Documentation of SDK-Tools for asanetwork

01/03

Version	Author	Date	Status	Comment
1.0	Martin Rothschink	7/2/2003	Released	For SDK Tools Version 1.0
3.0	Martin Rothschink	2/2/2011	Released	For SDK Tools Version 3.0
3.1	Martin Rothschink	3/10/2014	Released	For SDK Tools Version 3.1

Content

1 Introduction

1.1 General

The new SDK-Tools for asanetwork are intended for the development and the test of asanetwork-compatible applications. The tools support asanetwork-compatible commercial applications (Dealer Management Systems, DMS) as well as end devices (Clients, i.e. testing and checking instruments) in the asanetwork.

The new SDK-Tools offer extended capabilities, especially for the new vehicle identification and the transport of target data in asanetwork.

1.2 Overview and Application fields

Tool	Achievment	Main application
OrderSim	Simulation of an asanetwork-compatible commercial application (DMS). Creates and simulates order data, vehicle data, vehicle identification and target data for car exhausts.	Development of asanetwork-compatible testing and checking instruments (end devices, Clients) Reference behaviour for DMS functionality.
TestClient3	Simulation of asanetwork-compatible test and checking instruments. Processes order data and creates XML test results.	Development of asanetwork-compatible commercial applications (DMS). Reference behaviour for test and checking instruments.
AwnDebug3	Logging and visualization of all data exchanged in asanetwork. Saving of received data for later evaluation and debugging. Creating and editing of new or already existing records.	Troubleshooting during development and operation of asanetwork-compatible products.
AwnRemote	Sending of remote control commands to start an order on a distant testing instrument.	Starting of orders during the development of instruments operated by remote control.
AwnStatus	Sending and receiving of status notifications.	Simulation of status notifications during development.
LiveStreamSimulatior	Simulation of live transmission of brake measuring values.	Utilities for development of LiveStream-compatible receiving and controlling software.

Table 1, application fields of the new SDK-Tools

2 Order simulation with OrderSim

The program OrderSim is the further development of the order generator Office32, comprised until now in the asanetwork SDK. OrderSim is a Win32 console application and serves as a simulator for asanetwork orders, vehicle data, vehicle identification and target data for the exhaust emission check.

OrderSim implements the functions of a commercial application using asanetwork pre-settings.

2.1 Implemented services

Service	Priority	DiQual	DoQual	Application
Client order service	0	2	2	Sending and receiving of order data and extended vehicle data
Data storage service	5	1	0	Reception of test results parallel to the data storage service in network manager. Query of data is handled by network manager.
Identification data service	9	1	1	Sending and receiving of identification data. Query of data is handled by network manager.
Target data service for exhaust emission check	9	1	1	Sending and receiving of target data Query of data is handled by network manager.

Table 2, services used by OrderSim

2.2 Used data base

OrderSim uses fixed data to create orders. The following table shows the respective combinations of client, vehicle, extended vehicle data, identification data, target data and order positions.

Order/ Order position	Order DID	20030401	20030402	20030403	20030404	20030405
Vehicle		VW Golf	Opel Astra	Renault M. Senic	MB A-Class	BMW 7
Client		Johann Maier	Friedrich Ackermann	Ingeborg Redlich	Computer Service Auer	Xaver Rubenbaur
10 Car exhaust	AWNTXEM...	GKAT	GKAT	UKAT	Diesel	OBD
20 Brake	AWNTXBR000	X		X	X	X
30 Toe (side slip)	AWNTXSS000	X		X		X
40 Suspension	AWNTXSP000	X		X		X
50 Oil change	AWNTXOM0..	X		X	X	
60 wheel alignment	AWNTXWA000		X	X		X
70 Tyre checking	AWNTXTI000		X	X		
80 Wheel balancing	AWNTXWB000		X	X	X	
90 Tyre change	AWNTXML000				X	
100 Vehicle washing	AWNTXML000					X
0 Extended vehicle data			X	X	X	X
Identification data	AWNIXVEHID	X		X	X	X
Target data car exhausts	AWNDXEM000	X	X		X	X

Table 3, attribution of orders, vehicles and data

Note: The used client data are fictional; any similarity with real persons would be purely coincidental. The extended vehicle data originate from the TecDoc identification, as do the vehicle identification data; extra characteristics such as motor rotation speed, year of manufacture or brake data are fictional. Target data for car exhausts are also fictional and fit to the respective type of the checking, but not necessarily to the vehicle.

2.2.1 Orders and order positions

OrderSim generates 5 orders, corresponding to the respective order numbers 2003040x. Each order consists of up to 10 order positions with the position number 10 to 100. The allocation is effected in accordance with Table 3.

Order data are sent with the client order service xxxxx00000 with reference 5. Order data may be queried by other client order services with the Query function.

2.2.2 Extended vehicle records

For the orders 20030402 to 04 extended vehicle records containing additional information about the vehicle are available. These data are mainly used in Italy, in relation with the MCTC net.

Extended vehicle data are sent with the client order service xxxxx00000 with reference 3. Extended vehicle data may be queried by other client order services with the Query function.

2.2.3 Vehicle identification

For the orders 20030401 and 03 to 05 vehicle identification data in the new XML-form are available. These data are sent by OrderSim once with the identification data service AWNIXVEHID. OrderSim implements no

identification data server; a partial identification cannot be completed. Vehicle identification data can only be queried with the Query function and by the network manager.

At program start, OrderSim secures that there is only one record in the network manager at the time. For this purpose, while starting the Program eventually existing previous records are queried and deleted from the asanetwork.

2.2.4 Vehicle target data

For the orders 20030401, 02 and 04 to 05 exist vehicle target data for the exhaust emission check in the new XML-form. These data are sent by OrderSim with the vehicle target data service AWNDXEM000. OrderSim implements no vehicle target data server, a partial identification cannot be completed, and target data cannot be queried. Vehicle target data can only be queried with the Query function and by the network manager.

At Program start, OrderSim secures that there is only one record in the network manager at the time. For this purpose, while starting the Program eventually existing previous records are queried and deleted from the asanetwork.

2.3 Using OrderSim

2.3.1 Program parameters

OrderSim can be started and operated without designating parameters. The following parameters can be used to change the standard configurations:

Parameter	Meaning
`-n`	Order data are not deleted from the asanetwork at program end.
`-s`	Silent operation mode, received checking results are not displayed with notepad.
`-d`	Debug operation mode, all received order and result data are transferred to cust. dat and data. dat respectively.
`-i IP`	IP-Address of network manager in xxx.xxx.xxx.xxx format, e.g. 144.1.13.56
`-l DLoc`	Uses the specified DLoc instead of OFFICE1
`-f BaseDir`	Uses the File-Interface with the specified directory BaseDir instead of TCP/IP
`-y ms`	Delay in ms to simulate a slow DMS

Table 4, OrderSim parameters

2.3.2 Environment variables

Received records (Blobs) are saved in the temporary directory. OrderSim does not delete these files automatically. Therefore you should regularly delete files with the format 0000nnnn.awn from your temp dir.

The storage location can be changed by defining an environment variable. The search order is:

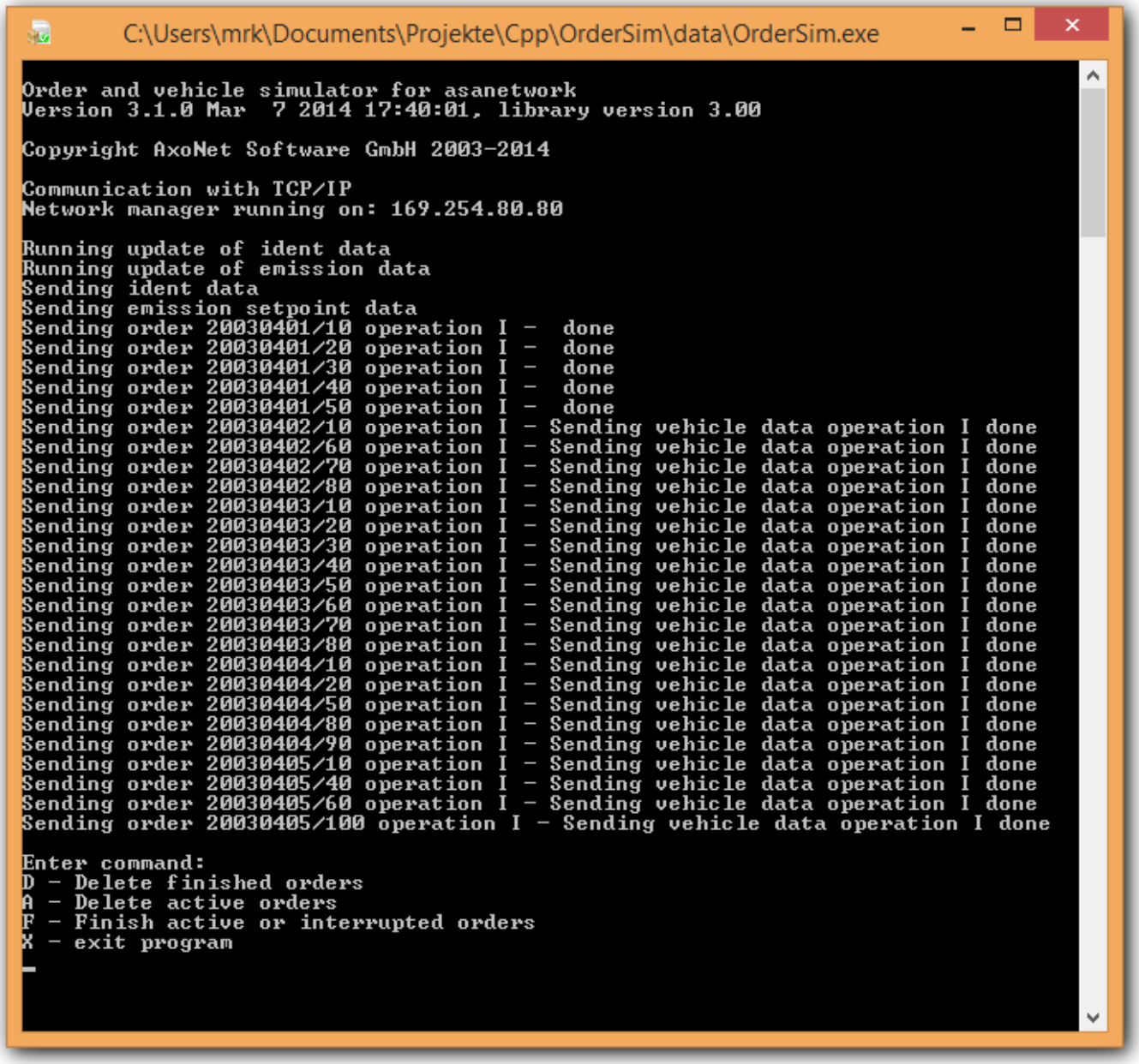
AWNTMP, TMP, TEMP, TMPDIR and finally TEMPDIR. The first path found is used. If none of the variables is defined, the directory of OrderSim is used.

Attention: If AWNTMP is defined globally this has effects on all asanetwork applications using the C or C++/Delphi interface. Alternatively, OrderSim can be start with a batch file defining a locally AWNTMP:

```
set AWNTMP=c:\MyTemp
OrderSim.exe
```

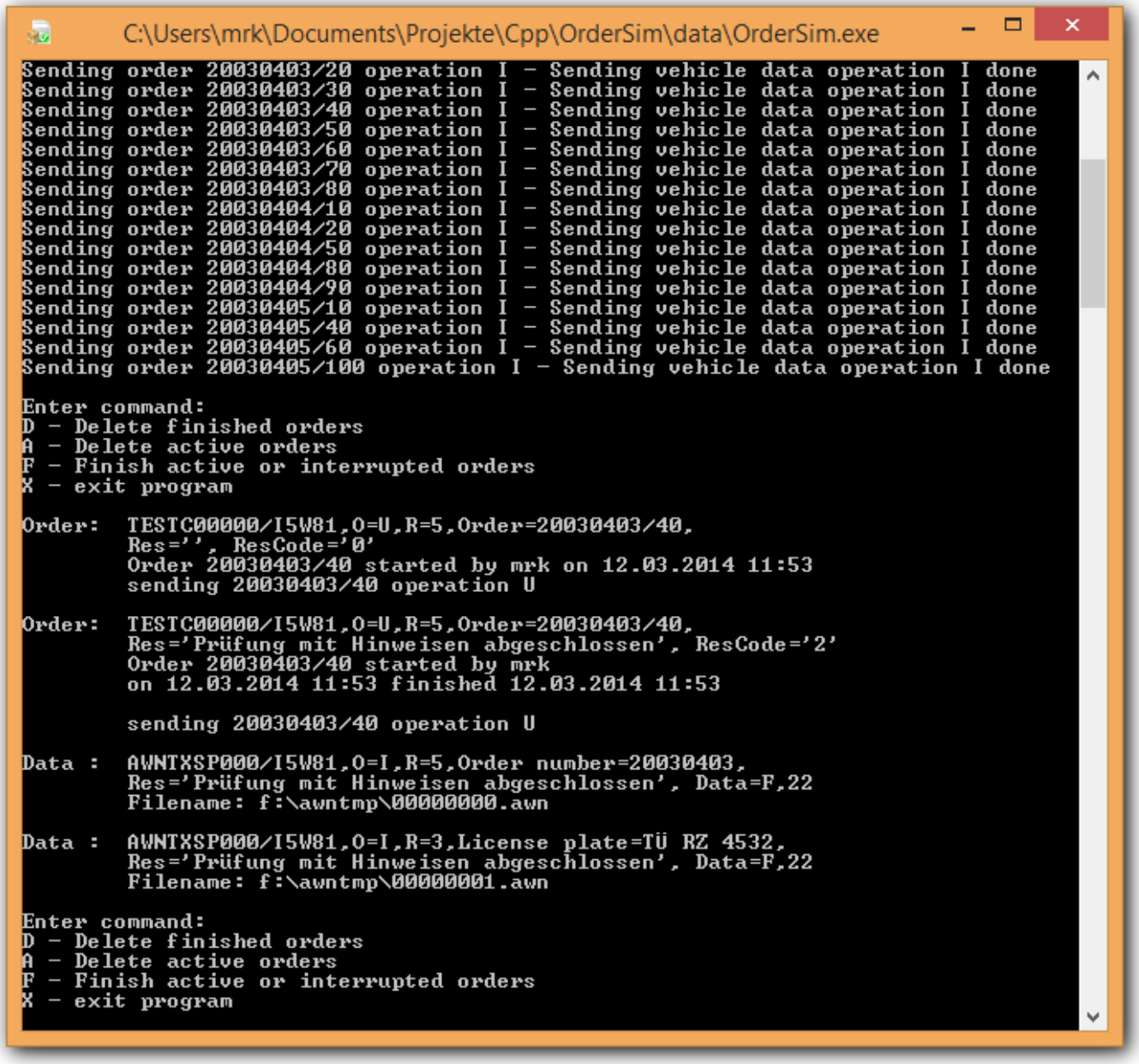
2.3.3 Runtime behaviour

After sending the date described above OrderSim waits for updates of order or extended vehicle data or a user command:



Picture 1, OrderSim display after start

According to the asanetwork protocol for commercial applications, detailed updates are processed and distributed again. Modifications in the order status are displayed. Data received over the data storage service is visualized with notepad.



