

Free Linux ACARS Decoder and ACARS Online Server

**It is not allowed to monitor all radio frequencies in every country!
Check with your local authorities prior to use ACARS data.**

acarsd – Free Linux ACARS Decoder and Server

2.1 Possible Command parameters

acarsd is equipped with several command parameters, which can influence the operation.

- ```
-d device This parameter is required when the soundcard is not activated
 by /dev/dsp. Device is the name of the device from which the
 sound information is retrieved.

-a When acarsd is being used as server then it possible to allow
 the clients to enter updates in their databases. This option is
 normally deactivated.
 When Option -a is used, 3 acarsd processes are running!

-e This parameter starts acarsd in the advanced mode: for every
 successful message a short statistic about the actual contact
 is displayed.
 This option should not be activated when a less powerful PC is
 being used!
 acarsd does not run per default in expanded mode!

-l logdir Using this switch allows the user save the log files in a
 separate directory. Without the -l switch, all log files will
 be stored in the same directory in which acarsd is installed.

-s Using this switch will start acarsd in the server mode, i.e. 2
 processes are running instead of only one. (with the -a switch
 even 3 processes are running). The server allows the users to
 connect to acarsd and follow the decoded signals live.
 Clients for Windows & Linux are available!
 Connects with a standard WebBrowser are not supported.

-c The output of acarsd is not sent to the screen when using this
 option. Only functions together with option -s. Using this
 option it is possible to run acarsd as a daemon.

-p port The acarsd server uses port 40000. With option -p <port> it is
 possible to assign a different port.
 Only a root user can assign ports less then 1024!

-i ip Default binding of the server is on all available IP addresses.
 By using -i <ip> it is possible to bind the server only on IP
 <ip>.

-r When this option is being used at the start of acarsd the
 database will be synchronized with the existing one and
 possible new entries will be loaded.

-R dir If it is necessary to create reports for exporting (in .pdf or
 .txt format) then option -R needs to be used. They will then be
 stored as .pdf.zip and/or .txt.zip
 The reports need to be put at the proper location using either
 a script or a program.
 A tool to create the pdf files is under development
```

- ```

-M sysname      This command allows automatic mailings. Users can register via
                 the acarsdclient and receive at midnight an email with the
                 ACARS report of the previous day. The report is sent as a
                 zipped file.
                 See par. 2.1.1

-0 days         All entries in the acarsd database older than days will be
                 deleted.
                 When this parameter is not being used, acarsd might occupy the
                 available memory with its database. Depending upon the
                 available memory and the air traffic in the local area this
                 value should be selected.

-H             When the PC running acars does not support SHARED MEMORY
                 (generally speaking all newer Linux versions support this) it
                 is possible to deactivate the shared memory using this
                 parameter. This will reduce the performance of the decoder,
                 because no permanent dataflow from the soundcard is guaranteed.

-A             This parameter forces the creation of two daily logfiles. One
                 for the acarsd application and one for the AIRNAV application.
                 The filename of the AIRNAV files starts with ANAD- and can be
                 used for flight tracking using the AIRNAV SUITE. See Par. 3

-F freq        This parameter informs the possible client(s) which frequency
                 is presently being used in the scanner.

-q            When living close to an uplink site it is possible to receive
                 the uplink and squitter messages. Using the -q parameter it is
                 possible to block them from filling up the screen.

-k            A parity check is contained in every transmission allowing the
                 verification of the message.
                 Only valid messages are displayed when acarsd is started with
                 option -k. Messages with an invalid checksum are displayed in
                 red and are not logged and are not forwarded to any connected
                 client.

-v            Disable the airport resolution for the contacted flights.
                 Use only if your system is very slow or low on memory

```


Enter the following command to start acarsd as private Decoder (**in some countries it is illegal to listen to certain frequencies!**):

The following screen opens:

