CMX V6.0 (UNIX)

Betrieb und Administration

Ausgabe Juni 2003

Kritik… Anregungen… Korrekturen…

Die Redaktion ist interessiert an Ihren Kommentaren zu diesem Handbuch. Ihre Rückmeldungen helfen uns, die Dokumentation zu optimieren und auf Ihre Wünsche und Bedürfnisse abzustimmen.

Für Ihre Kommentare stehen Ihnen Fax-Formulare auf den letzten Seiten dieses Handbuchs zur Verfügung.

Dort finden Sie auch die Adressen der zuständigen Redaktion.

Zertifizierte Dokumentation nach DIN EN ISO 9001:2000

Um eine gleichbleibend hohe Qualität und Anwenderfreundlichkeit zu gewährleisten, wurde diese Dokumentation nach den Vorgaben eines Qualitätsmanagementsystems erstellt, welches die Forderungen der DIN EN ISO 9001:2000 erfüllt.

cognitas. Gesellschaft für Technik-Dokumentation mbH
www.cognitas.de

Copyright und Handelsmarken

Dieses Handbuch wurde
auf chlorfrei gebleichtem
Papier gedruckt.

Inhalt

1 Einleitung

Das Transportzugriffssystem CMX (Communications Manager UNIX) ist das Basisprodukt der Kommunikationssoftware. CMX ermöglicht die Kommunikation zwischen Anwendungen in verschiedenen Rechnersystemen: Mit CMX lassen sich Anwendungsprogramme erstellen, die unabhängig vom Transportsystem mit anderen Anwendungen kommunizieren können.

Das Produkt CMX (UNIX) unterscheidet sich vom Transportzugriffssystem CMX (Solaris) in folgenden Punkten:

– Beim NAMEN Service gibt es nur eine Kommandozeilen-Schnittstelle.

– Transportzugriffsmethoden sind nur über Sockets, X/Open Transport Interface (XTI) und Transport Layer Interface (TLI) möglich.

An der Programmierschnittstelle unterscheidet sich CMX (UNIX) nicht von CMX (Solaris).

1.1 Zielgruppe des Handbuchs

Dieses Handbuch richtet sich an Systemverwalter. Um mit CMX arbeiten zu können, benötigen Sie Kenntnisse des Betriebssystems UNIX. Ferner sind für das Verständnis Kenntnisse über Prinzipien und Methoden der Datenfernverarbeitung hilfreich, insbesondere das OSI-Referenzmodell, wie es in ISO 7498 normiert ist.

1.2 Konzept des Handbuchs

Die Beschreibung des gesamten Produkts CMX umfasst zwei Handbücher:

– Das vorliegende CMX-Benutzerhandbuch für Systemverwalter

– CMX-Benutzerhandbuch „Anwendungen programmieren“ für den Programmierer von TS-Anwendungen (TS = Transport Service), die die CMX-Programmschnittstellen benutzen

Das Handbuch beschreibt in Kapitel 2 die Architektur der CMX (UNIX)-Kommunikationssoftware. Sie erhalten dort einen Überblick über die erhältlichen CMX (UNIX)-Produkte und ihre Funktionalität sowie wichtige Basisinformation zur CMX-Systemverwaltung.

Kapitel 3 enthält grundlegende Informationen zur Adressierung von Transportsystem-Anwendungen im Transport Name Service (TNS). Sie benötigen diese Kenntnisse, um Ihr Rechnernetz konfigurieren zu können.

Kapitel 4 liefert wichtige Hinweise zur Installation und Deinstallation von CMX (UNIX).

Kapitel 5 beschreibt Eingabe- und Dateiformat der CMX-Komponenten TNS.

Kapitel 6 beschreibt die Transport Provider Selection sowie deren Einsatz, Verwaltung und Pflege.

Kapitel 7 enthält Adress- und Kommando-Informationen zum Transportdienst RFC1006 über TCP/IP, der als Bestandteil von CMX ausgeliefert wird.

Kapitel 8 beschreibt die für die CMX-Systemverwaltung relevanten Kommandos mit ihrer Syntax.

Das CMX-Benutzerhandbuch „Anwendungen programmieren“ beschreibt die Programmschnittstellen von CMX, d. h. alle Programmaufrufe, die Sie benötigen, um selbst Anwendungen zu entwickeln.

Verweise auf andere Handbücher

Im Text wird mit „siehe Handbuch ’Handbuchtitel’ [*n*]" auf andere Handbücher verwiesen, die weiterführende Informationen enthalten. Dabei ist *n* eine Ziffer. Unter [*n*] finden Sie die Titel der entsprechenden Handbücher im Literaturverzeichnis zusammen mit einer kurzen Inhaltsangabe.

1.3 Readme-Dateien

Funktionelle Änderungen und Nachträge der aktuellen Produktversion zu diesem Handbuch entnehmen Sie bitte ggf. den produktspezifischen Readme-Dateien. Diese finden Sie im plattformspezifischen Verzeichnis. In der entsprechenden Readme-Datei sind auch die betriebssytemspezifischen Besonderheiten dokumentiert.

1.4 Darstellungsmittel

Soweit möglich hält sich die Kommandobeschreibung an ein festes Raster:

– Beschreibung des Kommandos
– Syntax
– Syntaxbeschreibung
– Ausgabeformat
– Ende-Status
– Fehlermeldungen
– Dateien
– Beispiel
– Siehe auch

Die hier aufgeführten Bestandteile werden im Folgenden erläutert.

Beschreibung des Kommandos

Im ersten Abschnitt jeder Kommandobeschreibung ist Folgendes dargestellt:

– die Arbeitsweise des Kommandos
– die unterschiedlichen Aufgaben der verschiedenen Kommando-Formate, falls mehrere Formate vorhanden sind
– in welcher Umgebung das Kommando zu verwenden ist (z. B. Einträge in Dateien, Zugriffsrechte)
– Hintergrundinformationen

Syntax

cmd[␣**-a**][␣**-b**][␣**-c**][␣**-d**␣arg1][␣**-f**␣arg2]␣datei␣...

Sie müssen *cmd* eingeben sowie für *datei* eine oder mehrere Dateien, die jeweils durch ein Leerzeichen voneinander getrennt werden. Sie können zusätzlich angeben:

– eine oder mehrere Optionen **-a, -b, -c**. Diese Optionen können Sie einzeln (**-a**␣**-b**␣**-c**) oder zusammen (**-abc**) angeben.

– die Option *-d*, wobei *arg1* durch ein Argument ersetzt werden muss

– die Option *-f*, wobei *arg2* durch ein Argument ersetzt werden muss

Halbfette Zeichen
Konstanten. Halbfett gedruckte Zeichen müssen genau wie dargestellt eingegeben werden.

Normale Zeichen
: Variablen. Diese Zeichen sind Stellvertreter für andere Zeichen, die Sie auswählen und eingeben.

[ ] Optionen. Argumente in eckigen Klammern sind optional und müssen nicht angegeben werden. Die eckigen Klammern sind nicht einzugeben, es sei denn, es wird ausdrücklich darauf hingewiesen.

␣ Leerzeichen, das Sie eingeben müssen.

.. Der vorherige Ausdruck kann wiederholt werden. Falls zwischen den Wiederholungen Leerzeichen eingegeben werden müssen, die nicht im Ausdruck enthalten sind, steht vor ... ein ␣(Leerzeichen).

{ | } Auswahlmöglichkeit. Wählen Sie genau einen der Ausdrücke aus, die durch den Strich getrennt sind.

<u>unterstrichen</u>
: Voreinstellung

Ist es bei einem Kommando möglich, für eine Option mehrere alternative Angaben zu machen, so wird die Kommandosyntax zweimal aufgeführt. Einmal wird für die entsprechende Option ein Stellungsparameter angegeben. In der zweiten Darstellung stehen statt des Stellungsparameters alle möglichen Angaben für die Option.

Die zweite Darstellung soll dem Profi ein schnelles Nachschlagen ermöglichen. Die Rasterung ist zwischen den beiden Darstellungen unterbrochen. Im ersten Raster steht jeweils die Darstellungsform mit Stellungsparametern.

Syntaxbeschreibung

Hier finden Sie die Beschreibung von Optionen und Argumenten (Eingabedateien, Parameter, Variablen etc.), die Sie beim Aufruf eines Kommandos eingeben können. Im Fließtext wird nicht zwischen Konstanten und Variablen unterschieden. Alle Syntaxelemente sowie Dateinamen, Pfadnamen und Kommandos sind dort in *kursiver* Schrift dargestellt.

Ausgabeformat

In diesem Abschnitt werden die Ausgabeformate der Kommandos erläutert.

Ende-Status

Ein Ende-Status ist der Wert, den ein Kommando nach seiner Ausführung an den aufrufenden Prozess zurückliefert. Der Wert gibt Auskunft darüber, wie das Kommando abgelaufen ist. Der Ende-Status ist ein Zahlenwert und wird in der Variablen ? abgelegt. Sie fragen den Ende-Status mit dem Befehl *echo $?* ab.

Der Ende-Status wird nur dann beschrieben, wenn er von folgendem Regelfall abweicht:

0 nach korrekter Durchführung des Kommandos

≠0 bei Fehler

Fehler

Hier werden wichtige Fehlermeldungen angegeben und erläutert sowie Hinweise zur Fehlervermeidung und -behebung gegeben.

Fehlermeldungen werden generell auf die Standard-Fehlerausgabe *stderr* ausgegeben. Normalerweise ist der Bildschirm die Standard-Fehlerausgabe.

Dateien

Hier werden Dateien angegeben, auf die das betreffende Kommando zugreift oder die von dem Kommando erzeugt werden.

Beispiel

Beispiele sollen die Hauptfunktion des Kommandos, den Einsatz wesentlicher Optionen sowie sinnvolle Kombinationen von Optionen und Argumenten veranschaulichen. In Anwendungsbeispielen sind Eingaben in das System dicktengleich halbfett dargestellt. Alle diese Eingabezeilen werden mit der ↵-Taste abgeschlossen. Die Taste wird daher am Ende der Zeilen nicht angegeben.

Ausgaben des Systems werden, außer im Fließtext, `dicktengleich` dargestellt. Im Fließtext erscheinen die Ausgaben *kursiv*.

Siehe auch

Hier finden Sie Verweise auf andere Kommandos, die eine ähnliche Funktionsweise haben oder mit dem betreffenden Kommando zusammenarbeiten. Außerdem wird auf weitere Literatur zu diesem Kommando verwiesen.

Hinweise und Warnungen

Dieses Symbol weist auf besonders wichtige Informationen hin, die Sie unbedingt beachten sollten.

Vorsicht!

Dieses Symbol weist auf Gefahren hin, die zu Datenverlust oder Geräteschaden führen können.

2 Architektur der CMX (UNIX)-Kommunikation

In diesem Kapitel erhalten Sie einen Überblick über die CMX (UNIX)-Kommunikationsarchitektur. Sie lernen die wesentlichen Eigenschaften dieser Architektur kennen und erfahren, welche Rolle CMX in der UNIX-Kommunikation spielt, und welche Dienste CMX Ihnen anbietet. Alle zentralen Begriffe, die in den nachfolgenden Kapiteln verwendet sind, werden hier erklärt.

Anhand der UNIX-Kommunikationsarchitektur wird die Vielzahl der Vernetzungsmöglichkeiten aufgezeigt, und die von CMX angebotenen Dienste werden aufgezählt. Sie erfahren so im Überblick, welche Leistungen von den Produkten erbracht werden.

Da für den Einsatz von CMX ein allgemeines Verständnis der Architektur der UNIX-Kommunikation erforderlich ist, beschreiben die folgenden Abschnitte, wie diese Leistungen erbracht werden. Anhand der Architektur wird die Terminologie eingeführt, wie sie in den Handbüchern zur UNIX-Kommunikation verwendet ist. Voraussetzung zum Verständnis sind lediglich Grundkenntnisse der Datenkommunikation sowie der Basis-Struktur des UNIX-Betriebssystems.

2.1 Leistungsumfang von CMX (UNIX)

Die UNIX-Kommunikationsarchitektur bietet für alle wichtigen Datennetze die entsprechenden Transportprofile. Zum Erstellen von Anwendungsprogrammen für die Kommunikation werden mehrere Programmschnittstellen angeboten.

Um zwischen den verschiedenartigen Transportprofilen und Programmschnittstellen zu vermitteln, präsentiert CMX den TS-Anwendungen ein einheitliches Bild des Transportsystems. Dadurch haben Sie den Vorteil, TS-Anwendungen unabhängig vom Transportsystem entwickeln zu können. Auf welche Netzwerkprofile die Anwendungen aufsetzen, wird erst zum Ablaufzeitpunkt durch die Konfigurierung entschieden.

CMX benötigt entsprechende Systemgrenzwerte für uneingeschränkte Kommunikation. Beachten Sie hierzu die Hinweise in der jeweiligen Readme-Datei.

2.2 Netzzugänge und Transportprofile

CMX unterstützt den Anschluss von UNIX-Systemen an ausgewählte Netze. Das bedeutet Integration in die wichtigsten Kommunikations-Architekturen TCP/IP, OSI und SNA und alle gängigen physikalischen Netze. Die folgende Abbildung zeigt diese Anschlussmöglichkeiten.

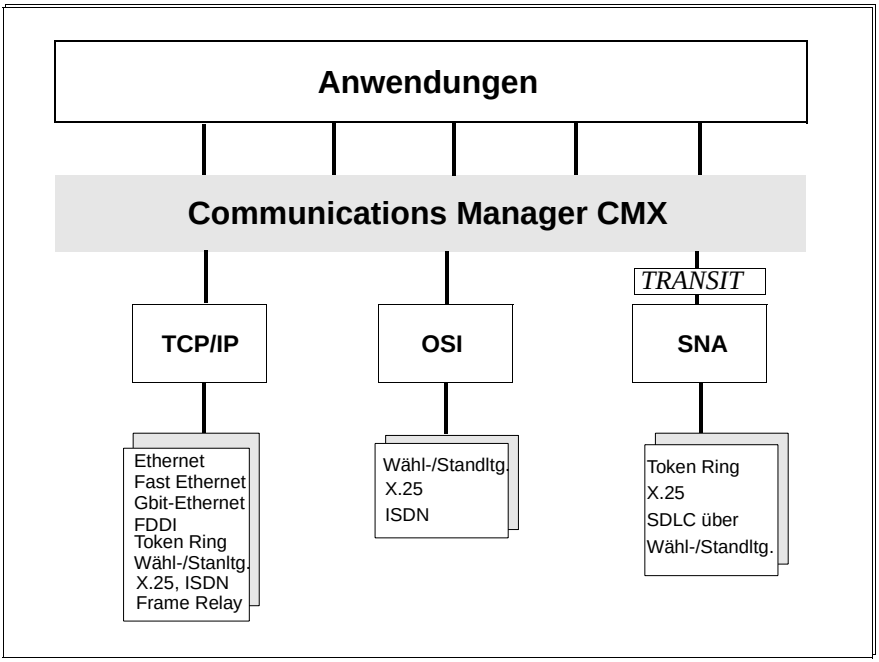


Bild 1: Netzzugänge und Transportprofile

Damit Rechner in lokalen Netzen oder über Weitverkehrsnetze hinweg kommunizieren können, müssen sie über eindeutige Adressen erreichbar sein. Die Adressierung ist jedoch von den verwendeten Protokollen abhängig und daher in den einzelnen Netzarchitekturen unterschiedlich. CMX unterstützt die Verwendung von TCP/IP-, ISO- und SNA-Netzadressen und ermöglicht dadurch netzunabhängige Kommunikation. Im Folgenden werden Netzarchitekturen sowie die Komponenten und Merkmale der wichtigsten Adressen vorgestellt.

2.2.1 TCP/IP-Architektur

TCP/IP kann über lokale und Weitverkehrsnetze aller Art betrieben werden.

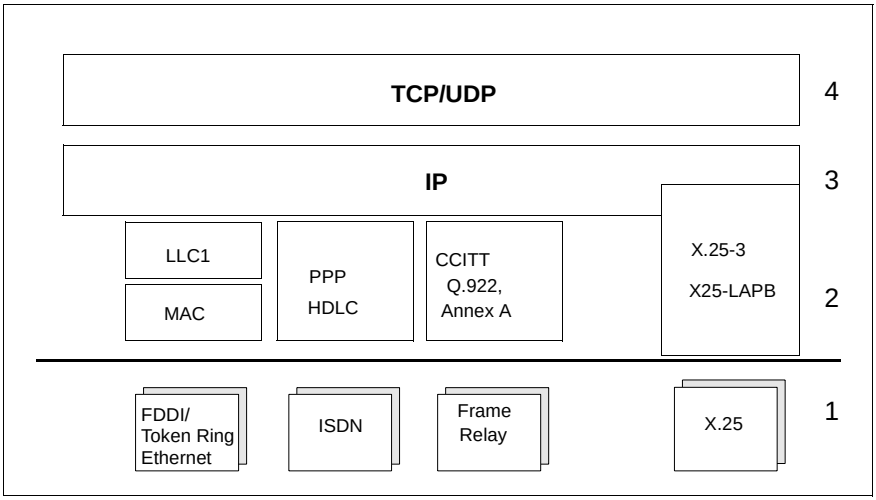


Bild 2: TCP/IP-Protokollstack

Schicht 2 des TCP/IP-Transportprofils für LANs ist in die MAC-Teilschicht und die LLC-Teilschicht gegliedert. In der MAC-Teilschicht (Medium Access Control) wird der Zugang auf das jeweilige physische LAN geregelt; in der LLC-Teilschicht (Logical Link Control) werden die Nachrichten ausgetauscht.

IP verwendet in Token Ring und FDDI-Netzen das verbindungslose Protokoll LLC1 mit einer IP-spezifischen Erweiterung namens SNAP. In Ethernet-LANs wird, abhängig von der MAC-Variante, kein LLC-Protokoll oder ebenfalls LLC1 verwendet.

Für IP über ISDN wird HDLC (mit oder ohne T.70-3) oder das Point-to-Point-Protokoll (PPP) verwendet. Oberhalb eines Frame-Relay-Subnetzes gilt das CCITT-Protokoll Q.922, Annex A, während über paketvermittelnde Netze X25-LAPB eingesetzt wird.

TCP/IP-Adressen

Eine Anwendung im TCP/IP-Netz (= Internet) wird über ihre Portnummer und die IP-Adresse des Rechners adressiert.

IP-Adresse

Es ist zwischen IPv4- und IPv6-Adressen zu unterscheiden:

– Eine IPv4-Adresse ist 32 Bit lang. Als Text, z. B. bei der Eingabe oder Ausgabe am Bildschirm, wird sie dargestellt, indem jeweils 8 Bit als Dezimalzahl wiedergegeben und durch Punkt voneinander getrennt werden (z. B. 139.22.112.88).

– Eine IPv6-Adresse ist 128 Bit lang. Hier werden in der Textdarstellung jeweils 16 Bit zusammengefasst, als Sedezimalzahl wiedergegeben und durch Doppelpunkt getrennt.

  Eine oder mehrere benachbarte Sedezimalzahlen mit Wert 0 können einmal pro Adresse als zwei unmittelbar aufeinanderfolgende Doppelpunkte abgekürzt werden (z. B. FE80..280:17FF:FE28.7BO8).

  Bei speziellen IPv6-Adressen, die aus IPv4-Adressen abgeleitet sind, kann der IPv4-Adressanteil auch in der IPv4-Schreibweise dargestellt werden (z. B. ::FFFF:139.22.112.88).

Portnummer

Eine Anwendung wird innerhalb des TCP/IP-Netzes mit der 2 Byte langen Portnummer adressiert. Die Kombination von Portnummer und IP-Adresse identifiziert den Sender bzw. Empfänger einer Nachricht im Netz eindeutig.

Beachten Sie bei der Vergabe von Portnummern für Ihre eigenen Anwendungen, dass für Standard-Anwendungen bestimmte Portnummern reserviert sind. So adressiert beispielsweise die Anwendung TELNET über die Portnummer 23, die Anwendung FTP über Portnummer 21.

Sie sollten für den eigenen Bedarf grundsätzlich nur Portnummern größer als 1024 konfigurieren. Die weltweit reservierten Portnummern werden regelmäßig als RFC (Request for Comment) von der International Electrotechnical Commission (IEC) veröffentlicht.

2.2.2 OSI-Architektur

Für die einzelnen WAN-Typen wurden von verschiedenen nationalen und internationalen Organisationen eine Reihe von OSI-Protokollprofilen festgelegt.

Im WAN (und ISDN) wird in der Regel auf der Schicht 4 das verbindungsorientierte Transportprotokoll IS8073, Klasse 0 bzw. 2 verwendet, das auf dem ebenfalls verbindungsorientierten Netzdienst X.25 oder auf T.70-3 aufsetzt.

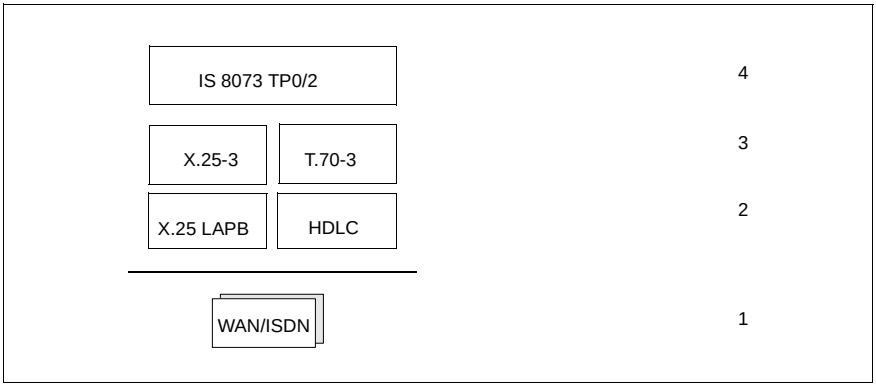


Bild 3: OSI-Protokollstacks

OSI-Transportadressen

OSI-Transportadressen bestehen aus der OSI-NSAP-Adresse und einem Transport-Selektor.

OSI-NSAP-Adresse

OSI-Adressen bestehen aus den Komponenten AFI (Authority and Format Identifier), IDI (Initial Domain Identifier) und DSP (Domain Specific Part). Nähere Informationen siehe Abschnitt „Adress-Komponenten und ihre Formate“ auf Seite 38). In vielen Fällen, insbesondere bei WAN-Anschlüssen, werden jedoch die Adressen des unterliegenden Subnetzes (z. B. X.25-Adressen) verwendet. Die OSI-NSAP-Adressen werden in diesen Fällen nicht benötigt.

Transport-Selektor

Mit dem Transport-Selektor (T-Selektor) meldet sich eine Anwendung beim OSI-Transportdienst an. Bei ankommendem Verbindungsaufbauwunsch identifiziert das Transportsystem anhand des T-Selektors die gerufene Anwendung.

2.3 Schnittstellen und Anwendungen

Oberhalb des Transportsystems, das durch eine der genannten Architekturen realisiert wird, fungiert CMX als Vermittler zu den Anwendungen, die das Transportsystem nutzen. Die Anwendungen können dabei unterschiedliche Programmschnittstellen nutzen, um auf die Transportsysteme zuzugreifen. Im Folgenden werden einige Standard-Anwendungen und ihre Schnittstellen genannt, die von CMX verwaltet und zum Ablauf gebracht werden.

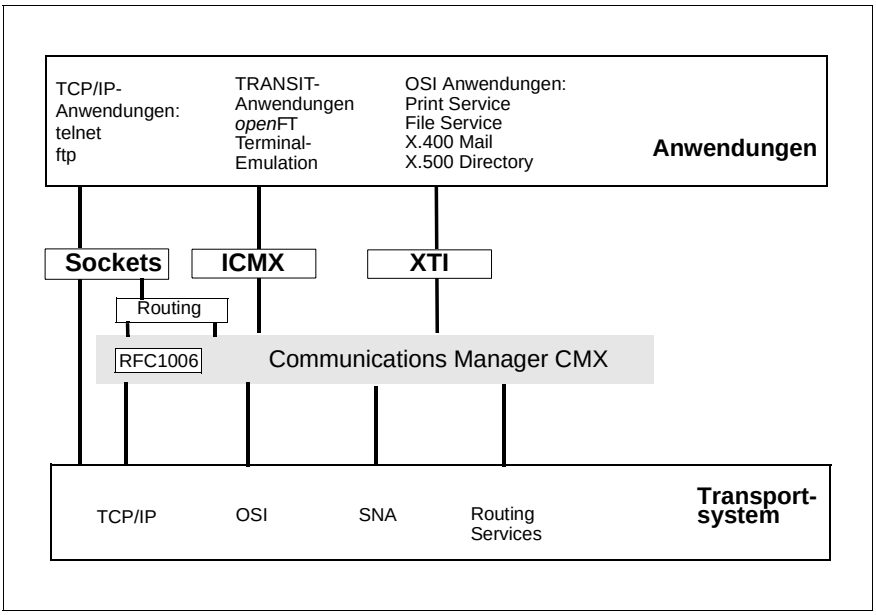


Bild 4: Standard-Anwendungen

Anwendungen, die auf der Sockets-Schnittstelle aufsetzen, sind zunächst auf das Transportsystem TCP/IP festgelegt. Mit den Routing Services von CMX können diese Anwendungen auch andere Transportsysteme nutzen. Die Sockets-Schnittstelle ist in allen UNIX-Systemen und auch in BS2000/OSD verfügbar. Sockets-Varianten gibt es für MS-DOS, Windows-Betriebssysteme. Typische Beispiele für Sockets-basierte Anwendungen sind *telnet* und *ftp*.

Die im Produkt CMX enthaltene ICMX-Schnittstelle eröffnet den Zugang zu allen Transportsystemen, die einen Transportdienst nach IS 8072 anbieten. Dazu gehören sowohl die reinen OSI-Transportsysteme als auch die verbreitete Variante RFC1006 über TCP/IP.

Die ICMX-Schnittstelle wird für alle gängigen UNIX-Systeme und im BS2000/OSD angeboten. Darüber hinaus gibt es Implementierungen für MS-DOS und Windows-Betriebssysteme.

Typische Beispiele für ICMX-Anwendungen sind z. B. *open*UTM, TRANSIT oder Anwendungen, die für den Endbenutzer konzipiert sind (z. B. EMDS, *open*FT oder MAIL.X).

Neben diesen Schnittstellen, die den Zugriff auf Basiskommunikationsdienste ermöglichen, bietet eine Reihe von Produkten weitere, „höhere“ Kommunikationsschnittstellen an, die die Anwendung von der Basiskommunikation unabhängig machen.

3 Adressierungskonzept

Die erforderlichen Daten, die Sie zum Konfigurieren von CMX benötigen, werden von der System-Komponente Transport Name Service (TNS) verwaltet. Dieses Kapitel beschreibt die Adressierung von Transportsystem-Anwendungen im TNS.

Jedes Netz und jedes Transportsystem verlangt spezielle Adress-Informationen, um die Kommunikationspartner adressieren zu können. CMX bietet den Transport Name Service TNS, mit dem Sie Namen und Adressen von TS-Anwendungen verwalten können.

TNS liest die Adress-Informationen aus einem Verzeichnis, dem TS-Directory (Transport Service Directory). In dem TS-Directory sind die Adress-Informationen jeder TS-Anwendung unter ihrem symbolischen Namen, dem GLOBALEN NAMEN der TS-Anwendung, abgelegt. Das TS-Directory enthält Informationen zu allen im lokalen System residierenden TS-Anwendungen und zu den potenziellen Kommunikationspartnern in fernen Systemen.

Eine TS-Anwendung arbeitet nur mit ihrem eigenen GLOBALEN NAMEN und mit den GLOBALEN NAMEN ihrer Kommunikationspartner.

