

COMPROTware:Testtool

Integriertes Test- und Analysetool für Fernwirktechn. Protokolle

Seit Mitte der 1990er Jahre hat ein umfassender Wechsel hin zu offenen und standardisierten Fernwirkprotokollen wie die IEC 60870-5-Protokollfamilie oder DNP3 stattgefunden. Ziel ist die Anbieterunabhängigkeit: Geräte und Systeme beliebiger Hersteller sollen in eine optimale Gesamtlösung integrierbar werden.

Mit **COMPROTware:Testtool** bietet **Real Thoughts** das herstellerunabhängige Test- und Analysewerkzeug für Fernwirkprotokolle in diesem Umfeld an.

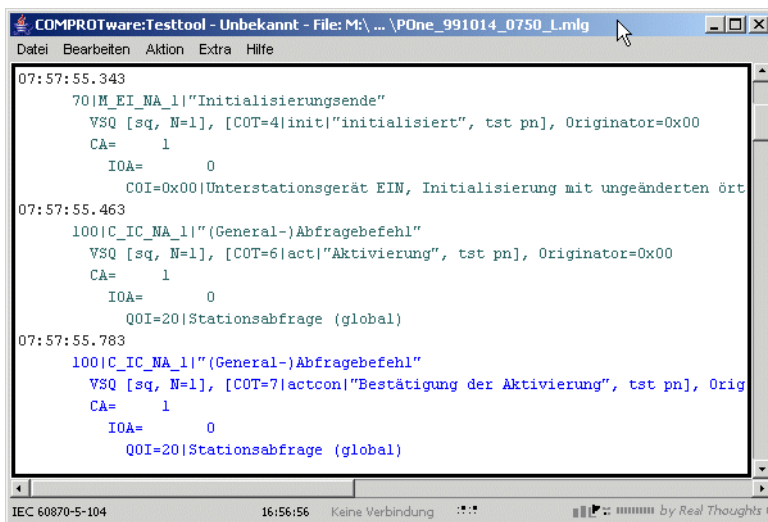
COMPROTware:Testtool kann sowohl die Zentral- oder Unterstation eines Fernwirksystems simulieren als

Look & Feel um den Anwender effizient bei seiner Aufgabe des Implementierens und Testens von Protokollen, der Auslieferung eines Leitsystems/einer Unterstation oder der Sicherung der Netzverfügbarkeit zu unterstützen.

- **COMPROTware:Testtool** läuft auf Microsoft Windows 2000, XP, Vista und 7.
- Ebenfalls erhältlich für PC Linux (auf Nachfrage).
- Verfügbare Sprachen sind Deutsch, Englisch, Französisch, Italienisch und Spanisch.

Ein Grundkonzept von **COMPROTware:Testtool** (kurz **CPTT**) ist es, offen zu sein:

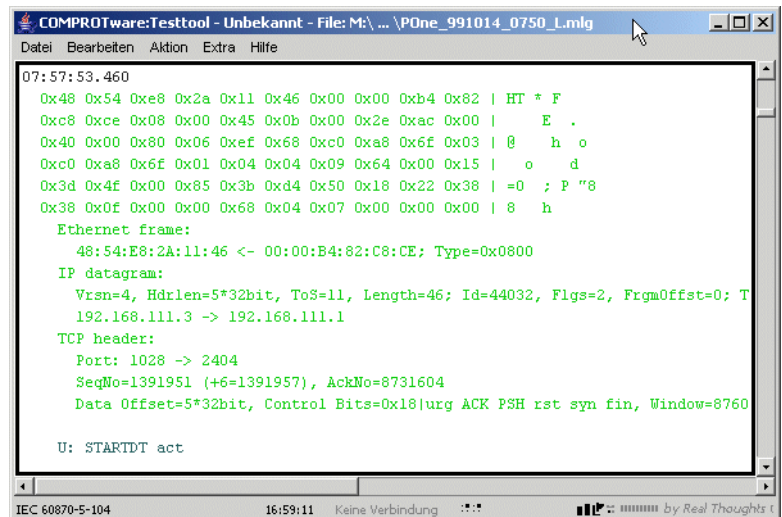
- Alle Konfigurationsdateien sind textbasiert und leicht zu verstehen. Damit kann eine externe Software einfach eigene Konfigurationen erzeugen.
- User Engine Classes (Plug-Ins) erlauben es alle Arten von benutzer-spez. Prozeduren in das Testtool zu integrieren.
- Zur nachträglichen Analyse inkl. Nachrichten-Decoder bedarf es keiner Lizenz: **CPTT** kann auf jedem Arbeitsrechner installiert und auch beliebig weitergegeben werden.
- Import aus einer Textdatei/von Pcap-Dateien und Export nach HTML, MS Word, Wireshark oder ASCII hält **CPTT** in Kontakt mit der Arbeitsumgebung.
- Über den Remote I/O Server lässt sich auch die Kommunikation von einem entfernten System mitprotokollieren.



```
COMPROTware:Testtool - Unbekannt - File: M:\... \POne_991014_0750_L.mlg
Datei Bearbeiten Aktion Extra Hilfe
07:57:55.343
70|M_EI_NA_1|"Initialisierungsende"
VSQ [sq, N=1], [COT=4|init|"initialisiert", tst pn], Originator=0x00
CA= 1
IOA= 0
COI=0x00|Unterstationsgerät EIN, Initialisierung mit ungeänderten ört
07:57:55.463
100|C_IC_NA_1|(General-)Abfragebefehl"
VSQ [sq, N=1], [COT=6|act|"Aktivierung", tst pn], Originator=0x00
CA= 1
IOA= 0
QOI=20|Stationsabfrage (global)
07:57:55.783
100|C_IC_NA_1|(General-)Abfragebefehl"
VSQ [sq, N=1], [COT=7|actcon|"Bestätigung der Aktivierung", tst pn], Orig
CA= 1
IOA= 0
QOI=20|Stationsabfrage (global)
IEC 60870-5-104 16:56:56 Keine Verbindung by Real Thoughts
```