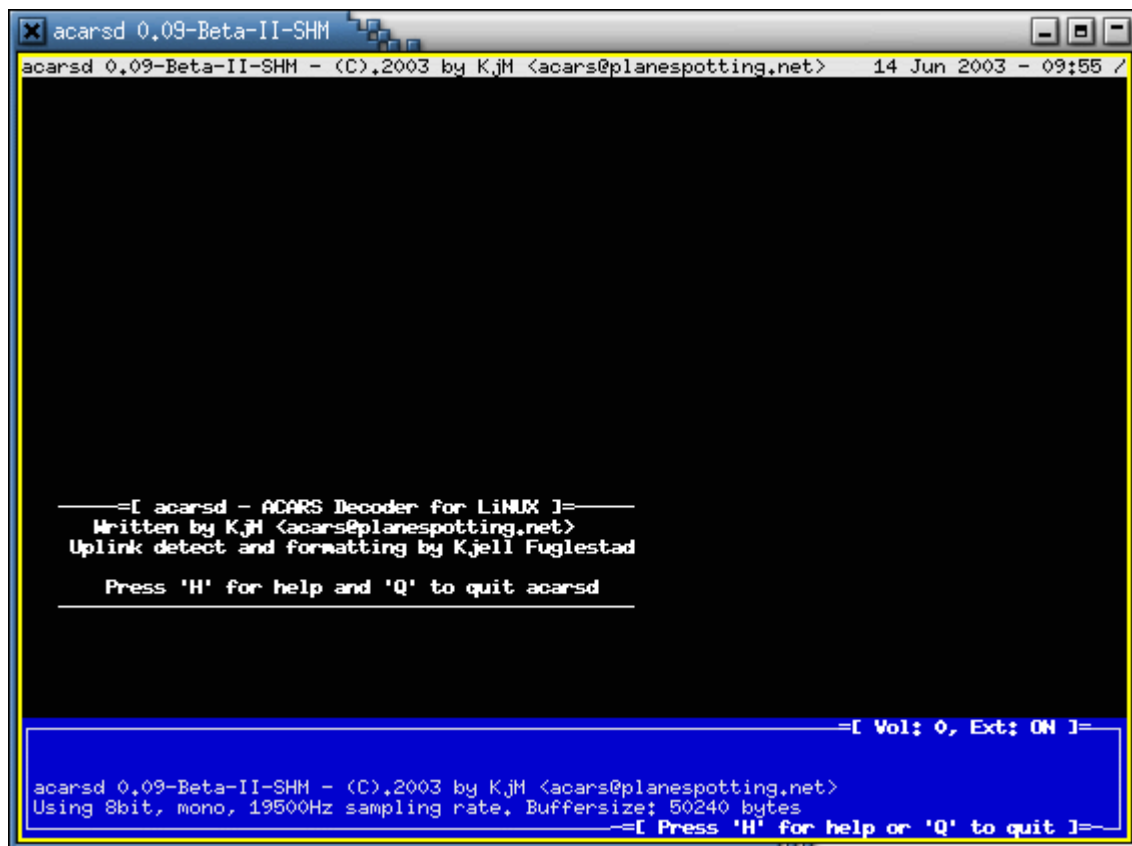


Figure 1

The version used will probably be different then the one shown in Figure 1

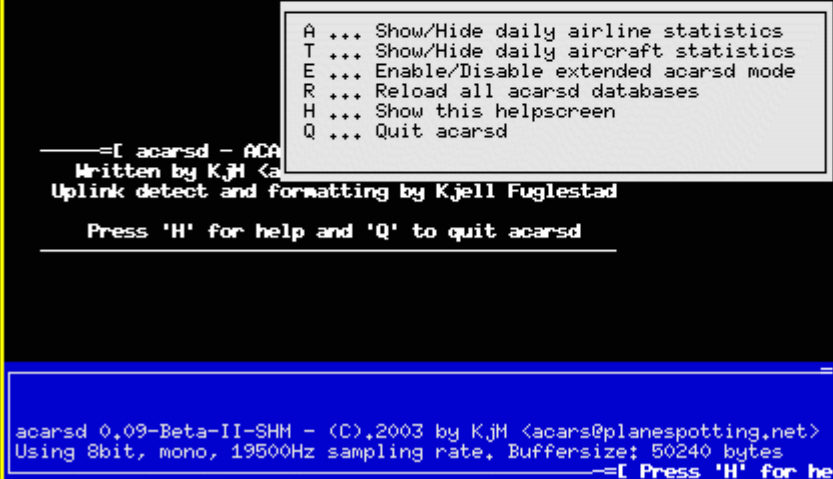
Verify that LINE-IN is set up for recording!

Adjust the volume control of the air band scanner.

When the Volume indicator is still set for **0**, therefore it appears that LINE-IN is not setup as record, or the interconnect cable is not connected properly. Start the sound mixer application (kmix, gmix, tkmix, cmix etc.) and test the recording source. Adjust the settings accordingly.

**The ideal Volume-setting is between 5 and 11!
A volume setting too high is preferred above a low one!**

When `acarsd` is not started as daemon it is always possible to exit the application by pressing **Q**.
All possible commands are shown pressing **H**.
Pressing **H** again returns the normal `acarsd` display.



