# Zur Rolle der Zeit in verteilten multimedialen Systemen

Winfried Kühnhauser and Martin Süßkraut

Technische Universität Ilmenau

**Zusammenfassung** Multimediale Anwendungssysteme besitzen einen extremen Bedarf an Rechenressourcen und können vielfach nur unter hohen, für viele Anwendungsszenarien unakzeptablen Qualitätseinschränkungen realisiert werden. Besonders deutlich wird dies in verteilten Anwendungen, die mobile Systeme und drahtlose Kommunikationsmedien umfassen und sich aus diesem Grund heute noch vielen attraktiven Anwendungsfeldern verschließen.
Ein wesentlicher Grund für den Ressourcenhunger multimedialer Anwendungssysteme liegt in der Verbindung aus Volumen und den Echtzeitanforderungen multimedialer Datenströme, die in derartigen Applikationen kommuniziert und verarbeitet werden. Paradigmen, Modelle und Mechanismen, die Ressourcenbedarf auf der einen Seite und Ressourcenverfügbarkeit auf der Anderen vereinen sind daher in diesem Anwendungsfeld von größter Bedeutung.
Thema dieses Beitrags sind adaptive Kommunikationsmodelle für verteilte multimediale Systeme. Adaptive Kommunikationsmodelle enthalten Strategien, die Informationen über den aktuellen Zustand des Kommunikationssystems bei der Suche nach einer Balance zwischen Ressourcenbedarf und Ressourcenverfügbarkeit einfließen lassen. Dieses Papier betrachtet in diesem Kontext die Rolle der Zeit.

## 1 Einleitung

Multimediale Anwendungssysteme erschließen Anwendungsszenarien, in denen Effizienz, Mühelosigkeit und Informativität der Kommunikation von Menschen mit oder mittels IT-Systemen eine besondere Rolle spielen. Zahllose Anwendungen in der Kommunikationstechnik, der Luft- und Raumfahrt, Telemedizin, Verkehrstelematik und Produktionstechnik werden durch multimediale Kommunikationstechniken grundsätzlich erst machbar.

Die bereits realisierten Systeme machen allerdings auch die Leistungsgrenzen heutiger IT-Systeme deutlich. Ursache hierfür sind die extensiven Prozessor-, Speicher- und Kommunikationsressourcen, die für die Verarbeitung und Kommunikation kontinuierlicher multimedialer Datenströme erforderlich sind. Multimediale Applikationen stellen zudem nichtfunktionale Anforderungen bezüglich der Rechtzeitigkeit, Zuverlässigkeit und IT-Sicherheit, denen die heutigen Kommunikationsmodelle und -infrastrukturen verteilter Systeme architektonisch und algorithmisch nicht gewachsen sind [EK02].

Thematisches Umfeld dieses Papiers sind adaptive Kommunikationsmodelle. Adaptive Modelle nutzen bei der Suche nach einer Balance zwischen Ressourcenbedarf und Ressourcenverfügbarkeit Informationen über den aktuellen Zustand des Kommunikationssystems und lassen diese in strategische Entscheidungen einfließen, die beispielsweise im Rahmen der Flusskontrolle oder der Reaktion auf Kommunikationsfehler zu treffen sind. In diesem thematischen Kontext liegt der Fokus des Papiers

auf der Fehlersemantik eines Kommunikationsmodells: derjenigen Modellkomponente, die mit der prinzipiellen Unmöglichkeit umgeht, in Umgebungen mit möglichen Kommunikationsfehlern eine präzise *exactly-once*-Semantik einer Kommunikationsoperation zu erzielen. Die Fehlersemantik eines Kommunikationsmodells spezifiziert in präziser Form diejenigen Garantien, die den Kommunikationspartnern auch unter Anwesenheit von Kommunikationsfehlern gegeben werden. Ihr Sinn ist es, die Fehlerbehandlung auf der Anwendungsebene zu vereinfachen, indem sie Anwendungsprogrammierern ein einfach zu handhabendes Modell zum Umgang mit Fehlern bereitstellt, welches ihm erlaubt, von den komplexen Details der Fehlerbehandlung zu abstrahieren und Fehlerbehandlungsstrategien auf hoher Abstraktionsebene zu formulieren.

