Authoring objektbasierter AV-Anwendungen

Dissertation

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

vorgelegt der Fakultät für Elektrotechnik und Informationstechnik
der Technischen Universität Ilmenau

von Dipl.-Ing. Uwe Kühhirt
geboren am 20. Mai 1967 in Zella-Mehlis

Gutachter:
1. Prof. Dr.-Ing. Karlheinz Brandenburg
   Technische Universität Ilmenau
2. Prof. Dr.-Ing. José Luis Encarnação
   Technische Universität Darmstadt
3. Prof. Dr.-Ing. Jörn Ostermann
   Leibniz Universität Hannover

Tag der Einreichung: 7. Juni 2005
Tag der wissenschaftlichen Aussprache: 9. April 2008

urn:nbn:de:gbv:ilm1-2008000059

Zusammenfassung

Die vorliegende Dissertation beschäftigt sich mit dem Authoring-Prozess objektbasierter AV-Anwendungen auf Basis des Objekt- und Szenenkonzeptes von MPEG-4. Diese moderne Beschreibungsform vereint die interaktiven Nutzungsmöglichkeiten digitaler Medien mit den Distributionsmöglichkeiten und dem Qualitätsniveau audiovisueller Medien. Mit MPEG-4 liegt ein umfangreicher Multimedia-Standard zur Beschreibung objektbasierter AV-Anwendungen vor. Dieser ist ein wichtiger Schritt in Richtung Konvergenz der elektronischen Medien. Die Umsetzung des Objekt- und Szenenkonzeptes hat tief greifende Auswirkungen auf die gesamte digitale Medienkette. Insbesondere der Produktionsprozess verändert sich dramatisch. Die Schaffung leistungsfähiger Autorensysteme ist daher eine wichtige Voraussetzung für die Verbreitung solcher Anwendungen.

Das Ziel der Arbeit war die Entwicklung und exemplarische Umsetzung von Konzepten und Komponenten für ein Autorensystem mit Unterstützung eines auf mehrere Autoren verteilten Authoring-Prozesses. Die Bearbeitung wurde in drei Aufgabenbereiche unterteilt: Authoring-Formate, Autorenwerkzeuge und Authoring-Server. Authoring-Formate speichern alle anfallenden Informationen zur Beschreibung einer objektbasierten AV-Anwendung. Um den Austausch der Szenenbeschreibungen zu erleichtern, kommt das in MPEG-4 beschriebene *Extensible MPEG-4 Textual Format (XMT)* in den Ausprägungen XMT-Ω und XMT-A zum Einsatz. Darauf aufbauend wurden eigene Authoring-Formate entwickelt, welche an die Anforderungen konkreter Anwendungen hinsichtlich Abstraktionsebene und Funktionsumfang angepasst sind. Autorenwerkzeuge sind die Schnittstellen des Autorensystems zu den Autoren. Im Fokus stehen grafisch-interaktive Werkzeuge zur Unterstützung eines intuitiven Arbeitens während des Authoring-Prozesses. Die Konzeption berücksichtigt die Anbindung an einen Authoring-Server. Der Authoring-Server bildet die technische Grundlage des Autorensystems für die verteilte Erstellung objektbasierter AV-Anwendungen. Er verwaltet alle anfallenden Daten und stellt diese den Autoren unter Berücksichtigung ihrer individuellen Berechtigungen zur Verfügung. Der Authoring-Server bildet die Schnittstelle zwischen den Produzenten der Medienobjekte und den Autoren. Weiterhin ermöglicht er eine Wiederverwendung von Szenen und Szenenelementen über Produktionsgrenzen hinweg. Der Authoring-Server erlaubt es Autoren und auch Medienproduzenten, gemeinsam an der Erstellung einer AV-Anwendung zu arbeiten. Dafür wurde ein flexibles Datenmanagement auf Basis einer XML-Datenbank entworfen.

Die entwickelten Konzepte orientieren sich an den Möglichkeiten von MPEG-4, sind aber auch auf andere multimediale Anwendungen übertragbar, die auf einem Szenengraphen beruhen. Auf dieser Basis können sowohl universell einsetzbare als auch spezialisierte Autorensysteme und Werkzeuge realisiert werden. Mehrere exemplarische Umsetzungen belegen die Funktionsfähigkeit der entwickelten Komponenten.

Abstract

This thesis is focused on the authoring process of object-based AV applications according to the object and scene concept of MPEG-4. The object-based approach embraces the extended interactive opportunities of multimedia applications as well as the distribution opportunities and the quality level of audiovisual media. The object and scene concept is described in the MPEG-4 standard which is an important step toward the convergence of electronic media. Producing of object-based AV applications has profound effects on the whole digital media chain. Especially the production process changes dramatically. The existence of efficient authoring tools and systems is an important requirement for the success of such applications.

The goal of this work was the development and exemplary implementation of concepts and components for an authoring system for object-based AV applications that supports collaborative work of several authors. The development was divided into three fields: authoring formats, authoring servers, and authoring tools. Authoring formats store all of the resulting information for the description of object-based AV applications during the authoring process. They describe a scene on a well adapted abstraction level and enable the exchange of scene data with other authors and systems. In the authoring system XML-based formats on the basis of the Extensible MPEG-4 Textual Format (XMT) are used. Some special authoring formats were developed which fulfill the requirements of concrete applications regarding abstraction level and function range. Components for the processing of these authoring formats were developed. Authoring tools are the interfaces of the authoring system to the authors. Interactive graphic tools for the support of an intuitive work during the authoring process are in the focus. A typical authoring tool contains several editors, which create and edit scene data stored in XML-based authoring formats. Furthermore, scene data can be edited at source code level. In the case of collaborative authoring an authoring tool is connected to an authoring server. An autarkic use is also possible. The authoring server is the technical basis for the collaborative work of several authors in the production of object-based AV applications. It manages all of the resulting data and gives access to authors with respect to their individual privileges. For this, a flexible data management was developed on the basis of a native XML database. It controls the access at the level of nodes within the scene graph and enables the collaboration of many authors in a novel way. The authoring server is also the interface between the producers of media objects and the authors. Furthermore, it enables the reusability of scenes and elements in different projects.