auch die Kommunikation zwischen beiden mitverfolgen. Man erhält dabei eine kontinuierliche Echtzeitdarstellung mit einer flexiblen Ausgabe der verschiedenen Kommunikationsschichten. Im Nachrichtenverkehr kann während der Simulation/des Mithörens beliebig navigiert, nach Text gesucht, die Darstellung gewechselt und Filter verändert werden. Jederzeit ist auch das Speichern in eine Datei möglich. Nachrichtenlisten können einmalig oder periodisch oder in einer Folge verschickt werden, eine Nachricht kann dabei auch beliebig fehlerhafte Informationen enthalten. Die nachträgliche (Off-line) Analyse anhand von Dateien erlaubt es, in aller Ruhe die richtigen Schlüsse zu ziehen.

Das Integrierte Testtool hat für sämtliche Übertragungsprotokolle das gleiche State-of-the-Art, fensterbasierte



```
COMPROTware:Testtool - Unbekannt - File: M:\... \POne_991014_0750_L.mlg
Datei Bearbeiten Aktion Extra Hilfe
07:57:53.460
0x48 0x54 0xe8 0x2a 0x11 0x46 0x00 0x00 0xb4 0x82 | HT * F
0xc8 0xce 0x08 0x00 0x45 0x0b 0x00 0x2e 0xac 0x00 | E .
0x40 0x00 0x80 0x06 0xef 0x68 0xc0 0xa8 0x6f 0x03 | @ h o
0xc0 0xa8 0x6f 0x01 0x04 0x04 0x09 0x64 0x00 0x15 | o d
0x3d 0x4f 0x00 0x85 0x3b 0xd4 0x50 0x18 0x22 0x38 | =0 ; P "8
0x38 0x0f 0x00 0x00 0x68 0x04 0x07 0x00 0x00 0x00 | 8 h
Ethernet frame:
48:54:E8:2A:11:46 <- 00:00:B4:82:C8:CE; Type=0x0800
IP datagram:
Vrsn=4, Hdrlen=5*32bit, ToS=11, Length=46; Id=44032, Flgs=2, FrgmOffset=0; T
192.168.111.3 -> 192.168.111.1
TCP header:
Port: 1028 -> 2404
SeqNo=1391951 (+6=1391957), AckNo=8731604
Data Offset=5*32bit, Control Bits=0x18|urg ACK PSH rst syn fin, Window=8760
U: STARTDT act
IEC 60870-5-104 16:59:11 Keine Verbindung by Real Thoughts
```

Eigenschaften

- Simulation von Master oder Slave Station bzw. Controlling oder Controlled Station.
 - ▲ Kontinuierliche Echtzeitdarstellung von Nachrichten (bidirektional, mit Zeitstempel) und von Schnittstellen-/Protokollmaschinen-Ereignissen.
 - ▲ Interaktives Erstellen der Sendenachrichten (auch während des Betriebs) von logischer bis hin zur Byteebene (um fehlerhafte Daten erzeugen zu können).
 - ▲ Einmaliges oder periodisches Senden von ausgewählten Nachrichtenlisten, die auch zu komplexen Abfolgen verknüpft werden können.
 - ▲ Mitschreiben, Speichern und Einlesen des Nachrichtenverkehrs zur nachträglichen Analyse.
 - ▲ Spezielle Operationsmodi und Parametersätze um die Protokollentwicklung zu unterstützen (z.B. für Stresssituationen, zur Lastanalyse, für Dauertests, ...).
- Mithören des Nachrichtenverkehrs (für serielle und netzwerk-basierte Kommunikation).
 - ▲ Kontinuierliche Echtzeitdarstellung von Nachrichten (bidirektional, mit Zeitstempel) und von Schnittstellen-Ereignissen.
 - ▲ Mitschreiben, Speichern und Einlesen des Nachrichtenverkehrs zur nachträglichen Analyse.
- Nachträgliche (Off-line) Analyse des Nachrichtenverkehrs.
 - ▲ Zugriff auf gespeicherten Nachrichtenverkehr.
 - ▲ Export des Nachrichtenverkehrs in Text-, HTML-, MS Word- oder Pcap-Dateien.
 - ▲ Import aus Text- oder Pcap-Dateien.
- Graphische Benutzeroberfläche.
 - ▲ Konfiguration des Protokollprofils und der Protokollmaschine.

- ▲ Jederzeit (auch während Simulation und Mithören) kann im Nachrichtenverkehr navigiert, beliebiger Text gesucht, die Darstellung gewechselt oder Filter gesetzt werden.
 - ▲ Protokoll-spez., für den Anwender leicht verständliche Formatierung.
- Performante und effiziente Implementierung der Simulation und der Nachrichtendarstellung gepaart mit robuster und intuitiver Graphischer Benutzeroberfläche.
 - User Engine Classes (Plug-Ins) für benutzereigene Verarbeitungsfunktionen.
 - Keine weitere Schnittstellen-Hardware notwendig.
 - Realisierte Protokolle (Stand März 2009):
 - ▲ IEC 60870-5-101 Ed. 2.0
 - ▲ IEC 60870-5-102
 - ▲ IEC 60870-5-103
 - ▲ IEC 60870-5-104 Ed. 2.0
 - ▲ IEC 61850 MMS und GOOSE (nur Mithören)
 - ▲ DNP3
 - ▲ DNP3 over LAN/WAN (TCP od. UDP-basiert)
 - ▲ MODBUS (RTU und ASCII)
 - ▲ MODBUS TCP/IP
 - ▲ ABB RP570/571
 - ▲ ABB SPA-Bus
 - ▲ Landis&Gyr TG 809(Die aktuelle Liste der unterstützten Protokolle ist auf unserer Website zu finden.)