Picture 2, OrderSim after order processing

In the shown example the order 20030403, position 40 was started and finished. For this order, a testing result with reference 5 and reference 3 respectively was received over the data storage service.

In the first line the respective sending service with DId and DLoc, operation identification and reference, are shown. In the second line the result and the result flag are shown, the third and eventually the fourth line show order start, operator and order end. The last line shows a confirmation about sending the modifications to the other network participants.

For the data storage service, in the third line the path to the received file is shown.

2.3.4 User commands

During the operation time of the program, the user is able to perform a series of tests to check the behaviour of his application.

Command	Effect
D	Deletes all finished order positions and extended vehicle data. The DMS executes this operation after closing the order.
A	Deletes all currently active order positions. Normally this should be prevented by the DMS; nevertheless your application should not show erroneous function.
F	Changes the order status of all currently active or cancelled orders to „finished“. Normally in the DMS this should only happen when order positions are cancelled, nevertheless your application should not show erroneous function.
X	Ends OrderSim and deletes all order positions from the asanetwork. Specify parameter –n to prevent this.

Table 5, OrderSim user commands

3 Testing instruments simulation with TestClient3

TestClient3 is a 32 bit windows application and is used by commercial applications as a simulation for test and checking instruments.

TestClient3 implements the functions of a testing instrument according to the asanetwork requirements.

3.1 Implemented services

Service	Priority	DiQual	DoQual	Application
Client order service	9	1	1	Sending and receiving of order data and extended vehicle data
Checking service(s)	9	1	1	Sending of checking results, login and logout optionally.
Identification data service	9	1	1	Sending of data queries and receiving of identification data, login and logout optionally.
Target data service for exhaust emission checking	9	1	1	Sending of data queries and receiving of target data, login and logout optionally.

Table 6, services implemented by TestClient3

3.2 Configuration of checking services with INI file

TestClient3 is able to support and process any checking service in any combination. They are configured in the file TestClient3.ini.

Note: Configuration data is saved in directory %APPDATA%\AxoNet Software GmbH\TestClient3!

The structure of the file looks, in extracts, like this:

```
[AWN]
DLoc=TestClnt25
Orders=AXONT00000
Results=AWNTXBR000;AWNTXSS000;AWNTXSP000;AWNTXEM000
```

The entry Results in section [AWN] defines all base services supported by TestClient3. For every one of these services an own section must follow. For the side slip service AWNTXSS000 this looks like this:

```
[AWNTXSS000]
Info=Side slip test
```

The Info entry describes the service. This information is transmitted as service description to the network manager at service login.

For services subdivided into further services, like e.g. the exhaust emission which contains more checking processes, an entry SubService with the further service has to be defined.

```
[AWNTXEM000]
SubService=AWNTXEM010;AWNTXEM020;AWNTXEM030;AWNTXEM040;AWNTXEM050;AWNTXEM060;AWNTXEM070
Info=Emission testing
```

For every one of these services an own section with the Info entry must be created.

All services defined in the INI file are announced to the network manager when TestClient3 starts, according to the asanetwork requiremenets. Received order positions containing one of these services as order identification are shown in the order list. A XML testing result is transmitted via the service defined in the order and configured in the INI file. This service is logged in only during the transmission of test results.

3.3 Used data base for test results

For every testing service contained in the INI file a homonymous XML file with the extension .awn exists in the subdirectory Result. For the toe checking AWNTXSS000 e.g. the file name is awntxss000.awn.

Each of these XML files contains a master record for the respective checking or test method. Before sending in the XML file, the TestClient3 replaces order, client and vehicle information with the data of the selected order position. The measured values it selves remain intact.

3.4 TestClient3 functionality

On start, TestClient3 announces his client order service and notifies the checking services to the network manager. An eventually existing order list is imported. The order list can be regenerated performing a query.

Newly received order positions that will show an order DId of a checking service defined in the INI file are incorporated to the order list. If there is an extended vehicle record to one of the order positions, a car symbol is shown before the order number.

Order positions from the order list can be activated, interrupted, cancelled or completed. A result flag can be set with a completed order. For every finished order position an XML test result is transmitted.

Client and vehicle details can be displayed to each order position. An identification or target data query can also be performed here. For this purpose, the respective service is logged in and after performing the query with the reference 4 logged out again.

After Program end the actual order list is saved on the hard disk in the subdirectory Save.

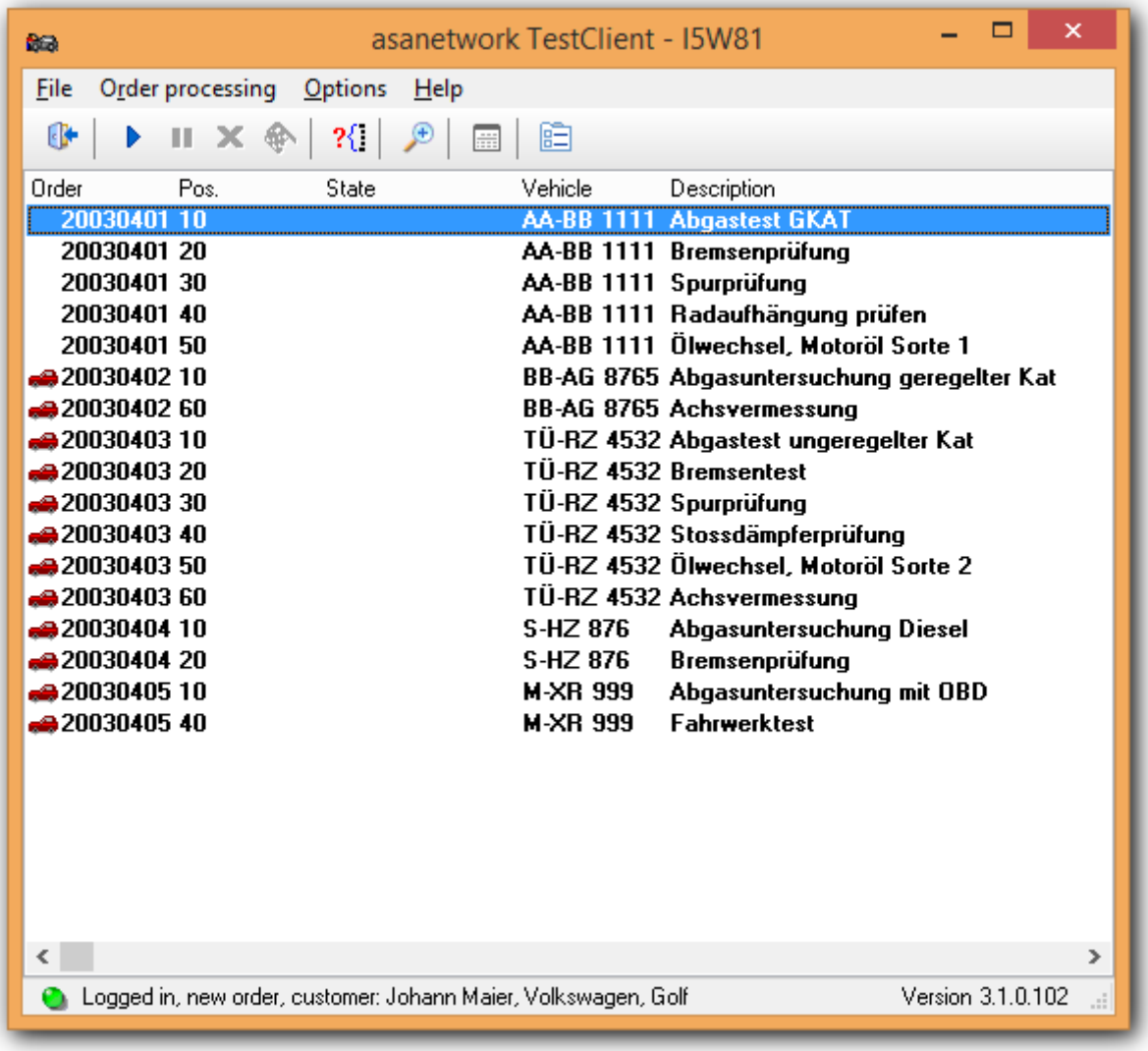
3.5 Application of TestClient3

> Note: If you wish to execute several instances of TestClient3 on the same PC, you need to run each instance with parameter –dloc XXX. XXX is a different DLoc.
>
> You can also create a link which assigns the correct parameter.

TestClient3 starts without specifying parameters and presents itself with its main window.

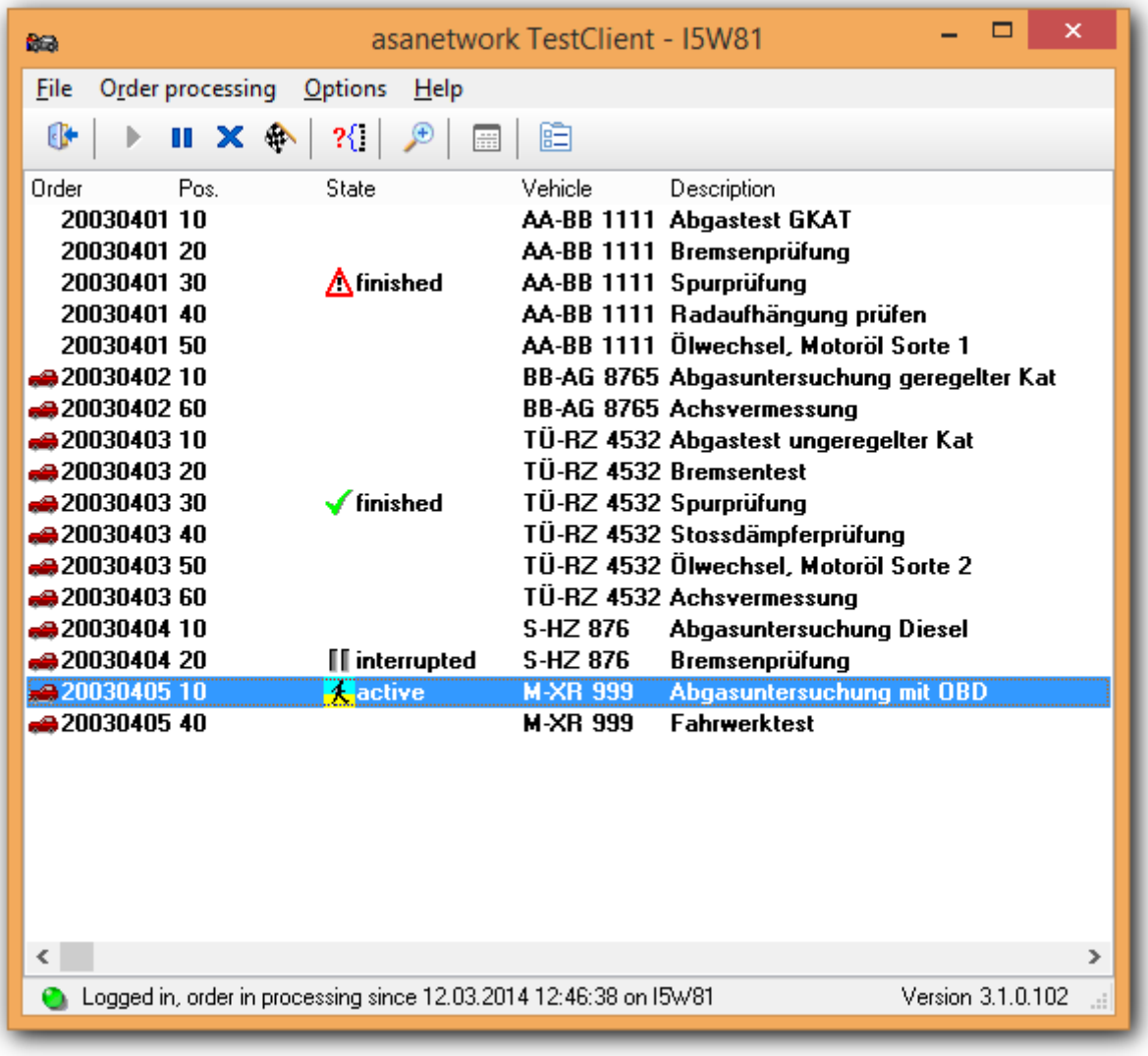
3.5.1 The main window

Besides a menu bar and a tool bar, the main window displays a list of available orders. In the status line connection status, order details and version number are shown.



