TECHNISCHE UNIVERSITÄT MÜNCHEN

INSTITUT FÜR INFORMATIK

Sonderforschungsbereich 342:
Methoden und Werkzeuge für die Nutzung
paralleler Rechnerarchitekturen

Home Shopping - Die Spezifikation einer Kommunikationsanwendung in Focus

Ursula Hinkel

TUM-I9808
SFB-Bericht Nr. 342/02/98 A
April 98

TUM–INFO–04-I9808-100/1.–FI

Anforderungen an: Prof. Dr. A. Bode
Sprecher SFB 342
Institut für Informatik
Technische Universität München
D-80290 München, Germany

Druck: Fakultät für Informatik der
Technischen Universität München

Home Shopping – Die Spezifikation einer Kommunikationsanwendung in FOCUS*

Ursula Hinkel

Institut für Informatik
Technische Universität München
hinkel@in.tum.de

April 1998

Zusammenfassung

Am Beispiel der Kommunikationsanwendung Home Shopping wird die Entwicklung einer Spezifikation im Rahmen der formalen Entwicklungsmethodik FOCUS vorgestellt. Ausgehend von einer gegebenen informellen Beschreibung des Systems wird schrittweise eine formale Spezifikation erstellt, wobei insbesondere graphische und tabellarische Beschreibungstechniken verwendet werden. Anhand der Fallstudie wird untersucht, inwieweit sich die vorhandenen Modellierungs- und Beschreibungstechniken von FOCUS für die Spezifikation der Dienststeuerung von Multimediadiensten eignen.

1 Motivation

Basierend auf der informellen Beschreibung des Home Shopping Systems in [KS98] wird in diesem Bericht eine formale Spezifikation des Systems mit FOCUS erstellt. FOCUS ([Foc98]) ist eine Methodik zur formalen Spezifikation und schrittweisen Entwicklung verteilter, reaktiver Systeme, die eine Reihe von Beschreibungstechniken und Verfeinerungskonzepten zur Verfügung stellt.

Ziel der Fallstudie ist es, die in FOCUS vorhandenen Modellierungsmöglichkeiten vorzustellen und zu vermitteln, wie eine formale Spezifikation entwickelt werden kann. Anhand der Fallstudie werden wir außerdem untersuchen, inwieweit sich die vorhandenen

*Diese Arbeit entstand im Teilprojekt C2 (Softwaretechnik für Kommunikationssysteme) des Bayerischen Forschungsverbundes FORSOFT und wurde gefördert von der Bayerischen Forschungsstiftung.

Modellierungs- und Beschreibungstechniken von Focus für die Spezifikation der Dienststeuerung von Multimediadiensten eignen.

Wir werden für die Spezifikationen praxisorientierte, vorzugsweise graphische Beschreibungstechniken einsetzen, die aus industriellen Beschreibungstechniken hervorgegangen und in die Methodik Focus eingebunden worden sind. Dazu zählen zum Beispiel Systemstruktur- und Ereignisdiagramme ([HSS96]). Eine vollständige Spezifikation des Home Shopping Systems konnte in der knappen Zeit, die für die Fallstudie zur Verfügung stand, nicht durchgeführt werden und war auch nicht Ziel dieser Arbeit.

Der vorliegende Bericht ist wie folgt gegliedert: In Abschnitt 2 stellen wir Auszüge aus der informellen Beschreibung des Home Shopping Systems vor und listen eine Reihe von Ergänzungen zu [KS98] auf, die im Laufe dieser Fallstudie gesammelt worden sind. In Abschnitt 3 geben wir eine kurze Einführung in die Entwicklungsmethodik Focus. Abschnitt 4 stellt unsere Vorgehensweise bei der formalen Modellierung des Home Shopping Systems vor. Die Struktur des Systems wird in Abschnitt 5 modelliert. Typische Systemabläufe werden in Abschnitt 6 vorgestellt. In Abschnitt 7 erfolgt die tabellarische Spezifikation einer Systemkomponente als Beispiel für eine Verhaltensspezifikation in Focus. Ein Ausblick auf die Modellierung von dynamischen und kontinuierlichen Verhaltensanteilen sowie die Einbindung pragmatischer Beschreibungstechniken in Focus ist in Abschnitt 8 enthalten. Abschnitt 9 faßt unsere Erfahrungen bei der Modellierung mit Focus zusammen.

2 Informelle Systembeschreibung

Die formale Modellierung des Home Shopping Systems basiert auf der informellen Beschreibung des Systems in [KS98], das die Funktionalität und die Hardwareumsetzung des Home Shopping Systems beschreibt. Es folgt eine kurze Charakterisierung des Systems aus [KS98]; dort findet sich auch eine ausführliche Beschreibung des Systems.

> *Ausgangspunkt für die Fallstudie ist das in [HKS97] beschriebene Home Shopping System. ... In der geplanten Realisierung dieses Systems kann ein Kunde zu Hause mittels WWW-Browser im virtuellen Katalog eines Kaufhauses blättern. Zu einigen der angebotenen Artikel können Videoclips aufgerufen werden, welche dann auf dem PC oder dem Fernseher des Kunden abgespielt werden. Hat der Kunde weitergehende Fragen oder möchte er eine Bestellung abgeben, so kann er mittels Videoverbindung einen Berater kontaktieren. Ein zentrales Element in diesem Szenario stellt hierbei die hohe Qualität der Videoübertragungen dar. Diese soll sich an der Qualität des herkömmlichen Fernsehens messen und übersteigt damit die derzeitige Leistungsfähigkeit des Internets.*

Während der Durchführung der Fallstudie hat sich gezeigt, daß [KS98] nur eine grobe Beschreibung des Systems gibt und wichtige Verhaltensaspekte des Systems nicht behandelt.

Wir haben deshalb die informelle Beschreibung um eine Reihe von Punkten ergänzt, so daß als Ergebnis unserer Fallstudie nun eine genauere Beschreibung des Systems vorliegt.

- Zwischen der IP-Adresse des Rechners, von dem aus sich der Kunde ins Internet begibt, sowie der Kundennummer des Dienstnutzers ist keine eindeutige Beziehung möglich, da der Dienstnutzer den Dienst von unterschiedlichen Rechnern aus beanspruchen kann und/oder unterschiedliche Dienstnutzer den Dienst mit derselben IP-Adresse aufrufen können, wenn sie einen Internet-Anschluß gemeinsam nutzen.
- Die Kommunikationsverbindung bei der Videokonferenz zwischen dem Berater und dem Kunden wird mittels Internettelephonie erfolgen.
- Bei der Anmeldung erhält der Kunde eine eindeutige Kundennummer und gibt die Identifikationsnummer seiner Set-Top-Box an.
- Die SmartCom, ein Server für die Dienststeuerung und das Dienstmanagement, kann auf möglicherweise schon bestehende Datenbestände zugreifen, wie z.B. Liste mit Kundennummer und zugehöriger Nummer der Set-Top-Box.
- Sobald sich der Kunde für die Übertragung eines Videofilms oder für eine Beratung entscheidet, werden die Formulardaten vom Kunden direkt an die SmartCom geleitet, die daraufhin eine Verbindung mit dem Kunden aufbaut.
- Der Bezug zwischen der SmartCom und dem WWW-Server ist zu präzisieren. So ist zum Beispiel unklar, ob die SmartCom selbst WWW-Seiten erzeugen und als Server dienen kann oder ob sie Daten für die Erstellung von WWW-Seiten an den WWW-Server sendet.
- Jede Set-Top-Box erhält eine Kanalnummer, die als eine Art „Schlüssel“ dient, um aus dem Datenstrom, der für alle Set-Top-Boxen identisch ist, die für sie bestimmten Daten zu filtern. Der Multiplexer erhält von der Set-Top-Box die Schlüssel mitgeteilt und mischt die Datenströme, die er vom Videoserver und den Beratern erhält, derart, daß die einzelnen Set-Top-Boxen anhand ihres Schlüssels die für sie bestimmten Daten aus dem Eingabestrom filtern können. Bei der Verwendung der Schlüssel handelt es sich nicht um Codierungs- bzw. Verschlüsselungsverfahren, sondern um ein Verfahren, das die Videofilme eindeutig den Set-Top-Boxen zuordnet.

Die Ergänzung von [KS98] um die eben genannten Punkte bildet ein wichtiges Ergebnis unserer Fallstudie. Es wurde deutlich, daß die Durchführung einer formalen Spezifikation zu einer sehr genauen Betrachtung der Problemstellung und der gegebenen informellen Beschreibung zwingt. Dadurch wird eine Konkretisierung der Aufgabenstellung erreicht. Voraussetzung dafür ist allerdings ein reger Informationsaustausch zwischen den Auftraggebern und den Erstellern der Spezifikation.

3 FOCUS

FOCUS ([Foc98]) ist eine Methodik zur Entwicklung verteilter, reaktiver Systeme. Die grundlegende Vorstellung besteht darin, Systeme als Netzwerke von Komponenten zu modellieren, die durch asynchronen Nachrichtenaustausch über gerichtete Kanäle kommunizieren.

FOCUS ist wie folgt charakterisiert:

- FOCUS verfügt über eine präzise, formale Basis auf mathematisch-logischen Konzepten (denotationelle Semantik), die auf Strömen und stromverarbeitenden Funktionen basiert [Kah74, Bro86].
- FOCUS bietet für die Spezifikation von verteilten, reaktiven Systemen eine Reihe von Beschreibungstechniken an, mit denen sich verschiedene Systemsichten spezifizieren lassen. Die Beschreibungstechniken sind sowohl graphisch als auch textuell ausgerichtet; die Semantik der Beschreibungstechniken ist mittels der formalen Basis von FOCUS präzise festgelegt. Die Beschreibungstechniken ermöglichen die Spezifikation eines Systems ohne expliziten Zugriff auf die mathematisch-logischen Konzepte von FOCUS.
- FOCUS ermöglicht eine vollständige Systementwicklung, ausgehend von einer abstrakten Anforderungsspezifikation über die Designphase bis hin zur Implementierungsphase. FOCUS verfügt dazu über einen Verfeinerungsbegriff mit verschiedenen Ausprägungen. Damit wird eine schrittweise Entwicklung von Systemspezifikationen methodisch unterstützt.
  - Verhaltensverfeinerung: Das Verhalten von Komponenten, die unterspezifiziert sind (d.h. ihr Verhalten ist nicht vollständig, z.B. nur für bestimmte Eingabenachrichten, festgelegt), wird konkretisiert.
  - Strukturelle Verfeinerung: Der strukturelle Aufbau des verteilten Systems wird präzisiert, indem die Komponenten zu Netzwerken von Komponenten verfeinert werden.
  - Interaktionsverfeinerung: Hierbei wird die Schnittstelle einer Komponente verfeinert, indem die Anzahl oder auch die Typen der Kanäle verändert werden.
- Basierend auf der formalen Basis stehen eine Reihe von Verifikationskonzepten zur Verfügung; damit können Eigenschaften einer Systemspezifikation überprüft werden.
- Durch das Case-Tool AUTOFOCUS ist eine prototypische Werkzeugunterstützung für die Systemspezifikation mit praxisorientierten, vorzugsweise graphischen Beschreibungsmitteln in FOCUS gegeben.