TNS macht die TS-Anwendungen also von der umgebenden Konfiguration der Kommunikations-Hardware und -Software unabhängig. Diese Unabhängigkeit bezieht sich z. B. auf:

– Art und Anzahl der Communication Controller (CC), die in Ihrem Rechner installiert sind

– die Topologie des Netzes, in das Ihr Rechner integriert ist

– die Controller-Profile, die in Ihrem Rechner ablaufen

All diese Merkmale und Daten brauchen bei Verwendung des TNS innerhalb einer TS-Anwendung nicht mehr berücksichtigt zu werden, egal, ob sie am Ort ihres Ablaufes, am Ort der Partneranwendung oder auf dem Weg dorthin (Addressing, Routing) vorliegen.

Die genannten Konfigurationsabhängigkeiten verwaltet der TNS im TS-Directory. Sie sind dort als Eigenschaften zu den TS-Anwendungen abgelegt. Die Identifikation der TS-Anwendungen erfolgt über einen hierarchisch strukturierten Namen, ihren GLOBALEN NAMEN. Die Programmierschnittstelle ICMX(L) bietet Abfragefunktionen, mit denen eine TS-Anwendung auf diese Eigenschaf-

ten zugreifen kann. Auf diese Weise wird die ICMX-Anwendung von der Verarbeitung physischer Adressen entlastet. Sie ist somit leichter in neuen Umgebungen einzusetzen und unabhängig von Änderungen im Netz.

In den folgenden Abschnitten werden die für den TNS wichtigen Elemente, der GLOBALE NAME, die Eigenschaften und das TS-Directory, beschrieben. Wie ein TS-Directory verwaltet wird, erfahren Sie im Abschnitt „Adress-Verwaltung in TS-Directories“ auf Seite 16.

Feste Begriffe, wie z. B. der GLOBALE NAME oder die Namen der Eigenschaften von TS-Anwendungen, werden in Großbuchstaben angegeben, um sie innerhalb des Textes kenntlich zu machen. TS-Anwendungen, die im lokalen Endsystem residieren, werden im Folgenden als lokale TS-Anwendungen bezeichnet. Entsprechend heißen TS-Anwendungen, die in fernen Endsystemen residieren, ferne TS-Anwendungen.

3.1 Adress-Verwaltung in TS-Directories

Der TNS verwaltet alle Eigenschaften der TS-Anwendungen im Transport Service Directory (TS-Directory). Das TS-Directory besteht aus Einträgen, von denen jeder die Eigenschaften einer TS-Anwendung enthält. Jeder Eintrag wird durch den GLOBALEN NAMEN identifiziert. Das TS-Directory enthält Einträge für alle TS-Anwendungen im lokalen Endsystem und für alle TS-Anwendungen in fernen Endsystemen, die als Kommunikationspartner in Frage kommen.

3.2 Identifizierung durch GLOBALEN NAMEN

Um Kommunikationsbeziehungen herstellen zu können, müssen TS-Anwendungen in der Lage sein, sich gegenseitig zu adressieren - ähnlich, wie sich Teilnehmer im Telefonnetz anhand ihrer Telefonnummern adressieren. So wie der Telefonnetzbetreiber den Teilnehmern Telefonnummern zuweist, so ordnen Sie den TS-Anwendungen ihre Identifikationen explizit zu: Sie geben jeder lokalen und jeder fernen TS-Anwendung einen eindeutigen GLOBALEN NAMEN. Global bedeutet hier, dass der Name unabhängig von den zu benutzenden Netzen gültig ist.

Unter einem GLOBALEN NAMEN versteht man einen hierarchisch strukturierten Anwendungsnamen. Dieser Name lässt sich in maximal 5 Teile gliedern (Namensteil 1 bis 5). Von diesen ist der Namensteil 1 in der Hierarchie der höchste, der Namensteil 5 der niedrigste. Denken Sie sich als Modell ein (welt-

weites!) Telefonnummernverzeichnis mit Länderkennzahl, Städtevorwahl und Telefonnummern oder ein Adressbuch mit Nationalitätskennzeichen, Postleitzahl, Zustellungsbezirk, Straße mit Hausnummer und Name des Empfängers.

Für einige Standard-Anwendungen gibt es Namenskonventionen. Beispiel:

– der Enterprise File Transfer *open*FT benutzt die GLOBALEN NAMEN *$FJAM, $FJAM001, $FJAM002* usw.

– Das Produkt EMDS benutzt für seine Anwendungen die GLOBALEN NAMEN *dss_000*, *drs_000, dss_001*, *drs_001* usw.

Unter einem *Netz* wird dabei eine Gesamtheit von Rechnern verstanden, die nach einem bestimmten Schema adressiert werden, wie zum Beispiel das Internet (Adressierung durch Internet-Adresse, Beispiel: *139.22.195.99*). Jeder Rechner in einem Netz ist durch seine Netzadresse (Synonym: *NSAP-Adresse*) eindeutig identifiziert. Ein Rechner, der in mehrere Netze eingebunden ist, hat je eine spezifische Netzadresse für jedes dieser Netze.

3.3 Adress-Information im GLOBALEN NAMEN

Bei der Adressierung einer Verbindung wird zwischen lokaler (= auf dem eigenen System residierender) und ferner (= auf dem Partnersystem residierender) TS-Anwendung unterschieden. Sie werden jeweils durch einen GLOBALEN NAMEN identifiziert. Sie unterscheiden sich jedoch in den Adress-Informationen, die dem GLOBALEN NAMEN zugeordnet sind.

Sie müssen die GLOBALEN NAMEN aller lokalen TS-Anwendungen und aller TS-Anwendungen an fernen Rechnern, mit denen eine lokale TS-Anwendung kommunizieren soll, in das TS-Directory aufnehmen. Jedem dieser GLOBALEN NAMEN, d. h. jedem Blatt im Namensbaum, können Sie bestimmte Eigenschaften zuordnen. Welche Eigenschaften Sie einer TS-Anwendung zuordnen können, hängt davon ab, ob die TS-Anwendung im lokalen oder in einem fernen Endsystem residiert.

Die Eigenschaften LOKALER NAME und TRANSPORTADRESSE sind für die Kommunikation besonders wichtig. Jeder TS-Anwendung im lokalen System müssen Sie die Eigenschaft LOKALER NAME zuordnen. Jeder TS-Anwendung in einem fernen Endsystem müssen Sie die Eigenschaft TRANSPORTADRESSE zuordnen.

i Unabhängig davon, ob die TS-Anwendung lokal oder fern ist, können Sie auch die Session-Komponente und die Presentation-Komponente erfassen.

Die hierarchische Struktur des GLOBALEN NAMENS bewirkt die Anordnung aller GLOBALEN NAMEN in einem Namensbaum. Ein Blatt (Leaf Entity) im Namensbaum entspricht einer TS-Anwendung. Einem Blatt kann eine Auswahl von Eigenschaften (Property) zugeordnet werden, z. B. TRANSPORTADRESSE.

Der Pfad von der Wurzel des Namensbaumes zum Blatt wird durch den GLOBALEN NAMEN der TS-Anwendung vorgegeben. Der Name kann aus bis zu 5 Namensteilen bestehen, die den Weg von der Wurzel des Baumes über die (bis zu 4) Knoten zum Blatt angeben. Es können auch Namensteile übersprungen werden. Ein „vollständiger" Namensbaum mit allen Namensteilen sieht beispielsweise wie folgt aus:

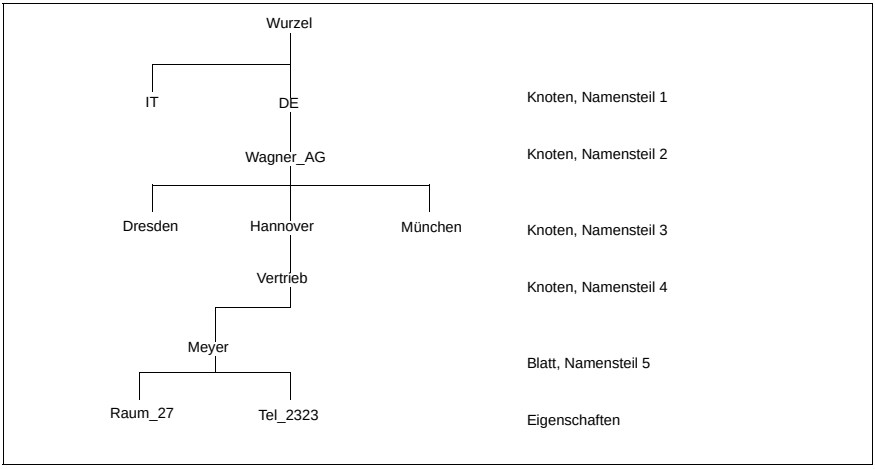


Bild 5: Beispiel für einen Namensbaum

Der zum Bild 8 korrespondierende GLOBALE NAME lautet in der Schreibweise des TNS:

```
Meyer.Vertrieb.Hannover.Wagner_AG.DE
```

Beachten Sie, dass hierbei Namensteil 5 zuerst genannt wird.

Übersicht der Merkmale eines GLOBALEN NAMENS

– Ein GLOBALER NAME ist ein Pfad im Namensbaum von der Wurzel zu einem Blatt

– Die Namensteile sind die Pfadkomponenten

– Die Festlegung von Knoten und Blatt erfolgt beim Einrichten des GLOBALEN NAMEN
– Die Namensteile 1 bis 4 können Pfadkomponenten zu einem Blatt sein
– Namensteil 5 kann nicht Pfadkomponente zu einem Knoten sein
– An einen Knoten kann ein weiterer Knoten oder ein Blatt unter Beachtung der Hierarchie angefügt werden
– Eigenschaften können nur einem Blatt zugeordnet werden

Im Namensbaum werden nur Blätter eingerichtet. Ein Knoten, an denen keine Blätter hängen, kann nicht explizit erzeugt werden. Ein Knoten wird jedoch implizit eingerichtet, wenn Blätter eingerichtet werden, und gelöscht, wenn alle Blätter gelöscht werden, die ihm zugeordnet sind.

GLOBALEN NAMEN strukturieren

Wie die Struktur des Namensbaumes angelegt wird, ob mit oder ohne Differenzierung nach Wurzel und Knoten und Blatt, bleibt dem einzelnen Anwendungsfall und der Gesamtkonfiguration aller TS-Anwendungen überlassen. Es liegt im Ermessen des Netzadministrators, wie „tief" er die Baumstruktur anlegt. Nicht zuletzt die Anzahl der TS-Anwendungen wird die Struktur prägen: Für wenige Anwendungen wird ein „flacher" Baum ohne Knoten ausreichen. Bei vielen Anwendungen sollten Sie sich der Strukturierung mit den Knoten bedienen, um deren Vorteile hinsichtlich Übersichtlichkeit, Zugriffsoptimierung etc. auszunutzen. Es empfiehlt sich, den Baum nach organisatorischen oder topologischen Gesichtspunkten zu strukturieren.

Bedeutung der Namensteile

Die Bezeichnung der Namensteile entspricht den Vorschlägen der internationalen Normungsgremien von ISO, CCITT und ECMA.

Für einen „vollständigen" Namensbaum ergibt sich folgende Zuordnung:

Namens-teil	Bezeichnung	Bedeutung	Länge in Byte	im Baum
1	TS_COUNTRY	Country	2	Knoten, Blatt
2	TS_ADMD	Administrative Domain	16	Knoten, Blatt
3	TS_PRMD	Private Domain	16	Knoten, Blatt

Tabelle 1: Bedeutung der Namensteile

Namens-teil	Bezeichnung	Bedeutung	Länge in Byte	im Baum
4	TS_OU	Organisation Unit	10	Knoten, Blatt
5	TS_PN	Personal Name	30	Blatt

Tabelle 1: Bedeutung der Namensteile

3.3.1 Lokale TS-Anwendung

Eine lokale *TS-Anwendung* muss sich beim Transportsystem anmelden, um Kommunikation betreiben zu können. Dabei muss sie anzeigen, welche Transportsysteme sie benutzen will. Die verschiedenen Transportsysteme werden durch unterschiedliche Adress-Formate identifiziert. In manchen Fällen bezeichnet das Adress-Format auch eine bestimmte Kombination von Transportsystem und Subnetz. Beispiele:

– Adress-Format WANSBKA (für Transportsystem OSI TP0/2) für OSI-Verbindungen über WAN gemäß ISO 8072

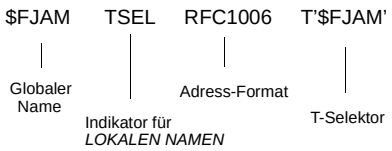
– Adress-Format RFC1006 (für Transportsystem RFC1006) für Rechnerkopplung über TCP/IP mit RFC1006-Konvergenzprotokoll

Für jedes Subnetz, über das die lokale Anwendung kommunizieren will, muss dem Transportsystem ein *T-Selektor* (Transport-Selektor) mitgeteilt werden, unter dem die Anwendung identifiziert werden kann. Der T-Selektor enthält unterschiedliche Adress-Informationen, z. B.

– LU-Name und LU-Nummer für ein SNA-Subnetz
– eine beliebige ASCII-Zeichenfolge für ein OSI-Subnetz

Einträge für lokale TS-Anwendungen beginnen jeweils mit dem Indikator TSEL.

Beispiel



Dem GLOBALEN NAMEN einer lokalen TS-Anwendung ordnen Sie daher einen Satz von T-Selektoren zusammen mit den Adress-Formaten zu. Die Summe dieser Einträge wird LOKALER NAME genannt.

Beispiel: Die durch den GLOBALEN NAMEN *$FJAM* bezeichnete Anwendung soll bei den Transportsystem RFC1006 (Adress-Format RFC1006) und dem lokalen Loopback (Adress-Format LOOPSBKA) bekannt gemacht werden. Dafür sind als LOKALER NAME drei TSEL-Einträge anzugeben:

```
$FJAM \
   TSEL RFC1006 T´$FJAM´
   TSEL LOOPSBKA T´$FJAM´
```

3.3.2 Ferne TS-Anwendung

Um eine *ferne TS-Anwendung* zu adressieren, werden folgende Informationen benötigt:

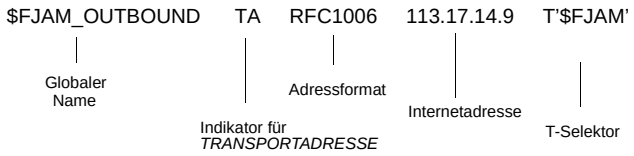
– in welchem fernen System läuft sie ab (Netzadresse)

– über welches Transportsystem kann sie erreicht werden (Adress-Format des Subnetzzugangs)

– wie wird sie im Transportsystem identifiziert (T-Selektor der fernen Anwendung)

Diese Informationen werden in CMX als TRANSPORTADRESSE bezeichnet. Die TRANSPORTADRESSE wird einem GLOBALEN NAMEN zugeordnet, durch den die ferne TS-Anwendung identifiziert wird.

Beispiel



4 Installation / Deinstallation von CMX (UNIX)

Die Installation bzw. Deinstallation von CMX (UNIX) ist abhängig von der jeweiligen Betriebssystemversion und wird in der Readme-Datei beschrieben.

Nachdem Sie CMX (UNIX) erfolgreich installiert haben, müssen Sie für die gewünschten lokalen und entfernten Kommunikationspartner die entsprechenden Einträge im TS-Directory vornehmen. Dazu dient der TNSX-Compiler *tnsxcom* (siehe Abschnitt „Konfigurieren mit tnsxcom“ auf Seite 27). Wenn Sie diese Einträge erfolgreich beendet haben, können Sie TS-Anwendungen oder Systemanwendungen, die auf CMX (UNIX) aufsetzen, ablaufen lassen.

5 TNS-Konfiguration

In diesem Kapitel erhalten Sie eine Darstellung der CMX-Kommandoschnittstelle zum Konfigurieren von TS-Anwendungen. Dabei wird das Kommando *tnsxcom* zum Konfigurieren im Transport Name Service (TNS) beschrieben. Mit TNS verwalten Sie die TS-Anwendungen und deren TRANSPORTADRESSE.

5.1 Vorgehensweise bei der Konfigurierung

Die folgenden zwei Abschnitte geben eine Übersicht der Arbeitsschritte beim Konfigurieren mit tnsxcom und fssadm an der Kommandoschnittstelle.

5.1.1 TNS: anwendungsspezifische Konfigurierung

Sie können eine TNS-Konfigurationsdatei im Format *tnsxfrm* mit einem beliebigen Editor erstellen und daraus im Kommandomodus ein TS-Directory erstellen. Die Syntax dieser Datei, die als Datenbasis für ein TS-Directory dienen soll, ist im Abschnitt „Syntax der TNS-Konfigurationsdatei“ auf Seite 29 beschrieben. Im Folgenden erhalten Sie eine Übersicht der wichtigsten Schritte beim Konfigurieren im TNS.

- Prüfen Sie die Konfigurationsdatei.

  Mit *tnsxcom -s␣file* wenden Sie eine Syntaxprüfung auf die Datei *file* an (Check-Modus). Der TNS protokolliert mögliche Syntaxfehler. Das TS-Directory wird nicht verändert.

- Erstellen Sie ein neues TS-Directory.

  Mit *tnsxcom -l␣file* füllen Sie ein bisher leeres TS-Directory mit den Einträgen aus der Datei *file* (Lade-Modus).

- Prüfen und aktualisieren Sie das TS-Directory.

  Mit *tnsxcom -S␣file* prüfen und aktualisieren Sie ein TS-Directory (check/update-Modus). Wie bei Option *-s* erfolgt in einem ersten Lauf zuerst die Syntaxprüfung auf die gesamte Datei *file*. Treten in *file* keine Syntaxfehler auf, so aktualisiert *tnsxcom* dann das TS-Directory in einem zweiten Lauf.

► TS-Directory erweitern.

Mit *tnsxcom -u␣file* fügen Sie einem bestehenden TS-Directory die Einträge aus der Datei *file* hinzu (Update-Modus). Enthält *file* Einträge zu GLOBALEN NAMEN, die bereits im TS-Directory vorhanden sind, so werden die alten Einträge überschrieben bzw. ergänzt.

► Interaktiv administrieren.

Anstatt die Einträge in einer Datei *file* vorzubereiten und dann an *tnsxcom* zu übergeben, können Sie nach Aufruf von *tnsxcom -i* die Konfigurationsdatei auch interaktiv bearbeiten. Abgesehen vom Erfassen neuer GLOBALER NAMEN sind im interaktiven Betrieb insbesondere folgende Eingaben möglich:

– Eintrag für TS-Anwendung aus dem TS-Directory löschen

Sie löschen den gesamten Eintrag einer TS-Anwendung aus dem TS-Directory mit folgendem Kommando:

```
Global_name␣DEL
```

Alle Eigenschaften, die der TS-Anwendung *Global_name* zugeordnet sind, und der GLOBALE NAME werden aus dem TS-Directory gelöscht. Die TS-Anwendung ist dem TNS dann nicht mehr bekannt.

– Eigenschaften einer TS-Anwendung anzeigen

Mit folgendem Kommando können Sie sich z. B. die zuvor erfassten oder geänderten Einträge einer TS-Anwendung zur Kontrolle am Bildschirm anzeigen lassen:

```
Global_name␣DISP
```

– TS-Directory wechseln

Sie können mit Hilfe des folgenden Kommandos in der Datei in ein anderes TS-Directory umschalten:

```
DIR␣n
```

Für *n* ist die Nummer des TS-Directory anzugeben, in das umgeschaltet werden soll.

Die folgenden Eingabesätze in der Datei beziehen sich dann auf dieses TS-Directory. Es wird solange bearbeitet, bis explizit in ein anderes TS-Directory umgeschaltet oder die Eingabe mit CTRL D beendet wird.

► TS-Directory auswählen.

TNS unterstützt bis zu 9 verschiedene TS-Directories. Alle oben aufgeführten Aktionen beziehen sich auf das Directory DIR1. Über die Option *-d␣* num (num=1,2...9) können Sie auch andere Directories bearbeiten.

5.2 Konfigurieren mit tnsxcom

Auf Shell-Ebene erstellt, aktualisiert und liest man TS-Directories mit Hilfe des TNS-Compilers *tnsxcom*. *tnsxcom* übersetzt Eingabesätze, die Sie im Format *tnsxfrm* an den Compiler übergeben, in das Format des TS-Directory und trägt die erzeugten Einträge in das TS-Directory ein. Ebenso liest der *tnsxcom* die Einträge der TS-Directories und übersetzt sie in ein abdruckbares Format. Der *tnsxcom* wird mit dem Kommando *tnsxcom* aufgerufen. Die Syntax von *tnsxcom* ist im Kommandokatalog (Abschnitt „TS-Directory erstellen, aktualisieren, lesen (tnsxcom)“ auf Seite 91) beschrieben.

In diesem Abschnitt wird beschrieben:

– welche Aktionen Sie mit dem *tnsxcom* durchführen können

– wie Sie die Eigenschaften erfassen müssen, die ins TS-Directory aufgenommen werden sollen (Format der Eingaben für den *tnsxcom*)

– in welchem Format die Adressen und T-Selektoren für die verschiedenen Transportsysteme angegeben werden müssen

– wie Sie GLOBALE NAMEN in den Eingabesätzen pauschal erweitern können, z. B. wenn Sie Änderungen an einem Zweig des Namensbaums vornehmen wollen (siehe Abschnitt „Namen mit gleichen höherwertigen Namensteilen“ auf Seite 48).

– wie der *tnsxcom* arbeitet

5.2.1 TS-Directories verwalten

Der TNS unterstützt gleichzeitig bis zu 9 verschiedene TS-Directories mit den Identifikationen 1-9. Die TS-Directories sind im Dateisystem als Dateiverzeichnisse *DIR1, DIR2, ... DIR9* abgelegt. Das TS-Directory *DIR1* ist das Standard-Directory, auf das TS-Anwendungen grundsätzlich zugreifen. Die anderen TS-Directories können Sie zum Beispiel als Sicherungskopien oder als experimentelle Bestände verwenden.

Wie Sie ein TS-Directory erstellen, ändern und Informationen dazu abfragen können, erfahren Sie im Abschnitt „Erfassungsregeln für TNS-Dateien“ auf Seite 46.

Folgende Aktionen können Sie mit dem *tnsxcom* ausführen:

– neue TS-Directories erzeugen.

  Mit dem *tnsxcom* können Sie neue TS-Directories DIR<n> (<n> = 1,...,9) erzeugen. Dazu erstellen Sie mit einem beliebigen Editor eine Datei. In dieser Datei erfassen Sie alle TS-Anwendungen mit ihren Eigenschaften, die in dieses TS-Directory eingetragen werden sollen, im Format *tnsxfrm* des *tnsxcom*. Dann rufen Sie den *tnsxcom* im LADE-Modus auf (*tsnxcom -l datei*). Er erzeugt aus den Sätzen der Datei die Einträge für das TS-Directory und schreibt diese in das zuvor leere TS-Directory. Dieses TS-Directory (DIR<n>) darf vorher nicht existieren. Sie dürfen es insbesondere nicht mit *mkdir* anlegen. Das neue TS-Directory DIR<n> wird implizit vom *tnsxcom* angelegt. Die Dateien des TS-Directory werden von ihm erzeugt.

– TS-Directories aktualisieren.

  Mit dem *tnsxcom* können Sie neue TS-Anwendungen in ein bestehendes TS-Directory aufnehmen bzw. bestehende Einträge zu TS-Anwendungen aus dem TS-Directory löschen. Sie können TS-Anwendungen neue Eigenschaften zuordnen, Eigenschaften ändern und löschen.

  Sie können bei der Aktualisierung eines TS-Directory genauso vorgehen wie bei der Erzeugung eines neuen TS-Directory und die Änderungen in einer Datei erfassen. Sie müssen *tnsxcom* dazu im UPDATE-Modus aufrufen (siehe Abschnitt „Syntax der TNS-Konfigurationsdatei“ auf Seite 29). Sie können ein TS-Directory aber auch aktualisieren, indem Sie dem *tnsxcom* die Änderungen interaktiv über die Standardeingabe übergeben. Dazu rufen Sie *tnsxcom* im INTERAKTIV-Modus auf (siehe Abschnitt „TS-Directory“ auf Seite 50). Das Format der Eingaben ist in beiden Modi gleich. Das Format der Eingaben ist auch unabhängig davon, ob Sie ein neues TS-Directory erstellen oder ein bestehendes aktualisieren wollen. Der *tnsxcom* muss lediglich im entsprechenden Modus aufgerufen werden.

– TS-Directories lesen.

  Ein TS-Directory besteht hauptsächlich aus nicht abdruckbaren Zeichen. Wollen Sie ein TS-Directory lesen, so können Sie es sich vom *tnsxcom* in eine abdruckbare Form aufbereiten und in eine Datei schreiben lassen. Diese Datei können Sie auch wieder als Eingabe für den *tnsxcom* verwenden.

Mit dieser Funktion können Sie ein TS-Directory eines anderen Rechners auf Ihren Rechner portieren. Sie müssen es an dem fremden Rechner mit dem *tnsxcom* in eine Datei schreiben und diese Datei an Ihrem Rechner einlesen. Hier übersetzen Sie das TS-Directory erneut mit dem *tnsxcom*.

Prüfen Sie vor dem Übersetzen, ob in den Einträgen des TS-Directory an Ihrem Rechner residierende TS-Anwendungen als lokale TS-Anwendungen erfasst sind. Entsprechend müssen die TS-Anwendungen, die an dem fremden Rechner residieren, als ferne TS-Anwendungen erfasst sein.

Das am fremden Rechner erzeugte TS-Directory können Sie auch in ein TS-Directory einfügen, das an Ihrem Rechner bereits existiert. Das TS-Directory hat dann die richtige Struktur für den TNS, auch wenn das Ausgangs-TS-Directory mit einer älteren TNS-Version erstellt wurde.

Welche dieser Aktionen der *tnsxcom* ausführen soll, bestimmen Sie durch die Angabe bestimmter Optionen beim Kommando *tnsxcom*.

5.2.2 Syntax der TNS-Konfigurationsdatei

Alle Einträge, die ins TS-Directory aufgenommen werden sollen, müssen in Form der folgenden Eingabesätze übergeben werden.

```
[Global_name␣]type[␣data-Felder]
```

Global_name, *type* und *data* bezeichnen Felder. Eckige Klammern zeigen an, dass es sich um optionale Felder handelt.

Die Bedeutung der einzelnen Felder innerhalb eines Eingabesatzes wird im Folgenden erläutert.

5.2.2.1 GLOBALER NAME

Im Feld *Global_name* geben Sie den GLOBALEN NAMEN der TS-Anwendung an, dem die in den folgenden Feldern beschriebene Eigenschaft zugeordnet werden soll. Wenn Sie für eine TS-Anwendung mehrere Sätze erfassen, so müssen Sie den GLOBALEN NAMEN nur im ersten Satz angeben. In den folgenden Sätzen können Sie das *name*-Feld leer lassen. Die Sätze müssen jedoch unmittelbar hintereinander stehen. D. h. ist in einem Satz das *name*-Feld leer, so gilt der zuletzt in einem Satz angegebene Wert.

Der GLOBALE NAME besteht aus 1 bis 5 hierarchisch angeordneten Namensteilen *Npi* (i = 1, 2, 3, 4, 5). *Np1* ist der Namensteil der höchsten Hierarchiestufe (TS_COUNTRY), *Np5* der der niedrigsten (TS_PN). Siehe hierzu auch Abschnitt „Adress-Information im GLOBALEN NAMEN“ auf Seite 17.