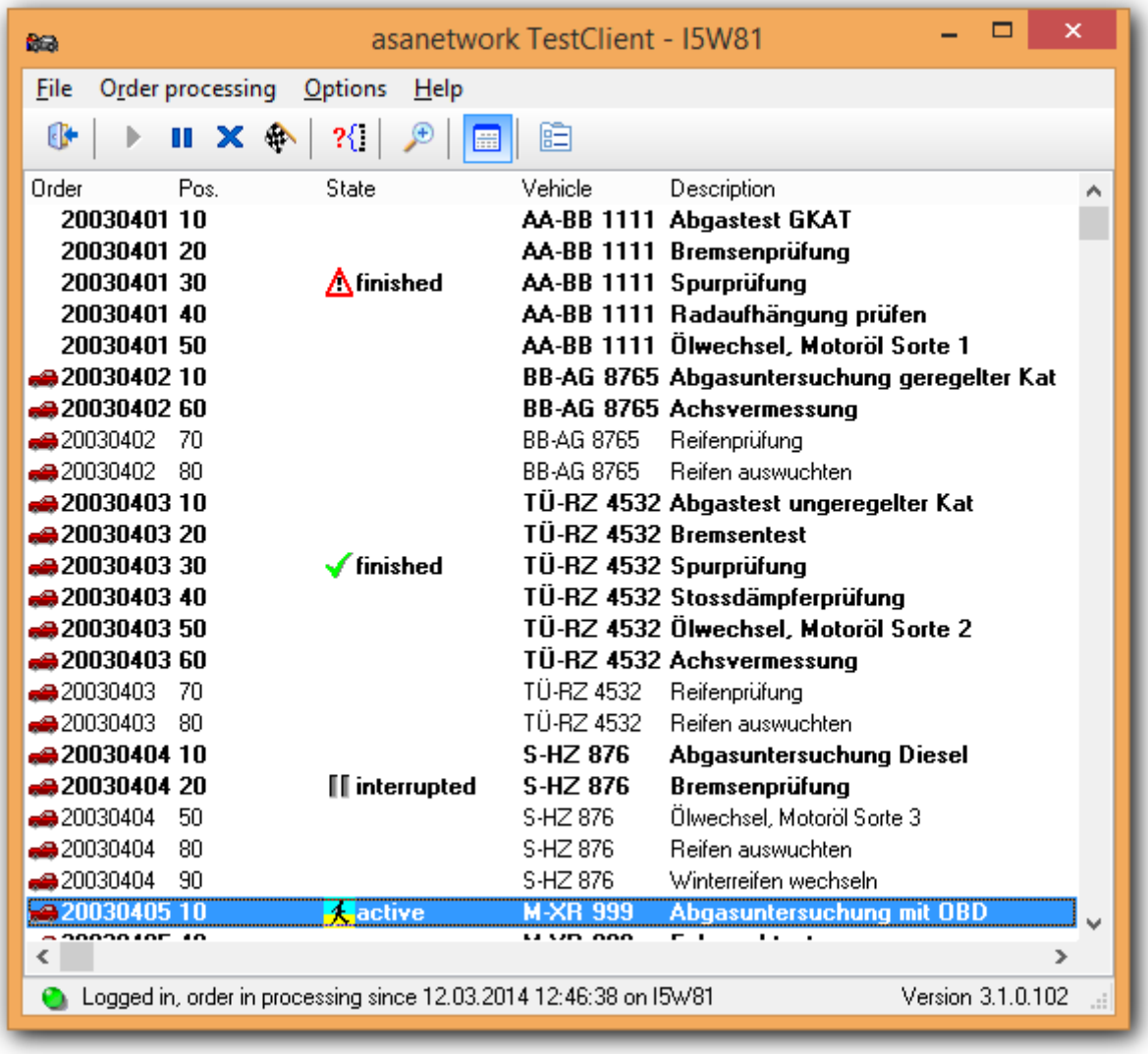
Picture 3, TestClient3 main window

Operations can be started by using the respective menu or by using the toolbar. An order can be started, interrupted, cancelled or completed. The respective status is displayed.



Picture 4, display of the order status in TestClient3

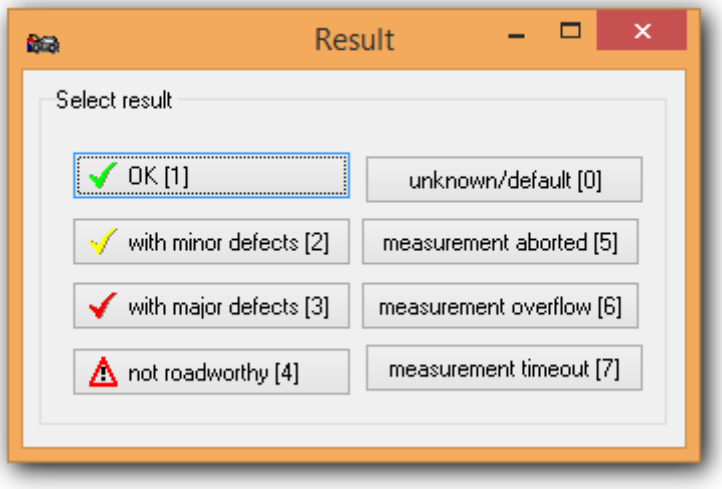
Optionally, all order positions (i.e. also the ones that cannot be selected in TestClient3) can be displayed.



Picture 5, display of all order positions

3.5.2 Result code

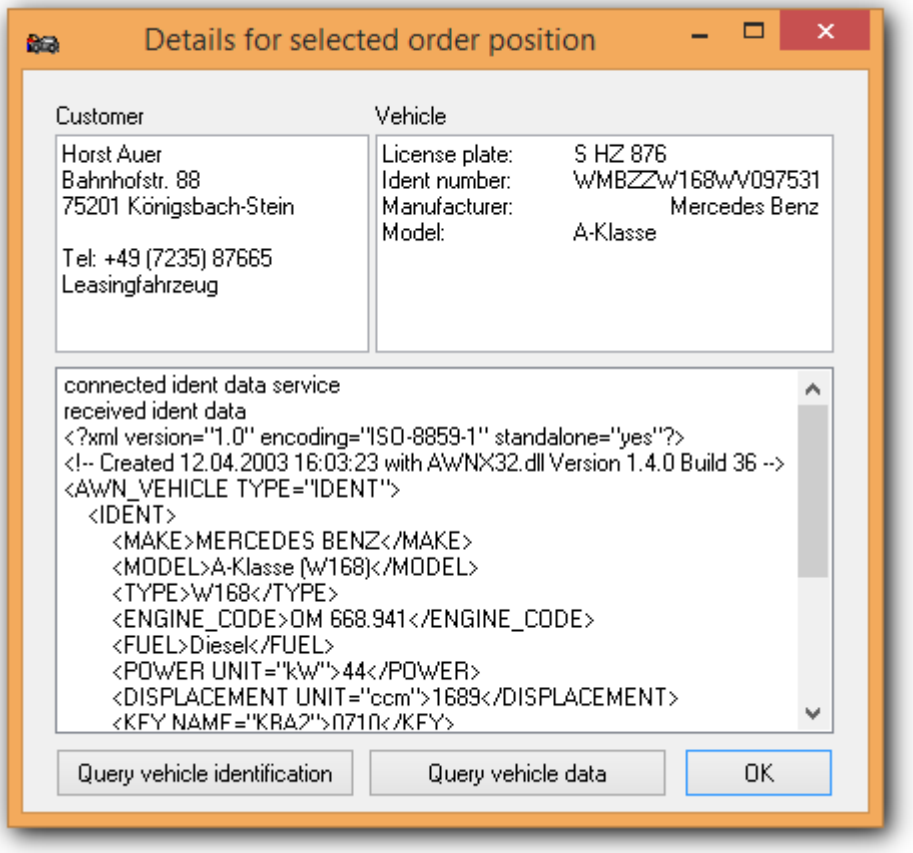
When an order position is completed, the result must be selected. This result is mapped to the RESULT attribute of the XML result.



Picture 6, result code

3.5.3 The detail window

To each order position a detail window can be showed. By clicking the buttons „Query Ident data“ and „Query setpoint data“ you can check if the respective data are available.

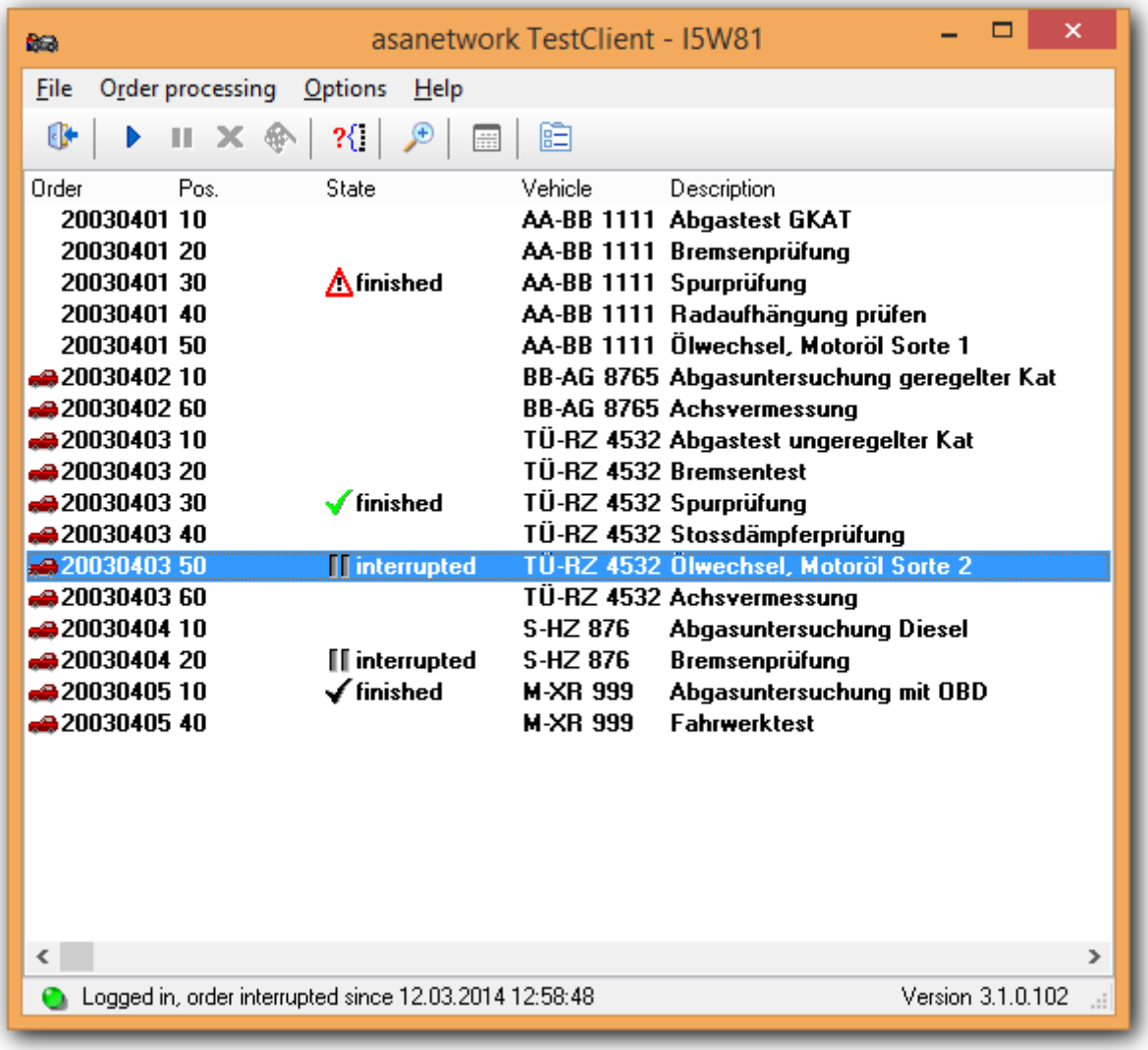


Picture 7, detail window

3.5.4 Oil management handling

All work orders, which include oil dispense (using Did AWNTXOM…) are handled by TestClient3 in accordance with asanetwork requirements for oil management systems.

If such an order position is started, it's automatically moved to state interrupted after 10s. If this is the first dispense, 80% of the nominal value are assigned and transmitted. This position can be started again, the actual value is increased by adding 10% of the nominal value.



Picture 8, Interrupted oil dispense

You can finished an interrupted oil dispense within the 10s time frame. The result will always be set to OK.

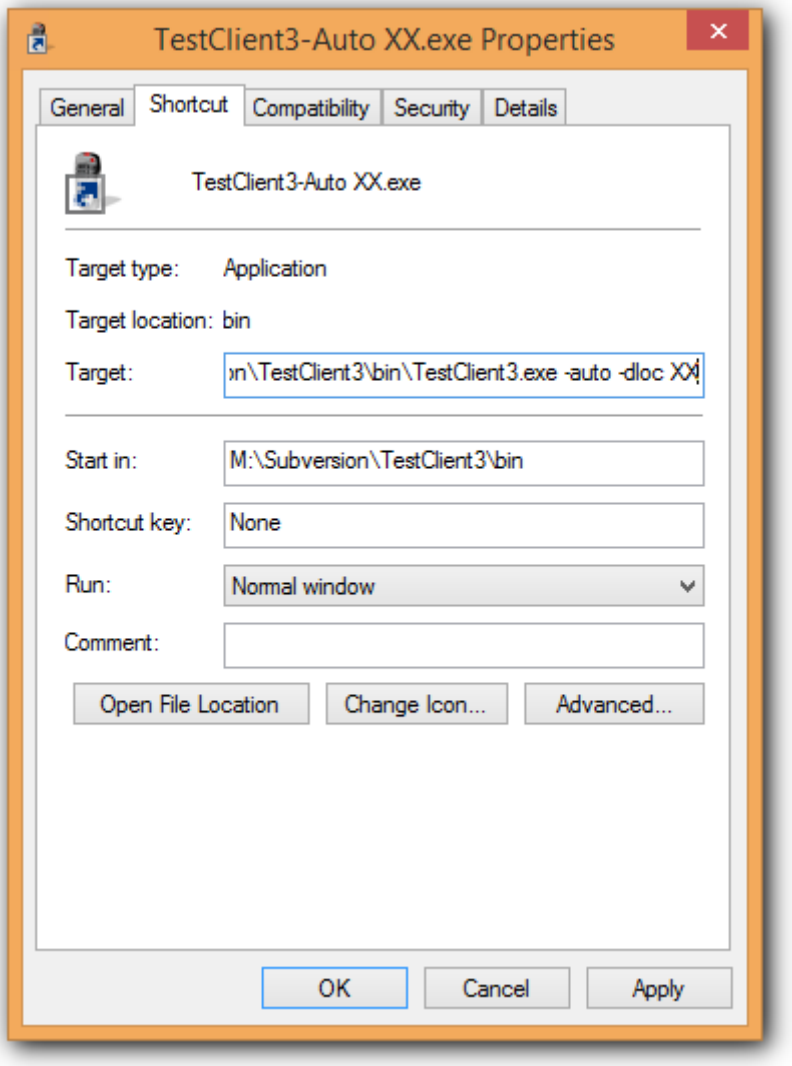
3.6 Automatic operation

TestClient3 can be used automatic testing. In that mode orders are selected randomly and then finished. You have to call TestClient3 with parameter –auto to enable that feature.