Für die Spezifikation der verschiedenen Sichten auf ein System stehen jeweils unterschiedliche Beschreibungstechniken zur Verfügung:

- Der Aufbau des Systems aus seinen Komponenten und seinen Kommunikationsverbindungen wird graphisch durch Systemstrukturdiagramme skizziert und auf textueller Ebene durch Netzwerkbeschreibungen spezifiziert.
- Das Verhalten des Gesamtsystems und seiner Komponenten kann graphisch durch Zustandsübergangsdiagramme sowie textuell durch funktionale Beschreibungen oder tabellarische Spezifikationen beschrieben werden.
- Für die Beschreibung von Beispielabläufen eines Systems bzw. einer Komponente können Erweiterte Ereignisdiagramme (EETs) verwendet werden.

Eine Einführung in diese Beschreibungstechniken geben beispielsweise [HSS96] und [SS95]. In den nachfolgenden Abschnitten werden wir die konkrete Anwendung einiger dieser Beschreibungstechniken anhand der Fallstudie vorstellen. Ein Überblick über die Methodik Focus und einige ihrer Anwendungsbereiche ist in [BHS98] zu finden.

4 Vorgehen

Bei der Modellierung des Home Shopping Systems mit Focus werden wir uns an folgender Vorgehensweise orientieren:

- Zunächst werden wir eine Abgrenzung zwischen dem zu modellierenden System und seiner Systemumgebung vornehmen. Dabei ist zu beachten, daß wir über das Verhalten der Systemumgebung keinerlei Festlegungen treffen werden. Das System selbst verhält sich gemäß der von uns zu entwickelnden Spezifikation. Für den Nachrichtenaustausch zwischen System und Systemumgebung sind Kanalverbindungen anzugeben. Für jeden Kanal ist der dazugehörige Nachrichtentyp zu definieren, um festzulegen, welche Nachrichten über den Kanal übertragen werden.
- Dann erfolgt der Aufbau des Systems aus seinen Komponenten und seinen internen Kommunikationsverbindungen. Dazu wird das System in Komponenten unterteilt. Die Komponenten, die miteinander durch Nachrichtenaustausch kommunizieren, werden durch Kanäle verbunden. Dabei ist wieder für jeden Kanal der dazugehörige Nachrichtentyp anzugeben.
- Es folgen eine Reihe von Verfeinerungsschritten, in denen die Struktur des Systems detaillierter beschrieben wird, bis die endgültige Strukturierung des Systems erreicht ist. Die vorhandenen Komponenten der untersten Hierarchieebene sollen nicht weiter in Komponenten zerlegt werden; wir bezeichnen sie als Basiskomponenten.
- Bereits nach der Abgrenzung des Systems zu seiner Umgebung oder aber nach der Strukturierung des Systems können Erweiterte Ereignisdiagramme (EETs) erstellt werden. Diese beschreiben exemplarische Systemabläufe, indem sie den Nachrichtenaustausch zwischen dem System und seiner Umgebung bzw. zwischen den einzelnen

Systemkomponenten angeben.

- Abschließend erfolgt die Verhaltensbeschreibung der einzelnen Basiskomponenten. Es wird definiert, wie eine Komponente auf die Nachrichten reagiert, die sie über ihre Eingabekanäle empfängt. Die Komponente kann als Reaktion Nachrichten an andere Komponenten senden und ihren lokalen Datenzustand verändern. Zunächst beschreiben wir das Verhalten informell mit schematisiertem Text. Aus dieser textuellen Form wird die formale Spezifikation abgeleitet. Diese besteht aus einer Menge von Funktionsgleichungen, die wir in tabellarischer Form angeben.

5 Strukturbeschreibung

Um die Struktur des Systems konzeptionell festzulegen, muß eine genaue Vorstellung der Aufgaben vorliegen, die das System zu erfüllen hat. Darauf aufbauend wird entschieden, wie diese Aufgaben auf einzelne Komponenten verteilt werden.

5.1 Erste Strukturbeschreibung

Wir beginnen mit der Black-Box-Sicht des Systems, die in Abbildung 1 graphisch durch ein Systemstrukturdiagramm spezifiziert ist. Für das System *Home Shopping* wird die Kanalschnittstelle festgelegt, ohne daß dabei seine innere Struktur betrachtet wird. Über die Kanäle $in_1, \ldots, in_n$ und $out_1, \ldots, out_n$ tauschen das System und die n Kunden (Dienstnutzer des Systems) Informationen aus: die Kunden rufen Webseiten ab. Über die Kanäle $vd_1, \ldots, vd_n$ sendet das System für die Übertragung von Videofilmen oder Videokonferenzen Videoströme an die Kunden. Mit der Definition der Schnittstelle ist eine Trennung zwischen dem System und seiner Umgebung vorgenommen – dies ist ein erster Modellierungsschritt. Die hier gewählte Trennung gibt die Sicht wieder, die ein Dienstnutzer auf das System hat.

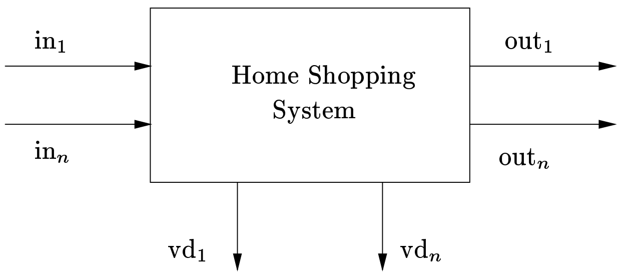

Abbildung 1: System aus Black-Box-Sicht

In einem nächsten Schritt wird die Struktur des Home Shopping Systems graphisch skizziert (siehe Abbildung 2). Dabei werden zwecks Übersichtlichkeit Komponenten, die mehrfach auftreten, nur einmal mit den dazugehörigen Kanalverbindungen erfaßt; die weiteren Komponenten werden durch einen Index, die Anzahl der Kanalverbindungen durch ein Intervall angedeutet. Dies ist eine Erweiterung der Syntax von Systemstrukturdiagrammen, wie sie bisher in FOCUS eingesetzt werden.

In Tabelle 1 sind für einige Kanäle die dazugehörigen Nachrichtenmengen aufgelistet. Die Nachrichten sind zum Teil mit Parametern versehen, die weitere Informationen einer Nachricht enthalten, wie zum Beispiel Information über die Anforderung des Kunden. Sämtliche Parameter sind vom Datentypen *Naturals* der natürlichen Zahlen. Datentypen werden in der algebraischen Spezifikationssprache SPECTRUM ([BFG$^+$93]) definiert. In [BFG$^+$93] ist die Definition von *Naturals* als abstrakter Datentyp enthalten, so daß wir hier auf die Angabe der Definition verzichten.

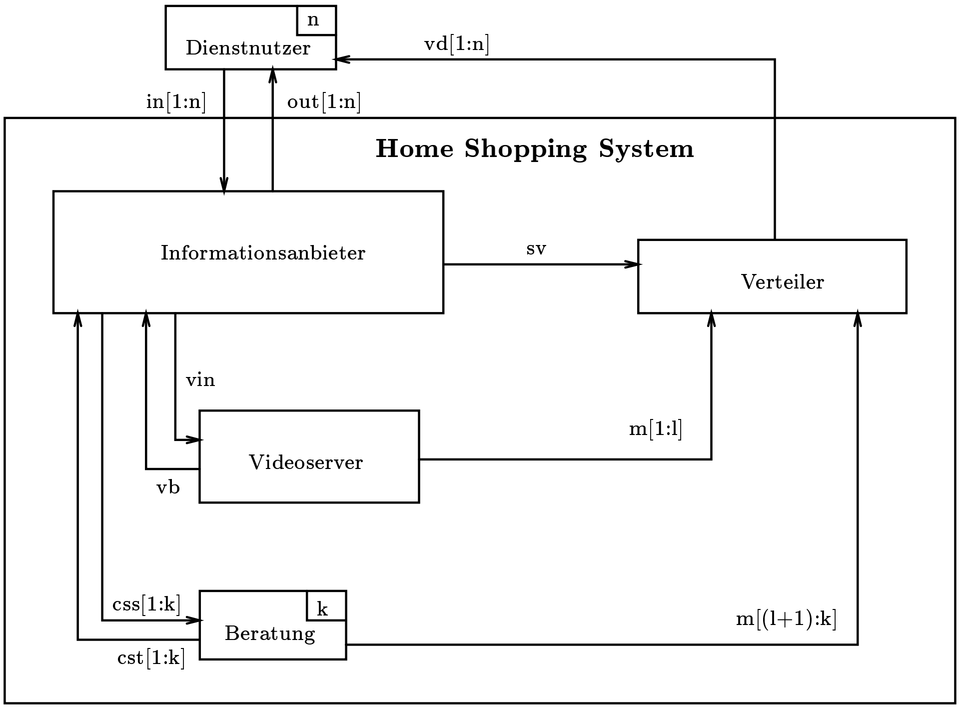

Abbildung 2: Systemstruktur von *Home Shopping*

Die graphische Darstellung dient zur Veranschaulichung der Systemstruktur, die in Abbildung 3 textuell mit ANDL [SS95] beschrieben wird. Mit der ANDL-Spezifikation werden der Aufbau des Systems aus seinen Komponenten und seinen Kommunikationsverbindungen auf textueller Ebene mit programmiersprachenähnlichen Konstrukten definiert. Ab einer gewissen Systemgröße sind graphische Darstellungen nicht mehr übersichtlich; die Information kann dann in einer textuellen Darstellung übersichtlicher und vollständig dargestellt werden. Das in der Graphik angedeutete mehrfache Auftreten von Komponenten wird in der ANDL-Spezifikation explizit spezifiziert.

Kanal	**Typ**	**Nachrichtenmenge**
in, c1	IN	request_info (ip, user_id, prod_id), request_video (ip, user_id, prod_id), request_conference (ip, user_id, prod_id)
out, c2	OUT	html_data (ip, user_id, prod_id)
vd, m	VD	video_begin, video_end, video_data
vin	VIN	start_video (session_nr, video_nr, channel), get_channel (session_nr)
vb	VST	free (session_nr, video_channel), trans_end (session_nr)
sv	SV	channels (channel, key), get_video (key)
smux	SV1	channel (channel, key)
sstb	SV2	get_video (key)
css	CSS	conference_session (ip, user_id, conf_info)
cst	CST	consultance_info (consultance_id, data)

Tabelle 1: Definition der Kanäle mit den dazugehörigen Nachrichtentypen

agent Home Shopping System
input channel $\mathrm{in}_1 : \mathrm{IN}, \ldots, \mathrm{in}_n : \mathrm{IN}$
output channel $\mathrm{out}_1 : \mathrm{OUT}, \ldots, \mathrm{out}_n : \mathrm{OUT},\ \mathrm{vd}_1 : \mathrm{VD}, \ldots, \mathrm{vd}_n : \mathrm{VD}$
is network

$$
\begin{array}{rcl}
\langle \mathrm{out}_1, \ldots, \mathrm{out}_n,\ \mathrm{vin}, & = & \mathrm{Informationsanbieter} \\
\mathrm{css}_1, \ldots, \mathrm{css}_k, \mathrm{sv}\rangle & & \qquad \langle \mathrm{in}_1, \ldots, \mathrm{in}_n,\ \mathrm{vb},\ \mathrm{cst}_1, \ldots, \mathrm{cst}_k\rangle; \\
\langle \mathrm{m}_1, \ldots, \mathrm{m}_l, \mathrm{vb}\rangle & = & \mathrm{Videoserver}\ \langle \mathrm{vin}\rangle; \\
\langle \mathrm{m}_{l+1},\ \mathrm{cst}_1\rangle & = & \mathrm{Beratung}_1\ \langle \mathrm{css}_1\rangle; \\
\vdots & = & \vdots \\
\langle \mathrm{m}_{l+k},\ \mathrm{cst}_k\rangle & = & \mathrm{Beratung}_k\ \langle \mathrm{css}_k\rangle; \\
\langle \mathrm{vd}_1, \ldots, \mathrm{vd}_n\rangle & = & \mathrm{Verteiler}\ \langle \mathrm{m}_1, \ldots, \mathrm{m}_l,\ \mathrm{m}_{l+1}, \ldots, \mathrm{m}_{l+k}, \mathrm{sv}\rangle
\end{array}
$$

end Home Shopping System

Abbildung 3: Spezifikation des Systems *Home Shopping*

5.2 Strukturelle Verfeinerungsschritte

Ausgehend von der ersten strukturellen Spezifikation werden wir im folgenden einige Verfeinerungsschritte durchführen und somit mehr Information über den Aufbau des Systems in die Spezifikation aufnehmen.

