Fakultät für Informatik
der Technischen Universität München

Methodik zur formalen Spezifikation des ISO/OSI Schichtenmodells

Christian Facchi

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender:	Univ.-Prof. Dr. P. P. Spies
Prüfer der Dissertation:	
1.	Univ.-Prof. Dr. M. Broy
2.	Univ.-Prof. Dr. E. Jessen

Die Dissertation wurde am 3. Mai 1995 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 7. Juni 1995 angenommen.

Kurzfassung

Die Standardisierungseinrichtung ISO hat mit dem ISO/OSI Schichtenmodell eine Grundlage für die Kommunikation in heterogenen Rechensystemen festgelegt. Eine Modularisierung erfolgt hierbei nach verschiedenen Aufgaben in sieben Schichten, wobei die Festlegung der schichtenspezifischen Aufgaben in den einzelnen Normen in größtenteils textueller Form vorliegt. Da informelle Definitionen unter anderen Nachteilen den der möglichen Mißinterpretation besitzen, wird in dieser Dissertation eine Methodik zur formalen Spezifikation des ISO/OSI Schichtenmodells vorgestellt, mit der die Diensterbringer aller Schichten beschreibbar sind. Mit dieser Methodik wird, im Gegensatz zu bisherigen Arbeiten, durch den Einsatz eines Modularisierungskonzepts bei der Spezifikation ein hoher Wiederverwendungsgrad erreicht. Dazu wird jeweils für mehrere der in den Normen verwendeten Beschreibungstechniken eine eigenständige formale Semantik angegeben, die selbstverständlich auch außerhalb des ISO/OSI Schichtenmodells anwendbar ist. So wird unter anderem erstmalig eine formale Semantik für die weit verbreiteten Time Sequence Diagrams beschrieben.

Durch die Entwicklung der formalen Semantik für die in den Normen verwendeten Beschreibungstechniken werden mit dieser Arbeit einige Ungenauigkeiten bei den Normen belegt und Lösungsvorschläge zur Behebung angeboten.

Die in den bisherigen Verfahren verwendeten Ansätze schränken das Verhalten der Dienstbenutzer, die kein Spezifikationsgegenstand der schichtenspezifischen Normen sind, ein. Die in dieser Arbeit entwickelte formale Semantik läßt dagegen ein freies Verhalten der Dienstbenutzer zu. Somit wird ausschließlich das Verhalten des Diensterbringers, der die zu implementierende Komponente darstellt, formal spezifiziert.

Aufbauend auf den formalen Semantiken werden die Beziehungen zwischen den einzelnen Beschreibungstechniken untersucht. Dabei wird die unterschiedliche Ausdrucksstärke der zuvor betrachteten Beschreibungstechniken festgestellt.

Danksagung

Ich möchte meinem Doktorvater Prof. Dr. Manfred Broy herzlich für die Gelegenheit, diese Arbeit zu erstellen, die Bereitstellung des Dissertationsthemas, die Betreuung und die Anmerkungen zu Vorversionen danken. Dank gebührt Prof. Dr. Eike Jessen für die sofortige Bereitschaft, das Zweitgutachten zu erstellen, und für seine wertvollen Hinweise zu einer Vorversion dieser Arbeit.

Insbesondere bei Ketil Stølen, Ph. D. möchte ich mich für Diskussionen und konstruktive Anmerkungen zu Vorversionen bedanken. Für das kritische Durchlesen von Vorversionen der Arbeit und wertvolle Rückkopplung haben sich Dr. Radu Grosu, Dr. Bernhard Neumair und Jan Philipps Dank verdient. Dank gilt auch allen Kolleginnen und Kollegen am Lehrstuhl, insbesondere Dr. Max Fuchs, Dr. Stephan Merz und Dr. Franz Regensburger für interessante Diskussionen.

Die liebevolle und syntaktische Unterstützung von Gitti Mühlbauer hat wesentlich zum Gelingen dieser Arbeit beigetragen. Abschließend möchte ich mich bei meinen Eltern, Marianne und Francesco Facchi, dafür bedanken, daß sie mir diese Ausbildung ermöglichten.

Inhaltsverzeichnis

1 Einleitung und Überblick **1**
- 1.1 Einführung in die Problemstellung . . . 1
  - 1.1.1 OSI Referenzmodell . . . 1
  - 1.1.2 Normen des OSI Referenzmodells . . . 3
- 1.2 Bearbeitung der Problemstellung in bestehenden Arbeiten . . . 4
  - 1.2.1 Aufgabenunabhängige Darstellung der Schichten . . . 5
  - 1.2.2 Verfeinerung der Diensterbringer . . . 6
  - 1.2.3 Formalisierungen im OSI Referenzmodell . . . 7
- 1.3 Motivation und Zielsetzung für den eigenen Ansatz . . . 9
- 1.4 Die wichtigsten Ergebnisse . . . 11
- 1.5 Aufbau der Arbeit . . . 12

2 Grundlagen **13**
- 2.1 Grundsätzliche Definitionen im OSI Referenzmodell . . . 13
- 2.2 Grundlagen des verwendeten Formalismus . . . 15
- 2.3 Grundsätzliche Modellierungsannahmen . . . 18

3 Formalisierung von Time Sequence Diagrams **19**
- 3.1 Informelle Darstellung von TSDs . . . 20
- 3.2 Ungenauigkeiten in den Normen bei der Verwendung von TSDs . . . 22
- 3.3 Formale Semantik mit Einschränkung des Umgebungsverhaltens . . . 26
  - 3.3.1 Konzepte zur Interpretation von TSDs . . . 26
  - 3.3.2 Syntax von TSD-DL . . . 30
  - 3.3.3 Denotationelle Semantik von TSD-DL . . . 31
  - 3.3.4 Beispiele für die Semantik einiger TSD-DL-Ausdrücke . . . 34
  - 3.3.5 Eigenschaften von TSD-DL . . . 36
  - 3.3.6 Methode zur Umwandlung von TSDs in TSD-DL-Ausdrücke . . . 43
  - 3.3.7 Beispiele für schematisch gewonnene TSD-DL-Ausdrücke . . . 46
- 3.4 Formale Semantik ohne Einschränkung des Umgebungsverhaltens . . . 47
  - 3.4.1 Konzepte der Semantik . . . 51
  - 3.4.2 Erweiterung um fehlerhafte Eingaben . . . 52
  - 3.4.3 Erweiterung um fehlende Eingaben . . . 54
  - 3.4.4 Kombination der Erweiterungen . . . 59
  - 3.4.5 Beispiele für Erweiterungen . . . 60
  - 3.4.6 Aufdeckung und Behebung der Nichtkompositionalität der Semantik 62

3.5 Anwendung der Formalisierungsmethodik auf die Dienstnormen 67
3.5.1 Untersuchung der TSD-DL-Ausdrücke bei den einzelnen Schichten . 67
3.5.2 Erweiterung von TSDs zur Einbeziehung der in realen Schichten zusätzlich angegebenen verbalen Anmerkungen 72
3.6 Bewertung der Formalisierungsmethodik 76
3.7 Bewertung von TSDs 77

4 Formalisierung der lokalen Einschränkungen 81
4.1 In den Normen verwendete Beschreibungsmethoden 82
4.2 Formale Semantik mit Einschränkung des Umgebungsverhaltens 86
4.3 Eigenschaften der formalen Spezifikation 88
4.4 Formale Semantik ohne Einschränkung des Umgebungsverhaltens 90
4.5 Anwendung der Formalisierungsmethodik auf die Schichten 93
4.5.1 Lokale Einschränkungen der Bitübertragungsschicht 93
4.5.2 Lokale Einschränkungen der Sicherungsschicht 96
4.5.3 Lokale Einschränkungen der Vermittlungsschicht 98
4.5.4 Lokale Einschränkungen der Transportschicht 100
4.6 Bewertung der Formalisierungsmethodik 101
4.7 Bewertung der in den Normen verwendeten Beschreibungstechniken 101

5 Formalisierung des Queuemodells 103
5.1 Informelle Darstellung des Queuemodells 104
5.2 Formale Semantik mit Einschränkung des Umgebungsverhaltens 107
5.2.1 Konzepte der Semantik 107
5.2.2 Formale Semantik der Präzedenzqueues 108
5.2.3 Formalisierung der Objekttabelle 110
5.2.4 Formalisierung des Einhaltens des Queuemodells 111
5.3 Eigenschaften des formalisierten Queuemodells 113
5.4 Formale Semantik ohne Einschränkung des Umgebungsverhaltens 114
5.5 Anwendung der Formalisierungsmethodik auf die Dienstnormen 114
5.5.1 Das Queuemodell der Bitübertragungsschicht 115
5.5.2 Das Queuemodell der Sicherungsschicht 116
5.5.3 Das Queuemodell der Vermittlungsschicht 119
5.5.4 Das Queuemodell der Transportschicht 121
5.6 Bewertung der in den Normen verwendeten Beschreibungstechnik 122
5.7 Bewertung der Formalisierungsmethodik 123

6 Vergleich der Beschreibungstechniken 125
6.1 Vergleich der einzelnen Beschreibungstechniken 126
6.2 Kombination der Beschreibungstechniken 131
6.3 Ergebnisse des Vergleichs der Beschreibungstechniken 135

7 Beschreibung einer Schicht 137
7.1 Schichtbeschreibung mit Einschränkung des Umgebungsverhaltens 137
7.2 Schichtbeschreibung ohne Einschränkung des Umgebungsverhaltens 139
7.3 Bewertung der Formalisierung 143

8 Betrachtung von mehreren Verbindungen **145**
8.1 Informelle Beschreibung eines Netzwerks von Verbindungen 145
8.2 Adreßidentifizierung der Dienstelemente 149
8.3 Formalisierung der Beschreibung von mehreren Verbindungen 150
8.4 Bewertung der Formalisierung 155

9 Zusammenfassung und Ausblick **157**
9.1 Zusammenfassung und Einordnung der wichtigsten Ergebnisse 157
9.2 Zukünftige Aktivitäten 159

Literaturverzeichnis **161**

Kapitel 1

Einleitung und Überblick

1.1 Einführung in die Problemstellung

Die internationale Standardisierungseinrichtung ISO hat das Open Systems Interconnection (OSI) Schichtenmodell zur Strukturierung der Rechnerkommunikation festgelegt. Ein Ziel des ISO/OSI Schichtenmodells ist es, eine gemeinsame Basis für die Kommunikation von verschiedenen Rechnern mit unterschiedlichen Betriebssystemen zur Verfügung zu stellen. Die Aufteilung in einzelne Schichten wurde unter anderem nach folgenden Kriterien vorgenommen [Zim80]:

- Die Aufteilung der Schichten ist nicht zu fein zu wählen, da sonst deren Anzahl zu groß wird, und der Beschreibungs- und Integrationsaufwand zu umfangreich wird.
- Die Schnittstellen zwischen den Schichten sind so zu bestimmen, daß die Dienstbeschreibung der Schichten und der Kommunikationsaufwand zwischen den Schichten gering ist.
- Wesentlich unterschiedliche Funktionen sind in verschiedenen Schichten anzusiedeln.
- Gleichartige Funktionen sind in einer Schicht zusammenzufassen.

Nach diesen Gesichtspunkten ist eine Aufteilung in sieben Schichten im OSI Schichtenmodell vorgenommen worden. Im weiteren wird dies auch mit dem Begriff OSI Referenzmodell bezeichnet.

1.1.1 OSI Referenzmodell

Das in Abbildung 1.1 dargestellte OSI Referenzmodell ist in der ISO Norm 7498 [ISO84a] festgelegt worden.

Eine Kommunikation zwischen zwei Rechnern erfolgt nur auf gleichen Schichten. Wenn beide Kommunikationspartner nicht direkt miteinander verbunden sind, wird die Information mit Hilfe von Vermittlungsrechnern ausgetauscht. Der tatsächliche, physikalische Datenaustausch erfolgt nur auf der untersten Schicht.

Die Aufgaben der Schichten werden informell in den Büchern [Bla91, Cyp91, JA93, Hal88, Ker92, Ros90, Tan88] dargestellt:

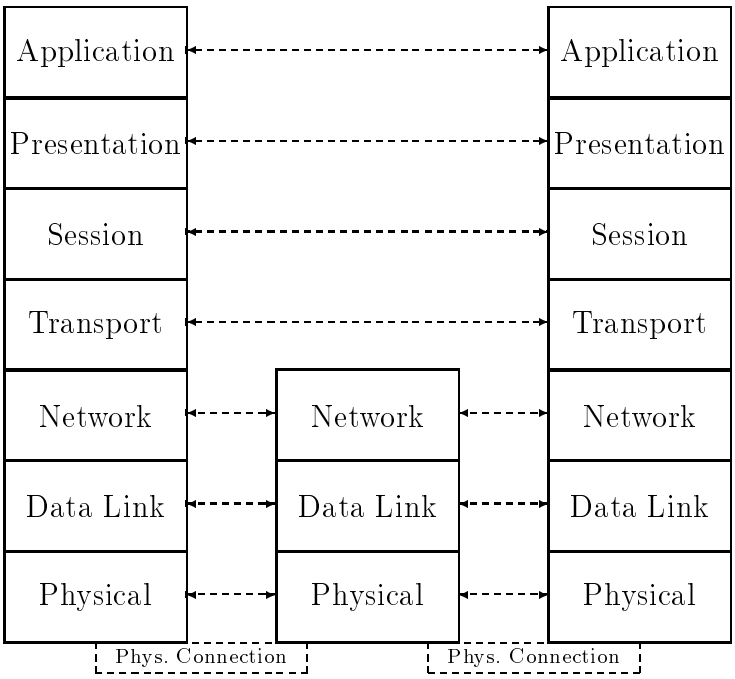

Abbildung 1.1: ISO/OSI Schichtenmodell

1. Die *Bitübertragungsschicht* (Physical Layer) stellt eine ungesicherte Systemverbindung zur Übertragung von Bits zur Verfügung. Dabei werden Daten nur zwischen real verbundenen Systemen ausgetauscht.

2. Die *Sicherungsschicht* (Data Link Layer) realisiert eine fehlerüberwachte, sichere Systemverbindung, das heißt eine Übertragung von Datenblöcken mit Fehlerprüfzeichen unabhängig vom physikalischen Medium.

3. Die *Vermittlungsschicht* (Network Layer) stellt eine Verbindung zwischen Endsystemen zur Verfügung. Dabei ist es möglich, daß die ausgetauschten Daten unter Zuhilfenahme von Vermittlungsrechnern, auch Transitsysteme genannt, übertragen werden. In dieser Schicht wird die Wegewahl vorgenommen.

4. In der *Transportschicht* (Transport Layer) werden beliebige Datenmengen zwischen zwei Endsystemen, die nicht notwendigerweise direkt verbunden sind, ausgetauscht.

5. In der *Kommunikationssteuerungsschicht* (Session Layer) werden Ausdrucksmittel zur Verwaltung einer Kommunikationsbeziehung zur Verfügung gestellt. Dabei wird vor allem ein Synchronisationsmanagement realisiert.

6. Die *Darstellungsschicht* (Presentation Layer) stellt Mittel zum darstellungsunabhängigen Austausch von Daten, sowie zur Datenkompression und Verschlüsselung zur Verfügung.

7. In der *Anwendungsschicht* (Application Layer) kommunizieren Anwendungen miteinander. Als Beispiele seien elektronische Mail Systeme genannt.

1.1.2 Normen des OSI Referenzmodells

Die offiziellen Spezifikationen der Standardisierungseinrichtung ISO für das OSI Schichtenmodell sind informell gehaltene Normen. Sie bestehen aus einer textuellen Beschreibung und sind stellenweise um formale Teile ergänzt. Grundlage für das ISO/OSI Schichtenmodell ist die Beschreibung aller Schichten im OSI Referenzmodell [ISO84a]. Eine detaillierte Version der Schichtspezifikation, in der sowohl der Dienst, als auch das Protokoll in jeweils einer Norm beschrieben wird, existiert für jede Schicht. Die zugehörigen Normenbezeichnungen werden in Tabelle 1.1 angeführt [JA90], in der nur die betreffende ISO-Normennummer angegeben ist. Die Spezifikation der Dienste der Anwendungsschicht setzt sich aus mehreren Normen für die jeweils unterschiedlichen Anwendungsbereiche zusammen.

Schichtbezeichnung	Spezifikationsgegenstand	Dienstspez.	Protokollspez.
Application Layer	Association Control	8649	8650
	Reliable Transfer	9066-1	9066-2
	Remote Operations	9072-1	9072-2
	CCR	9804	9805
	Directory Services	9594/3	9594/5
	Message Handling	10021-4,-5	10021-6
	File Transfer	8571/3	8571/4
	Virtual Terminal	9040	9041
Presentation Layer		8822	8823
Session Layer		8326	8327
Transport Layer		8072	8073,8602
Network Layer		8348	8878,8473
Data Link Layer		8886	8802
Physical Layer		10022	-

Tabelle 1.1: ISO-Normennummern

Die Standardisierungseinrichtung CCITT[1] hat einen Teil der ISO Normen mit geringfügigen Modifikationen übernommen. Das OSI Referenzmodell ist in der Empfehlung X.200 [CCI88b] festgelegt worden. Die jeweils zugehörigen Empfehlungsbezeichnungen sind in Tabelle 1.2 beschrieben [JA90]. Die Spezifikation der Dienste der Anwendungsschicht setzt sich wiederum aus mehreren Empfehlungen für die unterschiedlichen Anwendungsbereiche zusammen.

[1] 1993 wurde die CCITT in *Telecommunication Standards Sector of the International Telecommunication Union (ITU-T)* umbenannt. In dieser Arbeit wird die Bezeichnung CCITT an Stelle von ITU-T verwendet, wenn eine Veröffentlichung unter dem Namen CCITT erfolgt ist.

Schichtbezeichnung	Spezifikationsgegenstand	Dienstspez.	Protokollspez.
Application Layer	Association Control	X.217	X.227
	Reliable Transfer	X.218	X.228
	Remote Operations	X.219	X.229
	CCR	-	-
	Directory Services	X.511	X.519
	Message Handling	X.411, X.413	X.419
	File Transfer	-	-
	Virtual Terminal	-	-
Presentation Layer		X.216	X.226
Session Layer		X.215	X.225
Transport Layer		X.214	X.224
Network Layer		X.213	X.223
Data Link Layer		X.213	-
Physical Layer		X.211	-

Tabelle 1.2: CCITT-Empfehlungsnummern

1.2 Bearbeitung der Problemstellung in bestehenden Arbeiten

Da informelle Beschreibungen mehrere Nachteile mit sich bringen, sind die FDTs (Formal Description Techniques) eingeführt worden. In [Vis88, VS87, Bri86a] werden anstelle der Nachteile der informellen Beschreibungen die folgenden Vorteile von FDTs angeführt:

- Eindeutige Spezifikation eines Standards: Informelle Spezifikationen sind wegen der Verwendung der natürlichen Sprache mehrdeutig und können daher zur Mißinterpretation einer Norm führen. Dagegen liefern formale Spezifikationen eine exakte, eindeutige Beschreibung einer Norm, sofern die Spezifikationsmethode eine eindeutige Semantik hat.

- Verifizierbarkeit von Spezifikationen: Nur bei einer formalen Beschreibung kann ein exakter Beweis der Korrektheit eines Verfeinerungsschrittes durchgeführt werden.

- Ausgangsgrundlage für einen Conformance Test: Aus einer formalen Spezifikation können Testmuster gewonnen werden.

In [Vis88] wird ein dreistufiges Verfahren zur Erstellung einer Norm vorgeschlagen:

1. Beschreiben des Normvorschlags in natürlicher Sprache und anschließende Diskussion der an der Normierung beteiligten Partner.

2. Anfügen einer formalen Beschreibung als nicht verpflichtender Anhang.

3. Ersetzen der natürlichsprachlichen Beschreibung durch die formale Beschreibung als Referenz. Der Text in natürlicher Sprache kann dann noch als Kommentar verwendet werden.

1.2.1 Aufgabenunabhängige Darstellung der Schichten

Ein erster Ansatz zur abstrakten Beschreibung aller Schichten ist in Abbildung 1.2 dargestellt. Der Diensterbringer der Schicht N wird vereinfacht als eine *blackbox* angenommen, die der nächsthöheren Schicht (N+1) Dienste zur Verfügung stellt, wobei der Diensterbringer sich selbst nur auf Dienste der nächstniedrigeren Schicht (N-1) stützt. Die Schnittstelle, an der die Dienste zur Verfügung gestellt werden, wird als Dienstzugangspunkt (Service Access Point SAP) bezeichnet. Die physikalische Verbindung wird nur in der untersten Schicht realisiert. Hierbei ist es möglich, daß die Kommunikationspartner nicht direkt miteinander verbunden sind, da gegebenenfalls Verbindungsrechner zur Datenübertragung verwendet werden. Dann spricht man von einer virtuellen Verbindung.

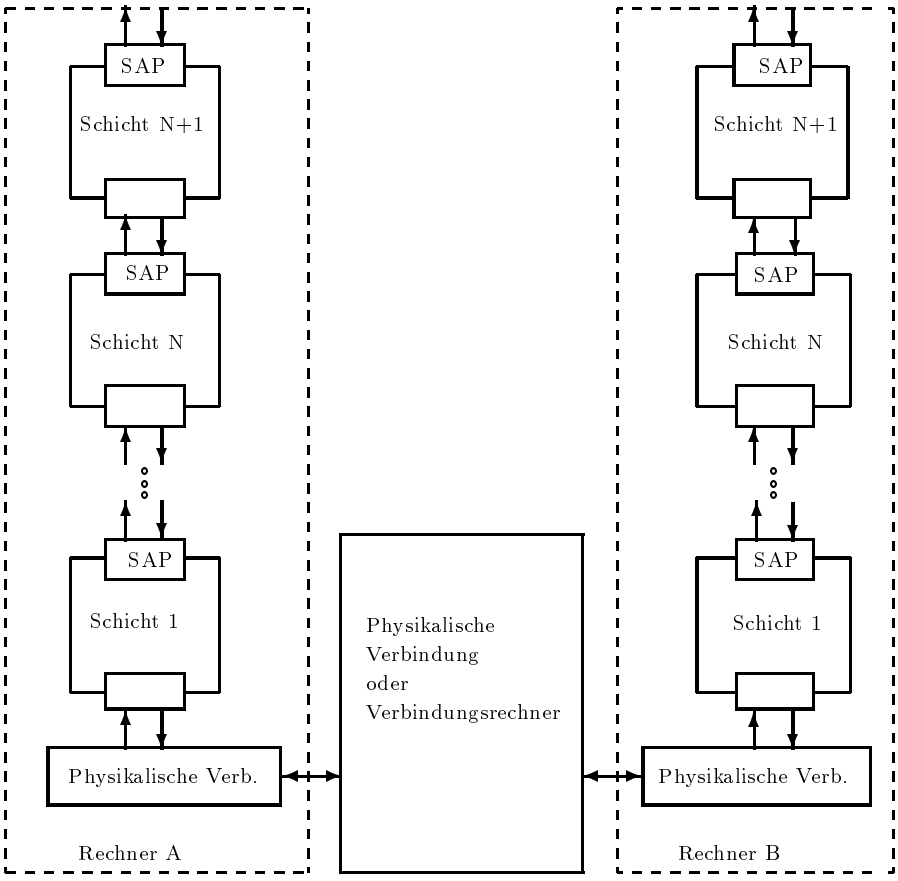

Abbildung 1.2: Abstrahiertes Schichtenmodell

Wird eine Schicht für sich betrachtet, so läßt sich eine weitere Abstraktion vornehmen [Lin83]. Dabei werden die sich meist auf verschiedenen Rechnern befindenden Kommunikationspartner einer Schicht, Diensterbringer genannt, zu einer Funktionseinheit zusammengefaßt. Dies ergibt das in Abbildung 1.3 dargestellte *blackbox*-Modell.

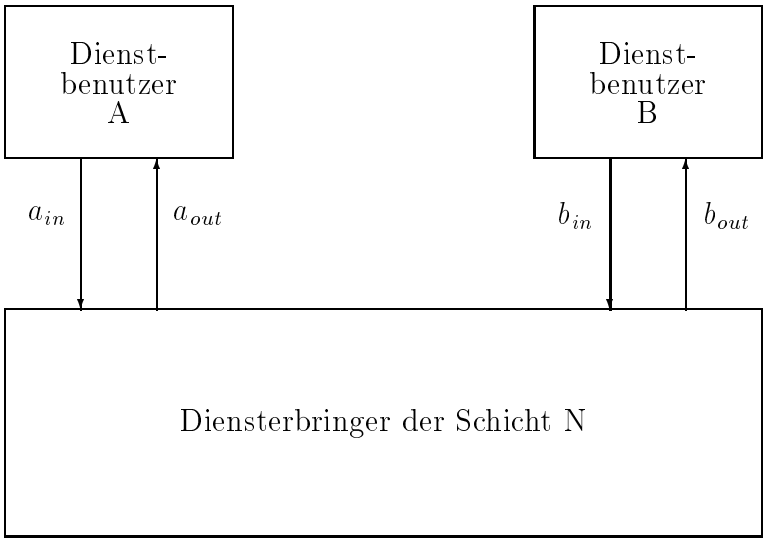

Abbildung 1.3: Abstrahierte Darstellung einer Schicht

In der Abbildung 1.3 werden nur noch die wesentlichen Merkmale einer Schicht angeführt. Diese Modellierung beschreibt den Austausch von Diensten zwischen zwei Benutzern über eine virtuelle Verbindung. Dabei bilden die SAP's, gekennzeichnet durch (a_{in}, a_{out}) und (b_{in}, b_{out}), die einzigen Schnittstellen zum Dienstbenutzer. Das Verhalten eines Diensterbringers wird auf diesem Abstraktionsniveau als Spezifikation der zeitlichen Abfolge von unteilbaren Dienstelementen (*service primitive*) aufgefaßt. Es wird spezifiziert, welche Reaktion ein Dienstelement hervorruft. Dies ist die abstrakteste Sicht eines Diensterbringers.

1.2.2 Verfeinerung der Diensterbringer

Berücksichtigt man bei der abstrakten Darstellung eines Diensterbringers (siehe Abbildung 1.3) den inneren Aufbau des Diensterbringers [JA90], so erhält man die in Abbildung 1.4 dargestellte Verfeinerung.

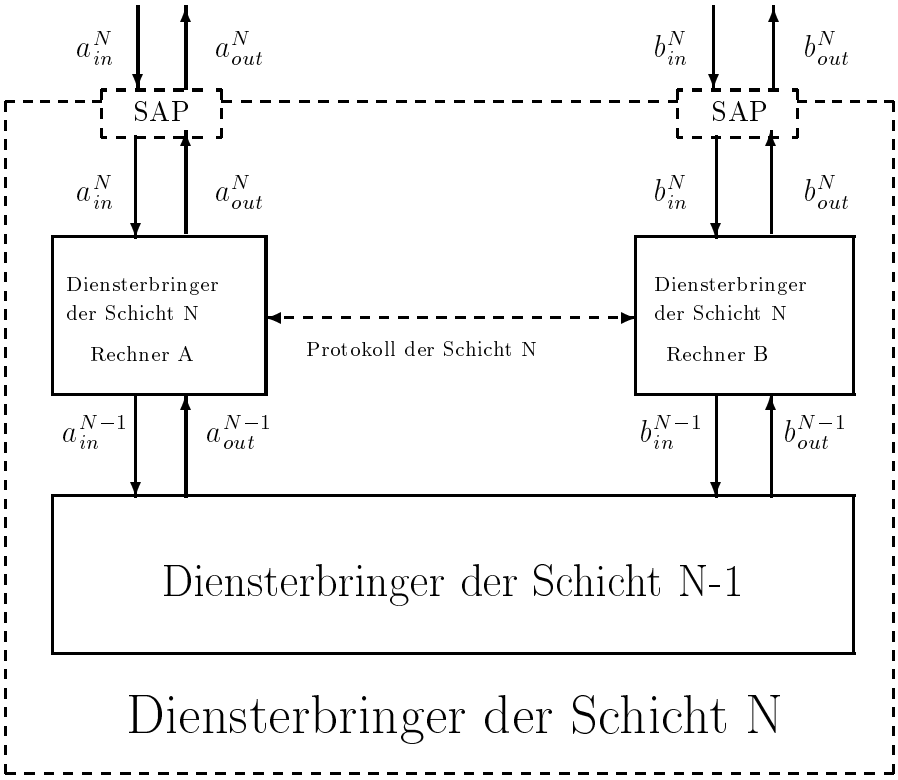

Abbildung 1.4: Abstrahierte rekursive Darstellung des Schichtenmodells

In Abbildung 1.4 wird der Diensterbringer der Schicht N durch zwei Komponenten realisiert, die sich gegebenenfalls auf unterschiedlichen Rechnern befinden. Dabei benutzen diese Komponenten den Diensterbringer der Schicht N-1 zum Austausch der Informationen. Die Regeln, denen diese Kommunikation unterliegt, werden als Protokoll der Schicht N bezeichnet.

Mit dieser Abstraktion ist die Möglichkeit der formalen Beschreibung einer beliebigen Schicht gegeben. Ebenso ist die formale Beschreibung einer Implementierung eines Diensterbringers, sowie des zugehörigen Protokolls möglich.

1.2.3 Formalisierungen im OSI Referenzmodell

Von der Standardisierungseinrichtung ISO werden zur Erstellung von Spezifikationen im OSI Schichtenmodell die Sprache Estelle [ISO87a], die auf erweiterten endlichen Automaten basiert, und die Sprache LOTOS [ISO89a], die aus der Kombination einer Prozeßalgebra mit einer Sprache zur Beschreibung von abstrakten Datentypen besteht, definiert. Nicht nur zur Anfertigung von OSI Spezifikationen wurde von CCITT die Spezifikationsprache SDL [CCI88a], die ebenso wie Estelle auf erweiterten endlichen Automaten beruht, aber zusätzlich abstrakte Datentypen enthält, eingeführt.

Die Beschreibung der Dienste ist als der abstrakteste Ansatz für die Anforderungsspezifikation einer Schicht zu wählen. In [VS87, VL86] werden dafür unter anderem folgende Argumente angegeben:

- Höchster Freiheitsgrad bei der Implementierung.

- Frühe Grundlage zur Verifikation, zum Beispiel einer Protokollbeschreibung gegenüber der Dienstbeschreibung.

Die Dienstspezifikation für die Transportschicht ist mit LOTOS in [SPLB86, BK85, ISO92] durchgeführt worden. In [SFA80] wird dazu eine sogenannte verteilte abstrakte Maschine verwendet. Die Kommunikationssteuerungsschicht wird mit LOTOS in [ISO89b] spezifiziert. Diese Dienstbeschreibungen nehmen unter Umständen Implementierungsdetails vorweg, da eine konstruktive Spezifikationstechnik verwendet wird. Außerdem werden einige Implementierungsdetails explizit festgelegt. Als Beispiel dafür dient die in [ISO92] vorgenommene Kodierung der Dienstelemente durch den Datentyp der natürlichen Zahlen. In [Got90] hingegen wird mit einer temporalen Logik, die um einen Zähloperator erweitert ist, eine deskriptive Spezifikation eines Diensterbringers durchgeführt. Jedoch erfolgt hierbei keine isolierte Betrachtung der einzelnen in den Normen verwendeten Beschreibungstechniken. Eine große Implementierungsnähe ist auch bei den Spezifikationen in den Sprachen SDL und Estelle zu beobachten, da beide auf dem Automatenkonzept beruhen.

Ansätze zur formalen Beschreibung von Protokollen sind unter anderem in [Boc90a, FO85, SM85, AF82] zu finden. In [SCB91, FO85] werden Protokolle ohne Zuhilfenahme der Dienstspezifikation entwickelt. Sehr häufig wird auf eine formale Dienstspezifikation verzichtet und nur eine Protokollspezifikation als Ausgangsbasis für weitere Verfeinerungsschritte verwendet. Somit wird die formale Technik nicht im höchsten Abstraktionsgrad eingesetzt.

Eine Protokollentwicklung mit einem automatischen Verfahren, das die durch einen endlichen Automaten modellierte Dienstspezifikation in ein Protokoll transformiert, wird in [SP91] dargestellt. Dies ist nur durch Verwendung einer konstruktiven Sprache möglich. Mit LOTOS als Spezifikationsprache wird eine halbautomatische Protokollentwicklung in [ES91, KBK89] durchgeführt. In einer eigens definierten Sprache ist dies in [BG86] vorgestellt worden. Eine Erweiterung von ML und CCS wird in [SGM90] zur nichtautomatischen Protokollentwicklung verwendet. Eine Validation, die etwaige Inkonsistenzen durch Tests aufzeigen kann, wird in [UP86] vorgestellt. Diese Techniken nehmen zwangsweise Implementierungsdetails vorweg.

Bei konstruktiven Spezifikationssprachen wird der Implementierungsbeweis im allgemeinen über das sogenannte beobachtbare Verhalten realisiert. Für LOTOS werden Beweisbeispiele in [VSvSB91, BS86] geführt. Mit den angegebenen Techniken wird aber die Korrektheit der Datenübermittlung nicht erfaßt. Somit ist die Modellierung einer Verfälschung der Daten bei der Übertragung nicht Bestandteil der Verifikation. In [Pep90] kann dies prinzipiell modelliert werden, aber dafür fehlt die feinere Aufteilung der Übertragungseinheiten in Dienstelemente.

Eine Verifikation von Spezifikationen, die mit erweiterten Automaten erstellt werden, beinhaltet nur den Ablauf von Ereignissen. Dadurch wird unter anderem die Deadlock-Freiheit zugesichert. Die bei den Diensten eingeschlossenen Parameter werden bei derartigen Verfahren nicht berücksichtigt.

1.3 Motivation und Zielsetzung für den eigenen Ansatz

Die von den Normierungsgremien festgelegten Normen bestehen zum größten Teil aus textuellen Erklärungen. Da dies, wie im Kapitel 1.2 dargestellt, wesentliche Nachteile mit sich bringt, wird in dieser Dissertation eine Methodik zur formalen Spezifikation des ISO/OSI Schichtenmodells vorgestellt. Dabei werden die formalen Methoden an frühest möglicher Stelle eingesetzt, damit die Vorteile von formalen Verfahren einen möglichst großen Wirkungsbereich haben. Es wird vor allem auf ein hohes Abstraktionsniveau bei der Spezifikation Wert gelegt, damit eine darauf aufbauende Implementierung nicht unnötig eingeschränkt wird[VL86].

Ausgangspunkt einer Protokollentwicklung ist die Beschreibung des Diensterbringers einer Schicht. Dabei wird der Diensterbringer einer Schicht als eine *blackbox* mit zwei Schnittstellen SAP zu den Dienstbenutzern betrachtet (siehe Abbildung 1.3). Das Verhalten dieses Diensterbringers wird mit Spurmengen dargestellt [BDD+93]. Eine derartige Semantik wird in den Normen nahegelegt, da sich die Erklärungen auf den Begriff der erlaubten Sequenzen von Aktionen stützen. Die Verwendung von Spuren bedeutet in der Entwicklungsmethodik für verteilte Systeme FOCUS [BDD+93] die Erstellung einer Anforderungsspezifikation, wobei die zwei Dienstbenutzer und der Diensterbringer als Komponenten betrachtet werden.

In den bisher bekannten Arbeiten werden die Eigenschaften des Diensterbringers einer Schicht als Gesamtheit betrachtet. Dies wird in [VSvSB91] als *monolithischer* Spezifikationsstil bezeichnet. Da der Wiederverwendungsgrad bei einer derartigen Spezifikation sehr gering ist, wird in dieser Dissertation eine modulare Spezifikation des Diensterbringers vorgenommen. Dies wird in [VSvSB91] als *constraint* orientiert bezeichnet. In dieser Arbeit werden Teileigenschaften des Diensterbringers erstmalig in voneinander unabhängige Beschreibungseinheiten, die dann einzeln mit einer Beschreibungstechnik spezifiziert werden, zusammengefaßt. Jeder dieser Beschreibungstechniken wird eine formale Semantik in Form einer Menge von akzeptierten Spuren gegeben. Die Semantik der unabhängigen Teile der Beschreibung wird mit einer Konjunktion kombiniert. Die Beschreibungstechniken lassen sich in schichtenunabhängige und schichtenspezifische Verfahren aufteilen. Ein Vorteil dieser modularen Beschreibung ist die hohe Wiederverwendbarkeit, da der schichtenspezifische Anteil nur von geringem Umfang ist. Insbesondere da bei den schichtenunabhängigen Beschreibungen in den Normen Teile verwendet werden, die einer formalen Beschreibung mehr oder weniger nahe kommen, ist eine Spezifikation der Diensterbringer aller Schichten nicht von Grund auf neu durchzuführen. In dieser Arbeit werden dazu Methoden zur Umwandlung der in den Normen verwendeten graphischen Techniken in eine formale Spezifikation angegeben. Ein weiterer Vorteil dieses Verfahrens ist die dadurch erfolgte formale Fundierung der in den Normen bereits verwendeten Beschreibungstechniken, die auch außerhalb des ISO/OSI Schichtenmodells eingesetzt werden.

Als schichtenunabhängige Beschreibungstechniken werden in dieser Arbeit *Time Sequence Diagrams (TSD)*, *lokale Einschränkungen (LC)* und das *Queuemodell (QM)* betrachtet, für die in der Literatur keine formale Semantik existiert. Sie sind in dieser Arbeit ausgewählt worden, da sie durch die Verwendung von graphischen Techniken einen

Ansatzpunkt für eine formale Semantik bieten. Für jede der oben angeführten Beschreibungstechniken wird eine spurbasierte Semantik [BDD+93] angegeben. Somit wird jeweils ein Bestandteil der Eigenschaften eines Diensterbringers in eine Menge von Spuren abgebildet. Die Kombination der jeweiligen Beschreibungsteile geschieht dann auf der Ebene der Semantik unter der Verwendung einer Konjunktion, die bei Spurmengen eine Schnittmengenbildung ist. Hierdurch wird in einer homogenen Basis die Semantik des Diensterbringers modular beschrieben. In den Normen ist eine derartige Modularisierung nicht explizit vorgenommen worden. In den betreffenden Teilen, in denen nur eine Beschreibungstechnik verwendet wird, wird klargestellt, daß in nachfolgenden Erklärungen weitere Einschränkungen der spezifizierten Eigenschaften erfolgen können. Es fehlt hierbei eine Erläuterung, wie das Zusammenspiel dieser Beschreibungstechniken ist.

In dieser Arbeit wird die Kombination der Beschreibungen auf der semantischen Ebene wie folgt festgelegt: Bezeichne $[\![.]\!]_{\mathcal{BT}}$ die spurbasierte Semantik der oben angeführten Beschreibungstechniken. Somit wird die von einer Dienstspezifikation erlaubte Spurmenge beschrieben durch:

$$[\![TSD_N]\!]_{\mathcal{TSD}} \cap [\![LC_N]\!]_{\mathcal{LC}} \cap [\![QM_N]\!]_{\mathcal{QM}} \cap \{t | spez_N(t)\}$$

In dieser Formel wird für die Beschreibung der Time Sequence Diagrams der Schicht N durch TSD_N die Semantik in Form einer Spurmenge mit $[\![TSD_N]\!]_{\mathcal{TSD}}$ angegeben. Mit dieser Spurmenge werden alle nach TSD_N erlaubten Abfolgen von Dienstelementen an beiden Dienstzugangspunkten erfaßt. Mit $[\![LC_N]\!]_{\mathcal{LC}}$ wird diejenige Spurmenge beschrieben, die den in dieser Arbeit als *lokale Einschränkungen* bezeichneten Bedingungen genügt. Bei den lokalen Einschränkungen wird eine Abfolge von Dienstelementen an nur einem Dienstzugangspunkt betrachtet. In den Normen werden hierzu meist Automaten verwendet. Die Einhaltung des in dieser Arbeit als *Queuemodell* bezeichneten Modells spiegelt sich in $[\![QM_N]\!]_{\mathcal{QM}}$ wider. Hierbei wird eine Abfolge von Dienstelementen bei beiden Dienstzugangspunkten beschrieben. Zur Ergänzung der Time Sequence Diagrams wird dabei detailliert auf die Parameter, mit denen zum Beispiel die Daten bei der Datenübertragung dargestellt werden, eingegangen. Mit dem Prädikat $spez_N$ wird die Einhaltung der schichtenspezifischen Eigenschaften, die von den zuvor angegebenen Beschreibungstechniken nicht erfaßt werden, sichergestellt. Ein Beispiel hierfür ist die Datenrepräsentation der Darstellungsschicht. Somit wird die Spezifikation eines Diensterbringers aus den Teilbeschreibungen TSD_N, LC_N, QM_N, sowie dem schichtenspezifischen Anteil, der durch das Prädikat $spez_N$ beschrieben wird, zusammengesetzt. Da die jeweiligen Teilbeschreibungen durch die in dieser Arbeit entwickelten Methoden zum größten Teil aus den Normen schematisch in eine formale Beschreibung umzuwandeln sind, ist nur der schichtenspezifische Anteil von Grund auf neu zu spezifizieren.

Die bisher bekannten Arbeiten beschreiben einen Diensterbringer implementierungsnah, da einerseits konstruktive Spezifikationssprachen wie zum Beispiel LOTOS, Estelle und SDL verwendet werden und andererseits einige Implementierungsdetails vorweggenommen werden. In dieser Arbeit wird hingegen eine Semantik auf einem sehr hohem Abstraktionsniveau angegeben, um eine Implementierung nicht unnötig einzuschränken.

In den Normen werden die Beschreibungstechniken so interpretiert, daß Eingabe- und Ausgabeaktionen des Diensterbringers nicht substantiell unterschieden werden. Durch das ebenfalls in den Normen nahegelegte Verwenden von Spuren wird in Kombination mit der

äquivalenten Behandlung der Ein- und Ausgabeaktionen das Verhalten der Umgebung, das heißt der beiden Dienstbenutzer, eingeschränkt. Dies ist unerwünscht[AL90], da nicht die Umgebung, die für die Eingabeaktionen ausschließlich zuständig ist, sondern die zu beschreibende Komponente, das ist in diesem Fall der Diensterbringer, das Implementierungsziel ist. Daher wird in dieser Arbeit eine formale Semantik, die die Einschränkung des Umgebungsverhaltens vermeidet, angegeben. Die Ausgangsbasis für diese Semantik sind die zuvor festgelegten Mengen von Spuren.

1.4 Die wichtigsten Ergebnisse

Durch die in dieser Arbeit vorgestellte Methodik wird das ISO/OSI Schichtenmodell und damit auch die Protokolle nach dem ISO/OSI Schichtenmodell in einfacher Weise formal spezifiziert. Die Verwendung eines modularen Spezifikationsstils ermöglicht im Gegensatz zu bisher benutzten Verfahren einen hohen Wiederverwendungsgrad der Spezifikationen bei einzelnen Schichten. Die isolierte Betrachtung einzelner Beschreibungstechniken bei der Bildung einer formalen Semantik gestattet es, die Ergebnisse dieser Arbeit auch außerhalb des ISO/OSI Schichtenmodells einzusetzen, da manche der Beschreibungstechniken allgemein bei Netzwerken benutzt werden.

Als Grundlage der Semantik der in dieser Arbeit vorgestellten Beschreibungstechniken werden Spuren verwendet. Ausgehend von der formalen Semantik werden einige Ungenauigkeiten in den Normen nachgewiesen. So werden zum Beispiel zwei Beschreibungsstile für TSDs in den Normen als äquivalent bezeichnet, obwohl, wie in dieser Arbeit nachgewiesen wird, sie eine unterschiedliche Ausdrucksmächtigkeit besitzen. Auch bei den LCs werden in den Normen zwei Beschreibungsstile eingeführt, ohne darauf hinzuweisen, daß sie von unterschiedlicher Mächtigkeit sind, wie dies in dieser Arbeit festgestellt wird.

Darüber hinaus wird eine Semantik präsentiert, die das Verhalten der Dienstbenutzer nicht mehr einschränkt. Dieser Aspekt wird in den Normen nicht betrachtet. Vielmehr ist die Deutung einiger Darstellungen der Beschreibungstechniken nicht exakt festgelegt, da die Einschränkung des Umgebungsverhaltens nicht durch ein sauberes Konzept ausgeschlossen wurde.

Die vorgestellte Semantik betrachtet unendliche Spuren als Erweiterung zu den Normen. Dies ist insbesondere im Hinblick auf Lebendigkeitseigenschaften und zum Erzielen eines hohen Abstraktionsniveaus der Spezifikationen notwendig.

Es zeigt sich zudem, daß bei einer auf einer einzigen Spurmenge basierten Semantik ohne Einschränkung des Verhaltens der Dienstbenutzer eine der in den Normen verwendeten Beschreibungstechniken, die TSDs, nicht kompositional ist. Durch die Verwendung eines Paars von Spurmengen wird die Kompositionalität der Semantik von TSDs sichergestellt.

Für die Erzeugung einer formalen Semantik für TSDs ist eine Prozeßalgebra eingeführt worden, die im Gegensatz zu den meisten anderen Prozeßalgebren unendliche Elemente auf einem sehr hohen Abstraktionsniveau beschreibt. Außerdem wird ein für TSDs gut anwendbarer, nichtdeterministischer Operator eingeführt.

Einige Beschreibungstechniken werden in dieser Arbeit ergänzt, so daß eine Spezifikation ohne Zuhilfenahme von Texten erfolgen kann. Dadurch wird die Exaktheit der Beschreibungstechniken sogar bei einem nichtformalen Gebrauch erhöht.

1.5 Aufbau der Arbeit

In Kapitel 2 erfolgt eine Einführung in die Grundlagen des ISO/OSI Schichtenmodells und des verwendeten Formalismus. Außerdem werden die bei der Modellierung verwendeten grundsätzlichen Annahmen dargestellt.

In den Kapiteln 3 bis 5 wird jede der in den Normen verwendeten Beschreibungstechniken isoliert betrachtet. In Kapitel 3 wird für Time Sequence Diagrams (TSD) eine formale Semantik definiert. Durch die Formalisierung werden einige Ungenauigkeiten in den Normen, die TSDs betreffen, aufgedeckt und anschließend behoben. Ebenso werden die einzelnen TSDs der verschiedenen Schichten formal betrachtet. Kapitel 4 beschreibt eine Formalisierung der lokalen Einschränkungen. Hierbei wird bei den zwei in den Normen angegebenen Spezifikationsmethoden eine unterschiedliche Mächtigkeit festgestellt und anhand der betreffenden Dienstnorm belegt. In Kapitel 5 wird das Queuemodell formalisiert. Da es in den Normen zum größten Teil textuell dargestellt wird, ist eine Erweiterung der informellen Darstellung um besser formalisierbare Teile notwendig. Bei der Formalisierung werden Ungenauigkeiten in den Normen offengelegt und behoben.

Kapitel 6 enthält einen formal fundierten Vergleich der zuvor betrachteten Beschreibungstechniken, bei dem sich herausstellt, daß bei der Spezifikation eines beliebigen Diensterbringers auf keine der betrachteten Beschreibungstechniken verzichtet werden kann.

Das Zusammensetzen der zuvor betrachteten Teile einer Dienstbeschreibung zur Spezifikation einer beliebigen Schicht im ISO/OSI Schichtenmodell erfolgt im Kapitel 7, in dem auch das Modularisierungsprinzip bei der Beschreibung einer beliebigen Schicht vorgestellt wird. Damit wird eine Methodik zur formalen Fundierung von beliebigen Dienstnormen eingeführt.

In den vorausgehenden Betrachtungen wird nur auf eine isolierte Punkt-zu-Punkt Verbindung eingegangen. In Kapitel 8 wird das in Kapitel 7 eingeführte Verfahren um die Beschreibung von mehreren Punkt-zu-Punkt Verbindungen erweitert. Dadurch kann ein realer Diensterbringers beschrieben werden.

Kapitel 9 gibt eine Zusammenfassung der wichtigsten Ergebnisse, sowie einen Hinweis auf die durch diese Arbeit angeregten zukünftigen Aufgaben.

Kapitel 2

Grundlagen

2.1 Grundsätzliche Definitionen im OSI Referenzmodell

Im folgenden werden grundlegende Begriffe zur Beschreibung von Rechnernetzen festgelegt. Diese Definitionen sind der ISO Norm 7498 [ISO84a] entnommen.

Definition 2.1

> Ein *reales offenes System* (Real Open System) ist eine Menge von Computern, von zugeordneter Software, von Peripheriegeräten, von Benutzern und von Übertragungsmitteln, die als Einheit Informationen verarbeiten und austauschen. Der Austausch von Informationen mit anderen realen offenen Systemen genügt den ISO/OSI Empfehlungen. □

Als Empfehlung wird eine Kombination aller diesbezüglichen Normen verstanden. Dies umfaßt das Referenzmodell [ISO84a], sowie alle in der Tabelle 1.1 angegebenen Normen.

Definition 2.2

> Ein *offenes System* (Open System) ist eine Darstellung innerhalb des Referenzmodells von Eigenschaften eines realen offenen Systems, die OSI betreffen. □

Ein offenes System ist also ein Modell eines Systems, dessen Bestandteile den OSI Standards entsprechen.

Definition 2.3

> Als *Schicht N* (Layer N) wird ein Teil in der hierarchischen Aufteilung der OSI Architektur bezeichnet, der aus Elementen besteht, die nur mit Elementen der nächsthöheren (N+1) beziehungsweise der nächstniedrigeren (N-1) Schicht interagieren. □

Definition 2.4

> Eine *(N)-Instanz* (Entity) ist ein aktives Element der Schicht N. □

Die offizielle Übersetzung von Entity ist nach DIN Instanz. Jedoch wird in deutscher Literatur oft der Begriff Einheit verwendet. Im Sprachgebrauch von FOCUS [BDD+93] entspricht eine Instanz einem Agenten.

Definition 2.5

Eine *(N) Partnerinstanz* (Peer Entity) ist eine Instanz der gleichen Schicht. □

Definition 2.6

Ein *(N) Dienst* (Service) beschreibt die Fähigkeit der Schicht N, gewisse Aufgaben für die Schicht N+1 zu erfüllen, die an den Grenzen der Schicht N zur Schicht N+1 bereitgestellt werden. □

Hierbei ist zu berücksichtigen, daß implizit auch alle darunterliegenden Schichten eingeschlossen werden, da eine Kommunikation nur über diese erfolgen kann.

Definition 2.7

Ein *(N) Dienstzugangspunkt* (Service Access Point oder abgekürzt SAP) ist die Schnittstelle zwischen einer Instanz der Schicht N und einer Instanz der Schicht N+1, an der die Dienste der Schicht N zur Verfügung gestellt wird. □

Definition 2.8

Ein *Dienstelement* (Service Primitive) ist eine abstrakte, implementierungsunabhängige Interaktion eines Dienstbenutzers mit einem Diensterbringer. □

Die Definition der Dienstelemente ist nicht Bestandteil von [ISO84a, CCI88b], sondern wurde aus [ISO87b, CCI88d] entnommen. Die Dienstelemente sind die kleinsten Kommunikationseinheiten, die zwischen den Instanzen zweier benachbarter Schichten ausgetauscht werden. Nach [Ker92] ist eine Interaktion in diesem Fall ein Kommando zur Inanspruchnahme eines Dienstes oder die Ergebnismeldung eines Dienstes. Als Beispiel hierfür dient der Wunsch eines Dienstbenutzers, die Nutzdaten x an seinen Kommunikationspartner zu senden. Dies wird mit dem Dienstelement $DATrequest(x)$ dargestellt. Nach [Ker92] besitzt ein Dienstelement zwei Arten von Informationen. Zum einen wird durch die Bezeichnung des Dienstelements, im Beispiel $DATrequest$, die Art der Interaktion gekennzeichnet. Zum anderen sind die jeweiligen Nutzdaten als Parameter angegeben.

Definition 2.9

Ein *Protokoll* (Protocol) der Schicht N ist eine Menge von (syntaktischen und semantischen) Regeln und Aufbaubeschreibungen, die das Kommunikationsverhalten von (N)-Instanzen beschreibt. □

In [ISO84a] wird noch genauer auf den Aufbau der (N)-Instanzen eingegangen. Dabei wird das Kommunikationsverhalten auf der Basis von Aktivitäten einer (N)-Instanz, den sogenannten (N)-Funktionen, beschrieben. Diese feinere Unterteilung ist aber in dieser Dissertation nicht von Bedeutung. In vielen anderen Definitionen fließt auch das Zeitverhalten explizit in die Definition des Protokolls ein [JA90, Sta90]. Dadurch wird dieser Bestandteil der Kommunikation hervorgehoben.

Definition 2.10

Eine *(N)-Verbindung* ist eine für die Datenübertragung durch die (N)-Schicht zwischen zwei oder mehreren (N+1)-Instanzen hergestellte Beziehung. □

Definition 2.11

Ein *(N)-Verbindungsendpunkt* ist der Endpunkt einer (N)-Verbindung innerhalb eines (N)-Dienstzugangspunkts. □

2.2 Grundlagen des verwendeten Formalismus

Die in dieser Arbeit dargestellte Semantik basiert im wesentlichen auf den in der Spezifikationsmethodik für verteilte Systeme FOCUS [BDD+93] verwendeten Formalismen. In diesem Kapitel wird eine kurze Einführung in die benötigten Konzepte vorgenommen. Für eine detailliertere Beschreibung sei auf [BDD+93] verwiesen.

Eine *Aktion* beschreibt ein unteilbares und punktförmiges, bezogen auf die zeitliche Ausdehnung, Ereignis in einem verteilten System. Somit sind Anfang und Ende eines Ereignisses nicht unterscheidbar, da sie immer zusammenfallen. Eine *Spur* ist eine endliche oder unendliche Sequenz von Aktionen. Mit einer Spur kann somit die Kommunikationsgeschichte eines Systems beschrieben werden. Hierbei werden in Anlehnung an die Dienstnormen die übertragenen Daten als Parameter der Aktionen beschrieben. Das *Verhalten eines verteilten Systems* wird als eine Menge von Spuren dargestellt. Jede einzelne Spur beschreibt eine erlaubte Abfolge von Aktionen. In einer Spurmenge werden alle erlaubten Abfolgen von Aktionen gleichberechtigt zusammengefaßt.

Definition 2.12

Für eine Menge von Aktionen S wird festgelegt[BDD+93]:

- ϵ bezeichnet die leere Spur
- & bezeichnet den Konstruktor für Spuren. Somit ist $a\&s$ diejenige Spur, bei der der Aktion a eine Spur s folgt.
- S^* bezeichnet die Menge der endlichen Spuren über S
- S^∞ bezeichnet die Menge der unendlichen Spuren über S
- $S^\omega := S^* \cup S^\infty$ bezeichnet die Menge aller Spuren über S
- Seien $a_1, a_2, \ldots, a_n \in S$, dann wird $<a_1, a_2, \ldots, a_n>$ als Abkürzung für $a_1\&a_2\& \ldots \&a_n\&\epsilon$ verwendet
- $s \circ t$ beschreibt die Konkatenation der Spuren s und t. Falls $s \in S^\infty$ gilt: $s \circ t = s$ Die Konkatenation ist induktiv über den Aufbau von s definiert.
- $s \sqsubseteq t$ beschreibt, daß s ein Präfix von t ist. Also $\exists s'.\ s \circ s' = t$ gilt. Man beachte, daß $\sqsubseteq$ eine partielle Ordnung darstellt.
- $\#s$ bezeichnet die Länge einer Spur s, die auch ∞ (unendlich) sein kann.

- Die Filteroperation $a©s$ liefert als Ergebnis die Teilspur von s, die nur das Element a beinhaltet. So gilt zum Beispiel: $a©<a,c,a,b,a> = <a,a,a>$. Der erste Operand kann zur Vereinfachung auch als Menge verwendet werden: $\{a,c\}©<a,c,a,b,a> = <a,c,a,a>$
- a^i bezeichnet eine nur aus der Aktion a bestehende Spur der Länge i. Somit gilt: $(\forall x.\ x©a^i \neq \epsilon \Rightarrow x = a) \wedge \#a^i = i$
- Bei einer Spur t wird der *Zugriff auf das i-te Element von t* definiert als: $t[i]$. Für $\#s \leq i$ liefert $t[i]$ das undefinierte Element $\perp$.
- Die Elementrelation $\in$ wird auf Spuren wie folgt festgelegt: Eine Aktion a ist Element einer Spur t genau dann, wenn $a \in t := \exists i.\ t[i] = a$ gilt.
- Eine *Kette* über einer beliebigen partiellen Ordnung $\sqsubseteq$ und einer Menge D ist eine Teilmenge $C \subseteq D$, für die gilt: $\forall c_1, c_2 \in C.\ c_1 \sqsubseteq c_2 \vee c_2 \sqsubseteq c_1$ Dies wird mit dem Prädikat *is_chain* dargestellt.
- Die *kleinste obere Schranke* $\bigsqcup S$ *einer Kette* S, die Teilmenge von D ist, ist für eine beliebige partielle Ordnung $\sqsubseteq$ definiert als:

$$(\forall d \in S.\ d \sqsubseteq \bigsqcup S) \wedge (\exists ub \in D.\ (\forall d \in S.\ d \sqsubseteq ub) \Rightarrow \bigsqcup S \sqsubseteq ub))$$

- Der *kleinste Fixpunkt* eines Funktionals f wird als $\mathsf{fix}f$ bezeichnet. Es gilt:

$$f(\mathsf{fix}f) = \mathsf{fix}f \wedge (\forall g.\ f(g) = g \Rightarrow \mathsf{fix}f \sqsubseteq g)$$

□

Mit diesen grundlegenden Definitionen lassen sich Spuren beschreiben. Als Sprache zur Beschreibung von Eigenschaften wird in dieser Arbeit Prädikatenlogik höherer Ordnung verwendet. Eine Eigenschaft eines verteilten Systems ist sowohl als Prädikat $P(t)$, als auch als Menge von Spuren $\{t \in Act^\omega | P(t)\}$ äquivalent beschreibbar. In dieser Arbeit werden Prädikate in Lambdanotation beschrieben: $\forall t \in Act^\omega.\ P(t)$ Dies entspricht der Darstellung von $\forall t \in Act^\omega :\ P(t)$ Es werden grundsätzlich zwei Arten von Eigenschaften bei verteilten Systemen unterschieden [AS87, Web92]:

Definition 2.13

Eine *Sicherheitseigenschaft* ist ein Prädikat $P(t)$ über Act^ω, für das gilt:

$$\forall t \in Act^\omega.\ (P(t) \Leftrightarrow \forall s \in Act^*.\ s \sqsubseteq t \Rightarrow P(s))$$

□

Definition 2.14

Eine *Lebendigkeitseigenschaft* ist ein Prädikat $P(t)$ über Act^ω, für das gilt:

$$\forall s \in Act^*.\ \exists t \in Act^\omega.\ P(s \circ t)$$

□

Es existiert eine alternative Definition der Lebendigkeitseigenschaft, bei der eine Lebendigkeitseigenschaft modulo einer Sicherheitseigenschaft betrachtet wird:

Definition 2.15

Sei S eine Sicherheitseigenschaft, dann beschreibt L eine *Lebendigkeitseigenschaft bezüglich* S, wenn gilt:

$$\forall s \in Act^*.\ (S(s) \Rightarrow \exists t \in Act^\omega.\ L(s \circ t) \wedge S(s \circ t))$$

□

Die folgenden Definitionen sind aus [Web92] entnommen:

Definition 2.16

Für eine Spurmenge T ist der *Präfixabschluß* $PRE(T)$ definiert als:

$$PRE(T) := \{t_p |\ \exists t \in T.\ t_p \sqsubseteq t\}$$

□

Definition 2.17

Für eine Spurmenge T ist der *Approximationsabschluß* $LIM(T)$ definiert als:

$$LIM(T) := \{\bigsqcup t |\ \exists C.\ C \subseteq T \wedge is_chain(C)\}$$

□

Darauf aufbauend erhält man:

Definition 2.18

Eine Spurmenge T ist genau dann gegenüber *Präfixbildung abgeschlossen*, wenn gilt:

$$T = PRE(T)$$

□

Definition 2.19

Eine Spurmenge T ist genau dann gegenüber *Approximation abgeschlossen*, wenn gilt:

$$T = LIM(T)$$

□

2.3 Grundsätzliche Modellierungsannahmen

Die Dienstelemente der Normen des ISO/OSI Schichtenmodells werden als unteilbare und punktförmige Aktionen modelliert. Somit besitzen die Dienstelemente keine zeitliche Ausdehnung. Ein Dienstelement beschreibt nach den ISO Normen [ISO84a, ISO87b] beziehungsweise nach den entsprechenden CCITT Empfehlungen [CCI88b, CCI88d] die Abstraktion einer Interaktion, die an den Dienstzugangspunkten sichtbar ist. Genauergenommen verbirgt sich hinter einem Dienstelement eine Sendeaktivität eines Dienstbenutzers und eine Empfangsaktivität eines Diensterbringers, oder umgekehrt. Diese Interaktion wird jedoch im ISO/OSI Schichtenmodell zu einer beobachtbaren Aktion abstrahiert. Die Annahme von punktförmigen Aktionen ist ebenso wie in vielen anderen Arbeiten [Boc90b, Got90, SGM90, ISO91b, ISO89b, ISO92, Hog89, Tur93, BHS91] getroffen worden, da sie nicht im Konflikt mit dem ISO/OSI Schichtenmodell steht. Die Berücksichtigung einer zeitlichen Ausdehnung von Dienstelementen würde nur die formalen Spezifikationen unnötig komplizieren. Eine derartige Vorgehensweise könnte, falls überhaupt, nur auf der Ebene der Hardwarebeschreibung erforderlich sein. Diese ist bei der Dienstbeschreibung im ISO/OSI Schichtenmodell vor den Dienstbenutzern verborgen, und somit nicht Spezifikationsgegenstand.