If you like to run multiple instances, add different DLocs:

1. Instance: -auto –dloc Test1
2. Instance: -auto –dloc Test2

Simply create two shortcuts and add the parameters:



Picture 9, Shortcut with parameter

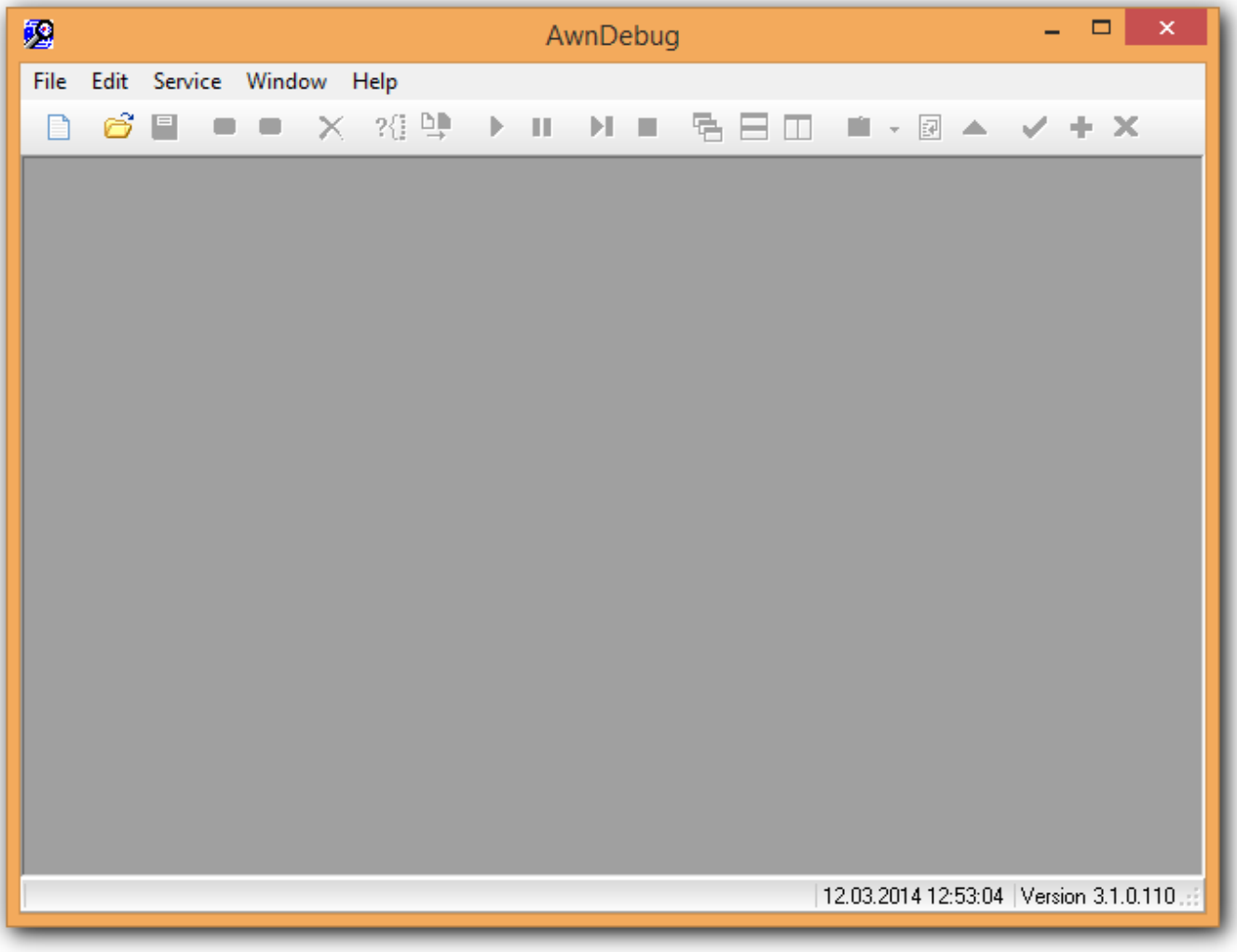
4 Logging with AwnDebug3

The Program AwnDebug3 is a 32 bit windows application for data logging and analysis in asanetwork.

AwnDebug3 is able to log any service classes as well as specific services.

4.1 Using AwnDebug3

AwnDebug3 starts without specifying parameters and presents itself with its main window.

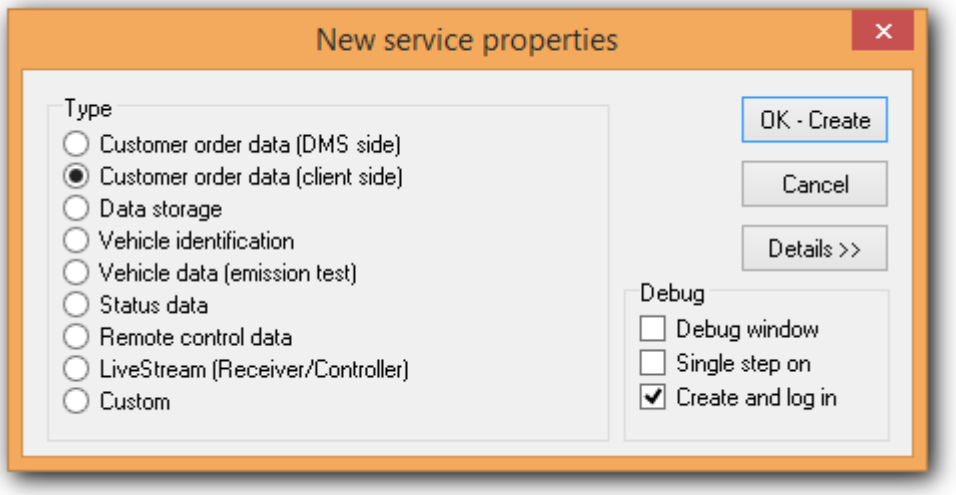


Picture 10, AwnDebug3 main window

Starting here you can either import and analyze or modify saved data or select a service for logging.

4.2 Creation of a new service for logging

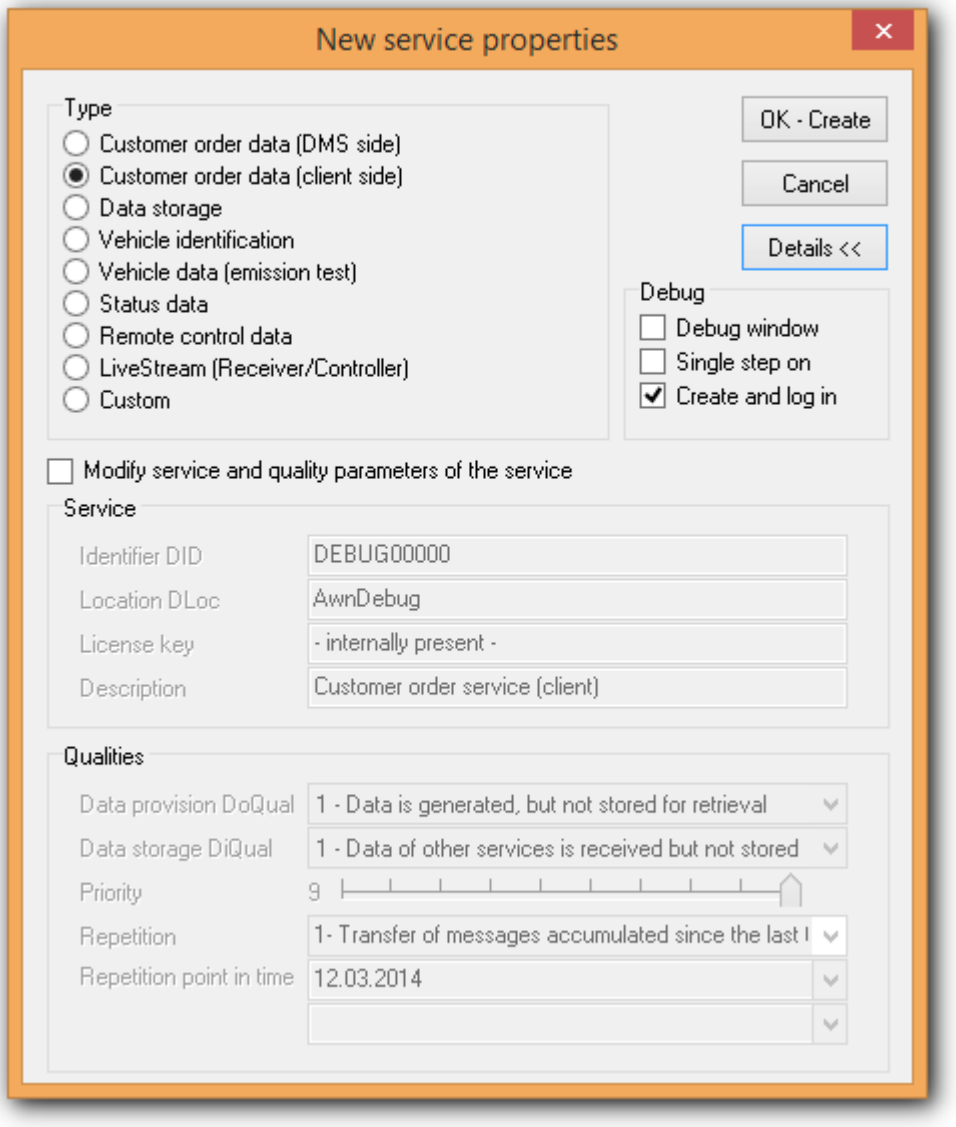
Click on the “New” symbol or select menu “File/New”. A window for setting service parameters appears.



Picture 11, standard service parameters

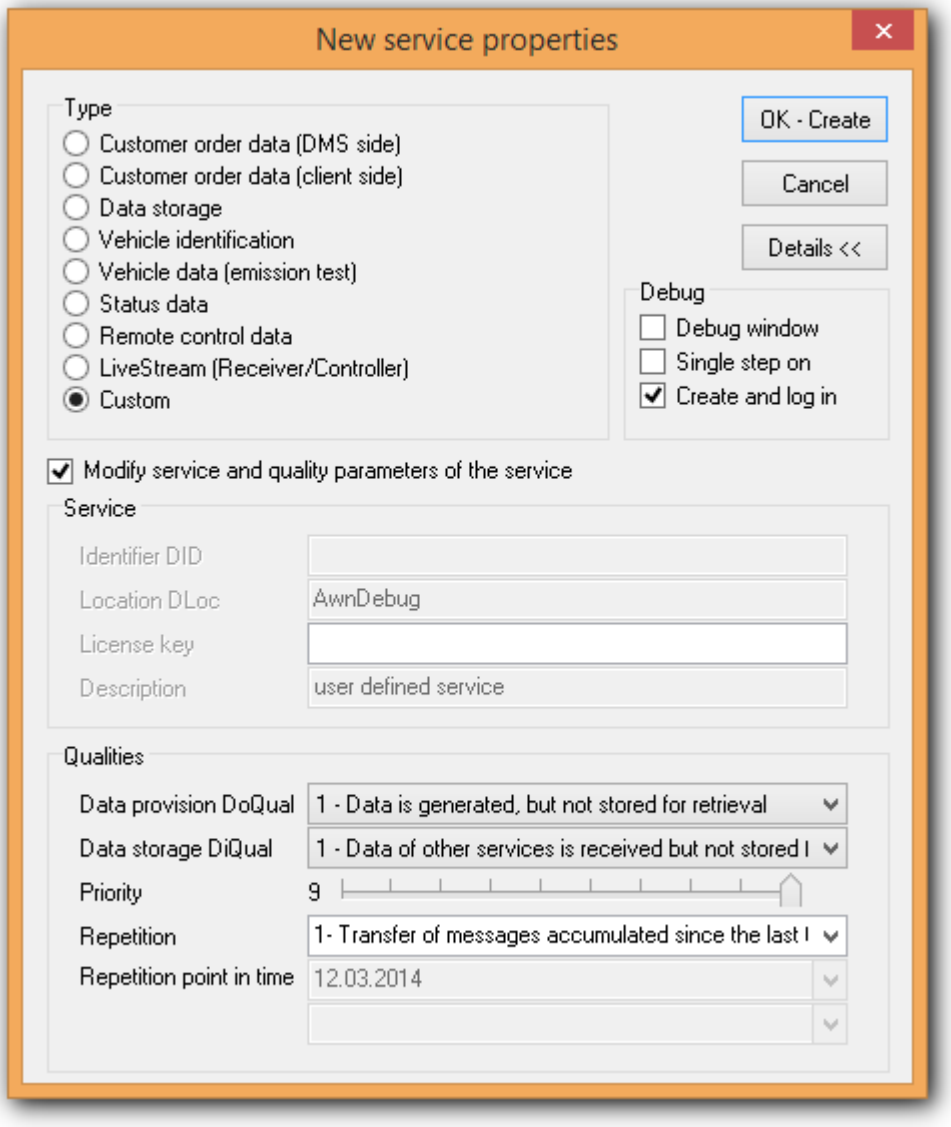
You can choose between the offered services, which are created with reasonable standard pre-settings. The debug parameters contain the display of the debug window from Awn2_32.dll, the activation of the Single Step operation modus (see there) as well as the possibility of logging in the service immediately.

If you wish to change the pre-settings, click on “Details” >>.



Picture 12, extended service parameters

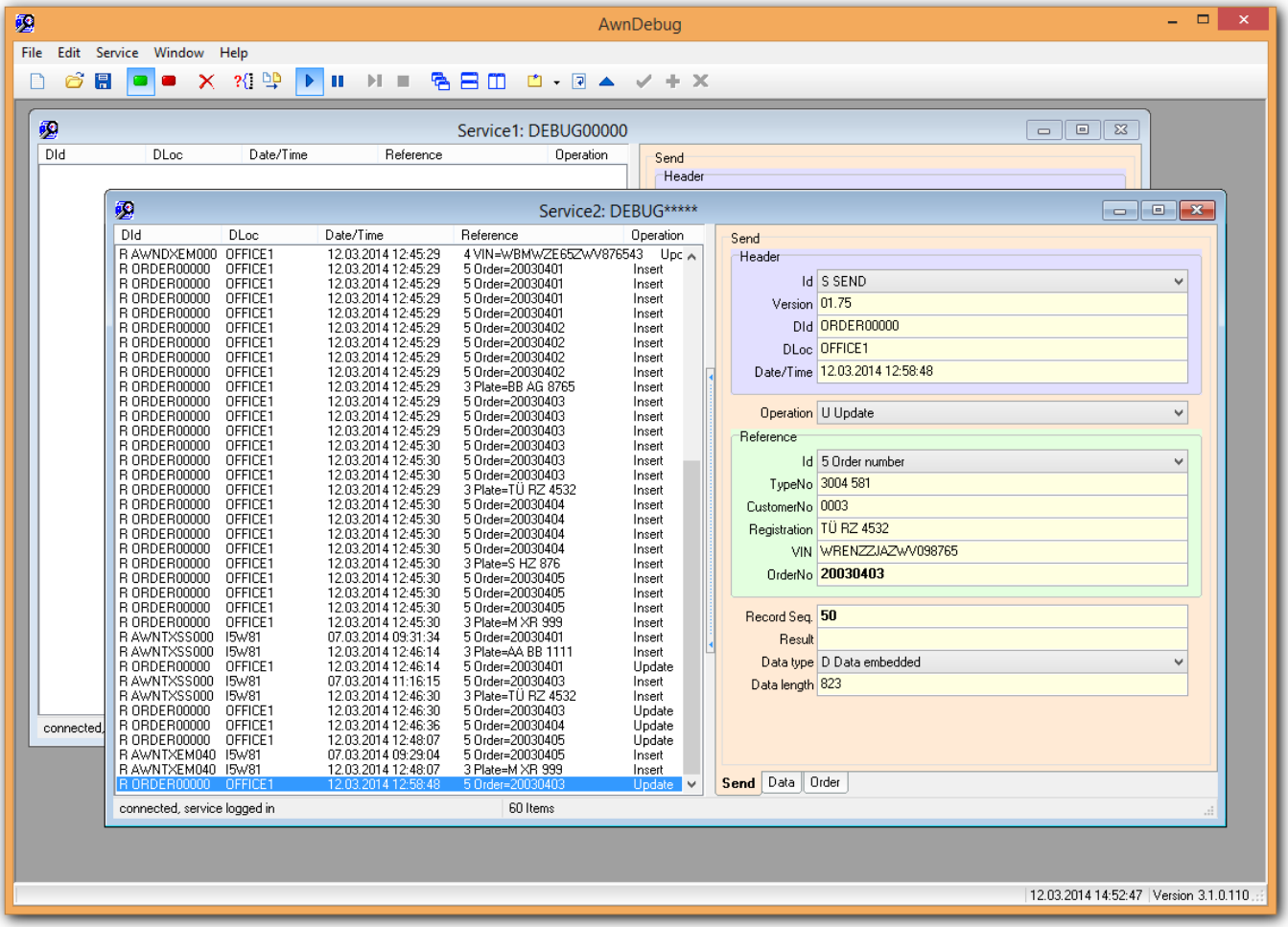
You see the pre-set values now. After checking “Modify service parameters” you can edit the service parameters. If you select the client specific service, these parameters are offered immediately for entering and you must enter the service name and the corresponding license key.