2.3 Starting acarsd as Server

To start acarsd as Server use the following commands:

```
./acarsd -e -r -s
```

as Daemon in the background

```
./acarsd -e -r -s -c &
```

```
as Daemon on Port 5000
```

```
./acarsd -e -r -s -c -p 5000 &
```

If acarsd is not started as Daemon, then the same screen opens as when starting a simple Client (see Par. 2.2)

It is also possible to run `acarsd` as a server, which uses the local interface of the network card (and can be reached from the internet). The switch would be **-i 127.0.0.1**

```
./acarsd -e -r -s -c -i 127.0.0.1 &
```

The sequence of the parameters has not been determined yet!

2.3.1 Is the Decoder operational?

During normal operation it is possible to determine whether the decoder is operational. In the top right hand corner is a symbol which changes after the receipt of every sound package. When acarsd is running on a very fast Pentium, the symbol appears to be turning continuously.

On a Pentium 133 every sound package requires about 2 seconds, so a visual change of the symbol is possible.

acarsd provides all information on-screen. A correction of the data is not done. acarsd is a ***Real Time Decoder!***

Messages decoded correctly without parity errors are displayed in grey. Messages containing text errors are shown in white while the actual text will not be displayed. Messages containing other errors will be displayed in binary (red) while the error message will be shown in cyan.

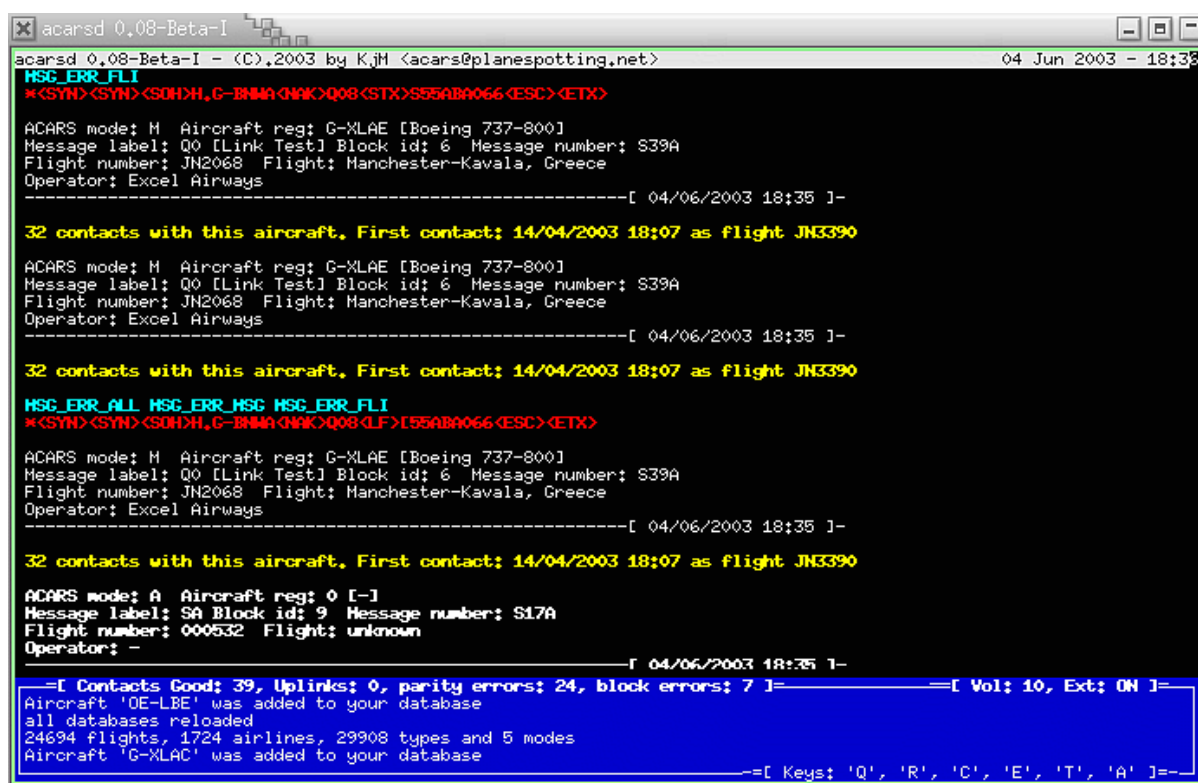


Figure 2

Description of the error flags:

MSG_ERR_ALL	Parity error within the message
MSG_ERR_REG	Error in the type of aircraft
MSG_ERR_LAB	Error in the message label
MSG_ERR_BLK	Error in the block ID of the message
MSG_ERR_MSG	Error in the message number
MSG_ERR_FLI	Error in the flight number
MSG_ERR_EXT	Parity error in the text block of the message

2.4.1 On-Screen Statistics

Statistics can be provided when acarsd is running in client mode (or not as daemon). Pressing key **A** opens the statistic window and shows the daily contacts of the various airlines (see Figure 3). Using key **T** provides statistics about the types of aircraft.