Strategien in adaptiven Fehlersemantiken, die einen sich dynamisch ändernden Kommunikationskontext einbeziehen, ruhen auf drei elementaren Säulen:

- dem *Stromkontext* zur Erkennung eines Fehlers und zur Beurteilung seiner Wichtigkeit
- dem *Ressourcenkontext* sowie
- dem *Zeitkontext* zur Beurteilung der Reparaturkosten und -möglichkeiten.

Arbeiten zur Rolle des Stromkontextes [Hof03,HK03] haben gezeigt, dass die Berücksichtigung des Stromkontextes in einer Fehlersemantik in MPEG-2 Datenströmen zu einer Qualitätsverbesserung von 10-20dB nach dem PSNR-Qualitätsmaß führt; bei Einhaltung der kritischen Qualitätsgrenze von 20dB PSNR führt dies zur Tolerierung einer um mindestens den Faktor 5 höheren Frameverlustrate. Diese Arbeiten machten allerdings gleichzeitig deutlich, dass derartige Ergebnisse nur dann zu erreichen sind, wenn der Faktor Zeit ausreichend scharf beurteilt werden kann. Ein Zeitmodell, welches genau dies erreicht wird in diesem Papier skizziert.

## 2 Fragen an ein Zeitmodell

Aufgabe eines Zeitmodells im Rahmens einer adaptiven Fehlersemantik für verteilte multimediale Anwendungssysteme ist es, zeitbezogene Kontextinformationen bereitzustellen, die eine effiziente und ökonomische Reaktion auf Kommunikationsfehler ermöglichen. In diesem Abschnitt werden Kernfragen diskutiert, die hierzu aus der Sicht der Fehlersemantik durch ein Zeitmodell beantwortet werden müssen.

Die Terminologie dieses Kapitels im Hinblick auf das Szenario multimedialer Anwendungssysteme folgt [EK02]. Insbesondere soll daran erinnert werden, dass in verteilten multimedialen Anwendungssystemen ein Datenstrom von einer Quelle zu einer Senke fließt und dabei weiter weitere Anwendungskomponenten (Filter) passieren kann, die den Strom manipulieren. Ein Datenstrom selbst besteht aus einer Folge diskreter Stromobjekte, von denen jedes ein Medienobjekt (wie beispielsweise ein Videobild oder ein Audiosample) transportiert.

### 2.1 Fehlererkennung: Erkennung von Zeitfehlern

Viele multimediale Anwendungen verarbeiten Datenströme unter Echtzeitbedingungen, in denen ein außerhalb seiner zeitlichen Spezifikationen eintreffendes Stromobjekt eine Fehlersituation bedeutet. Fehler dieser Klasse werden im Allgemeinen als Zeitfehler bezeichnet. Zur Erkennung von Zeitfehlern muss ein Fehlermodell Informationen über die Rechtzeitigkeit eines eintreffenden Stromobjekts liefern, also im Kern die Frage

**(1)** Ist das Stromobjekt zu spät?

beantworten. Zur Beantwortung dieser Frage braucht man ebenfalls den erwarteten Eintreffzeitpunkt des Stromobjektes, so dass hieraus die elementarere Fragestellung

**(1.a)** Zu welchem Zeitpunkt soll das Stromobjekt eintreffen?

abgeleitet werden kann.

### 2.2 Rechtzeitigkeit der Fehlerkorrektur: Notwendige Bedingung

Bei einer Verarbeitung in Echtzeit wird jedes Stromobjekt zu einem bestimmten Zeitpunkt bei der Senke erwartet. Trifft es später ein, ist es nutzlos. Sollte ein festgestellter Fehler nicht so korrigiert werden können, dass ein korrigiertes Stromobjekt noch rechtzeitig bei seiner Senke ankommen kann, so kann es unmittelbar verworfen werden. Dadurch werden die Ressourcen vor der Senke entlastet.

