COMPROTware:Testtool

Introduction and Training

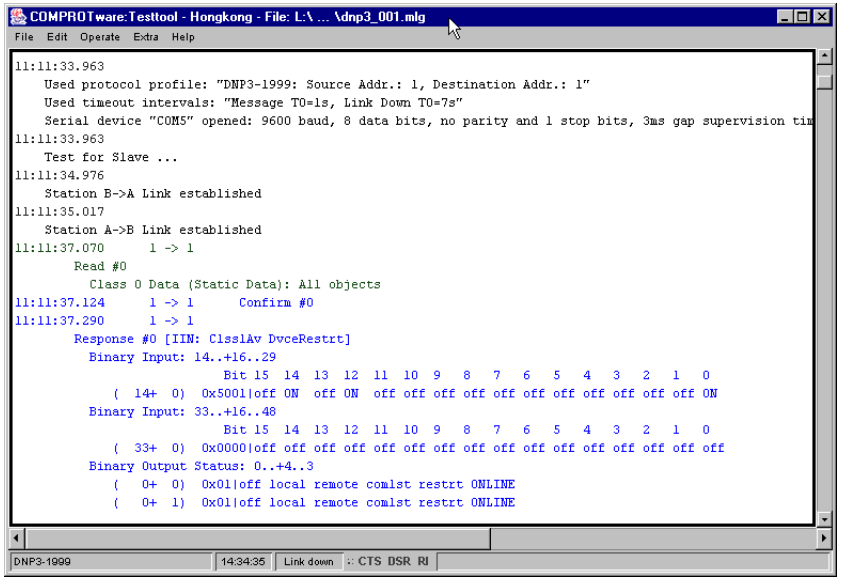
Real Thoughts GmbH
Haid-und-Neu-Straße 7
76131 Karlsruhe
Germany

Fon +49-721-6276730, Fax +49-721-6276731

Website www.realthoughts.de
E-mail info@realthoughts.de

General

- *CPTT* is an integrated testtool for transmission protocols of telecontrol systems
- ... combines different protocols in one user interface
- ... simulates master or slave station of a telecontrol system



(For details enlarge picture!)

Delivery

- The delivery includes
  - printed installation manual
  - CD-ROM with installation, documentation and examples
  - USB-, parallel port- or PCMCIA-dongle to license the protocols

CD-ROM

- CD-ROM contains the installation, documentation and examples.

Documentation

- The directory `\doc` contains the description of the software its self: Installation Manual, User's Manual, and Programer's Guide; all material comes in German and English, and the User's Manual additionally in Italian
- `\doc\Support` contains descriptions of ***CPTT***-related problems and advanced topics, like remote license updating
- In `\doc\Marketing` the data sheet, the end user price list and the presentation can be found.

Installation of CPTT

- The ***CPTT*** installation is based on Microsoft Windows Installer
- It is described in detail in the Installation Manual
- New releases are installed parallel to old ones; in case of problems with the new release you still have access to the old one. Old releases can be very quickly uninstalled using Microsoft Windows Installer
- The installation occurs in five steps (sequence at will, dongle **not** necessary):
  - Installation Java Runtime Environment for Windows x86 (32 Bit):
    http://java.com/download
  - Installation WibuKey Runtime for Windows (Windows 32/64 Bit, multilanguage):
    http://www.wibu.com/us/downloads-user-software.html
  - Installation of ***COMPROTware:Testtool***
    Start with `\iX86_WIN32\CPTT\setup.exe`
  - Installation WinPcap
    http://www.winpcap.org/
  - Installation of ***RIO Server***
    The installation of this option is described later on
- Normaly, the standard values can always be used

Licensing

- For simulation and monitoring a license is required; Offline Analysis can be done without a dongle any time
- *CPTT* may be installed on any computer; a license is only required for simulation and monitoring
- The protocol specific licenses are stored in the dongle; only if a license for a protocol is stored in the dongle simulation/monitoring is possible
- Dongles are programmable:
  - A bit pattern in the dongle is used to control which protocols are licensed
  - Only one license per protocol can be stored on a dongle, but for all protocols a license can be stored on a dongle.
  - With remote programming, by exchanging a context file and an update file, the content of a dongle can be altered over electronic media
- The dongles are deliverable as
  - Dongle for USB
  - Dongle for parallel port
  - Dongle for PCMCIA slot
  - For more information about dongles, see http://www.wibu.com/

Start

- As usual under MS Windows, ***CPTT*** is started from the Start menu or by double-clicking on the icon on the desktop
- On program startup the configuration of the last program execution is restored.
- ***CPTT*** can be started several times, even several simulations may run at the same time
- ***User Engine Classes*** (see further below) can be loaded and started during program startup

General Handling

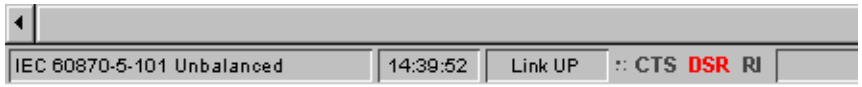
- The handling of ***CPTT*** meets the typical rules under MS Windows; with shortcut keys frequently used functions can be accessed very quickly
- The following file extensions are associated to ***CPTT***:
  - `.mlg` for Message Log files (protocol traffic logs)
  - `.mls` for Message List files (message lists)
  - `.cptt` for configuration files (parameter sets)
- ***CPTT*** destinguishes between four operating modes:
  - Pure display of protocol traffic (*Operate -> Stop*)
  - Simulation of a Master (*Operate -> as Master*)
  - Simulation of a Slave (*Operate -> as Slave*)
  - Monitoring (*Operate -> Monitor*)

Examples

- The delivered Messages List examples under `c:\Program files\realthoughts\comprotware\testtool\doc\<RELEASE>\MLS_Examples` with name suffix `*_Std_Example.mls` always carry out a complete demo of both Master and Slave; the Message Lists must be imported throught *Edit -> Message List Storage ...* and then the menu item *File->Open from ...*
- Depending on which protocol you have licensed:
  - Is it a serial line-based protocol? Then connect two serial ports and run one tool on one port as Master and another tool on the other port as Slave.
  - It's even easier with network-based protocols: one instance of the testtool runs as Slave, the other runs as Master Station (hence the IP address has to be set to 127.0.0.1 (Localhost))
- The delivered examples should help you understand ***CPTT*** and give you some insight on the protocol

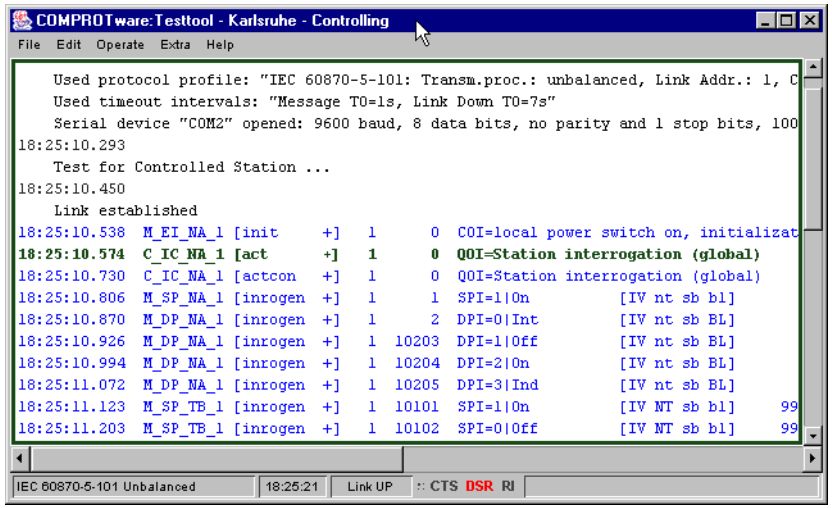
Simulation

- ***CPTT*** can simulate Master as well as Slave (Controlling and Controlled Station or Master and Outstation respectively) for all implemented transmission protocols
- First through *Edit -> Protocol Profile ...* choose the desired protocol from a protocol family; the station name makes the grouping of the different windows easier
- In the next window, the protocol specific parameter must be specified; for each protocol, there's a separate set of parameters and a separate input mask; the default values of ***CPTT*** meet common usage conditions
- Now with *Operate -> as Controlling*/*Operate -> as Master* or with *Operate -> as Controlled*/*Operate -> as Slave* you can simulate a control system or a sub device; *Operate -> Stop* stops the simulation
- In the window's title, the operation mode will be displayed: `Controlling`, `Master`, `Controlled`, `Slave`, `Monitor`, ...
- In the footer, the choosen protocol, the current time, state of the link (UP, down), the receiving of characters through animation and the modem signal CTS, DSR and RI are displayed:



Simulation - Continuation

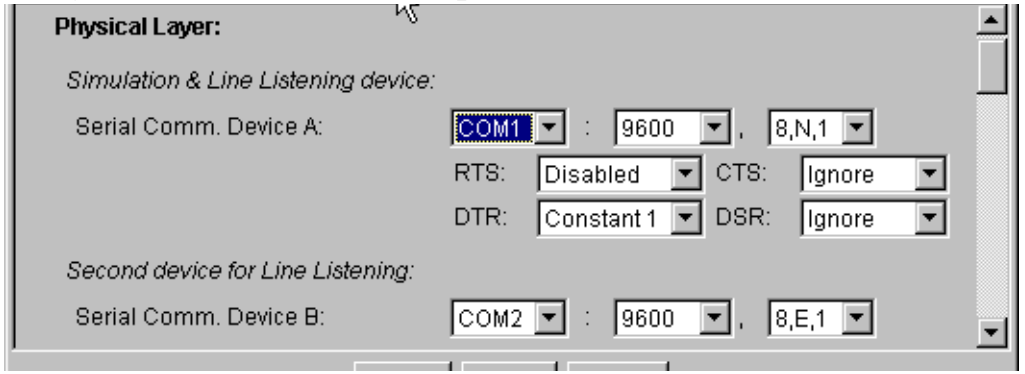
- The protocol traffic is classified by colors:
  - The border color of the display window indicates the operation mode
  - Messages sent by ***CPTT*** are in bold
  - In IEC 60870-5-104: ***green*** is always Controlling, ***blue*** is always Controlled
  - In serial protocols: ***green*** is always Master, ***blue*** is always Slave;
    except with IEC 60870-5-101 balanced: ***green*** is always dir, ***blue*** is always DIR