The developed concepts are oriented towards the possibilities of MPEG-4, but they are transferable to other multimedia applications which are based on a scene graph. With these concepts and components authoring systems and tools can be realized, there are both, universally applicable as well as specialized. Several exemplary applications show the operability of the developed components.

Danksagung

Die vorliegende Arbeit entstand im Rahmen meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Institut für Medientechnik der Technischen Universität Ilmenau. Mein Dank gilt den Mitarbeitern und Studierenden des Fachgebietes Elektronische Medientechnik, welche mir die Erstellung der vorliegenden Dissertation ermöglichten. Mein besonderer Dank gilt meinem Doktorvater Herrn Univ.-Prof. Dr.-Ing. Karlheinz Brandenburg und den beiden Gutachtern Herrn Prof. Dr.-Ing. José Luis Encarnação und Herrn Prof. Dr.-Ing. Jörn Ostermann. Weiterhin danke ich meinen ehemaligen Kollegen Herrn Dr.-Ing. Marco Rittermann, Herrn Dipl.-Ing. Christian Weigel und Herrn Dipl.-Inf. Mathias Schwark für die zahllosen Hinweise bei der Erstellung der Dissertation.

Ich widme diese Arbeit meiner Tochter Nicole.

Uwe Kühhirt

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

Die objektbasierte Beschreibung audiovisueller Inhalte eröffnet ein weites Feld neuer multimedialer Anwendungen und Dienste. Eine derart beschriebene Anwendung enthält Medienobjekte unterschiedlicher Art (z. B. rechteckiges oder beliebig geformtes Video, künstliche Welten, Sprache und Gesichter) und unterschiedlichen Ursprungs (natürlich oder synthetisch), deren örtliche und zeitliche Beziehungen durch eine Szenenbeschreibung festgelegt sind. Unter Verwendung geeigneter Encoder werden die Medienobjekte und die Szenenbeschreibung separat codiert und gemeinsam übertragen. Auf der Empfängerseite werden die einzelnen Medienobjekte decodiert und entsprechend der Szenenbeschreibung auf dem Endgerät dargestellt. Diese Verfahren sind im internationalen Multimedia-Standard *MPEG-4 (ISO/IEC 14496)* beschrieben [ISO01a], [PE02].

Anwendungen auf Basis des Objekt- und Szenenkonzeptes ermöglichen zahlreiche neue Leistungsmerkmale, wie z. B. eine erweiterte Skalierbarkeit, vielfältige Interaktionsmöglichkeiten oder eine Personalisierbarkeit der Inhalte. Dem Nutzer können beispielsweise zusätzliche Ansichten auf die Szene angeboten werden oder er kann individuell durch eine dreidimensionale Szene navigieren [DKR02]. Voraussetzung für die Verbreitung solcher Anwendungen sind leistungsfähige Endgeräte beim Nutzer, welche die meist sehr umfangreiche Verarbeitung der erhaltenen Daten bewältigen müssen. Die Leistungsfähigkeit aktueller und künftiger PCs mit 3D-Grafik-Beschleunigung ist hierfür eine gute Ausgangsbasis. Als Zielplattformen kommen aber vor allem auch „Wohnzimmergeräte“ in Betracht, wie z. B. Set-Top-Boxen für das digitale Fernsehen oder entsprechend ausgestattete Spielkonsolen. Auch Anwendungen für mobile Endgeräte sind in Zukunft denkbar.

Eine praktische Nutzung des Objekt- und Szenenkonzeptes für die Realisierung objektbasierter AV-Anwendungen erfordert neue Methoden der Generierung und Produktion von Inhalten. Hierfür kommen sowohl Verfahren aus der Welt der „klassischen“ Fernsehproduktion als auch Verfahren der Multimedia-Produktion zur Anwendung. Gerade die Schnittstelle zwischen diesen beiden Welten stellt neue Anforderungen an die Autoren sowie an die zugrunde liegenden Werkzeuge und Systeme. Neben der Produktion der vielfältigen Arten von Medienobjekten ist vor allem die Erstellung der mitunter sehr komplexen Szenenbeschreibungen von Interesse, welche auch die Interaktionsmöglichkeiten festlegen. Heute verfügbare Autorenwerkzeuge für multimediale Anwendungen sind meist auf eingegrenzte Anwendungsgebiete fokussiert bzw. wenden sich an bestimmte Nutzergruppen. Sie basieren häufig auf proprietären Technologien und Datenformaten und erschweren so die Erweiterbarkeit der Anwendungen sowie die Zusammenarbeit von Autoren. Es sind bisher keine universell einsetzbaren Systeme verfügbar, welche die gesamte Produktionskette für objektbasierte AV-Anwendungen umsetzen. Die medienübergreifenden Möglichkeiten von MPEG-4 werden derzeit nicht annähernd genutzt. Erst mit der Verfügbarkeit leistungsfähiger Autorensysteme werden sich objektbasierte AV-Anwendungen auf Basis des Objekt- und Szenenkonzeptes von MPEG-4 in der Medienwelt durchsetzen können.