Picture 13, client specific service parameters

4.3 The service window

After clicking “OK” the new service is created and the service window is shown. You can create any number of additional services. You can manage the order of the windows with the menu “Windows” or with the toolbar symbols cascade, horizontal or vertical segmenting respectively.



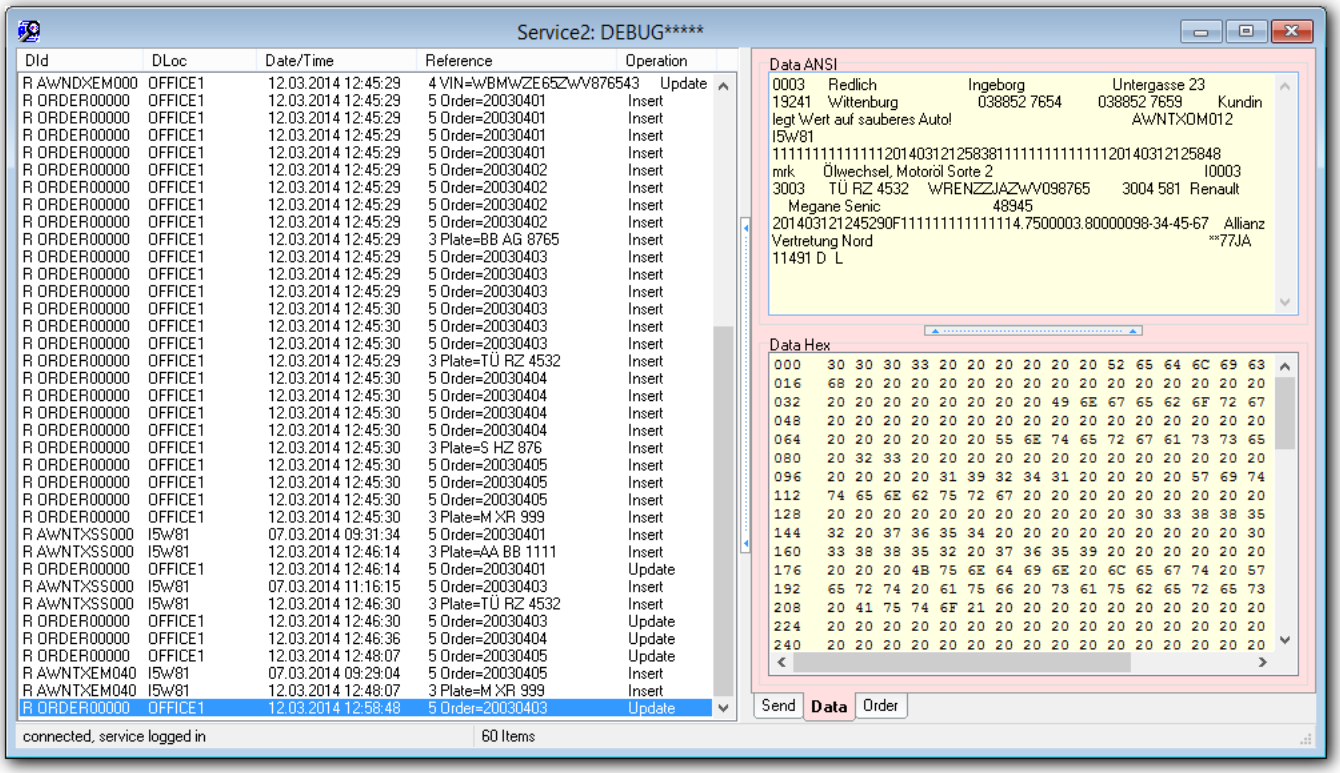
Picture 14, AwnDebug3 as MDI application with several service windows

4.4 The service window in detail

After its creation the service is logged in by default. Every service can be logged in or out using the menu “Service”. Alternatively this can be done using the toolbar symbols logout service or login service.

In the left part of the window, all received records are listed. For every record a flag (R – received, S – sent, empty when edited or modified), the DId and DLoc, the reference and the operation is displayed. They are ordered by time of reception. Using a local popup menu, single records or all of them can be deleted.

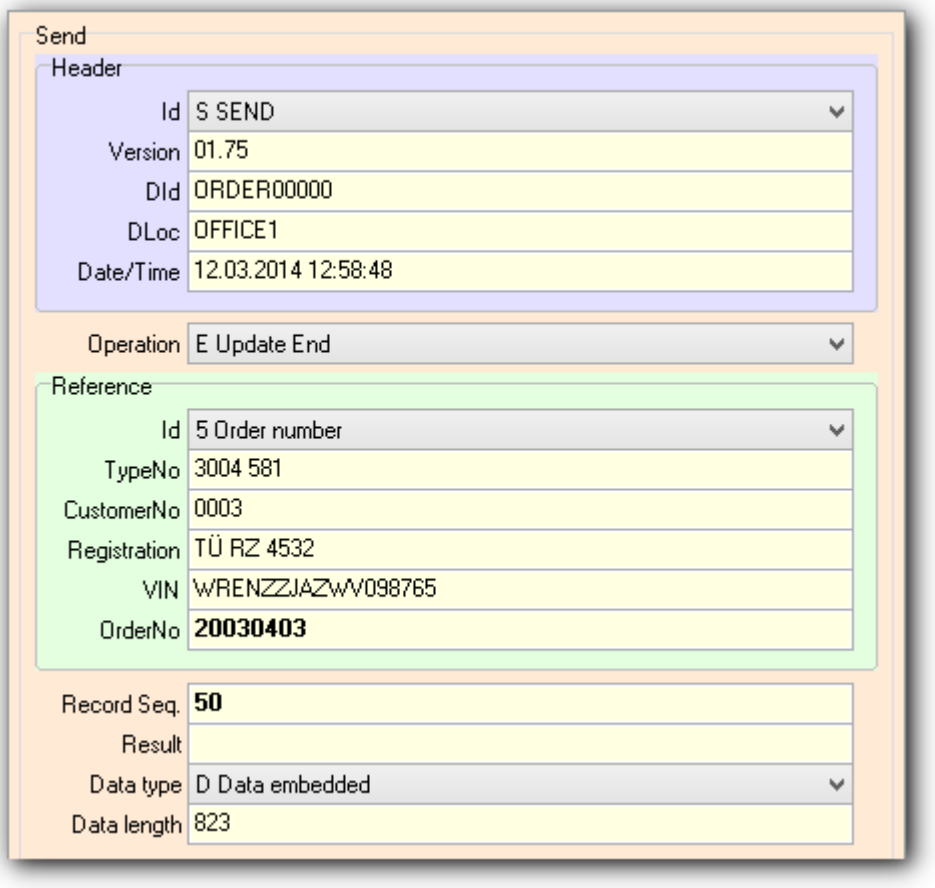
In the right part of the window, the details for the actually selected record are shown.



Picture 15, the service window

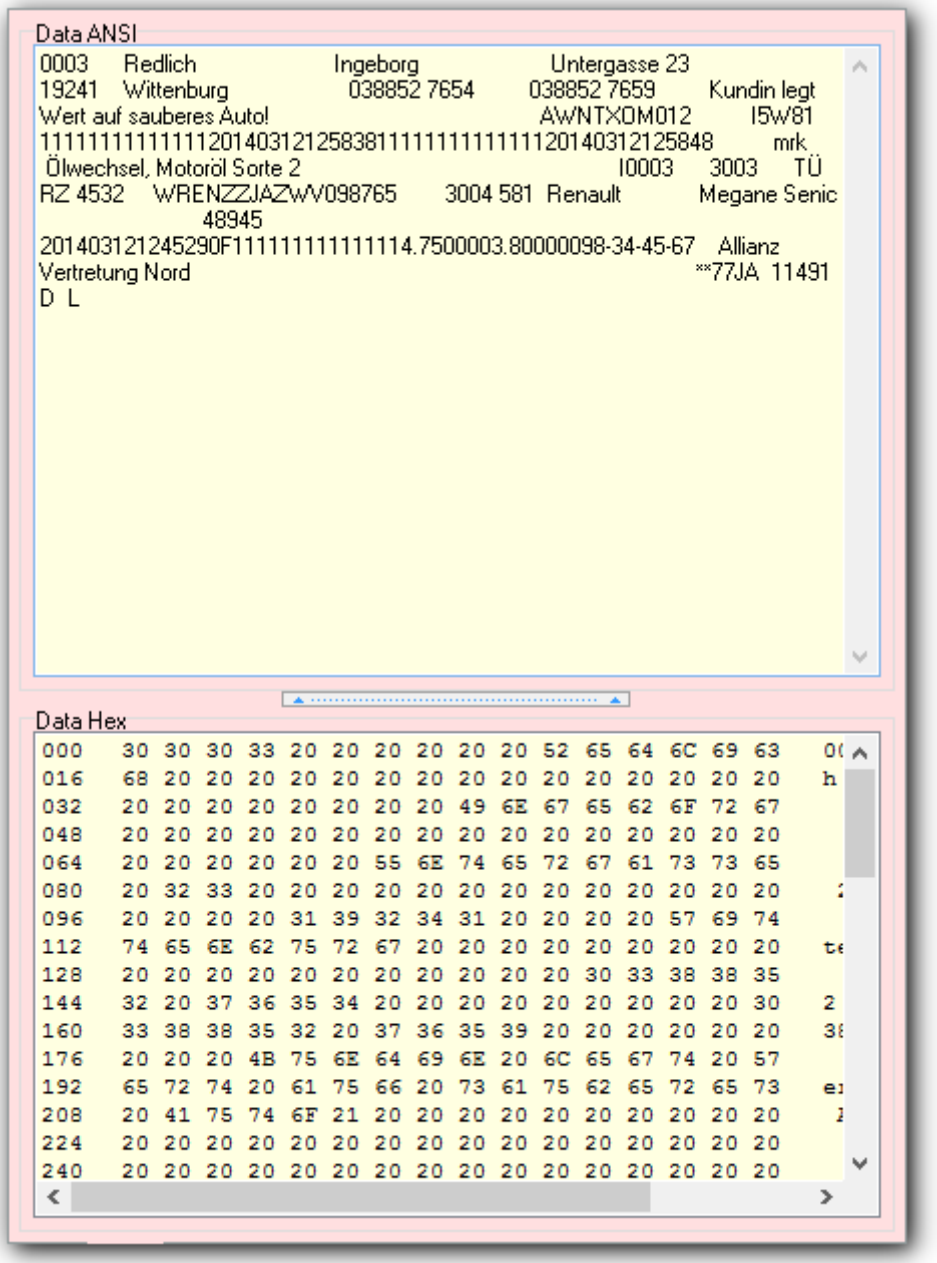
4.4.1 Data display and analysis

The selected record on the left is shown on the right side on different tabs. The first tab “Send” shows the header, reference and send data. The actual reference is marked with a bold font.



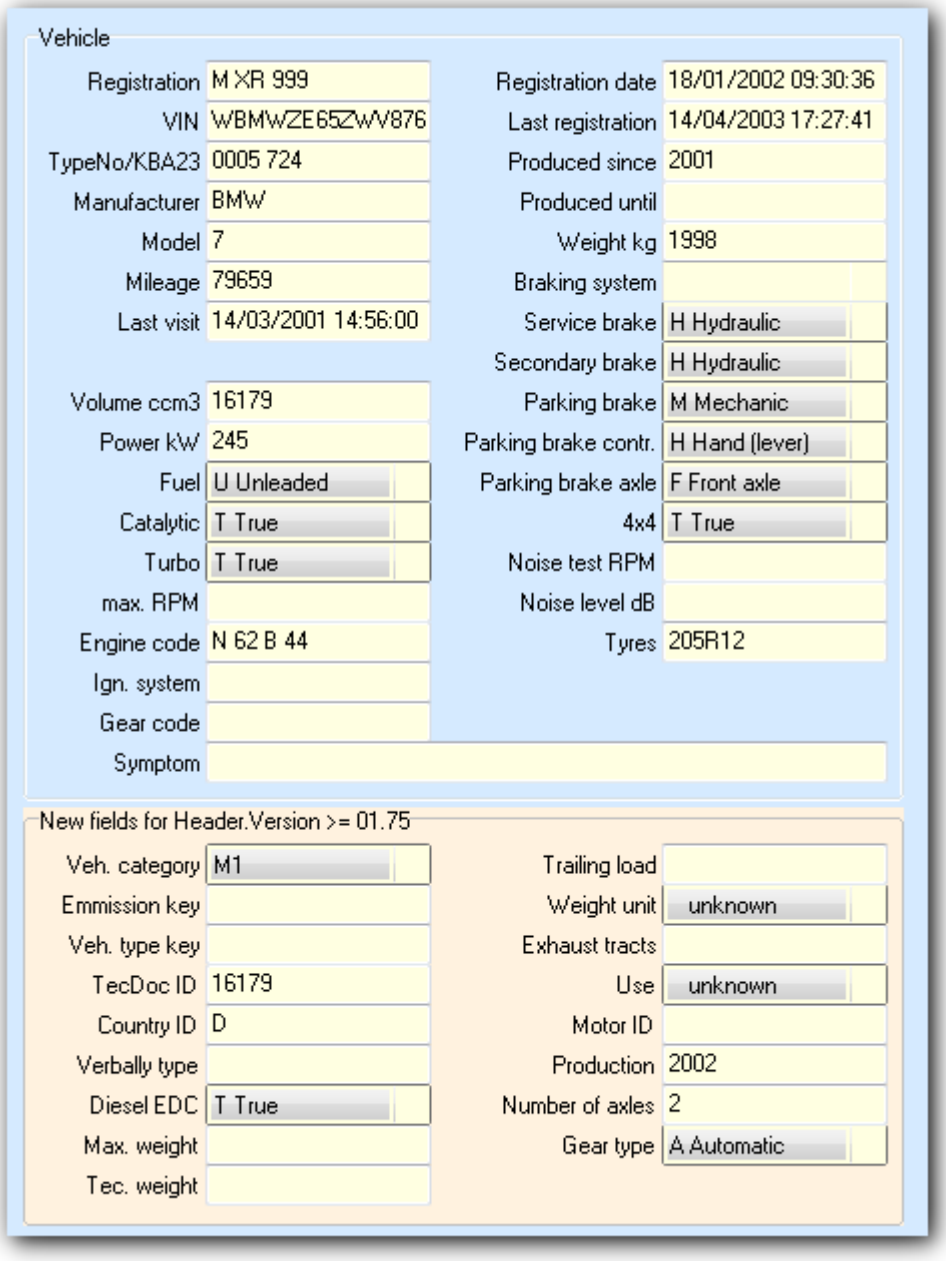
Picture 16, display send data

The second tab “Data” shows the raw data in ASCII format and as HEX Dump.



