CT - A P I 1.1

Anwendungsunabhängiges
Card**T**erminal **A**pplication **P**rogramming **I**nterface
für Chipkartenanwendungen

Autoren:
Deutsche Telekom AG / PZ Telesec
GMD - Forschungszentrum Informationstechnik GmbH
TÜV Informationstechnik GmbH
TELETRUST Deutschland e.V.
Stand: 14.10.1998

Inhalt

Vorwort 3
1. Zweck 3
2. Normative Verweisungen 3
3. Abkürzungen 3
4. CT-API-Schnittstelle 3
4.1 CT-API Aufbau 3
4.2 Typen-Definition 4
4.3 CT-API-Funktionen 4
4.3.1 CT_init 4
4.3.2 CT_data 5
4.3.3 CT_close 6
4.4 Allgemeine Funktionswerte der CT-API-Funktionen 7
Anhang A (normativ) 8
Kennzeichnung der CT-API-Funktionen mit einem "well known identifier" 8
Kennzeichnung der CT-API-Dateinamen mit einem wki 8
Anhang B (informativ)
Informationen für Programmierer 9
Windows 16 Bit DLL 9
Anhang C (informativ)
Programmierbeispiel 10
Anhang D (informativ)
Beispielablauf der CT-API-Funktion CT_data 11

Autoren (Ansprechpartner) :

Deutsche Telekom AG/ PZ Telesec
Untere Industriestr. 20
57250 Netphen

B. Kowalski, R. Moos
Tel.: (0271) 708-1600, Fax: -1625
E-Mail: Rainer.Moos@telekom.de
http://www.telesec.de

GMD
Rheinstrasse 75
D 64295 Darmstadt

L. Eckstein, B. Struif
Tel.: (06151) 869-205, Fax: -224
E-Mail: Levona.Eckstein@darmstadt.gmd.de
http://www.darmstadt.gmd.de

TÜV Informationstechnik GmbH
Leimbachstr. 227
D 57074 Siegen

J. Atrott, Dr. R. Baumgart
Tel.: (0271) 2278-192, Fax: -197
E-Mail: J.Atrott@tuvit.cubis.de
http://www.tuvit.de

TELETRUST Deutschland e.V. Geschäftsstelle
Eichendorffstrasse 16
D 99096 Erfurt

Prof. Dr.H. Reimer
Tel.: (0361) 3460531, Fax.: 3453957
E-Mail: teletrust@t-online.de
http://www.teletrust.de

Adressen:

RID German National Registration Authority
c/o GMD
Rheinstr. 75
64295 Darmstadt

B. Struif
Tel.: (06151) 869-206, Fax:-224
E-Mail: Bruno.Struif@darmstadt.gmd.de

Vorwort

Die Version CT-API 1.1 ist funktionell kompatibel zum CT-API vom 29.10.1993.
Folgende Änderungen wurden durchgeführt:

- *Ergänzung / Änderung von dad / sad-Adressen*
- *Einfügung einer Typen-Definition*
- *Änderung der ctn und pn Parameter auf 16 bit Format*
- *Änderung der lenr und response Parameter bei CT_data*
- *Erweiterung der Error-Returncodes*
- *Hinzufügung eines "well known identifiers" zur gleichzeitigen Nutzung verschiedener CT-API-Libraries und verschiedener CT-API-Dateien diverser Hersteller*
- *Überarbeitung der Texte und der Darstellungen*
- *Erweiterung des Autorenteams*

1. Zweck

Diese Spezifikation beschreibt die anwendungsunabhängigen Kartenterminal (Card-Terminal)-Funktionen, die benötigt werden, um auf einfache Weise das Handling und die Kommunikation mit Chipkarten realisieren zu können.
Die CT-API-Funktionen sind so konstruiert, daß

- Speicher- und Prozessor-Chipkarten verwendet,
- die zur Steuerung des Kartenterminals nötigen Kommandos übergeben werden und
- Kartenterminals unabhängig vom Anschluß (Port) angesprochen werden können.

An der CT-API-Schnittstelle ist nicht sichtbar, ob das CardTerminal als Einbauversion oder als externes Endgerät vorliegt.

Die CT-API Software sollte vom CT-Hersteller für die Systemumgebungen bereitgestellt werden, die für den Einsatz des betreffenden CT relevant sind.

