Eine Netzwerkarchitektur zum Einsatz des Material Exchange Formats für Live-Produktionen im professionellen Fernsehstudio

Dissertation
zur Erlangung des akademischen Grades
Doktoringenieur (Dr.-Ing.)

vorgelegt der Fakultät für Elektrotechnik und Informationstechnik
der Technischen Universität Ilmenau

von

Dipl.-Ing. Jan Röder
geboren am 25.06.1976
in Neubrandenburg

Tag der Einreichung: 25. Juni 2009
Tag der Verteidigung: 18. Dezember 2009

Gutachter: 1. Univ.-Prof. Dr.-Ing. Hans-Peter Schade
2. Univ.-Prof. Dr. rer. nat. habil. Jochen Seitz
3. Prof. Dr.-Ing. Rolf Hedtke

urn:nbn:de:gbv:ilm1-2010000020

Kurzfassung

Der Bereich der Liveproduktion im Fernsehstudio ist geprägt von hohen Anforderungen an Qualität, Zeitverhalten und Zuverlässigkeit bei der Erstellung von Audio- und Videomaterial zur Distribution über Broadcastkanäle. In der Vergangenheit konnten diese Anforderungen nur mit spezieller und damit kostenintensiver Gerätetechnik bewältigt werden. Mit der Entwicklung auf dem Elektroniksektor ist heute einerseits eine Vielzahl von zusätzlichen Distributionswegen (mobileTV, WWW, usw.) mit Inhalten zu versorgen. Andererseits stehen leistungsfähige Geräte auf Basis von Standard-IT-Technologien zur Verfügung, die senderseitig zur Produktion von Material eingesetzt werden können und zusätzlich Datenverarbeitung leisten, welche Produktionsabläufe effizienter gestaltet.

Die vorliegende Dissertation beschäftigt sich vor diesem Hintergrund mit der Anwendung von Standard-IT-Technologien im echtzeitkritischen Bereich der Fernsehstudioproduktion. Dabei besteht insbesondere das Ziel der Integration von Metadatenverarbeitung. Die Arbeit kombiniert dazu Standard-IT-Technologien und ergänzt diese um Konzepte, die die besonderen Anforderungen einer Liveproduktion im Fernsehproduktionsstudio berücksichtigen. Dazu zählt das Konzept des *File-Streaming*, das die Anwendung von Dateiformaten (wie MXF) auch im echtzeitkritischen Bereich der Fernsehstudioproduktion ermöglicht. *File-Streaming* impliziert aber eine synchrone Datenübertragung, die neue Fragestellungen insbesondere bei der Datenverteilung und -bearbeitung zur Folge hat.

Im Rahmen dieser Arbeit wird eine Übertragungstechnologie zum Datenaustausch im Studio aus Standardkomponenten modelliert (UDP/IP/Ethernet). Parameter zur Bewertung der Netzwerkleistung und Strategien zur Ressourcenteilung (QoS) werden diskutiert. Im weiteren Verlauf der Arbeit werden Prozessoren zur Verarbeitung von Essenzdaten verglichen und über die PC-Plattform in eine universelle Einheit zur Datenverarbeitung integriert. Die Analyse von Komponenten und Abläufen führt zu einer feingranularen Latenzbetrachtung, die eine Grundlage für Optimierungsstrategien mit dem Ziel einer latenzarmen Implementierung darstellt. Das Ziel der Metadatenintegration wird mit dem Einbinden des Material Exchange Formats (MXF) erreicht, das die synchronisierte Übertragung von Essenz- und Metadaten erlaubt. Die Arbeit identifiziert weiterhin Anwendungsszenarien, in denen Metadaten auch in echtzeitkritischen Live-Produktionen genutzt werden können. Eine prototypische Implementierung bildet abschließend die Grundlage zur Verifikation getroffener Aussagen.

Abstract

The live production of audio and video material in a TV studio requires high quality, low latency and total reliability of signals and systems for content distribution over broadcast channels. In the past these requirements could only be mastered with very special and thus cost-intensive broadcast equipment. With the progress in electronics and media today a multiplicity of additional distribution channels (like mobileTV, WWW, etc.) have to be supplied with content. On the other hand efficient IT based devices are available which can be used for data processing and therefore allowing efficient workflows on production site.

Against this background the thesis deals with the application of standard IT technologies within the real time critical area of TV studio production. With the goal of integrating the prospects of meta data processing into live production workflows the thesis combines and supplements standard IT technologies. In doing so the concept of *file streaming* allows the utilisation of file formats like Material eXchange Format (MXF) within real-time critical live TV production. However *file streaming* implies a synchronous data communication, which causes new issues particular in data handling and processing.

In the context of this work a transmission technology is modelled from standard components (UDP/IP/Ethernet). Parameters for the evaluation of the network performance and strategies for quality of service (QoS) are discussed. Beyond this several hardware media data processors are compared and integrated into a universal, PC based unit for data processing. The analysis of platform components lead to a fine-granular view of latency, which represents a basis for optimization strategies with the goal of a low latency implementation. Meta data integration is achieved by using the Material eXchange Format (MXF) which permits the synchronized transmission of essence and meta data. This thesis further identifies scenarios for the application of meta data in real time critical live TV productions. Finally a prototype implementation provides the basis to verify main findings of the thesis.

Danksagung

An dieser Stelle möchte ich allen danken, die mich auf meinem wissenschaftlichen Weg unterstützt und damit zum Entstehen dieser Arbeit beigetragen haben.

Mein ganz besonderer Dank gilt Professor Dr.-Ing. Hans-Peter Schade, der mich während meiner Zeit als wissenschaftlicher Mitarbeiter im Fachgebiet Audiovisuelle Technik an das Thema heranführte und mit immer währendem Druck die Erstellung dieser Arbeit überhaupt ermöglicht hat. Dank gebührt weiterhin Prof. Dr. rer. nat. habil. Jochen Seitz und Prof. Dr.-Ing. Rolf Hedtke für hilfreiche Anmerkungen und die Bereitschaft, Gutachten zur vorgelegten Arbeit zu erstellen.

Für die Unterstützung bedanke ich mich auch bei den Kollegen am Institut für Medientechnik für kritische Hinweise, Hilfbereitschaft und Unterstützung. Besonderer Dank gilt auch den zahlreichen Studierenden, die diese Dissertation im Rahmen ihrer eigenen Belegarbeiten mit viel Herzblut unterstützten.

Dank möchte ich auch an alle Freunde und Verwandten richten, die nicht müde wurden, mich zur Abgabe der Arbeit zu motivieren und ihren Teil zur formellen Korrektur der Arbeit beigetragen haben.

Überragender Dank gilt meiner Familie: Meinen Eltern für ihren unerschütterlichen Glauben an mich und für ein immer offenes Zuhause; und nicht zuletzt Jonte, Lotta und Anja für die Rücksicht und all die unvergesslichen Momente, die alle Strapazen unvergessen machen.

Inhaltsverzeichnis

1 Einleitung

Die Konvergenz von traditioneller Videotechnik und IT-Technologie auf dem Gebiet der professionellen Fernsehproduktion schreitet seit der Etablierung von nichtlinearen Videoschnittsystemen auf PC-Basis schnell fort. Ein wesentlicher Grund dafür ist die rasante Leistungssteigerung von PCs, die die Verarbeitung von in der Fernsehproduktion typischen Datenraten ermöglicht. Viele speziell für den Broadcastbereich entwickelte Geräte werden zunehmend durch Standard-PC-Komponenten[1] ersetzt (z.B. Videoserver).

Die Durchdringung des Marktes mit leistungsfähiger und gleichzeitig preiswerter Netzwerktechnik (z.B. Gigabit-Ethernet) fördert die Verbindung einzelner Geräte zu einem Netzwerk. Das erleichtert nicht nur den Austausch von Video- und Audiomaterial, sondern gestattet durch die Abkehr von der streng sequentiellen Arbeitsweise insbesondere in der Nachbearbeitung (Postproduktion) neue, effizientere Arbeitsabläufe (Workflows). Die Möglichkeit einer simultanen Produktion für verschiedene Distributionskanäle[2] beschleunigt die Entwicklung eines netzwerkbasierten Ansatzes. Die Nutzung von Metadaten trägt in diesem Bezug entscheidend zur effizienten Produktion audio-visueller Inhalte bei. Die Entwicklung im Bereich der Archivierung hin zu IT-basierten Speichertechnologien unterstützt die durchgängige Anwendung von Metadaten im gesamten Fernsehproduktionsprozess[3].

Voraussetzung für die Kommunikation der verschiedenen Teilsysteme in diesem Netzwerk ist die Verwendung geeigneter Austauschformate, die den Informationsfluss hersteller- und plattformunabhängig unterstützen. Andererseits ist eine zuverlässige und preiswerte Netzwerktechnologie notwendig. Beide Aspekte finden in der vorliegenden Arbeit Berücksichtigung.

Das *Material eXchange Format* (MXF) wurde als Austauschformat mit dem Ziel entwickelt, den interoperablen dateibasierten Datenaustausch zwischen Produktionsequipment zu ermöglichen und damit IT-Inseln zu einem leistungsfähigen Netzwerk zu verbinden. Inzwischen wird MXF auch als Speicherformat im Bereich Beitragserstellung und Nachrichtenproduktion verwendet. Ein Konzept für den Einsatz von MXF im echtzeitkritischen Bereich der Fernsehstudioproduktion existiert hingegen bisher nicht. Die Erarbeitung dieses durchgängigen Konzeptes ist Ziel der vorliegenden Arbeit.

[1] meist angepasste Standard-Hardware – *commodity hardware*
[2] z.B. HDTV-Ausstrahlung und mobileTV
[3] von Preproduktion, über Produktion, Postproduktion, bis zur Ausstrahlung/Archivierung

1.1 Definitionen und Bezeichnungen

Zum einheitlichen Sprachgebrauch innerhalb dieser Arbeit sind folgende Begriffe in Bezug auf die professionelle Fernsehproduktion definiert: Als **Essenz**en werden alle Daten bezeichnet, die in Form von z.B. audiovisuellen Datenströmen den primären Inhalt des Medienproduktes, also einer Fernsehsendung, darstellen. **Metadaten** sind Daten, die andere Daten beschreiben, beinhalten also in Bezug auf die Fernsehproduktion Informationen über die Essenz. Genau genommen unterscheidet man Metadaten von Zusatzdaten (wie z.B. Teletext) dadurch, dass Metadaten ohne die durch sie beschriebenen Daten wertlos sind, also keine Information im klassischen Sinne darstellen. Die Zusammenstellung von Essenzen und Metadaten bezeichnet man als **Content**. Wird Content mit Rechteinformationen ergänzt, spricht man von **Asset**s [Met03, S. 15].

Informationstechnologie (IT) bezeichnet die Technik der Informationserfassung, -übermittlung, -verarbeitung und -speicherung mithilfe von Computern und Telekommunikationseinrichtungen und ist damit ein zentraler Begriff dieser Arbeit. Aufgrund der kompakteren Schreibweise wird der Begriff „Fernsehproduktionsstudio“ im Rahmen dieser Arbeit mit „Studio“ abgekürzt.

Die Anwendung von Netzwerktechnologien ist ein integraler Bestandteil dieser Arbeit. Sender und Empfänger in einem Netzwerk werden in einem generischen Sinne als „Hosts“ bezeichnet, die sich am Netzwerkrand befinden. Im Hinblick auf die hohen Anforderungen an die Leistungsfähigkeit werden Hosts für den Einsatz im Studiokontext als „Workstations“ benannt. Geräte, die im Netzwerkinneren einen Teil des Übertragungskanals darstellen, werden „(Netzwerk)-Knoten“ genannt. Der Zusammenschluss mehrerer (lokaler) Netzwerke zu einem Netz aus Netzen wird als „Internetwork“ bezeichnet.

Im Zusammenhang mit der Übertragung und Bearbeitung von Daten im Studiobereich wird oft von „Echtzeit“ gesprochen. Im Rahmen dieser Arbeit wird der Begriff „Echtzeit“ als Eigenschaft eines Systems verwendet, eine definierte Datenverarbeitung in einer vorher berechenbaren Zeitspanne ohne zusätzliche Verzögerungen durchzuführen (z.B. Abspielen oder Übertragen von Audio und Video in der vorgegebenen Bild- oder Samplefrequenz, nicht schneller oder langsamer).

Viele englische Begriffe in dieser Arbeit wurden direkt in die deutsche Fachsprache übernommen. Dies betrifft insbesondere das Kapitel zu den MXF-Grundlagen (siehe S. 66 ff.). Zur besseren Lesbarkeit werden diese Fachbegriffe *kursiv* dargestellt.

1.2 Einordnung (in den Produktionsprozess)

Für eine Medienproduktion ist folgende Prozesskette üblich: Preproduktion, Produktion, Postproduktion und Distribution. Diese Einteilung gilt für alle Medienbranchen, angefangen von Film und Fernsehen (Television – TV) über Musik, Internet, Print und

Mobilfunk. Die Preproduktion umfasst Planung und Recherche von Content, während in der Produktion die Erstellung und damit die Anpassung von Content auf ein Vermittlungssystem erfolgt. Nachdem der Content während der Postproduktionsphase verfeinert, bearbeitet und getestet wird, erfolgt die Übergabe des Contents an die Zielgruppe durch die Distribution [KK05].

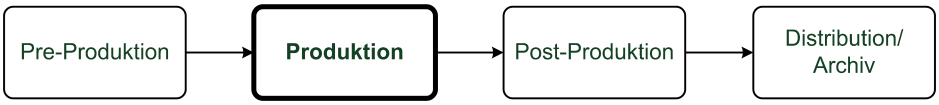


Abbildung 1.1: Medienproduktionsphasen nach [KK05] und Fokus dieser Arbeit

Diese Arbeit konzentriert sich auf die Phase der Produktion und dort vor allem auf den Studiobereich, der besonders für **Live-Produktionen** durch hohe Anforderungen an Qualität, Synchronität und Signalübertragung in Echtzeit gekennzeichnet ist (siehe Abbildung 1.1). Eine Live-Sendung ist eine aufwändige Produktion, in der das aufgenommene audiovisuelle Material komplett oder in Teilen ohne vorherige Speicherung in Echtzeit (abgesehen von Latenzzeiten der Übertragungswege) publiziert wird [Met03, S. 34]. Neben dieser Betriebsart „Live on Air“, bei der die Signale sofort gesendet werden, kann die Sendung auch aufgezeichnet werden („Live on Tape“) [Shm05, S. 709]. Die Anforderungen an Qualität und Synchronität sind in beiden Betriebsarten identisch.

1.3 Ziele und Aufbau der Arbeit

Die vorliegende Arbeit soll nachweisen, dass universelle Standard-Netzwerktechnologien im echtzeitkritischen Bereich der Fernsehstudioproduktion eingesetzt werden können, obwohl sie prinzipbedingt dafür ungeeignet scheinen. Es werden Anforderungen an Hard- und Softwarekomponenten definiert, deren Erfüllung zu einer stabilen Implementierung führt. Der Einsatz von Standard-Netzwerktechnologien erlaubt die Definition einer flexiblen Infrastruktur zur Be- und Verarbeitung von Daten auf Basis von PC-Technologie. Dafür sollen in dieser Arbeit Anwendungsszenarien beschrieben und diskutiert werden.

Drei wesentliche Aspekte sind bei Konzeption und Implementierung einer neuartigen Studioinfrastruktur zu beachten: Datenübertragung über Netzwerke, Datenbearbeitung auf PC-Plattformen und die Anwendung von Metadaten im linearen Produktionsprozess. Im Anschluss an das zweite Kapitel, das den Stand der Technik aufzeigt, wird jeder Aspekt in einem Kapitel diskutiert und Grundlagen des Aspektes um die neuen Erkenntnisse im Bereich der Fernsehstudioproduktion erweitert. Im sechsten Kapitel werden diese Aspekte in einem Gesamtkonzept zusammengefasst und dessen Funktion mittels einer prototypischen Implementierung nachgewiesen.

2 Stand der Technik

Die Entwicklung eines neuartigen Systems bedingt die genaue Analyse bereits existierender Lösungsansätze. Deshalb werden zunächst aktuelle Studioinfrastrukturen mit dem Ziel analysiert, Grenzen des aktuellen Systems aufzuzeigen und Anforderungen für zukünftige Systeme zu definieren.

Im Abschnitt 2.2 werden Ansätze aus themennahen Forschungsarbeiten vorgestellt, um einen Vergleich zu ermöglichen. Alternative Konzepte aus branchenübergreifenden Anwendungsfeldern sollen zeigen, dass die zu lösenden Fragestellungen nicht nur in der Medienproduktion bzw. im Fernsehproduktionsstudio aktuell und relevant sind.

2.1 Struktur eines konventionellen Fernsehproduktionsstudios

Für die Produktion von Live-Sendungen werden Fernsehproduktionsstudios benutzt, mit denen elektronische Bild- und Tonsignale produziert und nahezu verzögerungsfrei an den Zuschauer übertragen werden. Die gleichzeitige Nutzung mehrerer Kamerazüge ermöglicht dabei die Produktion von Bildsignalen aus verschiedenen Perspektiven, die sich zu einem abwechslungsreichen Programm mischen lassen. Innerhalb dieser Studios wurden zunächst analoge (Version 1), später digitale Essenzsignale (Version 2) übertragen und bearbeitet [Sha08]. Die Version 3 beschreibt den Austausch von Daten über generische Netzwerke und stellt ein Teilziel dieser Arbeit dar.

2.1.1 Signalübertragung

Der Signalaustausch zwischen verschiedenen Geräten zur Aufnahme, Bearbeitung und Speicherung von Audio- und Videosignalen erfolgt in einem konventionellen digitalen Studio auf Basis unkomprimierter digitaler Signale. Die Signalübertragung erfolgt isochron, d.h. Informationseinheiten fester Größe werden in regelmäßigen Abständen gesendet. Zeitliche Abweichungen existieren idealerweise nicht; praktisch werden sie minimiert. Aufgrund der vorhandenen Leitungslängen können Übertragungsfehler infolge von Leitungsdämpfung vernachlässigt werden.

Für jede Signalart (Audio, Video, Steuerung, Synchronisation usw.) existiert ein dediziertes Netzwerk aus Signalleitungen und zentralen Kreuzungspunkten (Kreuzschienen,

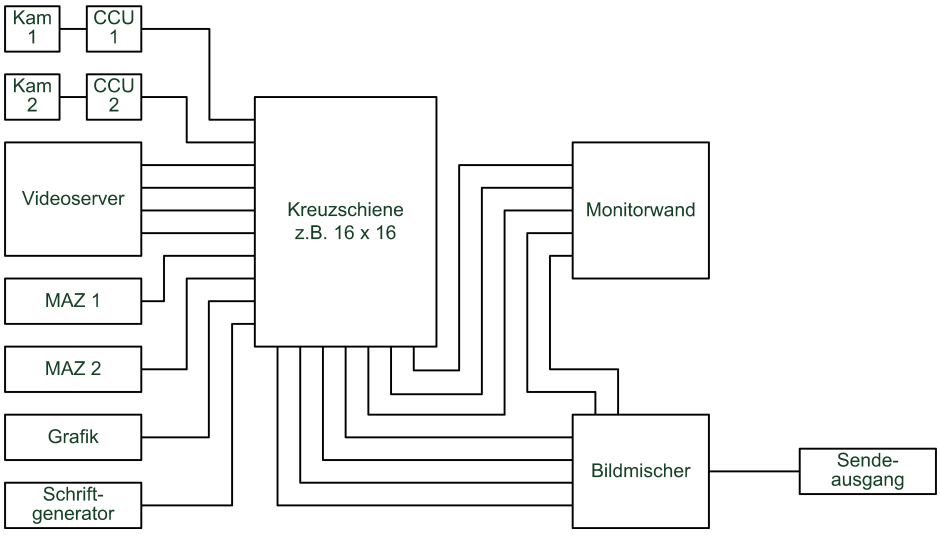


Abbildung 2.1: Vereinfachte Videoverkabelung in einem Produktionsstudio

Steckfeldern). Dieser Umstand macht die Verkabelung der Komponenten aufwändig. Gleichzeitig wird aber eine gegenseitige Signalbeeinflussung während der Übertragung minimiert und die logische Komplexität bleibt gering (ein Kabel = ein Signal). Abbildung 2.1 stellt ein vereinfachtes Videonetzwerk innerhalb eines Studios dar.

Videoschnittstellen

Studiokameras geben gammavorverzerrte Farbwertsignale (E'r, E'g, E'b, in der Regel als RGB-Signale bezeichnet) aus, die meist in analoger Signalform entweder parallel über Multicore-Kabel oder seriell über Triaxkabel vom Kamerakopf im Aufnahmesaal zu den *Camera Control Units* (CCU) übertragen werden. Die Signalverteilung im Studio ausgehend von den CCUs erfolgte zunächst in Form des analogen Composite-Signals (FBAS - s.u.) und wurde später von der Komponententechnik abgelöst. Eine zweite wesentliche Entwicklung war der Übergang von Analog- zu Digitaltechnik [Hed95]. Im Folgenden werden die wesentlichen Signalverarbeitungsschritte in einer CCU kurz beschrieben.

In der CCU werden aus den RGB-Signalen ein Leuchtdichtesignal Y und zwei Farbdifferenzsignale (R - Y) und (B - Y) gewonnen (Matrizierung). Nach Pegel- und Bandbreitenreduktion der Farbdifferenzkomponenten stehen Y, P_B und P_R als so genannte analoge Komponenten am Ausgang der CCU zur Verfügung. Die Verteilung innerhalb des analogen Studios erfolgt auf drei separaten Leitungen (75-Ohm-Koaxialkabel).

Zur Erzeugung eins PAL-codierten Sendesignals erfolgt eine Pegel- und Bandbreitenreduktion mit anderen Faktoren. Aus den Farbdifferenzsignalen (R - Y) und (B - Y) werden

die Signale U und V generiert, die entsprechend der vorherrschenden Videonorm von einem Chrominanzsignal C zusammengefasst werden. Für das in Deutschland verwendete PAL-Verfahren kommt die Quadraturamplitudenmodulation (QAM) zum Einsatz, welche die zwei Signale auf nur einer Trägerfrequenz vereint. Gemeinsam mit Austast- und Synchronsignalen werden Y und C zu einem *Composite*-Signal (auch FBAS - Farb-, Bild-, Austast- und Synchronsignal) zusammengeführt, das mit einer Bandbreite von 5 MHz lediglich ein 75-Ohm-Koaxialkabel zur Übertragung benötigt. Diese analoge Schnittstelle wird trotz eingeschränkter Signalqualität auch im digitalen Fernsehstudio eingesetzt, um eine große Anzahl von Videosignalquellen auf entsprechenden Vorschaumonitoren, die üblicher Weise in Form einer Monitorwand angeordnet sind, anzuzeigen.

Auf Basis des analogen Komponentensignals YP_BP_R nach EBU N-10 entstand eine digitale Repräsentation, in dem die Signale nach der ITU-R BT.601 abgetastet und quantisiert werden. Als Abtastfrequenz wurde für das Luminanzsignal ein Wert von 13,5 MHz definiert, aus dem sowohl für 525- (z.B. Nordamerika) als auch für 625-Zeilensysteme (Europa) eine ganzzahlige Anzahl von Abtastwerten pro Zeile resultiert. Für die Farbdifferenzkomponenten kommt die Hälfte diesen Wertes zum Einsatz (6,75 MHz). Dieses Verhältnis wird mithilfe des Abtastschemas 4:2:2 ausgedrückt. Im 625-Zeilen-System (Deutschland) setzt sich ein aktives Bild aus 720 × 576 Pixeln zusammen. [I601]

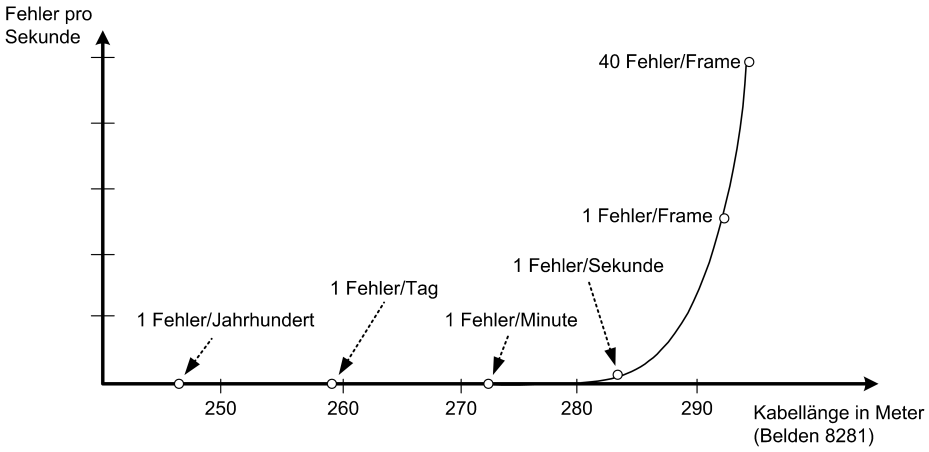


Abbildung 2.2: Fehlerwahrscheinlichkeit durch Leitungsdämpfung bei SDI [Shm05, S. 122]

Die Schnittstelle zur seriellen Übertragung des digitalen Komponentensignals wurde nach ITU-R BT.656 standardisiert und ist unter dem Namen SDI (*Serial Digital Interface*) bekannt geworden [I656]. Bei SDI sind Datenworte mit 8 oder 10 Bit zulässig, die Datenrate beträgt maximal 270 Mbit/s. Physikalisch ist ein Standard-Koaxialleiter mit 75 Ohm Wellenwiderstand vorgesehen, so dass bestehende Verkabelung weiter genutzt werden kann. Die SNRZI[1]-Kanalkodierung sorgt für ein annähernd gleichspannungsfreies

[1] *Scrambled Non Return to Zero Inverted*

und selbsttaktendes Signal. Abhängig von der Kabelqualität können Entfernungen von bis zu 250 m überbrückt werden, ohne dass Signalfehler durch Kabeldämpfung auftreten (siehe Abbildung 2.2). Ein Fehlerschutz ist nicht vorgesehen. Einzelne Bitfehler würden sich aufgrund der örtlichen und zeitlichen Unabhängigkeit der Daten auch nur auf einzelne Bildpunkte auswirken.

Neben dem aktiven Bildinhalt werden über die SDI-Schnittstelle auch die Horizontal- und Vertikalaustastlücken übertragen[2], die im analogen Signal zur Gewährleistung einer sicheren Synchronisation benötigt werden. Für digitale Signale werden die Austastlücken nicht benötigt und stehen somit zur Übertragung von Zusatzinformationen (*Auxiliary Data*) zur Verfügung. So können bis zu vier nach AES/EBU kodierte Audiokanäle in ein SDI-Signal integriert werden (Embedded Audio). Embedded Audio wird insbesondere bei der Weitverkehrsübertragung (z.B. zwischen Studios) eingesetzt. Innerhalb einer Studioumgebung werden Audio- und Videosignale getrennt voneinander übertragen, da Aufnahme, Ausgabe und insbesondere Bearbeitung jeweils auf unterschiedlichen Geräten für Audio und Video geschehen.

Mit der verstärkten Anwendung von Kompressionsverfahren im Bereich digitaler Videosignale entstand der Bedarf nach einer Schnittstelle, die den verlustfreien Transport von komprimierten Videosignalen ermöglicht. So entwickelten sich zunächst zwei inkompatible Ansätze (*serial digital data interface* – SDDI und *compressed serial data interface* – CSDI), deren Vorteile in einem SMPTE-Standard zum *serial data transport interface* (SDTI) zusammengeführt wurden (SMPTE 305M). SDTI ist ein unidirektionales Transportprotokoll für das Transportmedium SDI gemäß ITU-R BT.601/656 und kann somit vorhandene Infrastrukturen nutzen. Es erlaubt die Übertragung von komprimierten Videosignalen in „realtime“ oder auch „faster than realtime“. [Hof99] SDTI konnte seinem ursprünglichen Ansatz, der Erweiterung der SDI-Studioinfrastruktur nicht gerecht werden und befindet sich heute nur in dedizierten MAZ-zu-MAZ- und MAZ-zu-Server-Anwendungen im Einsatz [HK04b].

Auch für die unkomprimierte Übertragung von hochauflösenden Videosignalen wurden Schnittstellen standardisiert, die sich im Wesentlichen durch die erforderliche Datenrate und die Anzahl der benötigten Leitungen bzw. Kabel unterscheiden: „HD-SDI“ (1,5 Gbit/s, SMPTE 292M), „Dual-HD-SDI“ (3 Gbit/s, SMPTE 372M) und „3G“ (3 Gbit/s, SMPTE 424M)[3]. Des Weiteren kann Kompression genutzt werden, um 1080p-Signale über HD-SDI oder 1080i/720p-Signale über SDI zu übertragen (DiracPro)[BBC07].

Audioschnittstellen

Auch die Signalverteilung der Audiosignale erfolgt in modernen professionellen Studios hauptsächlich auf Basis der digitalen Signalübertragung. Die Signale werden vergleichbar

[2] getrennt durch *Start of Active Video* (SAV) und *End of Active Video* (EAV) Bitmuster
[3] Verzeichnis von SMPTE-Standards siehe Seite 121

mit den Videosignalen mithilfe von zentral bedienbaren Kreuzschienen oder Steckfeldern im Studio verteilt. Durchgesetzt hat sich dabei die AES/EBU[4]-Schnittstelle. Der Standard beinhaltet das Datenformat inkl. Kanalcode- und elektrischer Spezifikation für die Übertragung von zwei Audio-Kanälen über ein abgeschirmtes verdrilltes Leitungspaar bis 100 m Entfernung zwischen Sender und Empfänger. [Fin92]

Der linke und rechte Kanal eines Stereosignals werden im Multiplexverfahren übertragen. Durch die Verwendung des *Biphase Mark* Kanalcodes ist die Schnittstelle selbsttaktend. Die Abtastfrequenz des zu übertragenden Signals ist nicht festgelegt; die Frequenz 48 kHz wird insbesondere in Studioumgebungen oft verwendet. Ein Subframe beinhaltet die quantisierten Abtastwerte eines Audiokanals. Zwei Subframes werden zu einem Frame zusammengefasst und mit einer Frequenz, die der Abtastfrequenz des Audiosignals entspricht, übertragen. 192 Frames werden zu einem Block zusammengefasst. Für 48-kHz-Signale wird so alle 4 ms ein Block gesendet, was einer Frequenz von 250 Hz und einer Datenrate von $3,072 \cdot 10^6$ bit/s entspricht. [Poh00, S. 442 ff.]

Neben der Übertragung von hochqualitativen Audiosignalen als Bestandteil einer audiovisuellen Produktion müssen insbesondere in größeren Studios viele Beteiligte koordiniert werden. Dazu werden Gegensprechanlagen (auch: Intercom) verwendet. Zentraler Bestandteil eines Intercom-Systems ist eine Matrix, die ein- und ausgehende Signale miteinander verbindet. Im Gegensatz zu A/V-Kreuzschienen ermöglicht eine Intercom-Matrix auch eine Summierung verschiedener Eingänge auf einen Ausgang. Damit werden neben Punkt-zu-Punkt-Verbindungen auch die Betriebsarten Punkt-zu-Multipunkt und Multipunkt-zu-Multipunkt möglich. In großen Installationen können mehrere Matrizen miteinander verbunden werden. [Wai05]

Prinzipiell sind die Anforderungen an die Audiosignalqualität von Intercom-Systemen im Rahmen einer guten Sprachverständlichkeit relativ gering. Nachdem systemintern bekannte Schnittstellen der Essenz-Signalübertragung (z.B. AES/EBU) genutzt wurden, kommen zunehmend Voice-over-IP-Systeme zum Einsatz, die auch komprimierte Audiodaten übertragen.

2.1.2 Signalbearbeitung

Geräte der professionellen Videotechnik sind Konstruktionen für einen Markt mit kleinen Umsatzzahlen (im Vergleich zu PC-Technologie) und deshalb relativ kostenintensiv. Dazu zählen insbesondere Bildmischer, Audio- und Videokreuzschienen, Chromakeyer usw., deren Funktionalität in Hardware implementiert wird. In diesen Geräten kommen spezielle Prozessoren zum Einsatz, die Anforderungen an eine latenzarme und hochqualitative Signalverarbeitung erfüllen. In Abhängigkeit der speziellen Funktion werden dazu Digitale Signalprozessoren (*digital signal processor* – DSP) und anwendungsspezifische

[4] korrekt: AES3 - ein Standard der Audio Engineering Society, der nahezu identisch auch unter EBU Tech. 3250-E publiziert wurde

integrierte Schaltungen (*application specific integrated circuit* – ASIC) verwendet, die auf separaten Platinen modular in Elektronik-Boxen zusammen geschaltet werden. Ein handelsüblicher SDI-Videomischer erreicht so eine Eingangs- zu Ausgangslatenz von wenigen 10 μs (kleiner als eine Videozeile) [Tho02]. Zunehmend kommen in den Geräten auch vor Ort modifizierbare Logikbausteine (*field programmable gate array* – FPGA) zum Einsatz, was eine Funktionserweiterung über den Austausch von Konfigurationsdaten in Speicherzellen ermöglicht. Die Signaleingabe und -ausgabe erfolgt jeweils über o.g. audio/videospezifische Schnittstellen.

Bildmischer Die Umschaltung bzw. Mischung von Videosignalen wird durch Bildmischer (*video switcher*) umgesetzt. Für die fehlerfreie Mischung ist die Synchronität der angeschlossenen Signalquellen notwendig. Die Synchronität wird durch den zentralen Studiotakt gewährleistet. Die Bedienelemente eines Mischpultes sind ergonomisch angeordnet. Im Mischpult integrierte Kreuzschienen erlauben die Auswahl von Eingangssignalen, die über kaskadierbare Mix/Effekt(ME)-Stufen umgeschalten bzw. gemischt werden. Eine Umschaltung kann als Hartschnitt (*cut*) ausgeführt werden, d.h. der Wechsel zwischen den Signalen erfolgt störungsfrei in der vertikalen Austastlücke des Bildsignals. Die Umschaltung kann aber auch nach einer bestimmten Anzahl von Zeilen oder eines bestimmten Prozentsatzes der aktiven Zeilendauer vorgenommen werden (*wipe*). Die als *Mix* bezeichnete Bildmischung ist kein Schaltvorgang, hier sind beide Bildsignale ganzflächig mit einer bestimmten Intensität sichtbar.

Keyer Auch das Stanzverfahren (Keying) ist ein Schaltvorgang, bei dem das Schaltsignal aus dem Bildsignal abgeleitet wird. Beim Stanzverfahren muss sich der auszustanzende Bereich im Vordergrundsignal deutlich vom Umfeld abheben, damit daraus eindeutig das Schaltsignal gewonnen werden kann. Es werden große Helligkeitsunterschiede (*Luminance Key*) oder deutliche Farbunterschiede (*Chroma Key*) ausgenutzt. Beide Stanzarten und weitere Key-Verfahren[5] sind in vielen Bildmischern integriert. Um den hohen qualitativen Ansprüchen einer Blauwand- oder einer Virtual-Set-Produktion gerecht zu werden, wird oft ein *Chroma-Keyer* als separates Gerät verwendet. Mit patentierten und z.T. rechenintensiven Verfahren werden damit Artefakte wie Farbsäume (*blue spill*) vermieden und realitätsnahe Stanzmasken auch unter schwierigen Bedingungen (Glasflächen, Schatten, feine Strukturen z.B. Haare) erzeugt.

Eine zumindest teilweise Abkehr der aufwändigen und teuren Implementierung von Funktionalitäten in dedizierter Spezial-Hardware stellen Systeme dar, die im Wesentlichen als Signalquellen oder -senken fungieren.

[5] *Linear Key* für teiltransparente Flächen (Bauchbinden), *Extern Key* für die Nuttzung computergenerierten Stanzsignale (Logos), *Preset Pattern Key* für die Nutzung von Wipe-Mustern

Videoserver Magnetbandaufzeichnungssysteme (MAZ) waren lange Zeit die einzig wirtschaftliche Möglichkeit, Video- und Audiosignale im Studiobereich z.B. für Livesendungen einzuspielen (vorproduzierte Beiträge) bzw. aufzuzeichnen (Mitschnitt). Zunehmend werden diese Aufgaben von Videoservern übernommen, die technisch gesehen meist um Audio- und Videoschnittstellen erweiterte Standard-PC-Systeme mit entsprechender Speicherkapazität in Form von z.B. über Fibre Channel angebundenen Festplatten-Arrays (RAID: *redundant array of independent disks*). Videoserver sind speziell auf zeitkritische Audio- und Videodaten ausgerichtete Speicher- und Abspielsysteme, die gleichzeitig auf mehreren Kanälen aufnehmen und Videodateien auf anderen Kanälen abspielen können. Dabei kommen meist DV- oder MPEG-Kompressionsformate zum Einsatz. Der große Vorteil von Server-Systemen für die Audio- und Videoaufzeichnung gegenüber allen magnetbandgestützten Systemen ist der wahlfreie und schnelle Zugriff auf sämtliche Daten. Weiterhin kann die Nachbearbeitung des Materials in der Postproduktion direkt am auf dem Videoserver gespeicherten Material mittels nonlinearem Videoschnitt durchgeführt werden, die zeitaufwändige Nachbearbeitung auf Basis von Magnetbändern wird so umgangen. Der Einsatz von Videoservern ist die Voraussetzung für eine bandlose und damit effektive (Post)Produktion.

Schriftgenerator Schriftgeneratoren ermöglichen die Integration von computergenerierten Schriften und Zeichen in TV-Bilder. Mithilfe einer geeigneten Software werden verschiedene Schriftzüge und Hintergründe an einem PC erstellt. Eine integrierte Videokarte generiert daraus ein Video- und ein Keysignal, auf deren Basis ein Bildmischer das Einfügen in ein Videobild übernimmt.

Teleprompter Elektronisch vorhandene Sprechertexte können mithilfe von Teleprompter von einem Sprecher abgelesen werden, während er direkt in die Kamera schaut. Ein Teleprompter ist ein Kameravorsatzgerät bestehend aus einem Monitor und einem halbdurchlässigen Spiegel. Der Spiegel ist so angebracht, dass der Moderator den auf dem Monitor angezeigten Text lesen kann. Das Bildsignal für den Monitor wird von einem PC eingespielt, über dessen Software die Geschwindigkeit der Texte geregelt wird.

Bedienteile (Pulte, Tastaturen, Monitore usw.) der Signalverarbeitungsgeräte sind in der Videoregie untergebracht. Die abgesetzten Prozessoreinheiten befinden sich hingegen in 19-Zoll-Schränken in einem klimatisierten Technikraum. Die Geräusche, die im Wesentlichen durch Lüfter zur Kühlung der Hardware entstehen, werden somit abgeschirmt.

2.1.3 Synchronisation und Steuerung

Die Audio- und Videosignale in einem Studio werden unabhängig voneinander mit unterschiedlichen Frequenzen übertragen, können aber nicht unabhängig voneinander bearbeitet und ausgegeben werden. In zweifacher Hinsicht sind die Signale zu synchronisie-

ren: Erstens (1) muss bei einem Misch- bzw. Umschaltvorgang innerhalb einer Signalart Synchronität vorliegen, um Störungen zu vermeiden. Beim Umschaltvorgang von Videosignalen muss so gewährleistet sein, dass Bildsignal A und B synchron anliegen, um einen störungsfreien Schnitt in der Austastlücke zu ermöglichen. Zweitens (2) muss die Synchronität zwischen verschiedenen Signalarten gesichert werden. Ein bekanntes Beispiel dafür ist die Lippensynchronität[6], die besonders dann zu beachten ist, wenn eine Signalart (z.B. Video) eine Verzögerung durch einen Bearbeitungsvorgang erfährt (z.B. im virtuellen Studio: Verzögerung durch Berechnung des virtuellen Bildhintergrundes).

Das Problem (1) wird gelöst, indem jedes Gerät durch Anbindung an einen zentralen Taktgenerator (Studiotakt: Blackburst oder Tri-Level-Sync) im gleichen Takt betrieben wird, d.h. dass z.B. jedes Videogerät die Bildinformation im gleichen Frametakt ausgibt. Zusätzlich besteht an professionellen Geräten die Möglichkeit des Phasenabgleichs zwischen Videosignal und Studiotakt, weil die Verbindungsleitungen zum Taktgenerator meist ungleich lang sind und auch nicht alle Geräte die gleiche Signalverarbeitung aufweisen [Shm05, S. 663]. Die Problematik (2) wird durch Verzögerung (*Delay*) des jeweils nicht bearbeiteten Signals umgangen.

Insbesondere zur exakten Referenzierung von aufgezeichneten Videosignalen ist ein Zeitstempel erforderlich. *Timecode* stellt einen solchen Mechanismus zur Verfügung, in dem die einzelnen Bilder eindeutig mit Zeitwerten versehen werden. Die acht Ziffern des Timecodes stellen die Zeitwerte in Stunden, Minuten, Sekunden und Bildern (Frames) dar. Die Timecodearten *Longitudinal Timecode* (LTC) und *Vertical Interval Timecode* (VITC) werden für den MAZ-Schnittbetrieb notwendig und werden auch von Videoserver-ßystemen mit aufgezeichnet. Im Studio stellt der zentrale Taktgenerator auch Timcodewerte zur Verfügung.

Aus der Notwendigkeit, mehrere Magnetbandaufzeichnungsgeräte (MAZ) eines linearen Schnittsystems steuern zu können, entwickelten sich einfache Steuerschnittstellen mit begrenztem Befehlsumfang. Nachdem zunächst Ein/Aus-Befehle über GPI (*General Purpose Interface*) für einfache Start/Stop-Befehle genutzt wurden, konnten über RS-422 komplexere Befehle gesendet werden (*Fast Forward, Eject, Videoinsert* usw.). Im Studiobereich hat sich in Verbindung mit 9-poligen Sub D-Steckern die so genannte Sony 9 Pin-Schnittstelle als Quasistandard durchgesetzt, die neben den Fernsteuerdaten auch Timecode-Daten überträgt. [Shm05, S. 599 ff.] Das VDCP *(Video Device Communications Protocol)* erweitert die Sony 9 Pin-Schnittstelle um die Inhaltsabfrage eines Videoservers [Gen02].

[6] *A/V-Lip-Sync*, beschreibt die Gleichzeitigkeit der Wahrnehmung von Audio- und Videorepräsentation eines Ereignisses

2.1.4 Zusammenfassung: Grenzen der konventionellen Studioinfrastruktur

In Studios, die nach dem vorgestellten Ansatz arbeiten, können zuverlässig konventionelle Broadcast-Produktionen realisiert werden. Dennoch bringt dieser Ansatz systembedingt eine Reihe von Einschränkungen mit sich, die auch durch Weiterentwicklungen nicht oder nur sehr schwer erfüllbar sind. Nachfolgend sollen die **Grenzen** des konventionellen Ansatzes zusammengefasst werden.

1. Die Verarbeitung von Signalen erfolgt größtenteils auf dedizierten Geräten, die für wenige spezielle Aufgaben konzipiert wurden. Dies wirkt sich einerseits auf die Kosten der Geräte aus (Nischenmarkt Broadcast), andererseits sind die Möglichkeiten bezüglich der Automatisierung von Abläufen aufgrund geringer Flexibilität und der Vielzahl zu berücksichtigender Schnittstellen sehr eingeschränkt.
2. Die Verteilung der Signale erfolgt mit unterschiedlichen Schnittstellen für Audio-, Video-, Daten-, Synchronisations- und Steuerdaten. Dies macht die physikalische Verkabelung und die Synchronisation zwischen den Signalen aufwändig. Außerdem besteht die Notwendigkeit der Anpassung von Signalschnittstellen, sobald neue Essenzformate zum Einsatz kommen sollen.[7]
3. Die durchgängige Nutzung von Metadaten ist aufgrund fehlender Mechanismen und Schnittstellen schwer bzw. nicht möglich.

Alternative Ansätze sollten diese Unzulänglichkeiten kompensieren, müssen aber darüber hinaus folgende grundlegende **Anforderungen** an ein System zur professionellen Produktion von Live-Fernsehsendungen erfüllen.

1. Produktion von hochqualitativen Essenzdaten, d.h. gute Signalqualität auch nach mehreren Kopier- bzw. Bearbeitungsgenerationen
2. Synchronisation von Datenströmen sowohl für störungsfreie Signalmischung (z.B. Video-Mix), als auch gleichzeitige Repräsentation von Ereignissen durch verschiedene Essenzsignale (*Lip-Sync, Audio-Video-Delay)*
3. Punkt-zu-Multipunkt-Übertragungen innerhalb des Systems, z.B. für Signalvorschau-Möglichkeiten
4. latenzarme Datenverarbeitung, z.B. für Interviewsituationen
5. zuverlässige Funktionalität (24/7[8]), Havariesicherheit

[7] Durch die Entwicklung neuer Essenzformate können z.B. effizientere Kompressionsalgorithmen auch in der Fernsehproduktion Anwendung finden.
[8] 24 Stunden pro Tag, 7 Tage pro Woche

2.2 Related Work - Verwandte Arbeiten

Im Folgenden werden Bestrebungen zur Dezimierung genannter Einschränkungen des konventionellen Ansatzes einer Studioproduktion vorgestellt. Dabei werden Ergebnisse und Erkenntnisse zusammengestellt und um Defizite, Unvollständigkeiten bzw. Widersprüche ergänzt. Alle Bestrebungen haben eines gemeinsam: Sie lösen sich von der broadcasttypischen signalzentrischen Herangehensweise und orientieren sich an Modellen aus der Informationstechnik.

Der Forschungsschwerpunkt im Produktionsbereich der letzten Jahre lag zunächst auf Ansätzen und Systemen zur dateibasierten und damit speicherzentrierten Produktion von medialen Inhalten. Erst später wurde versucht, IT-Infrastrukturen auch in Live-Produktionsszenarien einzusetzen.

Wesentliche Basis der Entwicklungen von linearen zu nichtlinearen Abläufen in der Fernsehprogramm-Produktion schuf die **EBU/SMPTE Task Force** *„Harmonized Standards for the Exchange of Programme Material as Bitstreams“*[9]. Dazu wurden insbesondere in den Arbeitsgruppen *Systems*, *Wrappers & Metadata* und *Networks & Transfer Protocols* Anforderungen gesammelt, State-of-the-Art-Technologien verglichen und grundlegende Zusammenhänge mithilfe von Systemmodellen und Referenzarchitekturen modelliert [EBU98].

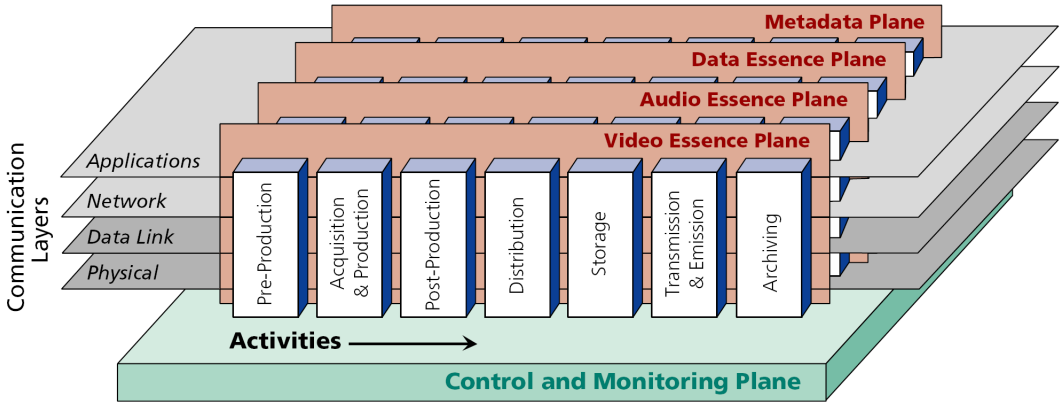


Abbildung 2.3: EBU/SMPTE Task Force System Modell [EBU98]

Das wichtigste Ergebnis der Arbeit dieser Task Force ist ein Systemmodell, das broadcasttypische Abläufe mit einem IT-typischen Schichtenmodell verbindet (siehe Abbildung 2.3). Die zentrale Rolle von Metadaten in zukünftigen Produktionsstrukturen wird darin deutlich und deren Handhabung bzw. Strukturierung in zentralen *Registries* (Wörterbüchern) definiert. Die Notwendigkeit einer eindeutigen Materialkennzeichnung (z.B. durch

[9] Anfang 1997 bis Mitte 1998

UMID[10]) wurde herausgearbeitet, sowie Anforderungen an Wrapperformate zur simultanen Verarbeitung von Content- und Metadaten zusammengetragen. Damit wurde die Grundlage der MXF-Standardisierung gelegt. Hinsichtlich des Austausches von Content über paketorientierte Schnittstellen entstand eine Referenzarchitektur, wobei Ethernet nicht als geeignete Technologie für den echtzeitkritischen Bereich bestimmt wurde.

Filebasierte Produktion

Bereits 1998 realisierte DELIYSKI ein bandloses Produktionssystem für News-Anwendungen auf Basis eines FibreChannel(FC)-Netzwerkes und Standard-PC-Hardware (Pentium PII mit Targa2000-Hardwareunterstützung). Mit dieser serverzentrierten Lösung konnten Audio- und Video-Streams ein- und ausgespielt werden. Damit war systemintern eine dateibasierte Verarbeitung möglich. [DB98]

Mit dem Ziel, die Kosten von Fernsehproduktionen in Europa zu reduzieren, startete im Jahre 2000 das IST[11]-Projekt G-FORS[12]. Auf Basis des MXF-Formates konnte des Projekt eine filebasierte Handhabung von Content in der Broadcast-Produktionskette demonstrieren. Ein Kamerasignal wurde dabei DV-kodiert in eine MXF-Datei geschrieben, die dann über Netzwerke für einen einfachen Videoschnitt zu Verfügung stand [WD02].

Auch die British Broadcasting Corporation (BBC) erforscht Methoden zur Kostenreduzierung in der Postproduktion und Distribution. Die Anwendung von Hardware-, Netzwerk- und Softwaretechnologie aus dem IT-Bereich stand im Zentrum der Bestrebungen von vielen Projekten, z.B. ATLANTIC[13] und ORBIT[14]. Insbesondere die Nutzung der MPEG2-Videokompression erlaubte dabei Datenraten, die Ende der 1990er Jahre nur mit ATM[15]-Technologie zu bewältigen waren [Bri99, BT00]. Die Mole[16]-Technologie hilft dabei, die Qualitätsverluste durch Rekompressionsdurchläufe bei der Verarbeitung von long-GOP-MPEG2 einzugrenzen [WG98].

Mit der Definition von *intimate metadata* wurden im IST MetaVision[17]-Projekt Möglichkeiten evaluiert, die Bildqualität unter Zuhilfenahme von Zusatzinformationen im Rahmen der Postproduktion zu verbessern bzw. zu stabilisieren. Dazu gehören z.B. die Bewegungsinformation innerhalb von Sequenzen, die mit höherer temporaler Auflösung auf-

[10] Unique Material Identifier, siehe SMPTE 330M
[11] Information Society Technologies, Teil des sechsten europäischen Rahmenprogramms zur Forschungsförderung 2002-2006
[12] Generic Formatting for Storage, Projektlaufzeit: zwei Jahre ab Januar 2000
[13] Advanced Television at Low bitrates And Networked Transmision over Integrated Communication systems
[14] Object Reconfigurable Broadcast Infrastructure Trail
[15] Asynchronous Transfer Mode ist eine Technik, die zur Datenübertragung kleine Zellen mit fester Länge benutzt und damit verbindliche Garantien bezüglich Durchsatz, Latenz usw. geben kann, vergleiche Abschnitt 3.3.1.1
[16] „Mole“ ist ein eingetragenes Trade Mark eines ATLANTIC-Partners
[17] IST-gefördertes Projekt, Laufzeit: Oktober 2000 - September 2003

genommen werden, oder Tiefenkarten für 3D-Produktionen. Weiterhin zählen darunter Bearbeitungsinformationen (EDL[18], Effekte usw.) und die Beschreibung eines Ausspielformates (z.b. örtliche und zeitliche Auflösung). [WTK+02] Die Idee des *nondestructive Editing* wurde von anderen Projekten aufgegriffen. Dabei kommen Beschreibungssprachen wie z.B. SMIL[19] zum Einsatz, die Referenzierung erfolgt meist auf Basis des UMID und häufig wird MXF als Wrapper-Format eingesetzt [TKY+05].

Ziel des Projektes ASSET[20] war die Spezifizierung und Entwicklung einer Middleware zur Integration von Produkten und Komponenten verschiedener Hersteller zu einer filebasierten Fernsehproduktionsplattform. Dazu wurden Software-Tools und Application Programming Interfaces (API) entwickelt und zu einem Plattform- und Programmiersprachenunabhängigem Framework zusammengefasst. Für den Austausch von Essenzen zwischen verschiedenen ASSET-Frameworks ist MXF vorgesehen, die intern verwendeten Nachrichten werden im XML-Format ausgetauscht [BC05, CVR+04].

