uses the same object

Response

Field	Type	Description
requestGuid	String (UUID)	This is unique identifier that you'll use to follow up your request

Refer to GET /observation/{requestGuid}/status

This is a separate operation because it will be reviewed manually by a Car-Pass employee before the change is accepted. As such the status call will return 'REVIEWING'
Do not keep repeating a status call for this. The review will be done between 1-2 business days, afterwards the status can be retrieved.

Error Responses

When a request is structurally not valid **or** data issues like unknown guids, or corrections when the status is not ISSUE an error message will be returned in this format. Normally this is something encountered during development and your software should be built in a way that it will not send requests that are not possible (missing mandatory fields, impossible combinations, wrong data types, …)

Response

Field	Type	Description
Errors	List	List of errors in the request
Field	String	Indicating for which field or fields the error occurred
Message	String	Details about what is wrong

Example

```
{
  "errors": [
    {
      "field": "vin, licensePlate",
      "message": "Provide either vin or licensePlate. Both are not accepted together and at least one must be provided."
    }
  ]
}
```

Annexes

Annex 1: Examples

The following describes a few common uses cases with example request and responses to help clarify the flow

Example: normal registration with issue and a correction

Step 1: submit the information entered by the user in the workshop

POST /observation

```
{
  "licensePlate": "1ABC001",
  "odometerReadingDate": "2022-09-15",
  "odometerReading": 37412,
  "observationType": "REPAIR_MAINTENANCE_SERVICE",
  "enterpriseOrganisation": "0400002000",
  "enteredOrganisation": "2000002001"
}
```

Response

```
{
  "requestGuid": "af66d153-5833-4fa2-8f34-f79f3f252a19"
}
```

Step 2: check status

*GET /observation/**af66d153-5833-4fa2-8f34-f79f3f252a19**/status*

Response

```
{
  "status": "ONGOING",
  "issues": null
}
```

Status is **ongoing,** the system is still validating the observation, try again

*GET /observation/**af66d153-5833-4fa2-8f34-f79f3f252a19**/status*

Response

```
{
  "status": "ISSUE",
  "issues": [
    {
      "type": "001",
      "description": {
        "nl": "Kilometerstand lager dan laatste.",
        "fr": "Kilométrage inférieur au précédent.",
        "de": "Kilometerstand niedriger als vorheriger."
      }
    }
  ]
}
```

Status is **ISSUE**, this means a correction or confirmation is required. This is shown to the user.
The user sees a type and wants to correct (this can be vin/licensePlate, odometerReadingDate and/or odometerReading!)

*POST observation/**af66d153-5833-4fa2-8f34-f79f3f252a19**/correction*

```
{
    "licensePlate": "1ABC001",
  "odometerReadingDate": "2022-09-15",
  "odometerReading": 57412,
  "observationType": "REPAIR_MAINTENANCE_SERVICE",
  "enterpriseOrganisation": "0400002000",
  "enteredOrganisation": "2000002001"
}
```

Response

```
{
  "requestGuid": "8e0c6bad-7a3f-421b-b0aa-c49e03904299"
}
```

Use the status call again to check the status of the correction. (new requestGuid!)
*GET /observation/**8e0c6bad-7a3f-421b-b0aa-c49e03904299**/status*

```
{
  "status": "PROCESSED",
  "issues": null
}
```

Status is Processed, no further action needed!

Example: Vehicle History with issue and a confirmation

Step 1: submit the information entered by the user (second hand car seller)
POST /observation

```
{
  "vin": "XYUIYAURZA1845118",
  "odometerReadingDate": "2022-09-15",
  "odometerReading": 37412,
  "observationType": "VEHICLE_HISTORY_DOCUMENT",
  "enterpriseOrganisation": "0400002000",
  "enteredOrganisation": "2000002001"
```

}

Response

```
{
  "requestGuid": "181a7b04-d735-40d3-8737-13e6f2e7e2db"
}
```

Step 2: check status

GET /observation/181a7b04-d735-40d3-8737-13e6f2e7e2db/status

Response

```
{
  "status": "ISSUE",
  "issues": [
    {
      "type": "209",
      "description": {
        "nl": "De kilometerstand is veel hoger dan verwacht.",
        "fr": "L'évolution du kilométrage est sensiblement plus élevée que par le passé.",
        "de": "Der Kilometerstand ist höher als der Normalwert."
      }
    }
  ]
}
```

The user double checks this and validates that his/her info is correct and wants to confirm the data.