Um derart ressourcenschonende Strategien realisieren zu können, muss die Frage

**(2)** Wird ein gegebenes Stromobjekt rechtzeitig bei der Senke eintreffen?

beantwortet werden können. Zur Beantwortung dieser Frage müssen wiederum als zwei Teilaspekte die Fragen

**(2.a)** Wann wird das Stromobjekt an der Senke gebraucht?
**(2.b)** Wie lange braucht das Stromobjekt bis zur Senke?

beantwortet werden.

### 2.3 Rechtzeitigkeit der Fehlerkorrektur: Hinreichende Bedingung

Als Reaktion auf einen Fehler lohnt sich die Wiederbeschaffung eines Stromobjektes nur, wenn es danach noch zeitgerecht weiter verarbeitet werden kann. Anderenfalls ist eine solche Maßnahme eine Ressourcenverschwendung, durch welche die weitere Verarbeitung des Datenstromes behindert werden kann. Ressourcenschonung impliziert hier, dass die Frage

**(3)** Kann das Stromobjekt rechtzeitig wiederbeschafft werden?

beantwortet wird. Auch aus dieser Frage ergeben sich zwei Unterfragen:

**(3.a)** Wie lange dauert die Wiederbeschaffung des Stromobjektes?
**(3.b)** Bis wann muss das Stromobjekt wiederbeschafft sein?

## 3 Das Zeitmodell

Dieser Abschnitt skizziert ein Zeitmodell, welches Antworten auf die Fragestellungen in Kapitel 2 gibt. Im Modell wird die Zeit $T$ durch reelle Zahlen aus $\mathbb{R}_0^+$ modelliert. Die Menge der Medienobjekte und die Menge der Stromobjekte werden durch $M$ bzw. $O$ beschrieben. Das vom Stromobjekt transportierte Medienobjekt wird durch $MO : O \rightarrow M$ definiert. $P$ steht für die Menge aus Quelle, Senke und allen Filtern der multimedialen Anwendung, im folgenden auch als Prozessoren bezeichnet.

Das Modell operiert mit vier verschiedenen Zeiten, welche die Erscheinungsformen der Zeit in multimedialen Dokumenten und Datenströmen im Kontext verteilter multimedialer Anwendungen reflektieren.

### 3.1 Medienzeit

Medienobjekte innerhalb eines multimedialen Dokumentes sind untereinander kausal geordnet. Implementiert wird diese Ordnung durch Zeitstempel, die bei der Erzeugung eines Medienobjekts als Metainformation dem Objekt aufgeprägt werden. Jedem Medienobjekt ist folglich eine *Medienzeit* $t_M : M \rightarrow T$ zugeordnet, die innerhalb eines Dokuments eindeutig ist. Die Medienzeit innerhalb eines multimedialen Dokuments beginnt mit dem ersten Medienobjekt bei 0 und steigt streng monoton an.

In einem periodischen Dokument lässt sich die Medienzeit aus der Periode berechnen. Ein Videodokument beispielsweise mit einer zeitlichen Auflösung von 25 Bildern je Sekunde hat $n$ Einzelbilder, die im Strom jeweils durch ein Medienobjekt $m_i, i \in 0, ..., n-1$ dargestellt werden; damit kann dann die Medienzeit $t_M$ eines jeden Medienobjekts $m_i$ berechnet werden durch

$$t_M(m_i) = i * \frac{1}{25} \ .$$

### 3.2 Strömungszeit

Die *Strömungszeit* $t_S : P \times O \rightarrow T$ beschreibt den Zeitpunkt des Eintreffens eines Stromobjekts bei einem Prozessor. Für Prozessoren, bei denen das Stromobjekt noch nicht eingetroffen ist (und es eventuell auch nie wird), ist die Strömungszeit nicht definiert. Die Quelle ist ein Sonderfall: $t_S$ bezeichnet hier den Zeitpunkt des Verlassens der Quelle.