Live-Produktion auf ATM-Netzen

Erst mit gestiegener Leistungsfähigkeit der Rechentechnik wurde auch der Aspekt *Streaming* über paketbasierte Netzwerke in großem Umfang aufgegriffen. Mitte der 1990er Jahre entstanden Konzepte, die – zunächst beschränkt auf die Anwendung in lokalen Netzwerken – die Signalverteilung von Audio- und Videosignalen über datenbasierte Netzwerke propagierten. Dazu wurden anfangs isochrone und mit konventionellen Broadcastschnittstellen kompatible Übertragungsverfahren (SDDI, CDSI, SDTI) verwendet, später zunehmend auch IT-Netzwerktechnologien wie ATM.

Stone und David stellten 1996 eine Architektur vor, die SDDI für die Echtzeitübertragung von Audio und Video, RS-422 für Echtzeit-Steuerinformationen und Ethernet für Offline-Datentransfer verwendete. Aus Effizienzgründen entschied man sich für eine Kompression der Videosignale in der Produktion. Visionär propagierten Stone und David bereits den *big pipe approach*, der ein generisches Netzwerk zur Verteilung aller Signale im Studio vorsieht. [SD96]

Das Projekt *High Definition Television Broadcast Technology*[21] griff die Problematik der komplexen Studioverkabelung auf entwickelte eine einfache Übertragungslösung auf Basis von ATM. Ein zentraler ATM-Router ermöglicht dabei die Verteilung von Essenz- und Steuerdaten. Auf Basis des *Digital Studio Command and Control* (DSCC) Protokolls [WWH98] wurde weiterhin eine Studioautomation implementiert, die wiederum auf

[18] *Edit Decision List*: eine Liste von In- und Out-Punkten für den Videoschnitt auf Basis von Timecode-Werten

[19] Synchronized Multimedia Integration Laguage

[20] Architectural Solution for Services Enhancing digital Television - ebenfalls ein IST-gefördertes Projekt, Laufzeit: Juli 2002 - Juni 2003

[21] Advanced Technology Program (ATP), 1995–2000, unterstützt vom National Institute of Standards and Technology (NIST)

CORBA (*Common Object Request Broker Architecture*) zurückgreift. Jedes Gerät im Studio wird dabei als Softwareobjekt repräsentiert, dessen Ressourcen über einen *Object Resource Broker* (ORB) angeboten werden. [Mar01]

Das Europäische ACTS[22]-Projekt „Distributed Video Production (DVP)“[23] brachte Pilotapplikationen professioneller Videoproduktion über ATM-Breitbandnetzwerke (LAN und WAN) hervor. Bereits Ende 1995 gelang es so, eine verteilte virtuelle Studioproduktion zwischen zwei 300 km entfernten Standorten zu realisieren. Dabei wurden komprimierte Videoinformation und zugehörige Trackingdaten[24] in getrennten *virtual channel connections* über ATM mit einer Netzwerkverzögerung von 4 ms übertragen. [KWG+96, BRR+02]

Aufbauend auf den Erkenntnissen von G-FORS hatte NUGGETS[25] die Konzeption und prototypische Implementierung eines Live-Produktionsszenarios basierend auf einer IT-Infrastruktur und dem MXF-Fileformat zum Ziel. Am Ende der Projektlaufzeit konnte die ferngesteuerte Produktion über ein IP-basiertes ATM-Weitverkehrsnetzwerk (WAN) demonstriert werden. Allerdings handelte es sich bei der „Live-Übertragung“ um einen File-Transfer mit sehr kurzen Verzögerungszeiten [RDR+03]. Der Protokollstapel IP/ ATM/SDH garantierte eine Gesamtlatenz[26] von 13 ms bei einem Jitter[27] von kleiner 1,5 ms [DHL+04]. Als wesentlicher Bestandteil der Gesamtverzögerung wurde die Kompression der Videodaten identifiziert; der Einfluss von MXF-Wrapping und Netzwerkübertragung war dagegen zu vernachlässigen.

Live-Produktion auf Ethernet-Netzen

Obwohl viele Untersuchungen ATM als ideale Netzwerktechnologie für echtzeitkritische Datenübertragung im Studio propagierten, kann ATM nicht im großen Ausmaß eingesetzt werden, weil Geräte und Technologie noch immer mit enormem finanziellen Aufwand verbunden sind. Deshalb wird aktuell an Möglichkeiten geforscht, wie Ethernet-Netzwerke, die keine verbindlichen Garantien bezüglich Durchsatz, Latenz usw. geben können, dennoch für den echtzeitkritischen Bereich genutzt werden können.

[22] Advanced Communications Technology and Services, ein Programm des „Fourth Framework Programme of European Community activities in the field of research and technological development and demonstration“ (1994 bis 1998)

[23] Sept. 1995 gestartet

[24] Trackingdaten repräsentieren extrinsische und intrinsische Kameradaten und kodieren damit die Position und Ausrichtung der Studiokamera sowie deren Parameter wie Brennweite, Öffnungsblende usw. in Virtual-Set-Anwendungen, um eine perspektivisch richtige Hintergrundgrafik berechnen zu können

[25] Networks Used in Globally Generic Television Systems - ebenfalls ein IST-gefördertes Projekt, Laufzeit: zwei Jahre ab April 2002

[26] Einweglatenz für Verbindungen innerhalb von Deutschland

[27] Varianz der Latenz

Die Nutzung von IP-basierenden Ethernet-Netzwerken ist nur unter speziellen Bedingungen möglich, wenn wie in Fernsehproduktionsstudios hohe Anforderungen an Echtzeit, Datenrate, Jitter und Fehlerfreiheit gestellt werden. Ein naheliegender Lösungsansatz, der des *Overprivisioning*, stellt dem Netzwerk immer ausreichend Bandbreite zur Verfügung. Staus können deshalb nicht entstehen und Verzögerungen sind minimal. Dieser Ansatz hat allerdings einen ineffizienten Ressourcenverbrauch zur Folge, d.h. eine Erweiterung des Netzwerkes (um z.B. unvorhersehbar viele zusätzliche Kommunikationsteilnehmer bzw. Datenströme) kann langfristig nicht ökonomisch realisiert werden. Eine weitere Lösungsmöglichkeit ist die Umsetzung von Quality-of-Service-(QoS)-Mechanismen. [BEF+00]

Aktuell werden in der Literatur zwei QoS-Mechanismen diskutiert: Das *Integrated Services* Modell (IntServ) ermöglicht verbindliche Garantien, indem Ressourcen vor der eigentlichen Kommunikation reserviert werden. Eine Skalierung der Netzwerkgröße ist mit IntServ problematisch, da für die Reservierung notwendige Statusinformationsmenge (und damit verbundener Verwaltungsaufwand) proportional mit der Anzahl der neuen Verbindungen steigt. *Differential Services* (DiffServ) ist ein klassenbasiertes Modell, das besser skaliert aber nur qualitative Garantien ermöglicht. Im Projekt PRO-NET wurde IntServ als bevorzugtes Modell für die Nutzung innerhalb von Studios und DiffServ für die Weitverkehrsverbindung (WAN) zwischen Studios identifiziert. [BEF+00, TBP+03]

Die Nutzung von IP-Netzwerken zur zuverlässigen Echtzeitübertragung über offene Wide-Area-Networks (WAN) wird zu Recht angezweifelt, da die Anzahl der Kommunikationsteilnehmer und der damit verbundene Datenverkehr nicht zuverlässig prognostiziert werden kann. Für eine verzögerungsarme Übertragung benötigt man zuverlässige Übertragungswege, die z.B. mit ATM oder separaten optischen Links zur Verfügung stehen. [NJH08]

Eine auf Standard-Technologien basierende Verwaltung und Steuerung von Audio-Datenströmen auf IP-Netzwerken wird in [SKM03][28] vorgestellt. Die Basis dafür ist die *Audio/Video Stream Specification* der *Object Management Group* (OMG). Mithilfe von CORBA (*Common Object Request Broker Architecture*) wurde die Steuerung implementiert, der Transport selbst erfolgt dabei über das *Real-Time Transport Protocol* (RTP) durch das *Java Media Framework* (JMF). Eine zentrale Kontrollinstanz (*studio manager*) kann sich so auf Verwaltung von Essenzen konzentrieren, anstatt einzelne Geräte und Kommunikationsendpunkte individuell ansprechen zu müssen.

[28] im Rahmen des EPSRC/LINK Projektes 'Production of Broadcast Content in an object-oriented IP-based Network' PRO-NET, Beginn: Januar 2000, Ziel: Nutzung von Standard-Netzwerken zum Austausch von Echtzeit-Audio- und Videoinformationen zwischen Fernsehproduktionstudios

IT-Netzwerk-Übertragung in anderen Zusammenhängen

Automatisierungstechnik In vielen Bereichen der Industrie müssen Anlagen und Anlagenteile gesteuert und überwacht werden, z.B. Produktionslinien in der Autoindustrie. Voraussetzung für eine stabile Produktion sind zuverlässige und hochperformante Infrastrukturen, die Befehle und Statusmeldungen in sehr engen Zeitschranken übertragen können. In der Vergangenheit hat sich dafür die so genannte Feldbus-Technologie etabliert, die eine Menge von proprietären Produkten zur Folge hatte. Um die Vorzüge etablierter IT-Standards nutzen zu können (z.B. FTP zum Dateiaustausch zwischen Geräten, HTTP zur Konfiguration über Webseiten), bestehen seit einiger Zeit Bestrebungen, Echtzeitkommunikation über Ethernet-Strukturen zu ermöglichen. [Fel05]

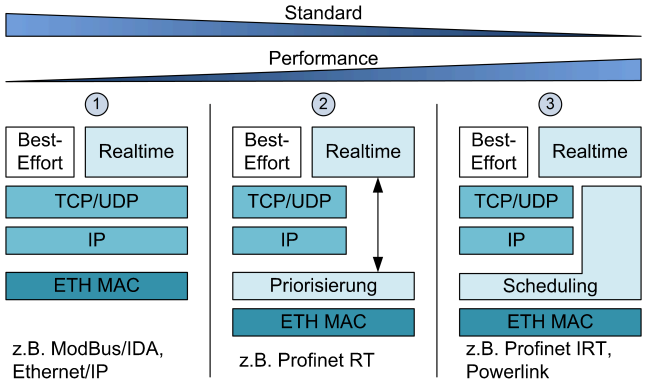


Abbildung 2.4: Klassifikation von Echtzeitverfahren der Industriellen Automation nach [JN04]

In der Automatisierungstechnik gilt es, relativ kurze Nachrichten in kurzer Zeit (Verzögerungen kleiner als 1 ms, bei z.T. max. 1 μs Jitter) zuverlässig über meist Bus-förmige Topologien zu übertragen [Fel05]. Abbildung 2.4 zeigt eine Übersicht über die Verfahren. Dabei beschreibt die Klasse 1 die Anwendung des Standard-TCP/IP-Protokollstapels ohne jegliche Änderung. In dieser Klasse nutzen verschiedene Echtzeit- und Best Effort-Protokolle (HTTP, SNMP, FTP) die Dienste von TCP/IP. In Klasse 2 werden Optimierungen der Softwareverarbeitung eingeführt, bei denen Echtzeit-Daten den TCP/IP-Stapel umgehen und damit Verzögerungszeiten minimiert und der maximale Durchsatz erhöht wird. Die Klasse 3 modifiziert den Protokollstapel auf Hardwareebene, indem die verfügbare Bandbreite in exklusive Blöcke für Echtzeit-Daten und Blöcke für Nicht-Echtzeitdaten unterteilt wird. Produkte der Klasse 3 können o.g. Anforderungen hinsichtlich der maximalen Verzögerungszeiten und der erforderlichen Jitterwerte erfüllen. [JN04]

Abbildung 2.4 zeigt weiterhin, dass mit abnehmender Standard-Kompatibilität (von links nach rechts) die erreichbare Echtzeit-Performanz ansteigt. Sollen Kosten durch den

Einsatz von Standard-Hardware in Grenzen gehalten werden, scheidet der Einsatz von Klasse-3-Produkten aus.

Bezüglich der Anforderungen an den Kommunikationskanal bzw. die Netzwerkstruktur ähneln sich Industrieautomation und TV-Studio in den wesentlichen Punkten. Gefordert wird [Fel05, Fur03]:

- eine fehlertolerante Übertragung (eingebaute Fehlererkennung und -korrektur)
- ausfallsicherer Betrieb (zuverlässige Funktionalität auch nach Systemabstürzen und unter extremen Umwelteinflüssen)
- definierte Dienstgüte (Quality of Service - QoS), z.B. garantierte Antwortzeit von $1 .. 100\ ms$, Jitter $< 1\ \mu s$ (je nach Einsatzzweck)
- flexible Installation (oft linienförmige Topologien, große Anzahl von Knoten)

Im Unterschied zur breitbandigen Anwendung im TV-Studio benötigt die Industrieautomation aber nur geringe Übertragungsraten, da sich die zu übertragenden Daten im Wesentlichen auf kurze Befehls- und Statusinformationen beschränken.

Hörfunkproduktion Die Nutzung von IT-Systemen inkl. entsprechender Netzwerke zur Übertragung und Bearbeitung von Daten im Hörfunkbereich ist weiter entwickelt. Seit etwa 1995 sind dort ausgereifte IT-Systeme am Markt, die an die hörfunkspezifischen Anforderungen angepasst sind. Dies ist zum einen mit den um bis zu Faktor 1000 geringeren Anforderungen an Datenraten und Datenmengen begründet. Zum anderen ist das Geschäftsvolumen im Hörfunksystem-Markt im Vergleich zum Fernsehen gering, so dass bei der Standardisierung von z.B. Dateiformaten, Kompressionsverfahren und Metadaten weniger Interessen zu berücksichtigen sind und der Standardisierungsprozess damit schneller verläuft. [HK04a]

Internetstreaming Auch im Endnutzerbereich spielen Echtzeit-Datenübertragungen eine Rolle. Beim Streaming von A/V-Daten über das Internet besteht beispielsweise ebenfalls die Anforderung, zeitkontinuierliche Medien möglichst fehlerfrei zu übertragen. Dabei handelt es sich (noch) um meist stark komprimierte Videos mit entsprechend geringer Datenrate, weil die prinzipiellen Zugangsvoraussetzungen[29] des Endkunden zum Internet beschränkt sind. Entsprechend gering ist die Erwartungshaltung der Nutzer: Mäßige Bildqualität, kleinere Störungen durch Übertragungsfehler und vor allem eine verzögerte Darstellung durch große Empfangspuffer werden i.d.R. akzeptiert.

[29] „Einwahl“ über *Internet Service Provider* (ISP), damit limitierte Maximalübertragungsbandbreiten in Download- und Uploadrichtung

2.3 Zusammenfassung

Traditionelle Studioinfrastrukturen haben inhärente Einschränkungen, die die flexible Nutzung z.B. für die Produktion von A/V-Inhalten für multiple Distributionswege nicht oder nicht hinreichend automatisiert unterstützen. Dazu zählen die Verwendung von funktionsspezifischen Geräten, die Schnittstellenvielfalt (inkl. des daraus resultierenden Verkabelungsaufwandes) sowie das Fehlen von unterstützenden Mechanismen zur durchgängigen Metadatenanwendung.

Die Lösung wird in der umfangreichen Anwendung von flexibler IT-Technologie (PC-Hardware und -Software) gesehen. Vergangene und aktuelle Forschungsaktivitäten in diesem Bereich beziehen sich auf die Anwendung von preiswerter PC-Technik zur Signalbearbeitung und IT-Netzwerktechnologie zur Signalübertragung sowie die prinzipiellen Anwendungsmöglichkeiten von Metadaten in der Medienproduktion. Auch die Entwicklung von Middleware zur Bewältigung von komplexen Steuerungsabläufen ist ein zentrales Thema.

Die Nutzung von „nicht echtzeitfähigen“ aber weit verbreiteten und preiswerten Netzwerktechnologien wie Ethernet zur Contentübertragung im Fernsehproduktionsstudio ist eine Forschungsrichtung, die von zwei Seiten motiviert wird. Zum einen wird die Leistungsfähigkeit zugrunde liegender Geräte ständig verbessert (kürzere Schalt- und Rechenzeiten, geringere Latenzen usw.). Zum anderen wird deutlich, dass zur Definition des Echtzeitbegriffs verwendete Parameter nur in speziellen Anwendungsfällen wirklich notwendig sind – in vielen Fällen entspannen sich die Anforderungen deutlich. Diese Arbeit betont diese Richtung und setzt sich mit den damit verbundenen Herausforderungen auseinander.

Weiterhin ist die Anwendung von Metadaten im Produktionsbereich aus Sicht des Autors noch nicht erschöpfend gelöst. Die sich daraus ergebenden Vorteile in Bezug auf effiziente Arbeitsweisen wurden bisher nur in den Bereichen Postproduktion, Distribution und Archivierung betont – der Bereich der Produktion wurde dabei fast vollständig ausgeklammert. An diesem Punkt setzt die vorliegende Arbeit an.

3 Datenübertragung über Netzwerke

Ausgangspunkt für ein erweiterungsfähiges Netzwerk zur Datenübertragung im Studio stellen standardisierte IT-Netzwerke dar. In diesem Kapitel werden deshalb zunächst Grundlagen der Datenübertragung über Netzwerke erläutert. Dies erfolgt zielführend zum Zweck der Auswahl von Technologien und Prinzipien vor dem Hintergrund der Anwendung im Studiobereich. Dazu werden aufbauend auf eine Klassifikation der Telekommunikationsnetzwerke Schichtenmodelle diskutiert und schichtenspezifisch Protokolle bzw. Formate identifiziert, die zur Implementierung in Frage kommen. Zunächst erfolgt die Definition einiger Begriffe.

Unter einer **Nachricht** versteht man eine mit dem Ziel der Weitergabe gebildete Information, die von einer Nachrichtenquelle ausgeht und an einer räumlich entfernten Nachrichtensenke aufgenommen wird. Zur Übertragung der Nachricht dienen **Signale**, d.h. Verläufe physikalischer Größen über die Zeit (z.B. Spannungen oder Stromstärken). Die Nachrichtenübertragung erfolgt über einen Nachrichtenkanal, an dessen Eingang im Allgemeinen eine Anpassung an die spezifischen Eigenschaften des Kanals stattfindet. Diese Kanalmodulation muss am Ausgang des Kanal wieder invertiert werden. **Daten** sind aus Signalen gewonnene Werte bzw. Zeichen auf Basis endlicher Wertebereiche [BMW05]. Zeitkontinuierliche Multimediadaten wie Audio und Video haben einen **Datenstrom** zur Folge, der aus einer Reihe von temporal abhängigen Elementen besteht.

Technische **Kommunikation** beschreibt den Prozess des Nachrichtenaustauschs zwischen technischen Systemen (auch: Maschinenkommunikation). „Telekommunikation“ – die Kommunikation über größere Entfernungen – umfasst neben der Daten- und Sprachübertragung auch die Hörfunk- und Fernsehübertragungstechnik sowie die Kommunikation über Rechner- und Mobilfunknetze.

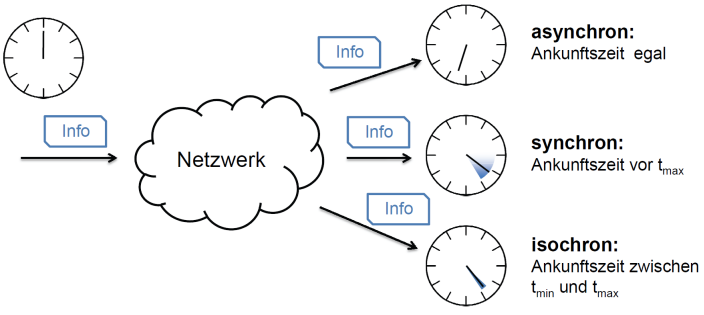


Abbildung 3.1: Übertragungsmodi nach [TS08, S. 185]

Bezüglich der Übertragung eines Datenstromes kann man folgende Übertragungsmodi voneinander abgrenzen: Unterliegen die Ankunftszeiten der Datenstromelemente keiner zeitlichen Beschränkung, spricht man von *asynchron*er Übertragung. Eine *synchron*e Übertragung erfolgt garantiert vor einem bestimmten Zeitpunkt, während die Ankunftszeit einer *isochron*en Übertragung sowohl durch die Angabe einer oberen (t_{max}) als auch einer unteren Grenze (t_{min}) sehr genau determiniert ist (siehe Abbildung 3.1). [1]

3.1 Telekommunikationsnetzwerke

Telekommunikationsnetzwerke lassen sich in zwei grundlegende Klassen einteilen: Leitungsvermittlung (*Circuit Switched Networks*) und Paketvermittlung (*Packet Switched Networks*) [KR05]. Die zur Übertragung benötigten Ressourcen (Leitungen, Puffer usw.) sind bei der Leitungsvermittlung reserviert, d.h. eine Datenübertragung kann nicht durch andere Übertragungen beeinflusst werden. Sender und Empfänger sind direkt miteinander verbunden. Mit dieser Sicherheit ist ein gewisses Maß an Ineffizienz verbunden: Von aktueller Kommunikation ungenutzte Ressourcen können nicht anderweitig genutzt werden. Für die gleichzeitige Nutzung leitungsvermittelter Netzwerke zur Übertragung mehrerer Datenströme können verschiedene Multiplexverfehren eingesetzt werden (*Time Division Multiplexing – TDM, Frequency Division Multiplexing – FDM*).

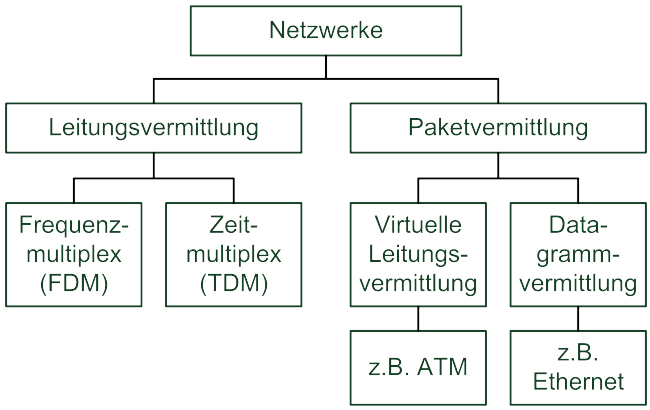


Abbildung 3.2: Telekommunikationsnetzwerk-Klassifikation nach [KR05]

Weil bei Paketvermittlung keine Ressourcen reserviert werden, können damit effiziente Kommunikationsnetzwerke implementiert werden. Eine Nachricht wird beim Sender (z.B. in Pakete) aufgeteilt und über Netzwerkknoten zum Empfänger weitergeleitet. Im

[1] Diese Definition widerspricht z.T. dem üblichen Sprachgebrauch im Broadcastbereich, in dem „synchron“ im Sinne eines Gleichlaufes mit einem zentralen Studiotakt verwendet wird und damit eher o.g. Definition von „isochron“ entspricht. Im Sinne der klaren und eindeutigen Verwendung von Begrifflichkeiten werden innerhalb dieser Arbeit o.g. Definitionen von „asynchron“, „synchron“ und „isochron“ angewendet.

Vergleich zur Leitungsvermittlung ergibt sich dadurch eine bessere Ressourcennutzung. Die Implementierung ist meist einfacher und preiswerter.

Paketvermittelte Kommunikation lässt sich nochmals in virtuelle Leitungsvermittlung (*Virtual Circuit Networks*) und Datagrammvermittlung (*Datagram Networks*) unterteilen. Bei ersterer wird eine virtuelle Leitung zwischen Sender und Empfänger etabliert, die auch alle dazwischen liegenden Netzwerkknoten umfasst. Diese virtuelle Leitung wird mit einer eindeutigen Identifikationsnummer (*Virtual Circuit Identifierer* – VCID) versehen, durch die indirekt Sender- und Empfängeradressen repräsentiert werden. Jeder Netzwerkknoten pflegt eine Tabelle, die jede VCID auf seine Anschlüsse abbildet. Bei jeder Änderung einer virtuellen Leitung müssen die Tabellen aller betroffenen Netzwerkknoten aktualisiert werden. Dies setzt komplexe Wartungsprotokolle voraus. Andererseits ermöglicht eine kompakte Tabelle einen schnellen Schaltvorgang im Netzwerkknoten, der in Kombination mit bekannten Routen Verzögerungen vorhersagbar macht und damit die Voraussetzung für eine Echtzeitkommuniaktion erfüllt.

Bei Datagrammvermittlung werden die einzelnen Pakete mit einer Empfängeradresse versehen dem Netzwerk überlassen. Anhand der Adresse erfolgt die Weiterleitung der Pakete durch die Netzwerkknoten zum Empfänger. Dabei können Pakete einer Nachricht verschiedene Wege durch das Netzwerk nehmen und resultierend aus unterschiedlichen Laufzeiten in falscher Reihenfolge am Ziel ankommen. Der hohe Aufwand des (virtuellen) Verbindungsaufbaus und -managements wird umgangen – allerdings auf Kosten der Vorhersagbarkeit von Verzögerungen, die Echtzeitkommunikation ohne zusätzlichen (Protokoll-)Aufwand nicht ermöglicht.

In der vorliegenden Arbeit soll die Anwendung von paketvermittelten Netzen mit dem Fokus auf Datagramm-Vermittlung im Studiobereich untersucht werden. Das Internet als prominentes Beispiel für ein heterogenes datagrammvermittelndes Netzwerk auf Basis des Internetprotokolls (IP) wird dazu in vielen Abschnitten Ausgangspunkt für Diskussionen sein.

3.1.1 Übertragungstechniken

Die Übertragung von Datenströmen kann mithilfe verschiedener Techniken realisiert werden, die sich bezüglich Anzahl der Kommunikationsteilnehmer und der Richtung des Informationsflusses unterscheiden. *Unicast* beschreibt die Ende-zu-Ende-Verbindung zwischen einem Client und einem Server. Dabei kann der Datenfluss in eine Richtung beschränkt sein (*simplex*) oder zeitlich nacheinander in beide Richtungen (*half-duplex*) bzw. gleichzeitig in beide Richtungen (*duplex*) möglich sein. Bei Übertragung von großen Datenmengen an mehrere Empfänger müssen entsprechend viele Verbindungen aufgebaut werden, die eine große Belastung des Senders und des Netzwerkes zur Folge haben und damit keine effiziente Datenübertragung ermöglichen.

Multicast bezeichnet den Datenempfang mehrerer Empfänger von einem Sender über ein Zwischensystem, das eine Empfängergruppe verwaltet. Eine Nachricht an die Adresse der Empfängergruppe wird durch das Zwischensystem dupliziert und an jeden Teilnehmer der Gruppe weitergeleitet. Multicast erlaubt einerseits eine ressourcensparende Übertragung, verlangt andererseits aber einen Verwaltungsaufwand auf den Zwischensystemen für die einzelnen Gruppen und die Gruppenzugehörigkeit der Empfänger. *Broadcast* bezeichnet einen Sonderfall der Multicastübertragung, der alle Kommunikationsteilnehmer eines Netzwerk(segment)s als Empfänger adressiert.

Entgegen der klaren Trennung in Sender und Empfänger o.g. Übertragungstechniken bilden *Peer-to-Peer*-Ansätze ein selbstorganisierendes Overlay-Netz aus Verbindungen zwischen den einzelnen Teilnehmern. Jeder Teilnehmer stellt einen Teil seiner lokalen Ressourcen zur Verfügung und leitet erhaltene Daten an andere Teilnehmer weiter. Verzögerungen (Delays), Verzögerungsschwankungen (Jitter) und Paketverluste durch Übertragungsfehler treten in Peer-to-Peer-Netzen verstärkt auf, da die Daten nicht direkt zwischen Quelle und Empfänger ausgetauscht werden. Des weiteren steigt die Wahrscheinlichkeit für Paketvertauschungen, was für multimediale Ströme, die latenzarm wiedergegeben werden sollen, besonders kritisch ist. [Str07, S. 4–5] Overlay-Netze erlauben die Abbildung logischer Netzwerke auf existierende Infrastrukturen (siehe Abbildung 3.3).

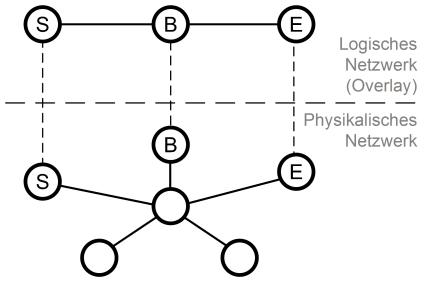


Abbildung 3.3: Sender (S), Bearbeitungsstationen (B) und Empfänger (E) eines Overlay-Netzes

In Bezug auf die effektive Signalverteilung von einer Quelle auf mehrere Senken, wie sie im Studio durch Kreuzschienen und Steckfelder realisiert wird, ist die Multicastübertragung zu bevorzugen. Viele hochratige Unicast-Datenströme (Video) würden die Kapazität des Senders und des Übertragungskanals ausschöpfen.

3.1.2 Netzwerktopologien

Hinsichtlich der Struktur der Vernetzung kann man verschiedene Topologien unterscheiden, die in Abbildung 3.4 aufgeführt sind. Diese Topologien lassen sich sowohl bezüglich dem jeweils notwendigen Planungs-, Installations- und Administrationssaufwand sowie der Eignung für die Übertragung vieler hochratiger Datenströme differenzieren.

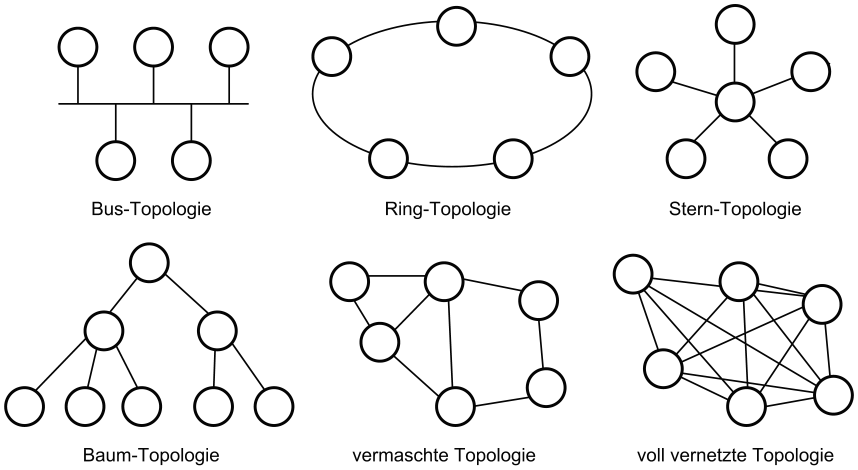


Abbildung 3.4: Netzwerktopologien [Ote08]

Die gemeinsame Nutzung eines physikalischen Übertragungskanals bei Bus- und Ring-Topologie erweist sich insbesondere für hochratige Datenübertragung als kritisch: Die zur Verfügung stehende Bandbreite muss zwischen allen Teilnehmern aufgeteilt werden. Zusätzlich sind Strategien zur Kollisionsvermeidung notwendig.

Sollen Kapazitätsprobleme vollständig ausgeschlossen werden, wächst der Planungs- und Administrationsaufwand für vermaschte Topologien stark an. Die Vollvernetzung bringt einen mit der Anzahl der Kommunikationsteilnehmer potentiell steigenden Verkabelungsaufwand mit sich. Bei vermaschter und voll vernetzter Topologie erweist sich das Routing, also das Festlegen des Signal- bzw. Datenweges, als aufwändig.

Sinnvoll nutzbare Topologien stellen letztlich die Stern- und die verwandte Baum-Topologie dar. Durch zentrale Elemente (z.B. *Switches*) entkoppelt, verfügt jeder Kommunikationsteilnehmer über eine exklusive Leitung zum zentralen Element; Kollisionen können nicht mehr auftreten und die volle Bandbreite steht exklusiv für den jeweiligen Teilnehmer zur Verfügung. Die zentralen Elemente müssen entsprechend leistungsstark ausgeprägt sein, damit alle für einen Anwendungsfall notwendigen Teilnehmer gleichzeitig und störungsfrei miteinander kommunizieren können.

3.1.3 Netzwerkleistung und Ressourcenzuteilung

Neben den funktionalen Aspekten von Netzwerken ist insbesondere für Echtzeitanwendungen eine Bewertung der Leistungsfähigkeit wichtig, um Netzwerke gemäß der Anforderungen zu dimensionieren. Folgende Definitionen und Erläuterungen erfolgen im Hinblick auf die Anwendung datagrammvermittelnder Netzwerke im Studiobereich. Bei der Übertragung von Datenströmen handelt es sich also um ein Versenden von Paketen, die –

jeweils mit einer Empfängeradresse versehen – in einer zeitlichen Abhängigkeit zueinander stehen. Für solche Echtzeitanwendungen wird die Leistungsfähigkeit eines Netzwerkes im Wesentlichen über Latenz und verfügbare Datenrate des Netzwerkes definiert.

Die **Latenz** bzw. Verzögerung beschreibt die Dauer der Ende-zu-Ende-Übertragung eines Paketes vom Senden des ersten Bits bis zum Empfang des letzten Bits. Im Wesentlichen können drei Ursachen für Latenz differenziert werden: Die endliche Ausbreitungsgeschwindigkeit der elektrischen Signale auf dem Leiter, die Größe der zu übertragenden Pakete und Wartezeiten in Netzwerkknoten (siehe Latenzmodell in Abschnitt 3.2). Prinzipiell kann die Latenz sehr feingranular theoretisch bestimmt und z.T. auch praktisch gemessen werden. Man unterscheidet zwischen der Einweg-Verzögerung *(One Way Delay* - OWD) und der Umlaufverzögerung (*Round Trip Delay* - RTD bzw. *Round Trip Time* - RTT) [Sie05, S. 16]. Die Bestimmung des OWD kann nicht in jedem Fall auf Basis der Halbierung der RRT erfolgen, da insbesondere in Netzwerken mit vielen Netzwerkknoten Hin- und Rückweg des Pakets nicht identisch sein müssen.

Die **Datenrate** bzw. der Durchsatz gibt an, wie viele Dateneinheiten pro Zeiteinheit (z.B. Bit pro Sekunde – bit/s) übertragen werden können. Die Datenrate ist ein Maß der Übertragungsgeschwindigkeit und wird meist als Durchschnittswert zum Ausdruck gebracht, der in Abhängigkeit des zugrunde liegenden Zeitintervalls den tatsächlichen Durchsatz mehr oder weniger mittelt. Die Vorsätze zu den Maßeinheiten für Datenraten werden (im Gegensatz zur Angabe bei Speichermengen) auf Basis von Zehnerpotenzen definiert (z.B. 1 Gigabit/s = 10^9 bit/s). An dieser Stelle ist der Begriff „Bandbreite“ abzugrenzen: Unter der Bandbreite eines Übertragungssystems versteht man den Frequenzbereich in Hertz (Hz), der für die störungsfreie Übertragung zur Verfügung steht. Aufgrund unterschiedlicher Kanalkodierungen können die nominellen Werte von Bandbreite und Datenrate voneinander abweichen.

Neben Datenrate und Latenz werden weitere Parameter der Netzwerkleistung verwendet. Zunächst ist für viele zeitabhängige Anwendungen das zeitliche Verhalten der Latenz wichtig. Dabei beschreibt der **Jitter** die Varianz der Latenz und damit kurzzeitige Abweichungen von den „Sollankunftszeiten“ der Pakete. Der Ausgleich von Jitter erfolgt empfängerseitig durch die Verwendung von Puffern, aus denen Pakete im gewünschten Takt wieder ausgelesen werden können. Die verwendete Puffergröße wirkt sich dabei auf die Latenz aus.

Die Leistungsparameter sind voneinander abhängig. Die Latenz steht in proportionalem Verhältnis zur Puffergröße und im umgekehrt proportionalem Verhältnis zur Übertragungsgeschwindigkeit. Mit steigender Datenrate kann so wahlweise die Puffergröße bei konstanter Latenz erhöht werden oder die Latenz bei konstanter Puffergröße verringert werden. Dieser Zusammenhang vereinfacht die Einhaltung der Echtzeitbedingung in synchronen Systemen erheblich. [Hän07]

Auch die Verzögerung bis zum Start der Datenübertragung (z.B. durch vorherigen Verbindungsaufbau) ist für das Zeitverhalten von Applikationen entscheidend. Ein weiterer

Parameter ist die Wahrscheinlichkeit für Blockierungen (*blocking propability*), die den Prozentsatz an gescheiterten Verbindungsversuchen beschreibt und damit insbesondere für (virtuelle) Leitungsvermittlung signifikant ist [Sha05, S. 9–10].

Die Qualität einer Übertragung über ein Netzwerk bezüglich Fehleranfälligkeit wird mit verschiedenen Parametern beschrieben. **Fehlerraten** beschreiben die Wahrscheinlichkeiten für das Auftreten von Übertragungsfehlern für Informationseinheiten und können sowohl in Bezug auf Bits (*Bit Error Rate* - BER) oder in Bezug auf Pakete angegeben werden. Bitfehler entstehen dabei vorzugsweise aufgrund physikalischer Eigenschaften des Übertragungskanals (z.B. Leitungsdämpfung) und haben meist Paketfehler zur Folge (wenn die Bitfehler nicht durch Fehlerkorrekturmechanismen behoben werden können). Paketfehler, die z.B. durch Vertauschung der Paketreihenfolge oder Verwerfen von Paketen entstehen, sind in der Regel auf Warteschlangenmechanismen in Netzwerkknoten bei Überlastsituationen zurückzuführen. Insbesondere bei Echtzeitanwendungen können Paketvertauschungen zu einer verspäteten Paketankunft führen, die einen Paketfehler zur Folge hat. Das Wissen darüber, in welcher Form z.B. Bitfehler auftreten (*Burstiness*), ist Grundlage für effektive Fehlerschutzmechanismen (z.B. *Forward Error Correction* – FEC).

Die Leistung eines Netzwerkes wird vielen Teilnehmern zur Verfügung gestellt, d.h. die Ressourcen des Netzwerkes müssen zugeteilt werden. Peterson und Davie stellen dafür eine Taxonomie vor, die acht Strategien unterscheidet, von denen wiederum nur zwei Strategien praktische Anwendung finden: *Best Effort* und *Quality of Service* (QoS) [PD07, S. 468–470]. *Best Effort* beschreibt die Eigenschaft eines Netzwerkes, Pakete weiterzuleiten, so lange Kapazitäten vorhanden sind; eine fehlerfreie Übertragung kann nicht garantiert werden. Für QoS oder Dienst(e)güte findet sich in der Literatur keine einheitliche Definition. Einerseits wird mit QoS die wahrgenommenen Güte eines Kommunikationsdienstes aus der Sicht der Anwender beschrieben, auf der anderen Seite bezieht man QoS als Bündel von Anforderung an ein Netzwerk während der Übertragung eines Datenstroms [Sha05, S. 8]. In beiden Fällen kommen o.g. Parameter als verifizierbare Bezugs- und Vergleichsgrößen zum Einsatz.

Die Internet Engineering Task Force (IETF) hat einige Mechanismen vorgeschlagen, über eine Erweiterung zur Internet-Architektur eine Unterstützung von QoS zu erreichen. Die bekanntesten Mechanismen und Modelle *Integrated Services Model* (IntServ), *Differentiated Services Model* (DiffServ), *Traffic Engineering* (TE), *Multi-Protocol Label Switching* (MPLS) und *Constraint-based Routing* (CBR) sollen im Folgenden kurz beschrieben werden [Sha05, S. 3–8].

IntServ sieht die Reservierung von Netzwerkressourcen entlang des Übertragungspfades vor. Neben *Best Effort* unterstützt *IntServ* zwei weitere Service-Klassen, die von interaktiven Applikationen mit zeitabhängigen Datenströmen genutzt werden können. Der *Controlled Load Service* (CS) begrenzt die Anzahl von Datenströmen, um das Verhalten des Netzwerkes bei niedriger Belastung zu bewahren. CS garantiert nur eine minimale Datenrate; Fehlerraten und Latenzen werden nicht zugesichert. Der *Guaranteed Service*

(GS) hingegen garantiert eine maximale Verzögerung und Fehlerfreiheit. Insgesamt stellt *IntServ* hohe Anforderungen an die Netzwerkknoten. Zum einen müssen die Verbindungsinformationen auf jedem Knoten verwaltet und gespeichert werden, woraus sich die begrenzte Skalierbarkeit von *IntServ* erklärt. Weiterhin müssen alle Netzwerkknoten das verwendete Reservierungsprotokoll (z.B. Ressource Reservation Protocol – RSVP) unterstützen, was mit Kosten verbunden ist.

DiffServ klassifiziert Datenströme, indem die Pakete durch korrespondierende Flags (Bits) gekennzeichnet werden. Je nach „Wertigkeit" des Pakets erfolgt eine mehr oder weniger bevorzugte Weiterleitung in den Netzwerkknoten. Auch *DiffServ* unterstützt neben *Best Effort* zwei weitere „*Services*"[2]. *Expedited Forwarding* (EF) „garantiert" einem entsprechend gekennzeichnetem Datenstrom eine einstellbare Abgangsdatenrate von Netzwerkknoten, indem es notfalls Pakete anderer Datenströme verwirft. *Assured Forwarding* (AF) stellt Klassen zur Verfügung, denen in den Netzwerkknoten verschiedene Ressourcen zugeordnet sind, wobei jede Klasse mehr Ressourcen nutzen kann, solange diese verfügbar sind. In Überlastsituationen werden einzelne Pakete innerhalb einer Klasse auf Basis einer weiteren Hierarchie verworfen. Im Vergleich zu *IntServ* entlastet *DiffServ* die Netzwerknoten, übernimmt aber keine QoS-Garantien.

Traffic Engineering beschreibt einen iterativen Prozess, der versucht, Staus und Verstopfungen im Netzwerk durch Gestaltung und Optimierung der Datenflüsse zu umgehen. Dazu werden Anforderungen identifiziert, der tatsächliche Bedarf an Verbindungswegen berechnet, geeignete Technologien identifiziert, QoS-Parameter ermittelt und die resultierenden Optimierungen umgesetzt. TE findet in Netzwerken Anwendung, deren Dimensionierung nicht oder nur im geringen Maße beeinflussbar ist. Die Anpassung des Netzwerkes (Auswahl und Konfiguration des Netzwerkequipments usw.) wird unter dem Begriff *Network Engineering* (NE) zusammengefasst [OS02, S. 7].

Multi-Protocol Label Switching ist eine Technologie, die die Weiterleitung von Paketen in Netzwerkknoten vereinfacht und damit beschleunigt. Dazu wird einem Paket bei Netzwerkeintritt ein Label angeheftet, das den Weg durch das Netzwerk vorbestimmt. Die Weiterleitung in den Netzwerkknoten erfolgt dann nicht mehr auf Basis der Zieladressinformation (aus der vom Knoten der weitere Weg durch das Netz respektive der korrespondierende Ausgangsport berechnet wird), sondern auf Grundlage des Labels. Der Vorteil der schnelleren Weiterleitung schrumpft mit tendenziell leistungsfähigeren Knoten. Dennoch kann MPLS in Verbindung mit TE dazu genutzt werden, eine explizite Route durch das Netzwerk festzulegen und damit helfen, Überlastsituationen zu vermeiden.

Constraint-based Routing berechnet Pfade durch das Netzwerk auf Grundlage von Bedingungen wie Datenrate, Pfadlänge, Netzwerktopologie u.a. und wählt den Pfad, der gestellte QoS-Anforderungen am besten erfüllt. Z.B. könnte ein langer aber wenig benutzter Pfad einem kürzeren aber viel benutzten Pfad vorgezogen werden. CBR hat einen

[2] im DiffServ-Kontext: *per-hop-behaviour* (PHB)

erhöhten Berechnungsaufwand in den Netzwerkknoten und größere Routingtabellen zur Folge.

Es wird deutlich, dass die Verknüpfung von *IntServ*, *DiffServ*, TE, MPLS und CBR in existierenden Netzwerken mit einer großen Anzahl von Netzwerkknoten und Teilnehmern möglich und sinnvoll ist. In kleineren lokalen Netzwerken nimmt die Bedeutung von MPLS und CBR ab. Zur möglichst selbstgesteuerten Etablierung einer Dienstegüte muss also eine Entscheidung zwischen einer teuren Implementierung auf Basis von Ressourcenreservierungen (*IntServ*) und einer „Verwaltung von Mängeln“ ohne Garantie (*DiffServ*) getroffen werden.

In relativ statischen lokalen Netzwerken mit bekannter Anzahl von Teilnehmern und definierten Anforderungen der Teilnehmer muss in Datagramm-vermittelten Netzwerken kein selbstgesteuerter QoS-Mechanismus (*IntServ/Diffserv*) etabliert werden, um eine zuverlässige und sichere Übertragung von hochratigen Datenströmen zu ermöglichen. Vor diesem Hintergrund erfolgt nun die Modellierung von Netzwerkkomponenten bezüglich des für Echtzeitanwendungen kritischen Parameters Latenz.

3.2 Latenzbetrachtungen zur Übertragung

Wenn Daten- und Fehlerrate eines Netzwerkes bekannt und angemessen sind, kristallisieren sich Latenz und deren Abweichung (Jitter) als bestimmende Parameter für Echtzeitanwendungen heraus. Deshalb soll im Folgenden ein Latenzmodell der Übertragung Aufschluss darüber geben, an welchen Stellen Optimierungsmechanismen angesetzt werden können.

Eine Netzwerkübertragung findet immer zwischen einem Sender und einem Empfänger statt, dazwischen können n Knoten liegen. Abbildung 3.5 verdeutlicht in diesem Sinne die Komponenten der Verzögerung, die sowohl bei Sender und Empfänger ($t_{\text{Sender}}, t_{\text{Empfänger}}$) als auch in jedem Netzwerkknoten (t_{Knoten}) entstehen. Als Hauptursache für Latenzen können Puffer identifiziert werden, die sich applikationsabhängig am Ein- und/oder Ausgang befinden.

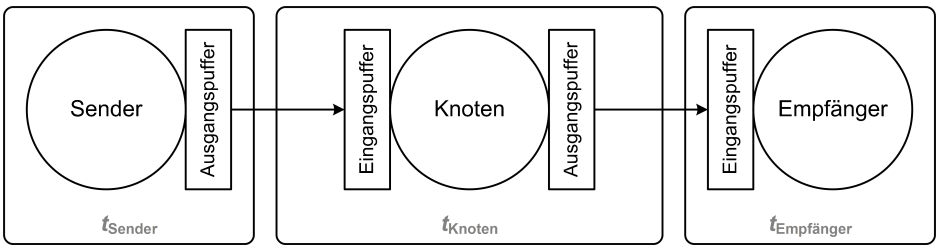


Abbildung 3.5: Latenzkomponenten einer einfachen Netzwerkübertragung

Innerhalb der Knoten bzw. zwischen Sender und Empfänger kann man Latenzkomponenten unterscheiden, die durch Netzwerkprotokoll-Verarbeitung (t_{nv}, *processing*), Zwischenspeicherung in Puffern (t_{zs}, *queuing*), Sendevorgang (t_{sv}, *transmission*) und Leitungsverzögerung (t_{lv}, *propagation*) entstehen [KR05]:

$$t_{\mathrm{Knoten}} = t_{\mathrm{nv}} + t_{\mathrm{zs}} + t_{\mathrm{sv}} + t_{\mathrm{lv}}$$

Zunächst muss jedes eintreffende Paket in einem Knoten verarbeitet werden, d.h. die Header-Information muss ausgewertet und die Zieladresse bestimmt werden. Diese Verzögerung durch die Verarbeitung wird in praktischen Implementierungen mit $t_{\mathrm{nv}} \leq 5\mu$s gemessen [KR08].

In Abhängigkeit des Verkehrsaufkommens (Anzahl der Datenströme und Anzahl der Pakete pro Datenstrom) müssen Pakete vor der Verarbeitung bzw. der Übertragung in Puffern zwischengespeichert werden. Die daraus resultierende Verzögerung t_{zs} ist abhängig vom verwendeten Schedulingalgorithmus, also der Bearbeitungsreihenfolge der Pakete, und hat eine praktische Größenordnung von μs bis wenigen ms. Da t_{zs} von Paket zu Paket variiert, erfolgt eine statistische Beschreibung.

Die Bezeichnung *Store and Forward* zielt auf die übliche Betriebsart von *Switches* ab, weil hier erst das gesamte Paket empfangen wird, um nach Überprüfung auf fehlerfreie Übertragung des Paketes mittels CRC[3]-Algorithmen und Bestimmung der Empfängeradresse mit dem erneuten Senden des Paketes (Bit für Bit) zu beginnen. Im Gegensatz dazu existiert die Betriebsart *Cut Through*, die die üblicherweise im Header des Paketes befindliche Empfängeradresse sofort nach Empfang der dafür notwendigen Bits erfasst und alle folgenden Bits des Paketes „durchschaltet“. Die Prüfung auf Fehlerfreiheit des Pakets entfällt, so dass die Latenz im Knoten minimiert wird; allerdings verbunden mit der Gefahr der Weiterleitung defekter Pakete. Beide Betriebsarten können zum *Adaptive Cut Through Switching* oder *Intelligent Switching* kombiniert werden, das erst oberhalb eines definierten Fehlerschwellenwertes in der (langsamen) *Store-and-Forward*-Betriebsart arbeitet, sonst im (schnellen) *Cut-Through*-Modus [Köh99].

Die Zeit, um einzelne Bits eines Pakets auf den Übertragungskanal zu senden (t_{sv}) und ist abhängig von der Paketgröße L und der Übertragungsrate R der Verbindung: $t_{\mathrm{sv}} = L/R$. Für die Übertragung von für Multimediainhalten üblichen großen Paketen erhöht sich somit der Anteil der Sendeverzögerung. Für eine Paketgröße von L = 1.500 Byte und einer Übertragungsrate von R = 1 GBit/s ergibt sich ein theoretischer Wert von 12 μs.

Letztlich trägt auch die Verzögerung durch die begrenzte Signalgeschwindigkeit auf dem physikalischen Leiter (Leitungsverzögerung) zur Gesamtlatenz bei. Unter der Annahme,

[3] Cyclic Redundancy Check

dass die Übertragungsgeschwindigkeit auf Kupferkabeln ca. 2/3 Lichtgeschwindigkeit beträgt, kann t_{lv} mit ca. 0,5 μs pro 100 m Kabellänge approximiert werden.

Die Summe der Knotenverzögerungen in einem Netzwerk mit mehreren Knoten ergibt sich somit zu $\sum_n t_{\mathrm{Knoten}}(n)$.

Auch für Sender und Empfänger kann man die Latenzanteile feingranular unterscheiden. Zunächst gibt es ebenfalls einen Anteil durch Netzwerkprotokollverarbeitung (beim Sender das Hinzufügen entsprechender *Header*, beim Empfänger invers). Auch die Pufferung in Eingangs- und Ausgangswarteschlangen vor der Verarbeitung bzw. vor dem Senden trifft auf Sender und Empfänger zu. Schließlich entsteht auch eine Latenz durch den Sendevorgang am Sender.

Für die Gesamtverzögerung eines reinen Übertragungsprozesses ergibt sich:

$$t_{\mathrm{ÜBERTRAGUNG}} = t_{\mathrm{Sender}} + \sum_n t_{\mathrm{Knoten}}(n) + t_{\mathrm{Empfänger}}$$

Aus diesem Zusammenhang können für eine Minimierung der Ende-zu-Ende-Verzögerung folgende Designrichtlinien für das Netzwerk abgeleitet und im Gesamtkonzept berücksichtigt werden (siehe Kapitel 6):

- kleine Anzahl von Netzwerkknoten (n) pro Übertragungspfad
- um Warteschlangen bzw. deren Einfluss (t_{zs}) zu minimieren, sollte die Datenrate des Netzwerkes angemessen dimensioniert werden und mit einer Reserve (Overhead) versehen werden
- eine hohe Datenrate mindert weiterhin den Einfluss der Verzögerung durch den Sendevorgang t_{sv}

3.3 Schichtenmodelle der Netzwerkkommunikation

Nachdem im letzten Abschnitt allgemeine Aussagen zu Telekommunikationsnetzwerken und deren Leistung getroffen worden sind, soll nun der Fokus auf konkrete Protokollimplementierungen gelenkt werden. Zur besseren Strukturierung werden zunächst gängige Schichtenmodelle vorgestellt. Gegliedert nach Schichten sollen dann korrespondierende Protokolle beschrieben und für die Anwendung im Studiobereich ausgewählt werden.