In einem ersten Schritt werden Informationen über das bisher nicht erfaßte Kommunikationsnetz in die Strukturbeschreibung aufgenommen. Damit wird detaillierter spezifiziert, welche Komponenten für den Nachrichtenaustausch zwischen dem System und der Systemumgebung erforderlich sind. Es wird dabei nur das abstrakte Verhalten der Übertragungsmedien erfaßt, nicht ihre technische Realisierung. Abbildung 4 zeigt die Struktur des Home Shopping Systems mit Komponenten, die die Übertragungsmedien darstellen. Wir bezeichnen dieses System mit *Home Shopping 2*.

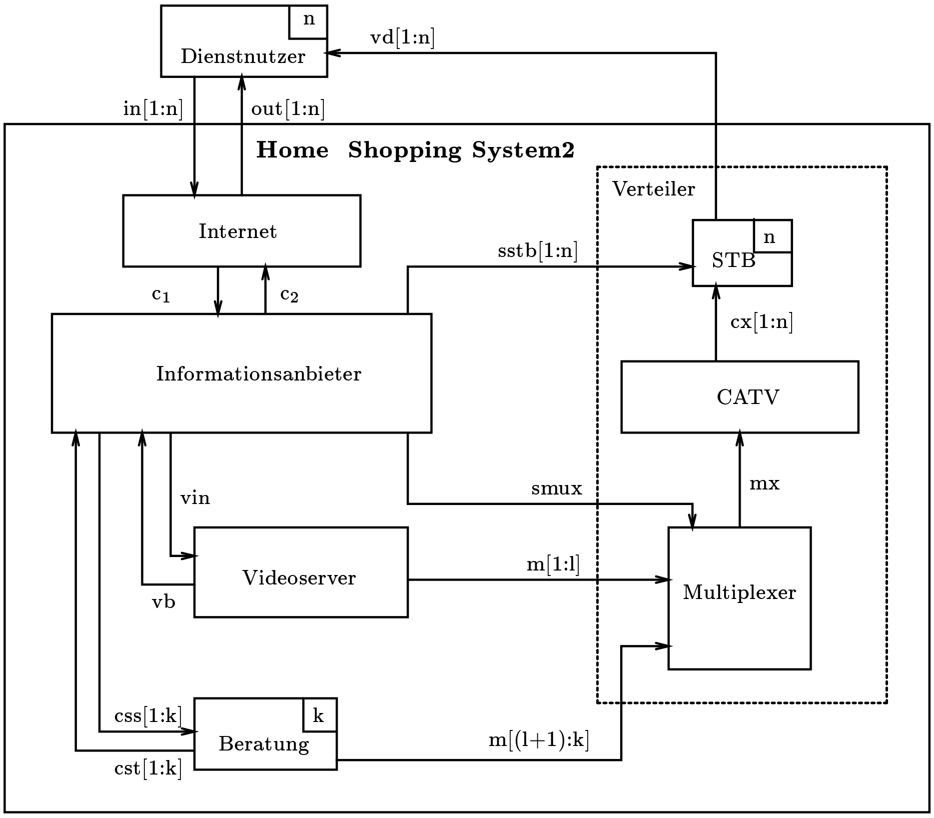

Abbildung 4: Die Komponente *Home Shopping 2*

Die abstrakte Verbindung zwischen dem Informationsanbieter und der Systemumgebung wird durch eine Verbindung ersetzt, die über das Internet erfolgt. Dazu wird eine Kanalverfeinerung durchgeführt; die Komponente *Internet* wird hinzugefügt.

Für den Nachrichtenfluß vom Verteiler zur Umgebung ist als Transportmedium das Kabelnetz vorgesehen; jeder Dienstnutzer verfügt über eine Set-Top-Box (STB), die die Videodaten, die an den Dienstnutzer gerichtet sind, aus dem Datenstrom filtert. Wir verfeinern die Komponente *Verteiler* strukturell in die Komponenten *Multiplexer*, *CATV* und *STB*.

Zudem ist eine Interaktionsverfeinerung erforderlich: Der Kanal sv wird in die Kanäle $smux$ und $sstb_1$ bis $sstb_n$ verfeinert. Über $smux$ teilt der Informationsanbieter dem Multiplexer mit, mit welchem Schlüssel welche Eingabeströme zu verschlüsseln sind. Über $sstb_i$ wird die Set-Top-Box informiert, mit welchem Schlüssel sie die für sie bestimmten Daten aus ihrem Eingabestrom filtern kann. Wie diese Information bei einer späteren Realisierung die Set-Top-Box erreicht, ist eine Implementierungsentscheidung und bleibt auf dieser Spezifikationsebene offen; wir erfassen die logische, nicht die physikalische Verbindung zwischen dem Informationsanbieter und der Set-Top-Box.

In der ANDL-Spezifikation von *Home Shopping 2* (Abbildung 5) sind die Veränderungen gegenüber der ersten Spezifikation *Home Shopping* hervorgehoben. In Abbildung 6 ist die Komponente *Verteiler* als ANDL-Netzwerk spezifiziert.

agent Home Shopping System 2

input channel $\mathrm{in}_1 : \mathrm{IN}, \ldots, \mathrm{in}_n : \mathrm{IN}$

output channel $\mathrm{out}_1 : \mathrm{OUT}, \ldots, \mathrm{out}_n : \mathrm{OUT}, \mathrm{vd}_1 : \mathrm{VD}, \ldots, \mathrm{vd}_n : \mathrm{VD}$

is network

$$
\begin{array}{rcl}
<\mathbf{out_1, \ldots, out_n, c_1}> & = & \mathbf{Internet} <\mathbf{in_1, \ldots, in_n, c_2}> \\
<\mathbf{c}_2,\ \mathrm{vin},\ \mathrm{css}_1, \ldots, \mathrm{css}_k, \mathbf{smux}, & & \\
\mathbf{sstb_1, \ldots, sstb_n}> & = & \mathrm{Informationsanbieter} \\
 & & <\mathbf{c_1},\ \mathrm{vb},\ \mathrm{cst}_1, \ldots, \mathrm{cst}_k>; \\
<\mathrm{m}_1, \ldots, \mathrm{m}_l, \mathrm{vb}> & = & \mathrm{Videoserver} <\mathrm{vin}>; \\
<\mathrm{m}_{l+1},\ \mathrm{cst}_1> & = & \mathrm{Beratung}_1 <\mathrm{css}_1>; \\
\vdots & = & \vdots \\
<\mathrm{m}_{l+k},\ \mathrm{cst}_k> & = & \mathrm{Beratung}_k <\mathrm{css}_k>; \\
<\mathrm{vd}_1, \ldots, \mathrm{vd}_n> & = & \mathrm{Verteiler} <\mathrm{m}_1, \ldots, \mathrm{m}_l, \mathrm{m}_{l+1}, \ldots, \mathrm{m}_{l+k}, \\
 & & \quad \mathbf{smux, sstb_1, \ldots, sstb_n}>
\end{array}
$$

end Home Shopping System 2

Abbildung 5: Spezifikation des Systems *Home Shopping 2*

agent Verteiler

input channel $\mathrm{m}_1 : \mathrm{VD}, \ldots, \mathrm{m}_{l+k} : \mathrm{VD}, \mathrm{smux} : \mathrm{SV1}, \mathrm{sstb}_1 : \mathrm{SV2}, \ldots, \mathrm{sstb}_n : \mathrm{SV2}$

output channel $\mathrm{vd}_1 : \mathrm{VD}, \ldots, \mathrm{vd}_n : \mathrm{VD}$

is network

$$
\begin{array}{rcl}
<\mathrm{mx}> & = & \mathrm{Multiplexer} <\mathrm{m}_1, \ldots, \mathrm{m}_{l+k}, \mathrm{smux}> \\
<\mathrm{cx}_1, \ldots, \mathrm{cx}_n> & = & \mathrm{CATV} <\mathrm{mx}> \\
<\mathrm{vd}_1> & = & \mathrm{STB}_1 <\mathrm{cx}_1, \mathrm{sstb}_1> \\
\vdots & = & \vdots \\
<\mathrm{vd}_n> & = & \mathrm{STB}_n <\mathrm{cx}_n, \mathrm{sstb}_n>
\end{array}
$$

end Verteiler

Abbildung 6: Spezifikation der Komponente *Verteiler*

Nun erfolgt die Aufteilung der Komponenten *Informationsanbieter*, *Videoserver* und *Beratung* in mehrere Teilkomponenten (siehe Abbildung 7). Zusätzlich werden die Kanäle c_1 und c_2 zwischen den Komponenten $Internet$ und $Informationsanbieter$ in die Kanäle $wwwin$ und usc bzw. $wwwout$ und scu verfeinert. Die Komponenten *SmartCom* und *WWWServer* verfügen damit über getrennte Verbindungen zu *Internet*.

Die dazugehörigen Netzwerkspezifikationen in ANDL werden wie bei den vorherigen Strukturdiagrammen erstellt und hier nicht angegeben. Aus diesem Grund haben wir in den Strukturdiagrammen auf die durchgehende Angabe von Kanalbezeichnern verzichtet. Die strukturellen Verfeinerungsschritte werden solange fortgesetzt, bis jede Komponente eine sogenannte Basiskomponente darstellt. Für eine Basiskomponente gilt, daß sich das Verhalten der Komponente unmittelbar durch ihr Ein-/Ausgabeverhalten beschreiben läßt und eine weitere Zerlegung in Teilkomponenten nicht mehr sinnvoll ist.