- About the state of the connection (displayed in the footer):
  - Link down: no connection established; with Master simulation ***CPTT*** tries to initiate a connection by sending restart frames
  - Link UP: connection established; messages on Application Layer are exchanged

Protocol Profiles

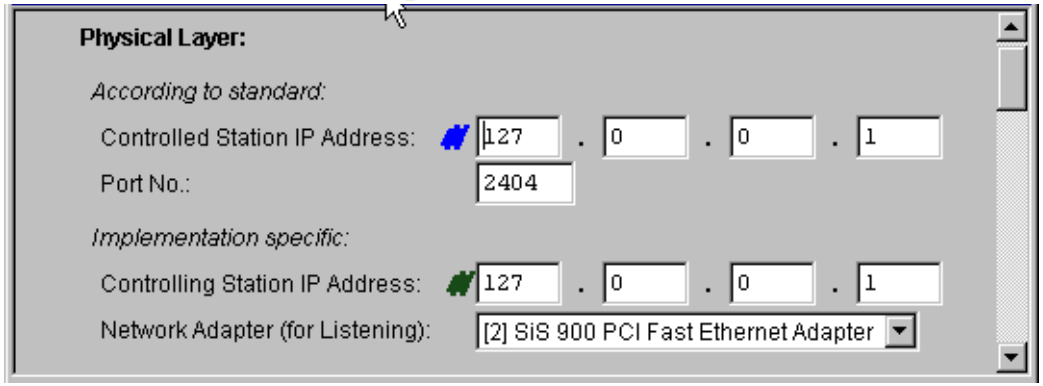
- Physical Layer with serial line-based protocols:



  - The simulation runs with Serial Comm. Device A;
    the setting for the modem signal displayed here represents a direct connection
  - Both Serial Comm. Device B and Serial Comm. Device A are used for the monitoring of serial line-based protocols; the modem settings will be ignored

Protocol Profiles - Continuation

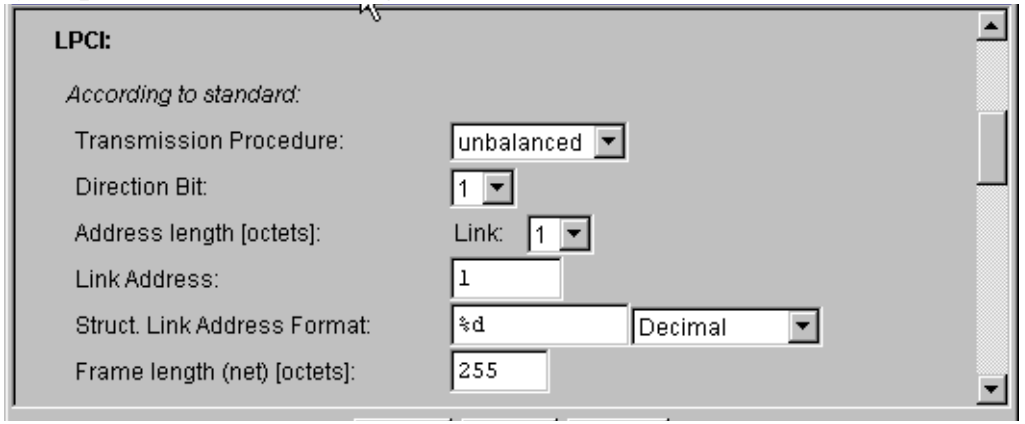
- Physical Layer in network-based protocols:



  - During simulation of a Controlling Station, the Controlled Station IP Address indicates the IP Address of the Controlled Station; 2404 is the port no. for the connection establishment according to the IEC 60870-5-104-standard
  - During monitoring, the Controlled Station IP Address and the Controlling Station IP Address are filter for the display of network packets; the value 255 is the wildcard value

Protocol Profiles - Continuation

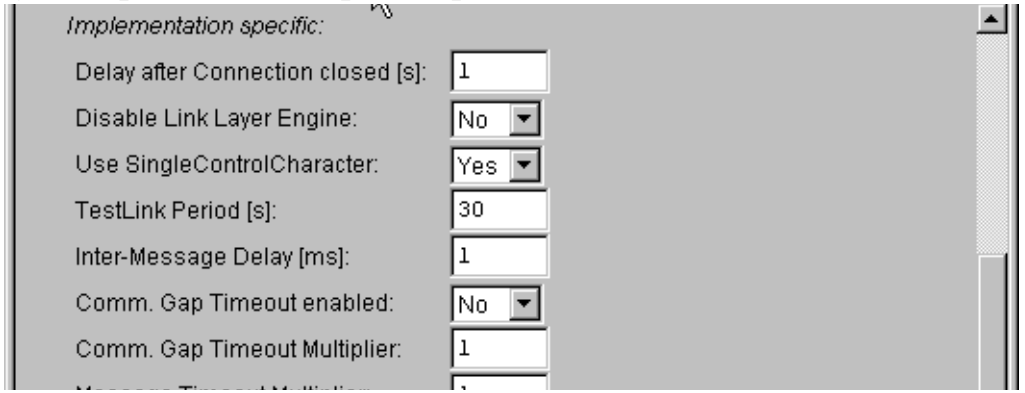
- Link Layer, parameters according to the standard:



  - Transmission procedure: unbalanced (polling) or balanced (spontaneous)

Protocol Profiles - Continuation

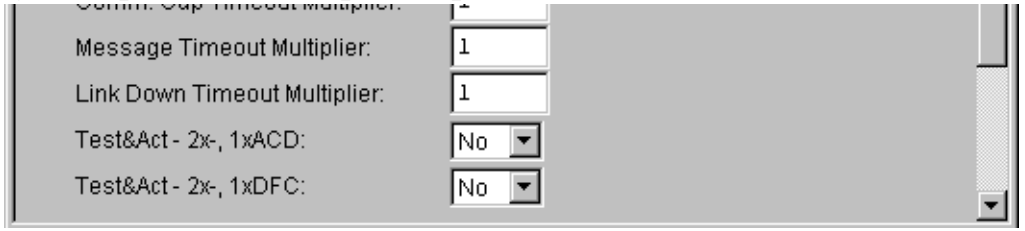
- Link Layer, implementation-specific parameters:



  - Delay after connection closed: delay after a connection loss, before a new connection is tried to be established
  - Disable Link Layer Engine: Link Layer Engine is deactivated (e.g. for connection reestablishment)
  - Use SingleControlCharacter: should the Single Control Character 0xe5 be used?
  - TestLink Period: duration period for TestLink-cycles
  - Inter-message Delay: min. delay between the last received and next sent message
  - Communication gaps are delays between two characters on a serial line; within a frame, gaps are not allowed; here the supervision of comm. gaps can be enabled and a timeout multiplier for tolerated gaps can be specified

Protocol Profiles - Continuation

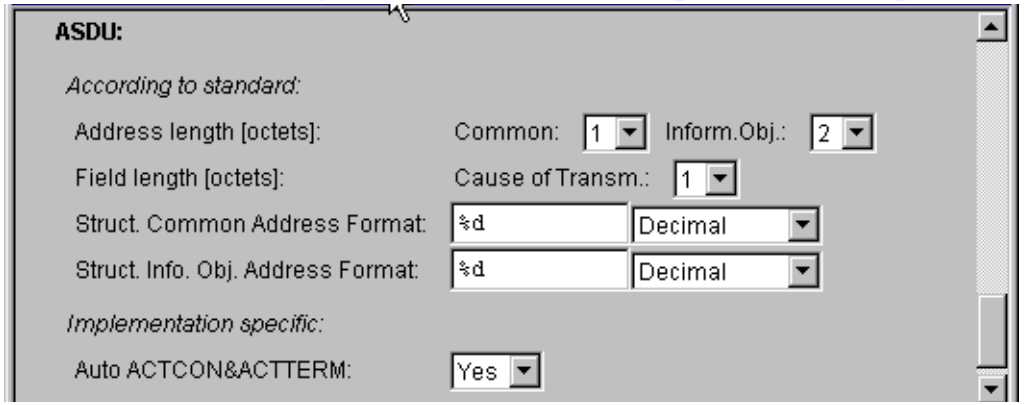
- Link Layer, implementation-specific parameters:



  - Message Timeout Multiplier: message reception timeout multiplier; reference time is calculated based on frame length and baudrate
  - Link Down Timeout Multiplier: multiplier for link down timeout; reference time is calculated based on frame length and baudrate
  - Test&Act - 2x-, 1xACD: if set, during Controlled Station simulation the ACD bit is set in every third response frame
  - Test&Act - 2x-, 1xDFC: if set, during Controlled Station simulation the DFC bit is set in every third response frame

Protocol Profiles - Continuation

- Application Layer, according to the standard and implementation-specific:



  - Auto ACTCON&ACTTERM: if set, then on each command an ACTCON and if necessary an ACTTERM will be sent

Simulation Examples

- Now, the examples for IEC 60870-5-101 or -104 resp. should be read: open the files `IEC60870_5_101n104_Std_Example.mls` and `IEC60870_5_101n104_All_TypeIdents.mls` (when reading the last file, don't delete the Message List Storage)
- ***CPTT*** can be started several times on a computer: use your computer to simulate both Master and Slave; with a network-based protocol use the IP address 127.0.0.1 (local-host) for simulation, with a serial line-based protocol connect two serial ports with a cable

Navigation

- Using the keys *cursor up* und *down*, *Page up* and *Page down* and *Pos1*, *STRG+Pos1* and *End*, you can navigate through the window.
- Normally the view is updated „in place“, so that the newest protocol traffic is shown continously.
- By scrolling up the snap-on is freed. The scroll back buffer allows the examination of previously exchanged messages.
- You return to the updated display by pressing the *End* key
- Navigation within the storage can be done anytime, even during monitoring and simulation

Formatting Options

- The window with the formatting options appears over the background menu (right mouse button) and *Formatting Options ...*; alternatively, the formatting options can be changed with the shortkeys <b>, <v>, <h> and <l>
- Formatting options (all combinations allowed):
  - Time only or with date
  - Verbose, multiline

```
09:45:28.465
      9|M_ME_NA_1|measured value, normalized value
        VSQ [SQ, N=7], [COT=20|inrogen, tst pn], Originator=0x00
        CA=4351
          IOA=131124
            NVA=0x110b|0.133148
            QDS [iv nt sb bl ov]
```

  or brief, single-lined

```
09:45:28.465  M_ME_NA_1 [inrogen  +]   4351   131124  NVA=0.133         [iv nt sb bl ov]
                                       (  131124+1 )  NVA=0.298         [iv nt sb bl ov]
                                       (  131124+2 )  NVA=-0.421        [iv nt sb bl ov]
                                       (  131124+3 )  NVA=0.260         [iv nt sb bl ov]
```

  - Hexdump included

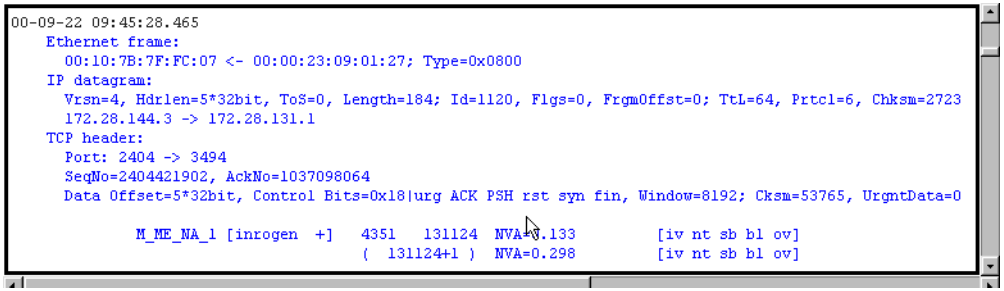
```
09:45:28.465
  0x68 0x22 0x18 0x00 0x02 0x00 0x09 0x87 0x14 0x00
  0xff 0x10 0x34 0x00 0x02 0x0b 0x11 0x00 0x17 0x26
  0x00 0x2b 0xca 0x00 0x57 0x21 0x00 0xc0 0xe4 0x00
  0xeb 0xd5 0x00 0xd7 0x21 0x00
      M_ME_NA_1 [inrogen  +]   4351   131124  NVA=0.133         [iv nt sb bl ov]
                               (  131124+1 )  NVA=0.298         [iv nt sb bl ov]
```

  - With or without Link Layer

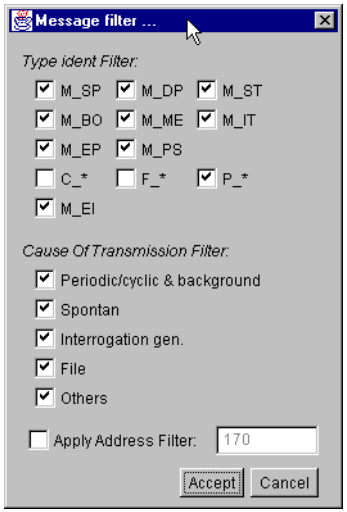
```
09:45:28.465
    I: SSN=12, RSN=1
      M_ME_NA_1 [inrogen  +]   4351   131124  NVA=0.133         [iv nt sb bl ov]
                               (  131124+1 )  NVA=0.298         [iv nt sb bl ov]
                               (  131124+2 )  NVA=-0.421        [iv nt sb bl ov]
                               (  131124+3 )  NVA=0.260         [iv nt sb bl ov]
                               (  131124+4 )  NVA=-0.213        [iv nt sb bl ov]
```

Formatting Options

- Specially for information captured on the network (not for simulation): display of Ethernet Frames and/or TCP Headers is possible

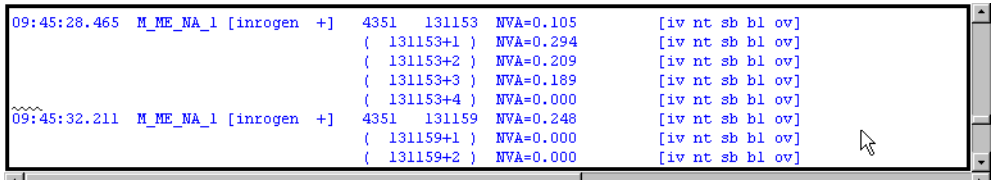


- Filter for type indication, cause of transmission und station address through *Formatting Options ... -> Message Filter ...*



Formatting Options

- The formatting options can be modified any time, even during simulation or monitoring
- A grey wavy line indicates that on that spot a message is being filtered (not displayed)



Display Information Elementes

- General rules for presentation:

```
09:45:28.465
      9|M_ME_NA_1|measured value, normalized value
        VSQ [SQ, N=7], [COT=20|inrogen, tst pn], Originator=0x00
        CA=4351
          IOA=131124
            NVA=0x110b|0.133148
            QDS [iv nt sb bl ov]
```

  - Alternative representations (e.g. decimal and clear text) are separated by `|`; helpful, because very often both coding and clear text or decimal and hexadecimal value are of interest
  - Groups are embraced in `[` and `]`, and are mostly named; this allows to easily link to hex representation
  - All bits are displayed, set bits in capital, clear bits in small letters; you always know which qualifiers an element has

- In the display mode `BRIEF` only a few representations are shown, in `VERBOSE` all necessary ones are there

IEC 60870-5-101/-104 - Structured Addresses

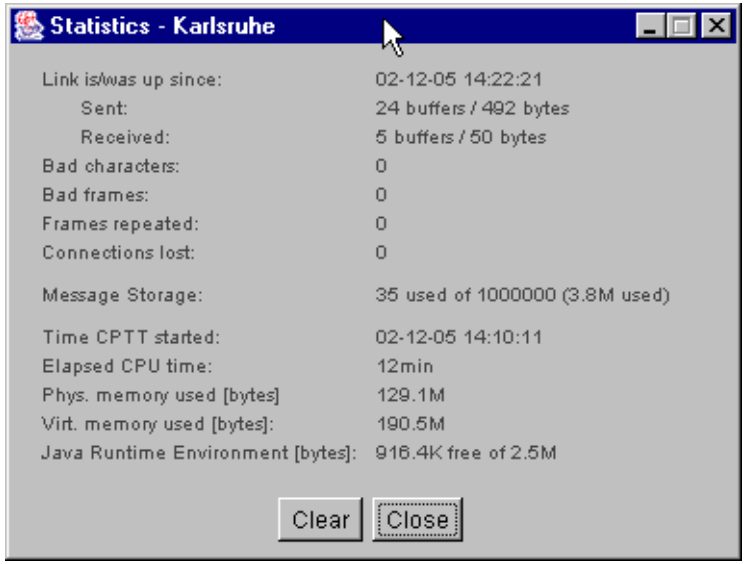
- Structured addresses devide the address octets into smaller units
- *CPTT* offers a very flexible display:
  - Bit groups can be named
  - Display in decimal and hexadecimal
  - Predefined templates make the selection easier
- Format: `V=%23_16d Fld=%15_8d Dev=%7_0d` is 8-8-8
  - Strings can contain any desired character
  - `%` (escape symbol) initiates a value, `%%` results in `%`
  - Optional: bit positions are specified from higher to lower bit; a higher bit position is separated from a lower one by _
  - For decimal representation use `d`, for hexadecimal `x`
  - If no bit positions are given, then the entire address field is used
  - `%23_16d` means that bit 23 to 16 (starting with bit 0) are shown decimal
  - `%d|%x` means that the address field is first shown in decimal and then in hexadecimal, separated by a `|`, e.g. like `192|0xc0`

Settings

- To control the memory use, the size of the Message Storage (memory for the protocol traffic) can be restricted to a value between 3.000 and 1.000.000 (*Edit -> General Preferences ...*)
- Pure Link Layer information may be dropped directly after reception (*Edit -> General Preferences ...*). Thus, the event polling doesn't waste any entries in Message Storage