Aufgrund der speziellen Anforderungen im Broadcastbereich entwickelten sich zunächst vertikale Systeme, d.h. alle Komponenten stammten von einem Hersteller. Die Spezifikation und Implementierung von offenen Schnittstellen konnte so entfallen, die Hersteller lieferten komplette Systeme und waren für deren Funktionalität verantwortlich. Mit der

zunehmenden Integration von IT-Technologie und dem gewachsenen Wunsch der Anwender, Anwendungssysteme herstellerübergreifend zusammenzustellen, war die Definition von offenen Schnittstellen unabdingbar. Anhand von Schichtenmodellen kann man Komponentengrenzen deutlich darstellen. Daran orientierte Systeme nennt man horizontale Systeme.

Als Grundlage für die Entwicklung von Kommunikationsprotokollen wird das ISO-OSI[4]-Referenzmodell mit sieben definierten Schichten verwendet. Untere Schichten stellen darin übergeordneten Schichten über Schnittstellen Dienste zur Verfügung und schirmen dabei die Details der darunter liegenden Schichten ab. In jeder dieser Schichten implementieren Protokolle Funktionalitäten des Netzwerkes, z.B. Flusssteuerung, Adressierung, Routing und Fehlererkennung.

Die (1) Bitübertragungsschicht (*physical layer*) sorgt für die Übertragung von Bits über ein physikalisches Übertragungsmedium, indem sie mechanische, elektrische und zeitorientierte Schnittstellen definiert. Aufgabe der (2) Sicherungsschicht[5] (*data link layer*) ist die zuverlässige Übertragung des Bitstromes, der zu diesem Zweck in Frames unterteilt wird. Die (3) Vermittlungsschicht[6] (*network layer*) handhabt die Auswahl der Paketrouten[7] über verschiedene Knoten des Netzwerkes (auch: *Routing*) und ist damit auch für die Adressierung der Kommunikationsteilnehmer verantwortlich. Ein verbreitetes Protokoll der Schicht 3 ist das *Internet Protocol* (IP). In der (4) Transportschicht werden Nachrichten zwischen Sender und Empfänger transportiert und deren Zustellung sichergestellt. Bekannte Beispiele für dieser Schicht sind das verbindungsorientierte *Transmission Control Protocol* (TCP) und das verbindungslose *User Datagram Protocol* (UDP). Die Schichten 1–4 werden unter dem Begriff „Transportsystem“ zusammengefasst.

Die Schichten 4–7 haben anwendungsnahe Funktionalitäten und werden deshalb auch als Anwendungssystem bezeichnet. Die (5) Sitzungsschicht (*session layer*) ermöglicht den Auf- und Abbau von Sitzungen, die Dienste wie die Dialogsteuerung (wer darf gerade Daten übertragen), Token-Verwaltung und Synchronisation zur Verfügung stellen. Während die unteren Schichten im Wesentlichen Bits übertragen, hat die (6) Darstellungsschicht (*presentation layer*) die Syntax und Semantik der übertragenen Information zum Inhalt. Weitere Aufgaben dieser Schicht sind Codierung, Kompression und Verschlüsselung. Die (7) Anwendungsschicht *(application layer*) schliesslich stellt dem Anwender Protokolle zur Verfügung die z.B. zur Datenübertragung (z.B. *Hyper-Text Transfer Protocol* – HTTP und *File Transfer Protocol* – FTP) genutzt werden können. [Tan04, S. 54–58][PD07, S. 27–28][HK04b]

Die Nummerierung der Schichten bezieht sich im Folgenden ausschließlich auf das OSI-Modell. Ein weiteres Schichtenmodell ist des Internet-Modell, das nach seinen wichtigsten Protokollen auch TCP/IP-Modell genannt wird. Im Vergleich zum OSI-Modell besteht

[4] *Open Systems Interconnection Reference Model* der *International Organization for Standardization*
[5] auch Verbindungsschicht
[6] auch Sitzungsschicht
[7] in Schicht 3 werden die zu übertragenden Informationen Pakete genannt

es aus nur vier Schichten, wobei Schichten 1 und 2 zur Netzwerkschicht und Schichten 5 bis 7 zur Anwendungsschicht zusammengefasst werden. Bezeichnend ist, dass in OSI-Schicht 3, also der Internet-Schicht des TCP/IP-Modells nur IP definiert ist. Das Internet-Modell ist kein strenges Schichtenmodell. Einer Anwendung steht es frei, die definierten Transportschichten zu umgehen und IP oder die darunter liegenden Netze direkt zu nutzen (siehe Abbildung 3.6).

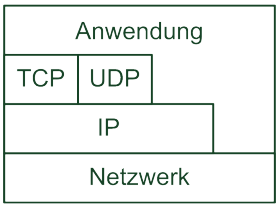


Abbildung 3.6: Alternative Betrachtung des Internetmodells [PD07, S. 28]

Vergleicht man das OSI- und das Internetmodell miteinander, ergeben sich folgende Unterschiede. Während das OSI-Modell vor der Entstehung entsprechender Protokolle entwickelt wurde, beschreibt das TCP/IP-Modell im Wesentlichen den im Internet gebräuchlichen Protokollstapel. Ohne praktische Implementierungserfahrungen entwickelt, beschreibt das OSI-Modell die Schichten abstrakt – was dazu geführt hat, dass z.B. Funktionen wie Adressierung, Flusskontrolle und Fehlerüberwachung mehrfach in den Schichten auftauchen [Tan04, S. 65]. Hingegen ist es mit dem TCP/IP-Modell schwierig bis unmöglich, TCP/IP-fremde Netze zu beschreiben.

Gemeinsame Überlegungen der EBU und der SMPTE, wie man IT-Standards zur Übertragung von Programmmaterial als Bitsröme über Netzwerke nutzen könne, hatten ein objektorientiertes Modell zur Folge (siehe Abbildung 2.3). Eine Achse dieses Modells reduziert die sieben Schichten des OSI-Modells auf vier Schichten, die für eine Anwendung im Broadcastbereich abgestimmt sind. Dabei bleiben die Schichten 1 und 2 identisch, die Schichten 3 bis 5 werden zur Netzwerkschicht und die Schichten 6 und 7 zur Anwendungsschicht zusammengefasst (siehe Abbildung 3.7).

Als Grundlage für diese Arbeit wird im Wesentlichen auf ein für Broadcastanwendungen angepasstes Schichtenmodell nach [HH04] zurückgegriffen (vergleiche Abbildung 3.7: „Studio Modell“), um die verwendeten Technologien voneinander abzugrenzen. Somit können die einzelnen Schichten zunächst losgelöst voneinander betrachtet und diskutiert werden, bevor auf dieser Grundlage Designentscheidungen gefällt werden. In diesem Sinne wird das weitere Kapitel wie folgt gegliedert: Der Netzwerkteil (OSI-Schichten 1 bis 3) hat grundlegende Netzwerktechnologien zur Datenübertragung zum Inhalt. Danach werden im Abschnitt „Netzwerkzugriff“ verschiedene Modi der Datenübertragung und entsprechende Protokolle diskutiert (OSI-Schichten 4 und 5). Auf dieser Grundlage aufbauend werden „Nutzdaten“ besprochen, welche neben verschiedenen Content- auch Containerformate umfassen (OSI-Schichten 6 und 7).

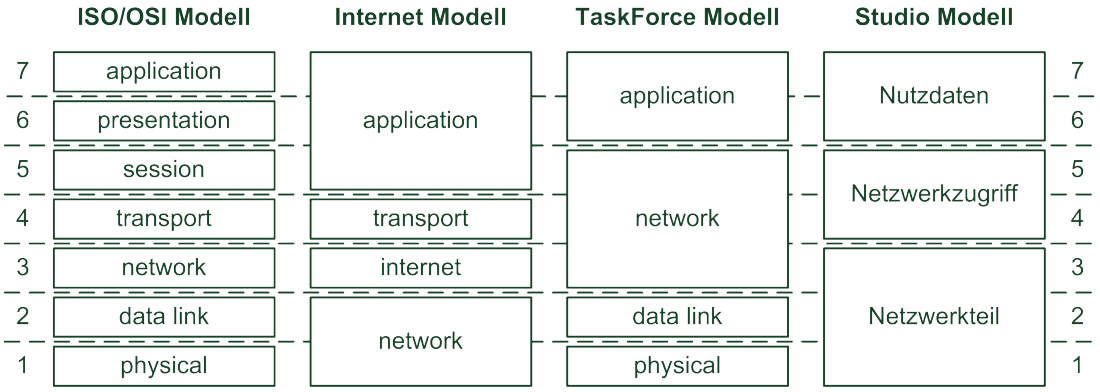


Abbildung 3.7: Vergleich verschiedener Schichtenmodelle zur Netzwerkkommunikation

3.3.1 Netzwerkteil

Der Netzwerkteil fasst die OSI-Schichten 1 bis 3 zusammen und gewährleistet so neben der Übertragung eines Bitstromes über ein physikalisches Medium auch die Adressierung der Teilnehmer und das Festlegen der Wegstrecke durch das Netzwerk (Routing). Beide Aspekte werden im Folgenden getrennt behandelt.

Zunächst werden die ausgewählten Netzwerktechnologien Asynchronous Transfer Mode (ATM) und Ethernet vorgestellt. Auf Basis von definierten Anforderungen für den Einsatz im Studiobereich erfolgt dann die Auswahl einer Netzwerktechnologie. Nach den Ausführungen zu den unteren Schichten erfolgt die detaillierte Betrachtung des IP-Protokolls als zentrales Element der Netzwerkschicht (OSI-Schicht 3).

3.3.1.1 Asynchronous Transfer Mode – ATM

Das grundlegende Konzept von ATM ist die Übertragung der Informationen in Form von Paketen (Zellen) mit einer festen Größe von 53 Byte, wovon 5 Byte für den Header und 48 Byte für die Nutzdaten verwendet werden. Aufgrund der festen Größe können die Zellen mithilfe von Hardware sehr schnell in Netzwerkknoten weitergeleitet werden (auch an mehrere Ausgangsports). Andere Netzwerktechnologien mit Paketen variabler Länge (z.B. Ethernet) müssen Pakete erst von einer Software analysieren lassen, was langsamer geschieht. ATM wurde ursprünglich für *Wide Area Networks* (WANs) konzeptioniert und unterstützte aufgrund seiner verbindungsorientierten Übertragung weder Broadcasting noch Multicasting. Diese Funktionalität wurde erst später mit einer Technik namens ATM-LAN-Emulation (LANE) aufwändig umgesetzt.

ATM-Netze arbeiten verbindungsorientiert, d.h. vor dem Senden von Daten muss zunächst ein Paket gesendet werden, das die Verbindung einrichtet (Signalisierung). Alle

auf dem Weg liegenden Netzwerkknoten verzeichnen das Vorhandensein der Verbindung[8] in ihren Tabellen und reservieren die benötigten Ressourcen. Der Header einer ATM-Zelle enthält eine Verbindungskennung, anhand der jeder Knoten die Zelle weiterleiten kann.

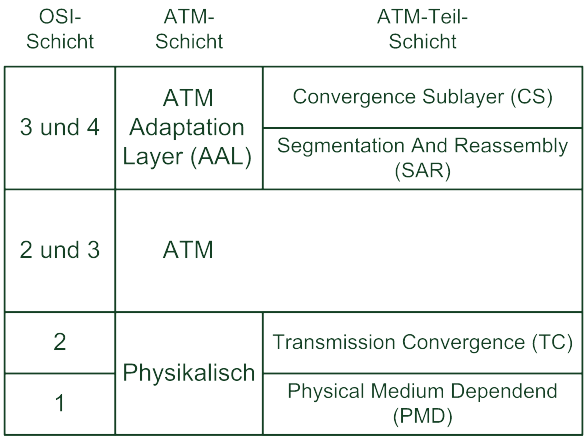


Abbildung 3.8: ATM-Schichten nach [Tan04, S. 83]

In Abbildung 3.8 ist das ATM-Referenzmodell abgebildet und dem OSI-Modell gegenübergestellt. Es besteht prinzipiell aus drei Schichten, die sich z.T. nicht direkt in das OSI-Modell überführen lassen. Die physikalische Schicht bezieht sich auf elektrische Parameter, wobei ATM-Zellen sowohl über Kupfer- und Glasfaserkabel übertragen werden können, als auch in den Nutzdaten anderer Trägersysteme verpackt werden können. Die ATM-Schicht definiert die Zellen und deren Transport inkl. Einrichtung und Freigabe virtueller Verbindungen. Weil die meisten Anwendungen nicht direkt mit Zellen arbeiten, stellt die ATM-Anpassungsschicht (auch *ATM Adaption Layer* – AAL) die Funktionalität zur Verfügung, größere Pakete zu fragmentieren (Segmentierung und Reassemblierung). [Tan04, S. 79–84], [PD07, S. 196–210]

AAL-1 unterstützt die Übermittlung von Audio- und Video-Livesignalen mit konstanter Bitrate (*constant bit rate* – CBR), die Einbettung von Zeitmarken, Möglichkeiten der Taktrückgewinnung und Vorwärts-Fehlerschutzmechanismen[9]. Für Daten ohne Synchronitätsanforderungen, wie z.B. File-Transfer über TCP/IP, steht AAL-5 zur Verfügung. [Ass04]

Anwendung im Rundfunkbereich findet ATM beim HYBNET (Hybrides Breitbandnetz), das auf Basis einer SDH-Ringstruktur mit einer Datenrate von 2,5 Gbit/s (STM-16) die Hauptstandorte der ARD-Rundfunkanstalten verbindet. Neben SDH-basierenden Verbindungen (im Wesentlichen für Fernsehprogrammverteilleitungen) wird ein Teil des HYBNETs für ATM-Anwendungen wie Audio- und Videofiletransfer, DVB-Zuführung und Intranet genutzt. [AAH03]

[8] auch *virtual circuit*, virtuelle Leitung
[9] *forward error correction* FEC, in diesem Falle ein Reed-Solomon-Code RS 128/124

3.3.1.2 Ethernet

1976 stellten Meltcalfe und Boggs das „Ethernet“ vor – ein paketbasiertes Netzwerk, dass eine Kommunikation zwischen Computern über ein Koaxial-Kabel erlaubt, indem Pakete variabler Länge ausgetauscht werden. Nach Verfeinerungen wurde Ethernet als IEEE[10] in den Standardfamilien 802.2 und 802.3 standardisiert und erreichte seitdem eine große Verbreitung. In den 1990er Jahren entwickelte sich Ethernet in drei Dimensionen weiter: höhere Übertragungsraten (10/100/1-Gigabit/10-Gigabit-Ethernet), weitere Übertragungsmedien (Kupferdraht – *twisted pair*, Glasfaserleitungen) und weitere Netzwerktopologien.

Im ursprünglichen Ethernet teilen sich alle Kommunikationsteilnehmer einen Kommunikationskanal (Bustopologie). Wenn ein Teilnehmer senden möchte, überprüft er zunächst, ob die Leitung bereits durch eine Kommunikation belegt ist (*Carrier Sense*). Falls die Leitung frei ist, beginnt der Teilnehmer mit dem Senden seiner Nachricht und prüft die Leitung auf Fehlerfreiheit (*Collision Detection*). Sollten Kollisionen auftreten (z.B. weil ein anderer Teilnehmer gleichzeitig versucht zu senden), wird der Sendevorgang abgebrochen und nach einer zufällig langen Zeitdauer erneut gestartet (*Carrier-Sense Multiple Access with Collision Detection* – CSMA/CD).

Abbildung 3.9: Aufbau eines Ethernetframes nach [EJK04]

Die Datenrahmen (Ethernet Frames) haben eine variable Länge (siehe Abbildung 3.9). Nach einer 56 Bit langen Präambel folgt ein 8-Bit-Muster, das den Beginn des Headers kennzeichnet (*Start of Frame Delimiter - SFD*). Danach folgen Ziel- und Absenderadresse mit je 48 Bit Länge. Diese MAC[11]-Adressen werden vom Hersteller des Netzwerkinterfaces bei der IEEE beantragt und für jede Karte individuell vergeben. Über bestimmte Zieladressen können Nachrichten auch an Gruppen gesendet werden (Broadcast und Multicast). Nach der Zieladresse kann optional ein 32 Bit VLAN-Tag eingefügt werden, um die Zugehörigkeit zu einem virtuellen LAN zu kennzeichnen[12]. Das folgende 16-Bit-Feld gibt die Länge des Frames in Bytes an (mind. 46 – bei Gigabit-Ethernet mind. 64 – bis maximal 1500 Nutzbytes[13]). Dieses Längenfeld kann spezielle Werte oberhalb von 1536 (0x0600) beinhalten. In diesem Fall wird es als Type-Feld erkannt und kennzeichnet Frames, deren Länge durch Mitzählen der Bits bestimmt wird. Diese Möglichkeit wird von IP-Paketen (0x0800) genutzt. Im Anschluss folgen die eigentlichen Nutzdaten – falls notwendig füllt das PAD-Feld den Datenbereich auf die Mindestgröße auf. Abschließend

[10] Institute of Electrical and Electronics Engineers
[11] *Media Access Control*
[12] VLANs werden zur Unterteilung von LANs genutzt
[13] so genannte Jumbo-Frames sehen eine Länge bis 9 KByte vor

Teilbereich	Übertragungsrate	Übertragungsmedium
10BASE5	10 Mbit/s	RG-8-Koaxialkabel 50 Ohm
10BASE2	10 Mbit/s	RG-58-Koaxialkabel 50 Ohm
10BROAD36	10 Mbit/s	Koaxialkabel 75 Ohm
10BASE-T	10 Mbit/s	Twisted-Pair-Kabel (TP)
10BASE-FL/-FB/-FP	10 Mbit/s	Lichtwellenleiter
100BASE-TX/-T2/-T4	100 Mbit/s	Twisted-Pair-Kabel (TP)
100BASE-FX	100 Mbit/s	Lichtwellenleiter
1000BASE-SX/-LX	1.000 Mbit/s	Lichtwellenleiter
1000BASE-CX	1.000 Mbit/s	Twinax-Kabel
1000BASE-T	1.000 Mbit/s	Twisted-Pair-Kabel (TP)
10GBASE-LX4/-ER/...	10.000 Mbit/s	Lichtwellenleiter
10GBASE-CX4	10.000 Mbit/s	IB4X-(Twinax)-Kabel
10GBASE-T	10.000 Mbit/s	Twisted-Pair-Kabel (TP)

Tabelle 3.1: Übertragungsraten und -medien im Ethernet-Standard nach [Rec08, S. 36]

wird ein 32-Bit-CRC[14] angefügt. [EJK04]

Im Ethernet-Standard werden für die verschiedenen Übertagungsmedien und den daraus resultierenden Datenkodierungen spezielle Teilbereiche vorgesehen. Tabelle 3.1 führt die wesentlichen Punkte zusammen.

Mit der Einführung des *Switched Ethernet* konnte die Stern-Topologie implementiert werden, die als zentralen Koppelpunkt einen *Switch* verwendet. Jeder Kommunikationsteilnehmer hat damit eine exklusive 1-zu-1-Verbindung zum *Switch*, weshalb Kollisionen nicht mehr auftreten können und das CSMA/CD-Verfahren nicht mehr angewendet werden muss. Dennoch kann im Ethernet keine sichere Übertragung im Sinne garantierter Fehlerraten, Verzögerungszeiten und Jitter-Werte realisiert werden. Einerseits können Pakete in Überlastsituationen verworfen werden (z.B. bei Puffer-Überlauf). Andererseits können Pakete eines Streams verschiedene Wege (Routen) durch das Netzwerk wählen.

3.3.1.3 Auswahl einer Netzwerktechnologie

ATM bietet über AAL-1 die garantierte und isochrone Übertragung von Datenströmen wie Video und Audio mit hohen Übertragungsraten sowie geringen Latenzen und bietet damit eine Dienstegüte, die eine Anwendung im Studiobereich möglich macht. Dennoch sind folgende Nachteile der ATM-Technologie zu beachten: Zunächst ist ATM eine für den Weitverkehrsbereich (WAN) definierte Technologie. Eine LAN-Unterstützung, die

[14] Cyclic Redundancy Check

Broadcasting und Multicasting bietet, muss erst aufwändig implementiert werden. Weiterhin hat ATM nur eine geringfügige Verbreitung gefunden, was sich in kostenintensiver Gerätetechnologie ausdrückt.

Ethernet ist als LAN-Technologie entwickelt worden und erfreut sich begünstigt durch das rasche Wachstum des Internets hoher Verbreitung. Dies hat positive Auswirkung auf die Kosten für Installation, Betrieb und Wartung von Ethernet-Netzwerken. Dienstegüte kann mit Ethernet dabei unter Nutzung von QoS-Mechanismen bzw. unter bestimmten Voraussetzungen (Vermeidung von Überlastsituationen, Einschränkung der Komplexität des Netzwerkaufbaus) erreicht werden.

ATM und Ethernet können für die Übertragung und Verteilung von hochratigen Datenströmen mit geringen Latenzen angepasst werden. Mit Blick auf zukünftige technologische Weiterentwicklungen (100-Gigabit-Ethernet) und Kosten für Implementierung, Betrieb und Wartung wird in dieser Arbeit Ethernet als Übertragungstechnologie gewählt. Die Anforderung besteht im Folgenden daraus, geeignete Technologien in höheren Schichten auszuwählen und für geeignete Bedingungen zu sorgen, damit die geforderten Parameter für die Anwendung im echtzeitkritischen Studiobereich erfüllt werden.

3.3.1.4 Internet Protocol – IP

Protokolle der Vermittlungsschicht (OSI-Schicht 3) haben die Aufgabe, Pakete von einer Quelle zum Ziel zu übertragen. Die Topologie des Netzwerkes muss über LAN-Grenzen hinaus bekannt sein, um geeignete Pfade zu wählen. Zentrale Funktionen von Layer-3-Protokollen sind demnach die Adressierung von Hosts und Knoten sowie die Bestimmung des Paket-Pfades durch das Netzwerk (*Routing*). Beide Aspekte sollen im Folgenden kurz am Beispiel des durch die große Verbreitung des Internet dominierenden *Internet Protocols* (IP) auf Basis von [PD07, Tan04] verdeutlicht werden.

Um Daten zwischen beliebigen Hosts in beliebigen miteinander verbundenen Netzwerken auszutauschen, ist ein **Adressierungsschema** notwendig, das globale Eindeutigkeit und eine hierarchische Struktur (die den Aufbau des Internetworks repräsentiert)[15] vereint. Um beide Anforderungen zu erfüllen, bestehen IP-Adressen aus einem Netzwerk- und einem Hostteil. Der Netzwerkteil kennzeichnet das Netzwerk, an dem der Host angeschlossen ist. Der Hostteil identifiziert den Host in diesem Netz eindeutig. Ein Netzwerkknoten, der an zwei Netzwerke angeschlossen ist (*Router*), verfügt dementsprechend über zwei IP-Adressen. Insofern kann eine IP-Adresse eher dem Interface als dem Host bzw. Knoten zugeordnet werden.

Wie in Abbildung 3.10 ersichtlich, werden IP-Adressen in drei Klassen eingeteilt, die sich jeweils durch unterschiedlich große Netz- und Hostteile differenzieren und damit einen hierarchischen Aufbau ermöglichen (Klasse A–C). In allen Fällen ist die Adresse 32 Bit

[15] Ethernet-Adressen sind global eindeutig, aber „flach“. Sie bieten keine Hinweise für Routing-Protokolle und sind insofern nicht als Adressierungsschema für Layer-3-Protokolle geeignet.

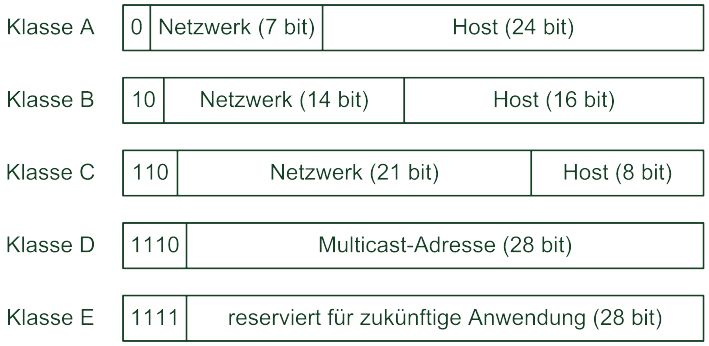


Abbildung 3.10: Klassifizierung und Aufbau von IPv4-Adressen nach [Tan04, S. 480]

lang. Die Klasse wird an den werthöchsten Bits identifiziert. Ursprünglich waren in diesem „klassenbehafteten“ Ansatz die Klasse-A für WANs, die Klasse-B für Campusnetze und die Klasse-C für LANs bestimmt, was sich auch in den jeweils möglichen Hostzahlen pro Netzwerk ausdrückt (Klasse A: 2^{24} = ca. $16 \cdot 10^9$, Klasse B: $2^{16} = 65.534$, Klasse C: $2^8 = 254$)[16]. IP-Adressen werden als vier durch Punkte getrennte Dezimalzahlen notiert (z.B. 192.168.178.21). Ein Teil der möglichen IP-Adressen ist für Multicastgruppen (Adressbereich 224.0.0.0 bis 239.255.255.255) und für zukünftige Anwendungen (Adressbereich 240.0.0.0 bis 255.255.255.255) reserviert.

Mithilfe des Adressierungsschemas können IP-Pakete in Netzwerkknoten weitergeleitet werden, d.h. auf Basis einer Weiterleitungstabelle korrespondierenden Ausgangsports zugeordnet werden. Zur Erstellung dieser Weiterleitungstabelle kommen z.T. komplexe verteilte Algorithmen zum Einsatz (*Routing*). Verschiedene Algorithmen (z.B. Distanzvektor-Routing, Link-State-Routing) und entsprechende Protokollimplementierungen (*Routing Information Protocol* – RIP, *Open Shortest Path First Protocol* – OSPF) werden dazu angewendet.

In Abbildung 3.11 wird der Aufbau eines IPv4-Headers deutlich. Nach der Kennzeichnung der Protokollversion (Feld „Version“) erfolgt die Angabe der Headerlänge („IHL“), da diese bei IPv4 variabel (min. 20 Byte, max. 40 Byte) ist. Das Feld „Diensttyp“ (*Type of Service* – TOS) wird zur Unterscheidung verschiedener Diensteklassen verwendet. Die Gesamtlänge bezieht sich auf das komplette Datagramm – also Header und Daten – und hat bei IPv4 einen Maximalwert von 65.535 (Bytes). In Abhängigkeit der *Maximum Transmission Unit* (MTU) des Basisnetzwerkes (bei Gigabit-Ethernet normalerweise 1.500 Byte) müssen IP-Datagramme fragmentiert werden. Die Felder „Identifikation“, „Flags“ und „Fragment-Offset“ enthalten dafür notwendige Informationen. Das Feld „Lebensspanne“ (*time to live* – TTL) verhindert Ressourcenverschwendung durch Routing-Schleifen. Das Feld „Protokoll“ identifiziert das im Protokollstapel höher gelegene Protokoll (z.B. TCP oder UDP). Anhand der Header-Prüfsumme kann mit einem einfachen Algorithmus

[16] reservierte bzw. ungültige Adressen bereits abgezogen

(Einer-Komplement-Arithmetik) verhindert werden, dass Datagramme mit fehlerhafter Header-Information weitergeleitet werden. Nach den Quell- und Zieladressen sieht IPv4 Platz für optionale Erweiterungen vor, die in Praxis jedoch selten Anwendung finden.

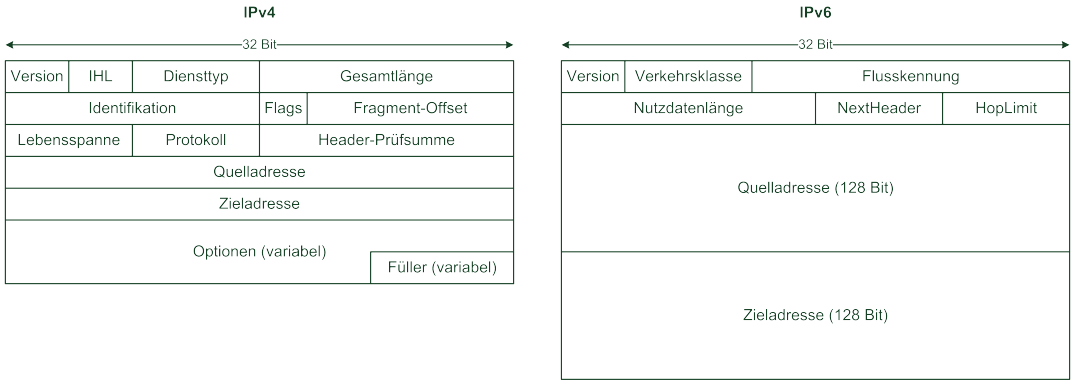


Abbildung 3.11: Headervergleich IPv4 und IPv6 nach [PD07, Tan04]

Trotz der Vielzahl an Erweiterungen ist ein IPv6-Header im Vergleich zu IPv4 einfacher aufgebaut. IPv6 sieht Header mit konstanter Länge von 40 Byte vor und wichtige Felder sind 64-bit-aligned, was den Zugriff in den Netzwerkknoten beschleunigt. Das Feld „Next-Header“ ersetzt das Feld „Protokoll“ und die Optionen bei IPv4, indem es auf eventuelle Zwischenheader nach dem IP-Header verweist (z.B. für Fragmentierung genutzt). Auch die Berechnung der Prüfsumme wird bei IPv6 entweder anderen Protokollen überlassen (z.B. TCP) oder in einen Zwischenheader ausgelagert. IPv6 vergrößert den Adressraum, indem Quell- und Zieladresse eine Größe von 128 Bit aufweisen. Obwohl die Adressinformation damit viermal länger als bei IPv4 ist, ist die Gesamtheadergröße bei IPv6 maximal doppelt so groß.

IPv6 weist in vielen Bereichen Vorteile gegenüber IPv4 auf (z.B. Autokonfiguration, Multicast, IPsec, Quality of Service, Renumbering). Bis auf den größeren Adressraum und Renumbering wurden aber alle Merkmale nach IPv4 portiert, so dass die Motivation zum Umstieg auf IPv6 eher gering ist.

IP-Multicast ermöglicht es einem Sender, Pakete effizient an mehrere Empfänger gleichzeitig zu übertragen. Dazu werden mehrere Empfänger unter einer Multicast-Adresse (siehe Abbildung 3.10) zusammengefasst. Die Pakete des Senders werden dann im zentralen Netzwerkknoten (Switch) kopiert und an alle Empfänger der Multicastgruppe weitergeleitet. Die Verwaltung der Multicastgruppe, also das Hinzufügen und Löschen von Empfängern, erfolgt zentral auf dem Switch durch Protokolle wie dem *Internet Group Management Protocol* (IGMP). IGMP baut wiederum auf IP auf, d.h. IGMP-Pakete werden in IP-Datagrammen gekapselt. Anmeldung und Beendigung der Mitgliedschaft werden durch den Empfänger mit IGMP-Paketen initiiert. Sowohl das Kopieren der Pakete im Switch als auch die Verwaltung der Multicastgruppe haben eine Latenz zur Folge (siehe Abschnitt 6.4).

In dieser Arbeit soll IP als Grundlage des Konzeptes verwendet werden, weil damit eine flexible und erprobte Lösung für die Aufgabenstellungen Adressierung und Routing zur Verfügung steht. Eine Alternative vor dem Hintergrund der Erweiterung des Ansatzes auf ein Internetwork für Studioanwendungen kann aus Sicht des Autors nicht identifiziert werden.

3.3.2 Netzwerkzugriff

In diesem Abschnitt werden hauptsächlich Transportprotokolle vorgestellt, die die Aufgabe haben, Daten unabhängig vom verwendeten physikalischen Netzwerk zuverlässig und kosteneffizient zwischen Prozessen auf verschiedenen Hosts zu übertragen. Durch das Internet haben sich im Wesentlichen zwei Protokolltypen etabliert, die verbindungslos (UDP) und verbindungsorientiert (TCP) agieren.

Das *User Datagram Protocol* (UDP) unterstützt die Übertragung von Datagrammen, ohne eine Verbindung aufzubauen. Der einfach gestaltete UDP-Header enthält Quell- und Zielportnummern, die den darüber liegenden Dienst identifizieren. Zusätzlich wird eine Längenangabe sowie eine Prüfsumme übertragen. UDP ermöglicht aufgrund seines einfachen Aufbaus eine latenzarme Verarbeitung in den Hosts, unterstützt aber keine Fehlerkorrektur. Diese muss im Bedarfsfall auf höheren Schichten implementiert werden, indem z.B. zusätzliche Redundanz im Paketstrom erzeugt wird und somit eine Korrektur defekter oder verlorener Pakete durch den Empfänger möglich wird (FEC). Wegen seines verbindungslosen Charakters ist UDP multicastfähig.

Die Prüfsumme stellt die Korrektheit der Nachricht sicher. Sollte bei der Verifizierung der Prüfsumme ein Fehler festgestellt werden, wird das Paket verworfen. Für Multimedia-Anwendungen kann es sinnvoll sein, das fehlerhafte Paket bei der Dekodierung zu verwenden und nicht auf die korrekt übertragenen Anteile zu verzichten. Diese Anforderung greift das *Lightweight User Datagram Protocol* (UDP lite) auf. Dabei wird die Prüfsumme nur über den Header und nicht über die Nutzdaten gebildet. Ein Paket wird daher nur verworfen, falls ein Fehler im Header auftritt und es so nicht mehr eindeutig zugeordnet werden kann.[17]

Das im Multimediabereich genutzte *Real-time Transport Protocol* (RTP) ist ein in der Anwendungsschicht implementiertes Transportprotokoll, das in der Regel auf UDP aufsetzt. Die Grundfunktion von RTP ist das Multiplexen verschiedener Echtzeit-Datenströme in einen einzigen UDP-Datenstrom. Dazu wird im RTP-Header eine Sequenznummer und ein Zeitstempel (relativ zum Beginn des Datenstromes) vermerkt. Das Zeitstempelverfahren reduziert nicht nur die Auswirkungen von Jitter, sondern erlaubt auch die

[17] Linux unterstützt UDP lite ab Kernel 2.6.20, unter Windows existiert derzeit noch keine von Microsoft dokumentierte Möglichkeit, UDP lite in Verbindung mit Windows Sockets zu nutzen [Ote08]

Synchronisation mehrerer Datenströme. RTP verfügt weder über Fluss- und Fehlerkontrolle, noch über Mechanismen zur Bestätigung fehlerfreier oder Anforderung erneuter Übertragung.

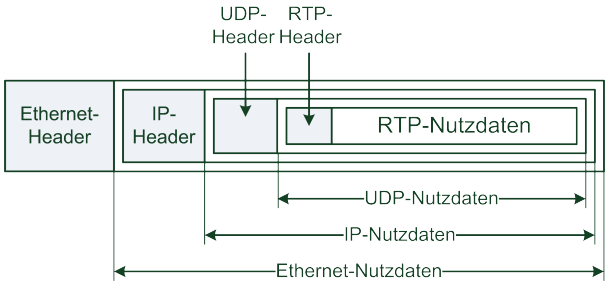


Abbildung 3.12: RTP-Paketverschachtelung nach [Tan04]

Während UDP ein einfaches aber schnelles Transportprotokoll z.B. zur Übertragung von Multimedia-Datenströmen zur Verfügung stellt, wird für viele Anwendungen eine zuverlässige Zustellung gefordert. Diese Aufgabe übernimmt das *Transmission Control Protocol* (TCP), das als Standard-Transportprotokoll des Internets eine große Verbreitung erfahren hat.

TCP ist ein verbindungsorientiertes Protokoll und stellt so Funktionen zum sicheren Auf- und Abbau sowie zur Überwachung einer Verbindung bereit. TCP umfasst einen Mechanismus zur Flusskontrolle, der es einem Empfänger ermöglicht, die vom Sender gesendete Datenmenge zu begrenzen. Weiterhin implementiert TCP einen Mechanismus zur Überlastkontrolle, der einen Sender davon abhält, das Netzwerk zu überfluten. Der TCP-Verbindungsaufbau wird durch ein Handshake-Verfahren realisiert, bei dem Sender und Empfänger Sequenznummern zu sendender Pakete austauschen. Verlorene oder in der Reihenfolge vertauschte Pakete können so erkannt und gegebenenfalls neu angefordert werden. Die Multicast-Übertragung eines TCP-Datenstromes ist wegen der notwendigen Bestätigungspakete des Empfängers nicht möglich.

Neben TCP und UDP wurden weitere Transportprotokolle entwickelt. Das *Datagram Congestion Control Protocol* (DCCP) beispielsweise implementiert eine unzuverlässige Übertragung wie bei UDP in Verbindung mit einem TCP-ähnlichen Überlast-Mechanismus [ATF+07]. Das *Reliable WAN Transfer Protocol* (RWTP) implementiert einen ratenbasierten Bestätigungsmechanismus, um eine konstant hohe Datenrate trotz großer Entfernungen zwischen den Hosts zu ermöglichen [Sie08].

Die verbindungsorientierte Übertragung via TCP weist für hochratige Datenströme, die mit kleinen Latenzen übertragen werden sollen, eine Reihe von Nachteilen auf. Zum einen verletzt die Verbindungsorientierung und die damit einhergehende Neuanforderung bzw. -sendung von verlorenen bzw. defekten Paketen die Echtzeitbedingung. Zum anderen können aufgrund des Überlastmechanismus keine konstant hohen Datenraten übertragen

werden, sobald die Kapazität des Übertragungskanals fast vollständig ausgenutzt wird. Weiterhin wird die Multicastübertragung prinzipbedingt ausgeschlossen.

Für die Anwendung im Studiobereich ist deshalb ein verbindungsloses Protokoll wie UDP (lite) ggf. in Verbindung mit RTP zu präferieren.

3.3.3 Nutzdaten

Innerhalb des Produktionsprozesses muss Content, also Essenz- und Metadaten, zwischen einzelnen Geräten bzw. Funktionsgruppen ausgetauscht werden. Für den dateibasierten Austausch wurden diverse Formate entwickelt, die im Folgenden kurz miteinander verglichen werden sollen.

Das Digital Picture Exchange Format (**DPX**) wurde während der Zeit des Wechsels von der optischen zur elektronischen Effektgestaltung in der Filmproduktion (Digital Intermediate) entwickelt. DPX beschreibt ein Format zum Transport unkomprimierter Mediendaten insbesondere für Filmabtastung und Rendering; komprimierte Formate sind nicht vorgesehen. Die Übertragung via DPX erfolgt dabei in Form von pixelbasierten Einzelbildern und ist auflösungsunabhängig. Seit 1994 ist DPX bei der Society of Motion Picture Engineers standardisiert (SMPTE S268M) [Ban04].

Die Streaming-Definition der ehemaligen Grass Valley Group (Thomson) für Programmmaterialaustausch zwischen *Profile*-Servern über FibreChannel oder Ethernet heisst General Exchange Format (**GXF**) und ist seit April 2001 als SMPTE S360M standardisiert. GXF ist für die „einfache“ Übertragung von fertig produziertem Material (OnAir-, Archiv-Bereich) gedacht, dementsprechend sind die speicherbaren Effektinformationen auf Video-Hartschnitte und Audio-Fades eingeschränkt. GXF greift nicht direkt auf die Key-Lenght-Value-Kodierung (KLV) zurück, erlaubt aber eine gekapselte Übertragung von KLV-Daten, sowie XML- und Metadaten im Userbereich [Pae01].

Das Material Exchange Format (**MXF**) ist ein ebenfalls bei der SMPTE standardisiertes Containerformat, mit dem nahezu beliebige Essenzen und Metadaten transportiert bzw. gespeichert werden können. MXF ist zum Austausch von (fast) fertigem Programmmaterial konzipiert worden und unterstützt neben Filetransfer auch Streaming. Der logische Aufbau von MXF wird von einem objektorientierten Datenmodell beschrieben. Durch eine *Zero Divergence Doctrine* wird sichergestellt, dass MXF kompatibel zu AAF, dem Advanced Authoring Format, ist.

Das Advanced Authoring Format (**AAF**) erlaubt im Vergleich zu MXF eine komplexere Beschreibung der Essenz (Übergänge usw.) und wird deshalb als Datenformat im Postproduktionsbereich angewandt. Vor dem Materialzugriff muss allerdings die gesamte Übertragung abgeschlossen sein – Streaming ist nicht möglich. AAF ist kein offizieller

	DPX	GXF	MXF	AAF
Standard	SMPTE S268M	SMPTE S360M	SMPTE S377M u.a.	Quasistandard
Anwendung	Übertragung von Mediendaten im Bereich Filmabtastung und Rendering	Übertragung fertiger Programmteile	Übertragung fertiger Programmteile	Postproduktion
(Video)-Essenz	unkomprimiert, ein Einzelbild pro Datei	komprimiert (MPEG-ES, DVCPRO), nur harte Schnitte	unkomprimiert und komprimiert (über Generic Container offen für alle Kompressionsformate), nur harte Schnitte	unkomprimiert und komprimiert, Quellmaterial + Bearbeitungsinformation (Übergänge möglich)
Referenzen	nein	intern	intern + extern	intern + extern
Metadaten	begrenzt im Userdatenbereich	im Userdatenbereich	unbeschränkte Komplexität	Beschreibung komplexer Editing-Funktionen
Streaming	nein	ja	ja	nein, zu komplex

Tabelle 3.2: Vergleich ausgewählter Austauschformate

Standard sondern baut auf dem OMFI[18] (Open Media Framework Interchange) Format auf. AAF hat sich als de facto-Standard im Postproduktionsbereich etabliert.

Bei der Auswahl eines Austauschformates für die Verwendung im echtzeitkritischen Bereich der Studioproduktion ist ein Streamingmechanismus zwingende Voraussetzung. Eine gepaarte Datei- und Streaming-Funktionalität ermöglicht einen reibungslosen Übergang von den Echtzeitanforderungen der Live-Produktion zur filebasierten Nachbearbeitung und Archivierung. Weiterhin sollte die Art und Komplexität der einzubindenden Metadaten zunächst nicht eingeschränkt sein. Um die Qualitätsansprüche im Fernsehstudio zu erfüllen, soll weiterhin der Transport von unkomprimierten Video- und Audioessenzen ermöglicht werden. MXF erfüllt diese Anforderungen und wird von Herstellern sowie Anwendern zunehmend anerkannt und angewendet (siehe Tabelle 3.2).

[18] proprietäres Format der Fa. Avid

3.4 Konklusion

Mit dem Ziel der Anwendung von offenen Standard-Netzwerktechnologien stellt dieses Kapitel zunächst Leitungs- und Paketvermittlung gegenüber. Aufgrund der effizienten Netzwerkauslastung wird die Übertragung auf Basis von Datagrammen im Studiobereich bevorzugt. Weiterhin ist damit die Hoffnung auf eine breite Nutzung von Standard-IT-Komponenten zur Bearbeitung und Übertragung von hochratigen Essenzströmen im echtzeitkritischen Bereich der Fernsehstudioproduktion verbunden. Ermutigend wirkt dabei das rasante Wachstum an Prozessor- und Netzwerkleistung durch die immer noch stetige Ausbreitung des Internets.

Ein Netzwerk im Studiobereich besteht aus einer Reihe von Workstations, also Geräten zur Erzeugung, Bearbeitung, Anzeige und Speicherung von Daten und weist Eigenschaften auf, die im Kontext der Dimensionierung und Protokollauswahl berücksichtigt werden können:

1. Die Anzahl der Workstations ist endlich, bekannt und annähernd konstant. Ein Studionetzwerk ist aus Sicherheits- und Datenschutzgründen nicht Teil eines offenen Internetworks.
2. Anforderungen und Eigenschaften der Workstations sind bekannt. Dazu zählen insbesondere Anzahl und zeitliche Verteilung von Datenströmen sowie deren maximale Datenraten.
3. Die Komplexität des Netzwerkes ist begrenzt. Eine Stern-Topologie mit einem zentralen Netzwerkknoten (Switch), an den alle Workstations angeschlossen sind, resultiert in einem einfachen Aufbau[19].

In Bezug auf die im Abschnitt 2.1 gestellten Anforderungen wurde gezeigt, dass eine effektive Signalverteilung in Netzwerken von einer Quelle auf mehrere Senken, wie sie im Studio durch Kreuzschienen und Steckfelder realisiert wird, über Multicastübertragungen realisiert werden muss. Weiterhin wurde festgestellt, dass die Stern- und die verwandte Baum-Topologie sinnvoll nutzbare Topologien für solche Netzwerke darstellen. Dabei müssen die zentralen Elemente leistungsstark ausgeprägt sein, damit alle für einen Anwendungsfall notwendigen Teilnehmer gleichzeitig und störungsfrei miteinander kommunizieren können.

Um die Anforderungen des Zielsystems sinnvoll in Netzwerktechnologien abzubilden, wurden Parameter zur Bewertung der Leistungsfähigkeit von Netzwerken diskutiert. Als für Echtzeitanwendungen entscheidende Parameter wurden die Latenz der Übertragung und die Varianz der Latenz (Jitter) identifiziert. Anhand eines Latenzmodells können nun Designrichtlinien für eine Minimierung der Ende-zu-Ende-Verzögerung abgeleitet werden.

[19] Die Ausfallsicherheit des Systems in Hinblick auf Havarieszenarien und daraus resultierenden Redundanzsystemen bleiben im Basiskonzept zunächst nicht berücksichtigt.

In Bezug auf die Anwendung von QoS-Mechanismen wird folgende These aufgestellt: In relativ statischen lokalen Netzwerken mit bekannter Anzahl von Teilnehmern und definierten Anforderungen der Teilnehmer muss in datagrammvermittelnden Netzwerken kein selbstgesteuerter QoS-Mechanismus (IntServ/DiffServ) etabliert werden, um eine zuverlässige und sichere Übertragung von hochratigen Datenströmen zu ermöglichen. Als Alternative wird die Anwendung von Network-Engineering-Methoden propagiert, die die geeignete Dimensionierung des Netzwerkes vorsehen.

Im Sinne einer strukturierten Abhandlung wurden verschiedene Netzwerk-Schichtenmodelle analysiert und eine dreiteilige Struktur vorgestellt, die für die Anwendung im Broadcastbereich geeignet ist. Im Netzwerkteil (1) wurden zunächst die Basis-Netzwerktechnologien ATM und Ethernet miteinander verglichen. Mit Blick auf zukünftige technologische Weiterentwicklungen (100-Gigabit-Ethernet) und Kosten für Implementierung, Betrieb und Wartung wird in dieser Arbeit Ethernet als Übertragungstechnologie gewählt. Die Anforderung besteht im Folgenden daraus, geeignete Technologien in höheren Schichten auszuwählen und für geeignete Bedingungen zu sorgen, damit die geforderten Parameter für die Anwendung im echtzeitkritischen Studiobereich erfüllt werden. In dieser Arbeit wird aufbauend auf Ethernet das Internet Protokoll (IP) als zentraler Bestandteil des Konzeptes verwendet, weil damit eine flexible und erprobte Lösung für die Aufgabenstellungen Adressierung und Routing zur Verfügung steht. Eine Alternative zu IP vor dem Hintergrund der Erweiterung des Ansatzes auf ein Internetwork für Studioanwendungen konnte nicht identifiziert werden.

Im Netzwerkzugriffsteil (2) kristallisierten sich die Vorteile von verbindingsloser gegenüber verbindungsorientierter Übertragung heraus, so dass für die Anwendung im Studiobereich ein verbindungsloses Protokoll wie UDP ggf. in Verbindung mit RTP zu präferieren ist.

Bei der Auswahl eines Austauschformates im Nutzdatenteil (3) wurde das Material Exchange Format (MXF) ausgewählt, weil es die Kernanforderungen Streaming-Fähigkeit, Metadatenkomplexität und Synchronisation von beliebigen Datenströmen erfüllt. Die Anforderung besteht nun darin, den MXF-„Streaming“-Mechanismus so zu implementieren, dass die im Studiobereich gestellten Anforderungen erfüllt werden können.

4 Datenverarbeitung auf PC-Plattformen

Elektronische Datenverarbeitung (EDV) bezeichnet allgemein jeden Prozess, bei dem aus Informationen (Eingangsdaten) durch automatisierte Verarbeitung in einem Computer andere Informationen (Ausgangsdaten) gewonnen werden. Unter „Verarbeitung“ versteht man dabei eine systematische Folge von Operationen, wie das Erfassen, Aufbereiten, Codieren, Speichern, Vergleichen, Zuordnen, Sortieren, Berechnen, Übertragen und Ausgeben. Unter dem Begriff „Bearbeitung“ werden i.A. Prozesse zur Manipulation von Daten eines Essenztyps verstanden (z.B. Bild-, Audio- und Videobearbeitung).

Essenz-Signalbearbeitung im Studio erfolgt aktuell zum großen Teil auf Basis speziell entwickelter Geräte, die durch aufwändige Entwicklung und geringe Umsatzzahlen vergleichsweise kostenintensiv sind. Mit deren steigender Leistungsfähigkeit können zunehmend Funktionen von preiswerter Standard-PC-Hardware übernommen werden. In diesem Kapitel sollen verschiedene Hardwareplattformen bezüglich der Eignung für die Bearbeitung im Studio anfallender Signale dargestellt und verglichen werden. Zunächst erfolgt eine Klassifikation von Geräten zur Audio- und Videosignalbearbeitung.

4.1 Klassifikation von Medienbearbeitungsgeräten

Kovalick schlägt zur Kategorisierung von A/V-Geräten vier Klassen vor, die die Schnittstellen der Geräte zur Erzeugung, Bearbeitung und Speicherung von Audio- und Videosignalen bzw. Daten in den unterschiedlichen Epochen der Studioproduktion widerspiegeln (siehe Abbildung 4.1). Es handelt sich hierbei um eine allgemeine Darstellung. Spezialfälle wie Kameras, die keinen Eingang aufweisen, können als solche einer jeweiligen Klasse untergeordnet werden.

Geräte der Klasse 1 besitzen analoge Ein- bzw. Ausgänge für Audio- und Videosignale und verfügen nicht über einen IT-Netzwerkanschluss. Diese Klasse fasst die Geräte der analogen Studiotechnik zusammen und hat in modernen Studiokomplexen keine Bedeutung mehr. Bei Geräten der Klasse 2 werden die analogen durch digitale Signalschnittstellen ersetzt (SDI für Video, AES/EBU für Audio). Zusätzlich steht an Klasse-2-Geräten eine IT-Netzwerkschnittstelle für Dateitransfer zur Verfügung. Im Unterschied zur Klasse 2 ermöglicht Klasse 3 auch Echtzeit-Datentransfer über Netzwerkschnittstellen wie Fibre Channel oder Ethernet. Klasse-4-Geräte schliesslich können als Klasse-3-Geräte ohne Audio- und Videoschnittstellen betrachtet werden.

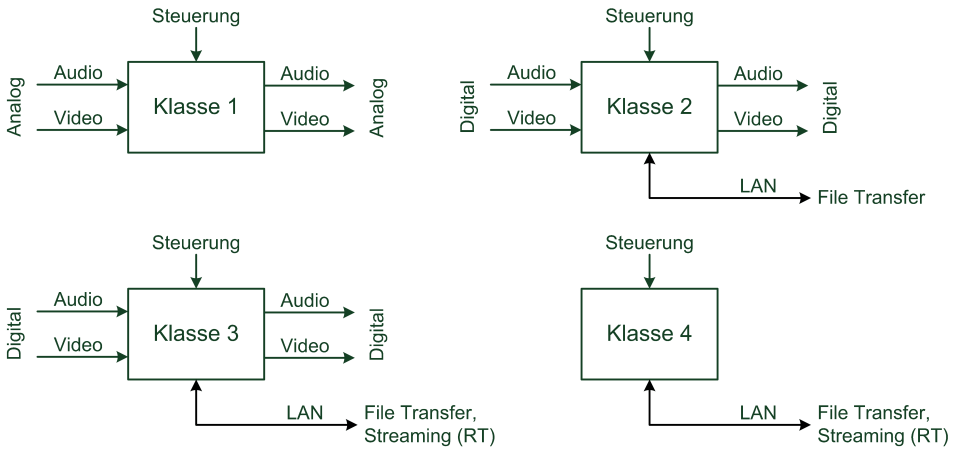


Abbildung 4.1: Klassifikation von A/V-Geräten nach [Kov06, S. 50 ff.]

4.2 Hardwareplattformen zur Medienbearbeitung

Die Berechnung von Ausgangs- aus Eingangswerten im Sinne einer elektronischen Datenverarbeitung erfolgt mithilfe von Hochleistungs-Schaltkreisen, auch Prozessoren, die dabei sowohl Steuer- als auch Rechenaufgaben übernehmen. Im Vergleich zu festverdrahteten Schaltkreisen (ASIC – *Application-Specific Integrated Circuit*), die auf Funktionalität und spezielle Eigenschaften (z.B. auf geringen Stromverbrauch) getrimmt werden können, erlauben programmierbare Prozessoren eine flexible Erweiterbarkeit der Funktionalität.