1.2 Zielsetzung

Das Ziel der vorliegenden Arbeit ist die Entwicklung und prototypische Umsetzung von Konzepten, Architekturen und Werkzeugen für die Erstellung objektbasierter AV-Anwendungen auf Basis des Objekt- und Szenenkonzeptes von MPEG-4. Zunächst sind audiovisuelle Anwendungen und der Ablauf deren Produktion zu analysieren und existierende Autorenwerkzeuge zu untersuchen. Aus den gewonnenen Erkenntnissen werden Anforderungen an die zu entwickelnden Komponenten eines Autorensystems abgeleitet. Besonderes Augenmerk liegt dabei auf der Unterstützung eines auf mehrere Autoren verteilten Authoring-Prozesses. Dieser erfordert ein geeignetes Datenmanagement für Medienobjekte und Szenenbeschreibungen. Ein weiterer Schwerpunkt ist die Entwicklung von Datenformaten für den Authoring-Prozess. Im Fokus stehen XML-basierte Sprachen für die Beschreibung objektbasierter AV-Anwendungen. Die erarbeiteten Konzepte werden prototypisch umgesetzt.

1.3 Gliederung

Das Kapitel 2 stellt die Grundlagen objektbasierter AV-Anwendungen vor. Zunächst werden multimediale Anwendungen und deren Beschreibung erläutert und eine Abgrenzung *objektbasierter AV-Anwendungen* vorgenommen. Anschließend werden die erweiterten Möglichkeiten und die technischen Grundlagen dieser Beschreibungsform vorgestellt. Ein Schwerpunkt ist hierbei der MPEG-4-Standard, insbesondere der *Part 1 (Systems)*. Abschließend werden Anwendungen und Projekte aus dem Umfeld vorgestellt.

Kapitel 3 widmet sich dem Erstellungsprozess multimedialer Anwendungen. Zunächst werden typische Abläufe untersucht und wichtige Elemente vorgestellt. Anhand mehrerer Beispiele wird die Erstellung objektbasierter AV-Anwendungen erläutert. Anschließend werden beispielhaft einige Autorensysteme und Forschungsprojekte sowie die wichtigsten Grundlagen des verteilten Authorings vorgestellt.

In Kapitel 4 wird ein modular aufgebautes Autorensystem für die Erstellung objektbasierter AV-Anwendungen auf Basis von MPEG-4 entwickelt. Zunächst werden die Anforderungen an ein künftiges Autorensystem formuliert und ein Authoring-Konzept entworfen. Darauf aufbauend werden Architekturen für ein Autorensystem und dessen Komponenten vorgeschlagen. Es werden Konzepte für XML-basierte Authoring-Formate und deren Verarbeitung sowie für modular aufgebaute Autorenwerkzeuge entwickelt.

Kapitel 5 befasst sich mit der Entwicklung eines universell einsetzbaren Authoring-Servers. Nach der Formulierung von Anforderungen werden verschiedene Ansätze für die Verwaltung strukturierter Daten untersucht. Es wird ein Datenmanagementsystem auf Basis einer XML-Datenbank und ein Konzept für den kontrollierten Zugriff auf Elemente der Szenengraphen entwickelt. Die erforderlichen Komponenten werden zu einer Architektur zusammengefügt.

In Kapitel 6 werden mehrere Projekte vorgestellt, welche die zuvor erarbeiteten Konzepte nutzen. Die Umsetzungen erfolgten am Institut für Medientechnik (IMT) im Rahmen des Forschungsprojektes IAVAS und am Fraunhofer-Institut für Digitale Medientechnologie (IDMT) in Ilmenau. Kapitel 7 fasst die Ergebnisse der Arbeit kompakt zusammen und gibt schließlich einen Ausblick auf mögliche künftige Arbeiten auf dem Gebiet.

2 Objektbasierte AV-Anwendungen

Dieses einleitende Kapitel befasst sich mit den technischen Grundlagen objektbasierter AV-Anwendungen. Ausgehend von herkömmlichen Multimedia-Anwendungen werden die neuen Möglichkeiten dieser Beschreibungsform dargelegt. Es wird eine kompakte Einführung in den Multimedia-Standard MPEG-4 gegeben, insbesondere in den *Part 1 (Systems)* [ISO01a], [ISO02b]. Abschließend werden einige Anwendungen und Projekte vorgestellt.

2.1 Objektbasierter Ansatz

2.1.1 Multimedia-Anwendungen

Multimediale Anwendungen sind aus unserem täglichen Leben nicht mehr wegzudenken. Bereits in den achtziger Jahren des letzten Jahrhunderts wurde am *Massachusetts Institute of Technology (MIT) Media Lab* in Boston an der multimedialen Aufbereitung von Inhalten gearbeitet. Einige dieser Visionen wurden in dem 1987 erschienenen Werk *„The Media Lab: Inventing the Future at MIT“* von Stewart Brand vorgestellt und machten damit das MIT in Cambridge berühmt [Bra88]. Ursprünglich verstand man unter *Multimedia* Computer-Anwendungen, in denen Audio- und Videoelemente integriert sind, um Inhalte anschaulicher oder unterhaltsamer zu präsentieren. Später kamen Interaktionsmöglichkeiten hinzu, die den Menschen vom passiven Betrachter zum aktiven Benutzer einer Anwendung machten. Es gibt bis heute keine eindeutige und wirklich schlüssige Definition des Begriffes *Multimedia*. Gemeinhin versteht man darunter die Aufbereitung von Inhalten unter Nutzung verschiedener digitaler Medien. Typische Elemente sind hierbei Text, Grafik, Animation, Video, Audio und Interaktion. Oftmals wird eine Anwendung erst dann als multimedial bezeichnet, wenn mindestens ein Medium zeitabhängig ist, also z. B. Video, Audio oder Animation. In [Ste00] findet sich folgende Definition:

> „Ein Multimediasystem ist durch die rechnergestützte, integrierte Erzeugung, Manipulation, Darstellung, Speicherung und Kommunikation von unabhängigen Informationen gekennzeichnet, die in mindestens einem kontinuierlichen (zeitabhängigen) und einem diskreten (zeitunabhängigen) Medium kodiert sind.“

Eine der ersten praktischen Multimedia-Anwendungen war die *Compact Disc Interactive CD-i*[1], die von Philips und Sony Mitte der achtziger Jahre entwickelt wurde. Das Ziel war die Schaffung einer nutzerfreundlichen Plattform mit exzellenter Audio- und Videowiedergabe als Alternative zu den verbreiteten *Home-Computern*. Erste Geräte kamen Anfang der neunziger Jahre auf den Markt. Durch die Verwendung von MPEG-1 für die Audio- und Videocodierung erreichte man bereits eine beachtliche Wiedergabequalität. In den folgenden Jahren wurde die CD-i durch die *DVD* verdrängt und spielt inzwischen kaum noch eine Rolle.

[1] http://www.icdia.org/

Heute haben multimediale Anwendungen in den verschiedensten Bereichen Bedeutung erlangt. Einige Beispiele hierfür sind:

- elektronische Bücher und Zeitschriften
- digitale Nachschlagewerke
- multimediale Präsentationen
- Virtual Reality
- Computerspiele
- Kiosk-Systeme (POI und POS)

Die Erstellung solcher Anwendungen erfolgt meist unter Nutzung proprietärer Autorenwerkzeuge bzw. spezieller Entwicklungssysteme. Die Verteilung ist über Netzwerke (z. B. Internet) oder Datenträger (z. B. CD-ROM oder DVD) möglich. Multimedia-Anwendungen werden häufig auch als „digitale Medien“ oder „interaktive Medien“ bezeichnet.

2.1.2 AV-Anwendungen

Auch im Bereich der audiovisuellen Medien (AV-Medien, z. B. Fernsehen und Video) hielt die digitale Technik Einzug. Dennoch entwickelte sich dieser Bereich lange Zeit parallel und weitgehend unabhängig vom Multimedia-Bereich.

Eine klassische AV-Anwendung besteht aus visuellen und auditiven Elementen, die zu einem Videosignal und einem Audiosignal (auch Mehrkanal) zusammengefügt und übertragen werden. Hierbei wird keine Rücksicht auf den Ursprung der Elemente genommen, d. h. alle visuellen Elemente durchlaufen denselben Video-Encoder und alle auditiven Elemente denselben Audio-Encoder. Das Ergebnis ist in der Regel ein zweidimensionales, rechteckiges Video und ein dazu synchrones Audiosignal. Die Verteilung erfolgt über Rundfunk oder Datenträger. Diese in Abbildung 2.1 dargestellte Art der Beschreibung

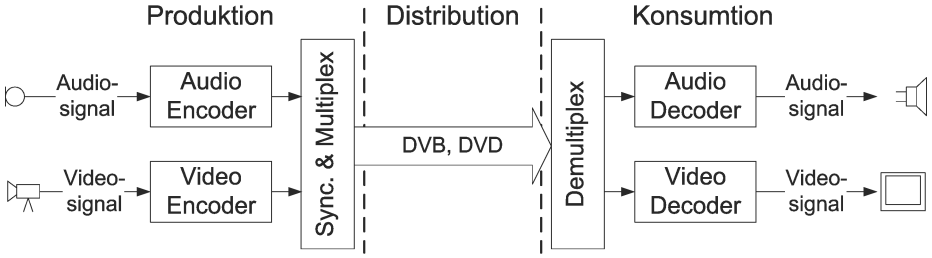

Abbildung 2.1: Klassische AV-Anwendung

nutzt die erweiterten Möglichkeiten moderner Produktionsmethoden nur unzureichend aus. So werden beispielsweise viele aktuelle Magazinsendungen unter Nutzung der virtuellen Studiotechnik produziert, bei der alle Informationen bereits als separate Objekte vorliegen. Dennoch werden diese Objekte vor der Ausstrahlung zu einem zweidimensionalen Video untrennbar zusammengefügt, welches dann über die verschiedenen Distributionswege verteilt wird. Die im Prinzip der objektbasierten Produktion liegenden Möglichkeiten (z. B. Personalisierungs- und Interaktionsmöglichkeiten) bleiben weitestgehend ungenutzt.

Eine objektbasierte AV-Anwendung besteht hingegen aus audiovisuellen Objekten, die getrennt voneinander produziert, bearbeitet, encodiert und übertragen werden. Dabei kommen für jede Objektform speziell angepasste Werkzeuge und Encoder zum Einsatz. Eine Szenenbeschreibung fügt die einzelnen Medienobjekte räumlich (*spatial*) und zeitlich (*temporal*) zu einer objektbasierten AV-Anwendung zusammen. Auf der Empfängerseite werden die einzelnen Objekte decodiert und entsprechend der Szenenbeschreibung dargestellt. Die Szene kann an allen Stellen dieser Kette durch Interaktionen beeinflusst werden. Abbildung 2.2 zeigt die digitale Medienkette für eine objektbasierte AV-Anwendung mit Interaktionsmöglichkeiten.