Parallelität wird in dieser Arbeit mit der Interleaving-Sicht modelliert. Dies ist nach [Hoa85] folgendermaßen festgelegt:

> Imagine there is an observer with a notebook who watches the process and writes down the name of each event as it occurs. We can validly ignore the possibility that two events occur simultaneously; for if they did, the observer would still have to record one of them first and then the other, and the order in which he records them would not matter.

Mit der Interleaving-Sicht wird die für einen Beobachter sichtbare Aktionsfolge bei zum Beispiel zwei gleichzeitigen Aktionen durch zwei Spuren, bei denen einmal die eine Aktion der anderen folgt und umgekehrt, modelliert. Dies ist natürlich in der Verwendung von punktförmigen Aktionen begründet. Basierend auf der Interleaving-Sicht wird in dieser Arbeit das Verhalten eines Diensterbringers mit einer Menge von Spuren über den Dienstelementen beschrieben. Hierbei ist der Diensterbringer wegen der zuvor aus [Hoa85] angeführten Gründe ausreichend genau beschrieben. Der in dieser Arbeit verwendete Spezifikationsansatz läßt sich, falls es trotzdem erwünscht ist, jederzeit dahingehend erweitern, daß echte Nebenläufigkeit beschreibbar wird. Hierfür sind anstelle von Spuren von Dienstelementen nur Spuren von Mengen von Dienstelementen zu verwenden. Innerhalb einer Menge werden gleichzeitige Dienstelemente beschrieben. Dies ist wegen der auf die zeitliche Ausdehnung bezogenen Punktförmigkeit der Aktionen ohne Problem durchzuführen. Eine derartige Vorgehensweise ist bei der Spezifikation unterlassen worden, da sie den Formalismus nur unnötig kompliziert.

Kapitel 3

Formalisierung von Time Sequence Diagrams

Zur Beschreibung der zeitlichen Abfolgen der Dienstelemente an den Dienstzugangspunkten[1] einer Schicht werden von ISO und CCITT in den betreffenden Normen *Time Sequence Diagrams* verwendet. Im folgenden werden diese als *TSDs* bezeichnet. TSDs werden bei der Spezifikation der Dienste aller Schichten bis auf die Darstellungsschicht benutzt. In ihr wird nur durch einen Verweis auf die TSDs der darunterliegenden Schicht, der Kommunikationssteuerungsschicht, Bezug genommen. Darauf aufbauend werden Unterschiede und Gemeinsamkeiten textuell dargestellt.

Die Interpretation von TSDs ist ursprünglich in einem Teil des technischen Berichts der ISO TR 8509 [ISO87b] festgelegt worden. Später folgte dann im Rahmen der Harmonisierung der Normen die CCITT Empfehlung X.210 [CCI88d]. Die beiden Definitionen unterscheiden sich nur in einigen Details, auf die in dieser Arbeit an geeigneter Stelle hingewiesen wird. Zur Zeit erfolgt eine Neudefinition in der ISO-Norm 10731 [ISO91a, ISO94]. Da diese Norm noch nicht verabschiedet ist, und sich die vorhandenen Dienstspezifikationen nur auf [ISO87b, CCI88d] stützen, wird in der vorgestellten Arbeit hauptsächlich auf letztere Bezug genommen.

Da bei der Dienstbeschreibung des ISO/OSI Schichtenmodells in den jeweiligen Normen, sowie bei der Definition der TSDs [ISO87b, CCI88d] nur TSDs mit genau zwei Kommunikationspartnern verwendet werden, beschränkt sich die folgende formale Fundierung ebenfalls darauf. In der Neufassung der Definition [ISO91a] werden im Hauptteil der Norm zwar nur TSDs mit mehr als zwei Kommunikationspartnern beschrieben, in [ISO94], der aktuelleren Version von [ISO91a], sind hingegen noch TSDs mit genau zwei Kommunikationspartnern vorhanden. Eine Erweiterung der vorgestellten Semantik auf mehrere Kommunikationspartner dürfte problemlos möglich sein.

[1] Da der Dienstzugangspunkt (SAP) die physikalische Schnittstelle eines Diensterbringers beschreibt, werden von einem SAP unter Umständen mehrere logische Verbindungen eingeschlossen, für die der Begriff *Verbindungsendpunkt* (Connection End Point, CEP) in den Normen eingeführt worden ist. In den folgenden Spezifikationen wird SAP mit CEP gleichgesetzt. Somit besitzt zur Vereinfachung jeder SAP genau einen CEP. Diese Vereinfachung ist gerechtfertigt, da der Einsatz eines einfachen Filteroperators dieses Verfahren verallgemeinert. Ein derartiges Verfahren wird in Kapitel 8 formal eingeführt. Dabei werden alle Dienstelemente, die eine Verbindung betreffen, herausgefiltert. Ein Prädikat berücksichtigt dann genau die zu einer Verbindung gehörenden Dienstelemente.

Die Definition der TSDs erfolgt in [ISO87b, CCI88d] auf äußerst knappe Weise in rein textueller Form. TSDs werden deshalb höchstens als semiformale Beschreibungstechnik bezeichnet, unter anderem in [BHS91], da bisher keine formale Semantik angegeben wurde. Die graphische Darstellung von TSDs ist in [ISO87b, CCI88d] nicht mit einer abgegrenzten Syntax klar festgelegt worden. Es werden nur zwei Beispiele für TSDs angegeben. Dazu kommt, daß in einer Dienstbeschreibung normalerweise eine Menge von TSDs verwendet wird, aber über die Komposition von TSDs in [ISO87b, CCI88d] keine Aussage getroffen wird. Die Festlegung von Eigenschaften der Komposition von TSDs ist ein wesentlicher Bestandteil der folgenden Kapitel. In diesem Kapitel wird eine formale Semantik für TSDs entwickelt. Diese Semantik ist nur ein Vorschlag, da sie von keiner Standardisierungseinrichtung offiziell autorisiert ist. Sie ist aus bestehenden Beispielen sowie Normen validiert worden. Ein wesentlicher Vorteil einer derartigen Semantik ist, daß mit ihr ein Ausgangspunkt für eine fundierte Diskussion über die Interpretation von TSDs geschaffen wird.

3.1 Informelle Darstellung von TSDs

TSDs sind ein graphisches Beschreibungsmittel zur Darstellung von Abfolgen von Dienstelementen bei der Dienstspezifikation. Sie beschreiben erlaubte Abfolgen von Ereignissen an den Dienstzugangspunkten. Hierbei werden sowohl Abfolgen von Ereignissen an einem einzelnen Dienstzugangspunkt zwischen Diensterbringer und Dienstbenutzer, als auch zwischen Partnerbenutzern beschrieben.

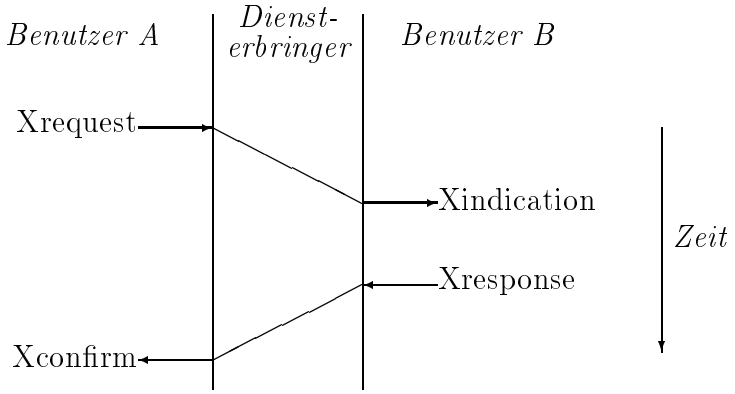

Abbildung 3.1: Prinzipieller Aufbau eines TSDs

In Abbildung 3.1 ist der prinzipielle Aufbau eines TSDs dargestellt. Jedes TSD wird durch zwei vertikale Linien, die die Dienstzugangspunkte[2] darstellen, in drei Felder aufgeteilt. In diesen Feldern wird eine Zuteilung der Dienstelemente auf den jeweiligen Dienstbenutzer A und B vorgenommen, indem die Dienstelemente auf der zugehörigen Benut-

[2]In [ISO87b, CCI88d] werden hier explizit Dienstzugangspunkte, und nicht Verbindungsendpunkte genannt. Auch aus diesem Grund ist die Vereinfachung von Verbindungsendpunkten zu Dienstzugangspunkten vorgenommen worden.

zerseite eingetragen werden. Die Richtung der Dienstelemente wird durch einen Pfeil angedeutet. Der Verlauf der Zeit ist generell von oben nach unten dargestellt. Somit ereignet sich zum Beispiel ein Xrequest vor einem Xconfirm. Es wird außerdem die zeitliche Abfolge von Dienstelementen an unterschiedlichen Dienstzugangspunkten durch eine geneigte Linie im Diensterbringerteil beschrieben. Somit ereignet sich ein Xrequest vor einem Xindication. Eine vorgeschriebene zeitliche Abfolge von Aktionen wird in dieser Arbeit als *zeitliche Präferenz* bezeichnet.

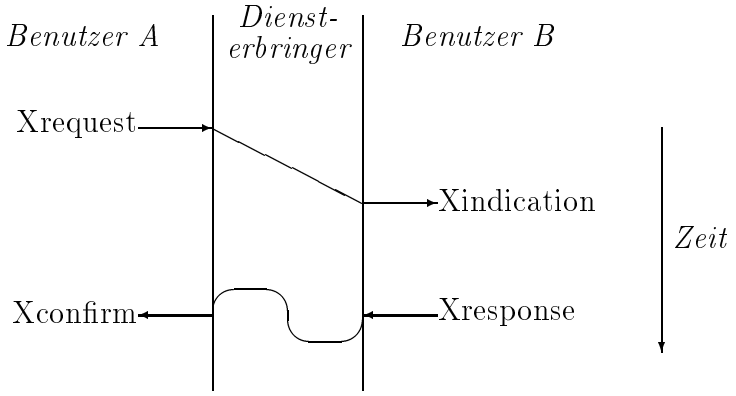

Abbildung 3.2: TS Diagramm mit unabhängigen Dienstelementen

In Abbildung 3.2 wird ein weiteres Grundelement von TSDs eingeführt, das Tildesymbol $\sim$, mit dem das Fehlen einer zeitlichen Abhängigkeit zwischen zwei Dienstelementen an unterschiedlichen Dienstzugangspunkten angedeutet wird. So wird durch das in Abbildung 3.2 dargestellte TSD kein Aussage über eine zeitliche Abhängigkeit der Dienstelemente Xresponse und Xconfirm getroffen. Da keine zeitliche Abhängigkeit besteht, ist es nach [ISO87b, CCI88d] erlaubt, das Symbol $\sim$ wegzulassen. Zur Verdeutlichung sollte dieses Symbol aber verwendet werden. Die zeitliche Präzedenz der Dienstelemente wird bei oben angeführten Darstellungen im Diensterbringer modelliert. Bei der Darstellung von TSDs entfällt die in den Abbildungen 3.1 und 3.2 kursiv dargestellte Beschriftung, die als Erklärungshilfe eingefügt wurde.

In [ISO87b, CCI88d] ist vermutlich aus historischen Gründen eine alternative Darstellungsweise für TSDs eingeführt worden, in der der Zeitverbrauch durch die Neigung der Pfeile bei den Dienstbenutzern angedeutet wird.

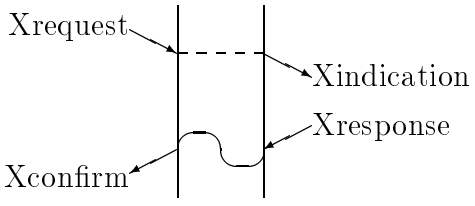

Abbildung 3.3: Zweite Darstellungsweise von TSDs

In dieser Arbeit wird der Stil 1, aus Abbildung 3.2, als *diensterbringerorientierter* Stil, der Stil 2, aus Abbildung 3.3, als *dienstbenutzerorientierter* Stil bezeichnet. Das einzige Unterscheidungskriterium ist der Ort, an dem der Zeitverlauf modelliert wird. Ansonsten sollten beide Darstellungsweisen als äquivalent betrachtet werden, da definiert wird, daß kein Unterschied bei der Abfolge von Dienstelementen auf die Verwendung der jeweiligen Stile zurückzuführen ist [CCI88d]:

> Each OSI Layer Service definition may adopt either of these styles for use in time-sequence-diagrams. No difference in the sequence of primitives being described is implied by the style used.

Bei der Interpretation von TSDs sind einige Definitionen nützlich.

Definition 3.1

> Eine durch ein TSD dargestellte, mögliche Abfolge von Dienstelementen wird als *Szenario eines TSDs* bezeichnet. □

Ein Szenario beschreibt also eine mögliche zeitliche Ordnung einer Abfolge von Dienstelementen. Ein Szenario wird semantisch mit einer Spur von Aktionen beschrieben. Ein TSD hingegen stellt unter Umständen mehrere Szenarien dar.

Definition 3.2

> Die *szenarische Interpretation von TSDs* beschreibt den einmaligen Ablauf eines mit TSDs dargestellten Sachverhalts. □

Bei der szenarischen Interpretation wird die eventuelle Wiederholung eines TSDs nicht betrachtet.

Die durch die Abbildungen 3.1 und 3.2 beschriebenen Szenarien umfassen jeweils vier Dienstelemente. Es treten auch TSDs mit einer geringeren Anzahl von Dienstelementen auf. Dabei werden obige Konventionen beibehalten.

Definition 3.3

> Die *vollständige Interpretation von TSDs* beschreibt eine beliebige Anzahl von Wiederholungen der Szenarien eines oder mehrerer TSDs. □

Die Unterscheidung zwischen szenarischer und vollständiger Interpretation eines TSDs wird in dieser Arbeit explizit eingeführt, da sie bei der exakten Definition der Semantik von TSDs erforderlich ist. In den Normen ist diese Unterscheidung nicht vorhanden.

3.2 Ungenauigkeiten in den Normen bei der Verwendung von TSDs

In diesem Kapitel werden einige Ungenauigkeiten, die bei der Betrachtung der Definition und der Anwendung von TSDs aufgefallen sind, angegeben. So wird anhand eines Beispiels die unterschiedliche Mächtigkeit beider Stile zur Darstellung von TSDs belegt.

Nach der Definition aus [CCI88d], die im vorigen Unterkapitel zitiert wird, sind die beiden Stile als äquivalent zu betrachten. Bei genauerer Untersuchung der beiden Darstellungsweisen stellt man jedoch fest, daß der diensterbringerorientierte Stil mächtiger ist als der dienstbenutzerorientierte Stil. Dies wird durch das TSD im diensterbringerorientierten Stil aus Abbildung 3.4 belegt, das mit den in [CCI88d] beschriebenen Mitteln im dienstbenutzerorientierten Stil nicht äquivalent dargestellt werden kann. Dieses TSD tritt in dieser Form in den Normen nicht auf, aber eine Verallgemeinerung davon, bei der die Aktionen mit einer Tilde verbunden sind, wird in den Normen zur Darstellung eines Verbindungsabbruchs durch den Diensterbringer verwendet. Das TSD aus Abbildung 3.4 ist zum Beispiel zur Modellierung eines Verbindungsabbruchs, der beim linken Dienstbenutzer zuerst angezeigt wird, verwendbar.

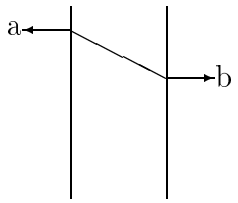

Abbildung 3.4: Im dienstbenutzerorientierten Stil nicht darstellbares TSD

In dem in Abbildung 3.4 dargestellten TSD wird eine Abfolge von zwei Ausgabeaktionen beschrieben, wobei die Ausgabeaktion a vor der Ausgabeaktion b erfolgen soll. Wird versucht, dieses TSD im dienstbenutzerorientierten Stil darzustellen, so ergibt sich ein Konflikt aus der Darstellung des Zeitverbrauchs, sowie der Ein- und Ausgabeorientierung. Im folgenden wird nachgewiesen, daß für das TSDs aus Abbildung 3.4 keine äquivalente Darstellung im dienstbenutzerorientierten Stil mit den in den Normen zur Verfügung gestellten Mitteln existiert.

Abbildung 3.5: Unbefriedigende Darstellung des problematischen TSDs im dienstbenutzerorientierten Stil

Einen ersten Ansatz zur Darstellung des TSDs aus Abbildung 3.4 im dienstbenutzerorientierten Stil ergibt das linke TSD aus Abbildung 3.5. Dieses TSD spiegelt jedoch nicht genau die Abfolgen des TSDs in Abbildung 3.4 wider, da die gewünschte zeitliche Reihenfolge der Ausgabeaktionen nicht mehr gewährleistet wird. Ein weiterer Lösungsansatz ist die einfache Umkehrung des Pfeils, der zum Ereignis a gehört, im rechten TSD

aus Abbildung 3.5. Bei diesem Ansatz tritt jedoch ein Konflikt mit dem Zeitverlauf auf. Dieses TSD legt einen negativen Zeitverbrauch im Benutzer *A* nahe. Da dies jedoch nicht realistisch ist, ist ein derartiges Konstrukt in den Normen nicht vorgesehen.

Abbildung 3.6: Unbefriedigende Darstellung des problematischen TSDs im dienstbenutzerorientierten Stil

Das in Abbildung 3.6 links dargestellte TSD zeigt einen Ansatz, in dem jedoch die Aktivität des Dienstelements *a* nur auf den Diensterbringer beschränkt wird. Außerdem wird die Bedingung verletzt, daß der Zeitverbrauch nur bei den Dienstbenutzern stattfindet. Das rechte TSD aus Abbildung 3.6 stellt den zeitlichen Sachverhalt korrekt dar, ist jedoch in [CCI88d] nicht vorgesehen, da die gestrichelte Linie nur zwischen den vertikalen Linien, die die jeweiligen SAPs beschreiben, verwendet wird. Durch ein Entfallen der Verlängerung der gestrichelten Linie wird eine zeitliche Beziehung zwischen den Ausgabeaktionen schwer erkennbar.

Zusammenfassend läßt sich feststellen, daß keine der präsentierten Lösungen zufriedenstellend ist. Das in Abbildung 3.4 dargestellte TSD läßt sich mit den in [CCI88d] angegebenen Mitteln nicht eindeutig in die dienstbenutzerorientierte Darstellungsweise bringen. Eine Behebung dieser Unstimmigkeit der Norm wäre der Ausschluß eines derartigen TSDs. Dies erfolgt in [CCI88d] nicht, da die Klasse der verwendbaren TSDs nicht explizit eingeschränkt wird. Es wird nur sehr vage auf die Verwendbarkeit im OSI Schichtenmodell hingewiesen. Eine naheliegende Lösung ist der Verzicht auf den dienstbenutzerorientierten Stil, wie dies auch im folgenden vorgeschlagen wird.

In [ISO87b] wird die Äquivalenz der beiden Darstellungsweisen vorsichtiger beschrieben. Zum einen wird der diensterbringerorientierte Stil als "logisch korrekte Darstellung" bezeichnet, zum anderen wird nur die Absicht festgestellt, daß beide Stile die gleiche Bedeutung haben sollen. Hierbei wird außerdem nur auf die zwei Beispiel-TSDs, und nicht auf die Menge aller TSDs, die im OSI Schichtenmodell verwendet werden, Bezug genommen:

> Both presentations are intended to convey the same basic meaning.

Die unterschiedliche Mächtigkeit wird in [ISO87b] ebenfalls nicht erwähnt.

Eine weitere Ungenauigkeit, die auf der Verwendung des dienstbenutzerorientierten Stils beruht, wird anhand eines TSDs belegt, das in der Definition der Bitübertragungsschicht [ISO88c, CCI88e] vorkommt. Es wird dabei eine zeitliche Präzedenz zwischen zwei Dienstelementen nahelegt, obwohl diese nicht vorhanden ist.

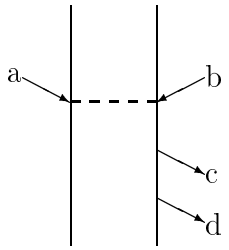

Abbildung 3.7: TSD der Bitübertragungsschicht

Die in Abbildung 3.7 verwendeten Dienstelemente werden abgekürzt. Bei dieser Darstellung wird über die zeitliche Reihenfolge der Aktionen a und b keine Aussage getroffen, da der Zeitverlauf lokal in den Dienstbenutzern dargestellt wird. Eine Interpretation der Abbildung 3.7 ergibt, daß die Aktion a sich vor c und b sich vor c ereignet.

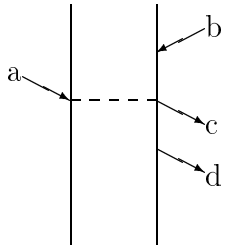

Abbildung 3.8: Modifizierte Darstellung des TSDs aus Abbildung 3.7

Bei dem in Abbildung 3.8 dargestellten TSD wird berücksichtigt, daß in Abbildung 3.7 zwischen den Aktionen a und b keine zeitliche Abhängigkeit beschrieben wird. Das TSD aus Abbildung 3.7 täuscht jedoch eine zeitliche Abhängigkeit zwischen den Ereignissen a und b vor, da eine Verbindung zwischen den beiden Dienstelementen existiert. Diese Ungenauigkeit der Darstellung ist durch die Verwendung des dienstbenutzerorientierten Stils entstanden, da bei diesem Stil ein Konflikt aus der Darstellung des Zeitverbrauchs sowie, der Ein- und Ausgabeorientierung der Ereignisse besteht. Im diensterbringerorientierten Stil ist eine derartige Ungenauigkeit nicht möglich. Aus diesem Grund wird in dieser Arbeit empfohlen, daß die Abbildung 3.8 anstelle von Abbildung 3.7 in [ISO88c, CCI88e] verwendet werden sollte.

Die höhere Mächtigkeit der Darstellung im diensterbringerorientierten Stil, sowie die für eine spätere Implementierung höhere Plausibilität durch die Modellierung des Zeitverbrauchs im Diensterbringer führt dazu, daß dieser Stil in dieser Arbeit bevorzugt wird.

3.3 Formale Semantik mit Einschränkung des Umgebungsverhaltens

Da in den Dienstnormen und in [ISO87b, CCI88d] TSDs nur zur Beschreibung von möglichen Abfolgen von Aktionen verwendet werden und hierbei keine Hinweise zur unterschiedlichen Behandlung von Ein- und Ausgabeaktionen erfolgen, ist die Definition einer formalen Semantik, die Ein- und Ausgabeaktionen nicht substantiell unterscheidet, naheliegend. In diesem Unterkapitel werden zuerst die semantischen Konzepte von TSDs erläutert. Im Anschluß daran wird die Sprache Time Sequence Diagram Description Language (TSD-DL) zur textuellen Beschreibung von TSDs eingeführt. Danach wird eine formale Semantik für TSD-DL angegeben, und eine Methode zur Umwandlung von graphischen TSDs in die TSD-DL-Ausdrücke angeführt.

3.3.1 Konzepte zur Interpretation von TSDs

Die Interpretation einzelner TSDs mit einer Menge von Spuren ist naheliegend, da sie nur eine Menge von Szenarien beschreiben. Dadurch wird die mögliche Abfolge von Dienstelementen unter Einbeziehung der zeitlichen Präzedenzen, die bei TSDs an den geneigten Linien oder an den Zeitachsen ablesbar sind, dargestellt. Hierbei ist eine Grundannahme, daß entweder alle Dienstelemente eines TSDs auftreten, oder keines. Das in Abbildung 3.2 dargestellte TSD ergibt bei szenarischer Interpretation:

<Xrequest, Xindication, Xresponse, Xconfirm>
<Xrequest, Xindication, Xconfirm, Xresponse>
<Xrequest, Xconfirm, Xindication, Xresponse>

Wichtig ist bei den oben angegebenen Szenarien, daß immer nur der Ablauf eines kompletten Szenarios möglich ist. Mit dieser Eigenschaft wird der Lebendigkeitsanteil bei der Interpretation von TSDs dargestellt. Wären Teilabläufe möglich, könnten zum Beispiel bei der Datenübertragung spontan Daten erzeugt werden oder aber auch verloren gehen. Auch ein Verbindungsaufbauwunsch X_CONNECTrequest könnte gegebenenfalls zu keiner Folgeaktion führen. Daher ist der Teilablauf eines Szenarios bei szenarischer Interpretation nicht gestattet, es sei denn, ein anderes TSD erlaubt diesen Teilablauf explizit als Gesamtheit.

Bei TSDs ist es erlaubt, daß die Aktionen an unterschiedlichen Dienstzugangspunkten die gleiche Bezeichnung besitzen und somit nicht unterscheidbar sind. Zur eindeutigen Bestimmung der Aktionen in den Spuren ist es daher notwendig, die Information über den Ursprung und die Orientierung[3] eines Dienstelements hinzuzufügen. So wird zum Beispiel beim TSD aus Abbildung 3.2 auf Seite 21 das Dienstelement $Xrequest$ vollständig mit der Aktion $Xrequest_A^{In}$ bezeichnet und $Xindication$ als $Xindication_B^{Out}$ bezeichnet. Hierdurch wird eindeutig festgelegt, daß das Dienstelement $Xrequest$ vom Dienstbenutzer

[3]Die explizite Bezeichnung der Orientierung ist im ISO/OSI Schichtenmodell unnötig, da sie nach [ISO87b, CCI88d] aus den Namen der Dienstelemente hervorgeht. Wenn jedoch TSDs auch außerhalb des ISO/OSI Schichtenmodells verwendet werden, wie dies häufig vorkommt, kann die Berücksichtigung der Orientierung erforderlich sein.

A an den Diensterbringer gesendet wird, sowie daß ein $Xindication$ vom Diensterbringer zum Dienstbenutzer B ausgegeben wird. Der Zusatz, ob eine Aktion eine Eingabe- oder Ausgabeaktion ist, ist wegen den in [ISO87b, CCI88d] getroffenen Konventionen nicht notwendig, da Eingabeaktionen stets auf *request* und *response* enden, sowie Ausgabeaktionen mit *indication* und *confirm* bezeichnet werden. Der Ursprung einer Aktion, das heißt die Bezeichnung des Dienstzugangspunkts, ist nicht in jedem Fall vernachlässigbar, da er nicht aus der Bezeichnung der Dienstelemente ablesbar ist. Zur Vereinfachung wird in dieser Arbeit die Bezeichnung des Ursprungs eines Dienstelements dann weggelassen, wenn eine eindeutige Zuordnung von Dienstelementen zum Ursprung möglich ist, das heißt, daß eine Aktion in den beschriebenen TSDs nur an einem Dienstzugangspunkt auftritt.

Als *Komposition* von TSDs wird das Zusammenwirken von mehreren TSDs bezeichnet. Die Interpretation der Komposition von TSDs ist problematisch, da auf der Ebene der graphischen Darstellung in den Normen für TSDs nur eine mengenartige Vereinigung zur Verfügung steht, und die Dienstbeschreibungen nur ein gleichberechtigtes Nebeneinanderstellen von TSDs verwenden.

Für die folgenden Überlegungen sei $P_{TSD_i}(t)$ das Prädikat, das die Spuren auszeichnet, die gemäß dem Aufbau von TSD_i korrekt erzeugt werden. Hierbei ist zu beachten, daß dieses Prädikat erst im Kapitel 3.3.3 definiert wird. Es wird an dieser Stelle nur eingeführt, damit die Bestimmung des Kompositionsoperators formal fundiert wird. Die Komposition zweier TSDs wird mit

$$P_{\{TSD_1,TSD_2\}}(t)$$

bezeichnet. Als erster Lösungsansatz zur Formalisierung der Komposition wird die Konjunktion der Prädikate gewählt:

$$P_{\{TSD_1,TSD_2\}}(t) = P_{\{TSD_1\}}(t) \wedge P_{\{TSD_2\}}(t)$$

Dazu läßt sich leicht ein Beispiel finden, das zeigt, daß diese Semantik nicht intendiert ist. Hierfür werden die TSDs aus Abbildung 3.9 angegeben.

Abbildung 3.9: Beispiel zur Komposition zweier TSDs mit einer Konjunktion

Die in Abbildung 3.9 dargestellten TSDs sind aus der Dienstbeschreibung der Transportschicht [ISO84c, CCI88h] entnommen und dahingehend modifiziert worden, daß die Dienstelemente zu den Ereignissen a, b und c abstrahiert wurden und die nachfolgenden Aktionen von TSD_2 weggelassen wurden, da sie für die folgenden Erklärungen nicht notwendig sind. Sei bei der prädikativen Beschreibung das linke TSD aus Abbildung 3.9 TSD_1, und das rechte TSD_2. Wird die Auffassung vertreten, daß TSDs die Menge der

möglichen Abläufe exakt beschreiben, und somit nach TSD_2 einem a irgendwann später ein c zu folgen hat, gilt:

$$\neg P_{\{TSD_2\}}(<a, b>)$$

Eine konjunktive Komposition würde somit das folgende Ergebnis liefern:

$$\neg P_{\{TSD_1, TSD_2\}}(<a, b>)$$

Dies ist aber gegen die Intuition, daß ein TSD zumindest ein mögliches Szenario beschreibt. Es wird sogar das von TSD_1 explizit zugelassene Szenario ausgeschlossen. Die Ursache ist, daß bei der konjunktiven Verknüpfung die Kausalität in beiden Diagrammen eingehalten werden muß. Das Dienstelement a ist somit kausal für b und c. Wählt man hierbei die Interpretation, daß nur die an einem TSD beteiligten Aktionen betrachtet werden, so folgt mit den Beispielen in Abbildung 3.9 aus der Gültigkeit von

$$P_{\{TSD_1, TSD_2\}}(<a, b>)$$

und der konjunktiven Verknüpfung die Gültigkeit von

$$P_{\{TSD_2\}}(<a>)$$

Aus

$$P_{\{TSD_1, TSD_2\}}(<a, c>)$$

folgt dann

$$P_{\{TSD_1\}}(<a>)$$

Somit gilt

$$P_{\{TSD_1, TSD_2\}}(<a>)$$

da nur die betreffenden Teile von TSDs betrachtet werden. Dadurch wird der Lebendigkeitsanteil der TSDs verletzt. Damit ist zum Beispiel ein Datenverlust bei der Übertragung möglich. Ähnliche Überlegungen führen dazu, daß dann

$$P_{\{TSD_1, TSD_2\}}(<a, b, c>)$$

gilt. Somit kann der Diensterbringer auf eine Aktion mehrmals reagieren. Wenn zum Beispiel b einen Verbindungsabbruch und c einen korrekten Verbindungsaufbau darstellt, so führt dies zu einem widersprüchlichen Verhalten. Anhand der Abbildung 3.10 läßt sich eine weitere Konsequenz verdeutlichen.

Abbildung 3.10: Beispiel zur Komposition zweier TSDs mit einer Konjunktion

Sei bei der prädikativen Beschreibung das linke TSD aus Abbildung 3.10 TSD_3, und das rechte TSD_4. Somit folgt aus

$$P_{\{TSD_3,TSD_4\}}(<a,c>)$$

und der konjunktiven Verknüpfung

$$P_{\{TSD_4\}}(<c>)$$

da bei TSD_4 nur die Aktionen b und c betrachtet werden dürfen. Gleiche Überlegungen führen dazu, daß aus

$$P_{\{TSD_3,TSD_4\}}(<b,c>)$$

die Gültigkeit von

$$P_{\{TSD_3\}}(<c>)$$

folgt. Dies ergibt

$$P_{\{TSD_3,TSD_4\}}(<c>)$$

was der Intuition der TSDs widerspricht, da weder die Aktion a noch die Aktion b zuvor erfolgt ist.

Ein zweiter Lösungsansatz ist die Verwendung einer Disjunktion:

$$P_{\{TSD_1,TSD_2\}}(t) = P_{\{TSD_1\}}(t) \vee P_{\{TSD_2\}}(t)$$

Dazu läßt sich wiederum ein Beispiel finden, das belegt, daß diese Semantik nicht intendiert ist. Hierfür werden die TSDs aus Abbildung 3.11 angegeben.

Abbildung 3.11: Beispiel zur Kombination zweier TSDs mit einer Disjunktion

Die in Abbildung 3.11 dargestellten TSDs sind wiederum der Transportschicht entnommen und die Dienstelemente abstrahiert worden. Sei bei der prädikativen Beschreibung das linke TSD von Abbildung 3.11 durch TSD_3 und das rechte TSD durch TSD_4 spezifiziert. Bei der Interpretation, daß nur die innerhalb eines TSDs beschriebenen Elemente auftreten dürfen, erhält man:

$$\neg P_{\{TSD_3\}}(<a,b,c,d>) \wedge \neg P_{\{TSD_4\}}(<a,b,c,d>)$$

Somit gilt:

$$\neg P_{\{TSD_3,TSD_4\}}(<a,b,c,d>)$$

Dies widerspricht der Intuition, daß die Szenarien sequentiell komponiert werden können. Bei einer alternativen Interpretation, bei der nur die an einem TSD beteiligten Aktionen betrachtet werden, und die nichtbeteiligten Aktionen vernachlässigt werden, erhält man:

$$P_{\{TSD_3\}}(<a, b, c>)$$

Mit der disjunktiven Komposition gilt:

$$P_{\{TSD_3, TSD_4\}}(<a, b, c>)$$

Dies würde aber die Beschreibung eines Datenverlusts bei der Datenübertragungsphase ermöglichen. Ein einziges korrektes Szenario, das in einer Spur enthalten wäre, würde dazu führen, daß diese Spur akzeptiert werden würde. Somit hätte man einen sehr groben Begriff der Korrektheit bezüglich einer Spur bezüglich einer Menge von TSDs.

Eine Lösung der Problematik liefert die Betrachtung der TSDs als eine Menge, die ein Generierungsprinzip für korrekte Spuren liefert. Dazu läßt sich die Komposition von TSDs in zwei unabhängige Operationen aufspalten:

- Bei der *alternativen Komposition* wird eine Auswahl zwischen zwei TSDs getroffen. Ein Szenario der Komposition besteht also aus einem Szenario, das durch eines der beiden kombinierten TSDs beschrieben wird.
- Bei der *repetitiven Komposition* wird die Wiederholung eines Szenarios beschrieben. Nur mit der repetitiven Komposition ist die vollständige Interpretation eines TSDs möglich.

Durch diese Aufspaltung wird bei der alternativen Komposition die Verwendung einer einfachen Disjunktion möglich, während die Interpretation der repetitiven Komposition aufwendiger ist. Die Verwendung einer einfachen sequentiellen Wiederholung ist nicht geeignet, einen Datentransport zu modellieren, bei dem keine Reihenfolgegarantie gefordert wird, da in diesem Fall ein Interleaving der beteiligten Aktionen möglich ist. In der folgenden Semantik wird aus diesem Grund zur vollständigen Interpretation von TSDs eine Interleaving Wiederholung zugrundegelegt.

3.3.2 Syntax von TSD-DL

Im folgenden wird die BNF-Syntax von Time Sequence Diagram Description Language (TSD-DL) angegeben. Ein TSD-DL-Ausdruck setzt sich aus Aktionen und Operatoren zusammen. Hierbei besteht eine Aktion aus der Bezeichnung des Dienstelements mit der zusätzlichen Information über dessen Richtung und die betreffende Quelle, das heißt den Dienstzugangspunkt. In einem ersten Schritt wird diese zusätzliche Information vernachlässigt.

Definition 3.4

Die *Syntax eines* TSD-DL-*Ausdrucks* wird definiert als:

(1) <TSD> ::= <event>
(2) <TSD> ::= <TSD> → <TSD>

(3) <TSD> ::= <TSD> $\sim$ <TSD>
(4) <TSD> ::= <TSD> $\circledast$ <TSD>
(5) <TSD> ::= <TSD> | <TSD>
(6) <TSD> ::= <TSD>$^{\underline{\omega}}$

□

In den Zeilen 1 bis 4 werden die Bestandteile zur Beschreibung von einzelnen TSDs eingeführt. Die Zeilen 5 und 6 beschreiben die Kompositionsoperatoren für TSDs. Zwei TSDs können mit dem Operator $\rightarrow$, mit dem eine zeitliche Präzedenz dargestellt wird, sequentiell kombiniert werden. Die Abwesenheit einer derartigen zeitlichen Präzedenz wird durch $\sim$ beschrieben. Die konjunktive Komposition von TSDs, die zur schematischen Darstellung von einzelnen TSDs benötigt wird, wird mit $\circledast$, die alternative Komposition von TSDs mit | dargestellt. Ein sehr mächtiger Kompositionsoperator ist $^{\underline{\omega}}$, der die Wiederholung eines TSD-DL-Ausdrucks beschreibt, und somit die vollständige Interpretation eines TSDs ermöglicht. Die Wiederholung ist definiert als eine beliebige Anzahl von Interleavings eines TSD-DL-Ausdrucks mit sich selbst. In der Grammatik für TSD-DL haben die Operatoren $\rightarrow$ und $\sim$ die höchste Priorität, dann folgen die Operatoren $\circledast$ und |. Die niedrigste Priorität hat der Operator $^{\underline{\omega}}$. Zur vereinfachten Benutzung von TSD-DL ist auch die Klammerung von Ausdrücken erlaubt. Dies wird durch die Regel <TSD> ::= <(TSD)> beschrieben.[4]

Das TSD aus Abbildung 3.1 kann in folgenden TSD-DL-Ausdruck zur szenarischen Interpretation umgewandelt werden:

$$(\mathsf{Xrequest} \rightarrow \mathsf{Xindication}) \rightarrow (\mathsf{Xresponse} \rightarrow \mathsf{Xconfirm})$$

Bei einer Dienstbeschreibung wird eine Menge von TSDs angeführt. Sei im folgenden TSD_i der TSD-DL-Ausdruck eines einzelnen TSDs, der der szenarischen Interpretation entspricht, dann ist

$$(TSD_1|TSD_2|\ldots|TSD_n)^{\underline{\omega}}$$

der TSD-DL-Ausdruck, der die Interpretation einer Menge von TSDs eines Dienstes beschreibt. Dies entspricht der vollständigen Interpretation einer alternativen Komposition von TSDs. Eine intuitive Vorstellung der Semantik ist das faire Mischen aller Szenarien der TSD_i, also auch der Wiederholungen.

3.3.3 Denotationelle Semantik von TSD-DL

Ein TSD-DL-Ausdruck beschreibt eine Menge von Spuren. Um die Beschreibung zu vereinfachen, werden Prädikate verwendet. Mit den in diesem Kapitel angegebenen Regeln kann ein TSD-DL-Ausdruck induktiv über dessen Aufbau in ein Prädikat mit der folgenden Bedeutung umgewandelt werden:

[4] Da diese Regel keine wesentliche Bedeutung zur Bildung einer formalen Semantik für TSDs beiträgt, wird sie bei der folgenden Definition einer Semantik für TSD-DL nicht berücksichtigt.

$P_{TSD-DLexpression}(t)$ ist genau dann wahr, wenn die Spur t gemäß den Regeln von $TSD-DLexpression$ aufgebaut ist.

Für die Definition der formalen Semantik von TSD-DL wird die strikte und stetige Funktion $filter(e, ps, xs)$ zu Hilfe genommen, die als Ergebnis diejenigen Elemente der Spur xs liefert, bei der die Elemente an der korrespondierenden Position von ps gleich dem Element e sind. Die Funktion $filter$ wird polymorph definiert, da sie später auf verschiedene Sorten angewendet wird.

filter: $\alpha \times$ Trace $\alpha \times$ Trace $\beta \longrightarrow$ Trace β

ps=$\epsilon \vee$ xs=$\epsilon \Rightarrow$ filter(e,ps,xs)=ϵ

filter(e,p&ps,x&xs) = if (e = p)
then x&filter(e,ps,xs)
else filter(e,ps,xs)

Mit den folgenden induktiven Definitionen wird die Semantik eines TSD-DL-Ausdrucks eindeutig festgelegt.

1. Definition des Basis TSDs:

P_a(t) = (t=<a>) iff $a : Event$

P_a(t) ist genau dann wahr, wenn t aus einer Spur mit genau einem Element a besteht.

2. Definition der sequentiellen Komposition:

$$P_{TS_1 \rightarrow TS_2}(t) = \exists s_1, s_2 : Trace\ Event. \\ t = s_1 \circ s_2 \wedge P_{TS_1}(s_1) \wedge P_{TS_2}(s_2)$$

Der Lebendigkeitsanteil von TSDs wird mit der grundlegenden Eigenschaft eines TSDs, daß alle beteiligten Ereignisse stattfinden, modelliert. Damit diese nicht verletzt wird, ist es erforderlich, daß die leere Spur nicht in die Definition der Semantik des Basisfalls der TSDs aufgenommen wird. Dies wird an folgendem Beispiel verdeutlicht, bei dem die Zeile 1 der Definition modifiziert wurde:

Beispiel 3.1

Anhand dieses Beispiels wird die Notwendigkeit von $\neg P_a(\epsilon)$ verdeutlicht. Um dies zu belegen, wird auf die Konsequenz eines modifizierten Prädikats P', bei dem $P'_a(\epsilon)$ gilt, hingewiesen. Indem auch Teilspuren akzeptiert werden, wird der Lebendigkeitsanteil von TSDs verletzt.

1'. Modifizierter Basisfall der TSDs:

$$P'_a(\mathrm{t}) = (\mathrm{t}{=}{<}\mathrm{a}{>} \vee \mathrm{t}{=}\epsilon) \qquad \text{iff } a : Event$$

Wenn wir jetzt die Definition der sequentiellen Komposition (Zeile 2) äquivalent benutzen würden, würde der TSD-DL-Ausdruck $a \rightarrow b$ nach einer Vereinfachung die folgende Semantik erhalten:

$$P'_{a \rightarrow b}(t) = (t = \epsilon \vee t = <a> \vee t = <b> \vee t = <a, b>)$$

Somit wäre eine Spur, die nur aus dem Element a oder b besteht, eine bezüglich dem TSD $a \rightarrow b$ korrekt erzeugte Spur. Dies widerspräche der Intention von TSDs. □

Eine Konsequenz des Ausschlusses der leeren Spur im Basisfall ist, daß diese nur durch den später definierten Wiederholungsoperator akzeptiert werden kann.

3. Definition der Interleaving Komposition:

$$\begin{aligned} P_{TS_1 \sim TS_2}(t) = \exists bs : Trace\ Bool. \\ \#bs = \infty \wedge P_{TS_1}(filter(true, bs, t)) \\ \wedge P_{TS_2}(filter(false, bs, t)) \end{aligned}$$

Die Interleaving Komposition wird verwendet, um die Abwesenheit einer zeitlichen Präzedenz in einem TSDs darzustellen. Durch die Forderung, daß bs eine unendliche Spur sei, wird sichergestellt, daß jedes Element von t berücksichtigt wird, sowie daß sowohl TS_1 als auch TS_2 bei Teilen von t erfüllt werden. Falls bei P_{TS_1} und P_{TS_2} die leeren Spuren nicht erfaßt werden, dies ist zum Beispiel bei einfachen Aktionen der Fall, so sind sowohl TS_1 als auch TS_2 zu berücksichtigen. So gilt zum Beispiel: $\neg P_{a \sim b}(<a>)$. Dies ist ebenfalls eine Folge des Ausschlusses des leeren Stroms im Basisfall, wie dies im folgenden verdeutlicht wird. Wenn TS_1 und TS_2 Aktionen sind, gilt $P_{TS_1}(t) \Rightarrow t \neq \epsilon$. Bei einfachen TSDs ist also ein $bs = false^\infty$ nicht möglich, da dann $filter(true, bs, t) = \epsilon$ und $\neg P_{TS_1}(\epsilon)$ gilt.

4. Definition der konjunktiven Komposition:

$$P_{TS_1 \circledast TS_2}(t) = P_{TS_1}(t) \wedge P_{TS_2}(t)$$

5. Definition der disjunktiven Komposition:

$$P_{TS_1 | TS_2}(t) = P_{TS_1}(t) \vee P_{TS_2}(t)$$

6. Definition der repetitiven Komposition:

$$P_{TS\underline{\omega}}(t) = \exists ns : Trace\ Nat.\ \#ns = \infty \wedge \forall n : Nat.\ P_{TS}(filter(n, ns, t)) \vee filter(n, ns, t) = \epsilon$$

Der mächtige Operator $\underline{\omega}$ wird verwendet, um ein beliebiges, gegebenenfalls unendliches faires Interleaving eines TSDs mit sich selbst darzustellen. Das heißt, daß eine Spur in eine beliebige Anzahl disjunkter Teile vollständig zerlegt wird. Dies geschieht durch die Verwendung der Funktion $filter$. Eine solche Zerlegung wird durch den Strom ns angegeben. Daß diese Zerlegung vollständig ist, wird durch die Quantifizierung über alle natürlichen Zahlen und durch die Bedingung $\#ns = \infty$ sichergestellt. Es gilt zum Beispiel: $P_{(a \to b)\underline{\omega}}(t)$ für die unendliche Spur $t = \mathsf{fix}\boldsymbol{\lambda} s.a\&b\&s$. Dies wird belegt durch: $ns = <1, 1, 2, 2, \ldots>$. Das gleiche Prädikat gilt für: $t = \mathsf{fix}\boldsymbol{\lambda} s.a\&a\&b\&s$, wegen $ns = <1, 2, 1, 3, 4, 2, \ldots>$. Die letzte Spur beschreibt eine Eigenschaft, die nicht aus dem endlichen Verhalten approximiert werden kann. Es gilt für jedes echte Präfix t' von t: $\neg P_{(a \to b)\underline{\omega}}(t')$ Nur bei der unendlichen Spur t gibt es für jedes Ereignis a ein korrespondierendes b.

Man beachte, daß wegen der Definition der repetitiven Komposition für alle TSD-DL-Ausdrücke TS $P_{TS\underline{\omega}}(\epsilon)$ gilt.

Definition 3.5

Die Semantik eines TSD-DL*-Ausdrucks expr* wird durch die folgende Spurmenge festgelegt:

$$[\![expr]\!]_{TSD} := \{t \in Act^{\omega} | P_{expr}(t)\}$$

Hierbei sei Act die Menge aller Aktionen. □

3.3.4 Beispiele für die Semantik einiger TSD-DL-Ausdrücke

Im folgendem wird ein Beispiel für die Semantik von TSD-DL-Ausdrücken angegeben.

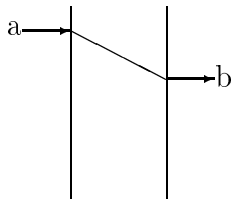

Abbildung 3.12: TSD mit zwei Dienstelementen

Das in Abbildung 3.12 dargestellte TSD kann bei einer vollständigen Interpretation durch den TSD-DL-Ausdruck $(a \to b)\underline{\omega}$ beschrieben werden, dessen Semantik im folgenden untersucht wird. Die Verwendung der induktiven Definition aus Kapitel 3.3.3 ergibt:

$$P_{(a\to b)^{\underline{\omega}}}(t) = \exists ns : Trace\ Nat.\ \#ns = \infty \wedge \forall n : Nat.\ P_{(a\to b)}(filter(n, ns, t)) \\ \vee filter(n, ns, t) = \epsilon$$

Im nächsten Schritt wird $P_{(a\to b)}(t)$ durch die Elimination des Existenzquantors vereinfacht.

$$\begin{aligned} P_{a\to b}(t) &= \exists s_1, s_2 : Trace\ Event.\ t = s_1 \circ s_2 \wedge (s_1 = <a>) \wedge (s_2 = <b>) \\ &= (t = <a, b>) \end{aligned}$$

Das Einsetzen der obigen Vereinfachung in $P_{(a\to b)^{\underline{\omega}}}(t)$ ergibt:

$$P_{(a\to b)^{\underline{\omega}}}(t) = \exists ns : Trace\ Nat. \#ns = \infty \wedge \forall n : Nat.\ filter(n, ns, t) = <a, b> \\ \vee filter(n, ns, t) = \epsilon$$

Zuerst werden die akzeptierten endlichen Spuren untersucht. Es ist zum Beispiel $P_{(a\to b)^{\underline{\omega}}}(<a, b, a, b>)$ erfüllt. Dies kann durch die Instantiierung von ns mit $n_1 \& n_1 \& n_2 \& n_2 \& ns'$ nachgewiesen werden. Hierbei sind n_1 und n_2 beliebige unterschiedliche, natürliche Zahlen und ns' ein beliebiger Strom von natürlichen Zahlen. Ähnliche Überlegungen führen dazu, daß $P_{(a\to b)^{\underline{\omega}}}(<a, a, b, b>)$ gilt. Dies wird durch $ns = n_1 \& n_2 \& n_1 \& n_2 \& ns'$ bewiesen. Daß $P_{(a\to b)^{\underline{\omega}}}(<b, a>)$ nicht gilt, kann mit $ns = n_1 \& n_2 \& ns'$ durch die Fallunterscheidung $n_1 = n_2$ und $n_1 \neq n_2$ bewiesen werden.

Als nächstes werden unendliche Spuren untersucht. $P_{(a\to b)^{\underline{\omega}}}(\mathsf{fix}\boldsymbol{\lambda} s.a \& b \& s)$ gilt. Dies wird durch die Instantiierung von ns mit $(\mathsf{fix}\boldsymbol{\lambda} f.\boldsymbol{\lambda} n.n \& n \& f.n + 1).1$ nachgewiesen.

Ein etwas weiterführender Fall ist gegeben durch $P_{(a\to b)^{\underline{\omega}}}(\mathsf{fix}\boldsymbol{\lambda} s.a \& a \& b \& s)$. Eine informelle Vorüberlegung führt dazu, daß es zu jedem a ein nachfolgendes b gibt. Dies kann durch $ns = (\mathsf{fix}\boldsymbol{\lambda} f.\boldsymbol{\lambda} n.2 * n - 1 \& 2 * n \& n \& f.n + 1).1$ nachgewiesen werden. Für jedes echte Präfix von $\mathsf{fix}\boldsymbol{\lambda} s.a \& a \& b \& s$ gilt diese Aussage nicht.

Daß $P_{(a\to b)^{\underline{\omega}}}(\mathsf{fix}\boldsymbol{\lambda} s.a \& b \& b \& s)$ nicht gilt, kann mit $ns = n_1 \& n_2 \& n_3 \& ns'$ und einer Fallunterscheidung gezeigt werden.

Dies läßt sich zusammenfassen zu:

Lemma 3.1

$$[\![(a \to b)^{\underline{\omega}}]\!]_{TS^{\mathcal{D}}} = \{t \in \{a, b\}^{\omega} | \#a \copyright t = \#b \copyright t \wedge \forall t' \sqsubseteq t.\ \#a \copyright t' \geq \#b \copyright t'\}$$

□

Beweis:

Sei $T := \{t \in \{a, b\}^{\omega} | \#a \copyright t = \#b \copyright t \wedge \forall t' \sqsubseteq t. \#a \copyright t' \geq \#b \copyright t'\}$
Zu zeigen ist: a) $t \in [\![(a \to b)^{\underline{\omega}}]\!]_{TS^{\mathcal{D}}} \Rightarrow t \in T$
b) $t \in T \Rightarrow t \in [\![(a \to b)^{\underline{\omega}}]\!]_{TS^{\mathcal{D}}}$

a) $t \in [\![(a \rightarrow b)^{\underline{\omega}}]\!]_{TSD} = P_{(a \rightarrow b)^{\underline{\omega}}}(t)$
$=$
$\exists\ ns : Trace\ Nat.\#ns = \infty \wedge \forall n : Nat.filter(n, ns, t) = <a, b>$
$\vee filter(n, ns, t) = \epsilon$
Es sind die folgenden drei Teilaussagen zu beweisen:
a1) $t \in \{a, b\}^{\omega}$
Widerspruchsannahme führt zu
$\forall\ ns.\#ns \neq \infty \vee \exists n : Nat.filter(n, ns, t) \neq <a, b> \vee filter(n, ns, t) \neq \epsilon$

a2) $\#a©t = \#b©t$ Widerspruchsbeweis äquivalent zu a1)
a3) $\forall t' \sqsubseteq t.\#a©t' \geq \#b©t'$
Widerspruchsannahme ist:
$\exists t' \sqsubseteq t.\#a©t' < \#b©t'$
und dies führt zu:
$\forall ns.\exists n : Nat.filter(n, ns, t) \neq <a, b> \vee filter(n, ns, t) \neq \epsilon$
b) $t \in T$
Aus t wird ein ns wie folgt konstruiert:
$t' \sqsubseteq t \wedge \#t' = k \wedge t'[k] = a \Rightarrow ns[k] = \#a©t'$
$t' \sqsubseteq t \wedge \#t' = k \wedge t'[k] = b \Rightarrow ns[k] = \#b©t'$
Somit kennzeichnet die k-te Position von ns die Anzahl der a beziehungsweise b bis zu dieser Position. Darauf aufbauend wird, falls es notwendig ist, ns mit beliebigen natürlichen Zahlen zu einer unendlichen Spur ergänzt.
Nach Konstruktion und wegen $\#a©t = \#b©t \wedge \forall t' \sqsubseteq t.\ \#a©t' \geq \#b©t'$ gilt:
$\forall n : Nat.filter(n, ns, t) = <a, b> \vee filter(n, ns, t) = \epsilon$
Somit: $P_{(a \rightarrow b)^{\underline{\omega}}}(t)$

□

3.3.5 Eigenschaften von TSD-DL

Im folgenden Kapitel werden einige Eigenschaften von Ausdrücken in TSD-DL bewiesen. Es werden unter anderem einige Vereinfachungsregeln, aber auch Kompositionsregeln angegeben. Insbesondere wird dadurch die Semantik des Kompositionsprinzips von TSDs formal beschrieben. Die wesentliche Grundlage für die folgenden Aussagen ist die Definition eines Äquivalenzbegriffs auf TSD-DL-Ausdrücke:

Definition 3.6

Für zwei TSD-DL-Ausdrücke TSD_1 und TSD_2 ist $TSD_1 \simeq TSD_2$ genau dann erfüllt, wenn $\forall t : P_{TSD_1}(t) = P_{TSD_2}(t)$. □

Somit gilt $TSD_1 \simeq TSD_2$ genau dann, wenn $[\![TSD_1]\!]_{TSD} = [\![TSD_2]\!]_{TSD}$ gilt. Die Verwendung der Äquivalenz von Spurmengen zur Beschreibung der Äquivalenz von TSD-DL-Ausdrücken ist sinnvoll, da in dieser Arbeit als Semantik von TSDs Spurmengen betrachtet

werden. Da die Gleichheit = eine Äquivalenzrelation ist, ist auch $\simeq$ eine Äquivalenzrelation.

Satz 3.1

Für beliebige TSD-DL-Ausdrücke TSD_1, TSD_2, TSD_3, TSD_4 und Aktionen a, b gelten folgende Aussagen:

$$(TSD_1|TSD_2)|TSD_3 \simeq TSD_1|(TSD_2|TSD_3) \quad (1)$$
$$(TSD_1 \to TSD_2) \to TSD_3 \simeq TSD_1 \to (TSD_2 \to TSD_3) \quad (2)$$
$$(TSD_1 \sim TSD_2) \sim TSD_3 \simeq TSD_1 \sim (TSD_2 \sim TSD_3) \quad (3)$$
$$TSD_1|TSD_2 \simeq TSD_2|TSD_1 \quad (4)$$
$$TSD_1 \sim TSD_2 \simeq TSD_2 \sim TSD_1 \quad (5)$$
$$TSD_1|TSD_1 \simeq TSD_1 \quad (6)$$
$$TSD_1 \to (TSD_2|TSD_3) \simeq (TSD_1 \to TSD_2)|(TSD_1 \to TSD_3) \quad (7)$$
$$TSD_1 \sim (TSD_2|TSD_3) \simeq (TSD_1 \sim TSD_2)|(TSD_1 \sim TSD_3) \quad (8)$$
$$(TSD_1|TSD_2) \to TSD_3 \simeq (TSD_1 \to TSD_3)|(TSD_2 \to TSD_3) \quad (9)$$
$$(TSD_1|TSD_2) \sim TSD_3 \simeq (TSD_1 \sim TSD_3)|(TSD_2 \sim TSD_3) \quad (10)$$
$$a \sim b \simeq (a \to b)|(b \to a) \quad (11)$$
$$(TSD_1^{\underline{\omega}}|TSD_2^{\underline{\omega}})^{\underline{\omega}} \simeq (TSD_1|TSD_2)^{\underline{\omega}} \quad (12)$$
$$TSD_1^{\underline{\omega}} \sim TSD_2^{\underline{\omega}} \simeq (TSD_1|TSD_2)^{\underline{\omega}} \quad (13)$$

□

Der Beweis der Aussagen (1) bis (11) ist durch Einsetzen der Definitionen relativ einfach schematisch zu erhalten. Da diese Beweise leicht nachvollziehbar sind, sind sie in dieser Arbeit weggelassen worden. Der Beweis der Aussagen (12) und (13) ist wegen der Wichtigkeit und zur Anführung der Beweismethodik angegeben worden. Ein weiterer Grund ist die Komplexität der Beweise von diesen beiden Aussagen, die die der Beweise der restlichen Aussagen bei weitem übersteigt.

Beweis:

Es wird Aussage zuerst (13) bewiesen: $TSD_1^{\underline{\omega}} \sim TSD_2^{\underline{\omega}} \simeq (TSD_1|TSD_2)^{\underline{\omega}}$

Zu zeigen ist:
(1) $P_{TSD_1^{\underline{\omega}} \sim TSD_2^{\underline{\omega}}}(t) \Rightarrow P_{(TSD_1|TSD_2)^{\underline{\omega}}}(t)$
Im weiteren werden die Zerlegungen mit den Operatoren $\underline{\omega}$ und $\sim$ durch eine einzige Zerlegung mit dem Operator $\underline{\omega}$ nachgebildet.
Annahme: $P_{TSD_1^{\underline{\omega}} \sim TSD_2^{\underline{\omega}}}(t)$
$= \exists bs.\#bs = \infty \wedge P_{TSD_1^{\underline{\omega}}}(filter(true, bs, t)) \wedge P_{TSD_2^{\underline{\omega}}}(filter(false, bs, t))$

Sei dieses bs geeignet instantiiert. Somit gilt:
$\exists ns'.\forall n'.(P_{TSD_1}(filter(n', ns', filter(true, bs, t)))$
$\vee filter(n', ns', filter(true, bs, t)) = \epsilon)$
$\wedge \exists ns''.\forall n''.(P_{TSD_2}(filter(n'', ns'', filter(false, bs, t)))$
$\vee filter(n'', ns'', filter(false, bs, t)) = \epsilon)$

Anschließend werden die Zerlegungen gemäß bs, ns', ns'' kombiniert, indem eine elementweise Potenzierung der Spuren ns' und ns'' erfolgt, damit eine spätere Kombination möglich ist.
Seien ns^* und ns^{**} wie folgt definiert: $ns^*[k] = 2^{ns'[k]}$ und $ns^{**}[k] = 3^{ns''[k]}$
Eine später Kombination von ns^* und ns^{**} zu einer einzigen Spur ist wegen der eindeutigen Unterscheidbarkeit der Elemente ohne weiteres möglich.
Sei $t'_i := filter(i, ns^*, filter(true, bs, t))$
und $t''_i := filter(i, ns^{**}, filter(false, bs, t))$
Nach der Annahme gilt: $P_{TSD_1}(t'_i) \vee t'_i = \epsilon$
und $P_{TSD_2}(t''_i) \vee t''_i = \epsilon$
Die Konstruktion von ns aus bs, ns^* und ns^{**} geschieht mit Hilfe der Spur p, die aus natürlichen aufsteigenden Zahlen besteht.
Sei nun $p = (\mathsf{fix}\boldsymbol{\lambda} f.\boldsymbol{\lambda} n.n \& f.n + 1).1$
Also $(p = <1, 2, 3, \ldots>)$
p dient zur Bezeichnung einer Spur der Positionen der Elemente.
Und $p'_i := filter(i, ns^*, filter(true, bs, p))$
p'_i beschreibt die Positionen der Elemente eines Teilstücks, das durch sukzessive Filterung mit i und $true$ erhalten wird, in der Ausgangsspur t.
Äquivalent dazu sei: $p''_i := filter(i, ns^{**}, filter(false, bs, p))$
Aus p'_i und p''_i läßt sich ein ns wie folgt konstruieren:
$\exists j.i = 2^j \Rightarrow filter(i, ns, p) = p'_i$
$\exists j.i = 3^j \Rightarrow filter(i, ns, p) = p''_i$
$\forall j.i \neq 2^j \wedge i \neq 3^j \Rightarrow filter(i, ns, p) = \epsilon$
Nach obiger Konstruktion erfüllt dieses ns die Eigenschaft:
$\forall n.(P_{TSD_1}(filter(n, ns, t))) \vee (P_{TSD_2}(filter(n, ns, t)) \vee filter(n, ns, t) = \epsilon)$
Somit gilt: $P_{(TSD_1|TSD_2)\underline{\omega}}(t)$

(2) $P_{(TSD_1|TSD_2)\underline{\omega}}(t) \Rightarrow P_{TSD\frac{\omega}{1} \sim TSD\frac{\omega}{2}}(t)$
Annahme: $P_{(TSD_1|TSD_2)\underline{\omega}}(t)$ Sei das darin verborgene ns geeignet bestimmt, so daß: $\forall n.P_{TSD_1}(filter(n, ns, t)) \vee P_{TSD_2}(filter(n, ns, t)) \vee filter(n, ns, t) = \epsilon$
Es werden im folgenden nach der Zerlegung in Teilstücke der Operator $|$ durch $\sim$ ersetzt.
Sei $t_i := filter(i, ns, t)$, sowie p wie in Teil (1) des Beweises definiert.
Diese Teilstücke werden nun unterschieden nach $P_{TSD_1}(t_i)$. Anschließend wird ein bs konstruiert. Falls $P_{TSD_1}(t_i)$ gilt, so werden die korrespondierenden Elemente von bs auf $true$ gesetzt. ns' ergibt sich dann direkt aus ns.
ns'' ist in diesem Fall so zu bestimmen, daß es keine Überlagerung gibt.
Konstruktion von bs,ns',ns'':
$P_{TSD_1}(t_i) \Rightarrow filter(i, ns', filter(true, bs, p)) = filter(i, ns, p)$

$$\wedge filter(i, ns'', filter(false, bs, p)) = \epsilon$$

Andernfalls wird äquivalent zum vorigen Schritt die zweite Komponente von $\sim$ gewählt.