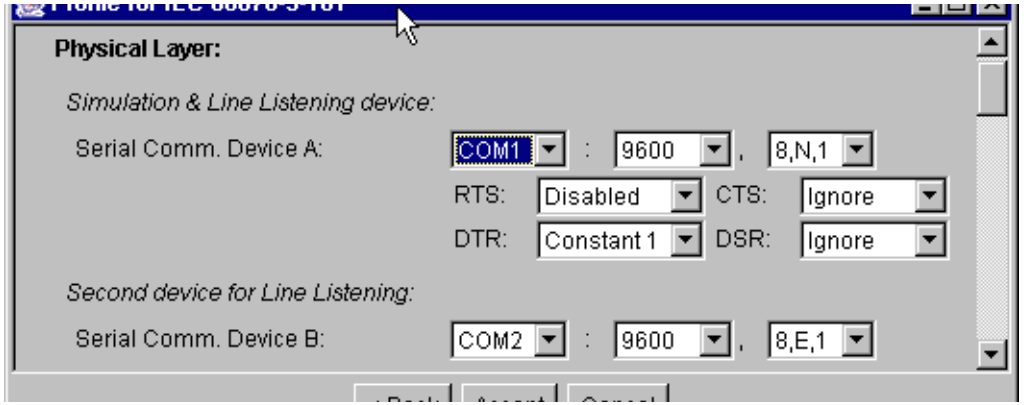
Statistic

- Miscellaneous statistical values about running/last connection and about ***CPTT*** are shown by the window *Extra -> Statistics*; the values are cyclically refreshed



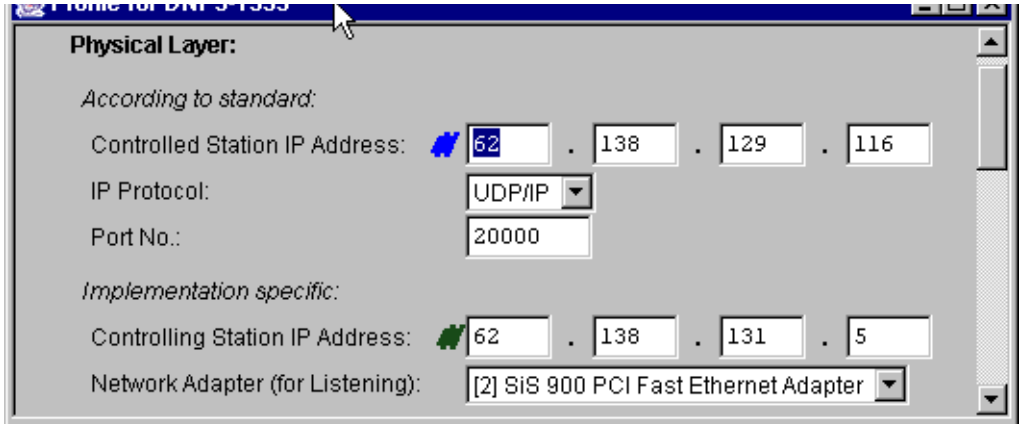
Monitoring

- Monitoring is possible with serial line-based protocols through two serial interfaces, and with network-based protocols through the network adapter
- Concerning monitoring of network-based protocols: with MS Windows no additional components are required but you get better results if WinPcap is installed; monitoring of IEC 61850 GOOSE is only available if WinPcap is installed
- Monitoring is started in *Operate -> Monitor* and through *Operate -> Stop* stopped again
- With serial line-based protocols: both serial devices used for monitoring will be choosen from the list of available devices within the protocol profile; the modem signal settings are ignored:



Monitoring - Continuation

- With network-based protocols: to do Network Monitoring, the IP address of both participants and the network adapter must be set in the protocol profile; if an IP address is unknown, 255 can be used a wildcard value:



  - Only with DNP3 over LAN/WAN: select transmission protocol: UDP/IP or TCP/IP
  - Network Adapter specifies the network adapter which is used to monitor on the network
  - Only the network traffic that passes by the network adapter can be monitored.
  - Switches are used for balancing the network load, hubs are used to distribute network packets; a switch only forwards network packets to its destination, preventing network monitoring on arbitrary packets

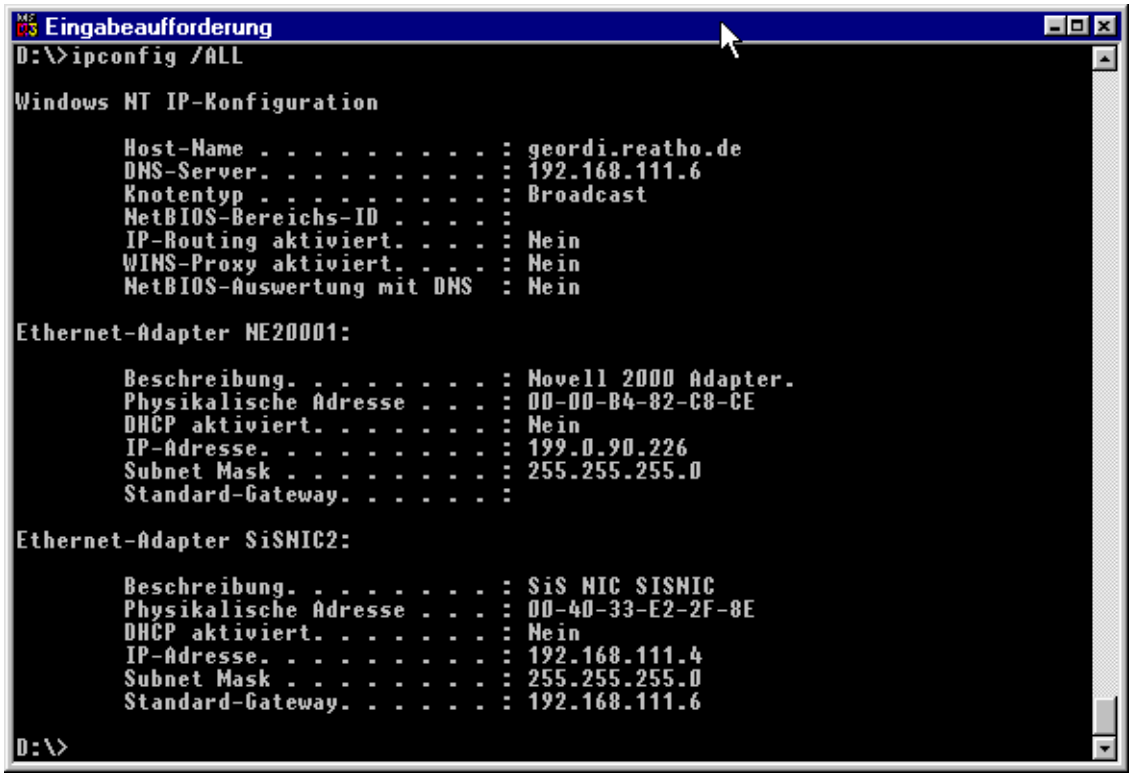
Network Monitoring

- Standardized model for information exchange:

- MAC-Address: 00-00-b4-82-c8-ce
  IP-Address: 192.168.111.1 with network mask ff:ff:ff:00 or /24
  TCP/UDP Ports: port no. at both client and server
- Hub: physical link between network adapters
  Switch: like a hub, but with filtering on the physical layer
  Router: connects networks on the IP layer

Network Monitoring

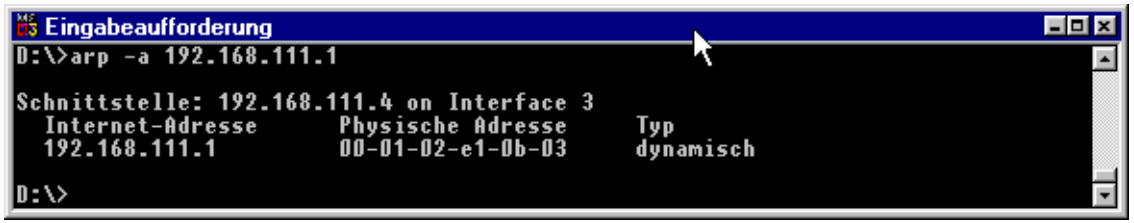
- Additional tools delivered by the operating system:
  - With all these tools, you get help through the option “-?“
  - ipconfig.exe - IP configuration of MS Windows



Network Monitoring

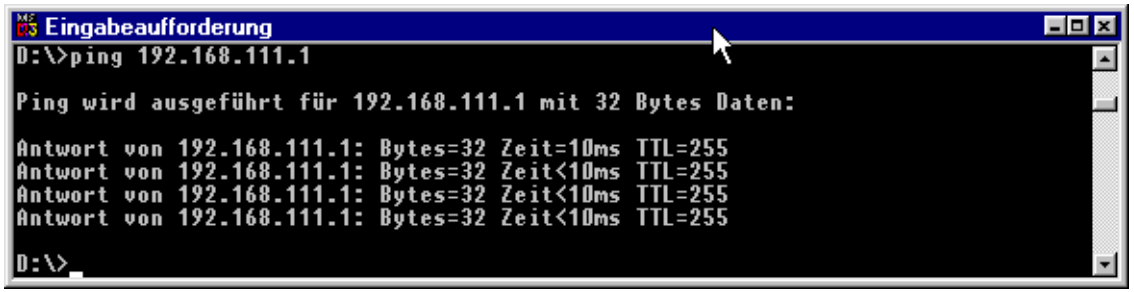
- arp.exe - IP/physical address mapping table (Address Resolution Protocol)
  Delivers list of all known pairs