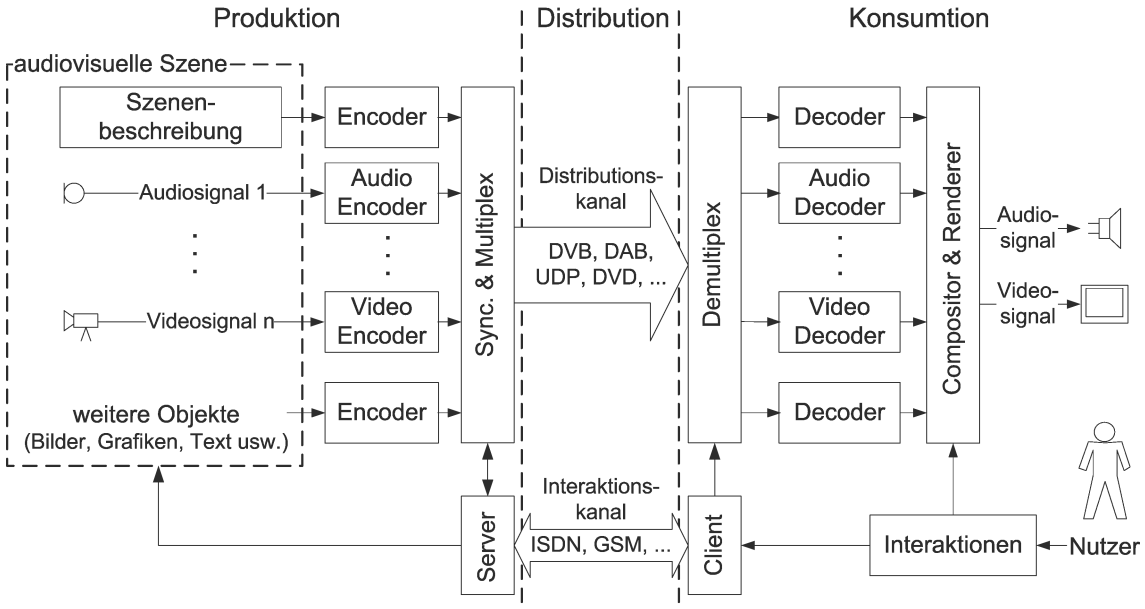

Abbildung 2.2: Objektbasierte AV-Anwendung

Objektbasierte AV-Anwendungen kombinieren die erweiterten Nutzungsmöglichkeiten von Multimedia-Anwendungen (z. B. Interaktivität und Skalierbarkeit) mit den Distributionsmöglichkeiten digitaler AV-Anwendungen auf einem definierbaren Qualitätsniveau. Eine andere treffende Bezeichnung für solche Anwendungen ist „interaktive audiovisuelle Medien“. Die wichtigsten Eigenschaften objektbasierter AV-Anwendungen sind:

- Separat codierte audiovisuelle Objekte sind in einer Szene angeordnet.
- Die Objekte können natürlichen oder synthetischen Ursprunges sein.
- Die Szene kann zweidimensional oder dreidimensional sein.
- Erscheinung und Ablauf der Anwendung sind durch Interaktionen beeinflussbar.
- Alle digitalen Distributionsformen sind prinzipiell nutzbar.
- Genaue Kenntnis des Endgerätes ist für die Beschreibung nicht notwendig.

Die digitale Technik ermöglicht die Nutzung aller Arten von Informationen auf allen Verarbeitungs-, Übertragungs- und Wiedergabesystemen. Die klassische Trennung zwischen verschiedenen Distributionswegen und Endgeräten wird weitgehend aufgehoben. Für dieses Zusammengehen der elektronischen Medien wurde der Begriff *Medienkonvergenz* geprägt.

2.1.3 Interaktionen

Eine herausragende Eigenschaft objektbasierter AV-Anwendungen ist die Möglichkeit, das Erscheinen und das Verhalten der Inhalte durch Interaktionen in vielfältiger Weise beeinflussen zu können. Praktisch alle Elemente einer Anwendung können interaktiv durch den Nutzer steuerbar sein. Zunächst eine Definition des Begriffes *Interaktion* nach DUDEN Deutsches Universalwörterbuch:

> „Interaktion, die (bes. Psych., Soziol.): aufeinander bezogenes Handeln zweier oder mehrerer Personen, Wechselbeziehung zwischen Handlungspartnern: sprachliche Kommunikation ist die wichtigste Form menschlicher Interaktion“

Im Sinne multimedialer Anwendungen versteht man unter Interaktion die aktive Einflussnahme auf eine Anwendung durch den Nutzer. Im Allgemeinen werden diese Interaktionsmöglichkeiten bereits während der Erstellung einer Anwendung durch den Autor festgelegt. Der Nutzer bestimmt dann zur Laufzeit, wann und in welcher Ausprägung er die angebotenen Interaktionsmöglichkeiten nutzen möchte. Das Spektrum denkbarer Interaktionen reicht von der einfachen Selektion von Elementen bis hin zu komplexen Eingriffen in die Szene. Einige Beispiele für Interaktionen sind [ISO02b]:

- Änderung des Betrachtungs- bzw. Hörpunktes der Szene,
  z. B. Navigation durch die Szene
- Aktivierung und Deaktivierung von Objekten,
  z. B. Auswahl eines Sprechers bzw. einer Sprache
- Verschiebung von Objekten an eine andere Position
- Änderung der Eigenschaften von Objekten,
  z. B. Farbe oder Transparenz eines Objektes
- Triggern von Ereignissen,
  z. B. Starten eines Videoelementes durch Anklicken eines Objektes

Diese Arten von Interaktionen werden *Objektinteraktionen* genannt. Interaktionsaufgaben werden in [Bol98] unterteilt in *Basisinteraktionsaufgaben* (Auswahl, Bestätigung, Texteingabe, Positionierung) und *zusammengesetzte Interaktionsaufgaben* (Dialogboxen, Manipulation, Konstruktion). Diese Interaktionsaufgaben lassen sich mit verschiedenen Mitteln lösen. In [Shn02] werden fünf grundlegende Interaktionsstile unterschieden:

> direkte Manipulation, Menüauswahl, Eingabefeld, Befehlssprache und natürliche Sprache.