Groß- und Kleinbuchstaben haben unterschiedliche Bedeutung („case sensitivity"). Enthalten die Namensteile Sonderzeichen (siehe Abschnitt „Zeichen mit Sonderbedeutung“ auf Seite 46), deren Sonderbedeutung eine Mehrdeutigkeit der Syntax verursachen würde, so müssen diese Sonderzeichen mit \ (Gegenschrägstrich) oder durch Apostrophierung entwertet werden.

Im Zweifelsfall sollten Sie jedes Sonderzeichen entwerten. Ist die Entwertung überflüssig, so wird sie ignoriert. Die maximalen Längen für *Npi* sind:

Namensteil Npi	1	2	3	4	5
Länge in byte	2	16	16	10	30

Tabelle 2: Maximale Länge der Namensteile

Die Anordnung der *Npi* in *name* erfolgt mit von links nach rechts ansteigender Hierarchie, wobei die *Npi* durch . (Punkt) getrennt werden (Np5.Np4.Np3.Np2.Np1). Leere *Npi* (i = 1, 2, 3, 4, 5) sind zulässig, das folgende Trennzeichen . (Punkt) muss jedoch angegeben werden. Beispiel:
.Np4..Np2.Np1

Endet ein GLOBALER NAME mit mindestens einem . (Punkt), so ist der GLOBALE NAME absolut, d. h. er ist direkt unter ROOT des Namensbaumes eingeordnet. Endet er dagegen nicht mit . (Punkt), so ist er relativ. Relativ angegebene GLOBALE NAMEN werden um einen Ursprung erweitert, sofern Sie einen Ursprung (Origin) definiert haben (siehe auch Abschnitt „Namen mit gleichen höherwertigen Namensteilen“ auf Seite 48).

Gültige Beispiele für die Angabe des GLOBALEN NAMENS sind:

Np5
: nur Namensteil 5, relativ (eventuell zu ROOT)

Np5.
: nur Namensteil 5, absolut

Np5.Np4
: nur Namensteile 5 und 4, relativ (eventuell zu ROOT)

Np5....Np1
: nur Namensteile 5 und 1, absolut

..Np3
nur Namensteil 3, relativ (eventuell zu ROOT)

.Np4..Np2.
nur Namensteile 4 und 2, absolut

5.2.2.2 Typ der Anwendung

Der Eintrag für den Typ bzw. die Eigenschaften der Anwendung hat folgende Syntax:

```
type[␣data-Felder]
```

Der Wert für *type* bestimmt den Typ des Eintrags (auch *Eigenschaft* genannt); d. h. *type* gibt an, ob es sich um einen LOKALEN NAMEN oder eine TRANSPORTADRESSE, eine *Session*- oder eine *Presentation-Komponente* handelt. In den *data*-Feldern ist der Wert der Eigenschaft anzugeben, z. B. die TRANSPORTADRESSE. Die einzelnen *data*-Felder sind durch Zwischenraum voneinander zu trennen.

Mögliche Werte für *type* sind:

– TSEL für Transport-Selektor-Eintrag einer lokalen Anwendung
– TA für TRANSPORTADRESSE einer fernen Anwendung
– PSEL für Presentation Selector
– SSEL für Session Selector

Sätze, mit denen Sie neue Einträge für TS-Anwendungen erzeugen wollen, und Sätze, mit denen Sie bestehende Einträge verändern oder erweitern wollen, haben die gleiche Form. Ist eine TS-Anwendung oder eine Eigenschaft einer TS-Anwendung im TS-Directory noch nicht vorhanden, so wird aus den Angaben des Satzes ein neuer Eintrag erzeugt. Ist bereits ein Eintrag vorhanden, so wird er entsprechend den Angaben in dem Satz geändert.

Sie müssen für jede Eigenschaft, die Sie erfassen wollen, mindestens einen Satz übergeben. Jeder TSEL-Eintrag eines LOKALEN NAMENS muss z. B. in einem eigenen Satz übergeben werden. Jeder Satz entspricht einer logischen Zeile. Ist es notwendig, dass sich ein Satz über mehrere Zeilen erstreckt, so muss das Zeilenende durch \ (Gegenschrägstrich) entwertet werden oder die Angaben müssen in () (runde Klammern) eingeschlossen werden.

Es ist nicht möglich, nur den GLOBALEN NAMEN einer TS-Anwendung ins TS-Directory aufzunehmen, ohne ihm eine Eigenschaft zuzuordnen.

Wie Sie die einzelnen Eigenschaften erfassen, ist im Folgenden beschrieben.

5.2.3 LOKALER NAME

Der LOKALE NAME einer TS-Anwendung besteht aus einem oder mehreren TSEL-Einträgen (einem TSEL-Eintrag pro Transportsystem, über das die TS-Anwendung kommunizieren soll). Für jeden T-Selektor, den Sie der lokalen TS-Anwendung *Global_name* zuordnen wollen, müssen Sie einen Satz übergeben. Der Satz muss wie folgt aufgebaut sein:

```
[Global_name␣]TSEL[␣addrform[␣data-Feld mit T-Selektor]]
```

Für *addrform* sind dieselben Angaben wie beim Erfassen der TRANSPORTADRESSE zulässig. Der im *data*-Feld angegebene Wert wird als T-Selektor für das Transportsystem *addrform* in das TS-Directory aufgenommen. Ist im TS-Directory für dieses Transportsystem bereits ein T-Selektor im LOKALEN NAMEN enthalten, wird dieser mit dem neuen Wert überschrieben. Enthält der LOKALE NAME bisher noch keinen T-Selektor für dieses Transportsystem, so wird dieser T-Selektor zum bisherigen LOKALEN NAMEN hinzugenommen. Ist das *data*-Feld leer, so wird ein bereits vorhandener T-Selektor für das entsprechende Transportsystem aus dem Eintrag im TS-Directory gelöscht. Hierbei erfolgt weder eine Warnung noch eine Fehlermeldung, wenn ein solcher T-Selektor nicht im LOKALEN NAMEN vorhanden ist. Ist der zu löschende T-Selektor die einzige Komponente des LOKALEN NAMENS, so wird die Eigenschaft LOKALER NAME für diese TS-Anwendung gelöscht.

Die beschränkte Länge des LOKALEN NAMENS erlaubt die Aufnahme von höchstens 8 verschiedenen T-Selektoren. Für mehrere Transportsysteme gleichlautende T-Selektoren beanspruchen dabei nur einen der 8 Speicherplätze. Hiervon ausgenommen sind die Transportsysteme mit *addrform* LANINET und EMSNA, d. h. die zu diesen Transportsystemen gehörigen T-Selektoren gelten unabhängig von ihrem Wert immer als von anderen verschieden.

T-Selektoren können in verschiedenen Formaten angegeben werden (siehe folgende Beispiele sowie Abschnitt „Adress-Komponenten und ihre Formate“ auf Seite 38). Ihre Länge ist auf 10 Zeichen beschränkt (im TRANSDATA-Format: 8 Zeichen). Die Sonderzeichen ’ (Apostroph) und \ (Gegenschrägstrich) müssen durch \ (Gegenschrägstrich) entwertet werden, falls sie zum T-Selektor gehören sollen.

Die folgende Tabelle enthält die erlaubten Angaben für T-Selektoren bei den verschiedenen Adress-Formaten. Die Bedeutung der Adress-Formate und T-Selektor-Formate sowie die Darstellungsformate für die T-Selektoren finden Sie ab Seite 36.

Adress-Format	**T-Selektor-Format**
EMSNA	LU-Name, LU-Nummer
LANINET	Portnummer
LOOPSBKA	T-Selektor
RFC1006	T-Selektor
SDLCSBKA	Stationsname
TRSNA	T-Selektor
WANNEA	Stationsname
WANSBKA	T-Selektor
WAN3SBKA	T-Selektor

Tabelle 3: Adress- und T-Selektor-Formate

Beispiel für LOKALE NAMEN

```
Global_name  type addrform T-Selektor
-------------------------------------------------------------------
loopleer     TSEL LOOPSBKA V''          ; leerer T-Selektor
laninet      TSEL LANINET  A'4712'      ; dezimale Portnummer
rfc1006      TSEL RFC1006  A'Cologne'
iso          TSEL WANSBKA  E'wansbka'   ; in EBCDIC abzulegen
x28          TSEL WAN3SBKA A'WAN3'      ; in ASCII abzulegen
wan1         TSEL WANNEA                ; Stationsname löschen
```

5.2.4 TRANSPORTADRESSE

Sätze, mit denen Sie die TRANSPORTADRESSE einer fernen TS-Anwendung erfassen, haben das folgende Format:

```
[Global_name␣]TA[␣addrform[␣data-Felder mit Adress-
Komponenten]]
```

TA ist der Indikator für eine TRANSPORTADRESSE.

Mit dem Adress-Format *addrform* geben Sie den Typ des verwendeten Transportsystems an. Beim Erfassen einer TRANSPORTADRESSE erzeugt TNS stets auch einen Eintrag für das Transportsystem. In den folgenden *data*-Feldern übergeben Sie die Adress-Komponenten.

Wollen Sie eine TRANSPORTADRESSE ändern, die bereits ins TS-Directory eingetragen ist, so geben Sie in den *data*-Feldern die neuen Adress-Komponenten an. Der Eintrag im TS-Directory wird dann mit der neuen TRANSPORTADRESSE überschrieben.

Wollen Sie eine TRANSPORTADRESSE aus dem TS-Directory löschen, so entfernen Sie die Einträge für die Adress-Komponenten. Die TRANSPORT-ADRESSE wird dann für diese TS-Anwendung aus dem TS-Directory gelöscht.

Im Folgenden sind die erlaubten Angaben für das Adress-Format *addrform* und die zu *addrform* anzugebenden Adress-Komponenten aufgelistet. In eckigen Klammern eingeschlossene Adress-Komponenten sind optional. Die Bedeutung der Adress-Formate und der Adress-Komponenten sowie das Format, in dem Sie die einzelnen Adress-Komponenten übergeben müssen, sind im Abschnitt „Adress-Komponenten und ihre Formate“ auf Seite 38 beschrieben.

addrform	**Adress-Komponenten**
EMSNA	LU-Name, Rechner-/Regionsnummer
LANINET	Internet-Adresse oder *HOST* Hostname, Portnummer
LOOPSBKA	T-Selektor
RFC1006	Internet-Adresse oder *HOST* Hostname, [*PORT* Portnummer], T-Selektor
SDLCSBKA	Rufnummer, [WAN CC/Leitungskennzeichen]
TRSNA	Sym-Dest-Name, T-Selektor
WANNEA	Stationsname, Rechner-/Regionsnummer, [SNPA-Information], [WAN CC/Leitungskennzeichen]
WANSBKA	OSI-NSAP, T-Selektor [TPI] [TPC] [WAN CC/Leitungskennzeichen]
WAN3SBKA	SNPA-Information [T-Selektor] [WAN CC/Leitungskennz.]

Tabelle 4: Adress-Formate und zugehörige Adress-Komponenten

Beispiel für TRANSPORTADRESSEN

```
Global_name  type addrform  Adress-Komponenten
-------------------------------------------------------------
neate     TA   WANNEA    T'$DIALOG' 1/18 WAN 1:1 2:3
X25      (TA   WANSBKA   X.121 4589004033 ; DTE-Adresse (IDI)
                         A'dtxp-33-01'    ; T-Selektor
                         2/0              ; TPC)
tcp/ip    TA   LANINET   128.0.1.23 A'4711'
rfc1006   TA   RFC1006   HOST D018B016 A'Cologne'
```

5.2.5 Session-Komponente

Die Session-Komponente erweitert die TRANSPORTADRESSE einer fernen TS-Anwendung zu einer SSAP-Adresse bzw. den LOKALEN NAMEN einer lokalen TS-Anwendung um einen S-Selektor. Die SSAP-Adresse ist die Adresse einer TS-Anwendung in der Kommunikationssteuerschicht (Session Layer, Schicht 5 des OSI-Referenzmodells).

Die Session-Komponente einer TS-Anwendung wird in folgendem Satz übergeben:

```
[Global_name␣]SSEL[␣data-Feld mit S-Selektor]
```

Falls das *data*-Feld leer ist, so wird die Session-Komponente aus der Eigenschaft TRANSPORTADRESSE bzw. LOKALER NAME entfernt. Ist bereits ein S-Selektor für diese TS-Anwendung im TS-Directory vorhanden, so wird dieser „alte" Wert durch den angegebenen Wert überschrieben. Es ist nicht notwendig, dass der TS-Anwendung vor dem Eintrag des S-Selektors bereits eine TRANSPORTADRESSE bzw. ein LOKALER NAME zugeordnet wurde.

Die folgende Tabelle enthält die erlaubten Angaben für den S-Selektor.

S-Selektor	**Bedeutung**
SSEL	SSEL-Eintrag löschen
SSEL V''	Leereintrag für S-Selektor
SSEL *ssel*	ssel-Darstellungsformate: A'string': string von maximal 16 Zeichen; Ablage in ASCII E'string': string von maximal 16 Zeichen; Ablage in EBCDIC X'string': gerade Anzahl von Hexziffern in string; max.32 T'string': nach TRANSDATA-Konventionen

Tabelle 5: Erlaubte Angaben für den S-Selektor

5.2.6 Presentation-Komponente

Die Presentation-Komponente erweitert die TRANSPORTADRESSE bzw. die SSAP-Adresse einer fernen TS-Anwendung zu einer PSAP-Adresse. Bei einer lokalen TS-Anwendung erweitert die Presentation-Komponente den LOKALEN NAMEN um einen P-Selektor.

Die PSAP-Adresse ist die Adresse einer TS-Anwendung in der Darstellungsschicht (Presentation Layer, Schicht 6 des OSI-Referenzmodells).

Die Presentation-Komponente einer TS-Anwendung wird in folgendem Satz übergeben:

```
[name␣]PSEL[␣data-Feld mit P-Selektor]
```

Falls das *data*-Feld leer ist, so wird die Presentation-Komponente aus der Eigenschaft TRANSPORTADRESSE bzw. LOKALER NAME entfernt. Ist bereits eine Presentation-Komponente für die angegebene TS-Anwendung im TS-Directory vorhanden, so wird dieser „alte" Wert durch den angegebenen Wert überschrieben. Es ist nicht notwendig, dass der TS-Anwendung vor dem Eintrag des P-Selektors bereits eine TRANSPORTADRESSE bzw. ein LOKALER NAME zugeordnet wurde. Falls in der TRANSPORTADRESSE bzw. in dem LOKALEN NAMEN noch keine Session-Komponente enthalten ist, so wird für die Session-Komponente beim Eintrag der Presentation-Komponente automatisch der Leereintrag (SSEL V'') erzeugt.

Die folgende Tabelle enthält die erlaubten Angaben für den P-Selektor.

P-Selektor	**Bedeutung**
PSEL	PSEL-Eintrag löschen
PSEL V"	Leereintrag für P-Selektor
PSEL *psel*	psel-Darstellungsformate: A'string': string von maximal 16 Zeichen; Ablage in ASCII E'string': string von maximal 16 Zeichen; Ablage in EBCDIC X'string': gerade Anzahl von Hexziffern in string; max.32 T'string': nach TRANSDATA-Konventionen

Tabelle 6: Erlaubte Angaben für den P-Selektor

5.2.7 Adress-Formate

Beim Erfassen der TRANSPORTADRESSEN und der TSEL-Einträge des LOKALEN NAMENS müssen Sie das Adress-Format *addrform* des Transportsystems übergeben, auf das sich die folgenden Angaben für die Adress-Komponenten bzw. für den T-Selektor beziehen.

In der folgenden Tabelle sind die verschiedenen erlaubten Werte für *addrform*, das zugehörige Transportsystem und alle zugehörigen Adress-Komponenten aufgeführt.

Beachten Sie bitte die Erläuterungen im Abschnitt „Syntax der TNS-Konfigurationsdatei“ auf Seite 29. Dort ist angegeben, welche Adress-Komponenten Sie im Einzelfall angeben müssen.

Im Anschluss an die Tabelle sind die Bedeutung der Adress-Komponenten und ihre Darstellungsformate beschrieben. Die Adress-Komponenten sind in alphabetischer Reihenfolge aufgelistet.

addrform	**Transportsystem für**	**Adress-Komponenten**
EMSNA	Kopplung mit TRANSIT über SNA-Backbone	LU-Name Rechner/Region
LANINET	Rechnerkopplung über TCP/IP	Internet-Adresse oder *HOST* Hostname Portnummer
LOOPSBKA	Interprozess-Kommunikation mit CMX	T-Selektor
RFC1006	Rechnerkopplung über TCP/IP mit RFC1006-Konvergenzprotokoll	Internet-Adresse oder *HOST* Hostname [*PORT* Portnummer] T-Selektor
SDLCSBKA	Stationskopplung an SNA-Netze über SDLC	Rufnummer WAN CC/Leitungskennzeichen
TRSNA	Rechnerkopplung über ein transparentes SNA-Netz	Sym-Dest-Name T-Selektor
WANNEA	Rechnerkopplung im WAN und ISDN mit Protokoll NEATE, NEAN	Stationsname Rechner/Region SNPA-Information WAN CC/Leitungskennzeichen
WANSBKA	Rechnerkopplung im WAN und ISDN mit ISO-Protokollen der Klasse 0 u. 2 (SBKA-Profil)	SNPA-Information oder OSI-NSAP T-Selektor TPI TPC WAN CC/Leitungskennzeichen
WAN3SBKA	heterogene Kopplung im X.25-Netz ohne Transportprotokoll	SNPA-Information T-Selektor WAN CC/Leitungskennzeichen

Tabelle 7: Adress-Formate, Transportsysteme und Adress-Komponenten

Die Adress-Komponenten zur Beschreibung der Netzadresse eines Endsystems (Ethernet-, Internet-Adresse, Hostname, Rechner, Region, Sym-Dest-Name) werden als Zeichenkette aus alphanumerischen Zeichen übergeben.

Die Adress-Komponenten, die die Adresse einer TS-Anwendung innerhalb des eigenen Systems beschreiben (T-Selektor, Stationsname usw.), werden eingeschlossen in Hochkommata als Zeichenkette übergeben. Vor dem T-Selektor ist ein Formatindikator anzugeben (siehe folgender Abschnitt).

5.2.7.1 Adress-Komponenten und ihre Formate

In diesem Abschnitt sind die Adress-Komponenten und ihre Darstellungsformate erläutert, die Sie beim Erfassen von TS-Anwendungen angeben.

Ethernet-Adresse

Geben Sie die Ethernet-Adresse des Endsystems an, in dem sich die ferne TS-Anwendung befindet.

Darstellungsformat:

Geben Sie genau 12 Hexadezimalziffern [0-9,A-F,a-f] an.

Internet-Adresse

Geben Sie die Internet-Adresse des fernen Endsystems an.

Darstellungsformat:

Geben Sie genau 4 Dezimalzahlen zwischen 0 und 255 an. Diese Zahlen müssen mit dem Sonderzeichen . (Punkt) getrennt werden.

Beispiel: 123.0.3.98

Hostname

Geben Sie den Hostnamen (mit dem Schlüsselwort HOST vorangestellt) des fernen Endsystems an.

Darstellungsformat:

Geben Sie maximal 60 Zeichen im ASCII-Format an.

Beispiele: PGTW1339 oder V116.mch.sni.de

LU-Name

Geben Sie für eine TS-Anwendung in einem SNA-System den VTAM-Applikationsnamen der SNA-Anwendung an. Für eine TS-Anwendung in einem TRANSDATA-Rechner, die über ein SNA-System erreichbar ist, geben Sie den Stationsnamen der TS-Anwendung an.

Darstellungsformat:

Der LU-Name ist als String ('LU-Name') zu übergeben. Es ist nur das TRANSDATA-Format zugelassen, das Sie mit dem Formatindikator T angeben. Der Stationsname darf maximal 8 Zeichen lang sein. Kürzere Eingaben werden automatisch durch Leerzeichen ergänzt. Längere Eingaben werden als Fehler abgewiesen.

LU-Nummer

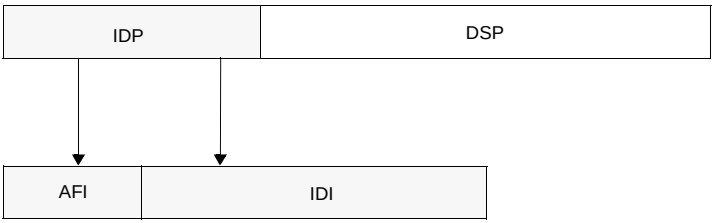
Die LU-Nummer können Sie alternativ zum LU-Namen angeben. Geben Sie die LU-Nummer (Locaddr) an, die in der TRANSIT-Konfiguration für den TRANSIT-Anschluss der TS-Anwendung angegeben ist.

Darstellungsformat:

Geben Sie eine Dezimalzahl zwischen 1 und 255 ein.

OSI-NSAP

Der Aufbau der hier anzugebenden OSI-Netzadressen ist in ISO 8348/Add.2 beschrieben. Er wird hier in Kurzform wiedergegeben:



IDP (Initial Domain Part)

Besteht aus zwei Teilen: dem AFI und dem IDI.

AFI (Authority and Format Identifier)

Die Kennziffer des AFI legt die Struktur und die Länge des IDP fest.

Es werden festgelegt:

- das IDI-Format
- die Institution, die die IDI-Werte festlegt
- welche Füllziffern bei der Codierung des IDP verwendet werden
- die abstrakte Syntax des Domain Specific Part (DSP)

IDI (Initial Domain Identifier)

Der IDI beschreibt den Adressierungsbereich und die Instanz für die Vergabe des DSP.

Es werden festgelegt:

– der Adressierungsbereich, aus dem die DSP-Werte stammen
– die Institution, die für die Vergabe der DSP-Werte in diesem Bereich zuständig ist

DSP (Domain Specific Part)

Der DSP gibt die Möglichkeit einer weiteren Detaillierung an. Die Semantik des DSP wird durch die über den IDI bezeichnete Institution festgelegt. Die abstrakte Syntax wird durch den AFI festgelegt. Die nachstehende Tabelle gibt die minimale/maximale Stellenzahl für den IDP (d. h. 2-stelliger AFI und zugehöriger IDI) und den DSP an. Zu beachten ist, dass für die binäre DSP-Syntax prinzipiell nur gerade DSP-Stellenanzahlen erlaubt sind (auch wenn der Maximalwert nicht erreicht wird!).

IDI-Format	**AFI**	**IDP min.**	**IDP max.**	**DSP-Syntax**	**DSP max.**
X.121	36	3	16	dec	24 Dezimalziffern
X.121	37	3	16	bin	12 x 2 Hexadezimalziffern
X.121	52	3	16	dec	24 Dezimalziffern
X.121	53	3	16	bin	12 x 2 Hexadezimalziffern
ISO_DCC	38	5	5	dec	35 Dezimalziffern
ISO_DCC	39	5	5	bin	17 x 2 Hexadezimalziffern
F.69	40	3	10	dec	30 Dezimalziffern
F.69	41	3	10	bin	15 x 2 Hexadezimalziffern
F.69	54	3	10	dec	30 Dezimalziffern
F.69	55	3	10	bin	15 x 2 Hexadezimalziffern
E.163	42	3	14	dec	26 Dezimalziffern
E.163	43	3	14	bin	13 x 2 Hexadezimalziffern
E.163	56	3	14	dec	26 Dezimalziffern
E.163	57	3	14	bin	13 x 2 Hexadezimalziffern

Tabelle 8: Minimale/maximale Stellenzahl für IDP und DSP
bin = binäre DSP-Syntax, dec = dezimale DSP-Syntax

IDI-Format	AFI	IDP min.	IDP max.	DSP-Syntax	DSP max.
E.164	44	3	17	dec	23 Dezimalziffern
E.164	45	3	17	bin	11 x 2 Hexadezimalziffern
E.164	58	3	17	dec	23 Dezimalziffer
E.164	59	3	17	bin	11 x 2 Hexadezimalziffern
ISO_ICD	46	6	6	dec	34 Dezimalziffern
ISO_ICD	47	6	6	bin	17 x 2 Hexadezimalziffern
Local	48	2	2	dec	38 Dezimalziffern
Local	49	2	2	bin	19 x 2 Hexadezimalziffern
Local	50	2	2	bin	19 x 2 Hexadezimalziffern
Local	51	2	2	bin	19 x 2 Hexadezimalziffern

Tabelle 8: Minimale/maximale Stellenzahl für IDP und DSP
bin = binäre DSP-Syntax, dec = dezimale DSP-Syntax

AFI und IDI werden direkt hintereinander eingetragen. IDP und DSP werden durch '+' voneinander getrennt.

Beispiel: 49+1234569876 oder 38123+556678

Portnummer

Geben Sie die TCP-Portnummer für die TS-Anwendung an.

Darstellungsformat:

Geben Sie eine Dezimalzahl zwischen 1 und 32767 an. Die Portnummer müssen Sie bei LANINET als String mit Formatindikator A (ASCII) übergeben.

Beispiel

A'4712'

Rechner

Geben Sie die Rechnernummer des Kommunikations- oder Verarbeitungsrechners an, in dem die ferne TS-Anwendung residiert.

Darstellungsformat:

Geben Sie bei WANNEA eine Dezimalzahl zwischen 0 und 255 ein, bei EMSNA und STANEA eine Dezimalzahl zwischen 0 und 31. Die Rechnernummer wird zusammen mit der Regionsnummer in der Form Rechnernummer/Regionsnummer (z. B. 7/16) angegeben.

Region

Geben Sie die Regionsnummer des Kommunikations- oder Verarbeitungsrechners an, in dem die ferne TS-Anwendung residiert.

Darstellungsformat:

Geben Sie eine Dezimalzahl zwischen 0 und 255 ein. Die Regionsnummer wird zusammen mit der Rechnernummer in der Form Rechnernummer/Regionsnummer (z. B. 7/16) angegeben.

Rufnummer

Geben Sie die Rufnummer an, unter der Sie das Partnersystem erreichen.

Darstellungsformat:

Geben Sie maximal 17 Dezimalziffern an.

SNPA-Information

Die Abkürzung SNPA steht für Subnetwork Point of Attachment und bezeichnet den Zugangspunkt zu einem Subnetz. Sie geben hier die Rufnummer, DTE-Adresse oder PVC-Nummer an, über die Sie den Partner erreichen.

Darstellungsformat:

Die SNPA-Information enthält eine Anschluss-Nummer, angeführt von einem subnetzspezifischen Schlüsselwort.

- für Rechnerkopplung im WAN oder ISDN mit ISO-Protokollen der Kl. 0/2 (Adress-Format WANSBKA):

  E.164 <ISDN-Nummer>
  20-stellige ISDN-Nummer

  E.163 <Telephonnummer>
  24-stellige Telephonnummer

  X.121 <IDI>
  17-stellige X.25-DTE-Adresse

PVC <PVC-Nummer>
X.25-PVC-Nummer(1 - 4095)

X.31 <ISDN-Nummer> X.121 <IDI>
Zweistufenwahl mit 20-stelliger ISDN-Nummer und 17-stelliger X.25-DTE-Adresse

X.32 <Rufnummer> X.121 <IDI>
Zweistufenwahl mit 24-stelliger Telefonnummer und 17-stelliger X.25-DTE-Adresse

- für heterogene Rechnerkopplung im X.25-Netz ohne Transportprotokoll (Adress-Format WAN3SBKA):

  E.164 <ISDN-Nummer>
  20-stellige ISDN-Nummer

  X.121 <IDI>
  17-stellige X.25-DTE-Adresse

  PVC <PVC-Nummer>
  X.25-PVC-Nummer (1 - 4095)

  X.31 <ISDN-Nummer> X.121 <IDI>
  Zweistufenwahl mit 20-stelliger ISDN-Nummer und 17-stelliger X.25-DTE-Adresse

  X.32 <Rufnummer> X.121 <IDI>
  Zweistufenwahl mit 24-stelliger Telefonnummer und 17-stelliger X.25-DTE-Adresse

Beispiele

```
E.163 08963641625
X.121 123456
PVC 123
```

Stationsname

Geben Sie den Stationsnamen (T-Selektor) aus der NEA-Adresse an. Mit dem Stationsnamen meldet sich die TS-Anwendung im Endsystem, auf dem sie residiert, beim Transportsystem an.