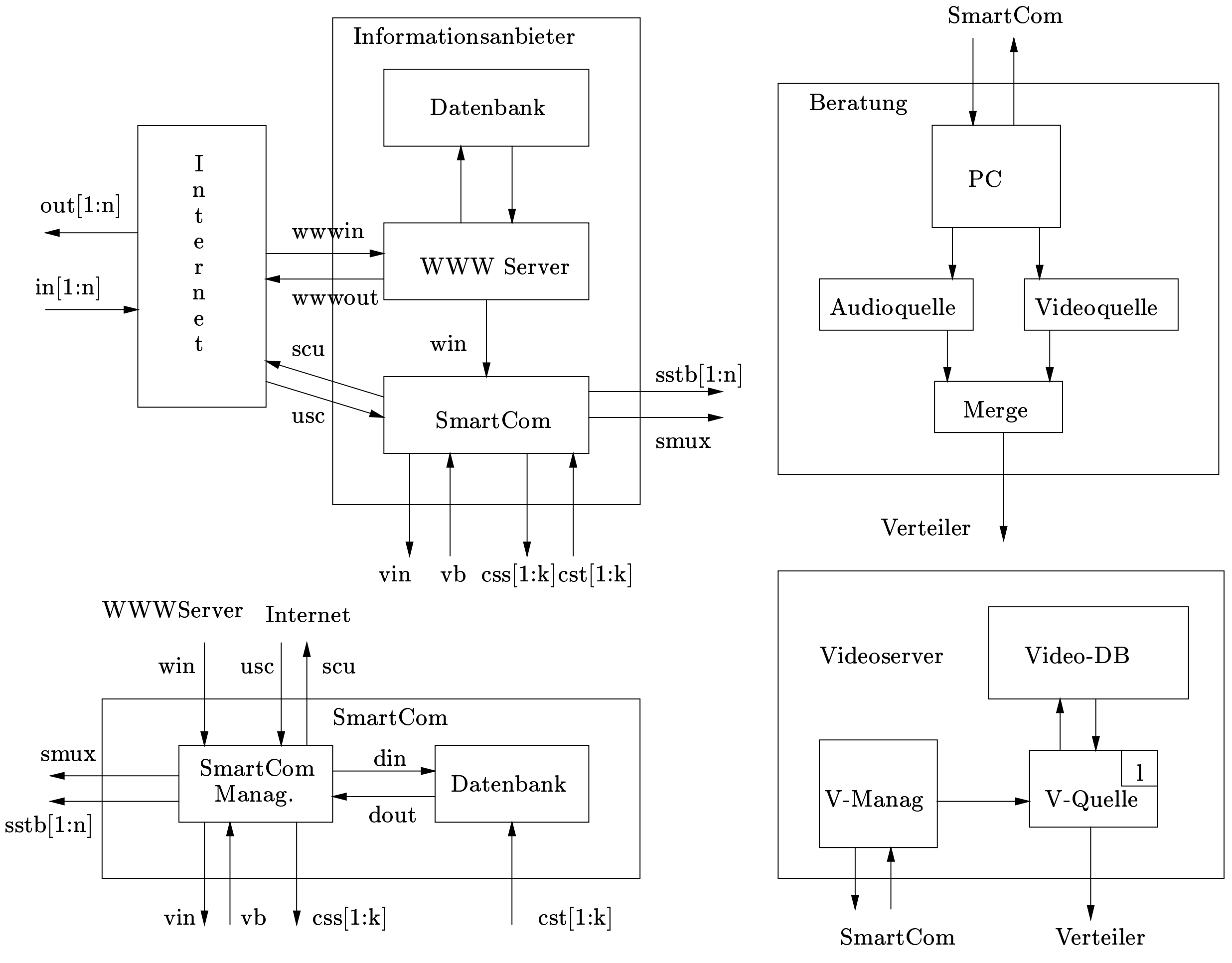

Abbildung 7: Komponenten aus *Home Shopping 2* nach struktureller Verfeinerung

6 Ablaufdiagramme

Ein wesentlicher Punkt bei der Erstellung der Spezifikation ist die Beschreibung der Interaktion zwischen dem System und seiner Umgebung sowie zwischen den einzelnen Systemkomponenten. Als Beschreibungstechnik für diesen Aspekt der Systembeschreibung liegen in FOCUS Erweiterte Ereignisdiagramme (EETs) in Anlehnung an Message Sequence Charts [IT96] vor.

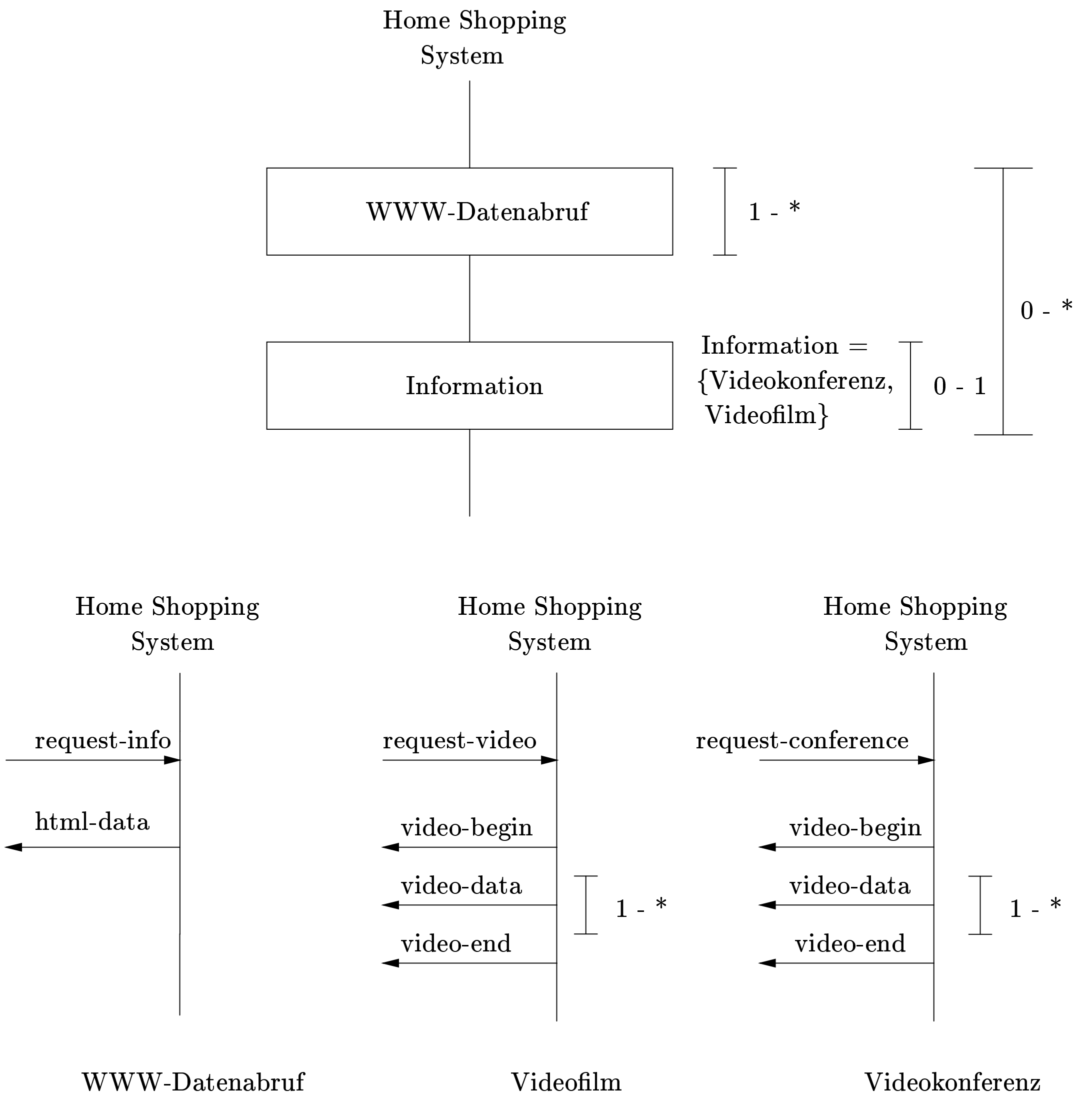

Abbildung 8: Nachrichtenaustausch zwischen System und Kunde

In Abbildung 8 ist ein EET dargestellt, das den Nachrichtenaustausch zwischen einem Kunden und dem System *Home Shopping* spezifiziert, wobei die Parameterwerte der Nachrichten nicht berücksichtigt sind. Wir betrachten das System dabei aus Black-Box-Sicht, d.h. ohne Berücksichtigung seiner internen Struktur.

Durch das obere EET ist sichergestellt, daß vor jedem Videofilm und jeder Videokonferenz ein Datenabruf über WWW erfolgt, in dem der Kunde die entsprechende Auswahl trifft. Das EET läßt aber auch einen Ablauf zu, bei dem der Kunde nur Daten über WWW abruft und keine anderen Informationsmöglichkeiten wahrnimmt.

7 Verhaltensspezifikation

Als Beispiel für eine Verhaltensbeschreibung einer Basiskomponente wählen wir die Komponente *SmartComManager*, im weiteren kurz SCM genannt. SCM weist ein komplexes Verhalten auf – sie ist die Dienststeuerung des Home Shopping Systems. Wir konzentrieren uns im wesentlichen auf die Verhaltensanteile, die die Übertragung von Videofilmen betreffen.

In Abschnitt 7.1 beschreiben wir das Verhalten von SCM zunächst informell und geben darauf aufbauend in Abschnitt 7.2 eine schematisierte textuelle Darstellung an. Anschließend leiten wir aus der schematischen Beschreibung das Ein/-Ausgabeverhalten ab und geben es tabellarisch an (siehe Abschnitt 7.3). Damit erhalten wir eine formale Spezifikation der Komponente. In Abschnitt 7.4 folgen einige Anmerkungen zu unserer Spezifikation.

7.1 Informelle Beschreibung von *SmartComManager*

Abbildung 9 zeigt die Kanalschnittstelle der Komponente SCM; Tabelle 2 enthält die Nachrichten, die über die Ein- und Ausgabekanäle von SCM gesendet werden, wobei nur die Nachrichten aufgeführt sind, die im Zusammenhang mit einer Videoübertragung stehen. SCM bildet die Dienststeuerung des Home Shopping Systems.

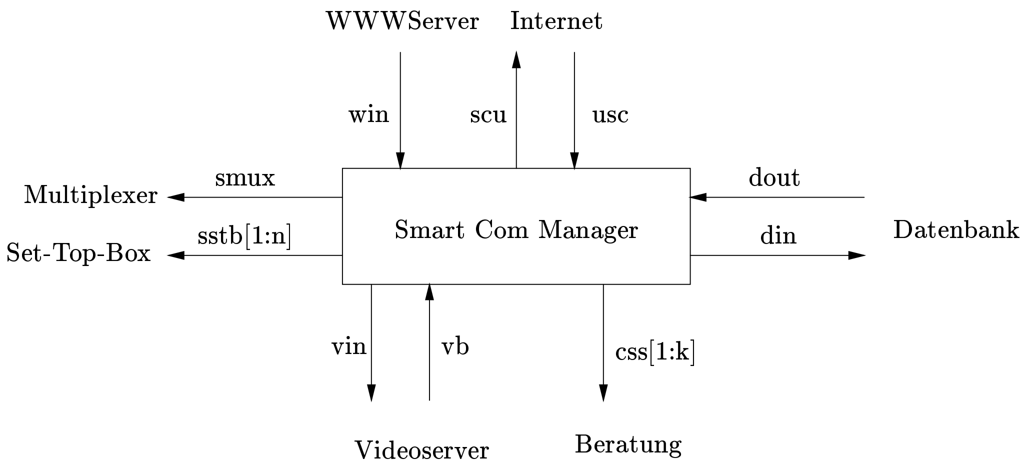

Abbildung 9: Schnittstelle von $SmartComManager$

Im folgenden stellen wir vor, wie die Anforderung eines Kunden für einen Videofilm durch SCM bearbeitet wird. Diese Beschreibung stellt eine Ergänzung von [KS98] dar, in der keine Beschreibung des Verhaltens der SmartCom enthalten ist. Die folgende informelle Beschreibung resultiert aus Gesprächen zwischen den Auftraggebern und den Erstellern der Spezifikation.