Für Bildschirm-basierte Anwendungen ist die direkte Manipulation der vorherrschende Interaktionsstil. Hierbei werden Aktionen durch direkte Zeigehandlungen auf grafische Darstellungen von Objekten ausgeführt [Pre99]. Für die Umsetzung von Interaktionsmöglichkeiten wird ein Eingabegerät benötigt. Dies kann beispielsweise eine Tastatur, eine Maus, eine Fernbedienung oder auch ein Mikrofon sein. Die Gestaltung von Interaktionsmöglichkeiten ist sowohl aus inhaltlicher als auch aus technischer Sicht eine große Herausforderung bei der Erstellung interaktiver Anwendungen. Folgende Arten von Interaktionen sind zu unterscheiden:

- *lokale Interaktionen:* Alle notwendigen Informationen werden über den Distributionskanal bereitgestellt und sind somit lokal verfügbar. Die Interaktion erfolgt vollständig innerhalb der Anwendung und erfordert keinen Informationsfluss vom Nutzer zum Anbieter. Beispiele: Videotext, EPG, einfache Spiele

- *Kommunikations-basierte Interaktionen:* Die Interaktion erfordert einen Informationsfluss vom Nutzer zum Anbieter – einen so genannten *Interaktionskanal*. Es lassen sich zwei Arten unterscheiden:
  - *unidirektionale Kommunikation:* Es erfolgt ein Informationsfluss vom Nutzer zum Anbieter ohne direkte Auswirkung auf den umgekehrten Informationsfluss. Beispiel: Televoting
  - *bidirektionale Kommunikation:* Es erfolgt ein Informationsfluss vom Nutzer zum Anbieter mit direkter Auswirkung auf den umgekehrten Informationsfluss. Informationen werden bei Bedarf vom Anbieter zum Nutzer übertragen. Beispiele: Telebanking, Teleshopping, Online-Spiele

Während der Nutzung interaktiver Anwendungen findet gewöhnlich ein bidirektionaler, meist stark unsymmetrischer Datenaustausch zwischen Anbieter und Nutzer statt. Der Distributionskanal ermöglicht die unidirektionale Übertragung von Inhalten vom Anbieter zum Nutzer mit hoher Datenrate. Der Interaktionskanal stellt eine Verbindung zum Anbieter der Inhalte her und ermöglicht somit einen bidirektionalen Datenaustausch. Beide Kanäle können dieselbe Technik nutzen (z. B. DSL) oder auch völlig unabhängig voneinander sein (z. B. Distributionskanal: DVB, Interaktionskanal: Telefonnetz). Die zu übertragenden Datenmengen unterscheiden sich meist deutlich. Für den Interaktionskanal ist daher oft eine Modem- oder ISDN-Verbindung zum Anbieter ausreichend. Die Forderung nach einem Interaktionskanal schränkt die Auswahl der nutzbaren Endgeräte mitunter ein.

2.1.4 Distribution

Die digitale Technik erlaubt vielfältige Formen der Verteilung multimedialer Anwendungen. Die Übertragung bzw. Speicherung ist prinzipiell mit allen digitalen Systemen möglich, die bestimmte Mindestanforderungen an Bandbreite bzw. Speicherplatz erfüllen. Vor allem das Internet und der digitale Rundfunk sind scheinbar ideal geeignet für die Verbreitung multimedialer Anwendungen. Als Distributionswege für objektbasierte AV-Anwendungen kommen beispielsweise in Frage:

- Datenträger: CD, DVD usw.
- Netzwerke:
  - Internet
  - Mobilfunk: GSM, GPRS, UMTS usw.
- digitaler Rundfunk: DVB und DAB

Distribution mit Hilfe von Datenträgern

Die Verteilung multimedialer Inhalte mit Hilfe von Datenträgern stellt die älteste Distributionsform dar. Mit der Verbreitung von optischen Speichermedien stand eine hinreichend

große Speicherkapazität zur Verfügung. Grenzen setzte zunächst die Leistungsfähigkeit der Endgeräte, z. B. bei der Videodecodierung. Mit der DVD ist nun die nächste Generation von Speichermedien verfügbar. Sie ist geeignet für die Verteilung größerer Datenmengen, wodurch beispielsweise auch Inhalte in HDTV-Qualität möglich werden. Weitere, noch leistungsfähigere Datenträger werden in Kürze eingeführt (z. B. High Density DVD [HD-DVD] und Blue-ray Disc).

Distribution über Netzwerke

Mit der Verbreitung des Internets gewann auch die Verteilung multimedialer Anwendungen über Netzwerke schnell an Bedeutung. Zunächst wurden Multimedia-Anwendungen dateibasiert übertragen, d. h. eine Anwendung muss vor der Nutzung vollständig heruntergeladen werden. Für die Verteilung objektbasierter AV-Anwendungen erscheint eine kontinuierliche Übertragung der Daten sinnvoller – so genannte Streaming-Verfahren, z. B. auf Basis von *RTP/RTSP*. Das Ziel ist die Übertragung und Nutzung der Inhalte ohne Zeitverlust, wodurch AV-Anwendungen bereits während des Ladevorganges nutzbar sind. Es werden nur die gerade benötigten Daten übertragen und zur Laufzeit decodiert. Im Falle interaktiver Anwendungen können einzelne Elemente durch den Nutzer gezielt angefordert werden. Mit den heute verfügbaren Datenraten lassen sich auch komplexe Inhalte adäquat übertragen. Neben den drahtgebundenen kommen zunehmend auch drahtlose Netzwerke zum Einsatz, z. B. auf Basis von Mobilfunk (GSM, GPRS und in Zukunft vor allem UMTS). Aus Sicht einer objektbasierten AV-Anwendung ist die konkrete Realisierung der Datenübertragung unerheblich, wenn bestimmte Mindestanforderungen an Datenrate und Verzögerungszeit eingehalten werden. Auch auf dem Gebiet der transparenten Distribution über Netzwerke erfolgt eine Konvergenz der Übertragungsverfahren. Unter dem Namen *Internet Streaming Media Alliance (ISMA)* haben sich Firmen und Institutionen zusammengeschlossen, um die Übertragung von Multimedia-Anwendungen über Netzwerke voranzutreiben und zu standardisieren [ISM].