Both statistics refer to the same day and are cleared automatically at 0.00 am. The window will be closed automatically when a message is decoded. Pressing either the A or T keys closes the window.

The screenshot shows the acarsd 0.04-Beta-XIV terminal window. The main window displays ACARS data for a message received on 23/04/2003 at 15:38. The data includes the aircraft registration D-AILT, type Airbus A319-100, and flight number LH3419. A statistics window is open, showing a list of airlines and their contact counts. The statistics window is titled "Daily Airlines Statistics" and lists the following airlines and counts:

Airline	Contact Count
Lufthansa	0052
KLH Royal Dutch Airlines	0017
Emirates	0015
Swiss International Air Lines/Crossair	0015
Austrian Airlines	0014
Air France	0012
Hapag-Lloyd	0011
Croatia Airlines	0010
Gulf Air	0009
Lauda Air	0007
Adria Airways	0006
Scandinavian Airlines	0006
Singapore Airlines	0005
Airfoyle	0004
Cyprus Airways	0004
EVA Airways Corporation	0004
FedEx Express/Federal Express	0004
Aero Lloyd	0003
Aeroflot	0003
Braathens	0003

Below the statistics window, the terminal shows the following text:

```

4 contacts with this aircraft. First contact: 23/04/2003 11:54 as flight BH595
ACARS Mode: H Aircraft Registration: D-AILT
Aircraft Type: Airbus A319-100
Message Label: H2 Block ID: 8 Mess
Flight Number: LH3419 Flight: Istanbul
Operator: Lufthansa
Message Content: -
Q2+FMH<R>J:(B:U89S EEAA:
5 contacts with this aircraft. First contact: 23/04/2003 11:54 as flight BH595
ACARS Mode: K Aircraft Registration: D-AILT
Aircraft Type: Boeing 737-700
Message Label: 00 Block ID: 8 Mess
Flight Number: BU1596
Operator: Braathens
2 contacts with this aircraft. First contact: 23/04/2003 11:54 as flight BH595
Contacts Good: 270, with parity errors: 143
Aircraft 'D-AILT' was added to your database
Contacts Good: 271, with parity errors: 143
Contacts Good: 272, with parity errors: 143

```

Figure 3

2.4.2 Usage of other keys

Pressing **Q** closes acarsd.

When changes have been made to the databases it is necessary to press **R**. This informs acarsd about changes in the databases.

When acarsd is running as daemon it is necessary to send the main process (the one with the lowest PID) a HUP signal: kill -HUP <processid>.

The database server (when acarsd is running as a server and has been started with **-a**) informs the decoder using SIGHUP updates which are being sent from the clients.

Updates from the database server are processed every 5 minutes to prevent blocking the decoder!

Pressing **E** allows the user to activate or de-activate the extended mode (this is only functional when acarsd is operated as client or not as a daemon).

Next to the volume indication is the information whether or not acarsd is running in the extended mode.

Extended Mode: OFF = no extended mode

Extended Mode: ON = extended mode active

Extended Mode means:

Acarsd keeps statistics about the contacts of the various aircraft. Within the extended mode acarsd will also try to translate the message label into clear text. When acarsd is running in the daemon mode (no screen output), then it is possible to switch the mode as follows:

kill -USR1 <pid of the ACARS decoder>

The PID of the decoder can be determined using the command:

```
ps auxw |grep ACARS| grep -v grep
```

The second value in the response is the PID

2.4.3 Changing the size of the window

It is always possible to change the size of the window when acarsd is running in normal mode. Be aware that when changing the window size, the actual contents of the window are cleared.

3 Log files

acarsd writes log files, which automatically rotate at midnight. The log files are stored in the same directory as acarsd unless the option `-l` has been used during startup.