2. Normative Verweisungen

ISO 3166-1: 1997 (5th edition)
Codes for the representation of names of countries and their subdivisions – Part 1

ISO/IEC 7816-3: 1997 (2nd edition)
Identifikation cards - Integrated circuit(s) cards with contacts
Part 3 - Electronic signals and transmisson protocols

ISO/IEC 7816-4: 1995
Identification cards - Integrated circuit(s) cards with contacts
Part 4 - Inter-industry commands for interchange

3. Abkürzungen

API	Application Programming Interface
CT	Card Terminal
ctn	Card Terminal Number
CTM	Card Terminal Manufacturer
dad	Destination ADdress
ICC	Integrated Circuit(s) Card
HTSI	Host Transport Service Interface
lenc	LENgth Command
lenr	LENgth Response
pn	Port Number
res	RESult of function
RFU	Reserved for Future Use
RID	Registered application provider IDdentifier
sad	Source ADdress
wki	Well Known Identifier

4. CT-API-Schnittstelle

4.1 CT-API Aufbau

Die CT-API-Funktionen werden von einem sogenannten HTSI-Modul (Host-Transport-Service-Interface) erbracht. Die Einordnung eines HTSI-Moduls in seine Systemumgebung zeigt Abb. 1.

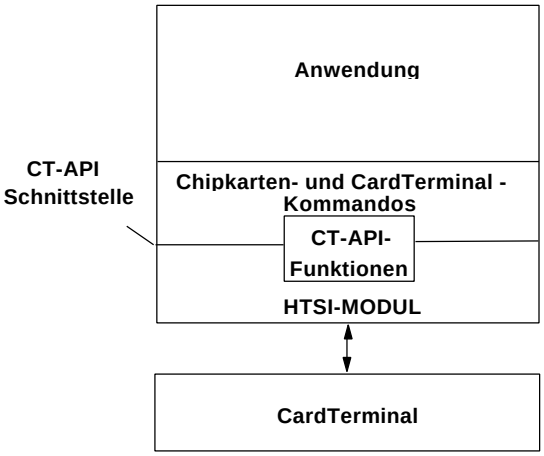

Abb. 1 Die Einordnung eines HTSI-Moduls in seine Systemumgebung

4.2 Typen-Definition

Das CT-API verwendet Funktionen, deren Parameter und Funktionsergebnisse in folgender Notation beschrieben werden.

Ordinaler Typ	Bedeutung
IU_X	Vorzeichenlose Ganzzahl (integer unsigned); Index x zeigt die verwendete Bitlänge in dezimaler Form an (z.B.: IU_{16} = 0 ... 65535)
IS_X	Vorzeichenbehaftete Ganzzahl (integer signed); Index x zeigt die verwendete Bitlänge in dezimaler Form an (z.B.: IS_8 = -128 ... 127)

Tab.1:Typen-Notation

4.3 CT-API-Funktionen

Das CT -API umfaßt folgende Funktionen, die von einem HTSI-Modul bereitgestellt werden.

CT-API-Funktion	Bedeutung
CT_init	Initiieren der Host/CT-Verbindung
CT_data	Senden eines Kommandos an ein CardTerminal bzw. an eine Chipkarte und Rückgabe der Antwort
CT_close	Beenden der Host/CT-Verbindung

Tab.2: CT-API-Funktionen

Kann eine Funktion ordnungsgemäß ausgeführt werden, wird als Funktionswert 0 zurückgeliefert, ansonsten zeigt ein negativer Funktionswert die Fehlerursache an. Diese Funktionswerte sind für alle CT-API-Funktionen im Abschnitt 4.4 aufgeführt.

Normative und informative Einzelheiten zur Nutzung der CT-API-Funktionen sind im Anhang B bzw. Anhang C beschrieben.

Die CT-API-Funktionen sind aus Kompatibilitätsgründen in jeder Implementation gleichnamig zu benennen und mit gleichen Parametern zu versehen.
Zusätzlich ist für den Fall der gleichzeitigen Verwendung mehrerer CT-API-Libraries in einem Anwendungssystem eine CT-API-Library mit "well known identifier" (siehe Anhang A) bereitzustellen.