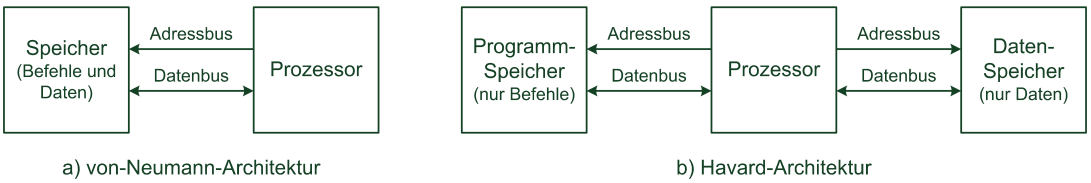


Abbildung 4.2: Prozessorarchitekturen nach [Smi99, S. 511]

Zwei wesentliche Prozessorarchitekturen können unterschieden werden (vergleiche Abbildung 4.2). Bei der von-Neumann-Architektur verbindet ein Bussystem den Prozessor mit dem Hauptspeicher, der sowohl Befehle (Programm) als auch Daten beherbergt, und den Ein- und Ausgabeschnittstellen, an denen weitere Komponenten angeschlossen sein können. Damit sind mindestens zwei Taktzyklen erforderlich, bevor ein Befehl ausgeführt werden kann. Die Havard-Architektur beschreibt die Ablage der Daten und Befehle in physikalisch getrennten Speichern. Der Zugriff erfolgt über getrennte Busse, sodass ein Befehl und die zugehörigen Daten innerhalb eines Taktes transferiert werden können.

4.2.1 Mikroprozessoren (CPU)

Mit der Entwicklung von programmierbaren Logikbausteinen auf einem Chip begann Anfang der 1970er Jahre die Ära der Mikroprozessoren, die durch ständige Weiterentwicklung in der 1980er Jahren den Aufbau von *Personal Computern* (PC) und deren große Verbreitung ermöglichte. In vielen elektronischen Geräten bildet der Mikroprozessor die Grundlage zur Steuerung komplexer elektronischer Systeme. Der programmierbare Mikroprozessor dient als zentrale Datenverarbeitungseinheit innerhalb der PC-Architektur und wird als *Central Processing Unit* (CPU) bezeichnet. CPUs verwalten oft die Hardwareplattformen, auf denen andere Prozessortypen implementiert sind.

Eine CPU besteht aus den zwei grundlegenden Bestandteilen Rechenwerk (*arithmetic logic unit* – ALU) und Steuerwerk (*control unit* – CU) und ist nach der von-Neumann-Architektur aufgebaut. Das Steuerwerk ruft Befehle aus dem Hauptspeicher ab und bestimmt den Befehlstyp. Das Rechenwerk führt für den jeweiligen Befehl notwendige Operationen (Maschinenbefehle) aus, z.B. die Addition oder Subtraktion zweier Werte.

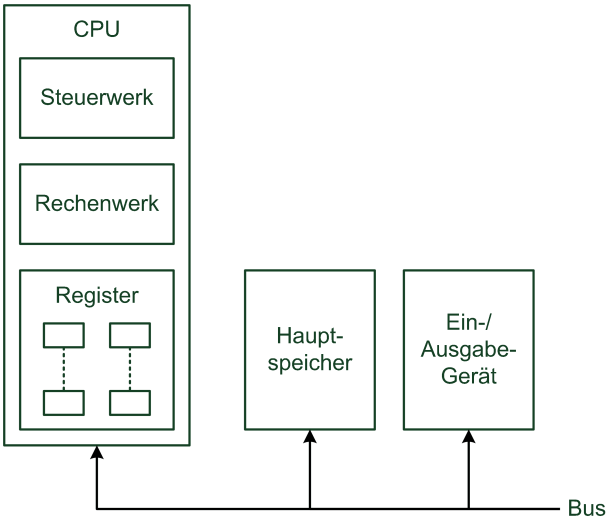


Abbildung 4.3: Einfacher Computer mit CPU [Tan06, S. 71]

Weiterhin verfügt eine CPU über internen Hochgeschwindigkeitsspeicher in Form von Registern, die Zwischenergebnisse und Steuerinformationen aufnehmen können. Mit definierten Befehlen können Daten aus dem Speicher in Register und umgekehrt geschrieben werden.

Viele CPUs verfügen über Multimedia- oder SIMD[1]-Erweiterungen, um die Rechenleistung bei multimedialen Anwendungen zu erhöhen. Dabei handelt es sich um Befehlssätze,

[1] *single instruction stream, multiple data*

die die parallele Verarbeitung von mehreren ähnlichen Datensätzen mit einem Befehlsaufruf ermöglichen. Diese Erweiterungen werden unter Namen wie *MMX*, *3DNow!* oder *SSE* vermarktet. Die Rechenleistung von CPUs wird durch diese Erweiterungen erhöht, erreicht aber nicht das Niveau von speziell für Videoverarbeitung konzipierten Prozessoren.

Mikroprozessoren sind vielfältig programmierbar. Dazu können Hochsprachen wie C oder C++ zum Einsatz kommen, die über Compiler in für den Prozessor verwertbare Maschinenenbefehle umgewandelt werden.

4.2.2 Digitale Signalprozessoren (DSP)

Digitale Signalprozessoren *(digital signal processors* – DSP) sind Mikroprozessoren, die auf Signalverarbeitung spezialisiert sind. Zur Verarbeitung von analogen Signalen sind meist Analog-Digital- bzw. Digital-Analog-Wandler integriert. DSPs führen mathematische Kalkulationen in einer vorausberechenbaren Zeitdauer aus und sind damit für Echtzeit-Verarbeitungsprozesse geeignet. DSPs sind nach der Harvard-Architektur aufgebaut (siehe Abbildung 4.4).

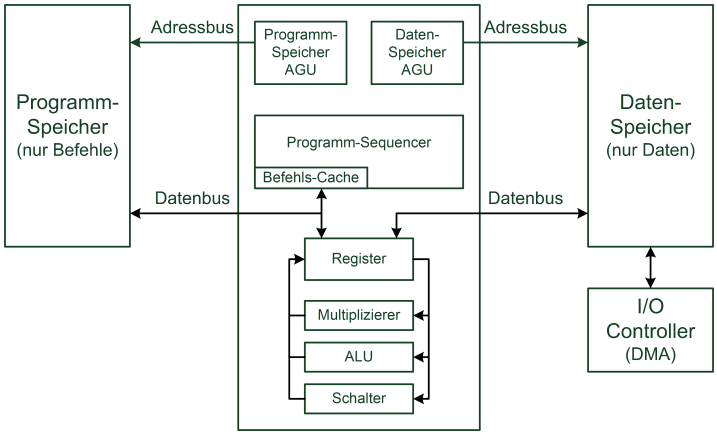


Abbildung 4.4: Typischer Aufbau eines DSP [Smi99, S. 513]

Typische DSPs enthalten weiterhin Multiplikationsakkumulatoren (MACs), die die mathematischen Operation $A^* = A + B \times C$ in einem Prozessorzyklus abarbeiten können. Das führt zu einer enormen Erhöhung der Geschwindigkeit für die Berechnung von Algorithmen wie der schnellen Fouriertransformation (*fast fourier transformation* – FFT) und der Faltung. Damit wird deutlich, dass DSPs Grundlage performanter Filter-Implementierungen darstellen können.

Die hohe Berechnungsgeschwindigkeit von DSPs ist auf mehrere Punkte zurückzuführen. Adressgeneratoren (*address generation unit* – AGU) erlauben z.B. die Umsetzung

von Schleifen ohne zusätzlichen Softwareaufwand, sodass Schleifen schneller ausgeführt werden können. Weiterhin berechnen AGUs Speicheradressen parallel zu arithmetischen Operationen. Das Laden von Daten vor der eigentlichen Nutzung (*prefetching*) und das *precoding* der Instruktionen ermöglichen außerdem eine hohe Ausführungsgeschwindigkeit der Befehle. *Precoding* unterstützt das Aufteilen der Berechnungen auf Teilaufgaben (*pipelining*), die dann für mehrere Befehle parallel durchgeführt werden können.

Zwei Arten der internen Zahlendarstellung sind bei DSPs unterscheidbar. Ein Teil kommerzieller DSPs ist den Festkomma-DSPs (*fixed-point*) zuzuordnen, die Zweier-Komplement-Zahlendarstellung verwenden. Das Zahlenformat von Gleitkomma-DSPs (*floating-point*) besteht aus Mantisse mit fester Breite und Exponenten. Die Produktion von Gleitkomma-DSPs ist gegenüber den Festkomma-DSPs mit höherem Hardwareaufwand verbunden und damit auch teurer. Außerdem sind sowohl ihre Chip-Fläche als auch der Leistungsverbrauch größer. Die gegenüber Festkomma-DSPs komplexeren Rechenwerke können im Allgemeinen nicht so hoch getaktet werden. Dieser Nachteil wird allerdings durch den erhöhten Aufwand, den man bei Festkomma-DSPs für komplexere Algorithmen betreiben muss, wieder aufgehoben. Auch die Zahlenformate haben verschiedene Vorzüge. Während bei Fixed-Point-DSPs Zahlen äquidistant sind und eine über den gesamten Zahlenbereich gleichbleibende Genauigkeit besitzen, erreichen Floatingpoint-DSPs eine wesentlich höhere Dynamik und sind weniger anfällig für Zahlenüber- und -unterläufe, allerdings ist deren Genauigkeit vom Exponenten abhängig. Häufig besitzen moderne DSPs die Möglichkeit, in beiden Modi zu arbeiten.

Verglichen mit Allzweckprozessoren sind DSPs kompakt in der Bauform und kostengünstig in der Anschaffung. Weil sie um ein Vielfaches sparsamer im Energieverbrauch sind, bedürfen DSPs kaum einer zusätzlichen Kühlung. DSPs und CPUs entwickeln sich in gewisser Weise in die gleiche Richtung, was insbesondere die interne Struktur (Mehrkern), Optimierungsmechanismen (Prefetch) sowie auch die Programmierung (C-Sprachen, Matlab) betrifft. Prefetching ist allerdings kein Alleinstellungsmerkmal von DSPs. Auch GPUs und CPUs nutzen massiv Prefetching-Operationen (L2-Cache, Dual-Core, Quadcore), dennoch bleiben die grundlegenden Eigenschaften der Harvard- (DSP) und von-Neumann-Architektur (CPU) bestehen.

4.2.3 Programmierbare Logikbausteine (FPGA)

Ein FPGA (*field programmable gate array*) ist ein programmierbarer integrierter Schaltkreis. Durch spezifische Konfiguration interner Strukturen können Schaltungen realisiert werden, die von einfachen Synchronzählern bis zu hochkomplexen Mikroprozessoren reichen. FPGAs werden vor allem dort eingesetzt, wo es auf schnelle Signalverarbeitung und flexible Änderung der Schaltung ankommt, z.B. beim Schaltungsentwurf von anwendungsspezifischen, integrierten Schaltungen (*application specific integrated circuit – ASIC*). Verbesserungen an den implementierten Funktionen können so nachträglich vorgenommen werden, ohne dabei direkt die physikalische Hardware ändern zu müssen.

Ein FPGA besteht aus Logikelementen, hauptsächlich *Flip-Flops*[2] und ladbaren Tabellen (*Lookup-Tables*), die elektronisch entsprechend der vom Entwickler gewünschten Funktion miteinander verknüpft werden können. Ein (*Lookup-Table*) kann eine beliebige kombinatorische Funktion (NAND, XOR, AND, Multiplexer etc.) aus den Eingangssignalen realisieren. Die Flip-Flops dienen dazu, Signalwerte zwischenzuspeichern, um sie im nächsten Takt weiterverarbeiten zu können. Aktuelle FPGAs bestehen aus bis zu einigen zehntausend Logikelementen. Für rechenintensive Designs, z.B. in der Signalverarbeitung, enthalten viele FPGAs Multiplizierer direkt auf dem Chip, die in einem einzigen Taktzyklus Multiplikationen durchführen können.

Die maximale Verarbeitungsgeschwindigkeit eines FPGAs ist vor allem von der verwendeten Halbleitertechnologie (Prozess, Strukturgrößen) und der internen Schaltungstopologie (Komplexität der Logikelemente) abhängig. Je nach der Anzahl und Komplexität der pro Takt durchzuführenden Operationen ergeben sich reale Systemtaktfrequenzen von ca. 10-200 MHz, obwohl die reinen Logikelemente höhere Taktfrequenzen (300-500 MHz) aufweisen. Im Vergleich zu anderen Prozessortypen haben FPGAs große Abmessungen und sind relativ kostenintensiv. Typische Anwendungsgebiete für FPGAs sind digitale Signalverarbeitung (Filter, schnelle Fourier-Transformation), Netzwerk-Protokoll-Implementierungen (Ethernet-MAC-Layer) und Kodierung von digitalen Videosignalen. Besonders in Bereichen kurzer Innovationszyklen haben rekonfigurierbare FPGAs viele Vorteile gegenüber anwendungsspezifischen Schaltungen (ASICs). Dazu zählen schnelle Marktreife, nachfolgende Fehlerbehebungen und Anpassung an neue Entwicklungen. Zusätzlich steht eine große Auswahl an vordefinierten Hardwareblöcken und Softwaremakros zur Verfügung, die Standardfunktionalitäten und -schnittstellen für die Signalverarbeitung bereitstellen und somit Entwicklungszeiten weiter verkürzen [KG06].

Zur Schaltungsentwicklung kommen Hardwarebeschreibungssprachen wie VHDL[3] oder Verilog zum Einsatz, die Funktion und Struktur komplexer Schaltungen beschreiben können. Als Programmspeicher werden die FPGA-internen RAM-Blöcke oder externe Speicher-ICs (SDRAM, SRAM) genutzt. Für einige Prozessorkerne stehen Hochsprachen wie C, C++ etc. zur Verfügung, andere werden in Assembler programmiert.

4.2.4 Grafikprozessoren (GPU)

Grafikprozessoren (*graphics processing unit* – GPU) sind stark parallel arbeitende Bearbeitungseinheiten, die sich in der Vergangenheit von spezialisierten Prozessoren mit festgelegten Funktionen zu vollständig parallel programmierbaren Prozessoren entwickelt haben und damit eine leistungsstarke Grundlage für rechenintensive Applikationen zur Verfügung stellen.

[2] einfache elektronische Schaltung zur Speicherung einer Datenmenge von einem Bit
[3] VHSIC Hardware Description Language, VHSIC = Very High Speed Integration Language

Eine GPU basiert auf der Harvard-Architektur; Daten und Befehle werden also in getrennten Speichern abgelegt. Parallele Berechnungen werden dadurch begünstigt. Weiterhin ermöglicht die separate Speicherung von Daten und Befehlen auch das Laden von Daten in den Prozessor im Vorhinein (*prefetching*). Der Schwerpunkt bei Grafikprozessoren liegt auf einem hohen Datendurchsatz und nicht auf einer möglichst schnellen Abarbeitung der Rechenvorgänge.

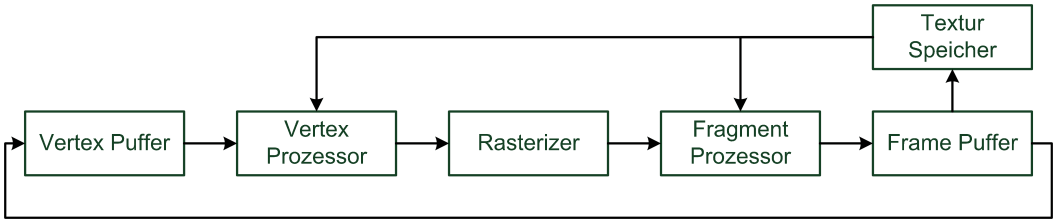


Abbildung 4.5: Grafikpipeline nach [OLG+05]

Der Aufbau einer klassischen GPU folgt der sequentiellen Abarbeitung einzelner Berechnungsschritte der so genannten Grafikpipeline (siehe Abbildung 4.5). Die Grafikpipeline beschreibt den Weg von der vektoriellen, mathematischen Beschreibung einer 3D-Szene zum gerasterten Bild auf dem Monitor. Ausgangspunkt ist also eine Liste von 3D-Koordinaten (*vertices*); das Ergebnis ist ein Pixelbild im Framepuffer, das auf einem Monitor angezeigt wird. Die Daten werden durch die einzelnen Schritte der Pipeline verarbeitet, wobei zu jedem Zeitpunkt jeder Schritt gleichzeitig die ihm zugestellten Daten berechnet. Innerhalb einer programmierbaren Grafikpipeline können ehemals genau spezifizierte (Vertex- und Fragment-) Prozessoren vom Nutzer frei programmiert werden. Moderne Grafikkarten nutzen nur einen vollprogrammierbaren Prozessortyp (*unified shader*), der alle Schritte der Pipeline übernehmen kann.

General Purpose Computing on GPUs (GPGPU) bezeichnet die Nutzung von GPUs für beliebige Anwendungen, die von der parallelen Verarbeitung Gebrauch machen können. Dazu zählt neben physikalischer Simulation auch Signal- und Bildverarbeitung. Für die ersten GPGPU-Applikationen wurden Fragment-Prozessoren über eine Assemblerähnliche Sprache programmiert. Einfacher konnte die Programmierung von GPUs auf Basis der *High Level Shading Language* (HLSL) erfolgen, die als Teil von DirectX9 mit dem Betriebssystem Microsoft Windows zur Verfügung steht. Fragment- und Vertexprozessoren können mit Shadersprachen wie *Cg* (NVIDIA) und *OpenGL Shading Language* programmiert werden. Weitere Hochsprachen wie *Sh* und *Brook*, *Accelerator* (Microsoft), CUDA[4] (NVIDIA) und *Stream*[5] (AMD) stellen weiter Programmierschnittstellen zur Verfügung. [Mün08]

Bei der GPU-Programmierung gilt es zwei Eigenschaften zu berücksichtigen. Einerseits müssen zu bearbeitende Daten vor der Berechnung vom Hauptspeicher des PCs zum

[4] *Computed Unified Device Architecture*
[5] vormals Close To Metal – CTM

Grafikspeicher der Grafikkarte transferiert werden. Andererseits verfügen aktuelle Grafikkarten kaum über prozessornahen Cache. Speicherzugriffe auf den Grafikspeicher der Grafikkarte sind nur vergleichsweise langsam (wenn auch bis zu zehn mal schneller als der Zugriff einer CPU auf den Hauptspeicher) realisierbar. So sind insbesondere Algorithmen zur Berechnung auf GPUs geeignet, die eine hohe Anzahl von Berechnungen bei gleichzeitig geringer Anzahl von Speicherzugriffen zur Folge haben. Eine weitgehende Unabhängigkeit der Daten voneinander trägt zusätzlich zu einem hohen Grad an Paralellverarbeitung bei, die prinzipbedingt von GPUs sehr gut unterstützt wird.

4.2.5 Integration auf PC-Basis

Es wird deutlich, dass die verschiedenen Prozessortypen zu flexiblen, leistungsfähigen und dennoch preiswerten Systemen kombiniert werden können. In handelsüblichen PC-Systemen können so FPGA-, DSP- und GPU-Architekturen über Schnittstellenkarten integriert und von einer CPU gesteuert werden. So können für verschiedene Anwendungsfälle jeweils geeignete Prozessoren ausgewählt werden. Zu beachten ist dabei, dass die Daten vom und zum Hauptspeicher des Systems kopiert werden müssen, was je nach verwendeter Schnittstelle eine nicht zu vernachlässigende Zeitdauer in Anspruch nimmt.

Eine Herausforderung besteht nun darin, Eigenschaften von PC-Systemen bei der Integration externer Prozessoren zu beachten. Deshalb soll nachfolgend die PC-Architektur hardwareseitig und softwareseitig beleuchtet werden.

4.3 PC-Architektur

Personal Computer (PCs) bestehen sowohl hardware- als auch softwareseitig aus verschiedenen, vergleichsweise preiswerten Komponenten. Mit der steigenden Leistungsfähigkeit dieser Komponenten wird der Einsatz von PC-Technologie auch im professionellen Bereich der Studiotechnik möglich. Deshalb wird im Folgenden überblicksartig auf die Bereiche Hardwareaufbau und Betriebssystem eingegangen.

4.3.1 Hardwareaufbau

Ein Desktop-PC ist traditionell nach der *bridge*-basierten Struktur aufgebaut. Der Hauptprozessor (CPU) ist dabei über einen schnellen Prozessorbus mit dem Cache-Speicher[6]

[6] schneller, CPU-naher Zwischenspeicher

verbunden. Die so genannte *north bridge*[7] verbindet den Prozessorbus mit einem Systembus (auch PCI-Bus) und stellt außerdem oft AGP[9]-Schnittstellen zur Verfügung. In manchen Architekturen wird auch der Hauptspeicher auch mit dem Prozessorbus verbunden, um kurze Zugriffszeiten zu ermöglichen. Der Systembus bindet weiterhin Komponenten wie SCSI-Host-Adapter oder LAN-Controller an, die über die PCI-Schnittstelle auf dem Mainboard des Rechners untergebracht werden. Eine zweite Brücke (*south bridge*) steuert Festplatten und USB-Komponenten an und koppelt zusätzlich den PCI-Bus an den ISA-Bus, der langsamere Systemkomponenten wie Eingabeperipherie- oder Diskettenkontroller beliefert. [Mär03]

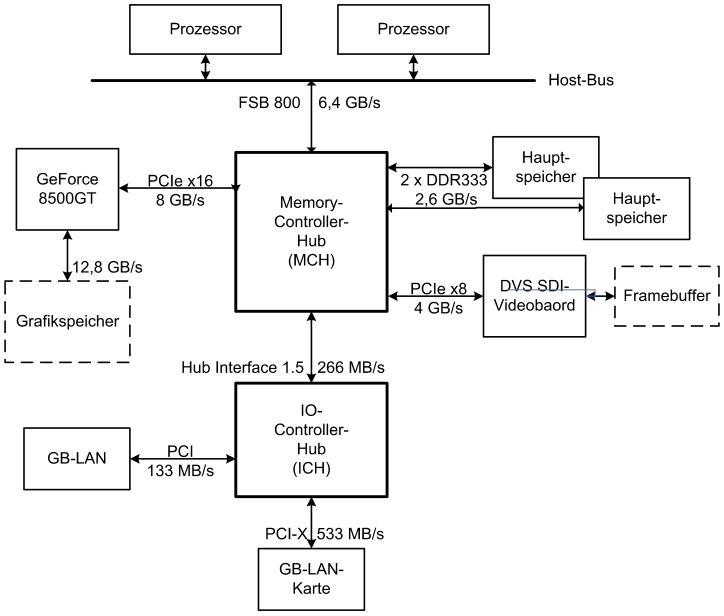


Abbildung 4.6: Aktuelle Hub-basierte Rechnerarchitektur [Ote08]

Modernere PC-Architekturen verwenden zentrale Vermittlungseinrichtungen (so genannte *Hubs*), die nicht mehr über den Systembus, sondern über schnelle, serielle Punkt-zu-Punkt-Verbindungen miteinander verbunden sind. Die *North-Bridge* wird durch einen *Memory-Controller-Hub* (MCH) ersetzt, der als Bindeglied zwischen Prozessorbus, Hauptspeicher, Grafikkarte und einem zweiten *Hub* (*I/O-Controller-Hub* – ICH) fungiert. Der ICH ermöglicht die Kommunikation mit Standard-PCI-Komponenten, wie LAN-Controllern und anderen niederratigen Peripheriegeräten. [Mün08] Abbildung 4.6 zeigt die Hub-basierte Struktur eines handelsüblichen PC-Systems[10].

Das Hub-Konzept hat zwei entscheidende Vorteile gegenüber dem Bridge-Konzept: Erstens können höhere Übertragungsraten realisiert werden. Zweitens wird der Durchsatz

[7] auch als *Host-to-PCI*[8]-Bridge bezeichnet
[9] *Accelerated Graphics Port*, ein Anschluss zur Anbindung der Grafikkarte
[10] am Beispiel des im Testaufbau verwendeten Intel-PCs „Funok“, siehe Abschnitte 6.5.3.1 ff.

zwischen den Hubs nicht von einem globalen Bus begrenzt, auf welchem nur zwei Komponenten gleichzeitig Daten austauschen können, sondern die Übertragung kann jeweils zwischen zwei Komponenten ausgehandelt werden. [Fli05, S. 289]

In zukünftigen PC-Architekturen nimmt der Prozessor eine immer zentralere Position ein. Die bisher übliche Trennung in Speicher- und IO-Controller wird aufgelöst und der Speichercontroller in den Prozessor integriert. Für Consumer-Modelle sind sogar die vollständige Integration der PCIe-Schnittstelle und die Anbindung des IO-Controllers an die CPU vorgesehen. Abbildung 4.7 zeigt die nächste Generation von Intel[11]-Rechnern. [Ote08]

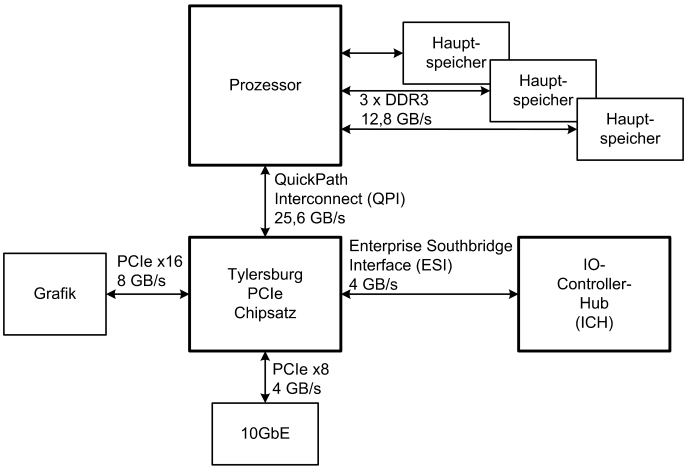


Abbildung 4.7: Zukünftige Rechnerarchitektur [Ote08]

Daten, die zunächst nicht vom Prozessor verarbeitet werden sollen, können ohne die Mitwirkung des Prozessors über einen direkten Speicherzugriff (*Direct Memory Access* – DMA) transportiert werden. DMA wird von einem Prozessor-unabhängigen Controller initiiert und entlastet dadurch die CPU von Eingabe/Ausgabe-Aufgaben. Über mehrere Registersätze können DMA-Controller mehrere Übertragungen gleichzeitig steuern.

Weil interne Busstrukturen häufig den Flaschenhals bei der Übertragung hochratiger Datenströme darstellen, sollen im Folgenden kurz verschiedene Systembusse hinsichtlich ihrer Leistungsfähigkeit verglichen werden (siehe Tabelle 4.1).

Die älteste Variante ist der PCI[12]-Bus, welcher bis zu 133 MB/s übertragen kann. Die für den Server-Markt bestimmte Erweiterung PCI-Extended (PCI-X) erlaubt eine Datenrate von üblicherweise 1.033 MB/s. Bei PCI und PCI-X teilen sich alle angeschlossenen Geräte die zur Verfügung stehende Übertragungsrate. In modernen Rechner werden beide Bussysteme durch die PCI-Express-Schnittstelle (PCIe) ersetzt. Diese bietet mehrere

[11] Prozessor-Hersteller
[12] *Peripheral Component Interconnect*

Bus-Standard	Bus-Breite	Taktfrequenz	Maximale Datenrate
PCI (verbreitet)	32 bit	33 MHz	133 MB/s
PCI (maximal)	64 bit	66 MHz	533 MB/s
PCI-X (verbreitet)	64 bit	133 MHz	1.033 MB/s
PCI-X (maximal)	64 bit	533 MHz	4.266 MB/s
PCIe je Lane	seriell	–	500 MB/s
PCIe 2.0 je Lane	seriell	–	1.000 MB/s

Tabelle 4.1: Überblick über PCI-Standards [Ote08]

serielle Verbindungen (*Lanes*), die den Geräten exklusiv zugewiesen werden. Jede *Lane* überträgt bis zu 250 MB/s in jede Richtung. Durch die Bündelung von *Lanes* kann die Übertragungsrate erhöht werden. Handelsübliche Chipsätze stellen 24 *Lanes* zur Verfügung, wobei in der Regel jeweils ein Steckplatz mit 16 *Lanes* (gewöhnlich für den Grafikadapter) und ein Steckplatz mit 8 *Lanes* verbaut wird. Damit steht eine Gesamtdatenrate von 8 GB/s bzw. 4 GB/s zur Verfügung. [Ote08]

4.3.2 (Echtzeit)-Betriebssysteme

Die Aufgaben eines Betriebssystems lassen sich in zwei Bereichen zusammenfassen. Einerseits abstrahiert ein Betriebssystem die Details der Hardwarenutzung (Speicherzugriffe usw.) und stellt eine vereinfachte Repräsentation der darunter liegenden Hardware zur Verfügung, die eine einfache Programmierung durch den Benutzer erlaubt. Andererseits fungiert ein Betriebssystem als „Ressourcenmanager“, der den Zugriff auf gemeinsam genutzte Komponenten (Prozessor, Speicher, Drucker usw.) regelt. [Tan05]

Betriebssysteme kommen den Anforderungen von spezifischen Einsatzgebieten nach. So existieren Mainframe-Betriebssysteme, die darauf ausgelegt sind, viele Prozesse mit hohen Anforderungen an Ein- und Ausgabegeschwindigkeit gleichzeitig auszuführen. Auch Server-Betriebssysteme bedienen viele Nutzer zur gleichen Zeit und stellen über Netzwerke Hard- und Softwareressourcen zur Verfügung. Aufgrund hoher technologischer Anforderungen und vergleichsweise geringer Verbreitung sind Mainframe- und Server-Betriebssysteme mit höheren Kosten für Anschaffung, Betrieb und Wartung verbunden, als PC-Betriebssysteme. Ursprünglich wurden PC-Betriebssysteme wie z.B. Windows 98/2000/XP als Schnittstelle für einzelne Benutzer eingesetzt. Mit der zunehmenden Leistungsfähigkeit von PC-Komponenten und umfassender Vernetzung auch im privaten Bereich werden die Unterschiede zwischen PC- und Server-Betriebssystemen kleiner. Auch eingebettete Systeme wie z.B. PDAs[13] und Mobiltelefone verfügen über entsprechend angepasste Betriebssysteme. Insbesondere in zeitkritischen Systemen z.B. der ma-

[13] *personal digital assistant*

schinellen Fertigung muss ein Betriebssystem seine Aufgaben unter Einhaltung zeitlicher Grenzen erfüllen. Nachfolgend soll der Fokus auf Echtzeitsysteme bzw. Echtzeitbetriebsysteme gerichtet werden.

Ein Echtzeitsystem ist eine Kombination aus Hardware und Software, welche Daten empfängt, verarbeitet und die Ergebnisse innerhalb einer vorgegebenen definierbaren Zeitspanne wieder ausgibt. Es werden allgemein zwei Klassen von Echtzeit unterschieden: Harte Echtzeit beschreibt die Anforderung, dass die Datenverarbeitung in jedem Fall zu einem vorher definierten Zeitpunkt abgeschlossen sein muss. Ein Überschreiten der „Deadline“ kann nicht toleriert werden (z.B. Roboterarmsteuerung bei einem Fertigungsprozess). Weiche Echtzeit erlaubt hingegen eine Datenverarbeitung in dehnbaren Zeitgrenzen, d.h. die Überschreitung der Zeitgrenze durch einen Verarbeitungsprozess hat keine schwerwiegenden Folgen (z.B. Multimediaanwendungen).

Echtzeitbetriebssysteme besitzen einen so genannten Mikrokern(el)[14], der eine Systemuhr, den Scheduler (Taskmanager) und die Semaphorenverwaltung[15] beinhaltet und damit den Ablauf von Prozessen steuert. Zwischen dem Mikrokernel und dem Anwendungsprogramm befindet sich das Echtzeitbetriebssystem.

Ein Rechenprozess (*task*) ist eine organisatorische Einheit, die aus ausführbarem Programmcode, eigenem Speicher und Variablen besteht. Ein Prozess besitzt zudem eine Priorität und einen Zustand (ruhend, bereit, laufend, wartend, beendet). Prozesse werden mit ihrer Priorität in Listen eingetragen, die der Kernel verwaltet. Ablaufbereite Prozessen werden so in einer „*Ready Queue*“ gesammelt und nach einem Scheduling-Algorithmus von der CPU verarbeitet.

Verschiedene Scheduling-Algorithmen können unterschieden werden. In Prioritäts-gesteuerten Systemen bekommt derjenige Prozess die CPU zugeteilt, der ablaufbereit ist und die höchste Priorität besitzt. Ein für Echtzeitsysteme wichtiges Verfahren ist das prioritätsbasierte preemptive Scheduling mit dynamischen Prioritäten. Jeder Prozess in einer Softwareapplikation muss einer Priorität zugeordnet sein, wobei höhere Prioritätswerte schnellere Reaktion bedeuten. Bei preemptiven Verfahren können aktuelle bearbeitete Prozesse in der Verarbeitung verdrängt, also in den Zustand „bereit“ versetzt, werden, wenn entsprechend höher priorisierte Prozesse Prozessorzeit beanspruchen. So ist die bevorzugte Bearbeitung und damit die garantierte Fertigstellung von Prozessen zu einem definierten Zeitpunkt möglich.

Standard-PC-Betriebssysteme wie *Microsoft Windows XP* und Standard-Linux-Distributionen sind nicht deterministisch bzw. echtzeitfähig, auch wenn Weiterentwicklungen

[14] Ein Mikrokernel verfügt im Gegensatz zu einem monolithischen Kernel nur über grundlegende Funktionen (Funktionen zur Speicher- und Prozessverwaltung, sowie zur Synchronisation und Kommunikation). Alle weiteren Funktionen werden als eigene Prozesse (Server) oder als Programmbibliothek im Benutzer-Modus implementiert

[15] Semaphore werden zur Prozesssynchronisation eingesetzt, wenn die parallele Ausführung mehrerer Prozesse/Threads eine zeitliche Abstimmung der Ausführungen erfordert

dazu geführt haben, dass z.B. ein aktuelles Linuxsystem weiche Echtzeitanforderungen erfüllt. Mit der Implementierung entsprechender Erweiterungen[16] entsteht ein echtzeitfähiges Linux. Hierbei werden die Echtzeittasks von einem (von Linux unabhängigen) erweiterbaren Minikernel verwaltet. Der eigentliche Linux-Betriebsystemkern stellt aus Sicht dieses Minikernels einen Prozess mit niedriger Priorität dar, das heißt, er kann von allen anderen Echtzeitprozesse zu jeder Zeit unterbrochen werden. Typische Aufgaben für RT-Linux finden sich im Bereich der Mess-, Steuer- und Regelungstechnik, wo harte Deadlines unter 50 Mikrosekunden gefordert sind. Weitere Beispiele für Echtzeitbetriebssysteme sind *Microsoft Windows CE* und *Mentor Graphics VRTX*.

Mit zunehmender Leistungsfähigkeit der zugrunde liegenden Hardware werden Berechnungszeiten für Prozesse verkürzt. Werden in einem konkreten Anwendungsfall vergleichsweise geringe Echtzeitanforderungen gestellt, d.h. die Deadline einen relativ großen Zeitraum abspannt (z.B. mehrere Millisekunden), können auch Betriebssysteme ohne preemptive Scheduling-Mechanismen zu Einsatz kommen. Voraussetzung dafür ist eine geringe Prozessorlast durch konkurrierende Anwendungen bzw. die Zuweisung entsprechender Prioritäten.

4.4 Latenzbetrachtungen zur Be- und Verarbeitung

Die Gesamtsystemlatenz einer Infrastruktur für Studioanwendungen setzt sich im Wesentlichen aus zwei Komponenten zusammen. Zunächst verursacht die Datenübertragung eine messbare Verzögerung. Sie umfasst alle Prozesse im Netzwerk, insbesondere in den Netzwerkknoten, aber auch im Sender und Empfänger (siehe Abschnitt 3.2). Zusätzlich tragen andere Prozesse bei Sender und Empfänger zur Gesamtlatenz bei – insbesondere, wenn die Hosts (Workstations) zur Mediendaten-Verarbeitung genutzt werden. Im Anschluss sollen alle Latenzbestandteile einer Workstation zusammengestellt werden, die nicht schon über das Latenzmodell der Netzwerkübertragung abgedeckt sind. Auch hier ist das Ziel die Identifikation von vermeidbaren Verzögerung für eine möglichst geringe Gesamtsystemlatenz.

Am Beispiel von Videoessenzdaten im Studiobereich können die Bestandteile der Verzögerung in einer Workstation identifiziert werden: Latenzen durch Verarbeitungsprozesse (t_{prozess}), durch Kompression ($t_{\text{kompression}}$), durch Fehlerschutz ($t_{\text{fehlerschutz}}$), durch Pufferung (t_{puffer}) und durch die Verwendung von Containerformaten wie MXF[17], AVI[18], Quicktime usw. ($t_{\text{container}}$). Alle Bestandteile summieren sich zur Workstation-Latenz ($t_{\text{WORKSTATION}}$).

[16] RT-Linux, entwickelt am „New Mexico Institute of Technologie“, siehe http://www.rtlinux.org
[17] Material eXchange Format
[18] Audio Video Interleave

$$t_{\text{WORKSTATION}} = t_{\text{prozess}} + t_{\text{kompression}} + t_{\text{fehlerschutz}} + t_{\text{puffer}} + t_{\text{container}}$$

Verarbeitungsprozesse können beispielsweise die Anwendung von Filteralgorithmen, Bildmischungen oder Effekte umfassen. In Abhängigkeit der Komplexität der verwendeten Algorithmen, der notwendigen Datenbasis (z.B. ein oder mehrere aufeinander folgende Vollbilder) und der zur Verfügung stehenden Rechenleistung des Workstations (auf CPU, GPU, DSP oder FPGA) kann t_{prozess} Größenordnungen von einigen Mikrosekunden bis wenigen hundert Millisekunden erreichen. Insbesondere hochauflösende Videodaten (HDTV) haben große Datenraten zur Folge, so dass Videodaten mit Rücksicht auf vorhandene Signalschnittstellen bzw. Datennetzwerken zur Übertragung komprimiert werden. Die dadurch entstehende Latenz $t_{\text{kompression}}$ kann in Abhängigkeit der verwendeten Algorithmen bis zu mehreren 10 ms betragen. Durch speziell latenzoptimierte Verfahren wie Dirac Pro (SMPTE VC-2) lassen sich aber auch geringe Latenzen von wenigen HDTV-Zeilen erreichen (0,25 ms) [Dav08].

Weil oft davon ausgegangen werden muss, dass der Übertragungskanal nicht störungsfrei zur Verfügung steht, kann den zu übertragenden Daten senderseitig zusätzlich Redundanz hinzugefügt werden, die im Fehlerfall empfängerseitig zur Berechnung der Originaldaten herangezogen werden. Die Leistungsfähigkeit dieser Fehlerschutzmechanismen steigt mit dem Umfang der hinzugefügten Redundanz. Mit ihm steigt aber auch die dadurch verursachte Verzögerung $t_{\text{fehlerschutz}}$, die schnell Werte von mehreren 100 ms erreicht [NJH08].

Hauptsächlich bei der Verwendung isochroner Signalübertragung (z.B. SDI) können zusätzliche Wartezeiten entstehen, wenn durch vorgelagerte Prozesse eine Informationseinheit (z.B. ein Voll- oder Halbbild) nicht oder nicht vollständig am Ausgang zum Versenden zur Verfügung steht (t_{puffer}). Die Anwendung von Containerformaten wie MXF ist mit einer theoretisch minimalen Latenz $t_{\text{container}}$ versehen, weil das Neuanordnen von Elementen im Datenstrom vergleichsweise schnell umgesetzt werden kann. Dennoch können durch die Anwendung von vollbildbasierten SDKs und in Verbindung mit isochronen Signalschnittstellen (s.o.) schnell Verzögerungen von mehreren Vollbildern (x * 40 ms) entstehen (vergleiche Abschnitt 6.5.3.5).

4.5 Konklusion

Der Einsatz von Standard-IT-Hardware und -Software im Bereich der professionellen Fernsehproduktion wird maßgeblich aus Kostengründen forciert. Schon heute werden z.B. Videoserver auf PC-Technologie erfolgreich im funktionskritischen Bereich eingesetzt. Durch die rasante Weiterentwicklung von Prozessorgeschwindigkeiten, Netzwerkübertragungsraten und Speicherkapazitäten können PC-basierte Systeme zunehmend auch für

echtzeitkritische Prozesse während einer Live-Produktion zur Anwendung kommen. Um den nach wie vor hohen Anforderungen der Be- und Verarbeitung von Content nachzukommen, werden PC-Komponenten zu angepassten, leistungsfähigen „Workstations“ zusammengesetzt.

Die ohnehin hohe Rechenleistung der Hauptprozessoren (CPUs) kann sehr flexibel erweitert werden, indem FPGAs und DSPs zur signalspezifischen Bearbeitung über hochratige Schnittstellen an das System angebunden werden. Insbesondere der Bereich GPGPU ist Gegenstand vielfältiger Forschungsarbeiten, von deren Ergebnissen auch die professionelle Fernsehproduktion profitiert. Zu beachten ist in diesem Zusammenhang die hinreichende Dimensionierung der Systembusse hinsichtlich maximaler Durchsätze, um Flaschenhälse zu vermeiden. Weiterhin ist die Notwendigkeit der internen Datenverteilung über Speicherkopierprozesse mit zusätzlichen Latenzen verbunden, die im Gesamtzusammenhang berücksichtigt werden müssen. Das aufgestellte Latenzmodell für Verarbeitungsknoten identifiziert daneben weitere Prozesse auf einer Workstation, die Rückschlüsse auf die Konzeption einer Gesamtarchitektur zulassen.

Bezüglich des Einsatzes von Standard-Betriebsystemen wie z.B. *Microsoft Windows XP* und Standard-Linux-Distributionen, die nicht deterministisch arbeiten, ist experimentell zu bestimmen, ob und unter welchen Voraussetzungen die Anforderungen im Studiobereich erfüllt werden können. Grundlegende Voraussetzung dafür ist eine geringe Prozessorlast durch konkurrierende Anwendungen bzw. die Zuweisung entsprechender Prioritäten echtzeitkritischer Prozesse.

Insgesamt tragen die beschriebenen Weiterentwicklungen dazu bei, dass langfristig signalspezifische Schnittstellen wie AES/EBU und SDI durch universelle IT-basierte Schnittstellen ersetzt werden und damit die Tendenz zu Klasse-4-Geräten für die Medienbearbeitung im Studio nach der Klassifizierung von Kovalick bestätigt wird.

5 Anwendung von Metadaten in linearen Produktionsprozessen

Metadaten stellen Daten über Daten dar, beinhalten also in Bezug auf die Fernsehproduktion Informationen über die Essenz. Metadaten dienen unterschiedlichen Zwecken und können auf diese Weise klassifiziert werden. Im MXF-Kontext wird zwischen strukturellen (notwendig zur Dekodierung einer MXF-Datei) und beschreibenden Metadaten (für Annotationen) unterschieden (vergleiche Abschnitt 5.2.3). Auch die zeitliche Gültigkeit kann als Kriterium zur Klassifikation von Metadaten herangezogen werden. Beispielsweise können sich Metadaten auf die gesamte Dauer der Essenz (statische Metadaten) oder nur auf definierte Zeitintervalle (dynamische Metadaten) beziehen. Darüber hinaus kann die funktionsspezifische Gruppierung von Metadaten vorgenommen werden. So fassen z.B. „intimate Metadata“ Zusatzinformationen zur Verbesserung bzw. Stabilisierung der Bildqualität im Rahmen der Postproduktion zusammen (vergleiche [WTK+02]). Metadaten können durch geeignete Mechanismen (z.B. Sprach- und Mustererkennung) automatisch erzeugt werden oder manuell in entsprechende Systeme eingegeben werden.

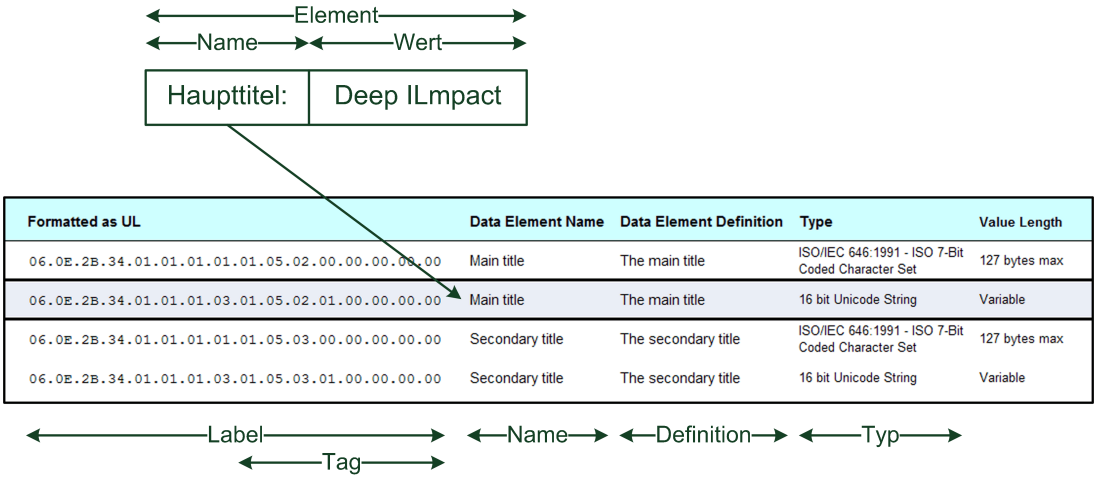
Die durchgängige Repräsentation von Daten in einer genormten Art und Weise erlaubt die maschinelle Nutzung des Datenbestandes ohne Nutzerinteraktion und damit die Automatisierung von Arbeitsabläufen zur Be- und Verarbeitung des Datenbestandes. Dies trifft auch für den echtzeitkritischen Bereich der Live-Studioproduktion zu. In diesem Kapitel sollen grundlegende Prinzipien insbesondere der Metadatenorganisation und deren Einfluss auf zentrale Aspekte bezüglich der Nutzung in einer IT-Netzwerk-basierten Architektur verdeutlicht werden.

Zu diesem Zweck werden zunächst für den Bereich der Fernsehproduktion zentrale Metadaten-Standards zueinander in Beziehung gebracht. Das für diese Arbeit zentrale Material Exchange Format (MXF) erfährt dann eine detailliertere Beleuchtung. Mögliche Anwendungsszenarien von MXF im Studio bilden dann die Voraussetzung von Designentscheidungen für das Konzept einer Nutzung von Metadaten in einer IT-Netzwerk-basierten Architektur.

5.1 Standards für anwendungsübergeifende Nutzung

Sowohl Essenz- als auch Metadaten beschreiben jeweils die Einheit von Elementen, die aus einem Namen und einem Wert bestehen. Abbildung 5.1 stellt diesen Zusammenhang

an einem Beispiel dar. Zum Zwecke der globalen Eindeutigkeit werden die Namen in Verzeichnissen aufgelistet und mit eindeutigen Identifikatoren (*Tags* und *Labels*) versehen.



Formatted as UL	Data Element Name	Data Element Definition	Type	Value Length
06.0E.2B.34.01.01.01.01.01.05.02.00.00.00.00.00	Main title	The main title	ISO/IEC 646:1991 - ISO 7-Bit Coded Character Set	127 bytes max
06.0E.2B.34.01.01.01.03.01.05.02.01.00.00.00.00	Main title	The main title	16 bit Unicode String	Variable
06.0E.2B.34.01.01.01.01.01.05.03.00.00.00.00.00	Secondary title	The secondary title	ISO/IEC 646:1991 - ISO 7-Bit Coded Character Set	127 bytes max
06.0E.2B.34.01.01.01.03.01.05.03.01.00.00.00.00	Secondary title	The secondary title	16 bit Unicode String	Variable

Abbildung 5.1: Begriffsdefinitionen für Metadaten am Beispiel SMPTE RP.210

5.1.1 SMPTE-Metadatenframework

Für die Film- und Fernsehindustrie wichtige Standardisierungsarbeit leistet die *Society of Motion Picture and Television Engineers* (SMPTE). Seit der durchgängigen Digitalisierung der Produktionsabläufe rückt dabei neben den Definition von Essenzdaten-Kodierverfahren und Übertragungsschnittstellen zunehmend der Bereich der Daten- und Metadatenorganisation in den Vordergrund. Mit der Standardisierung von Verzeichnissen (*Registries*[1]) wird ein einheitliches Metadaten-Vokabular und damit der interoperable Austausch und die umfassende Anwendung von Metadaten ermöglicht. Diese Verzeichnisse stellen Listen dar, in denen Einträge durch Kennzeichner (*Universal Labels*[2]) eindeutig referenziert werden. Es existieren vier *Registries*, die strukturell durch eigene Standard-Dokumente definiert und jeweils als Excel-Tabellen[3] umgesetzt sind. Im Sinne einer offenen Erweiterbarkeit und einer Rückwärtskompatibilität dürfen die Einträge eines Verzeichnisses ergänzt, aber nicht verändert werden. Abbildung 5.2 verdeutlicht die Zusammenhänge zwischen den SMPTE-Verzeichnissen.

Im *SMPTE Metadata Dictionary* sind die Namen von Metadatenelementen in einer Baumstruktur einer Klassifikation untergeordnet. Momentan sind zehn Klassen definiert (vergleiche SMPTE 335M). Dazu zählt auch eine Klasse, in der Organisationen eigene, nicht öffentliche Einträge vornehmen können. Zusätzlich sind nähere Beschreibungen

[1] auch *Dictionaries* (Wörterbücher)
[2] 16-Byte-Kennzeichner, siehe SMPTE 298M
[3] dokumentiert durch *Recommended Practices*

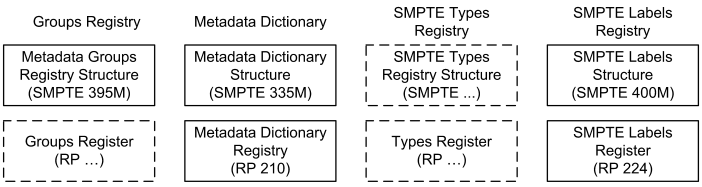


Abbildung 5.2: SMPTE Metadaten-Framework

(Definitionen) der Namen und (Daten)-Typen der Werte dokumentiert, wobei sowohl primitive (Integer) als auch komplexe Datentypen (Gruppen von Metadaten) definiert sind. Die *Groups Registry* stellt die Auflistung möglicher Metadatengruppen dar; an der Umsetzung in eine *Recommended Practice* wird noch gearbeitet. Auch die *Types Registry*, die alle Datentypen aufnehmen soll, die Metadaten annehmen können, befindet sich noch im Aufbau. Die *Labels Registry* schließlich stellt alle SMPTE-Labels zur Identifizierung von Essenz-, Meta- und anderen Daten dar.

5.1.2 Metadatenmodelle

Die für einen bestimmten Anwendungsbereich relevanten Metadaten werden mithilfe von Metadatenmodellen zusammengefasst. Dabei werden Metadatenelemente gruppiert (*Sets*) und Beziehungen zwischen diesen Gruppen definiert. Ein Metadatenset wird auch als Klasse bezeichnet, die Metadatenelemente einer Gruppe stellen die Attribute der Klasse dar.

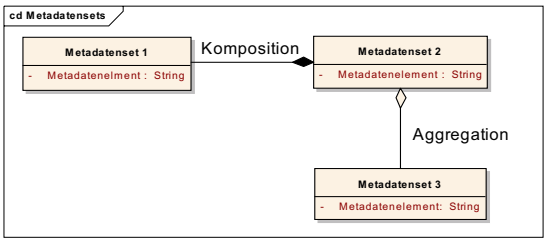


Abbildung 5.3: UML-Klassendiagramm: Aggregation und Komposition [Zim06]

Jedes Attribut einer Klasse besitzt einen definierten Datentyp. Metadatenmodelle können mithilfe von UML[4]-Diagrammen visualisiert werden. UML-Klassenmodelle spezifizieren (gerichtete) Beziehungen zwischen darin enthaltenen Metadatensets. Referenzen unterscheiden sich in Aggregation[5] und Komposition[6] (vergleiche Abbildung 5.3). [Zim06]

[4] Unified Modelling Language – eine graphische Modellierungssprache für den Entwurf objektorientierter Softwaresysteme

[5] schwache „Teile-Ganzes-Beziehung“: Zeiger von einem Metadatenset („Ganzes“) zu einem anderen Metadatenset („Teil“)

[6] starke „Teile-Ganzes-Beziehung“: ein Metadatenset („Ganzes“) „besitzt“ ein anderes Metadatenset

Zur Nutzung in MXF wurde das Descriptive Metadata Scheme-1 (**DMS-1**) spezifiziert (vergleiche SMPTE 380M). DMS-1 enthält drei *Descriptive Metadata Frameworks*, die der Beschreibung der Essenz in MXF dienen (siehe Abbildung 5.4). So können Informationen über das gesamte Programm, Programmteile oder einzelne Szenen angegeben werden. Zur Synchronisation dieser Metadaten mit der Essenz werden die drei Frameworks genauso wie Audio, Video oder Timecode mittels logischer Spuren (*Tracks*) in der MXF-Datei angeordnet, wobei jedes Framework in einem eigenen Track angelegt ist. Zur Referenzierung der Metadatenelemente wird das SMPTE *Metadata Dictionary* genutzt (SMPTE RP 210).