COMPROTware:Testtool basiert darauf, dass die Implementierung eines Integrierten Testtools für fernwirktechn. Übertragungsprotokolle Vorteile beim Einsatz eines jeden einzelnen Protokolls bringt: Kennt der Anwender für ein Protokoll das Werkzeug und dessen Einsatzmöglichkeiten, dann kennt er es auch für alle weiteren. Dieser Grundsatz erlaubt dem Anwender auf gleichbleibend hohem Standard die Entwicklung, Inbetriebnahme und Wartung von RTUs, IEDs und Leitsystemen.

Kontakt zu uns ...

Real Thoughts GmbH

Haid-und-Neu-Straße 7, 76131 Karlsruhe, Germany

Vanity-No. +49-700-REALTHOU

Fon +49-721-6276730

Fax +49-721-6276731

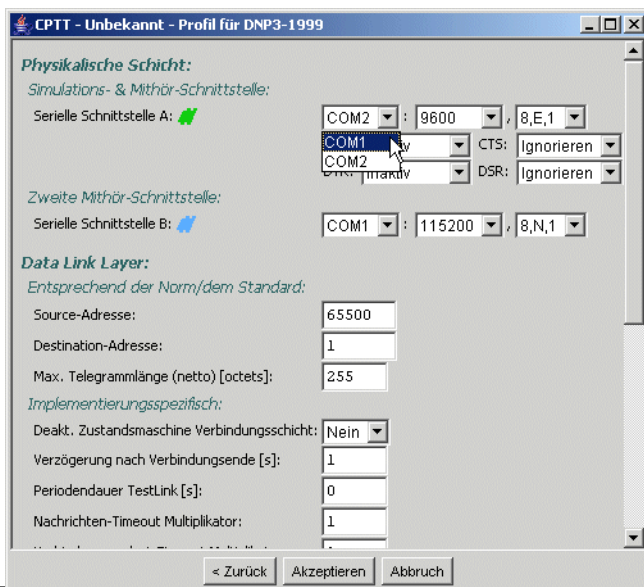
Website www.realthoughts.de

E-Mail info@realthoughts.de

Irrtum und Änderungen vorbehalten. Stand: August 2011

Real Thoughts
GmbH

„Echtzeitapplikationen und Graphikanwendungen“



Besondere Merkmale

■ Allgemein:

- ▲ **COMPROTware:Testtool** (kurz **CPTT**) ist ein **Integriertes Test- und Analysetool** für fernwirtsch. Übertragungsprotokolle. Es unterstützt den Anwender bei seiner Aufgabe den sicheren Betrieb eines Fernwirtsystems zu gewährleisten, ist nützliches Werkzeug bei der Implementierung von Protokollen, sorgt für die effiziente Inbetriebnahme von Leistsystemen und Unterstationen, hilft bei der Fehlersuche und Problemanalyse und ermöglicht den Test von neuen Geräten und Protokollen.
- ▲ **CPTTs Graphische Benutzeroberfläche** ist **intuitiv** und **effizient** zu **handhaben** und sorgt für den einfachen Umgang mit dem dargestellten Nachrichtenverkehr.
- ▲ **CPTT** integriert **alle Protokolle in einem einzigen Tool**. Dadurch kann schnell zwischen den Protokollen gewechselt werden. Und die Bedienung des einen Protokolls entspricht auch der Bedienung jedes anderen.
- ▲ **CPTT** ist eine **moderne Software**, deren Kern in C++ und deren Graphische Benutzeroberfläche in Java implementiert ist. Dabei werden die Vorteile beider Sprachen gewinnbringend vereint um dem Anwender eine **zuverlässige, robuste und vertrauenswürdige Software** an die Hand zu geben.
- ▲ **CPTT** ist **zukunftsicher**, denn **CPTT** kann auf Microsoft Windows 2000, XP, Vista und 7 (jeweils als 32Bit Applikation) eingesetzt werden. Eine PC Linux-Fassung kann auf Anfrage bereitgestellt werden.
- ▲ **CPTT** ist **mehrsprachig**: Deutsch, Englisch, Französisch, Italienisch und Spanisch.
- ▲ **CPTT** kann **mehrfach auf einem Rechner** betrieben werden, jede Instanz kann über eine eigene Schnittstelle und mit einem anderen Protokoll/Profil mit einer Gegenstation kommunizieren.
- ▲ Die Grundeinstellungen im Protokollprofil führen zusammen mit den Message List-Vorlagen der Installation und dem Trainingsdokument zu einer **umgehend einsetzbaren Simulationsumgebung**, so dass ohne aufwendige Vorbereitungen mit der Arbeit begonnen werden kann.
- ▲ Die **CPTT-Konfiguration** bestehend aus den **Voreinstellungen**, dem **Protokollprofil** und den **Message Lists** kann **gespeichert** und **wiedereingelesen** werden. Einmal ermittelte gerä-

te-spez. Einstellung können somit schnell wiederhergestellt werden.