$\neg P_{TSD_1}(t_i) \Rightarrow filter(i, ns'', filter(false, bs, p)) = filter(i, ns, p)$
$\wedge filter(i, ns', filter(true, bs, p)) = \epsilon$

Nach Konstruktion erfüllt dieses bs die Eigenschaft:

$$\exists bs. \exists ns'. \forall n'. (P_{TSD_1}(filter(n', ns', filter(true, bs, t)))$$
$$\vee filter(n', ns', filter(true, bs, t)) = \epsilon)$$
$$\wedge \exists ns''. \forall n''. (P_{TSD_2}(filter(n'', ns'', filter(false, bs, t)))$$
$$\vee filter(n'', ns'', filter(false, bs, t)) = \epsilon)$$

Somit gilt: $P_{TSD_1^{\underline{\omega}} \sim TSD_2^{\underline{\omega}}}(t)$

□

Beweis:

Es wird Aussage (12) bewiesen: $(TSD_1^{\underline{\omega}} | TSD_2^{\underline{\omega}})^{\underline{\omega}} \simeq (TSD_1 | TSD_2)^{\underline{\omega}}$

Zu zeigen ist:

(1) $P_{(TSD_1|TSD_2)^{\underline{\omega}}} \Rightarrow P_{(TSD_1^{\underline{\omega}}|TSD_2^{\underline{\omega}})^{\underline{\omega}}}(t)$

Die Gültigkeit der Aussage ist sehr einfach zu beweisen, wenn bei $(TSD_1^{\underline{\omega}} | TSD_2^{\underline{\omega}})^{\underline{\omega}}$ die inneren Operatoren $^{\underline{\omega}}$ vernachlässigt werden.

Annahme: $P_{(TSD_1|TSD_2)^{\underline{\omega}}}(t)$

$\exists ns.\ \#ns = \infty \wedge \forall n. (P_{TSD_1}(filter(n, ns, t)))$
$\vee\ (P_{TSD_2}(filter(n, ns, t))$
$\vee\ filter(n, ns, t) = \epsilon)$

Sei $ns' := \mathsf{fix}\boldsymbol{\lambda} s.0 \& s$ und $ns'' = ns'$, somit sind die inneren Operatoren $^{\underline{\omega}}$ bei $(TSD_1^{\underline{\omega}} | TSD_2^{\underline{\omega}})^{\underline{\omega}}$ vernachlässigbar, und nach Konstruktion gilt:

$\exists ns.\ \forall n.\ \exists ns', ns''.\ \#ns' = \infty$
$\wedge\ \#ns'' = \infty$
$\wedge\ \forall n', n''. (P_{TSD_1}(filter(n', ns', filter(n, ns, t)))$
$\vee\ (P_{TSD_2}(filter(n'', ns'', filter(n, ns, t)))$
$\vee\ filter(n, ns, t)) = \epsilon)$

Somit gilt: $P_{(TSD_1^{\underline{\omega}}|TSD_2^{\underline{\omega}})^{\underline{\omega}}}(t)$

(2) $P_{(TSD_1^{\underline{\omega}}|TSD_2^{\underline{\omega}})^{\underline{\omega}}}(t) \Rightarrow P_{(TSD_1|TSD_2)^{\underline{\omega}}}$

Der Beweis beruht auf dem Diagonalisierungsverfahren. Hierbei wird die jeweils zweistufige Zerlegung in $(TSD_1^{\underline{\omega}} | TSD_2^{\underline{\omega}})^{\underline{\omega}}$ durch eine einfache Zerlegung mit nur einem Operator $^{\underline{\omega}}$ nachgebildet.

Annahme: $P_{(TSD_1^{\underline{\omega}}|TSD_2^{\underline{\omega}})^{\underline{\omega}}}(t)$

$=\exists ns.\ \#ns = \infty \wedge \forall n.\ P_{TSD_1^{\underline{\omega}}}(filter(n, ns, t)) \vee P_{TSD_2^{\underline{\omega}}}(filter(n, ns, t))$
$\vee\ filter(n, ns, t) = \epsilon$

Sei dieses ns in folgendem so bestimmt, daß diese Eigenschaft zutrifft.
Wegen der späteren eindeutigen Indexvergabe bei den Teilstücken wird jedes Element von ns zur Basis 2 potenziert. Somit sei ns_1 wie folgt definiert:
$ns_1[k] = 2^{ns[k]}$
Zur Abkürzung für die aus der Zerlegung resultierenden Teilstücke wird
$t_i := filter(i, ns_1, t)$ festgelegt.
Da das Potenzieren die Gültigkeit der Aussage nicht verändert, gilt weiterhin:
$\forall i.\ (P_{TSD\frac{\omega}{1}}(t_i) \vee P_{TSD\frac{\omega}{2}}(t_i) \vee t_i = \epsilon)$
$= \forall i.(\exists ns'.\#ns' = \infty \wedge \forall n'.P_{TSD_1}(filter(n', ns', t_i)) \vee filter(n', ns', t_i) = \epsilon)$
$\vee (\exists ns''.\#ns'' = \infty \wedge \forall n''.P_{TSD_2}(filter(n'', ns'', t_i)) \vee filter(n'', ns'', t_i) = \epsilon)$
$\vee t_i = \epsilon$
Sei $ns'(i)$ das zu i gehörende ns' beziehungsweise $ns''(i)$ das zu ns'' gehörende.
Seien $ns^*(i)$ und $ns^{**}(i)$ wie folgt definiert: $ns^*(i)[k] = 3^{ns'(i)[k]}$ und
$ns^{**}(i)[k] = 5^{ns''(i)[k]}$, wodurch eine eindeutige Indexvergabe der Teilstücke möglich ist.
Sowie $t'_{i,j'} := filter(j', ns^*(i), t_i)$ und $t''_{i,j''} := filter(j'', ns^{**}(i), t_i)$
Somit $\forall i, j', j''.\ P_{TSD_1}(t'_{i,j'}) \vee t'_{i,j'} = \epsilon \vee P_{TSD_2}(t''_{i,j''}) \vee t''_{i,j''} = \epsilon \vee t_i = \epsilon$
Sei $p = (\mathsf{fix}\boldsymbol{\lambda} f.\boldsymbol{\lambda} n.n\&f.n + 1).1$ die Angabe der Positionen. Mit Hilfe von p wird nun ein ns_{neu} konstruiert, das die zweifache Zerlegung durch ns_1, $ns^*(i)$ sowie durch ns_1, $ns^{**}(i)$ in einer Zerlegung vereint.
Und $p_i = filter(i, ns_1, p)$ Somit entspricht p_i der Auflistung der Positionen der Elemente von t, die in t_i enthalten sind.
Sowie $p'_{i,j'} := filter(j', ns^*(i), p_i)$
$p'_{i,j'}$ spiegelt die Zerlegung nach ns_1 und $ns^*(i)$ wider.
Und $p''_{i,j''} := filter(j'', ns^{**}(i), p_i)$
$p''_{i,j''}$ spiegelt die Zerlegung nach ns_1 und $ns^{**}(i)$ wider.
Mit $p'_{i,j'}$ und $p''_{i,j''}$ läßt sich ein ns_{neu} konstruieren:
$\exists i, j'.\ k = 2^i * 3^{j'} \Rightarrow filter(k, ns_{neu}, p) = p'_{i,j'}$
$\exists i, j''.\ k = 2^i * 5^{j''} \Rightarrow filter(k, ns_{neu}, p) = p''_{i,j''}$
$\forall i, j.\ k \neq 2^i * 3^j \wedge k \neq 2^i * 5^j \Rightarrow filter(k, ns_{neu}, p) = \epsilon$
Durch die Gleichheit der Positionen sind die durch die Zerlegung ns_{neu} erhaltenen Teilstücke gleich mit den aus ns_1 und ns^* beziehungsweise aus ns_1 und ns^{**} resultierenden.
Nach obiger Konstruktion erfüllt dieses ns_{neu} somit die Eigenschaft:
$\forall n.\ (P_{TSD_1}(filter(n, ns_{neu}, t))) \vee (P_{TSD_2}(filter(n, ns_{neu}, t))$
$\vee\ filter(n, ns_{neu}, t) = \epsilon)$
Somit gilt: $P_{(TSD_1|TSD_2)\underline{\omega}}(t)$

□

Die Aussagen (12) und (13) aus Satz 3.1 beschreiben die Kompositionalität der Semantik von TSDs. Seien TSD_1, TSD_2 jeweils TSD-DL-Ausdrücke, die Mengen von TSDs beschreiben. Eine Kombination der einzelnen TSDs wird durch den Operator | für die alternative Komposition beschrieben. Hierdurch wird nur der Aspekt der szenarischen Interpretation dargestellt. Will man nun alle Abläufe inklusive der Wiederholungen bestimmen

– dies modelliert die vollständige Interpretation – so findet der Wiederholungsoperator $\underline{\omega}$ Verwendung. In Aussage (12) wird das Kompositionsprinzip beschrieben: Betrachtet man die vollständige Interpretation durch die TSD-DL-Ausdrücke $TSD_1^{\underline{\omega}}$ und $TSD_2^{\underline{\omega}}$, kombiniert diese alternativ und erlaubt auch Wiederholungen, so erhält man einen semantisch äquivalenten TSD-DL-Ausdruck, wenn TSD_1 und TSD_2 alternativ kombiniert werden, und danach eine vollständige Interpretation betrachtet wird.

Definition 3.7

Seien T_1 und T_2 Spurmengen, dann ist das *faire Mischen zweier Spurmengen*, für das der Operator $\bowtie$ verwendet wird, definiert als:

$$T_1 \bowtie T_2 := \{t |\ \exists bs.\ \#bs = \infty \wedge filter(true, bs, t) \in T_1 \wedge filter(false, bs, t) \in T_2\}$$

□

Das faire Mischen zweier Spurmengen beschreibt die Menge, die entsteht, wenn jeweils zwei Elemente aus den beiden Spurmengen fair gemischt werden. Es ist somit unmöglich, daß nur ein echtes Präfix aus T_1 oder T_2 gemischt wird. Es erfolgt nur ein vollständiges Mischen von jeweils einem Element einer Spurmenge mit einem der anderen.

Eine direkte Folge der Aussage (13) im Satz 3.1 ist das Kompositionsprinzip für TSDs:

Korollar 3.2

Für beliebige TSD-DL-Ausdrücke TSD_1, TSD_2 gilt:

$$[\![TSD_1^{\underline{\omega}}]\!]_{\mathcal{TSD}} \bowtie [\![TSD_2^{\underline{\omega}}]\!]_{\mathcal{TSD}} = [\![(TSD_1 | TSD_2)^{\underline{\omega}}]\!]_{\mathcal{TSD}}$$

Hierbei ist $\bowtie$ wie in Definition 3.7 festgelegt. □

Eine Verwendung der Aussage (12) aus Satz 3.1 ergibt weiter:

Korollar 3.3

Für beliebige TSD-DL-Ausdrücke TSD_1, TSD_2 gilt:

$$[\![TSD_1^{\underline{\omega}}]\!]_{\mathcal{TSD}} \bowtie [\![TSD_2^{\underline{\omega}}]\!]_{\mathcal{TSD}} = [\![(TSD_1^{\underline{\omega}} | TSD_2^{\underline{\omega}})^{\underline{\omega}}]\!]_{\mathcal{TSD}}$$

□

Somit wird die Komposition von graphischen TSDs, die in den Normen durch eine mengenartige Vereinigung vorgenommen wird, auf der Semantik mit einem fairen Interleaving der Spurmengen modelliert.

Satz 3.4

Für beliebige TSD-DL-Ausdrücke TSD_1, TSD_2 gilt die folgende Aussage:

$$(TSD_1 \simeq TSD_2) \Rightarrow (TSD_1^{\underline{\omega}} \simeq TSD_2^{\underline{\omega}})$$

□

Beweis:

Voraussetzung: $P_{TSD_1}(t) = P_{TSD_2}(t)$

Zu zeigen ist:
(1) $P_{TSD_1^{\underline{\omega}}}(t) \Rightarrow P_{TSD_2^{\underline{\omega}}}(t)$
Annahme: $P_{TSD_1^{\underline{\omega}}}(t)$, das heißt:
$\exists ns.\forall n.(P_{TSD_1}(filter(n, ns, t))$
$\vee filter(n, ns, t) = \epsilon)$
Wegen der Voraussetzung $TSD_1 \simeq TSD_2$ gilt:
$\exists ns.\forall n.(P_{TSD_2}(filter(n, ns, t))$
$\vee filter(n, ns, t) = \epsilon)$
Somit: $P_{TSD_2^{\underline{\omega}}}(t)$

(2) $P_{TSD_2^{\underline{\omega}}}(t) \Rightarrow P_{TSD_1^{\underline{\omega}}}(t)$
Äquivalent zu (1)

□

Satz 3.5

Für beliebige TSD-DL-Ausdrücke $TSD_1, TSD_2, TSD_3, TSD_4$ gilt die folgende Aussage:

$$(TSD_1 \simeq TSD_2 \wedge TSD_3 \simeq TSD_4) \Rightarrow (TSD_1|TSD_3 \simeq TSD_2|TSD_4)$$

□

Beweis:

Die Aussage folgt unmittelbar aus dem Einsetzen der Definitionen. □

Satz 3.6

Für beliebige TSD-DL-Ausdrücke TSD_1, TSD_1', TSD_2, TSD_2',..., TSD_n, TSD_n' gilt die folgende Aussage:

$$\begin{array}{ccc} (TSD_1 \simeq TSD_1' \wedge & \ldots & \wedge\, TSD_n \simeq TSD_n') \\ & \Rightarrow & \\ (TSD_1|\ldots|TSD_n)^{\underline{\omega}} & \simeq & (TSD_1'|\ldots|TSD_n')^{\underline{\omega}} \end{array}$$

□

Beweis:

Induktion über n unter Verwendung des Satzes 3.5. Die Aussage erhält man durch die anschließende Verwendung von Satz 3.4. □

Mit Satz 3.6 ist die Erhaltung der Äquivalenz einer Schichtbeschreibung beim Austausch eines TSD-DL-Ausdrucks, der ein einzelnes TSD beschreibt, sichergestellt. Somit können äquivalente TSD-DL-Ausdrücke innerhalb einer Schichtbeschreibung ausgetauscht werden, ohne daß die Semantik verändert wird.

3.3.6 Methode zur Umwandlung von TSDs in TSD-DL-Ausdrücke

In diesem Kapitel wird eine Methode zur Umwandlung der graphischen Darstellung von TSDs in TSD-DL-Ausdrücke beschrieben. Mit dieser Methode wird einem einzelnen beliebigen TSD eine formale Semantik gegeben, wobei nur die szenarische Interpretation berücksichtigt wird. Anschließend wird ein Verfahren angegeben, mit dem dann die TSD-DL-Ausdrücke der einzelnen TSDs kombiniert werden, so daß eine komplette Beschreibung der TSDs eines Dienstes möglich ist.

Konzepte der Interpretation eines beliebigen TSDs

Bei einfachen TSDs ist die intuitive Vorgehensweise, die Verbindung von Aktionen an zwei SAPs mit einer Linie durch den TSD-DL-Operator $\rightarrow$, und die Abwesenheit einer zeitlichen Präzedenz mit $\sim$ zu modellieren, ausreichend für die Umwandlung der graphischen Darstellung eines TSDs in einen TSD-DL-Ausdruck.

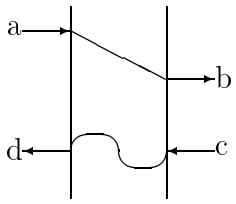

Abbildung 3.13: Beispiel eines TSDs

Das in Abbildung 3.13 dargestellte TSD ist in [ISO87b, CCI88d] zur Definition von TSDs verwendet worden. Dieses TSD wirft jedoch eine Interpretationsschwierigkeit auf. Eine mögliche Interpretation ist der TSD-DL-Ausdruck $(a \rightarrow b) \rightarrow (c \sim d)$, bei dem die Aktionen c und d der Aktion b nachgeordnet sind. Eine andere Interpretation beschränkt sich darauf, daß zwischen b und d keinerlei zeitliche Präzedenz existiert. Dies führt zum TSD-DL-Ausdruck $a \rightarrow ((b \rightarrow c) \sim d)$. Diese eigentlich naheliegende Definition ist in der Dienstspezifikation der Vermittlungsschicht [ISO87c, CCI88g] als eine verbale Ergänzung zum TSD hinzugefügt worden. In der Sicherungsschicht [ISO88b, CCI88f] fehlt dieser Zusatz. Bedeutet dies, daß hier nur die erste Interpretation zulässig ist? Es ist schwer, eine derartige Mißinterpretation in den Normen nachzuweisen, da bei Betrachtung der Mengen der akzeptierten Spuren die erste Interpretation eine echte Teilmenge der zweiten Interpretation ist. Sogar wenn bei genauerer Betrachtung der Implementierung durch ein Protokoll in den betreffenden Normen die zweite, also die umfassendere Interpretation, belegt wird, führt dies zu keinem Widerspruch mit der Dienstnorm, da bei dieser nur der Genauigkeit verschaffende Zusatz der Interpretation fehlt.

Für das Folgende wird die Verwendung der zweiten Interpretation festgelegt, da sie die geringsten Einschränkungen an einen Diensterbringer stellt, indem sie nur die wesentlichen Präzedenzen berücksichtigt. Eine Folge davon ist zudem, daß die zweite Interpretation die erste beinhaltet.

Eine schematische Interpretationsmethode für einzelne TSDs

Eine Möglichkeit, den in den Normen verwendeten TSDs eine auf TSD-DL basierte formale Semantik zu geben, ist die Angabe eines TSD-DL-Ausdrucks für jedes einzelne in den Normen vorkommende TSD. Da die Anzahl der in den ISO beziehungsweise CCITT Dokumenten vorkommenden TSDs beschränkt ist, ist diese Vorgehensweise ausreichend, den derzeitigen Stand zu beschreiben. Eine umfassendere Anwendbarkeit erhält man jedoch durch die Angabe einer schematischen Methode, die ein beliebiges TSD in einen TSD-DL-Ausdruck umwandelt. Diese Methode ist die Schnittstelle zu einer formalen Semantik. Bei der in dieser Arbeit eingeführten Methode werden nur TSDs mit disjunkten Aktionen berücksichtigt, da derartige TSDs in den ISO beziehungsweise CCITT Dokumenten unter Einbeziehung der Ursprungsinformation über den SAP ausschließlich vorkommen. Die Methode ist in vier Schritte aufgeteilt:

1. *Entwurf der Basispräzedenzrelation:*
   Als Ausgangspunkt dieses Verfahrens ist jede zeitliche Präzedenz eines TSDs zu ermitteln. Eine zeitliche Präzedenz wird mit dem TSD-DL-Symbol $\rightarrow$ beschrieben. Das *Präzedenzpaar* $a \rightarrow b$ stellt sicher, daß das Ereignis a vor dem Ereignis b erfolgt. Die *Basispräzedenzrelation* $BP(TSD)$ ist eine Menge von Präzedenzpaaren. Es werden zwei Arten von Präzedenzen unterschieden: Zum einen diejenigen zeitlichen Präzedenzen, die an einem Dienstzugangspunkt beobachtet werden. Diese sind somit an den vertikalen Linien eines TSDs ablesbar. Die zweite Art von zeitlichen Präzedenzen sind zwischen Ereignissen an verschiedenen Dienstzugangspunkten sichtbar. Diese werden durch die Verbindung zweier Ereignisse mit einer Linie erkennbar. Da die Tilde keine zeitliche Präzedenz beschreibt, wird sie nicht berücksichtigt. Somit erhält man für das in Abbildung 3.13 auf Seite 43 dargestellte TSD die Basispräzedenzrelation:

   $$\{a \rightarrow d, b \rightarrow c, a \rightarrow b\}$$

   Ein weiteres Beispiel für die Basispräzedenzrelation ist das TSD aus Abbildung 3.14.

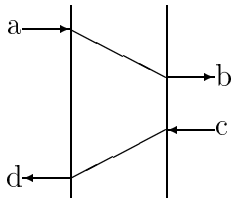

Abbildung 3.14: TSD mit vier Elementen

Das in Abbildung 3.14 dargestellte TSD ergibt die Basispräzedenzrelation:

$$\{a \to d, b \to c, a \to b, c \to d\}$$

2. *Entwurf des Basispräzedenzausdrucks:*
   Der *Basispräzedenzausdruck* $BPexpr(TSD)$ ist die Darstellung der Basispräzedenzrelation in TSD-DL. Dabei werden alle Elemente der Basispräzedenzrelation mit dem Operator $\circledast$ für die konjunktive Komposition verknüpft:

   $$BPexpr(TSD) \in \{bp_1 \circledast bp_2 \circledast \ldots \circledast bp_n | \{bp_1, \ldots, bp_n\} = BP(TSD)\}$$

   Daß $BPexpr(TSD)$ wohldefiniert ist, erkennt man daran, daß alle Elemente der Menge $\{bp_1 \circledast bp_2 \circledast \ldots \circledast bp_n | \{bp_1, \ldots, bp_n\} = BP(TSD)\}$ semantisch äquivalent sind. Dies ist eine Folge der Kommutativität und Assoziativität der Interpretation des Operators $\circledast$. Somit ist wegen der semantischen Äquivalenz der Elemente die Verwendung des Elementsymbols $\in$ bei der Definition sinnvoll.

3. *Bildung des Basis-*TSD-DL*-Ausdrucks:*
   Der *Basis-*TSD-DL*-Ausdruck* $Basic(TSD)$ beschreibt den Lebendigkeitsanteil eines TSDs, das heißt, entweder treten alle Aktionen auf oder keine. Dies wird durch einen TSD-DL-Ausdruck dargestellt, bei dem alle beteiligten Aktionen eines TSDs, die mit $ACT(TSD)$ bezeichnet werden, mit dem TSD-DL Operator $\sim$ verbunden werden:

   $$Basic(TSD) \in \{act_1 \sim act_2 \sim \ldots \sim act_n | \{act_1, \ldots, act_n\} = ACT(TSD)\}$$

   Dies ist nach den zu $BPexpr(TSD)$ äquivalenten Überlegungen wohldefiniert, da $\sim$ ein kommutativer und assoziativer Operator ist. Somit ergibt sich für die in Abbildungen 3.13 und 3.14 dargestellten TSDs:

   $$Basic(TSD_1) = Basic(TSD_2) = a \sim b \sim c \sim d$$

4. *Bildung des* TSD-DL*-Ausdrucks:*
   Der TSD-DL-Ausdruck, der die Semantik eines TSDs bei szenarischer Interpretation beschreibt, ist die konjunktive Verknüpfung des Basis-TSD-DL-Ausdrucks mit dem Basispräzedenzausdruck:

   $$(Basic(TSD)) \circledast (BPexpr(TSD))$$

Mit dem oben angegebenen Verfahren ist es möglich, für einzelne TSDs einen TSD-DL-Ausdruck zu bestimmen, der einer szenarischen Interpretation eines TSDs entspricht.

Zur Konstruktion eines TSD-DL-Ausdrucks, der eine Schicht beschreibt, werden die TSD-DL-Ausdrücke der einzelnen TSDs wie folgt kombiniert:

$$(TSD_1 | TSD_2 | \ldots | TSD_n)^{\underline{\omega}}$$

Hierbei ist TSD_i ein TSD-DL-Ausdruck, der die szenarische Interpretation eines einzelnen TSDs beschreibt. Somit wird die szenarische Interpretation des TSD-Anteils eines Dienstерbringers einer Schicht durch die alternative Kombination der einzelnen TSDs mit dem

TSD-DL-Ausdruck $(TSD_1|TSD_2|\ldots|TSD_n)$ dargestellt. Da nur eine vollständige Interpretation alle möglichen Abläufe erfaßt, wird die Wiederholung dieser TSDs durch die Verwendung des Operators $\underline{\omega}$ einbezogen.

Die einzelnen TSDs werden unter Umständen durch TSD-DL-Ausdrücke beschrieben, die vereinfachbar sind. Eine adhoc Lösung ist der Beweis der Äquivalenz eines durch Überlegung gefundenen Kandidaten mit dem durch das Verfahren bestimmten TSD-DL-Ausdrucks bei einer szenarischen Interpretation, also ohne Wiederholungsoperator. Dies ist sehr einfach, da nur endliche Spuren betrachtet werden. Der vereinfachte TSD-DL-Ausdruck kann nun in die Dienstbeschreibung eingesetzt werden, da dies wegen Satz 3.6 semantikerhaltend ist.

3.3.7 Beispiele für schematisch gewonnene TSD-DL-Ausdrücke

In diesem Kapitel werden einige Beispiele für schematisch gewonnene TSD-DL-Ausdrücke nach dem Verfahren aus Kapitel 3.3.6 vorgestellt. Hierbei wird keine Wiederholung berücksichtigt, da zur Veranschaulichung nur die szenarische Interpretation genügt.

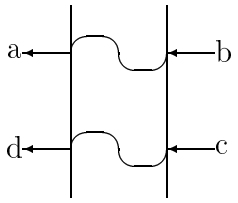

Abbildung 3.15: Beispiel eines TSDs

Das in Kapitel 3.3.6 beschriebene Verfahren führt für das in Abbildung 3.15 dargestellte TSD zum TSD-DL-Ausdruck:

$$(a \sim b \sim c \sim d) \circledast (a \rightarrow d) \circledast (b \rightarrow c)$$

mit der Semantik:

$$[\![(a \sim b \sim c \sim d) \circledast (a \rightarrow d) \circledast (b \rightarrow c)]\!]_{TSD} = \{ \quad <a,d,b,c>, <a,b,d,c>, <a,b,c,d>, \\ <b,a,d,c>, <b,a,c,d>, <b,c,a,d>\}$$

Es gilt:

$$((a \sim b \sim c \sim d) \circledast (a \rightarrow d) \circledast (b \rightarrow c)) \simeq ((a \rightarrow d) \sim (b \rightarrow c))$$

Das in Abbildung 3.8 auf Seite 25 dargestellte TSD führt zum TSD-DL-Ausdruck:

$$(a \sim b \sim c \sim d) \circledast (b \rightarrow c) \circledast (c \rightarrow d) \circledast (a \rightarrow c)$$

mit der Semantik:

$$[\![(a \sim b \sim c \sim d) \circledast (b \rightarrow c) \circledast (c \rightarrow d) \circledast (a \rightarrow c)]\!]_{TSD} = \{<a,b,c,d>, <b,a,c,d>\}$$

Es gilt:

$$((a \sim b \sim c \sim d) \circledast (b \rightarrow c) \circledast (c \rightarrow d) \circledast (a \rightarrow c)) \simeq ((a \sim b) \rightarrow (c \rightarrow d))$$

Es ist auch möglich, daß TSDs ohne Tildesymbol nebenläufige Teile besitzen, wie mit Abbildung 3.16 demonstriert wird.

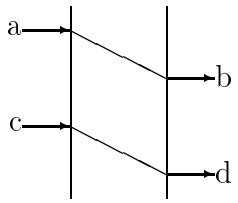

Abbildung 3.16: TSD mit nebenläufigen Teilen

Die schematische Umsetzung des TSDs aus Abbildung 3.16 nach dem in Kapitel 3.3.6 verwendeten Verfahren ergibt den TSD-DL-Ausdruck:

$$(a \sim b \sim c \sim d) \circledast (a \rightarrow c) \circledast (b \rightarrow d) \circledast (a \rightarrow b) \circledast (c \rightarrow d)$$

mit der Semantik:

$$[\![(a \sim b \sim c \sim d) \circledast (a \rightarrow c) \circledast (b \rightarrow d) \circledast (a \rightarrow b) \circledast (c \rightarrow d)]\!]_{TSD} = \{<a, b, c, d>, <a, c, b, d>\}$$

Es gilt:

$$((a \sim b \sim c \sim d) \circledast (a \rightarrow c) \circledast (b \rightarrow d) \circledast (a \rightarrow b) \circledast (c \rightarrow d)) \simeq (a \rightarrow ((b \sim c) \rightarrow d))$$

3.4 Formale Semantik ohne Einschränkung des Umgebungsverhaltens

In Kapitel 3.3 ist eine Semantik für TSDs definiert worden, die auf der Beschreibung von Sequenzen von Aktionen basiert. Diese Definition wird durch die betreffenden Normen, unter anderem in [ISO87b, CCI88d], nahegelegt. Bei der Interpretation der TSDs werden hierbei Eingabe- und Ausgabeaktionen nicht substantiell unterschieden, da bei der Bildung der Sequenzen keine unterschiedliche Behandlung der Aktionen erfolgt.

Es ist sinnvoll, den Diensterbringer einer Schicht als offenes System nach [BDD+93] zu betrachten[5], da in den Dienstnormen nur die Eigenschaften des Diensterbringers spezifiziert werden. Hierbei wird der Diensterbringer als Komponente betrachtet, an die die Umgebung, das sind die Dienstbenutzer, Anfragen stellt. Dies wird in Abbildung 1.3 auf Seite

[5] In Focus wird bei der Sicht auf ein verteiltes System ein offenes von einem geschlossenem System unterschieden. Letzteres beschreibt einen von der Außenwelt isolierten Komplex, der Aktionen eigenständig ausführt. Bei einer Sicht auf ein offenes System wird eine Komponente und die Umgebung, die das Verhalten der Komponente beeinflussen kann, unterschieden. Man beachte, daß der Begriff offenes System im ISO/OSI Schichtenmodell in einem anderen Zusammenhang, siehe dazu Definition 2.2, verwendet wird.

6 verdeutlicht. Eine derartige Vorgehensweise ist vorzuziehen, da nur der Diensterbringer implementiert werden soll, und nicht die Dienstbenutzer. Aufgrund der oben angeführten gleichartigen Behandlung von Eingabe- und Ausgabeaktionen beschreibt die in Kapitel 3.3 angegebene Semantik eine Einschränkung des Umgebungsverhaltens, indem möglicherweise auch eine Reihenfolge der Eingabeaktionen vorgeschrieben wird. Es ist jedoch dem Diensterbringer nicht möglich, auf diese Aktionen Einfluß zu nehmen, da sie nicht von ihm verursacht werden. Dies ist bei einer späteren Implementierung des Diensterbringers zu berücksichtigen. Deswegen ist ein freies Umgebungsverhalten die Grundannahme, die bei einer Implementierung zu beachten ist. Aus diesem Grund ist eine Einschränkung des Umgebungsverhaltens unter anderem nach [AL90, AL93, SDW93, BS94] in jedem Fall zu vermeiden. Derartige Einschränkungen der Dienstbenutzer mit der bisher verwendeten Semantik werden am folgenden Beispiel verdeutlicht.

Beispiel 3.2

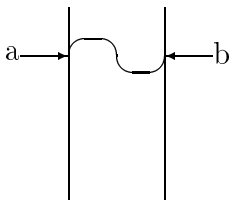

Abbildung 3.17: TSD mit Einschränkung des Umgebungsverhaltens

Das in Abbildung 3.17 dargestellte TSD ist aus der Dienstdefinition der Transportschicht [ISO84c, CCI88h] entnommen. Dabei sind die Bezeichnungen der Dienstelemente zu a, b vereinfacht worden. Die Semantik dieses TSDs ist:

$$[\![(a \sim b)^{\underline{\omega}}]\!]_{TSD} = \{t \in \{a,b\}^{\omega} | \#a \copyright t = \#b \copyright t\}$$

Somit beschreibt dieses TSD, daß die gleiche Anzahl der Eingabeaktionen a und b zu erfolgen hat. Dadurch ist das Umgebungsverhalten eingeschränkt, da ausschließlich die Umgebung, das heißt die Dienstbenutzer, die Aktionen a und b erzeugt. Das Verhalten der Komponente wird hierbei nur insofern festgelegt, als keine Ausgabeaktion erfolgt. □

Bei den Texten, die das Beschreibungsmittel TSD definieren [CCI88d, ISO87b], wird versucht, das Problem der Einschränkung des Umgebungsverhaltens implizit durch den Punkt 2 in der folgenden Definition zu beheben:

TSDs geben an:

1. die Folge von Ereignissen an jeder Übergabestelle zwischen Benutzern und Erbringer;

2. soweit zutreffend, die Folge von Ereignissen zwischen Partner-Benutzern.

Dies ist natürlich keine Grundlage für eine formale Semantik, da zum einen der Begriff "soweit zutreffend" nicht näher definiert wird, und zum anderen nicht beschrieben wird, was sonst geschieht. Desweiteren sind auch TSDs denkbar, die das Umgebungsverhalten an einem SAP einschränken, also das Verhalten eines einzigen Benutzers beschränken. Dadurch ist die zuvor aus [CCI88d, ISO87b] zitierte Definition nicht mehr anwendbar.

Beispiel 3.3

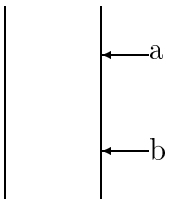

Abbildung 3.18: TSD mit Einschränkung des Umgebungsverhaltens

Beim TSD in Abbildung 3.18 wird eine Beschränkung des Umgebungsverhaltens an einem Dienstzugangspunkt beschrieben. Somit ist der Punkt 2 der zuvor aus [CCI88d, ISO87b] zitierten Definition nicht mehr zutreffend. □

Zur Vermeidung der Einschränkung des Umgebungsverhaltens wird in dieser Arbeit der Assumption/Commitment Stil, der auch als Relay/Guarantee Stil bezeichnet wird, herangezogen [AL90, AL93, Bro94, Jon83, SDW93]. Hierzu wird das korrekte Verhalten einer Komponente mit dem Prädikaten-Paar (A, C) beschrieben. Das Einhalten der Spielregeln durch die Umgebung wird mit der Assumption A dargestellt, das der Komponente mit dem Prädikat C. Das korrekte Verhalten einer Komponente ist dadurch definiert, daß sie ihrer Verpflichtung nachzukommen hat, falls sich die Umgebung an die Spielregeln hält. Somit ist auch ein Ablauf korrekt, wenn die Umgebung die Spielregeln verletzt hat. Bei der spurbasierten Semantik bedeutet dies:

$$P(t) := (A(t) \Rightarrow C(t))$$

Auf Spurmengen angewendet erhält man:

$$T := \{t|\ C(t)\} \cup \{t|\ \neg A(t)\}$$

Bei dieser Darstellung ist es jedoch möglich, daß zuerst die Komponente ihre Verpflichtung verletzt und dadurch nachfolgend die Umgebung ihrerseits die Spielregeln, die sie aber bis zu diesem Zeitpunkt erfüllt hat, nicht mehr einhalten kann. Dadurch ist die obige Formel erfüllt, obwohl die Komponente als erstes die Spielregeln verletzt hat. Um dies zu beheben, wird in [AL93] eine andere Notation zur Widerspiegelung dieses Sachverhalts verwendet:

$$A \longrightarrow\!\triangleright\, C$$

Damit wird gefordert, daß C mindestens genauso lang erfüllt sein muß wie A. Auf Spurmengen angewendet erhält man:

$$T := \{t|\ A(t) \wedge C(t)\} \cup \{t|\ \exists t'.\ t' \sqsubseteq t \wedge \neg A(t') \wedge (\exists t''.\ C(t' \circ t''))\}$$

Da die Menge $\{t|A(t) \wedge C(t)\}$ bereits im Kapitel 3.3 beschrieben wurde, ist in diesem Kapitel nur noch die Menge $\{t|\ \exists t'.\ t' \sqsubseteq t \wedge \neg A(t') \wedge (\exists t''.\ C(t' \circ t''))\}$ für TSDs zu bestimmen. Diese Menge beschreibt diejenigen Spuren, bei denen die Umgebung als erstes die Spielregeln verletzt. Sei t eine zu dieser Menge gehörende Spur, dann ist es erforderlich, daß es ein Präfix t' gibt, bei dem die Assumption verletzt ist. Für dieses Präfix t' muß sich der Diensterbringer an die Spielregeln gehalten haben. Dies kann in erster Näherung mit $C(t')$ dargestellt werden. Da jedoch meist das Commitment einen Lebendigkeitsanteil besitzt, ist diese Aussage nicht zutreffend. So ist die Erfüllung des Lebendigkeitsanteils des Commitments oft zum Verletzungszeitpunkt der Assumption nicht möglich. Somit genügt eine Überprüfung des Sicherheitsanteils des Commitments, der mindestens genauso lang erfüllt zu sein hat, wie die Assumption. Dies wird durch die Existenz der Spur t'' und $C(t' \circ t'')$ sichergestellt.

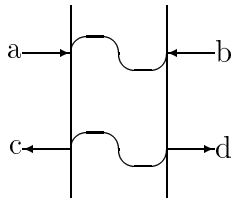

Abbildung 3.19: TSD mit Einschränkung des Umgebungsverhaltens

Das in Abbildung 3.19 dargestellte TSD, das aus der Dienstdefinition der Vermittlungsschicht [ISO87c, CCI88h] entnommen ist, beschreibt sowohl eine Einschränkung des Umgebungsverhaltens, als auch des Komponentenverhaltens. Dieses TSD wird bei Verwendung des Assumption/Commitment Stils folgendermaßen interpretiert: Falls die Umgebung eine gleiche Anzahl von Eingabeaktionen a und b zur Verfügung stellt, so reagiert die Komponente mit der gleichen Anzahl von Ausgabeaktionen c und d

Die Dienstnormen unterscheiden bei der Interpretation von TSDs nicht zwischen Anforderungen an die Dienstbenutzer und Anforderungen an den Diensterbringer. Durch die Verwendung des Assumption/Commitment Stils bei der Spezifikation werden beide Anteile klar getrennt, und somit wird eine saubere Vorgehensweise bei der Dienstspezifikation ermöglicht. Das Einhalten der Spielregeln durch die Dienstbenutzer wird mit der Assumption Komponente der Spezifikation, das Einhalten der Spielregeln durch den Diensterbringer mit der Commitment Komponente beschrieben. Durch diese Aufteilung ist eine nachträgliche Erweiterung der Semantik um die Spuren, bei denen die Dienstbenutzer zuerst die Spielregeln verletzen, schematisch möglich. Durch die Bezugnahme auf den Assumption/Commitment Stil wird die folgende Semantikerweiterung anschaulich verdeutlicht.

In diesem Kapitel wird die im Kapitel 3.3 gegebene formale Semantik dahingehend erweitert, daß das Umgebungsverhalten nicht mehr eingeschränkt wird. Dazu wird eine

Erweiterung der Spurmenge beschrieben. Es zeigt sich jedoch, daß dann die TSDs mit einer auf einfachen Spurmengen basierenden Semantik nicht mehr kompositional sind.

3.4.1 Konzepte der Semantik

Bezeichne I die Menge der Eingabeaktionen, das heißt diejenigen Aktionen, die die Umgebung tätigt, und O die Menge der Ausgabeaktionen, also Aktionen des Diensterbringers. Beide Mengen seien disjunkt. Dies bedeutet keine Einschränkung, da die Aktionen bei Bedarf mit einer Ursprungsinformation, wie im Kapitel 3.3.1, ergänzt werden können und somit unterscheidbar sind. Act bezeichne die Menge aller Aktionen: $Act := I \cup O$

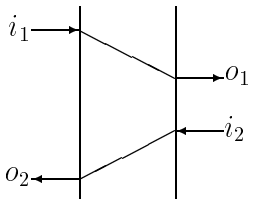

Abbildung 3.20: TSD_1

Das TSD aus Abbildung 3.20 hat bei einer vollständigen Interpretation die Semantik unter der Einschränkung des Umgebungsverhaltens:

$$[\![((i_1 \rightarrow o_1) \rightarrow (i_2 \rightarrow o_2))^{\underline{\omega}}]\!]_{TSD} = T_1$$

Für die Bestimmung einer Semantik ohne Einschränkung des Umgebungsverhaltens ist die Definition der Mengen I und O notwendig. Im folgenden wird für das konkrete Beispiel TSD_1 aus Abbildung 3.20 für die Bezeichnung der Eingabemenge I_1 und der Ausgabemenge O_1 verwendet.

$$I_1 := \{i_1, i_2, \ldots\}$$
$$O_1 := \{o_1, o_2, \ldots\}$$

Um die Semantik von TSDs ohne Einschränkung des Umgebungsverhaltens zu beschreiben, ist die ursprüngliche Spurmenge um folgende Elemente zu erweitern:

fehlerhafte Eingabeaktionen: Damit werden Eingabeaktionen bezeichnet, nach deren Auftreten die Ergänzung zu einer korrekten Spur nicht mehr möglich ist. Derartige Spuren sind zu berücksichtigen, da damit eine erstmalige Verletzung des Sicherheitsanteils der Assumption erfolgt. Bei TSD_1 aus Abbildung 3.20 fällt die Spur $<i_2>$ darunter. Hier erfolgt die Eingabeaktion i_2 ohne die zuvor notwendigen Aktionen i_1 und o_1. Darunter fallen auch alle Spuren, die mit einer von i_1 verschiedenen Eingabeaktion beginnen.

fehlende Eingabeaktionen: Das sind Eingabeaktionen, die zur Vervollständigung eines TSDs noch ausstehen. Dies bedeutet eine Verletzung des Lebendigkeitsanteils der

Assumption. Ein Beispiel hierfür ist bei TSD_1 aus Abbildung 3.20 die Spur $<i_1, o_1>$, da die Eingabeaktion i_2 von der Umgebung nicht mehr zur Verfügung gestellt wird, und somit der Diensterbringer unverschuldet eine Vervollständigung des TSDs nicht mehr bewirken kann.

Eine Erweiterung um diese Elemente, sowie die ursprüngliche Spurmenge, beschreiben dann die Semantik von TSDs ohne Einschränkung des Umgebungsverhaltens. Das Einhalten der Spielregeln wird in [BDDW91] dagegen mittels einer sogenannten Strategie modelliert. Ausgehend von dieser Strategie wird die Spurmenge konstruiert, die ein korrektes Systemverhalten widerspiegelt. In der vorgestellten Arbeit wird ein anderes Verfahren vorgestellt. Ausgehend von einer Spurmenge, die ein korrektes Verhalten der Umgebung vorschreibt, wird die Spurmenge dahingehend erweitert, daß ein beliebiges Umgebungsverhalten möglich ist. Diese Zweiteilung der Semantik wird durch die Normen nahegelegt, da nur Überlegungen zur Interpretation unter Einschränkung des Umgebungsverhaltens vorgenommen werden. Ein weiterer Grund dafür ist, daß bei den bisherigen Formalisierungen einer Dienstnorm ausschließlich der Anteil spezifiziert wurde, bei dem sowohl die Assumption als auch das Commitment erfüllt ist.

3.4.2 Erweiterung um fehlerhafte Eingaben

Ein Grundprinzip bei der Modellierung von reaktiven Systemen ist, daß es der Umgebung jederzeit erlaubt ist, eine Eingabe zu tätigen. Dieses Prinzip wird zum Beispiel bei den I/O Automaten [LT89] mit dem Begriff *input enabledness* bezeichnet.

Es werden zwei Arten von Eingaben unterschieden. Zum einen Eingabeaktionen, die zu einer korrekten Spur ergänzt werden können, und solche, die nicht mehr zu einer korrekten Spur ergänzt werden können. Letztere werden im folgenden mit dem Begriff *fehlerhafte Eingaben* bezeichnet. Dies entspricht einer erstmaligen Verletzung des Sicherheitsanteils der Assumption. Erfolgt eine fehlerhafte Eingabeaktion, so hat die Umgebung ihre Verpflichtung vor der Komponente verletzt. Der Diensterbringer kann sich ab diesem Zeitpunkt beliebig verhalten. Diese Erweiterung einer Spurmenge T unter Berücksichtigung der Menge von Eingabeaktionen I und Ausgabeaktionen O wird mit $E_{false_inp}(T, I, O)$ bezeichnet.

Definition 3.8

Für eine Spurmenge T, eine Menge von Eingabeaktionen I und eine Menge von Ausgabeaktionen O ist die *Erweiterung um fehlerhafte Eingaben* $E_{false_inp}(T, I, O)$ festgelegt als:

$$\begin{aligned} E_{false_inp}(T, I, O) \quad := \{t \circ i\&x \mid \quad & i \in I, x \in (I \cup O)^\omega \\ & \wedge\ t \in PRE(T) \\ & \wedge\ t \circ <i> \notin PRE(T)\} \end{aligned}$$

□

Hierbei beschreibt PRE, wie in Kapitel 2.2 erwähnt, den Präfixabschluß einer Spurmenge. Die Spur $t \circ i\&x$ ist genau dann in der Erweiterung um fehlerhafte Eingaben

enthalten, wenn t ein Präfix einer korrekten Spur, und $t \circ <i>$ kein Präfix einer korrekten Spur ist, und somit eine korrekte Ergänzung nicht mehr möglich ist.

Beispiel 3.4

Seien im folgenden T_1, I_1, O_1 wie auf Seite 51 für Abbildung 3.20 definiert. Es gilt für alle Spuren $x \in (I \cup O)^\omega$:

$$i_2 \& x \notin T_1$$

$$i_2 \& x \in E_{false_inp}(T_1, I_1, O_1)$$

Die erste Aussage ist zutreffend, da der Eingabeaktion i_2 ein $<i_1, o_1>$ vorauszugehen hat. Die Erweiterung um fehlerhafte Eingaben beinhaltet jedoch diese Spur mit der vorzeitig erfolgten Eingabeaktion i_2. Es ist auch jede Spur, die mit einer von i_1 verschiedenen Aktion beginnt, in $E_{false_inp}(T_1, I_1, O_1)$ enthalten:

$$a \in I_1 \backslash \{i_1\} \Rightarrow a \& x \in E_{false_inp}(T_1, I_1, O_1)$$

□

Satz 3.7

Für die Spurmengen T', T'' und die Mengen von Aktionen I, O gilt:

$$PRE(T') = PRE(T'') \Rightarrow E_{false_inp}(T', I, O) = E_{false_inp}(T'', I, O)$$

□

Beweis:

Die Aussage folgt direkt aus der Annahme $PRE(T') = PRE(T'')$ und dem Einsetzen in die Definition 3.8. □

Es gilt sogar eine Verschärfung der Aussage. Bei obigem Satz genügt in der Prämisse auch die Beschränkung der Gleichheit auf endliche Präfixe. Dazu wird zuerst die Menge der endlichen Präfixe definiert.

Definition 3.9

Für eine Spurmenge T ist der *endliche Präfixabschluß* $FINPRE(T)$ definiert als:

$$FINPRE(T) := \{t_p |\ \#t_p < \infty \wedge \exists t.\ t_p \sqsubseteq t \wedge t \in T\}$$

□

Satz 3.8

Für die Spurmengen T', T'' und die Mengen von Aktionen I, O gilt:

$$FINPRE(T') = FINPRE(T'') \Rightarrow E_{false_inp}(T', I, O) = E_{false_inp}(T'', I, O)$$

□

Beweis:

Dazu genügt es zu zeigen, daß in der Definition von $E_{false_inp}(T,I,O)$ jede Verwendung von $PRE(T)$ durch $FINPRE(T)$ ersetzt werden kann.

$$\begin{aligned} E_{false_inp}(T,I,O) \quad &= \{t \circ i\&x | \quad i \in I, x \in (I \cup O)^\omega \\ &\qquad \wedge\ t \in PRE(T) \wedge t \circ <i> \notin PRE(T)\} \end{aligned}$$

Fallunterscheidung nach $\#t \neq \infty$ ergibt:

$$\begin{aligned} E_{false_inp}(T,I,O) \quad &= \{t \circ i\&x | \quad i \in I, x \in (I \cup O)^\omega \wedge \\ &\qquad (\#t \neq \infty \wedge t \in PRE(T) \wedge t \circ <i> \notin PRE(T)) \\ &\qquad \vee (\#t = \infty \wedge t \in PRE(T) \wedge t \circ <i> \notin PRE(T))\} \end{aligned}$$

Der zweite Fall $\#t = \infty$ führt zu einem Widerspruch, da dann $t = t \circ <i>$ gilt und damit $t \in PRE(T) \wedge t \notin PRE(T)$. Somit erhält man:

$$\begin{aligned} E_{false_inp}(T,I,O) \quad &= \{t \circ i\&x | \quad i \in I, x \in (I \cup O)^\omega \wedge \\ &\qquad (\#t \neq \infty \wedge t \in PRE(T) \wedge t \circ <i> \notin PRE(T))\} \\ &= \{t \circ i\&x | \quad i \in I, x \in (I \cup O)^\omega \wedge \\ &\qquad t \in FINPRE(T) \wedge t \circ <i> \notin FINPRE(T)\} \end{aligned}$$

Woraus die Aussage folgt.

□

Somit führt die Gleichheit der Präfixmengen zur Gleichheit der Erweiterung um fehlerhafte Eingaben $E_{false_inp}(T,I,O)$ dieser Mengen. Man beachte, daß aus der Gleichheit der Präfixmengen nicht die Gleichheit der Spurmengen folgt.

Beispiel 3.5

Sei

$$T' := [\![(a \rightarrow b)^{\underline{\omega}}]\!]_{\mathcal{TSD}}$$

und

$$T'' := [\![((a \rightarrow b)|a)^{\underline{\omega}}]\!]_{\mathcal{TSD}}$$

Es gilt: $T' \neq T''$ da $<a> \notin T'$ und $<a> \in T''$. Wegen $FINPRE(T') = FINPRE(T'')$ gilt für beliebige Mengen I und O: $E_{false_inp}(T',I,O) = E_{false_inp}(T'',I,O)$ □

3.4.3 Erweiterung um fehlende Eingaben

Der zweite Bestandteil der Erweiterung beschreibt das Ausbleiben einer Eingabeaktion. Der Diensterbringer hat hierbei seinen Anteil erledigt, nur fehlt zur Vervollständigung

noch mindestens eine Eingabeaktion, der wiederum auch Aus- und Eingabeaktionen folgen können. So gilt für das in Abbildung 3.20 auf Seite 51 dargestellte TSD_1:

$$<i_1, o_1> \notin T_1$$

$$<i_1, o_1> \notin E_{false_inp}(T_1, I_1, O_1),$$

da die Aktionen i_2 und o_2 noch ausstehen. Dies beschreibt eine Verletzung des Lebendigkeitsanteils der Assumption. Eine Erweiterung, die derartige Spuren berücksichtigt, wird wie folgt definiert:

Definition 3.10

Für eine Spurmenge T, eine Menge von Eingabeaktionen I und eine Menge von Ausgabeaktionen O ist die *endliche Erweiterung um fehlende Eingaben* $E'_{missing_inp}(T, I, O)$ festgelegt als:

$$\begin{aligned} E'_{missing_inp}(T, I, O) \quad := \{t| \quad & t \notin T \\ & \wedge \exists i \in I.\ t \circ <i> \in PRE(T) \\ & \wedge (\forall o \in O.\ t \circ <o> \notin PRE(T) \vee <o> \in PRE(T))\} \end{aligned}$$

□

Durch $t \notin T \wedge t \circ <i> \in PRE(T)$ wird sichergestellt, daß noch mindestens eine Eingabeaktion zu erfolgen hat. Durch diese Bedingung wird ebenfalls sichergestellt, daß t endlich ist. Wäre t unendlich, so würde $t = t \circ <i>$, und somit $t \notin T \wedge t \in PRE(T)$ gelten. Da wegen der Unendlichkeit von t dann $t \in PRE(T) \Rightarrow t \in T$ gälte, würde dies zum Widerspruch führen. Die Ausgabeaktionen werden in zwei Gruppen eingeteilt. Zum einen gibt es noch ausstehende Ausgabeaktionen, bei denen die Komponente die Spur t noch weiter zu ergänzen hat. Hierbei hat der Diensterbringer seinerseits noch zu agieren. Somit ist eine derartige Spur in den Erweiterungen um fehlerhafte Eingaben nicht enthalten. Zum anderen gibt es die spontanen Ausgabeaktionen, die ohne eine vorhergehende Eingabe erfolgen können. Aufgrund der Definition der Wiederholung bei TSDs, die als ein beliebiges Interleaving eines TSDs mit sich selbst festgelegt ist, können diese Ausgabeaktionen jederzeit auftreten, ohne bisherige Verpflichtungen des Diensterbringers zu erfüllen. Sie erzeugen höchstens neue Verpflichtungen. Diese Aktionen sind daher bei der Betrachtung, ob der Diensterbringer seine Verpflichtungen erfüllt hat, nicht mehr zu berücksichtigen. Bei spontanen Aktionen gilt: $<o> \in PRE(T)$. Das folgende Beispiel verdeutlicht eine derartige Aktion.

Beispiel 3.6

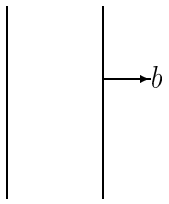

Abbildung 3.21: TSD mit einer spontanen Ausgabeaktion

Im TSD der Abbildung 3.21 kann die Ausgabeaktion b ohne eine vorhergehende Eingabeaktion erfolgen. Sei T die Spurmenge dieses TSDs. Somit gilt für jede Spur t in diesem Fall:

$$t \in PRE(T) \Rightarrow t \circ <b> \in PRE(T)$$

□

Derartig spontan auftretende, ungetriggerte Aktionen sind die einzigen Ausgabeaktionen, die noch erfolgen dürfen, falls die Spur in den Erweiterungen um fehlende Eingaben enthalten sein soll. Dies ist durch

$$\forall o \in O.\ t \circ <o> \notin PRE(T) \vee <o> \in PRE(T)$$

sichergestellt.

Die Anwendung der Definition 3.10 auf das in Abbildung 3.20 auf Seite 51 dargestellte TSD ergibt:

$$<i_1, o_1> \in E'_{missing_inp}(T_1, I_1, O_1)$$

da :

$$<i_1, o_1> \notin T_1 \wedge <i_1, o_1, i_2> \in PRE(T_1) \wedge \forall o \in O_1.\ <i_1, o_1, o> \notin PRE(T_1)$$

Wird ein solches TSD durch eine Funktion implementiert, so ist wegen Stetigkeitsüberlegungen auch die Approximation von unendlichen Elementen einzubeziehen. Bei der folgenden Überlegung erzeuge die Umgebung immer wieder die Eingabeaktion i_1, jedoch nie die Eingabeaktion i_2. Auf jedes i_1 reagiere der Diensterbringer mit einem o_1. Darauf erzeuge die Umgebung erneut ein i_1. Eine Folge der Stetigkeit einer derartigen Funktion ist, daß die Spur $\mathsf{fix}\boldsymbol{\lambda} s.i_1 \& o_1 \& s$ akzeptiert wird. Dies führt zu folgender Erweiterung der Definition 3.10:

Definition 3.11

Für eine Spurmenge T, eine Menge von Eingabeaktionen I und eine Menge von

Ausgabeaktionen O ist die *Erweiterung um fehlende Eingaben* $E_{missing_inp}(T, I, O)$ festgelegt als:

$$E_{missing_inp}(T, I, O) := LIM(E'_{missing_inp}(T, I, O))\backslash T$$

□

Ohne den expliziten Ausschluß von T wären auch Elemente in der Erweiterung, bei denen sowohl die Assumption als auch das Commitment erfüllt ist. Diese Elemente werden unter Umständen durch den Approximationsprozeß erzeugt. Sie werden in der Definition 3.11 explizit ausgeschlossen, da in dieser Arbeit eine echte Erweiterung der Spurmengen festgelegt wird und somit die Erweiterung nur diejenigen Spuren beschreibt, bei denen die Assumption zuerst verletzt ist. Dadurch wird eine saubere methodische Trennung der Spuren, bei denen die Assumption verletzt ist, von denjenigen, bei denen sowohl die Assumption als auch das Commitment erfüllt ist, erreicht.

Beispiel 3.7

In den Normen einiger Schichten [ISO88c, ISO88b, ISO87c, ISO84c, ISO84b, CCI88e, CCI88f, CCI88g, CCI88h] tritt das TSD aus Abbildung 3.22 auf.

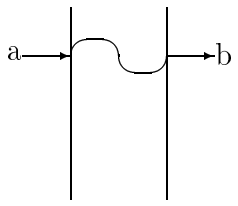

Abbildung 3.22: Beispiel TSD

In dem in Abbildung 3.22 gezeigten TSD wird festgelegt, daß die Eingabeaktion a keine Präzedenz zur Ausgabeaktion b besitzt. Gemäß dem in Kapitel 3.3.6 angegebenen Verfahren besitzt das TSD bei vollständiger Interpretation die Semantik unter Einschränkung des Umgebungsverhaltens:

$$[\![(a \sim b)^{\underline{\omega}}]\!]_{TSD}$$

Eine Folgerung aus Satz 3.1 auf Seite 37 ist:

$$[\![(a \sim b)^{\underline{\omega}}]\!]_{TSD} = [\![(a \rightarrow b | b \rightarrow a)^{\underline{\omega}}]\!]_{TSD}$$

Eine graphische Darstellung dieses Sachverhalts ergibt die Abbildung 3.22.

Abbildung 3.23: Aufspaltung des Beispiel TSDs

Eine mögliche Implementierung des TSDs aus Abbildung 3.22 ist die Interpretation ausschließlich durch das linke TSD aus Abbildung 3.23. Gerechtfertigt wäre dies durch die problematische Implementierung des rechten TSDs, da es eine Komponente beschreibt, die eine Kenntnis der Zukunft besitzt: Die Komponente gibt ein b aus auf den Verdacht hin, daß später ein a eingegeben wird. Dadurch werden jedoch etwaige Implementierungen beschränkt, da die Spur $<b, a>$ nicht mehr berücksichtigt wird. Sicherlich ist die alleinstehende Implementierung des rechten TSDs aus Abbildung 3.23 problematisch, jedoch ist eine Implementierung in Kombination mit anderen TSDs ohne weiteres möglich, wie dies mit Abbildung 3.24 verdeutlicht wird.

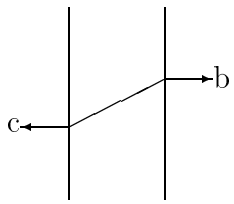

Abbildung 3.24: Beispiel eines TSDs zur Implementierbarkeit

Durch die Hinzunahme des TSDs aus Abbildung 3.24 ist auch das rechte TSD aus Abbildung 3.23 ohne weiteres implementierbar. Ein so beschriebener Diensterbringer gibt ein b aus, wartet auf die Eingabe a. Erfolgt diese Eingabe nicht, so gibt die Komponente ein c aus.

Diese Überlegungen führen dazu, daß auch das rechte TSD aus Abbildung 3.23 bei der Semantik zu berücksichtigen ist. Es gilt für $a \in I$ und $b \in O$:

$$E_{missing_inp}([\![(a \sim b)^{\underline{\omega}}]\!]_{TSD}, I, O) = \{t \in \{a, b\}^{\omega} | \#a \copyright t < \#b \copyright t\}$$

□

Satz 3.9

Für eine Spurmenge T und die Mengen von Aktionen I, O gilt:

$$PRE(T) = T \Rightarrow E_{missing_inp}(T, I, O) = \emptyset$$

□

Beweis:

Es gilt:

$$\begin{aligned} E'_{missing_inp}(T,I,O) \quad := \{t| \quad & t \notin T \\ & \wedge \exists i \in I.\ t \circ <i> \in PRE(T) \\ & \wedge (\forall o \in O.\ t \circ <o> \notin PRE(T) \vee <o> \in PRE(T))\} \end{aligned}$$

Aus $\exists i \in I.\ t \circ <i> \in PRE(T)$ folgt $t \in PRE(T)$ und aus $PRE(T) = T$ folgt der Widerspruch $t \in T$, womit $E'_{missing_inp}(T,I,O) = \emptyset$ folgt. Und aus

$$E_{missing_inp}(T,I,O) := LIM(E'_{missing_inp}(T,I,O))\backslash T$$

folgt die Behauptung. □

Aus der Präfixabgeschlossenheit einer Spurmenge folgt, daß die Erweiterung um fehlende Eingaben keine zusätzlichen Spuren liefert.