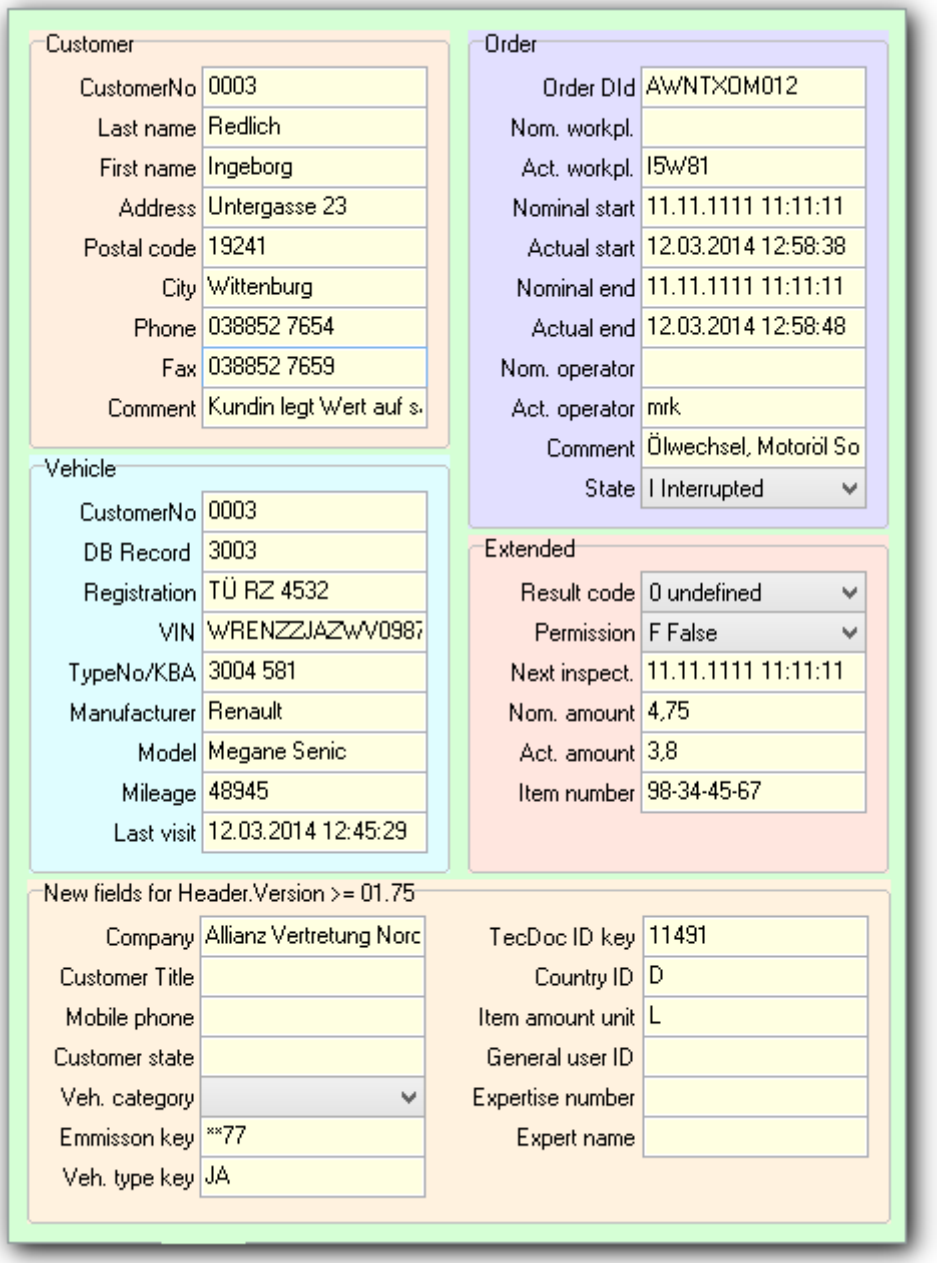
Picture 17, display raw data

For extended vehicle records, the tab “Vehicle” shows the vehicle details.



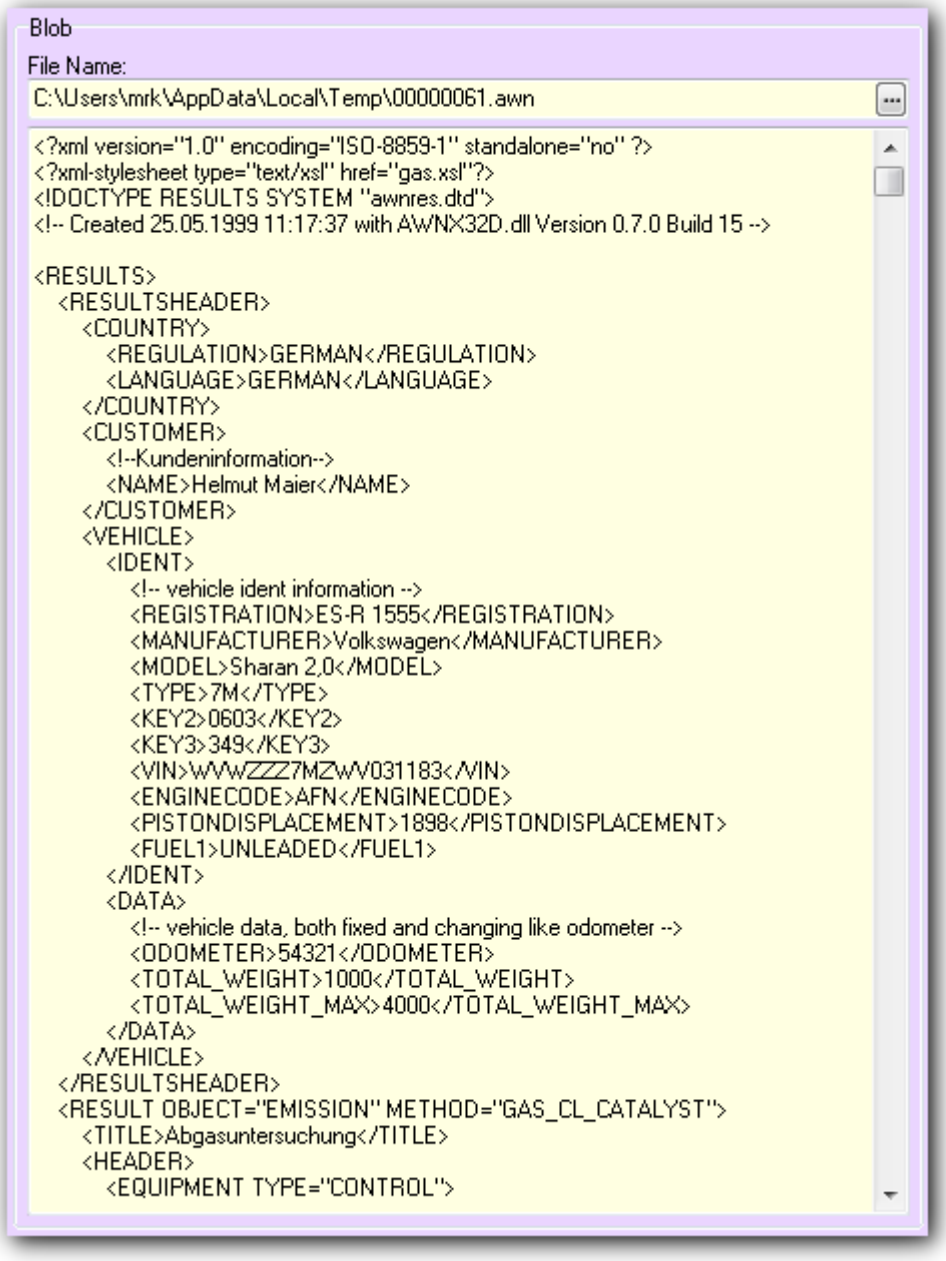
Picture 18, display extended vehicle data

Accordingly, for orders, order details are shown.



Picture 19, display order data

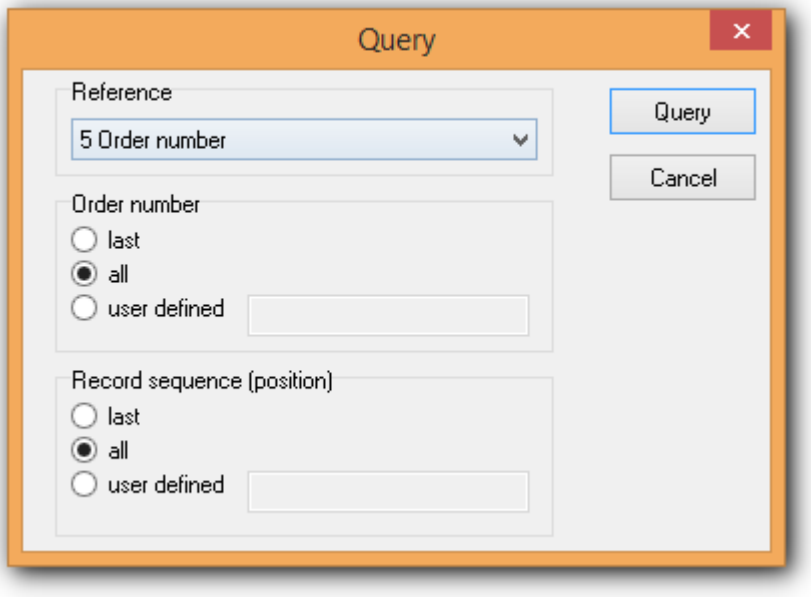
If the received data contains a file, it is displayed on the tab “File” together with the file name.



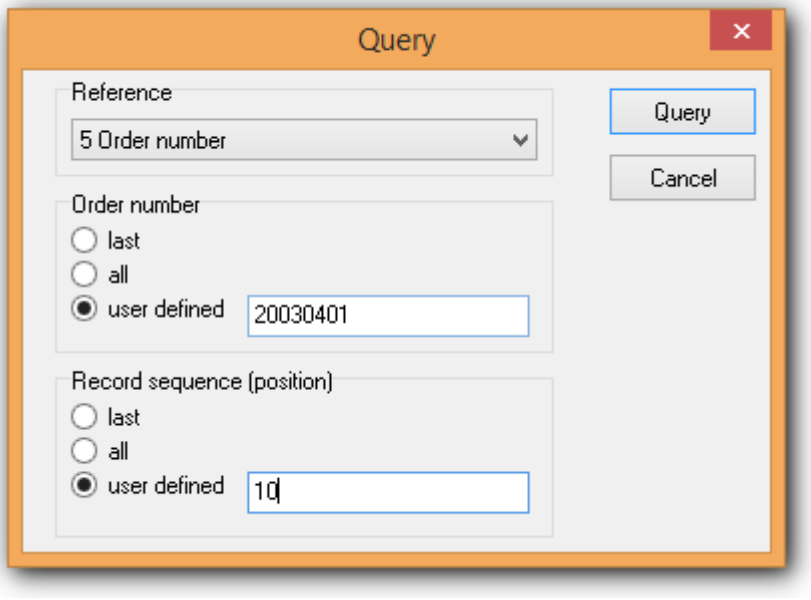
Picture 20, display file data (blobs)

4.4.2 Query of data

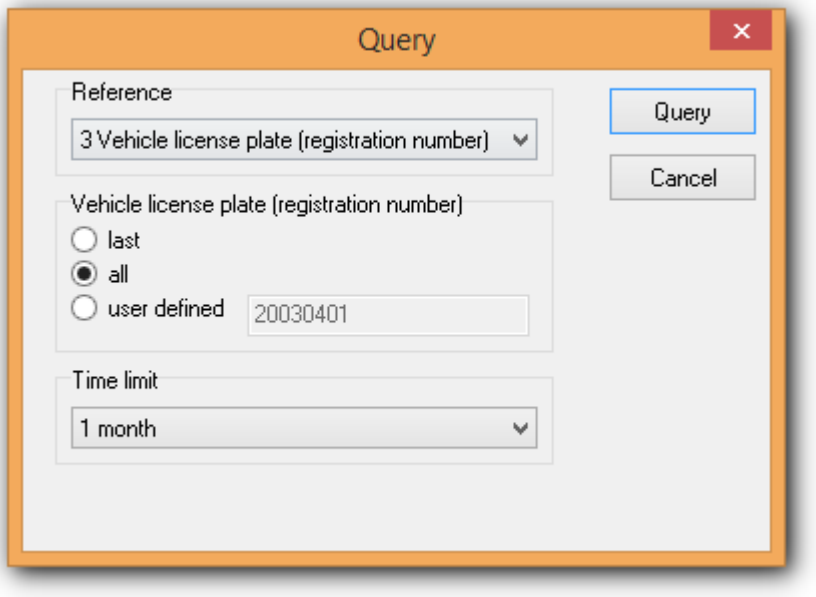
Using the service menu or the tool button ?{ a query of data can be started any time. A dialog window is shown, where you must select reference and enter reference data.