Darstellungsformat:

Der Stationsname ist als String 'Stationsname' mit vorangestelltem Formatindikator zu übergeben. Es sind die Formatindikatoren T, A, E und X zugelassen. Beim T-Selektor des Adress-Formats SDLCSBKA ist jedoch

nur der Formatindikator T erlaubt. Der Formatindikator bestimmt das im String anzugebende Format (siehe Abschnitt „Formatindikatoren“ auf Seite 45).

Der Stationsname darf bei allen Formaten maximal 8 Zeichen lang sein. Das entspricht beim Formatindikator X 16 Hexadezimalziffern.

Kürzere Eingaben werden beim TRANSDATA-Format durch Leerzeichen, ansonsten durch NIL ergänzt. Längere Eingaben werden als Fehler abgewiesen.

Sym-Dest-Name

Geben Sie den Symbolic Destination Name aus der TRANSIT-Konfiguration an. Der Sym-Dest-Name bezeichnet das LU6.2-Programm (LU = Logical Unit) für TRANSIT-LU6.2 auf der Partner-LU.

Darstellungsformat:

Der Sym-Dest-Name ist als Zeichenkette mit genau 8 Zeichen zu übergeben. Die Zeichenkette darf nur Großbuchstaben [A-Z] und Ziffern [0-9] enthalten.

TPI (optional)

Geben Sie bei der TRANSPORTADRESSE mit Adress-Format WANSBKA die Transportprotokoll-Identifikation (TPI) an, wenn diese beim Verbindungsaufbau zu der fernen TS-Anwendung erwartet wird.

Darstellungsformat:

Den TPI müssen Sie als String mit Formatindikator X (Hexadezimalformat) übergeben. Der Wert muss eine gerade Anzahl von Hexadezimalziffern enthalten. Maximal dürfen Sie 32 Hexadezimalziffern angeben.

TPC (optional)

Mit der Eingabe der Transportprotokollklasse (TPC) können Sie die Aushandlung der Transportprotokollklasse gemäß ISO 8073 beim T-CONNECT.request steuern. Wenn Sie kein TPC eingeben, gilt der Standardwert (2/2).

Darstellungsformat:

Für TPC können Sie *2/0, 2/2, 0/0* oder *0/-* angeben. Die Werte bedeuten:

– 2/0 bevorzugt Klasse 2, alternativ Klasse 0
– 2/2 bevorzugt Klasse 2, alternativ Klasse 2
– 0/0 bevorzugte Klasse 0, alternative Klasse 0
– 0/- nur Klasse 0, keine Alternative

T-Selektor

Mit dem T-Selektor meldet sich die TS-Anwendung im Endsystem, auf dem sie residiert, beim Transportsystem an.

Darstellungsformat:

Der T-Selektor muss als String 'T-Selektor' mit vorangestelltem Formatindikator T, A, E, X oder V angegeben werden. Der Formatindikator bestimmt das im String anzugebende Format bzw. dessen Codierung. Im Hexadezimalformat (Formatindikator X) ist eine gerade Anzahl von Ziffern anzugeben, maximal 20 (in der TRANSPORTADRESSE maximal 64). Im ASCII-Format (A) und im EBCDIC-Format (E) dürfen Sie maximal 10 (in der TRANSPORTADRESSE 32) Zeichen angeben. Im TRANSDATA-Format (T) dürfen Sie maximal 8 Zeichen angeben.

Geben Sie den Formatindikator V an, so wird die folgende Angabe für den T-Selektor ignoriert. In das TS-Directory wird ein leerer Eintrag für den T-Selektor aufgenommen.

Formatindikatoren

Die verschiedenen Formatindikatoren haben folgende Bedeutung:

T (TRANSDATA-Format)

Der T-Selektor wird im TRANSDATA-Format für Stationsnamen angegeben. D. h. der String darf nur aus Großbuchstaben, Ziffern und den Sonderzeichen '$', '#' und '@' bestehen, höchstens 8 Zeichen lang sein und nicht mit einer Ziffer beginnen. Der T-Selektor wird dann intern in EBCDIC.DF.03 (Internationale/Deutsche DF-Version 03) abgelegt und mit Leerzeichen auf 8 Stellen ergänzt.

A (ASCII-Zeichenformat)

Jedes eingegebene Zeichen wird im ISO-7-Bit-Code abgelegt. Die Zeichenkette darf maximal 10 (in der TRANSPORTADRESSE 32) oder 8 Zeichen lang sein, je nach Wahl des Transportsystems.

E (EBCDIC-Zeichenformat)

Jedes eingegebene Zeichen wird im EBCDIC-Code EBCDIC.DF.03 (Internationale/Deutsche DF-Version 03) abgelegt. Die Zeichenkette darf maximal 10 (in der TRANSPORTADRESSE 32) bzw. 8 Zeichen lang sein, je nach Wahl des Transportsystems.

X (Hexadezimalformat)

Der T-Selektor wird als Hexadezimalstring übergeben. Der String muss eine gerade Anzahl von Hexadezimalziffern [0-9,A-F,a-f] enthalten. Je ein Ziffernpaar wird als ein Byte (Zeichen) abgelegt, wobei die 1. Ziffer den Wert der höherwertigen und die 2. Ziffer den der niederwertigen Bits beschreibt. X'3a' entspricht zum Beispiel der Bitdarstellung '0011 1010' (höchstwertiges Bit am weitesten links).

V (Leerformat)

Mit diesem Formatindikator können Sie einen Leereintrag erzeugen. Der entsprechende T-Selektor existiert dann, hat aber keinen Wert.

Einen Leereintrag erzeugen Sie durch Angabe von V''. Geben Sie nach V einen nicht leeren String an, dann wird dieser String ignoriert.

WAN CC/Leitungskennzeichen

CCs und Leitungen, die für die Verbindung genutzt werden können.

Darstellungsformat:

Liste von (durch Leerzeichen getrennten) CC-Nummern. Zu jeder CC-Nummer kann optional, durch Doppelpunkt getrennt, eine Liste von Komma-separierten Leitungsnummern angegeben werden. Eine Leitungsnummer kennzeichnet einen Leitungsanschluss auf dem CC.

Zulässig sind die Leitungsnummern 0, 1, 2, 3, 4, 32, 33 und 34 und CC-Nummern von 1 bis 255. Welche Kombinationen sinnvoll sind, hängt von Ihrer Systemkonfiguration ab. Beachten Sie auch die Hinweise in den Freigabemitteilungen. Die Liste wird von dem Schlüsselwort WAN angeführt.

Beispiel: WAN 1:1,2 2:33

5.2.8 Erfassungsregeln für TNS-Dateien

5.2.8.1 Zeichen mit Sonderbedeutung

Außer dem Leerzeichen haben folgende Zeichen eine Sonderbedeutung:

$

Dollar leitet eine INCLUDE-, ORIGIN- oder VERSION-Anweisung ein.

$ muss entwertet werden, wenn $INCLUDE, $ORIGIN oder $VERSION als GLOBALE NAMEN definiert werden.

;

Semikolon leitet einen Kommentar ein, der Rest der laufenden Zeile wird ignoriert.

()

Runde Klammern können verwendet werden, um Felder über Zeilengrenzen hinaus zu einem Eingabesatz zusammenzufassen. Insbesondere können damit Felder an beliebiger Position innerhalb des Eingabesatzes mit einem Kommentar versehen werden (ein Kommentar zeigt i.a. das Zeilenende an). Folgendes Beispiel beschreibt *einen* Eingabesatz zur Angabe der TRANSPORTADRESSE der TS-Anwendung „X.25“.

```
X\.25 (  TA WANSBKA
X.121 45890040033    ; DTE-Adresse
A'dtxp-33-01'        ; T-Selektor
2/0                  ; Transport Protocol Class (TPC)
)
```

\

Gegenschrägstrich dient zur Entwertung der Sonderbedeutung des nachfolgenden Zeichens. Falls das dem \ folgende Zeichen keine Sonderbedeutung hat, wird \ ignoriert.

.

Punkt dient zur Trennung von Namensteilen bei der Angabe des GLOBALEN NAMENS.

’

Apostroph; innerhalb eines von Apostrophen eingeschlossenen Strings ist die Sonderbedeutung der Zeichen $; ( ) . * @ und des Leerzeichens aufgehoben. Die Zeichen repräsentieren dort sich selbst.

Strings für T-, S- und P-Selektoren sind immer mit Apostrophen einzuschließen.

*

Stern ist reserviert für zukünftige Zwecke. Es ist nicht erlaubt, * als einziges Zeichen eines Namensteils des GLOBALEN NAMENS anzugeben. In diesem Fall kann * auch nicht entwertet werden.

@

Kommerzielles AT ist reserviert für zukünftige Zwecke.

Die Sonderbedeutung eines Zeichens wird durch \ (Gegenschrägstrich) oder durch Apostrophierung aufgehoben. Der Zeilentrenner wird ignoriert, falls ihm ein \ vorangeht oder falls er mit ( ) (runden Klammern) entwertet ist.

5.2.8.2 Namen mit gleichen höherwertigen Namensteilen

Wollen Sie Einträge für mehrere TS-Anwendungen erfassen, die zu Blättern an einem Ast des Namensbaums gehören, so müssen Sie die gemeinsamen Namensteile der zugehörigen GLOBALEN NAMEN in den Angaben für *name* nicht laufend wiederholen. Sie können die gemeinsamen Namensteile als *origin* vorgeben. Alle relativ angegebenen Namen in den *name*-Feldern werden dann um . (Punkt) und den Wert von *origin* erweitert. Sie müssen dann in den *name*-Feldern also nur noch die Namensteile angeben, die nicht im *origin* enthalten sind. Der relativ angegebene Wert von *name* muss zusammen mit dem Wert von *origin* einen syntaktisch korrekten GLOBALEN NAMEN bilden.

Ein GLOBALER NAME im Feld *name* ist relativ zu einem *origin*, wenn er *nicht* mit . (Punkt) endet. Endet er mit einem . (Punkt), so ist er absolut (relativ zu ROOT). Ein absolut angegebener GLOBALER NAME wird nicht erweitert, auch wenn Sie einen *origin* vorgeben. Ist kein *origin* vorgegeben, so ist jede Angabe für einen GLOBALEN NAMEN absolut (relativ zu ROOT).

Beispiel

```
Feldinhalt       Wert                resultierender
von name         von origin          GLOBALER NAME
-------------------------------------------------
myappl           myhost.sttz.Mch-P.D
myappl .myhost.  sttz.Mch-P.D
np5.np4          .np2                np5.np4..np2
```

Den Wert für *origin* können Sie wie folgt durch eine Steuerzeile in der Eingabedatei festlegen:

$ORIGIN[␣origin]

Für *origin* geben Sie den Ursprung an, um den alle relativen Namensangaben erweitert werden sollen. *$ORIGIN␣origin* muss der einzige Inhalt des entsprechenden Satzes sein. Wird für *origin* kein Wert angegeben, so werden die GLOBALEN NAMEN der folgenden Eingabesätze nicht erweitert. Eine $ORIGIN-Anweisung ändert die Festlegung für *origin* bei der Kommando-Eingabe im Format *tnsxfrm*.

Die Ursprungsdefinition mit $ORIGIN gilt nur bis zur nächsten $ORIGIN-Anweisung oder bis zum Ende dieser Datei.

5.2.8.3 Eingabedateien verschachteln

Sie können Ihre Eingaben für den TNS in mehrere Dateien aufteilen (z. B. zur produktspezifischen Trennung der TNS-Einträge).

Durch $INCLUDE-Anweisungen können Sie die Dateien ineinander verschachteln. Eine $INCLUDE-Anweisung innerhalb einer Eingabedatei wird durch den Inhalt der angegebenen Datei ersetzt.

Eine $INCLUDE-Anweisung ist ein Satz mit dem einzigen Inhalt:

$INCLUDE␣file

Für *file* ist der Name der Datei anzugeben, die eingefügt werden soll. *file* muss aus Einträgen im Format *tnsxfrm* bestehen. Die Datei *file* darf weitere $INCLUDE-Anweisungen enthalten. Diese dürfen aber keine direkte oder indirekte Rekursion auslösen. Maximal können 10 $INCLUDE-Anweisungen geschachtelt werden.

Die Einstellung des Ursprungs ($ORIGIN-Anweisung) wird an die untergeordnete INCLUDE-Stufe weiter vererbt. Bei Rückkehr in die übergeordnete INCLUDE-Stufe wird der ursprüngliche Wert für den Ursprung dieser Stufe wieder eingestellt.

5.2.8.4 Versionsangabe für Format und Syntax

Eine $VERSION-Anweisung ist ein Satz mit dem einzigen Inhalt:

$VERSION␣version

Für *version* ist die Versionsnummer 6.0 anzugeben.

5.2.8.5 Migration

Dieser Abschnitt ist nur von Interesse, wenn Sie Ihre TNS-Konfiguration von einem Reliant-UNIX-Rechner mit älteren CMX-Versionen in den Solaris-Rechner integrieren wollen.

Die Syntax des Formats *tnsxfrm* hat sich von CMX V3.0 auf V4.0 in einigen Adress-Formaten geändert, von CMX 4.0 auf 5.x gab es jedoch keine inkompatiblen Änderungen. TNS bietet für Dateien, die unter CMX V3.0 erstellt wurden, eine Migration in das Format von CMX V5.x. Diese Migration wird automatisch angestoßen. TNS erkennt die Version des Dateiformats anhand des $VERSION-Satzes (siehe Abschnitt „Versionsangabe für Format und Syntax“

oben). Nachfolgend die Gegenüberstellung der betroffenen Adress-Formate in der CMX V3.0- und der CMX V5.x-Syntax. Von CMX V4.0 zu CMX V5.x ist keine Migration nötig.

Wenn Sie mit RFC1006 arbeiten, beachten Sie auch die Hinweise im Kapitel „Verbindungen über RFC1006 konfigurieren“ auf Seite 57.

```
3.0: TA ISDNSBKA idi tsel [tpi] [tpc]
5.0: TA WANSBKA E.164 idi tsel [tpi] [tpc]

3.0: TA WANSBKA idi tsel [tpi] [tpc]
5.0: TA WANSBKA X.121 idi tsel [tpi] [tpc]

3.0: TA WAN3SBKA idi
5.0: TA WAN3SBKA X.121 idi

3.0: TA ISDNSBKA LNR line tsel [tpi] [tpc] CC Wijk
5.0: TA WANSBKA tsel [tpi] [tpc] WAN i:line j:line k:line

3.0: TA WANSBKA PVC pvcNumber LNR line tsel [tpi] [tpc] CC Wijk
5.0: TA WANSBKA PVC pvcNumber tsel [tpi] [tpc] WAN i:line j:line
k:line

3.0: TA WANSBKA LNR line tsel [tpi] [tpc] CC Wijk
5.0: TA WANSBKA tsel [tpi] [tpc] WAN i:line j:line k:line

3.0: TA WAN3SBKA PVC pvcNumber LNR line CC Wijk
5.0: TA WAN3SBKA PVC pvcNumber WAN i:line j:line k:line

3.0: (MSA ISDNSBKA idi1
     TA WANSBKA idi2 tsel)
5.0: TA WANSBKA X.31 idi1 X.121 idi2 tsel
```

5.2.9 TS-Directory

Mit dem Kommando *tnsxcom* können Sie Dateien des Formates *tnsxfrm* in TS-Directories überführen. Dabei können Sie verschiedene Modi einstellen für Funktionen wie Syntaxprüfung, Aktualisierung oder Neuerstellung des TS-Directories. Das Kommando hat folgende Syntax:

tnsxcom[␣**-d**␣num][␣**-l**␣-**s**␣-**S**␣-**u**␣-**i**][␣file ...]

Die Optionen haben folgende Bedeutung:

-d␣num

Nummer des TS-Directory, das bearbeitet werden soll. Sie können die Nummern 1 bis 9 angeben. Ohne Angabe wird 1 (entspricht DIR1) eingestellt.

-l

LOAD-Modus

tnsxcom nimmt die Einträge einzeln aus der Datei *file* und füllt das (bisher leere) TS-Directory mit den syntaktisch korrekten Einträgen.

-s

CHECK-Modus

tnsxcom wendet nur die Syntaxprüfung auf die Datei *file* an und protokolliert mögliche Syntaxfehler. Das TS-Directory wird nicht verändert.

-S

CHECK_UPD-Modus

Wie bei Option *s* erfolgt in einem ersten Lauf zuerst die Syntaxprüfung auf die gesamte Datei *file*. Treten in *file* keine Syntaxfehler auf, so aktualisiert *tnsxcom* dann das TS-Directory in einem zweiten Lauf.

-u

UPDATE-Modus

tnsxcom nimmt die Einträge einzeln aus der editierbaren Datei *file* und mischt die syntaktisch richtigen Einträge in das TS-Directory durch Erfassung bisher nicht vorhandener oder Aktualisierung existierender Einträge (Option *u* ist der Standardwert von *option*).

-i

INTERAKTIV-Modus

tnsxcom liest Einträge im Format *tnsxfrm* von *stdin*, nachdem er durch Ausgabe eines Promptzeichens seine Eingabebereitschaft angezeigt hat, und mischt sie in das TS-Directory. Bisher im TS-Directory noch nicht vorhandene Einträge werden eingefügt, existierende Einträge werden aktualisiert.

file ...

Name der Datei mit Einträgen im Format *tnsxfrm*, die im Fall *option* = *l*, *s*, *S* oder *u* von *tnsxcom* ausgewertet werden soll. Es können mehrere Dateien angegeben werden.

Im Fall *option* = *d* ist der Name der Datei anzugeben, in die *tnsxcom* den Inhalt des TS-Directory aufbereiten soll.

Beispiel

Der folgende Aufruf überführt die Einträge aus der Datei *input.dir* in das bisher leere TS-Directory 2:

```
tnsxcom -d 2 -l input.dir
```

5.2.9.1 Eintrag für TS-Anwendung aus dem TS-Directory löschen

Wollen Sie den gesamten Eintrag einer TS-Anwendung aus dem TS-Directory löschen, so übergeben Sie dem *tnsxcom* folgenden Satz.

```
[name␣]DEL
```

Der *tnsxcom* löscht dann alle Eigenschaften, die der TS-Anwendung *name* zugeordnet sind, und den GLOBALEN NAMEN aus dem TS-Directory.

5.2.9.2 Eigenschaften einer TS-Anwendung anzeigen

Wenn Sie Ihre Eingaben für den *tnsxcom* interaktiv vornehmen, dann können Sie sich die aktuell im TS-Directory eingetragenen Eigenschaften einer TS-Anwendung am Bildschirm anzeigen lassen. Dazu übergeben Sie einen Satz mit folgendem Format:

```
[name␣]DISP
```

Damit können Sie sich z. B. die zuvor erfassten oder geänderten Einträge einer TS-Anwendung zur Kontrolle anzeigen lassen.

Geben Sie einen Satz mit dem obigen Format in einer Datei an, die Sie dann an den *tnsxcom* übergeben, so wird beim Compilieren eine Warnung ausgegeben, und der Eintrag wird vom *tnsxcom* ignoriert.

5.2.9.3 Angabe des TS-Directory

Sie können mit Hilfe eines Eingabesatzes in der Datei in ein anderes TS-Directory umschalten. Der Satz muss folgendes Format haben:

```
DIR␣n
```

Für *n* ist die Nummer des TS-Directory anzugeben, in das umgeschaltet werden soll.

Die folgenden Sätze beziehen sich dann auf dieses TS-Directory. Es wird solange bearbeitet, bis explizit in ein anderes TS-Directory umgeschaltet oder die Eingabe beendet wird.

5.2.9.4 Beispiel für tnsxcom-Einträge

Folgende Beispieldatei soll die Syntax von *tnsxcom* verdeutlichen:

```
; RFC1006-Transportadresse einer über IP-Adresse 10.25.1.27
; erreichbaren Anwendung mit T-Selektor im TRANSDATA-Format
;
; name   type data
rfcanw01 TA   RFC1006 10.25.1.27 PORT 102 T'RFCANW01'
;
; Der NEA-Partner $DIALOG im Rechner 1/18,
;
;name   type data
wan1anw TA   WANNEA T'$DIALOG' 1/18
;
; Zwei Anwendungen, die über Interprozeßkommunikation
; miteinander kommunizieren
;
;name   type data
ipclok  TA   LOOPSBKA A'IPC-LOK'
        TSEL LOOPSBKA A'IPC-LOK'
ipcrem  TA   LOOPSBKA A'IPC-REM'
        TSEL LOOPSBKA A'IPC-REM'
;
;Transportadresse eines WAN-Partners, der über OSI-
;Transportprotokoll und über DTE-Adresse 123456 erreicht wird.
;
;name   type data
wananw01 TA   WANSBKA X.121 123456 A'ANW01'
```

6 Transport Provider Selection

Die Transport Provider Selection ist notwendig, um bei Kopplung über XTI/TLI aus Adressen, die von TNSX geliefert wurden, den Transport Provider zu ermitteln und die Adressen für ihn umzuformen. Dazu stellt Transport Provider Selection einem Anwendungsprozess eine einfache konsistente Schnittstelle in Form einer Datei zur Verfügung.

Die von Transport Provider Selection zur Verfügung gestellte Standard-Schnittstelle kennt die vorhandenen Transport Provider. Diese Kenntnis erlaubt den Anwendungen eines Benutzers durch entsprechende Wahl des Netzwerks zugeordnete Transport Provider zu verwenden. Dadurch sind die Anwendungen unabhängig von Protokollen und Providern.

In diesem Kapitel ist beschrieben, wie

– der Systemverwalter die Konfigurationsdatei für Transport Provider vewaltet und pflegt,

– der Benutzer einen Transport Provider für seine Anwendungen wählt.

6.1 Transport Provider Selection verwalten und pflegen

Der zentrale Teil der Transport Provider Selection ist die Netzwerk-Konfigurationsdatei */opt/lib/cmx/cmxti.cfg*. In dieser Datei sind für einen bestimmten Rechner die unterstützten Netzwerke und die damit verknüpften Transport Provider eingetragen. Der Systemverwalter kann neue Einträge hinzufügen, indem er die Datei */opt/lib/cmx/cmxti.cfg* editiert.

Die für Transport Provider Selection notwendigen Einträge in der Datei *cmxticfg* liegen in folgenden Feldern vor:

Network	Transport Provider	cmxticfg access library	Options

Network
: bezeichnet eine Adress-Form (*addrform* im Abschnitt „Konfigurieren mit tnsxcom“ auf Seite 27), die durch die Transport Provider Selection unterstützt wird (z. B. WANSBKA).

Transport Provider
: bezeichnet den Zugang zu einem XTI/TLI Transport Provider. Im Allgemeinen ist es der Pfadname eines „special file“ (z. B. `/dev/ticots`).

cmxticfg access library
: bezeichnet den Pfad und den Namen der *cmxticfg*-Bibliothek. Diese Bibliothek ist ein gemeinsam benutzbarer Modul wie beispielsweise */opt/lib/cmx/loopsbka.so*.

Options
: Plattformspezifische Optionen. Siehe Readme-Datei.

Kommentarzeilen beginnen mit einem ’#’ in der ersten Spalte der Zeile.

Für die unterstützten Netzwerke, mit denen der Rechner verbunden ist, sind durch Voreinstellungen zulässige Einträge definiert.

Example

```
#
# The following entry may not be removed
# It is necessary for redirection
       LOOPSBKA     /dev/ticots     /opt/lib/cmx/loopsbka.so
```

Weitere voreingestellte Werte entnehmen Sie der Freigabemitteilung bzw. der Datei selbst.

Der Netz-Eintrag LOOPSBKA darf *nicht* geändert oder gelöscht werden; er wird für die Verbindungsumlenkung benötigt.

6.2 Einsatz der Transport Provider Selection

Der Benutzer wählt für eine Anwendung indirekt einen Transport Provider durch die Angabe einer TRANSPORTADRESSE bzw. des LOKALEN NAMEN (z. B. WANSBKA). Die Anwendung ist somit protokoll- und provider-unabhängig. Das Wählen zwischen den verfügbaren Netzwerken liegt in der Verantwortung der Anwendung.

Es werden die Routinen zur Adress-Umsetzung zum benötigten Zeitpunkt (aktiver/passiver Verbindungsaufbau und beim Anmelden beim Transport Provider) an die Anwendung gebunden.

7 Verbindungen über RFC1006 konfigurieren

Der Internet *Request for Comments* RFC1006 wurde durch das Internet Architecture Board (IAB) herausgegeben, um oberhalb von TCP/IP einen OSI-Transportdienst gemäß ISO 8072 zu definieren.

Das Transmission Control Protocol (TCP) ist verbindungsorientiert, d. h., vor der eigentlichen Datenübertragung wird eine logische Verbindung aufgebaut. TCP sorgt dafür, dass Daten in der richtigen Reihenfolge zur Anwendung gelangen und dass keine Daten verlorengehen oder verfälscht werden.

Beim Transportdienst gemäß ISO 8072 werden Nachrichten durch Endemarken voneinander abgegrenzt; der Datenfluss ist nachrichtenorientiert. Beim TCP gibt es solche Endemarken nicht; ein kontinuierlicher Datenstrom wird übertragen. TCP-Anwendungen können nur anhand des Dateninhalts erkennen, wo eine logische Nachricht zu Ende ist und die nächste beginnt.

Ein weiterer Unterschied zwischen TCP/IP und dem ISO-Transportdienst besteht beim Verbindungsabbau: TCP garantiert, dass beim Verbindungsabbau alle vorher gesendeten Daten den Empfänger erreichen (orderly release). Gemäß ISO 8073 liegt die Verantwortung dafür, dass Verbindungen erst nach erfolgtem Datenaustausch beendet werden (abortive release), nicht beim Transportsystem, sondern bei den Anwendungen selbst.

7.1 Übersicht der Konfigurationsdaten

Die folgende Abbildung gibt einen Überblick der Konfigurationsdaten, die für den Verbindungsaufbau über RFC1006 von Bedeutung sind.

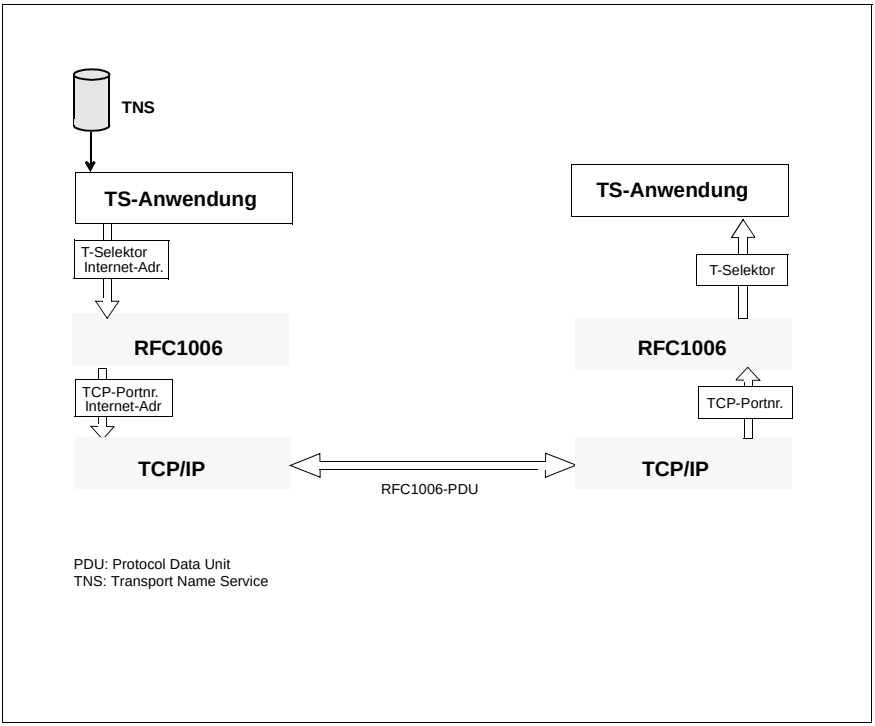


Bild 6: Verbindungsaufbau über RFC1006 - Überblick

Eine Transportverbindung über RFC1006 ist eindeutig einer TCP-Verbindung zugeordnet. Zum Aufbau der TCP-Verbindung werden Internet-Adresse und Portnummer benötigt. Das Paar Internet-Adresse und Portnummer wird im Folgenden als TCP-Adresse bezeichnet. Eine TCP-Adresse identifiziert eindeutig einen TCP-Verbindungsendpunkt. Somit wird eine TCP-Verbindung eindeutig durch ein Paar von TCP-Adressen, nämlich des lokalen und fernen Endpunkts, identifiziert. Eine TS-Anwendung meldet sich mit einem sogenannten Transport-Selektor, kurz T-Selektor genannt, beim RFC1006 an. Sie wird auf der Protokollebene des RFC1006 über diesen T-Selektor adressiert.