The log files always use the following scheme:

acarsd-YYYYMMDD.log

YYYY	Year
MM	Month
DD	Day

Example for the contents of a log file (running in acarsd mode):

```
----- [ 01/04/2003 10:08 ]-
ACARS Mode: 0 Aircraft Registration: VP-BAH
Aircraft type: Boeing 737-400
Message Label: 5U Block ID: 3 Message Number: M65A
Flight Number: SU0281
Operator: Aeroflot
Message Content: -
    01 WXRQ 0281/01 UUEE/LIRF .VP-BAH
/TYP 1/STA LIRF/STA /STA

----- [ 01/04/2003 10:27 ]-
ACARS Mode: A Aircraft Registration: D-AIAR
Aircraft type: Airbus A300
Message Label: _ Block ID: 9 Message Number: S49A
Flight Number: LH3422
Operator: Lufthansa
Message Content: -
```

When acarsd is started with option `-A`, in addition to the acarsd log file a 100% compatible AIRNAV file will be generated.

The filename is:

ANAD-YYYYMMDD.log

YYYY	Year
MM	Month
DD	Date

These files can be used with AirNav Suite, and it should be possible to use them with other commercial available products.

9.3 Possible Operations

The following Operations are presently used

```
#define AS_WELCOME      1    only Server -> Client  
                           Sends the servername and port (separated by tab) to the client.  
#define AS_BYE         2    only Client -> Server  
                           The client signalsizes that he wants to disconnect from the  
                           server.  
#define AS_FINFO       3    not used  
#define AS_CONTACT     4    Server -> Client  
                           The server sends the actual statistics as pure text to the  
                           client.  
#define AS_VOLUME      5    Server -> Client  
                           The server sends the volume string to the client.  
#define AS_HIT         6    not used anymore since Beta VII  
#define AS_PARITY      7    Server -> Client  
                           The server indicates to the client that the following  
                           transmission contains parity errors and should not be indicated  
                           as "positive" anymore.  
#define AS_EXT         8    Server -> Client  
                           The characteristic of the actual aircraft is sent to the client.  
                           (Only when the server is operating in the expanded mode.)  
#define AS_FETCH       9    Server -> Client  
                           The server signals the client that the following aircraft  
                           information is to be displayed. String: TYPE\tREG\tFLUGNUMMER  
#define AS_TYPE        10   Server -> Client  
                           After a successful decode the TYPE information is sent to the  
                           client so an update of the statistics can take place.  
#define AS_DBINFO     11   Server -> Client  
                           Is only sent once after a connect. The server informs the client  
                           about the size of the database. String: FLÜGE\tTYPEN\tAIRLINES  
#define AS_INFO       12   Server -> Client  
                           Will only be sent once after login. Quantity of records in the  
                           local ACARS database.  
#define AS_AIRLINE    13   Server -> Client  
                           This is sent to the client after a successful decode, so the  
                           client can update his statistics.  
#define AS_REG        14   Client -> Server  
                           The client looks for a registration.  
                           The answer is either AS_RECHCNT or AS_RECHERR  
#define AS_RECH       15   Server -> Client  
                           The result of a search is sent to the client.  
                           String: UNIX Zeitstempel\tFlugnummer\tFlug  
                           Or  
                           UNIX Zeitstempel\tReg\tType  
#define AS_RECHCNT    16   Server -> Client  
                           Will be sent as a positive answer to a search.  
                           String: Quantity of hits\tSuchstring  
#define AS_FLIGHT     17   Client -> Server  
                           The client looks for a flight number.  
                           The answer is either AS_RECHCNT or AS_RECHERR  
#define AS_HIT2       18   Server -> Client  
                           A decoded message is transmitted.  
                           The strings are separated by TAB.
```

```
#define AS_TXTREP      19      Client -> Server
                                The client would like to have a daily report as a zipped text
                                file.
                                The filename is provided as a string
#define AS_PDFREP      20      Client -> Server
                                The client would like to have a daily report as a zipped pdf
                                file.
                                The filename is provided as a string
#define AS_REPDIS      21      Server -> Client
                                The server sends this to the client to indicate that no reports
                                are available.
                                This code will be sent even when a client requests a report.
#define AS_REPSIZE      22      Server -> Client
                                The server sends the client the size of the following report.
#define AS_REPNAME      23      Server -> Client
                                The server sends the client the filename of the following
                                report.
#define AS_AFLIGHT      30      Client -> Server
                                The client wants to add the flight to the database of the server
                                SYNTAX: Flightnumber\tVON-NACH
#define AS_AAIRLINE      31      Client -> Server
                                The client wants to add the airline to the database of the
                                server
                                SYNTAX: IATA or ICAO Code\tAIRLINE
#define AS_ATYPE      32      Client -> Server
                                The client wants to add the type to the database of the server
                                SYNTAX: Reg\tTypenbezeichnung
#define AS_ADDED      35      Server -> Client
                                The Server informs the Client that the update has been done
#define AS_NOTADDED      36      Server -> Client
                                The requested update was NOT added to the database
#define AS_CCLIENT      50      Client -> Server
                                The string is a chatmessage which will be sent to other clients
                                in chat mode.
#define AS_CMESS      51      Server -> Client
                                Chatmessage which needs to be displayed in the client.
#define AS_CON      52      Client -> Server
                                Client activates the chat
#define AS_COFF      53      Client -> Server
                                Client deactivates the chat
#define AS_CINFO      54      Server -> Client
                                The Server informs the Client about the logins and logouts. The
                                texts also include the nicknames of the users.
#define AS_SUBSCR      60      The user subscribes to the mailings.
                                The text is the mailing address
#define AS_UNSUBSCR      61      The user unsubscribes from the mailings.
                                The authentication code must be sent as the keycode, otherwise
                                no unsubscribe will take place
#define AS_ACTIVATE      62      The authentication code of the mailing is activated.
                                The authentication code and the mailing address are forwarded
#define AS_PAUSED      63      The active mailing is being halted.
#define AS_REPINFO      64      A report is being transmitted from the server to the client.
#define AS_NOSUBSC      65      A subscription is canceled
#define AS_ADMINMSG      90      Server -> Client
                                Admin messages are shown on the Client.
#define AS_FREQ      91      Server -> Client
                                The Server informs the Client which frequency the scanner is
                                tuned to.
#define AS_BUSY      100      Server -> Client
                                The server reached the limit of x clients.
                                The socket will be closed after this message.
#define AS_PERROR      101      not used anymore since Beta II
#define AS_CERROR      102      not used anymore since Beta II
#define AS_RERROR      103      Server -> Client
                                The following message has an error in the registration or flight
                                number and should therefore not be identified as positive.
#define AS_RECHERR      104      Server -> Client
                                Possible answer on a search. No hit!
#define AS_NOREP      105      Server -> Client
                                Requested report is not available or cannot be read.
#define AS_NOUPD      106      Server -> Client
                                The server informs the client directly after a connect that no
                                updates are allowed.
#define AS_ALIVE      150      Server -> Client
```