4.3.1 CT_init

Mit der Funktion CT_init wird die zur Kommunikation benötigte Host-Schnittstelle (bei MS-DOS z.B.: COM1: oder COM2:, bei UNIX z.B. ttya oder ttyb) ausgewählt, an welcher das CardTerminal angeschlossen ist, wobei automatisch die Defaultwerte für die Kommunikation eingestellt werden. Die CT_init-Funktion muß vor Beginn der Kommunikation mit dem CardTerminal und der Chipkarte aufgerufen werden. Gleichzeitig wird dabei dem betreffenden CardTerminal eine eindeutige logische Adresse in Form einer, vom Programmierer frei wählbaren, CardTerminal-Number (ctn) zugeordnet. Zusätzlich wird eine Port-Number (pn) verwendet. Sie bezeichnet die physikalische Schnittstelle, über die das betreffende CardTerminal angesteuert wird.
Die Zuordnung bzw. das Zuordnungsprinzip von Port-Number zu physikalischer Schnittstelle ist nicht verbindlich vorgegeben und ist der Dokumentation des Herstellers zu entnehmen.
Ein CardTerminal-Hersteller kann z.B. einer Tastaturschnittstelle die Port-Number 1 zuweisen, ein anderer Hersteller aber die Port-Number 1 für den Ausgang 1 eines 8-Bit breiten I/O-Ports oder für die serielle Schnittstelle COM1 eines Personal Computers verwenden.
Das Anwendungsprogramm kann diese unterschiedlichen Port-Ausprägungen trotzdem jeweils mit CardTerminal Number 1 und Port-Number 1 ansprechen, falls die entsprechende API-Library des Herstellers dazugebunden wird. Die entsprechenden

Module im Anwendungsprogramm brauchen somit nicht modifiziert zu werden.

Funktion:

CT_init (ctn, pn)

Funktionsparameter:

Parame-ternname	Parametertyp	Bedeutung
ctn	Eingabepara-meter, Typ IU_{16}	Logische Card-Terminal Number
pn	Eingabepara-meter, Typ IU_{16}	Port Number der physikalischen Schnittstelle

Tab 3: Parameter zu CT_init

Funktionswerte:

Parametertyp	Bedeutung
Rückgabepa-rameter, Typ IS_8	Funktionswerte s. Abschnitt 4.4

Tab 4: Funktionswerte zu CT_init

4.3.2 CT_data

Die Funktion CT_data dient dem Senden von Chipkarten- bzw. CardTerminal-Kommandos und liefert die Antwort auf das Kommando an das aufrufende Programm zurück.

Funktion:

CT_data (ctn, dad, sad, lenc, command, lenr, response)

Funktionsparameter:

Parame-ternname	Parametertyp	Bedeutung
ctn	Eingabepara-meter, Typ IU_{16}	Logische Card Termi-nal Number
dad	Ein/Rückgabepa-rameter, Typ IU_8	Destination Address
sad	Ein/Rückgabepa-rameter, Typ IU_8	Source Address
lenc	Eingabeparame-ter, Typ IU_{16}	Länge des Kommandos (Command) in Byte
command	Eingabepara-meter: Referenz-adresse eines Feldes mit Ele-menten vom Typ IU_8, die das Kommando ent-halten	Chipkarten-bzw. CT-Kommando
lenr	Ein/Rückgabepa-rameter, Typ IU_{16}	Übermittlung der max. Puffergröße des response Feldes an die Funktion und Rück-gabe der tatsächlichen Länge der Antwort in Byte
response	Eingabeparame-ter: Referenza-dresse eines Feldes mit Ele-menten vom Typ IU_8, in die die Antwort einge-tragen wird (Felddeklaration i.d. Applikation)	Antwort auf das Kom-mando

Tab. 5: Parameter zu CT_data

Bei Aufruf der Funktion CT_data steht in der Regel bei Source-Address der Adresswert für ”Host”. Bei bestimmten Anwendungen kann jedoch hier der Adresswert für ”Remote Host” auftreten. Als Destination Address tritt bei CT-Kommandos der Adresswert für das CardTerminal auf und bei ICC-Kommandos ein Adresswert für die Chipkarte. Da ein Kartenterminal mehrere Chipkarten-Schnittstellen besitzen kann, ist in der Destination Address der Adresswert für die betreffende Chipkarte anzugeben. Bei Rückgabe der Antwort durch die aufgerufene Funktion werden auch die Werte in den Parametern dad und sad, entsprechend von Empfänger und Sender, gesetzt.
Tab. 6 und 7 zeigen die bisher festgelegten Adresswerte.