7.1.1 Konfigurationsdaten für lokale TS-Anwendungen

Um eine Transport-Verbindung über RFC1006 aufzubauen, definieren Sie im Regelfall eine Transportsystem-Anwendung auf Ihrem eigenen System (im Folgenden lokale TS-Anwendung genannt) und eine weitere für jede Partner-Anwendung, die Sie erreichen wollen (im Folgenden ferne TS-Anwendung genannt).

Um eine lokale TS-Anwendung im TNS zu konfigurieren, müssen Sie im Regelfall folgende Daten angeben:

- Einen GLOBALEN NAMEN, mit dem die Anwendung im TNS identifiziert werden kann. Zu Struktur und Merkmalen des GLOBALEN NAMENS siehe Abschnitt „GLOBALER NAME“ auf Seite 29.
- Das Adress-Format über das die Verbindung aufgebaut werden soll. Der RFC1006 wird über die Formate *RFC1006* oder *LANINET* identifiziert.
- Beim Adress-Format LANINET eine Portnummer, für die der RFC1006 gegebenenfalls einen TCP-Listener für den passiven Verbindungsaufbau einrichtet.

  Beispiel:

  ```
  TSEL LANINET A'1100'
  ```

  Wenn sich die TS-Anwendung für passiven Verbindungsaufbau anmeldet, richtet der RFC1006 gegebenenfalls einen TCP-Listener mit Portnummer 1100 ein.

  Diese Angabe wird nur dann benötigt, wenn ein Partnersystem **nicht** die RFC1006-Standard-TCP-Portnummer 102 als Endpunkt der TCP-Verbindung adressiert. Das ist zum Beispiel bei RFC1006-Implementierungen älterer CMX-Versionen als CMX V5.0[1] der Fall, und in *open*FT-Anwendungen, die auf Rechnern anderer Hersteller ablaufen.
- Beim Adress-Format RFC1006 den T-Selektor,
  - über den ferne TS-Anwendungen die lokale TS-Anwendung, die für passiven Verbindungsaufbau angemeldet ist, adressieren und
  - der beim aktiven Verbindungsaufbau dem Partnersystem als Adressinformation im RFC1006-Protokoll in Form des Parameters *calling TSAP-ID* mitgeteilt wird.

[1] oder auch bei CMX (UNIX)

Zum Beispiel verwenden *open*FT-Anwendungen für den passiven Verbindungsaufbau den T-Selektor *T'$FJAM'* und für den aktiven Verbindungsaufbau die T-Selektoren *T'$FJAM001'*, *T'$FJAM002'*, usw.

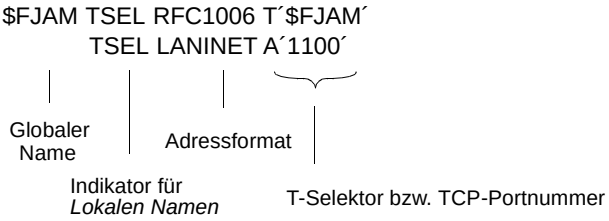
Sie können die Adress-Formate LANINET und RFC1006 jeweils allein oder beide zusammen angeben.

Wenn Sie nur das Adress-Format LANINET angegeben haben, handelt es sich um eine sogenannte reine LANINET-Anwendung. Wenn eine solche TS-Anwendung für passiven Verbindungsaufbau angemeldet ist, beansprucht sie einen eigenen TCP-Listener, d.h. keine andere TS-Anwendung kann sich mit dieser Portnummer für passiven Verbindungsaufbau anmelden. Umgekehrt scheitert die Anmeldung der reinen LANINET-Anwendung, falls bereits ein TCP-Listener mit dieser Portnummer existiert. Beim aktiven Verbindungsaufbau einer reinen LANINET-Anwendung wird die angegebene Portnummer dem Partnersystem als Adressinformation über das RFC1006-Protokoll in Form des Parameters *calling TSAP-ID* mitgeteilt.

Wenn Sie beide Adress-Formate angegeben haben, können sich mehrere TS-Anwendungen denselben TCP-Listener und die dazugehörige Portnummer teilen. Der mit dem Adress-Format RFC1006 angegebene T-Selektor muss jedoch eindeutig sein und kann nur genau einer für passiven Verbindungsaufbau angemeldeten TS-Anwendung angehören. Sowohl nur mit Adress-Format RFC1006 als auch für beide Adress-Formate angemeldete TS-Anwendungen können Verbindungsanforderungen über den TCP-Listener empfangen.

T-Selektor und Portnummer mit den entsprechenden Adress-Format-Angaben stehen bei lokalen TS-Anwendungen im LOKALEN NAMEN, der mit dem Indikator TSEL vor jeder Adress-Format-Angabe gekennzeichnet wird. Der LOKALE NAME ist einem GLOBALEN NAMEN zugeordnet.

Beispiel-Eintrag für eine lokale TS-Anwendung



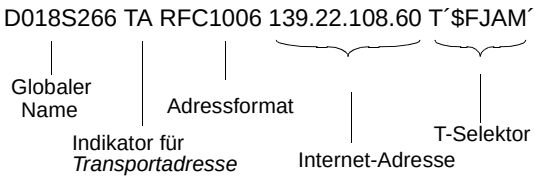
7.1.2 Konfigurationsdaten für ferne TS-Anwendungen

Um eine ferne TS-Anwendung im TNS zu konfigurieren, müssen Sie folgende Daten angeben:

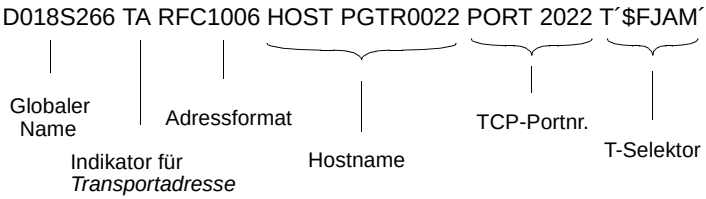
- Den GLOBALEN NAMEN, mit dem die Anwendung im TNS identifiziert werden kann. Zu Struktur und Merkmalen des GLOBALEN NAMENS siehe Abschnitt „GLOBALER NAME“ auf Seite 29.
- Die Internet-Adresse des fernen Systems (oder alternativ den Hostnamen).
- Das Adress-Format, das den Transport Service Provider identifiziert, über den das ferne System erreicht werden soll. Der RFC1006 wird über die Formate *RFC1006* oder *LANINET* identifiziert.
- Beim Adress-Format RFC1006
  - den T-Selektor, der zur Identifizierung der TS-Anwendung im fernen System dient und der dem Partnersystem als Adressinformation über das RFC1006-Protokoll in Form des Parameters *called TSAP-ID* mitgeteilt wird
  - optional eine Portnummer, die den fernen Endpunkt der TCP-Verbindung identifiziert. Ohne die Portnummer-Angabe wird Portnummer 102 adressiert.
- Beim Adress-Format LANINET die Portnummer, die den fernen Endpunkt der TCP-Verbindung identifiziert und die zusätzlich als Adressinformation im RFC1006-Protokoll in Form des Parameters *called TSAP-ID* übertragen wird.

Internet-Adresse oder Hostname, Format, evtl. die TCP-Portnummer sowie der T-Selektor stehen bei fernen TS-Anwendungen in der TRANSPORTADRESSE, die mit dem Indikator TA gekennzeichnet wird. Die TRANSPORTADRESSE ist einem GLOBALEN NAMEN zugeordnet.

Beispiel-Einträge für eine ferne TS-Anwendung



Im obigen Beispiel wird die Anwendung über die Standardportnummer 102 adressiert.



7.2 Verbindung zu fernem Partnersystem aufbauen

In diesem Abschnitt finden Sie Informationen zu den Abläufen beim Verbindungsaufbau und zur Vorgehensweise bei der Konfigurierung am Beispiel eines Verbindungsaufbaus zwischen zwei Systemen. Die Abläufe im aktiven und passiven System werden jeweils aus der Sicht des betreffenden Systems in einem eigenen Abschnitt beschrieben.

7.2.1 Aktiver Verbindungsaufbau

Zunächst ermittelt die CMX-Anwendung über den TNS ihren LOKALEN NAMEN und meldet sich mit diesem für den aktiven Verbindungsaufbau bei CMX an. Danach liest die CMX-Anwendung die TRANSPORTADRESSE (Internet-Adresse oder Hostname, T-Selektor) der Partner-CMX-Anwendung aus der TNS-Datenbasis. Ist in der TNS-Datenbasis der Hostname angegeben, so wird dieser dynamisch beim Abruf der Adresse in die Internet-Adresse übersetzt.

Die CMX-Anwendung übergibt diese Information an RFC1006. Dieser baut mittels Internet-Adresse und Standard-Portnummer 102 eine TCP-Verbindung zum RFC1006-TCP im Partnersystem auf. Über diese sendet dann der RFC1006 den T-Selektor der lokalen CMX-Anwendung in Form des Parameters *calling TSAP-ID* und den T-Selektor der fernen CMX-Anwendung in Form des Parameters *called TSAP-ID* in einem RFC1006-Protokoll-Element (CR TPDU) an den fernen RFC1006. Der weitere Ablauf ist im folgenden Abschnitt aus der Sicht des passiven Systems beschrieben.

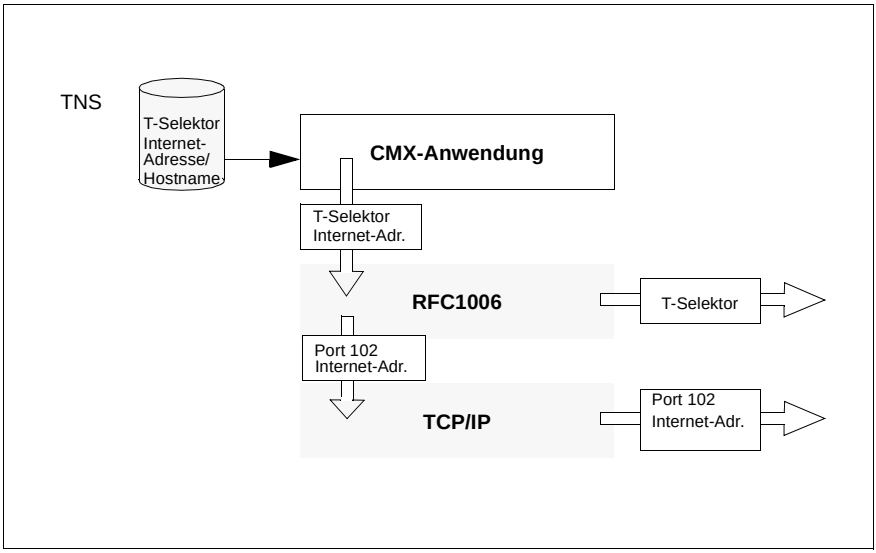


Bild 7: Aktiver Verbindungsaufbau zwischen Systemen mit CMX

Angenommen, das ferne System mit CMX hätte den Hostnamen PGRTV009 und die Internet-Adresse 76.3.13.11, dann wäre für die ferne Anwendung einer der beiden folgenden Einträge erforderlich:

```
rloginrem TA RFC1006 HOST PGRTV009 T'rlogin'
oder
rloginrem TA RFC1006 76.3.13.11 T'rlogin'
```

7.2.2 Passiver Verbindungsaufbau

Die CMX-Anwendung liest vor dem Verbindungsaufbau ihren LOKALEN NAMEN aus dem TNS und meldet sich mit diesem für passiven Verbindungsaufbau bei CMX an.

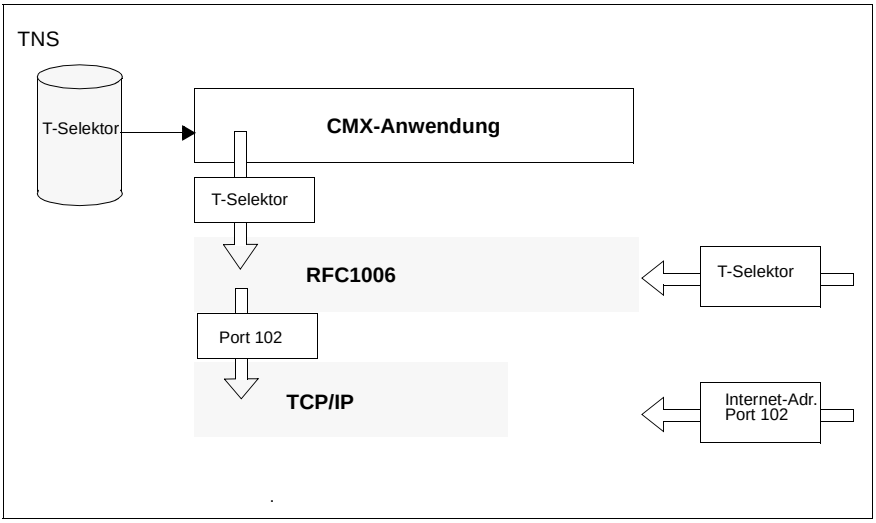


Bild 8: Passiver Verbindungsaufbau zu Partnersystemen mit CMX

Ausgelöst durch die Initiative der Partneranwendung erhält der RFC1006 über den obengenannten TCP-Listener eine Verbindungsanforderung mit der Quell-Adresse (Internet-Adresse und Portnummer) des fernen TCP-Verbindungsendpunkts und der Zieladresse . Nachdem diese TCP-Verbindung etabliert ist, empfängt der RFC1006 über diese ein RFC1006-Protokoll-Element (CR TPDU). Dieses Protokoll-Element enthält in Form des Parameters *calling TSAP-ID* den T-Selektor der rufenden CMX-Anwendung und in Form des Parameters *called TSAP-ID* den T-Selektor der gerufenen CMX-Anwendung, nämlich der angemeldeten lokalen CMX-Anwendung. Dieser CMX-Anwendung wird dann eine Verbindungsanforderung zugestellt und dabei die Internet-Adresse des fernen TCP-Verbindungsendpunkts und der T-Selektor der rufenden CMX-Anwendung in Form der TRANSPORTADRESSE mitgeteilt. Die lokale CMX-Anwendung hat dann die Möglichkeit über den TNS mittels dieser TRANS-PORTADRESSE den GLOBALEN NAMEN der Partner-CMX-Anwendung zu ermitteln.

Ihre Partneranwendung muss im TNS in folgender Weise konfiguriert werden (lokale Anwendung, für Verbindungsaufbau ankommend von der Partneran-wendung):

```
rloginlocal TSEL RFC1006 T´rlogin´
```

8 Administration und Wartung

Dieses Kapitel enthält die Beschreibung der Wartungs- und Administrationsfunktionen, wie sie über die Kommandoschnittstelle aktiviert werden können. Sie finden die CMX-Kommandos in alphabetischer Reihenfolge.

Übersicht der Kommandos

CMX-spezifische Kommandos

cmxdec
: CMX-Meldungen decodieren, siehe Abschnitt „CMX-Meldungen decodieren (cmxdec)“ auf Seite 67.

TNS-spezifische Kommandos

tnsxchk
: Zustand eines TS-Directories prüfen, siehe Abschnitt „TS-Directory prüfen (tnsxchk)“ auf Seite 89.

tnsxcom
: TS-Directory-Einträge hinzufügen, ändern und löschen, siehe Abschnitt „TS-Directory erstellen, aktualisieren, lesen (tnsxcom)“ auf Seite 91.

tnsxd
: TNS-Dämon starten, siehe Abschnitt „Transport Name Service Daemon starten (tnsxd)“ auf Seite 95.

tnsxinfo
: Informationen über ein TS-Directory anzeigen, siehe Abschnitt „Informationen zum TS-Directory anzeigen (tnsxinfo)“ auf Seite 97.

tnsxprop
: Einträge eines TS-Directories anzeigen, siehe Abschnitt „Eigenschaften von TS-Anwendungen ausgeben (tnsxprop)“ auf Seite 104.

Trace-Kommandos

cmxl
: CMX-Bibliotheks-Trace steuern und aufbereiten, siehe Abschnitt „CMX-Bibliotheks-Trace steuern und aufbereiten (cmxl)“ auf Seite 71.

neal

NEA-Bibliotheks-Trace steuern und aufbereiten, siehe Abschnitt „NEABX-Bibliotheks-Trace steuern und aufbereiten (neal)“ auf Seite 85.

tnsxt

TNS-Trace-Information sicherstellen und aufbereiten, siehe Abschnitt „Sicherstellen und Aufbereiten der Trace-Information (tnsxt)“ auf Seite 107.

Die einzelnen Kommandos werden nachfolgend in alphabetischer Reihenfolge beschrieben.

8.1 CMX-Meldungen decodieren (cmxdec)

Mit dem Programm *cmxdec* können Sie ICMX- und XTI-Meldungen decodieren. Dabei handelt es sich um folgende Meldungsarten:

– Fehlermeldungen, die in den Include-Dateien <*cmx.h*>, <*neabx.h*> und <*tnsx.h*> der ICMX bzw. in der Include-Datei <*xti.h*> von XTI definiert sind. Diese Meldungen werden an den Programmschnittstellen ICMX(L), ICMX(NEA) und XTI erzeugt.

– Meldungen, die aus fehlerhaft abgelaufenen Systemaufrufen von ICMX oder XTI resultieren, d. h. Fehlermeldungen, die in der Datei <*errno.h*> definiert sind.

ICMX- und XTI-Fehlermeldungen, Systemmeldungen und Verbindungsabbaugründe werden i.a. in Form eines numerischen Codes, Dezimalzahl oder Hexadezimalzahl mit führendem „0x“, ausgegeben. Eine Fehlermeldung, die an der Programmschnittstelle zum TNS erzeugt wurde, ist nur eindeutig interpretierbar, wenn je ein Wert für den Fehlertyp, die Fehlerklasse und den Fehlerwert angegeben wird. Aus diesem Grund wird ein entsprechender Fehlercode in Form von drei Dezimalzahlen ausgegeben. Die Werte dieser Codes können negativ sein.

Ein Fehlercode bzw. der Code für einen Verbindungsabbau (reason) wird von *cmxdec* entschlüsselt, wenn Sie den Typ der Meldung (XTI-Fehlermeldung, ICMX(L)-Fehlermeldung usw.) und den angegebenen Code des Fehlers an *cmxdec* übergeben. *cmxdec* gibt dann den symbolischen Wert, der in der entsprechenden Include-Datei definiert ist, auf der Standardfehlerausgabe aus.

Bei entsprechender Umgebung des Aufrufers (Variable LANG) liefert *cmxdec* erläuternde Texte zu den Meldungen. Die Texte sind sprachabhängig und optional. Die Messagekataloge in deutscher und englischer Sprache werden immer zusammen mit CMX ausgeliefert.

cmxdec[␣**-c**][␣**-d**][␣**-n**][␣**-s**][␣**-t**][␣**-x**]␣code ...

Die Optionen geben den Typ der in *code* angegebenen Meldung an. Voreinstellung ist *-c*. *cmxdec* versteht die im Folgenden beschriebenen Optionen. Sie schließen sich gegenseitig aus.

-c

Der in *code* angegebene Wert ist eine ICMX-Fehlermeldung, wie sie von *t_error()* an der Schnittstelle ICMX(L) geliefert wird.

Ausgabeformat:

```
ICMX(L) Fehlerdecodierung (6.0)
CODE 0x%x = %d (TYP %d KLASSE %d WERT %d)
      symbolischer Wert und Erklärung für TYP
      symbolischer Wert und Erklärung für KLASSE
      symbolischer Wert und Erklärung für WERT
```

Bezeichnet WERT dabei eine Systemfehlermeldung, so wird in der letzten Zeile in der Regel statt eines symbolischen Wertes der Zahlenwert eingetragen. Die Erklärung erfolgt dann in Englisch.

-d

Der in *code* angegebene Wert ist ein Verbindungsabbaugrund, wie er von *t_disin()* (ICMX(L)), *x_disin()* (ICMX(NEA)) oder *t_rcvdis()* (XTI) geliefert wird. Beachten Sie, dass für XTI/Internet (XTI über TCP/IP) der Abbaugrund als Systemfehler, definiert in <*errno.h*>, decodiert wird.

Ausgabeformat:

```
CMX Reasondecodierung (6.0)
REASON 0x%x = %d:
ICMX(L):
      symbolischer Wert und Erklärung
XTI/Internet:
      symbolischer Wert und Erklärung
XTI/ISO:
      symbolischer Wert und Erklärung
```

-n

Der in *code* angegebene Wert ist eine NEABX-Fehlermeldung, wie sie von *x_error()* an der Schnittstelle ICMX(NEA) geliefert wird.

Ausgabeformat:

```
ICMX(NEA) Fehlerdecodierung (6.0)
CODE 0x%x = %d (TYP %d KLASSE %d WERT %d)
      symbolischer Wert und Erklärung für TYP
      symbolischer Wert und Erklärung für KLASSE
      symbolischer Wert und Erklärung für WERT
```

-s

Der in *code* angegebene Wert ist eine Systemmeldung, wie sie von Systemaufrufen geliefert wird.

Ausgabeformat:

```
CMX Systemfehlerdecodierung (6.0)
CODE 0x%x = %d
      Erklärung in Englisch
```

-t

Die drei in *code* angegebenen numerischen Werte sind eine TNS-Fehlermeldung, wie sie von den TNS-Aufrufen im Standardkopf geliefert wird.

Ausgabeformat:

```
ICMX(TNS) Fehlerdecodierung
TYP %x=%d KLASSE =x%x=%d WERT 0x%x=%d)
      symbolischer Wert und Erklärung für TYP
      symbolischer Wert und Erklärung für KLASSE
      symbolischer Wert und Erklärung für WERT
```

-x

Der in *code* angegebene Wert ist eine XTI-Fehlermeldung, wie sie von *t_error()* geliefert wird.

Ausgabeformat:

```
XTI Fehlerdecodierung
CODE 0x%x = %d
      symbolischer Wert und Erklärung
```

code

Für *code* ist der von ICMX, NEABX, TNS bzw. XTI gelieferte numerische Fehlercode, der Code einer Systemmeldung oder der Code eines Verbindungsabbaugrunds anzugeben, den *cmxdec* decodieren soll. Bei den Optionen *c, d, n, s, x* müssen Sie für *code* eine Dezimalzahl oder eine Hexadezimalzahl mit führendem „0x“ oder „0X“ angeben. Bei Option *t* müssen Sie für *code* drei vorzeichenbehaftete Dezimalzahlen oder Hexadezimalzahlen mit führendem „0x“ oder „0X“ angeben.

Ein vorgegebener numerischer Wert hat für ICMX(L) und XTI unterschiedliche Bedeutung.

TYP, KLASSE und WERT entsprechen den bei ICMX, NEABX und TNS verwendeten Klassifizierungen der Fehlercodes, REASON ist der Grund für einen Verbindungsabbau. Der numerischen Angabe folgt die symbolische Definition gemäß den Include-Dateien <*cmx.h*>, <*neabx.h*>, <*xti.h*> und <*tnsx.h*>. Daran schließt sich der erklärende Text zu dieser Meldung an.

Wird für *code* ein ungültiger oder nicht definierter Wert angegeben, so entschlüsselt *cmxdec* diesen Wert möglichst weitgehend in Bezug auf die symbolische Darstellung des Verbindungsabbaugrundes oder auf Typ, Klasse und Wert der Fehlermeldung. Als erklärenden Text zum Wert von *code* wird von *cmxdec* die Meldung *Nicht decodierbar* ausgegeben.

8.2 CMX-Bibliotheks-Trace steuern und aufbereiten (cmxl)

Der Trace der CMX-Bibliothek wird über die Umgebungsvariable CMXTRACE aktiviert und gesteuert. Die Trace-Einträge eines Prozesses werden kompakt und binär in einem dynamisch angelegten Puffer gesammelt und periodisch in temporäre Dateien geschrieben. Die Aufbereitung dieser Dateien erfolgt entkoppelt durch *cmxl*.

Für Multithreading (MT) gelten einige Abweichungen, auf die im Text hingewiesen wird.

Steuerung des Trace - CMXTRACE

Jeder CMX-Aufruf *t_attach* eines Prozesses wertet die Umgebungsvariable CMXTRACE aus und aktiviert gegebenenfalls den Trace. CMXTRACE muss vor dem Starten der Anwendung, d. h. vor dem ersten *t_attach* des zu überwachenden Prozesses, gesetzt worden sein. Nach dem Aktivieren des Trace wird die temporäre Datei *CMXLa<pid>* mit der Prozess-Identifikation *<pid>* eröffnet, falls sie nicht bereits eröffnet ist. Für die Dateien werden die Zugriffsrechte `rw-------` (0600) vergeben. Anschließend wird dynamisch Speicher für die Pufferung der Trace-Einträge belegt.

Beim Multithreading werden die Trace-Einträge aller Threads eines Prozesses in die temporäre Datei *CMXLa<pid>* geschrieben.

Speicher und Dateien bleiben für die Lebensdauer des Prozesses belegt.

Die in CMXTRACE angegebenen Optionen steuern den Trace. Die Optionen *s*, *S*, *D* und *G* bestimmen den Umfang der Protokollierung. Die Optionen *p, r* steuern Pufferung und Rundschreiben der Datei.

Syntax der Umgebungsvariable

CMXTRACE="[␣**-s**][␣**-S**][␣**-D**][␣**-p**␣fac][␣**-r**␣wrap][␣**-f**␣directory][␣**-G**␣dirx]";

export␣**CMXTRACE**

Die Optionen *-s, -S* und *-D* bestimmen die Art des Trace. Um den Trace zu aktivieren, muss ein Wert angegeben werden.

-s

Es erfolgt eine gewöhnliche Protokollierung der ICMX(L)-Aufrufe, ihrer Argumente, der Optionen und Benutzerdaten.

-S

Es erfolgt eine ausführliche Protokollierung der Aufrufe, ihrer Argumente, des Inhalts eventueller Optionen, der Benutzerdaten in ihrer Gesamtlänge.

Die Optionen *s* und *S* schließen sich gegenseitig aus.

-D

Es erfolgt eine ausführliche Protokollierung der Aufrufe mit zusätzlichen Informationen über Systemaufrufe. Diese Angabe kann nur zusätzlich zu *s* oder *S* gemacht werden.