Die Komponente SCM ist über das Internet mit den Kunden des Home Shopping Systems verbunden und setzt die Anforderungen ($request$), die sie von den Kunden erhält, in Nachrichten um, die den gewünschten Informationsdienst anstoßen. Für die Videoübertragung fordert SCM mit der Nachricht *get_data* Information aus der Datenbank an und erhält als Antwort die Nachricht *snd_data* mit folgenden Parametern: der Nummer des Videofilms *video_nr*, der zu dem Produktbereich *prod_id* gehört, den der Kunde gewählt hat; den Schlüssel *key* für die Set-Top-Box des Kunden und den Namen des Kanals *stb_channel*, über den diese Set-Top-Box erreichbar ist. Vom Videoserver fordert SCM mit der Nachricht *get_channel* die Nummer eines freien Kanals *channel_nr* an, über den der Videofilm abgespielt werden wird. Erst wenn all diese Informationen mit dem Erhalt der Nachrichten $free$ und *snd_data* für einen Auftrag vorliegen, kann SCM die Übertragung des Videofilms durch den Videoserver veranlassen (*start_video*). Der Multiplexer wird informiert, mit welchem Schlüssel *key* die Videodaten, die er vom Videoserver auf Kanal *channel* erhält, in den Ausgabestrom zu mischen sind. An die Set-Top-Box des Kunden sendet SCM die Nachricht *get_video* mit dem Schlüssel *key*, damit diese die für sie bestimmten Videodaten aus ihrem Eingabestrom filtern kann.

Alle Nachrichten, die im Zusammenhang mit dem Auftrag eines Kunden für eine Videoübertragung stehen, sind mit einer Auftragsnummer versehen, die SCM bei Erhalt eines Kundenauftrags vergibt. Anhand dieser Nummer kann SCM eintreffende Informationen von Videoserver und Datenbank den verschiedenen Aufträgen zuordnen und entscheiden, ob für einen Auftrag alle notwendigen Informationen vorliegen. Ist dies nicht der Fall, so werden die Informationen solange zwischengespeichert, bis die restlichen Informationen eingetroffen sind.

Kanal	**Typ**	**Nachrichtenmenge** (alle Parameter vom Typ $Naturals$)
usc	IN	request_video (user_id, prod_id) (*kurz: request(user_id, prod_id)*)
din	DIN	get_data (session_nr, user_id, prod_id), put_data (session_nr, user_id, prod_id, channel, stb_chan, key)
dout	DOUT	snd_data (session_nr, video_nr, stb_channel, key)
vb	VST	free (session_nr, channel_nr), end_trans (session_nr)
vin	VIN	start_video (session_nr, video_nr, channel), get_channel (session_nr)
smux	SV1	channels (channel, key)
sstb	SV2	get_video (key)

Tabelle 2: Nachrichtenmengen für die Ein- und Ausgabekanäle von SCM

Der **Datenzustand** von SCM setzt sich aus zwei Variablen zusammen: Um die Aufträge der Kunden eindeutig identifizieren zu können, verfügt SCM über einen Zähler. Darüber hinaus ist eine Variable erforderlich, die einen lokalen Datenspeicher von SCM repräsentiert. In dieser Variablen speichert SCM für jede Auftragsnummer die dazugehörigen Informationen. Nach erfolgreicher Beendigung einer Videoübertragung (*end_trans*) werden diese Informationen für den Zweck der Vergebührung oder Statistik mit der Nachricht *put_data* an die Datenbank gesendet und aus dem Speicher gelöscht.

7.2 Schematisierte Beschreibung von *SmartComManager*

Um das Verhalten von SCM formal zu spezifizieren, ist die Beziehung zwischen den Ein- und den Ausgabeströmen von SCM festzulegen. Dies erfolgt bei unserer Vorgehensweise schrittweise operationell, indem die Reaktion der Komponente auf einzelne Nachrichten beschrieben wird; die Eingabeströme werden nachrichtenweise abgearbeitet. Als Reaktion auf eine Eingabenachricht kann die Komponente ihren internen Datenzustand ändern und Nachrichten auf ihre Ausgabekanäle senden.

Auf der funktionalen Ebene besteht in FOCUS die Möglichkeit, das Zeitverhalten einer Komponente explizit zu modellieren. Dazu wird eine globale, diskrete Systemzeit vorausgesetzt. Wir wählen für unsere Spezifikation eine synchrone Zeitmodellierung: in jedem Zeitintervall kann höchstens eine Nachricht gesendet oder empfangen werden. Die Nachrichten in den Strömen sind eindeutig Zeitintervallen zugeordnet. Wird in einem Zeitintervall keine Nachricht versendet, so wird dies durch das spezielle Symbol $\surd$ (Zeittick) beschrieben. Mit diesem einfachen Zeitmodell können wir zum Beispiel die Situation spezifizieren, daß an einem Eingabekanal einer Komponente zum aktuellen Zeitpunkt keine Nachricht vorliegt.

Wir beschreiben das Ein-/Ausgabeverhalten nun informell, aber schematisch, indem wir für die verschiedenen Kombinationen von Eingabenachrichten die Reaktion von SCM beschreiben.

Seien z der Wert des Zählers und σ die aktuelle Belegung des lokalen Speichers bestehend aus den Aufträgen mit den dazugehörigen Informationen.

Erhält die Komponente SCM auf Kanal *usc* die Nachricht $request(user, product)$, so

- erzeugt sie einen neuen Eintrag in σ, gekennzeichnet mit z, in dem die Werte *user* und *product* abgespeichert werden,
- erhöht den internen Zähler z

und sendet

- auf Kanal *din* die Nachricht $get\_data(z, user, product)$,
- auf Kanal *vin* die Nachricht $get\_channel(z)$ an den Videoserver.

Erhält SCM

- auf Kanal $dout$ die Nachricht $snd\_data(n, vnr, stb\_chan, key)$,
- auf Kanal vb die Nachricht $free(n, c)$ vom Videoserver,

so sendet sie

- auf Kanal vin die Nachricht $start\_video(n, vnr, c)$ an den Videoserver,
- auf Kanal stb_chan die Nachricht $get\_video(key)$ an die Set-Top-Box des Kunden,
- auf Kanal $smux$ die Nachricht $channel(c, key)$ an den Multiplexer

und

- speichert die Informationen aus den Nachrichten ab.

Erhält SCM

- auf Kanal $dout$ die Nachricht $snd\_data(m, vnr, stb\_chan, key)$
- auf Kanal vb die Nachricht $free(n, c)$ vom Videoserver,

mit $m \neq n$, so werden die Informationen aus den Nachrichten in den Speicher eingetragen und folgende Fallunterscheidung durchgeführt:

1. Weder für Auftrag n noch für Auftrag m liegen alle Informationen vor, so daß für keinen der Aufträge eine Videoübertragung gestartet werden kann.
2. Für einen der beiden Aufträge liegen bereits Informationen vor, so daß die Videoübertragung für diesen Auftrag gestartet werden kann.
3. Für beide Aufträge liegen bereits Informationen vor, so daß für beide Aufträge die Videoübertragungen gestartet werden können.

Das Starten der Videoübertragung erfolgt wie im vorherigen Fall.

Erhält SCM

- auf Kanal $dout$ den Zeittick $\surd$,
- auf Kanal vb die Nachricht $free(n, c)$ vom Videoserver,

so werden die Informationen aus der Nachricht $free$ gespeichert und folgende Fallunterscheidung durchgeführt:

- Für den Auftrag n liegt bereits die Information aus der Datenbank vor, so daß die Videoübertragung für diesen Auftrag gestartet werden kann.
- Für den Auftrag n liegt noch keine Information aus der Datenbank vor, so daß noch keine Videoübertragung gestartet werden kann.

Erhält SCM

- auf Kanal *dout* die Nachricht $snd\_data(m, vnr, stb\_chan, key)$ von der Datenbank,
- auf Kanal *vb* den Zeittick $\surd$,

so werden die Informationen aus der Nachricht *snd_data* gespeichert und folgende Fallunterscheidung durchgeführt:

- Für den Auftrag m liegt bereits die Information des Videoservers vor, so daß die Videoübertragung für diesen Auftrag gestartet werden kann.
- Für den Auftrag m liegt noch keine Information des Videoservers vor, so daß noch keine Videoübertragung gestartet werden kann.

Erhält SCM auf Kanal *vb* die Nachricht $trans\_end(n)$ vom Videoserver, so bedeutet dies, daß für den Auftrag mit Nummer n die Übertragung des Videos beendet ist. SCM sendet auf Kanal *din* die Nachricht *put* mit den zum Auftrag n gehörigen Informationen an die Datenbank, wobei diese Informationen aus der Speichervariablen σ für n entnommen werden. Anschließend werden die Einträge für Auftrag n aus σ gelöscht.

7.3 Formale Spezifikation von *SmartComManager*

Das Verhalten einer Komponente wird, wie in Abschnitt 7.2 beschrieben, durch die Beziehung zwischen ihren Ein- und Ausgabeströmen charakterisiert. Diese Beziehung wird durch die Angabe einer Funktion definiert, die für jeden Eingabekanal einen Strom von Eingabenachrichten erhält und diese auf Ströme von Ausgabenachrichten abbildet. Jeder Komponente werden auf diese Weise eine oder mehrere Funktionen zugeordnet. Letzteres bedeutet, daß das Verhalten der Komponente nicht eindeutig festgelegt ist und es deshalb mehrere mögliche Beziehungen zwischen den Ein- und Ausgabeströmen und damit mehrere Funktionen gibt. Durch Verhaltensverfeinerung kann das Verhalten der Komponente in weiteren Schritten konkretisiert werden, so daß die Zahl der Funktionen, die ein mögliches Verhalten festlegen, abnimmt.

Um das Verhalten der Funktionen zu definieren, verwenden wir einen konstruktiven Spezifikationsstil auf der Basis von rekursiven Funktionsgleichungen. Eine Funktionsgleichung genügt im wesentlichen folgendem Schema:

$$\begin{aligned} &f\,[d_1, \ldots, d_i]\,(\{in_1 \to m_1,\, in_2 \to m_2,\, \ldots,\, in_l \to m_l\} \,\&\, s) \;= \\ &\qquad\qquad \{out_1 \to n_1,\, out_2 \to n_2,\, \ldots,\, out_k \to n_k\} \,\&\, f\,[d'_1, \ldots, d'_i]\,(s) \end{aligned}$$

Diese Funktionsgleichung steht für folgende textuelle Beschreibung:

> Die Funktion f erhält auf ihren Eingabekanälen $in_1, \ldots, in_l$ die Nachrichten $m_1, \ldots, m_l$, reagiert darauf mit der Ausgabe der Nachrichten $n_1, \ldots, n_k$ auf ihre Ausgabekanäle $out_1, \ldots, out_k$ und arbeitet mit dem Reststrom s weiter.