sad in sedezimaler Notation (Hex)	Sender
02	HOST
05	REMOTE HOST

dad in sedezimaler Notation (Hex)	Empfänger

00	ICC1 (Chipkarte 1)
01	CT
02	ICC2 (Chipkarte 2)
...	...
0E	ICC14 (Chipkarte 14)
XX	other values reserved

Tab. 6: sad - und dad Werte beim Senden von command

sad in sedezimaler Notation (Hex)	Sender
00	ICC1 (Chipkarte 1)
01	CT
02	ICC2 (Chipkarte 2)
...	...
0E	ICC14 (Chipkarte 14)
dad in sedezimaler Notation (Hex)	**Empfänger**
02	HOST
05	REMOTE HOST
XX	other values reserved

Tab. 7: sad - und dad Werte beim Empfangen von response

Funktionswerte:

Parametertyp	Bedeutung
Rückgabeparameter, Typ IS_8	Funktionswerte s. Abschnitt 4.4

Tab 8: Funktionswerte zu CT_data

Nutzungshinweise:

a) CardTerminal-Kommandos

Die CT_data-Funktion erlaubt sowohl die Benutzung standardisierter bzw. herstellerneutraler CT-Kommandos (inter-industry-commands) als auch die Verwendung herstellerspezifischer CT-Kommandos.

b) Kommandos an Speicherchipkarten

Unterstützt das Kartenterminal die Kommunikation mit Speicherchipkarten, dann ist ein Kommando entsprechend den dafür vorgesehenen Konventionen zu übergeben (z.B. als inter-industry command, falls eine entsprechende Umsetzungsfunktion auf Chip-spezifische Kommandos vorhanden ist).

c) Kommandos an Prozessorchipkarten

Kommandos an Prozessorchipkarten werden grundsätzlich transparent (d.h. ohne jegliche Änderung) zur Chipkarte durchgereicht. Implementationsabhängiger Overhead (z.B. Sicherungsprotokoll) wird automatisch hinzugefügt.

Abb. 1 im Anhang D zeigt am Beispiel eines CardTerminals die Ausführung eines Chipkartenkommandos für Prozessorchipkarten mit Hilfe der CT-API-Funktion CT_data.

4.3.3 CT_close

Die Funktion CT_close bildet das Äquivalent zur Funktion CT_init. Sie beendet die Kommunikation zum jeweiligen CardTerminal, welches mit CT_init einer logischen CardTerminal Number zugewiesen wurde. Die Funktion muß vor Ende des Programms aufgerufen werden, um ggf. belegte Resourcen freizugeben.

Funktion:

CT_close (ctn)

Funktionsparameter:

Parametername	Parametertyp	Bedeutung
ctn	Eingabeparameter, Typ IU_{16}	Logische CardTerminal Number

Tab. 9: Parameter zu CT_close

Funktionswerte:

Parametertyp	Bedeutung
Rückgabeparameter, Typ IS_8	Funktionswerte s. Abschnitt 4.4

Tab 10: Funktionswerte zu CT_close

4.4 Allgemeine Funktionswerte der CT-API-Funktionen

Die CT-API-Funktionen geben die in Tab. 11 aufgeführten Funktionswerte zurück. Funktionswerte sind generell vom Typ IS_8.
Returncodes sollen, sofern es die verwendete Programmiersprache unterstützt, in Form von globalen, typisierten Konstanten

("OK", "ERR_INVALID", u.s.w.) implementiert werden.

Returncode	Funktionswert	Bedeutung
OK	0	Funktionsaufruf war erfolgreich
ERR_INVALID	-1	ungültiger Parameter oder Wert
ERR_CT	-8	CT Fehler 1)
ERR_TRANS	-10	nicht behebbarer Übertragungsfehler 2)
ERR_MEMORY	-11	Speicherzuordnungsfehler im HTSI 3)
ERR_HOST	-127	Funktionsabbruch durch Host / Betriebssystem
ERR_HTSI	-128	HTSI-Fehler

Tab. 11: Funktionswerte der CT - API-Funktionen

1) Auf das CT kann temporär nicht zugegriffen werden (Bearbeitung anderer oder interner Prozesse). Problem kann anwendungstechnisch gelöst werden.

2) Übertragungsfehler im Sinne von mechanischen, elektrischen und protokolltechnischen Ausfällen; Neuinitialisierung des CT erforderlich.

3) Hier wird signalisiert, daß ein Speicherzuordnungsfehler aufgetreten ist (z.B. wenn die Datenmenge größer ist als der zur Verfügung gestellte Puffer).