Wird ein Datenstrom in Echtzeit verarbeitet, so unterscheidet sich die Medienzeit eines Medienobjekts und Strömungszeit des korrespondierenden Stromobjekts auf einem Prozessor um einen konstanten Summanden, dessen Größe von der Entfernung des Prozessors von der Quelle abhängt. Im Allgemeinen kann ein Datenstrom ähnlich wie bei einem Videorecorder jedoch auch schneller (z.B. für Kopieroperationen) oder langsamer (Zeitlupe) abgespielt werden, als dies vom zugrundeliegenden Dokument vorgegeben ist. Im Modell wird diese Eigenschaft durch den Geschwindigkeitsfaktor $v \in \mathbb{R}$ ausgedrückt. Für $v = 1$ wird der Datenstrom mit der vom Dokument vorgegebenen Geschwindigkeit verarbeitet, für $v > 1$ schneller, für $v < 1$ langsamer. $v < 0$ bedeutet, dass der Datenstrom rückwärts abgearbeitet wird. Im Falle von $v = 0$ wurde die Verarbeitung des Datenstromes pausiert.

### 3.3 Stromglobale logische Uhren

Zur Bestimmung der Strömungszeit wird eine Echtzeituhr benötigt, deren Uhrzeit $t_G :\rightarrow T$ global für alle Prozessoren des Datenstromes ist. Diese stromglobale Uhr steht in direktem Bezug zur Echtzeit und wächst streng monoton. Da in verteilten Systemen im Allgemeinen eine globale Echtzeituhr nicht vorausgesetzt werden kann, wird sie mittels Algorithmen zur Uhrensynchronisation realisiert, die auf der Grundlage der lokalen Rechneruhren arbeiten.

Die Strömungszeit $t_S$ wird mittels der stromglobalen logischen Uhr $t_G$ errechnet mittels

$$t_S(p, o) = t_G \ ,$$

sobald ein Stromobjekt $o$ bei einem Prozessor $p$ eintrifft.

### 3.4 Lokale Uhren

Grundlage der Implementierung der stromglobalen logischen Uhr $t_G$ sind die *lokalen Uhren* $t_L : P \rightarrow T$ der beteiligten Prozessoren. Da für zwei unterschiedliche Prozessoren $p$ und $p'$ auf Grund der unterschiedlichen Driftraten ihrer Hardware-Uhren nicht angenommen werden kann, dass $t_L(p) = t_L(p')$ über einen längeren Zeitraum gilt, wird $t_G$ aus den $t_L(p)$ durch einen Uhrensynchronisationsalgorithmus erzeugt, der auf der Grundlage der lokalen physischen Uhren und ihren bekannten Driftraten eine vorgegebene Synchronisationsschärfe der lokalen Uhren $t_L(p)$ herstellt.

## 4 Die Antworten des Modells

Das vorangehende Kapitel skizzierte ein Zeitmodell mit vier Formen der Zeit und beschrieb ihre Rollen im Modell und ihre Beziehungen untereineinder. Dieses Kapitel zeigt, wie auf dieser Grundlage die in Kapitel 2 gestellten Fragen zu beantworten sind.

Da es in verteilten Anwendungen keinen globalen Systemzustand gibt, kann es sein, dass einige Größen in den folgenden Antworten nicht zu jedem Zeitpunkt und bei jedem Prozessor bekannt sind. Größen wie die Medienzeit können aber auch deshalb unbekannt sein, weil sie sich nicht ableiten lassen[1]. In derartigen Fällen lassen sich die betreffenden Fragen dann aus prinzipiellen Gründen nicht beantworten.

### 4.1 Fehlererkennung: Erkennung von Zeitfehlern

Zur Erkennung von Zeitfehlern war die Frage

**(1)** Ist das Stromobjekt zu spät?

zu beantworten und hierzu als Konkretisierung die Frage

**(1.a)** Zu welchem Zeitpunkt soll das Stromobjekt eintreffen?

Der erwartete Eintreffzeitpunkt eines Stromobjektes $o \in O$ bei einem Prozessor $p \in P$ lässt sich auf verschiedene Art und Weise vorhersagen.