Zusätzlich verfügt die Funktion über lokale Variablen, deren aktuelle Belegung $d_1, \ldots, d_i$ sich durch die Verarbeitung der Eingabenachrichten in $d'_1, \ldots, d'_i$ verändert.

Für das Verhalten von SCM ergeben sich komplexe Funktionsgleichungen, die nur schwer lesbar und erfaßbar sind. Deshalb geben wir die Gleichungen nicht explizit an, sondern erfassen den Informationsgehalt der Gleichungen in tabellarischer Form.

spec $SCM : \left(IN^{\omega}_{\surd} \times DOUT^{\omega}_{\surd} \times VST^{\omega}_{\surd}\right) \rightarrow \left(DIN^{\omega}_{\surd} \times VIN^{\omega}_{\surd} \times SV1^{\omega}_{\surd} \times \left(SV2^{\omega}_{\surd}\right)_{(1:n)}\right)$
data $z : Naturals = 1$ $\sigma : Buffer = \sigma_{init}$
input channels : $usc,\ dout,\ vb$ **output channels** : $din,\ vin,\ smux,\ sstb_1, \ldots, sstb_n$
behaviour: Ein-/Ausgabetabellen, siehe Tabellen 4, 5, 6

Tabelle 3: Tabellenspezifikation

Die Tabellenspezifikation in Tabelle 3 setzt sich aus vier Teilen zusammen:

spec gibt den Namen und die Funktionalität der zu spezifizierenden Komponente an. Die Funktionalität setzt sich aus den Ein- und den Ausgabeströmen der Komponente zusammen. Dabei bezeichnet $N^{\omega}_{\surd}$ die Menge aller Ströme, bestehend aus Nachrichten des Typs N und dem Zeittick $\surd$. $N_{(1:n)}$ steht abkürzend für das n-fache Vorkommen des Stromtyps N.

data definiert den Datenzustand von SCM, der sich aus der Variablen z für den Zähler und der Variablen σ für den lokalen Speicher zusammensetzt. Initialisiert werden diese Variablen mit 1 bzw. mit dem Wert σ_{init} für den leeren Speicher (siehe auch nachfolgende Erläuterungen zum Datenzustand auf Seite 19).

input/output channels gibt die Schnittstelle der Komponente an, bestehend aus ihren Ein- und Ausgabekanälen. (Wir haben nur diejenigen Kanäle in die Schnittstellenbeschreibung mit aufgenommen, die für die Erbringung des Videodienstes erforderlich sind.)

behaviour referenziert eine Ein-/Ausgabetabelle, in der die Beziehung zwischen den Ein- und Ausgabenachrichten in Abhängigkeit des Datenzustands von SCM beschrieben wird.

Die drei folgenden Tabellen 4, 5 und 6 beschreiben das Ein-/Ausgabeverhalten von SCM. Im folgenden geben wir einige Erläuterungen zum Aufbau der Tabellen und dem Datenzustand von SCM.

Eine **Ein-/Ausgabetabelle** setzt sich aus der Angabe der Eingabekanäle, Vorbedingungen **PRE**, den Ausgabekanälen und Nachbedingungen **POST** zusammen. Ist die Spalte **PRE** leer, so können die Eingabenachrichten bei beliebigem Datenzustand von SCM verarbeitet werden. Liegen Vorbedingungen vor, so können die Eingabenachrichten nur verarbeitet werden, wenn der aktuelle Datenzustand von SCM diese Bedingungen erfüllt. Nach der Verarbeitung der Eingabenachrichten erfüllt der Datenzustand von SCM die unter **POST** aufgeführten Nachbedingungen. z und σ bezeichnen den Datenzustand von SCM vor, z' und σ' den Datenzustand nach Verarbeitung der Eingabenachrichten.

Aufgrund des großen Umfangs der Tabelle wird die Tabelle in mehrere Teiltabellen zerlegt. So werden die Belegung der Eingabekanäle und die Vorbedingungen (siehe Tabelle 4) getrennt von der Belegung der Ausgabekanäle und den Nachbedingungen (siehe Tabelle 5, 6) aufgeführt. In der Spalte für den Ausgabekanal $sstb_i$ wird neben der Nachricht auch ein Index angegeben. Damit wird festgelegt, über welchen der n Kanäle $sstb_1, \ldots, sstb_n$ die Nachricht gesendet werden soll.

Erläuterungen zum Datenzustand:

Wir geben hier eine kurze Beschreibung des Datenzustands von SCM; die für den Zustand erforderlichen Datentypen lassen sich durch die in SPECTRUM ([BFG+93]) gegebenen abstrakten Datentypen definieren und werden hier nicht vorgestellt.

Die Variable z bezeichnet den aktuellen Wert des Zählers vom Typ $Naturals$.

Die Variable σ des Typs *Buffer* bildet den lokalen Speicher für die Auftragsnummern von SCM. Für jede Auftragsnummer von SCM sind in σ die dazugehörigen Informationen gespeichert.

σ_{init} ist der leere Speicher von SCM am Beginn des Systemablaufs, in dem noch keine Daten gespeichert sind.

σ_n liefert die Informationen für die Auftragsnummer n und besteht aus sechs Feldern mit Einträgen vom Typ $Naturals$:

1. Kundennummer u
2. Produktbezeichner p
3. Kanal für Übertragung des Videofilms c
4. Nummer des Videofilms v
5. Schlüssel für die Set-Top-Box k
6. Kanal s, auf dem die Set-Top-Box erreichbar ist

Belegung der Eingabekanäle mit Vorbedingung				
	usc	$dout$	vb	PRE
1	$request(us, pr)$	$\surd$	$\surd$	
2	$request(us, pr)$	$snd\_data(n, vn, stb, key)$	$free(m, c)$	$n = m$
3a	$request(us, pr)$	$snd\_data(n, vn, stb, key)$	$free(m, c)$	$n \neq m$ $db\_exists(\sigma_m), c\_exists(\sigma_n)$
3b				$n \neq m$ $\neg db\_exists(\sigma_m), c\_exists(\sigma_n)$
3c				$n \neq m$ $db\_exists(\sigma_m), \neg c\_exists(\sigma_n)$
3d				$n \neq m$ $\neg db\_exists(\sigma_m), \neg c\_exists(\sigma_n)$
4a	$request(us, pr)$	$snd\_data(n, vn, stb, key)$	$trans\_end(r)$	$n \neq r$ $c\_exists(\sigma_n)$
4b				$n \neq r$ $\neg c\_exists(\sigma_n)$
5a	$request(us, pr)$	$snd\_data(n, vn, stb, key)$	$\surd$	$c\_exists(\sigma_n)$
5b				$\neg c\_exists(\sigma_n)$
6a	$request(us, pr)$	$\surd$	$free(m, c)$	$c\_exists(\sigma_m)$
6b				$\neg c\_exists(\sigma_m)$
7	$request(us, pr)$	$\surd$	$trans\_end(r)$	
8	$\surd$	$\surd$	$\surd$	
9a	$\surd$	$snd\_data(n, vn, stb, key)$	$free(m, c)$	$n = m$
9b				$n \neq m$ $db\_exists(\sigma_m), c\_exists(\sigma_n)$
9c				$n \neq m$ $\neg db\_exists(\sigma_m), c\_exists(\sigma_n)$
9d				$n \neq m$ $db\_exists(\sigma_m), \neg c\_exists(\sigma_n)$
9e				$n \neq m$ $\neg db\_exists(\sigma_m), \neg c\_exists(\sigma_n)$
10a	$\surd$	$snd\_data(n, vn, stb, key)$	$trans\_end(r)$	$n \neq r$ $c\_exists(\sigma_n)$
10b				$n \neq r$ $\neg c\_exists(\sigma_n)$
11a	$\surd$	$snd\_data(n, vn, stb, key)$	$\surd$	$c\_exists(\sigma_n)$
11b				$\neg c\_exists(\sigma_n)$
12a	$\surd$	$\surd$	$free(m, c)$	$c\_exists(\sigma_m)$
12b				$\neg c\_exists(\sigma_m)$
13	$\surd$	$\surd$	$trans\_end(r)$	

Tabelle 4: Belegung der Eingabekanäle und Angabe der Vorbedingungen

Belegung der Ausgabekanäle mit Nachbedingung					
	din	*vin*	*smux*	$stb_i,\ i$	POST
1	$get\_data(z, us, pr)$	$get\_channel(z)$	$\surd$	$\surd$	$z' = z + 1, \sigma'_z = [us, pr, 0, 0, 0, 0]$
2	$get\_data(z, us, pr)$	$get\_channel(z)$ $start\_video(n, vn, c)$	$channel(c, key)$	$get\_video(key), i = stb$	$z' = z + 1, \sigma'_z = [us, pr, 0, 0, 0, 0]$ $\sigma'_n = [*, *, c, vn, stb, key]$
3a	$get\_data(z, us, pr)$	$get\_channel(z)$ $start\_video(n, vn, c(\sigma_n))$ $start\_video(m, v(\sigma_m), c)$	$channel(c(\sigma_n), key)$ $channel(c, k(\sigma_m))$	$get\_video(key), i = stb$ $get\_video(k(\sigma_m)), i = s(\sigma_m)$	$z' = z + 1, \sigma'_z = [us, pr, 0, 0, 0, 0]$ $\sigma'_n = [*, *, *, vn, stb, key]$ $\sigma'_m = [*, *, c, *, *, *]$
3b	$get\_data(z, us, pr)$	$get\_channel(z)$ $start\_video(n, vn, c(\sigma_n))$	$channel(c(\sigma_n), key)$	$get\_video(key), i = stb$	$z' = z + 1, \sigma'_z = [us, pr, 0, 0, 0, 0]$ $\sigma'_n = [*, *, *, vn, stb, key]$ $\sigma'_m = [*, *, c, *, *, *]$
3c	$get\_data(z, us, pr)$	$get\_channel(z)$ $start\_video(m, v, c(\sigma_m))$	$channel(c, k(\sigma_m))$	$get\_video(k(\sigma_m)), i = s(\sigma_m)$	$z' = z + 1, \sigma'_z = [us, pr, 0, 0, 0, 0]$ $\sigma'_m = [*, *, c, *, *, *]$ $\sigma'_n = [*, *, *, vn, stb, key]$
3d	$get\_data(z, us, pr)$	$get\_channel(z)$	$\surd$	$\surd$	$z' = z + 1, \sigma'_z = [us, pr, 0, 0, 0, 0]$ $\sigma'_m = [*, *, c, *, *, *]$ $\sigma'_n = [*, *, *, vn, stb, key]$
4a	$get\_data(z, us, pr)$ $put\_data(\sigma_r)$	$get\_channel(z)$ $start\_video(n, vn, c(\sigma_n))$	$channel(c(\sigma_n), key)$	$get\_video(key), i = stb$	$z' = z + 1,\ \sigma'_z = [us, pr, 0, 0, 0, 0]$ $\sigma'_n = [*, *, *, vn, stb, key],\ \sigma' = \sigma \setminus \sigma_r$
4b	$get\_data(z, us, pr)$ $put\_data(\sigma_r)$	$get\_channel(z)$			$z' = z + 1,\ \sigma'_z = [us, pr, 0, 0, 0, 0]$ $\sigma'_n = [*, *, *, vn, stb, key],\ \sigma' = \sigma \setminus \sigma_r$
5a	$get\_data(z, us, pr)$	$get\_channel(z)$ $start\_video(n, vn, c(\sigma_n))$	$channel(c(\sigma_n), key)$	$get\_video(key), i = stb$	$z' = z + 1, \sigma'_z = [us, pr, 0, 0, 0, 0]$ $\sigma'_n = [*, *, *, vn, stb, key]$
5b	$get\_data(z, us, pr)$	$get\_channel(z)$			$z' = z + 1, \sigma'_z = [us, pr, 0, 0, 0, 0]$ $\sigma'_n = [*, *, *, vn, stb, key]$
6a	$get\_data(z, us, pr)$	$get\_channel(z)$ $start\_video(m, v, c(\sigma_m))$	$channel(c, k(\sigma_m))$	$get\_video(k(\sigma_m)), i = s(\sigma_m)$	$z' = z + 1,\ \sigma'_z = [us, pr, 0, 0, 0, 0]$ $\sigma'_m = [*, *, c, *, *, *]$
6b	$get\_data(z, us, pr)$	$get\_channel(z)$	$\surd$	$\surd$	$z' = z + 1,\ \sigma'_z = [us, pr, 0, 0, 0, 0]$ $\sigma'_m = [*, *, c, *, *, *]$
7	$get\_data(z, us, pr)$ $put\_data(\sigma_r)$	$get\_channel(z)$	$\surd$	$\surd$	$z' = z + 1,\ \sigma'_z = [us, pr, 0, 0, 0, 0]$ $\sigma' = \sigma \setminus \sigma_r$