-p␣fac

Durch die Dezimalziffer *fac* wird der Faktor der Pufferung bestimmt. Die Pufferung erfolgt im Betrag *fac* * BUFSIZ mit BUFSIZ gemäß *<stdio.h>*. Wird *fac* = 0 angegeben, so wird jeder Trace-Eintrag sofort (ungepuffert) in die Datei geschrieben.

fac=0...8.
-p␣fac nicht angegeben: Es wird *fac=1* angenommen.

-r␣wrap

Durch die Dezimalzahl *wrap* wird angegeben, dass nach *wrap* * BUFSIZ byte (BUFSIZ gemäß <*stdio.h*>) in die zweite temporäre Datei *<directory>/CMXMa<pid>* protokolliert werden soll.

Die zweite Datei *CMXMa<pid>* behandelt der Trace genauso wie *CMXLa<pid>*.

Nach jeweils *wrap* * BUFSIZ byte schaltet der Trace zwischen *CMXLa<pid>* und *CMXMa<pid>* um. Dabei geht der alte Inhalt der jeweiligen Datei verloren.

-r wrap nicht angegeben: Es wird *wrap* = 512 angenommen.

-f␣directory

Die Trace-Dateien werden in das angegebene Directory geschrieben. *-f directory* nicht angegeben: Für *directory* wird */var/tmp* angenommen.

-G␣dirx

Es erfolgt eine ausführliche Protokollierung mit zusätzlichen Informationen über DIR.X-Aufrufe. Die Protokolldatei wird im laufenden Directory unter dem Namen *logfile.<pid>* abgelegt.

dirx ist folgendermaßen zu belegen:

0x0
: Verfolgen TNS-Namen

0x02
: Verfolgen DIR.X-Namen

0x04
: Verfolgen Konfiguration

0x08
: Verfolgen interne Aufrufe

0x10
: Verfolgen „alarms“

Diese Werte können binär kombiniert werden.

Die Option wird in der MT-Bibliothek ignoriert.

Aufbereitung des Trace - cmxl

cmxl liest die vom Trace erzeugten Einträge aus der temporären Datei *file*, verarbeitet sie entsprechend den angegebenen Optionen und gibt das Ergebnis auf *stdout* aus.

Syntax

cmxl[␣**-c**][␣**-d**][␣**-e**][␣**-t**][␣**-v**][␣**-x**][␣**-D**]␣file ...

Die Optionen geben an, welche Trace-Einträge aus *file* aufbereitet werden sollen. Es ist möglich, mehrere der im Folgenden beschriebenen Werte pro Aufruf von *cmxl* anzugeben. Lediglich die Optionen *v* und *x* schließen einander aus.

Keine Option angegeben: Es wird *cdex* angenommen.

-c

Die Aufbereitung erfolgt für die ICMX(L)-Aufrufe:

- zur An- und Abmeldung der TS-Anwendung bei CMX
- zum Verbindungsaufbau und -abbau
- zur Verbindungsumlenkung

-d

Die Aufbereitung erfolgt für die ICMX(L)-Aufrufe:

- zum Datenaustausch
- zur Flussregelung

-e

Die Aufbereitung erfolgt für die ICMX(L)-Aufrufe zur Ereignisbehandlung.

-t

Es erfolgt zusätzlich zu der Protokollierung der Fehlermeldungen eine explizite Aufbereitung der *t_error()*-Aufrufe.

Fehlermeldungen werden immer protokolliert, auch wenn diese Option nicht angegeben wird.

-v

Es erfolgt eine ausführliche Aufbereitung der ICMX(L)-Aufrufe, ihrer Argumente, der Optionen, der Benutzerdaten. Der Umfang der Aufbereitung der Daten ist abhängig von den bei CMXTRACE angegebenen Optionen.

-x

Es erfolgt eine eingeschränkte Aufbereitung der Aufrufe und ihrer Argumente *ohne* Optionen und Benutzerdaten.

-D

Es erfolgt eine ausführliche Aufbereitung mit zusätzlichen Informationen über Systemaufrufe.

file ...

Name einer oder mehrerer Dateien mit Trace-Einträgen, die aufbereitet werden sollen.

Ausgabeformate (CMXTRACE, NEATRACE, cmxl, neal)

Die Beschreibung der Trace-Informationen in diesem Abschnitt erfordert Kenntnisse über die Programmschnittstellen ICMX(L) bzw. ICMX(NEA) von CMX. Die Programmschnittstellen sind im Handbuch „CMX, Anwendungen programmieren“ [1] beschrieben.

Das Format der durch *cmxl* und *neal* aufbereiteten Trace-Informationen ist gleich. Aus diesem Grund wird hier nur das Ausgabeformat von *cmxl* beschrieben. Die Ausgabe von *neal* ist damit ebenfalls interpretierbar. Die Angaben t_... sind lediglich durch x_... zu ersetzen mit Ausnahme der Aufrufe *t_vdatarq()* und *t_vdatain()*.

Die von *cmxl* aufgearbeiteten Trace-Informationen werden in folgendem Format ausgegeben:

```
1. ICMX(L) Trace Expansion              Expanded: Jun  9 13:50:53

2. Trace collected with data length 0 starting Jun  9 13:50:20.
3. Application was running in 32-bit singlethreaded mode.
4. Trace expanded by CMX 6.0 from file "CMXLa02830" with options
5.  c expand ICMX connection handling calls (set by default)
6.  d   expand ICMX data and flow control calls (set by default)
7.  D   expand system calls (sockets etc.; implies v)
8.  e   expand ICMX event handling calls (set by default)
9.  v   verbose mode showing args, options, user data
10. X   expand XTI system calls in case of ICMX over XTI
11. using T_MSG_SIZE=256, FD_SETSIZE=1024, openmax=0,
12. fac=0, wrap=4194304.

13. Traced System and CMX: Linux gibnix 2.4.19-4GB #1 Fri Sep 13
    13:14:56 UTC 2002 i686; CMX CMX-J-6.0A00-01

14. hh:mm:ss.msc ICMX Call t_***** or (intermediate) results

1. Zeile        zeitstempel t_xxxxx(args in %d, 0x%x, %s)
2. Zeile        [Optionen und Benutzerdaten in %d, %x, %s]
3. Zeile                    [TRANSPORTADRESSE, LOKALER NAME]
4. Zeile        [Ergebnisse, Ereignisse in %d, %x, %s]
1. Zeile        zeitstempel ... nächster Aufruf ...
```

Die Kopfzeilen 1-14 werden einmal zu Beginn der Ausgabe der Trace-Informationen ausgegeben. Sie enthalten u.a.:

- Versionsbezeichnung und CMX-Version (hier V6.0)
- Startdatum (datum) und Startzeit (hh:mm:ss) des Trace
- Angabe, ob es sich um den 32- oder 64-Bit-Modus handelt
- Hinweis, ob die Anwendung singlethreaded oder multithreaded ist

– die ausgewählten Optionen zur Aufbereitung (hier sind es die Standardoptionen *c, d, D, e, v, X*)
– Name der aufbereiteten Trace-Datei (trace file)
– die Version der Programmschnittstelle.

Die Ausgabe der Trace-Informationen zu den einzelnen Funktionsaufrufen erfolgt in mehreren Zeilen unterschiedlichen Formats. Im Folgenden sind die Formate dieser Zeilen beschrieben (Bezeichnungen der Zeilenformate: 1. Zeile bis 4. Zeile).

1. Zeile

Jeder protokollierte Funktionsaufruf erhält am Anfang der 1. Zeile einen Zeitstempel. Bei *cmxl* hat der Zeitstempel die Form hh:mm:ss:msc mit hh = Stunden, mm = Minuten, ss = Sekunden und msc = Millisekunden. Bei *neal* hat der Zeitstempel die Form hh:mm:ss.

Es folgen der protokollierte Funktionsaufruf *(t_xxxxx)* und in runden Klammern die Werte der Argumente *(args)* in der von ICMX(L) verlangten Reihenfolge. Die Argumente werden in dezimaler (%d), hexadezimaler (0x%x) oder symbolischer (%s) Form dargestellt.

Zur Interpretation der protokollierten Werte ist Folgendes zu beachten:

– Bei den Argumenten *datap, fromaddr, name, opt, toaddr* wird die übergebene Adresse (0x%x) dargestellt.
– Bei den Argumenten *chain, flags, cmode, pid, reason* ist hier der entsprechende Wert (0x%x, %d, oder %s) dargestellt, auch wenn die Argumente eigentlich die Adressen dieser Werte sind.
– Beim Argument *tref* ist der entsprechende Wert (0x%x) dargestellt. Ausnahmen sind die Aufrufe: *t_conrq()* und *t_event()*. Dort wird für *tref* die übergebene Adresse dargestellt.
– Bei der Protokollierung von Datenlängen, z. B. *datal* ist der beim Aufruf gültige Wert (%d) dargestellt. Bei folgenden Aufrufen wird zusätzlich der eventuell modifizierte Wert (%d), der nach der Rückkehr gültig ist, ausgegeben:

  t_concf(), t_conin(), t_datain(), t_vdatain(), t_xdatin(), t_redin() und *t_disin()*

  Beide Werte sind durch >< (Größerzeichen Kleinerzeichen) voneinander getrennt.

Die 2. Zeile und die 3. Zeile werden nur ausgegeben, wenn die Option *v* bei *cmxl* angegeben wurde und der Trace entsprechende Informationen gesammelt hat (Option *-S*).

2. Zeile

In der 2. Zeile sind die Optionen mit Optionsnummern und Optionsfeldern protokolliert. Sie stehen in der Reihenfolge, wie sie in der Optionsstruktur gemäß der Headerfile *<cmx.h>* deklariert sind.

Zur Interpretation der protokollierten Werte ist Folgendes zu beachten:

– Bei den Optionsfeldern *t_maxl, t_optnr, t_timeout* und *t_xdata* wird der übergebene Wert (0x%x, %d oder %s) dargestellt. Bei *t_udatap* wird die übergebene Adresse dargestellt.

– Bei dem Argument *t_udatal* wird der beim Aufruf gültige Wert (%d) dargestellt. Bei den Aufrufen *t_conin()*, *t_concf()*, *t_disin()* und *t_redin()* wird zusätzlich der eventuell modifizierte Wert (%d), der bei der Rückkehr gültig ist, ausgegeben. Die beiden Werte werden durch >< (Größerzeichen Kleinerzeichen) voneinander getrennt.

3. Zeile

Die Zeilen mit dem Format „3. Zeile" protokollieren die TRANSPORTADRESSE, den LOKALEN NAMEN und Benutzerdaten, sofern diese vom Trace protokolliert und von *cmxl* aufbereitet wurden. Es folgen die Daten, dargestellt in hexadezimaler und in abdruckbarer Form.

Ein Beispiel für die Ausgabe in der 3. Zeile:

```
Distanz   hexadezimal dargestellte Daten    . abdruckbar
  0     4c4f4b41 4c455220 4e414d45 20242424 .LOKALER NAME $$$
```

4. Zeile

In der 4. Zeile eines Eintrags wird das Ergebnis des Aufrufs protokolliert. Im Falle eines Fehlers wird T_ERROR eingetragen. Wurde der Aufruf erfolgreich durchgeführt, so wird das Ergebnis nur protokolliert, wenn es von T_OK abweicht. Das Ergebnis wird dann zusammen mit den Informationen protokolliert, die durch den Aufruf zurückgeliefert wurden.

Dies ist bei folgenden Aufrufen das folgende Ergebnis:

– *t_attach*: das Ergebnis T_NOTFIRST
– *t_conrq*: die gelieferte Transportreferenz (tref)
– *t_event*: das gemeldete Ereignis und die zugehörige Transportreferenz (tref)
– *t_datain, t_vdatain, t_xdatin*: die Restlänge noch zu lesender Daten
– *t_datarq, t_vdatarq, t_xdatrq*: die Ergebnisse T_DATASTOP, T_XDATSTOP

Die Darstellung der protokollierten Ergebnisse der Aufrufe erfolgt in der Regel auf folgende Weise:

– dezimal (%d) bei Längen oder Wertangaben
– symbolisch (%s), wenn eine entsprechende Definition in der Datei <*cmx.h*> existiert
– hexadezimal (0x%x) in allen anderen Fällen

Sind für ein Argument oder Optionsfeld zwar Symbole definiert, aber der Wert entspricht keinem der zulässigen Symbole, so erfolgt die Ausgabe hexadezimal (0x%x) mit einem Fragezeichen (?). Zum leichteren Auffinden sind die Ergebnisse T_ERROR mit mehreren # gekennzeichnet.

Beispiel für die Aufbereitung durch cmxl

Für die Aufbereitung des Trace wurden die Optionen *c, d, e, x* ausgewählt. D. h. es werden alle Aufrufe für An- und Abmeldung bei CMX, Verbindungsaufbau, -abbau und -umlenkung, Datenaustausch und Flussregelung aufbereitet mit ihren Argumenten ohne Optionen, ohne Benutzerdaten.

```
ICMX(L) Trace Expansion                          Expanded: Jun  9 14:22:34

Trace collected with data length 0 starting Jun  9 13:50:20.
Application was running in 32-bit singlethreaded mode.
Trace expanded by CMX 6.0 from file "CMXLa02830" with options
c    expand ICMX connection handling calls (set by default)
d    expand ICMX data and flow control calls (set by default)
e    expand ICMX event handling calls (set by default)
x    restricted mode excluding args, options, user data (set by default)
using T_MSG_SIZE=256, FD_SETSIZE=1024, openmax=0,
fac=0, wrap=4194304.

Traced System and CMX: Linux gibnix 2.4.19-4GB #1 Fri Sep 13 13:14:56
UTC 2002 i686; CMX CMX-J-6.0A00-01

hh:mm:ss.msc ICMX Call t_***** or (intermediate) results
13:50:20.002 t_getloc(0x804a138, NULL)
       glob:
      0  44454d4f 5f504430  31                          |DEMO_PD01    |
       loc 0x4007aae0: LANINET RFC1006
      0  01002600 0e000000  00000001 04003430          |  &        40|
     10  30310000 00000000  04000400 50443031          |01       PD01|
     20  00000000 0000                                 |             |
13:50:20.003 t_attach(0x804a340, 0x804a03c)
       name: LANINET RFC1006
      0  01002600 0e000000  00000001 04003430          |  &        40|
     10  30310000 00000000  04000400 50443031          |01       PD01|
```

```
     20  00000000 0000                                  |             |
13:50:20.003 string: IPv6 netdb functions available
       t_uattid 0x0 t_attid 0x0 t_sptypes 0x100
       t_ccbits 0x4
13:50:20.003  returns T_OK (0)
13:50:20.003 t_callback(0x8048abc, (nil), NULL)
13:50:20.003  old callback routine was: 0x0
13:50:20.003 t_event(0x804a468, T_WAIT, 0x804a120)
13:50:21.870  returns T_CONIN (5) tref 0x30000001
13:50:21.871 t_conin(0x30000001, 0x804a340, 0x804a3e0, 0x804a0e8)
       toaddr: LANINET RFC1006
      0  01002600 0e000000  00000001 04003430          |  &        40|
     10  30310000 00000000  04000400 50443031          |01       PD01|
     20  00000000 0000                                 |             |
       fromaddr: RFC1006
      0  02001f00 04001000  1500300d 03490000          |       0  I  |
     10  00070001 7f000001  fe800441 443031            |          AD01|
```

Ausgabe zusätzlich mit der Option -v.

```
ICMX(L) Trace Expansion                          Expanded: Jun  9 14:28:28

Trace collected with data length 0 starting Jun  9 13:50:20.
Application was running in 32-bit singlethreaded mode.
Trace expanded by CMX 6.0 from file "CMXLa02830" with options
c       expand ICMX connection handling calls (set by default)
d       expand ICMX data and flow control calls (set by default)
e       expand ICMX event handling calls (set by default)
v       verbose mode showing args, options, user data
using T_MSG_SIZE=256, FD_SETSIZE=1024, openmax=0,
fac=0, wrap=4194304.

Traced System and CMX: Linux gibnix 2.4.19-4GB #1 Fri Sep 13 13:14:56
UTC 2002 i686; CMX CMX-J-6.0A00-01

hh:mm:ss.msc ICMX Call t_***** or (intermediate) results
13:50:20.002 t_getloc(0x804a138, NULL)
       glob:
      0  44454d4f 5f504430  31                         |DEMO_PD01    |
       loc 0x4007aae0: LANINET RFC1006
      0  01002600 0e000000  00000001 04003430          |  &        40|
     10  30310000 00000000  04000400 50443031          |01       PD01|
     20  00000000 0000                                 |             |
13:50:20.003 t_attach(0x804a340, 0x804a03c)
       opt:
       t_optnr T_OPTA2 (3)  t_apmode T_PASSIVE (2)
       t_conlim 1
       name: LANINET RFC1006
```

```
      0  01002600 0e000000  00000001 04003430          |  &        40|
     10  30310000 00000000  04000400 50443031          |01       PD01|
     20  00000000 0000                                 |             |
13:50:20.003 string: IPv6 netdb functions available
       t_uattid 0x0 t_attid 0x0 t_sptypes 0x100
       t_ccbits 0x4
13:50:20.003  returns T_OK (0)
13:50:20.003 t_callback(0x8048abc, (nil), NULL)
13:50:20.003  old callback routine was: 0x0
13:50:20.003 t_event(0x804a468, T_WAIT, 0x804a120)
       opt:
       t_optnr T_OPTE1 (1)  t_timeout T_NOLIMIT (-1)
13:50:21.870  returns T_CONIN (5) tref 0x30000001
       t_attid 0x0  t_uattid 0x0
       t_ucepid 0x0  t_evdat 0x0 (0)
13:50:21.871 t_conin(0x30000001, 0x804a340, 0x804a3e0, 0x804a0e8)
       opt:
       t_optnr T_OPTC1 (1)  t_udatap (nil)  t_udatal 0><0
       t_xdata T_NO (0)  t_timeout T_NO (0)
       toaddr: LANINET RFC1006
      0  01002600 0e000000  00000001 04003430          |  &        40|
     10  30310000 00000000  04000400 50443031          |01       PD01|
     20  00000000 0000                                 |             |
       fromaddr: RFC1006
      0  02001f00 04001000  1500300d 03490000          |       0  I  |
     10  00070001 7f000001  fe800441 443031            |        AD01 |
```

Dateien

*CMXLa*pid, *CMXMa*pid
: Dateien mit kompakten Trace-Einträgen in Binärformat.

Falls bei CMXTRACE nicht anders angegeben, werden die Dateien *CMXLa<pid>* und *CMXMa<pid>* für den CMX-Bibliothekstrace im Verzeichnis */var/tmp* angelegt.

8.2.1 Hinweise für MT

Ein in CMX V6.0 erzeugter CMX-Bibliothekstrace kann threadspezifisch aufbereitet werden. Dafür wird die Shell-Prozedur *cmxl_mt* bereitgestellt. Sie wertet die zuvor mit *cmxl* aus dem Binärformat erzeugte ASCII-Datei *<tracefile>* aus (**cmxl ... > *<tracefile>***). Für Diagnosezwecke sollte jeweils die komplette ASCII-Datei zur Verfügung gestellt werden.

Die Prozedur erlaubt folgende Aufruf-Syntax:

cmxl_mt[␣**-h**][␣**-t**␣<tid> | all][␣<file> ...]

Optionen und Argumente

<tid> Thread-ID

<file> ASCII-Trace-Datei

cmxl_mt (Aufruf ohne Argumente)
Die ASCII-Tracedatei wird von stdin gelesen; es werden lediglich der Trace-Header und eine Liste der in der Tracedatei vorkommenden Thread IDs auf stdout ausgegeben.

-h
Anzeige der Syntax-Beschreibung.

-t <tid>
Die ASCII-Tracedatei wird von stdin gelesen; es werden alle Einträge für die Thread-ID <tid> auf stdout ausgegeben.

-t all
Die ASCII-Tracedatei wird von stdin gelesen; für jeden Thread werden dessen Einträge in die Datei <file>.<tid> ausgegeben. Der Trace-Header und die Liste der Thread-IDs werden in der Datei <file>.hdr abgelegt

<file ...>
Es wird die ASCII-Tracedatei <file> gelesen und lediglich der Trace-Header und die Thread-IDs auf stdout ausgegeben.

-t <tid> <file>
Es wird die ASCII-Tracedatei <file> gelesen und die Einträge für den Thread <tid> auf stdout ausgegeben.

-t all <file>
Es wird die ASCII-Tracedatei <file> gelesen; für jeden Thread werden dessen Einträge in die Datei <file>.<tid> ausgegeben. Der Trace-Header und die Liste der Thread-IDs werden in der Datei <file>.hdr abgelegt.

Ausgabeformate

Das Ausgabeformat bei MT ändert sich gegenüber der bisherigen Ausgabe nur geringfügig. In der durch *cmxl* aufbereiteten ASCII-Datei ist zusätzlich die Thread ID in der Form "[*<tid>*]" ausgewiesen (siehe unten). Alle nachfolgenden Trace-Einträge sind solange diesem Thread zugeordnet, bis eine andere Thread ID ausgewiesen wird. Die Bedeutung der ausgegebenen Zeilen bleibt wie bisher.

Beispiele

Ausgabeformat des durch cmcl aufbereiteten <tracefile>:

```
ICMX(L) Trace Expansion                          Expanded: Jun  9 15:17:33

Trace collected with data length 0 starting Jun  9 15:16:32.
Application was running in 32-bit multithreaded mode.
Trace expanded by CMX 6.0 from file "CMXLa03042" with options
c       expand ICMX connection handling calls (set by default)
d       expand ICMX data and flow control calls (set by default)
e       expand ICMX event handling calls (set by default)
v       verbose mode showing args, options, user data
using T_MSG_SIZE=256, FD_SETSIZE=1024, openmax=1024,
fac=0, wrap=4194304.

Traced System and CMX: Linux gibnix 2.4.19-4GB #1 Fri Sep 13 13:14:56
UTC 2002 i686; CMX CMX-J-6.0A00-01

hh:mm:ss.msc ICMX Call t_***** or (intermediate) results
[1024]
15:16:32.929 t_getloc(0xbffff3c1, NULL)
       glob:
      0  44454d4f 5f504430  31                    |DEMO_PD01       |
       loc 0x804e1c4: LANINET RFC1006
      0  01002600 0e000000  00000001 04003430     |  &           40|
     10  30310000 00000000  04000400 50443031     |01          PD01|
     20  00000000 0000                           |                |
[1026]
15:16:32.935 t_attach(0x804ce80, 0xbf7ffadc)
       opt:
       t_optnr T_OPTA1 (1)  t_apmode T_REDIRECT (4)
       t_conlim 1
       name: LANINET RFC1006
      0  01002600 0e000000  00000001 04003430     |  &           40|
     10  30310000 00000000  04000400 50443031     |01          PD01|
     20  00000000 0000                           |                |
15:16:32.936  returns T_OK (0)
15:16:32.936 t_event(0xbf7ffa50, T_WAIT, 0xbf7ffa2c)
       opt:
       t_optnr T_OPTE1 (1)  t_timeout T_NOLIMIT (-1)
[2051]
15:16:32.936 t_attach(0x804ce80, 0xbf5ffadc)
       opt:
       t_optnr T_OPTA1 (1)  t_apmode T_REDIRECT (4)
       t_conlim 2
       name: LANINET RFC1006
      0  01002600 0e000000  00000001 04003430     |  &           40|
```

```
     10  30310000 00000000  04000400 50443031     |01          PD01|
     20  00000000 0000                            |                |
15:16:32.937  returns T_OK (0)
15:16:32.937 t_event(0xbf5ffa50, T_WAIT, 0xbf5ffa2c)
       opt:
       t_optnr T_OPTE1 (1)  t_timeout T_NOLIMIT (-1)
```

Ausgabeformat der Headerdatei <tracefile>.hdr:

```
ICMX(L) Trace Expansion                          Expanded: Jun  9 15:17:33

Trace collected with data length 0 starting Jun  9 15:16:32.
Application was running in 32-bit multithreaded mode.
Trace expanded by CMX 6.0 from file "CMXLa03042" with options
c       expand ICMX connection handling calls (set by default)
d       expand ICMX data and flow control calls (set by default)
e       expand ICMX event handling calls (set by default)
v       verbose mode showing args, options, user data
using T_MSG_SIZE=256, FD_SETSIZE=1024, openmax=1024,
fac=0, wrap=4194304.

Traced System and CMX: Linux gibnix 2.4.19-4GB #1 Fri Sep 13 13:14:56
UTC 2002 i686; CMX CMX-J-6.0A00-01

hh:mm:ss.msc ICMX Call t_***** or (intermediate) results

Found Thread Id's
    4101
    1024
    8201
    3076
    1026
    7176
    6151
    5126
    2051
```

Ausgabeformat einer threadspezifischen Aufbereitung der Datei <tracefile>.<tid>:

```
Trace filter for thread ID 1026, entries separated from file <file>

hh:mm:ss.msc ICMX Call t_***** or (intermediate) results
15:16:32.935 t_attach(0x804ce80, 0xbf7ffadc)
        opt:
        t_optnr T_OPTA1 (1)  t_apmode T_REDIRECT (4)
        t_conlim 1
        name: LANINET RFC1006
       0  01002600 0e000000  00000001 04003430     |  &           40|
      10  30310000 00000000  04000400 50443031     |01          PD01|
      20  00000000 0000                           |                 |
15:16:32.936  returns T_OK (0)
15:16:32.936 t_event(0xbf7ffa50, T_WAIT, 0xbf7ffa2c)
        opt:
        t_optnr T_OPTE1 (1)  t_timeout T_NOLIMIT (-1)
15:16:34.557  returns T_REDIN (8) tref 0x30000001
        t_attid 0x0  t_uattid 0x0
        t_ucepid 0x0  t_evdat 0x0 (0)
```

8.3 NEABX-Bibliotheks-Trace steuern und aufbereiten (neal)

Der Trace der NEABX-Bibliothek wird über die Umgebungsvariable NEATRACE aktiviert und gesteuert. Die Trace-Einträge eines Prozesses werden kompakt und binär in einem dynamisch angelegten Puffer gesammelt und periodisch in temporäre Dateien geschrieben. Die Aufbereitung dieser Dateien erfolgt entkoppelt durch *neal*.

Steuerung des Trace - NEATRACE

Jeder CMX-Aufruf *t_attach* eines Prozesses wertet die Umgebungsvariable NEATRACE aus und aktiviert gegebenenfalls den Trace. NEATRACE muss vor dem Starten der Anwendung, d. h. vor dem ersten *t_attach* des zu überwachenden Prozesses, gesetzt worden sein. Nach dem Aktivieren des Trace wird die temporäre Datei *NEAL<pid>* mit der Prozess-Identifikation *<pid>* eröffnet, falls sie nicht bereits eröffnet ist. Für die Dateien werden die Zugriffsrechte 0600 vergeben. Anschließend wird dynamisch Speicher für die Pufferung der Trace-Einträge belegt.

Speicher und Dateien bleiben für die Lebensdauer des Prozesses belegt.

Die in NEATRACE angegebenen Optionen steuern den Trace. Die Optionen *s*, *S* bestimmen den Umfang der Protokollierung. Die Optionen *p,d, r* steuern Pufferung, Datenlänge und Rundschreiben der Dateien.

Die Variable NEATRACE wird in folgender Syntax angegeben:

NEATRACE= [**-s**|**S**] [␣**-p**␣fac] [␣**-d**␣lng␣**-r**␣wrap] [␣**-f**␣file]

export␣NEATRACE

-s und *-S* bestimmen die Art des Trace. Sie dürfen nur einen der beiden Werte angeben. Um den Trace zu aktivieren, müssen Sie einen Wert angeben.

-s

Gewöhnliche Protokollierung: Es werden alle Aufrufe und deren Argumente protokolliert. Optionen und Benutzerdaten werden nicht protokolliert.