Folgereaktionen auf Fehlerfälle sind in Abstimmung mit den Empfehlungen der HTSI-Hersteller zu implementieren.

Anhang A (normativ)

Kennzeichnung der CT-API-Funktionen mit einem "well known identifier"

In einer Entwicklungsumgebung, in der die Funktionen der verschiedenen CT-API-Libraries genau identifizierbar sein müssen, sind die Funktionsnamen um einen ”well known identifier (wki)” zu erweitern. Der wki besteht aus
- der CardTerminal Manufacturer Id (CTM Id, 5 Byte) und
- der HTSI Id (2 Byte).

Die CT-API-Funktionen haben somit die Form

CTxxxxxyy_init
CTxxxxxyy_data
CTxxxxxyy_close

wobei xxxxx die CTM Id und yy die HTSI Id darstellen. Die CTM Id wird in Abstimmung mit dem CT-Hersteller von der RID German National Registration Authority vergeben und registriert. Sie besteht aus einem 2-Byte langen country code in alpha-Codierung entsprechend ISO 3166 (z.B. DE für Deutschland) und einer 3-Byte langen alphanumerischen Herstellerkennung. Die HTSI Id wird vom CT-Hersteller vergeben.
Abb. 1 zeigt am Beispiel eines Anwendungssystems mit zwei CT-API-Libraries und zwei Kartenterminals das generelle Konstruktionsprinzip.

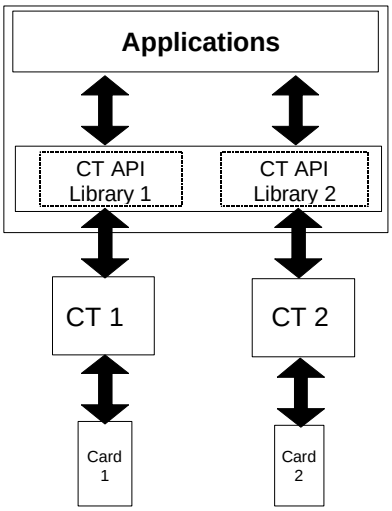

Abb. 1: Anwendungsystem mit zwei unterschiedlichen CT-API-Libraries und zwei unterschiedlichen Kartenterminals.

Kennzeichnung der CT-API-Dateinamen mit einem wki

Falls es erforderlich ist, firmenspezifische Software signifikant unterscheiden zu können (Dynamic Link Libraries, Systemtreiber u.ä.), ist es notwendig, Dateinamen ebenfalls mit einem wki auszustatten. Der wki besteht in diesem Fall aus
- der CardTerminal Manufacturer Id (CTM Id, 3 Byte) und
- dem individuellen Teil (0-3 Byte).

Die CT-API-Dateinamen haben somit die Form

ctxxx[yyy]

wobei xxx die CTM Id und yyy den individuellen Teil darstellen.
Die CTM Id wird in Abstimmung mit dem CT-Hersteller ebenfalls von der RID German National Registration Authority vergeben und registriert. Sie besteht aus einer 3-Byte langen alphanumerischen Herstellerkennung. Der individuelle Teil ist optional und kann vom CT-Hersteller frei gewählt werden.

Anhang B (informativ)

Informationen für Programmierer

Windows 16 Bit DLL

Die Funktionen müssen in Pascal-Konvention und mit ”far”-Aufruf deklariert werden.
Ebenfalls müssen Parameter zur Übergabe von Zeigern als ”far” deklariert werden.

Anhang C (informativ)