3.4.4 Kombination der Erweiterungen

Es ergibt sich für die Semantik von TSDs ohne Einschränkung des Umgebungsverhaltens für einen beliebigen TSD-DL-Ausdruck $Texpr$:

Definition 3.12

Die *Semantik eines* TSD-DL-*Ausdrucks* $Texpr$ *bei freiem Umgebungsverhaltens* wird für die Eingabemenge I und die Ausgabemenge O festgelegt als:

$$[\![Texpr]\!]^{(I,O)}_{TSD} := E([\![Texpr]\!]_{TSD}, I, O)$$

Wobei:

$$E(T,I,O) := T \cup E_{false_inp}(T,I,O) \cup E_{missing_inp}(T,I,O)$$

□

Bei der Definition von E beschreibt T den Anteil, bei dem sowohl die Komponente als auch die Umgebung ihren Verpflichtungen nachgekommen ist, also sowohl die Assumption als auch das Commitment erfüllt ist. Zu T werden die Spuren hinzugefügt, bei denen die Umgebung als erstes die Spielregeln nicht eingehalten hat. Hierbei beschreibt E_{false_inp} eine Verletzung des Sicherheitsanteils der Assumption, und $E_{missing_inp}$ eine Verletzung des Lebendigkeitsanteils.

Satz 3.10

Nicht alle TSDs beschreiben eine präfixabgeschlossene Menge. □

Beweis:

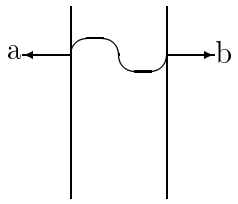

Abbildung 3.25: TSD, das keine präfixabgeschlossene Spurmenge beschreibt

Die Semantik ohne Einschränkung des Umgebungsverhaltens des TSDs aus Abbildung 3.25 ist bei einer vollständigen Interpretation:

$$[\![(a \sim b)^{\underline{\omega}}]\!]_{TSD}^{(\emptyset,\{a,b\})}$$

Es gilt:

$$<a,b> \in [\![(a \sim b)^{\underline{\omega}}]\!]_{TSD}^{(\emptyset,\{a,b\})}$$

Aber für das Präfix $<a>$ gilt:

$$<a> \notin [\![(a \sim b)^{\underline{\omega}}]\!]_{TSD}^{(\emptyset,\{a,b\})}$$

□

TSDs stellen somit keine reine Sicherheitseigenschaft dar. Das TSD aus Abbildung 3.25 wird in den Normen häufig zur Beschreibung eines vom Diensterbringer verursachten Verbindungsabbruchs verwendet.

3.4.5 Beispiele für Erweiterungen

In diesem Kapitel werden einige Beispiele für die Erweiterung von Spurmengen, bei denen die Umgebung die Spielregeln verletzt, angegeben. Hierzu werden TSDs aus den Normen verschiedener Dienstspezifikationen verwendet. Mit der Angabe dieser Beispiele werden sowohl die Erweiterungen als auch die Semantik einiger in den Normen verwendeter TSDs verdeutlicht.

Beispiel 3.8

Das TSD aus Abbildung 3.22 auf Seite 57 besitzt bei vollständiger Interpretation die Erweiterung um fehlende Eingaben für $I = \{a, i_1, i_2, \ldots\}$ und $O = \{b, o_1, o_2, \ldots\}$:

$$E_{missing_inp}([\![(a \sim b)^{\underline{\omega}}]\!]_{TSD}, I, O) = \{t \in \{a,b\}^{\omega} | \#a©t < \#b©t\}$$

Es gilt für das rechte TSD aus Abbildung 3.23 auf Seite 58:

$$E_{missing_inp}([\![(b \rightarrow a)^{\underline{\omega}}]\!]_{TSD}, I, O) = E_{missing_inp}([\![(a \sim b)^{\underline{\omega}}]\!]_{TSD}, I, O)$$

Somit sind beide TSDs bezüglich der Erweiterungen um fehlende Eingaben äquivalent, da die Spurmengen, bei denen der Lebendigkeitsanteil verletzt ist, identisch sind. Es gilt jedoch:

$$[\![(a \sim b)^{\underline{\omega}}]\!]_{TSD} \neq [\![(b \rightarrow a)^{\underline{\omega}}]\!]_{TSD}$$

Dies ist sinnvoll, da beide TSDs bei Einschränkung des Umgebungsverhaltens unterschiedliche Spurmengen beschreiben, obwohl die Verletzung des Lebendigkeitsanteils der Assumption bei beiden TSDs gleich ist. Genau dies wird von der in dieser Arbeit festgelegten Semantik modelliert. □

Beispiel 3.9

Das in Abbildung 3.19 auf Seite 50 dargestellte TSD besitzt bei szenarischer Interpretation, also ohne Wiederholungsoperator, die Semantik unter Beeinflussung des Umgebungsverhaltens:

$$T_2 := [\![(a \rightarrow c) \sim (b \rightarrow d)]\!]_{TSD}$$

Es gilt:

$$E_{missing_inp}(T_2, \{a, b\}, \{c, d\}) = \{<a, c>, <b, d>\}$$

Somit:

$$<a> \notin E_{missing_inp}(T_2, \{a, b\}, \{c, d\})$$

Dies entspricht der Intuition, da der Diensterbringer in diesem Fall ja noch eine Aktion durch die Ausgabe von c vornehmen kann. □

Beispiel 3.10

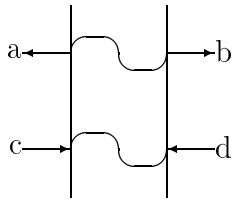

Abbildung 3.26: TSD aus der Transportschicht mit Einschränkung des Umgebungsverhaltens

Das TSD aus Abbildung 3.26 ist aus der Transportschicht [ISO84c, CCI88h] entnommen. Es besitzt ohne Wiederholungsoperator, also bei szenarischer Interpretation, die Semantik:

$$T_3 := [\![(a \rightarrow c) \sim (b \rightarrow d)]\!]_{TSD} = T_2$$

Es gilt:

$$E_{missing_inp}(T_3, \{c,d\}, \{a,b\}) := \{ \quad <a,b>, <a,c,b>, <a,b,c>, <b,a,c>, \\ <b,d,a>, <b,a,d>, <a,b,d>\}$$

□

Beispiel 3.11

Für die von dem in Abbildung 3.17 auf Seite 48 dargestellten TSD akzeptierte Spurmenge T_4 mit Wiederholung, also bei einer vollständigen Interpretation, gilt:

$$T_4 = [\![(a \sim b)^{\underline{\omega}}]\!]_{TSD} = \{t | t \in \{a,b\}^\omega \wedge \#a©t = \#b©t\}$$

$$E_{missing_inp}(T_4, \{a,b\}, \emptyset) = \{t \in \{a,b\}^\omega | \#a© \neq \#b©t\}$$

□

3.4.6 Aufdeckung und Behebung der Nichtkompositionalität der Semantik

Eine wesentliche Eigenschaft von Beschreibungstechniken ist die Kompositionalität der verwendeten Sprachmittel [AL90, Bro92, BS94, dR85, Jon90, Zwi89]. Nach [Gun92] ist die Semantik einer Sprache kompositional, wenn die Semantik eines Ausdruck anhand der Semantik der Teilausdrücke und des syntaktischen Aufbaus des Ausdrucks bestimmt werden kann. Durch die Kompositionalität wird eine modulare Entwicklung ermöglicht. Es wäre wünschenswert, daß jedes TSD durch eine eigene Funktion implementiert wird. Dies ist mit den zuvor genannten Überlegungen möglich. Danach könnten diese Funktionen durch Kombinationsfunktionen verbunden werden. Es zeigt sich jedoch im nachstehenden Satz, daß eine derartige Kombinationsfunktion auf einfachen Spurmengen nicht möglich ist, da TSDs mit den zuvor eingeführten Erweiterungen nicht mehr kompositional sind. Bei der Semantik unter Einschränkung des Umgebungsverhaltens war dies durch das faire Mischen zweier Spurmengen mit $\bowtie$ problemlos möglich, wie dies im Korollar 3.3 festgestellt wird. Die Nichtkompositionalität der spurbasierten Semantik ohne Einschränkung des Umgebungsverhaltens ist ein Folge davon, daß aus der Spurmenge, die die Semantik eines TSDs beschreibt, diejenigen Spuren, bei denen der Lebendigkeitsanteil der Assumption verletzt wird, nicht mehr aus der gesamten Spurmenge extrahierbar sind.

Satz 3.11

Es existiert **keine** Funktion ρ, so daß für alle T_A, T_B, die die Spurmengen von beliebigen TSDs unter Einschränkung des Umgebungsverhaltens beschreiben, gilt:

$$E(T_A \bowtie T_B, I, O) = E(T_A, I, O) \; \rho \; E(T_B, I, O)$$

Hierbei sei E wie in Definition 3.12 auf Seite 59 festgelegt. □

Beweis:

Die Aussage wird mit einem Widerspruchsbeweis gezeigt:

Annahme: Es gibt eine Funktion ρ, so daß $E(T_A \bowtie T_B, I, O) = E(T_A, I, O)\ \rho\ E(T_B, I, O)$ gilt.

Die Beweisidee läßt sich wie folgt skizzieren: Es werden zwei durch TSD-DL-Ausdrücke beschriebene Spurmengen T_1 und T_2 angegeben, bei denen die Erweiterungen zu freiem Umgebungsverhalten gleich sind: $E(T_1, I, O) = E(T_2, I, O)$ Dann wird eine weitere durch einen TSD-DL-Ausdruck beschriebene Spurmenge T' angegeben, die bei Zusammensetzung jeweils mit T_1 und T_2 durch den Operator $\bowtie$ zu unterschiedlichen Spurmengen führt: $E(T_1 \bowtie T', I, O) \neq E(T_2 \bowtie T', I, O)$. Da ρ eine Funktion ist, gilt: $E(T_1, I, O)\ \rho\ E(T', I, O) = E(T_2, I, O)\ \rho\ E(T', I, O)$, was zum Widerspruch führt.

Beweis:

Abbildung 3.27: TSD_1

Der TSD-DL-Ausdruck, der die vollständige Interpretation der TSDs aus Abbildung 3.27 beschreibt, ist:

$$TSD_1 := ((b \rightarrow a)|b)^{\underline{\omega}}$$

mit der Semantik:

$$T_1 := [\![TSD_1]\!]_{TSD} = \{t \in \{a,b\}^\omega | \forall t' \sqsubseteq t.\ \#a©t' \leq \#b©t'\}$$

Hierbei ist zur Vereinfachung a_A^{In} mit a, sowie b_B^{Out} mit b ersetzt worden. Dies ist ohne weiteres möglich, da diese Aktionen nur eindeutig verwendet werden, und somit die Zusatzinformation über den Ursprung und die Orientierung der Dienstelemente redundant ist.

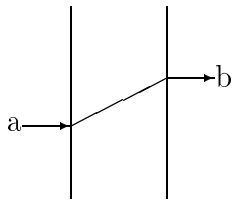

Abbildung 3.28: TSD_2

Der TSD-DL-Ausdruck, der die vollständige Interpretation des obigen TSDs beschreibt, ist:

$$TSD_2 := (b \rightarrow a)^{\underline{\omega}}$$

mit der Semantik:

$$T_2 := [\![TSD_2]\!]_{TSD} = \{t \in \{a,b\}^\omega | \#a©t = \#b©t \wedge \forall t' \sqsubseteq t.\ \#a©t' \leq \#b©t'\}$$

Somit gilt:

$$T_2 \subsetneq T_1$$

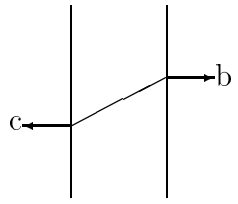

Abbildung 3.29: TSD'

Der TSD-DL-Ausdruck, der die vollständige Interpretation des in Abbildung 3.29 dargestellten TSDs beschreibt, ist:

$$TSD' := (b \rightarrow c)^{\underline{\omega}}$$

mit der Semantik:

$$T' := [\![TSD']\!]_{TSD} = \{t \in \{b,c\}^\omega | \#c©t = \#b©t \wedge \forall t' \sqsubseteq t.\ \#c©t' \leq \#b©t'\}$$

Zur Vereinfachung ist c_A^{Out} mit c ersetzt worden.

Seien im folgenden I und O als disjunkte Mengen definiert:

$$I = \{a, i_1, i_2, \ldots\}$$

$$O = \{b, c, o_1, o_2, \ldots\}$$

Nach Annahme gibt es ein ρ, so daß $E(T_1 \bowtie T', I, O) = E(T_1, I, O)\ \rho\ E(T', I, O)$
sowie $E(T_2 \bowtie T', I, O) = E(T_2, I, O)\ \rho\ E(T', I, O)$
Nach Lemma 3.3 auf Seite 65 gilt $E(T_1, I, O) = E(T_2, I, O)$
Da ρ eine Funktion ist, gilt somit: $E(T_1 \bowtie T', I, O) = E(T_1, I, O)\ \rho\ E(T', I, O) =$
$= E(T_2, I, O)\ \rho\ E(T', I, O) = E(T_2 \bowtie T', I, O)$
Also: $E(T_1 \bowtie T', I, O) = E(T_2 \bowtie T', I, O)$
Dies ist jedoch ein Widerspruch zu Lemma 3.2 von Seite 65:
$E(T_1 \bowtie T', I, O) \neq E(T_2 \bowtie T', I, O)$

□

Lemma 3.2

Für die zuvor durch die Abbildungen 3.27, 3.28 und 3.29 bestimmten Spurmengen T_1, T_2, T' gilt:

$$E(T_1 \bowtie T', I, O) \neq E(T_2 \bowtie T', I, O)$$

□

Beweis:

Es gilt:
$<b> \in E(T_1 \bowtie T', I, O)$, da $<b> \in T_1 \bowtie T'$
Es gilt $<b> \notin E(T_2 \bowtie T', I, O)$, da:
$<b> \notin T_2 \bowtie T'$ (Offensichtlicher Widerspruchsbeweis)
$<b> \notin E_{false_inp}(T_2 \bowtie T', I, O)$, da $b \in O$
$<b> \notin E_{missing_inp}(T_2 \bowtie T', I, O)$,
da $c \in O \wedge <b, c> \in PRE(T_2 \bowtie T') \wedge <c> \notin PRE(T_2 \bowtie T')$, also c eine Ausgabeaktion und keine spontane Aktion ist.
Woraus die Behauptung folgt.

□

Lemma 3.3

Für die zuvor durch die Abbildungen 3.27 und 3.28 bestimmten Spurmengen T_1, T_2 gilt:

$$E(T_1, I, O) = E(T_2, I, O)$$

□

Beweis:

Zu zeigen ist:
$E(T_1, I, O) = E(T_2, I, O)$
Es gilt:
$E_{false_inp}(T_1, I, O) = E_{false_inp}(T_2, I, O)$, wegen $FINPRE(T_1) = FINPRE(T_2)$ und Lemma 3.8 auf Seite 53
$E_{missing_inp}(T_1, I, O) = \emptyset$ (Offensichtlicher Widerspruchsbeweis)
Äquivalent zum Beweis von Lemma 3.1 auf Seite 35 und unter Einbeziehung der Definition von $E_{missing_inp}$ erhält man:
$E_{missing_inp}(T_2, I, O) = \{t \in \{a,b\}^\omega | (\#a©t \neq \#b©t \wedge \forall t' \sqsubset t.\ \#a©t' \leq \#b©t'\}$
Somit: $E_{missing_inp}(T_2, I, O) \cup T_2 = T_1$
Da $E_{missing_inp}(T_1, I, O) = \emptyset$ gilt, folgt:
$E_{missing_inp}(T_2, I, O) \cup T_2 = E_{missing_inp}(T_1, I, O) \cup T_1$
Da $E_{false_inp}(T_1, I, O) = E_{false_inp}(T_2, I, O)$ gilt, folgt: $E(T_1, I, O) = E(T_2, I, O)$

□

Da nach Kapitel 3.3.5 der Operator $\bowtie$ das Kompositionsprinzip von TSDs mit Einschränkung des Umgebungsverhaltens beschreibt, folgt aus dem Satz 3.11, daß TSDs mit den Erweiterungen um fehlerhafte und fehlende Eingaben nicht mehr kompositional sind.

Korollar 3.12

Es gibt **keine** Funktion ρ, so daß für alle TSD-DL-Ausdrücke TSD_1, TSD_2, gilt:

$$[\![(TSD_1 | TSD_2)^{\underline{\omega}}]\!]^{(I,O)}_{TSD} = [\![(TSD_1)^{\underline{\omega}}]\!]^{(I,O)}_{TSD} \ \rho\ [\![(TSD_2)^{\underline{\omega}}]\!]^{(I,O)}_{TSD}$$

□

Die Ursache der Nichtkompositionalität der Semantik ist, daß der Anteil, bei dem der Lebendigkeitsanteil der Assumption verletzt ist, aus der Vereinigung der Spurmengen nicht mehr extrahiert werden kann. Zur Beschreibung einer kompositionalen Semantik ohne Einschränkung des Umgebungsverhaltens bietet sich die folgende adhoc Lösung an, bei der die Semantik eines TSD-DL-Ausdrucks als ein Paar von Spurmengen dargestellt wird.

Definition 3.13

Die *kompositionale Semantik eines* TSD-DL*-Ausdrucks* $Texpr$ *ohne Einschränkung des Umgebungsverhaltens* wird für die Eingabemenge I und die Ausgabemenge O festgelegt als:

$$[\![Texpr]\!]^{(I,O)}_{TSDkomp} := (\quad E_{false_inp}([\![Texpr]\!]_{TSD}, I, O) \cup E_{missing_inp}([\![Texpr]\!]_{TSD}, I, O) \\ , [\![Texpr]\!]_{TSD})$$

□

Hierbei beschreibt die erste Komponente die Menge derjenigen Spuren, bei denen die Assumption als erstes verletzt wird, und die zweite Komponente die Menge der Spuren, bei denen sowohl die Assumption als auch das Commitment erfüllt ist.

Definition 3.14

Die *Kompositionsfunktion ρ_{TSD} der kompositionalen Semantik ohne Einschränkung des Umgebungsverhaltens* wird für die Spurmengen NA_1, NA_2, AC_1 und AC_2, sowie die Eingabemenge I und die Ausgabemenge O festgelegt als:

$$\begin{aligned}(NA_1, AC_1)\rho_{TSD}^{(I,O)}(NA_2, AC_2) \;:=\; (\;& E_{false_inp}(AC_1 \bowtie AC_2, I, O) \\ & \cup\, E_{missing_inp}(AC_1 \bowtie AC_2, I, O) \\ & , AC_1 \bowtie AC_2)\end{aligned}$$

□

Daß die durch die Definition 3.13 festgelegte Semantik kompositional ist, läßt sich durch folgende Überlegung leicht nachvollziehen. Die Definition der Kompositionsfunktion ρ_{TSD} stützt sich nur auf eine Verwendung der Funktion $\bowtie$ bei der Semantikkomponente der Spurmengen unter Beeinflussung des Umgebungsverhaltens. Nach Korollar 3.3 beschreibt die Funktion $\bowtie$ genau das Kompositionsprinzip der Semantik unter Beeinflussung des Umgebungsverhaltens.

3.5 Anwendung der Formalisierungsmethodik auf die Dienstnormen

In diesem Kapitel wird die Anwendung der zuvor eingeführten formalen Fundierung auf verschiedene Dienstbeschreibungen vorgenommen. Daran anknüpfend werden einige Erweiterungen der graphischen Darstellung von TSDs vorgeschlagen.

3.5.1 Untersuchung der TSD-DL-Ausdrücke bei den einzelnen Schichten

Die TSDs der einzelnen Schichten werden im weiteren formalisiert, wobei nur einzelne TSDs herausgegriffen werden, deren Interpretation interessant erscheint. Auf einen vollständigen TSD-DL-Ausdruck, der den TSD-Anteil einer Schicht beschreibt, wird verzichtet, da eine derartige Auflistung das Verständnis für TSDs nicht erhöht und mit den zuvor angeführten Methoden einfach durchgeführt werden kann. Einige Dienstelemente sind durch einfachere Identifikatoren ersetzt worden. Es werden nur diejenigen Schichten erwähnt, die explizit TSDs verwenden. Außerdem wird festgestellt, welcher der beiden Stile zur Darstellung von TSDs, die in dieser Arbeit in Kapitel 3.1 als dienstbenutzerorientierter oder diensterbringerorientierter Stil bezeichnet werden, verwendet wird.

Bitübertragungsschicht

In der Definition der Bitübertragungsschicht [ISO88c, CCI88e] wird der dienstbenutzerorientierte Stil verwendet. Es treten hauptsächlich ein- oder zweielementige TSDs auf. Da die Umformung der ein- oder zweielementigen TSDs in TSD-DL-Ausdrücke keine weiterführenden Erkenntnisse bringt, werden nur die mehrelementigen TSDs als Beispiele angeführt.

Wie im Kapitel 3.1 beschrieben, sollte das TSD aus Abbildung 3.7 auf Seite 25 durch das TSD aus Abbildung 3.8 auf Seite 25 ersetzt werden. Dies kann nun durch die folgende Äquivalenz formal belegt werden:

$$\begin{array}{rcl} ((a \sim b \sim c \sim d) & \circledast & (a \rightarrow c) \circledast (a \rightarrow d) \circledast (c \rightarrow d) \circledast (b \rightarrow c) \circledast (b \rightarrow d)) \\ & \simeq & \\ ((a \sim b) & \rightarrow & (c \rightarrow d)) \end{array}$$

Bei einem weiteren TSD aus der Bitübertragungsschicht fehlt das Tildesymbol, das nur zur Erhöhung der Verständlichkeit verwendet wird, zwischen den Dienstelementen c und d. Dies ist jedoch nicht weiter von Bedeutung.

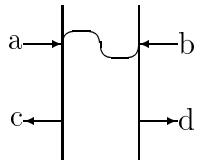

Abbildung 3.30: TSD aus der Bitübertragungsschicht

Der der Abbildung 3.30 entsprechende, vereinfachte TSD-DL-Ausdruck ist bei Betrachtung der szenarischen Interpretation:

$$(a \rightarrow c) \sim (b \rightarrow d)$$

Bei allen in der Bitübertragungsschicht angegebenen TSDs befindet sich bei den Dienstprimitiven ein Kommentar, der sich auf Zustandsübergänge bezieht, die in einem in der Norm später eingeführten Automaten verwendet werden. Da diese Angaben nur in dieser Schicht erfolgen, sind sie bei der Formalisierung nicht berücksichtigt worden.

Sicherungsschicht

In der Sicherungsschicht [ISO88b, CCI88f] wird der diensterbringerorientierte Stil zur Darstellung von TSDs verwendet. Die Umformung in TSD-DL-Ausdrücke ist mit dem in Kapitel 3.3.6 beschriebenen Verfahren einfach möglich. Da ein Großteil der beschriebenen TSDs im Kapitel 3.3 als Beispiele verwendet wird, wird auf eine explizite Darstellung verzichtet. Die formalen Betrachtungen der TSDs führen jedoch zur Aufdeckung von Redundanzen bei den in dieser Schicht verwendeten TSDs.

Abbildung 3.31: Redundanz bei TSDs der Sicherungsschicht

Beide der durch Abbildung 3.31 dargestellten TSDs sind in der Dienstspezifikation der Sicherungsschicht mit äquivalenten Bezeichnungen der Dienstelemente vorhanden. Das linke TSD beschreibt jedoch eine Teilmenge des rechten TSDs, da:

$$[\![(a \rightarrow b)^{\underline{\omega}}]\!]_{TSD} \subset [\![(a \sim b)^{\underline{\omega}}]\!]_{TSD}$$

Es stellt sich daher die Frage, weshalb das linke TSD in der Norm vorhanden ist. In [ISO88b, CCI88f] fehlt ein derartiger Hinweis.

Abbildung 3.32: Redundanz bei TSDs der Sicherungsschicht

Bei den in Abbildung 3.32 dargestellten TSDs tritt eine äquivalente Redundanz auf. Es gilt:

$$[\![(e \rightarrow (h \sim (f \rightarrow g)))^{\underline{\omega}}]\!]_{TSD} \subset [\![((e \rightarrow h) \sim (f \rightarrow g))^{\underline{\omega}}]\!]_{TSD}$$

Vermittlungsschicht

In der Vermittlungsschicht [ISO87c, CCI88g] wird der dienstbenutzerorientierte Stil verwendet. Die in den Abbildungen 3.31 und 3.32 gezeigten Redundanzen sind ebenfalls vorhanden. In dieser Norm wird bei der Verwendung von TSDs erstmalig mit Abbildung 3.33 auf Parameter bei den Dienstelementen eingegangen. Da in dieser Arbeit aus den in Kapitel 3.2 angegebenen Gründen der diensterbringerorientierte Stil bevorzugt wird, erfolgt die Darstellung der TSDs aus Abbildung 3.33 im Gegensatz zu den Normen in diesem Stil.

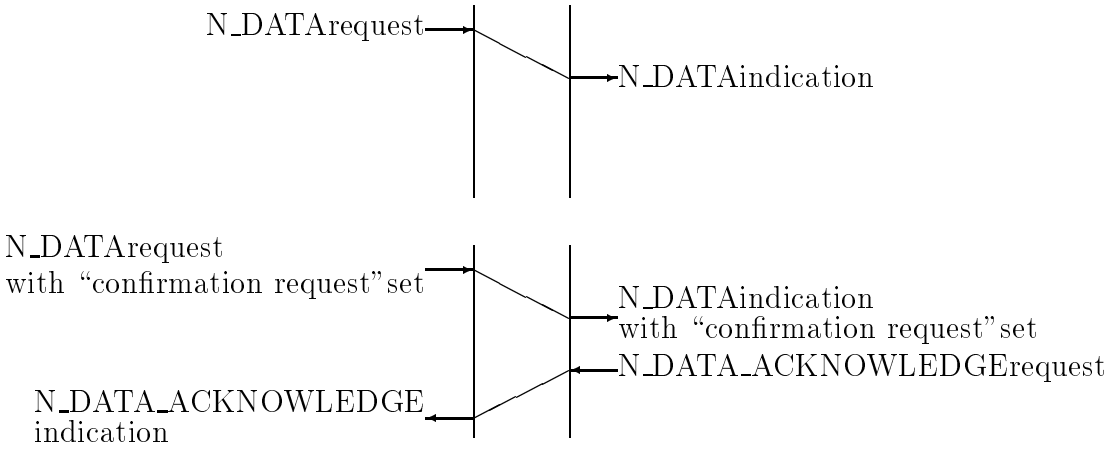

Abbildung 3.33: TSDs für die Datenübertragung bei der Vermittlungsschicht

Das obige TSD aus Abbildung 3.33 beschreibt eine herkömmliche Datenübertragung, das untere TSD eine quittierte Datenübertragung. Die Unterscheidung der Dienstelemente wird mittels eines zusätzlichen Parameters bei der Datenübertragung, der in [ISO87c, CCI88g] nicht explizit angegeben wird, vorgenommen. Eine derartige Unterscheidung kann problemlos in die Formalisierung der TSDs einbezogen werden.

Transportschicht

In der Dienstspezifikation der Transportschicht [ISO84c, CCI88h] wird der dienstbenutzerorientierte Stil verwendet. Die in Abbildung 3.31 dargestellte Redundanz tritt ebenfalls auf. Die Umsetzung der TSDs in TSD-DL-Ausdrücke kann problemlos vorgenommen werden.

Kommunikationssteuerungsschicht

Bei der Dienstnorm der Kommunikationssteuerungsschicht [ISO84b, CCI88i] wird der dienstbenutzerorientierte Stil verwendet. Es ist keine Redundanz bei den verwendeten TSDs vorhanden. Die Umwandlung der TSDs in TSD-DL-Ausdrücke gestaltet sich problemlos.

INRES Dienst

Der INRES Dienst wird in [Hog89, Hog91] verwendet, um einen vereinfachten Diensterbringer zu beschreiben, der die wesentlichen Aspekte eines Dienstes berücksichtigt. Der INRES Dienst ist *kein* Bestandteil des OSI Schichtenmodells, wird aber oft bei der Verwendung von formalen Beschreibungstechniken als Beispiel angeführt, da er in [Hog89, Hog91] mit SDL, LOTOS und Estelle formalisiert ist. Wegen seiner häufigen Präsenz bei formalen Betrachtungen wird er an dieser Stelle angeführt. Der Verbindungsaufbau beim INRES Dienst setzt sich nach [Hog89] aus TSDs, die in Abbildung 3.34 dargestellt werden, zusammen.

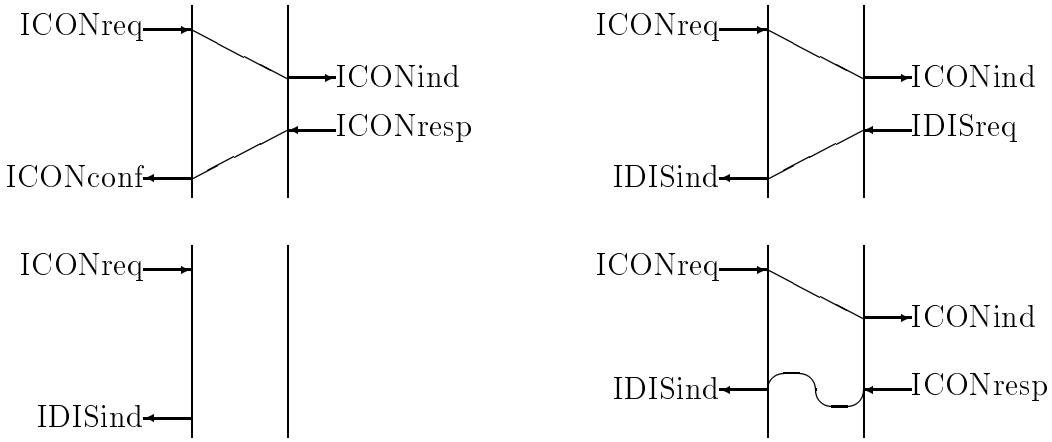

Abbildung 3.34: Verbindungsaufbau beim INRES Dienst

Bei den in Abbildung 3.34 dargestellten TSDs wird ein Reagieren des rechten Dienstbenutzers vorausgesetzt. Er hat nach der Übermittlung eines *ICONind* die Wahl, mit einem *ICONresp* oder einem *IDISreq* zu reagieren. In der formalen Beschreibung in SDL, LOTOS und Estelle in [Hog89, Hog91] wird jedoch auch die Spezifikation der Passivität des rechten Dienstbenutzers einbezogen. Als Plausibilitätserklärung dafür sei hierbei auf eine Implementierung verwiesen, bei der ein sogenannter Timeout eintreten kann. Auf der Abstraktionsstufe zur Beschreibung eines Diensterbringers gestaltet sich die Einführung eines Timeouts problematisch, da kein expliziter Zeitbegriff zur Verfügung steht, und somit ein solches Detail vor den Verwendern einer Dienstspezifikation verborgen bleibt. Da dieser Fall von den TSDs aus Abbildung 3.34 nicht erfaßt wird, wird in einer späteren Version der Beschreibung des INRES Dienstes [Hog91] vorgeschlagen, ein zusätzliches TSD zu verwenden, mit dem die Abstraktion eines Timeouts durch die Angabe einer Sequenz von Dienstelementen beim Verbindungsaufbau beschrieben wird.

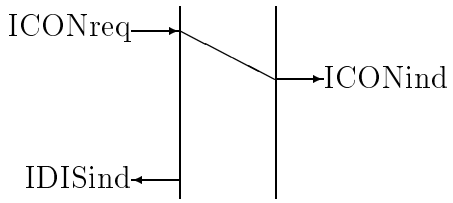

Abbildung 3.35: TSD zur Ergänzung des Verbindungsaufbaus

Mit den im Kapitel 3.4.3 vorgeschlagenen Erweiterungen ist jedoch auch eine andere Deutung möglich, bei der der Ablauf $<ICONreq, ICONind>$ gemäß der in Abbildung 3.34 dargestellten TSDs korrekt ist. Bei dieser Deutung ist die in [Hog91] getätigte Implementierungsentscheidung, daß ein Timeout erfolgt, nicht getroffen worden. Dies müßte bei dem in dieser Arbeit vorgestellten Verfahren ebenfalls durch das zusätzliche TSD aus Abbildung 3.35 vorgenommen werden. Eine weitere Interpretationsschwierigkeit ist

in [Hog91] vorhanden. Angenommen, ein derartiger Timeout würde erfolgen, dann führte dies zu der Spur $<ICONreq, ICONind, IDISind>$. Wenn der zweite Dienstbenutzer nach dem $IDISind$ mit einem $ICONresp$ antwortet, ist dies durch das rechte untere TSD aus Abbildung 3.34 abgedeckt. Antwortet er hingegen mit einem $IDISreq$, so wird dies von keinem der TSDs aus den Abbildungen 3.34 und 3.35 berücksichtigt. Daher wird in dieser Arbeit die in Abbildung 3.36 dargestellte Modifikation vorgeschlagen.

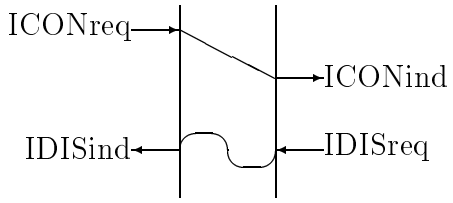

Abbildung 3.36: Modifikation eines TSDs des INRES Dienstes

Mit dem in Abbildung 3.36 vorgestellten TSD läßt sich der zuvor genannte Ablauf modellieren, da nun ein $IDISreq$ nach einem $IDISind$ möglich ist. Hierbei ist das rechte obere TSD aus Abbildung 3.34 durch das TSD aus Abbildung 3.36 zu ersetzten.

3.5.2 Erweiterung von TSDs zur Einbeziehung der in realen Schichten zusätzlich angegebenen verbalen Anmerkungen

Mit TSD-DL ist es möglich, den TSD Anteil einer Norm weitgehend zu beschreiben. Die Formalisierung hat jedoch gezeigt, daß für eine exakte Beschreibung eine Erweiterung der graphischen Darstellung von TSDs notwendig ist. Im folgenden wird eine Variante für TSD-DL angegeben, *TSD-DLext* genannt, mit der solche Erweiterungen beschrieben werden können. Anschließend wird eine formale Semantik für TSD-DLext angegeben.

Formalisierung einer Verbindungsunterbrechung

Bei vielen Dienstspezifikationen beschreiben die darin verwendeten TSDs einen etwaigen Datenverlust nicht vollständig. Einige Spezifikationen behelfen sich hier mit textuellen Zusätzen.

Als erstes Beispiel dient der INRES Dienst. In [Hog89, Hog91] wird die Datenübertragung beim INRES Dienst mit den TSDs aus Abbildung 3.37 informell spezifiziert.

Abbildung 3.37: Datenübertragung beim INRES Dienst

Betrachtet man die beiden in Abbildung 3.37 dargestellten TSDs, so folgt daraus, daß jedem Datenübertragungswunsch ($IDATreq$) entweder genau eine Datenbestätigung ($IDATind$) oder genau ein Anzeigen eines Verbindungsabbruchs ($IDISind$) zugeordnet ist. In der Praxis ist folgender Ablauf möglich: Ein Dienstbenutzer sendet fünf $IDATreq$; der Diensterbringer stellt dem anderen Dienstbenutzer das erste Datenpaket mit einem $IDATind$ zu. Danach wird die Verbindung unterbrochen. Bei obiger exakter Interpretation der TSDs sollte der Diensterbringer viermal ein $IDISind$ ausgeben. In der formalen Beschreibung des INRES Dienstes mit den Beschreibungstechniken LOTOS, SDL und Estelle in [Hog89, Hog91] genügt es jedoch, daß genau ein $IDISind$ ausgegeben wird. Aus diesem Grund ist in [BHS91] die Hinzunahme eines weiteren TSDs vorgeschlagen worden.

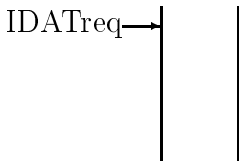

Abbildung 3.38: Erweiterung der Datenübertragung beim INRES Dienst

Mit den in den Abbildungen 3.37 und 3.38 gezeigten TSDs kann beim oben angegebenen Ablauf der Diensterbringer genau ein $IDISind$ ausgeben. Durch die Hinzunahme des TSDs aus Abbildung 3.38 ist jedoch eine unsichere Datenübertragung möglich geworden, da ein Reagieren auf ein $IDATreq$ nicht mehr unbedingt notwendig ist. Dieser Aspekt wird erst später bei der Beschreibung der Eigenschaften einer Verbindung berücksichtigt. Da auch komplexere TSDs unterbrochen werden können, würde eine derartige Vorgehensweise auch die Anzahl der TSDs enorm erhöhen. Es ist hierbei jeder mögliche Teilablauf mit einem eigenen TSD zu berücksichtigen. Eine konsequentere Vorgehensweise ist die Einführung einer neuen Art von TSDs in dieser Arbeit, die durch Abbildung 3.39 verdeutlicht wird. Die formale Semantik dieses Konstrukts wird im Anschluß an die graphische Einführung aller Erweiterungen von TSDs angegeben.

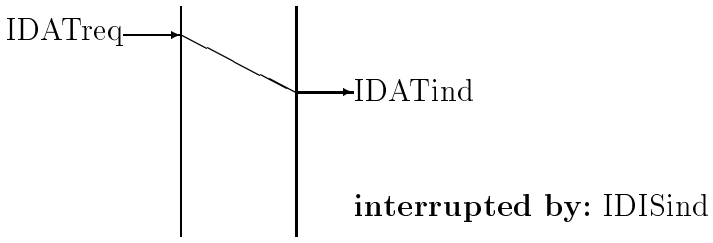

Abbildung 3.39: TSD mit globalem Kill-Operator

Hierbei wird definiert, daß bei einer vollständigen Interpretation das TSD mit einem $IDISind$ unterbrochen werden kann. Es können bei dem in Abbildung 3.39 dargestellten TSD auch mehrere offene $IDATreq$ unterbrochen werden.

Anstelle des Dienstelements $IDISind$ ist es natürlich möglich, auch komplexere TSDs einzusetzen. Mit dieser Methode ist zum Beispiel der textuelle Zusatz in der Transportschichtspezifikation [ISO84c, CCI88h], daß jedes TSD durch ein $DISind$ oder ein $DISreq$ unterbrochen werden kann, mit TSDs darstellbar.

Einbeziehung von Parametern bei Dienstelementen

Normalerweise werden TSDs zur Spezifikation von Sequenzen von Dienstelementen ohne Parameter verwendet. Bei der Datenübertragung sind jedoch gerade die Parameter von Bedeutung, da sie der essentielle Teil der Kommunikation sind. Eine naheliegende Einbeziehung der Parameter ergibt das TSD aus Abbildung 3.40.

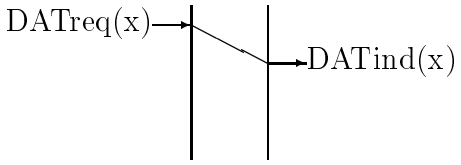

Abbildung 3.40: Datenübertragung mit Parametern

Durch das in Abbildung 3.40 dargestellte TSD werden die bei den meisten Dienstelementen vorhandenen Parameter einbezogen. Genauergenommen müßte hierbei eine Datentyperweiterung vorgenommen werden, da bisher nur Konstante als parameterlose Dienstelemente verwendet wurden, aber jetzt mehrstellige Sortenkonstruktoren erforderlich sind. Semantisch wird dies durch die Verwendung von freien Algebren modelliert [BFG+93, EGL89, EM85, EM90, Gal86, Wec92]. Das dargestellte TSD beschreibt ausschließlich einen korrekten Datenaustausch, da die Parameter bei beiden Dienstelementen identisch sind. In den Dienstspezifikationen des OSI Schichtenmodells ist jedoch das Konzept der Restfehlerrate eingeschlossen. Hierunter sind vom Diensterbringer nicht erkennbare Übertragungsfehler zu verstehen, die eine Folge des in darunterliegenden Schichten verborgenen Protokolls sind. Ein Beispiel hierfür ist ein verfälschter Datensatz, bei dem die Prüfsummen gleich sind. Zur formalen Beschreibung dieses Sachverhaltes werden in dieser Arbeit *Parameterrestriktionen* eingeführt. Die formale Semantik der Parameterrestriktionen wird im Anschluß eingeführt.

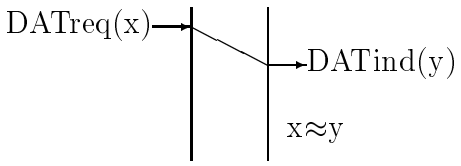

Abbildung 3.41: Datenübertragung mit Parameterrestriktion

Bei dem in Abbildung 3.41 dargestellten TSD ist $\approx$ eine Äquivalenzrelation, mit der die Äquivalenzklasseneinteilung bezüglich nicht erkennbarer Fehler vorgenommen wird.

Anstelle von $\approx$ ist jedes beliebige Prädikat erlaubt. Die hierbei zugrundeliegende Intention ist, daß für dieses TSD die Bedingung $x \approx y$ erfüllt sein muß. Wenn dies nicht der Fall ist, so ist dieses TSD nicht anwendbar.

TSDs mit Parameterrestriktion können auch verwendet werden, um die Aushandlung der *Quality of Service*-Parameter, die die Dienstgüte einer Verbindung beschreiben, beim Verbindungsaufbau zu spezifizieren. Hierzu ist nur die Äquivalenzrelation $\approx$ durch eine geeignete partielle Ordnung zu ersetzen.

Syntax und Semantik von TSD-DLext

Die zuvor genannten Erweiterungen von TSDs werden mit folgenden syntaktischen Ergänzungen von TSD-DL aus Kapitel 3.3 beschreibbar:

(7) <TSD> ::= <TSD> † <TSD>
(8) <TSD> ::= **Var** <varlist> **With** <pred> : <TSD>

Hierbei ist <varlist> eine Liste von Variablenbezeichnern, die durch ein Komma getrennt sind, und <pred> ein beliebiges Prädikat, das nur die in <varlist> definierten Identifikatoren als freie Variablen enthält. Mit $P_{TS_1 \dagger TS_K}(t)$ wird dargestellt, daß t bezüglich des TSD-DLext-Ausdrucks TS_1 korrekt ist, oder aber durch eine Spur unterbrochen wurde, die TS_K erfüllt. Somit wird eine mögliche Unterbrechung von TS_1 durch TS_K modelliert. Das Einbeziehen von Parametern wird zum Beispiel bei $P_{\textbf{Var } x,y \textbf{ With } Pred(x,y):TS}(t)$ dadurch erreicht, daß bei einer Spur, die dieses Prädikat erfüllt, jeweils bei den in diesem TSD verwendeten Parametern x und y das Prädikat $Pred(x,y)$ erfüllt ist.

Das in Abbildung 3.39 dargestellte TSD führt zum TSD-DLext-Ausdruck:

$$(IDATreq \rightarrow IDATind)^{\underline{\omega}} \dagger IDISind$$

Die in Abbildung 3.41 vorgenommene Parameterrestriktion wird durch den folgenden TSD-DLext-Ausdruck beschrieben:

$$\textbf{Var } x, y \textbf{ With } x \approx y : DATreq(x) \rightarrow DATind(y)$$

Die formale Semantik des Operators † ist:

$$\begin{aligned} P_{TS_1 \dagger TS_K}(t) = \quad & P_{TS_1}(t) \\ \vee\ & \exists s_1, s_k, t' : Trace\ Event.\quad t = s_1 \circ s_k \\ & \qquad\qquad \wedge\ P_{TS_K}(s_k) \wedge s_1 \neq \epsilon \wedge P_{TS_1}(s_1 \circ t') \end{aligned}$$

Es existieren zwei Möglichkeiten, daß eine Spur t den TSD-DLext-Ausdruck $TS_1 \dagger TS_K$ erfüllt. Entweder t ist eine Spur, die nur TS_1 erfüllt, ohne daß TS_K vorkommt. Oder aber t ist das Präfix einer Spur, die TS_1 erfüllt und an die eine Spur angefügt wird, die TS_K erfüllt. Die formale Semantik der Parameterrestriktion ist:

$$P_{\textbf{Var } x,y \textbf{ With } Pred(x,y):TS}(t) = \exists x, y.\ Pred(x,y) \wedge P_{TS}(t)$$

Zur Vereinfachung ist nur ein zweistelliges Prädikat verwendet worden. Die implizite Bindung von Parametern bei den Dienstelementen in TS wird mittels eines Existenzquantors gelöst. Somit gehören nur die Dienstelemente, die die Parameter x und y verwenden, zu einem TSD, wenn für diese Parameter $x \approx y$ gilt.

3.6 Bewertung der Formalisierungsmethodik

Mit der in Kapitel 3.3 angegebenen Methodik ist es möglich, die TSDs der Dienstspezifikation jeder Schicht zu formalisieren. Die informell spezifizierten Eigenschaften werden mit einer eindeutigen Semantik unterlegt. Dabei sind durchaus einige Interpretationsentscheidungen zu treffen, die in den Normen [ISO87b, CCI88d] nicht beschrieben werden. An dieser Tatsache erkennt man sehr deutlich, daß sogar bei intuitiv leicht verständlichen Konzepten eine informelle Spezifikation dringend einer semantischen Fundierung bedarf. Dies ist bei den in Abbildung 3.7 von Seite 25 und Abbildung 3.13 von Seite 43 erläuterten Beispielen der Fall. Eine informelle Spezifikation ist dennoch als Kommentar nach dem in [Vis88] beschriebenen Verfahren notwendig.

Im Gegensatz zu den aus der Literatur bekannten Verfahren wird die Spezifikation der TSDs als eigenständiges Spezifikationsmerkmal betrachtet. Dies ist ein methodisch sauberer Ansatz, da nur eine in sich abgeschlossene Einheit betrachtet wird. Mit der angegebenen Semantik ist es jetzt möglich, TSDs als eigenständige Beschreibungstechnik zu betrachten. Eine monolithische Beschreibung eines Diensterbringers, wie dies in der Literatur vorgenommen wird, ist als nächster Schritt im Hinblick auf eine spätere Implementierung möglich. Dieser ist durch die in dieser Arbeit angegebene Semantik verifizierbar.

Durch die Angabe eines Verfahrens zur Umwandlung beliebiger TSDs in TSD-DL-Ausdrücke in Kapitel 3.3.6, sowie einer formalen Semantik von TSD-DL sind TSDs formal fundiert worden. Eine geringfügige Schwachstelle bei der angegebenen formalen Semantik von TSDs ist der Übergang von der graphischen Darstellung von TSDs zu TSD-DL-Ausdrücken mit dem in Kapitel 3.3.6 angegebenen Verfahren, da dessen Ansatzpunkt beim Erfassen der Basispräzedenzrelation verbal, also informell beschrieben wurde. Dies wäre durch die Angabe einer graphischen Syntax von TSDs zu beheben. Dieser Vorschlag ist technisch sehr aufwendig und würde zur formalen Fundierung von TSDs keine wesentlichen Aspekte beitragen. Deswegen wird in dieser Arbeit die geringfügige Nichtformalität des Transformationsverfahrens aus Kapitel 3.3.6 in Kauf genommen. Durch dieses Verfahren werden in der Regel nicht die einfachsten TSD-DL-Ausdrücke ermittelt. Dies ist aber zur Bildung einer formalen Semantik nicht notwendig.

Mit der angegebenen Semantik ist auch eine Deutung von unendlichen Abläufen möglich. Derartige Überlegungen sind nicht einmal rudimentär in den definierenden Normen [ISO87b, CCI88d] vorhanden. Erst mit einer solchen Interpretation ist die Deutung von Lebendigkeitseigenschaften möglich. Dies ist notwendig, da TSDs sowohl Lebendigkeitseigenschaften als auch Sicherheitseigenschaften beschreiben. Es ist bei den beschriebenen Verfahren auch möglich, daß eine unendliche Spur korrekt bezüglich einer Menge von TSDs ist, obwohl alle endlichen Präfixe dieser Spur bezüglich dieser Menge von TSDs inkorrekt sind. Als Beispiel hierfür dient:

$$P_{(a \to b)^{\underline{\omega}}}(\mathsf{fix}\boldsymbol{\lambda} s.a\&a\&b\&s)$$

Diese Semantik unterscheidet sich dadurch von den meisten Semantiken von Prozeßalgebren, siehe dazu unter anderem [BW90, Hoa85], bei denen die unendlichen Elemente ausschließlich durch den Approximationsprozeß erreichbar sind und somit nur eine eingeschränkte Klasse von Lebendigkeitseigenschaften beschreibbar ist [Web92]. Diese Seman-

tik würde dazu führen, daß die obige Aussage nicht zuträfe, da alle endlichen Präfixe dieser Spur nicht in der Spurmenge enthalten sind. Durch die verwendete Semantik erhält man einen höheren Abstraktionsgrad, da eventuelle Implementierungsgedanken erst später eine Rolle spielen.

Ein weiterer Unterscheidungspunkt ist die Verwendung eines nichtdeterministischen Operators bei herkömmlichen Prozeßalgebren: Sei zum Beispiel der TSD-DL-Ausdruck für einen Teil des Verbindungsaufbau beim INRES Diensts, der in Abbildung 3.34 auf Seite 71 dargestellt ist, festgelegt als:

$$((a \rightarrow b) \rightarrow (c \rightarrow d))|((a \rightarrow b) \rightarrow (e \rightarrow f))$$

In der Prozeßalgebra BPA_ϵ, die in [BW90] definiert ist, wird dieser Sachverhalt folgendermaßen modelliert:

$$(a \cdot b \cdot c \cdot d) + (a \cdot b \cdot e \cdot f)$$

Jedoch gilt bei BPA_ϵ:

$$x \cdot (y + z) \neq (x \cdot y) + (x \cdot z)$$

da der Zeitpunkt der nichtdeterministischen Entscheidung bei beiden Seiten unterschiedlich ist [BW90, Hoa85]. Da aber eine Vernachlässigung des Zeitpunkts der nichtdeterministischen Entscheidung bei *Message Sequence Charts* (MSCs) [CCI92] von Bedeutung ist, wird in [BM94] die Einführung eines *delayed choice* Operators vorgeschlagen, bei dem die Negation der obigen Gleichung zutrifft. Eine derartige Erweiterung der Prozeßalgebra ist ebenfalls zur Darstellung von TSDs notwendig. In TSD-DL ist die Vernachlässigung des Zeitpunkts der nichtdeterministischen Entscheidung problemlos darzustellen.

Ein wesentlicher Punkt bei der Interpretation von TSDs ist eine Semantik ohne Einschränkung des Umgebungsverhaltens. Dies ist in [ISO87b, CCI88d] nur in sehr geringem Maße berücksichtigt worden. Mit den im Kapitel 3.4 angegebenen Erweiterungen wird eine Semantik festgelegt, die eine spätere Implementierungsphase unterstützt. So ist im Kapitel 3.4.5 die Semantik von verschiedenen TSDs angegeben worden, deren Interpretation nicht offensichtlich ist. Aufbauend auf dieser Semantik hat sich im Kapitel 3.4.6 gezeigt, daß die auf einer Spurmenge basierende Semantik von TSDs nicht kompositional ist. Die Nichtkompositionalität konnte jedoch durch die Verwendung eines Tupels von Spurmengen als Grundlage der Semantik behoben werden.

3.7 Bewertung von TSDs

TSDs sind ein weit verbreitetes Beschreibungsmittel. Sie werden im Zusammenhang mit dem OSI Schichtenmodell in einem Teil der CCITT Empfehlung [CCI88d], sowie im technischen Bericht der ISO [ISO87b] definiert und werden in den Normen zur Dienstspezifikation der OSI Schichten verwendet. Die jeweiligen Normennummern sind dazu in Tabelle 1.1 auf Seite 3 beziehungsweise in Tabelle 1.2 auf Seite 4 angegeben. TSDs finden aber auch außerhalb des OSI Schichtenmodells Verwendung. So werden sie oftmals bei einer beliebigen Rechnerkommunikation als bevorzugtes Beschreibungsmittel zur Darstellung von Kommunikationsabläufen benutzt.

Ein Grund für die weite Verbreitung von TSDs ist die einfache graphische Darstellungsweise und ein sehr einfaches intuitives Verständnis von TSDs. Es ist möglich, mit

TSDs auch einen komplexen Kommunikationsverlauf auf leicht verständliche Weise darzustellen.

Erstaunlicherweise gibt es trotz des hohen Verbreitungsgrads von TSDs keine eindeutige Semantikbeschreibung. Eigentlich kann man die informellen Beschreibungen in [ISO87b, CCI88d], die sich nur in einigen Formulierungsdetails unterscheiden, als rudimentär bezeichnen, da die Definition den sehr geringen Umfang einer Seite einnimmt, und im wesentlichen nur die zeitlichen Präzedenzen beschrieben werden. So wird zum Beispiel kein Hinweis zur Interpretation der Komposition von TSDs angegeben. Ein wesentlicher Nachteil ist, daß TSDs bei der Verwendung manchmal nur einen Beispielscharakter haben. Es wird hierbei ausgesagt, daß TSDs nur die häufigsten Beispiele einer Kommunikation beschreiben. Eine derartige Interpretation ist in der Erneuerung des technischen Berichts der ISO [ISO87b] durch [ISO94] vorgenommen worden. Diese Einschränkung läßt jeden beliebigen Ablauf zu. Somit haben TSDs eine sehr geringe Ausdruckskraft, es sei denn, daß sie ausschließlich zur Illustration einiger Beispiele gedacht sind. In der vorgestellten Arbeit wird dieses Manko durch die Angabe einer formalen Semantik umgangen, indem TSDs als Erzeugungsprinzip für eine Spurmenge betrachtet werden, das heißt, daß keine Spuren akzeptiert werden, die nicht explizit durch TSDs beschrieben werden. Ein hierbei aufgetretenes Problem ist die Komposition von TSDs, die in [ISO87b, CCI88d, ISO91a, ISO94] nicht einmal erwähnt wird. In Kapitel 3.3 ist die Komposition als ein faires Interleaving auf Spurmengen definiert worden.

Betrachtet man die von TSDs akzeptierten Sprachmengen, so kann man versuchen, deren endliche Anteile in die Chomsky Hierarchie einzuordnen. Zum einen lassen sich mit regulären Grammatiken Sprachmengen beschreiben, die mit TSDs nicht darstellbar sind. Ein Beispiel hierfür ist die Sprachmenge $\{<a,b>^n \mid n \in \mathbb{N}\}$. Eine derartige Spurmenge läßt sich mit TSDs wegen des Interleavingkonzepts nicht beschreiben, da dann auch Elemente wie zum Beispiel $<a,a,b,b>$ in der Sprachmenge enthalten sind[6]. Zum anderen sind kontextfreie Grammatiken nicht zur Darstellung von TSDs ausreichend. Dies ist zum Beispiel an $[\![(a \sim b \sim c)^{\underline{\omega}}]\!]_{TSD}$ erkennbar. In der akzeptierten Sprachmenge sind alle Spuren mit der gleichen Anzahl von a, b und c enthalten. Wegen des Pumping Lemmas [AU72] ist diese Sprachmenge nicht mit einer kontextfreien Grammatik beschreibbar. Es ist vermutlich ein Ansatz denkbar, der kontextsensitive Grammatiken zur Beschreibung von TSDs verwendet. Da dann aber die Beschreibung nicht in der gewünschten Abstraktheit durchgeführt wird, dies ist bei den unendlichen Spuren erkennbar, sind derartige Untersuchungen in dieser Arbeit nicht durchgeführt worden.

Bei TSDs werden in den Normen zwei als äquivalent nahegelegte Stile verwendet. Genauere Überlegungen haben jedoch im Kapitel 3.2 gezeigt, daß der diensterbringerorientierte Stil mächtiger ist. Obwohl in [ISO87b] dieser Stil sogar als der logisch korrekte bezeichnet wird, wird in der Praxis der dienstbenutzerorientierte Stil häufiger verwendet. In der Neufassung der definierenden Norm [ISO94] wird ausschließlich der diensterbringerorientierte Stil im Hauptteil verwendet. Der dienstbenutzerorientierte Stil wird im Anhang nur noch vorgestellt. Durch die Ergebnisse dieser Arbeit ist eine fundierte Argumentation für die Verwendung des diensterbringerorientierten Stils möglich geworden.

Ein grundlegendes Problem, das bei einem ersten Schritt in Richtung Implementierung

[6] Dies wird im Kapitel 6 formal mit Satz 6.3 belegt.

aufgetreten ist, ist die Einschränkung des Umgebungsverhaltens bei einer einheitlichen Behandlung von Eingabe- und Ausgabeaktionen in einer Spursemantik. Eine Behebung dieser Einschränkung ist in [ISO87b, CCI88d, ISO91a, ISO94] nur in einem Nebensatz erwähnt. Im Kapitel 3.4 wird eine Semantik definiert, die das Umgebungsverhalten nicht mehr einschränkt. Eine auf einer einzelnen Spurmenge basierende Semantik führt jedoch dazu, daß TSDs nicht mehr kompositional sind. Dies ist eine wesentliche Einschränkung bei der Verwendung von TSDs, da dadurch bei der Dienstimplementierung eine modulare Vorgehensweise im allgemeinen nicht mehr möglich ist, das heißt, es kann nicht jedes TSD einzeln implementiert und diese dann mit Kombinationsfunktionen zusammengefügt werden. Somit ist bei der Implementierung eine andere Vorgehensweise erforderlich. Es werden die TSDs einer Schicht, als Gesamtheit betrachtet, für eine Implementierung herangezogen. Eine Alternative zu dieser unbefriedigenden Vorgehensweise ist die Verwendung von Paaren von Spurmengen als Semantik von TSDs. Die Notwendigkeit eines solchen Vorgehens ist a priori aus [ISO87b, CCI88d, ISO91a, ISO94] nicht ersichtlich. Sie ist aber durch die in Kapitel 3.4 definierte formale Semantik erkennbar geworden.

Die Beschränkung des Umgebungsverhaltens hat auch den Normenerstellern Schwierigkeiten bereitet. Dies ist an einigen Indizien zu erkennen:

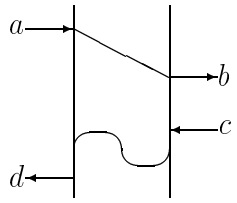

Abbildung 3.42: TSD aus der Datensicherungsschicht

In dem in Abbildung 3.42 dargestellten TSD, das aus der Datensicherungsschicht [ISO88b, CCI88f] entnommen ist, sind die Aktionen c und d nicht auf der gleichen Höhe angebracht worden. Es stellt sich hierbei die Frage, ob damit eine Kausalität zwischen c und d ausgedrückt werden soll.

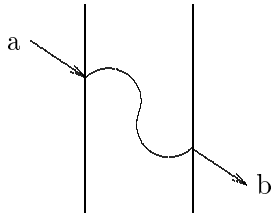

Abbildung 3.43: TSD aus der Vermittlungsschicht

In dem in Abbildung 3.43 dargestellten TSD, das aus der Vermittlungsschicht stammt, ist die Tilde geneigt dargestellt. Durch dieses Schrägstellen der Tilde könnte eine versteckte Kausalität nahegelegt werden. An den obigen Beispielen ist deutlich ersichtlich, daß

den Normenerstellern einige Details Probleme bereitet haben, die in dieser Arbeit gelöst werden, indem eine eindeutige Semantik angegeben wird. Ebenfalls ein Indiz dafür ist, daß bei der Neufassung der Definition von TSDs in [ISO94] TSDs mit dem Tildesymbol nur im Anhang angeführt sind. Dabei wird festgestellt, daß dieser Anhang kein wesentlicher Bestandteil der Norm ist. Durch diese vorgestellte Arbeit ist eine exakte Deutung derartiger TSDs möglich.

Durch die formale Fundierung von TSDs hat sich die Notwendigkeit einer geringfügigen Erweiterung der graphischen Darstellung von TSDs im Kapitel 3.5.2 ergeben. Mit diesen erweiterten TSDs können in den Normen vorkommende, zusätzliche verbale Einschränkungen einbezogen werden. Ein äquivalentes Problem ist die Beschreibung von Vollduplex-Verbindungen. Bei keiner OSI Dienstnorm ist bei strenger Auslegung der TSDs ein gleichzeitiger Datentransfer in beide Richtungen über eine bestehende Verbindung möglich, da gemäß den verwendeten TSDs nur ein Datentransfer vom verbindungsaufbauenden Dienstbenutzer zum verbindungsakzeptierenden Dienstbenutzer möglich ist. Zur Modellierung dieses Sachverhalts ist es erforderlich, ein zusätzliches TSD einzufügen, das auch einen Datentransfer in die entgegengesetzte Richtung beschreibt.