POST observation/181a7b04-d735-40d3-8737-13e6f2e7e2db/confirm

Response

```
{
  "requestGuid": "0991bea7-e9ae-42d6-b170-cbed5fc936cc"
}
```

Check status again

*GET observation/**0991bea7-e9ae-42d6-b170-cbed5fc936cc**/status*

```
{
  "status": "PROCESSED",
  "issues": null
}
```

Status is processed, now the vehicle history document can be retrieved

*GET observation/**0991bea7-e9ae-42d6-b170-cbed5fc936cc**/status/vhr?language=NL*

Response

```
{
  "file": "JVBERi0xLjQKJeLjz9MKMSAwIGVPRgo=",
  "publicUrl": "https://public.beta.car-pass.be/vhr/8aea64b-4b36-4703-9199-62fea6f46442"
}
```

File is a base64 encoded string.

Annex 2: Basic Authentication header

The Authorization header is constructed as follows:

1. Username and password are combined into a string "username:password"
2. The resulting string is then encoded using the RFC2045-MIME variant of Base64, except not limited to 76 char/line
3. The authorization method and a space i.e. "Basic " is then put before the encoded string.

The web service can only be accessed with a username and password. These credentials must always be provided while making a request to Car-Pass.

Annex 3: Good practices – Car-Pass approved DMS

Car-Pass attaches the utmost importance to the quality of the data communicated by the enterprises in the car sector. After all, any errors will later end up on the Car-Pass certificate for the vehicle and will cast doubt on the correct odometer reading with a potential buyer. In the worst case, this could result in a sale falling through.
Correcting errors is a difficult task, both for Car-Pass and for the enterprise that has communicated the incorrect odometer reading. Prevention is therefore better than cure. In practice, it appears that many errors can be avoided at the source if the software used by the enterprise is based on a sound design. That is why Car-Pass has set a number of criteria that DMS software must meet in order to guide the user as much as possible when transferring odometer readings, thereby helping the avoidance of errors. If the software meets these requirements, it may carry the "Car-Pass approved DMS" label. Car-Pass has approved the following software.

Transfer of the data
The DMS must, as required by law, send the data to Car-Pass when the vehicle is still in the enterprise, regardless of the final closing of the file or billing.
The developer must be able to demonstrate that the data is transmitted in one of the following ways:

a) Automatic transfer (automatically after entering data on, for example, a work order and without the user having to explicitly press a button).
b) Manually, with installation of the necessary guarantees (warnings, reminders, etc.) that the data is being *sent before the vehicle file is closed or before the vehicle leaves the repair shop*.

Checks to ensure correct transfer and subsequent receipt by Car-Pass
The DMS system must contain the necessary checks to verify whether the data has actually been sent to Car-Pass and whether Car-Pass has received it properly.

1) In the first phase of the web service, Car-Pass sends back a requestID.
   Is this requestID captured and stored, so that this requestID can be used even after restarting the PC/server?
2) Before launching the status/{requestGuid}/vhr, is there a delay time of 1 seconds as minimum?
3) Is there an adequate strategy to retrieve the data when the status/{requestGuid}/vhr is equal to 'ONGOING'?
4) Has the status/{requestGuid}/vhr been captured so that any issues can be corrected? (see §4 Corrections)
5) If, for technical reasons, the data cannot be sent for a certain period of time, it must be saved and the user must receive a warning. After the problems have been solved, the user and/or the DMS vendor should be able to send all overdue data with the correct date of work.