- ▲ Die Installation (unter MS Windows) erfolgt über **Microsoft Windows Installer**, dem Installations- und Konfigurationsdienst von Microsoft.
 - ▲ Durch die **Integration** von **CPTT in Microsoft Windows** kann effizient mit Dateien gearbeitet werden, ohne dass ein Bruch in der Bedienphilosophie auftritt.
 - ▲ Alle Konfigurations- und Beschreibungsdateien (nicht die Message Log-Dateien) sind textbasiert und leicht zu verstehen, durch die Import-/Export-Funktionalität ist **CPTT** bereit für den Informationsaustausch mit Anderen und durch die User Engine Classes beliebig erweiterbar. Damit ist **CPTT offen, flexibel und anpassungsfähig für die Zusammenarbeit mit anderen Anwendungen**.
 - ▲ **CPTT** wird **ständig erweitert und gepflegt**: Das **Feedback der Anwender** zusammen mit unserer Erfahrung (aus eigenen Projekten) fließt fortlaufend in das Werkzeug ein und sorgt für eine stetige Fortentwicklung. Technologische Veränderungen und Neuerungen bei Protokollen wie auch bei Betriebssystemen werden umgehend in **CPTT** eingebracht.
 - ▲ Wir erweitern **CPTT** sukzessive um neue Protokolle und Funktionalitäten.
 - ▲ Über unsere Website <http://www.realthoughts.de/> kann **CPTT** aus dem Internet aktualisiert, die aktuelle Approved Release oder der neueste Entwicklungsschnappschuss heruntergeladen werden.
- Realisierte Protokolle:
- ▲ Die vollständige IEC 60870-5-Familie: **IEC 60870-5-101 Ed. 2.0, -102, -103 und -104 Ed. 2.0**.
 - ▲ **IEC 61850 MMS and GOOSE (nur Mithören)**.
 - ▲ **DNP3** und **DNP3 over LAN/WAN**.
 - ▲ **MODBUS** und **MODBUS TCP/IP**.
 - ▲ **ABB RP570/571** und **ABB SPA-Bus**.
 - ▲ **Landis&Gyr TG 809**.
(Die aktuelle Liste der unterstützten Protokolle ist auf unserer Website zu finden.)
 - ▲ Sobald irgendein Protokoll lizenziert ist, steht das Protokoll **Raw data** zur Verfügung. Über dieses (Pseudo-)Protokoll können beliebige Zeichenströme mitgehört werden und eine einfache Simulation auf Basis von hexadezimalen Zeichenketten ist möglich.
 - ▲ Über die Graphischen Benutzeroberfläche wird

ein Protokoll aus einer Protokollfamilie ausgewählt und das Profil (gemäß Standard und erweitert um **CPTT**-spez. Parameter) bearbeitet.

- ▲ Bei allen Übertragungsprotokollen können Zentralstation (**Master, Controlling Station**) oder Unterstation (**Slave, Controlled Station**) eines

Fernwirksystems simuliert und der Nachrichtenverkehr **mitgehört** werden.

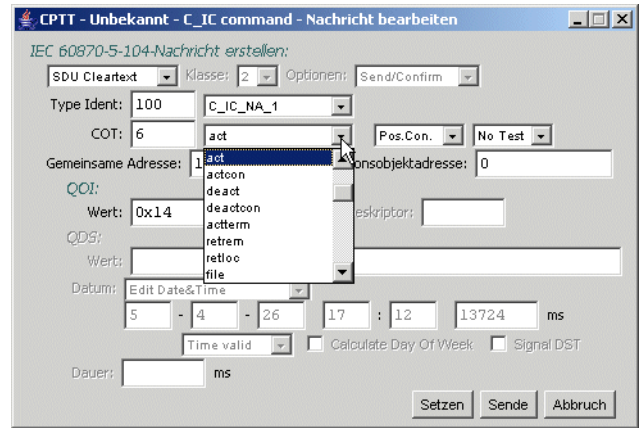
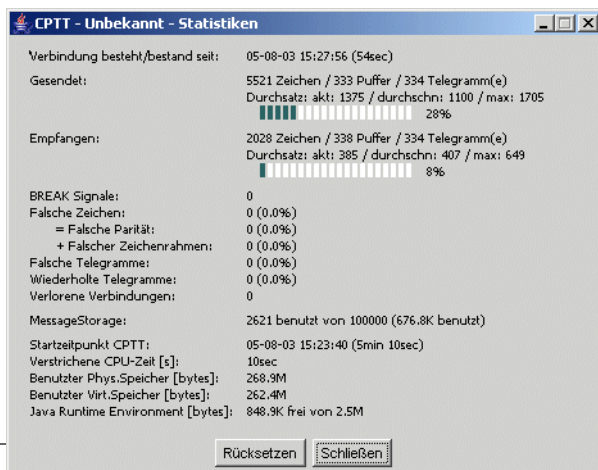
- Zu IEC 60870-5-101/-104:
 - ▲ Die Implementierung basiert auf den Normen **IEC 60870-5-101 - Ed. 2.0, März 2003** und **IEC 60870-5-104 - Ed. 2.0, Juni 2006**.
 - ▲ Die IEC 60870-5-101-Übertragungsverfahren symmetrisch und unsymmetrisch sind implementiert, ebenso alle Adress-/Feldlängen auf Verbindungsschicht- und Anwendungsschicht.
 - ▲ **Alle Typkennungen, Übertragungsursachen, ...** von IEC 60870-5-101 **Ed 2.0** und -104 sind realisiert.
 - ▲ **Eingabe** von Strukturierten Adressen und **Darstellung** beliebig formatierbarer **Strukturierter Adressen** (mit beschreibendem Text für jede Ebene).
 - ▲ Eine Implementierung des Filetransfers wird als Beispiel einer User Engine Class mitgeliefert.
- Zu IEC 60870-5-102:
 - ▲ Die Implementierung basiert auf der Norm **IEC 60870-5-102 - Ed. 1.0, Juni 1996**.
- Zu IEC 60870-5-103:
 - ▲ Die Implementierung basiert auf der Norm **IEC 60870-5-103 - Ed. 1.0, Dezember 1997**.
- Zu IEC 61850:
 - ▲ Die Implementierung basiert auf der Norm

IEC 61850-8-1 - Ed. 1.0, Mai 2004.