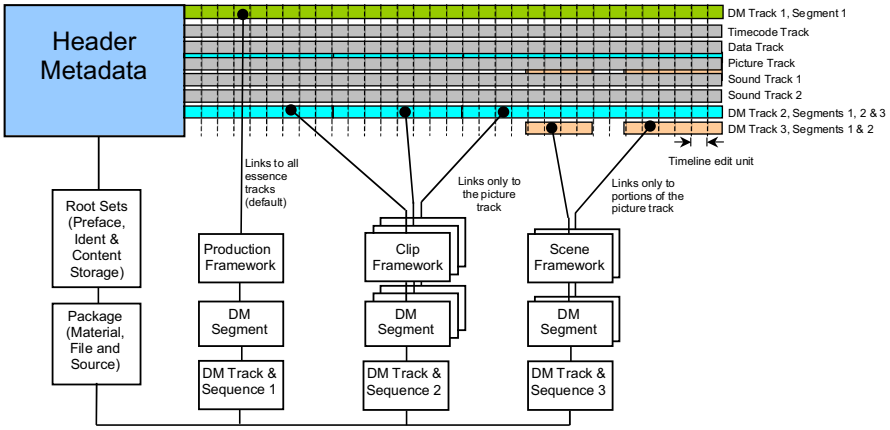


Abbildung 5.4: DMS-1 Frameworks und deren Beziehungen zum MXF-Datenmodell nach SMPTE 380M

DMS-1 ist zwar für die Beschreibung auszutauschenden Materials umfassend, aber nicht ausreichend, um ganze Produktionsprozesse zu unterstützen. Zudem kann DMS-1 nur in Verbindung mit MXF existieren, ohne MXF gehen sämtliche Bezüge zu den zugehörigen Essenzen verloren.

In Zusammenarbeit mit der Rundfunk-Industrie hat das IRT[7] mit dem *Broadcast Metadata Exchange Format* (**BMF**) ein weiteres Datenmodell für den Metadatenaustausch in der Fernsehproduktion entwickelt. BMF deckt das gesamte Spektrum an Bereichen, die ein Programmbeitrag durchlaufen kann, ab: Von der redaktionellen Arbeit, Planung, Rechteverwaltung, Disposition, Produktion, Bearbeitung, Sendung bis hin zur Archivierung und Wiederverwendung. Wesentliche Anforderungen an die Entwicklung des Datenmodells waren die Kompatibilität zu FESADneu[8] und dem Metadaten-Plugin-Mechanismus von MXF.

[7] Institut für Rundfunktechnik, München

[8] Weiterentwicklung der Fernsehdatenbank FESAD, die in allen Produktionshäusern der ARD genutzt wird. Neuerungen sind die Erweiterung um Multimedia- und Rechteverwaltung, also Content- und Asset-Management.

Grundsätzlich lässt sich BMF in zwei Teile gliedern. Das „Produktionselement“ folgt dem redaktionellen Konzept und umfasst die Abbildung der logischen Struktur, die inhaltliche Beschreibung sowie alle erforderlichen prozessbegleitenden Informationen. Dabei ist es nicht zwingend erforderlich, dass das beschriebene Material auch vorhanden ist. Eine Sendung, ein Beitrag oder eine Szene kann somit auch im Vorfeld inhaltlich geplant werden. Der mit „Physikalische Realisierung und Speicherung“ (PRS) bezeichnete zweite Teil kann ohne ein Produktionselement nicht existieren. PRS umfasst die Materialbeschreibung aus der logischen Sicht, also die Beschreibung der Signale und deren Speicherung.

Verfahrensvorschriften für die Transformation zwischen den Datenmodellen von BMF und MXF wurden im Rahmen einer Diplomarbeit am IRT ausgearbeitet, so dass BMF als erweiterungsfähiges Datenmodell im MXF-Nutzungskontext genutzt werden kann [Zim06].

5.2 Material eXchange Format

Obwohl das Material eXchange Format (MXF) mit dem Ziel entwickelt und standardisiert wurde, auf möglichst einfachem Weg Interoperabilität beim Datenaustausch zu gewährleisten, ist der modulare Standard mittlerweile auf mehrere 100 Seiten Umfang angewachsen (siehe Abbildung 5.5). Die Implementierung selbst einfacher Applikationen ist damit relativ komplex. *Software Development Kits* (SDKs)[9] bieten dem Entwickler über entsprechende Schnittstellen (Klassen, Objekten, Funktionen) eine abstrahierende Möglichkeit, MXF-Anwendungen zu implementieren.

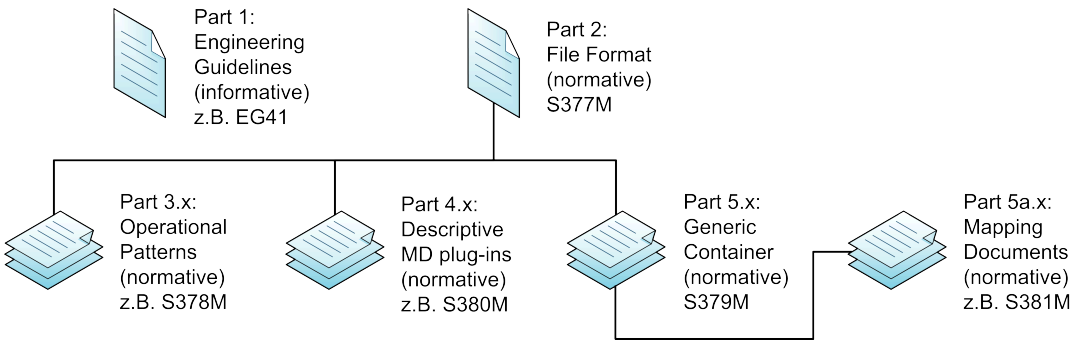


Abbildung 5.5: Strukturierung der MXF-Standarddokumente nach [EG41] (vergleiche Tabelle A.1 im Anhang)

Das MXF-Dateiformat ist die physische Implementierung eines (logischen) Datenmodells. Der mitunter komplexe Aufbau einer MXF-Datei und die damit verbundenen Anwendungsmöglichkeiten werden am besten deutlich, wenn sowohl die logische als auch die

[9] kommerziell und frei verfügbar, freie SDKs unterstützen aktuelle MXF-Standard-Entwicklungen erst mit zeitlicher Verzögerung und bieten meist keine Anwendungsunterstützung (Support)

physische Sichtweise auf MXF erläutert werden. Dies soll im Folgenden auf Basis von [Shi04] und [DW04] geschehen. Für eine detaillierte Beschreibung sei auf die korrespondierenden Standarddokumente der SMPTE und [WDW06] verwiesen.

5.2.1 Das MXF-Datenmodell

Im MXF-Datenmodell wird die Zusammensetzung der Datei mithilfe von den logischen Repräsentationen *Package* und *Tracks* beschrieben. Das Datenmodell ist kompatibel zu AAF und wird in Form von so genannten strukturellen Metadaten in der Datei gespeichert (*Header Metadata – HMD*).

Packages

Der logische Verbund *Package* verbindet einzelne Spuren (*Tracks*) aus unterschiedlichen Essenz- (Audio, Video, Daten) sowie beschreibenden Metadaten miteinander und erlaubt deren Synchronisierung. *File Packages* stellen jene Metadaten dar, die sich direkt auf in der MXF-Datei gespeicherte Essenz beziehen (*Input Timeline*). *Material Packages* hingegen repräsentieren die wiedergegebenen Inhalte der MXF-Datei, wenn diese abgespielt wird (*Output Timeline*).

Ein *Track* hat einen Synchronisationspunkt (*Origin*), der zum Ausrichten der Startpositionen der Tracks innerhalb eines *Packages* dient. Jeder *Track* hat genau eine *Sequence*. Damit wird eine Kompatibilität zum Datenmodell des Advanced Authoring Formats (AAF) sichergestellt[10]. Eine *Sequence* wiederum hat ein oder mehrere *Source Clips*. Ein *Source Clip* eines *Material Packages* bezieht sich meist auf einen *Track* in einem *File Package*, das wiederum eine Eins-zu-Eins-Beziehung mit einem *Essence Container*[11] hat. Ein *Source Clip* im *File Package* repräsentiert so einen konkreten Inhalt einer Essenzspur (siehe Abbildung 5.6).

Operational Patterns

Der Standard erlaubt sehr komplexe MXF-Dateien. So können mehrere *File Packages* mit verschieden langen *Tracks* verwendet werden, die von einfachen (Hardware-)Komponenten nicht verarbeitet werden können[12]. Um dennoch einen interoperablen Austausch über MXF zu ermöglichen, muss die Komplexität des Aufbaus einer MXF-Datei limitiert werden. Zu diesem Zweck wurden *Generalized Operational Patterns* definiert (SMPTE

[10] Zero Divergence Doctrine
[11] ein Essence Container enthält die physisch gespeicherte Essenz
[12] z.B. Sony e-VTR, einer IMX-MAZ, die über eine Netzwerkschnittstelle auf dem Band gespeicherte Audio- und Videoessenzen im IMX-Format als MXF-File ausgeben kann (und umgekehrt)

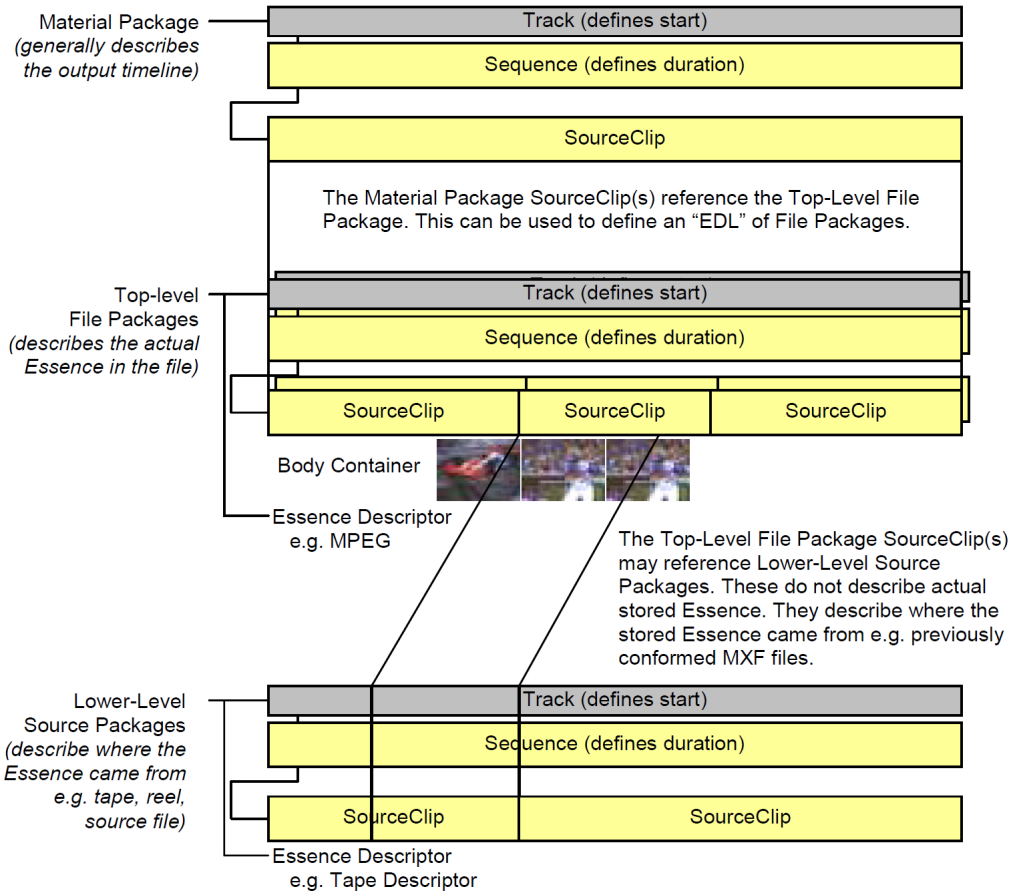


Abbildung 5.6: Beziehungen zwischen den verschiedenen Package-Arten [EG41]

S377M), die in jeweils eigenen Standarddokumenten genauer beschrieben werden (siehe Anhang).

In der Abbildung 5.7 werden neun verschiedene *Operational Patterns* abgegrenzt, die sich aus je drei Möglichkeiten in zwei Freiheitsgraden ergeben. Horizontal wird die *Item Complexity* abgetragen: *Single Item* (OP1x) erlaubt nur ein geschlossenes also nicht segmentiertes *Material Package*, das die gleiche Dauer hat, wie das oder die zugehörige(n) *File Package(s)*. *Play-list Item* (OP2x) erlaubt mehrere *Material Package* Segmente, die aber jeweils vollständig von einem *File Package* referenziert werden. Bei *Edit Items* (OP3x) schließlich dürfen *File Packages* auch länger als referenzierende *Material Package* Segmente sein (*random access*, nur harte Schnitte für Video). In vertikaler Richtung wird mit der (*Package Complexity)* die Anzahl zu einer Zeit aktiven *File Packages* definiert: *Single Package* (OPxa) erlaubt nur ein *File Package*, bei *Ganged Packages* (OPxb) darf sich das *Material Package* gleichzeitig aus mehreren (auch externen) *File Packages* zusammensetzen. Mehrere alternative *Material Packages* definiert die dritte Zeile der Matrix

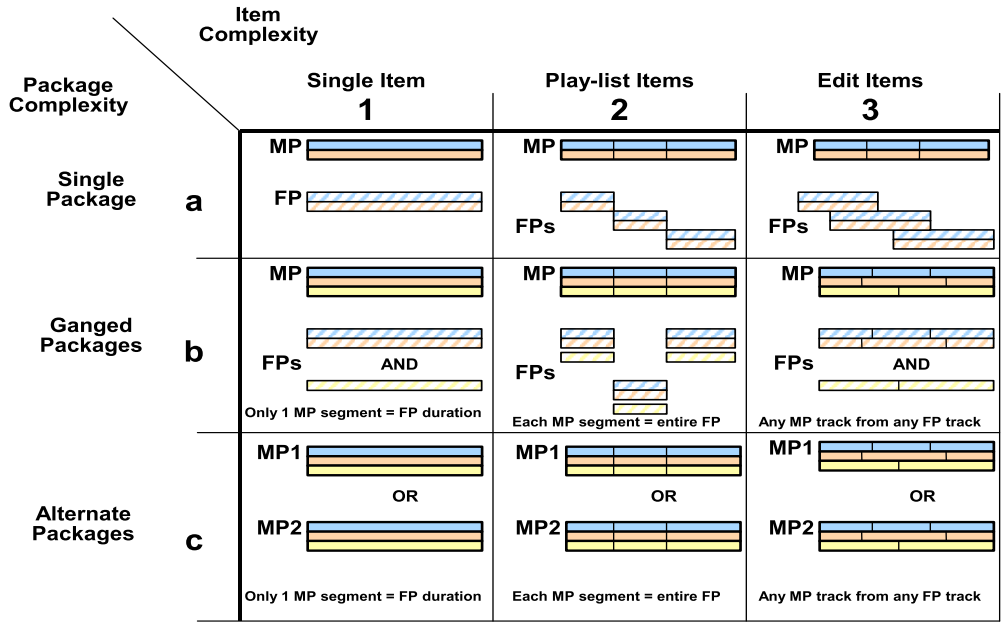


Abbildung 5.7: Operational Patterns [EG41]

(*Alternate Packages*, OPxc). Für Streaminganwendungen scheiden prinzipbedingt alle OPs mit *Edit Items* (OP3x) und mit *Alternate Packages* (OPxc) aus.

Beschränkungen können auch in Form von *Spezialized Operational Pattern* definiert werden. So wurde ein „*OP Atom*“ standardisiert, bei dem eine MXF-Datei nur eine Essenzform (Audio oder Video) beinhalten darf (SMPTE S390). So entfällt z.B. für ein NLE-System[13] die Notwendigkeit des Demultiplexens und erneuten Multiplexens bei der Bearbeitung der Audio- und Videoessenzen.

Der MXF-Standard erlaubt die Referenzierung von internen und externen File-Packages, d.h. Essenzen eines Tracks können sich auch in anderen Dateien befinden. Die Kopplung erfolgt dabei über den *Unique Material Identifier* (UMID). Eine Anwendung stellt das MXF-Master(ing)-Format dar, bei dem eine übergeordnete MXF-(Master)-Datei externe Referenzen auf weitere MXF-Dateien beinhaltet, die z.B. alternative Sprachversionen zu einer Videosequenz repräsentieren können.

Unique Material Identifier – UMID

Mithilfe des *Unique Material Identifiers* kann eine weltweit eindeutige Kennzeichnung von Content implementiert werden (SMPTE S330M). Ein UMID wird lokal erzeugt, ist

[13] Non Linear Editing System – nichtlineares Videoschnittsystem, meist auf Computerbasis

aber global einzigartig. Die ersten zwölf Byte des UMID stellen ein *Universal Label* dar, das über SMPTE *Metadata Dictionary* nach SMPTE RP 210 den UMID identifiziert. Das Längenfeld gibt die Länge der nachfolgenden Information an und unterscheidet somit zwischen *Basic* und *Extended UMID*. Um Kopien des Materials oder nur Teile davon zu kennzeichnen, wird eine *Instance Number* verwendet, die bei Bedarf entsprechend inkrementiert wird. Die nachfolgende *Material Number* enthält die eindeutige Kennzeichnung des Quellmaterials, die aus diversen Informationen (z.B. Generierungszeitpunkt, Netzwerkadressierung des erzeugenden Gerätes, Zufallszahlen) berechnet wird.

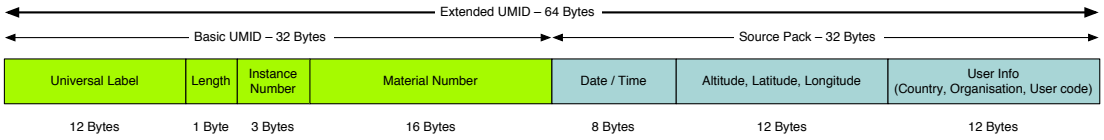


Abbildung 5.8: Aufbau des Extended UMID [Shi04]

Bei Bedarf kann ein *Basic UMID* um ein *Source Pack* zu einem *Extended UMID* erweitert werden. Die ersten acht Bytes des *Source Packs* ermöglichen eine frame- bzw. fieldgenaue Adressierung der Essenz. Die nächsten zwölf Bytes kennzeichnen einen Ort mithilfe der Höhen-, Längen- und Breitengrade, die z.B. über einen an die Kamera angeschlossenen GPS[14]-Empfänger aquiriert werden können. Die letzten zwölf Bytes erlauben Nutzerinformationen im Sinne alphanumerischer Angaben für Land, Unternehmen und beliebige Anwendercodes.

Deskriptoren

Packages enthalten im Wesentlichen Informationen zum zeitlichen Ablauf. Deskriptoren (*Descriptors*) können verschiedene Einträge zu technischen Parametern, z.B. Samplingfrequenz, gespeicherte Bildauflösung oder Anzahl der Audiospuren, enthalten. Abbildung 5.9 gibt einen Überblick über existierende Deskriptoren und stellt sie in ihrer Vererbungshierarchie dar. Aus dem an der Spitze des „Vererbungsbaums“ stehenden *Generic Descriptor* können alle anderen Deskriptoren abgeleitet werden. Der *File Descriptor* findet als Grundlage für alle spezielleren *Essence Descriptors* Anwendung und enthält Parameter über die gespeicherte Essenz. Ein *Physical Descriptor* beschreibt die ursprüngliche Herkunft von Essenzen oder vorangegangene Transcodierungen.

Angaben zum physikalischen Speicherort einer Essenz werden über einen *Locator* gemacht, der im jeweils verwendeten Deskriptor eingetragen wird. Ein *Network Locator* kennzeichnet einen Speicherort per URL[15]. Ein *Text Locator* erfasst den physikalischen

[14] Global Positioning System
[15] Uniform Resource Locator - identifiziert eine Ressource über das verwendete Netzwerkprotokoll und den Ort der Ressource in Computernetzwerken

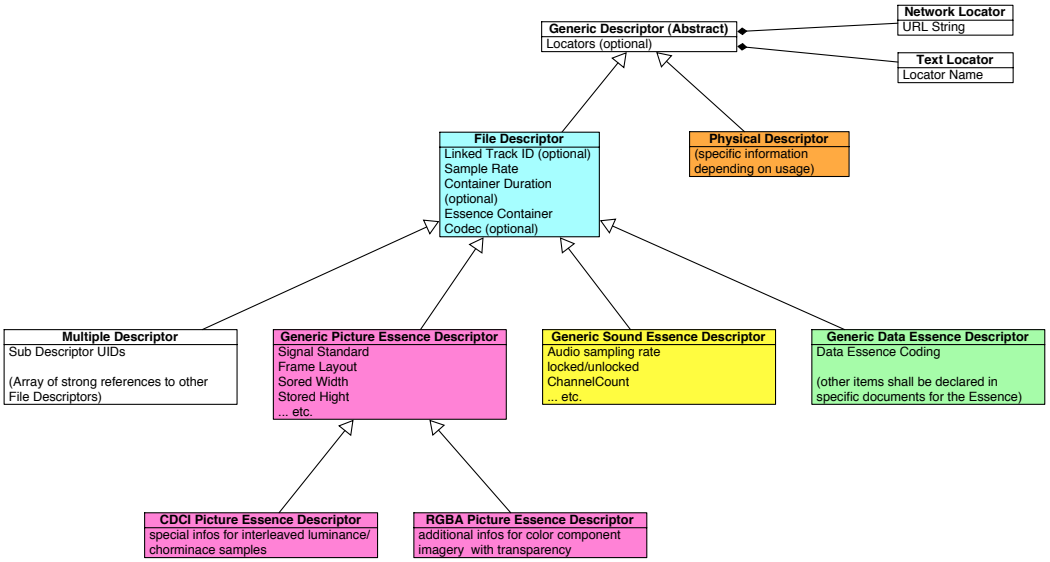


Abbildung 5.9: Vererbunghierarchie der Deskriptoren [Shi04]

Lagerort mithilfe herkömmlicher Textinformation (z.B. Kassettenname und Regalnummer).

Essence Container und Generic Container

Für die Kapselung von Essenztypen (z.B. DV-DIF, MPEG usw.) werden bei MXF *Essence Container* verwendet. Im MXF-Standard selbst ist ein *Generic Container* und dessen Einbindung in ein MXF-File definiert (SMPTE 379M). In weiteren Dokumenten wird das Mapping einzelner Essenztypen in den *Generic Container* standardisiert (S381M: MPEG Streams, 383M: DV-DIF usw.) und somit eine flexible Erweiterbarkeit des MXF-Standards für neue Essenztypen erreicht.

Abbildung 5.10 stellt den *Generic Container* als komplexe Struktur dar. Ein *Essence Container* enthält danach ein oder mehrere *Content Packages*. Ein *Content Package* umfasst den Dateninhalt für eine Zeitdauer von einem Frame (bei *frame based wrapping*) oder von einem gesamten Clip (bei *clip based wrapping*). Ein *Content Package* besteht aus bis zu fünf unterschiedlichen *Content Items*, die Datenströme einer Essenzart zusammenfassen. Jedes *Content Item* besteht aus bis zu 127 *Content Elements*. Ein *Content Element* stellt dabei eine unteilbare, zu einem Track gehörige Einheit dar. Folgende *Content Elements* können in den jeweiligen *Content Items* enthalten sein:

- *Picture Element*: ein Video-Frame oder -Field
- *Audio Element*: Audio-Samples eines *Tracks* für einen Kanal

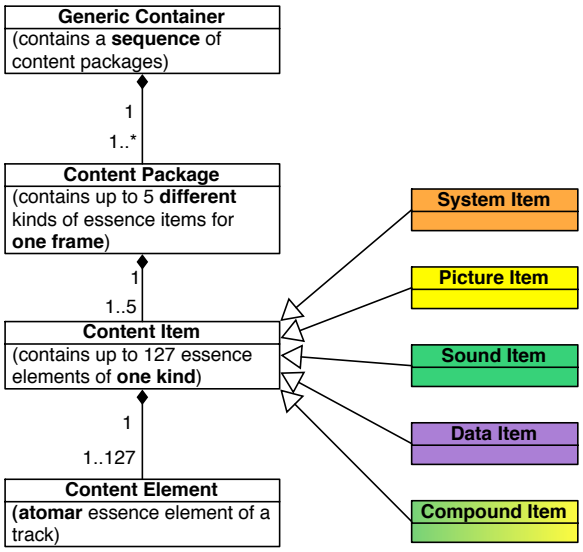


Abbildung 5.10: UML-Diagramm für den Aufbau eines Generic Container [Shi04]

- *System Element*: Meta- bzw. Steuerdaten-Elemente (auch Timecode), die mit der restlichen Essenz zeitabhängig aufgezeichnet wurden
- *Data Element*: sonstige Datenelemente wie Videotext oder Untertitel
- *Compound Element*: Multiplex aus A/V- und Metadaten (z.B. DV-DIF)

Jedes *Content Element* des *Essence Containers* wird physisch als KLV[16]-Tripel gespeichert.

5.2.2 Die physische Realisierung des MXF-Datenmodells

Der potentiell komplexe Aufbau einer MXF-Datei muss nun in einem seriellen Datenstrom abgebildet werden, der als Datei gespeichert werden kann. Zur Beschreibung des Vorgangs soll zunächst der prinzipielle Aufbau einer MXF-Datei erläutert werden, bevor auf die Kodierung des Bitstromes eingegangen wird.

Dateiaufbau

Im einfachsten Fall besteht eine MXF-Datei aus einem *File Header* und einem *File Footer* (siehe Abbildung 5.11). Prinzipiell beherbergt der *File Header* die kodierte logische Struktur der Datei (in den Header Metadaten) sowie weitere, z.T. optionale Elemente, die den Zugriff auf die Essenzinformationen erleichtern können (*Index Tables*, *Run In*,

[16] Key-Length-Value-Kodierung, siehe Abschnitt 5.2.2

usw.). Weiterhin kann der *File Header* bereits Essenz in Form eines *Essence Containers* enthalten. Sobald die Unterteilung der Essenzinformation in mehrere Partitionen sinnvoll ist (z.B. *frame based wrapping* für Streaming), wird diese im *File Body* abgespeichert (siehe Abbildung 5.12).

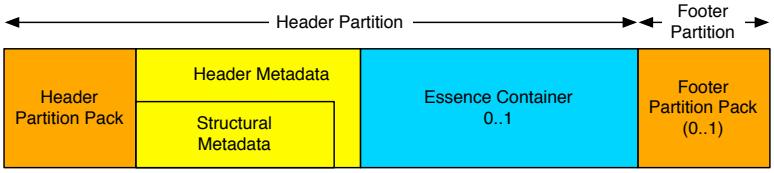


Abbildung 5.11: Physische Struktur einer einfachen MXF-Datei [Shi04]

Partitionen

Durch Partitionen wird der Inhalt einer MXF-Datei in einen seriellen Datenstrom überführt. Sie dienen dabei als Synchronisationseinheiten. Für eine Streaming-Anwendung muss so für hinreichend kleine Partitionen gesorgt werden, damit die Verzögerungszeit beim Aufschalten eines neuen Empfängers möglichst gering bleibt.

Drei verschiedene Arten von Partitionen sind definiert [Shi04]:

- **Header Partition**, die den Anfang einer MXF-Datei und Ihrem KLV-Multiplex kennzeichnet und bereits einen *Essence Container* beinhalten kann.
- **Body Partition**, die Essenz- und/-oder Metadaten einbettet. Bis zu 2^{64-1} *Body Partitions* sind pro MXF-Datei zugelassen.
- **Footer Partition**, die den Abschluss einer MXF-Datei kennzeichnet und ggf. vollständige Metadaten enthält. In einigen *Operational Patterns* kann diese Partition entfallen.

Alle Partitionen beginnen mit einem *Partition Pack*, der wichtige Informationen über die Partition enthält. So finden sich dort Partitionsart (Header/Body/Footer), Status (open/closed, incomplete/complete), *Operational Pattern* oder *KAGSize* (s.u.). Danach folgen Header Metadaten, Indextabellen und *Essence Container* (siehe Abbildung 5.12), wobei jede Partition max. einen *Essence Container* enthält.

Der Partitionsstatus bezieht sich auf die Vollständigkeit von entsprechenden Metadaten: Sind alle zwingend benötigten Metadaten vorhanden und korrekt, ist eine Partition geschlossen (*closed*), anderenfalls offen (*open*). Während der Dateierzeugung noch unbekannte Metadaten (*best effort elements*) werden auf einen definierten Wert gesetzt und die Partition als *incomplete* gekennzeichnet. *Complete* ist eine Partition mit korrekten Werten für alle *best effort elements*. Ein Beispiel für ein *best effort element* ist die Dauer (*duration*), die mit dem Wert „-1“ als unbekannt gekennzeichnet wird [WDW06, S. 72].

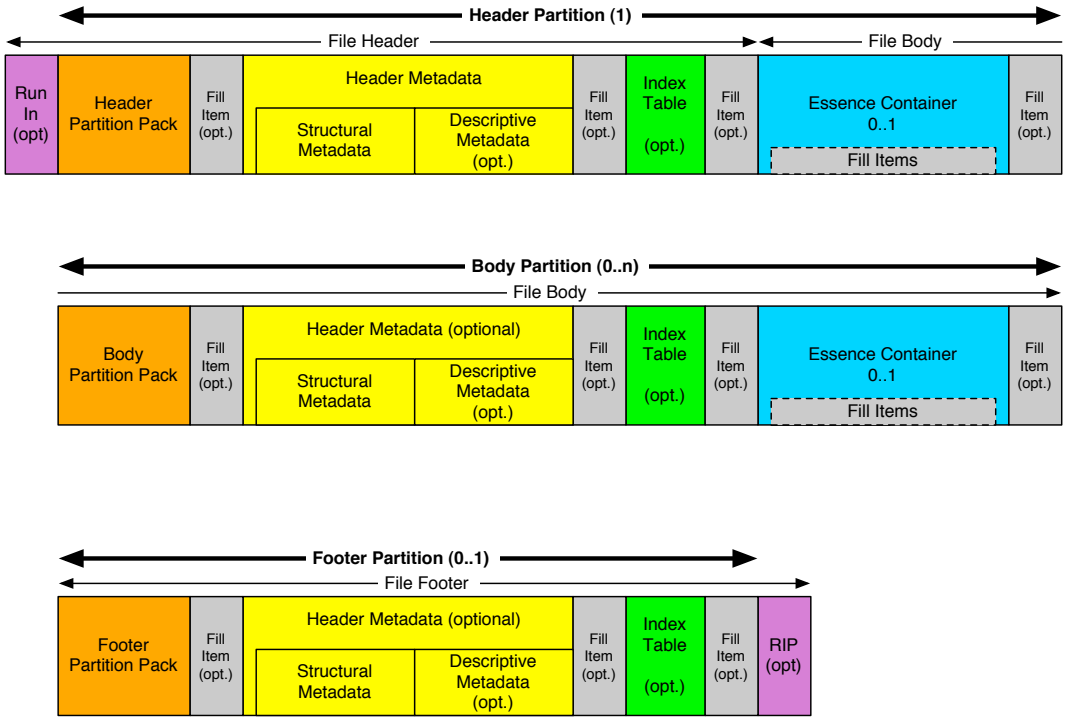


Abbildung 5.12: Physische Struktur einer MXF-Datei mit allen Optionen [Shi04]

Kodierung des Bitstromes: KLV

Der Bitstrom einer MXF-Datei wird mithilfe der *Key-Length-Value*-Kodierung kodiert, die im SMPTE Standard S336M definiert ist. Diese Kodierung kapselt verschiedene Dateneinheiten und ist unabhängig von Anwendung, Speichermedium oder verwendetem Transportprotokoll. Aus folgenden Elementen wird ein Daten-Tripel definiert:

- **Key**: Ein 16 Byte langer Schlüssel identifiziert enthaltene Nutzdaten durch ein Wörterbuch, das eine Liste mit Schlüsseln und Datenattributen enthält (zentrales SMPTE *Metadate Dictionary* oder lokale Kopie).
- **Length**: Wert gibt die Länge der enthaltenen Nutzdaten in Byte an
- **Value**: Hier werden die eigentlichen Nutzdaten abgelegt (z.B. für Essenzdaten ein *Essence Container*).

Daten, deren Schlüssel für einen Decoder unbekannt sind, weil diese nicht in seinem *Dictionary* stehen, können mithilfe der Längeninformation übersprungen und damit ignoriert werden (*Dark Data*). Diese Tatsache ermöglicht eine offene Erweiterung der Funktionalität von MXF, ohne die Leistungsfähigkeit einzelner MXF-Systeme einzuschränken [Hön02].

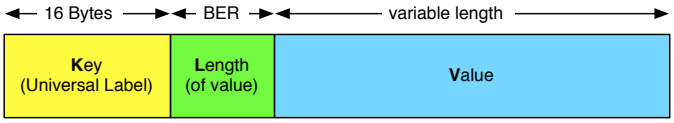


Abbildung 5.13: Key-Length-Value-Kodierung [Shi04]

Als Value können weitere KLV-Tripel eingebettet sein. Damit werden einfache Hierarchien und Gruppierung möglich. Mit dem Ziel einfacher (Hardware-)Implementierungen wurde die Verschachtelungstiefe im MXF-Standard auf zwei KLV-Ebenen eingeschränkt.

5.2.3 Metadaten in MXF

Im MXF-Kontext werden zwei Arten von Metadaten unterschieden. Während die strukturellen Metadaten (*structural metadata*) den Inhalt der MXF-Datei über das MXF-Datenmodell beschreiben, bieten beschreibende Metadaten (*decriptive metadata* – DM) die Möglichkeit, Klartext-Annotationen zur Essenz zu ergänzen. Strukturelle Metadaten werden von einer Applikation zur Dekodierung des Datenstromes benötigt und sind deshalb vorzugsweise maschinen-lesbar. Dazu zählen auch technische Metadaten wie Abtastfrequenzen, Bild-Seiten-Verhältnisse und verwendete Kodieralgorithmen. Beschreibende Metadaten sind für die Nutzung durch den Anwender vorgesehen, z.B. Angabe der Produktionsnummer, Name des Kameramanns oder Titel einer Sendung. Diese produktionsbegleitenden bzw. produktionsrelevanten Metadaten sind essentiell für Archivierung, Dokumentation und Recherche von Material.

Um beschreibende Metadaten mit strukturellen Informationen zu verknüpfen, werden sie als *DM Tracks* in einem *Package* angelegt („Plug-In-Mechanismus“). Ein *DM Track* in einem *Material Package* bezieht sich auf den Inhalt der *Output Timeline* der MXF-Datei, ein *DM Track* in einem *Source Package* beschreibt den Inhalt einer intern oder extern gespeicherten Essenz.

Folgende drei Arten von *DM Tracks* werden im MXF-Standard unterschieden:

- *DM Static Track*: beschreibt den gesamten Track, enthaltene Metadaten sind somit über die gesamte Dauer statisch
- *DM Timeline Track*: enthält eine lückenlose Sequenz von DM Segments ohne Überlappungen (durch Startposition und Dauer in ihrer Gültigkeit zeitlich beschränkt)
- *DM Event Track*: *DM Segments* können beliebige Position und Dauer erhalten (inkl. gleichzeitige „Events“ und Dauer „0“)

Für einen standardisierten interoperablen Austausch dieser Informationen sind Metadatenmodelle wie DMS-1 oder BXF vorgesehen (siehe Abschnitt 5.1.2).

Bezüglich der Speicherung von Metadaten bei der Nutzung im Produktionsprozess bestehen, unabhängig vom Einsatzzweck der Metadaten, zwei grundsätzliche Möglichkeiten. Einerseits können Metadaten mit der Essenz übertragen und gespeichert werden (dezentraler Ansatz); andererseits kann die Verwaltung der Metadaten zentral (z.B. in einer Datenbank) erfolgen. Die gemeinsame Übertragung von Meta- und Essenzinformation bedingt einen geeigneten Synchronisationsmechanismus, den das MXF-Dateiformat zur Verfügung stellt. Weiterhin ist die Speicherung von Essenz- und Metainformationen in getrennten Dateien möglich. Die Synchronisierung kann auch hier mithilfe eindeutiger Identifikatoren (z.B. UMID) und mithilfe externer Referenzen erfolgen.

Vorteil der dezentralen Variante ist, dass die Konsistenz von Meta- und Essenzinformation einfach zu bewerkstelligen ist. Weiterhin liegen die für die Essenzverarbeitung notwendigen Metadaten direkt vor, was die Steuerung vereinfacht und den Content auch ohne Vorhandensein einer übergeordneten Datenbank nutzbar macht. Eine Suche über Metadaten vieler verschiedener Essenzen ist aber aufwändig, da jede einzelne Content-Einheit danach durchsucht werden muss.

Das Argument der effizienten Suche favorisiert den zentralen Ansatz, wenn die Informationen offline, also nicht im echtzeitkritischen Zusammenhang, vorliegen. Insbesondere im Live-Produktionsbereich wird Essenz durch zeitkontinuierliche Datenströme repräsentiert. Die zugehörigen Metadaten müssen weder im herkömmlichen Sinne suchbar noch von zentraler Stelle aus manipulierbar sein, solange der Datenstrom noch nicht in Form einer Datei gespeichert ist. Der dezentrale Ansatz bietet hier weiterhin den Vorteil der einfachen Speicherung von Metadaten ohne eine zentrale Instanz, während der Datenstrom einzelne Stationen linear durchläuft.

Beide Ansätze können unter Nutzung des MXF-Dateiformates vereint werden. MXF erlaubt über externe Referenzen die Abspeicherung von Essenz- und Metadaten in separaten Dateien und prinzipiell auch in Datenbanken. Je nach Anwendungsfall kann so der optimale Ansatz im echtzeitkritischen Bereich gewählt werden. Nach der Speicherung des Datenstreams als Datei auf einem Datenserver ist eine zentrale Speicherung der Metadaten sinnvoll, um die Arbeitsabläufe der Postproduktion zu unterstützen.

5.2.4 MXF-Streaming

Der Begriff „Streaming“ wird normalerweise im Zusammenhang mit der isochronen Übertragung von zeitabhängigen Medien benutzt. Der MXF-Standard definiert im Wesentlichen eine Datei-Spezifikation; dementsprechend wird bezüglich der Anwendung von MXF unter zeitlichen Bedingungen von *streamable files* gesprochen. Eine „streaming-fähige“ MXF-Datei erlaubt die Verwendung (z.B. Anzeige) von bereits übertragener Information, während der Dateitransfer noch nicht abgeschlossen ist (*read while write)*. Unter

der Voraussetzung, dass das darunter liegende Transportnetzwerk entsprechende Dienste anbietet, kann mit „MXF-Streaming“ eine synchrone Datenübertragung auf Dateibasis umgesetzt werden.

Der Aufbau einer streaming-fähigen MXF-Datei muss in der Komplexität begrenzt werden, wenn eine hinreichend kurze Verzögerungszeit durch kleine Puffer am Empfänger realisiert werden soll. Bezüglich der Zusammensetzung der Datei sollte *frame based wrapping* Anwendung finden, d.h. der *File Body* sollte eine Folge von kleinen Partitionen, die jeweils ein *Content Package* enthalten, das wiederum jeweils die Daten eines Frames enthält [WDW06, S. 66]. Weiterhin sollte „Input-Timeline = Output-Timeline“ gelten. Somit ist für Streaming-Anwendungen das Operational Pattern OP1a prädestiniert. Daneben existieren weitere Richtlinien, die eine einfache Implementierung des Empfängers ermöglichen und damit einer weiteren Minimierung der Latenz zuträglich sind [EG41].

Trotz expliziter Erwähnung in den Standarddokumenten existieren aktuell nur wenige Implementierungen, die die dort beschriebenen Möglichkeiten des MXF-Streaming umsetzen. Die Standardisierung der Kapselung von MXF in IP wird zur Zeit von einer Arbeitsgruppe der EBU angeregt. Nach einer Manifestierung in einem Request for Comment (RFC) bei der Internet Engineering Task Force (IETF) sollen die Erkenntnisse an den SMPTE-Standardisierungsprozess übergeben werden.

5.3 MXF-Anwendungsszenarien im Studio

In den folgenden Abschnitten sollen Anwendungspotentiale für Metadaten im Live-Studiobetrieb identifiziert werden, um daraus Rückschlüsse auf das Konzept einer IT-Netzwerk-basierten Architektur zu ziehen.

5.3.1 Anpassung an weitere Distributionskanäle

Eine wesentliche Motivation der Anwendung von flexibler Standard-IT-Technologien im Fernsehstudio besteht in der möglichst automatischen Anpassung von TV-Content an die Anforderungen weiterer Distributionskanäle, wie z.B. mobile-TV[17]. Neben der Weiterverwertung von umfangreich in den Archiven vorhandenem Content (in der Postproduktion) sollte zukünftig schon die Content-Produktion simultan für möglichst viele Distributionskanäle möglich sein.

Die produktionsseitigen Anforderungen an einen Distributionskanal ergeben sich aus dem Nutzungskontext. Bei mobile-TV beispielsweise ist die Größe des verwendeten Bildschirmes und damit die örtliche Auflösung der Videoessenz begrenzt. Weitere technische Einschränkungen des Distributionskanals, wie z.B. die mögliche Datenübertragungsraten

[17] Fernsehen auf tragbaren Geräten mit entsprechend kleinen Bildschirmen

vom Sender zum mobilen Empfänger, verwendete Audio- und Videokompressionsalgorithmen sowie begrenzte Akkulaufzeiten können zusätzlich Parameter wie die zeitliche Auflösung (Bildwiederholfrequenz) oder Farbauflösung (Anzahl der Quantisierungstufen pro Farbkanal) beeinflussen. Eine Reihe von Parametern ist bereits bei der Aufnahme des Contents zu berücksichtigen (Aufnahmeperspektive, Geschwindigkeit von Kamerabewegungen), andere können im Rahmen einer Bearbeitung auf die Anforderungen von Distributionskanälen angepasst werden (z.B. Bildausschnitt). Grundlage für die Anpassung nach der Aufnahme sind Metadaten, die Essenzen in der Weise beschreiben, dass Distributionskanal-Parameter automatisch berücksichtigt werden können. Am Beispiel der Anpassung der örtlichen Auflösung soll im Folgenden die Anwendung von Metadaten im Studiobereich verdeutlicht werden.

Ausgangspunkt der Betrachtungen ist ein hochauflösendes Breitbild-Videosignal[18] als Grundlage für ein HDTV-Sendesignal. Die Nutzung dieses Ausgangssignals für die Distribution auf kleine Bildschirme mobiler Empfangsgeräte setzt eine Anpassung der örtlichen Auflösung auf z.B. 320 × 240 aktive Bildpunkte voraus. Bei der einfachen Skalierung können Bilddetails verloren gehen. Zudem entstehen bei der Anpassung des Bildseitenverhältnisses Verzerrungen oder Bildinhalte gehen verloren, indem Randbereiche beschnitten werden. Optimaler ist die Festlegung eines sinnvoll auf kleinen Bildschirmen darstellbaren Bildausschnittes innerhalb des HD-Videobildes. Dieser Bildausschnitt wird dabei durch Position und Größe eines Rechtecks beschrieben, welches durch zwei Koordinaten definiert wird. Diese Koordinaten können in Form von Metadaten einfach abgespeichert werden (siehe Abbildung 5.14).

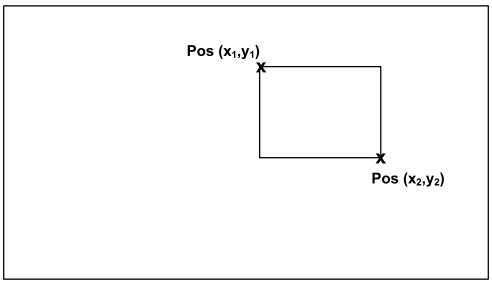


Abbildung 5.14: Bildausschnittswahl zur Anpassung von TV-Content an alternative Distributionskanäle am Beispiel der Berücksichtigung kleiner Bildschirme bei mobile-TV

Position und Größe des für kleine Bildschirme relevanten Teils der hochauflösenden Bildinformation dienen später dazu, einen Encoder für die mobile-TV-Distribution zu steuern. Auch die Unterstützung weiterer Distributionskanäle ist darauf aufbauend realisierbar.

In ähnlicher Weise können Textinformationen für Bauchbinden und Einblendungen zunächst als Textinformationen mit der Essenz verknüpft übertragen und erst am Encoder

[18] 16:9 Bildseitenverhältnis, 1920 × 1080 aktive Bildpunkte

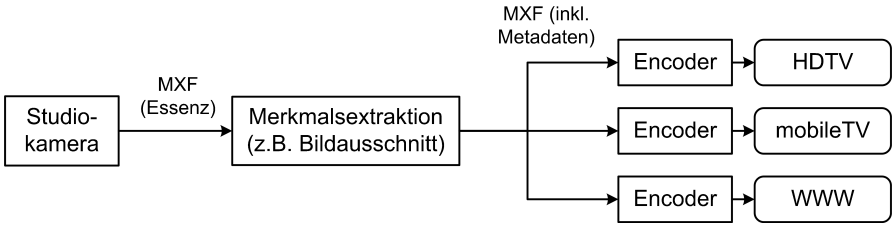


Abbildung 5.15: Prinzip der Anpassung von TV-Content an alternative Distributionskanäle in der Produktion

unter Anwendung vorgefertigter Templates (z.B. für Schriftarten und -größen) in das Sendebild integriert werden. Der Encoder kann in diesem Fall ein *Virtual Set Sytem* sein, dass über entsprechende Schnittstellen offen und erweiterungsfähig ist.

5.3.2 Mehrfachnutzen von Textinformationen

Insbesondere bei Nachrichtenproduktionen kommt Informationen in Textform eine große Bedeutung zu. Die Ansagen einer Nachrichtensendung werden im Vorfeld redaktionell erstellt und während der Sendung live von einem Sprecher[19] verlesen. Traditionell benutzen die Sprecher dafür ausschließlich einen Ausdruck auf Papier, mit der Verbreitung von Teleprompter-Systemen[20] kommen auch elektronische Textrepräsentationen zum Einsatz.

Die Sprechertexte in elektronischer Form können in vielfältiger Weise wiederverwendet werden – sowohl in „Echtzeit“ während einer Livesendung, als auch in der Postproduktion für eine Aufzeichnung. Die Erzeugung eines Untertitels für Hörgeschädigte ist ein Beispiel für die Anwendung während einer Livesendung. Die Nutzung eines Teleprompter-Systems erlaubt die automatische Synchronisierung der Textinformation mit der aufgenommenen Audio- und Videoessenz. Als Datenstrom einer MXF-Datei kann der Sprechertext synchron übertragen werden und im System für die Untertiteleintastung automatisch wiederverwendet werden. Auch für die selbstgesteuerte Generierung von Teletext- und Webseiten kann der Sprechertext verwendet werden. Weiterhin sind die automatische Erstellung von Newsletter-Emails und Broadcastdiensten für mobile Endgeräte wie *Smartphones* denkbar. Schließlich bietet die mit den Essenzdaten verknüpfte Sprechertextinformation eine Datenbasis, die über eine Volltextsuche das Auffinden von Beiträgen im Archiv erleichtern kann. Die Nutzung von Sprechertexten für die automatische Generierung von Webseiten wurde bereits prototypisch umgesetzt [Möb07].

[19] mit dem Ziel der kurzen Schreibweise werden im Folgenden alle Tätigkeitsfelder mit der männlichen Bezeichnung versehen, die in Realität gleichberechtigt von Frauen und Männern ausgeübt werden

[20] Ein Teleprompter spiegelt die Bildschirmausgabe eines PCs in den Strahlengang einer Studiokamera ein, so dass es für den Moderator vor der Kamera sichtbar wird, ohne von der Kamera aufgenommen zu werden. Die Bildschirmausgabe gibt den Sprechertext wieder, der sich in einer beeinflussbaren Geschwindigkeit über den Bildschirm bewegt.

5.3.3 Virtual Set Anwendungen

Die in MXF-Dateien und -Datenströmen enthaltenen Metadaten können dazu genutzt werden, Abläufe im Virtuellen Studio zu steuern. Während der Produktion können diese Metadaten weiterhin durch das Virtual Set System generierte Informationen strukturiert aufnehmen. Diese können dann in der Nachbearbeitung, Archivierung oder Wiederverwertung des Materials verwendet werden. Die gespeicherten Informationen können dabei in einem festen Bezug zum Timecode des Materials stehen oder sich auf die komplette Produktion beziehen.

Ein naheliegendes Einsatzgebiet ist die Speicherung der Trackingdaten der Studiokamera zusammen mit ihrem Bildmaterial. Dadurch besteht immer ein fester zeitlicher Bezug der Trackingdaten zum Bild und alle zusammengehörigen Informationen liegen gemeinsam in einer einzigen Datei vor. So lässt sich durch die Reduzierung des Verwaltungsaufwandes die Effizienz steigern und gleichzeitig die Fehlerquote reduzieren.

Metadaten in MXF-Files können außerdem dazu verwendet werden, Ereignisse im Virtual Set zu steuern. So lassen sich beispielsweise Schrift- oder Grafikeinblendungen zu einem Einspieler einer Nachrichtensendungen automatisch und zum korrekten Zeitpunkt aus den im MXF-File enthaltenen Daten generieren. Umgekehrt ist mit MXF aber auch die Aufzeichnung solcher Events, die während der Produktion vom Virtual Set System generiert werden, möglich. Bei der Wiederverwertung des Programmmaterials lassen sich so nachträglich entsprechende Einblendungen (z.B. Informationen, die nur während einer Livesendung von Bedeutung sind) aktivieren oder deaktivieren.

Ebenfalls möglich ist die Abbildung rein redaktioneller Informationen auf Metadaten, auf deren Grundlage für bestimmte Genres ein automatisierter Sendeablauf ermöglicht wird. Für eine Magazinsendung ist z.B. ein vorbereiteter Ablauf vorstellbar, der Start und Dauer von Zuspielungen und Sprecherpassagen sekundengenau festlegt und damit eine automatisierte Produktion mit weniger Personal ermöglicht.

5.3.4 Interaktives Fernsehen

Ähnliche Anwendungsszenarien für MXF wie in der Virtuellen Fernsehproduktion können für das Interaktive Fernsehen identifiziert werden. Aus den enthaltenen Metadaten lassen sich unter Einsatz eines entsprechenden Frameworks automatisch Applikationen für das Interaktive Fernsehen generieren. So kann z.B. ein seitenbasiertes Informationssystem aus Standbildern, Texten und Templates, die in den MXF-Dateien abgelegt wurden, Zusatzangebote zum normalen Fernsehbild erzeugen. Durch die Verwendung entsprechend angepasster Vorlagen ist auf diese Weise die gleichzeitige Bedienung unterschiedlicher Zielplattformen, wie der *Multimedia Home Platform* (MHP), Internet Streams (SMIL - *Synchronized Multimedia Integration Language*), Handheld Applikationen (UMTS, DVB-H, DXB) usw., möglich.

Ein weiteres Einsatzgebiet für MXF existiert in der Archivierung interaktiver Sendungen. Die zu einer Sendung gehörenden MHP-Applikationen lassen sich in einer einzigen Datei zusammen mit dem Audio- und Videomaterial der Sendung archivieren. Bei der wiederholten Ausstrahlung kann die Applikation automatisch extrahiert und mit den erforderlichen Synchronisationsinformationen an den MHP Playoutserver weitergegeben werden. Auch hier können so Fehler reduziert und die Effizienz erhöht werden.

Mit der zu erwartenden Erhöhung der Leistungsfähigkeit von Endgeräten ist es denkbar, dass bestimmte Rechenvorgänge, die momentan aufgrund der benötigten Rechenleistung noch sendeseitig im Fernsehstudio erfolgen, in Zukunft auf Empfängerseite bewältigt werden. Dies setzt vor allem voraus, dass Metadaten aus dem Produktionsprozess über Sendergrenzen hinweg einfach und ohne Mehraufwand zum Empfänger gelangen. Möglich ist z.B. das Rendering von 3D-Szenerien für Virtual Set Anwendungen auf der Settop Box beim Empfänger. Dabei kann für die Distribution die objektorientierte Beschreibung (Szenengraph) des MPEG-4-Standards genutzt werden.