Or only the physical address to an IP address



Network Monitoring

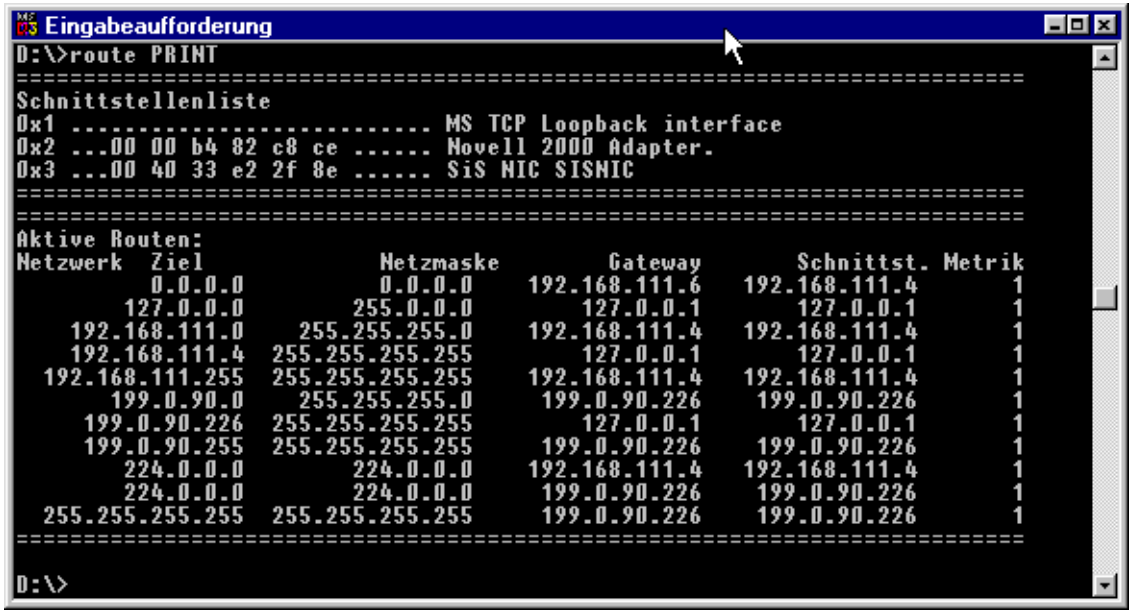
- ping.exe - checks accessibility on IP layer



- pathping.exe

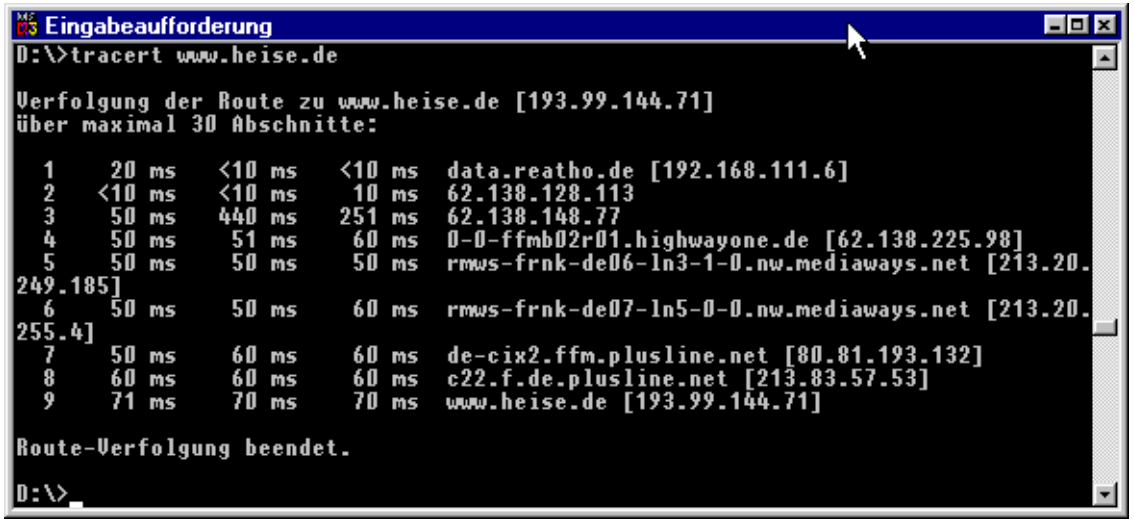
Network Monitoring

- route.exe - network-routing table
  Which target (host or network) is reachable using which gateway through which interface?



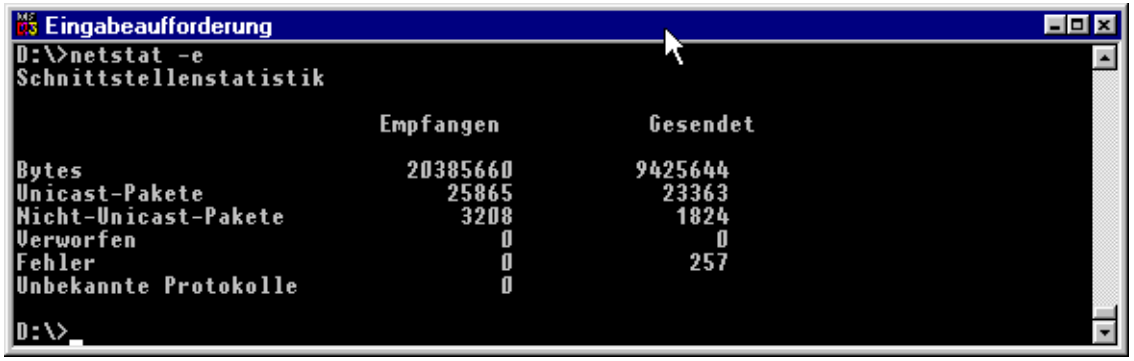
Network Monitoring

- tracert.exe - way to a host
  Show me all hosts on the way to my target

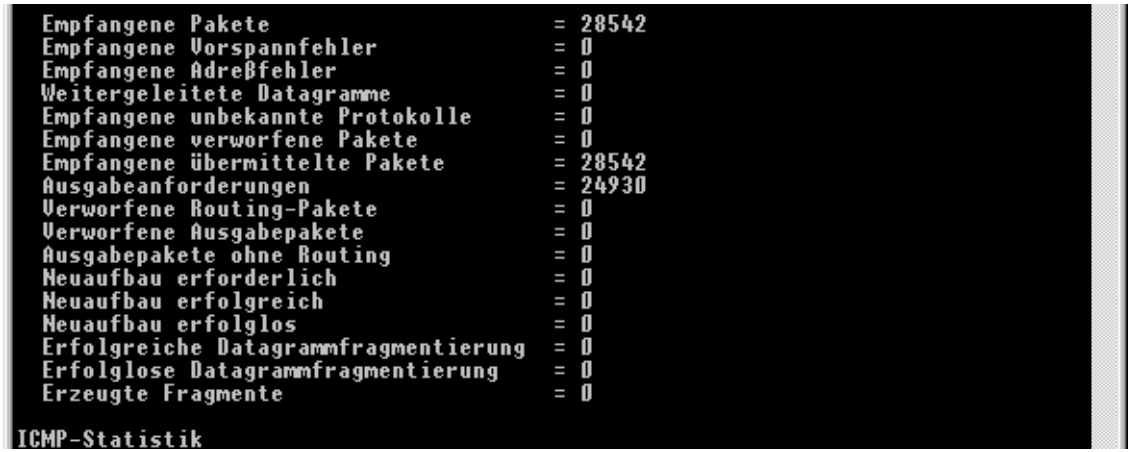


Network Monitoring

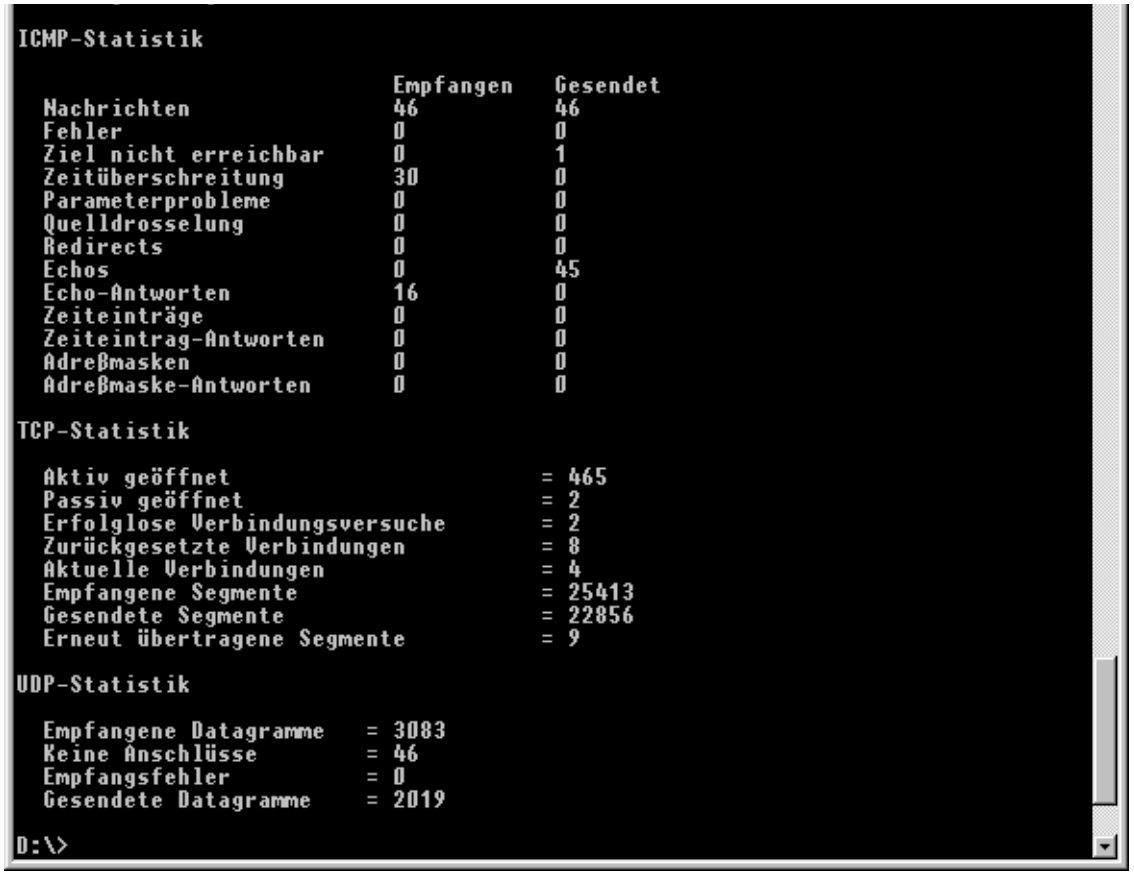
- netstat.exe -e
  Statistic Ethernet-Frames