In den Normen werden bei den TSDs die eventuell vorhanden Parameter der Dienstelemente nicht berücksichtigt. Diese sind jedoch insbesondere bei der Datenübertragung von wesentlicher Bedeutung. Mit den im Kapitel 3.5.2 präsentierten Überlegungen ist ein Einbeziehen der Parameter problemlos möglich.

Ein weiterer Kritikpunkt ist die Beschreibung der Kommunikation eines Dienstbenutzers mit dem Diensterbringer als eine einzelne Aktion. Bei einer Kommunikation ist es nach [Vis93] sinnvoller, diese Aktion in eine Sendeaktion und eine Empfangsaktion aufzuspalten, da in der Realität auch zwei Aktionen stattfinden: Das Senden einer Nachricht und der Empfang einer Nachricht, dem ein Warten vorausgeht. Dabei wird eine Synchronisation per Handshake angenommen. Eine derartige Modellierung wird auch bei den *Message Sequence Charts* (MSC) [CCI92] verwendet.

Die Zeitbetrachtung bei den TSDs führt dazu, daß zwei Ereignisse, die an unterschiedlichen Dienstzugangspunkten erfolgen, in eine zeitliche Relation zu setzen sind. Dabei ist eine globale Sicht der Zeit erforderlich, da eine zeitliche Präzedenz zweier Ereignisse beurteilt wird. Eine derartige globale Sicht ist jedoch etwas eigenartig, da sich beide Dienstzugangspunkte unter Umständen auf verschiedenen Rechnern oder an auf dem Globus entgegengesetzten Orten befinden können. Dann ist ein Beobachter notwendig, der beide Dienstzugangspunkte zugleich betrachtet und die unterschiedlichen Zeitmaßstäbe ineinander überführen kann.

Abgesehen von den oben erwähnten Interpretations- und prinzipiellen Schwierigkeiten sind TSDs in weiten Bereichen zur Beschreibung von speziellen Aspekten der Rechnerkommunikation gut einsetzbar. Durch die in dieser Arbeit erfolgte formale Fundierung ist dies nun sogar mit einer eindeutigen Semantik möglich. Ob diese Semantik der Intuition eines Erstellers einer Dienstspezifikation entspricht, ist ein offener Diskussionspunkt. Durch die Angabe einer formalen Semantik ist jedoch die Grundlage zu einer fundierten Diskussion über die Interpretation von TSDs gelegt worden.

Kapitel 4

Formalisierung der lokalen Einschränkungen

Bei den unteren Schichten einschließlich der Transportschicht ist die zeitliche Abfolge der Dienstelemente an einem Verbindungsendpunkt[1] eingeschränkt. Dies ist ein lokal gebildetes Kriterium, da es Eigenschaften an nur einem Dienstzugangspunkt beschreibt, also nur den SAP zum Dienstbenutzer A, oder den zum Dienstbenutzer B alleinstehend betrachtet. In dieser Arbeit wird dafür der Begriff *lokale Einschränkungen*, der mit *LC (local constraints)* abgekürzt wird, verwendet.

Bei der Definition der lokalen Einschränkungen wird die Richtung der Dienstelemente nicht betrachtet, das heißt, es wird nicht bei der Anwendung der Beschreibungstechnik substantiell unterschieden, ob ein Dienstelement Eingabe oder Ausgabe des Diensterbringers ist, da die Unterscheidung implizit in einer anderen Norm über die Namensgebung vorgenommen wird. So sind nach [ISO87b, CCI88d] auf *request* sowie *reponse* endende Dienstelemente Eingabeaktionen, auf *indication* sowie $confirm$ endende Ausgabeaktionen. Somit berücksichtigt die Beschreibungstechnik LC keine explizite Unterscheidung von Eingabe- und Ausgabeaktionen, da beide gleich behandelt werden. Damit wird bei den in den Normen beschriebenen Verfahren nicht nur das Komponentenverhalten, sondern zusätzlich das Umgebungsverhalten eingeschränkt. Aus diesem Grund wird äquivalent zur Formalisierung von TSDs im Kapitel 3 eine Zweiteilung der Semantik vorgenommen. Zuerst wird bei der Beschreibung des Diensterbringers eine Einschränkung der Umgebung in Kauf genommen, da die Normen eine derartige Semantik nahelegen. Hierbei werden nur Sequenzen von erlaubten Aktionen ohne Unterscheidung der Ein- und Ausgabeaktionen betrachtet. Als zweiter Schritt werden die dadurch erhaltenen Spurmengen äquivalent zu dem in Kapitel 3.4 beschriebenen Verfahren dahingehend erweitert, daß das Umgebungsverhalten nicht mehr eingeschränkt wird.

[1] Wie in Kapitel 3 erwähnt werden im folgenden Verbindungsendpunkte mit Dienstzugangspunkten gleichgestellt.

4.1 In den Normen verwendete Beschreibungsmethoden

In allen unteren Schichten einschließlich der Transportschicht wird ein endlicher Automat zur Beschreibung der lokalen Einschränkungen an einem Dienstzugangspunkt verwendet. An den Kanten des Automaten werden die erlaubten Dienstelemente eingetragen. Alle anderen nicht angegebenen Dienstelemente sind unzulässig. In dieser Arbeit wird diese Methode zur Beschreibung der lokalen Einschränkungen mit dem Begriff *Automatenmethode* bezeichnet. Zur Erläuterung der Automatenmethode wird der Automat der lokalen Einschränkungen der Transportschicht verwendet [ISO84c, CCI88h], der in Abbildung 4.1 angegeben ist.

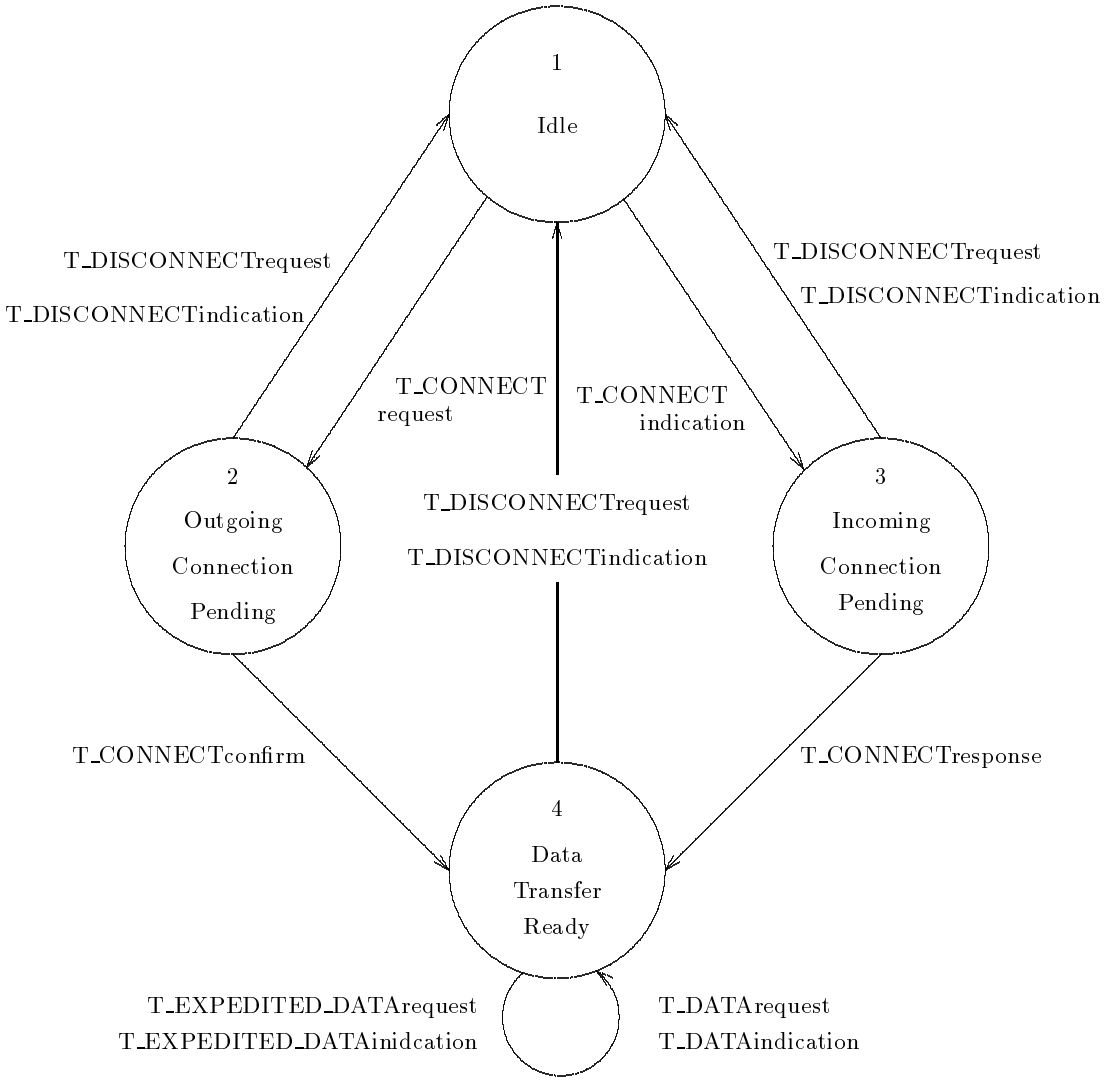

Abbildung 4.1: Erlaubte Aktionen an einem SAP in der Transportschicht

In Abbildung 4.1 werden die Dienstelemente zur Markierung der Zustandsübergänge

verwendet. Bei einer korrekten Sequenz von Dienstelementen ist ein Übergang innerhalb der angegebenen Zustandsmenge möglich. Als Anfangszustand ist hierbei der Zustand 1 ausgezeichnet. Durch den in Abbildung 4.1 dargestellten Automaten wird sichergestellt, daß zum Beispiel nach einem *T_DISCONNECTrequest* kein *T_DATArequest* in unmittelbarer Folge auftreten kann. So ist im Anfangszustand nur ein *T_CONNECTrequest* oder ein *T_CONNECTindication* möglich. Innerhalb der Normen wird ein gemeinsamer Automat für beide SAPs verwendet.

Nahezu in allen Schichten werden deterministische endliche Automaten verwendet. Eine Ausnahme bildet die Bitübertragungsschicht [ISO88c, CCI88e], in der ein endlicher Automat mit nichtdeterministischen Übergängen verwendet wird. Bei einer Transformation in einen deterministischen Automaten mit den gängigen Verfahren [AHU74] ändert sich zwar die Bedeutung der Zustände, aber dies ist bei der Formalisierung der lokalen Einschränkungen an einem Dienstzugangspunkt nicht weiter von Bedeutung.

Die Automatenmethode beruht auf der Beschreibung von internen Zuständen eines Diensterbringers. Nach [VSvSB91] ist dies ein *state-oriented* Spezifikationsstil und wird auch als abstrakte Implementierung bezeichnet. Die bei späteren Dienst- oder Protokollimplementierungen vorkommenden Zustände sind aber noch detaillierter.

Ein Nachteil der Automatenmethode ist eine durch die Einführung von Zuständen unter Umständen für die Protokollentwicklung vorweggenommene Implementierungsentscheidung, die bei einer Dienstbeschreibung nicht getroffen werden sollte. Damit ist die Forderung nach möglichst hoher Abstraktheit nicht erfüllt. Dies hat den Normenerstellern auch deutliches Unbehagen bei der Dienstbeschreibung der Transportschicht bereitet. Aus diesem Grund ist der folgende Zusatz nach der Automatenbeschreibung in [ISO84c, CCI88h] von den Normenerstellern eingefügt worden:

> The use of a state transition diagram to describe the allowable sequences of TS primitives does not impose any requirement or constraint on the internal organization of any implementation of the Transport Service.

Zur Umgehung dieses Nachteils wird in den Normen für die Transportschicht [ISO84c, CCI88h] als Ergänzung eine weitere Darstellungsform gewählt, bei der auf interne Zustände nicht näher eingegangen wird. Es werden in einer Tabelle zu jedem Dienstelement die erlaubten Nachfolgedienstelemente angegeben. In dieser Arbeit wird diese Methode zur Beschreibung der lokalen Einschränkungen mit dem Begriff *Tabellenmethode* bezeichnet.

	Diesem Dienstelement ↓									
kann das Dienstelement ↓ folgen	T_CONNECTrequest	T_CONNECTconfirm	T_CONNECTindication	T_CONNECTresponse	T_DATArequest	T_DATAindication	T_EXPEDITED_DATArequest	T_EXPEDITED_DATAindication	T_DISCONNECTrequest	T_DISCONNECTindication
T_CONNECTrequest										
T_CONNECTconfirm	+									
T_CONNECTindication										
T_CONNECTresponse			+							
T_DATArequest		+		+	+	+	+	+		
T_DATAindication		+		+	+	+	+	+		
T_EXPEDITED_DATArequest		+		+	+	+	+	+		
T_EXPEDITED_DATAindication		+		+	+	+	+	+		
T_DISCONNECTrequest	+	+	+	+	+	+	+	+		
T_DISCONNECTindication	+	+	+	+	+	+	+	+		

Tabelle 4.1: Tabelle der lokalen Einschränkungen der Transportschicht

Bei der aus der Dienstspezifikation der Transportschicht [ISO84c, CCI88h] entnommenen Tabelle 4.1 wird das Zeichen + für die Existenz einer möglichen Nachfolge von zwei Dienstelementen verwendet. Somit kann einem $T_CONNECTrequest$ unmittelbar nur ein $T_CONNECTconfirm$, $T_DISCONNECTrequest$ oder ein $T_DISCONNECTindication$ folgen. Ein Unterschied zwischen den oben angeführten Beschreibungsmethoden, auf den in [ISO84c, CCI88h] nicht hingewiesen wird, ist, daß bei der Tabellenmethode jede einelementige Spur akzeptiert wird. Dies ist eine Folge davon, daß nur die Korrektheit zweier unmittelbar aufeinanderfolgender Dienstelemente berücksichtigt wird. Bei der Automatenmethode wird dies durch die Angabe eines Startzustands vermieden. Eine Abhilfe bei der Tabellenmethode wäre ein Zusatz, der die Menge der im Anfangszustand möglichen Dienstelemente beschreibt, die im folgenden als Startaktionen bezeichnet werden. Die explizite Bezeichnung von Startaktionen erfolgt in [ISO84c, CCI88h] nicht, sollte aber aus oben angeführten Gründen unbedingt eingeschlossen werden.

Lemma 4.1

Jede tabellarische Beschreibung der lokalen Einschränkungen kann durch einen Automaten mit gleichem Sprachschatz ersetzt werden. □

Beweis:

Konstruktion eines nichtdeterministischen endlichen Automaten aus einer Tabelle: Die Zustandsmenge wird aus der Menge der Dienstelemente konstruiert. Dabei wird die Intuition zugrundegelegt, daß der Zustandsname eine vorher erfolgte Aktion beschreibt. s_{act} ist somit die Bezeichnung eines Zustands, der ausgehend von einem beliebigen Zustand mit der vorher erfolgten Aktion act erreicht wurde. s_ϵ sei der Startzustand, der spontan in die Menge der Startaktionen überführt. Sind keine erlaubten Startaktionen angegeben, so sind alle Zustände von s_ϵ aus spontan erreichbar. Die Zustandsübergangsfunktion wird ausschließlich durch die Tabelle beschrieben. Ist zum Beispiel b eine korrekte Nachfolgeaktion der Aktion a, dann führt die Aktion b im Zustand s_a zum Folgezustand s_b. Nach Konstruktion sind die akzeptierten Wortmengen gleich. (Induktion über den Aufbau der akzeptierten Worte) □

Lemma 4.2

Es gibt einen Automaten, dessen akzeptierte Wortmenge nicht mit der Tabellenmethode beschrieben werden kann. □

Beweis:

Gegeben sei der Automat aus Abbildung 4.2.

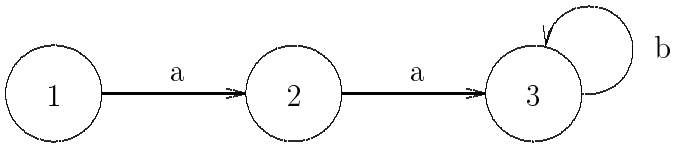

Abbildung 4.2: Mit der Tabellenmethode nicht darstellbarer Automat

Somit ist ein b nur dann erlaubt, wenn es einem doppelten a folgt.

Widerspruchsannahme: Es gibt eine Tabelle, die die gleiche Wortmenge wie der Automat aus Abbildung 4.2 akzeptiert. In einer Tabelle gibt es für den Eintrag unter der Zeile für a und der Spalte für a zwei Möglichkeiten:

1. a ist eine erlaubte Nachfolgeaktion von a, somit ist auch $<a, a, a>$ in der akzeptierten Wortmenge enthalten. Dies führt zu einem Widerspruch, da der Automat aus Abbildung 4.2 diese Spur nicht akzeptiert.
2. a ist keine erlaubte Nachfolgeaktion von a. Damit wird $<a, a>$ nicht akzeptiert, obwohl $<a, a>$ in der akzeptierten Wortmenge enthalten ist. Dies führt zum Widerspruch.

□

Satz 4.1

Die Menge der mit der Tabellenmethode festlegbaren Spurmengen ist eine echte Teilmenge der mit der Automatenmethode festlegbaren Spurmengen. □

Beweis:

Die Aussage folgt direkt aus den Lemmata 4.1 und 4.2. □

Aus Satz 4.1 folgt, daß nicht jede lokale Einschränkung, die man mit der Automatenmethode darstellen kann, mit der Tabellenmethode beschreibbar ist. Da aber die Tabellenmethode als abstraktere Beschreibungstechnik gegenüber der Automatenmethode zu bevorzugen ist, steht man vor dem Problem, daß dies wegen der geringeren Ausdruckskraft der Tabellenmethode nicht jederzeit möglich ist.

Wegen der größeren Ausdrucksstärke der Automatenmethode wird sie als Grundlage der folgenden Formalisierung verwendet.

4.2 Formale Semantik mit Einschränkung des Umgebungsverhaltens

Als Ausgangspunkt für eine formale Beschreibung wird ein nichtdeterministischer endlicher Automat nach [AHU74] verwendet, der geringfügig modifiziert wird:

Definition 4.1

Ein *nichtdeterministischer endlicher Automat* M ist ein Quintupel $(S, A, \delta, Start, F)$, wobei:

1. die Zustandsmenge S eine endliche Menge,
2. das Alphabet A eine endliche Menge von Zeichen,
3. $\delta : S \times A \rightarrow \wp(S)$ eine mengenwertige Zustandsübergangsfunktion,
4. $Start \subseteq S$ die Menge der Anfangszustände,
5. $F \subseteq S$ die Menge der Endzustände ist.

□

Im folgenden wird S_M als Zustandsmenge für den Automaten M verwendet. Bei den restlichen Bestandteilen des Quintupels wird ebenso der Index $_M$ zur Identifizierung eines bestimmten Automaten angefügt.

Bei der Definition der akzeptierten Wortmenge unter Verwendung der Endzustände kann bei unendlichen Spuren auch eine alternierende Zustandsfolge vorkommen. Es gibt somit keinen Fixpunkt der Zustandsmenge. Eine Möglichkeit zur Beschreibung der akzeptierten Wortmenge wäre die Verwendung von Büchiautomaten [Tho90], bei denen ein unendliches Wort genau dann akzeptiert wird, wenn ein Element aus der Endzustandsmenge bei einer Berechnung unendlich oft vorkommt. Dies ist jedoch zur Beschreibung der lokalen Einschränkungen nicht notwendig, da jeder angegebene Zustand ein Endzustand ist.

Definition 4.2

Die *von einem nichtdeterministischen endlichen Automaten M akzeptierte Sprachmenge* ist in Form eines Prädikates definiert als:

$$L_M(t) := \exists s :\ Trace\ S_M.\ \#s = \#t + 1 \wedge s[0] \in Start_M \\ \wedge\ \forall i.\ 0 \leq i \leq \#t \Rightarrow s[i+1] \in \delta_M(s[i], t[i])$$

□

Die Umwandlung der in den Normen verwendeten graphischen Darstellung in den oben angegebenen Formalismus wird wie folgt durchgeführt: Als Alphabet A wird die Menge aller vorkommenden Aktionen verwendet. Diese Aktionen sind die Kantenmarkierungen. Die Zustandsübergangsfunktion erhält man durch einfaches Ablesen:

1. Ist vom Zustand s_1 eine gerichtete Kante zum Zustand s_2 mit der Markierung a vorhanden, so ist $s_2 \in \delta(s_1, a)$ zu setzen.

2. Die Zustandsübergangsfunktion δ wird nur durch die Regel 1 bestimmt.

Als Anfangszustand wird der in den Normen benannte Zustand verwendet. Die Umsetzung der Norm in die oben angegebene Notation gestaltet sich somit problemlos.

In den Normen wird nur ein Automat bei einer Schichtbeschreibung angegeben, obwohl für jeden SAP ein eigener Automat benötigt wird. In dieser Arbeit wird deshalb der Automat einer Dienstbeschreibung dupliziert, somit erhält man einen einzelnen Automaten für jeden Dienstzugangspunkt, der dann mit einem Kriterium zur Unterscheidung gekennzeichnet wird. Dadurch erhält man die beiden Automaten M_A und M_B. Hierbei beschreiben die Indizes $_A$ und $_B$ den jeweiligen SAP, an dem lokal die Abfolge von Dienstelementen betrachtet wird. Diese Unterscheidung ist notwendig, da in den Normen ein einzelner Automat für beide SAPs verwendet wird, aber gleich bezeichnete Dienstelemente an verschiedenen SAPs möglich sind. Die Indizes M_A und M_B werden an die jeweiligen Zustände, Zustandsmengen, Alphabete und Zustandsübergangsfunktionen zur Unterscheidung angefügt. Die Korrektheit der lokalen Einschränkungen einer Schicht wird mit dem Prädikat $LC_{(M_A,M_B)}(t)$ bestimmt.

Definition 4.3

Für zwei endliche deterministische Automaten M_A und M_B wird die *Korrektheit der lokalen Einschränkungen* definiert durch:

$$LC_{(M_A,M_B)}(t) := L_{M_A}(A_{M_A}©t) \wedge L_{M_B}(A_{M_B}©t)$$

□

An der strikten Trennung der Automaten und Aktionen bezüglich der SAPs ist deutlich ersichtlich, daß bei der Bildung von lokalen Einschränkungen bei beiden SAPs nur jeweils ein Dienstzugangspunkt lokal betrachtet wird. Die Semantik der Automatenbeschreibung einer Schicht wird wie folgt definiert:

Definition 4.4

Für zwei endliche deterministische Automaten M_A und M_B wird die *Semantik der lokalen Einschränkungen* definiert durch:

$$[\![(M_A, M_B)]\!]_{\mathcal{LC}} := \{t | LC_{(M_A,M_B)}(t)\}$$

□

Beispiel 4.1

Seien M_A und M_B die durch die Abbildung 4.1 auf Seite 82 definierten Automaten der Transportschicht, wobei der Automat am SAP des linken Dienstbenutzers den Index $_A$ und der des rechten Dienstbenutzers den Index $_B$ erhält. So gilt:

$$T_CONNECTrequest_A\&<T_CONNECTconfirm_A> \in [\![(M_A, M_B)]\!]_{\mathcal{LC}}$$

$$T_CONNECTindication_B\&T_CONNECTresponse_B\& \\ <T_DATAindication_B> \in [\![(M_A, M_B)]\!]_{\mathcal{LC}}$$

Ein Beispiel für die Verknüpfung von Dienstelementen, die an beiden SAPs auftreten, ist:

$$T_CONNECTrequest_A\&T_CONNECTindication_B\& \\ T_CONNECTresponse_B\&<T_CONNECTconfirm_A> \in [\![(M_A, M_B)]\!]_{\mathcal{LC}}$$

Auch die folgende Permutation der obigen Spur, die von den TSDs der Transportschicht nicht zugelassen wird, ist in der Menge der lokalen Einschränkungen enthalten:

$$T_CONNECTindication_B\&T_CONNECTrequest_A\& \\ T_CONNECTconfirm_A\&<T_CONNECTresponse_B> \in [\![(M_A, M_B)]\!]_{\mathcal{LC}}$$

□

4.3 Eigenschaften der formalen Spezifikation

Die Verwendung von Büchiautomaten [Tho90] zur Beschreibung von akzeptierten unendlichen Spuren ist nicht notwendig, da das Prädikat L_M und damit auch LC kettenvollständig beziehungsweise *admissible* [Pau87] ist:

Lemma 4.3

Sei M ein nichtdeterministischer Automat nach Definition 4.1, sowie $\{t_i\}$ eine Kette, dann gilt:

$$(\forall i.\ L_M(t_i)) \Rightarrow L_M(\bigsqcup_i t_i)$$

□

Beweis:

Annahme:

$$\forall i.\ L_M(t_i)$$

Sei

$$t := \bigsqcup_i t_i$$

Sei M' der zu M äquivalente deterministische Automat, der mit den Verfahren aus [AHU74] erhalten wurde. Somit gilt:

$$\begin{aligned}\forall i.\ \exists s_i.\ \#s_i = \#t_i + 1 \wedge\ & s_i[0] \in Start_{M'} \\ \wedge\ \forall j.\ 0 \leq j < \#t_i + 1 \Rightarrow\ & s_i[j+1] \in \delta_{M'}(s_i[j], t_i[j])\end{aligned}$$

Nach der Konstruktion aus [AHU74] beschreibt $s_i[j]$ aus dem deterministischen Automaten M' eine Menge von Zuständen aus dem Automaten M. Somit beschreiben beide Automaten die gleiche akzeptierte Spurmenge.

Da M' deterministisch ist, gilt:

$$|\delta_{M'}(s_i[j], t_i[j])| = 1$$

Daraus folgt, daß $\{s_i\}$ ein Kette ist. (Beweis durch Widerspruch). Sei

$$s := \bigsqcup_i s_i$$

Daraus folgt dann $L_M(t)$ □

Lemma 4.4

Sei M ein nichtdeterministischer Automat nach Definition 4.1, dann gilt:

$$L_M(t) \Rightarrow \forall t' \sqsubseteq t.\ L_M(t')$$

□

Beweis:

Es gilt:

$$\exists s.\ \#s = \#t + 1 \wedge s[0] \in Start_M \wedge \forall j.\ 0 \leq j \leq \#t \Rightarrow s[j+1] \in \delta_M(s[j], t[j])$$

Sei $s' \sqsubseteq s$ und $\#s' = \#t' + 1$, dann gilt:

$$\exists s'.\ \#t' = \#s' + 1 \wedge s'[0] \in Start_M \wedge \forall j.\ 0 \leq j \leq \#t' \Rightarrow s'[j+1] \in \delta_M(s'[j], t'[j])$$

□

Satz 4.2

Seien M_A und M_B nichtdeterministische Automaten nach Definition 4.1, dann beschreibt $[\![(M_A, M_B)]\!]_{\mathcal{LC}}$ eine Sicherheitseigenschaft. □

Beweis:

Die Aussage folgt direkt aus der Definition 4.4 und der Kombination der Lemmata 4.3 und 4.4, sowie aus der Tatsache, daß die konjunktive Verknüpfung zweier Sicherheitseigenschaften wieder eine Sicherheitseigenschaft ergibt [Web92]. □

4.4 Formale Semantik ohne Einschränkung des Umgebungsverhaltens

Im Kapitel 4.2 wird keine substantielle Unterscheidung zwischen Ein- und Ausgabeaktionen getroffen. Ebenso wie bei der im Kapitel 3.3 für TSDs verwendeten Methode wird hierdurch das Umgebungsverhalten eingeschränkt. Da die lokalen Einschränkungen nach Satz 4.2 eine Sicherheitseigenschaft darstellen, ist die Erweiterung um fehlende Eingaben, die eine Verletzung des Lebendigkeitsanteils der Assumption darstellen, nach Satz 3.9 hinfällig. Eine Erweiterung um fehlerhafte Eingaben, die ein jederzeitiges Agieren der Umgebung erlauben, wird äquivalent zu dem im Kapitel 3.4.2 bei den TSDs verwendeten Verfahren vorgenommen. Für die lokalen Einschränkungen bei freiem Umgebungsverhalten ergibt sich:

Definition 4.5

Für zwei endliche deterministische Automaten M_A und M_B, die Eingabemenge I und die Ausgabemenge O wird die *Semantik der lokalen Einschränkungen unter freiem Umgebungsverhalten* definiert als:

$$[\![(M_A, M_B)]\!]_{\mathcal{LC}}^{(I,O)} := [\![(M_A, M_B)]\!]_{\mathcal{LC}} \cup E_{false_inp}([\![(M_A, M_B)]\!]_{\mathcal{LC}}, I, O)$$

□

Hierbei ist $E_{false_inp}(T, I, O)$ wie in Definition 3.8 auf Seite 52 festgelegt. Somit wird eine jederzeitige Eingabe an den Diensterbringer ermöglicht. Ist eine Spur nach einer derartigen Eingabe nicht mehr korrekt fortsetzbar, so kann sie beliebig ergänzt werden, da der Diensterbringer sich seinerseits korrekt verhalten hat und nur ein Dienstbenutzer einen Fehler gemacht hat, den der Diensterbringer nicht mehr korrigieren kann.

Beispiel 4.2

Seien M_A und M_B die durch Abbildung 4.1 auf Seite 82 definierten Automaten der Transportschicht, deren SAPs mit den Indizes $_A$ und $_B$ unterschieden werden, und enthalte I diejenigen Dienstelemente, die auf *request* sowie *response* enden, O die auf *indication* und $confirm$ endenden, so gilt:

$$<T_DATArequest_A> \notin [\![(M_A, M_B)]\!]_{\mathcal{LC}}$$

Aber:

$$<T_DATArequest_A> \in [\![(M_A, M_B)]\!]_{\mathcal{LC}}^{(I,O)}$$

Dies gilt, da die Eingabeaktion $T_DATArequest_A$ vorzeitig erfolgt. Ein Beispiel für die Verknüpfung von Dienstelementen, die an beiden SAPs auftreten, ist:

$$T_CONNECTrequest_A \& T_CONNECTindication_B \& \\ <T_DATArequest_A> \in [\![(M_A, M_B)]\!]_{\mathcal{LC}}^{(I,O)}$$

□

Lemma 4.5

Beschreibt die Menge X eine Sicherheitseigenschaft, so beschreibt die Menge $X \cup E_{false_inp}(X, I, O)$ ebenfalls eine Sicherheitseigenschaft. □

Beweis:

1) Zu zeigen ist, wenn $\{t_i\}$ eine Kette ist, dann gilt:
$(\forall t_i.\ t_i \in X \cup E_{false_inp}(X, I, O)) \Rightarrow \bigsqcup_i t_i \in X \cup E_{false_inp}(X, I, O)$

Fallunterscheidung nach $\exists i.\ t_i \in E_{false_inp}(X, I, O)$:
a) $\exists i'.\ t_{i'} \in E_{false_inp}(X, I, O)$ wegen der Konstruktion von E_{false_inp} können dann beliebige Elemente an $t_{i'}$ angefügt werden. Aus $t_{i'} \sqsubseteq \bigsqcup_i t_i$ folgt: $\bigsqcup_i t_i \in E_{false_inp}(X, I, O)$
b) $\forall i.\ t_i \in X$
Da X eine Sicherheitseigenschaft ist, folgt: $\bigsqcup_i t_i \in X$

2) $t \in X \cup E_{false_inp}(X, I, O) \Rightarrow (\forall t'.\ t' \sqsubseteq t \Rightarrow t' \in X \cup E_{false_inp}(X, I, O))$
Fallunterscheidung nach $t \in X$:
a) $t \in X$: Da X eine Sicherheitseigenschaft ist, gilt: $\forall t'.\ t' \sqsubseteq t \Rightarrow t' \in X$
b) $t \in E_{false_inp}(X, I, O)$:
Somit: $\exists t'', i'', x.\ t = t'' \circ i'' \& x \wedge t'' \in PRE(X) \wedge t'' \circ <i''> \notin PRE(X)$
Damit gilt entweder: $t' \sqsubseteq t'' \Rightarrow t' \in X$, da X eine Sicherheitseigenschaft beschreibt.
Oder bei $t'' \sqsubset t'$gilt: $t' \in E_{false_inp}(X, I, O)$, da beliebige Elemente an $t'' \& <i''>$ angefügt werden können.

□

Satz 4.3

Seien M_A und M_B nichtdeterministische endliche Automaten nach Definition 4.1 und I, O Mengen von Aktionen, dann beschreibt $[\![(M_A, M_B)]\!]_{\mathcal{LC}}^{(I,O)}$ eine Sicherheitseigenschaft. □

Beweis:

Die Aussage folgt aus Lemma 4.5 und Satz 4.2. □

Im nächsten Kapitel wird die Formalisierung der lokalen Einschränkungen bei den einzelnen Schichten untersucht. Hierbei stellt sich die Frage, ob bei denjenigen Schichten, die ausschließlich die Automatenmethode benutzen, die Tabellenmethode verwendbar ist. Im folgenden wird eine für das Schichtenmodell ausreichende Klasse von Automaten ausgezeichnet, bei denen eine Transformation in eine Tabelle möglich ist. Dazu ist es erforderlich, den betrachteten nichtdeterministischen Automaten M mit den üblichen Verfahren [AHU74] in einen deterministischen Automaten M' umzuwandeln. Bei einem deterministischen Automaten gilt dann in der Definition 4.1: $\delta_{M'}(s,a) = \emptyset \vee \exists s'.\ \delta_{M'}(s,a) = \{s'\}$. Im folgenden wird für deterministische Automaten $\delta_{M'_f}$ als Funktion verwendet: $\delta_{M'_f}(s,a) = s'$. Falls der Zustandsübergang nicht möglich ist, also $\delta_{M'}(s,a) = \emptyset$ gilt, wird bei deterministischen Automaten $\delta_{M'_f}(s,a) = \perp$ gesetzt. Die Funktion $\delta_{M'_f}$ ist zudem strikt definiert.

Zur Bildung eines Kriteriums für die Beurteilung der Transformierbarkeit in eine Tabelle wird die Menge aller Zustände, die durch die Aktion a in einem Schritt erreichbar sind, als $STATES_M(a)$ festgelegt:

Definition 4.6

Für einen deterministischen endlichen Automaten M und $a \in A_M$ wird die *Menge der von der Aktion a in einem Schritt erreichbaren Zustände* $STATES_M(a)$ definiert als:

$$STATES_M(a) := \{s' | \exists s \in S.\ \delta_M(s,a) = s'\}$$

□

Satz 4.4

Gilt für einen deterministischen endlichen Automaten M:

$$\forall a \in A_M.\ |STATES_M(a)| \leq 1,$$

so läßt er sich in eine bezüglich der akzeptierten Wortmenge äquivalente Tabelle in Kombination mit einer Menge der erlaubten Startaktionen umwandeln. □

Beweis:

Konstruktion einer Tabelle, die zur Vereinfachung als Relation R bezeichnet wird, aus dem deterministischen Automaten M:

$$(a,b) \in R \ :\Leftrightarrow \exists s', s \in S_M.\ \delta_M(\delta_M(s,a),b)) = s'$$

Die Menge der erlaubten Startaktionen $StartAct$ eines deterministischen Automaten M wird festgelegt als:

$$StartAct := \{a \in A_M | \exists s \in Start_M.\ \delta_M(s,a) \in S_M\}$$

Der Sprachschatz einer Relation R wird wie folgt festgelegt:

$$L(R, StartAct) := \{\epsilon\} \cup \{t | t[0] \in StartAct \wedge \forall t', a, b.\ t' \circ <a,b> \sqsubseteq t \Rightarrow (a,b) \in R\}$$

Der Sprachschatz des Automaten wird nach Definition 4.2 mit L_M bezeichnet.

Beweis der Äquivalenz beider Sprachen:

1) $L_M(t) \Rightarrow t \in L(R, StartAct))$
Widerspruchsannahme: $t \notin L(R, StartAct))$
$t[0] \notin StartAct$ kann wegen $L_M(t)$ nicht erfüllt sein, somit
$\Rightarrow \exists t', a, b.\ t' \circ <a, b> \sqsubseteq t \wedge (a, b) \notin R$
$\Rightarrow \exists t', a, b.\ t' \circ <a, b> \sqsubseteq t \wedge \forall s, s' \in S_M.\ \delta_M(\delta_M(s, a), b)) \neq s'$
Somit: $\neg L_M(t)$

2) $t \in L(R, StartAct)) \Rightarrow L_M(t)$
Für $t = \epsilon$ gilt $L_M(t)$ sowieso.
Es gilt $t[0] \in StartAct \wedge \forall t', a, b.\ t' \circ <a, b> \sqsubseteq t \Rightarrow (a, b) \in R$
Sei s wie folgt induktiv definiert:
$s[0] \in Start_M$, wobei dies eindeutig ist, da es bei deterministischen Automaten einen eindeutigen Startzustand gibt.
$s[i+1] = \delta_M(s[i], t[i])$
Es gilt für $i < \#t$: $|STATES_M(t[i])| = 1$, wegen Voraussetzung und
$\forall i < \#t.\ (t[i], t[i+1]) \in R$
Es gilt somit: $STATES_M(t[i]) = \{s[i+1]\}$
Beweis per Widerspruch: $STATES_M(t[i]) = \{s'\} \wedge s' \neq s[i+1]$
Somit: $\delta_M(s[i], t[i]) = s'$. Nach Definition von $s[i]$ gilt $s[i+1] = s'$ und dies führt zum Widerspruch
Somit erhält man durch Induktion über i: $L_M(t)$

□

Damit ist eine Unterscheidung der Schichten nach dem Kriterium der Verwendbarkeit der Tabellenmethode möglich. Jedoch sollte wegen der größeren Abstraktion die Tabellenmethode immer dann verwendet werden, wenn dies möglich ist.

4.5 Anwendung der Formalisierungsmethodik auf die Schichten

In diesem Kapitel werden die Formalisierungen der lokalen Einschränkungen bei den einzelnen Schichten betrachtet. So wird detailliert überprüft, ob die jeweils angegebene Automatendarstellung in eine äquivalente Tabellendarstellung überführt werden kann. Dies ist bei der Bitübertragungsschicht, Sicherungsschicht und Vermittlungsschicht von besonderem Interesse, da in den Normen bei diesen ausschließlich die Automatenmethode verwendet wird. Mit Satz 4.4 kann nachgewiesen werden, daß die Angabe einer äquivalente Tabellendarstellung nur in der Sicherungsschicht und Vermittlungsschicht möglich ist.

4.5.1 Lokale Einschränkungen der Bitübertragungsschicht

In der Bitübertragungsschicht [ISO88c, CCI88e] wird ein nichtdeterministischer, endlicher Automat verwendet.

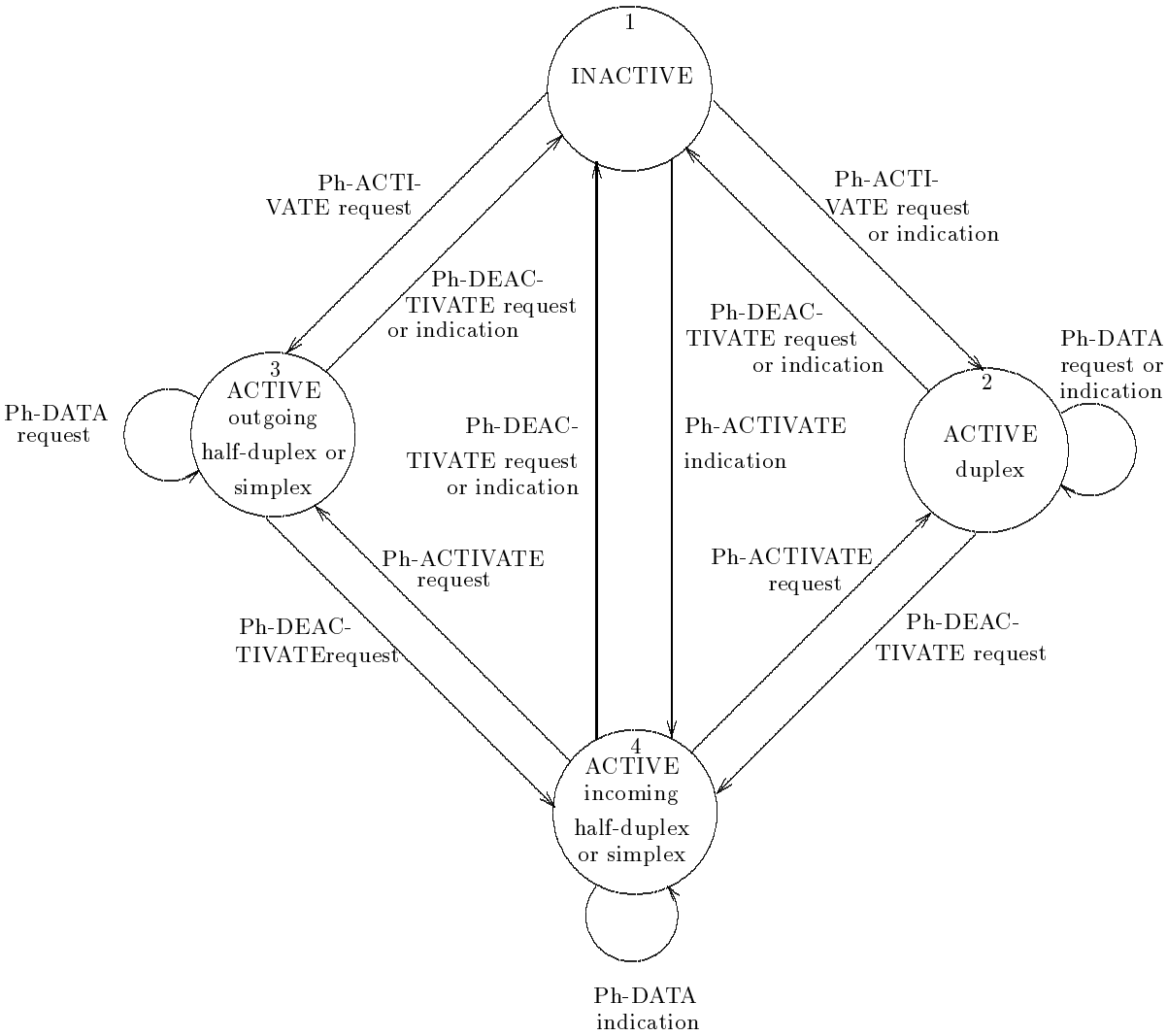

Abbildung 4.3: Nichtdeterministischer Automat der Bitübertragungsschicht

In der Norm wird kein expliziter Anfangszustand festgelegt. Als Anfangszustand bietet sich der Zustand 1 (INACTIVE) an. Der in Abbildung 4.3 dargestellte Automat kann durch folgenden äquivalenten, deterministischen Automaten ersetzt werden.

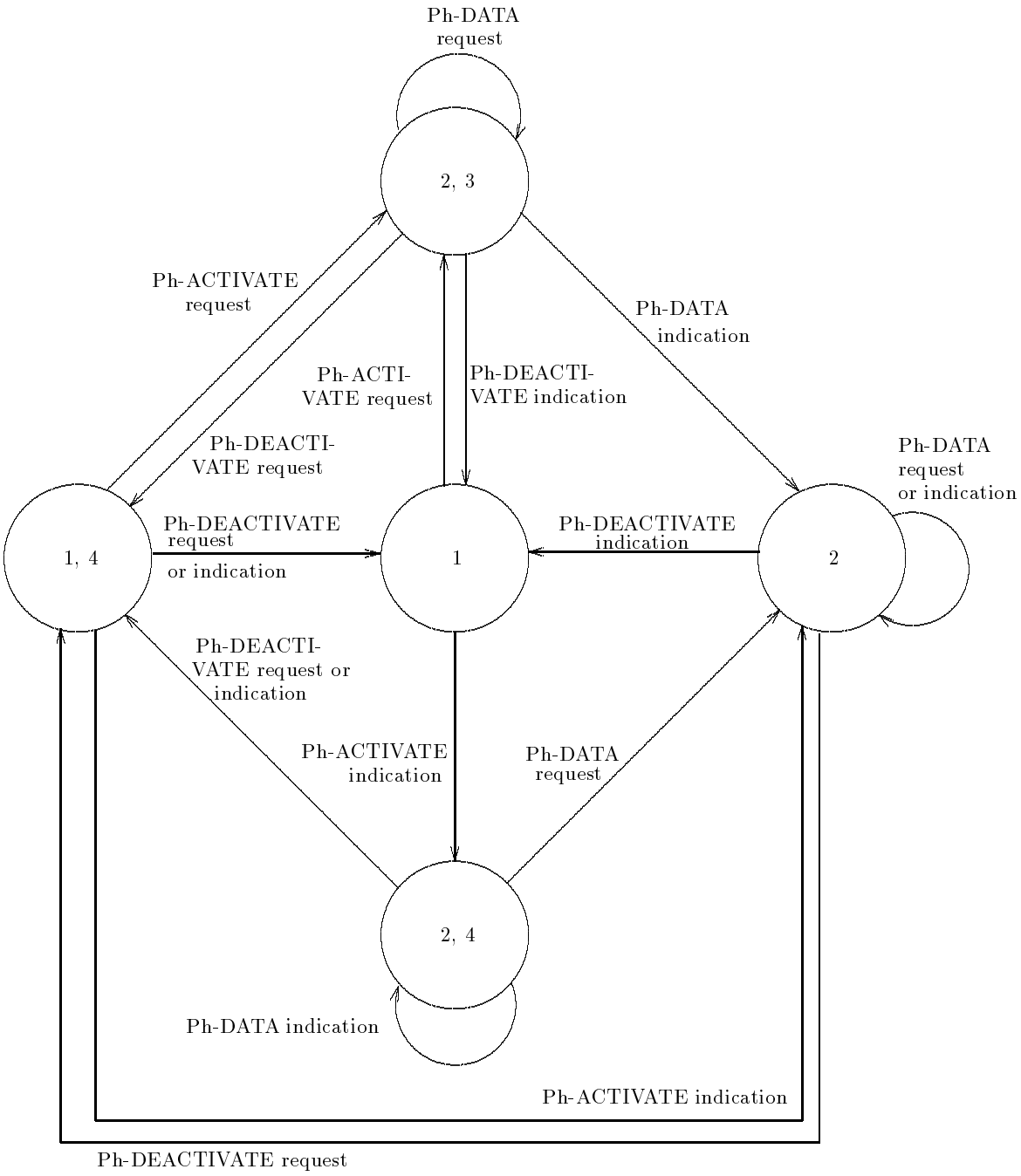

Abbildung 4.4: Deterministischer Automat der Bitübertragungsschicht

Der in Abbildung 4.4 dargestellte Automat besitzt den Startzustand 1. Dieser Automat genügt nicht der Prämisse aus Satz 4.4, daß jede Aktion in höchstens einem Zustand mündet. Er kann darüber hinaus nicht durch eine äquivalente Tabelle ersetzt werden. Die Spur

$$Ph_ACTIVATErequest\&Ph_DEACTIVATErequest\& \\ <Ph_DEACTIVATErequest>$$

ist Element der vom Automaten beschriebenen Sprache, wohingegen

$$Ph_ACTIVATErequest\&Ph_DEACTIVATErequest\& \\ Ph_DEACTIVATErequest\&<Ph_DEACTIVATErequest>$$

nicht Element dieser Sprachmenge ist. Daß die erste Spur von der Tabelle akzeptiert wird, ist mit

$$(Ph_DEACTIVATErequest, Ph_DEACTIVATErequest) \in R$$

zu erreichen. Dadurch ist auch die oben genannte zweite Spur Element der Sprachmenge. Somit gibt es keine Tabelle, in der die erste Spur akzeptiert wird, die zweite jedoch nicht.

4.5.2 Lokale Einschränkungen der Sicherungsschicht

In der Sicherungsschicht [ISO88b, CCI88f] beschreibt der in Abbildung 4.5 dargestellte deterministische endliche Automat die lokalen Einschränkungen. In den Normen wird im Gegensatz zur Bitübertragungsschicht der Zustand 1 (idle) explizit als Anfangszustand bezeichnet.

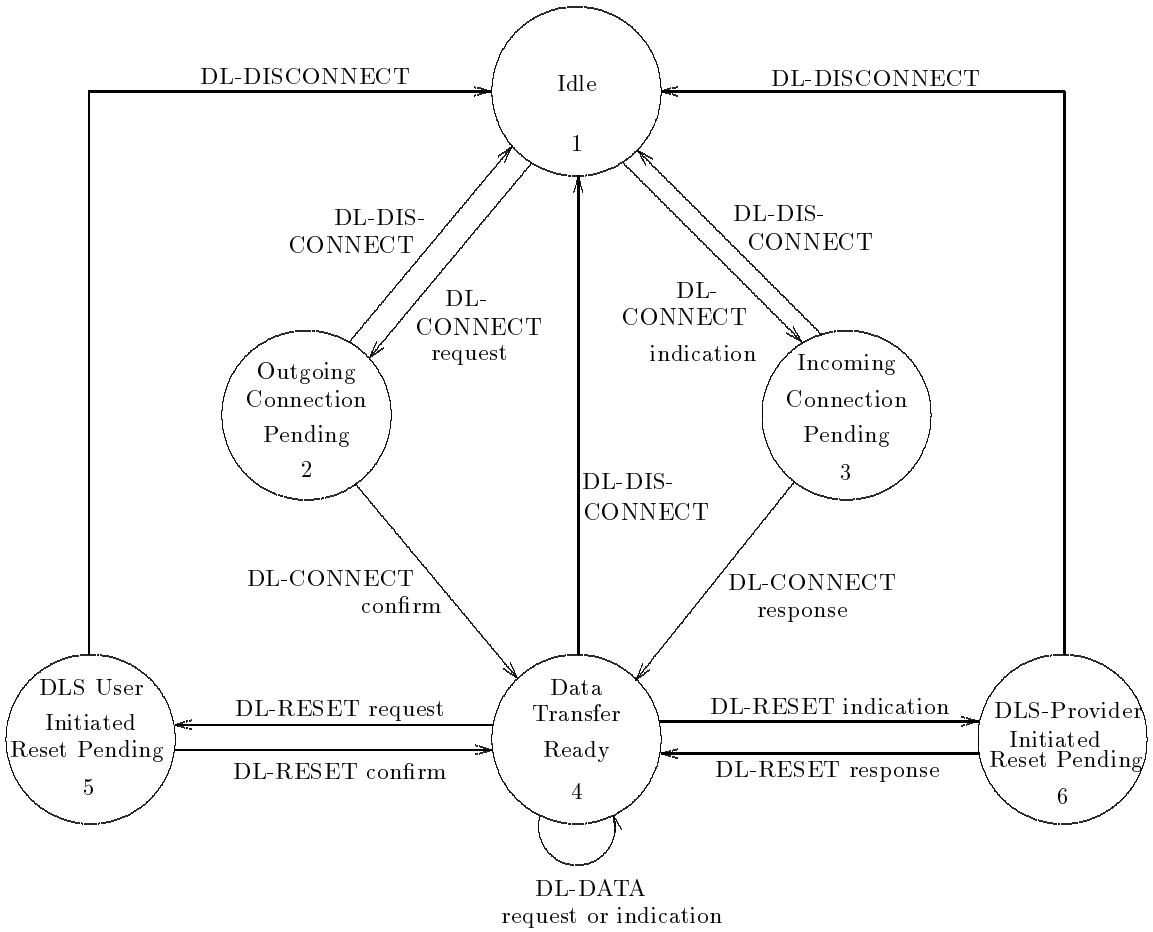

Abbildung 4.5: Automat der Sicherungsschicht

Die Aktion DL_DISCONNECT wird in der Abbildung 4.5 nach [ISO88c, CCI88e] als Abkürzung für DL_DISCONNECTrequest und DL_DISCONNECTindication verwendet. Es gilt:

$$\forall a \in A.\ |STATES_M(a)| \leq 1$$

Nach Satz 4.4 lassen sich die lokalen Einschränkungen durch die in den Normen nicht angegebene Tabelle 4.2 darstellen:

	Diesem Dienstelement ↓											
kann das Dienstelement ↓ folgen	DL_CONNECTrequest	DL_CONNECTconfirm	DL_CONNECTindication	DL_CONNECTresponse	DL_RESETrequest	DL_RESETconfirm	DL_RESETindication	DL_RESETresponse	DL_DATArequest	DL_DATAindication	N_DISCONNECTrequest	N_DISCONNECTindication
DL_CONNECTrequest											+	+
DL_CONNECTconfirm	+											
DL_CONNECTindication											+	+
DL_CONNECTresponse			+									
DL_RESETrequest		+		+		+		+	+	+		
DL_RESETconfirm					+							
DL_RESETindication		+		+		+		+	+	+		
DL_RESETresponse							+					
DL_DATArequest		+		+		+		+	+	+		
DL_DATAindication		+		+		+		+	+	+		
DL_DISCONNECTrequest	+	+	+	+	+	+	+	+	+	+		
DL_DISCONNECTindication	+	+	+	+	+	+	+	+	+	+		

Tabelle 4.2: Tabelle der lokalen Einschränkungen der Sicherungsschicht

Hierbei ist es möglich, daß nach einem Verbindungsabbau mit einem *DL_DISCONNECTindication* eine neue Verbindung mit einem *DL_CONNECTrequest* aufgebaut wird. Dies ist durch die schematische Umformung möglich und im Gegensatz zur Tabelle 4.1 der Transportschicht durchgeführt worden. Die Menge der erlaubten Startaktionen wird festgelegt als:

$$StartAct = \{DL_CONNECTrequest, DL_CONNECTindication\}$$

4.5.3 Lokale Einschränkungen der Vermittlungsschicht

Die lokalen Einschränkungen in der Vermittlungsschicht [ISO87c, CCI88g] beschreibt der in Abbildung 4.6 dargestellte deterministische endliche Automat. In den Normen wird der Zustand 1 (idle) explizit als Anfangszustand ausgewiesen.

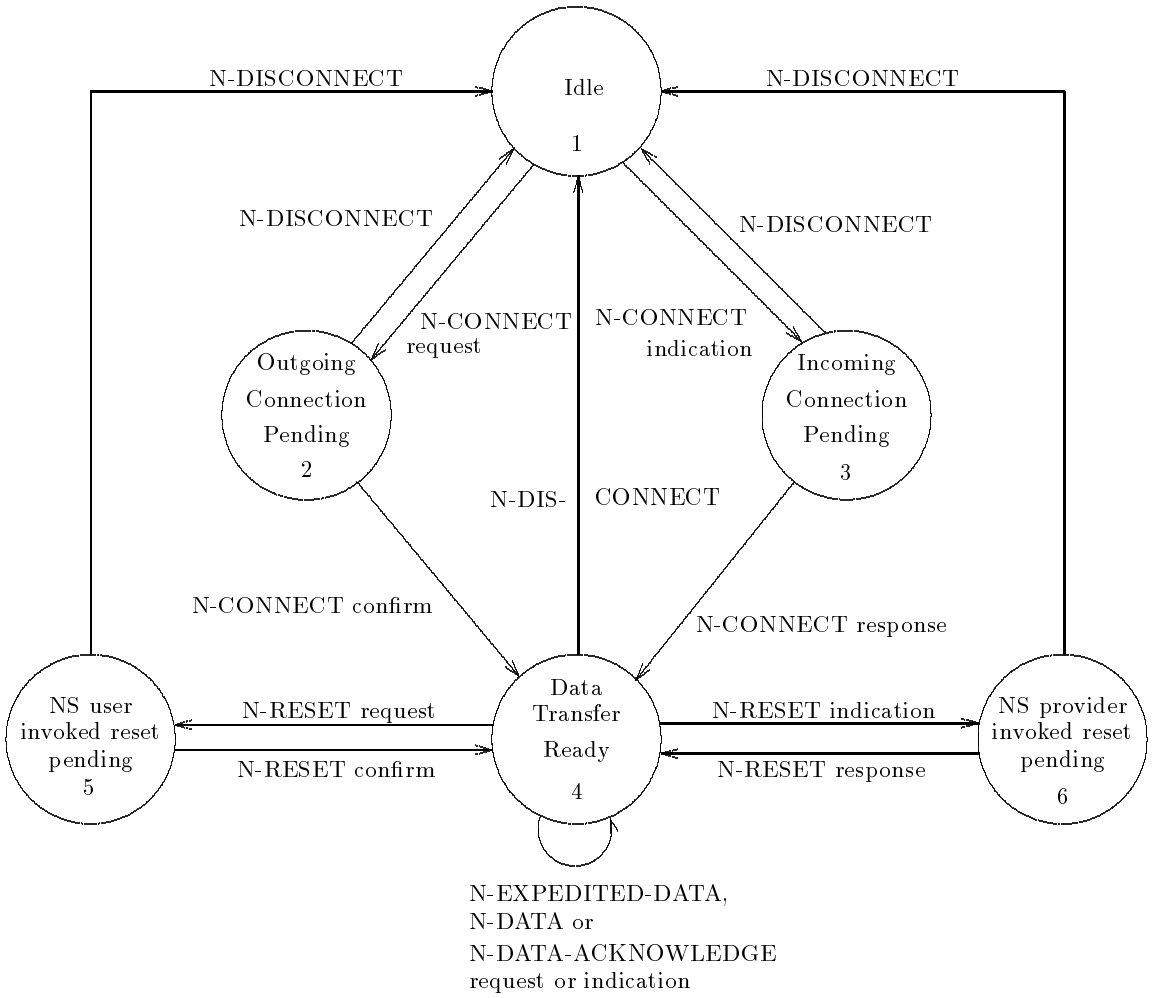

Abbildung 4.6: Automat der Vermittlungsschicht

Der Automat der Vermittlungsschicht ist nahezu äquivalent zu dem der Sicherungsschicht. Es werden nur einige Zusatzaktionen bei der Datenübertragung eingeführt, die einen priorisierten sowie quittierten Datenaustausch beschreibbar machen. Diese Dienstelemente sind natürlich nur im Zustand 4, der die Bereitschaft zum Datenaustausch darstellt, möglich.

Es gilt:

$$\forall a \in A.\ |STATES_M(a)| \leq 1$$

Nach Satz 4.4 lassen sich die lokalen Einschränkungen durch die Tabelle 4.3 darstellen, die in den Normen nicht angegeben ist.

	Diesem Dienstelement ↓															
kann das Dienstelement ↓ folgen	N_CONNECTrequest	N_CONNECTconfirm	N_CONNECTindication	N_CONNECTresponse	N_RESETrequest	N_RESETconfirm	N_RESETindication	N_RESETresponse	N_DATArequest	N_DATAindication	N_DATA_ACKNOWLEDGErequest	N_DATA_ACKNOWLEDGEindication	N_EXPEDITED_DATArequest	N_EXPEDITED_DATAindication	N_DISCONNECTrequest	N_DISCONNECTindication
N_CONNECTrequest															+	+
N_CONNECTconfirm	+															
N_CONNECTindication															+	+
N_CONNECTresponse			+													
N_RESETrequest		+		+		+		+	+	+	+	+	+	+		
N_RESETconfirm					+											
N_RESETindication		+		+		+		+	+	+	+	+	+	+		
N_RESETresponse							+									
N_DATArequest		+		+		+		+	+	+	+	+	+	+		
N_DATAindication		+		+		+		+	+	+	+	+	+	+		
N_DATA_ACKNOWLEDGErequest		+		+		+		+	+	+	+	+	+	+		
N_DATA_ACKNOWLEDGEindication		+		+		+		+	+	+	+	+	+	+		
N_EXPEDITED_DATArequest		+		+		+		+	+	+	+	+	+	+		
N_EXPEDITED_DATAindication		+		+		+		+	+	+	+	+	+	+		
N_DISCONNECTrequest	+	+	+	+	+	+	+	+	+	+	+	+	+	+		
N_DISCONNECTindication	+	+	+	+	+	+	+	+	+	+	+	+	+	+		

Tabelle 4.3: Tabelle der lokalen Einschränkungen der Vermittlungsschicht

Die Menge der erlaubten Startaktionen wird festgelegt als:

$$StartAct = \{N_CONNECTrequest, N_CONNECTindication\}$$

4.5.4 Lokale Einschränkungen der Transportschicht

In der Dienstspezifikation der Transportschicht [ISO84c, CCI88h] beschreibt der in Abbildung 4.1 auf Seite 82 dargestellte Automat die lokalen Einschränkungen. Es ist als expliziter Anfangszustand idle angegeben. Die Tabelle 4.1 von Seite 84 befindet sich in der Norm ohne einen verbalen Hinweis auf die erlaubten Startaktionen. Die Menge der erlaubten Startaktionen wird festgelegt als:

$$StartAct = \{T_CONNECTrequest, T_CONNECTindication\}$$

Bei dieser Beschreibung ist nach einem $T_DISCONNECTindication$ kein neues Dienstelement mehr erlaubt. Bei einer strengen Auslegung der Tabelle ist es nur möglich, eine einzige Verbindung zu erhalten. Damit wären die beiden Beschreibungsmethoden nicht äquivalent. Eine weniger strenge Interpretation beschränkt die Gültigkeit der Tabelle auf eine Verbindung. Damit wird die Tabelle innerhalb jeder neuen Verbindung überprüft. Ein derartiger Hinweis fehlt an dieser Stelle in der Norm. Es wird nur bei der

Automatenbeschreibung erwähnt, daß das Ende einer Verbindung eine neue Betrachtung anstößt. Bei der tabellarischen Beschreibung ist nur der Zusatz vorhanden, daß die Tabelle einen zur Automatenbeschreibung äquivalenten Sachverhalt beschreibt.