Lugmayr beschreibt die Anwendung bei regionalsierten Werbeeinblendungen. Neben der Videobildinformation werden dabei Positionsdeskriptoren an die Settop-Box des Rezipienten gesendet, die dort für die Einblendung von Werbeinformationen zur Verfügung stehen. [LNK04, S. 185 f.]

5.3.5 Nichttechnische Anwendungen

Ein weiterer in der Praxis komplexer Vorgang kann mithilfe von Metadaten deutlich vereinfacht werden: Rechteverwaltung und Honorarabrechnung. Diesbezügliche Aktivitäten können besonders gut mit dem Metadatenmodell BMF abgebildet werden. Die selbstgesteuerte Protokollierung von Tätigkeiten bei der technischen Abwicklung der Produktion kann weiterhin zu einer schnellen und genauen Abrechnung von Honorarkräften genutzt werden, da diese Informationen über Schnittstellen in Business-Software-Systeme wie SAP importiert werden können. Über die Produktionsphase hinaus stellen diese Informationen eine Datenbasis für Workflow-Management-Systeme dar, die Abläufe in nachgelagerten Bereichen, z.B. der Postproduktion, effizienter gestalten können.

5.4 Konklusion

Auf Basis von Metadaten können auch im echtzeitkritschen Studiobereich Arbeitsschritte selbstgesteuert und damit effizient ausgeführt werden. Neben vielen technischen Anwendungsfällen können Metadaten auch organisatorische Abläufe unterstützen.

Voraussetzung dafür ist eine robuste Referenz der Metadaten auf dazugehörige Essenzdaten (bzw. umgekehrt). Nur so können exakte zeitliche Abläufe und Zuordnungen realisiert werden. MXF stellt Mechanismen dafür zur Verfügung.

Für die Anwendung im Studiobereich unter Live-Bedingungen, wo Datenströme sequentiell in Quasi-Echtzeit durch Be- und Verarbeitungsstationen geleitet werden, ist die Speicherung von Metadaten mit der Essenz zu bevorzugen (dezentraler Ansatz). Erst nach der Speicherung auf einem Datenträger erweist sich die zentrale Metadatenhaltung z.B. in einer Datenbank als sinnvoll.

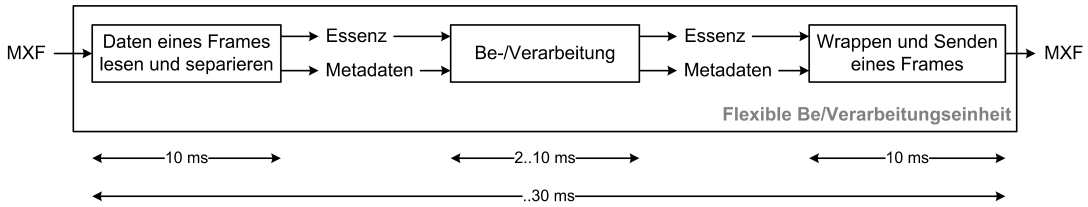


Abbildung 5.16: MXF-basierte Metadatenspeicherung im Studio

Abbildung 5.16 verdeutlicht die Speicherung und Verwendung von Metadaten unter Nutzung von MXF und stellt in dieser Arbeit die Basis für die Einbindung von Metadaten dar. Die beispielhaft angedeuteten Teilprozessdauern sollen zeigen, dass diese Vorgehensweise auch in engen Zeitschranken Anwendung finden kann. Voraussetzung hierfür ist eine gebündelte Content-Übertragung nach dem Konzept des Filestreaming (vergleiche Abschnitt 6.3).

6 Eine Netzwerkarchitektur für Studioanwendungen

Ziel des Gesamtkonzeptes ist eine Studioarchitektur auf Basis eines standardisierten, preiswerten und dennoch robusten Netzwerkes, um die Vielzahl von Essenzdatenschnittstellen (wie SDI und AES/EBU) zu ersetzen. Ein solches generisches Netzwerk soll neben der latenzarmen und synchronen Übertragung von Essendaten vor allem ermöglichen, mit Essenzdaten verknüpfte Metainformation zu transportieren, damit diese jederzeit in den Bearbeitungsknoten als Grundlage für automatisierte Abläufe zur Verfügung stehen. Damit verbunden ist die Anforderung, dass auch Bearbeitungsprozesse in engen zeitlichen Schranken ermöglicht werden.

Daraus folgt die Notwendigkeit von drei Teilkonzepten, die miteinander integriert werden müssen: (1) Ein Konzept zur physikalischen Übertragung, (2) ein Konzept zur Essenzdatenverarbeitung auf Basis von Standardhardware und (3) ein Konzept zur gewinnbringenden Anwendung von Metadaten. (1) und (2) sichern die Funktionalität bisheriger Mechanismen im Studio ab, während (3) eine Erweiterung darstellt und damit die neue Architektur motiviert. Diese drei Teilkonzepte wurden in den vorangegangenen Kapiteln diskutiert und sollen nun zu einem Gesamtkonzept für eine prototypische Implementierung vereint werden. Daraus abgeleitete Designentscheidungen bilden den Anfang der Gesamtsystemkonzeption. Als Grundlage zur theoretischen Verifikation des Gesamtsystems wird im Anschluss daran ein Gesamtlatenzmodell aus den Latenzmodellen für Übertragung und Bearbeitung (vergleiche Abschnitte 3.2 und 4.4) zusammengefügt.

Die Anforderungen an die Datenübertragung und Bearbeitung im Studiobereich lassen sich durch drei Punkte zusammenfassen: Signalqualität, Rechtzeitigkeit (geringe Latenzen) und Synchronisation (zwischen Datenströmen im Studio und innerhalb von Datenströmen zwischen gleichen und unterschiedlichen Essenztypen). In diesem Sinne erfolgt nach der Beschreibung des Gesamtlatenzmodells die Diskussion von für die Funktionalität des Gesamtsystems wichtigen Technologien (meist am Beispiel Video). Dazu zählen Konzepte für die schnelle Übertragung von Dateien (*File-Streaming*) und für die Datenverteilung im Studio. Den Abschluss bildet die Beschreibung einer prototypischen Implementierung und deren Evaluierung durch eine Reihe von Messungen, die die Kernaussagen des Systemkonzeptes für eine Netzwerkarchitektur verfizieren.

6.1 Systemdesign

Die Diskussion von Entscheidungsaspekten hilft nachfolgend dabei, die gestellten Anforderungen des Zielsystems zu berücksichtigen (siehe Abschnitt 2.1.4) und trägt dazu bei, ein einfaches und stabiles Gesamtsystem zu entwerfen.

Anwendung eines Standard-Netzwerk-Protokollstapels

Das Kapitel 3 beschreibt die Auswahl eines Protokollstapels, der für ein begrenzt komplexes Netzwerk mit bekannter Quantität und Qualität der Netzwerkteilnehmer (vergleiche Abschnitt 3.4) die Anforderungen an ein Übertragungssystem im Studio erfüllt. Abbildung 6.1 stellt diesen praxiserprobten Protokollstapel dar.

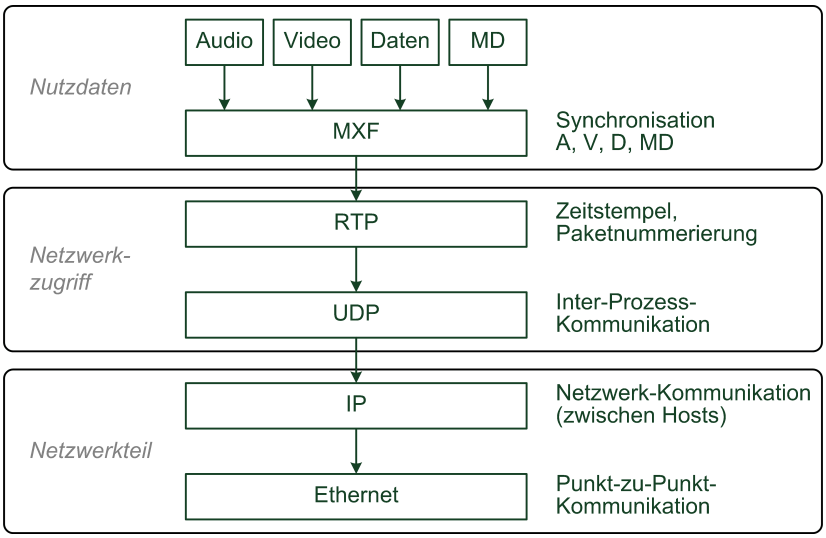


Abbildung 6.1: Protokollsicht auf das Gesamtsystem

Im Netzwerkteil wird mit Blick auf zukünftige technologische Weiterentwicklungen (100-Gigabit-Ethernet) und Kosten für Implementierung, Betrieb und Wartung Ethernet als Übertragungstechnologie gewählt. Darauf aufbauend stellt das Internet-Protokoll (IP) eine flexible und erprobte Lösung für die Aufgabenstellungen Adressierung, Routing und Multicastübertragung zur Verfügung, zu der vor dem Hintergrund der Erweiterung des Systems auf ein Internetwork für Studioanwendungen keine Alternative besteht. Die Umgehung von IP und darüber liegender Schichten (wie in der Automatisierungstechnik, siehe Abbildung 2.4), um eine latenzärmere Übertragung auf Ethernet-Basis zu ermöglichen, kommt nicht in Betracht, um IT-Standard-Komponenten für Übertragung und Bearbeitung verwenden zu können.

Über dem zentralen IP-Protokoll kommt mit dem *User Datagram Protocol* (UDP) ein Transportprotokoll zum Einsatz, das aufgrund seines verbindungslosen Charakters die Zustellung von Paketen innerhalb enger Zeitgrenzen erlaubt, aber nicht garantiert. Zur Verifikation der theoretisch verlustfreien und rechtzeitigen Übertragung wird das in der Anwendungsschicht angesiedelte *Realtime Transport Protocol* (RTP) verwendet. Es ermöglicht über Paketnummerierung und Zeitstempel die Prüfung auf Vollständigkeit und korrekte Reihenfolge der Pakete eines Datenstroms.

Über den bisher geschilderten Protokollstapel können prinzipiell beliebige Datenformate in der Nutzdatenschicht übertragen werden. Durch die Kapselung von Essenz- und Metadaten in einem Containerformat wie MXF kann so auf genau spezifizierte Mechanismen zur Synchronisation aller Daten zurückgegriffen werden. Dies betrifft insbesondere Metadaten, die im Kontext der automatischen Be- und Verarbeitung der Essenzdaten Grundlage für automatische Abläufe darstellen.

Verzicht auf QoS-Mechanismen

Die in Abschnitt 3.1.3 vorgestellten Verfahren zur Unterstützung von QoS entfalten Ihre Leistungsfähgkeit erst in hinreichend großen (Inter)-Netzwerken mit vielen Netzwerknoten und Hosts. Die Komplexität einer Netzwerkarchitektur für Studioanwendungen ist hingegen gering: Alle Geräte sind mit einem zentralen Switch verbunden. Löser schlägt unter diesen Bedingungen eine zentrale Kontrollinstanz vor, die Art und Umfang des Datenverkehrs überwacht und somit Echtzeitdatenverkehr ermöglicht [Lös06, S. 27]. Eine solche Kontrollinstanz sichert die Übertragung auch im Systemkonzept.

Unkomprimierte Essenzen

Im Broadcastbereich, speziell in der Produktion und Kontribution[1], gibt es die Tendenz zur Nutzung umkomprimierter Essenzen. Am Beispiel Video hat dies folgende Gründe [Sim04]:

- Qualität: unkomprimierte Datenübertragung vermeidet visuelle Artefakte, die ggf. erst nach mehreren Bearbeitungsgenerationen sichtbar werden
- Zeit: der Transport unkomprimierter Daten kann mit sehr kleinen Latenzen durchgeführt werden, weil die Anwendung zeitaufwändiger Kompressionsalgorithmen unterbleibt
- Kosten: unkomprimierte Datenübertragung vermeidet Kosten für Anschaffung und Monitoring leistungsfähiger Encoder und Decoder (Hardware und Software)

[1] Zuführung von Rohmaterial, fertig geschnittenen Beiträgen oder einer Live-Sendung, z.B. von einem Übertragungswagen zum Hauptstudio oder zwischen zwei Studios

Prinzipiell sind unkomprimierte Videosignale robuster gegenüber Übertragungsfehlern, da sich diese im Falle von Bitfehlern nur auf einzelne Bildpunkte auswirken. Bei komprimierten Videosignalen ist im Fehlerfall ein Bildbereich (z.B. Block) oder das gesamte Bild betroffen (in Abhängigkeit des Kompressionsverfahrens). Bei Interframe-Verfahren können sich Fehler auch zeitlich fortpflanzen [ATW02]. Des Weiteren gehen insbesondere effiziente Kompressionsverfahren mit einer Verzögerung einher, die bei unkomprimierter Signalübertragung vermieden werden können; allerdings auf Kosten eines erhöhten Bandbreitenbedarfes.

Aus diesen Gründen werden im Zielsystem Essenzdaten unkomprimiert transportiert. Des Weiteren werden die Essenzen nativ übertragen. Am Beispiel Video wird also nicht ein SDI-Frame nach ITU-R BT.656 mit horizontalen und vertikalen Austastlücken verwendet, sondern nur der sichtbare Bereich eines Vollbildes von 720×576 Bildpunkten. So wird die Datenmenge und damit die Datenrate minimiert. Zusatzdaten, die optional in den Austastlücken übertragen werden können, werden durch Mechanismen eines Containerformates wie MXF synchron transportiert.

Verzicht auf Fehlerschutz

Bei der Übertragung von Daten über Datagramm-Netzwerke können Fehler auf unterschiedlichen Ebenen auftreten. Übertragungsfehler können sich in Bitfehlern, Paketfehlern und Unterbrechungen (z.B. durch Kabeldefekt) äussern, wobei Bitfehler Paketfehler nach sich ziehen können. Um Auswirkungen von Bitfehlern zu umgehen, existieren zwei wesentliche Möglichkeiten. Fehlerhaft übertragene Daten können einerseits erneut angefordert und übertragen werden (*Backward Error Correction* – BEC). Alternativ erlauben zusätzlich übertragene redundante Daten die Korrektur von Fehlern auf Basis mathematischer Algorithmen (*Forward Error Correction* – FEC). Ein oft genutzter Algorithmus im Mulimedia-Bereich ist die Reed-Solomon-Codierung (RS). Die FEC-Leistungsfähigkeit in Bezug auf korrigierbare Fehlerstellen nimmt mit dem Umfang der Redundanzdaten zu. Die Technik der Verwürfelung (auch: *Scrambling*), bei der Daten vor der Übertragung in eine alternative Reihenfolge gebracht werden, kann die Auswirkungen von Bitfehlern reduzieren bzw. die Korrektur von Burstfehlern[2] durch FEC ermöglichen. Unterbrechungen können bei entsprechend redundanten Systemen mit der verlustfreien Umschaltung (*Hitless Switchover*) auf alternative Pfade behoben werden.

Bei Echtzeitanwendungen ist die Latenz durch BEC meist nicht tolerabel. Zudem ist die Anwendung von BEC in Multicastübertragungen prinzipbedingt sehr komplex; aus der Sicht des Autors existieren noch keine effizienten Algorithmen dafür. Auch FEC hat in Abhängigkeit des verwendeten Algorithmus und dem Umfang der Redundanzdaten eine nicht zu vernachlässigende zusätzliche Latenz zur Folge [NJH08]. Des Weiteren verursacht FEC einen Mehrbedarf an zu übertragenden Daten, der sich in einer höheren Datenrate

[2] mehrere Bitfehler in engem zeitlichen Zusammenhang

äußert. Aus diesen Gründen wird im Systemkonzept auf die Anwendung von BEC und FEC verzichtet. Stattdessen wird aufgrund der begrenzten Komplexität eines Studionetzwerkes eine kollisions- sowie verdrängungsfreie und damit fehlerlose Datenübertragung vorausgesetzt (vergleiche [Lös06]).

Nutzung von Standard-Hardware

Durch den Einsatz von handelsüblichen PC- und Netzwerktechnik-Komponenten stehen leistungsfähige aber dennoch preiswerte Rechen-, Übertragungs- und Speicherkapazitäten zur Verfügung. Für den Einsatz im professionellen Bereich werden dennoch Anforderungen an die Hardware gestellt, die von angepasster Hardware (Workstation) erfüllt wird:

- große Prozessortaktraten, die die Verarbeitung hochratiger Datenströme ermöglichen
- Systembus-Architekturen, die keinen Engpass für hochratige und schnelle Kopierprozesse darstellen
- Netzwerkkarten (NICs) mit Interrupt-Moderation, die die Hauptprozessorlast des PCs für die Netzwerkverarbeitung begrenzen und damit Übertragungsfehler vermeiden sowie Prozessorkapazität für Be- und Verarbeitungsprozesse bereitstellen; zusätzliche Prozessoren (z.B. DSPs) können auf den NICs nebenläufige Prozesse der Netzwerkverarbeitung (z.B. Echo- und Cross-Talk-Unterdrückung) übernehmen und die PC-CPU entlasten
- Switches mit ausreichend vielen Netzwerkports, großem Backplane-Durchsatz und Multicastfähigkeit (Unterstützung hinreichend vieler Multicast-Gruppen)

Nutzung von Standard-Betriebssystemen

Bei der Contentverarbeitung im Live-Studio müssen Zeitgrenzen eingehalten werden (harte Echtzeit). Sind die damit definierten Zeitspannen relativ groß (z.B. mehrere Millisekunden für ein Halb- oder Vollbild), wird der Einsatz von verbreiteten Betriebssystemen (Linux, Microsoft Windows) unter bestimmten Voraussetzungen möglich. Dazu zählt die sorgfältige Systemkonfiguration (z.B. Festlegung hoher Prioritäten für Prozesse echtzeitkritischer Applikationen) und die Berücksichtigung bekannter Optimierungsstrategien bei der Applikationsentwicklung (wie sorgfältige Speicherverwaltung – siehe *Garbage Collection* und *Swapping* – sowie Minimierung von Kontext-Wechseln und Kopiervorgängen – siehe auch [Tan04, S. 613 ff.]) erlaubt. Der Einsatz von Standard-Betriebssystemen gestattet kurze Einarbeitungszeiten für Anwender und Wiederverwendung existierender Softwarekomponenten in der Entwicklung (dies schliesst auch *Open Source Software* ein). Prinzipiell ermöglicht die Umsetzung von Funktionalitäten in Software eine flexible Weiterentwicklung.

Übersicht über das Gesamtsystem

Unter Berücksichtigung der o.g. Designaspekte entsteht ein einfach strukturiertes Netzwerk aus Standardkomponenten, das in Abbbildung 6.2 skizziert wird. Ein zentraler Switch stellt den Kreuzungspunkt zwischen vielen PCs (Workstations) dar, die spezifische Aufgaben der Essenz- und Contentverarbeitung übernehmen. In Abhängigkeit der Leistungsfähigkeit der Workstations und des Havariekonzeptes[3] können einzelne Workstations mehrere Aufgaben übernehmen.

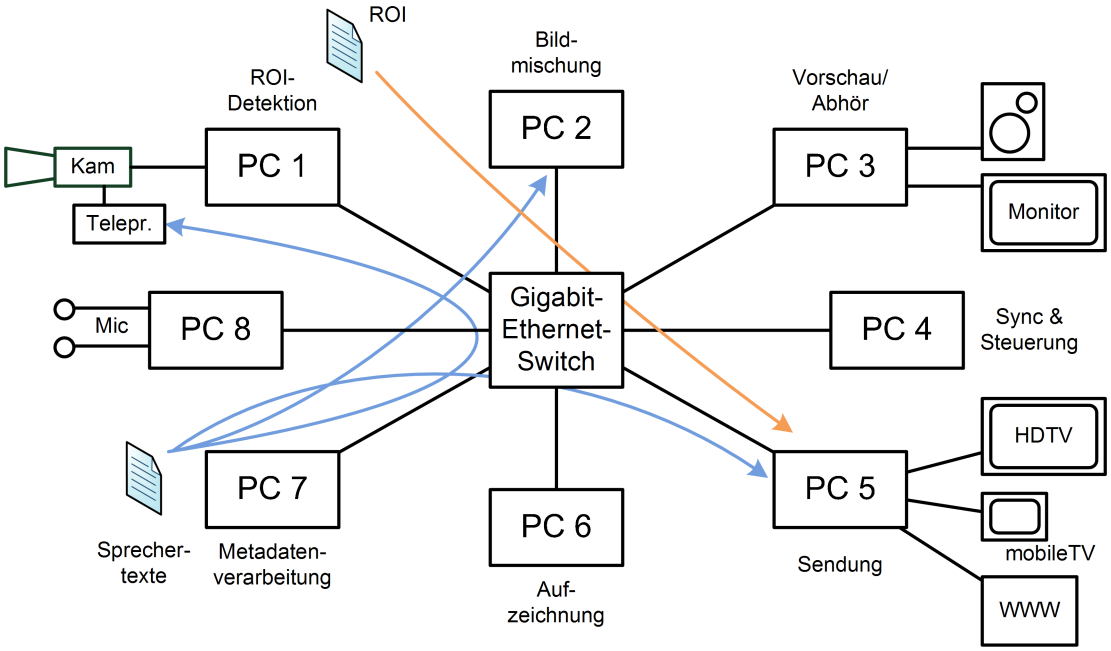


Abbildung 6.2: Prinzipieller Systemaufbau mit Beispielanwendungsfällen

Anhand zweier Anwendungsfälle (vergleiche Abschnitte 5.3.1 und 5.3.2) wird die Nutzung von Metadaten während einer Liveproduktion verdeutlicht. Sprechertexte können demnach zeitgleich von einem Telepromter-System an der Studiokamera zur Unterstützung des Sprechers eingespiegelt (PC 1), von einem (universellen) Bildmischer als Untertext eingeblendet (PC 2) und/oder als Basis für eine textbasierte Veröffentlichung im Internet über eine Webseite verwendet werden. Die Festlegung eines Bildausschnittes für kleine Bildschirme (*Region of Interest* Detektion auf PC 1) hat Metadaten zur Folge, die im Rahmen der Distribution für verschiedene Kanäle genutzt werden kann (PC 5). Das offene und flexible Netzwerk zwischen den Workstations transportiert dabei beliebige Datenströme (Essenz und/oder Metadaten) im Verbund oder einzeln.

[3] das nicht Bestandteil dieser Arbeit ist

6.2 Gesamtsystem-Latenzmodell

Die Gesamtlatenz eines Netzwerkes für Studioanwendungen setzt sich aus zwei grundsätzlichen Komponenten zusammen. Zum einen entstehen Verzögerungen durch die synchrone Datenübertragung über paketorientierte Netzwerke (Ethernet), zum anderen haben Prozesse zur Contentverarbeitung Latenzen zur Folge (vergleiche Abbildung 6.3). Beide Komponenten bestehen, wie in den Abschnitten 3.2 und 4.4 beschrieben, wiederum aus Teilkomponenten. Diese Teilkomponenten werden in Abbildung 6.3 als Schichten dargestellt. Dabei ist es möglich, einzelne Schichten zu umgehen und damit die Latenz zu reduzieren.

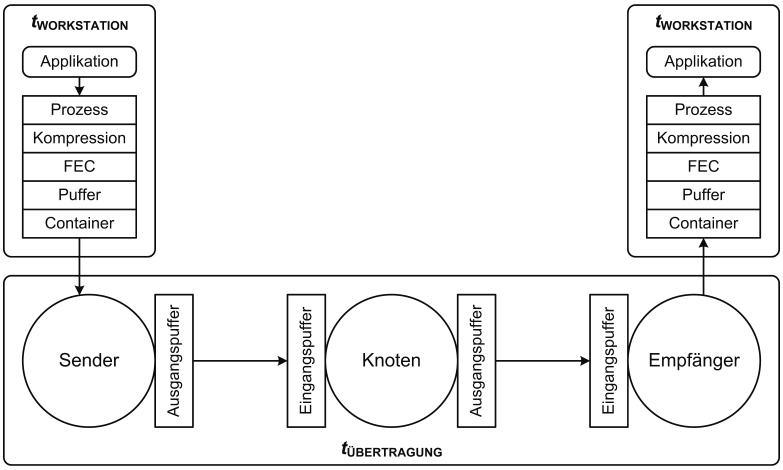


Abbildung 6.3: Abgrenzung der Komponenten im Gesamtlatenzmodell

$$t_{\text{GESAMT}} = \sum_{m-1} t_{\text{ÜBERTRAGUNG}} + \sum_{m} t_{\text{WORKSTATION}}(m) \tag{6.1}$$

$$t_{\text{ÜBERTRAGUNG}} = t_{\text{Sender}} + \sum_{n} t_{\text{Knoten}}(n) + t_{\text{Empfänger}} \tag{6.2}$$

$$t_{\text{WORKSTATION}} = t_{\text{prozess}} + t_{\text{kompression}} + t_{\text{fehlerschutz}} + t_{\text{puffer}} + t_{\text{container}} \tag{6.3}$$

n stellt dabei die Anzahl der Netzwerkknoten dar (die bei der jeweiligen Übertragung involviert sind), m die Anzahl der beteiligten Workstations. Die Minimierung der Netzwerkknoten trägt zur Reduktion der Komplexität, der möglichen Fehlerquellen und der Latenz bei. Insofern gehen alle folgenden Konzeptschritte daven aus, dass nur ein Netzwerkknoten (ein zentraler Switch) zum Einsatz kommt (n=1). Die Latenz einer Übertragung vereinfacht sich dann zu:

$$t_{\text{ÜBERTRAGUNG}} = t_{\text{Sender}} + t_{\text{Knoten}} + t_{\text{Empfänger}}$$

In einem typischen Anwendungsfall der Fernsehstudioproduktion wird das Signal einer Kamera auf einem Mischer mit einem Effekt versehen und auf einem Videomonitor zur Anzeige gebracht. Das Beispiel umfasst zwei Übertragungen (Kamera – Mischer, Mischer – Monitor) und einen Bearbeitungsvorgang. Auf Kompression, Fehlerschutz sowie Synchronisation auf den Studiotakt wird verzichtet, so dass bei der Bearbeitung auf dem Mischer nur die Latenz zur Berechnung des Effektes zu berücksichtigen ist. Übertragungs- und Effektlatenz werden mit jeweils 10 ms abgeschätzt. Die theoretische Gesamtlatenz berechnet sich dann zu:

$$\begin{aligned} t_{\text{GESAMT}} &= 2 \cdot t_{\text{ÜBERTRAGUNG}} + t_{\text{WORKSTATION}} \\ &= 2 \cdot 10\,ms + 10\,ms = 30\,ms \end{aligned}$$

Die Latenz liegt hat die Größenordnung einer Vollbilddauer und entspricht damit einer typischen Verzögerung eines Videosignals im digitalen Komponentenstudio. Die Verifikation dieser einfachen Berechnung anhand eines Prototypen erfolgt in Abschnitt 6.5.

6.3 Das Konzept des File-Streaming

Neben der Contentverarbeitung hat die Datenübertragung einen wesentlichen Einfluss auf die Gesamtlatenz. Ziel ist es nun, ein Konzept der latenzarmen, aber dennoch offenen und erweiterbaren Contentübertragung zu definieren, das den synchronen Transport beliebiger Daten erlaubt. Ausgehend von der Übertragung von AV-Daten in Videosystemen werden drei prinzipielle Methoden unterschieden [Kov06, S. 55 ff.]:

1. *File-Transfer*
2. *Streaming*
3. *Direct-to-Storage*

File-Transfer bezeichnet eine zuverlässige Datenübertragung, die in Abhängigkeit der zur Verfügung stehenden maximalen Datenrate des Kommunikationskanals schneller oder langsamer als Echtzeit stattfinden kann. *Streaming* ist charakterisiert durch eine nicht auf Fehlerfreiheit geprüfte isochrone Datenübertragung in Echtzeit. *Direct-to-Storage* beschreibt den Lese- und Schreibzugriff eines an einem Datenspeicher angeschlossenen Rechners.

Produktionsabläufe in einem Fernsehstudio setzen sich traditionell aus einer Folge von Prozessen zusammen, die linear durchlaufen werden. Innerhalb eines solchen Produktionsablaufes haben die drei Methoden zur Audio- und Videoübertragung unterschiedliche Zeitverhalten. Alle Bearbeitungsprozesse nehmen eine definierte Zeit in Anspruch, die aufgrund der isochronen Übertragung über die SDI-Schnittstelle auf ganzzahlige Vielfache einer Vollbilddauer (in 50 Hz Systemen 40 ms) aufgerundet werden müssen. Diese Verzögerung wird durch eine zusätzliche Pufferung umgesetzt. Ein Dateitransfer ist nicht an diesen Takt gebunden. Das Beispielszenario in Abbildung 6.4 bezieht sich auf einen Arbeitsablauf, der die Bearbeitungsprozesse Skalierung, Filterung und MPEG-Kodierung sequentiell auf Essenzmaterial anwendet, das auf einem A/V-Server gespeichert ist. Die Dauer der Teilprozesse ist dabei mit dem Ziel der einfachen Beschreibung willkürlich festgelegt.

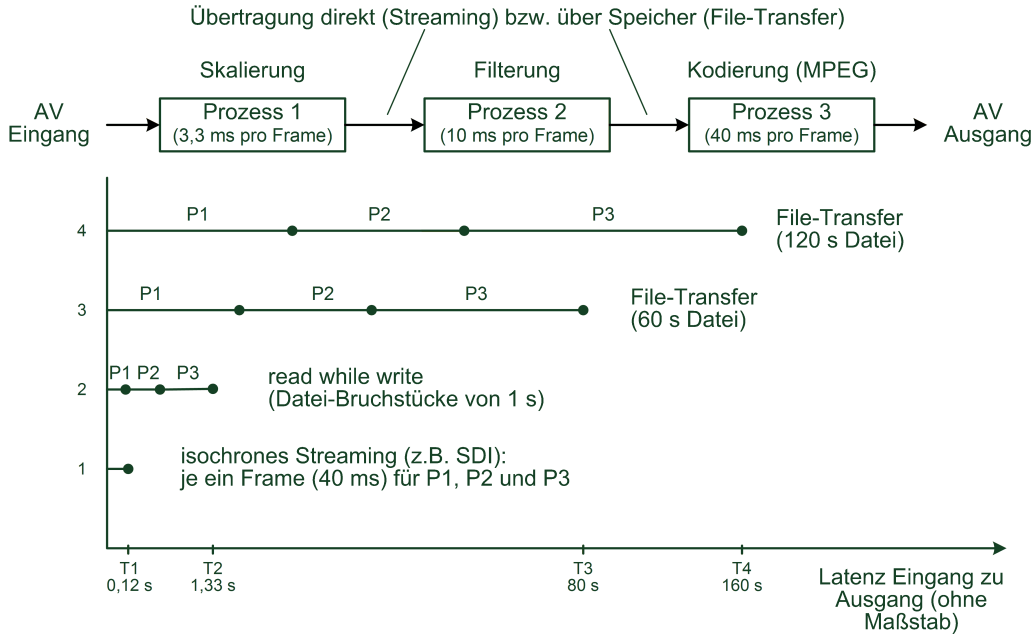


Abbildung 6.4: Vergleich der Konzepte File-Transfer, Direct to Storage und isochrones Streaming bei einer linearen Prozessverarbeitung nach [Kov06, S. 360]

Isochrones *Streaming* (z.B. SDI, AES/EBU) (1) ruft für das Beispielszenario in Abbildung 6.4 eine Eingangs- zu Ausgangsverzögerung von drei Frames (120 ms) hervor, die sich aus der Summe der Verzögerungen der Prozesse 1 bis 3 zusammensetzt (je ein Frame für P1, P2 und P3, da bei jeder Übertragung über die isochrone Schnittstelle auf den Vollbildtakt gewartet werden muss).

Der darüber dargestellte *read while write* Modus (2) beschreibt eine Form der *Direct-to-Storage*-Methode, bei der eine Datei in einem Speicher in Bruchstücke geteilt wird, die im Beispiel eine Dauer von einer Sekunde aufweisen. Diese Bruchstücke werden sequentiell

von den Prozessen P1 bis P3 bearbeitet, was in einer Gesamtlatenz von 1,33 s zum Ausdruck kommt. Jeder Prozess greift dabei auf den Speicher zu, um entsprechende Dateibruchstücke zu laden. P2 kann erst beginnen, wenn P1 abgeschlossen ist. Eine zusätzliche Verzögerung durch das Warten auf den Vollbildtakt entfällt.

$$\begin{aligned} t_{\text{read while write}} &= 1\,s \cdot 25\,Frames/s \cdot (t_{\text{P1}} + t_{\text{P2}} + t_{\text{P3}}) \\ &= 1\,s \cdot 25\,Frames/s \cdot (3,3 + 10 + 40)\,ms/Frame \\ &= 1.332,5\,ms \end{aligned}$$

Bei den oberen zwei Beispielen für den *File-Transfer* (3 und 4) wird die Auswirkung der Dateigröße auf die Gesamtlatenz deutlich, wenn die Dateien jeweils als vollständige Einheit sequenziell bearbeitet werden. Für eine 60-sekündige Datei werden Verzögerungen durch P1 von 5 Sekunden, durch P2 von 15 Sekunden und durch P3 von 60 Sekunden angenommen. Für die doppelte Dateilänge ergeben sich auch doppelte Latenzwerte durch die einzelnen Prozesse.

Das in dieser Arbeit zugrunde liegende Netzwerk verwendet eine Mischung aus allen drei Methoden: Übertragen wird eine Datei (*File-Transfer*), allerdings unterteilt in sehr kleine Bruchstücke (vgl. *Direct-to-Storage*), so dass insgesamt eine Latenz erreicht wird, die mit isochronem *Streaming* vergleichbar ist. Dabei werden die Daten direkt zwischen den Prozessen ohne Zwischenspeicherung auf einem A/V-Server übertragen, indem ein Client-Server-Prozess zwischen den korrespondierenden Rechnern etabliert wird. Im Folgenden wird diese Methode der Datenübertragung mit *File-Streaming* bezeichnet. *File-Streaming* ermöglicht die Anwendung von ausgewählten Dateiformaten wie MXF in der Live-Studioproduktion.

Im Folgenden soll auf Basis des Gesamtlatenzmodells (vergleiche Abschnitt 6.2) die *File-Streaming*-Methode mit den oben genannten Methoden verglichen werden. Vereinfachend verzichtet das Beispiel auf Kompression, Fehlerschutzmechanismen, Pufferung und MXF-Kapselung; die Latenzen werden also nur von den Bearbeitungsprozessen bestimmt ($t_{\text{WORKSTATION}} = t_{\text{prozess}}$). Für jeden Übertragungsprozess sei eine Latenz von 10 ms angenommen[4].

$$\begin{aligned} t_{\text{GESAMT}} &= \sum_{m+1} t_{\text{ÜBERTRAGUNG}} + \sum_{m} t_{\text{WORKSTATION}}(m) \\ &= (m+1) \cdot t_{\text{ÜBERTRAGUNG}} + \sum_{m} t_{\text{WORKSTATION}}(m) \\ &= 4 \cdot 10\,ms + 3,3\,ms + 10\,ms + 40\,ms = 93,3\,ms \end{aligned}$$

[4] Messungen belegen diese Größenordnung, siehe Abschnitt 6.5

Das Ergebnis stellt eine pessimistische Schätzung dar, die davon ausgeht, dass zur Verarbeitung jeweils die gesamte Vollbildinformation vorliegen muss. Abhängig vom verwendeten Algorithmus kann die Berechnung ggf. schon vor der vollständigen Übertragung beginnen und so die Gesamtlatenz weiter minimiert werden. Abbildung 6.5 stellt am oben beschriebenen Beispiel theoretische Werte der Latenz für *File-Streaming* und isochronem *Streaming* gegenüber. Es wird deutlich, dass durch den Verzicht auf Synchronisation nach jedem Bearbeitungsschritt die Latenz der Netzwerkübertragung mehr als kompensiert werden kann. Im konkreten Beispiel ist der Gesamtworkflow mit *File-Streaming* auf Datagramm-Netzwerkbasis um 26,7 *ms* kürzer als isochrones *Streaming* über SDI.

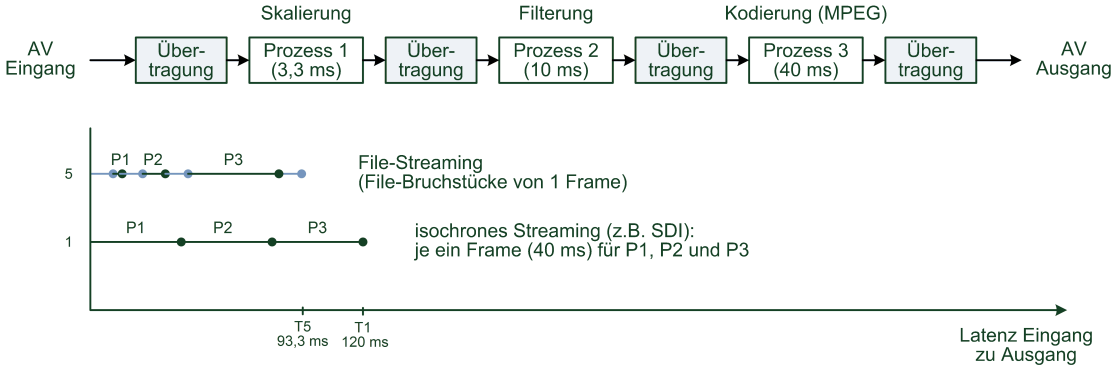


Abbildung 6.5: Vergleich der Konzepte isochrones *Streaming* und *File-Streaming* bei einer linearen Prozessverarbeitung

Folgende Trends zur Minimierung der Gesamtsystemlatenz werden in Abbildung 6.5 verdeutlicht:

1. Minimierung der Latenz in jedem Prozess und
2. Minimierung teilnehmender Systeme (Workstations).

Beide Aspekte werden durch eine leistungsstarke PC-Architektur (unter Nutzung zusätzlicher Prozessor-Ressourcen, z.B. GPU) unterstützt: Kürzere Prozess-Rechenzeiten erlauben die Verringerung der einzelnen Prozesslatenzen. Damit wird die Integration vieler Prozesse auf einer Workstation möglich, was die Anzahl der Übertragungen in einem linearen Ablauf minimiert.

Solange Daten in einer IT-basierten Umgebung ohne Synchronisationszwang augetauscht werden, kann *File-Streaming* in definierten Fällen kürzere Verzögerungszeiten aufweisen, als traditionelles isochrones *Streaming*. Voraussetzung dafür ist es, dass nicht nur die reine Übertragungszeit betrachtet wird, sondern dass Übertragung *und* Bearbeitung als Einheit wahrgenommen werden. Mit steigender Rechenleistung sind viele Bearbeitungsprozesse schneller durchführbar und der Zeitgewinn nimmt entsprechend zu.

6.4 Datenverteilung im Studio

Das Prinzip der sequentiellen Signalbearbeitung im Fernsehstudio bleibt im Livebetrieb auch für neue Architekturen bestehen. Die grundlegenden Anforderungen an die Signalverteilung müssen auch von alternativen Konzepten zur isochronen Essenzdatenübertragung erfüllt werden. Dazu zählen insbesondere die Signalverteilung auf viele Empfänger (1 zu n) und das störungsfreie Umschalten zwischen Essenzsignalen (Audio bzw. Video). Im konventionellen Studio werden dafür Kreuzschienen eingesetzt.

6.4.1 Multicast-Übertragung

Die distributive Eigenschaft der Datenverteilung auf viele Empfänger kann in Netzwerken mithilfe der Unicast-Übertragung oder der Multicast-Übertragung abgebildet werden. Die Etablierung mehrerer Unicast-Übertragungen kann nur bis zum Erreichen der Datenratenobergrenze zwischen Sender und Switch skaliert werden, d.h. die Anzahl der Empfänger ist begrenzt (siehe Abbildung 6.6).

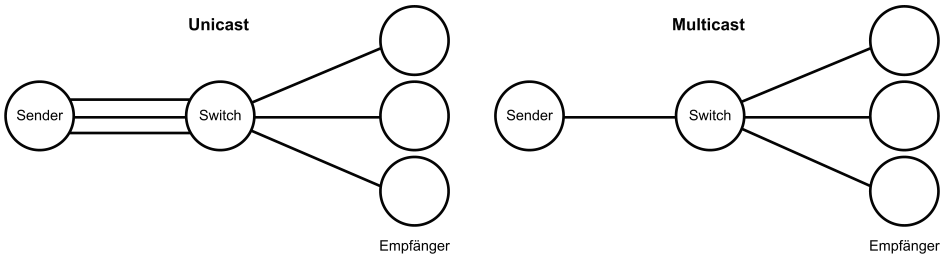


Abbildung 6.6: Unicast und Multicast

Multicast erlaubt eine effizientere Übertragung vom Sender bis zum Netzwerkknoten, aber es entstehen Latenzen durch erweiterte Verarbeitungs- und Verwaltungsaufgaben des Switches (siehe Abschnitt 3.3.1.4). Die Latenz durch die Erzeugung von Paketkopien $t_{\mathrm{multicast}}$ ist dabei von der Anzahl der Empfänger in einer Multicastgruppe abhängig. Zusätzliche Latenzen entstehen durch den Verwaltungsaufwand (Einrichtung, An- und Abmeldung von Empfängern usw.) der Multicastgruppe (t_{MCadmin}). Sowohl $t_{\mathrm{multicast}}$ als auch t_{MCadmin} sind Bestandteil der in Abschnitt 3.2 definierten Latenz durch die Netzwerkprotokoll-Verarbeitung t_{nv}. Die Verifikation realer Latenzwerte durch die Multicastübertragung erfolgt in Abschnitt 6.5.3.3.

Für eine effiziente Datenverteilung im Studio wird die Verwendung von IP-Multicast propagiert. Mit dem Ziel einer latenzarmen Implementierung muss dabei für jeden Sender eine Multicastgruppe zur Verfügung stehen und eingerichtet sein. Damit wird t_{MCadmin} minimiert. Da der Empfänger nach der Aufschaltung auf die Multicastgruppe zwar Pakete empfängt, diese aber erst nach dem Start eines neuen Vollbildes sinnvoll verarbeiten bzw. anzeigen kann, entsteht im ungünstigsten Fall eine zusätzliche Verzögerung am

Empfänger in der Größenordnung eines Vollbildes (40 ms). Diese Verzögerung ist ein inhärentes Problem einer nicht isochronen Übertragung und tritt prinzipbedingt auch bei Unicast auf. Eine Umschaltung zwischen verschiedenen Datenströmen kann aus diesem Grund mit Standard-Switches weder im Unicast- noch im Multicast-Übertragungsmodus hinreichend störungsfrei erfolgen.

6.4.2 Burstmodus

Die Netzwerk- und Prozessorauslastung bei der Übertragung von hochratigen Datenströmen unter Verwendung von Standard-PC-Komponenten wurde von Magaña et. al untersucht [MAV04]. Darin werden zwei Übertragungsarten unterschieden (siehe Abbildung 6.7). Der kontinuierliche Modus (a) erzeugt eine gleichmäßige Netzwerklast, indem Pakete eines Videostromes gesendet werden, sobald sie verfügbar sind. Damit werden geringe Übertragungslatenzen erreicht, allerdings auf Kosten einer konstant hohen Prozessorauslastung des Empfängers. Dieser wartet aktiv auf eintreffende Pakete und kann sofort auf deren Empfang reagieren. Praktisch erfolgt auf dem Empfangsrechner beim Empfang eines Pakets ein Interrupt, der die Verarbeitung des Paketes durch den Prozessor initiiert. In Verbindung mit dem dafür notwendigen Kontextwechsel (der auch CPU-Zeit benötigt) ist der Prozessor so mit Empfang und Verarbeitung des Netzwerkverkehrs ausgelastet.

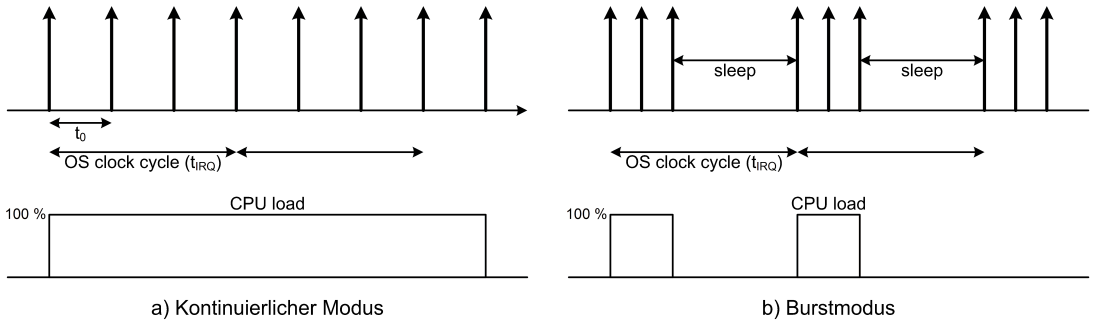


Abbildung 6.7: Auswirkung von kontinuierlichem Modus und Burstmodus auf die Prozessorauslastung [MAV04]

Die Totalauslastung der Empfänger-CPU wird vermieden, indem mehrere Pakete gesammelt und gebündelt unter Ausnutzung der maximalen Übertragungsrate gesendet werden. Der Burst-Modus (b) erlaubt es, dass der Prozessor zwischen den Sendezyklen neben der Bewältigung des Netzwerkverkehrs Berechnungskapazität für weitere Prozesse zur Verfügung hat. Die periodische Auslastung der Netzwerkkapazität ist für geswitchte Ethernetnetzwerke unproblematisch, solange der zentrale Netzwerkknoten hinreichend große Backplanekapazität zur Verfügung stellt (was i.d.R. der Fall ist). Nachteilig wirkt sich der Burst-Modus auf die Latenz aus.

Neben der sendeseitigen Beeinflussung des Netzwerkverkehrs bieten moderne Netzwerk-Controller mit Interrupt-Moderation die Möglichkeit, die CPU zu entlasten, indem sie

mehrere Empfangspakete sammeln, bevor ein Interrupt ausgelöst wird. OTERO hat mit praktischen Messungen nachgewiesen, dass bereits ein kurzzeitiges Pausieren des Sende-Threads[5] für eine Millisekunde (1 ms) ausreicht, um die CPU-Auslastung signifikant zu senken [Ote08].

Trotz der damit verbundenen Latenz greift die in dieser Arbeit beschriebene Architektur auf den Burstmodus zurück, weil dieser auch im Zusammenhang mit der Problematik des störungsfreien Umschaltens positiven Einfluss ausübt.

6.4.3 Synchronität

In traditionellen Studios werden alle Geräte mit einem zentralen Takt versorgt (vergleiche Abbildung 6.9), auf dessen Grundlage alle Signale zu jeder Zeit an jeder Stelle im Studio synchron vorliegen (isochrone Übertragung). Damit wird ein störungsfreier Umschaltvorgang ohne örtliche und zeitliche Einschränkung ermöglicht. Komplexere Signalbearbeitungsprozesse, die eine größere, nicht mehr kompensierbare Latenz verursachen[6], haben dabei eine Pufferung der Signale bis zum nächsten Takt zur Folge.

Synchrone Übertragung

Die Nutzung eines paketorientierten Netzwerkes im Studio erlaubt die isochrone Datenübertragung nur unter Einschränkungen, die mit dem Ziel der für beliebige Daten offenen Kommunikation nicht vereinbar sind. Eine synchrone Übertragung[7] hingegen kann durch sorgfältige Konfiguration von Hard- und Softwarekomponenten in Kombination mit einer einfachen Pufferung der Daten beim Empfänger erreicht werden. In Abhängigkeit der zeitlichen Anforderung einer Applikation besteht die wesentliche Aufgabe darin, die Größe des Puffers zu optimieren. Je größer die Puffergröße, desto geringer sind die Anforderungen an das verwendete Transportnetzwerk und umso größer die durch den Puffer verursachte Latenz. Für die Anwendung in der professionellen Live-Studioproduktion wird eine maximale Latenz von einem Vollbild (40 ms für 50i- bzw. 25p-Systeme) angestrebt.

Das *File-Streaming*-Konzept (vergleiche Abschnitt 6.3) sieht eine Minimierung der Synchronisationspunkte vor, um die Gesamtsystemlatenz gering zu halten. Dies trifft insbesondere für Netzwerkknoten bei der Übertragung, aber auch auf Bearbeitungsgeräte wie beispielsweise Mischer zu. Eine Herausforderung bei der synchronen Übertragung stellt allerdings die Umsetzung eines störungsfreien Umschaltvorgangs für Essenzsignale dar (vergleiche Abschnitt 6.4.4).

[5] Ein Thread ist Teil eines Prozesses und bezeichnet einen Ausführungsstrang in der Abarbeitung eines Programms

[6] z.B. aufwändige Bildeffekte mit einer Berechnungsdauer von einigen Millisekunden

[7] „synchrone Übertragung“ beschreibt hier, dass Daten garantiert vor einer determinierten Ankunftszeit empfangen werden, vergleiche Kapitel 3

Abbildung 6.8 beschreibt die zeitlichen Zusammenhänge bei einer synchronen Übertragung eines Bildsignales zwischen einer Kamera und einem Monitor über eine Bearbeitungsstufe z.B eines Bildmischers (Effekt) im Burst-Modus. Nach der Übertragung können unregelmäßige Abstände zwischen den übertragenen Halbbilddaten auftreten (z.B. durch variable Paketlaufzeiten). Die Berechnung des Effektes hat eine wahrscheinlich konstante Verzögerung pro Halbbild zur Folge. Eine Synchronisation auf den Vollbildtakt vor der Berechnung des Effektes findet explizit nicht statt.

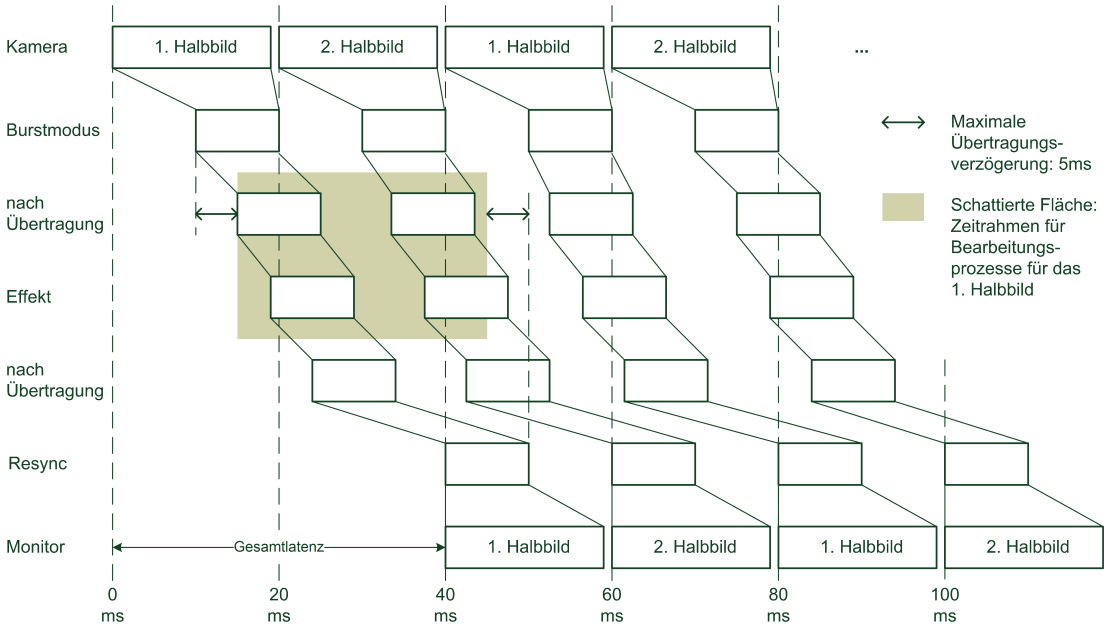


Abbildung 6.8: Auswirkungen der synchronen Übertragung und Synchronisation auf einen zentralen Wiedergabetakt

Zur Vermeidung von Bildstörungen bei der Anzeige muss in einer Resynchronisationsstufe einerseits eine Neutaktung auf die Halbbildfrequenz und andererseits eine Ausrichtung am zentralen Takt erfolgen. Beides kann zeitgleich über eine Pufferung mit anschliessendem getakteten Auslesen realisiert werden. Die Größe der Gesamtlatenz (im Beispiel 40 ms) kann auf der Grundlage erreichbarer Übertragungszeiten und notwendiger Bearbeitungslatenzen im Netzwerk dimensioniert werden.