Picture 21, query of all orders



Picture 22, query of a specific order position



Picture 23, query of data with reference license plate, max. 1 month old

4.4.3 Sending of data

The particular selected record can be sent using the service menu or the tool button Send. The sent entry is sent with the DId and DLoc of the service logged in and is added as a new record with flag S.

4.4.4 Creating / modifying of data

A new record can be created using the editing menu “New” or the tool button Create. The newly created or another selected record can be changed to processing mode with the editing menu or the tool button edit. All changes in the respective tab are applied after executing Save. Alternatively, the modified record can be stored with as a new record. Modifications can be cancelled by clicking the button .

If you made changes on a tab and now are trying to pass to another tab, you are asked to save, add or delete the data.

4.5 Single Step operation mode

Normally, the service logged in automatically sends a “ready for receive” after receiving a record. By activating the Single Step operation mode on the service menu or by using the tool button no automatic “ready for receive” is sent. By clicking on the next record can be received. A click on ends the Single Step operation mode, while a click on Stop notifies the service online, i.e. outstanding data remain without transmission.

4.6 Storage of received or created data

As soon as records exist, using the file menu or the tool button all shown data can be saved in a new file.

4.7 Import of saved data

By using the file menu or the tool button Open, saved records can be loaded for analysis. In this operation mode no service is logged in and no data can be sent. Also, file attachments (blobs) are not automatically deleted if the record is deleted or the window is closed.

4.8 Typical use cases for AwnDebug3

4.8.1 Logging of data exchange between DMS and testing instrument (orders)

Please create a client order service (DMS page). The client order service logs all data exchanged between the DMS and the testing instrument(s) and/or applications.

The trace can be saved in a file and sent e.g. by email to be examined by Support. Support can import the file in AwnDebug3 and analyze it comfortably.

Limitations:

Queries of checking instruments are by default not logged, because DiQual/DoQual is set on 1.

> If changed to 2, incoming queries are logged, but this can maybe influence the behaviour of the application because a query is not transmitted afterwards.
>
> Answers to the query from DMS cannot be logged, because they are sent directly to the query-maker. If necessary, this can be tested by starting a query from AwnDebug3.

4.8.2 Logging of data storage (e.g. results)

Please create a data storage service. The data storage service logs all data traffic which is not sent only to DMS or other receivers because of priorities. Typically, a trace of sent testing results can occur with this.

> The statements from the chapter above are also valid for the evaluation and the limitations. While saving, the trace is saved in a file and every blob in another file with the same base name. For later evaluation all files must be available.

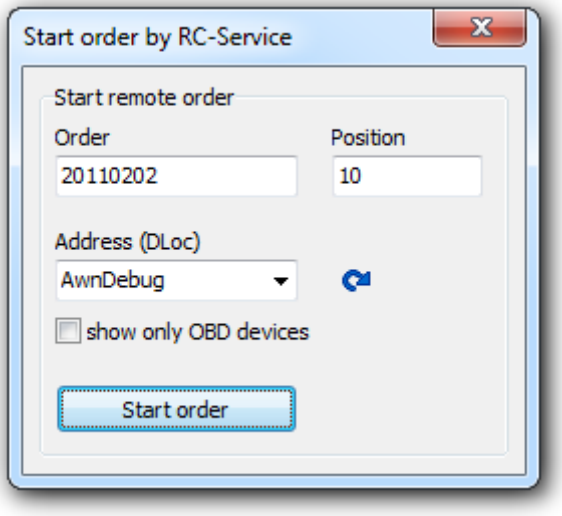
5 Remote start of orders with AnwRemote

AwnRemote enables the sending of start commands to start an order on another instruments. Internally, the service AWNAXRCORD is used for this purpose.

Tip: AwnDebug3 permits the logging of this data.

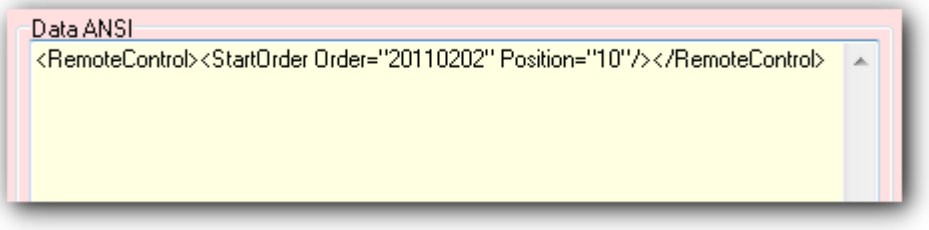
5.1 Using AwnRemote

After starting AwnRemote the order number and position must be entered. Then the receiver DLoc is set, because here direct addressing is used.



Picture 24, AwnRemote

AwnDebug3 can show the commands as XML data:



Picture 25, AwnRemote data

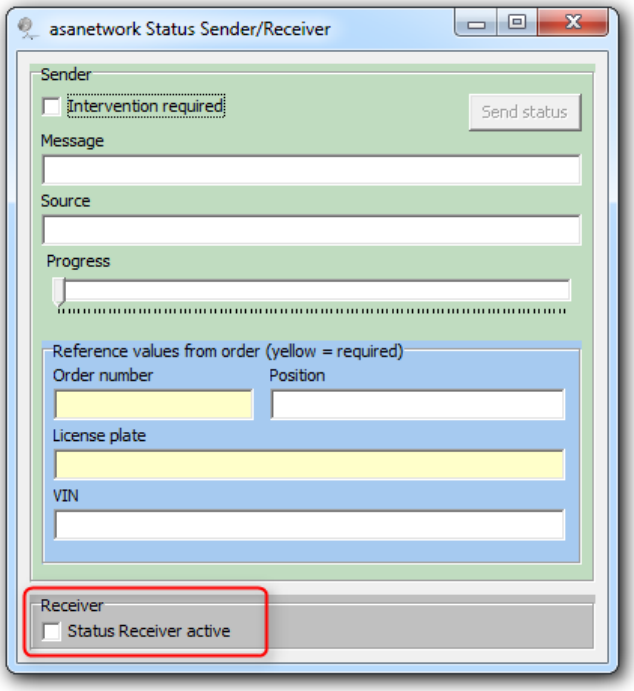
6 Display status information with AwnStatus

AwnStatus allows sending and receiving of status information. Status information can be transmitted by checking instruments to notify the status of enduring operations to commercial software.

AwnStatus can be used both as receiver and as sender.

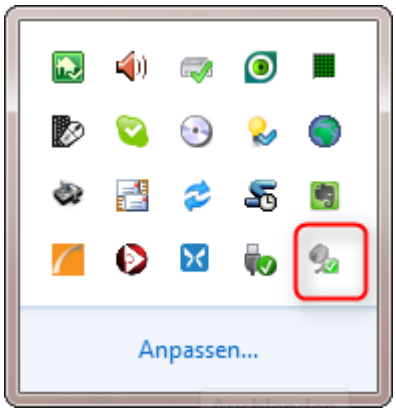
6.1 Using AwnStatus as status receiver

To use AwnStatus as a receiver, the checkbox below must be activated.



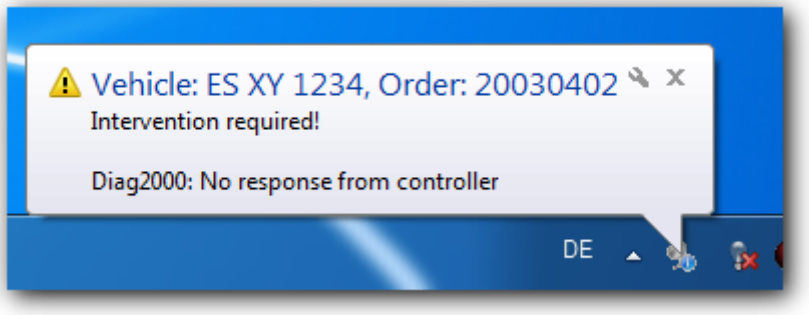
Picture 26, AwnStatus

In the tray bar the icon is now visible with a green check mark.



Picture 27, AwnStatus receiver active

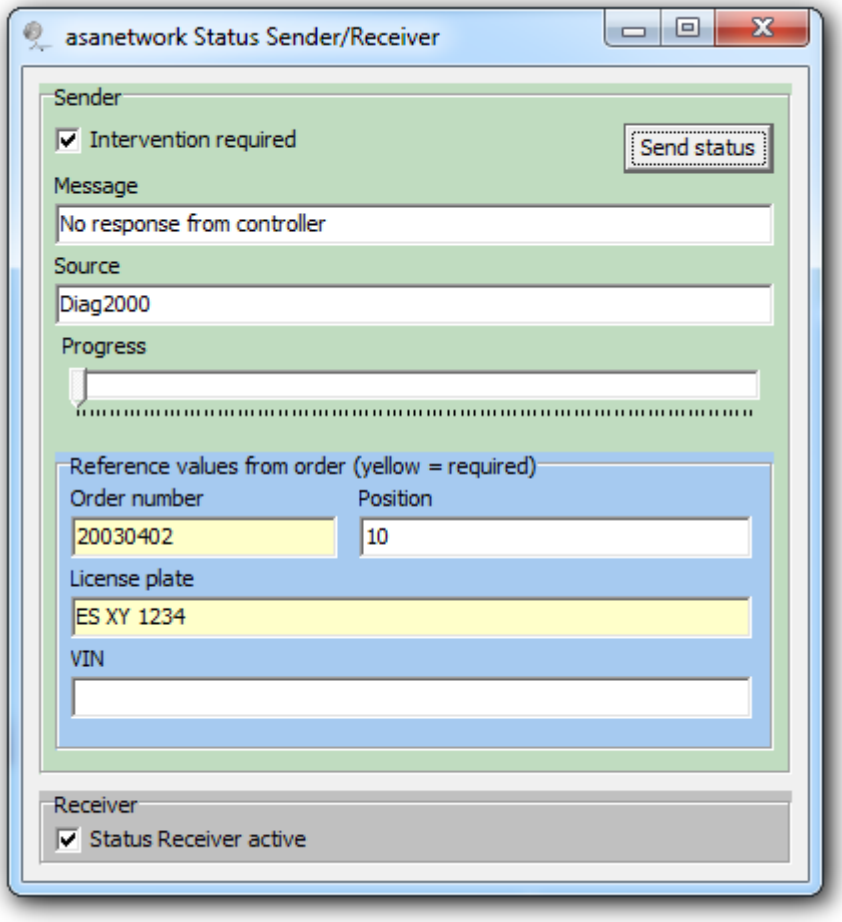
Received notifications are displayed as balloon messages:



Picture 28, received status notification

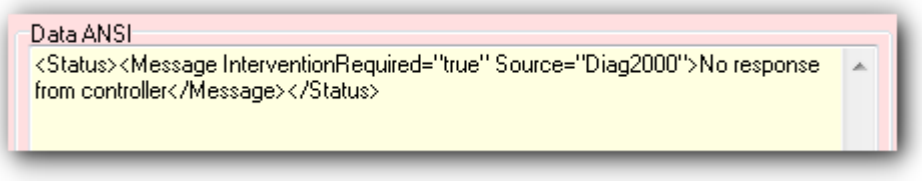
6.2 AwnStatus as status sender

In order to send a status notification, at least the order number and the license number must be entered:



Picture 29, sending of status notifications

Same as above, with AwnDebug3 the data content can be traced:



Picture 30, AwnStatus data

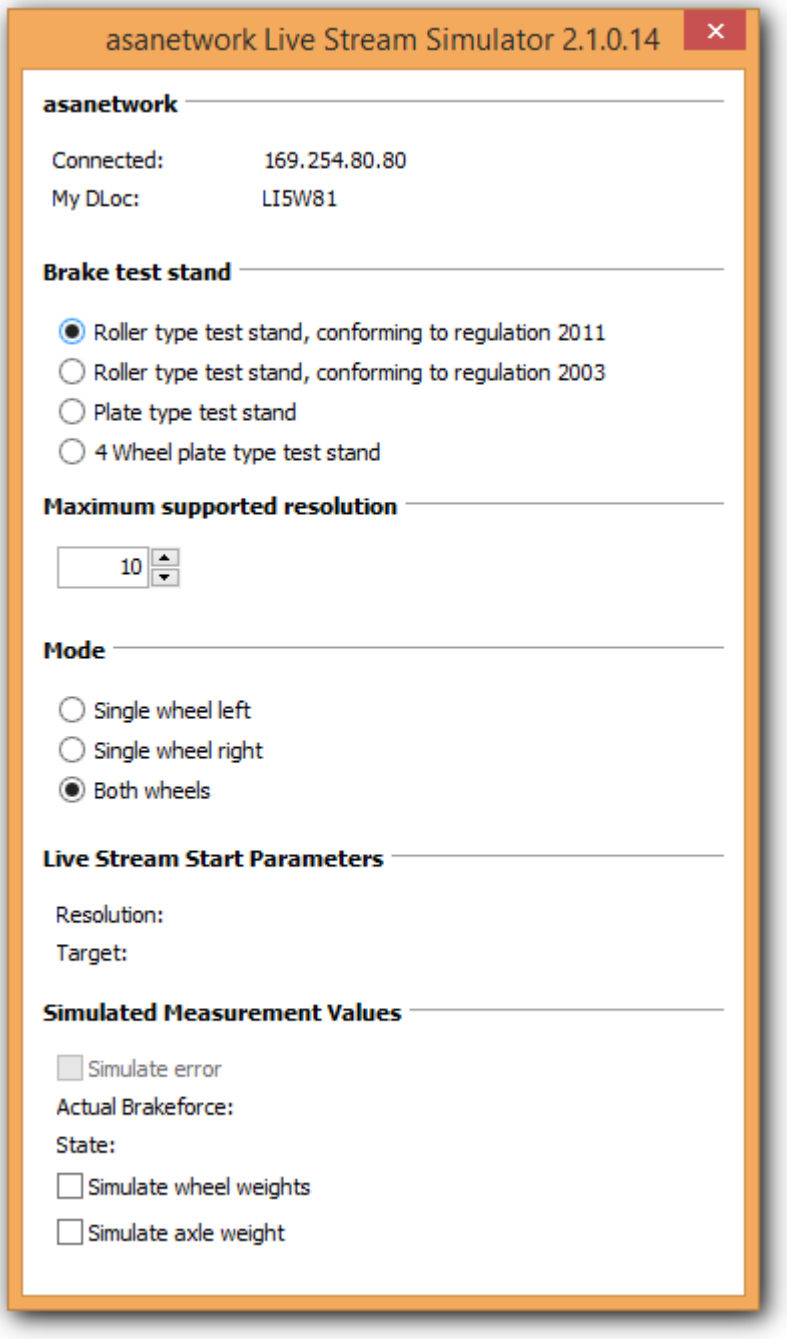
7 Transmission of real time values with LiveStreamSimulator

LiveStreamSimulator implements data transmission for a brake bench which after request sends continous measuring values.