4.6 Bewertung der Formalisierungsmethodik

Mit der angegebenen Beschreibungsmethodik lassen sich die lokalen Einschränkungen an einem Dienstzugangspunkt für eine beliebige Schicht sehr einfach spezifizieren. Dies erfolgt in einem schichtenspezifischen Schritt durch die Spezifikation der herauszufilternden Elemente am jeweiligen Dienstzugangspunkt, sowie der Zustandsübergangsfunktion und der Zustände des in der Norm dargestellten Automaten. Interpretationsschwierigkeiten bei endlichen Abläufen, wie bei den TSDs, kommen nicht vor, da die in den Normen vorhandenen Beschreibungsformen sehr nahe an einem formalen Modell orientiert sind.

Da die lokalen Einschränkungen eine Sicherheitseigenschaft darstellen, ist die Interpretation von unendlichen Strömen wesentlich einfacher als bei den TSDs.

Da die Parameter der Dienstelemente in den Normen ebenso wie bei den TSDs nicht einbezogen werden, sind sie bei der Formalisierung nicht berücksichtigt worden. Das Einbeziehen der Parameter ist problemlos mit der in Kapitel 3.5.2 erwähnten Abstraktionsfunktion möglich.

4.7 Bewertung der in den Normen verwendeten Beschreibungstechniken

Mit der in den ISO/OSI Normen beziehungsweise CCITT Empfehlungen angegebenen Automatenmethode sind die Einschränkungen an einem Dienstzugangspunkt für einen Dienst einer beliebigen Schicht beschreibbar. Dieses Verfahren ist bei den lokalen Einschränkungen verwendbar, die durch reguläre Ausdrücke darstellbar sind. Ein nicht mehr mit dieser Methode darstellbarer Spezialfall ist zum Beispiel eine Klammerstruktur. Nach einem genau n-maligen Ereignis a folgt ebenso oft das Ereignis b. Komplexere Bedingungen bereiten ebenfalls Darstellungsprobleme. Derartige Ausdrücke finden jedoch in den Normen keine Verwendung, da dieser Aspekt bereits von den TSDs berücksichtigt wird. Deswegen ist die Ausdruckskraft dieser Methode ausreichend.

Ein Nachteil der Automatendarstellung ist eine für den Protokollentwickler unter Umständen vorweggenommene Implementierungsentscheidung, die bei einer Dienstbeschreibung nicht getroffen werden sollte. Dies hat den Normenerstellern auch deutliches Unbehagen bei der Dienstbeschreibung der Transportschicht bereitet. Aus diesem Grund ist die tabellarische Methode in der Dienstbeschreibung der Transportschicht [ISO84c, CCI88h] zusätzlich als äquivalente Erklärung eingeführt worden. Wie in Kapitel 4.5.4 festgestellt wird, beschreibt die Tabelle nur höchstens eine Verbindung, der Automat hingegen unter Umständen mehrere. Zusätzlich wird bei der Tabellenmethode das erste Element nicht benannt. Somit ist bei strenger Auslegung ein Unterschied zwischen beiden Darstellungsweisen vorhanden, auf den in [ISO84c, CCI88h] nicht hingewiesen wird. Dieser Unterschied wäre aber mit einem kleinen textuellen Zusatz in den Normen, der die

im Anfangszustand möglichen Aktionen kennzeichnet, zu beheben. Daß dies trotz des hohen Formalisierungsniveaus bei den lokalen Einschränkungen den Normenerstellern nicht aufgefallen ist, verdeutlicht die Notwendigkeit des Einsatzes von formalen Methoden.

Obwohl die Tabellenmethode eigentlich wegen der höheren Abstraktheit der Automatenmethode vorzuziehen ist, konnte in dieser Arbeit durch Satz 4.1 gezeigt werden, daß dies wegen der geringeren Ausdrucksmächtigkeit nicht immer möglich ist. Dies ist auch in den Normen am Beispiel der Bitübertragungsschicht [ISO88c, CCI88e] erkennbar, da in Kapitel 4.5.1 erstmalig gezeigt wurde, daß keine bezüglich der akzeptierten Wortmenge äquivalente Tabelle existieren kann. Somit wird mit dieser Arbeit belegt, daß die Beschreibung der lokalen Einschränkungen problematisch ist und das folgende Verfahren angewendet werden sollte: Wegen des höheren Abstraktionsgrads ist die Tabellenmethode zu verwenden, solange sie zur Darstellung genügt. Ist dies nicht der Fall, so ist auf die universellere, aber implementierungsnähere Automatenmethode zurückzugreifen.

Kapitel 5

Formalisierung des Queuemodells

Ein drittes, in den unteren Schichten bis einschließlich der Transportschicht verwendetes Beschreibungsmittel ist die abstrahierte Darstellung einer Warteschlange, die in dieser Arbeit als *Queuemodell (QM)* bezeichnet wird.

Mit den in den Kapiteln 3 und 4 formalisierten Beschreibungstechniken TSD und LC lassen sich bestimmte Eigenschaften eines Diensterbringers nicht darstellen. Darunter fällt zum Beispiel die die Reihenfolgeerhaltung bei der Datenübertragung.

Beispiel 5.1

Es sei der Wunsch eines Dienstbenutzers, die Nutzdaten x zu übertragen, durch das Dienstelement $DATreq(x)$ gekennzeichnet. Die Nutzdaten werden dem Kommunikationspartner durch das Dienstelement $DATind(x)$ zugestellt. Werden die Nutzdaten x_1 und x_2 durch einen reihenfolgeerhaltenden Diensterbringer übertragen, so sind die folgenden Abläufe möglich:

$$<DATreq(x_1), DATind(x_1), DATreq(x_2), DATind(x_2)>$$
$$<DATreq(x_1), DATreq(x_2), DATind(x_1), DATind(x_2)>$$

Wäre dies ausschließlich mit den Beschreibungstechniken TSD und LC spezifiziert, dann wäre auch die Spur

$$<DATreq(x_1), DATreq(x_2), DATind(x_2), DATind(x_1)>$$

in der Spurmenge enthalten.[1] Damit wäre der Diensterbringer nicht mehr reihenfolgeerhaltend, da die Nachricht x_2 vor der zuerst gesendeten Nachricht x_1 zugestellt werden würde. □

Zur Beschreibung dieses Verhaltens ist in den Normen eine Warteschlangendarstellung gewählt worden. In einem ersten Ansatz ist hierbei die Vorstellung einer FIFO Struktur vorhanden, in der alle Daten gespeichert werden. Der Inhalt dieser Warteschlangen spiegelt die dem Kommunikationspartner noch nicht zugestellten Daten wider. Da durch die

[1] Eine formale Begründung dieses Sachverhalts erfolgt im Kapitel 6 durch Satz 6.8.

Verwendung einer reinen FIFO Struktur die Beschreibung einer priorisierten Datenübertragung nicht möglich ist, wird das Warteschlangenmodell in den Normen erweitert. In diesem Kapitel werden die in den Normen angegebenen Beschreibungen ergänzt, sofern dies erforderlich ist, und formal fundiert.

Bei der Anwendung des Queuemodells wird, wie bei den lokalen Einschränkungen, die Richtung der Dienstelemente nicht substantiell unterschieden. Eine Unterscheidung von Eingabe- und Ausgabedienstelementen erfolgt nur implizit in einer anderen Norm [ISO87b, CCI88d] durch die Benennung der Dienstelemente, ebenso wie bei den lokalen Einschränkungen, siehe Kapitel 4. Damit wird bei den in den Normen beschriebenen Verfahren nicht nur das Komponentenverhalten, sondern zusätzlich das Umgebungsverhalten eingeschränkt. Aus diesem Grund wird äquivalent zur Formalisierung von TSDs im Kapitel 3 und von lokalen Einschränkungen im Kapitel 4 eine Zweiteilung der Semantik vorgenommen. Zuerst wird das Verhalten des Diensterbringers mit einer Einschränkung des Umgebungsverhaltens beschrieben. Hierbei werden nur Sequenzen von erlaubten Aktionen ohne Unterscheidung der Ein- und Ausgabeaktionen betrachtet. Als zweiter Schritt werden die dadurch erhaltenen Spurmengen äquivalent zu dem in den Kapiteln 3.4 und 4.4 beschriebenen Verfahren dahingehend erweitert, daß die Einschränkung des Umgebungsverhaltens entfällt.

5.1 Informelle Darstellung des Queuemodells

Das Queuemodell wird in keiner Norm explizit beschrieben, sondern nur als Beschreibungstechnik bei der Spezifikation der Diensterbringer angewendet. In [JA90] erfolgt zwar eine methodische Zusammenfassung der wichtigsten Eigenschaften, da [JA90] aber keine offiziell autorisierte Sicht des Queuemodells darstellt, stützen sich die folgenden Erklärungen auf die Anwendungen in den Normen.

Eine Spezifikation in den Normen, die diese Methode verwendet, beinhaltet immer eine zur Abbildung 5.1 äquivalente Darstellung.

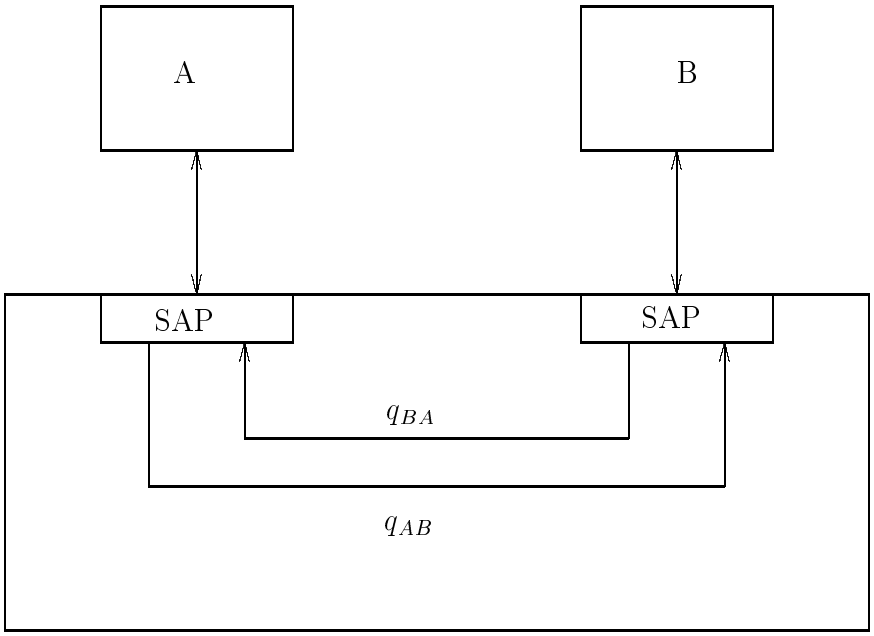

Abbildung 5.1: Queuemodell

In Abbildung 5.1 wird ein Queuemodell für eine Vollduplex-Verbindung skizziert, da zwei entgegengerichtete Warteschlangen für eine Verbindung verwendet werden. Die einzige formale Aussage, die aus Abbildung 5.1 extrahierbar ist, ist die Verwendung zweier Warteschlangen zum Datenaustausch. In diese Warteschlangen werden Objekte eingetragen, die gemäß der FIFO-Strategie abgearbeitet werden. Ein Objekt[2] stellt ein Element in der Warteschlange dar, dessen Parameter der eines entsprechenden Dienstelements ist. So wird zum Beispiel das Dienstelement $DATind(x)$, das das Übermitteln der Nutzdaten x an einen Dienstbenutzer beschreibt, durch das Datenobjekt $DO(x)$ in der Warteschlange dargestellt.

In textuellen Erklärungen folgen einige Hinweise über das Eintragen und Entfernen von Objekten. Bei allen Dienstnormen ist festgelegt, daß die Warteschlangen eine endliche Kapazität besitzen. Die in einer Implementierung tatsächliche Größe einer Warteschlange wird gegenüber den Dienstbenutzern verborgen gehalten.

In den Normen wird festgelegt, daß die meisten Objekte von den jeweiligen Dienstbenutzern in die Warteschlangen eingetragen werden. Darunter fallen Verbindungsaufbau-, Daten-, Synchronisations- und Verbindungsabbauobjekte. Nur bei Aktionen, die der Diensterbringer von sich aus bewirken kann, wie zum Beispiel bei einem diensterbringerorientierten Verbindungsabbruch, ist es ihm möglich, Objekte in die Warteschlangen einzutragen. Als Beleg hierfür dient die Spezifikation der Transportschicht[ISO84c, CCI88h], in der die hauptsächliche Aktivität den Dienstbenutzern übergeben wird:

The objects which may be placed in a queue by a TS user (see §§12, 13 and

[2]Obwohl der Begriff Objekt wegen seiner schillernden Verwendung nur mit erhöhter Vorsicht bei einer Dissertation in Informatik zu benutzen ist, wird er in dieser Arbeit dennoch verwendet, da er in den Normen explizit in diesem Zusammenhang vorkommt.

14) are:

a) connect objects ...

...

The only objects which can be placed in a queue by the TS provider are disconnect objects (representing $T - DISCONNECT$ primitives and their parameters).

Sicherlich ist damit gemeint, daß der jeweilige Dienstbenutzer Objekte bereitstellt, die unter der Kontrolle des Diensterbringers in die Warteschlangen eingetragen werden. Dadurch wäre sichergestellt, daß die Kontrolle der Aktivität beim Diensterbringer bliebe. Dies wäre schon wegen des Modularisierungskonzepts und der Datenkapselung unabdingbar. Wie sollte ein Dienstbenutzer Objekte in eine Warteschlange eintragen, die sich in einer Komponente befindet, auf die er nur über die Schnittstellen, das heißt die SAPs, zugreifen kann. Insbesondere wird die Verletzung der Datenkapselung daran erkennbar, daß die Warteschlangen eine begrenzte Kapazität haben. Somit müßte ein Dienstbenutzer jederzeit Kenntnis vom Füllungsstand der Warteschlangen haben, wenn die Aktivität bei ihm läge, da der Diensterbringer die Annahme nicht verweigern kann, falls er keine Aktivitätskontrolle besitzt. Für die Überprüfung des Füllungsstands der Warteschlangen sind jedoch keinerlei Dienstelemente vorgesehen. Oben genannte Annahme wird ebenso beim Entfernen von Objekten aus den Warteschlangen getroffen.

In [JA90] werden die Operationen etwas deutlicher als in den Normen, aber dennoch sehr vage unterschieden:

- Schreiboperationen:
  - Dienstbenutzer: Die Dienstelemente, die auf *request* und *response* enden, bewirken den Eintrag eines Objekts in die Warteschlange.
  - Diensterbringer: Manche der Dienstelemente, die auf *indication* enden, bewirken den Eintrag eines Objekts in die Warteschlange.
- Leseoperationen:
  - Die Dienstelemente, die auf *indication* und $confirm$ enden, bewirken das Löschen eines Objekts in der Warteschlange.

In den Normen erfolgt noch der Hinweis, daß ein Verbindungsabbruchobjekt unter Umständen zum Löschen des unmittelbaren Vorgängerelements in einer Warteschlange führen kann. Zur Veranschaulichung werden hierfür die sogenannten Präzedenztabellen verwendet, die zusätzlich das Überholen von Elementen beschreiben. Damit wird eine nichtdeterministische Eigenschaft der Warteschlangen festgelegt, da ein Löschvorgang beziehungsweise ein Überholvorgang nicht zwingenderweise erfolgen muß. Mit der Überholeigenschaft von Elementen kann außerdem eine priorisierte Datenübertragung beschrieben werden. Dazu wird in der Präzedenztabelle festgelegt, daß ein priorisiertes Datenobjekt ein normales Datenobjekt überholen kann.

In den Dienstbeschreibungen wird auch auf die Rahmenbedingungen der Dienstelemente eingegangen. So ist zum Beispiel ein Verbindungsaufbauobjekt bei einigen Schichten nur dann erlaubt, wenn die Warteschlangen leer sind.

5.2 Formale Semantik mit Einschränkung des Umgebungsverhaltens

Es wird eine detailliertere und schematischere Beschreibung als die in den Normen verwendete eingeführt, mit deren Hilfe eine exakte und universelle Spezifikation des Queuemodells der verschiedenen Schichten möglich ist. Ausgehend von dieser Beschreibung wird dann eine formale Semantik definiert.

5.2.1 Konzepte der Semantik

In den Normen erfolgt nur teilweise eine Zuordnung von Dienstelementen zu Objekten und Operationen auf Warteschlangen in textueller Form. Zur exakten Beschreibung dieses Sachverhalts wird in dieser Arbeit das Konzept der *Objekttabellen* neu eingeführt. Die jeweiligen SAPs werden durch das Anfügen eines Indizes an die Dienstelemente gekennzeichnet. Es gibt die Operationen *add*, *remove* und *none*. Die Warteschlange vom SAP A zum SAP B wird mit dem Index $_{AB}$ beschrieben.

Dienstelement	Objekt	Operation	Bedingung
CONNECTrequest$_A$	CO	add$_{AB}$	isempty(q_{AB})
DATAindication$_B$	DO	remove$_{AB}$	
Dummy		none	

Tabelle 5.1: Objekttabelle

In der Tabelle 5.1 wird der Zusammenhang zwischen Dienstelementen und Operationen, die in drei Kategorien eingeteilt werden, festgelegt. Die erste Kategorie von Operationen beschreibt das Einfügen von Objekten in eine Warteschlange. So besteht wegen des Eintrags von add_{AB} in Tabelle 5.1 ein Zusammenhang zwischen dem Dienstelement $CONNNECTrequest_A$ und dem Eintragen eines Verbindungsaufbauobjekts CO in die Warteschlange q_{AB}. Hierbei ist sicherzustellen, daß die Warteschlange q_{AB} leer ist. In der Bedingung kann auf die aktuelle Warteschlange mit dem Identifikator q_{AB} oder q_{BA}, sowie auf die gegebenenfalls modifizierte Warteschlange mit dem Identifikator q'_{AB} oder q'_{BA} Bezug genommen werden. Die zweite Kategorie sind Operationen, die Elemente aus der Warteschlange entfernen können. So ist gemäß obiger Tabelle wegen des Eintrags $remove_{AB}$ ein $DATAindication_B$ nur dann möglich, wenn sich ein Datenobjekt DO in der Warteschlange q_{AB} an geeigneter Position befindet. Unter die dritte Kategorie fallen Elemente, die keine Operation bewirken und somit die Warteschlange unverändert lassen. In der Objekttabelle wird dies durch den Eintrag *none* erkennbar.

Ein weiterer Bestandteil der Beschreibung ist die in den Normen verwendete *Präzedenztabelle*, in der die Präzedenzen der Objekte definiert werden:

	CO	DO	EXP	DIS
CO	N/A	–	–	DES
DO	N/A	–	A/A	DES
EXP	N/A	–	–	DES
DIS	N/A	–	–	DES

Tabelle 5.2: Präzedenztabelle

In Tabelle 5.2 steht CO für ein Verbindungsaufbauobjekt, DO für ein Datenobjekt EXP für ein priorisiertes Datenobjekt, und DIS für ein Verbindungsabbauobjekt. Ein A/A (advance ahead) bedeutet, daß das Objekt EXP ein Objekt DO überholen kann. Für einen korrekt implementierten Diensterbringer ist es jedoch nach den Dienstnormen nicht erforderlich, daß ein derartiges Überholen stattfindet, da es nicht explizit gefordert wird. Dies führt zum bereits erwähnten Nichtdeterminismus. Daß ein Objekt durch ein anderes nicht überholt, beziehungsweise gelöscht werden kann, wird durch den Eintrag – gekennzeichnet. Der Eintrag DES beschreibt das mögliche Löschen eines Vorgängerelements, wobei das den Löschvorgang auslösende Objekt erhalten bleibt. Im Gegensatz dazu bedeutet ein $DESboth$, das nicht in obiger Tabelle vorkommt, das mögliche Löschen von beiden Elementen. Der Eintrag $DESboth$ wird in den Normen nicht verwendet, ist aber in der Sicherungsschicht und der Vermittlungsschicht notwendig und deswegen in dieser Arbeit eingeführt worden. So kann nach obiger Tabelle nur das Objekt DIS ein beliebiges Vorgängerelement löschen, wobei es selbst erhalten bleibt. Der Eintrag N/A (not available) beschreibt, daß eine derartige Abfolge in den Dienstbeschreibungen nicht vorgesehen ist. Da dies meist als eine Folge der TSDs und LCs schon ausgeschlossen ist, wird dieser Eintrag im weiteren nicht behandelt.

5.2.2 Formale Semantik der Präzedenzqueues

Als grundlegende Datenstruktur wird eine Warteschlange verwendet. Die Präzedenztabelle wird als ein Tripel von Relationen auf Objekten $(Prec, Destr, DestrBoth)$ formalisiert. So bedeutet $(obj_1, obj_2) \in Prec$, daß das Objekt obj_1 eine Präzedenz über das Objekt obj_2 hat, somit kann obj_1 ein unmittelbar vorausgehendes obj_2 überholen. Die zweite Relation $Destr$, die die Löscheigenschaft eines Vorgängerelements beschreibt, ist ebenfalls aus der Präzedenztabelle ablesbar, ebenso die Relation $DestrBoth$.

Der Nichtdeterminismus beim Zugriff auf die Warteschlangen wird durch die Einführung von *Reduktionsmengen* bezüglich der Relationen $(Prec, Destr, DestrBoth)$ modelliert. Hierbei beschreibt $[q]_{(Prec,Destr,DestrBoth)}$ die Menge aller Warteschlangen, die aus q hervorgegangen sind, wenn ein beliebiges Überholen oder Löschen von Objekten gemäß der Relationen $(Prec, Destr, DestrBoth)$ erfolgt ist.

Definition 5.1

Für eine Warteschlange q und die Relationen über Objekte[3] $Prec, Destr, DestrBoth$

[3]Die Objekte sind hierbei die Elemente, die in die Warteschlangen eingetragen werden. Sie werden

ist die *Reduktionsmenge* $[q]_{(Prec,Destr,DestrBoth)}$ *für eine Warteschlange* wie folgt definiert:

$$\begin{aligned} q_1\&a(x)\&b(y)\&q_2 \in [q_1\&b(y)\&a(x)\&q_2]_{(Prec,Destr,DestrBoth)} &\Leftrightarrow (a,b) \in Prec \\ q_1\&<a(x)> \in [q_1\&<b(y),a(x)>]_{(Prec,Destr,DestrBoth)} &\Leftrightarrow (a,b) \in Destr \\ q_1 \in [q_1\&<b(y),a(x)>]_{(Prec,Destr,DestrBoth)} &\Leftrightarrow (a,b) \in DestrBoth \end{aligned}$$

Zur Abkürzung wird bei dem Rest der Definition $Rels = (Prec, Destr, DestrBoth)$ gesetzt:

$$\begin{aligned} q &\in [q]_{Rels} \\ q \in [q']_{Rels} \wedge q' \in [q'']_{Rels} &\Rightarrow q \in [q'']_{Rels} \\ \{<a>\} &= [<a>]_{Rels} \\ \{\epsilon\} &= [\epsilon]_{Rels} \end{aligned}$$

□

Mit dieser Definition wird festgelegt, daß durch die Relation $Prec$ ein Vertauschen der Reihenfolge von Elementen möglich ist. Ein Löschen von Elementen gemäß der Relation $Destr$ beschreibt das Entfernen eines Vorgängerelements. Bei $DestrBoth$ werden beide Elemente gelöscht. Das Entfernen von Elementen durch eine Löschoperation ist somit immer nur vom Ende einer Warteschlange möglich. Mit dieser Definition ist dadurch bei Warteschlangen mit unendlicher Länge keine Löschaktion möglich, was sinnvoll ist. Derartige Warteschlangen sind im Queuemodell sowieso ausgeschlossen.

Das Lesen einer Nachricht geschieht nach dem FIFO Prinzip vom Anfang der Warteschlange aus. Das explizite Löschen durch Löschobjekte ist im Gegensatz dazu nur am Ende einer Warteschlange möglich. Der Aufbau der Reduktionsmengen wird an folgendem Beispiel erläutert:

Beispiel 5.2

In dieser vereinfachten Warteschlange sei das Objekt $DAT(x)$ für eine normale Datenübertragung, $EXP(x)$ für ein priorisiertes Datenobjekt festgelegt. Zur Darstellung der Priorisierung wird die Priorisierungsrelation $P := \{(EXP, DAT)\}$ definiert. Es wird im folgenden die Reduktionsmenge für die Warteschlange $<DAT(x_1), DAT(x_2), EXP(x_3)>$ bestimmt. In dieser Warteschlange folgt den normalen Datenelementen x_1 und x_2 das priorisierte Datenelement x_3. Nach der obigen Definition gilt für die Reduktionsmenge:

$$\begin{aligned} [<DAT(x_1),DAT(x_2),EXP(x_3)>]_{(P,\emptyset,\emptyset)} = \{&<DAT(x_1),DAT(x_2),EXP(x_3)>, \\ &<DAT(x_1),EXP(x_3),DAT(x_2)>, \\ &<EXP(x_3),DAT(x_1),DAT(x_2)>\} \end{aligned}$$

durch die Objekt-, Präzedenz- und Destruktionstabellen festgelegt.

Wird nun ein Löschelement DIS hinzugenommen, sowie die Destruktionsrelation $D = \{(DIS, DAT), (DIS, EXP)\}$ So gilt:

$$\begin{aligned}[<DAT(x_1), DAT(x_2), DIS(x_4)>]_{(P,D,\emptyset)} &= \{<DAT(x_1), DAT(x_2), DIS(x_4)>, \\ &\quad <DAT(x_1), DIS(x_4)>, \\ &\quad <DIS(x_4)>\}\end{aligned}$$

Wird jetzt die Destruktionsrelation als Relation zur Beschreibung, daß beide Elemente gelöscht werden, verwendet, dies wird durch das Tupel $(P, \emptyset, D)$ dargestellt, dann erhält man:

$$\begin{aligned}[<DAT(x_1), DAT(x_2), DIS(x_4)>]_{(P,\emptyset,D)} &= \{<DAT(x_1), DAT(x_2), DIS(x_4)>, \\ &\quad <DAT(x_1)>\}\end{aligned}$$

□

Da bei jeder Verbindung zwei Warteschlangen vorhanden sind, wird die Definition auf Tupel von Warteschlangen kanonisch unter Einbeziehung von spontan erzeugten Objekten erweitert.

Definition 5.2

Für ein Tupel von Warteschlangen (q_{AB}, q_{BA}) ist die *Reduktionsmenge* $[(q_{AB}, q_{BA})]_{(Prec,Destr,DestrBoth,SO)}$ definiert als:

$$\begin{aligned}&(q'_{AB}, q'_{BA}) \in [(q_{AB}, q_{BA})]_{(Prec,Destr,DestrBoth,SO)} = \\ &\quad (q'_{AB} \in [q_{AB}]_{(Prec,Destr,DestrBoth)} \wedge q'_{BA} \in [q_{BA}]_{(Prec,Destr,DestrBoth)}) \\ &\quad \vee \exists s \in SO.\ (q'_{AB} \& s = q_{AB} \wedge q'_{BA} \& s = q_{BA})\end{aligned}$$

□

Die Reduktionsmenge eines Tupels von Warteschlangen setzt sich aus den Reduktionsmengen der Warteschlangen des Tupels zusammen. Hierbei ist das Anfügen von spontan erzeugten Objektsequenzen möglich, deren Menge mit SO bezeichnet wird. Mit den spontanen Objektsequenzen ist die Modellierung eines selbstständigen Agierens eines Diensterbringers zum Beispiel bei einem Verbindungsabbruch möglich. Wie in den Beschreibungen der Normen wird hierbei eine spontan erzeugte Objektsequenz an beide Warteschlangen gleichzeitig angefügt.

Die Relationen $Prec$, $Destr$, $DestrBoth$ werden durch einfaches Ablesen der in den Normen angegebenen Präzedenztabelle erhalten. Das Tabellenelement A/A bewirkt einen Eintrag in die Relation $Prec$, DES in $Destr$, sowie $DESboth$ in $DestrBoth$.

5.2.3 Formalisierung der Objekttabelle

Ein weiterer Bestandteil des Queuemodells ist die in dieser Arbeit eingeführte Objekttabelle, mit der die Zuordnung von Dienstelementen zu Objekten und Operationen auf

die Warteschlangen beschrieben wird. Die Objekttabelle wird als eine Menge von Objektdefinitionen formalisiert. Die Tabelle 5.1 auf Seite 107 wird durch die Menge $ObjTab$ beschrieben, die wie folgt festgelegt ist:

$$\begin{aligned} ObjTab = \quad & \{add(CONNECTrequest_A, CO, AB, \lambda(q_A, q_B).isempty(q_A)), \\ & remove(DATAindication_B, DO, AB, \lambda(q_A, q_B).true), \\ & none(Dummy)\} \end{aligned}$$

Es stehen hierbei die Konstruktoren *add*, *remove* und *none* zur Verfügung, mit denen die Wirkung der Dienstelemente auf die Warteschlangen klassifiziert wird. Die zu der Operation zugehörige Warteschlange wird durch die Angabe des Indizes $_{AB}$ oder $_{BA}$ gekennzeichnet. Das Vorhandensein eines Parameters wird bei den Dienstelementen und Objekten zur Vereinfachung immer implizit angenommen und nicht gesondert in der Objekttabelle angegeben. Mit der Menge $ObjTab$ ist somit eine Zuordnung von Dienstelementen zu Objekten, Warteschlangen, Aktionen und Bedingungen definiert. Diese Zuordnung muß nicht notwendigerweise eindeutig sein, da ein Dienstelement unter Umständen auch mehrfach auftreten kann.

5.2.4 Formalisierung des Einhaltens des Queuemodells

Ausgehend von Präzedenz- und Objekttabelle wird die Korrektheit einer Aktionsfolge bezüglich des Queuemodells bewertet. Hierfür wird zu jedem Zeitpunkt der Zustand beider Warteschlangen betrachtet.

Definition 5.3

Ein *Queuemodell* QM ist ein Quintupel (P, D, Db, SO, O), wobei P eine Präzedenzrelation, D und Db eine Destruktionsrelation, SO die Menge der spontanen Aktionssequenzen und O eine Objekttabellenmenge ist. □

Definition 5.4

Die *von dem Queuemodell QM akzeptierte Spurmenge* wird durch das Prädikat $IsQM_{[QM]}(t)$ beschrieben:

$$\begin{aligned} IsQM_{[QM]}(t) = \exists qs, max \neq \infty. \\ \forall m.\ m < \#t \\ \Rightarrow (\#qs[m] < max \\ \wedge\ IsCorrectTransition_{[QM]}(qs[m], t[m], qs[m+1])) \end{aligned}$$

Hierbei ist qs eine Spur von Warteschlangentupeln, die eine mögliche Zustandsfolge beider Warteschlangen eines Diensterbringers widerspiegelt. Mit der natürlichen Zahl max wird die in den Normen geforderte Eigenschaft modelliert, daß die Länge

der Warteschlange beschränkt ist. Bei obiger Definition wird zur Vereinfachung angenommen, daß die Anwendung der Längenfunktion auf ein Tupel das Maximum der Längen der Komponenten des Tupels bestimmt.

Das Prädikat $IsCorrectTransition_{[QM]}(q, act, q')$ überprüft, ob ein Übergang von q nach q' durch die von dem Dienstelement act ausgelöste Operation gemäß der Angaben des Queuemodells $QM = (P, D, Db, SO, O)$ möglich ist.

$$IsCorrectTransition_{[QM]}((q_{AB}, q_{BA}), act(x), (q'_{AB}, q'_{BA})) =$$

$$\begin{aligned}
&\exists add(act, obj, where, cond) \in O. \\
&\quad cond(q_{AB}, q_{BA}) \\
&\quad \wedge\ (where = AB \Rightarrow (q'_{AB}, q'_{BA}) \in [(q_{AB} \circ <obj(x)>, q_{BA})]_{(P,D,Db,SO)}) \\
&\quad \wedge\ (where = BA \Rightarrow (q'_{AB}, q'_{BA}) \in [(q_{AB}, q_{BA} \circ <obj(x)>)]_{(P,D,Db,SO)}) \\
&\vee\ \exists remove(act, obj, where, cond) \in O. \\
&\quad cond(q_{AB}, q_{BA}) \\
&\quad \wedge\ (where = AB \Rightarrow (obj(x) \& q'_{AB}, q'_{BA}) \in [(q_{AB}, q_{BA})]_{(P,D,Db,SO)}) \\
&\quad \wedge\ (where = BA \Rightarrow (q'_{AB}, obj(x) \& q'_{BA}) \in [(q_{AB}, q_{BA})]_{(P,D,Db,SO)}) \\
&\vee\ \exists none(act) \in O. \\
&\quad (q'_{AB}, q'_{BA}) \in [(q_{AB}, q_{BA})]_{(P,D,Db,SO)}
\end{aligned}$$

□

Bei der Definition von $IsCorrectTransition_{[QM]}$ wird ein Tupel von Warteschlangen betrachtet, da das Erzeugen von spontanen Elementen, siehe dazu Definition 5.2, nur bei beiden Warteschlangen gleichzeitig vorgenommen wird. Bei einem Dienstelement, das eine Einfügeoperation in eine Warteschlange verursacht – das ist an dem Konstruktor *add* in der Objekttabelle O erkennbar – wird das zugehörige Objekt an das Ende der betreffenden Warteschlange, die mit *where* genauer bezeichnet wird, angefügt. Das dabei entstehende Warteschlangentupel ist aus der Reduktionsmenge, die in der Definition 5.2 festgelegt ist, zu wählen. Verursacht ein Dienstelement wegen der Angabe von *remove* das Auslesen eines Elements, so ist zuerst zu prüfen, ob in der Reduktionsmenge des betreffenden Warteschlangentupels eine Warteschlange mit dem zugehörigen Objekt an erster Stelle enthalten ist. Der Rest dieser Warteschlange beschreibt dann den Neuzustand dieser Komponente des Tupels. Bei einem Dienstelement, das keine Operation erfordert – dies ist an dem Konstruktor *none* zu erkennen – kann das Tupel der Warteschlangen aus der Reduktionsmenge gewählt werden. Lassen sich derartige Operationen nicht realisieren, so ist das betreffende Dienstelement nicht konform mit dem angegebenen Queuemodell.

Definition 5.5

Die *Semantik eines Queuemodells* QM wird definiert als:

$$[\![QM]\!]_{\mathcal{QM}} := \{t | IsQM_{[QM]}(t)\}$$

□

Die Semantik eines Queuemodells QM wird als die Menge aller Spuren bezeichnet, bei denen $IsQM_{[QM]}$ erfüllt ist.

5.3 Eigenschaften des formalisierten Queuemodells

Ein Merkmal des Queuemodells ist, daß es eine präfixabgeschlossene Menge von Spuren beschreibt.

Satz 5.1

Für ein Queuemodell QM' beschreibt $[\![QM']\!]_{\mathcal{QM}}$ eine präfixabgeschlossene Menge:

$$[\![QM']\!]_{\mathcal{QM}} = PRE([\![QM']\!]_{\mathcal{QM}})$$

□

Beweis:

1) $PRE([\![QM']\!]_{\mathcal{QM}}) \subseteq [\![QM']\!]_{\mathcal{QM}}$ gilt trivialerweise
2) $[\![QM']\!]_{\mathcal{QM}} \subseteq PRE([\![QM']\!]_{\mathcal{QM}})$
Die Aussage folgt direkt aus der Definition 5.4.

□

Die Präfixabgeschlossenheit des Queuemodells ist eine Folge davon, daß in fast allen Dienstnormen keine Lebendigkeitseigenschaft des Queuemodells gefordert wird. Eine explizite Lebendigkeitseigenschaft wird nur in der Sicherungsschicht bei der Übertragung eines Löschobjekts erwartet. In [ISO88b, CCI88f] wird festgelegt, daß Objekte, die in einer Warteschlange nicht regulär ausgegeben werden, nach Ablauf einer nicht näher spezifizierten Zeitspanne durch das Eintragen eines Löschobjekts entfernt werden. Diese Modellierung ist im Queuemodell nicht notwendig, da dieser Aspekt schon vom Lebendigkeitsanteil der TSDs erfaßt wird.

Beispiel 5.3

Mit einem Queuemodell wird im allgemeinen keine reine Sicherheitseigenschaft beschrieben. Als Beispiel hierfür dient das durch die Objekttabelle 5.1 auf Seite 107 festgelegte Queuemodell QM'. Es gilt:

$$CONNECTrequest_A(x)^\infty \notin [\![QM']\!]_{\mathcal{QM}}$$

Bei einer Eingabe von unendlichen vielen $CONNECTrequest$ ist es erforderlich, daß die Länge einer Warteschlange unbegrenzt ist. Da eine endliche Länge der Warteschlangen in Definition 5.4 und den Normen gefordert wird, ist diese Spur nicht in $[\![QM']\!]_{\mathcal{QM}}$ enthalten, obwohl alle endlichen Präfixe dieser Spur in $[\![QM']\!]_{\mathcal{QM}}$ enthalten sind. Somit stellt ein Queuemodell im allgemeinen keine reine Sicherheitseigenschaft dar. □

5.4 Formale Semantik ohne Einschränkung des Umgebungsverhaltens

Bei der im Kapitel 5.2 vorgestellten Semantik des Queuemodells wird das Umgebungsverhalten eingeschränkt. So ist in einigen Normen ein Verbindungsaufbau mit der Eingabeaktion $CONNECTrequest$ nur bei einer leeren Warteschlange erlaubt. Von der vorgestellten Semantik wird ein $CONNECTrequest$ bei einer nichtleeren Warteschlange nicht zugelassen. Somit wird das Umgebungsverhalten eingeschränkt. Eine Semantik des Queuemodells ohne Beeinflussung des Umgebungsverhaltens erhält man äquivalent zu der im Kapitel 4.4 für die lokalen Einschränkungen beschriebenen Methode. Da das Queuemodell nach Satz 5.1 eine bezüglich Präfixen abgeschlossene Eigenschaft beschreibt, ist die Einbeziehung von fehlenden Eingaben nach Satz 3.9 nicht notwendig. Für die Semantik des Queuemodells QM bei freiem Umgebungsverhalten ergibt sich:

Definition 5.6

Die *Semantik eines Queuemodells QM ohne Einschränkung des Umgebungsverhaltens* wird definiert als:

$$[\![QM]\!]_{\mathcal{QM}}^{(I,O)} := [\![QM]\!]_{\mathcal{QM}} \cup E_{false_inp}([\![QM]\!]_{\mathcal{QM}}, I, O)$$

Hierbei ist E_{false_inp} wie in Definition 3.8 festgelegt. □

Mit der Definition 5.6 wird eine jederzeitige Aktion der Dienstbenutzer ermöglicht.

Satz 5.2

Für das Queuemodell QM beschreibt $[\![QM]\!]_{\mathcal{QM}}^{(I,O)}$ eine bezüglich Präfixen abgeschlossene Eigenschaft. □

Beweis:

Die Aussage folgt aus Satz 5.1 und wird äquivalent zum Beweis des Lemmas 4.5 von Seite 91, Teil 2) durchgeführt. □

5.5 Anwendung der Formalisierungsmethodik auf die Dienstnormen

In diesem Kapitel wird die zuvor beschriebene Methode auf die Schichten, die das Queuemodell in den Normen benutzen, angewendet. Hierbei wird inbesondere auf die Objekttabellen eingegangen, da diese in den Normen nicht existieren, sondern nur textuell beschrieben werden. Zur Vereinheitlichung des Konzepts ist in dieser Arbeit davon ausgegangen worden, daß jedes Dienstelement mit einem Parameter versehen ist. Diese Annahme bedeutet keine wesentliche Einschränkung, da die meisten Dienstelemente Parameter besitzen. Für eine exakte Behandlung von Dienstelementen sowohl mit als auch ohne

Parameter wäre eine Unterscheidung innerhalb der Objekt-, Präzedenz- und Destruktionstabellen denkbar. Durch diese Vorgehensweise werden jedoch die Tabellen aufwendiger, da beide Fälle zu berücksichtigten sind. Um dies zu vermeiden, werden in dieser Arbeit alle Dienstelemente mit Parameter versehen und, falls notwendig, zu Dienstelementen ohne Parameter abstrahiert. Die Angabe einer derartigen Abstraktionsfunktion ist sehr einfach, da nur die jeweiligen Parameter nicht betrachtet werden. Mit dieser Vorgehensweise läßt sich der Sachverhalt eindeutig modellieren und die Darstellung der Tabellen ist dennoch einfach.

5.5.1 Das Queuemodell der Bitübertragungsschicht

In der Bitübertragungsschicht [ISO88c, CCI88e] ist die Beschreibung des Queuemodells sehr einfach gehalten. Es wird festgelegt, daß Bitströme zur Verbindung verwendet werden, die ausschließlich Protokolldateneinheiten der Bitübertragungsschicht transportieren. Desweiteren wird festgelegt, daß die Daten in den Bitströmen reihenfolgeerhaltend transportiert werden. Aus diesen minimalen Angaben wird in dieser Arbeit das passende Queuemodell entwickelt. Die Bitströme entsprechen wegen der geforderten FIFO Struktur den Warteschlangen.

Dienstelement	Objekt	Operation	Bedingung
Ph_ACTIVATErequest$_A$		none	
Ph_ACTIVATEindication$_B$		none	
Ph_DATArequest$_A$	DO	add$_{AB}$	
Ph_DATAindication$_B$	DO	remove$_{AB}$	
Ph_DEACTIVATErequest$_A$		none	
Ph_DEACTIVATEindication$_B$		none	
Ph_ACTIVATErequest$_B$		none	
Ph_ACTIVATEindication$_A$		none	
Ph_DATArequest$_B$	DO	add$_{BA}$	
Ph_DATAindication$_A$	DO	remove$_{BA}$	
Ph_DEACTIVATErequest$_B$		none	
Ph_DEACTIVATEindication$_A$		none	

Tabelle 5.3: Objekttabelle der Bitübertragungsschicht

In der Objekttabelle 5.3 wird festgelegt, daß nur die Datenübertragungsdienstelemente eine Wirkung auf die Warteschlangen haben. Die Menge, die diese Objekttabelle beschreibt, wird als PhO bezeichnet. Da keine Präzedenztabelle vorhanden ist, wird für die Präzedenzrelation und die Destruktionsrelation die leere Menge verwendet. Da keine spontane Aktion erfolgen kann, wird die Menge der spontanen Aktionssequenzen als leere Menge definiert. Somit beschreibt $[\![\emptyset, \emptyset, \emptyset, \emptyset, PhO]\!]^{(I,O)}_{\mathcal{QM}}$ die Spurmenge des Queuemodells der Bitübertragungsschicht.

Bei obiger Formalisierung ist ebenso wie in [ISO88c, CCI88e] nicht auf einige Grundeigenschaften eingegangen worden. So ist zum Beispiel nicht festgelegt worden, daß eine Datenübertragung nur bei einer bestehenden Verbindung erfolgen kann. Diese Eigenschaft folgt aber schon aus den lokalen Einschränkungen, die im Kapitel 4.5.1 beschrieben werden. Da ein Hinweis fehlt, daß die jeweiligen Warteschlangen zum Zeitpunkt des Verbindungsaufbaus leer sein müssen, entspricht es der Dienstspezifikation, daß Daten der vorhergehenden Verbindung in einer späteren Verbindung übertragen werden können. Eine Aussage, was mit den Daten bei einem diensterbringerorientierten Verbindungsabbau geschehen soll, ist ebenfalls nicht vorhanden. Diese Daten müßten in einer späteren Verbindung übertragen werden. Dies ist aber nicht bei allen Verbindungen erwünscht. Obige Definition entspricht aus diesem Grund nur dann der Intuition, wenn jeweils höchstens eine Verbindung betrachtet wird und keine Sequenz von Verbindungen. Da dies in [ISO88c, CCI88e] nahegelegt wird, ist vermutlich die obige Vereinfachung getroffen worden. Eine Variante wäre eine Beschreibung des Queuemodells in einem höheren Detaillierungsgrad unter Einbeziehung von Verbindungsaufbau- und Verbindungsabbauobjekten. Da dies in [ISO88c, CCI88e] nicht erfolgt ist, unterbleibt es in dieser Arbeit. Eine derartige Spezifikation ist aber problemlos zu erstellen, wie dies an den folgenden Queuemodellen der anderen Schichten zu erkennen ist.

5.5.2 Das Queuemodell der Sicherungsschicht

In der Definition der Sicherungsschicht [ISO88b, CCI88f] erfolgt eine wesentlich genauere Beschreibung des Queuemodells als bei der Bitübertragungsschicht. Es werden Verbindungsaufbau-, Daten-, Reset-, Synchronisations- und Verbindungsabbauobjekte eingeführt. Ein Verbindungsaufbauobjekt wird als Folge eines $DL_CONNECTrequest$ oder eines $DL_CONNECTresponse$ in die Warteschlange eingetragen. Solange dieses Objekt nicht aus der Warteschlange entfernt ist, darf kein anderes Objekt als ein Verbindungsabbauobjekt in die Warteschlange eingetragen werden. Es wird in [ISO88b, CCI88f] nicht explizit erwähnt, daß die Warteschlangen beim Verbindungsaufbau leer sein müssen. Daraus ergibt sich die folgende Objekttabelle:

Dienstelement	Objekt	Operation	Bedingung
DL_CONNECTrequest$_A$	CO	add$_{AB}$	Max1ConObj
DL_CONNECTindication$_B$	CO	remove$_{AB}$	
DL_CONNECTresponse$_B$	CO	add$_{BA}$	Max1ConObj
DL_CONNECTconfirm$_A$	CO	remove$_{BA}$	
DL_CONNECTrequest$_B$	CO	add$_{BA}$	Max1ConObj
DL_CONNECTconfirm$_B$	CO	remove$_{AB}$	

Tabelle 5.4: Objekttabelle des Verbindungsaufbaus der Sicherungsschicht

In der Tabelle 5.4 ist der Teil der Objekttabelle der Sicherungsschicht dargestellt, der den Verbindungsaufbau durch die Angabe der Operationen mit dem Verbindungsaufbauobjekt CO beschreibt. Das Prädikat $Max1ConObj$ stellt sicher, daß höchstens ein

Verbindungsaufbauobjekt in den Warteschlangen enthalten ist:[4]

$$Max1ConObj'(q_{AB}, q_{BA}) = (\#CO©q_{AB} + \#CO©q_{BA} \leq 1)$$

Bei dieser Definition könnten sich jedoch bereits Datenelemente in den Warteschlangen befinden. Dies ist in den Normen nicht explizit ausgeschlossen, aber es ist dennoch unerwünscht. Somit ist die folgende Definition sinnvoller:

$$Max1ConObj(q_{AB}, q_{BA}) = ((q_{AB}, q_{BA}) = (<CO>, \epsilon) \vee (q_{AB}, q_{BA}) = (\epsilon, <CO>))$$

Da gemäß der Definition der Sicherungsschicht ein simultaner Verbindungsaufbau von beiden Dienstbenutzern möglich ist, ist die Bedingung, daß bei einem $DL_CONNECTrequest$ die Warteschlangen keine Verbindungsaufbauobjekte enthalten, zu restriktiv. Es ist somit nur sicherzustellen, daß maximal ein Verbindungsaufbauobjekt in den Warteschlangen enthalten ist. Für den Fall eines simultanen Verbindungsaufbaus sind die Dienstelemente $DL_CONNECTrequest_B$ und $DL_CONNECTconfirm_B$ auch in der Tabelle erfaßt worden.

Dienstelement	Objekt	Operation	Bedingung
DL_DATArequest$_A$	DO	add$_{AB}$	NoConObj
DL_DATAindication$_B$	DO	remove$_{AB}$	
DL_DATArequest$_B$	DO	add$_{BA}$	NoConObj
DL_DATAindication$_A$	DO	remove$_{BA}$	

Tabelle 5.5: Objekttabelle der Datenübertragung der Sicherungsschicht

In der Tabelle 5.5 wird der Teil der Objekttabelle angegeben, der für die Datenübertragung verwendet wird. Hierbei stellt $NoConObj$ sicher, daß kein Datenobjekt während eines noch offenen Verbindungsaufbaus übertragen wird:

$$NoConObj(q_{AB}, q_{BA}) = (\#CO©q_{AB} + \#CO©q_{BA} = 0)$$

In den Normen [ISO88b, CCI88f] ist bei der Datenübertragung eine Aufteilung in kleinere Übertragungseinheiten beim Eintrag in die Warteschlangen möglich. Somit kann ein $DL_DATArequest$ zu einem Eintrag von mehreren Objekten in die Warteschlangen führen. In dieser Arbeit wird eine derartige Konkretisierung nicht vorgenommen, da diese Objekte beim Empfang nur zu einem $DL_DATAindication$ führen und somit die Aufteilung in kleinere Einheiten vor den Dienstbenutzern verborgen wird.

In [ISO88b, CCI88f] wird ein Markierungsobjekt zur Synchronisation, das im folgenden mit SO bezeichnet wird, und ein Resetobjekt RO eingeführt. Erzeugt der Diensterbringer ohne Beteiligung eines Dienstbenutzers einen Reset, so wird in beide Warteschlangen ein RO vor einem SO angefügt. In diesem Fall ist das spontane Objekt die Sequenz

[4] Man beachte hierbei, daß ein zweimaliger Verbindungsaufbau, der an einem SAP durchgeführt wird, mit dieser Definition möglich ist. Die lokalen Einschränkungen schließen aber einen derartigen Ablauf aus.

$<RO, SO>$. Wünscht ein Dienstbenutzer einen Reset, so wird in der von ihm ausgehenden Warteschlange ein RO eingetragen, bei der entgegengesetzten Warteschlange ein RO gefolgt von einem SO [ISO88b, CCI88f]:

> A reset procedure initiated by a DLS user is represented by the addition, by the DLS provider, of a reset object into the queue from the Reset initiator to the peer DLS user and the insertion of a reset object followed by a synchronization mark object into the other queue.

Dies führt als Konsequenz dazu, daß ein $DL_RESETresponse$ des anderen Dienstbenutzers keine Operation auf eine Warteschlange hervorruft. Nach der Auffassung der Normen erfolgt dies bereits als Folge des $DL_RESETrequest$. In dieser Arbeit wird die Auffassung vertreten, daß erst beim Bestätigen durch ein $DL_RESETresponse$ die Sequenz $<RO, SO>$ in die Warteschlange eingefügt wird, da dies eine direkte Folge ist. Eine Modifikation dieses Sachverhalts ist durch eine Erweiterung der Objekttabellen auf zwei mögliche Operationen problemlos durchführbar. Da in diesem Fall die Darstellung aufwendiger wird und die vorgeschlagene Vereinfachung den Sachverhalt ausreichend genau modelliert, ist dies in dem in dieser Arbeit präsentierten Queuemodell nicht berücksichtigt worden.

Das Löschen von Synchronisationobjekten SO ist nach [ISO88b, CCI88f] nur in zwei Fällen möglich:

- Ein Verbindungsabbauobjekt DIS kann ein vorausgehendes SO löschen. Dabei bleibt das Objekt DIS danach in der Warteschlange.
- Ein RO löscht möglicherweise sowohl ein vorausgehendes SO, als auch sich selbst.

Auf das Entfernen von Resetobjekten wird nur in der Präzedenztabelle hingewiesen. Eine durch die Ausgabe eines Dienstelements verursachte Löschung wird nicht erwähnt. In dieser Arbeit wird dies durch ein $DL_RESETindication$ sowie einem $DL_RESETconfirm$ bewirkt.

Dienstelement	Objekt	Operation	Bedingung
DL_RESETrequest$_A$	RO	add$_{AB}$	NoConObj
DL_RESETindication$_B$	RO	remove$_{AB}$	
DL_RESETresponse$_B$	RO&SO	add$_{BA}$	NoConObj
DL_RESETconfirm$_A$	RO	remove$_{BA}$	
DL_RESETrequest$_B$	RO	add$_{BA}$	NoConObj
DL_RESETindication$_A$	RO	remove$_{BA}$	
DL_RESETresponse$_A$	RO&SO	add$_{AB}$	NoConObj
DL_RESETconfirm$_B$	RO	remove$_{AB}$	

Tabelle 5.6: Objekttabelle der Synchronisation der Sicherungsschicht

Für den Verbindungsabbau ergibt sich:

Dienstelement	Objekt	Operation	Bedingung
DL_DISCONNECTrequest$_A$	DIS	add$_{AB}$	
DL_DISCONNECTindication$_A$	DIS	remove$_{BA}$	
DL_DISCONNECTrequest$_B$	DIS	add$_{BA}$	
DL_DISCONNECTindication$_B$	DIS	remove$_{AB}$	

Tabelle 5.7: Objekttabelle des Verbindungsabbaus der Sicherungsschicht

Die aus den Tabellen 5.4, 5.5, 5.6 und 5.7 gewonnene Objekttabelle wird als DLO bezeichnet.

Aus den Normen ergibt sich die folgende Präzedenztabelle:

	CO	DO	RO	SO	DIS
CO	N/A	-	-	N/A	DES
DO	N/A	-	DES	N/A	DES
RO	N/A	-	DES	-	DES
SO	N/A	-	DESboth	N/A	DES
DIS	N/A	N/A	N/A	N/A	DES

Tabelle 5.8: Präzedenztabelle der Sicherungsschicht

In der Tabelle 5.8 erfolgt im Unterschied zu [ISO88b, CCI88f] in der zu RO gehörenden Spalte und in der Zeile zu SO der Eintrag $DESboth$ anstelle eines DES. Dies ist notwendig, da es in diesem Fall wegen der verbalen Beschreibung in der Norm zu einer Löschung von beiden Elementen kommt. Der Eintrag DES würde nur zu einer Löschung des SO führen und das Objekt RO erhalten. Beschreibe DLP die zu dieser Tabelle gehörige Präzedenzrelation, sowie DLD und $DLDb$ die Destruktionsrelationen. Die Menge der spontanen Aktionssequenzen ist wie folgt definiert:

$$DLS := \{<RO, SO>, <DIS>\}$$

Somit beschreibt $[\![DLP, DLD, DLDb, DLS, DLO]\!]^{(I,O)}_{\mathcal{QM}}$ die Spurmenge des Queuemodells der Sicherungsschicht.

5.5.3 Das Queuemodell der Vermittlungsschicht

Das in der Vermittlungsschicht [ISO87c, CCI88g] verwendete Queuemodell ist dem der Sicherungsschicht sehr ähnlich. Es wird nur zusätzlich eine Quittierung bei der Datenübertragung mit einem Bestätigungsobjekt, das im weiteren mit AO bezeichnet wird, einbezogen. Außerdem existiert eine priorisierte Datenübertragung, für die das Objekt EXP verwendet wird. Somit erhält man die folgende Objekttabelle:

Dienstelement	Objekt	Operation	Bedingung
N_CONNECTrequest$_A$	CO	add$_{AB}$	isempty
N_CONNECTindication$_B$	CO	remove$_{AB}$	
N_CONNECTresponse$_B$	CO	add$_{BA}$	isempty
N_CONNECTconfirm$_A$	CO	remove$_{BA}$	
N_DATArequest$_A$	DO	add$_{AB}$	NoConObjBA
N_DATAindication$_B$	DO	remove$_{AB}$	
N_DATArequest$_B$	DO	add$_{BA}$	NoConObjBA
N_DATAindication$_A$	DO	remove$_{BA}$	
N_EXPEDITED_DATArequest$_A$	EXP	add$_{AB}$	NoConObjBA
N_EXPEDITED_DATAindication$_B$	EXP	remove$_{AB}$	
N_EXPEDITED_DATArequest$_B$	EXP	add$_{BA}$	NoConObjBA
N_EXPEDITED_DATAindication$_A$	EXP	remove$_{BA}$	
N_DATA_ACKNOWLEDGErequest$_A$	AO	add$_{AB}$	NoConObjBA
N_DATA_ACKNOWLEDGEindication$_B$	AO	remove$_{AB}$	
N_DATA_ACKNOWLEDGErequest$_B$	AO	add$_{BA}$	NoConObjBA
N_DATA_ACKNOWLEDGEindication$_A$	AO	remove$_{BA}$	
N_RESETrequest$_A$	RO	add$_{AB}$	NoConObjBA
N_RESETindication$_B$	RO	remove$_{AB}$	
N_RESETresponse$_B$	RO&SO	add$_{BA}$	NoConObjBA
N_RESETconfirm$_A$	RO	remove$_{BA}$	
N_RESETrequest$_B$	RO	add$_{BA}$	NoConObjBA
N_RESETindication$_A$	RO	remove$_{BA}$	
N_RESETresponse$_A$	RO&SO	add$_{AB}$	NoConObjBA
N_RESETconfirm$_B$	RO	remove$_{AB}$	
N_DISCONNECTrequest$_A$	DIS	add$_{AB}$	
N_DISCONNECTindication$_A$	DIS	remove$_{BA}$	
N_DISCONNECTrequest$_B$	DIS	add$_{BA}$	
N_DISCONNECTindication$_B$	DIS	remove$_{AB}$	

Tabelle 5.9: Objekttabelle der Vermittlungsschicht

Ein weiterer Unterschied zur Sicherungsschicht ist der Ausschluß eines simultanen Verbindungsaufbaus. Aus diesem Grund wird gefordert, daß die Warteschlangen beim Verbindungsaufbau leer sind:

$$isempty(q_{AB}, q_{BA}) = (q_{AB} = \epsilon \wedge q_{BA} = \epsilon)$$

Solange das vom Dienstbenutzer B durch ein $N_CONNECTresponse$ verursachte CO in der Warteschlange ist, darf nach [ISO87c, CCI88g] kein anderes Objekt als ein DIS in die Warteschlangen eingetragen werden. Dies stellt das Prädikat $NoConObjBA$ sicher:

$$NoConObjBA(q_{AB}, q_{BA}) = (CO \copyright q_{BA} = \epsilon)$$

Die in [ISO87c, CCI88g] angegebene Präzedenztabelle ist ebenfalls geringfügig erweitert:

	CO	DO	EXP	AO	RO	SO	DIS
CO	N/A	N/A	-	-	-	N/A	DES
DO	N/A	-	A/A	A/A	DES	N/A	DES
EXP	N/A	-	-	-	A/A	N/A	DES
AO	N/A	-	-	A/A	-	N/A	DES
RO	N/A	-	-	-	DES	-	DES
SO	N/A	-	-	-	DESboth	N/A	DES
DIS	N/A	N/A	N/A	N/A	N/A	N/A	DES

Tabelle 5.10: Präzedenztabelle der Vermittlungsschicht

Die Menge der spontanen Aktionssequenzen ist wiederum als $\{<RO, SO>, <DIS>\}$ definiert. Mit diesen Angaben ist eine einfache Formalisierung des Queuemodells der Vermittlungsschicht mit der zuvor beschriebenen Methode möglich.

5.5.4 Das Queuemodell der Transportschicht

Die Definition des Queuemodells in [ISO84c, CCI88h] ist der der Vermittlungsschicht sehr ähnlich, nur entfallen die Objekte *SO* und *RO* zur Synchronisation, sowie das Datenbestätigungsobjekt *AO*, da diese Dienste in der Transportschicht nicht zur Verfügung gestellt werden. Somit erhält man die folgenden Objekttabelle:

Dienstelement	Objekt	Operation	Bedingung
T_CONNECTrequest$_A$	CO	add$_{AB}$	isempty
T_CONNECTindication$_B$	CO	remove$_{AB}$	
T_CONNECTresponse$_B$	CO	add$_{BA}$	isempty
T_CONNECTconfirm$_A$	CO	remove$_{BA}$	
T_DATArequest$_A$	DO	add$_{AB}$	NoConObjBA
T_DATAindication$_B$	DO	remove$_{AB}$	
T_DATArequest$_B$	DO	add$_{BA}$	NoConObjBA
T_DATAindication$_A$	DO	remove$_{BA}$	
T_EXPEDITED_DATArequest$_A$	EXP	add$_{AB}$	NoConObjBA
T_EXPEDITED_DATAindication$_B$	EXP	remove$_{AB}$	
T_EXPEDITED_DATArequest$_B$	EXP	add$_{BA}$	NoConObjBA
T_EXPEDITED_DATAindication$_A$	EXP	remove$_{BA}$	
T_DISCONNECTrequest$_A$	DIS	add$_{AB}$	
T_DISCONNECTindication$_A$	DIS	remove$_{BA}$	
T_DISCONNECTrequest$_B$	DIS	add$_{BA}$	
T_DISCONNECTindication$_B$	DIS	remove$_{AB}$	

Tabelle 5.11: Objekttabelle der Transportschicht

Die Präzedenztabelle ist:

	CO	DO	EXP	DIS
CO	N/A	-	-	DES
DO	N/A	-	A/A	DES
EXP	N/A	-	A/A	DES
DIS	N/A	N/A	N/A	DES

Tabelle 5.12: Präzedenztabelle der Transportschicht

Als spontane Aktion ist nur das Einfügen eines Verbindungsabbauobjekts erlaubt. Somit ist die Menge der spontanen Aktionssequenzen als $\{<DIS>\}$ definiert.