- ▲ Die Mappings MMS und GOOSE sind implementiert.
- ▲ Das Mithören ist implementiert, Master- bzw. Slave-Simulation stehen im Moment nicht zur Verfügung.
- Zu DNP3 und DNP3 over LAN/WAN:
 - ▲ Die Implementierung basiert auf den **DNP3 Basic 4 Documents plus Technical Bulletins**.
 - ▲ **Verbindungs-, Transport- und Anwendungsschicht** sind **vollständig implementiert**, **Unsol. Response** werden unterstützt, alle Object Groups/Variations bis **Level 3** sind realisiert, DNP3 over LAN/WAN sowohl für **TCP/IP** als auch für **UDP/IP**.
- Zu MODBUS:
 - ▲ Die Implementierung erfolgte auf Basis des **MODICON-Standard-Dokuments, Rev. J, Juni 1996**.
 - ▲ Es sind die beiden Übertragungsmodi RTU und ASCII realisiert.
 - ▲ **Automatisches Generieren von Antworten** auf Anforderung bei Unterstationssimulation.
- Zu MODBUS TCP/IP:
 - ▲ Die Implementierung erfolgte auf Basis des **MODBUS Messaging on TCP/IP Implementation Guide V1.0b, 2006**.
 - ▲ **Automatisches Generieren von Antworten** auf Anforderung bei Unterstationssimulation.
- Zu ABB RP57x:
 - ▲ Die Implementierung erfolgte auf Basis des Dokuments **RTU Protocol 570 and 571, Rev. 09, August 1998** und **Implementation of the RTU Protocol 570 and 571, Rev. 02, November 1998**.
- Zu ABB SPA-Bus:
 - ▲ Die Implementierung erfolgte auf Basis des Dokuments **SPA-Bus Communication Protocol V2.5, Januar 1996**.
 - ▲ **Automatisches Generieren von Antworten** auf Anforderung bei Unterstationssimulation.
- Zu Landis&Gyr TG 809:
 - ▲ Die Implementierung basiert auf der **TG 809-US Konventionsbeschreibung von 1995**.
- Message View:
 - ▲ Ist das Fenster mit der **Darstellung der Nachrichten**, die im **Message Storage gespeichert** sind.
 - ▲ **Der Klartext** für Nachrichten wird vom Nachrichten-Decoder erst zur Darstellung selbst aus den Rohdaten bestimmt. Dabei stehen eine Vielzahl

von Darstellungsmodi und Filtern zur Verfügung:

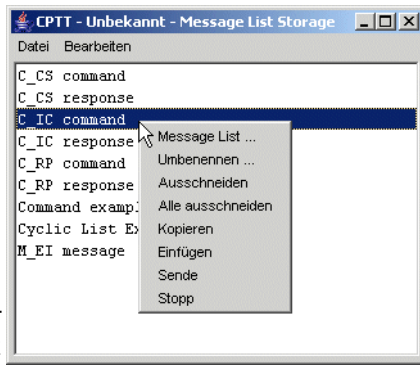
- Der digitale Zeichenstrom ist in eine **für den Anwender leicht erfassbare Darstellung** aufbereitet: Bits werden in Klartext gewandelt, Messwerte in der korrekten Repräsentation wiedergegeben. Dabei bleibt **CPTT** immer dicht an den Vorgaben (Bezeichnungen, Konventionen, ...) des realisierten Übertragungsprotokolls.
- Die Farbe der Nachrichten gibt deren Ursprung an. Fehlerhafte Daten und Ausgaben zu Fehlern werden in Rot dargestellt.
- Nachrichten können von einer **prägnanten, einzeiligen** bis hin zu einer **umfangreichen, mehrzeiligen** Darstellung angezeigt werden.
- Mit oder ohne **Verbindungsschicht**.
- Mit oder ohne **Hexdump** der Nachricht.
- **Zeitstempel** als Uhrzeit oder auch mit Datum.
- Bei IEC 60870-5-104, IEC 61850, MODBUS TCP/IP oder DNP3 over LAN/WAN (über TCP/IP oder UDP/IP) kann bei mitgehörten Nachrichten zusätzlich der **Ethernet Frame** und der **TCP/UDP Header** angezeigt werden.
- **Filter für Unterstationsadresse, Nachrichtentyp, ...** reduzieren die dargestellte Informationsmenge auf das vom Anwender gewünschte, überschaubare Maß. **Wellenlinien** markieren die Stelle im Message View, an der **Nachrichten gefiltert wurden**.
- ▲ Die Cursortasten (Pfeiltasten) erlauben die **schnelle Navigation** durch die Nachrichten inkl. Bild hoch/runter, Anfang und Ende. Am Ende angelangt „rastet“ die Anzeige ein und zeigt von nun an die neuesten Informationen. Sobald an eine Stelle navigiert wird, verharrt die Ansicht dort während die Simulation/das Mithören weiterläuft.
- ▲ Die **Suchen**-Funktion findet einen beliebigen Text im Message Storage, vor- und rückwärts.
- ▲ Über **Shortkeys** sind alle wichtigen Funktionen schnell erreichbar: Wechseln der Darstellungsmodi, Ändern der Nachrichtenfilter, Versenden einer



einzelnen Nachricht, Starten der Simulation oder des Mithörens, ...