The simulated measuring values rise from 0 to approximately 1500N and then stay fluctuating around this value.

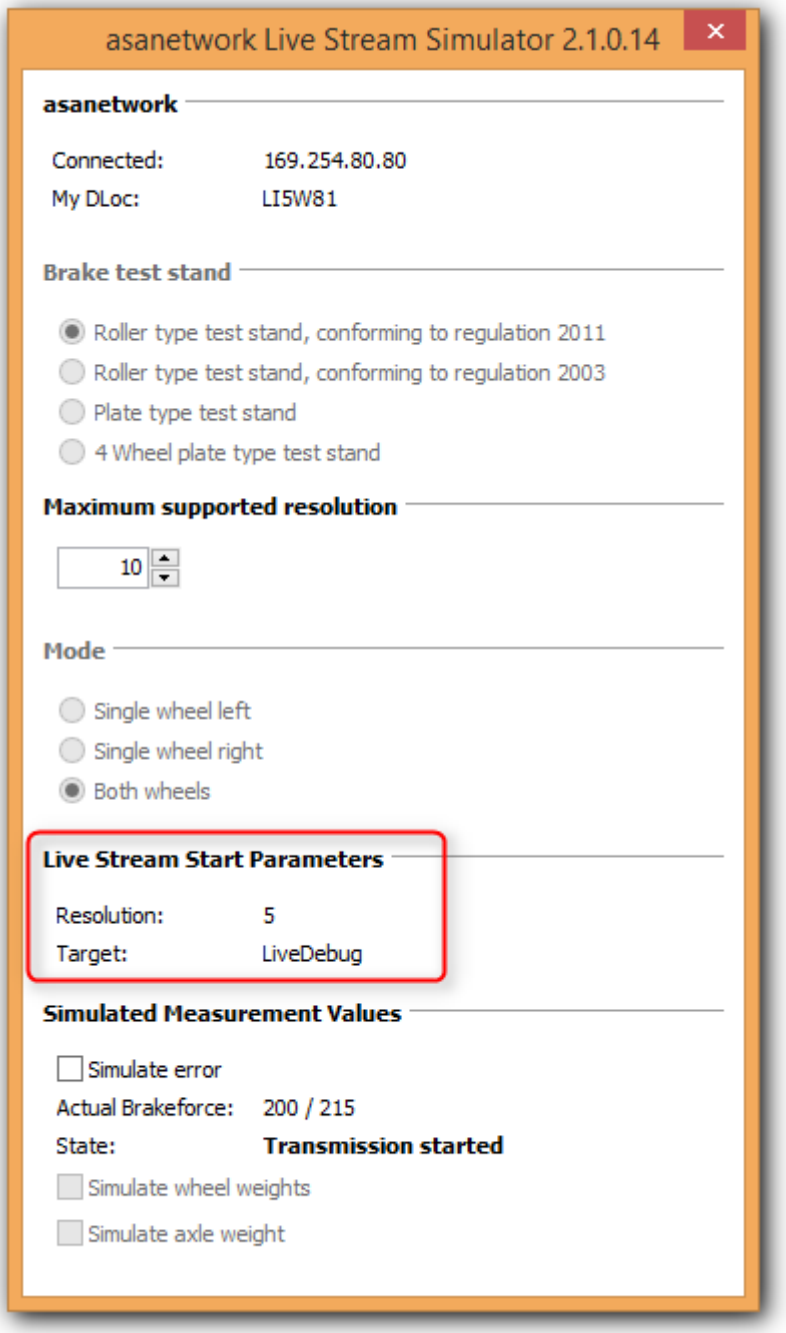
7.1 Using LiveStreamSimulator

The simulator runs without further configuration, same as a brake bench. For simulation purposes, you can select the brake bench type and between single tire measuring left/right or axis measuring before the start of transmission. Also, at runtime errors can be set and transmitted.



Picture 31, LiveStreamSimulator in idle state

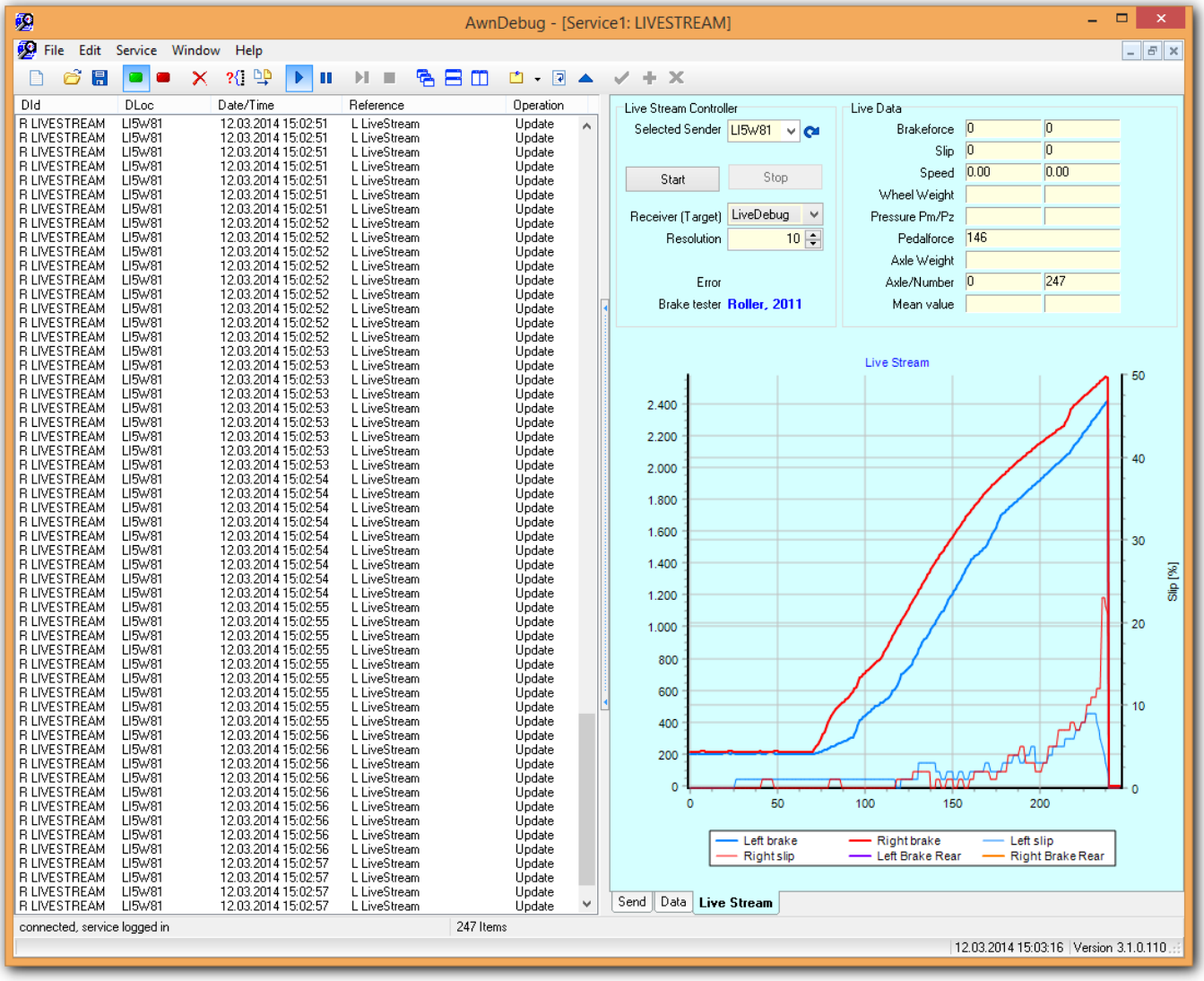
The simulator displays the receiver of the measuring values and the required resolution. Permitted values for the resolution are between 1 – max. resolution. Any request for a higher resolution will create an error.



Picture 32, LiveStreamSimulator running

7.2 Using LiveStream controller in AwnDebug3

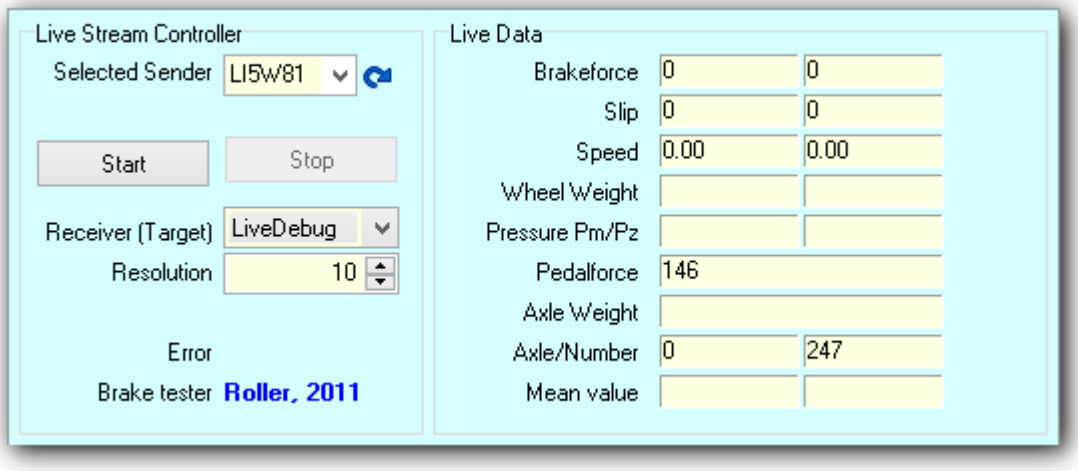
AwnDebug3 includes a LiveStream controller and display. Measuring values are shown graphically:



Picture 33, LiveStream support in AwnDebug3

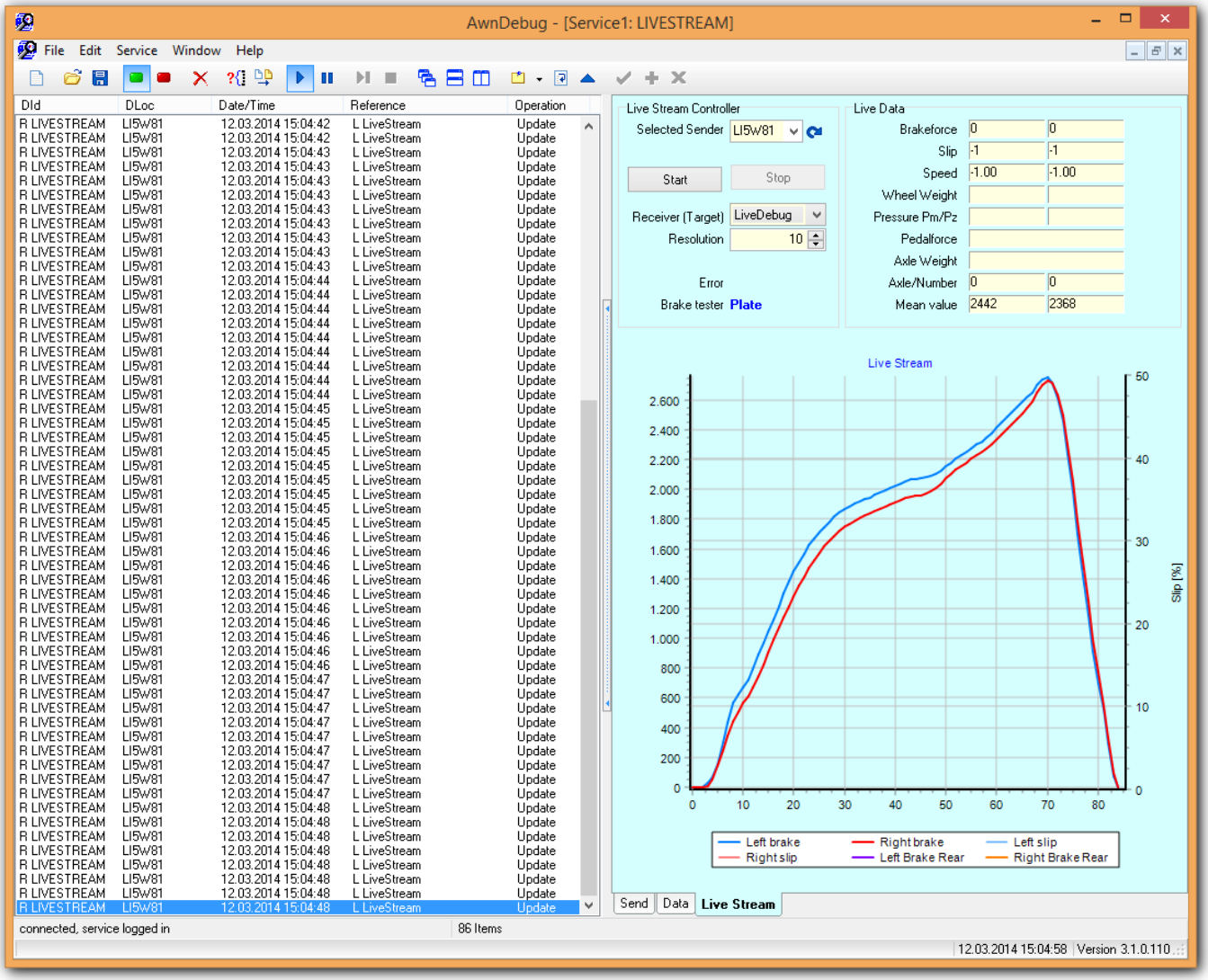
The control requires that you first select the used brake bench or simulator (sender). The combobox shows only the currently active senders, i.e. logged in to the network manager. By clicking on the refresh button on the right, the list can be updated any time.

After choosing a sender, the transmission can be activated with Start and finished with Stop.
Optionally, the transmission can be limited to LiveDebug as the only receiver and the resolution can be adapted.



Picture 34, Live Stream control in detail

The mean value is only used for plate brake benches:



Picture 35, LiveStream with plate brake tester

8 Appendix

8.1 List of Tables and Pictures

8.2 History

8.2.1 Version 3.1 of 3/10/2014

Updated and enhanced for version 3.1.

8.2.2 Version 3.0 of 2/2/2011

Extended version with AwnRemote, AwnStatus and LiveStreamSimulator.

8.2.3 Version 1.0 of 7/2/2003

First official version for the new SDK Tools 1.0