5.6 Bewertung der in den Normen verwendeten Beschreibungstechnik

Das Queuemodell besitzt im Vergleich zu den TSDs und den lokalen Einschränkungen den geringsten formalen Charakter, da die Beschreibung in den Normen zum größten Teil textuell erfolgt. Die einzigen nichttextuellen Teile sind die Präzedenztabellen, die sehr gut schematisch formalisierbar sind, und die graphische Darstellung der Warteschlangen, die jedoch eine sehr geringe formale Relevanz besitzt. Bei den Präzedenztabellen

der Sicherungs- und der Vermittlungsschicht ist eine Ungenauigkeit erkennbar, da das Auslöschen eines Vorgängerelements und das Auslöschen beider Elemente mit dem gleichen Tabelleneintrag beschrieben wird. Da beide Operationen in der zusätzlichen textuellen Erklärung unterschieden werden, werden in dieser Arbeit im Kapitel 5.5.2 zwei unterschiedliche Einträge eingeführt.

Ein Nachteil der Beschreibungen in den Normen ist die Aktivitätsübergabe an die Dienstbenutzer, die im Kapitel 5.1 kritisiert wird, da hierbei das Prinzip der Datenkapselung verletzt wird. In dieser Arbeit wird daher vorgeschlagen, daß der Diensterbringer die ausschließliche Kontrolle über die Warteschlangen besitzt, wie dies sicherlich in den Normen gedacht ist.

Eine weitere Ungenauigkeit der Normen ist die nur teilweise erfolgte Zuordnung von Dienstelementen zu Operationen auf Warteschlangen. Dies erfolgt zudem in rein textueller Form. Diese Zuordnung ist insbesondere bei Dienstelementen, die ein Entfernen von Objekten aus den Warteschlangen bewirken, nicht beschrieben. Da dies aber für eine exakte Spezifikation unabdingbar ist, werden in dieser Arbeit Objekttabellen eingeführt. Hierbei wird jedem Dienstelement eine Operation auf den Warteschlangen zugewiesen. Die Darstellung der Objekttabellen ist auch für einen informellen Gebrauch eindeutiger verwendbar als die in den Normen benutzte textuelle Form. Zudem eignen sich die Objekttabellen für eine spätere Formalisierung.

5.7 Bewertung der Formalisierungsmethodik

Die in dieser Arbeit eingeführte Formalisierungsmethode für das Queuemodell dient zur exakten Beschreibung der Verbindungseigenschaften eines Diensterbringers. Mit ihr ist es möglich, die Präzedenztabelle und die Objekttabelle für beliebige Schichten auf eine sehr einfache Weise in eine formale Darstellung umzuwandeln. Der Nichtdeterminismus des Diensterbringers wird durch die Einführung von Reduktionsmengen, die die möglichen Zustände der Warteschlangen beschreiben, spezifiziert.

Die Einbeziehung von Dienstelementen ohne Parameter erfolgt in dieser Arbeit mit einer Abstraktionsfunktion, wie dies im Kapitel 5.5 dargestellt wird. Dadurch sind die Tabellen und der Formalismus nicht unnötig kompliziert geworden.

Eine Einbeziehung der Verfälschung von Daten bei der Übertragung ist in diesem Modell nicht direkt beschreibbar. Eine Erweiterung, die dies behebt, ist äquivalent zu der bei TSDs in Kapitel 3.5.2 beschriebenen Methode problemlos durchzuführen.

Kapitel 6

Vergleich der Beschreibungstechniken

In diesem Kapitel werden die Beschreibungstechniken TSD für Time Sequence Diagrams, LC für die lokalen Einschränkungen und QM für das Queuemodell miteinander formal in Beziehung gesetzt.[1] Dies wird überhaupt erst durch die in dieser Arbeit eingeführte formale Semantik für diese Beschreibungstechniken möglich. Es wird im folgenden belegt, daß die einzelnen Beschreibungstechniken nicht die gleiche Ausdruckskraft besitzen, auch in Kombination miteinander, und somit keine von ihnen bei der Erstellung von beliebigen Dienstspezifikationen vernachlässigbar ist. Die Ergebnisse der Untersuchung sind natürlich nur für die in dieser Arbeit festgelegte formale Semantik zutreffend, da die informelle Darstellung der Beschreibungstechniken in den Normen für einen formalen Vergleich nicht ausreichend ist. Bei dieser Betrachtung wird nur die Semantik unter Einschränkung des Umgebungsverhaltens berücksichtigt, da es in einem ersten Ansatz genügt, nur die Spuren, bei denen sich sowohl der Diensterbringer als auch die Dienstbenutzer an die Spielregeln halten, zu betrachten. Insbesondere da bei den Dienstnormen kein Ansatz zu Überlegungen für ein freies Umgebungsverhalten bei der Interpretation der Beschreibungstechniken vorhanden ist. Eine Verwendung dieser Ergebnisse bei der Semantik mit freiem Umgebungsverhalten ist wegen des folgenden Satzes problemlos möglich:

Satz 6.1

Für die Erweiterung um fehlende und fehlerhafte Eingaben nach Definitionen 3.8 und 3.10 gilt:

$$E_{false_inp}(T, \emptyset, O) = T$$

$$E_{missing_inp}(T, \emptyset, O) = T$$

□

[1] Zur Vereinfachung wird im folgenden jeweils TSD, LC und QM sowohl zur Bezeichnung der jeweiligen in dieser Arbeit festgelegten formalen syntaktischen Darstellung einer Beschreibungstechnik, als auch der in den Normen benutzten informellen Darstellung verwendet. Genaugenommen müßte zwischen der in dieser Arbeit festgelegten Syntax der Beschreibungstechnik und der in den Normen verwendeten Darstellung unterschieden werden.

Beweis:

Die Aussage folgt direkt aus den Definitionen der Erweiterung um fehlende und fehlerhafte Eingaben und der leeren Menge von Eingabeaktionen. □

Da es bei den Beschreibungstechniken erlaubt ist, die Eingabemenge beliebig zu bestimmen, ist das Ergebnis des Vergleichs der Semantiken unter Einschränkung des Umgebungsverhaltens auf die Semantiken mit freiem Umgebungsverhalten anwendbar.

6.1 Vergleich der einzelnen Beschreibungstechniken

In diesem Unterkapitel werden die Unterschiede der Beschreibungstechniken TSD, LC und QM untersucht. Es wird formal belegt, daß bei einer isolierten Betrachtung die einzelnen Beschreibungstechniken eine unterschiedliche Ausdruckskraft besitzen und untersucht, ob eine von ihnen durch eine andere ersetzt werden kann. Hierfür sind einige grundlegende Definitionen notwendig.

Definition 6.1

Sei BT eine Beschreibungstechnik. Dann sei $expr_{BT}$ die Sorte aller Ausdrücke, die in der Beschreibungstechnik BT syntaktisch korrekt sind. □

Mit der Definition von $expr_{BT}$ ist es möglich, die Ausdrücke zu beschreiben, die in einer beliebigen Beschreibungstechnik syntaktisch korrekt formuliert sind. Damit sind auch andere als die erwähnten Beschreibungstechniken erfaßt. Zum Beispiel werden mit $expr_{TSD}$ alle Ausdrücke erfaßt, die der Grammatik von TSD-DL entsprechen und TSDs beschreiben. Diese Definition ist bewußt sehr vage gehalten, damit jede mögliche Beschreibungstechnik erfaßt werden kann.

Definition 6.2

Sei BT eine Beschreibungstechnik. Dann bezeichne $[\![BT]\!]$ die *spurbasierte Semantik der Beschreibungstechnik*. $[\![BT]\!]$ bestimmt die Abbildung von Elementen aus $expr_{BT}$ auf Mengen von Spuren.

$$[\![BT]\!] \in expr_{BT} \rightarrow \wp(act^{\omega})$$

□

Somit wird die Semantik eines in einer Beschreibungstechnik formulierten Ausdrucks durch eine Menge von Spuren festgelegt. Eine derartige Spurmenge wird in dieser Arbeit auch als *akzeptierte Abläufe* eines Ausdrucks einer Beschreibungstechnik bezeichnet.

Beispiel 6.1

Für die im Kapitel 3 angegebene Semantik von TSDs gilt bei TSD-DL-Ausdrücken $expr$, die TSDs beschreiben:

$$[\![TSD]\!] := \boldsymbol{\lambda} expr.\ [\![expr]\!]_{TSD}$$

□

Definition 6.3

Sei BT eine Beschreibungstechnik. Dann bezeichne $\Gamma(BT)$ den *Sprachschatz einer Beschreibungstechnik*:

$$\Gamma(BT) := \{[\![BT]\!](e) | e \in expr_{BT}\}$$

□

Der Sprachschatz einer Beschreibungstechnik ist die Menge aller akzeptierten Abläufe. Jedes Element eines Sprachschatzes einer Beschreibungstechnik ist somit eine Menge von Spuren über Aktionen, die der Semantik eines in einer Beschreibungstechnik formulierten Ausdrucks entsprechen.

Definition 6.4

Seien BT_1 und BT_2 zwei Beschreibungstechniken. Dann ist BT_2 *mindestens so mächtig* wie BT_1, wenn gilt:

$$BT_1 \subseteq BT_2 := \Gamma(BT_1) \subseteq \Gamma(BT_2)$$

□

Der Sprachschatz jeder Beschreibungstechnik ist eine Menge von Spurmengen. Jedes Element des Sprachschatzes stellt die von einem Ausdruck einer Beschreibungstechnik beschriebene Spurmenge, das sind die akzeptierten Abläufe, dar. Nach Definition 6.4 ist eine Beschreibungstechnik mächtiger als eine andere, wenn sich mit ihr mehr Spurmengen beschreiben lassen.

Beispiel 6.2

Bezeichnet man die bei den lokalen Einschränkungen im Kapitel 4 verwendete Automatenmethode mit LC_{Aut}, sowie die Tabellenmethode mit LC_{Tab}, so gilt wegen Satz 4.1 $LC_{Tab} \subseteq LC_{Aut}$, aber $LC_{Aut} \not\subseteq LC_{Tab}$ □

Definition 6.5

Seien BT_1und BT_2 zwei Beschreibungstechniken. Dann ist BT_1 *äquivalent zu* BT_2, wenn gilt:

$$BT_1 = BT_2 := \Gamma(BT_1) = \Gamma(BT_2)$$

□

Definition 6.6

Seien BT_1und BT_2 zwei Beschreibungstechniken. Dann sind BT_1 und BT_2 *wechselseitig unterschiedlich mächtig*, wenn gilt:

$$BT_1 \perp BT_2 := \Gamma(BT_1) \not\subseteq \Gamma(BT_2) \wedge \Gamma(BT_2) \not\subseteq \Gamma(BT_1)$$

□

Aus $BT_1 \perp BT_2$ folgt, daß mit jeder der Beschreibungstechniken BT_1 und BT_2 eine Spurmenge darstellbar ist, die mit der anderen nicht beschrieben werden kann.

Aufbauend auf diesen Definitionen werden nun die in den vorigen Kapiteln formalisierten Beschreibungstechniken TSD, LC und QM in Beziehung gesetzt.

Satz 6.2

Die Beschreibungstechniken, mit denen Time Sequence Diagrams und die lokalen Einschränkungen dargestellt werden, sind wechselseitig unterschiedlich mächtig:

$$TSD \perp LC$$

□

Beweis:

1) $TSD \not\subseteq LC$ gilt, da wegen Satz 3.10 TSDs in der Regel keine reine Sicherheitseigenschaft beschreiben, LC aber nach Satz 4.2 schon.

2) Die Gültigkeit von $LC \not\subseteq TSD$ wird am Automat aus Abbildung 6.1 gezeigt.

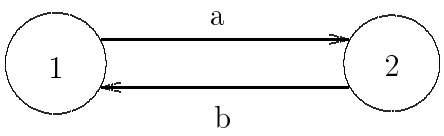

Abbildung 6.1: Automat, der mit TSDs nicht äquivalent dargestellt werden kann

Der akzeptierte Sprachschatz des in Abbildung 6.1 dargestellten Automaten ist beim Anfangszustand 1:

$$T := \{t | t \sqsubseteq (\mathsf{fix}\boldsymbol{\lambda} s.a\&b\&s)\}$$

Es gilt:

$$<a> \in T$$

Wenn die Spurmenge T durch einen TSD-DL-Ausdruck nach dem im Kapitel 3.3.6 beschriebenen Verfahren dargestellt wird, und diese TSD-DL-Ausdrücke beschreiben ja die TSDs, so folgt mit Satz 6.3 auf Seite 129, wenn $t_1 = t_2 = <a>$ gesetzt wird:

$$<a, a> \in T$$

Da aber $<a, a> \notin T$ gilt, führt dies zum Widerspruch. □

Mit dem folgenden Satz wird festgestellt, daß, falls zwei Spuren in einer durch TSDs beschriebenen Spurmenge enthalten sind, dann auch das Interleaving von beiden in der Spurmenge enthalten ist.

Satz 6.3

Sei T die Spurmenge eines TSD-DL-Ausdrucks, der nach dem im Kapitel 3.3.6 beschriebenen Verfahren TSDs darstellt, so gilt für die Spuren t_1 und t_2:

$$(t_1 \in T \wedge t_2 \in T) \Rightarrow \{t_1\} \bowtie \{t_2\} \subset T$$

□

Beweis:

Da die Spurmenge T durch einen TSD-DL-Ausdruck nach dem im Kapitel 3.3.6 beschriebenen Verfahren darstellt wird, existiert ein TSD-DL-Ausdruck TSD' für den gilt:

$$T = [\![(TSD')^{\underline{\omega}}]\!]_{TSD}$$

Es gilt nach der Voraussetzung:

$$t_1 \in [\![(TSD')^{\underline{\omega}}]\!]_{TSD} \wedge t_2 \in [\![(TSD')^{\underline{\omega}}]\!]_{TSD}$$

Nach Korollar 3.2 und wegen $(t_1 \in T_1 \wedge t_2 \in T_2) \Rightarrow \{t_1\} \bowtie \{t_2\} \subset T_1 \bowtie T_2$ gilt:

$$\{t_1\} \bowtie \{t_2\} \subset [\![(TSD'|TSD')^{\underline{\omega}}]\!]_{TSD}$$

Nach Satz 3.1 Aussage (6) gilt:

$$\{t_1\} \bowtie \{t_2\} \subset [\![(TSD')^{\underline{\omega}}]\!]_{TSD}$$

Und somit

$$\{t_1\} \bowtie \{t_2\} \subset T$$

□

Satz 6.4

Die Beschreibungstechniken, mit denen Time Sequence Diagrams und das Queuemodell dargestellt werden, sind wechselseitig unterschiedlich mächtig:

$$TSD \perp QM$$

□

Beweis:

1) $TSD \nsubseteq QM$ gilt, da wegen Satz 3.10 TSDs in der Regel nicht präfixabgeschlossen sind, QM aber nach Satz 5.1 schon.

2) Die Gültigkeit von $QM \nsubseteq TSD$ wird an folgender Objekttabelle gezeigt:

Dienstelement	Objekt	Operation	Bedingung
a	DO	add_{AB}	
b	DO	remove_{AB}	

Tabelle 6.1: Objekttabelle

Nach der Objekttabelle 6.1 sind also a, b zwei Dienstelemente, die eine reihenfolgeerhaltende Datenübertragung beschreiben. Sei QM' das dazugehörige Queuemodell. Somit gilt: $<a(x_1), b(x_1)> \in [\![QM']\!]_{\mathcal{QM}}$ und $<a(x_2), b(x_2)> \in [\![QM']\!]_{\mathcal{QM}}$ Sei nun T die Spurmenge, die durch TSDs beschrieben wird. Es gilt: $T = [\![QM']\!]_{\mathcal{QM}}$. Nach Satz 6.3 gilt:$<a(x_1), a(x_2), b(x_2), b(x_1)> \in T$. Dies steht aber im Widerspruch zu $<a(x_1), a(x_2), b(x_2), b(x_1)> \notin [\![QM']\!]_{\mathcal{QM}}$

□

Satz 6.5

Die Beschreibungstechniken, mit denen das Queuemodell und die lokalen Einschränkungen dargestellt werden, sind wechselseitig unterschiedlich mächtig:

$$QM \perp LC$$

□

Beweis:

1) $QM \not\subseteq LC$ läßt sich äquivalent zum Teil $QM \not\subseteq TSD$ des Beweises zum Satz 6.4 zeigen. Anhand der Objekttabelle 6.1 wird wiederum eine reihenfolgeerhaltende Datenübertragung verwendet, die sich aber mit LCs nicht beschreiben läßt, wie man dies an der Definition 4.3 unmittelbar sieht, da beide Dienstelemente an unterschiedlichen SAPs auftreten und somit der akzeptierte Sprachschatz $\{a, b\}^\omega \cup \{a^n | n \in \mathbb{N}\} \bowtie \{b^m | m \in \mathbb{N}\}$ mit beliebigen Parametern ist.

2) Die Gültigkeit von $LC \not\subseteq QM$ wird mit dem Automaten aus Abbildung 6.1 von Seite 128 gezeigt. Dessen Sprachschatz ist, wenn man die Zustände 1 und 2 als Anfangszustände zuläßt:

$$T := \{t | t \sqsubseteq (\mathsf{fix}\boldsymbol{\lambda} s.a\&b\&s) \vee t \sqsubseteq (\mathsf{fix}\boldsymbol{\lambda} s.b\&a\&s)\}$$

Für das Queuemodell sind die folgenden Eintragungen in der Objekttabelle möglich. Hierbei sei O_a das zu a gehörende und O_b das zu b gehörende Objekt, sowie QM' das zugehörige Queuemodell, dessen Objekttabelle wie folgt definiert wird[2]:

[2]Zur Vereinfachung werden im folgenden die Parameter der Dienstelemente vernachlässigt, da sie von den lokalen Einschränkungen nicht berücksichtigt werden. Man kann die Dienstelemente ohne Parameter als Abstraktion der mit Parametern versehenen Dienstelemente betrachten.

1. a sei vom Typ none oder b sei vom Typ none: Damit gilt $[\![QM']\!]_{\mathcal{QM}} = \{a,b\}^\omega$ und somit $<a,a> \in [\![QM']\!]_{\mathcal{QM}}$. Da diese Spur nicht Element aus T ist, führt dies zum Widerspruch.
2. a, b seien vom Typ remove: Somit ist es wegen der Definition von T erforderlich, daß $<O_a,O_b>$ und $<O_b,O_a>$ in der Menge der spontanen Objektsequenzen liegen. Dadurch gilt: $<a,b,b,a> \in [\![QM']\!]_{\mathcal{QM}}$, was zum Widerspruch führt.
3. a, b seien vom Typ add: Damit kann die Queue unendlich groß werden. Da dies vom Queuemodell explizit ausgeschlossen wird, führt dies zum Widerspruch.
4. a sei vom Typ add und b vom Typ remove: Somit gilt $<b,a> \notin [\![QM']\!]_{\mathcal{QM}}$
5. a sei vom Typ remove und b vom Typ add: Somit gilt $<a,b> \notin [\![QM']\!]_{\mathcal{QM}}$

□

Anhand der durch die Sätze 6.2, 6.4 und 6.5 festgestellten Ergebnisse ist formal belegt worden, daß keine der Beschreibungstechniken TSD, LC und QM durch eine andere von diesen drei Beschreibungstechniken ersetzbar ist. Jede dieser Beschreibungstechniken kann bestimmte Teileigenschaften ausdrücken, die mit keiner der beiden anderen darstellbar sind.

6.2 Kombination der Beschreibungstechniken

In diesem Unterkapitel wird die Mächtigkeit von kombinierten Beschreibungstechniken miteinander verglichen. Als Kombination von Beschreibungstechniken wird die Konjunktion beziehungsweise die Schnittmengenbildung gewählt.

Definition 6.7

Sei $I \subsetneq \mathbb{N}$. Für $i \in I$ seien BT_i Beschreibungstechniken. Dann ist die *Kombination einer Menge von Beschreibungstechniken* definiert als:

$$\Cap\{BT_i |\ i \in I\} := \{\bigcap_{i \in I} T_i |\ T_i \in \Gamma(BT_i)\}$$

□

Mit dieser Kombination wird das Zusammenwirken von mehreren Beschreibungstechniken in Form einer Schnittmengenbildung modelliert. $\Cap$ wird im folgenden zur Abkürzung auch als Infixoperator verwendet. Es gilt: $BT_1 \Cap BT_2 := \Cap\{BT_1, BT_2\}$ So beschreibt $BT_1 \Cap BT_2$ den möglichen Sprachschatz zweier Beschreibungstechniken, der durch Schnittmengenbildung erhalten wird.

Die möglichen Spurmengen, die durch die Kombination von TSD, LC und QM erhalten werden können, werden mit $\Cap\{TSD, LC, QM\}$ beschrieben.

Aufbauend auf diesen Definitionen wird im folgenden die unterschiedliche Ausdrucksmächtigkeit der konjunktiven Kombination zweier Beschreibungstechniken mit der dritten untersucht.

Satz 6.6

Die Beschreibungstechnik der Time Sequence Diagrams und die Kombination der Beschreibungstechnik der lokalen Einschränkungen mit der des Queuemodells sind wechselseitig unterschiedlich mächtig:

$$TSD \perp LC \Cap QM$$

□

Beweis:

Der Beweis läßt sich in zwei Teile aufspalten:

1. $TSD \not\subseteq LC \Cap QM$
   In der Regel beschreiben nach Satz 3.10 TSDs keine bezüglich Präfixen abgeschlossene Spurmengen. LC und QM beschreiben jedoch nach den Sätzen 4.2 und 5.1 ausschließlich derartige Spurmengen. Da die Schnittmenge zweier präfixabgeschlossener Spurmengen wiederum präfixabgeschlossen ist, sind alle Elemente von $LC \Cap QM$ präfixabgeschlossen. Somit folgt die Aussage.
2. $LC \Cap QM \not\subseteq TSD$
   Der Beweis wird äquivalent zum Beweis zu Satz 6.4, Fall $QM \not\subseteq TSD$ geführt. Es ist möglich, mit LC keine Einschränkung, also die Spurmenge $\{a, b\}^\omega$, zu beschreiben.

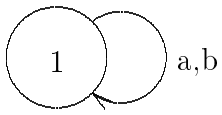

Abbildung 6.2: Automat, der a, b akzeptiert

Sei diese Darstellung der lokalen Einschränkungen mit LC' bezeichnet. Somit gilt für das im Beweis zu Satz 6.4 definierte Queuemodell QM': $[\![LC']\!]_{\mathcal{LC}} \cap [\![QM']\!]_{\mathcal{QM}} = [\![QM']\!]_{\mathcal{QM}}$. Woraus die Behauptung folgt.

□

Satz 6.7

Die Beschreibungstechnik der lokalen Einschränkungen und die Kombination der Beschreibungstechnik der Time Sequence Diagrams mit der des Queuemodell sind wechselseitig unterschiedlich mächtig:

$$LC \perp TSD \Cap QM$$

□

Beweis:

Der Beweis läßt sich in zwei Teile aufspalten:

1. $LC \not\subseteq TSD \Cap QM$

   Der Beweis wird äquivalent zum Beweis zu Satz 6.5, Fall $LC \not\subseteq QM$ geführt. Nur wird hierbei die mögliche Wirkung von TSDs zusätzlich einbezogen. Im folgenden wird nur eine Ergänzung der 5 Fälle des Beweises zu Satz 6.5 in Teil 2 um TSDs angegeben:

   Ergänzung zu 1. Mit TSDs können die Spuren weiter eingeschränkt werden. Aus $<a,b> \in [\![TSD']\!]_{\mathcal{TSD}}$ folgt wegen Satz 6.3: $<a,a,b,b> \in [\![TSD']\!]_{\mathcal{TSD}}$ Somit gilt dann trotzdem $<a,a,b,b> \in [\![TSD']\!]_{\mathcal{TSD}} \cap [\![QM']\!]_{\mathcal{QM}}$. Was zum Widerspruch führt.

   Ergänzung zu 2. Die Spur $<a,b,b,a>$ kann wegen Satz 6.3 bei TSDs nur dann ausgeschlossen werden, wenn auch $<a,b>$ ausgeschlossen wird.

   Ergänzung zu 3.-5. Da das Queuemodell ohnehin zuwenig Spuren beschreibt, kann dies nicht mit einer Schnittmengenbildung mit TSDs behoben werden.

2. $TSD \Cap QM \not\subseteq LC$

   Der Beweis läßt sich äquivalent zum Beweis zu Satz 6.5, Fall $QM \not\subseteq LC$ führen. Wiederum ist durch das QM eine reihenfolgeerhaltende Datenübertragung spezifiziert.

Abbildung 6.3: TSDs, die alle Spuren über a, b akzeptieren

Sei $TSD' = (a|b)^{\underline{\omega}}$ der dieses TSD beschreibende TSD-DL-Ausdruck. Es gilt: $[\![TSD']\!]_{\mathcal{TSD}} = \{a,b\}^{\omega}$. Somit gilt auch: $[\![TSD']\!]_{\mathcal{TSD}} \cap [\![QM']\!]_{\mathcal{QM}} = [\![QM']\!]_{\mathcal{QM}}$, woraus die Behauptung folgt.

□

Satz 6.8

Die Beschreibungstechniken, mit denen das Queuemodell und die Kombination der Time Sequence Diagrams mit den lokalen Einschränkungen dargestellt werden, sind wechselseitig unterschiedlich mächtig:

$$QM \perp TSD \Cap LC$$

□

Beweis:

Der Beweis läßt sich in zwei Teile aufspalten:

1. $QM \not\subseteq TSD \Cap LC$

   Der Beweis läßt sich äquivalent zum Beweis zu Satz 6.5, Fall $QM \not\subseteq LC$ führen. Wiederum ist durch das QM eine reihenfolgeerhaltende Datenübertragung beschrieben. Da die beiden Dienstelemente a,b an verschiedenen SAPs auftreten, werden durch LC entweder alle Spuren aus $\{a,b\}^\omega$ oder aus $\{a^n | n \in \mathbb{N}\} \bowtie \{b^m | m \in \mathbb{N}\}$ mit beliebigen Parametern akzeptiert. Anhand von Satz 6.3 erkennt man, daß TSDs ebenfalls keinen Aspekt zur Reihenfolgeerhaltung beitragen. Somit ist auch die Schnittmengenbildung nicht erfolgreich.

2. $TSD \Cap LC \not\subseteq QM$

   Es sei ein TSD beschrieben wie in Satz 3.10, das eine nicht präfixabgeschlossene Menge charakterisiert. Mit LC seien alle Spuren aus $\{a,b\}^\omega$ beschrieben. Somit schränkt LC die durch TSD beschriebene Spurmenge nicht weiter ein. Die Schnittmenge beider Spurmengen ist dann nicht präfixabgeschlossen und kann wegen Satz 5.1 mit QM nicht beschrieben werden.

□

Mit den Aussagen aus den Sätzen 6.6, 6.7 und 6.8 folgt, daß keine der drei semi-formalen Beschreibungstechniken durch eine konjunktive Kombination der restlichen Beschreibungstechniken adäquat ersetzt werden kann.

Satz 6.9

Seien A, B und C Beschreibungstechniken, und act eine Menge von Aktionen. Wenn $\Gamma(A) \subseteq \wp(act^\omega) \wedge \Gamma(A) \neq \emptyset \wedge act^\omega \in \Gamma(C)$, dann gilt:

1. $B \not\subseteq A \Rightarrow B \Cap C \not\subseteq A$
2. $A \subseteq A \Cap C$

□

Wenn in der Beschreibungstechnik C die ganze Spurmenge über act^ω beschrieben werden kann, so ändert die konjunktive Komposition nicht die Teilmengeneigenschaft.

Beweis:

Seien A, B und C Beschreibungstechniken. Wenn $\Gamma(A) \subseteq \wp(act^\omega) \wedge \Gamma(A) \neq \emptyset$, dann gilt: $\exists T_1.\ T_1 \in \Gamma(A) \wedge T_1 \in \wp(act^\omega)$.

Da $T_1 \in \wp(act^\omega) \wedge act^\omega \in \Gamma(C)$ folgt: $T_1 \cap act^\omega = T_1$

1. Aus $B \not\subseteq A$ folgt: $\exists T_1.\ T_1 \in \Gamma(A) \wedge T_1 \notin \Gamma(B)$
   Und somit folgt mit $T_1 \cap act^\omega = T_1$: $T_1 \notin B \Cap C$

2. Aus $T_1 \in \Gamma(A)$ folgt mit $T_1 \cap act^\omega = T_1$: $T_1 \in A \Cap C$

□

Satz 6.10

Für die Beschreibungstechniken gelten bei einer endlichen Aktionenmenge act die folgenden Aussagen:

1. $TSD \Cap LC \Cap QM \not\subseteq TSD \Cap LC$
2. $TSD \Cap LC \Cap QM \not\subseteq TSD \Cap QM$
3. $TSD \Cap LC \Cap QM \not\subseteq LC \Cap QM$
4. $TSD \Cap LC \Cap QM \supseteq TSD \Cap LC$
5. $TSD \Cap LC \Cap QM \supseteq TSD \Cap QM$
6. $TSD \Cap LC \Cap QM \supseteq LC \Cap QM$

□

Beweis:

Die Aussagen folgen direkt aus der Anwendung des Satzes 6.9 auf die Sätze 6.6, 6.7 und 6.8, sowie der Tatsache, daß der Sprachschatz von jeder der Beschreibungstechniken nicht leer ist, und daß bei einer endlichen Aktionenmenge, das trifft bei den Dienstelementen zu, die triviale Spurmenge act^ω beschreibbar ist. □

Somit zeigt sich, daß jede der drei Beschreibungstechniken TSD, LC und QM einen eigenständigen Beitrag zur Beschreibung von Eigenschaften ermöglicht.

6.3 Ergebnisse des Vergleichs der Beschreibungstechniken

Betrachtet man den unterschiedlichen Formalisierungsgrad der in den Normen verwendeten Beschreibungstechniken TSD, LC und QM, so fällt auf, daß LC den ausgeprägtesten formalen Charakter dieser Beschreibungstechniken besitzen, da sie auf dem theoretisch fundierten Konzept der Automaten basiert. An zweiter Stelle folgen die TSDs aufgrund ihrer graphischen Darstellung. Leider sind bei TSDs in den Normen wesentliche Hinweise auf deren Semantik unterblieben, wie dies im Kapitel 3 verdeutlicht wurde. Dieser Nachteil konnte jedoch in dieser Arbeit behoben werden. Die am wenigsten formale Beschreibungstechnik ist das Queuemodell, da es fast ausschließlich aus textuellen Erklärungen besteht. In dieser Arbeit wird durch die Einführung der Objekttabellen der Formalisierungsgrad dieser Beschreibungstechnik in ausreichender Weise erhöht.

Durch die lokalen Einschränkungen werden Eigenschaften an jeweils einem SAP isoliert betrachtet. Beim Queuemodell dagegen wird normalerweise ausschließlich der Austausch

von Dienstelementen und somit die Abfolge von Dienstelementen an beiden SAPs dargestellt. Mit TSDs ist sowohl die Abfolge von Dienstelementen an einem SAP als auch an beiden SAPs beschreibbar.

Durch die Ergebnisse der vorherigen Kapitel ist gezeigt worden, daß bei einer formalen Betrachtung jede der in den Normen verwendeten Beschreibungstechniken TSD, LC und QM notwendig ist, da jede einen eigenständigen Beitrag zur Spezifikation von Eigenschaften eines Diensterbringers liefert. Somit läßt sich die Existenz der einzelnen Beschreibungstechniken formal begründen. Natürlich sind die Ergebnisse dieses Vergleichs nur für die Formalisierung der Beschreibungstechniken feststellbar. Da jedoch die Formalisierung einer exakten Interpretation der Beschreibungstechniken entspricht, sind diese Ergebnisse auch auf die in den Normen verwendete informelle Darstellung anwendbar.

Ein weiterer Aspekt ist die Untersuchung, ob in den einzelnen Dienstnormen der jeweiligen Schichten des ISO/OSI Schichtenmodells eine Beschreibungstechnik redundant ist, da zum Beispiel ein ausschließlich durch diese Beschreibungstechnik darstellbarer Aspekt nicht in dieser Norm benötigt wird. Da derartige Untersuchungen nur zum Teil auf den getroffenen Aussagen beruhen und unter Umständen neuartige Beweise erfordern, würde dies den Rahmen dieser Arbeit erheblich erweitern. Diese Untersuchungen werden daher in dieser Arbeit nicht mehr durchgeführt.

Kapitel 7

Beschreibung einer Schicht

In diesem Kapitel wird die Kombination der in den Kapiteln 3, 4 und 5 dargestellten Beschreibungstechniken zu einer Schichtbeschreibung untersucht. Zuerst wird dies für eine Semantik unter Beeinflussung des Umgebungsverhaltens realisiert. Darauf aufbauend wird eine Kombination der Teilsemantiken festgelegt, so daß ein freies Umgebungsverhalten möglich ist.

7.1 Schichtbeschreibung mit Einschränkung des Umgebungsverhaltens

Mit den in den Kapiteln 3, 4 und 5 dargestellten Beschreibungstechniken lassen sich die wesentlichen Eigenschaften einer beliebigen Schicht darstellen. Es werden jedoch bei manchen Schichten spezielle Eigenschaften gefordert, die mit den bisher formalisierten Beschreibungstechniken nicht darstellbar sind. Als Beispiel hierfür sei das Konzept der Datenrepräsentation in der Darstellungsschicht in [ISO88a, CCI88j] genannt. Die Datenrepräsentation ist bei der Beschreibung von verteilten Anwendungen von wesentlicher Bedeutung. Wollen zum Beispiel zwei Dienstbenutzer eine ganze Zahl austauschen, dann kann es möglich sein, daß die Bitdarstellung bei den Dienstbenutzern aufgrund von verschiedener Hardware unterschiedlich ist. Es genügt also nicht, daß der Diensterbringer die Bitdarstellung übermittelt, sondern er muß gegebenenfalls eine Umwandlung in die lokalen Formate vornehmen. Die bisher formalisierten Beschreibungstechniken TSD, LC und QM berücksichtigen keine Datentypen für die Nutzdaten. Somit ist dieser Aspekt der Datenübertragung mit TSD, LC und QM nicht darstellbar und dadurch ist eine bisher nicht formalisierte Beschreibungstechnik notwendig. In den Normen wird dazu die Beschreibung der Datenrepräsentation mit ASN.1 (Abstract Syntax Notation One) [ISO90, CCI88c] verwendet. Die Einführung einer eigenen Beschreibungstechnik, die diesen Aspekt abdeckt, ist bei einer Methodik zur Beschreibung aller Schichten nur von untergeordnetem Interesse. Damit der Umfang dieser Arbeit nicht unnötig vergrößert wird, sind derartige Eigenschaften einer Schichtenspezifikation bei der vorgestellten Methodik mit dem schichtenspezifischen Prädikat $special_N$, das in dieser Arbeit nicht näher formalisiert wird, spezifiziert. Somit setzt sich die Spezifikation einer beliebigen Schicht wie folgt zusammen:

Definition 7.1

Die *Beschreibung einer Schicht* wird definiert als:

$$(TSD_N, LC_N, QM_N, special_N)$$

wobei TSD_N ein TSD-DL-Ausdruck, der TSDs beschreibt, LC_N eine Beschreibung der lokalen Einschränkungen, QM_N ein Queuemodell und $special_N$ ein schichtenspezifisches Prädikat über Spuren ist. □

Hierbei ist $special_N$ das in dieser Arbeit nicht explizit dargestellte schichtenspezifische Prädikat. In den unteren Schichten ist dieses Prädikat nicht notwendig, da deren Eigenschaften mit TSDs, LC und QM bereits ausreichend beschrieben sind. Bei der Darstellungsschicht beschreibt es zum Beispiel die korrekte Datenrepräsentation gemäß ASN.1. In der Kommunikationssteuerungsschicht wird zum Beispiel mit $special_N$ das Synchronisationsmanagement exakt beschrieben, wobei auch die Behandlung von Haupt- und Nebensynchronisationspunkten detaillierter erfaßt wird, als dies mit TSDs, LC und QM möglich ist. Die Spezifikation dieser schichtenspezifischen Prädikate dürfte problemlos möglich sein. Da diese Beschreibungen nur auf jeweils eine Schicht angewendet werden können und somit keinen wesentlichen Beitrag für eine Methodik zur Beschreibung aller Schichten liefern, unterbleibt eine detaillierte Spezifikation des Prädikats $special_N$.

Mit den in den vorigen Kapiteln beschriebenen Teilsemantiken unter Beeinflussung des Umgebungsverhaltens wird die Schichtbeschreibung durch eine einfache Schnittmengenbildung realisiert:

Definition 7.2

Die *Semantik einer Schichtbeschreibung* $(TSD_N, LC_N, QM_N, special_N)$ *unter Einschränkung des Umgebungsverhaltens* wird definiert als:

$$\begin{aligned}&[\![(TSD_N, LC_N, QM_N, special_N)]\!]_{\mathcal{L}} := \\ &\quad [\![TSD_N]\!]_{\mathcal{TSD}} \cap [\![LC_N]\!]_{\mathcal{LC}} \cap [\![QM_N]\!]_{\mathcal{QM}} \cap \{t | special_N(t)\}\end{aligned}$$

□

Auch bei der Kombination der Teilsemantiken wird das Umgebungsverhalten eingeschränkt. So ist zum Beispiel eine jederzeitige Eingabeaktion bei den meisten Beschreibungen nicht mehr möglich. Falls nur eine der beteiligten Beschreibungen eine Eingabeaktion zu einem Zeitpunkt nicht mehr erlaubt, so ist diese Aktion bei der Kombination wegen der Schnittmengenbildung ebenso nicht erlaubt. Exemplarisch sei hierfür auf die lokalen Einschränkungen der Transportschicht, die im Beispiel 4.2 auf Seite 90 dargestellt wurden, verwiesen, nach der ein $T_DATrequest$ nur bei einer bestehenden Verbindung eingegeben werden darf. Somit schränkt die in Definition 7.2 festgelegte Semantik das Umgebungsverhalten ein. Da dies unerwünscht ist, wird im nächsten Unterkapitel eine Semantik angegeben, bei der dies nicht der Fall ist.

7.2 Schichtbeschreibung ohne Einschränkung des Umgebungsverhaltens

Im folgenden wird die Kombination der Teilsemantiken der Beschreibungstechniken ohne Einschränkung des Umgebungsverhaltens untersucht. Da in den Kapiteln 3, 4 und 5 die Erweiterung um diejenigen Teile, bei denen die Umgebung ihre Verpflichtung verletzt hat, nur bei den jeweiligen Beschreibungstechniken isoliert vorgenommen wurde, stellt sich in diesem Unterkapitel die Frage, in welcher Weise die Spurmengen kombiniert werden sollen, so daß ein freies Umgebungsverhalten möglich ist. Wählt man in einem ersten Ansatz ebenso wie bei der Semantik mit eingeschränktem Umgebungsverhalten die Schnittmengenbildung, so erhält man:

$$[\![(TSD_N, LC_N, QM_N, special_N)]\!]_{\mathcal{L}}^{(I,O)} := \\ [\![TSD_N]\!]_{\mathcal{TSD}}^{(I,O)} \cap [\![LC_N]\!]_{\mathcal{LC}}^{(I,O)} \cap [\![QM_N]\!]_{\mathcal{QM}}^{(I,O)} \cap \{t | special_N^{(I,O)}(t)\}$$

Hierbei beschreibt $special_N^{(I,O)}$ die Erweiterung des Prädikats $special_N$, so daß ein freies Umgebungsverhalten möglich ist. Diese Kombination entspricht der in [AL90, AL93] angegebenen. Man beachte aber, daß diese Art der Kombination in [AL90, AL93] für unterschiedliche Komponenten verwendet wird. Bei der Schichtbeschreibung wird jedoch nur eine Komponente mehrmals, jeweils unterschiedliche Aspekte betreffend, beschrieben. Für einen ersten Ansatz der Kombination soll dies aber dennoch untersucht werden, da der Kombinationsoperator wegen der Schnittmengenbildung sehr einfach ist. Eine Darstellung dieser Art der Kombination in Logiknotation mit dem Operator $\longrightarrow\!\triangleright$ aus [AL93] ergibt:

$$(A_{TSD_N} \longrightarrow\!\triangleright C_{TSD_N}) \wedge (A_{LC_N} \longrightarrow\!\triangleright C_{LC_N}) \wedge (A_{QM_N} \longrightarrow\!\triangleright C_{QM_N}) \wedge (A_{special_N} \longrightarrow\!\triangleright C_{special_N})$$

Die Auswirkung dieser Semantikdefinition ist jedoch die Einschränkung des Umgebungsverhaltens, die im Hinblick auf eine spätere Implementierung vermieden werden soll. Dies wird im folgenden formal belegt.

Bei dieser Definition der Kombination kann festgestellt werden, daß eine Eingabeaktion jederzeit möglich ist. Hierzu wird in dieser Arbeit der Begriff der *jederzeitigen Eingabemöglichkeit* für Spurmengen festgelegt:

Definition 7.3

> Bei einer Spurmenge T und einer Menge von Eingabeaktionen I ist eine *Eingabe jederzeit möglich*, wenn gilt:
>
> $$InpEn(T, I) := (t \in PRE(T) \Rightarrow \forall i \in I.\ t \circ <i> \in PRE(T))$$

□

Wenn bei einer Spurmenge die Eingabe jederzeit möglich ist, dann wird auch im folgenden äquivalent die Bezeichnung verwendet, daß sie *input enabled* ist. Eine Spurmenge ist genau dann input enabled, wenn die durch das Anfügen einer Eingabeaktion an jedes Präfix der Spurmenge entstehende Spur in dieser Spurmenge enthalten ist, also eine Eingabeaktion zu jedem Zeitpunkt nicht zum Verlassen der Spurmenge führt.

Satz 7.1

Es gilt für die in der Definition 3.8 festgelegte Erweiterung um fehlende Eingaben, eine Spurmenge T und Mengen von Aktionen I, O:

$$InpEn(X \cup E_{false_input}(X, I, O), I)$$

□

Beweis:

Voraussetzung: $t \in PRE(X \cup E_{false_input}(X, I, O))$

Fallunterscheidung nach $t \in PRE(X)$:

1. $t \notin PRE(X)$. Somit gilt $t \in PRE(E_{false_input}(X, I, O))$. Daraus folgt wegen der Definition von E_{false_input}: $\forall x.\ t \circ x \in PRE(E_{false_input}(X, I, O))$, also auch $t \circ <i> \in PRE(E_{false_input}(X, I, O))$
2. $t \in PRE(X)$. Wenn $t \circ <i> \in PRE(X)$ gilt, folgt die Behauptung.
   Falls $t \circ <i> \notin PRE(X)$ gilt, trifft $t \circ <i> \in PRE(E_{false_input}(X, I, O))$ wegen der Definition von E_{false_input} zu.

□

Somit ist eine beliebige Spurmenge, die mit E_{false_input} erweitert wurde, input enabled. Dies ist nicht verwunderlich, da E_{false_input} eine jederzeitige Eingabe ermöglichen soll.

Die Eigenschaft der jederzeitigen Eingabemöglichkeit von zwei Mengen ist gegenüber der Schnittmengenbildung abgeschlossen:

Satz 7.2

Es gilt für die Spurmengen X_1, X_2 und die Menge von Aktionen I:

$$(InpEn(X_1, I) \wedge InpEn(X_2, I)) \Rightarrow InpEn(X_1 \cap X_2, I)$$

□

Beweis:

Voraussetzung: $(InpEn(X_1, I) \wedge InpEn(X_2, I))$

Annahme: $t \in PRE(X_1 \cap X_2)$

Daraus folgt: $t \in PRE(X_1) \cap PRE(X_2)$

Wenn $\#t = \infty$ gilt, dann ist $t \circ <i> = t$ erfüllt. Woraus die Behauptung folgt.

Für $\#t \neq \infty$ liefert Induktion unter Verwendung von $InpEn$: $t \circ i^\infty \in PRE(X_1)$

Daraus folgt: $t \circ i^\infty \in X_1$

Äquivalent dazu: $t \circ i^\infty \in X_2$

Somit: $t \circ i^\infty \in X_1 \cap X_2$ und $t \circ <i> \in PRE(X_1 \cap X_2)$ □

Somit gilt bei der vorgeschlagenen Semantik, die man durch die Schnittmengenbildung der Teilsemantiken mit freiem Umgebungsverhalten erhält, daß dabei eine jederzeitige Eingabe möglich ist.

Obwohl bei der Schnittmengenbildung die Eigenschaft input enabled erhalten bleibt, kann jedoch eine Verletzung des Lebendigkeitsanteils der Assumption durch die Schnittmengenbildung entstehen, wie dies am Beispiel 7.1 deutlich wird:

Beispiel 7.1

Seien für $I = \{i\}$ und $O = \{a, b, c\}$ die folgenden Spurmengen definiert:

$$\begin{aligned} X_1 &= \{<a>, <i>\&x, <a,i,b>, <a,i,b,i>\&x, <a,i,i>\&x \mid x \in \{I \cup O\}^\omega\} \\ X_2 &= \{<a>, <i>\&x, <a,i,c>, <a,i,c,i>\&x, <a,i,i>\&x \mid x \in \{I \cup O\}^\omega\} \end{aligned}$$

Diese Mengen sind zum Beispiel aus $T_1 = \{<a>, <a,i,b>\}$ und $T_2 = \{<a>, <a,i,c>\}$ mit $X_i = T_i \cup E_{false_inp}(T_i, I, O)$, für $i = 1$ oder $i = 2$, entstanden.

Es gilt:

$$InpEn(X_1, I) \wedge InpEn(X_2, I)$$

Aber:

$$<a,i> \notin X_1 \cap X_2 = \{<a>, <i>\&x, <a,i,i>\&x \mid x \in \{I \cup O\}^\omega\}$$

Es hat also, falls auf ein a die Eingabe eines i folgt, ein zweites i zu kommen. Dies ist eine Lebendigkeitsanforderung an die Umgebung. Was ist, wenn dieses zweite i nicht mehr kommt? Aus diesem Grund sollte auch die Spur $<a,i>$ in der Schnittmenge enthalten sein. □

Eine adhoc Lösung wäre die Bildung eines Abschlußes $E(X)$, der zusätzlich derartige Spuren beinhaltet, und somit in Beispiel 7.1 die Spur $<a,i>$ in die Spurmenge $E(X_1 \cap X_2)$ aufnimmt. Da aber dadurch eine neuartige Erweiterungsoperation nach der Schnittmengenbildung zu erfolgen hat, wird in dieser Arbeit sowohl die Erweiterung um fehlende, als auch um fehlerhafte Eingaben nach der Konjunktion der Teilsemantiken unter Einschränkung des Umgebungsverhaltens anstelle dieses Abschlußes verwendet. In einem ersten Ansatz ist es möglich, die folgende Definition der Kombination zu verwenden:

$$(A_{TSD_N} \wedge A_{LC_N} \wedge A_{QM_N} \wedge A_{special_N}) \Rightarrow (C_{TSD_N} \wedge C_{LC_N} \wedge C_{QM_N} \wedge C_{special_N})$$

Dies entspricht:

$$\neg A_{TSD_N} \vee \neg A_{LC_N} \vee \neg A_{QM_N} \vee \neg A_{special_N} \vee (C_{TSD_N} \wedge C_{LC_N} \wedge C_{QM_N} \wedge C_{special_N})$$

In einer Mengendarstellung verspricht dies eine einfache Lösung für die Definition der Kombination der Semantiken. Daß dies leider nicht zum gewünschtem Ergebnis führt, wird mit Beispiel 7.2 gezeigt:

Beispiel 7.2

Durch die zuvor festgelegten Definitionen der Erweiterung E_{false_inp} und $E_{missing_inp}$, die eine erstmalige Verletzung der Assumption darstellen, sowie der Formulierung der Commitment Komponente gemäß Definition 7.2, erhält man für die Schichtbeschreibung $(TSD_N, LC_N, QM_N, special_N)$:

$$
\begin{aligned}
&[\![(TSD_N, LC_N, QM_N, special_N)]\!]_{\mathcal{L}}^{(I,O)} \\
&\quad = E_{false_inp}([\![TSD_N]\!]_{\mathcal{TSD}}, I, O) \ \cup \ E_{missing_inp}([\![TSD_N]\!]_{\mathcal{TSD}}, I, O) \\
&\qquad \cup \ E_{false_inp}([\![LC_N]\!]_{\mathcal{LC}}, I, O) \\
&\qquad \cup \ E_{false_inp}([\![QM_N]\!]_{\mathcal{QM}}, I, O) \\
&\qquad \cup \ E_{false_inp}(\{t | special_N(t)\}, I, O) \ \cup \ E^{special_N}_{missing_inp}(\{t | special_N(t)\}, I, O) \\
&\qquad \cup \ [\![(TSD_N, LC_N, QM_N, special_N)]\!]_{\mathcal{L}}
\end{aligned}
$$

Hierbei ist $E^{special_N}_{missing_inp}$ die Formulierung der Verletzung des Lebendigkeitsanteils der Assumption für das schichtenspezifische Prädikat $special_N$. Da jedoch bei den obigen Erweiterungen E_{false_inp} und $E_{missing_inp}$ als Voraussetzung die Komponente die Spielregeln mindestens solange einzuhalten hat, bis die Umgebung diese verletzt, und dies bei obigem Vorschlag nicht berücksichtigt wird, ist diese Definition nicht korrekt. Sei zum Beispiel die Spur $<i, o, i>$ in $E_{false_inp}([\![LC_N]\!]_{\mathcal{LC}}, I, O)$ enthalten, aber die Spur $<i, o>$ in der Präfixmenge der Gesamtbeschreibung $[\![(TSD_N, LC_N, QM_N, special_N)]\!]_{\mathcal{L}}$ nicht. Mit der obigen Definition wäre $<i, o, i>$ in $[\![(TSD_N, LC_N, QM_N, special_N)]\!]_{\mathcal{L}}^{(I,O)}$ enthalten, was nicht der Vorstellung entspricht, da diese Spur kein erlaubtes Präfix der Gesamtbeschreibung, sondern nur ein erlaubtes Präfix einer Teilbeschreibung ist. □

Dies ist nicht weiter verwunderlich, da nicht berücksichtigt wurde, daß bis zum Zeitpunkt der Verletzung der Assumption das Commitment erfüllt sein muß. Dies wird unter Verwendung des Operators $\longrightarrow\!\!\triangleright$ aus [AL93] wie folgt dargestellt:

$$(A_{TSD_N} \wedge A_{LC_N} \wedge A_{QM_N} \wedge A_{special_N}) \longrightarrow\!\!\triangleright (C_{TSD_N} \wedge C_{LC_N} \wedge C_{QM_N} \wedge C_{special_N})$$

Die Grundidee der Vereinigung von Spurmengen wird aus Beispiel 7.2 übernommen, da damit die Kombination sehr einfach wird:

Definition 7.4

Die *Semantik einer Schichtbeschreibung* $(TSD_N, LC_N, QM_N, special_N)$ *bei freiem Umgebungsverhalten* wird für die Eingabemenge I und die Ausgabemenge O festgelegt als:

$$[\![(TSD_N, LC_N, QM_N, special_N)]\!]_{\mathcal{L}}^{(I,O)} = T_{all} \ \cup \ E_{false_inp}(T_{all}, I, O)$$

Wobei:

$$
\begin{aligned}
T_{all} = (\ & (E_{missing_inp}([\![TSD_N]\!]_{TSD}, I, O) \ \cup \ E^{special_N}_{missing_inp}(\{t | special_N(t)\}, I, O)) \\
& \cap PRE([\![(TSD_N, LC_N, QM_N, special_N)]\!]_{\mathcal{L}})) \\
& \cup [\![(TSD_N, LC_N, QM_N, special_N)]\!]_{\mathcal{L}}
\end{aligned}
$$

□

Hierbei ist $E^{special_N}_{missing_inp}$ wiederum die Darstellung einer Verletzung des Lebendigkeitsanteils der Assumption für das schichtenspezifische Prädikat $special_N$. Die Menge T_{all} beschreibt diejenigen Spuren, bei denen der Lebendigkeitsanteil der Assumption verletzt wird, vereinigt mit denjenigen Spuren, bei denen sowohl die Assumption als auch das Commitment erfüllt ist. Diese Menge wird wie folgt konstruiert: Zuerst werden die Mengen, die eine Verletzung des Lebendigkeitsanteils bei TSDs und beim schichtenspezifischen Prädikat beschreiben, vereint. Die daraus resultierende Menge wird mit der Präfixmenge der Gesamtbeschreibung $[\![(TSD_N, LC_N, QM_N, special_N)]\!]_{\mathcal{L}}$ geschnitten. Dadurch ist sichergestellt, daß der Diensterbringer bezogen auf die Gesamtbeschreibung seinen Verpflichtungen nachgekommen ist. Dies ist, wie in Beispiel 7.2 dargestellt wurde, ohne Schnittmengenbildung nicht mehr gewährleistet. Die daraus resultierende Menge wird dann mit der Spurmenge vereint, bei der sowohl die Assumption als auch das Commitment der Gesamtbeschreibung erfüllt ist. Die Verletzung des Sicherheitsanteils der Assumption der Gesamtbeschreibung wird anschließend mit $E_{false_inp}(T_{all}, I, O)$ beschrieben.

7.3 Bewertung der Formalisierung

Bei der Semantik einer Schichtbeschreibung mit Beeinflussung des Umgebungsverhaltens werden die Semantiken der Teilbeschreibungen in Definition 7.2 konjunktiv verknüpft. Dieses Verfahren ist auch intuitiv sehr naheliegend, da jede Beschreibungstechnik einen einzelnen Teilaspekt einer Schichtbeschreibung erfaßt. Dadurch ist eine klarere Trennung der Konzepte als bei den Dienstnormen erfolgt, bei denen eine Vermischung der Teilbeschreibungen vorgenommen wurde.

Die Semantik einer Schichtbeschreibung mit freiem Umgebungsverhalten kann dagegen nicht durch eine einfache Schnittmengenbildung der Semantiken der Teilbeschreibungen mit freiem Umgebungsverhalten vorgenommen werden, da durch die Schnittmengenbildung unter Umständen eine Lebendigkeitsanforderung an die Dienstbenutzer entstehen kann. Aus diesem Grund werden in dem in dieser Arbeit vorgestellten Verfahren zuerst die Semantiken der Teilbeschreibungen mit Beeinflussung des Umgebungsverhaltens konjunktiv kombiniert. Dann wird die aus der Schichtbeschreibung entstandene Spurmenge um diejenigen Spuren ergänzt, die eine Verletzung des Sicherheits- und Lebendigkeitsanteils eines Dienstbenutzers darstellen.

Mit dieser Methode ist es sehr einfach möglich, die formale Beschreibung eines Diensterbringers zu erstellen. Hierfür ist es erforderlich, daß die jeweiligen Teile der Normen in der in den Kapiteln 3, 4 und 5 vorgestellten Weise umgesetzt werden. Daran anschließend erfolgt eine Formalisierung der Eigenschaften, die von TSDs, LCs und QM nicht erfaßt sind, mit einem schichtenspezifischen Prädikat. Dieses schichtenspezifische Prädikat ist

nicht bei jeder Schicht erforderlich, da die Beschreibungstechniken TSD, LC und QM sehr mächtig sind.

Kapitel 8

Betrachtung von mehreren Verbindungen

In den dienstbeschreibenden Normen wird nur jeweils eine einzige Punkt–zu–Punkt Verbindung, die mit der in den vorigen Kapiteln vorgestellten Methode eindeutig beschrieben werden kann, betrachtet. Auch wenn es in den Normen nicht explizit bei den einzelnen Dienstbeschreibungen erwähnt wird, so ist es doch von wesentlichem Interesse, wie sich mehrere Punkt–zu–Punkt Verbindungen zu einem Netzwerk zusammensetzen. Auf diese Zusammensetzung von mehreren Verbindungen wird hauptsächlich in der Definition des Basisreferenzmodells in [ISO84a, CCI88b] durch die Einführung eines Verbindungsbegriffs eingegangen. In diesem Kapitel wird ausgehend von der Darstellung in den Normen eine Formalisierung des Konzepts beschrieben.

8.1 Informelle Beschreibung eines Netzwerks von Verbindungen

Betrachtet man den Diensterbringer einer Schicht, so steht dieser in einem offenen System einer beliebigen Anzahl von Dienstbenutzern zur Verfügung. Die jeweiligen Dienstbenutzer haben als Schnittstellen die Dienstzugangspunkte (SAPs) beim Diensterbringer. In den Dienstnormen steht deswegen eine beliebige Anzahl von SAPs zur Verfügung. Bei der Modellierung in [ISO92, ISO89b] wird sogar explizit eine unendliche Anzahl erlaubt.

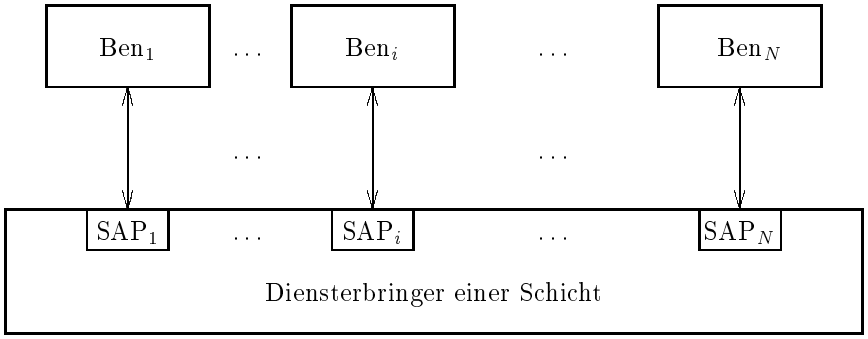

Abbildung 8.1: Diensterbringer mit mehreren Dienstbenutzern

In der folgenden Erklärung ist i aus $\{1, 2, \ldots, N\}$ zu wählen. In Abbildung 8.1 wird der Diensterbringer einer beliebigen Schicht mit den N Dienstzugangspunkten SAP$_i$, die die physikalischen Schnittstellen zu den jeweiligen Dienstbenutzern Ben$_i$ bezeichnen, dargestellt. Da bei einem Dienstzugangspunkt durch Zeitmultiplexing auch mehrere Verbindungen gleichzeitig existieren können, ist im ISO/OSI Schichtenmodell [ISO84a, CCI88b] der Begriff der *Verbindungsendpunkte (connection endpoint CEP)* eingeführt worden. Mit den Verbindungsendpunkten wird eine feinere Aufteilung der Dienstzugangspunkte nach logischen Gesichtspunkten vorgenommen, das heißt, die Verbindungsendpunkte stellen keine physikalischen Schnittstellen dar, sondern nur ein Hilfsmittel zur logischen Aufteilung von Verbindungen. Betrachtet man einen Benutzer Ben$_i$ und den zugeordneten SAP$_i$ aus Abbildung 8.1 näher, so erkennt man, daß sich hinter der oben beschriebenen Beziehung zwischen Ben$_i$ und SAP$_i$ jeweils mehrere Punkt-zu-Punkt Verbindungen verbergen können, wie dies in Abbildung 8.2 verdeutlicht wird.