Tabelle 5: Belegung der Ausgabekanäle und Angabe der Nachbedingungen (Teil 1)

Belegung der Ausgabekanäle mit Nachbedingung

	din	vin	$smux$	$stb_i,\ i$	POST
8	$\surd$	$\surd$	$\surd$	$\surd$	
9a	$\surd$	$start\_video(n, vn, c)$	$channel(c, key)$	$get\_video(key), i = stb$	$\sigma'_n = [*, *, c, vn, stb, key]$
9b	$\surd$	$start\_video(n, vn, c(\sigma_n))$ $start\_video(m, v(\sigma_m), c)$	$channel(c(\sigma_n), key)$ $channel(c, k(\sigma_m))$	$get\_video(key), i = stb$ $get\_video(k(\sigma_m)), i = s(\sigma_m)$	$\sigma'_n = [*, *, *, vn, stb, key]$ $\sigma'_m = [*, *, c, *, *, *]$
9c		$start\_video(n, vn, c(\sigma_n))$	$channel(c(\sigma_n), key)$	$get\_video(key), i = stb$	$\sigma'_n = [*, *, *, vn, stb, key]$ $\sigma'_m = [*, *, c, *, *, *]$
9d	$\surd$	$start\_video(m, v, c(\sigma_m))$	$channel(c, k(\sigma_m))$	$get\_video(k(\sigma_m)), i = s(\sigma_m)$	$\sigma'_m = [*, *, c, *, *, *]$ $\sigma'_n = [*, *, *, vn, stb, key]$
9e	$\surd$	$\surd$	$\surd$	$\surd$	$\sigma'_m = [*, *, c, *, *, *]$ $\sigma'_n = [*, *, *, vn, stb, key]$
10a	$put\_data(\sigma_r)$	$start\_video(n, vn, c(\sigma_n))$	$channel(c(\sigma_n), key)$	$get\_video(key), i = stb$	$\sigma'_n = [*, *, *, vn, stb, key],\ \sigma' = \sigma \setminus \sigma_r$
10b	$put\_data(\sigma_r)$	$\surd$	$\surd$	$\surd$	$\sigma'_n = [*, *, *, vn, stb, key],\ \sigma' = \sigma \setminus \sigma_r$
11a	$\surd$	$start\_video(n, vn, c(\sigma_n))$	$channel(c(\sigma_n), key)$	$get\_video(key), i = stb$	$\sigma'_n = [*, *, *, vn, stb, key]$
11b	$\surd$	$\surd$	$\surd$	$\surd$	$\sigma'_n = [*, *, *, vn, stb, key]$
12a	$\surd$	$start\_video(m, v, c(\sigma_m))$	$channel(c, k(\sigma_m))$	$get\_video(k(\sigma_m)), i = s(\sigma_m)$	$\sigma'_m = [*, *, c, *, *, *]$
12b	$\surd$	$\surd$	$\surd$	$\surd$	$\sigma'_m = [*, *, c, *, *, *]$
13	$put\_data(\sigma_r)$	$\surd$	$\surd$	$\surd$	$\sigma' = \sigma \setminus \sigma_r$

Tabelle 6: Belegung der Ausgabekanäle und Angabe der Nachbedingungen (Teil 2)

Die Angaben hinter den Einträgen sind die Selektoren, mit denen auf die einzelnen Felder einer Auftragsnummer n zugegriffen wird. So erhalten wir beispielsweise den Kanal für die Videoübertragung von Auftragsnummer n mit $c(\sigma_n)$.

Wird für eine neu vergebene Auftragsnummer ein Eintrag im Speicher erzeugt, so werden Felder, für die noch keine aktuellen Daten vorliegen, mit 0 initialisiert. Es wird vorausgesetzt, daß 0 keine gültige Belegung eines Feldes durch aktuelle Daten darstellt.

Werte von Feldern, die bei einer Zuweisung unverändert bleiben sollen, werden mit $*$ gekennzeichnet. So ist zum Beispiel $\sigma_n = [*, *, d, *, e, *]$ eine abkürzende Schreibweise dafür, daß die Felder 3 und 5 des Eintrags für Auftragsnummer n mit den Werten d und e belegt werden, die Belegungen der Felder $1, 2, 4$ und 6 jedoch unverändert bleiben.

$\sigma \setminus \sigma_r$ bezeichnet den Speicher, aus dem der Eintrag für die Auftragsnummer r gelöscht ist.

Für die Abfrage, ob für eine Auftragsnummer m bestimmte Felder mit gültigen Werten belegt sind, führen wir die Prädikate *c_exists* und *db_exists* ein:

$c\_exists(\sigma_m) \Leftrightarrow c(\sigma_m) \neq 0$

$db\_exists(\sigma_m) \Leftrightarrow n(\sigma_m) \neq 0 \land k(\sigma_m) \neq 0 \land s(\sigma_m) \neq 0$

7.4 Diskussion der Spezifikation

Wir haben im vorherigen Abschnitt die Spezifikation der Komponente *SCM* in Form von Ein-/Ausgabetabellen angegeben. Mit dieser Spezifikation ist nun eindeutig festgelegt, wie die Komponente in jedem Zeitintervall die aktuell anliegenden Nachrichten von ihren Eingabekanälen liest und verarbeitet. Für Eingabenachrichten, die in der Tabelle nicht aufgeführt sind, ist das Verhalten der Komponente nicht festgelegt und somit beliebig; wir sprechen in diesen Fällen von Unterspezifikation. Das gleichzeitige Lesen von mehreren Eingabekanälen ist die allgemeinste Vorstellung von einer Komponente mit mehreren Ein-/Ausgabekanälen.

Wir haben uns für die Modellierung mit mehreren Eingabekanälen entschieden, da mangels einer informellen Beschreibung der Komponente *SCM* keine Information darüber vorlag, auf welche Weise *SCM* Eingabenachrichten verarbeitet. In der Diskussion über die vorliegende Spezifikation stellte sich die Frage, ob für diese Komponente auch folgende Verhaltensmodellierung adäquat ist: sämtliche Eingabeströme der Komponente werden zu einem einzigen Eingabestrom gemischt; die Komponente verfügt somit über nur mehr einen Eingabestrom, den sie sequentiell verarbeitet. Mit dieser alternativen Modellierung geht ein gewisser Grad an Freiheit bei der Spezifikation verloren. Die Nachrichten, die eine Komponente über die verschiedenen Eingabekanäle erreichen, werden von dieser erst nach einer Zeitverzögerung verarbeitet, da die Nachrichten hintereinander in aufeinanderfolgenden Zeitintervallen verarbeitet werden. Zudem gilt, daß die Komponente bei jedem Lesen der

Eingabe nur noch einen Teil der gleichzeitig möglichen Verarbeitungsschritte durchführt[1].

In zukünftigen Arbeiten ist zu untersuchen, ob die Vorstellung eines einzelnen Eingabestroms für Komponenten einer Dienststeuerung eine geeignete Sichtweise ist oder ob es Komponenten gibt, die tatsächlich auf die Nachrichten unterschiedlicher Eingabekanäle gleichzeitig reagieren müssen. Darüber hinaus ist es in FOCUS möglich, einen Eingabekanal in mehrere Kanäle bzw. mehrere Eingabekanäle in einen Kanal zu verfeinern, so daß auf jeder Abstraktionsebene eines Modells die Art von Kommunikationsverbindungen modelliert werden kann, die für die Abstraktionsebene adäquat ist.

8 Weitere Modellierungsaspekte

In den vorangegangenen Abschnitten haben wir die Spezifikation des Home Shopping Systems in FOCUS vorgestellt. Dabei wurde das System als Netzwerk von Komponenten spezifiziert, dessen Struktur während des Systemablaufs fest ist. Im folgenden erläutern wir, wie sich dynamisches und kontinuierliches Verhalten in einer Systemspezifikation ausdrücken lassen; diese Verhaltensaspekte sind charakteristisch für Dienststeuerungen von Multimediadiensten. Daneben diskutieren wir die Einbindung pragmatischer Beschreibungstechniken in die Systementwicklung mit FOCUS.

8.1 Dynamische Verhaltensanteile

Bei der Modellierung verteilter Systeme ist es oft notwendig und wünschenswert, nicht von einer festen Anzahl von Komponenten auszugehen, sondern zusätzlich das dynamische Erzeugen neuer Komponenten während des Systemablaufs zuzulassen sowie die Kommunikationsstruktur variabel zu gestalten. Allgemein gilt für Kommunikationssysteme, daß Komponenten während des Systemablaufs hinzukommen oder gelöscht werden (etwa neue Vermittlungsknoten, neue Anschlüsse) und Aufbau, Abbau und Umleiten von Verbindungen typische Vermittlungsaktivitäten darstellen. Kommunikationssysteme sind somit in hohem Maße dynamische Systeme; diese Eigenschaft ist in der Dienststeuerung zu berücksichtigen.

Für das FOCUS-Modell, das unserer Systemspezifikation zugrunde liegt, existiert eine Erweiterung, mit der die Modellierung von dynamischen Systemen in zweierlei Hinsicht unterstützt wird:

- Für die Kanäle, die innerhalb des Systems und zur Systemumgebung existieren, werden Schreib- und Leserechte vergeben. Eine Komponente darf nur auf Kanäle zugreifen, für die sie die Lese- und Schreibrechte besitzt. Während des Systemablaufs

[1] Das Mischen von Strömen zu einem einzigen Eingabestrom ähnelt dem Vorgehen in SDL ([IT93]). Jeder SDL-Prozeß verfügt über einen Eingabepuffer, in den alle eintreffenden Signale eingetragen werden.