- netstat.exe -s
  Statistic IP-Layer



Network Monitoring



Network Monitoring

- Typical connection establishment

```
11:58:36.777
    IP datagram: 171.26.177.141 -> 171.26.177.150
    TCP header:
      Port: 1034 -> 2404
      SeqNo=41421, AckNo=0
      Data Offset=6*32bit, Control Bits=0x02|urg ack psh rst SYN fin, Window=8192; Cksm=
        TCP Option: Kind=2|Maximum Segment Size, Length=4: 1460
11:58:36.777
    IP datagram: 171.26.177.150 -> 171.26.177.141
    TCP header:
      Port: 2404 -> 1034
      SeqNo=3813791165, AckNo=41422
      Data Offset=6*32bit, Control Bits=0x12|urg ACK psh rst SYN fin, Window=8192; Cksm=
        TCP Option: Kind=2|Maximum Segment Size, Length=4: 1460
11:58:36.777
    IP datagram: 171.26.177.141 -> 171.26.177.150
    TCP header:
      Port: 1034 -> 2404
      SeqNo=41422, AckNo=3813791166
      Data Offset=5*32bit, Control Bits=0x10|urg ACK psh rst syn fin, Window=8760; Cksm=
11:58:37.108
    IP datagram: 171.26.177.141 -> 171.26.177.150
    TCP header:
      Port: 1034 -> 2404
      SeqNo=41422, AckNo=3813791166
      Data Offset=5*32bit, Control Bits=0x18|urg ACK PSH rst syn fin, Window=8760; Cksm=

    U: STARTDT act
11:58:37.108
    IP datagram: 171.26.177.150 -> 171.26.177.141
    TCP header:
      Port: 2404 -> 1034
```

Network Monitoring

- While connection is established

```
11:58:37.168
    IP datagram: 171.26.177.150 -> 171.26.177.141
    TCP header:
      Port: 2404 -> 1034
      SeqNo=3813791172, AckNo=41444
      Data Offset=5*32bit, Control Bits=0x18|urg ACK PSH rst syn fin, Window=8192; Cksm

    I: SSN=0, RSN=1
      C_IC_NA_1 [actcon   +]     12        0  QOI=Station interrogation (global)
11:58:37.268
    IP datagram: 171.26.177.141 -> 171.26.177.150
    TCP header:
      Port: 1034 -> 2404
      SeqNo=41444, AckNo=3813791188
      Data Offset=5*32bit, Control Bits=0x10|urg ACK psh rst syn fin, Window=8738; Cksm
11:58:37.278
    IP datagram: 171.26.177.150 -> 171.26.177.141
    TCP header:
      Port: 2404 -> 1034
      SeqNo=3813791188, AckNo=41444
      Data Offset=5*32bit, Control Bits=0x18|urg ACK PSH rst syn fin, Window=8192; Cksm

    I: SSN=1, RSN=1
      M_SP_NA_1 [inrogen  +]     12      263  SPI=1|On          [iv nt sb bl]
                                         267  SPI=1|On          [iv nt sb bl]

    I: SSN=2, RSN=1
      M_SP_NA_1 [inrogen  +]     12    10016  SPI=0|Off         [iv nt sb bl]
                               (   10016+1 )  SPI=0|Off         [iv nt sb bl]
                               (   10016+2 )  SPI=0|Off         [iv nt sb bl]
                               (   10016+3 )  SPI=0|Off         [iv nt sb bl]
```

Network Monitoring

- Typical connection shutdown

```
11:58:36.527
    IP datagram: 171.26.177.150 -> 171.26.177.141
    TCP header:
      Port: 2404 -> 1033
      SeqNo=3802081444, AckNo=41503
      Data Offset=5*32bit, Control Bits=0x11|urg ACK psh rst syn FIN, Window=8192; Cksm=
11:58:36.527
    IP datagram: 171.26.177.141 -> 171.26.177.150
    TCP header:
      Port: 1033 -> 2404
      SeqNo=41503, AckNo=3802081445
      Data Offset=5*32bit, Control Bits=0x10|urg ACK psh rst syn fin, Window=8220; Cksm=
11:58:36.527
    IP datagram: 171.26.177.141 -> 171.26.177.150
    TCP header:
      Port: 1033 -> 2404
      SeqNo=41503, AckNo=3802081445
      Data Offset=5*32bit, Control Bits=0x11|urg ACK psh rst syn FIN, Window=8220; Cksm=
11:58:36.537
    IP datagram: 171.26.177.150 -> 171.26.177.141
    TCP header:
      Port: 2404 -> 1033
      SeqNo=3802081445, AckNo=41504
      Data Offset=5*32bit, Control Bits=0x10|urg ACK psh rst syn fin, Window=8192; Cksm=
```

Message Storage

- During simulation or monitoring, the entire protocol traffic is saved in the Message Storage
- The content of the Message Storage can be written into a file
- Always raw data is saved in the Message Log files, not processed data; for Offline Analysis all formatting options keeps available
- The protocol profile is also saved, so that when the file is loaded again, the right parameters are immediately set
- The current content of the Message Storage can be saved through *File -> Save as ...* ; don't forget to use the filename extension `.mlg` in the directory dialog

Logging

- Additionally, during simulation or monitoring the protocol traffic can be logged in a file (*File -> Log to file ...* and *File -> Close Log file*); this file can be of any length, allowing extensive recordings (that last days) to be created, that would otherwise not fit in the Message Storage

Access to Message Log files

- Through *File -> Open from ...* a Message Log file can be read
- A very long log file can be divided into smaller pieces through *Extra -> Split Log file ...*

Offline Analysis

- For postprocessing of the recorded protocol traffic
- The Offline Analysis **requires no license**
- Enables you to calmly go through the protocol traffic, to document and to draw the right conclusions
- Through *File -> Open from ...* a Message Log file can be retrieved again; with the cursor keys, the protocol traffic can be navigated, and using the shortcut keys the desired formatting options can be choosen

Exporting to file

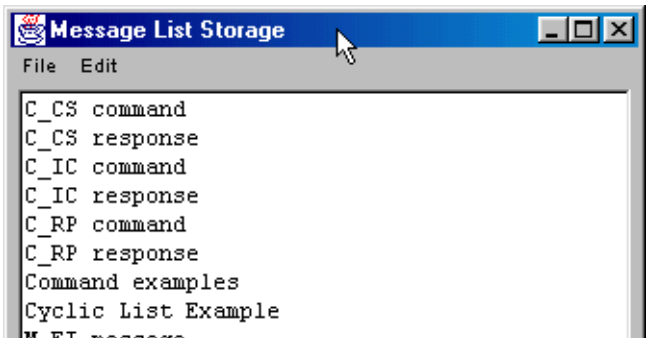
- *Extra -> Export to file ...* allows the content of the Message Storage to be exported to a file based on the previously choosen formatting options

Erase Message Storage

- *Clear* in the background menu deletes the Message Storage content

Message List Storage

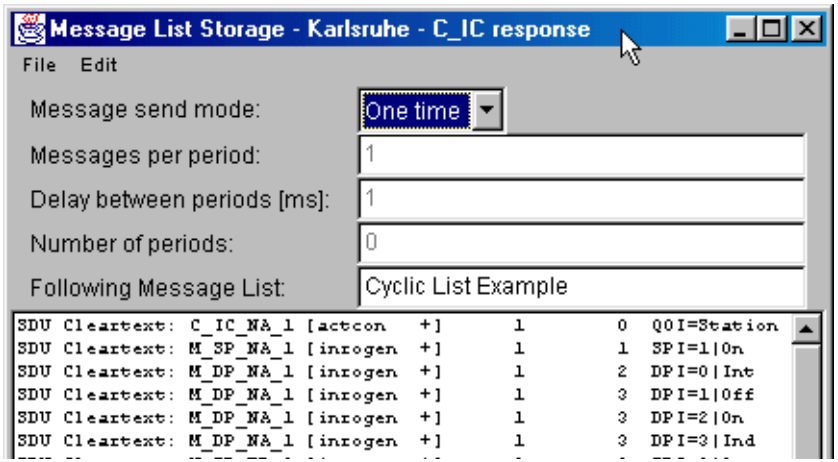
- The Message List Storage contains all the message lists
- Through *Edit -> Message List Storage ...* a view of the Message List Storage can be opened



- The message lists names can be choosen freely, but must be unique;
  in destinct protocols, destinct message lists have a special meaning, e.g. `C_IC response` contains the answer to a IEC 60870-5-101/-104 general interrogation
- In menu *File* through *New*, a new (empty) message list can be created, *Open from ...* imports message lists and *Save as ...* saves all message lists in Message List Storage in a file;
  the background menu and the menu *Edit* allow the editing, renaming, cutting, copying, pasting, sending and stopping of a message list;
  double-clicking on a list name opens a editing window for the list

Message Lists

- A single message list can be stored in a file (*File -> Save as ...*), by *Edit* meta-messages can be edited, copied, pasted, and deleted; double-clicking on a meta-message opens the editing window
- A number of attributes and a list of meta-messages together setup a message list (the interpretation depends on the choosen protocol!)