-S

Ausführliche Protokollierung: Es werden alle Aufrufe, deren Argumente, der Inhalt der Optionen und die Benutzerdaten protokolliert.

-p␣fac

Durch die Dezimalziffer *fac* wird der Faktor der Pufferung bestimmt. Die Pufferung erfolgt im Betrag *fac* * BUFSIZ mit BUFSIZ gemäß der UNIX-Headerdatei <*stdio.h*>.

Wird *fac* = 0 angegeben, so wird jeder Trace-Eintrag sofort (ungepuffert) in die Datei geschrieben.

Wertebereich für *fac*=0...8.
-p␣fac nicht angegeben: Es wird *fac=0* angenommen.

-d␣Ing

Die Dezimalzahl *Ing* gibt in Byte an, bis zu welcher Länge TIDUs bei Angabe der Steueroption *-S* protokolliert werden. Der Standardwert ist 16.

Ing=0...16 ...65535.

Bei Angabe 0 erfolgt keine Protokollierung, sonst bis zu der angegebenen Maximallänge.

-r␣wrap

Durch die Dezimalzahl *wrap* wird angegeben, dass nach *wrap* * BUFSIZ byte (BUFSIZ gemäß <*stdio.h*>) in die zweite temporäre Datei *NEAM*.pid protokolliert werden soll.

Die zweite Datei *NEAM*.pid behandelt der Trace genauso wie *NEAL*.pid.

Nach jeweils *wrap* * BUFSIZ Byte schaltet der Trace zwischen *NEAL*.pid und *NEAM.pid* um. Dabei geht der alte Inhalt der jeweiligen Datei verloren.

-r wrap nicht angegeben: Es wird *wrap=256* angenommen.

-f␣file

Diese Option dient zur Angabe eines Dateiverzeichnisses *file*, in dem die Trace-Dateien *NEA[LM]*.pid hinterlegt werden sollen. Dabei kann das Argument sowohl ein relativer als auch ein absoluter Pfadname sein.

Aufbereitung des Trace - neal

neal liest die vom Trace erzeugten Einträge aus den Dateien, die Sie für *file* angegeben haben. Die Einträge verarbeitet *neal* entsprechend der angegebenen Optionen und gibt das Ergebnis auf *stdout* aus. Das Kommando hat folgende Syntax:

neal[␣**-c**][␣**-d**][␣**-e**][␣**-n**][␣**-v**][␣**-x**][␣**-D**][␣**-p**]␣file ...

Die gewählten Optionen geben an, welche Trace-Einträge aufbereitet werden sollen. Es ist möglich, mehrere der im Folgenden beschriebenen Werte pro Aufruf von *neal* anzugeben. Lediglich die Optionen *-v* und *-x* schließen einander aus. Ist keine Option angegeben, so wird *-cdex* angenommen.

-c

Die Aufbereitung erfolgt für die Funktionsaufrufe:

– zur An- und Abmeldung der TS-Anwendung bei NEABX
– zum Verbindungsaufbau und -abbau

-d

Die Aufbereitung erfolgt für die Funktionsaufrufe:

– zum Datenaustausch
– zur Fluss-Regelung

-e

Die Aufbereitung erfolgt für die Funktionsaufrufe zur Ereignisbehandlung.

-v

Es erfolgt eine ausführliche Aufbereitung der Funktionsaufrufe, ihrer Argumente, der Optionen, der Benutzerdaten. Der Umfang der Aufbereitung der Daten ist abhängig davon, ob bei NEATRACE die Option *-s* oder die Option *-S* angegeben wurde.

-x

Es erfolgt eine eingeschränkte Aufbereitung der Funktionsaufrufe und ihrer Argumente *ohne* Optionen und Benutzerdaten.

-D

Wurde bei der Trace-Steuerung die Option *D* angegeben, dann kann bei *neal* die interne Debugging-Information ausgewertet werden.

-p

Wird die Option *p* gesetzt, so müssen Sie für den Parameter *file* die Prozess-Identifikation (pid) des Prozesses einer TS-Anwendung angeben, für den die Trace-Einträge aufbereitet werden sollen.

file ...

Anzugeben ist der Name einer oder mehrerer Dateien, die die aufzubereitende Trace-Informationen enthalten. Falls u. a. die Option *p* angegeben wurde, ist für file nur die Prozess-Identifikation des Prozesses anzugeben, für den Trace-Einträge aufbereitet werden sollen. *neal* sucht dann im Dateiverzeichnis */var/tmp* nach allen zu diesem Prozess gehörenden Trace-Dateien.

Ausgabeformate

Das Format der durch *neal* aufbereiteten Trace-Information ist im Abschnitt „CMX-Bibliotheks-Trace steuern und aufbereiten (cmxl)“ auf Seite 71 beschrieben.

Dateien

NEAL.pid, NEAM.pid

Dateien mit kompakten Trace-Einträgen in Binärformat.

Falls bei NEATRACE nicht anders angegeben werden die Dateien für den NEABX-Bibliothekstrace *NEAL<pid>* und *NEAM<pid>* im Directory */var/tmp* abgelegt.

Siehe auch

cmxl

8.4 TS-Directory prüfen (tnsxchk)

Das Programm *tnsxchk* sucht Fehler in dem angegebenen TS-Directory. *tnsxchk* untersucht zunächst die Zeiger und Indizes innerhalb der Dateien des zu überprüfenden TS-Directory DIR<num> (num = 1...9) auf Plausibilität. Erkennt es hierbei keinen Fehler, so prüft *tnsxchk* das Format der Einträge innerhalb der Dateien. *tnsxchk* gibt an, ob eine der Dateien fehlerhaft ist oder nicht. Zur Diagnose liefert *tnsxchk* auf Anforderung detaillierte Informationen zu den gefundenen Fehlern. *tnsxchk* erkennt auch, wenn das TS-Directory nicht die vom TNS-Dämon erwartete Struktur hat. Das Kommando hat folgende Syntax:

tnsxchk[␣**-d**␣num][␣**-v**][␣**-f**]

-d␣num

num gibt die Nummer des zu überprüfenden TS-Directory DIR<num> an.

Ohne Angabe wird *num* = 1 eingestellt.

-v

tnsxchk prüft, ob das TS-Directory DIR<num> die Version hat, die der TNS-Dämon verwenden kann.

Ohne Angabe von *-v* überprüft *tnsxchk* alle Dateien des TS-Directory.

-f

tnsxchk liefert detaillierte Informationen zu den gefundenen Fehlern. Diese Informationen sind sehr komplex und werden nur für die Diagnose durch den Kundendienst benötigt.

Ausgabeformat

Die Ausgabe von *tnsxchk* wird durch eine Kopfzeile eingeleitet, die den Programmnamen, den Namen des überprüften TS-Directory, das Datum und den Zeitpunkt des Programmstarts enthält.

Danach wird der Inhalt der Datei VERSION des TS-Directory ausgegeben. Die in der Datei VERSION enthaltenen Werte sind betriebssystemabhängig. Ein Beispiel für die Ausgabe des Kommandos *'tnsxchk-v'* ist:

```
VERSION:
========
VERSION=V6.0 BYTEORDER=LSB LONG=32BIT INTEGER=32BIT SHORT=16BIT
```

Für jede Datei des TS-Directory, die *tnsxchk* überprüft, gibt *tnsxchk* folgende zwei Zeilen aus:

```
tnsxchk: datei wird überprüft:
tnsxchk: ergebnis
```

Für *datei* wird der absolute Pfadname der gerade geprüften Datei angegeben. Für *ergebnis* wird „OK" oder „Fehlerhaft" ausgegeben.

Dateien

DIR<num>
TS-Directory (<num> = 1,...,9)

Die folgenden Dateien sind Dateien innerhalb des TS-Directory DIR <num> (i = 1,...,5)

NAMEP<i>

NPVALUES

PROPERTIES

PRVALUE

ROOT

FREENPV

FREENPRV

VERSION

Die TS-Directories DIR1-DIR9 befinden sich unter dem Verzeichnis */opt/lib/cmx*.

Die Datei LOG für das pauschale Ändern von TNS-Einträgen wird im Directory */var/tmp* abgelegt.

Die Datei *tnsxd.trc* zur Protokollierung der TNS-Zugriffe befindet sich im Directory */opt/lib/cmx*. Dort gibt es auch die Datei *tnsxd.pid*, in der die aktuelle Prozessnummer des aktiven *tnsxd*-Prozesses abgelegt ist.

Siehe auch

tnsxd

8.5 TS-Directory erstellen, aktualisieren, lesen (tnsxcom)

tnsxcom bearbeitet Einträge des Formates *tnsxfrm* (standard TNS entry format). Über den Parameter *option* kann die Betriebsweise des TNSXCOM eingestellt werden. Folgende Betriebsweisen können angegeben werden:

UPDATE
: Aktualisieren des TS-Directory entsprechend der Einträge in der Datei *file* (den Dateien *file* ...).

INTERAKTIV
: Aktualisieren des TS-Directory entsprechend der Angaben von stdin

LOAD
: Erstellen eines zuvor leeren TS-Directory aus den Einträgen in der Datei *file* (den Dateien *file* ...).

DUMP
: (Sortierte) Ausgabe des Inhalts eines TS-Directory im Format *tnsxfrm* in die Datei *file*.

CHECK
: Überprüfen der Syntax der Datei *file*

CHECK_UPD
: Überprüfen der Syntax der Datei *file* und bei korrekter Syntax Aktualisieren des TS-Directory entsprechend der Einträge in *file*.

Bei Abwesenheit von Optionen compiliert *tnsxcom* die Einträge aus *file* im UPDATE-Modus (zeilenweise) in das TS-Directory.

tnsxcom mit der Betriebsweise UPDATE, INTERAKTIV, LOAD und CHECK_UPD (*option = u, i, l, S*) kann nur vom Systemverwalter aufgerufen werden.

Sobald *tnsxcom* eine Datei *file* abgearbeitet hat, gibt der TNSXCOM folgende Werte auf *stderr* aus:

- die Anzahl der aufgetretenen Fehler und Warnungen,
- die während der Bearbeitung der Datei *file* abgelaufene Zeit (real),
- den Anteil von real, den der *tnsxcom*-Prozess verbraucht hat, in %
- die verbrauchte Zeit aufgeteilt in Benutzermodus (user) und Systemmodus (sys).

Wurden mehrere Dateien von *tnsxcom* bearbeitet (*file* ...), so gibt *tnsxcom* am Ende der Bearbeitung diese Werte für den gesamten Compilierlauf aus.

Syntax

tnsxcom[␣**-d**␣num][␣**-**option][␣**-t**][␣**-p**␣prmt][␣**-o**␣orig][␣file ...]

tnsxcom[␣**-d**␣num][␣**-DilsSu**][␣**-t**][␣**-p**␣prmt][␣**-o**␣orig][␣file ...]

-d␣num

Nummer des TS-Directory, das bearbeitet werden soll. Mögliche Angabe für *num*: 1,2,...,9

Ohne Angabe wird *num* = 1 eingestellt.

-option

Betriebsweise von *tnsxcom* festlegen.

Ohne Angabe wird *option* = *u* eingestellt. Die Optionen *D*, *i*, *l*, *s*, *S*, *u* schließen einander aus. Folgende Angaben für *option* sind möglich.

D

DUMP-Modus

tnsxcom bereitet den Inhalt des TS-Directory in Einträgen im Format *tnsxfrm* in die Datei *file* auf, die wiederum als Eingabe verwendbar ist. Die Ausgabe erfolgt sortiert in aufsteigender ASCII-Sortierreihenfolge nach den GLOBALEN NAMEN gemäß ihrer Namenshierarchie und für jeden Namensteil. Es wird also zuerst nach Namensteil[1] sortiert. Existieren im TS-Directory mehrere TS-Anwendungen mit demselben Namensteil[1], so werden diese nach Namensteil[2] sortiert usw.

i

INTERAKTIV-Modus

tnsxcom liest Einträge im Format *tnsxfrm* von *stdin*, nachdem er durch Ausgabe einer Promptzeichenfolge (siehe *-p prmt*) seine Eingabebereitschaft angezeigt hat, und mischt sie in das TS-Directory, indem er bisher im TS-Directory noch nicht vorhandene Einträge erfasst oder existierende Einträge aktualisiert.

l

LOAD-Modus

tnsxcom nimmt die Einträge einzeln aus der Datei *file* und füllt das (bisher leere) TS-Directory mit den syntaktisch korrekten Einträgen.

s

CHECK-Modus

tnsxcom wendet nur die Syntaxprüfung auf die Datei *file* an und protokolliert mögliche Syntaxfehler. Das TS-Directory wird nicht verändert.

S

CHECK_UPD-Modus

Wie bei Option *s* erfolgt in einem ersten Lauf zuerst die Syntaxprüfung auf die gesamte Datei *file*. Treten in *file* keine Syntaxfehler auf, so aktualisiert *tnsxcom* dann das TS-Directory in einem zweiten Lauf.

u

UPDATE-Modus

tnsxcom nimmt die Einträge einzeln aus der editierbaren Datei *file* und mischt die syntaktisch richtigen Einträge in das TS-Directory durch Erfassung bisher nicht vorhandener oder Aktualisierung existierender Einträge.

Option *u* ist der Standardwert von *option*.

-t

Trace von *tnsxcom* einschalten.

Der Trace wird eingeschaltet und dessen Ausgabe in der Datei *tnsxcom.trc* im laufenden Dateiverzeichnis protokolliert. Bei Angabe von *option* = *D* wird die Option *t* ignoriert.

-o␣orig

orig gibt den Ursprung vor. Der Ursprung ist eine Folge von Namensteilen. Der Inhalt des *name*-Feldes eines *tnsxfrm*-Eintrags wird um . (Punkt) und dem Wert von *orig* ergänzt, falls der Inhalt nicht mit . (Punkt) endet. Das Resultat muss ein syntaktisch korrekter GLOBALER NAME mit maximal fünf Namensteilen sein.

-p␣prmt

die Zeichenfolge *prmt* wird im interaktiven Modus *i* als Prompt verwendet und sonst ignoriert.

Keine Angabe von *-p prmt*: es wird *prmt* = * angenommen.

file ...

Name der Datei mit Einträgen im Format *tnsxfrm*, die im Fall *option* = *l*, *s*, *S* oder *u* von *tnsxcom* ausgewertet werden soll. Es können mehrere Dateien angegeben werden. Im Fall *option* = *D* ist der Name der Datei anzugeben, in die *tnsxcom* den Inhalt des TS-Directory aufbereiten soll.

Fehler

Fehlermeldungen, die sich auf Syntax oder Semantik der Einträge in *file* beziehen, gibt *tnsxcom* zusammen mit dem Namen der Datei und der Zeilennummer auf *stderr* aus.

Dateien

tnsxcom.trc

Name der Tracedatei im Dateiverzeichnis, in dem *tnsxcom* aufgerufen wurde.

Die TS-Directories DIR1-DIR9 befinden sich unter dem Verzeichnis */opt/lib/cmx*.

Die Datei LOG für das pauschale Ändern von TNS-Einträgen wird im Directory abgelegt.

Die Datei *tnsxd.trc* zur Protokollierung der TNS-Zugriffe befindet sich im Verzeichnis */var/tmp*. Dort finden Sie auch die Datei *tnsxd.pid*, in der die aktuelle Prozess-Nummer des aktiven *tnsxd*-Prozesses abgelegt ist.

Der TNS-Dämon *tnsxd* wird im Verzeichnis */opt/etc* aufgerufen.

Siehe auch

tnsxd, *tnsxprop*, sowie Beschreibung des Eingabeformats für den TNSXCOM im Abschnitt „Syntax der TNS-Konfigurationsdatei“ auf Seite 29.

Beispiel

Der Aufruf *tnsxcom -d 2 -l -o np4..np2.np1 input1 input2* überführt die Einträge aus den Dateien input1 und input2 in das leere TS-Directory 2; ein *inhalt* jeden *name*-Feldes, der nicht mit '.' endet, soll dabei zu "*inhalt.np4..np2.np1*" (mit leerem Namensteil 3) erweitert werden.

8.6 Transport Name Service Daemon starten (tnsxd)

tnsxd startet den TNSX-Daemon. Der TNSX-Daemon ist der zentrale Serverprozeß für die Anforderungen von TS-Anwendungen an den Transport-Name-Service (TNSX). Der Start erfolgt üblicherweise in der Startup-Datei des Systems.

Zusammen mit dem TNSX-Daemon kann auch der Verfolger für die Anforderungen an den TNSX gestartet werden. Den Namen des Dateiverzeichnisses, in dem Sie den TNSX-Daemon starten können, entnehmen Sie bitte der Freigabemitteilung.

tnsxd[␣**-t**][␣**-c**␣num][␣**-l**]

-t

Verfolger von Anfang an einschalten (im laufenden Betrieb kann der Verfolger mit *tnsxt* ein- und ausgeschaltet werden).

-c␣num

Die Option gibt an, dass nach *num* Einträgen die Verfolgerdatei rund zu schreiben ist.

Ohne Angabe wird *num* = 100 eingestellt.

-l

Die Option gibt an, dass die Ausgabe der Verfolgerinformation ungepuffert erfolgen soll. Dies ist nur in Fehlerfällen sinnvoll.

Endestatus

Beim Aufruf startet *tnsxd* einen Sohnprozess im Hintergrund, der mit den TS-Anwendungen kommuniziert. Bei erfolgreichem Start des Sohnprozesses beendet sich der Vaterprozessmit dem Status 0, sonst ungleich 0. Bei Beendigung durch SIGHUP beendet sich der Sohnprozess mit Status 0, in allen anderen Fällen mit Status 1.

Fehler

Die folqenden Fehlermeldungen können auf *stderr* ausgegeben werden:

```
"syntax: tnsxd [-1] [-t] [-c n] "
"tnsxd: error: bad option <-c n>
         max. %d trace -entries possible."
"tnsxd: error %d opening %s"
"tnsxd: error %d mknod'ing %s"
"tnsxd: error %d reading %s"
"tnsxd: error %d locking %s" "tnsxd: error %d writing %s"
"tnsxd: bad result {is %d, should %d) writing %s"
"tnsxd: process already exists"
"tnsxd: error %d in fork"
"tnsxd: cauqht signal %d"
```

Die angezeigten Fehlernummern (%d) können mit dem Kommando *cmxdec -s* entschlüsselt werden. Der Dateiname (%s) ist einer der Namen im folgenden Abschnitt.

Dateien

/opt/lib/cmx/tnsxd.pid
: Datei mit der PPID von *tnsxd*

/opt/lib/cmx/tnsxd.trc
: Datei mit dem Trace von *tnsxd*

/opt/lib/cmx/tnsxd.trc2
: Backup-Trace-Datei

/opt/lib/cmx/lock_dsa
: Lock-datei zur Koordinationen der Clients

/opt/lib/cmx/DIR[1-9]
: Pfad der TS-Verzeichnisse

/dev/cmx/dsa_request
: Anforderungs-FIFO

/dev/cmx/dsa_result
: Ergebnis-FIFO

Siehe auch

tnsxlock, *tnsxt*

8.7 Informationen zum TS-Directory anzeigen (tnsxinfo)

tnsxinfo gibt den Inhalt, die Grenzwerte und die aktuelle Belegung eines TS-Directory in lesbarer Form auf der Standardausgabe *stdout* aus. Die ausgegebenen Informationen erhält *tnsxinfo* durch direkten Zugriff auf das jeweilige TS-Directory, d. h. ohne Anforderung an den TNS-Dämon.

tnsxinfo liefert statistische Informationen über die Belegung der TS-Directories, bereitet die GLOBALEN NAMEN der TS-Anwendungen des TS-Directory in einer Baumstruktur auf und gibt die den TS-Anwendungen zugeordneten Eigenschaften aus.

Syntax

tnsxinfo[␣**-d**␣num][␣**-g**][␣**-p**][␣**-v**]

-d␣num
: definiert *num* als die Nummer des zu verwendenden TS-Directory. Ohne Angabe wird *num* = 1 eingestellt.

-g
: *tnsxinfo* gibt den Inhalt der Datei VERSION im TS-Directory, die Grenzwerte für die Belegung und Informationen über die aktuelle Belegung des TS-Directory aus. Alle im TS-Directory vorhandenen GLOBALEN NAMEN werden in einer Baumstruktur aufbereitet ausgegeben.

-p
: *tnsxinfo* gibt den Inhalt der Datei VERSION im TS-Directory, die Grenzwerte für die Belegung und Informationen über die aktuelle Belegung des TS-Directory aus. Zusätzlich werden alle im TS-Directory enthaltenen Eigenschaften aufbereitet ausgegeben.

-v
: *tnsxinfo* gibt nur den Inhalt der Datei VERSION des TS-Directory aus.

Wird keine der Optionen *-g, -p, -v* angegeben, so gibt *tnsxinfo* den Inhalt der Datei VERSION und Tabellen mit den Grenzwerten des TS-Directory und Informationen über seine aktuelle Belegung auf *stdout* aus.

Ausgabeformat

Die Ausgabe von *tnsxinfo* beginnt immer mit zwei Kopfzeilen und dem Inhalt der Datei VERSION im TS-Directory.

Die Kopfzeilen enthalten:

– Name und Version des Programms

– aktuelles Datum und Zeit des Programmstarts

– Name des TS-Directory, auf das sich die Ausgabe bezieht.

Ausgabe der Grenzwerte und der aktuellen Belegung

Grenzwerte und aktuelle Belegung des TS-Directory werden in Form von drei Tabellen ausgegeben. Die in den Tabellen enthaltenen Informationen können unvollständig sein, wenn zur Zeit der Informationsabfrage der TNS-Daemon läuft (z. B. wenn Daten zur Zeit der Abfrage über den TNS-Daemon verändert werden oder der TNS-Daemon Daten zum schnelleren Zugriff in einem cache hält). Die erste Tabelle enthält die Angaben zu den einzelnen Namensteilen der GLOBALEN NAMEN. Die Tabelle ist wie folgt aufgebaut:

```
GRENZWERTE FUER NAMENSTEILE UND EIGENSCHAFTEN:
Namensteil TS_COUNTRY TS_ADMD TS_PRMD  TS_OU  TS_PN
Index             1       2       3      4      5
Laenge            2      16      16     10     30
Maximum         113     223    1093   4093  16381
Grenze          100     200    1000   4000  16000
belegt            1       2       2      2      5
```

Die erste Zeile der Tabelle enthält die symbolischen Bezeichnungen der Namensteile des GLOBALEN NAMENS (siehe Abschnitt „Syntax der TNS-Konfigurationsdatei“ auf Seite 29). Darunter steht der Index des entsprechenden Namensteils. Index 1 gehört zu Namensteil1 usw. Die Einträge in den weiteren Zeilen haben folgende Bedeutung:

Laenge
: Erlaubte Maximallänge der einzelnen Namensteilwerte in Byte

Maximum
: Größe der Hash-Tabellen in den einzelnen Dateien NAMEP*i* (i = 1,...,5).

Grenze
: Anzahl der Namensteile mit Index *i*, die Sie maximal in das TS-Directory eintragen können.

belegt
: Anzahl der Namensteile mit Index *i*, die derzeit im TS-Directory existieren.

Die zweite Tabelle enthält Grenzwerte und aktuelle Belegung der Eigenschaften, der Namensteil- und Eigenschaftswerte (dies entspricht der maximalen Länge und der aktuellen Länge der Dateien NPVALUES und PRVALUES).

```
             Eigenschaften    Namensteilwerte    Eigenschaftswerte
Maximum             320000             560700             21406500
belegt                  14                 54                  134
Luecken (max.)       -/-             0 (100)              0 (50)
```

Die Einträge haben folgende Bedeutung:

Maximum
: maximale Anzahl der Eigenschaften, die Sie im TS-Directory ablegen können bzw. maximale Länge der Dateien NPVALUES, die alle Namensteilwerte der GLOBALEN NAMEN enthält, und PRVALUES, die alle Werte der Eigenschaften enthält (siehe Abschnitt „Syntax der TNS-Konfigurationsdatei“ auf Seite 29). Die Längen werden in Byte angegeben.

belegt
: Anzahl der Eigenschaften, die im TS-Directory abgelegt sind bzw. aktuelle Länge der Dateien NPVALUES und PRVALUES in Byte.

Luecken (max.)
: Anzahl der Lücken in den Dateien PROPERTIES (sie enthält die Namen der Eigenschaften), NPVALUES und PRVALUES, die durch Löschen entstanden sind. Wenn die Anzahl der Lücken den in Klammern angegebenen Maximalwert übersteigt, wird das TS-Directory reorganisiert und die Lücken geschlossen.

Die dritte Tabelle enthält die erlaubte Maximallänge der verschiedenen Eigenschaftswerte in Byte.

```
properties:
type            Laenge(Byte) type             Laenge(Byte)
TS_EMPTYPROP               0   TS_TRANS               200
TS_NEABX                   1   TS_TRSYS                 1
TS_LNAME                 200   TS_USER1PR             200
TS_USER2PR               200   TS_USER3PR             200
```

Die Bezeichnungen für die Eigenschaften haben folgende Bedeutung:

```
TS_EMPTYPROP   Leereintrag,
TS_LNAME       LOKALER NAME,
TS_USER2PR     USER2,
TS_TRANS       TRANSPORTADRESSE,
TS_TRSYS       TRANSPORTSYSTEM,
TS_USER1PR     USER1,
TS_USER3PR     USER3.
```

Die Eigenschaften sind in Abschnitt „Konfigurieren mit tnsxcom“ auf Seite 27 beschrieben.

Ausgabe der GLOBALEN NAMEN

Die im TS-Directory abgelegten GLOBALEN NAMEN werden in Form eines Namensbaums ausgegeben (Beispiel für einen Namensbaum siehe Abschnitt „GLOBALER NAME“ auf Seite 29). Hier wird nur ein Teil der im Beispiel von *tnsxinfo* gelieferten Informationen abgedruckt.

```
GLOBALE NAMEN:
     ----i-[Npi] [x] NonLeaf, Name part i, Value Npi, Index x
    ====i=[Npi] [x] [p] Leaf, Name part i, Value Npi, Index x,
Propindex p

     ROOT
     |----1-[co] [53]
     |       |----2-[admd1] [82]
     |       |       |----3-[prmd1] [461]
     |       |       |       |----4-[ou1] [873]
     |       |       |       |       |====5=[pn4] [875] [10]

       :
       :
```

Jede einfach gestrichelte Linie `---i-` führt zu einem Knoten im Namensbaum. Jede doppelt gestrichelte Linie `===i=` führt zu einem Blatt im Namensbaum. Jedem Blatt ist eindeutig eine TS-Anwendung zugeordnet. Der Weg im Namensbaum von der Wurzel ROOT bis zu dem Blatt entspricht dem GLOBALEN NAMEN der TS-Anwendung. Nur den Blättern sind Eigenschaften zugeordnet.

Die angegebenen Werte haben folgende Bedeutung:

i
: Index des dargestellten Namensteils.

[Npi]
: Wert des dargestellten Namensteils.

[x]
: Hash-Tabellenindex des dargestellten Namensteils in der Datei NAMEP*i*. Der Wert liegt zwischen 1 und dem in der 1. Tabelle angegebenen Maximum für die Hashtabelle.

[p]
: Index der ersten Eigenschaft der TS-Anwendung in der Tabelle in der Datei PROPERTIES. Dieser Wert wird nur ausgegeben, wenn Sie von *tnsxinfo* auch die Ausgabe der Eigenschaften angefordert haben.