Distribution über Rundfunk

Die verfügbaren Infrastrukturen des digitalen Rundfunks erlauben die Distribution umfangreicher Inhalte. In Europa sind zwei digitale Rundfunknetze verbreitet: das *Digital Audio Broadcasting (DAB)* und das *Digital Video Broadcasting (DVB)*. Letzteres nutzt für die Übertragung der Programme einen MPEG-2-Transportstrom. Dieser kann neben den eigentlichen Fernsehprogrammen einschließlich zugehöriger Zusatzinformationen auch weitere digitale Inhalte transportieren. Folgende Möglichkeiten der Datenübertragung unter Nutzung des MPEG-2-Transportstromes stehen zur Verfügung:

- *Data Piping:* fragmentierte, nicht-synchrone Übertragung
- *Data Streaming:* Möglichkeit der synchronen Übertragung
- *Multiprotocol Encapsulation:* Übertragung verschiedener Protokolle, vorzugsweise IP
- *Data & Object Carousel:* zyklische Übertragung von Daten

Tabelle 2.1 stellt einige dieser Verfahren gegenüber.

	Stream Encapsulation	IP Encapsulation	Data- und Object-Carousel
Transport	PES	DSM-CC	DSM-CC
Format	Streaming	Streaming	komplette Dateien
Weiteres	• FlexMux-Tool optional • gut geeignet für Broadcast	• Broadcast, Multicast, Unicast • Adressierung über MAC-Adresse • benötigt Rückkanal • Interaktion mit Server möglich	• benötigt lokale Speichermöglichkeit • vollständiger Download muss abgewartet werden
Anwendung	Broadcast von Streaming-Inhalten	interaktive AV-Anwendungen	Zusatzapplikationen und Szenensequenzen

Tabelle 2.1: Übertragungsverfahren unter Nutzung des MPEG-2-Transportstromes

Auch zahlreiche wissenschaftliche Projekte beschäftigten sich mit der Übertragung interaktiver Anwendungen über Rundfunk. Hier einige Beispiele bereits abgeschlossener Projekte:

- Sambits – System for Advanced Multimedia Broadcast and IT Services (siehe Unterabschnitt 2.5.2); IRT, EBU, Philips, Siemens, BR, ...
- NexTV – New media consumption in Extended interactive TeleVision environment [Nex]; Philips, T-Nova, GMD, Sony, SUN, France Telecom, ...
- Song – portalS Of Next Generation [Son]; blaxxun Interactive, T-Nova, EPFL, Siemens, ...

Im Verlauf dieser Projekte wurde gezeigt, dass die verfügbaren digitalen Rundfunknetze für die Verteilung interaktiver Anwendungen nutzbar sind. Sowohl auf der Seite der Anbieter als auch auf der Seite der Nutzer sind hierfür neue Systeme und Geräte erforderlich.

2.1.5 Konsumtion

Voraussetzung für die Konsumtion objektbasierter AV-Anwendungen sind leistungsfähige Endgeräte, welche die teilweise sehr komplexen audiovisuellen Inhalte decodieren und adäquat darstellen können. Die Auswahl an möglichen Endgeräten wird immer größer. Während für neuartige Anwendungen zunächst der PC als flexible, frei programmierbare Plattform eine gute Ausgangsbasis darstellt, sind viele weitere Arten von Endgeräten denkbar. Mögliche Plattformen für die Nutzung objektbasierter AV-Anwendungen sind beispielsweise Personalcomputer, Fernsehgeräte (mit STB), Spielkonsolen (z. B. *Playstation* oder *Xbox*) und mobile Endgeräte (z. B. Mobiltelefone und PDA). Tabelle 2.2 zeigt mögliche Kombinationen von Distributionswegen und Endgeräten. Weiterhin ist der für viele Anwendungen notwendige Interaktionskanal zu berücksichtigen. Die Konsumtion lässt sich nach verschiedenen Kriterien einteilen:

	PC	Fernseher	Spielkonsolen	mobile Endgeräte
Datenträger	×	○	×	○
Rundfunk	○	×	○	○
Internet	×	○	○	○
Mobilfunk				×

Tabelle 2.2: Distributions- und Konsumtionsmöglichkeiten (×=Standard, ○=möglich)

- nach dem Ort der Nutzung: stationär oder mobil
- nach der Art des Distributionskanals: Datenträger, Netzwerk oder Rundfunk
- nach der Art der Interaktion: lokale oder kommunikationsbasierte Interaktion

Eine Unterscheidung zwischen stationärer und mobiler Nutzung verliert auf Grund der raschen Entwicklung drahtloser Netzwerke (z. B. GPRS, DVB-T und WLAN) an Bedeutung. Es sind verschiedene Kombinationen von Distributions- und Interaktionskanal möglich. Eine Enzyklopädie könnte beispielsweise auf DVD vertrieben werden; während der interaktiven Nutzung werden über den Interaktionskanal (z. B. DSL oder ISDN) weitere Inhalte zum Nutzer übertragen. Ein ähnliches Szenario findet sich beim interaktiven Fernsehen; die Distribution erfolgt dabei über Rundfunk (DVB).