Der schattierte Bereich in Abbildung 6.8 kennzeichnet den Zeitrahmen für die Berechnung des Effektes für das erste Halbbild. Der Rahmen wird auf der linken und rechten Seite durch die definierte Gesamtverzögerung des Beispielsystems (40 ms) und die maximale Übertragungsverzögerung (5 ms) begrenzt. Die Daten des 1. Halbbildes müssen vor und nach der Berechnung vollständig im Zeitrahmen vorliegen, um die maximale Gesamtverzögerung garantiert nicht zu überschreiten. Der Zeitrahmen ist im Beispiel 30 ms groß. Wird davon noch die Bearbeitungsprozessdauer (im Beispiel ebenfalls 5 ms) subtrahiert, so ist der Zeitrahmen für die Ankunft des 1. Halbbildes 25 ms groß. Sind z.B. für die

Effektberechnung Bilddaten eines zweiten Zuspielers (Kamera) notwendig, wird deutlich, dass die Wahrscheinlichkeit der rechtzeitigen Ankunft (im Zeitrahmen) der Daten des 1. Halbbildes der zweiten Quelle kleiner als 100% ist. D.h. für einen Bildmischer typische Übergänge (Mix oder Wipe) oder störungsfreie Umschaltvorgänge zwischen zwei Videoquellen sind nicht zuverlässig innerhalb der definierten 40 ms Gesamtlatenz möglich. Die Gesamtlatenz muss auf 2 Vollbilder (80 ms) angehoben werden. Diese Aussage gilt auch für progressiv arbeitende Systeme.

Synchronisation

Die Problematik der Synchronisation im Studio bezieht sich auf zwei Dimensionen. Einerseits benötigt jedes Gerät zur Aufnahme und Wiedergabe von audio-visueller Information einen einheitlichen zentralen Takt, der die Gleichzeitigkeit der Aufnahme bzw. Wiedergabe von Informationseinheiten (Frames, Samples) sicherstellt. Diese Funktionalität wird aktuell durch die Verteilung eines zentralen Referenzsignals (Black Burst, Tri-Level-Sync) bewerkstelligt, das eine Reihe von Nachteilen aufweist. So ist es nicht einfach möglich, Audiosysteme mit 59,94-Hz-Videosystemen zu verkoppeln. 50- und 59,94-Hz-Systeme benötigen weiterhin eigene Referenzsignale, was eine Multi-Standard-Produktion erschwert. Wegen der relativ großen Bandbreite von ca. 5 MHz ist die Verteilung von Black-Burst-Signalen über größere Distanzen nur mit aufwändiger Verstärkung möglich.

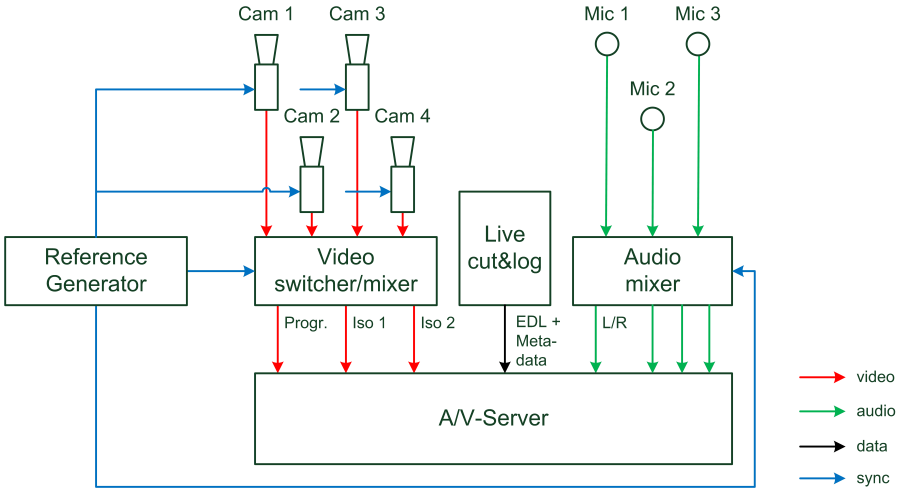


Abbildung 6.9: Multikamera Studioszenario nach [EBU08]

Andererseits müssen die Informationseinheiten bearbeitungsresistent gekennzeichnet werden, damit z.B. für alle Abläufe nach der Aufnahme (Postproduktion, Distribution, Archiv) die verschiedenen Essenz- und Metadatenströme miteinander verkoppelt bleiben. Der aktuell genutzte Timecode-Mechanismus nach SMPTE S12M kann dafür nicht mehr

alle Anforderungen erfüllen. So können Vollbildfrequenzen größer 30 Hz nicht robust gekennzeichnet werden, was neben 50- bzw. 60-Hz-progressiv-Formaten insbesondere Slow-Motion-Aufnahmen betrifft. Aufgrund der nicht ganzzahligen Bildfrequenz bei NTSC-Video ist die Audio-/Video-Synchronisation aufwändig (*drop frames*). Außerdem ist die Kennzeichnung auf eine 24-Stunden-Periode beschränkt.

Beider Dimensionen nimmt sich eine gemeinsame Arbeitsgruppe aus EBU und SMPTE[8] an. Bis zum Februar 2008 stellte sie Nutzeranforderungen zusammen, sammelte bis zum Juni 2008 über einen „Request for Technology“ Vorschläge für eine Referenztechnologie und diskutiert und kombiniert seitdem die eingegangenen Vorschläge. Die Referenztechnologie soll gleichzeitig traditionelle Synchronisationssignale aus der analogen Welt ersetzen und eine eindeutige und bearbeitungsresistente Verkopplung zwischen Essenz- und Metadaten ermöglichen [EBU08]. Nach Abschluss ihrer Arbeit wird die Task Force ihre Ergebnisse zur Standardisierung bei der SMPTE einreichen („Request for Standardisation“). Das Konzept der vorliegenden Arbeit sieht die Einbindung dieses neuen Standards vor. Um den Ergebnissen der Task Force nicht vorzugreifen baut das Konzept auf vorhandene Mechanismen zur Synchronisation auf und lässt Raum zur Integration der neuen Referenztechnologie.

Die Synchronisation zwischen Essenz- und Metadaten innerhalb eines Datenstromes erfolgt über die Mechanismen des Material Exchange Formates (vergleiche Abschnitt 5.2). Darüber hinaus müssen Daten in Be- und Verarbeitungsprozessen auch stromübergreifend synchronisiert werden. Dafür werden Zeitmarken *(time related labels, time stamps)* verwendet, die als Teil einer Referenztechnologie zukünftig standardisiert werden.

6.4.4 Störungsfreies Umschalten

Neben ihrer distributiven Eigenschaft ermöglicht eine Kreuzschiene im Studiobereich den störungsfreien Umschaltvorgang, der z.B. bei Video in der vertikalen Austastlücke erfolgt und so ohne Bildstörungen umgesetzt werden kann. Voraussetzung für diese Vorgehensweise ist die Signalverteilung über isochrone Schnittstellen wie SDI sowie die einheitliche Taktung aller Geräte durch den Studiotakt.

Da die Übertragung über datagrammbasierte Netzwerke nicht isochron erfolgt, gibt es keine deterministischen Phasenbeziehungen zwischen den einzelnen Datenströmen am zentralen Switch. Am Beispiel Videoübertragung bedeutet das, dass die Daten eines Vollbildes zweier Ströme in den seltensten Fällen zeitgleich (am Switch oder Bildmischer) vorliegen. Deshalb ist eine adaptive Pufferung beim Umschaltvorgang unumgänglich. Abbildung 6.10 stellt drei Varianten für die Umsetzung des Umschaltvorgangs gegenüber.

Zunächst kann der Umschaltvorgang auf dem Switch (a) erfolgen, der mit zusätzlicher Intelligenz ausgestattet werden muss, um einen störungsfreien Schnitt an den Vollbild-

[8] Task Force on Time Labeling and Synchronization

grenzen der beteiligten Datenströme zu realisieren. Diese Variante kann latenzarm implementiert werden, widerspricht aber zur Zeit dem Ziel, Standardhardware einzusetzen. Zunehmend werden in Netzwerkknoten programmierbare Prozessoren integriert, die die Analyse von Netzwerkpaketen bis zur Applikationsschicht erlauben („Layer-7-Switch“). Somit könnten zukünftig Vollbildgrenzen eines Videodatenstromes detektiert und ein störungsfreier Umschaltvorgang auf dem Switch ermöglicht werden.

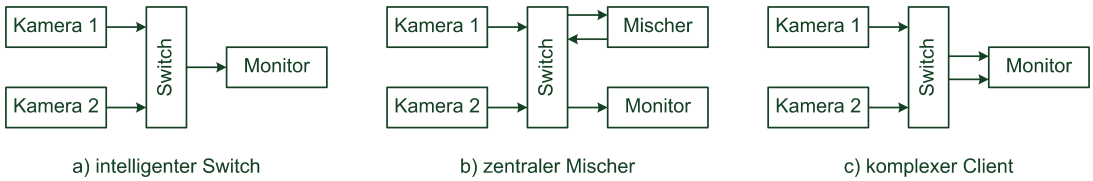


Abbildung 6.10: Varianten zur Umsetzung eines störungsfreien Umschaltvorgangs

Weiterhin kann der Umschaltvorgang durch eine Workstation im Netzwerk (Mischer) erfolgen, die daneben weitere Aufgaben übernimmt (b). Hierbei ist die zusätzliche Latenz durch die Übertragung vom und zum Switch zurück zu berücksichtigen. Schließlich ist es denkbar, die Funktionalität des störungsfreien Umschaltens in alle Empfänger zu implementieren (c), die zu diesem Zweck mit jeweils zwei Datenströmen zeitgleich versorgt werden müssen. Neben der Netzwerkauslastung wachsen damit auch Komplexität und Kosten jedes Empfängers. Theoretisch ist auch eine Kombination aus (a) und (b) denkbar: Ein mit entsprechend vielen Netzwerkschnittstellen (NICs) ausgestatteter PC als zentrales Netzwerkelement übernimmt neben der Bearbeitung der Essenzinformationen auch netzwerkspezifische Aufgaben eines Switches. Praktisch ist diese Lösung nur begrenzt kostengünstig realisierbar. Mit dem Ziel einer zeitnahen Implementierung wird im vorliegenden Konzept der zentrale Mischer (Variante b) ausgewählt. Die grundlegende Problematik des störungsfreien Umschaltvorgangs ist aber in allen drei Fällen identisch.

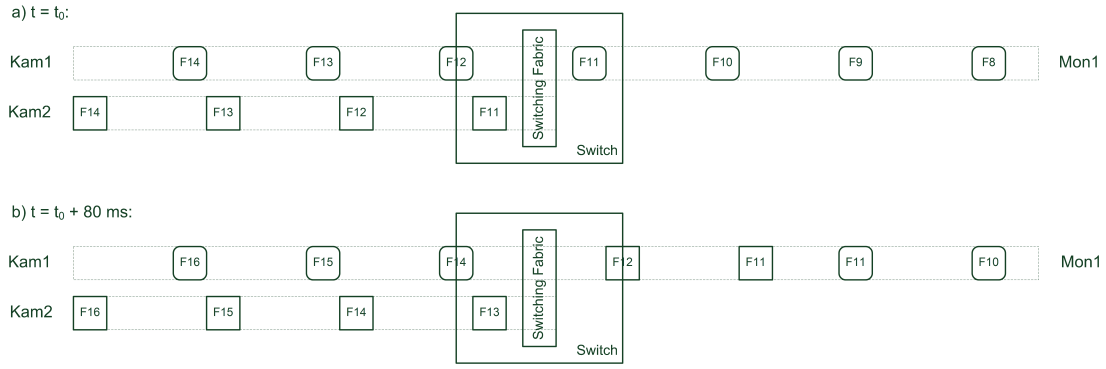


Abbildung 6.11: Realisierung eines störungsfreien Umschaltvorgangs

In Verbindung mit dem File-Streaming (siehe Abschnitt 6.3) und dem Burst-Modus,

der auch mit dem Ziel freier Prozessorkapazitäten der am Sendeprozess beteiligten PCs forciert wird (siehe Abschnitt 6.4.2), kann ein störungsfreier Umschaltvorgang folgendermaßen realisiert werden. Die Pakete eines Videostromes werden nicht kontinuierlich sondern (z.B vollbildweise) zusammengefasst auf das Netzwerk gesendet. Daraus ergibt sich für den Datenstrom ein stoßartiger periodischer Zeitverlauf. In Abbildung 6.11 werden die Daten eines Vollbildes in den ersten 10 ms eines 40 ms Intervalls übertragen, danach ist der Übertragungskanal für 30 ms nicht belegt. Diese Pause wird für den Umschaltvorgang genutzt. Sollte diese Pause nicht ausreichen, muss ein Datenstrom gepuffert werden. Durch mehrmaliges Umschalten zwischen zwei Quellen können Latenzen kumuliert werden, was durch intelligente Schnittpunktwahl zu vermeiden ist.

6.4.5 Datenflusssteuerung auf PC-Basis

Die Etablierung eines Datenflusses in einem IT-Netzwerk wird vom Sender initiiert. Für eine zentrale Steuerung aller Datenflüsse in einem System mit einer Vielzahl von Sendern ist es deshalb notwendig, ein Steuerungsprotokoll einzusetzen. Diese Aufgabe wird von der zentralen Verwaltungsinstanz übernommen, die damit gleichzeitig Überlastsituationen vermeidet.

Flexible Gestaltungsmöglichkeiten von Softwaresystemen erlauben die Nutzung im Studiobereich verbreiteter Metaphern und damit Benutzungsoberflächen, die mit geringem Einarbeitungs- und Schulungsaufwand von einem Nutzer bedient werden können. Neben der gewohnten Kreuzschienenmetapher, die einer ausgewählten Senke eine Datenquelle zuordnet, ermöglicht die graphorientierte Metapher eine intuitive Bedienung. Sie ordnet Quellen und Senken räumlich oder logisch an, wobei sich der Datenfluss jederzeit anhand gerichteter Verbindungslinien nachvollziehen lässt (siehe Abbildung 6.12). Eine PC-basierte Datenflusssteuerung erlaubt ein Umschalten zwischen verschiedenen Metaphern und lässt sich so an Nutzerpräferenzen anpassen. Die Metaphern arbeiten dabei auf der gleichen Steuerungslogik.

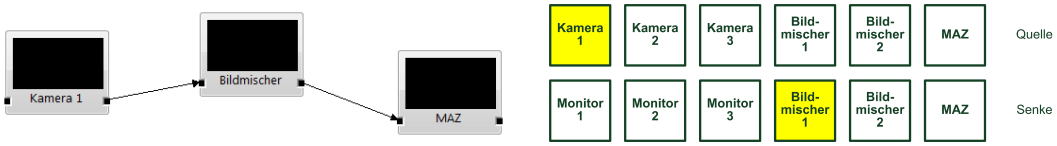


Abbildung 6.12: Methaphern für die Oberfläche einer Datenflusssteuerung [Ote08]

Unabhängig von der Wahl der Metapher kann die Steuerungsoberfläche von beliebigen Clients aus aufgerufen und bedient werden, wenn die Darstellung clientseitig in HTML-Browsern erfolgt. Zu diesem Zweck können etablierte Webtechnologien wie AJAX[9] ge-

[9] *Asynchronous JavaScript and XML* bezeichnet ein Konzept der asynchronen Datenübertragung zwischen einem Server und dem Browser, das den partiellen HTML-Seitenaufbau ermöglicht und damit einen kompletten Neuaufbau vermeidet

nutzt werden. Die Oberflächen können sowohl mithilfe üblicher PC-Eingabegeräte wie Maus und Tastatur als auch über berührungssensitive Bildschirme intuitiv bedient werden. [Ote08]

Im zugrunde liegenden Konzept werden Daten über IP-Multicast übertragen. Jedem Sender ist dabei eine Multicastgruppe zugeordnet, an die die Daten übertragen werden. Der Steuerungsaufwand beschränkt sich zunächst auf die An- und Abmeldung von Empfängern an diesen Multicastgruppen. Da Protokolle zur Organisation von Multicastübertragungen wie IGMP (*Internet Group Management Protocol*) vorsehen, dass diese An- und Abmeldung vom Empfänger ausgeht, muss dem Empfänger über das Steuerungsprotokoll die entsprechende Multicastadresse vermittelt werden. Daneben muss das Steuerungsprotokoll auch Befehle übertragen können, die Parameter wie Priorisierung der Threads und Prozesse oder das Einstellen des Burstintervalls ermöglichen. Die beschriebene Funktionalität erlaubt so eine latenzarme und sichere Übertragung und kann durch ein einfach aufgebautes Steuerungsprotokoll mit entsprechend kurzen Befehlen implementiert werden.

Neben der Anforderung, kurzfristige Änderungen am Datenflussrouting vorzunehmen, sollte die zentrale Steuerungsinstanz Voreinstellungen unterstützen, die vergleichbar mit einer Kreuzschiene nach dem Herstellen der Betriebsbereitschaft bereits sinnvolle Datenübertragungen (z.B. zwischen Kameras und Vorschaumonitoren) etabliert. Die Voreinstellungen können effektiv in XML-Dateien abgelegt werden.

6.5 Systemevaluation

Nachfolgend sollen die theoretischen Erkenntnisse des Gesamtsystemkonzeptes anhand konkreter Messungen an einem Prototypen evaluiert werden. Dazu werden zunächst die prototypische Implementierung und verwendete Messverfahren beschrieben. Im Anschluss erfolgt die detaillierte Darstellung der durchgeführten Messreihen. Die Auswertung der Messergebnisse stellt zum Schluss die Grundlage für die rekursive Parameterbestimmung des Prototyps und für Optimierungsansätze dar.

6.5.1 Prototyp: Das Projekt ITTV

Zur Verifikation des Systemkonzeptes wurde im Rahmen des ITTV-Projektes[10] ein Prototyp implementiert, der wesentliche Funktionalitäten des Datentransports und dessen Steuerung bereitstellt. Die prototypische Implementierung basiert auf einem Gigabit-Ethernet, weil notwendige Komponenten (NICs, Switch, Kabel) zum Zeitpunkt des Aufbaus preiswert zur Verfügung standen. Die damit verbundene Netto-Datenrate von ca.

[10] ein Zusammenschluss Forschungsarbeiten im Bereich „IT-basierte TV-Produktion“ am Institut für Medientechnik der TU Ilmenau

800 MBit/s (siehe Abschnitt 6.5.3.1) erlaubt die umkomprimierte Übertragung von professionellen Videosignalen in Standardauflösung nach ITU.R BT.601. Für die Übertragung von HD-Video benötigte Datenraten über ein GBit/s können durch 10-Gigabit-Ethernet-Equipment bereitgestellt werden, sobald dieses preiswert verfügbar ist.

In Abbildung 6.13 ist der Aufbau eines Prototypen in einer Minimalversion dargestellt. Der Verbund aus drei PCs deckt die Hauptaufgaben eines Studio-Netzwerkes ab: Aufnahme, Bearbeitung und Anzeige von Essenzdaten. Dieses Netzwerk kann bis zu einer Grenze, die durch die physikalischen Anschlussmöglichkeiten und Verarbeitungsressourcen des zentralen Switches bestimmt wird, erweitert werden.

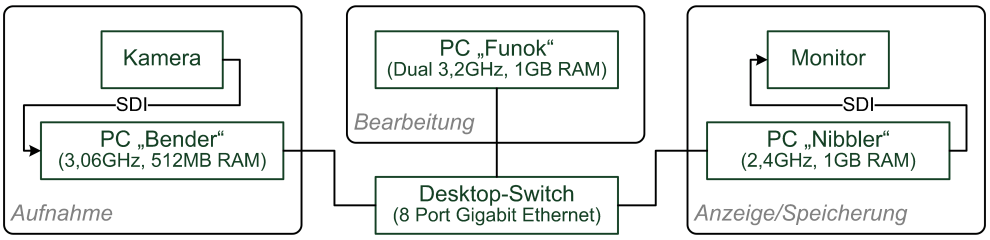


Abbildung 6.13: Praktischer Aufbau (Minimalvariante)

Die Aufgabenbereiche stellen flexible Möglichkeiten zur Implementierung von Geräten und Grundlage für die Erweiterung des Netzwerkes dar:

- Aufnahme: Videosignalquellen (Kameras), Audiosignalquellen (Mikrophone), Datenquellen (Grafik- oder Schriftgeneratoren)
- Bearbeitung: Bildmischer, Chromakeyer, Delays
- Anzeige/Speicherung: Vorschau-Monitore, Teleprompter, Videoserver

Im Rahmen des ITTV-Projektes erfolgte die prototypische Implementierung eines universellen Bearbeitungssystemes auf Basis einer Consumer-Grafikkarte. Der Prototyp setzt die Funktionalität eines einfachen Bildmischers um, d.h. er kann zwei Eingangsvideoströme auf ein Ausgangsvidoestrom mischen/umschalten. Die dafür notwendigen Algorithmen werden nicht auf der CPU des PCs, sondern auf einer über PCIe angeschlossenen Grafikkarte berechnet. Neben der herkömmlichen SDI-Schnittstelle steht dabei auch die oben beschriebene Netzwerkschnittstelle für die Zu- und Abführung von Videosignaldaten zur Verfügung. [Mün08]

Insbesondere die Geräte zur A/V-Signalaufnahme und -anzeige stellen momentan für den Broadcastbereich spezialisierte Einheiten dar. Zunehmend werden diese Geräte aber von der steigenden Leistungsfähigkeit des „Prosumer“-Bereiches und den damit verbundenen Umsatzzahlen profitieren, sodass hier z.T. deutliche Kosteneinsparungen zu verzeichnen sein werden. Im Bereich der Bearbeitung wird sich der Trend der Integration mehrerer Prozesse auf einem Gerät mit steigender Leistungsfähigkeit der PC-Hardware fortsetzen.

Die Signalanzeige im Videobereich profitiert schon heute von der Verfügbarkeit großer LCD[11]-Panels für Multisignal-Displays.

6.5.2 Messverfahren

Für die feingranulare Latenzmessung von einzelnen Programmteilen wurde auf den Prozessor-internen *High Performance Timer* (HPT) zurückgegriffen, der im Takt der CPU einen Wert inkrementiert. Die theoretisch erreichbare Genauigkeit ist somit von der Leistungsfähigkeit des Prozessors abhängig: Bei einer Taktfrequenz von 3 GHz beträgt diese $\frac{1}{3\ GHz} = 0,33\ ns$. Bei der Nutzung von höheren Programmiersprachen ist dieser Wert praktisch nicht erreichbar, da ein Funktionsaufruf aus mehreren Maschinenbefehlen besteht. Zusätzliche Ungenauigkeiten durch Unterbrechungen des Funktionsaufrufs (durch den Taskscheduler) können vermieden werden, wenn für den Prozess bzw. Task die maximale Prioritätsklasse (*Realtime* und *Time-Critical*) gesetzt wird. Die nutzbare Genauigkeit des HPT liegt dann bei einer Mikrosekunde (1 μs). Die Aufbereitung und Ausgabe der Messwerte wurde über einen zeitunkritischen Kontrollthread realisiert (vergleiche Abbildung 6.14).

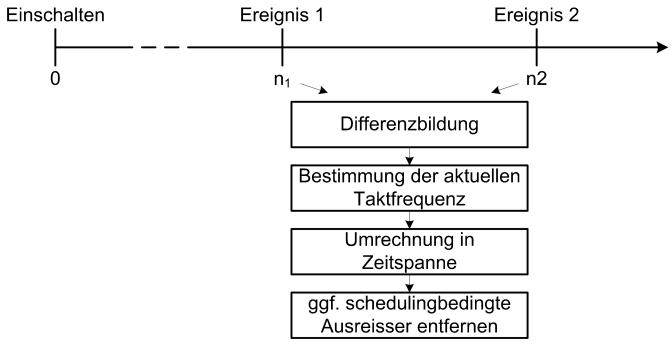


Abbildung 6.14: Zeitspannenmessungen mit der High Performance Timer [Ote08]

Weiterhin wurde auf den im Betriebssystem integrierten Systemmonitor mit zugehöriger Protokollfunktion zurückgegriffen. Der Systemmonitor erlaubt die Messung vieler systeminterner Vorgänge. Für die Messungen am Prototypen wurden Indikatoren der Bereiche „CPU“, „Netzwerk“ und „laufende Prozesse“ ausgewählt.

6.5.3 Messreihen und Messergebnisse

Im Folgenden werden Messaufbauten beschrieben und resultierende Ergebnisse diskutiert. Zunächst erfolgt die Bestimmung der erreichbaren Nettodurchsatzraten in Standardnetzwerken und deren Abhängigkeit. Latenzmessungen des oben beschriebenen Netz-

[11] Liquid Crystal Display

werkstapels belegen im Anschluss die Praxistauglichkeit des theoretischen Konzeptes. Danach wird der Einfluss der Multicastübertragung und der Be- und Verarbeitung auf PC-Plattformen – insbesondere auf Basis der Nutzung von GPUs – bestimmt. Zum Schluss wird das Gesamtsystem ausgemessen.

Die Videobildinformation wird dabei mithilfe von Capture-Hardware vollbildweise im Hauptspeicher abgelegt. Alle Messungen gehen von dieser Voraussetzung aus. Die für den Capturevorgang benötigte Latenz wird bewusst nicht berücksichtigt, da mittelfristig Bildquellen mit Netzwerkschnittstellen verfügbar sein werden, deren Verwendung diese Latenz umgeht (vergleiche Abbildung 6.8 auf Seite 97).

6.5.3.1 Abhängigkeit des erreichbaren Durchsatzes

Zur Bestimmung der maximal möglichen Netto-Datenrate in einem einfachen Gigabit-Netzwerk können Netzwerkgeneratoren genutzt werden, die mithilfe einer Client-Server-Anwendung über einen definierten Zeitraum ein Bitmuster mit maximal zur Verfügung stehenden Übertragungsrate senden.

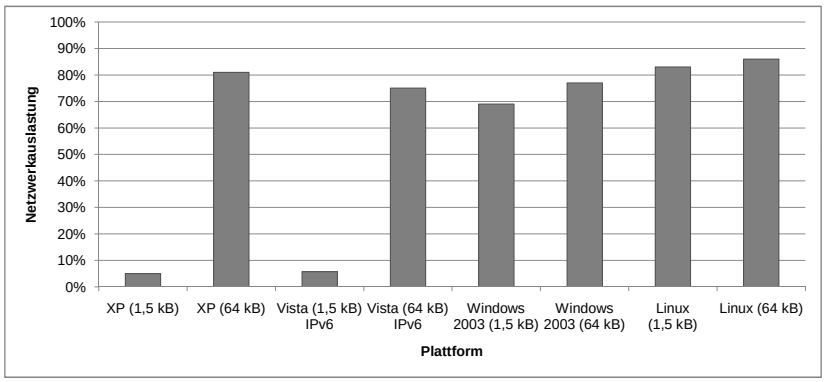


Abbildung 6.15: Netzwerkdurchsatz in Abhängigkeit von IP-Paketgröße und Betriebssystem [Ote08]

In Abbildung 6.15 ist die Auswirkung kleiner IP-Paketgrößen auf den erreichbaren Durchsatz bei den Standardbetriebssystemen *Windows XP* und *Vista* zu erkennen. Für die Messungen mit Windows XP wurde eine MTU[12] von 9 kB eingestellt, für alle anderen Tests betrug die MTU 1,5 kB. Bei IP-Paketgrößen von 1,5 kB findet keine Fragmentierung statt, dennoch bleibt der Durchsatz sehr gering. Serverbetriebssysteme wie Windows 2003 oder Linux-Distibutionen[13] erreichen auch für kleine Paketgrößen Durchsatzwerte von 70 bis über 80 Prozent. Mit Paketgrößen von 64 kB kann mit allen getesteten Betriebssystemen eine Auslastung der verfügbaren Netzwerkkapazität zu ca. 80 Prozent

[12] Die *Maximum Transmission Unit* gibt die maximale Größe eines Ethernet-Frames und damit die Obergrenze für die Größe von IP-Paketen an, wenn Fragmentierung vermieden werden soll

[13] hier getestet: Ubuntu 7.02

erreicht werden. Bei großen Paketgrößen ist der Protokolloverhead geringer, als bei kleinen Paketgrößen. Nachteilig an großen IP-Paketen ist die Tatsache, dass bei Verlust oder Beschädigung eines Ethernet-Frames 64 kB des IP-Paketes verloren gehen. [Ote08]

Aus den Messergebnissen folgt, dass bei der Anwendung von Standardbetriebssystemen wie *Microsoft Windows XP* die im Konzept geforderten hohen Durchsatzraten nur mit großen Paketen erreicht werden können. Die Zuweisung eines hohen Wertes für die MTU erlaubt weiterhin einen minimierten Overhead durch Fragmentierung.

6.5.3.2 Latenzmessung des implementierten Netzwerkstapels

Zum Nachweis der Echtzeitfähigkeit des in Abschnitt 6.1 vorgestellten Netzwerkstapels wurden feingranulare Messungen am Prototyp mithilfe des *High Performance Timers* (HPT) durchgeführt. Dabei wurde die zusätzliche Latenz der MXF-Kapselung zunächst noch nicht berücksichtigt. Abbildung 6.16 zeigt den Versuchsaufbau: Gemessen wurde die Latenz der Übertragung zwischen zwei PCs (Speicher zu Speicher) über einen zwischengelagerten Switch.

Abbildung 6.16: Messaufbau zur Latenzmessung des implementierten Netzwerkstapels

Die gemessene Gesamtlatenz des prototypisch realisierten Übertragungssystems besteht aus vier Anteilen, die sich mithilfe des aufgestellten Latenzmodells (siehe Abschnitt 6.2) genau bestimmen lassen. Mit den Einschränkungen, dass nur ein Netzwerkknoten verwendet wird ($n = 1$) und dass die Bearbeitung der übertragenen Daten am Empfänger bis auf eine Pufferung eines vollständigen Halbbildes ausbleibt[14], ergibt sich folgende Zusammensetzung der Latenz einer Speicher-zu-Speicher-Übertragung am Prototypen:

$$\begin{aligned} t_{Prototyp} &= \sum_{m-1} t_{\text{ÜBERTRAGUNG}} + \sum_{m} t_{\text{WORKSTATION}}(m) \\ &= t_{\text{Sender}} + t_{\text{Knoten}} + t_{\text{Empfänger}} + t_{\text{puffer}} \end{aligned}$$

Die softwareseitigen Verarbeitungszeiten der Server- und Clientapplikation (t_{Sender} und $t_{\text{Empfänger}}$) konnten genau bestimmt werden, indem kurz vor dem DMA[15]-Transfer an der SDI-Karte und nach dem Übergeben des ersten Paketes an den Netzwerktreiber (bzw. umgekehrt für den Client) HPT-Werte gemessen wurden. Auf gleiche Weise wurde die

[14] die Videodaten werden später auf einem über SDI angeschlossenen Videomonitor angezeigt
[15] *Direct Memory Access* – Speicherdirektzugriff

Dauer des Sendevorgangs eines Halbbildes (t_{puffer}) bestimmt. Zur Berechnung der Übertragungsdauer[16] (t_{Knoten}) wurden vom Client Testpakete zum Server zurückgesendet. Mit der Annahme, dass die Dauer der Übertragung von Client zum Server genauso groß ist, wie umgekehrt, wurde die Einweglatenz durch Halbierung der Zweiwegelatenz berechnet. Über 100 Messungen gemittelt wurden folgende Werte bestimmt (siehe Tabelle 6.1).

	Mittelwert [μs]	**Standardabweichung [μs]**
Software Server (t_{Sender})	1000	561
Netzwerk (t_{Knoten})	1103	55
Bilddauer (t_{puffer})	5335	588
Software Client ($t_{\text{Empfänger}}$)	37	18
Gesamt	7471	605

Tabelle 6.1: Latenz des Netzwerkstapels [Ote08]

Die Latenzmessung des implementierten Netzwerkstapels erfolgte über eine Unicast-Übertragung. Otero zeigte in weitergehenden Versuchen, dass eine fehlerfreie Übertragung durch den Einsatz von Burst-Übertragungsmodus und Netzwerkkarten mit Interrupt-Moderation begünstigt wird [Ote08]. Beide Maßnahmen verhindern die Überlastung der CPU mit Netzwerkverarbeitungsaufgaben, sodass keine Pakete aufgrund fehlender Berechnungskapazität verworfen werden müssen.

Die durchgeführten Messungen gehen pessimistisch von einer Speicher-zu-Speicher-Übertragung aus, d.h. die bestimmte Latenz umfasst die Pufferung eines Halbbildes im Empfänger (t_{puffer}), die ca. 70% der Übertragungslatenz darstellt. Wenn der Bearbeitungsalgorithmus des Empfängers eine Verarbeitung der Bilddaten sofort nach dem Eintreffen der ersten Pakete erlaubt (ohne vorherige Pufferung), kann t_{puffer} im Rahmen der Gesamtsystemlatenz (Quelle – Bearbeitung – Senke) minimiert werden. Für eine studiosynchrone Bildausgabe ist ein Ausgangspuffer zwingend erforderlich.

6.5.3.3 Einfluss der Multicastübertragung

Multicastübertragungen haben einen Berechnungs- und Verwaltungsmehraufwand im Switch zur Folge. Zum einen muss ein Paket vom Switch kopiert und mehrfach gesendet werden. Zum anderen fordert die Etablierung einer Multicastgruppe und die An- und Abmeldung an bzw. von der Multicastgruppe eine zu berücksichtigende Zeitdauer. Beide Latenzen können experimentell bestimmt werden.

Zur Ermittlung des Berechnungsmehraufwandes im Switch wurde der Messaufbau aus Abschnitt 6.5.3.2 um bis zu drei Empfänger erweitert (siehe Abbildung 6.17), die Messmethode blieb identisch. Zunächst wurde der Multicastgruppe nur ein Empfänger zuge-

[16] beinhaltet sowohl die Latenz durch den Netzwerkknoten selbst, als auch die Verzögerung durch die endliche Ausbreitungsgeschwindigkeit der Signale auf dem Kabel

Abbildung 6.17: Messaufbau zur Bestimmung des Multicasteinflusses auf t_{Knoten}

ordnet (1-Multicast), später zwei bzw. drei (2- bzw. 3-Multicast). Es zeigte sich, dass die Latenz jedes zugefügten Multicast-Empfängers um im Mittel 155 μs anstieg, während bereits etablierte Empfänger ihre Verzögerung beibehielten.

$t_{\textbf{Knoten}}$	**Mittelwert [μs]**	**Standardabweichung [μs]**
Unicast	832	59
1-Multicast	993	23
2-Multicast, 1. Empfänger	1047	67
2-Multicast, 2. Empfänger	1205	50
3-Multicast, 3. Empfänger	1357	32

Tabelle 6.2: Latenz in Abhängigkeit der verwendeten Übertragungstechnik [Ote08]

Die festgelegte IP-Paketgröße des Messaufbaus hatte eine Fragmentierung in 37 Ethernetrahmen zur Folge. Im Vergleich Unicast und 1-Multicast benötigt die Multicastübertragung ca. 55 μs länger. Daraus kann geschlossen werden, dass der Switch für die Analyse eines Ethernetrahmens im Mittel ca. 1,5 μs benötigt. Aus der durchschnittlich 155 μs größeren Latenz für jeden zusätzlichen Multicastempfänger folgt eine Latenz durch Weiterleitung eines Ethernetframes von ca. 4,2 μs.

Der zeitliche Verwaltungsmehraufwand durch die Multicastübertragung im Switch wurde von Otero am gleichen Prototypen bestimmt: Für die Etablierung einer Multicastgruppe und das Hinzufügen eines Empfängers wurde eine Latenz zwischen 600 und 650 ms gemessen. Deutlich minimieren lässt sich diese Verzögerung, wenn die Multicastgruppe bereits existiert: Gemessen vom Auslösen des Beitritts zur Multicastgruppe bis zum Empfang erster Daten am Empfänger beträgt die Latenz ca. 10 ms. [Ote08]

Folgende Rückschlüsse lassen sich bezüglich des Prototypen ziehen. Für eine latenzarme Umsetzung müssen Multicastgruppen für jeden Sender definiert werden. Damit wird t_{MCadmin} minimiert. Die Abhängigkeit der Multicast-Verarbeitungslatenz $t_{\text{multicast}}$ von der Anzahl der Empfänger hat zur Folge, dass weiterhin die Empfängeranzahl optimiert, d.h. auf die real genutzte Empfänger begrenzt werden muss (vergleiche Abschnitt 6.4.1).

6.5.3.4 Bearbeitungslatenz

Auch am Bildmischerprototypen, der die Berechnung von Übergangseffekten auf GPU-Hardware erlaubt, wurde eine Reihe von Latenzmessungen durchgeführt, deren Umfang in Abbildung 6.18 verdeutlicht wird.

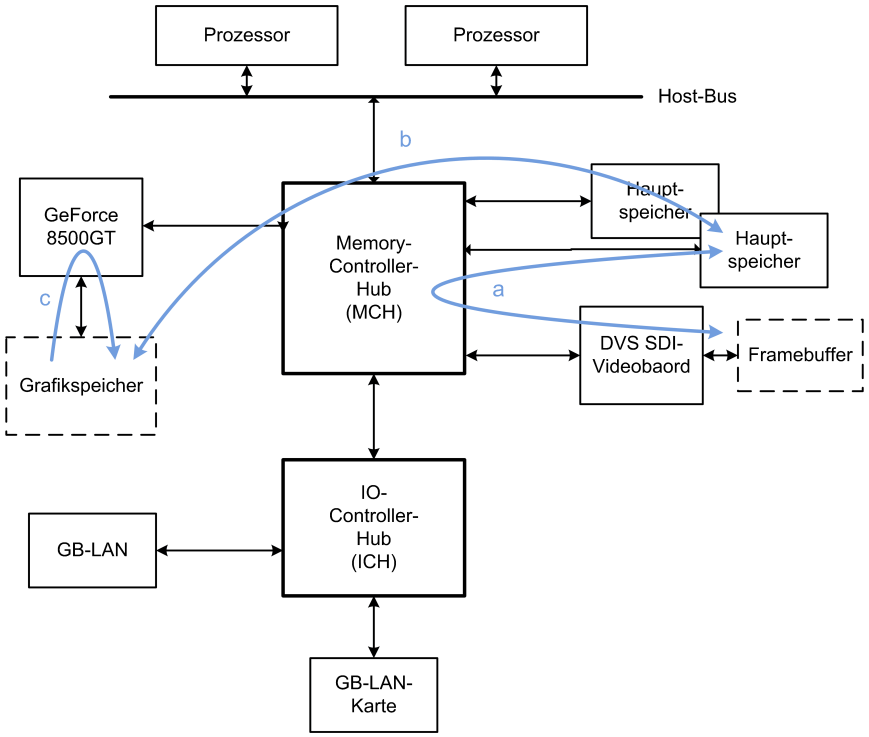


Abbildung 6.18: Visualisierung gemessener Latenzen am Bildmischer-Prototyp

Videodaten, die vollbildweise im Speicher der Schnittstellenkarte zur Verfügung stehen, müssen zunächst in den Hauptspeicher des PC-Systems transferiert werden (a). Von dort erfolgt in einem zweiten Schritt ein Kopierprozess in den Speicher der Grafikkarte (b), bevor die Berechnung auf den Bilddaten nach vorgegebenem Algorithmus stattfinden kann (c). Auf gleichem Weg werden die berechneten Daten wieder auf den Speicher der Schnittstellenkarte kopiert und ausgegeben.

Ausgehend von der Verwendung der SDI-Schnittstelle an den Ein- bzw. Ausgängen für Videosignale wurde zunächst die Latenz ermittelt, die durch das Ansammeln der Daten eines Videovollbildes im Speicher der SDI-I/O-Karte und deren Transfer über DMA in der Hauptspeicher des PC-Systems entsteht. Start- und Endzeitpunkt für die Messung mithilfe des *High Performance Timers* (HPT) wurden vor und nach Codezeilen der vom SDI-I/O zur Verfügung gestellten Funktionen zur Reservierung, Kopieerstellung

und Freigabe eines entsprechenden Speicherbereichs auf der SDI-Karte[17] gesetzt. In einer weiteren Messreihe wurde auf Grundlage einer Demoanwendung des Herstellers DVS GmbH die Latenz des DMA-Transfers eines Vollbildes bestimmt, wenn das Vollbild bereits vollständig im Speicher der SDI-Karte vorliegt. Tabelle 6.3 fasst die Messergebnisse zusammen.

	Mittelwert [μs]	**Standardabweichung [μs]**
Input: ein Vollbild	38.557	123
Input: zwei Vollbilder	39.931	299
Output: ein Vollbild (`getbuffer`)	17	10
Output: ein Vollbild (`putbuffer`)	5.758	155
DMA-Transfer Karte → Hauptspeicher (ein Vollbild)	1.930	98
DMA-Transfer Hauptspeicher → Karte (ein Vollbild)	2.027	104

Tabelle 6.3: Latenz des Kopiervorgangs zwischen SDI-Karte und Hauptspeicher (a) [Mün08]

Die verwendete SDI-Schnittstellenkarte[18] bietet die Möglichkeit, parallel zwei SDI-Eingänge zu verarbeiten und einen SDI-Ausgang zur Verfügung zu stellen. Dementsprechend wurden in den Messreihen ein Eingang (ein Vollbild) und zwei Eingänge (zwei Vollbilder) verwendet. Der geringfügige Unterschied zwischen beiden Messwerten ist ein Beleg dafür, dass beide Eingänge weitestgehend unabhängig voneinander und parallel durch die Karte verarbeitet bzw. transportiert werden. Bei der Nutzung von zwei Eingängen überschreitet der Messwert gelegentlich die Vollbildrate von 40 ms. Ein Bildfehler wird vermieden, indem der Speicher auf der SDI-Karte mehrere Frames umfasst und so das folgende Vollbild aufgezeichnet werden kann, obwohl das aktuelle noch nicht vollständig kopiert wurde. Das SDI-I/O-SDK fängt solche Überschreitungen mit einem internen Sicherheitspuffer ab, der die minimale Eingangs- zu Ausgangsverzögerung des Systems auf zwei Frames anhebt.

Der Rücktransport der Daten eines Vollbildes vom Hauptspeicher auf die SDI-Schnittstellenkarte (inkl. Reservierung, Kopieerstellung und Freigabe des Speicherbereichs) erfolgt mit ca. 6 ms hingegen schneller, da nicht auf die Ankunft der Daten im Speicher gewartet werden muss. Tabelle 6.3 zeigt weiterhin, dass der eigentliche DMA-Transfer der Daten in beide Richtungen mit ca. 2 ms sehr schnell durchgeführt wird. Auch der DMA-Transfer zwischen Hauptspeicher des PC-Systems und dem Bildspeicher der Grafikkarte erfolgt sehr schnell (vergleiche Tabelle 6.4).

Neben der Transportzeit vom Hauptspeicher auf die Grafikkarte ist für die Gesamtverarbeitungslatenz die Dauer der Berechnungen auf der GPU von Interesse. Tabelle 6.5 stellt

[17] `getbuffer-` bzw. `putbuffer`-Schleife
[18] DVS Centaurus II, PCIe-Version, Hersteller: Digital Video Systems (DVS) GmbH

	Mittelwert [μs]	Standardabweichung [μs]
Hauptspeicher → Grafikkarte (ein Vollbild)	1.005	56
Hauptspeicher → Grafikkarte (zwei Vollbilder)	2.480	131
Grafikkarte → Hauptspeicher (ein Vollbild)	911	46

Tabelle 6.4: Latenz des DMA-Transfers zwischen Hauptspeicher und Speicher der Grafikkarte (b) [Mün08]

die Ergebnisse durchgeführter Messreihen zusammen. Zugrunde gelegt wurde dabei der Mix-Effekt, der beide Eingangsvideobilder in Abhängigkeit eines Alphawertes gleichzeitig auf dem Ausgangsbild darstellt. Um die Vergleichbarkeit der Berechnung auf den Grafikkarten und CPUs sicher zu stellen, muss neben den reinen GPU-Berechnungszeiten der DMA-Transfer vom und zum Hauptspeicher des PC-Systems berücksichtigt werden. Tabelle 6.5 lässt erkennen, dass ältere Grafikkarten-Hardware im Vergleich mit aktuelleren CPU-Modellen langsamer sein kann. Münstermann hat durch weitergehende Messungen nachgewiesen, dass die Anzahl von Threads pro Zeile bei der Implementierung von Kerneln Auswirkungen auf die Berechnungszeit hat. Die optimale Konfiguration muss aber für jeden Kernel in Abhängigkeit des verwendeten Algorithmus neu bestimmt werden [Mün08].

	Mittelwert [μs]	Standardabweichung [μs]
GPU (GeForce 8500) 16 Streamprozessoren, 900 MHz Shadertakt, Kernel mit 16 Threads/Zeile	9.355	13
GPU (GeForce 8800) 112 Streamprozessoren, 1.500 MHz Shadertakt, Kernel mit 16 Threads/Zeile	1.965	15
CPU (Intel Dual Xeon; 3,2 GHz)	11.521	225
CPU (AMD Athlon 64; 2,21 GHz)	4.847	225

Tabelle 6.5: Nettolatenz für die Berechnung eines Mix-Effektes (c) [Mün08]

6.5.3.5 Gesamtsystemtest

Für die MXF-Implementierung am Prototyp wurde ein Software Development Kit (SDK)[19] verwendet, dass über objektorientierte Schnittstellen eine einfache Umsetzung des komplexen MXF-Standards erlaubt. Die Bestimmung der Verzögerung, die durch die MXF-

[19] MXF::SDK von MOG Solutions, das in Zusammenarbeit mit dem Institut für Rundfunktechnik (IRT) in München entwickelt wurde; eingesetzte Version: 4.0.1.161

Verarbeitung entsteht, wurde durch Einfügen entsprechender HPT-Messpunkte im Programmcode erreicht. Für die reine MXF-Verarbeitung innerhalb des SDKs wurde eine Latenz für das Verpacken eines Vollbildes von ca. 1,2 ms bestimmt [Ote08]. Problematisch ist die vollbildorientierte Arbeitsweise des SDKs an den Schnittstellen nach außen und auch an SDK-internen Schnittstellen.

In einem einfachen Gesamtsystemtest wurde die Eingangs-zu-Ausgangs-Latenz mithilfe eines Bildmischers mit Frame-Store[20] bestimmt. Dazu wurden anhand einer geeigneten Videoseqenz[21] das Eingangs- und Ausgangssignal zeitgleich (Mix-Effekt mit $\alpha = 0,5$) angezeigt und (mit der Einstellung einer korrespondierenden Verzögerung über den Frame-Store) zur Deckung gebracht. Abbildung 6.19 verdeutlicht den Versuchsaufbau.

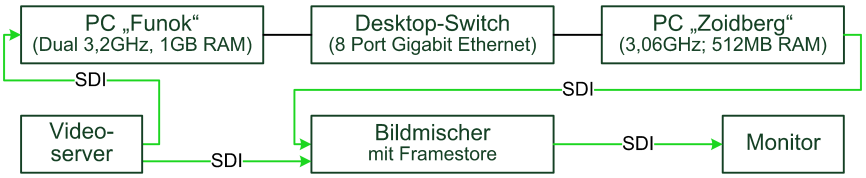


Abbildung 6.19: Messaufbau Gesamtsystem-Latenzmessung

Prinzipbedingt können Latenzmesswerte mit diesem Versuchsaufbau nur im ganzzahligen Vollbildbereich gemessen werden. Da fast alle verwendeten Prozesse (SDI-I/O, MXF-Verarbeitung) nur vollbildgetaktete Schnittstellen nutzen, ist diese Genauigkeit für prinzipielle Aussagen hinreichend. Um eine Zuordnung einzelner Latenzkomponenten zu ermöglichen, wurden verschiedene Messungen durchgeführt (siehe Abbildung 6.20).

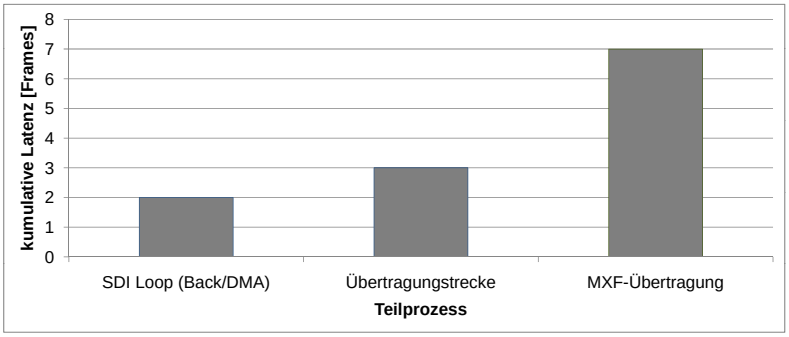


Abbildung 6.20: Kumulative Darstellung der Latenzkomponeten des Übertragungssystems [Ote08]

Die benutzten SDI-I/O-Karten der Fa. DVS stellen die am Eingang anliegenden Videosignale auch am Ausgang zur Verfügung. Dabei werden die Signale wahlweise direkt (*Loop*

[20] ermöglicht die Verzögerung von Videosignalen in Vollbildschritten über eine Zwischenspeicherung von ganzen Vollbildern

[21] ein in der Bildebene rotierender Schriftzug erlaubt durch Winkeländerung über die Zeit eine einfache Beurteilung der Deckungsgleichheit zweier gleichzeitig sichtbarer Bilder

Back) oder nach einem DMA-Transfer in den PC-Hauptspeicher und zurück am Ausgang ausgegeben. In beiden Fällen kann eine Verzögerung von zwei Frames erreicht werden. Wird weiterhin eine Speicher-zu-Speicher-Übertragung (vergleiche Abschnitt 6.5.3.2) zwischen den PCs etabliert, erhöht sich die Latenz aufgrund der Synchronisation auf den Studiotakt bei der Darstellung auf drei Frames, obwohl die Übertragung selbst weniger als 10 ms in Anspruch nimmt. In einem nächsten Schritt wurde die Verzögerung der MXF-Verarbeitung mit zusätzlichen vier Frames bestimmt.

6.5.4 Diskussion der Messergebnisse und Optimierungsansätze

Bei der Auswahl eines Betriebssystems für die am Netzwerk angeschlossenen Workstations ist der erreichbare Netzwerkdurchsatz zu beachten. Die Messungen in Abschnitt 6.5.3.1 haben gezeigt, dass Standard-Desktop-Betriebssysteme wie *Microsoft Windows XP* und *Vista* bei kleinen Paketgrößen die zur Verfügung stehende Übertragungskapazität des Netzwerkes nicht nutzen können. Erst bei größeren Paketgrößen kann das Netzwerk zu 80 % ausgelastet werden. Andere Desktopbetriebssysteme (diverse Linux-Derivate) und vor allem Serverbetriebssysteme wie *Microsoft Windows 2003 Server* weisen die Einschränkung bei kleinen Paketgrößen nicht auf.

Im direkten Zusammenhang mit zusätzlicher Einschränkung der nutzbaren Datenrate steht die Problematik der Fragmentierung, also der eventuellen Aufteilung großer Layer-3-(IP)-Pakete auf Layer-2-(Ethernet)-Rahmen. Die Wahl der MTU beeinflusst zum einen das Nutzdatenverhältnis des resultierenden Datenstromes und zum anderen die Größe des Verlustes bei fehlerhaft übertragenen Daten. Bei Verwendung von handelsüblicher Netzwerkhardware (insbesondere Switches) und dem Einsatz von Multicast-Übertragungen kann die MTU nicht beliebig dimensioniert werden. Im Prototypen ist die MTU so auf 1.500 kB festgeschrieben.

Die Latenzmessungen des implementierten Netzwerkstapels (vergleiche Abschnitt 6.5.3.2) belegen die Einsparpotentiale durch Vermeidung von Puffern zur Synchronisation auf einen festen Systemtakt während der Bearbeitung. Eine Speicher-zu-Speicher-Übertragung zwischen zwei PCs über einen Ethernetswitch ist mit einem Standard-Netzwerkstapel in weniger als 10 ms realisierbar. Der dabei größte Anteil, der zur Übertragung einer Bilddauer benötigt wird, ist direkt abhängig vom durch das Netzwerk zur Verfügung gestellten Durchsatz und kann somit durch schnellere Netzwerke (100-Gbit-Ethernet) oder Portbündelung (z.B. über *Link Aggregation Control Protocol* – LACP) minimiert werden. Weiterhin können die Daten im Abhängigkeit des Bearbeitungsalgorithmus am Empfänger bereits bearbeitet bzw. angezeigt werden, bevor die Übertragung des vollständigen Voll- bzw. Halbbildes abgeschlossen ist. Der Einsatz des Burstübertragungsmodus vermeidet weiterhin die Auslastung der System-CPU mit Netzwerkaufgaben und stellt den Workstations CPU-Kapazität zur allgemeinen Nutzung frei.