- ▲ Bei der Simulation und beim Mithören erhält man eine **kontinuierliche Echtzeitdarstellung** mit einer **flexiblen Ausgabe** der **verschiedenen Kommunikationsschichten**. Die Ausgabe ist jederzeit - auch während der Simulation/des Mithörens - änderbar.
- ▲ Während der Simulation/des Mithörens werden in einem eigenen Fenster fortlaufend ermittelte **statistische Größen** ausgegeben: Zähler für gesendete/empfangene Zeichen, Puffer und Telegramme, aktueller, durchschn. und maximaler Durchsatz, fehlerhafte Zeichen und Telegramme (auch in Prozent relativ zur Gesamtzahl), wiederholte Telegramme und Verbindungsverluste, ...
- **Message Storage:**
 - ▲ Ist der zentrale Speicher für alle gesendeten/empfangenen Nachrichten. Jede Nachricht hat einen eigenen **Zeitstempel in Millisekundauflösung**.
 - ▲ Der Message Storage speichert **bis zu 10.000.000 Einträge**. Die Größe ist konfigurierbar, um den sicheren Betrieb des Rechners zu gewährleisten.
 - ▲ Zusätzlich **kann festgelegt** werden, dass reine **Verbindungsschichttelegramme nicht gespeichert** werden, um damit Ressourcen im Arbeitsspeicher und auf der Platte zu sparen.
 - ▲ Der Message Storage kann in einer **Message Log-Datei gespeichert** und von dieser auch wieder **eingelesen** werden.
- **Message List:**
 - ▲ Die Message List ist eine Liste von Meta-Nachrichten. Eine Meta-Nachricht kann ein **Verbindungsschichttelegramm** sein oder eine **Anwendungsschichtnachricht** (nur Nutzdaten). Desweiteren kann eine Meta-Nachricht eine **Verzögerung** bewirken, das **kontrollierte Runterfahren** einer Verbindung oder deren **abrupten**

Abbruch auslösen oder die **Ausgabe eines Anwender-texts** veranlassen.



- ▲ Die Anzahl der Message Lists und die Zahl der Nachrichten pro Liste ist nicht beschränkt. Die einzelnen Message Lists werden durch ihren Namen identifiziert.
- ▲ Nachrichten werden als hexadezimale Zeichenkette oder mit **interaktiver Unterstützung auf logischer Ebene als Klartext** eingegeben. Es können sowohl korrekte als auch beliebig falsche Nachrichten erzeugt werden.
- ▲ Eine Message List kann **ein einziges Mal** oder **periodisch ausgeführt** werden.
- ▲ Abfolgen von Message Lists für umfassende Testabläufe können definiert werden.
- ▲ **Aus dem Nachrichtenverkehr** können für jede Richtung getrennt **Message Lists erzeugt** werden, die z.B. alle Datenpunkte einer Station enthalten (nicht für alle Protokolle verfügbar).
- ▲ Message Lists können einzeln oder als Sammlung in einer Datei abgespeichert werden.
- ▲ Mit Hilfe üblicher Windows-Funktionen wie **Copy&Paste** können sehr einfach Message Lists oder einzelne Nachrichten aufgebaut und vervielfältigt werden.
- ▲ Auch während des Simulationsbetriebs können Message Lists und Nachrichten bearbeitet werden.
- ▲ Protokoll-spez. gibt es reservierte Namen für Message Lists für besondere Aufgaben: Nach dem Verbindungsaufbau von IEC 60870-5-101 wird z.B. die Message List „C_IC command“ gesucht und ausgeführt. Diese Message List kann z.B. die Generalabfrage enthalten.
- ▲ Message Lists können einzeln per Windows-Button ausgelöst oder gestoppt werden.
- **Kommunikation:**
 - ▲ Die Kommunikation erfolgt protokoll-abhängig über (alle vom Betriebssystem zur Verfügung gestellten) serielle Anschlüsse oder über das Netzwerk.
 - ▲ Über serielle Anschlüsse: Übertragungsgeschwindigkeiten von **50-115200 Baud** sind möglich¹. Alle Standard-Telegrammformate (**Bits/Zeichen, Stopbits, Paritätsbit**) sind realisiert. Zur **Simulation** wird **ein Anschluss** benötigt, für das **Mithö-**

ren typischerweise zwei (bei bestimmten Protokollen und unter besonderen Voraussetzungen reicht auch ein Anschluss aus). Es ist **keine weitere Hardware notwendig**. USB-Seriell-Konverter und PCMCIA-Karten werden unterstützt. Durchdachte und effiziente Ansteuerung der seriellen Anschlüsse gewährleistet, dass auch bei langsamen Baudraten kein ruckhaftes Verhalten auftritt und dass auch bei hohen Baudraten das Mithören zu einem sehr guten Ergebnis führt.

- ▲ Über Netzwerk: Die Simulation basiert auf dem **Standard-Betriebssystem-IP-Protokollstack**. Mithören ist über Windows-Treiber oder über WinPcap realisiert. Die vorhandenen Netzwerkkarten werden als Netzwerkanschluss verwendet.
- **Simulation:**
 - ▲ **CPTT simuliert** immer beides sowohl **Zentral als auch Unterstation** eines Fernwirkprotokolls (nicht bei IEC 611850).
 - ▲ Die Simulation basiert auf der implementierten **Protokollmaschine mit Prozeduren für Verbindungs- und Anwendungsschicht**. Die Protokollmaschine baut eine korrekte Verbindung auf, tauscht sicher Telegramme aus, usw.
 - ▲ Die Protokollmaschine umfasst auch Generalabfrage, Befehlsrückmeldungen und Zeitsynchronisation.
 - ▲ Man erhält eine **kontinuierliche Echtzeitdarstellung** von Nachrichten (bidirektional, mit Zeitstempel) und von Schnittstellen-/Protokollmaschinen-Ereignissen (wie Paritätsfehler, falsche Zeichen auf der Strecke, falsche Checksumme, unerwartetes Telegramm, falsches Telegrammfolgebit, ...).
 - ▲ Der **Nachrichtenverkehr** während der Simulation kann in einer **Message Log-Datei fortlaufend über Tage und Wochen mitgeschrieben** werden.
- **Mithören:**
 - ▲ Möglich bei **serieller Kommunikation** und bei **Kommunikation über Netzwerk** (TCP/IP oder UDP/IP, z.B. bei IEC 60870-5-104, IEC 61850, MODBUS TCP/IP oder DNP3 over LAN/WAN). Auch **GPRS/PPP-Kommunikation** hin zu einem **GSM-Modem** kann mitgehört und dargestellt werden.
 - ▲ Man erhält eine **kontinuierliche Echtzeitdarstellung** von Nachrichten (bidirektional, mit Zeitstempel) und von Schnittstellen-Ereignissen (wie Paritätsfehler, falsche Zeichen auf der Strecke, fal-