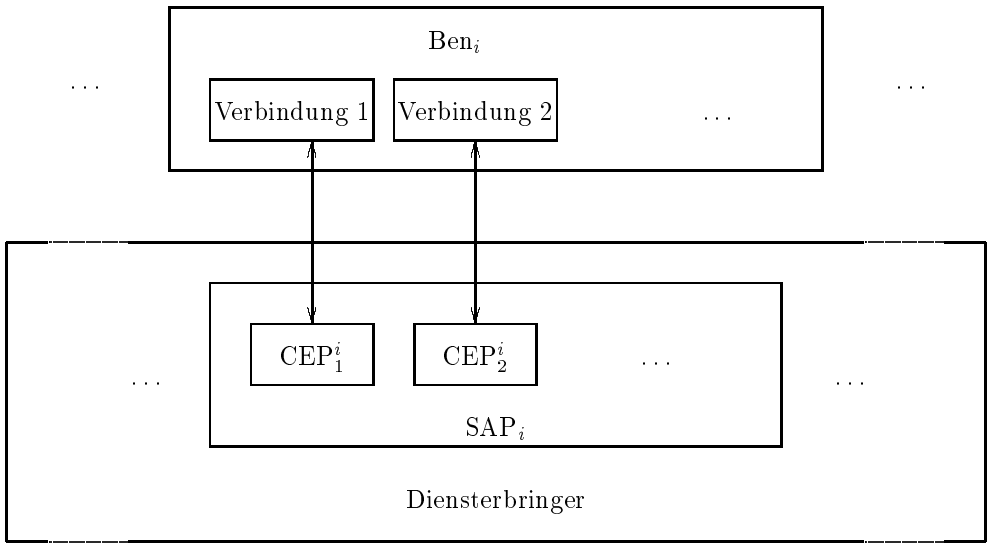

Abbildung 8.2: Aufteilung des SAP$_i$ in mehrere Verbindungsendpunkte

In Abbildung 8.2 wird die feinere Aufteilung eines SAPs in mehrere Verbindungsendpunkte CEP_1^i, CEP_2^i, ... dargestellt. Dies entspricht der detaillierteren Darstellung der Beziehung zwischen Ben_i und SAP_i aus Abbildung 8.1. Nach [ISO84a, CCI88b] ist jeder Verbindungsendpunkt durch einen eindeutigen Identifikator gekennzeichnet. Durch diese Konkretisierung läßt sich jede einzelne Verbindung in Abbildung 8.2 genau darstellen. Die Eindeutigkeit der Verbindungsendpunktidentifikatoren wird in [ISO84a, CCI88b] nicht nur innerhalb eines SAPs, sondern bezüglich aller SAPs gefordert und wird dadurch erreicht, daß der SAP-Identifikator das Präfix von jedem CEP-Identifikator ist.[1]

Die wesentliche Aufgabe dieses Kapitels ist es, die Aufteilung einer Spur, die alle Verbindungen erfaßt, in Teile dieser Spur, die jeweils eine einzige Punkt-zu-Punkt Verbindung beschreiben, darzustellen. Es werden somit aus der Mehrzahl der Verbindungsendpunkte durch eine Zuordnung zwei miteinander kommunizierende Verbindungsendpunkte bestimmt. Mittels dieser Zuordnung werden dann die jeweiligen Teilspuren aus der Gesamtbeschreibung extrahiert.

Betrachtet man die einzelnen Verbindungen, die innerhalb eines Diensterbringers möglich sind, so ist jeder CEP mit genau dem CEP des entsprechenden Kommunikationspartners verbunden. Dies wird mit Abbildung 8.3 gezeigt.

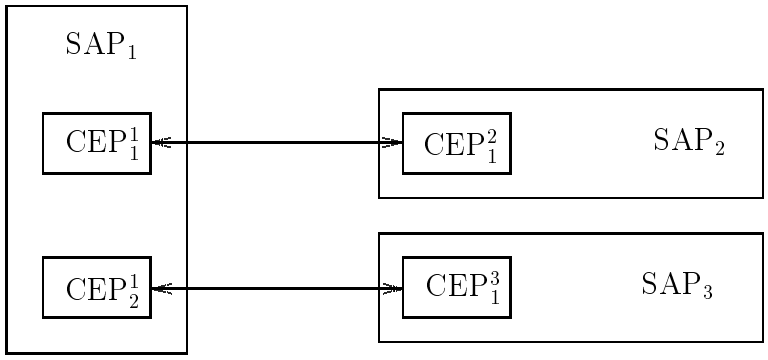

Abbildung 8.3: Zwei Punkt-zu-Punkt Verbindungen

In Abbildung 8.3 werden zwei Verbindungen innerhalb eines Diensterbringers dargestellt. Die jeweiligen Dienstbenutzer werden in dieser Abbildung nicht mehr angegeben. Sie verbergen sich hinter den Verbindungsendpunkten. Es besteht zum Beispiel eine Verbindung von SAP_1, CEP_1^1 mit SAP_2, CEP_1^2. Diese Darstellung entspricht einer logischen Sicht auf die jeweiligen Verbindungen, da als jeweiliges Zuordnungskriterium die Verbindungsendpunkte berücksichtigt werden. Würde man eine Aufteilung nach physikalischen Gesichtspunkten, das heißt nach SAPs, betrachten, so wären unter Umständen mehrere Verbindungsendpunkte zu einem SAP zusammenzufassen.

[1] Eine derartige Eindeutigkeit bezüglich aller SAPs ist nach [JA93] überflüssig, da bei diesem Verfahren der SAP-Identifikator redundant ist. Es genügt eine innerhalb der SAPs eindeutige Unterscheidung von Verbindungsendpunkten.

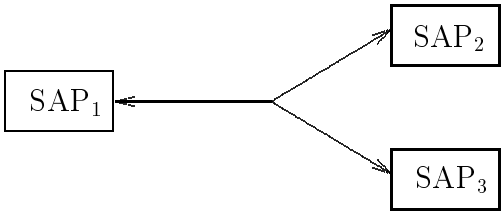

Abbildung 8.4: Abstraktion zweier Punkt-zu-Punkt Verbindungen

Abbildung 8.4 ist eine Abstraktion nach physikalischen Gesichtspunkten der Abbildung 8.3. Bei dieser Abbildung kommuniziert der SAP_1 durch Multiplexing gleichzeitig mit SAP_2 und SAP_3. Da dies der graphischen Darstellung einer Mehrpunktverbindung, also Verbindung eines Partners mit mehreren anderen Kommunikationspartnern, nahe käme und dies nicht der Realität entspricht, wird die Aufteilung in den Normen nach logischen Gesichtspunkten gewählt.

Betrachtet man die einzelnen Dienstelemente, das heißt, die in den vorigen Kapiteln beschriebenen Grundeinheiten der ausgetauschten Informationen, so ist sicherzustellen, daß der Dienstbenutzer jedes Dienstelement genau einem Verbindungsendpunkt zuordnet. Andernfalls ist eine nachträgliche Zuordnung der Dienstelemente zu einem vom Dienstbenutzer nicht gewünschten Verbindungsendpunkt möglich, da sie dann nur vom Diensterbringer vorgenommen werden kann. Eine derartige durch den Dienstbenutzer getroffene Zuordnung zu einem Verbindungsendpunkt kann man auch als Gruppierung von Dienstelementen zu einem Verbindungsendpunkt innerhalb eines SAPs bezeichnen. Bei einer realen Implementierung erfolgt dies durch das Betriebssystem, das eine eindeutige Vergabe der Verbindungsendpunktidentifikatoren sicherstellt. Ohne diese Vorgruppierung durch Verbindungsendpunktidentifikatoren wäre eine Zuordnung von Dienstelementen zu einzelnen Verbindungen nur nichtdeterministisch möglich. Da nicht jedes Dienstelement einen SAP-Identifikator für die Zieladresse besitzt, beim ISO/OSI Schichtenmodell zum Beispiel fehlt dieser bei der verbindungsorientierten Schichtenbeschreibung bei den Dienstelementen für die Datenübertragung, könnte ein Dienstbenutzer innerhalb eines SAPs keine Entscheidung treffen, ob eine bestimmte Zieladresse angesprochen werden soll. Ein Verbindungsendpunktidentifikator ist sogar bei einer expliziten Angabe einer Zieladresse notwendig, da mehrere unterscheidbare Verbindungen zwischen zwei SAPs existieren können, wie dies in Abbildung 8.4 dargestellt wird. Somit ist eine dem Dienstbenutzer zugängliche Unterscheidung in Form der zuvor erläuterten Vorgruppierung notwendig.

An folgendem Beispiel eines Austauschs von Dienstelementen zwischen Diensterbringer und Dienstbenutzer, wie er in Abbildung 8.3 vorkommen könnte, wird das Konzept des Verbindungsaufbaus verdeutlicht.

Beispiel 8.1

Ein Dienstbenutzer, der an SAP_2 einen Zugang zum Diensterbringer hat, sendet einen Verbindungsaufbauwunsch $CONreq(SAP_1)$. Durch den Parameter SAP_1 wird dem Diensterbringer mitgeteilt, daß eine Verbindung mit SAP_1 gewünscht wird. Der

Diensterbringer stellt daraufhin an SAP_1 ein $CONind$ zur Verfügung, wobei er vollkommen frei einen Verbindungsendpunkt bestimmt. Keiner der Dienstbenutzer hat auf die Auswahl der Verbindungsendpunkte bei dieser Ausgabeaktion einen Einfluß. Dieser sei in diesem Fall CEP_1^1. Somit besteht eine Verbindung von SAP_2, CEP_1^2 mit SAP_1, CEP_1^1. Die weiteren Dienstelemente, die innerhalb dieser Verbindung ausgetauscht werden, haben keinerlei Angabe über ihre Zieladresse. Sie können aber durch ihre Ursprungsadresse einer Verbindung zugeordnet werden. So wird zum Beispiel jedes $DATreq$ am SAP_2, CEP_1^2 der oben beschriebenen Verbindung zugeordnet und somit gibt der Diensterbringer an SAP_1, CEP_1^1 ein $DATind$ aus, falls dies möglich ist. □

8.2 Adreßidentifizierung der Dienstelemente

Zur Modellierung der formalen Semantik eines Diensterbringers erhält in dieser Arbeit jedes Dienstelement einen eindeutigen *Adreßidentifikator*, mit dem der physikalische und logische Ursprung eines Dienstelements, also dessen SAP und CEP, ermittelt werden kann. Dieses Verfahren wird auch bei der Spezifikation der Transportschicht [ISO92] und der Kommunikationssteuerungsschicht [ISO89b] mit LOTOS verwendet. Hierfür wird der *Dienstzugangspunktidentifikator* $SAPid$, der den jeweiligen SAP eindeutig beschreibt, und der *Verbindungsendpunktidentifikator* $CONid$, mit dessen Hilfe die jeweiligen Verbindungen innerhalb eines SAPs unterschieden werden, eingeführt. Ein *Adreßidentifikator* ist die Kombination eines SAPidentifikators mit einem Verbindungsendpunktidentifikator. Die Zusammenfassung eines Dienstelements mit einem Adreßidentifikator wird als *adressiertes Dienstelement* bezeichnet.

Definition 8.1

Ein *adressiertes Dienstelement* ist ein Tripel $(SAPid, CONid, primitiv)$, wobei $SAPid$ ein eindeutiger Dienstzugangspunktidentifikator, $CONid$ ein Verbindungsendpunktidentifikator und $primitiv$ ein Dienstelement ist. □

Mit der Komponente $SAPid$ kann der physikalische Ursprung eines adressierten Dienstelements bestimmt werden, wohingegen mit der Komponente $CONid$ die Gruppierung der adressierten Dienstelemente nach logischen Gesichtspunkten erkennbar ist. Bei einer Punkt–zu–Punkt Verbindung sind die jeweiligen Verbindungsendpunktidentifikatoren der beiden Kommunikationspartner im allgemeinen unterschiedlich, sie können aber auch gleich sein. Dies bedeutet eine größere Abstraktion, als im ISO/OSI Schichtenmodell gefordert wird. Bei einer Verschärfung des Kriteriums der Eindeutigkeit bezüglich aller SAPs ist der folgende Ansatz ohnehin erfüllt. Mit der Komponente $CONid$ erfolgt nur eine Zuordnung von adressierten Dienstelementen zu einer Verbindung bei einem Kommunikationspartner.

8.3 Formalisierung der Beschreibung von mehreren Verbindungen

Bei der Beschreibung einer beliebigen Verbindung wird davon ausgegangen, daß zu jedem Zeitpunkt ein adressiertes Dienstelement Bestandteil einer *Verbindungszuordnung* ist. Eine Verbindungszuordnung $(SAPid_A, CONid_A, SAPid_B, CONid_B)$ besteht aus der Zuordnung eines Adreßidentifikators zu einem anderen, wobei durch die Indizes $_A$ und $_B$ die jeweiligen Verbindungspartner unterschieden werden. Spezifiziert wird die Abfolge der Verbindungszuordnungen in dieser Arbeit mit einer Spur, die die Verbindungszuordnungen zu jedem Zeitpunkt beschreibt. Die jeweilige Verbindungszuordnung wird nichtdeterministisch bestimmt. Dies entspricht der Einführung von *Prophecy*-Variablen [AL88].

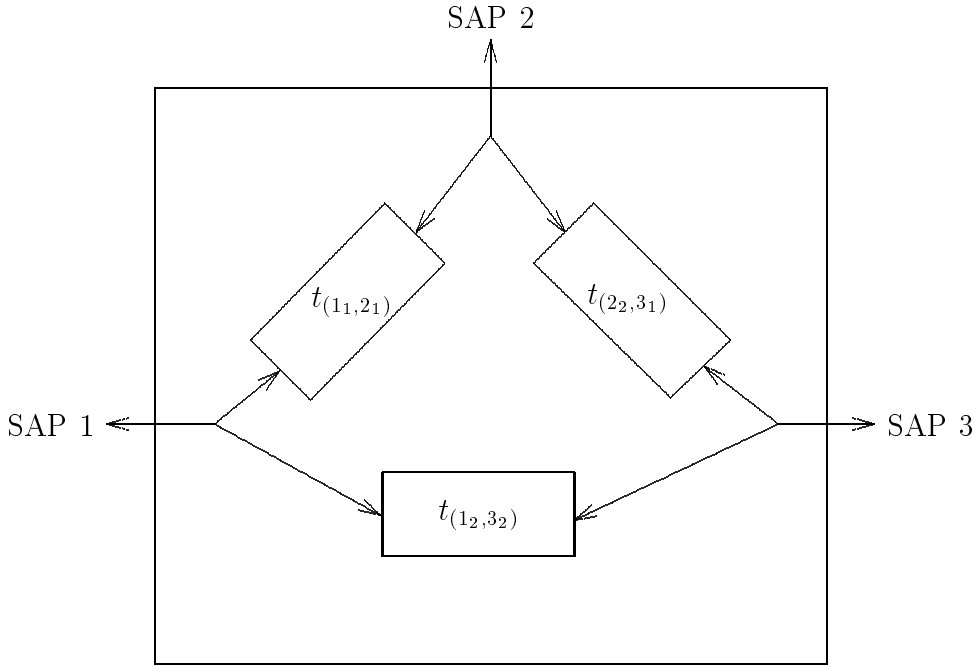

Abbildung 8.5: Aufspalten einer Mehrfachverbindung in Einzelverbindungen

In Abbildung 8.5 wird die Beschreibung mehrerer Verbindungen an einem Beispiel mit den drei SAPs 1, 2 und 3, die die physikalischen Schnittstellen beschreiben, verdeutlicht. Nach dieser Abbildung existieren zu einem Zeitpunkt 6 Verbindungsendpunkte, die die logischen Schnittstellen beschreiben, da an jedem SAP genau zwei Verbindungen dargestellt sind. Diese Darstellung entspricht der physikalische Sicht auf die Verbindungen, in deren Beschreibung zusätzlich die Spuren einbezogen werden. Eine Spur ist hierbei eine Sequenz von adressierten Dienstelementen. Im folgenden wird dargestellt, wie sich die Spur t, die alle Verbindungen beschreibt, aus den Teilspuren, die jeweils eine einzelne Punkt-zu-Punkt Verbindung beschreiben, zusammensetzt. Die jeweiligen Verbindungs-

endpunkte werden durch Indizes unterschieden. So wird der Verbindungsendpunkt 2 am SAP 1 wird 1_2 bezeichnet. Die Spur $t_{(1_1,2_1)}$ beschreibt denjenigen Ausschnitt der Spur t, der die Punkt-zu-Punkt Verbindung von 1_1 mit 2_1 erfaßt. Mit diesen Abkürzungen wird die Korrektheit aller Verbindungen einer Schicht durch das Prädikat $LayerOk_N$ dargestellt. Dieses Prädikat stützt sich auf das Prädikat zur Darstellung der Korrektheit einer einzelnen Punkt-zu-Punkt Verbindung einer Schicht $SingleConnOk_N$, das die in Kapitel 7 beschriebene Spurmenge zur Spezifikation der Schicht N charakterisiert.

$$\begin{aligned} LayerOk_N(t) \;:=\; & SingleConnOk_N(t_{(1_1,2_1)}) \\ & \wedge\; SingleConnOk_N(t_{(2_2,3_1)}) \\ & \wedge\; SingleConnOk_N(t_{(1_2,3_2)}) \end{aligned}$$

Ausgehend von dieser vereinfachten Darstellung wird das Verfahren verallgemeinert. Die Spur t wird mit einer Funktion in die Teilspuren für jeweils eine Verbindung aufgespalten. Hierzu wird die Prophecy-Variable p, eine Spur von Verbindungszuordnungen, verwendet. Daß die Prophecy-Variable p global gewählt wird, ist notwendig, da der jeweilige Verbindungspartner nur beim Verbindungsaufbau als Parameter vorhanden und somit nicht bei jedem Dienstelement verfügbar ist. Außerdem ist jedes Dienstelement genau zwei Verbindungspartnern zuzuordnen. In einem ersten Ansatz könnte die Funktion $filter((s_A, c_A, s_B, c_B), p, t)$ aus Kapitel 3.3.3 zum Herausfiltern der jeweils zu einer Verbindung gehörenden Elemente verwendet werden. Hierbei würden dann die adressierten Dienstelemente die genaue Adreßidentifizierung (s, c) durch die jeweiligen Identifikatoren besitzen. Zur Vereinfachung der Beschreibung einer Punkt-zu-Punkt Verbindung wird dies abstrahiert, indem man die Dienstelemente an beiden Verbindungsendpunkten nur durch die Indizes $_A$ und $_B$, anstatt durch den Adreßidentifikator unterscheidet. Die Bildung der Teilspuren wird durch die stetige und strikte Funktion $ExtractSingleConn$, die die zu der Verbindung (s_A, c_A, s_B, c_B) gehörenden Dienstelemente aus der Spur t gemäß der Sequenz der Verbindungszuordnungen p extrahiert, realisiert. Hierbei wird an die Dienstelemente, die zu (s_A, c_A) gehören, die SAP Bezeichnung $_A$ angefügt, um die bei einer Punkt–zu–Punkt Verbindung notwendige Unterscheidung der Kommunikationspartner zu erzielen. Äquivalent dazu erhalten die zu (s_B, c_B) gehörenden Dienstelemente den Index $_B$.

Durch die Abbildung 8.6 wird die Aufteilung von mehreren Punkt-zu-Punkt Verbindungen in einzelne Punkt-zu-Punkt Verbindungen durch einen möglichen inneren Aufbau eines Diensterbringers beschrieben.

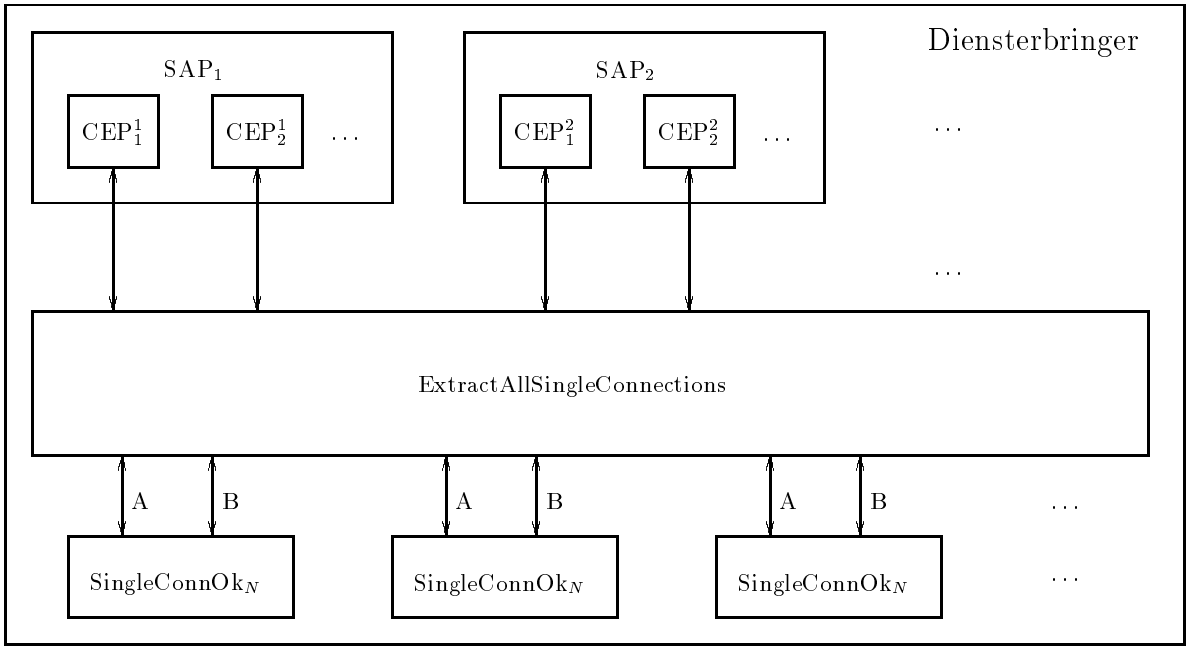

Abbildung 8.6: Aufteilung von mehreren Punkt-zu-Punkt Verbindungen innerhalb eines Diensterbringers

In Abbildung 8.6 wird dargestellt, wie die einzelnen Punkt-zu-Punkt Verbindungen jeweils aus der Beschreibung von mehreren Punkt-zu-Punkt Verbindungen innerhalb eines Diensterbringers herausgefiltert werden. Hierbei werden mit der Komponente *ExtractAllSingleConnections* jeweils die zu zwei verbundenen CEPs gehörenden Dienstelemente einer einzigen Punkt-zu-Punkt Verbindung zugeordnet. Modelliert wird die Funktionsweise der Komponente *ExtractAllSingleConnections* durch eine beliebige Anzahl von Instanzen der Funktion *ExtractSingleConn*, die im folgenden definiert wird. Die Funktion *ExtractSingleConn* filtert aus einer Spur die zu genau zwei verbunden CEPs zugeordnete Teilspur heraus:

Definition 8.2

Die Funktion *ExtractSingleConn* ist für die Adreßidentifikatoren (s_A, c_A) und (s_B, c_B), sowie die Verbindungszuordnung p, die Spur von Verbindungszuordnungen ps, das adressierte Dienstelement t und die Spur von adressierten Dienstelementen ts wie folgt definiert.

$$ps = \epsilon \vee ts = \epsilon \Rightarrow ExtractSingleConn(s_A, c_A, s_B, c_B, ps, ts) = \epsilon$$

$$ExtractSingleConn(s_A, c_A, s_B, c_B, p\&ps, (s, c, act)\&ts) =$$
$$\quad \text{if } (s_A, c_A, s_B, c_B) = p$$
$$\quad\quad \text{then if } (s, c) = (s_A, c_A)$$
$$\quad\quad\quad \text{then } Make_A(act)\&ExtractSingleConn(s_A, c_A, s_B, c_B, ps, ts)$$

$$\text{else } Make_B(act)\&ExtractSingleConn(s_A, c_A, s_B, c_B, ps, ts)$$
$$\text{else } ExtractSingleConn(s_A, c_A, s_B, c_B, ps, ts)$$

Die Funktionen $Make_A$ und $Make_B$ bilden die Dienstelemente auf die jeweils unterschiedlichen Dienstelemente der SAPs ab. So gilt zum Beispiel:

$$Make_A(DATreq) = DATreq_A$$
$$Make_B(DATreq) = DATreq_B$$

□

Somit werden nur die Dienstelemente, die zur Verbindung von (s_A, c_A) mit (s_B, c_B) aufgrund der Verbindungszuordnung ps gehören, aus der Spur ts herausgefiltert. Dabei erfolgt eine Abstraktion der Adreßidentifikatoren zu einer einfachen Ursprungsangabe, die durch das Anfügen der Indizes $_A$ und $_B$ an die Dienstelemente verdeutlicht wird.

Als nächster Punkt werden alle korrekten Sequenzen von Verbindungszuordnungen beschrieben. Dies ist notwendig, da die Prophecy-Variable p nicht völlig uneingeschränkt gewählt werden kann, sondern einige Rahmenbedingungen einhalten muß. Das Prädikat $ProphecyOk$ stellt sicher, daß ein adressiertes Dienstelement zu jedem Zeitpunkt einer Verbindung zugeordnet ist. Außerdem stellt es sicher, daß die Zuordnung von adressierten Dienstelementen zu einer Verbindung eindeutig ist, das heißt, daß während einer Verbindung ein adressiertes Dienstelement mit gleichem Adreßidentifikator zu keiner anderen Verbindung gehört. Letzteres modelliert die Vorgruppierung von Verbindungen an einem SAP mittels des Verbindungsendpunktidentifikators.

Definition 8.3

Das Prädikat $ProphecyOk$ ist für die Spur von Verbindungszuordnungen ps und die Spur von adressierten Dienstelementen ts wie folgt definiert.

$$\begin{aligned}
&ProphecyOk(ps, ts) = \\
&\quad (\forall m.\ (ps[m] = \bot \wedge ts[m] = \bot) \\
&\qquad \vee\ (ps[m] = (s_A, c_A, s_B, c_B) \wedge (\exists prim.\ (s_A, c_A, prim) = ts[m] \\
&\qquad\qquad \vee\ (s_B, c_B, prim) = ts[m]))) \\
&\wedge\ \forall s_A, c_A, s_B, c_B, k, m. \\
&\qquad (ps[k] = (s_A, c_A, s_B, c_B) \wedge ps[m] = (s_A, c_A, s_B, c_B) \wedge k < m) \\
&\qquad \Rightarrow (\forall s'_A, c'_A, s'_B, c'_B, l. \\
&\qquad\qquad (k < l < m) \wedge (s'_A, c'_A, s'_B, c'_B) = ps[l] \\
&\qquad\qquad \wedge\ ((s'_A, c'_A) = (s_A, c_A) \vee (s_B, c_B) = (s'_B, c'_B)) \\
&\qquad\qquad\quad \Rightarrow ps[k] = ps[l]))
\end{aligned}$$

□

Durch das Prädikat $ProphecyOk$ wird im ersten Teil sichergestellt, daß jedes adressierte Dienstelement einer Verbindung zugeordnet wird. Im zweiten Teil wird gefordert, daß während einer bestehenden Verbindung keiner der beiden an dieser Verbindung beteiligten Adreßidentifikatoren einer anderen Verbindung zugeordnet wird. Eine bestehende

Verbindung ist durch eine gleiche Verbindungszuordnung zu verschiedenen Zeitpunkten gekennzeichnet. Es ist zu beachten, daß die an einer bestimmte Verbindungszuordnung beteiligten Adreßidentifikatoren nach Beendigung einer Verbindung nicht neu berücksichtigt werden. Dies ist eine Folge des für die gesamte Spezifikation eindeutigen Verbindungsendpunktidentifikators. Diese Annahme ist auch bei der Spezifikation der Transportschicht [ISO92] und der Kommunikationssteuerungsschicht [ISO89b] in LOTOS getroffen worden. Eine modifizierte Beschreibung, die dies nicht berücksichtigt, wird am Ende des Kapitels in Form eines schichtenspezifischen Prädikats angegeben.

In der folgenden Definition beschreibt das Prädikat $LayerOk_N$, daß in der Spur t alle Verbindungen der Dienstspezifikation der Schicht N genügen.

Definition 8.4

Die *Korrektheit bezüglich aller Verbindungen einer Schicht N einer Spur t wird mit dem Prädikat* $LayerOk_N$ festgelegt:

$$
\begin{aligned}
&LayerOk_N(t)= \\
&\quad \exists p.\ ProphecyOk(p,t) \\
&\qquad \wedge\ \forall s_A, c_A, s_B, c_B, t_s.\ t_s = ExtractSingleConn(s_A, c_A, s_B, c_B, p, t) \\
&\qquad\qquad \Rightarrow (ConnParamOk_N(s_A, c_A, s_B, c_B, t_s) \\
&\qquad\qquad\qquad \wedge\ SingleConnOk_N(t_s))
\end{aligned}
$$

□

Die Allquantifizierung über die möglichen Verbindungszuordnungen (s_A, c_A, s_B, c_B) entspricht der Betrachtung aller möglichen Verbindungen. Somit werden alle möglichen Verbindungen untersucht und die beschreibenden Spuren an die jeweilige Instanz der Variablen t_s gebunden. Die Prädikate $ConnParamOk_N$, sowie $SingleConnOk_N$ sind schichtenspezifisch, wobei $SingleConnOk_N$ die Darstellung der im Kapitel 7 festgelegten Spurmenge, die eine korrekte Punkt–zu–Punkt Verbindung einer beliebigen Schicht beschreibt, mit einem Prädikat ist. Bei einigen Dienstelementen kommen explizit die Dienstzugangspunktidentifikatoren als Parameter vor. Dies ist zum Beispiel bei einem Verbindungsaufbauwunsch der Fall. Die Überprüfung der Korrektheit dieser Parameter wird mit dem Prädikat $ConnParamOk_N$ vorgenommen. Das ist notwendig, da die nichtdeterministische Auswahl durch die Verbindungszuordnung einer durch die Parameter der Dienstelemente festgelegten Zuordnung bezüglich der Zieladresse widersprechen kann. Dieses Prädikat ist schichtenspezifisch, da die Zieladresse nur als Parameter bei den schichtenspezifischen Dienstelementen auftreten.

Das Prädikat $LayerOk_N$ zeichnet nur dann korrekte Verbindungen aus, wenn die Verbindungsendpunktidentifikatoren eindeutig verwendet werden, das heißt, daß sie nur innerhalb einer Verbindung verwendet werden und nach Beendigung dieser Verbindung nicht wieder. Ist es möglich, daß zwei unterschiedliche, aufeinanderfolgende Verbindungen den gleichen Verbindungsendpunktidentifikator besitzen, also ist eine Wiederverwendung des Verbindungsendpunktidentifikators in anderen Verbindungen erlaubt, dann ist das Prädikat $ProphecyOk$ durch das folgende schichtenspezifische Prädikat $ProphecyOk_N$ zu

ersetzen. Hierbei wird spezifiziert, daß alle mit einem bestimmten Verbindungsendpunktidentifikator versehenen, adressierten Dienstelemente während einer Verbindung, die durch ein Verbindungsaufbaudienstelement und ein Verbindungsabbaudienstelement erkennbar ist, zu genau einer Verbindung gehören. Dazu sind schichtenspezifisch die Dienstelemente zu beschreiben, die einen Verbindungsaufbau beziehungsweise Verbindungsabbau kennzeichnen.

$$
\begin{aligned}
&ProphecyOk_N(p,t) = \\
&\quad (\forall m.\ (p[m] = \bot \wedge t[m] = \bot) \\
&\qquad \vee\ (p[m] = (s_A, c_A, s_B, c_B) \wedge (\exists prim.\ (s_A, c_A, prim) = t[m] \vee (s_B, c_B, prim) = t[m]))) \\
&\qquad \wedge \forall s_A, c_A, s_B, c_B, k. \\
&\qquad\quad (p[k] = (s_A, c_A, s_B, c_B) \wedge (t[k] = (s_A, c_A, CONreq) \vee t[k] = (s_B, c_B, CONreq)) \\
&\qquad\qquad \Rightarrow \exists m.\ m \in PositionDisind(s_A, c_A, s_B, c_B, t) \wedge k < m \\
&\qquad\qquad\qquad \wedge \forall s'_A, c'_A, s'_B, c'_B, l.\ (k < l < m \wedge (s'_A, c'_A, s'_B, c'_B) = p[l] \\
&\qquad\qquad\qquad\qquad \wedge ((s'_A, c'_A) = (s_A, c_A) \vee (s_B, c_B) = (s'_B, c'_B))) \\
&\qquad\qquad\qquad\qquad \Rightarrow p[k] = p[l]))
\end{aligned}
$$

$$
\begin{aligned}
&m \in PositionDisind(s_A, c_A, s_B, c_B, t, k) \Leftrightarrow \\
&\qquad (t[m] = (s_A, c_A, DISind) \vee t[m] = (s_B, c_B, DISind) \vee m = \infty)
\end{aligned}
$$

8.4 Bewertung der Formalisierung

Durch die Verwendung der in diesem Kapitel eingeführten Formalisierung lassen sich mehrere Punkt-zu-Punkt Verbindungen mit Spurmengen beschreiben. Hierzu wird mit einer Prophecy-Variablen gemäß einer Zuordnung von Verbindungsendpunkten eine Teilspur erzeugt, die genau eine Verbindung beschreibt. Eine derartige Beschreibung erfolgt in den jeweiligen Dienstnormen nur in äußerst geringem Umfang. Durch dieses Kapitel wird der Aspekt von mehreren Verbindungen in Ergänzung zu den Normen eindeutig festgelegt.

Kapitel 9

Zusammenfassung und Ausblick

In diesem Kapitel werden die wesentlichen Ergebnisse zusammengefaßt und deren Bezug zu anderen Arbeiten hergestellt. Daran anschließend werden die durch diese Dissertation angeregten, zukünftigen Aktivitäten angeführt.

9.1 Zusammenfassung und Einordnung der wichtigsten Ergebnisse

Mit der in dieser Arbeit vorgestellten Methodik ist eine formale Beschreibung der Dienstnormen des ISO/OSI Schichtenmodells problemlos durchführbar. Durch die isolierte Betrachtung der einzelnen Beschreibungstechniken ist eine modulare Vorgehensweise bei der formalen Spezifikation eines Diensterbringers im Gegensatz zu den bisherigen Verfahren möglich geworden. Aus der Modularisierung resultiert auch ein hoher Wiederverwendungsgrad bei den jeweiligen in den Normen informell angegebenen Dienstspezifikationen, die dadurch im Gegensatz zu den monolithischen Verfahren [ISO89b, ISO92] mit geringem Aufwand in eine formale Spezifikation umgewandelt werden können.

Im Rahmen der modularen Vorgehensweise sind die in den Normen verwendeten Beschreibungstechniken für TSDs, LC und QM in dieser Arbeit erstmalig unabhängig formal betrachtet worden. In den Dienstnormen wird die Interpretation dieser Beschreibungstechniken nicht klar genug getrennt. Damit erfolgt eine Vermischung der jeweiligen Konzepte. Dies ist daran ersichtlich, daß zum Beispiel nach den Dienstnormen bei der Beschreibung der erlaubten Abfolgen mit TSDs diesen keine exakte Bedeutung zugewiesen wird. Es erfolgt nur ein Hinweis, daß die jeweiligen Beschreibungsteile den genauen Sachverhalt nicht widerspiegeln. Eine derartige Vorgehensweise ist auch bei anderen Teilen der Dienstnormen zu beobachten. Zur Festlegung einer eindeutigen Semantik wird in dieser Arbeit eine getrennte Betrachtung der einzelnen Beschreibungstechniken durchgeführt. Jeder einzelnen dieser Beschreibungstechniken wird ein exakte Semantik zugewiesen. Somit beschreibt jede einen Aspekt der Dienstnormen in eindeutiger Weise mittels einer exakten Interpretation. Die Semantiken der Teilbeschreibungen werden mit einer genau definierten Operation kombiniert. Dadurch erfolgt eine exakte, im Gegensatz zu der in den Dienstnormen vorhandenen vagen, Beschreibung der Eigenschaften eines Diensterbringers.

Durch die Entwicklung der formalen Semantik sind einige Ungereimtheiten bei den Beschreibungstechniken aufgedeckt und behoben worden. So erwiesen sich die beiden in [CCI88d] als äquivalent nahegelegten Stile zur Darstellung von TSDs als unterschiedlich mächtig. Der in [ISO87b] etwas vorsichtig als "logisch korrekte Darstellung" bezeichnete Stil ist nach Kapitel 3 mächtiger. Die erstmalig in dieser Arbeit belegte größere Mächtigkeit liefert die notwendige Argumentation für die Verwendung der "logisch korrekten Darstellung". Mit der bei der Semantik für TSDs eingeführten Prozeßalgebra lassen sich einige Eigenschaften eines Diensterbringers, im Gegensatz zu anderen Prozeßalgebren, in einer besonders abstrakten Form beschreiben. Dadurch wird die von [VS87] für eine Implementierung geforderte möglichst hohe Abstraktheit einer Dienstspezifikation erfüllt. Außerdem sind die TSDs erstmalig dahingehend erweitert worden, daß die Parameter bei den Dienstelementen berücksichtigt werden. Erst dadurch ist eine exakte Beschreibung der Datenübertragung möglich geworden. Ebenfalls ist durch die Einführung eines Unterbrechungsoperators bei TSDs ein vorzeitiger Abbruch einer Verbindung mit TSDs beschreibbar geworden. Bei der Spezifikation der lokalen Einschränkungen in Kapitel 4 durch Automaten und Tabellen erwies sich die Automatenmethode als die mächtigere. Desweiteren wurde durch diese Arbeit die Notwendigkeit der Einführung von Startaktionen, die in den Dienstnormen nicht vorgenommen wurde, bei den Tabellen ersichtlich. Die Einführung von Objekttabellen beim Queuemodell im Kapitel 5 ermöglichte die Formalisierung dieser Beschreibungstechnik, da der sich damit beschäftigende Teil der Normen fast nur aus Text besteht und somit nicht schematisch zur Bildung einer formalen Semantik herangezogen werden kann. Die Objekttabellen sind aber nicht nur bei der Formalisierung verwendbar, sondern auch beim informellen Gebrauch des Queuemodells, da sie auf leicht verständliche Weise erhöhte Klarheit schaffen.

Durch den verwendeten Formalismus ist, im Gegensatz zu den mit LOTOS, Estelle und SDL erstellten Spezifikationen, eine besonders hohe Abstraktion der Beschreibungen erreicht worden. Dies ist unter andererem an der Behandlung von unendlichen Abläufen erkennbar.

Basierend auf den jeweiligen formalen Semantiken der Beschreibungstechniken konnten diese bezüglich der Ausdrucksmächtigkeit verglichen werden. Als Ergebnis dieses Vergleichs wird in dieser Arbeit erstmalig festgestellt, daß jede der Beschreibungstechniken nötig ist, um spezielle Eigenschaften eines Diensterbringers darzustellen.

Bei den bei Rechnernetzen verwendeten Spezifikationsformalismen wird keine spezielle Unterscheidung von Eingabe- und Ausgabeaktionen vorgenommen und daher mit der jeweiligen Technik bei der Semantikbeschreibung das Verhalten der Dienstbenutzer, die nicht implementiert werden sollen, eingeschränkt. Dies ist aber unter anderem nach [AL90] in jedem Fall zu vermeiden. Deshalb wird in dieser Arbeit, im Gegensatz zu den meisten aus der Literatur bekannten Verfahren, das Verhalten der Dienstbenutzer mittels einer geeigneten Erweiterung der Semantik nicht mehr eingeschränkt. Mit dem freien Verhalten der Dienstbenutzer ist eine notwendige Voraussetzung zur Implementierung der Diensterbringer geschaffen worden. Außerdem ist durch die Angabe einer Semantik mit freiem Umgebungsverhalten die exakte und sinnvolle Interpretation einiger TSDs erst möglich geworden. Zudem hat sich durch diese Arbeit herausgestellt, daß eine auf einfachen Spurmengen basierende Semantik von TSDs mit freiem Umgebungsverhalten nicht mehr kompositional ist. Dies konnte durch eine Erweiterung der Semantik auf ein Tupel

von Spurmengen problemlos beseitigt werden.

9.2 Zukünftige Aktivitäten

Mit der beschriebenen Methodik ist die Spezifikation eines Diensterbringers im ISO/OSI Schichtenmodell auf einem abstrakten Niveau möglich. Somit ist der Ausgangspunkt für die Entwicklung eines Diensterbringers festgelegt worden. Es stellt sich hierbei die Frage, wie die Implementierung eines Diensterbringers zu erfolgen hat. Als Ausgangspunkt für solche Untersuchungen kann zum Beispiel die Spezifikationsmethode Focus [BDD+93] verwendet werden. Hierbei ist die Adaption eines Verfeinerungsbegriffs auf das ISO/OSI Schichtenmodell zu untersuchen. Ein weiterer offener Punkt im Zusammenhang mit der Implementierung eines Diensterbringers ist die Entwicklung von Beschreibungstechniken für die Protokollnormen. Hierzu sollte der Diensterbringer in zwei deterministische Automaten, die Protokollautomaten, aufgeteilt werden, deren Beschreibung mit gängigen Verfahren sehr einfach durchzuführen ist. Der Nichtdeterminismus des Diensterbringers wird dabei auf das von den Protokollautomaten benutzte Medium verlagert, das wiederum durch die in dieser Arbeit festgelegte Methodik spezifiziert werden kann, da es den Diensterbringer der nächstniedrigeren Schicht darstellt. Ebenfalls einfach vorzunehmen ist die Beschreibung des Aufbaus von Protokolldaten.

In dieser Arbeit wird die Güte eines Dienstes nur als Parameter bei der Aushandlung der Dienstgüte beim Verbindungsaufbau berücksichtigt. Die Einhaltung der Dienstgüte wird durch die vorgestellten Methoden während einer bestehenden Verbindung nicht sichergestellt, da derartige Überlegungen nicht Bestandteil der Dienstnormen sind. So wird zum Beispiel in den Normen nicht eindeutig festgelegt, was bei einem Nichteinhalten der Bedingung geschieht. Unter diese Eigenschaften fallen zum Beispiel die exakte Betrachtung von Fehlerraten und maximalen Verzögerungen. Eine Berücksichtigung dieser Eigenschaften bei einer formalen Spezifikation eines Diensterbringers erfordert die Einführung eines quantitativen Zeitbegriffs, da sonst einige Größen, wie zum Beispiel die maximalen Verzögerungen, nicht erfaßbar sind. Nach der Einführung eines quantitativen Zeitbegriffs läßt sich das Einhalten der Dienstgüte einfach spezifizieren. Jedoch sind dann die Dienstnormen geeignet zu interpretieren.

Die in dieser Arbeit vorgestellte Methode ist zur formalen Spezifikation der Diensterbringer des ISO/OSI Schichtenmodells entworfen worden. Eine Untersuchung der Verwendbarkeit dieser Methodik bei neuartigeren Dienstbeschreibungen, wie zum Beispiel bei auf ATM [LB92, MS95] basierenden Diensten, ist mit den dafür zur Zeit entstehenden Standards, über die in [KSM94] ein Überblick gegeben wird, durchzuführen.

Ein weiterer Untersuchungspunkt ist die Anwendung der Ergebnisse zur Bildung einer Semantik mit freiem Umgebungsverhalten bei anderen Beschreibungstechniken. Hierfür bieten sich MSC [CCI92], LOTOS [ISO89a], Estelle [ISO87a] und SDL [CCI88a] an. Dazu ist die Semantik der jeweiligen Beschreibungstechniken dahingehend zu erweitern, daß ein freies Verhalten der Dienstbenutzer möglich ist. Dies kann entweder durch die Anpassung der für Spurmengen präsentierten Verfahren auf die jeweiligen Semantiken, oder durch die einfacher zu realisierende Abbildung der jeweiligen Semantiken auf Spurmengen mit der anschließenden Verwendung der in dieser Arbeit dargestellten Verfahren erfolgen.

Literaturverzeichnis

[AF82] S. Alfonzetti and A. Faro. A fomalization of the session protocol. In IEEE, editor, *Proceedings Computer Networking Symposium (dec 10 1982)*, pages 85–90. IEEE, 1982.

[AHU74] Alfred V. Aho, John E. Hopcraft, and Jeffrey D. Ullman. *The design and analysis of computer algorithms*. Addison–Wesley, 1974.

[AL88] Martin Abadi and Leslie Lamport. The existence of refinement mappings. Technical Report 29, Digital System Research Center, 1988.

[AL90] Martin Abadi and Leslie Lamport. Composing specifications. Technical Report 66, Digital System Research Center, October 1990.

[AL93] Martin Abadi and Leslie Lamport. Conjoining specifications. Technical Report 118, Digital System Research Center, December 1993.

[AS87] Bowen Alpern and Fred B. Schneider. Recognizing saftey and liveness. *Distributed Computing*, 2(1):117–126, 1987.

[AU72] Alfred V. Aho and Jeffrey D. Ullman. *The Theory of Parsing, Translation, and Compiling (Volume 1: Parsing)*. Prentice–Hall, 1972.

[BB87] Tommaso Bolognesi and Ed. Brinksma. Introduction to the ISO specification language LOTOS. *Computer Networks and ISDN Systems*, 14(1):25–60, 1987.

[BD87] S. Budkowski and P. Dembinski. An introduction to Estelle: A specification language for distributed systems. *Computer Networks and ISDN Systems*, 14(1):1–24, 1987.

[BDD+93] M. Broy, F. Dederichs, C. Dendorfer, M. Fuchs, T. F. Gritzner, and R. Weber. The design of distributed systems — an introduction to FOCUS — REVISED VERSION. SFB-Bericht 342/2/92 A TUM-I9202-2, Technische Universität München, January 1993.

[BDDW91] Manfred Broy, Frank Dederichs, Claus Dendorfer, and Rainer Weber. Characterizing the behaviour of reactive systems by trace sets. SFB-Bericht 342/2/91 A TUM-I9102, Technische Universität München, 1991.

[BFG+93] Manfred Broy, Christian Facchi, Radu Grosu, Rudi Hettler, Heinrich Hussman, Dieter Nazareth, Franz Regensburger, Oscar Slotosch, and Ketil Stølen. The requirement and design specification language SPECTRUM: An informal introduction version 1.0 (part I and II). Technical Report TUM-I9311 and TUM-I9312, Technische Universität München, 1993.

[BG86] G. v. Bochmann and R. Gotzhein. Deriving protocol specifications from service specifications. In ACM, editor, *Communications Architectures and Protocols*, pages 148–157. ACM, 1986.

[BH88] Ferenc Belina and Dieter Hogrefe. The CCITT-specification and description language SDL. *Computer Networks and ISDN Systems*, 16(4):311–343, 1988.

[BHS91] Ferenc Belina, Dieter Hogrefe, and Amardeo Sarma. *SDL with Applications from Protocol Specification*. Prentice Hall, 1991.

[BK85] E. Brinksma and G. Karjoth. A specification of the OSI transport service in LOTOS. In Yechiam Yemini, Robert Strom, and Shaula Yemini, editors, *Protocol Specification, Testing and Verification, IV*, pages 227–253. North-Holland, 1985.

[Bla91] Uyless Black. *OSI: A Model for Computer Communications Standards*. Prentice-Hall, 1991.

[BM94] J. C. M. Baeten and S. Mauw. Delayed choice: an operator for joining Message Sequence Charts. In Dieter Hogrefe and Stefan Leue, editors, *Participant's Proceedings of the seventh international Conference on Formal Description Techniques (FORTE 94)*, pages 327–341, 1994.

[Boc90a] Gregor v. Bochmann. Protocol specification for OSI. *Computer Networks and ISDN Systems*, 18:167–184, 1990.

[Boc90b] Gregor v. Bochmann. Specifications of a simplified transport protocol using different formal description techniques. *Computer Networks and ISDN Systems*, 18:335–377, 1990.

[Bri86a] E. Brinksma. On the formal specification of OSI services and protocols. In P.J. Kuehn, editor, *New Communication Services: A Challenge To Computer Technology*, pages 159–165. North-Holland, 1986.

[Bri86b] E. Brinksma. A tutorial on LOTOS. In Michel Diaz, editor, *Protocol Specification, Testing, and Verification, V*, pages 171–195. North-Holland, 1986.

[Bro92] Manfred Broy. Compositional refinement of interactive systems. Technical Report 89, Digital System Research Center, 1992.

[Bro94] Manfred Broy. A functional rephrasing of the assumption/commitment specification style. Technical Report TUM-I9417, Technische Universität München, 1994.

[BS86] Ed Brinksma and Giuseppe Scollo. Formal notations of implementation and conformance in LOTOS. Technical Report INF-86-13, Universiteit Twente, 1986.

[BS94] Manfred Broy and Ketil Stølen. Specification and refinement of finite dataflow networks – a relational approach. Technical Report SFB-Bericht Nr. 342/7/94 A, Technische Universität München, 1994.

[BW90] J.C.M. Baeten and W.P. Weijland. *Process Algebra*, volume Cambridge Tracts in Theoretical Computer Science 18. Cambridge University Press, 1990.

[CCI88a] CCITT. Functional specification and description language (SDL). Blue Book, Recommendation Z.100, November 1988.

[CCI88b] CCITT. X.200, reference model of Open System Interconnection for CCITT applications. Blue Book, FASCICLE VIII.4, Recommendations X.200-X.219, November 1988.

[CCI88c] CCITT. X.208, specification of abstract syntax notation one (ASN.1). Blue Book, FASCICLE VIII.4, Recommendations X.200-X.219, November 1988.

[CCI88d] CCITT. X.210, Open System Interconnection layer service definition conventions. Blue Book, FASCICLE VIII.4, Recommendations X.200-X.219, November 1988.

[CCI88e] CCITT. X.211, physical service definition of open system interconnection for CCITT applications. Blue Book, FASCICLE VIII.4, Recommendations X.200-X.219, November 1988.

[CCI88f] CCITT. X.212, data link service definition of open system interconnection for CCITT applications. Blue Book, FASCICLE VIII.4, Recommendations X.200-X.219, November 1988.

[CCI88g] CCITT. X.213, network service definition of open system interconnection for CCITT applications. Blue Book, FASCICLE VIII.4, Recommendations X.200-X.219, November 1988.

[CCI88h] CCITT. X.214, transport service definition for Open System Interconnection for CCITT applications. Blue Book, FASCICLE VIII.4, Recommendations X.200-X.219, November 1988.

[CCI88i] CCITT. X.215, session service definition for Open System Interconnection for CCITT applications. Blue Book, FASCICLE VIII.4, Recommendations X.200-X.219, November 1988.

[CCI88j] CCITT. X.216, presentation service definition for Open System Interconnection for CCITT applications. Blue Book, FASCICLE VIII.4, Recommendations X.200-X.219, November 1988.

[CCI92] CCITT. Message sequence chart (MSC). Recommendation Z.120, November 1992.

[Cyp91] R. J. Cypser. *Communications for cooperating systems: OSI, SNA, and TCP/IP*. Addison-Wesley, 1991.

[dR85] W. P. de Roever. The quest for compositionatility. In F. J. Neuhold and G. Chroust, editors, *Formal models in programming, Proceedings IFIP 85*, pages 181–205. Elsevier Science Publishers B. V. (North-Holland), 1985.

[EGL89] H.-D. Ehrich, M. Gogolla, and U. W. Lipeck. *Algebraische Spezifikation abstrakter Datentypen*. B.G. Teubner, 1989.

[EM85] H Ehrig and B. Mahr. *Fundamentals of Algebraic Specification 1*. Number 6 in EATCS Monographs on Theoretical Computer Science. Springer-Verlag, 1985.

[EM90] H Ehrig and B. Mahr. *Fundamentals of Algebraic Specification 2: Module Specifications and Constraints*. Number 20 in EATCS Monographs on Theoretical Computer Science. Springer-Verlag, 1990.

[ES91] Peter van Eijk and Jeroen Schot. An exercise in protocol synthesis. Technical Report TIOS 91/015, Universiteit Twente, 1991.

[FO85] W.S. Ford and C.J. O'neil. Design and specification of an OSI network interface protocol. In J.M. Bennett and T. Pearcey, editors, *The New World of the Information Society*, pages 596–602. North-Holland, 1985.

[Gal86] J. Gallier. *Logic for Computer Science*. Harper and Row, 1986.

[GGE+85] E. Giese, K. Görgen, Hinsch E., G. Schulze, and Truöl K. *Dienste und Protokolle in Kommunikationssystemen*. Springer-Verlag, 1985.

[Got90] Reinhard Gotzhein. Specifying communication services with temporal logic. In L. Logrippo, R.L. Probert, and H. Ural, editors, *Protocol Specification ,Testing and Verification, X*, pages 295–309. North-Holland, 1990.

[Gun92] Carl A. Gunter. *Semantics of Programming Languages: Structures and Techniques*. Foundations of Computing. The MIT Press, 1992.

[Hal88] Fred Halsall. *Data Communications, Computer Networks and OSI*. Addison-Wesley, 1988.

[Hoa85] C.A.R. Hoare. *Communicating Sequential Processes*. Prentice Hall, 1985.

[Hog89] Dieter Hogrefe. *Estelle, LOTOS und SDL*. Springer-Verlag, 1989.

[Hog91] Dieter Hogrefe. OSI formal specification case study: the Inres protocol and service. Technical Report IAM-91-012, Universität Bern, 1991.

[Hol91] Gerard J. Holzmann. *Design and Validation of Computer Protocols*. Prentice Hall, 1991.

[ISO84a] ISO. ISO 7498: Information processing systems - open systems interconnection - basic reference model, 1984.

[ISO84b] ISO. ISO 8326: Information processing systems - open systems interconnection - basic connection oriented session service definition, 1984.

[ISO84c] ISO. ISO DP 8072: Information processing systems - open systems interconnection - transport service definition, 1984.

[ISO87a] ISO. Estelle, a formal description technique based on an extended state transition model. ISO Draft International Standard DIS9074, September 1987.

[ISO87b] ISO. Information processing systems - Open Systems Interconnection - service conventions. Technical Report ISO TR 8509, ISO, 1987.

[ISO87c] ISO. ISO 8348: Information processing systems - open systems interconnection - network service definition, 1987.

[ISO88a] ISO. ISO 8822: Information processing systems - open systems interconnection - connection oriented presentation service definition, 1988.

[ISO88b] ISO. ISO 8886: Information processing systems - data communication - data link service definition, 1988.

[ISO88c] ISO. ISO DP 10022: Information processing systems - open systems interconnection - physical layer service definition, 1988.

[ISO89a] ISO. LOTOS, a formal description technique based on the temporal ordering of observational behaviour. ISO International Standard 8807, February 1989.

[ISO89b] ISO/IEC. Information technology - Open System Interconection - LOTOS description of the session service. Technical Report ISO/IEC/TR 9571, International Organization for Standardization Geneva, 1989.

[ISO90] ISO. ISO 8822: Information processing systems - open systems interconnection - specification of abstract syntax notation one (ASN.1), 1990.

[ISO91a] ISO. Revised text of CD 10731, information technology - Open Systems Interconnection - conventions for the definition of OSI services. Technical Report ISO/IEC JTC 1/SC 21 N 6341, ISO, 1991.

[ISO91b] ISO/IEC. Information technology - Open System Interconection - guidelines for the application of Estelle, LOTOS and SDL. Technical Report ISO/IEC/TR 10167, International Organization for Standardization Geneva, 1991.

[ISO92] ISO/IEC. Information technology - telecommunications and information exchange between systems - formal description of ISO 8072 in LOTOS. Technical Report ISO/IEC/TR 10023, International Organization for Standardization Geneva, 1992.

[ISO94] ISO. Final DIS text of ISO/IEC 10731, information technology - Open Systems Interconnection - conventions for the definition of OSI services. Technical Report ISO/IEC JTC 1/SC 21 N 8604, ISO, 1994.

[JA90] Bijendra N. Jain and Ashok K. Agrawala. *Open System Interconnection: Its Architecture and Protocols*. Elsevier, 1990.

[JA93] Bijendra N. Jain and Ashok K. Agrawala. *Open System Interconnection: Its Architecture and Protocols – Revised Edition*. McGraw-Hill, 1993.

[Jon83] C. B. Jones. Specification and design of (parallel) programs. In R. E. A. Mason, editor, *Proceedings Information Processing 83*, pages 321–331. North-Holland, 1983.

[Jon90] B. Jonsson. On decomposing and refining specifications of distributed systems. In *Proceeedings REX Workshop on Stepwise Refinement of Distributed Systems*, number 430 in Lecture Notes in Computer Science, 1990.

[KBK89] F. Khendek, G. Bochmann, and Christian Kant. New results on deriving protocol specifications from service specifications. In *Communications Architectures and Protocols*, volume 19 of *Computer Communication Review*, pages 136–146. ACM, 1989.

[Ker92] Helmut Kerner. *Rechnernetze nach OSI*. Addison-Wesley, 1992.

[KSM94] Jaime Bae Kim, Tatsuya Suda, and Yoshimura Masaaki. International standardization of B–ISDN. *Computer Networks and ISDN Systems*, 27:5–27, 1994.

[LB92] Jean-Yves Le Boudec. The asynchronous transfer mode: a tutorial. *Computer Networks and ISDN Systems*, 24:279–309, 1992.

[Lin83] Peter F. Linington. Fundamentals of the layer service definitions and protocol specifications. *Proceedings of the IEEE*, 71(12):1341–1345, 1983.

[LT89] Nancy Lynch and Mark Tuttle. An introduction to input/output automata. *CWI Quarterly*, 2(3):219–246, 1989.

[MS95] David E. McDysan and Darren L. Spohn. *ATM: Theory and application*. McGraw–Hill Series on Computer Communication. McGraw–Hill, 1995.

[Pau87] L.C. Paulson. *Logic and Computation, Interactive proof with Cambridge LCF*, volume 2 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1987.

[Pep90] Peter Pepper. Development of communication protocols by transforming temporal specifications. Unpublished draft, 1990.

[Ros90] Marshall T. Rose. *The Open Book: A Practical Perspective on OSI*. Prentice Hall, 1990.

[San88] Don Sannella. A survey of formal software development methods. Technical Report ECS-LFCS-88-56, University of Edinburgh, 1988.

[SCB91] Deepinder Sidhu, Anthony Chang, and Thomas P. Blumer. Experience with formal methods in protocol development. *Computer Communication Review ACM SIGCOMM*, 21(2):81–101, 1991.

[SDW93] Ketil Stølen, Frank Dederichs, and Rainer Weber. Assumption/commitment rules for networks of asynchronously communicating agents. Technical Report TUM-I9303, SFB-Bericht Nr. 342/2/93 A, Technische Universität München, 1993.

[SFA80] S. Schindler, U. Flasche, and D. Altenkrueger. The OSA project formal specification of the ISO transport service. In IEEE, editor, *Processings Computer Networking Symposium (dec 10 1980)*, pages 136–157. IEEE, 1980.

[SGM90] Terry Stroup, Norbert Götz, and Michael Mendler. Stepwise refinement of layered protocols by formal program development. In E. Brinksma, G. Scollo, and C. A. Vissers, editors, *Protocol Specification Testing,and Verification, IX*, pages 71–85. North-Holland, 1990.

[SM85] G. Scollo and F. Minissale. On the specification in LOTOS of OSI protocols. In Giacomo Bucci and Giorgio Valle, editors, *Computing 85 A Broad Perspective of Current Developments*, pages 197–207. North-Holland, 1985.

[SP91] Kassem Saleh and Robert Probert. Automatic synthesis of protocol specifications from service specifications. In *10th International Phoenix Conference on Computers and Communications*. IEEE, 1991.

[SPLB86] G. Scollo, G. Pappalardo, L. Logrippo, and E. Brinksma. The OSI transport service and its formal description in LOTOS. In Laszlo Csaba, Katie Tarnay, and Tibor Szentivanyi, editors, *Computer Network Usage: Recent Experiences*, pages 465–489. North-Holland, 1986.

[Sta90] William Stallings. *Handbook of Computer Communication Standards*, volume 1. Howard W. Sams & Company, 2nd edition edition, 1990.

[Tan88] Andrew S. Tanenbaum. *Computer Networks (second Edition)*. Prentice Hall, 1988.

[Tho90] Wolfgang Thomas. Automata on infinite objects. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, pages 135–191, 1990.

[Tur93] Kenneth J. Turner, editor. *Using Formal Description Techniques.* John Wiley & Sons, 1993.

[UP86] Hasan Ural and Robert L. Probert. Step-wise validation of communication protocols and services. *Computer Networks and ISDN Systems*, 11:183–202, 1986.

[Vis88] C.A. Vissers. FDTs for distributed systems. In R. Speth, editor, *Research Into Networks and Distributed Applications*, pages 39–49. North-Holland, 1988.

[Vis93] Chris A. Vissers. Private communication. 1993.

[VL86] Chris A. Vissers and Luigi Logrippo. The importance of the service concept in the design of data communication protocols. In *Protocol Specification, Testing, and Verification*, volume V, pages 3–17, 1986.

[VS87] C.A. Vissers and G. Scollo. Formal specification in OSI. In Günter Müller and Robert P. Blanc, editors, *Networking in Open Systems*, volume 248 of *Lecture Notes in Computer Science*, pages 338–360. Springer-Verlag, 1987.

[VSvSB91] Chriss A. Vissers, Giuseppe Scollo, Marten van Sinderen, and Ed Brinksma. Specification styles in distributed systems design and verification. *Theoretical Computer Science*, 89:179–206, 1991.

[Web92] Rainer Weber. Eine Methodik für die formale Anforderungsspezifikation verteilter Systeme. SFB-Bericht 342/13/92 A TUM-I9219, Technische Universität München, 1992.

[Wec92] Wolfgang Wechler. *Universal Algebra for Computer Scientists.* Number 25 in EATCS Monographs on Theoretical Computer Science. Springer-Verlag, 1992.

[Zim80] Hubert Zimmermann. OSI reference model - the ISO model of architecture for open systems interconnection. In Craig Patridge, editor, *Innovations in Internetworking*, 1980.

[Zwi89] J. Zwiers. *Compositionality, Concurrency and Partial Correctness: Proof Theories for Networks of Processes and their Relationship.* Number 321 in Lecture Notes in Computer Science. Springer, 1989.