Programmierbeispiel

Das folgende Beispiel zeigt den Gebrauch der CT-API-Funktionen unter der Verwendung von ANSI C

```
#include <stdio.h>
#include <conio.h>
#include "ct_api.h"

#define   MAXMEM   1000

int main(void)
{
 unsigned char sad, dad;                  /* source/destination address,*/
                                          /* als Byte deklariert  */
 unsigned char command[300];              /* Feld für Kommandotext mit max. 300 Zeichen Länge */
 unsigned char response [MAXMEM];         /* Feld für Antwort der Funktion hier: max 1000 Zeichen */
 unsigned short int ctn, lenr;            /* card terminal number, Länge der Antwort */
 char res;                                /* Funktions(rückgabe)wert */

        ctn=1;                           /*Card Terminal 1*/

/* logische Terminal-Nummer 1 und Port COM2 auswählen (herstellerabhängig!)*/
 if (CT_init(ctn,2) != OK) return(1);              /* falls Returncode nicht OK, PRG beenden */

/* CT-Kommando REQUEST ICC (20 12 01 00 00 ) aufbauen und senden */
 printf  ("\nBitte Karte einlegen und eine Taste drücken!\n");
 getch ();

        sad=2;                           /* source = Host */
        dad=1;                           /* destination = Card Terminal */
        lenr=MAXMEM;                     /* maximale Länge der Antwort als Info an die Funktion */
        command[0]=0x20;                 /* CLA */
        command[1]=0x12;                 /* INS */
        command[2]=0x01;                 /* P1 */
        command[3]=0x00;                 /* P2 */
        command[4]=0x00;                 /* L */

/* Funktion (CT_data) aufrufen */

 res=CT_data (ctn,&dad,&sad,5,command,&lenr,response);

/* Fehlerbehandlung hier nicht beschrieben */
                        ...
/* nach Auswerfen der Karte mit CT-Kommando EJECT ICC */
 res=CT_close(ctn);

return(0);
}
```

Anhang D (informativ)

Beispielablauf der CT-API-Funktion CT_data

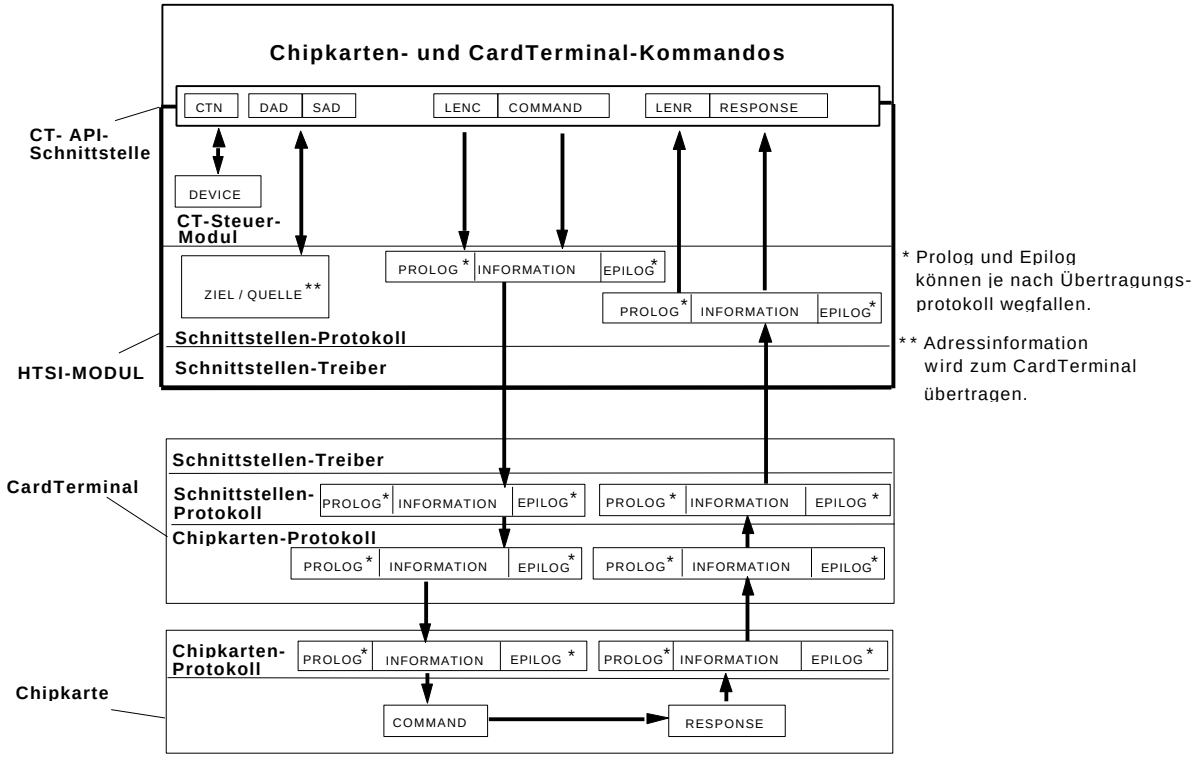

Abb. 1: Beispiel für den Ablauf der CT-API-Funktion CT_data.
Andere Implementationen können von dieser Grafik abweichen.