sche Checksumme, ...).

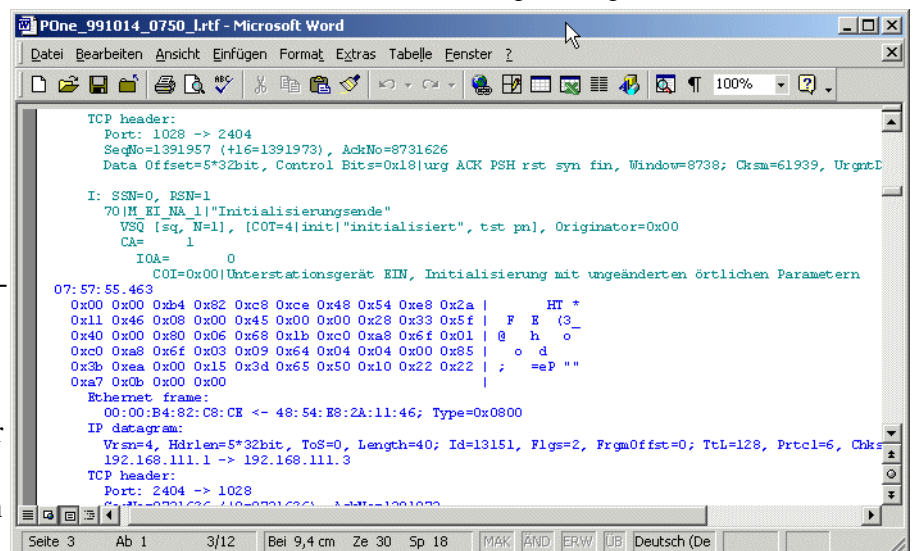
- ▲ Der **Nachrichtenverkehr** während des Mithörens kann in einer **Message Log-Datei fortlaufend über Tage und Wochen mitgeschrieben** werden.
- ▲ Das **Mithören von Paketen auf dem Netzwerk** ist systembedingt nur bei Anschluss über einen **Hub** oder einen **entsprechend konfigurierbaren Switch** auf dem Netzwerkpfad zwischen den Teilnehmern möglich.
- Nachträgliche (Off-line) Analyse:
 - ▲ Bei der nachträglichen Analyse wird in Message Log-Dateien gespeicherter Nachrichtenverkehr über den Nachrichten-Decoder des Protokolls im Message View dargestellt. Die Analyse erlaubt es, in aller Ruhe die richtigen Schlüsse zu ziehen und die notwendige Dokumentation zu erstellen.
 - ▲ Ergebnisse können einfach über Export aus **CPTT** heraus in die Dokumentation einfließen.
 - ▲ **Für die nachträgliche (Off-line) Analyse ist keine Lizenz erforderlich.**
- Protokollmaschine:
 - ▲ Die Protokollmaschine **implementiert ein Kommunikationsprotokoll** und legen damit Aktion/Reaktion fest. Sie sorgt für einen korrekten Verbindungsaufbau, für einen korrekten und vollständigen Austausch von Nachrichten, für Telegrammwiederholungen usw.
 - ▲ Neben den **Protokollprofil-Parametern** stehen noch weitere **implementierungs-spez. Parameter** bereit, um den Protokollentwickler und den Abwickler bei Tests zu unterstützen:

- **Stressszenarien** können entwickelt werden.
- **Fehlerfälle** sind **nachbildbar**.
- **Lastanalysen und Dauertests** sind durchführbar.
- Die Protokollmaschine kann **komplett deaktiviert** werden, um alle Aktionen per Hand auslösen zu können oder um ein Protokoll als User Engine Class zu realisieren.

- Message Log-Datei:
 - ▲ Der Inhalt des Message Storage oder der kontinuierliche Nachrichtenverkehr während der Simulation/beim Mithören kann in einer Message Log-Datei (mit)geschrieben werden.
 - ▲ **Enthält Rohdaten** und keine aufbereiteten Daten. Zur Darstellung werden die Rohdaten **entspre-**

chend den Darstellungsoptionen aufbereitet

- ▲ Jede Nachricht ist mit einem **Zeitstempel in Millisekundenauflösung** versehen. **Textmeldungen** beschreiben **Ereignisse der Schnittstelle**, zeigen **Protokollmaschinenereignisse an** und erläutern Abläufe in **CPTT**.
- ▲ Die **Beschreibung des Protokollprofils ist mitgespeichert** und wird beim Einlesen der Datei wiederhergestellt. Eine **gespeicherte Datei** ist nach dem Einlesen **sofort auf dem Bildschirm lesbar** (trotzdem bleibt das Profil anpassbar).
- ▲ Wird in der **Größe nur durch den freien Plattenplatz beschränkt**. Damit ist auch eine Aufzeichnungsdauer im Bereich von Tagen und Wochen kein Problem.
- ▲ Kann über eine in **CPTT** integrierte Funktion in **handliche, kleine Portionen aufgesplittet** werden.
- ▲ Das Speichern reiner **Verbindungsschichttelegramme kann unterdrückt** werden, um kleinere Dateien zu erhalten.
- Import/Export:
 - ▲ Import und Export öffnet **CPTT** hin zu anderen Programmen, hilft **weitergehende Analysen** durchzuführen und unterstützt den Anwender bei der **Dokumentation seiner Ergebnisse**.
 - ▲ **Importiert** werden können zeilenorientierte Hexdumps aus einer **Textdatei** oder netzwerk-basierte Kommunikation aus einer **Wireshark (Pcap)-Datei**.
 - ▲ Der Inhalt des Message Storage kann in eine