ändert sich die Menge der verfügbaren Ein- und Ausgabekanäle einer Komponente, so daß dadurch eine dynamische Veränderung der Kommunikationsverbindungen modellierbar ist.

- Das Erzeugen einer neuen Komponente während des Systemablaufs wird in der Spezifikation des Systems durch einen Verfeinerungsschritt modelliert. Durch einen rekursiven Funktionsaufruf innerhalb der auf Seite 17 vorgestellten Gleichungen wird eine neue Komponente erzeugt und in die bestehende Systemstruktur eingebunden.

Für die Modellierung von dynamischem Systemverhalten steht eine Sammlung von Leitfäden und Spezifikationsschemata zur Verfügung ([HS97]), wodurch ein systematisches Vorgehen bei der Spezifikation unterstützt wird.

8.2 Kontinuierliche Verhaltensanteile

Ein Multimediasystem ist nach [Ste93] durch die rechnergesteuerte, integrierte Erzeugung, Manipulation, Darstellung, Speicherung und Kommunikation von unabhängigen Informationen gekennzeichnet, die in mindestens einem kontinuierlichen (zeitabhängigen) und einem diskreten (zeitunabhängigen) Medium kodiert sind. Damit sind in der Dienststeuerung von Multimediadiensten sowohl kontinuierliche als auch diskrete Ströme zu behandeln – es handelt sich um ein *hybrides System*. Die in dieser Arbeit erstellte Focus-Modellierung basiert ausschließlich auf diskreten Strömen. So wird zum Beispiel die Übertragung eines Videofilms durch eine Sequenz von Datenpaketen und nicht als kontinuierlicher Datenstrom modelliert. Diese Abstraktion ist jedoch adäquat, da wir an Verhaltensaspekten interessiert sind, die in Zusammenhang mit Aufgaben der Dienststeuerung, nicht jedoch mit der Regelung von kontinuierlichen Strömen verbunden sind. In zukünftigen Arbeiten ist es denkbar, die Umsetzung von kontinuierlichen Verhaltensanteilen in Focus vorzunehmen. Dazu ist die Wahl eines alternativen Focus-Modells für hybride Systeme erforderlich.

In [GS98] wird eine neuartige graphische Beschreibungstechnik, HyCharts, für solche Systeme definiert. HyCharts sind an die in der Industrie weit verbreiteten Statecharts [Har87] angelehnt. Sie sind sowohl anschaulich als auch formal fundiert und - im Gegensatz zu anderen Bereibungstechniken in diesem Gebiet - modular.

8.3 Einbindung pragmatischer Beschreibungstechniken

Im Rahmen der Systementwicklung, wie wir sie in Abschnitt 4 skizziert haben, ist es möglich, pragmatisch orientierte, ingenieurtechnische Beschreibungstechniken aus dem industriellen Anwendungsbereich einzubinden. So können zum Beispiel für die Verhaltensbeschreibung von Komponenten SDL [IT93] oder Statecharts [Har87] eingesetzt werden. Voraussetzung dafür ist jedoch, daß diese Beschreibungstechniken eine formale Semantik im semantischen Modell von Focus erhalten. So sind die in Abschnitt 4 verwendeten

Erweiterten Ereignisdiagramme in Anlehnung an Message Sequence Charts MSC [IT96] in Focus integriert worden. Für SDL und Teile von UML existiert bereits eine formale Fundierung, so daß diese Beschreibungstechniken in den Entwicklungsprozeß mit Focus aufgenommen werden können. So kann untersucht werden, ob sich das Verhalten einer Komponente anstelle von Funktionsgleichungen auch graphisch durch eine existierende pragmatische Beschreibungstechnik definieren läßt.

Auch wenn graphische Beschreibungstechniken bei Anwendern in der Industrie auf hohe Akzeptanz stoßen, ist zu bedenken, daß bei umfangreichen Spezifikationen graphische Darstellungen unübersichtlich werden und somit textuelle Darstellungen kompakter sind.

9 Schlußbemerkung

In diesem Bericht wurde die schrittweise Entwicklung einer Systemspezifikation mit Focus am Beispiel eines Home Shopping Systems vorgestellt. Im folgenden fassen wir die wesentlichen Ergebnisse dieser Arbeit zusammen.

Ausgangspunkt für die Spezifikation mit Focus war eine informelle Beschreibung des Home Shopping Systems. Bereits bei der Modellierung der Architektur in Focus wurden einige Unklarheiten in der informellen Beschreibung aufgedeckt. Auch für die Spezifikation der Komponente *SmartComManager* waren Informationen erforderlich, die in der informellen Beschreibung nicht enthalten sind. Hier wurde deutlich, welch komplexes Verhalten die einzelnen Komponenten des Systems aufweisen und daß die informelle Beschreibung nur ein sehr grobes Bild des zu implementierenden Systems wiedergibt. Unsere Erfahrungen zeigen, daß die Entwicklung einer formalen Spezifikation auf abstraktem Niveau zu einer intensiven Auseinandersetzung mit der Aufgabenstellung führt. Als Folge davon werden Ungenauigkeiten und Lücken in der informellen Anforderungsspezifikation aufgedeckt. Wesentlich ist, daß diese Punkte bereits am Beginn der Systementwicklung und nicht erst bei der Implementierung des Systems aufgedeckt werden.

Anhand der Spezifikation des Home Shopping Systems wurde deutlich, daß Focus einen umfassenden Rahmen für die Modellierung unterschiedlicher Sichten auf ein System bildet: wir haben Struktur, Nachrichtenaustausch und Verhalten von Komponenten in Focus modelliert; dynamische Aspekte, Zeitanforderungen und kontinuierliche Verhaltensanteile sind ebenfalls in Focus spezifizierbar. Damit ist mit den Spezifikationsmitteln von Focus eine adäquate Entwicklung von Kommunikationssystemen, insbesondere von Dienststeuerungen, möglich. Diese Erfahrung wird durch [Sch97] bestätigt, das eine Videokonferenz im Rahmen von TINA ([Tin98]) mit Beschreibungstechniken aus Focus formalisiert. Von besonderem Interesse für die Modellierung von Aufgaben der Dienststeuerung sind die Erweiterten Ereignisdiagramme. Um zeitliche Anforderungen erfassen zu können, ist hier eine Erweiterung der Ereignisdiagramme um Zeitaspekte wünschenswert.

Mit der vorliegenden Fallstudie haben wir gezeigt, daß Focus einen geeigneten Entwicklungsrahmen für Kommunikationssysteme darstellt. In zukünftigen Arbeiten werden wir ein methodisches Vorgehen für die formale Spezifikation von Dienststeuerungen für Multimediasysteme entwickeln.

Danksagung

Diese Arbeit wurde in Kooperation mit dem Teilprojekt A6 des Sonderforschungsbereichs 342 „Werkzeuge und Methoden für die Nutzung paralleler Rechnerarchitekturen“ durchgeführt. Ich danke Max Breitling aus A6 für die kooperative Zusammenarbeit und die genaue Durchsicht einer Vorversion dieses Berichts. Katharina Spies danke ich für konstruktive Hinweise zu einer Vorversion dieses Berichtes. Weiterhin danke ich Wolfgang Kellerer und Peter Sties aus dem Teilprojekt C2 von FORSOFT für ihre Rolle als „Auftraggeber“ der Spezifikation und für Kommentare zu Vorversionen des Berichts.

Literatur

[BFG+93] Manfred Broy, Christian Facchi, Radu Grosu, Rudi Hettler, Heinrich Hußmann, Dieter Nazareth, Franz Regensburger, Oscar Slotosch und Ketil Stølen. The Requirement and Design Specification Language SPECTRUM- Part I. Technischer Bericht TUM-I9311, Technische Universität München, Mai 1993.

[BHS98] Max Breitling, Ursula Hinkel und Katharina Spies. Formale Entwicklung verteilter reaktiver Systeme mit FOCUS. In H. König, Hrsg., *erscheint als Beitrag des 8. GI/ITG Fachgespräch – Formale Beschreibungstechniken für verteilte Systeme*. Technische Universität Cottbus, 1998.

[Bro86] Manfred Broy. A theory for nondeterminism, parallelism, communication and concurrency. *Theoretical Computer Science*, (45):1 – 61, 1986.

[Foc98] Homepage des FOCUS-Projektes mit allen relevanten Veröffentlichungen. Verfügbar im World Wide Web unter http://www4.informatik.tu-muenchen.de/proj/focus/, 1998.

[GS98] Radu Grosu und Thomas Stauner. A Modular Visual Model for Hybrid Systems. Technischer Bericht TUM-I9801, Technische Universität München, Institut für Informatik, 1998.

[Har87] David Harel. Statecharts: a visual formalism for complex systems. *Science of Computer Programming*, 8:231–274, 1987.

[HKS97] Ursula Hinkel, Wolfgang Kellerer und Peter Sties. Multimediale Anwendungen in der Telekommunikation – Szenarien und Abläufe. FORSOFT C2-Bericht, Technische Universität München, 1997.

[HS97] Ursula Hinkel und Katharina Spies. Spezifikationsmethodik für mobile, dynamische FOCUS-Netze. In A. Rennoch A. Wolisz, I. Schieferdecker, Hrsg., *Formale Beschreibungstechniken für verteilte Systeme, 7. GI/ITG Fachgespräch 1997*. GMD Verlag (St.Augustin), 1997.

[HSS96] Franz Huber, Bernhard Schätz und Katharina Spies. AutoFocus - Ein Werkzeugkonzept zur Beschreibung verteilter Systeme. In Ulrich Herzog, Hrsg., *Formale Beschreibungstechniken für verteilte Systeme, 6. GI/ITG Fachgespräch – Formale Beschreibungstechniken für verteilte Systeme*. Universität Erlangen-Nürnberg, 1996.

[IT93] ITU-T. *Recommendation Z.100, Specification and Description Language (SDL)*. CCITT, 1993.

[IT96] ITU-T. *Z.120 – Message Sequence Chart (MSC)*. ITU-T, Geneva, 1996.

[Kah74] G. Kahn. The Semantics of a Simple Language for Parallel Programming. *Information Processing*, Seiten 471 – 475, 1974.

[KS98] Wolfgang Kellerer und Peter Sties. Entwicklung zukünftiger multimedialer Kommunikationsanwendungen am Beispiel eines Home Shopping Systems. Interner Bericht des Projekts FORSOFT – C2, Technische Universität München, 1998.

[Sch97] Alexander Schmidt. A TINA-Related Specification of a Videoconference. Interner Bericht, Technische Universität München, 1997.

[SS95] Bernhard Schätz und Katharina Spies. Formale Syntax zur logischen Kernsprache der FOCUS- Entwicklungsmethodik. SFB-Bericht 342/16/95 A, Technische Universität München, Institut für Informatik, 1995.

[Ste93] Ralf Steinmetz. *Multimedia-Technologie*. Springer Verlag Berlin, 1993.

[Tin98] Homepage des TINA-Konsortiums mit allen relevanten Veröffentlichungen. Verfügbar im World Wide Web unter http://www.tinac.com, 1998.