- Falls $t_S(q, o)$ mit $q \in P$ Quelle von $o$ bekannt ist, gilt:

  $$\mathit{Eintreffzeit}(p, o) \approx t_S(q, o) + \overline{\mathit{Laufzeit}}(q, p)$$

  wobei $\overline{\mathit{Laufzeit}}(q, p)$ ein gewichteter Durchschnitt der Laufzeiten von der Quelle $q$ zum Prozessor $p$ der zuletzt bei $p$ eingetroffenen Stromobjekte $o' \in O$, mit $\mathit{Laufzeit}(q, p, o') = t_S(p, o') - t_S(q, o')$ ist.
- Eine alternative Möglichkeit ist:

  $$\mathit{Eintreffzeit}(p, o) \approx t_S(p, o') + \frac{t_M(MO(o)) - t_M(MO(o'))}{v}$$

  mit $o' \in O$, $t_S(p, o')$ bekannt und $v$ ist seit der Übertragung von $o'$ unverändert. Da aber die Übertragung von $o'$ Unregelmäßigkeiten unterworfen sein kann, ist es besser mit einem mittleren $\overline{\mathit{Eintreffzeitpunkt}}$ analog zur $\overline{\mathit{Laufzeit}}$ zu arbeiten.

Falls ein Stromobjekt $o \in O$ bei einem Prozessor $p \in P$ noch nicht eingetroffen ist und

$$t_G > \mathit{Eintreffzeit}(p, o)$$

gilt, dann trifft $o$ bei $p$ zu spät ein.

[1] Beispielsweise ist die Medienzeit in aperiodischen Dokumenten ohne weitere Metainformationen i. A. nicht vorhersagbar.

### 4.2 Rechtzeitigkeit der Fehlerkorrektur: Notwendige Bedingung

Notwendige Bedingung für eine erfolgreiche Fehlerkorrektur ist ihre Rechtzeitigkeit; daher war die Frage

**(2)** Wird ein gegebenes Stromobjekt rechtzeitig bei der Senke eintreffen?

zu beantworten. Zwei Teilaspekte sind von Wichtigkeit:

**(2.a)** Wann wird das Stromobjekt an der Senke gebraucht?

Gibt die Anwendung eine Ende-Zu-Ende Verzögerung $EZE \in T$ vor, dann gilt:

$$EintreffzeitSenke(o) = t_S(q, o) + EZE$$

$\forall o \in O$ mit $q \in P$ Quelle von $o$.

Ist $EZE$ nicht definiert, dann kann der Eintreffzeitpunkt an der Senke auf den allgemeinen Fall (siehe 4.1) zurückgeführt werden:

$$EintreffzeitSenke(o) = Eintreffzeit(s, o)$$

$\forall o \in O$ mit $s \in P$ Senke von $o$.

**(2.b)** Wie lange braucht das Stromobjekt bis zur Senke?

Die Restlaufzeit eines Stromobjektes $o \in O$ von einem Prozessor $p \in P$ aus wird so berechnet:

$$Restlaufzeit(p, o) = t_S(s, o) - t_S(p, o)$$

$\forall o \in O \wedge p \in P$ mit $s \in P$ Senke von $o$.

Da im Allgemeinen $t_S(s, o)$ nicht bekannt ist, wenn $o$ noch nicht die Senke $s$ erreicht hat, kann $Restlaufzeit(p, o)$ mit Hilfe der Restlaufzeiten von bereits bei $s$ angekommenen Stromobjekten abgeschätzt werden.

Falls gilt:

$$EintreffzeitSenke(o) \geq Restlaufzeit(p, o) + t_G$$

dann kann das Stromobjekt $o \in O$ seine Senke vom Prozessor $p \in P$ aus noch rechtzeitig erreichen.

Ist $Restlaufzeit(p, o)$ nicht bekannt, kann sie für obige Ungleichung mit

$$Restlaufzeit(p, o) > \min_{o' \in O \ \wedge \ Restlaufzeit(p,o') \ bekannt} Restlaufzeit(p, o')$$

sinnvoll abgeschätzt werden.