Die hohe Verzögerung durch die MXF-Verarbeitung ist im Wesentlichen auf vollbildbasierende Schnittstellen des verwendeten SDKs zurückzuführen. Für den Einsatz im echtzeitkritischen Bereich ist die Latenz von vier Vollbildern zu groß. Hier muss eine latenzoptimierte MXF-Implementierung vorgenommen werden.

Die Multicastübertragung stellt die Signalverteilung von einem Sender auf viele Empfänger mit geringen zusätzlichen Latenzen zur Verfügung (vergleiche Abschnitt 6.5.3.3). In einer Standardkonfiguration kann ein störungsfreies Umschalten zwischen zwei Signalquellen aber nicht erfolgen, da (1) der Umschaltzeitpunkt (also das Ab- und vor allem das Neuanmelden an einer Mutlicastgruppe) zeitlich nicht genau determiniert werden kann und (2) die Übertragung nicht isochron erfolgt, verschiedene Datenströme also nicht synchron am Switch anliegen.

Die Messungen der Latenzen durch die Bearbeitung von Essenzdaten auf Standard-PC-Systemen (vergleiche Abschnitt 6.5.3.4) zeigt deutlich, dass das zugrundeliegende Konzept von der weiter fortschreitenden Leistungsfähigkeit von CPUs und GPUs profitiert. Die mit den Leistungssprüngen zwischen den Prozessor-Generationen verbundene Reduzierung der Bearbeitungslatenz ermöglicht die Berechnung von komplexeren Algorithmen in kürzerer Zeit, so dass einzelne Workstations immer mehr Berechnungsaufgaben übernehmen können.

Insbesondere der Gesamtsystemtest (vergleiche Abschnitt 6.5.3.5) macht den Einfluss getakteter Übertragung und Bearbeitung deutlich. Vollbildbasierte Schnittstellen zur Videobearbeitung lassen so die Gesamtlatenz schnell auf mehrere Frames ansteigen. Die Minimierung dieser Schnittstellen und der damit verbundenen Puffer erlaubt eine deutliche Reduzierung der Gesamtsystemlatenz. Vor dem Hintergrund eines Übergangsszenarios kann aber mittelfristig nicht vollständig auf isochrone Schnittstellen verzichtet werden. Langfristig ist die vollständige Integration des Netzwerkanschlusses in die Geräte (Kamera usw.) anzustreben.

7 Zusammenfassung, Ergebnisse und Ausblick

Die vorliegende Dissertation setzt sich mit der Anwendung von Standard-IT-Technologien im echtzeitkritischen Bereich der Fernsehstudioproduktion auseinander. Das abschließende Kapitel fasst die Arbeit zusammen, stellt die Ergebnisse heraus und benennt im Ausblick zukünftige Forschungsaufgaben.

Zusammenfassung

Die Anwendung von paketbasierten Übertragungsverfahren ist für die Datenübertragung im echtzeitkritischen Bereich der Studioproduktion eine attraktive Alternative zu existierenden verbindungsorientierten isochronen Signalschnittstellen. Damit wird eine flexible Infrastruktur möglich, die für die Übertragung beliebiger Daten genutzt werden kann und die somit die Reduktion der Anzahl physikalischer Schnittstellen für Audio-, Video- und Steuerdaten erlaubt. Weil diese Infrastruktur aus Standard-IT-Komponenten und -protokollen besteht, kann auf einen großen Erfahrungs- und Kenntnisstand bei Konzeption, Betrieb und Wartung aufgebaut werden. Die Anforderung liegt im Wesentlichen darin, ursprünglich nicht für Echtzeitkommunikation ausgelegte Technologien entsprechend anzupassen bzw. zu modifizieren.

Vor diesem Hintergrund kombiniert die vorliegende Arbeit Standard-IT-Technologien und ergänzt diese um Konzepte, die die besonderen Anforderungen einer Liveproduktion im Fernsehproduktionsstudio berücksichtigen. Dazu zählen insbesondere hohe Essenzsignalqualität, latenzarme Verarbeitung und zuverlässige Funktionalität. Die vorliegende Arbeit konzentriert sich auf die zwei ersten Aspekte. Zur Bewältigung der komplexen Aufgabenstellung ist diese Arbeit im Kern in vier Teile gegliedert.

Zunächst wird im Teil „Datenübertragung über Netzwerke“ (Kapitel 3) eine Übertragungstechnologie aus Standardkomponenten modelliert (UDP/IP/Ethernet). Parameter zur Bewertung der Netzwerkleistung und Strategien zur Ressourcenteilung (QoS) werden diskutiert. Weiterhin stellt eine Modellierung der Übertragungslatenzkomponenten die Grundlage für Optimierungsansätze dar.

Im Teil „Datenverarbeitung auf PC-Plattformen“ (Kapitel 4) werden Prozessoren zur Verarbeitung von Essenzdaten verglichen und über die PC-Plattform in eine universelle Einheit zur Verarbeitung von Datenströmen für Fernsehstudioanwendungen (Workstation) integriert. Die Analyse PC-interner Komponenten und Abläufe führt zu einer feingranularen Latenzbetrachtung, die für Optimierungsstrategien genutzt werden kann.

Die bisher beschriebene Architektur erlaubt die Übertragung und Bearbeitung beliebiger Daten. Insbesondere Metadaten können somit zu optimierten Abläufen beitragen. Dem Aspekt der Nutzung von Metadaten wird der Teil „Anwendung von Metadaten in linearen Produktionsprozessen“ gerecht. Hauptaugenmerk liegt dabei auf dem Material Exchange Format (MXF), das die gekoppelte Übertragung von Essenz- und Metadaten ermöglicht. „MXF-Streaming“ erlaubt dabei die Nutzung des MXF-Dateiformates auch unter den Anforderungen einer latenzarmen Übertragung und Verarbeitung. Die Arbeit stellt weiterhin Anwendungsszenarien vor, in denen Metadaten auch in echtzeitkritischen Live-Produktionen genutzt werden können.

Die drei o.g. Teile werden im Kapitel 6 zu „Eine(r) Netzwerkarchitektur für Studioanwendungen“ zusammengefasst. Die dabei identifizierten Problemstellungen werden mit Technologien und Konzepten einer Lösung zugeführt. Dazu zählt das Konzept des *File-Streaming*, das die Anwendung von Dateiformaten (wie MXF) auch im echtzeitkritischen Bereich der Fernsehstudioproduktion ermöglicht. *File-Streaming* impliziert aber eine synchrone Datenübertragung, die neue Fragestellungen insbesondere bei der Datenverteilung und -bearbeitung zur Folge hat. Eine prototypische Implementierung bildet abschließend die Grundlage zur Verifikation getroffener Aussagen.

Ergebnisse

Die in dieser Arbeit postulierte Netzwerkarchitektur basiert auf der synchronen Übertragung von Paketen, d.h. es existiert eine zeitliche Grenze für die Ankunft der Pakete, deren Laufzeit durch die Eigenschaften des Netzwerkes beeinflusst wird. Diese Grenze kann in Netzwerken mit geringer Komplexität (ein zentraler Switch verbindet alle Teilnehmer) und bekannter Qualität und Quantität der Teilnehmer bzw. der von ihnen produzierten Datenströme zuverlässig eingehalten werden, ohne zusätzliche QoS-Mechanismen wie *IntServ* oder *DiffServ* zu verwenden. Dennoch ist die Netzwerkarchitektur für die Implementierung von QoS-Mechanismen offen, wenn dies durch das Auftreten zusätzlichen Datenverkehrs notwendig sein sollte. Eine Synchronisation auf einen zentralen Takt findet nur bei der Aufnahme und Wiedergabe von Essenzdaten zur Sicherstellung einer gleichzeitigen Präsentation statt. Sowohl während der Übertragung als auch bei der Bearbeitung von Daten wird zur Vermeidung von kumulierten Latenzen auf diese Synchronisation auf eine zentrale Taktung verzichtet.

Die Bemessung der Gesamtsystemlatenz und die damit verbundene Festlegung einer zeitlichen Grenze für die Ankunftszeit der Pakete beim Empfänger wird durch zwei Komponenten bestimmt. Zum einen ist die Übertragungszeit zwischen den jeweils beteiligten Rechnern (Speicher zu Speicher) zu berücksichtigen. Die vorliegende Arbeit hat gezeigt, dass unter Nutzung eines Standardprotokollstapels diese Komponente in unter 10 ms Latenz pro Halbbild eines 50i-SD-Videosignales zu realisieren ist. Voraussetzung dafür

ist eine entsprechend schnelle MXF-Implementierung. Ist zur Verarbeitung der Bildinformation am Empfänger nicht das gesamte Bild im Speicher notwendig, kann mit der Verarbeitung sofort nach dem Eintreffen der ersten Pakete begonnen werden. Unter dieser Voraussetzung kann die einfache Übertragungslatenz signifikant auf ca. 5 ms reduziert werden. Die Infrastruktur verzichtet mit dem Ziel einer latenzarmen Implementierung auf Fehlerschutzmechanismen und Kompression der Essenzsignale.

Die zweite Komponente zur Bemessung der Gesamtsystemlatenz entsteht durch die Be- und Verarbeitung von Contentinformationen auf zwischengelagerten Clients (Workstations). In Abhängigkeit der Komplexität des verwendeten Algorithmus und der zur Berechnung verwendeten Hardware kann diese Verzögerung Bruchteile einer Vollbilddauer betragen. Insbesondere die hohe Leistungsfähigkeit von zusätzlichen Prozessoren wie GPUs kann sehr flexibel für die Berechnung von Bildeffekten genutzt werden.

In der Summe ist die Übertragung und Verarbeitung (Aufnahme mit Kamera – Bearbeitung auf Workstation – Anzeige auf Monitor) der Bildinformation innerhalb der Gesamtlatenz von einem Vollbild möglich. Anspruchsvoller ist die Realisierung von Verarbeitungsprozessen, zu denen die Informationen zweier Bildsignale notwendig sind (z.B. Bildübergänge *Cut*, *Mix* und *Wipe*). Aufgrund der nichtisochronen Übertragung müssen Pakete einer der beiden Ströme gepuffert werden. Betrachtungen in dieser Arbeit (vergleiche Abschnitt 6.4.3) zeigen, dass der bei einer Gesamtlatenz von einem Vollbild existierende Toleranzschlauch nicht alle Fälle abdecken kann und die Gesamtlatenz auf zwei Vollbilder angehoben werden muss.

Eine Kernaussage dieser Dissertation ist also: Wird eine Gesamtlatenz zwischen Kameraaufnahme und Monitoranzeige von zwei Vollbildern im Rahmen eines Produktionsnetzwerkes toleriert, können Standard-IT-Komponenten zu einer offenen und erweiterungsfähigen Netzwerkarchitektur kombiniert werden.

Die vorgestellte Netzwerkarchitektur bildet die Grundlage für die Verarbeitung von Essenz- und insbesondere Metadaten im echtzeitkritischen Bereich der Studioliveproduktion. Daneben kann diese Infrastruktur auch für die Übertragung von Steuerdaten genutzt werden. Diese Dissertation identifiziert Anwendungsszenarien für Metadaten in diesem Bereich und propagiert die Anwendung des Material Exchange Formates (MXF) zur Synchronisation von Meta- und Essenzdaten. In den Bearbeitungsknoten eines Produktionsnetzwerkes liegen so insbesondere Metadaten synchron zu Essenzdaten vor. Die Anwendung des Dateiformates MXF wird ermöglicht durch die im MXF-Standard vorgesehene *Low-Delay*-Applikation von MXF (MXF-Streaming) und dem in dieser Arbeit vorgeschlagenen Konzept des *File-Streaming*.

Folgende Punkte fassen die Ergebnisse der vorliegenden Dissertation zusammen:

- Identifikation von Anforderungen an ein Netzwerk zur Datenübertragung im Studio
- Konzept einer auf Standardprotokollen und Standard-IT-Komponenten basierenden Infrastruktur für die Datenübertragung im Studio
- Konzept des *File-Streaming*, das die Anwendung eines Dateiformates im echtzeitkritischen Bereich (MXF-Streaming) erlaubt
- Konzept zum störungsfreien Umschalten in einem Ethernet-Netzwerk
- Latenzmodell für das Gesamtsystem als Grundlage systematischer Optimierung
- Evaluation von Anwendungsszenarien für Metadaten im Produktionsprozess
- funktionstüchtiger Prototyp zur Verifikation zentraler Aussagen der Arbeit

Im Vergleich zur traditionellen Architektur eines Fernsehproduktionsstudios (vergleiche Abschnitt 2.1) bietet die in dieser Arbeit vorgestellte Netzwerkinfrastruktur den Vorteil der offenen Erweiterbarkeit, der insbesondere für die Übertragung und Verarbeitung von Metadaten nutzbar ist. Dabei kommen Standard-IT-Komponenten zum Einsatz, die in Verbindung mit ihrer steigenden Leistungsfähigkeit langfristig zur Kostenreduktion für Anschaffung, Betrieb und Wartung führen können. Tabelle 7.1 stellt wesentliche Unterschiede zusammen.

	traditionelles Studio	**Netzwerkstudio**
Signalübertragung	isochron	synchron
Übertragungslatenz (Kamera – Bildmischer – Monitor)	0 – 2 Frames (in Abhängigkeit der Komplexität des Mischeralgorythmus)	konstant 2 Frames
Signalverteilung	Kreuzschiene	Switch
störungsfreies Umschalten	einfach und jederzeit in Austastlücken möglich	komplex durch adaptive, intelligente Pufferung
flexibler Zusatzdatentransport	z.T. (Austastlücken)	ja (MXF)
Erweiterbarkeit	bedingt gegeben	offen gegeben

Tabelle 7.1: Vergleich traditioneller Ansatz und Netzwerkansatz

Im Gegensatz zu anderen Forschungsarbeiten auf diesem Gebiet (vergleiche z.B. [MT08, TM08]) baut die in dieser Arbeit propagierte Netzwerkinfrastruktur nicht auf einer isochronen Datenübertragung auf. Mit der Vorgabe einer konstanten Gesamtsystemlatenz von zwei Frames (80 ms) werden Latenzen durch Bearbeitung und Übertragung abgedeckt. Verzögerungen in dieser Größenordnung entstehen bei der Berechnung komplexer

Bildeffekte auf Basis digitaler Broadcastmischer auch im konventionellen Studio. Die synchrone Übertragung erlaubt eine schnelle Verarbeitung von Daten auf Basis komplexer Algorithmen, weil Wartezeiten auf den Referenztakt vermieden werden. In spezifischen Anwendungsfällen kann so eine höhere Übertragungsgeschwindigkeit als bei der isochronen Datenübertragung erreicht werden.

Die Anwendung von Standard-IT-Technologie im echtzeitkritischen Bereich der Fernsehstudioproduktion ist im Wesentlichen auf die hohe Leistungssteigerungen von Standardkomponenten wie Prozessoren, Speicher und Netzwerke zurückzuführen. Entsprechend jung ist auch der Forschungsbereich, der sich der Lösung der damit aufgeworfenen Problemstellungen widmet. Eine Überprüfung der im Teil zur Datenverteilung im Studio vorgestellten Konzepte insbesondere des störungsfreien Umschaltvorgangs steht noch aus. Daraufhin bedarf auch die im Studiobereich unabdingbare Langzeitstabilität eines Nachweises anhand einer anwendungsreifen Implementierung.

Ausblick

Die in dieser Arbeit vorgestellte Netzwerkinfrastruktur wurde mit dem Ziel der offenen Erweiterung konzipiert. Diese Infrastruktur unterstützt so eine Skalierung in mehrerlei Hinsicht.

Die Erhöhung der örtlichen und zeitlichen Auflösung von Videoessenzen (z.B. Ultra HDTV) hat eine erhöhte Datenrate und größere Bildwechselfrequenzen zur Folge. Die damit steigenden Anforderungen an die zugrundeliegende Infrastruktur werden durch die Weiterentwicklung von Prozessor- und Netzwerktechnologie abgefangen (100 Gigabit Ethernet bereits im Standardisierungsprozess). Damit verbunden ist die Verwendung alternativer Übertragungsmedien (Glasfaser, Luft).

Durch die zentrale Rolle des IP-Protokolls im vorliegenden Konzept ist die Ausweitung auf WAN-Bereiche möglich (verteilte Produktion). In Anbetracht nicht vorhersehbarer Datenströme über Internets ist dabei die Anwendung von QoS-Mechanismen unumgänglich.

Anhang

A MXF-Standards der SMPTE

Teil	Nr.	Standard
1 (Engineering Guidelines)	EG 41	MXF Engineering Guideline
	EG 42	MXF Descriptive Metadata
2 (File Format)	S377M	File Format Specification
3.x (Operational Patterns)	S378M	OP 1a (Single Item, Single Package)
	S390M	Specialized OP „Atom“ (Simplified Repres. of a Single Item)
	S391M	OP 1b (Single Item, Ganged Packages)
	S392M	OP 2a (Play-List Items, Single Package)
	S393M	OP 2b (Play-List Items, Ganged Packages)
	S407M	OP 3a and 3b
	S408M	OP 1c, 2c and 3c
4.x (DM Plug Ins)	S380M	Descriptive Metadata Scheme-1 (Standard, Dynamic)
5.x (Generic Container)	S379M	MXF Generic Container
	S388M	MXF Generic Container Reverse Play System Element
	S394M	System Scheme 1 for the MXF Generic Container
	S405M	Elements and Individual Data Items for the MXF Generic Container System Scheme 1
	S410M	Generic Stream Partition
5.ax (Mapping Documents)	S381M	MPEG Streams (Dynamic)
	S382M	AES3 and Broadcast Wave Audio
	S383M	DV-DIF Data
	S384M	Uncompressed Pictures
	S385M	SDTI-CP Essence and Metadata
	S386M	Type D-10 Essence Data
	S387M	Type D-11 Essence Data
	S388M	A-law Coded Audio
	S422M	JPEG 2000 Codestreams
	S436	Mappings for VBI Lines and Ancillary Data Packets
	S2019-4	VC-3 Coding Units

Tabelle A.1: SMPTE Standards zum Material Exchange Format (MXF)

B Standards der SMPTE

Nr.	Standard
S292M	Bit-Serial Digital Interface for High-Definition Television Systems (HD-SDI)
S298M	Universal Labels for Unique Identification of Digital Data
S305M	Serial Data Transport Interface (SDTI)
S330M	Unique Material Identifier (UMID)
S335M	Metadata Dictionary Structure
S336M	Data Encoding Protocol Using Key-Length-Value
S372M	Dual Link 292M Interface for 1920 x 1080 Picture Raster
S395M	Metadata Groups Registry Structure
S400M	SMPTE Labels Structure
S424M	3 Gb/s Signal/Data Serial Interface
S429-4	D-Cinema Packaging – MXF JPEG 2000 Application
S429-6	D-Cinema Packaging – MXF Track File Essence Encryption
S434	XML Encoding for Metadata and File Structure Information
S2021	Broadcast Exchange Format (BXF)
RP210.10	Recommended Practice: Metadata Dictionary Registry of Metadata Element Descriptions
RP224.9	Recommended Practice: SMPTE Labels Register

Tabelle B.1: weitere SMPTE Standards

C Verwendete Hardware

Rechner: (jeweils Windows XP SP2)

„Funok“:

- Dual Xeon 3,2GHz, 1GB RAM, Supermicro X6DAT-G
- GeForce 7960GT Dual DVI PCIe x16
- Intel Pro/1000MT PCI-X 66MHz
- SDI-I/O: DVS Centaurus II PCIe

„Zoidberg“:

- IBM IntelliStation Z-Pro (3,06GHz, 512MB, 40GB)

„Fry“, „Bender“, „Leela“, „Nibbler“, „Amy“, „Zapp“:

- 2,4GHz P4, 1GB RAM, nur PCI-Bus
- 80GB System (davon 20GB WIN), 230 GB Daten (1/2 Linux, 1/2 NTFS)
- RAID-Controller Promise FastTrak TX2000
- Intel Pro/100VE onboard und Intel Pro/1000MT PCI
- Geforce 4Ti

Switch:

Linksys SLM2008

- Smart Managed Switch (IGMP-Unterstützung)
- 8-Port-Gigabit-Ethernet

Bildmischer: Thomson Kayak

Videoserver: Thomson Grass Valley Profile XP 1104

Literaturverzeichnis

[AAH03] Albrecht, W.; Amatucci, G.; Heimann, R.; Mirgel, H.-G.: *HYBNET, das neue Netz der ARD.* FKT, 57. Jahrgang, Nr. 5/2003, S. 201–210

[Ass04] Assmus, Ulf: *Grundlagen der Netzwerktechnik (6) – ATM-Technik.* FKT, 58. Jahrgang, Nr. 3/2004, S. 113–119

[ATF+07] Akar, G. B.; Tekalp, A.M.; Fehn, C.; Civanlar, M.R.: *Transport Methods in 3DTV - A Survey.* IEEE Transactions on Circuits and Systems for Video Technology 11/17(2007), S. 1622–1630

[ATW02] Apostolopoulos, John; Tan, Wai-tian; Wee, Susie: *Video Streaming: Concepts, Algorithms, and Systems.* Whitepaper HP Laboratories Palo Alto, Im Web: `http://www.hpl.hp.com/techreports/2002/HPL-2002-260.pdf`, letzter Abruf: 1.11.2008

[Ban04] Bancroft, Dave: *The Digital Picture Exchange File Format.* In: Gilmer, Brad et. al : File Interchange Handbook for images, audio and metadata; Amsterdam, Boston, Heidelberg u.a.; Focal Press, 2004, S. 61–99

[BBC07] BBC Research: Dirac Pro. April 2007, Im Web: `http://www.bbc.co.uk/rd/projects/dirac/diracpro.shtml`, letzter Abruf: 1.11.2008

[BC05] Body, M.; Cousin,B.: *Efficient media asset transfer in a unified framework managing broadcasting systems.* International Conference on Distributed Frameworks for Multimedia Applications, DFMA 2005, S. 121–127

[BEF+00] Baugé, T.; Egan, R.; Flegkas, P. et al: *QoS Provisioning in an IP-based Studio Production Network.* Proceedings of PGNET 2000, Im Web: `http://www.ee.surrey.ac.uk/Personal/G.Pavlou/Publications/Conference-papers/Bauge-00.pdf`, Abruf: 29.10.2008

[BMW05] Bruns, Kai; Meyer-Wegener, Klaus: *Medieninformatik.* In: Taschenbuch der Medieninformatik; München [u.a.]: Fachbuchverlag Leipzig im Carl Hanser-Verlag, 2005, S. 17–27

[Bri99] Brightwell, P.: *A distributed television production environment using MPEG-2 compression and IT infrastructure.* IEE European Workshop on Distributed Imaging (Ref. No. 1999/109), S. 19/1–19/6

[Bro01] Brodde, E.: *Das MOS-Protokoll Version 2.5 – Chance für die Zukunft der Newsroom-Computer?.* FKT, 55. Jahrgang, Nr. 1-2/2001, S. 44–51

[BRR+02] Boutaba, R.; Ren, N. N.; Rasheed, Y.; Leon-Garcia, A.: *Distributed Video Production: Tasks, Architecture and QoS Provisioning.* Multimedia Tools Appl. 1-2/16(2002), S. 99–136.

[BT00] Brightwell, P. J.; Tudor, P. N.: *A Distributed Programme-Making Environment Using IT-Based Technology.* Proceedings International Broadcasting Convention, 2000, S. 540–546, Im Web: `http://www.bbc.co.uk/orbit/publications/ibc2000.pdf`, letzter Abruf: 30.10.2008

[CVR+04] Cordeiro, M.; Viana, P.; Ruela, J. et al: *The ASSET Architecture – Integrating Media Applications and Products through a Unified API.* SMPTE motion imaging journal, 9/113(2004), S. 307–312

[Dav08] Davies, Thomas: *Wavelet-Kompression für die Videoproduktion.* FKT, 62. Jahrgang, Nr. 1-2/2008, S. 40–44

[DB98] Deliyski, A.; Baberkov, I.: *An untraditional approach to the development of untraditional tapeless TV technology.* EBU Technical Review, Sumer 1998, Im Web: `http://www.ebu.ch/en/technical/trev/trev_276-deliysky.pdf`, letzter Abruf: 30.10.2008

[DHL+04] Devlin, B. Heber, H. Lacotte, J. P. Ritter, U. van Rooy, J. Ruppel, W. Viollet, J. P.: *Nuggets and MXF: Making the Networked Studio a Reality.* SMPTE Motion Imaging Journal 7-8/113(2004), S. 243–251

[DW04] Devlin, Bruce; Wilkinson; Jim: *The Material Exchange Format.* In: Gilmer, Brad et. al : File Interchange Handbook for images, audio and metadata; Amsterdam, Boston, Heidelberg u.a.; Focal Press, 2004, S. 123–176

[EBU08] *EBU/SMPTE Task Force on Time Labeling and Synchronization.* Request for Technology, Februar 2008, Im Web: `http://www.smpte.org/standards/tf_home/TFTS_RFT_270208_final.pdf`, letzter Abruf: 04.03.2008

[EBU98] EBU Technical Review: *EBU/SMPTE Task Force for Harmonized Standards for the Exchange of Programme Material as Bitstreams.* Final Report: Analyses and Results, August 1998, Im Web: `http://www.ebu.ch/CMSimages/en/tec_ebu-smpte_taskforce_final_report_tcm6-40406.pdf`, letzter Abruf: 17.08.2007

[EG41] SMPTE Engineering Guideline 41, *Material Exchange Format (MXF).* White Plains, NY, USA : Society of Motion Picture and Television Engineers, 2003

[EJK04] Endemann, Wolfgang; Jostschulte, Klaus; Kays, Rüdiger: *Grundlagen der Netzwerktechnik (4) – Ethernet-Techniken (IEEE 802.3).* FKT, 58. Jahrgang, Nr. 3/2004, S. 100–105

[Fel05] Felser, M.: *Real-Time Ethernet – Industry Prospective.* Proceedings of the IEEE, Bd. 6/93(2005), S. 1118–1129

[Fin92] Finger, Robert A.: *AES3-1992: The Revised Two-Channel Digital Audio Interface.* Audio Engineering Society Journal 3/40(1992), S. 107–116

[Fli05] Flik, Thomas: *Mikroprozessortechnik und Rechnerstrukturen.* Berlin [u.a.]: Springer, 2005, 7. Auflage

[Fur03] Furrer, Frank J.: *Industrieautomation mit Ethernet-TCP/IP und Webtechnologie*. Heidelberg: Hüthig Verlag, 2003, 3. Aufl.

[Gen02] Genzel, U.: *Automationssysteme in der Fernsehproduktion*. FKT, 56. Jahrgang, Nr. 4/2002, S. 186–189

[Hän07] Hänsch, B.: *Breitbandiges Streaming über Datennetze - Audio & Video over IP*. FKT, 61. Jahrgang, Nr. 8-9/2007, S. 443–450

[Hed95] Hedtke, Rolf: *Verteilung digitaler Videodaten im Studio*. FKT, 49. Jahrgang, Nr. 6/1995, S. 366–376

[Hei04] Heitmann, Jürgen: *Future program production: TV and IT - will one replace the other?*. Vortrag, Broadcast Asia 2004, Im Web: `http://www.avmediatec.com/BAsia04.pdf`, letzter Abruf: 15.08.2007

[HH04] Hedtke, Rolf; Hofmann, Hans: *Grundlagen der Netzwerktechnik (2) – Ein Überblick*. FKT, 58. Jahrgang, Nr. 3/2004, S. 85–90

[HK04a] Hedtke, Rolf; Klemmer, Wolfram: *IT und Netzwerke in der Fernsehproduktion*. FKT, 58. Jahrgang, Nr. 3/2004, S. 79–80

[HK04b] Hedtke, Rolf; Knör, Reinhard: *Grundlagen der Netzwerktechnik (2) – SDTI in der Retrospektive*. FKT, 58. Jahrgang, Nr. 3/2004, S. 91–93

[Hof99] Hoffmann, Hans: *Der Weg zum SDTI – Serial Data Transport Interface*. FKT, 53. Jahrgang, Nr. 1-2/1999, S. 38–44

[Hön02] Höntsch, Ingo: *Fileformate für die vernetzte Fernsehproduktion - Die Bedeutung von Dateiformaten aus Sicht der TV-Produktion*. FKT, 56. Jahrgang, Nr.11/2002, S. 631–639

[I601] ITU-R BT.601-5, *Studio Encoding Parameters of Digital Television for Standard 4:3 and wide-screen 16:9 Aspect Ratios*. Recommendation International Telecommunication Union, 1995

[I656] ITU-R BT.656-4, *Interfaces for Digital Component Video Signals in 525-line and 625-line television Systems operating at the 4:2:2 level of Recommendation ITU-R BT.601 (Part A)*. Recommendation International Telecommunication Union, 1998

[JN04] Jasperneite, J.; Neumann, P.: *How to guarantee real-time behavior using Ethernet*. 11th IFAC Symposium on Information Control Problems in Manufacturing (INCOMŽ2004), 5 - 7 April 2004, Salvador-Bahia, Brazil, Im Web: `http://www.jasperneite.net/paper/incom2004.pdf`, letzter Abruf: 03.11.2008

[Kay05] Kayser, Irene: *Managementsysteme im Broadcastumfeld*. FKT, 59. Jahrgang, Nr. 5/2005, S. 209–212

[KG06] Katz, David J.; Gentile, Rick: *Embedded media processing*. Amsterdam [u.a.] : Elsevier [u.a.], 2006

[KK05] Krömker, Heidi ; Klimsa, Paul: *Einführung*. In: Handbuch Medienproduktion - Produktion von Film, Fernsehen, Hörfunk, Print, Internet, Mobilfunk und Musik, Wiesbaden, Verl. für Sozialwiss., 2005, S. 15–35

[Köh99] Köhler, Rolf-Dieter: *Auf dem Weg zu Multimedia-Netzen – VPN, VLAN-Techniken, Datenpriorisierung.* Köln : FOSSIL-Verl., 1999

[Kov06] Kovalick, Al: *Video systems in an IT environment – the essentials of professional networked media.* Amsterdam [u.a.] : Elsevier/Focal Press, 2006

[KR05] Kurose, James F. ; Ross, Keith W.: *Computer networking : a top-down approach featuring the internet.* Boston, Mass. [u.a.]: Pearson/Addison Wesley, 3. Aufl. 2005

[KR08] Kurose, James F. ; Ross, Keith W.: *Computer networking : a top-down approach.* Boston, Mass. [u.a.]: Pearson/Addison Wesley, 4. Aufl. 2008

[Kug06] Kuge, T.: *Development of JPEG 2000 HDTV program production system.* Proceedings of IEEE International Conference on Image Processing 2006, S. 505–508

[KWG+96] Kaul, M.; Wasserschaff, M.; Gibbs, S.; Breiteneder, C.; Steinberg, D.: *Studio on demand by distributed video production over ATM.* International Broadcasting Convention 1996 (Conf. Publ. No. 428), S. 161–166, Im Web: `http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=642883\&isnumber"-=14018`, letzter Abruf: 30.10.2008

[LG05] Lausberg, Tobias; Gierlinger, Friedrich: *Integration von Kontroll- und Monitoringsystemen in das TV-Produktionsumfeld.* FKT, 59. Jahrgang, Nr. 5/2005, S. 213–218

[LNK04] Lugmayr, Artur; Niiranen, Samuli; Kalli, Seppo: *Digital interactive TV and metadata : future broadcast multimedia.* Berlin [u.a.] : Springer, 2004

[Lös06] Löser, Jork: *Low-Latency Hard Real-Time Communication over Switched Ethernet.* Dissertation, Technische Universität Dresden, 2006

[MAV04] Magaña, E.; Aracil, J.; Villadangos, J.: *Packet Video Broadcasting with General-Purpose Operating Systems in an Ethernet.* Multimedia Tools and Applications 1/24(2004), S. 5–28

[Mar01] Marlowe, F.: *High definition television broadcast technology - a NIST/ATP project.* Proceedings of the Second International Workshop Digital and Computational Video, 2001, IEEE Computer Society, Washington, DC; S. 13–20

[Mär03] Märtin, Christian: *Einführung in die Rechnerarchitektur – Prozessoren und Systeme.* München [u.a.] : Fachbuchverl. Leipzig im Carl Hanser Verl., 2003

[MB76] Metcalfe, Robert M.; Boggs, David R.: *Ethernet: distributed packet switching for local computer networks.* Communications of the ACM, 7/19(1976), S. 395–404

[Met03] Metzger, Thomas: *Metadaten im Fernsehproduktionskanal und deren Austausch sowie Einbindung in Applikationen.* Diplomarbeit, Fachhochschule Südwestfalen, Januar 2003

[Möb07] Möbius, Rico: *MXF-Schnittstelle für Teleprompteranwendungen.* Diplomarbeit, TU Ilmenau, 2007

[MT08] Macé, Gaël; Teener, Michael Johas: *Using Ethernet in the HD studio.* Broadcast Engineering, 1. Juni 2008, Im Web: `http://broadcastengineering.com/hdtv/using-ethernet-hd-studio-0601/`, letzter Abruf: 23.01.2009

[Mün08] Münstermann, Tim: *Konzeption und Umsetzung eines GPU-basierten Produktionsmischers.* Diplomarbeit, TU Ilmenau, Dezember 2008

[NJH08] Naegele-Jackson, Susanne; Holleczek, Peter: *Verteilte Interaktive TV Produktion.* In: Schade, Hans-Peter und Röder, Jan: *Live-Studioproduktion 3.0 – IT-basiert in die Zukunft. Technische Universität Ilmenau, 7. Oktober 2008. Tagungsband*, Universitätsverlag TU Ilmenau, 2008, S. 53–62

[NR05] Nowak, Arne; Röder, Jan: *Möglichkeiten des MXF-Formates bei der parallelen Produktion für verschiedene Produktionskanäle mit Virtual Set Systemen.* In: Elektronische Medien (Dortmunder Fernsehseminar, 26.–28. September 2005), Berlin [u.a.], VDE-Verlag, 2005, S. 77–82

[NSR05] Nowak, Arne; Schade, Hans-Peter; Röder, Jan: *Virtual Reality in Television Production.* Computer Simulation in Information and Communication Engineering CSICE'05, Sofia (Bulgarien), 20. - 22.10.2005

[OLG+05] Owens, John D.; Luebke, David; Govindaraju, Naga; Harris, Mark; Krüger, Jens; Lefohn, Aaron E.; Purcell, Timothy J.: *A Survey of General-Purpose Computation on Graphics Hardware.* In: Eurographics 2005, State of the Art Reports, August 2005, S. 21–51

[OS02] Osborne, Eric; Simha, Ajay: *Traffic engineering with MPLS.* Indianapolis, Ind. : Cisco, 2002

[Ote08] Otero Aguilar, Steve: *Konzeption und Realisierung einer netzwerkbasierenden Architektur für die Anwednung im Fernsehproduktionsstudio.* Diplomarbeit, TU Ilmenau, 2008

[Pae01] Paefgen, Norbert: *General Exchange Format (GXF) - Datentransfer zwischen Videoservern und Schnittsystemen.* FKT, 55. Jahrgang, Nr. 12/2001, S. 735–744

[PD07] Peterson, Larry L.; Davie, Bruce S.: *Computernetze : eine systemorientierte Einführung.* dpunkt.verlag: Heidelberg, 4. Auflage, 2007

[Poh00] Pohlmann, Ken C.: *Principles of digital audio.* New York [u.a.] : McGraw-Hill, 2000, 4. Aufl.

[RBK06] Röder, Jan; Brecht, Rike; Kunert, Tibor: *Effective production of television content for mobile devices.* In: Conference Proceedings of the 10th IEEE International Symposium on Consumer Electronics (ISCE). St. Petersburg, Russland, 28. Juni–1. Juli 2006, S. 439–443

[RDR+03] Ruppel, Wolfgang; Dethloff, Carsten; Ritter, Uwe; Heber, Hans: *Netzwerkbasierte Live-Produktion unter Nutzung des MXF-Fileformates*, In: Vorträge des 10. Dortmunder Fernsehseminars vom 29. September bis 1. Oktober 2003 in Dortmund; Berlin [u.a.] : VDE-Verl., 2003

[Rec08] Rech, Jörg: *Ethernet : Technologien und Protokolle für die Computervernetzung*, Heise : Hannover, 2. akt. Auflage, 2008

[RNE06] Röder, J.; Nowak, A.; Erdmann, M.: *Ethernet für Realtime-Anwendungen.* Proceedings 51. Internationales Wissenschaftliches Kolloquium (IWK), 11.–15. September 2006, Ilmenau-

[RO08] Röder, Jan; Otero, Steve: *Eine MXF-Schnittstelle für die Anwendung im Fernsehproduktionsstudio.* In: Tagungsband zum Workshop „Studioproduktion 3.0“, 07. Oktober 2008, Ilmenau, S. 81–92

[Röd06] Röder, Jan: *Das Material Exchange Format im netzwerkbasierten TV-Studio.* FKTG-Jahrestagung, Potsdam, 15.-18.05.2006

[Röd07] Röder, Jan: *Das Material Exchange Format (MXF) im netzwerkbasierten TV-Studio.* FKT, 61. Jahrgang, Berlin : Schiele & Schön, 1-2/2007, S. 23–27

[S292] SMPTE 292M - 1998, *Bit-Serial Digital Interface for High-Definition Television Systems.* White Plains, NY, USA : Society of Motion Picture and Television Engineers, 1998

[S336] SMPTE 336M - 2001, *Television - Data Encoding Protocol Using KLV.* White Plains, NY, USA : Society of Motion Picture and Television Engineers, 2001

[S372] SMPTE 372M - 2002, *Television - Dual Link 292M Interface for 1920 x 1080 Picture Raster.* White Plains, NY, USA : Society of Motion Picture and Television Engineers, 2002

[S390] SMPTE 390M - 2004, *Television – Material Exchange Format (MXF) – Specialized Operational Pattern „Atom“.* White Plains, NY, USA : Society of Motion Picture and Television Engineers, 2004

[S424] SMPTE 424M - 2005, *Television - 3 Gb/s Signal/Data Serial Interface.* White Plains, NY, USA : Society of Motion Picture and Television Engineers, 2005

[SD96] Stone, J. J.; David, M. W. A.: *Studio integrated operations and networking.* IEE Colloquium on Studio Workstations and Networking, 244/1996, S. 7/1-7/6

[Sha05] Sharafeddine, Sanaa: *IP Network Planning for Realtime Services with Statistical Qos Guarantees.* Dissertation, Technische Universität München, 2005

[Sha08] Schade, Hans-Peter: *Vorwort: Live-Studioproduktion 3.0.* In: Schade, Hans-Peter und Röder, Jan: *Live-Studioproduktion 3.0 – IT-basiert in die Zukunft. Technische Universität Ilmenau, 7. Oktober 2008. Tagungsband*, Universitätsverlag TU Ilmenau, 2008, S. 9/10

[Shi04] Schinas, Konstantin: *Bestimmung von Testparametern für MXF-Decoder und Implementierung einer Anwendung zur Erzeugung von MXF-Referenzdateien.* Diplomarbeit, TU Ilmenau, Januar 2004

[Shm05] Schmidt, Ulrich: *Professionelle Videotechnik - analoge und digitale Grundlagen, Filmtechnik, Fernsehtechnik, HDTV, Kameras, Displays, Videorecorder, Produktion und Studiotechnik.* Berlin [u.a.] : Springer, 2005, 4. Aufl.

[Sie05] Siemens, Eduard: *Verteiltes Messen der Dienstgüte und Netzwerk-Performance in IP-Netzen.* Dissertation, Universität Hannover, 2005

[Sie08] Siemens, Eduard: *Unkomprimierte Filmdaten über IP-basierte Netze.* FKT, 62. Jahrgang, Nr. 10/2008, S. 541–547

[Sim04] Simpson, W.D.: *Uncompressed HD and SD video transport over IP networks.* In: The IEE 2-Day Seminar on IT to HD: Visions of Broadcasting in the 21st Century (Ref. No. 2004/10760), 30 Nov.-1 Dec. 2004, S. 75–84

[SKM03] Song, T.; Kaleshi, D.; Munro, A.: *CORBA-Based Stream Control and Management for IP-Based Production Studio Networks.* Lecture Notes in Computer Science: Management of Multimedia Networks and Services, 2839/2003, Springer Berlin, Heidelberg, S. 1-17

[Smi99] Smith, Steven W.: *The Scientist and Engineer's Guide to Digital Signal Processing.* California Technical Publishing: San Diego, 2. Auflage, 1999, Im Web: http://www.dspguide.com/, letzter Abruf: 22.12.2008

[SR08] Schade, Hans-Peter; Röder, Jan (Herausgeber): *Live-Studioproduktion 3.0 – IT-basiert in die Zukunft.* Tagungsband zum Workshop am 7. Oktober 2008 (Gebundene Ausgabe), TU Ilmenau Universitätsverlag, ISBN: 978-3939473329, Im Web: `http://www.db-thueringen.de/servlets/DocumentServlet?id=10930`

[Str07] Strufe, Thorsten: *Ein Peer-to-Peer-basierter Ansatz für die Live-Übertragung multimedialer Datenströme.* Dissertation, TU Ilmenau, 2007

[Tan04] Tanenbaum, Andrew S.: *Computernetzwerke.* Pearson Studium, München [u.a.]: 4. überarb. Aufl., 2004

[Tan05] Tanenbaum, Andrew S.: *Moderne Betriebssysteme.* Pearson Studium, München [u.a.]: 2. überarb. Aufl., 2005

[Tan06] Tanenbaum, Andrew S.: *Computerarchitektur : Strukturen – Konzepte – Grundlagen.* Pearson Studium, München [u.a.]: 5. Aufl., 2006

[TBP+03] Trimintzios, Panos A.; Baugé, Timothy; Pavlou, George; Flegkas, Paris; Egan, Richard: *Quality of service provisioning through traffic engineering with applicability to IP-based production networks.* Computer Commun. 8/26(2003), S. 845–860, Im Web: `http://personal.ee.surrey.ac.uk/Personal/G.Pavlou/Publications/Journal-papers/Trimin-03a.pdf`, letzter Abruf: 29.10.2008

[Tho02] Thomson Broadcast Solutions: *DD5 / DD10 / DD 20 / DD30 – Planning & Installation Manual.* BTS Media Solutions GmbH, 2002, Im Web: `http://www.grassvalley.com/docs/Planning_Guides/switchers/dd10/DD10_Planning_and_Insallation.pdf`, letzter Abruf: 20.12.2008

[TKY+05] Takeuchi, S.; Kaneko, Y.; Yamamoto, M. et al.: *A Program Production System Using ID and File-Data Over IP Networks.* SMPTE Motion Imaging Journal 4/114(2005), S. 132–138

[TM08] Teener, Michael Johas; Macé, Gaël : *Ethernet in the HD studio.* Broadcast Engineering, 1. Mai 2008, Im Web: `http://broadcastengineering.com/hdtv/ethernet-hd-studio/`, letzter Abruf: 23.01.2009

[TS08] Tanenbaum, Andrew S.; Steen, Maarten van: *Verteilte Systeme : Prinzipien und Paradigmen.* Pearson Studium, München [u.a.], 2. aktualisierte Aufl., 2008

[Wai05] Wainwright, Jochen: *Technik Design und Arichtektur von Intercom-Systemen.* FKT, 59. Jahrgang, Nr. 3/2005, S. 90–94

[WD02] Wilkinson, Jim; Devlin, Bruce: *The Material Exchange Format (MXF) and its Application.* SMPTE Motion Imaging Journal, September 2002

[WDW06] Wells, Nick (Hrsg.); Devlin, Bruce; Wilkinson, Jim; Beard, Matt: *The MXF book – Introduction to the Material eXchange Format.* Focal Press, Amsterdam [u.a.], 2006

[WG98] Wells, N.; Gilchrist, N.: *ATLANTIC: Preserving video and audio quality in an MPEG-coded environment.* Im Web: `http://www.bbc.co.uk/atlantic/pdf-files/ibc98ndwnhcg.pdf`, letzter Abruf: 30.10.2008

[WTK+02] Walland, P. W.; Thomas, G.; Koppetz, M.; Cardoso, J.S.; Erseghe, T.; Hericourt, F.: *The Application of Intimate Metadata in Post Production.* Proceedings of Int. Broadcasting Convention (IBC 2002) Im Web: `http://www.bbc.co.uk/rd/pubs/whp/whp-pdf-files/WHP041.pdf`, letzter Abruf: 30.10.2008

[WWH98] Ward, C.; Wine, C.M.; Homan, D.B.: *Command and Control Architecture for a Digital Studio.* International Patent WO/1998/011724, 1998

[Zim06] Zimmermann, Rico: *Transformation des Datenmodells BMF in das implizite Datenmodell von MXF.* Diplomarbeit, TU Ilmenau, IRT München, 2006

Abkürzungsverzeichnis

A/V	Audio und Video
AAF	Advanced Authoring Format
AES	Audio Engineering Society
ASIC	Application Specific Integrated Circuit
ATM	Asynchronous Transfer Mode
BBC	British Broadcasting Corporation
BEC	Backward Error Correction
BER	Bit Error Rate
BMF	Broadcast Metadata exchange Format
CBR	Constant Bit Rate, *auch:* Constraint-Based Routing
CCU	Camera Control Unit
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
CSDI	Compressed Serial Data Interface
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
DM	Decriptive Metadata (MXF-Kontext)
DMA	Direct Memory Access
DPX	Digital Picture Exchange format
DSCC	Digital Studio Command and Control
DSP	Digital Signal Processor
DV-DIF	Digital Video - Digital Interface Format
EBU	European Braodcasting Uniion
FC	FibreChannel
FEC	Forward Error Corection
FDM	Frequency Division Multiplexing
FPGA	Field Programmable Gate Array
FTP	File Transfer Protocol
GPI	General Purpose Interface
GPS	Global Positioning System
GPU	Graphics Processing Unit
GXF	General eXchange Format
HMD	Header MetaData (MXF-Kontext)
HPT	High Performance Timer
HTTP	Hyper-Text Transfer Protocol
HYBNET	HYbrides Breitband NETz

ICMP	Internet Control Message Protocol
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IGMP	Internet Group Management Protocol
IMX	Interoperability Material eXchange (Sony Markenname für D-10)
IP	Internet Protocol
IST	Information Society Technologies
ISO	International Organization for Standardization
IT	InformationsTechnologie (Information Technology)
KLV	Key Length Value
LAN	Local Area Network
LANE	ATM-LAN-Emulation
LCD	Liquid Crystal Display
MAC	Media Access Control
MAZ	Magnet(band)AufZeichnung(sgerät)
MIB	Management Information Base
MOS	Media Object Server (Protokoll)
MPLS	Multi-Protocol Label Switching
MTU	Maximum Transfer Unit
MXF	Material eXchange Format
NLE	Non Linear Editing (System)
NRCS	NewsRoom Computer System
NRZI	Non Return to Zero Inverted
OMFI	Open Media Framework Interchange (Format)
ORB	Object Resource Broker
OSI	Open System Interconnection
OWD	One Way Delay
PAL	Phase Alternating (by) Line
PC	Personal Computer
QAM	QuadraturAmplitudenModulation
QoS	Quality of Service
RFC	Request for Comments (IETF)
RIP	Random Index Pack (MXF-Kontext)
RS	Reed-Solomon(-Kodierung)
RSVP	Resource reSerVation Protocol
RTCP	Real-Time Control Protocol
RTD	Round Trip Delay
RTP	Realtime Transport Protocol
RTT	Round Trip Time
SDDI	Serial Digital Data Interface
SDI	Serial Digital Interface
SDK	Software Development Kit
SDTI	Serial Data Transport Interface

SMIL	Synchronized Multimedia Integration Laguage
SMPTE	The Society of Motion Picture and Television Engineers
SNMP	Simple Network Management Protocol
SNRZI	Scrambled Non Return to Zero Inverted
TCP	Transmission Control Protocol
TDM	Time Division Multiplexing
TDMA	Time Division Multiple Access
TE	Traffic Engineering
TP	Twisted Pair (Kabel)
TRL	Time Related Label
UDP	User Datagram Protocol
UMID	Unique Material IDentifier
URL	Uniform Resource Locator
VDCP	Video Device Communications Protocol
VLAN	Virtual LAN
WAN	Wide Area Network

Abbildungsverzeichnis

Tabellenverzeichnis

Erklärung

Ich versichere, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen direkt oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet.

Weitere Personen waren an der inhaltlich-materiellen Erstellung der vorliegenden Arbeit nicht beteiligt. Insbesondere habe ich hierfür nicht die entgeltliche Hilfe von Vermittlungs- bzw. Beratungsdiensten (Promotionsberater oder anderer Personen) in Anspruch genommen. Niemand hat von mir unmittelbar oder mittelbar geldwerte Leistungen für Arbeiten erhalten, die im Zusammenhang mit dem Inhalt der vorgelegten Dissertation stehen.

Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form einer Prüfungsbehörde vorgelegt.

Ich bin darauf hingewiesen worden, dass die Unrichtigkeit der vorstehenden Erklärung als Täuschungsversuch angesehen wird und den erfolglosen Abbruch des Promotionsverfahrens zu Folge hat.

(Jan Röder)

Ilmenau, 25. Juni 2009

Thesen

1. Die Nutzung eines datagrammvermittelnden Netzwerkes für die Datenverarbeitung im Live-Fernsehproduktionsstudio erfüllt alle wesentlichen bisherigen Anforderungen und bietet darüber hinaus vielfältige Vorteile durch die Nutzung von Metadaten.

2. In relativ statischen lokalen Netzwerken mit bekannter Anzahl von Teilnehmern und definierten Anforderungen der Teilnehmer muss in datagrammvermittelten Netzwerken kein selbstgesteuerter QoS-Mechanismus (IntServ/Diffserv) etabliert werden, um eine zuverlässige und sichere Übertragung von hochratigen Datenströmen zu ermöglichen.

3. Solange Daten in einer IT-basierten Umgebung ohne Synchronisationszwang ausgetauscht werden, kann *File-Streaming* in definierten Fällen kürzere Verzögerungszeiten aufweisen, als traditionelles isochrones *Streaming*. Voraussetzung dafür ist es, dass nicht nur die reine Übertragungszeit betrachtet wird, sondern die Einheit von Übertragung *und* Bearbeitung als einheitliches System wahrgenommen wird.

4. Das Prinzip der sequentiellen Signalbearbeitung im Fernsehstudio bleibt im Livebetrieb auch für neue Architekturen bestehen. Grundlegende Anforderungen sind die Signalverteilung auf viele Empfänger (1 zu n) und das störungsfreie Umschalten zwischen Essenzsignalen (Audio bzw. Video).

5. Langfristig werden signalspezifische Schnittstellen wie AES/EBU und SDI durch universelle IT-basierte Schnittstellen ersetzt.

6. Das Internet Protocol (IP) stellt eine flexible und erprobte Lösung für die Aufgabenstellungen Adressierung und Routing zur Verfügung. Eine Alternative vor dem Hintergrund der Erweiterung des Ansatzes auf ein Internetwork für Studioanwendungen kann aus Sicht des Autors nicht identifiziert werden.

7. Mit zunehmender Leistungsfähigkeit der zugrunde liegenden Hardware werden Berechnungszeiten für Contentverarbeitungsprozesse verkürzt. Sind die Echtzeitanforderungen eines Einsatzszenarios vergleichsweise gering, können auch Betriebssysteme ohne preemptive Scheduling-Mechanismen zu Einsatz kommen. Voraussetzung dafür ist eine geringe Prozessorlast durch konkurrierende Anwendungen bzw. die Zuweisung entsprechender Prioritäten.

8. Die ohnehin hohe Rechenleistung der Hauptprozessoren (CPUs) kann sehr flexibel erweitert werden, indem FPGAs, DSPs und GPUs zur signalspezifischen Bearbeitung über hochratige Schnittstellen an das System angebunden werden.

9. Die durchgängige Repräsentation von Daten in einer genormten Art und Weise erlaubt die maschinelle Nutzung des Datenbestandes ohne Nutzerinteraktion und damit die Automatisierung von Arbeitsabläufen zur Be- und Verarbeitung des Datenbestandes. Dies trifft auch für den echtzeitkritischen Bereich der Live-Studioproduktion zu.

10. Für die Anwendung im Studiobereich unter Live-Bedingungen, wo Datenströme sequentiell durch Be- und Verarbeitungsstationen geleitet werden, ist die Speicherung von Metadaten mit der Essenz zu bevorzugen (dezentraler Ansatz). Erst nach der Speicherung auf einem Datenträger erweist sich die zentrale Metadatenhaltung z.B. in einer Datenbank als sinnvoll.

11. Wird eine Gesamtlatenz zwischen Kameraaufnahme und Monitoranzeige von zwei Vollbildern im Rahmen eines Produktionsnetzwerkes toleriert, können Standard-IT-Komponenten zu einer offenen und erweiterungsfähigen Netzwerkarchitektur nach dem vorliegenden Konzept kombiniert werden.