HTML- oder Textdatei, nach MS Word (RTF-Datei) oder in eine **Wireshark (Pcap)-Datei** exportiert werden.

- User Engine Classes:
 - ▲ Ziel der User Engine Classes ist es, weitergehende Funktionen in Form von **Plug-ins** zur Funktionalität von **CPTT** hinzuzufügen. Man hat die Möglichkeit, alle Arten von **benutzer-spez. Prozeduren** in **CPTT** zu integrieren:
 - Man kann **Überwachungsfunktionen kodieren**, um die Kommunikation zu verifizieren (z.B. die Integrität von Meldungs-änderungen).
 - Oder man implementiert eine **eigene Graphischer Benutzeroberfläche**, die **geräte-spez. Details** (z.B. Pseudodatenpunkte mit Systeminformationen) nachbildet.
 - Auch **Anwendungsschicht-Prozeduren** (z.B. einen **Filetransfer**) sind als User Engine Classes realisierbar.
 - ▲ Die Java-Umgebung sorgt dabei dafür, dass User Engine Classes **CPTT selbst nicht beeinträchtigen** (nicht zur Blockade oder zum Absturz bringen).
 - ▲ In der Dokumentation „Programmer’s Guide“ ist die Schnittstelle zwischen einer User Engine Class und **CPTT** beschrieben.
 - ▲ Verschiedene Beispiele für User Engine Classes werden mitgeliefert, darunter auch ein **IEC 60870-5-101/-104-Filetransfer**.
- Remote I/O Server:
 - ▲ Der Remote I/O Server **trennt die Darstellung des Nachrichtenverkehrs von dessen Empfang**. Er erlaubt es in **CPTT** Nachrichtenverkehr darzustellen, der nicht direkt am Darstellungsrechner sondern über (mehrere) entfernte Rechner empfangen wird. Zwischen Darstellungs- und Empfangsrechner muss nur eine Netzwerkverbindung bestehen.
 - ▲ Quelle für den Nachrichtenverkehr kann eine **serielle Schnittstelle, ein Netzwerkadapter oder eine Datei** sein.
 - ▲ Der Remote I/O Server ist ein eigenständiges Programm, verfügbar für verschiedene Rechnerarchitekturen, z.B. **MS Windows, PC Linux, Solaris, ...**. Für die Integration in kundeneigene Geräte liefern wir zusätzlich den Quellcode mit.
 - ▲ Typischer Einsatz ist die Fernüberwachung von Kommunikationssystemen oder die Integration des Remote I/O Servers in ein Gerät (RTU, IED, Gateway, ...) zur Diagnose des Nachrichtenverkehrs an dessen Schnittstelle.
- ▲ **Die Option „Remote I/O Client“ in CPTT ist getrennt zu lizenzieren.**
- Dokumentation:
 - ▲ **Multilinguale gedruckte Installationsanleitung** in Deutsch und Englisch.
 - ▲ **Schulungsdokument** in Deutsch und Englisch in elektronischer Form.
- Lizenzierung:
 - ▲ Eine **Lizenz ist notwendig für die Simulation von Zentral- oder Unterstation** eines fernwirktechn. Übertragungsprotokolls und für das **Mithören des Nachrichtenverkehrs** zwischen zwei Stationen.
 - ▲ **Für jedes Protokoll ist eine eigene Lizenz notwendig.**
 - ▲ **Für die nachträgliche (Off-line) Analyse ist keine Lizenz erforderlich.**
 - ▲ **CPTT darf auf jedem beliebigen Computersystem installiert und auch beliebig weitergegeben werden.** Wie schon gesagt: Nur für die Nutzung der Protokollsimulation und des Mithörens ist eine Lizenz erforderlich.
 - ▲ Die Lizenz/Lizenzen sind **auf einem Dongle gespeichert**:
 - Dongles sind für verschiedene Schnittstellen verfügbar: **Druckeranschluss oder USB.**
 - Die Dongles werden **für jeden Kunden individuell programmiert**. Die Programmierung ist **nachträglich vor Ort beim Anwender** durch eine **Updatedatei änderbar**. Dadurch können sehr schnell und einfach weitere Lizenzen auf einem Dongle hinzugefügt werden, ohne dass dieser eingeschickt werden muss.

¹⁾ Abhängig von der Leistungsfähigkeit des Computersystems.

²⁾ Nicht mit allen Netzwerkadapters ist das Mithören auf dem Netzwerk möglich. Die Vollständigkeit des mitgehörten Nachrichtenverkehrs hängt von der Leistungsfähigkeit des Computersystems ab.

Real Thoughts GmbH

Haid-und-Neu-Straße 7

76131 Karlsruhe

Germany

Vanity-No. +49-700-REALTHOU

Fon +49-721-6276730

Fax +49-721-6276731

Website www.realthoughts.de

E-Mail info@realthoughts.de

Irrtum und Änderungen vorbehalten. Stand: August 2011