### 4.3 Rechtzeitigkeit der Fehlerkorrektur: Hinreichende Bedingung

Hinreichende Bedingung für eine erfolgreiche Fehlerkorrektur ist eine positive Beantwortung der Frage

**(3)** Kann das Stromobjekt rechtzeitig wiederbeschafft werden? eintreffen?

Wiederum sind zwei Teilaspekte von Wichtigkeit:

**(3.a)** Wie lange dauert die Wiederbeschaffung des Stromobjektes?

Die $\mathit{Wiederbeschaffungsdauer}(p, o)$ ist abhängig vom Wiederbeschaffungsalgorithmus. Die Dauer der Anforderung des Stromobjektes von einem Prozessor $p'$ kann nicht vom Zeitmodell beantwortet werden.

Die Dauer einer erneuten Verarbeitung eines Stromobjektes $o$ vom Prozessor $p'$ zu Prozessor $p$ kann mit $\overline{\mathit{Laufzeit}}(p', p)$ abgeschätzt werden.

**(3.b)** Bis wann muss das Stromobjekt wiederbeschafft sein?

Je nach Auslegung der Frage sind zwei Antworten möglich:

1. $\mathit{WiederbeschaffungBis}(p, o) = \mathit{Eintreffzeit}(p, o)$
2. $\mathit{WiederbeschaffungBis}(p, o) = \mathit{EZE} - \mathit{Restlaufzeit}(p, o)$

mit $p \in P$ und $o \in O$.

Falls

$$t_G + \mathit{WiederbeschaffungsDauer}(p, o) < \mathit{WiederbeschaffungBis}(p, o)$$

gilt, kann $o \in O$ noch rechtzeitig für die Verarbeitung auf $p \in P$ wiederbeschafft werden.

## 5 Zusammenfassung

Adaptive Fehlersemantiken für Kommunikationsmodelle verteilter multimedialer Systeme benötigen für eine effiziente und ressourcenschonende Behandlung von Kommunikationsfehlern Wissen über zeitliche Beziehungen zwischen den Objekten innerhalb der Datenströme und ihrer Umgebung. Die Bereitstellung dieses Wissens ist Aufgabe des in diesem Papier skizzierten Zeitmodells.

Ausgehend von den Anforderungen einer adaptiven Fehlersemantik entstand das Modell aus einer Untersuchung der Rollen und Erscheinungsformen der Zeit bei der Kommunikation multimedialer Datenströme. Obwohl nicht Gegenstand dieses Papiers sind die Informationen des Zeitmodells außer in Fehlersemantiken in zahlreichen weiteren Komponenten einer multimedialen Kommunikationsinfrastruktur, so beispielsweise in der Fluss-, Synchronisierungs- und Adaptionssteuerung, von strategischer Bedeutung.

Die nächsten Schritte werden die Implementierung und experimentelle Validation des Modells in Bezug auf seine Leistungen und Grenzen sein.

## Literatur


[EK02] Alexander Eichhorn and Winfried E. Kühnhauser. Datenströme in multimedialen Systemen. In *Proceedings of NET.OBJECTDAYS 2002 – 8th Workshop on Multimedia Information and Communication Systems*, pages 687–695, October 2002. ISBN 3-9808628-1-X.

[Hel94] Tobias Helbig. Development and Control of Distributed Multimedia Applications. Technical report, University of Stuttgart / IPVR, 1994.

[HK03] Martin Hoffmann and Winfried E. Kühnhauser. Towards a Structure-Aware Failure Semantics for Streaming Media Communication Models, 2003. Journal submission, May 2003.

[Hof03] Martin Hoffmann. On Failure Semantics for Multimedia Streaming Applications. Diploma Thesis 2003-02-07/015/IN95/2255, Technical University of Ilmenau, February 2003.

[ZSM02] Aidong Zhang, Yuqing Song, and Markus Mielke. NetMedia: Streaming Multimedia Presentations in Distributed Environments. *IEEE MultiMedia*, 9(1):56–73, January 2002.