- The attributes determine:
  - If the list is processed once (`One time`) or cyclically (`Cyclic`)
  - In cyclic processing: how many messages should be sent in a period, the delay between periods and the number of periods
  - The next message list

Specific Message Lists

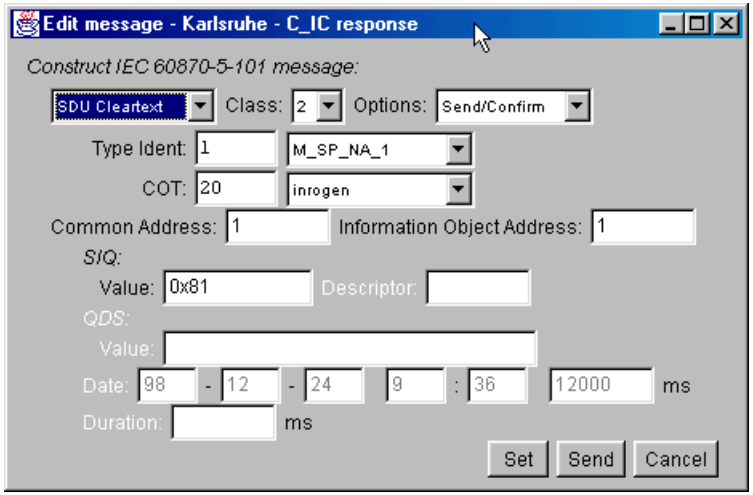
- Message list names reserved for distinct protocols:
  - IEC 60870-5-101/-104:
    - `M_EI message` - initialization message (after connection establishment)
    - `C_IC command` - general interrogation (after connection establishment)
    - `C_IC response` - response to general interrogation
    - `C_RP response` - response to reset process command
    - `C_CS response` - response to synchronization command
  - IEC 60870-5-102/-103:
    - `Init message` - initialization message (after connection establishment)
    - `GI request` - general interrogation (after connection establishment)
    - `GI response` - response to general interrogation
    - `ResetProcess response` - response to reset process command
    - `ClockSync response` - response to synchronization command
  - DNP3 and DNP3 over LAN/WAN:
    - `GI request` - general interrogation (after connection establishment)
    - `Reset Device Restarted` - command to reset the Device Restarted bit
    - `Timesync` - response to time synchronization request
    - `Read Class1` - response to indication class 1 data available
    - `Read Class2` - response to indication class 2 data available
    - `Read Class3` - response to indication class 3 data available

Specific Message Lists - Continuation

- ABB RP570/571:
  - `Init message` - initialization message (after connection establishment)
  - `Status Check request` - general interrogation (after connection establishment)
  - `Status Check response` - response to general interrogation
  - `Coldstart response` - response to reset process command (FCOM 1)
  - `VersionId response` - response to inquiry firmware version (FCOM 4)
- MODBUS:
  - `Cyclic Queries` - cyclic list with queries
  - `Response Read coils 1...8` - response; adjust value range
  - `Response Read input discretes 1...8` - response; adjust value range
  - `Response Read multiple registers 1...8` - response; adjust value range
  - `Response Read multiple registers 1...8` - response; adjust value range
  - `Response Read input registers 1...8` - response; adjust value range
  - `Response Read exception status`
  - `Response Read general reference 1/2/3/4` - response; adjust value range
  - `Response Read/write registers 1...8` - response; adjust value range
  - `Response Read FIFO queue 8` - response; adjust value range

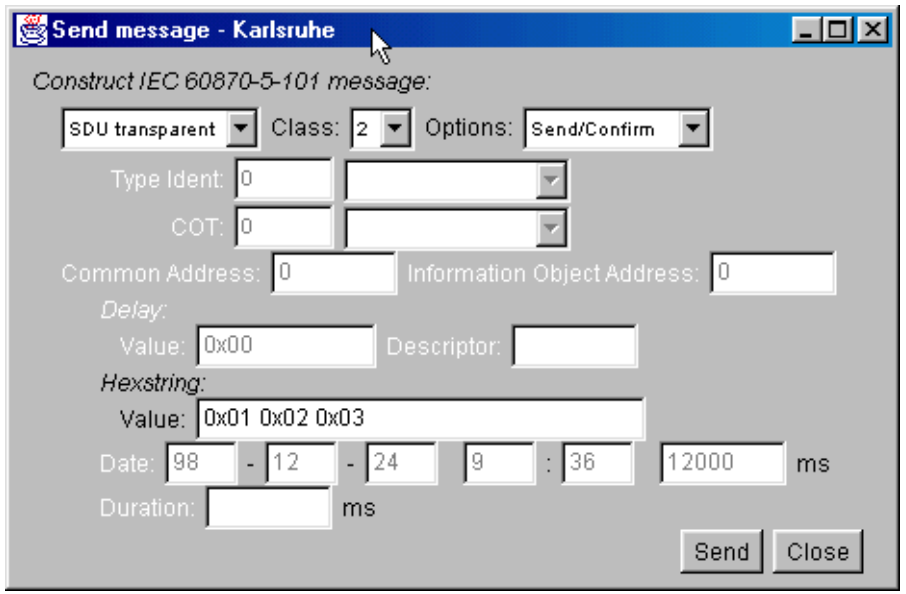
Meta-Messages

- The message lists are composed of single meta-messages;
  the meta-messages can consist of:
  - A complete frame (Link Layer frame) as a hexstring (PDU transparent)
  - Payload data (Application Layer message) in clear text or as a hexstring (SDU clear text or SDU transparent)
  - User String
  - Delay
  - Abrupt, uncontrolled connection break-off (Abort)
  - Controlled shutdown of the connetion (Shutdown)
- The clear text entry of messages offers a very fast and efficient way of entering message lists



Sending of single messages

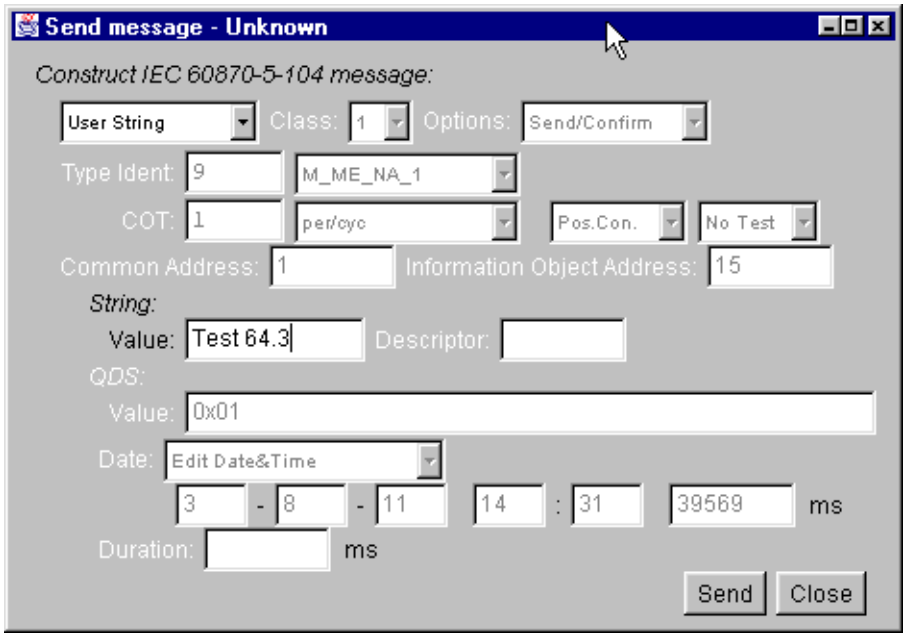
- Besides the possibility of editing and sending message lists, single messages can easily be sent;
  therefore, *Send message...* is available in the background menu of the display windows



- The advantage here is that the entry text boxes can be changed rapidly and the message can be sent immediately

User Comments

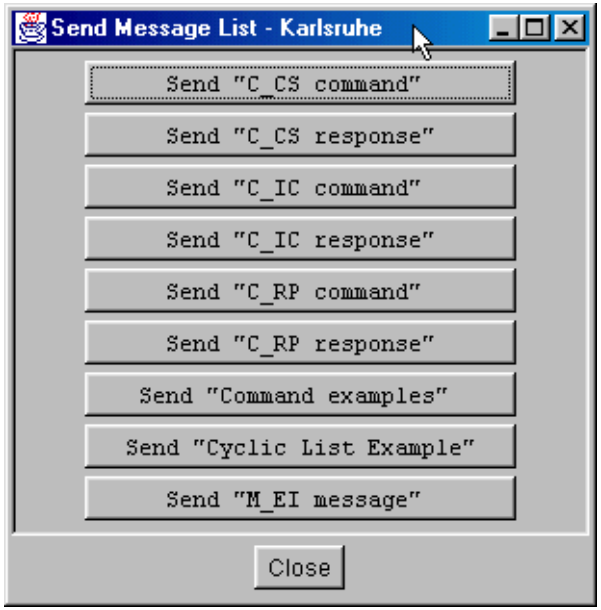
- Throught the window “Send Message“ it is possible to add user comments:



They are immediately added to the message storage

Sending Message Lists

- A single window allows it to easily activate message lists and to stop their execution; therefore, in the background select menu item *Send Message List ...*:



- Multiple message lists can be simultaneously executed; but every message list can only be processed once.
- Through *Send* the execution will begin and with *Stop* it will end

Preparing Configurations

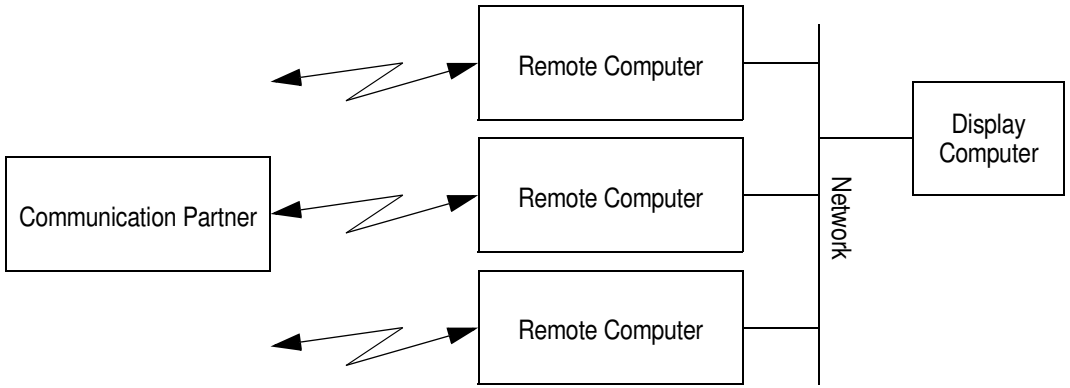
- Configurations can be prepared comfortably right from your desk and then saved; in doing so an important step in the preparation for an on-site customer visit can be previously solved
- The following belong to a configuration: protocol profile, loaded ***User Engine Classes***, general preferences and position/size of windows
- Save the configuration using *File -> Save Configuration as ...*, import the configuration using *File -> Open Configuration from ...*
- If ***CPTT*** must run several times on the desktop with the same configurations, there is a trick: start ***CPTT***, set configuration and then save it using *File -> Save Configuration to user default*; the new user configuration has been saved; now you can run ***CPTT*** as often as wanted... all ***CPTTs*** have the same configuration

User Engine Classes

- Are plug-ins for ***CPTT***, that are implemented in Java. In the Programmer's Guide the API of the ***User Engine Classes*** is described; a precondition for the development of ***User Engine Classes*** is an installed Java Development Kit
- Because Java is very similar to ANSI C and C++, it is easy to write ***User Engine Classes***
- Java helps to protect programs against crashes from ***User Engine Classes***
- An example for ***User Engine Classes*** will be delivered:
  IEC 60870-5-101/-104-Filetransfer
- Through *Edit -> User Engine Classes ...* you are able to choose and load ***User Engine Classes***; if the path in the text field is cleared, then the class will be unloaded again
- Many other applications are imaginable:
  - File transfer and other Application Layer procedures
  - Recreating of device specific pseudo points
  - Controlling the basic functions of a device

Remote I/O Server

- The Remote I/O Server **separates the display from the reception of protocol traffic**. It enables data streams, which are not directly available on a *CPTT*-running display computer but can be monitored by remote computers, to be analyzed and displayed by *CPTT*. The *RIO Server* exchanges data with a communication partner through the remote computer's communication interface (serial interface, network, files). The data frames are handed off to *CPTT* through the network, to be analyzed and displayed.



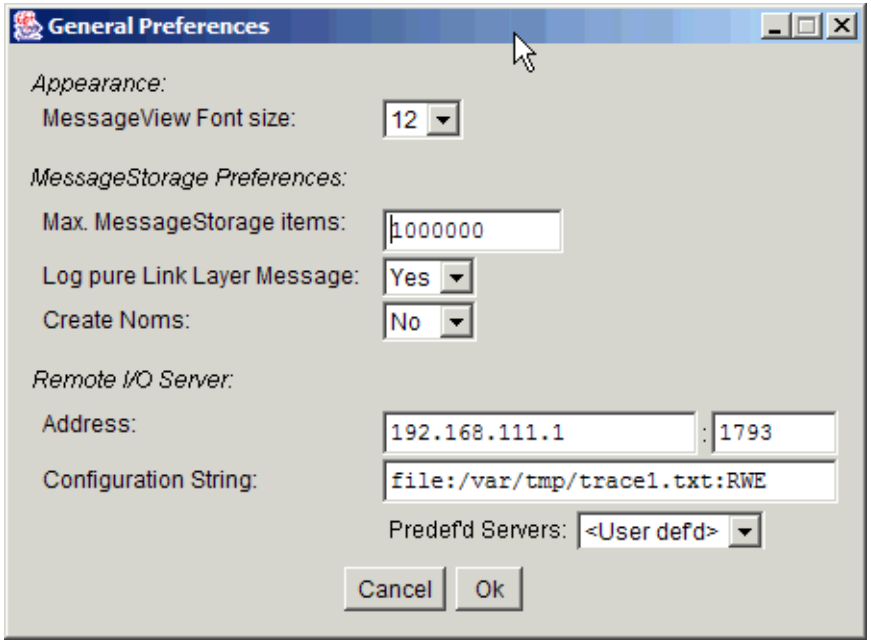
- The *RIO Server* is an independent program, that operates separately from *CPTT* on another computer. While *CPTT* is limited to an MS Windows computer, the light weight *RIO Server* runs on many computer architectures (MS Windows, Linux, Solaris, ...).

RIO Server

- The executable version of ***RIO Servers*** for the required computer architecture can be found on the CD-ROM in the directory `\CPRioServer`:
  - `CPRioServer_iX86_WIN32` for MS Windows
  - `CPRioServer_iX86_Linux` for PC Linux 2.x
  - `CPRioServer_sun4_SunOS_5` for Sun SPARC Solaris 7
- Copy the executable version on your target system.
- Start the ***RIO Server*** on the remote computer. So that the ***RIO Server*** is always available, it should be started when the computer is booted or the supervised application is started:
  - Under MS Windows: Put RIO Server in the registry
  - Under UNIX or Linux within the `/etc/rd.d` scripts
  - Or through a script related to the supervised application

RIO Client in CPTT

- The configuration of the ***RIO Clients*** in ***CPTT*** can be carried out through *Edit->General Preferences*:



  - In this window, the address (computer name or IP address) of the ***RIO Servers***, the TCP port no. and the configuration strings can be entered. The configuration string must be specified from the ***RIO Servers*** perspective (see below).

RIO Client in CPTT

- The following applies for the inputs:
  - If there is a correct ***RIO Server*** address given, with immediate effect communication takes place through the ***RIO Server***. If you like to continue without ***RIO Server***, just clear the address.
  - If predefined settings (as CPTT configuration files) were read, then these can be selected from *Predef'd Servers* and accepted.
- "@Rio" in the title of the window shows that the communication occurs indirectly through the ***RIO Server***.
- Predefined ***RIO Server*** configuration can be read through *File->Open Configuration from*. The maintenance of these configurations must occur through an external editor.
- The configuration string have the following structure:
  <Medium>:<location>[;<location]:<Format>
  - `dblsrl:com1,9600,8n1;com2,9600,8n1:PPP`
    From two serial interfaces (com1 and com2 each running on 9600 baud, 8 data-, 1 stop- and no parity bits) in format PPP (Point-to-Point Protocol) successive PPP-frames are caught and forwarded to ***CPTT***.
  - `file:/var/tmp/trace1.txt;/var/tmp/trace2.txt:RWE`
    From a file, path and filename is "/var/tmp/trace1.txt" or "/var/tmp/trace2.txt", in format RWE all added data frames are consecutively read out and forwarded to ***CPTT***.

We have observed...

- Using *Edit -> Protocol Profiles ...* a protocol can be choosen from a protocol family, and its profile can be determined,
- the simulation as Controlling Station/Master or as Controlled Station/Slave is started through *Operate -> as Controlling/as Master* or *Operate -> as Controlled/as Slave*,
- the simulation can be finished again through *Operate -> Stop*,
- the display formatting of the protocol traffic can be changed using the *background menu* (right mouse button) -> *Formatting Options ...*
- the messages can be saved on a disk (*File -> Save as ...*) and be opened again (*File -> Open from ...*)
- The size of the Message Storage can be controlled using *Edit -> General Preference,* as well as whether all Link Layer information should be saved
- For long lasting tests, the protocol traffic can be saved directly into a file through *File -> Log to file ...*, logging will be stopped through *File -> Close Log file*
- This is specially useful during monitoring (*Operate -> Monitor* und *Operate -> Stop*)
- Large Message Log files can be split into smaller files using *Extra -> Split Log file ...*,
- the content of the Message Storage, with the selected formatting options, can be saved in a file by using *Extra -> Export to file*

We have observed... - Continuation

- Message lists are processed by *Edit -> Message List Storage ...*, ***User Engine Classes*** (plug-ins) can be added to ***CPTT*** through *Edit -> User Engine Class ...*
- For questions, *Help -> About* gives you the contact information to contact us!

Real Thoughts GmbH
Haid-und-Neu-Straße 7
76131 Karlsruhe
Germany

Fon +49-721-6276730, Fax +49-721-6276731

Website www.realthoughts.de
E-mail info@realthoughts.de