9.4 Communication with Clients

The server always responds to one request per client. Therefore all clients have the same priority and possible DOS attacks should not stop the server.

*Requests and reports are never transmitted simultaneously!
Reports have priority over requests!*

9.5 Available Clients

Graphical Clients for LINUX and Windows are available at www.acarsd.org for downloading.

9.6 Live ACARS on the web.

It is possible to view live ACARS directly on the web. Browse to <http://www.planespotting.net/acarsd/LIVE/>. Required for this is a modern browser, and Javascript and cookies need to be activated. This has been tested using Mozilla, Netscape (version 6) and MS Internet Explorer, and this did not cause any problems.

10 Licenses and Tools

acarsd may only be distributed in an unmodified form. Translation of the ASM code is absolutely prohibited.

The following tools/libraries are internally used within acarsd:

Collections by Stefan Briesenick sbriesen@gmx.de

These tools allow the creation of very fast databases. Collections can be used universally, so that all internal statistics and all required data in a sorted form can be linked within these Collections.

11 Thanks

First of all I'd like to thank my family. They only saw my backside during the development phase of acarsd.

Thanks also go to

MM	Our specialist for sound analysis and good graphics ☺
Peter Brucker	The roof man and the one with the longest antenna. I was also allowed to test acarsd on several of his PCs.
Ingo Richardt	He never gave up until acarsd had more contacts than ANAD. Also thanks for puma. ☺
Siegfried Huss	He's continuously testing and without him we probably wouldn't have a database with more than 25,000 entries
Armin Zundel	My supervisor has good ideas, and these have been implemented in acarsd
Kay Berkling	Thanks for the tips about FFT and other mechanism.
KD Baer	For the various pictures and the very good advises
Marinus Jacobs	For the translation into English. If you had to read my English, you would be able to get acarsd up and running. ☺
Gabor Kerekes	For the information about the modification of the ANAD log files and the life testing of these.
Kjell Fuglestad	Thanks for the information about SQ messages, Uplinks and the use of them, and not to forget the amount of info provided and the support with the beta versions.
Francois Guillet	Provided details about testing messages using the parity check.