The DMS system displays the error messages with the correct issue type and description
In the DMS system, the outstanding problems must be visible, showing the correct problem number, the error code (issue type) and the description of the problem.

The DMS system allows issues to be corrected

The system allows corrections to be made. Under the term corrections is meant:

1. Modifying the VIN, mileage or date by using the correct requestGuid and observationType
2. Confirming the data that caused the issue as correct by using the confirm flow

If the input data has caused more than one issue, then all related issues will be closed when one of them is corrected. If the garage sends a correction to Car-Pass, the company's local database must be updated accordingly. The feedback on this correction will be loaded correctly.

The DMS shows all issues and which ones still need to be corrected.

Built-in checks to avoid system-based errors
The DMS system carries out a check to ensure if the entered kilometer reading is possible (precheck of data in the own system)

a. The DMS system gives a warning if the odometer reading that is entered is lower than the previous odometer reading in the garage database (local issue 001)
b. The DMS system gives a warning if the odometer reading that is entered is excessively high compared to the previous odometer reading in the garage database.(local issue 209)
c. The DMS system gives a warning if the odometer reading that is entered is the same as a previous odometer reading in the garage database
d. The DMS system gives a warning system if the odometerReadingDate is in the future
e. The DMS system gives a warning if the odometerReadingDate is in the past.
f. The input field for the mileage may only allow numbers and no special characters nor blanks.
g. When the data has already been processed by Car-Pass, the DMS prevents the same source from sending the identical data (same VIN, km and date) multiple times. (DMS should therefore not cause an issue_type 150)

Check to ensure that 17 characters are entered for VIN
If a VIN is entered with less than 17 characters, the user receives a warning. As the VIN of oldtimers often contain less than 17 characters, it should nevertheless be possible to send the data to Car-Pass.
When a VIN with more than 17 characters is entered, the user receives an error message. It should not be possible to send the data to Car-Pass.
An empty or impossible chassis number (00000000000000000) is not allowed. Only alphanumeric values are allowed.

The system makes it possible to exclude operations that are not mandatory to report (e.g. till sales, etc.) and certain categories of vehicles from being transferred to CAR-PASS (trucks, foreign number plates, etc.)
The user must have the option of excluding these operations/vehicles from being transferred to Car-Pass.

The system transfers the correct combination of date/odometer reading

The system links the correct date to the correct odometer reading, i.e. the date when the vehicle was present in the repair shop and the odometer value was recorded.

Possible errors that give rise to failure on this criterion (not exhaustive list):

- Using the invoice date
- Using the odometer reading at the date of the expertise and link it to the date of repair
- Using the date the appointment was made
- Re-sending the odometer reading when creating a credit note / re-invoicing
- Claims under guarantee: cut-off date
- Etc.

The user must be able to check his username and password

Test: The DMS system must have a screen showing the contact details that were entered at the time of activating the Car-Pass user account. These are at the minimum the following:

- The username (which should be the establishment number of the garage)
- The enterprise number (of the establishment), should be the VAT number
- The password
- The email address
- Optionally: telephone number of the Car-Pass helpdesk and link to the contact form.

If the garage changes its password and/or username locally in the DMS system, at least one warning must appear warning that it must also be changed on the Car-Pass website.

Vehicle History Request (optional)

The DMS allows the vehicle history to be consulted, the PublicUrl to be saved and associated issues to be processed. The DMS system opens the pdf document that was sent and allows to print the vehicle history.

Automatic or manual forwarding after ANY entry of the odometer reading of the vehicle for sale. As long as this odometer reading is not adjusted, no new VHR may be requested by the DMS.

It is not intended that the requests are stored locally and later queried in batch.

In order to create a VHR, Car-Pass has to contact several third parties (e.g. car manufacturers, DIV,...). Car-Pass waits a maximum of 1 minute for a response. Meanwhile the response status remains ONGOING. Thus, the DMS program may not repeat the RequestOdometerReadingStatus more frequently than 1 time per 10 seconds as long as the response status is ONGOING and should not abort the RequestOdometerReadingStatus query within 90 seconds.