Ausgabe der Eigenschaften

Die Eigenschaften aller TS-Anwendungen in dem TS-Directory werden wie folgt aufbereitet: (Es ist nur ein Teil der im Beispiel von *tnsxinfo* gelieferten Informationen abgedruckt.)

```
PROPERTIES: 14 Eintraege
  #  xfnl next [t property    lng   off] [np# ind]
  0: XF..    1 [I TS_...        1     0] [ 5  864]
           0  00
  1: X.N.    2 [I TS_TRANS     20     1] [ 5  864]
           0  02001400 01004000  0a005bc4 c9c1d3d6
          10  c7401208
  2: X..L   -1 [I TS_TRSYS      1    15] [ 5  864]
           0  00
  3: XF..    4 [I TS_...        1    16] [ 5  868]
           0  00
  4: X.N.    5 [I TS_TRANS     37    17] [ 5  868]
           0  02002500 02001000  1b00300f 03490000
          10  00090001 08001410  1996fe80 08d3c1d5
          20  c1d5e640 40
  5: X.N.    6 [I TS_TRSYS      1    3c] [ 5  868]
           0  02
```

In der ersten Zeile der Tabelle gibt *tnsxinfo* an, wieviele Eigenschaften in dem TS-Directory eingetragen sind. Für jede dieser Eigenschaften werden mehrere Zeilen ausgegeben. Die erste Zeile enthält die Werte zu den in der Kopfzeile der Tabelle angegebenen Größen, die folgenden Zeilen den Wert der Eigenschaft in hexadezimaler Form.

Die Größen der Kopfzeile haben die folgende Bedeutung:

#
: Index der Eigenschaft in der Datei PROPERTIES des TS-Directory.

xfnl
: gibt an, ob es sich um die erste oder die letzte Eigenschaft der TS-Anwendung in der Liste handelt. Mögliche Werte für xfnl sind:

  X
  : Eintrag ist belegt.

  F
  : erster Eintrag einer Eigenschaft der TS-Anwendung.

  L
  : letzter Eigenschaftseintrag für diese TS-Anwendung.

  N
  : Folgeeintrag einer Eigenschaft für die TS-Anwendung.

  Die Werte F, L, N schließen einander aus.

next
: Index # der nächsten Eigenschaft, die zu derselben TS-Anwendung gehört.

t
: Typ der Eigenschaft. Es wird derzeit nur der Wert *I* für TS_ITEM ausgegeben. Der Typ TS_GROUP (G) wird zur Zeit noch nicht unterstützt.

property
: Bezeichnung der Eigenschaft. Es werden die Bezeichnungen ausgegeben, die auch bei der Ausgabe der erlaubten Längen der Eigenschaftswerte verwendet wurden (siehe oben).

lng
: Länge des Wertes der Eigenschaft in Byte.

off
: Offset in der Datei PRVALUES des TS-Directory, in der die Eigenschaftswerte stehen. Offset gibt den Abstand vom Dateianfang bis zum Beginn dieses Eigenschaftswertes in Byte an.

np#
: Index Namensteil, dem die Eigenschaft zugeordnet ist (np# = 1,...,5).

ind
: Hash-Tabellenindex dieses Namensteils in der Datei NAMEPnp#

Fehler

Besitzt das in *-d num* angegebene TS-Directory eine falsche Datei VERSION, so gibt *tnsxinfo* den Inhalt der Datei VERSION und die in VERSION erwarteten Werte zusammen mit einer Fehlermeldung aus.

tnsxinfo gibt ebenso eine Fehlermeldung aus, wenn er die Datei VERSION eines TS-Directory nicht öffnen kann und die Datei ROOT eine falsche Länge hat. Beides deutet auf ein TS-Directory hin, das mit einer alten CMX-Version erzeugt und nicht gemäß Abschnitt „Migration“ auf Seite 49 umgesetzt wurde.

Dateien

DIR<num>
: TS-Directory (<num> = 1, ...,9)

8.8 Eigenschaften von TS-Anwendungen ausgeben (tnsxprop)

tnsxprop gibt die Werte aller Eigenschaften in einem abdruckbaren Format auf *stdout* aus, die in einem TS-Directory für die angegebenen TS-Anwendungen enthalten sind.

Mit Hilfe des Parameters *option* kann man festlegen, in welchem Format die Eigenschaften ausgegeben werden sollen.

Die TS-Anwendungen werden durch die Parameterwerte von *name* bestimmt. Die Parameterwerte für *name* können auch aus der Datei *file* an *tnsxprop* übergeben werden. Wird weder für *name* noch für *file* eine Angabe gemacht, so bereitet *tnsxprop* die Eigenschaften aller TS-Anwendungen des TS-Directory im angegebenen Format auf.

Syntax

tnsxprop[␣**-d**␣num][␣**-**option][␣**-f**␣file][␣name ...]

tnsxprop[␣**-d**␣num][␣**-hS**][␣**-f**␣file][␣name ...]

-d␣num

Nummer des TS-Directory, das verwendet werden soll.

Mögliche Angabe für *num*: 1,2,...,9

Ohne Angabe wird *num* = 1 eingestellt.

-option

Mit Hilfe des Parameters wird das Format festgelegt, in dem die Eigenschaften ausgegeben werden sollen.

Ohne Angabe wird *option* = *S* eingestellt.

Die im Folgenden beschriebenen Optionen *h* und *S* schließen sich gegenseitig aus. Mögliche Angaben für *option*:

h

Aufbereitung der Eigenschaften in hexadezimaler Darstellung. Die Ausgabe der Eigenschaftswerte erfolgt als Hexadezimalziffern-String zusammen mit der entsprechenden Bitdarstellung, wobei das niederwertigste Bit ganz rechts steht.

S

Aufbereitung der Eigenschaften in symbolischer Darstellung im Format *tnsxfrm*. Siehe hierzu Abschnitt „Syntax der TNS-Konfigurationsdatei“ auf Seite 29.

-f␣file

Für *file* ist der Name einer Datei anzugeben, die die GLOBALEN NAMEN der TS-Anwendungen enthält, deren Eigenschaften abgefragt werden sollen. Die GLOBALEN NAMEN sind wie unter *name* beschrieben anzugeben.

name

Für *name* ist der GLOBALE NAME der TS-Anwendung im TS-Directory wie folgt anzugeben:

NP5.NP4.NP3.NP2.NP. Die NPi sind die Namensteile des GLOBALEN NAMENS. Dabei ist NP5 der Namensteil[5], also der Namensteil der unteren Hierarchiestufe. NP1 ist der Namensteil[1], also der in der Hierarchie höchste Namensteil. Die Namensteile sind in von links nach rechts aufsteigender hierarchischer Reihenfolge anzugeben.

Ist bei einem GLOBALEN NAMEN einer der Namensteile nicht belegt (z. B. NP4) und folgt diesem Namensteil noch ein Namensteil höherer Hierarchie (z. B. NP3), so ist von dem nicht belegten Namensteil das Trennzeichen (.) anzugeben. Eine Folge von Trennzeichen am Ende des Wertes von *name* kann weggelassen werden.

Enthalten die Namensteile Sonderzeichen, deren Sonderbedeutung eine Mehrdeutigkeit der Syntax verursachen würde, so müssen diese Sonderzeichen mit \ (Gegenschrägstrich) entwertet werden. Im Zweifelsfall sollten Sie jedes Sonderzeichen entwerten. Ist die Entwertung überflüssig, so wird sie von *tnsxprop* ignoriert.

Gibt man für einen der Namensteile den Wert * (Stern) an, so liefert *tnsxprop* die Eigenschaften von allen TS-Anwendungen, die für diesen Namensteil einen beliebigen Wert haben und in den anderen Namensteilen mit der Angabe in *name* übereinstimmen (Filtermodus TS_RESTRICTED).

Dateien

DIR num
: Name des betreffenden TS-Directory (*num* = 1,...,9)

Beispiel 1

Der TS-Anwendung im TS-Directory 1, die nur Namensteil[5] mit dem Wert "Beispiel_1" hat, sind alle möglichen Eigenschaften zugeordnet. Diese Eigenschaften sollen in hexadezimaler Darstellung auf *stdout* ausgegeben werden. Geben Sie folgendes Kommando ein: ␣

Beispiel 2

Für die TS-Anwendung "Beispiel_1" im TS-Directory 1 sollen die Eigenschaften in symbolischer Darstellung auf *stdout* ausgegeben werden (Standardwert von *option*). Geben Sie folgendes Kommando ein: ␣

Hinweis

Die Aufrufe *tnsxprop -S > DAT1* und *tnsxcom -D DAT1* sind äquivalent, beide schreiben den Inhalt des TS-Directory 1 in symbolischer Darstellung in die Datei *DAT1*.

8.9 Sicherstellen und Aufbereiten der Trace-Information (tnsxt)

tnsxt kann von jedem Benutzer aufgerufen werden. Mit Hilfe von *tnsxt* kann der Trace der Zugriffe auf den TNS jederzeit gestartet und angehalten werden. *tnsxt* sammelt die letzten Zugriffe zum TNS in abdruckbarer Darstellung in einem Ringpuffer, solange der Trace eingeschaltet ist. Der Ringpuffer wird in der Datei *tnsxd.trc* angelegt und bleibt während der gesamten Lebensdauer des Systems bestehen. Der Trace kann jederzeit gestartet werden, der Start muss jedoch vor den zu überwachenden Zugriffen erfolgen. Hält man den Trace an, so kann der Ringpuffer ausgegeben werden. Bei Fortstart des Trace wird der Ringpuffer fortgeschrieben.

Syntax

tnsxt␣on|off

on
: der Trace wird (fort)gestartet,

off
: der Trace wird angehalten.

Dateien

/opt/lib/cmx/tnsxd.trc
: Datei mit Trace-Einträgen

Siehe auch

tnsxd

Fachwörter

Anwendung

Eine Anwendung ist ein System von Programmen, das ein bestimmtes Dienstangebot eines EDV-Systems anwendet, um einem menschlichen oder maschinellen Nutzer eine höherwertige Dienstleistung anzubieten. Kommunikationsanwendungen sind Anwendungen, die die Kommunikationsfunktionen eines EDV-Systems nutzen, um unter Nutzung eines Netzes systemübergreifende Dienstleistungen zu erbringen.

Den meisten Anwendungen wird ein Präfix zur Kennzeichnung des untergelagerten Dienstangebots vorangestellt (CMX-Anwendung, UTM-Anwendung, DCAM-Anwendung, Motif-Anwendung und Windows-Anwendung, etc.). Beispiele für Kommunikationsanwendungen sind Filetransfer, Terminalemulation, Electronic Mail, World Wide Web Browser und Server, Transaktionssysteme wie UTM, allgemein alle Anwendungen nach dem Client/Server-Prinzip.

API (Application Program Interface)

APIs sind Programmschnittstellen, die die Funktionen eines Programmsystems zur Verfügung stellen. Als Programmierer nutzen Sie die APIs bei der Programmierung von Anwendungen. APIs bieten Funktionen zum Verbindungsmanagement, zum Datenaustausch und zur Abbildung von Namen in Adressen. APIs im CMX-Umfeld sind Sockets, ICMX und XTI.

CC (Communications Controller)

Ein CC ist eine Baugruppe zum Anschluss eines UNIX-Rechners an ein Netz. Sie benötigen einen CC, um Ihren Rechner physisch an ein Subnetz anzuschließen, es sei denn, der Anschluss ist auf einer anderen Baugruppe, z. B. der Mutterplatine, mit integriert (onboard-Anschluss).

CLI (Command Line Interface)

CLI ist die Summe der Kommandos für *OA&M* von *CMX*. Als Administrator können Sie Initialisierungs-, Überwachungs-, Steuer- und Wartungsfunktionen von *CMX*und den *Communication Services* über die UNIX-Kommandozeile vornehmen.

CMX (Communications Manager UNIX)

CMX erbringt Kommunikationsdienste zur Nutzung von CMX-Anwendungen im Netz und ermöglicht die Programmierung von CMX-Anwendungen. CMX vereinheitlicht die Dienste unterschiedlicher Netze und

ermöglicht damit die Nutzung derselben CMX-Anwendung unabhängig vom unterliegenden Netz. Als Laufzeitsystem vermittelt CMX zwischen aktuellem Netzangebot und CMX-Anwendungen und bietet dem Netzadministrator einheitliche Funktionen für *OA&M* (Operation, Administration, Maintenance). Als Entwicklungssystem bietet CMX Schnittstellen (APIs) und Verfahren zur Programmierung von netzunabhängigen CMX-Anwendungen.

CMX-Anwendungen

CMX-Anwendungen sind Anwendungen, die die Dienste von CMX nutzen. CMX-Anwendungen haben im Netz eine Adresse, die *TRANSPORTADRESSE*. Sie identifizieren sich untereinander durch symbolische Namen, dem *GLOBALEN NAMEN* einer Anwendung.

Communication Services

Communication Services dienen der Verknüpfung von heterogenen Netzen verschiedener Architektur bzw. unterschiedlicher Technologie. Durch Einsatz von Communication Services können beispielsweise unterschiedliche LAN-WAN-Kopplungen realisiert werden, wobei der entsprechende Communication Service eine Software-Komponente z. B. auf einem Server ist.

GLOBALER NAME einer Anwendung

Jede CMX-Anwendung identifiziert sich selbst und ihre Kommunikationspartner im Netz durch symbolische, hierarchische GLOBALE NAMEN. Ein GLOBALER NAME besteht aus bis zu fünf Namensteilen (NP[1- 5]), die Sie zur Definition der Anwendung (NP5), des Rechners (NP4) und (bis zu drei) administrativer Domänen (NP[3-1]) verwenden können.

Beispiel: Der GLOBALE NAME "IhreAnwendung.D018S065.mch-p.sni.de" bedeutet: "IhreAnwendung" residiert im Host "D018S065" in der Domäne "mch-p.sni.de".

Bei der Wahl eines GLOBALEN NAMENS müssen Sie als Administrator die Vorgaben und Empfehlungen der speziellen Anwendung beachten.

Als Administrator können Sie dem GLOBALEN NAMEN einer Anwendung 1:1 eine *TRANSPORTADRESSE* oder einen *LOKALEN NAMEN* der Anwendung zuordnen. Als Programmierer können Sie die von CMX erwartete TRANSPORTADRESSE oder den LOKALEN NAMEN mit Hilfe der Funktionsaufrufe des *Transport Name Service* (TNS) aus dem GLOBALEN NAMEN gewinnen.

LOKALER NAME einer Anwendung

Eine CMX-Anwendung meldet sich in ihrem lokalen Rechner mit dem LOKALEN NAMEN bei CMX zur Kommunikation an. Der LOKALE NAME besteht aus einem oder mehreren *T-Selektoren*, die jeweils das Transportsystem bezeichnen, über das die CMX-Anwendung kommunizieren soll. Als Administrator können Sie mit dem LOKALEN NAMEN die Kommunikation einer CMX-Anwendung über bestimmte Transportsysteme ermöglichen oder ausschließen und etwaige Anforderungen der CMX-Anwendung nach bestimmten T-Selektor-Werten, z. B. beim Filetransfer, erfüllen.

Beispiel: Eine Anwendung soll den T-Selektor "cmxappl" (in Kleinbuchstaben!) für die Kommunikation über das TCP/IP- RFC1006 Transportsystem und den T-Selektor "$CMXAPPL" (in Großbuchstaben!) für die Kommunikation über das NEA-Transportsystem verwenden.

Den LOKALEN NAMEN einer Anwendung können Sie als Administrator in CMX mit dem Kommando tnsxcom dem *GLOBALEN NAMEN* der Anwendung zuordnen. Als Programmierer können Sie den von CMX erwarteten LOKALEN NAMEN mit Hilfe der Funktionsaufrufe des *Transport Name Service* (TNS) aus dem GLOBALEN NAMEN gewinnen.

Netz

Ein Netz ist ein Verbund zusammenwirkender Übertragungskomponenten (Leitungen, Vermittlungsknoten, Verfahren) mit einheitlich definierten Diensten, Protokollen und Zugangseinrichtungen für EDV-Systeme. Ein Netz verbindet Rechner zur Nutzung systemübergreifender Anwendungen miteinander. Das Netz eines Netzbetreibers kann sofort für Anwendungen oder zur Definition darauf aufbauender, überlagerter, privater Netzstrukturen genutzt werden. Im UNIX-Umfeld sind folgende Netze relevant: das Internet, SNA-, TRANSDATA- und OSI-Netze.

Ein Netz kann aus einem oder mehreren *Subnetzen* bestehen, die über das homogene Ende- zu-Ende-Protokoll des Netzes verknüpft sind. Die oben genannten Beispielnetze können Überlagerungen aus öffentlichen oder privaten Subnetzen wie dem X.25-Netz, dem Telefon- oder Datennetz, dem ISDN oder ATM-Netz und verschiedenen privaten, lokalen Netzen basierend auf Ethernet, Token Ring und FDDI sein.

Netzadresse

Jeder Rechner in einem *Netz* ist durch seine Netzadresse eindeutig identifiziert. Ein Rechner kann in unterschiedliche Netze eingebunden sein und hat dann für jedes dieser Netze eine spezifische Netzadresse. Im

Internet hat ein Rechner eine Internet-Adresse (IP-Adresse), die sich zusammensetzt aus Netz- und Host-Nummer (z. B. 129.144.89.171). Im NEA-Netz hat ein Rechner eine NEA-Netzadresse, die sich zusammensetzt aus Rechner-/Regions-Nummer (z. B. 124/213). Die OSI-Netzadresse (NSAP-Adresse) setzt sich zusammen aus dem Initial Domain Part (IDP) und dem Domain Specific Part (DSP) und hat das Format: IDP+DSP
(z. B. 470058+0144458100007391100308001411961301).

Subnetz

Ein Subnetz ist ein technisch oder administrativ homogener Teil eines *Netzes*. Subnetze sind u.a. das X.25-Netz, das Telefon- oder Datennetz, das ISDN oder ATM-Netz und verschiedene private, lokale Netze basierend auf Ethernet, Token Ring und FDDI. Der Zugang zu einem Subnetz kann über einen oder mehrere Subnetz-Anschlüsse erfolgen. Ein Subnetz-Anschluss wird durch seine *Subnetz-Adresse* identifiziert.

Subnetz-Adresse

Die Subnetz-Adresse beschreibt eindeutig einen Subnetz-Anschluss, der den Zugang zum *Subnetz* ermöglicht. Die Subnetz-Adresse ist beispielsweise eine ISDN-Rufnummer, eine DTE-Adresse oder eine Ethernet-Adresse.

Subnetz-ID

Die Subnetz-ID, auch SNID genannt, benennt eine Gruppe gleichartiger Subnetz-Anschlüsse, die in dasselbe *Subnetz* führen. Die Subnetz-ID gibt die Art des Subnetzes an und identifiziert, um welche Gruppe von Zugängen zu diesem Subnetz es sich handelt. Eine Subnetz-ID steht beispielsweise für zwei ISDN-Anschlüsse oder für mehrere X.25-Anschlüsse in einem Subnetz.

TNS (Transport Name Service)

Der TNS ist eine Komponente von *CMX*, die die korrekte Abbildung der *GLOBALEN NAMEN* von CMX-Anwendungen im Netz in *TRANSPORTADRESSEN* und *LOKALE NAMEN* unterstützt. Als Administrator konfigurieren Sie die von Ihnen gewählte Zuordnung von GLOBALER NAME zu TRANSPORTADRESSE für ferne Anwendungen sowie die Zuordnung von GLOBALER NAME zu LOKALER NAME für lokale Anwendungen. Als Programmierer von Anwendungen können sie diese Abbildungen über ein *API* nutzen und damit allein mit den *GLOBALEN NAMEN* von Anwendungen ohne Bewertung der Abbilder arbeiten.

Der TNS bietet die netzweite Identifikation von Anwendungen durch logische GLOBALE NAMEN und deren Abbildung in eine entsprechende *Netzadresse*. Damit können Sie die Anwendungen vom Wissen um ihre Netzadressen entkoppeln.

TRANSPORTADRESSE einer Anwendung

Eine rufende CMX-Anwendung übergibt die TRANSPORTADRESSE eines gerufenen Kommunikationspartners beim Aufbau der Kommunikation an *CMX*. CMX verwendet die TRANSPORTADRESSE, um den Kommunikationspartner im Netz zu lokalisieren und eine *Route* durch das Netz zu bestimmen. Die TRANSPORTADRESSE hängt im allgemeinen von der logischen und physischen Struktur des Netzes (und seiner Subnetze) ab. Sie enthält die für Ihr Netz spezifischen Vorgaben Ihrer/Ihres Netzbetreiber(s).

Als Administrator können Sie unabhängig von der Anwendung die TRANSPORTADRESSE und damit die Kommunikationswege beeinflussen. Bestandteile einer TRANSPORTADRESSE sind: eine Netzadresse zur eindeutigen Bestimmung des fernen Rechners, auf dem die Anwendung residiert, der Typ des *Transportsystems*, über das die ferne Anwendung erreicht werden kann, und der *T-Selektor*, der die ferne Anwendung im fernen Rechner identifiziert. Beispiele für Netzadressen sind: die Internet-Adresse in der Punktnotation "192.11.44.1" und die X.25-Adresse (DTE-Adresse) als Ziffernstring "45890010123".

Als Administrator können Sie dem *GLOBALEN NAMEN* einer Anwendung 1:1 eine TRANSPORTADRESSE der Anwendung zuordnen.

Als Programmierer können Sie die von CMX erwartete TRANSPORTADRESSE mit Hilfe der Funktionsaufrufe des *Transport Name Service* (TNS) aus dem GLOBALEN NAMEN gewinnen.

Transportsystem

Das Transportsystem bezeichnet die unteren vier Schichten des *OSI-Referenzmodells*. Ein *CCP* implementiert die vier Schichten des Transportsystems. Das Transportsystem sorgt für den gesicherten Datenaustausch zwischen Rechnern, deren *Anwendungen* miteinander kommunizieren, und zwar unabhängig von den darunterliegenden Netzstrukturen. Das Transportsystem verwendet dazu Protokolle.

T-Selektor

Der T-Selektor identifiziert eine Kommunikationsanwendung innerhalb des Rechners, auf dem die Anwendung abläuft. Der T-Selektor bildet zusammen mit der *Netzadresse* des Rechners die *TRANSPORTADRESSE* einer Anwendung, mit der diese Anwendung innerhalb eines Netzes eindeutig adressiert werden kann. Das Format und der Wertebereich des T-Selektors hängen vom Typ des *Netzes* ab.

Abkürzungen

ASCII
American Standard Code of Information Interchange

CC
Communication Controller

CCITT
Comité Consultatif International Télégraphique et Téléphonique

CMX
Communications Manager UNIX

DCAM
Data Communication Access Method

EBCDIC
Extended Binary Coded Decimals Interchange Code

EBNF
Extended Backus Naur Form

ETHN
ETHERNET

ETSDU
Expedited Transport Service Data Unit

FT
File Transfer

ICMX
Programming Interface CMX

ISDN
Integrated Services Digital Network

ISO
International Organization for Standardization

ITU
International Telecommunication Union

ITU-T
Telecommunication Standardization Sector

LAN
Local Area Network

MIB
Management Information Base

NEA
Netzwerk-Architektur bei TRANSDATA-Systemen

NSAP
Network Service Access Point

OSI
Open Systems Interconnection

PDN
Programmsystem für Datenfernverarbeitung und Netzsteuerung

PID
Process Identifier

PSDN
Packet Switched Data Network

PSTN
Public Switched Telephone Network

PVC
Permanent Virtual Circuit

SNA
Systems Network Architecture

SNID
Subnetz-Identifikation

SNPA
Subnet Point of Access

SVC
Switched Virtual Circuit

STA
Stationskopplung

TCEP
Transport Connection Endpoint

TCP/IP
Transmission Control Protocol/Internet Protocol

TEP
Transport Endpoint

TIDU
Transport Interface Data Unit

TNS
Transport Name Service

TPDU
Transport Protocol Data Unit

TR
Token-Ring

TREF
Transport Reference

TS
Transport Service

TSAP
Transport Service Access Point

TSDU
Transport Service Data Unit

TSTAT
TEP-Status

WAN
Wide Area Network

XTI
X/OPEN Transport Interface

Literatur

Die Handbücher sind online unter *http://manuals.fujitsu-siemens.com* zu finden oder in gedruckter Form gegen gesondertes Entgelt unter *http://FSC-manual-shop.com* zu bestellen.

[1] **CMX V6.0**
Communications Manager UNIX
CMX-Anwendungen programmieren

Zielgruppe
Programmierer

Inhalt
Das Handbuch beschreibt die Programmierschnittstellen von CMX, d. h. alle Werkzeuge, die Sie benötigen, um selbst TS-Anwendungen zu entwickeln.

[2] **XTI V5.1**
X/Open Transport Interface
User Guide

Zielgruppe
Programmierer von TS-Anwendungen.

Inhalt
Das Handbuch enthält implementierungsabhängige Ergänzungen zu den Funktionsaufrufen von XTI.

[3] **TRANSIT-SERVER V3.6** (UNIX)
Administration von TRANSIT
Basishandbuch, Teil 1 und 2

Zielgruppe
SINIX-Anwender in SNA-Netzen

Inhalt
Basisbeschreibung der TRANSIT-Produkte in SINIX

[4] **CS-GATE** (SINIX, Reliant UNIX)
Benutzerhandbuch

Zielgruppe
Dieses Handbuch richtet sich an Netz- und Systemadministratoren.

Inhalt
Das Handbuch beschreibt, wie Sie mit CS-GATE als Gateway LAN-WAN-, LAN-LAN- und LAN-WAN-LAN-Übergänge realisieren. Es erklärt Adressierung, Adress-Umsetzungs-Funktionen und Konfigurierung sowie Steuerung und Diagnose von CS-GATE.

Stichwörter

Fujitsu Siemens Computers GmbH
Handbuchredaktion
81730 München

Fax: 0 700 / 372 00000

email: manuals@fujitsu-siemens.com
http://manuals.fujitsu-siemens.com

Kritik
Anregungen
Korrekturen

Absender

Kommentar zu CMX V6.0 (UNIX)
Betrieb und Administration

Fujitsu Siemens Computers GmbH
Handbuchredaktion
81730 München

Fax: 0 700 / 372 00000

email: manuals@fujitsu-siemens.com
http://manuals.fujitsu-siemens.com

Kritik
Anregungen
Korrekturen

Absender

Kommentar zu CMX V6.0 (UNIX)
Betrieb und Administration

Information on this document

On April 1, 2009, Fujitsu became the sole owner of Fujitsu Siemens Computers. This new subsidiary of Fujitsu has been renamed Fujitsu Technology Solutions.

This document from the document archive refers to a product version which was released a considerable time ago or which is no longer marketed.

Please note that all company references and copyrights in this document have been legally transferred to Fujitsu Technology Solutions.

Contact and support addresses will now be offered by Fujitsu Technology Solutions and have the format *...@ts.fujitsu.com*.

The Internet pages of Fujitsu Technology Solutions are available at *http://ts.fujitsu.com/*...
and the user documentation at *http://manuals.ts.fujitsu.com*.

Hinweise zum vorliegenden Dokument

Zum 1. April 2009 ist Fujitsu Siemens Computers in den alleinigen Besitz von Fujitsu übergegangen. Diese neue Tochtergesellschaft von Fujitsu trägt seitdem den Namen Fujitsu Technology Solutions.

Das vorliegende Dokument aus dem Dokumentenarchiv bezieht sich auf eine bereits vor längerer Zeit freigegebene oder nicht mehr im Vertrieb befindliche Produktversion.

Bitte beachten Sie, dass alle Firmenbezüge und Copyrights im vorliegenden Dokument rechtlich auf Fujitsu Technology Solutions übergegangen sind.

Kontakt- und Supportadressen werden nun von Fujitsu Technology Solutions angeboten und haben die Form *...@ts.fujitsu.com*.

Die Internetseiten von Fujitsu Technology Solutions finden Sie unter *http://de.ts.fujitsu.com/*..., und unter *http://manuals.ts.fujitsu.com* finden Sie die Benutzerdokumentation.