2.2 Medienobjekte

Multimediale Anwendungen bestehen aus Informationseinheiten, die räumlich und zeitlich im Zusammenhang stehen. Diese Informationseinheiten kapseln Informationen und werden als *Objekte* betrachtet. In der Literatur werden diese Objekte als *audiovisuelle Objekte (AVO)* oder *Medienobjekte (MO)* bezeichnet. Die Medienobjekte bilden die Knoten eines multimedialen Beziehungsnetzwerkes. Sie werden über Kanten miteinander verbunden, welche die Beziehungen zwischen den Objekten repräsentieren [Bol98].

2.2.1 Einteilung

Die Medienobjekte sind in ihrer Beschreibung prinzipiell unabhängig voneinander. Ein Aspekt objektbasierter AV-Anwendungen ist die Komposition von Medienobjekten unterschiedlichster Art und Herkunft zu einer audiovisuellen Anwendung. Medienobjekte lassen sich nach verschiedenen Kriterien einteilen:

1. nach ihrer Herkunft: natürlich oder synthetisch
2. nach der Art der Perzeption: visuell oder auditiv
3. nach ihrer Dimension: zweidimensional oder dreidimensional
4. nach ihrem zeitlichen Verhalten: zeitunabhängig (diskret) oder zeitabhängig (kontinuierlich)

Tabelle 2.3 gibt einige Beispiele für die genannte Einteilung.

Objekt	Herkunft		Wahrnehmung		Dimension		zeitl. Verhalten	
	natürl.	synth.	visuell	auditiv	2D	3D	diskret	kont.
Text		×	×		×	×	×	
Standbild	×		×		×	×	×	
Video	×		×		×	×		×
Grafik		×	×		×	×	×	
Animation		×	×		×	×		×
Ton	×			×				×
MIDI		×		×				×

Tabelle 2.3: Beispiele für die Einteilung der Medienobjekte

Natürliche Medienobjekte liegen dann vor, wenn sie natürlichen Ursprungs sind, d. h. wenn sie mit Kameras oder Mikrofonen aufgenommen wurden [ISO98]. Natürliche visuelle Objekte liegen gewöhnlich als Pixelbilder vor; natürliche auditive Objekte werden durch digitalisierte Wellenformen repräsentiert. Sie werden entsprechend ihrer Herkunft und ihrer Eigenschaften encodiert. Für die Produktion natürlicher Medienobjekte werden überwiegend Verfahren und Werkzeuge aus dem Bereich der professionellen Studiotechnik eingesetzt. Für multimediale Anwendungen kommen drei Arten von natürlichen Medienobjekten in Frage:

Standbild (Fotos), Bewegtbild (Video) und Ton (Audio).

Auf die Art Standbild soll hier nicht näher eingegangen werden. Unter *Bewegtbild* versteht man im Allgemeinen eine kontinuierliche Folge von Einzelaufnahmen einer meist natürlichen Szene. Erfolgt die Aufnahme, Speicherung, Übertragung und Wiedergabe vollständig auf elektronischem Weg, spricht man von Video. Heute wird Video überwiegend digital verarbeitet. Für multimediale Anwendungen lassen sich folgende Arten von digitalem Video unterscheiden:

- rechteckiges zweidimensionales Video: *herkömmliches Video*
- Panorama-Video
- geformtes zweidimensionales Video: *shaped Video*
- dreidimensionale Videoobjekte: *3DVO*

Herkömmliches Video besteht aus rechteckigem, zweidimensionalem Bewegtbild und ist wichtiger Bestandteil multimedialer Anwendungen. In objektbasierten Anwendungen reicht diese einfachste Form von Video jedoch oft nicht aus. Wenn verschiedene Medienobjekte zu einer audiovisuellen Szene verbunden werden sollen, schränkt die stets rechteckige Form eines solchen Videoobjektes den Gestaltungsspielraum erheblich ein. Erweiterte Möglichkeiten bieten beliebig geformte Videoobjekte (siehe Unterabschnitt 2.2.2) und 3D-Videoobjekte (siehe Unterabschnitt 2.2.3).

Synthetische Objekte liegen dann vor, wenn die Generierung mit Hilfe eines Rechners stattfindet. Typische Beispiele hierfür sind:

Text, zwei- und dreidimensionale Grafik sowie synthetisches Audio.

Im Unterschied zu natürlichen Medienobjekten werden synthetische Medienobjekte parametrisch beschrieben, z. B. durch eine textuelle Objektbeschreibung oder einen Parameter-Strom. Sie sind daher gut bearbeitbar und steuerbar. Dies hat tief greifende Auswirkungen auf die spätere Integration in eine audiovisuelle Anwendung. Beispiele für neuartige synthetische Objekte sind parametrisch beschriebene menschliche Gesichter und Körper.

In objektbasierten AV-Anwendungen können alle bekannten Arten von Medienobjekten vorkommen. Insbesondere die Kombination von natürlichen und synthetischen Objekten eröffnet vielfältige neue Möglichkeiten. Im Folgenden werden beispielhaft einige ausgewählte Formen von Medienobjekten beschrieben, die in objektbasierten AV-Anwendungen eine besondere Bedeutung erlangen und in herkömmlichen Multimedia-Anwendungen bisher kaum zum Einsatz kamen.

2.2.2 Beliebig geformte Videoobjekte

Beliebig geformte Videoobjekte sind unter dem Namen *arbitrarily shaped video* bekannt [OJSC97]. Durch Hinzufügen einer Maske zu einem Video kann eine Information über die Form des Objektes übermittelt werden. Diese Masken werden z. B. mit Hilfe des Chroma-Key-Verfahrens gewonnen. Man unterscheidet zwei Arten (siehe Abbildung 2.3):

- *Binäre Maske:* Der Alpha-Kanal wird mit einer binären Maske belegt. Unterscheidbar sind nur vollständig transparente und vollständig opake[2] Bereiche.
- *Graustufen-Maske:* Der Alpha-Kanal wird mit einer Graustufen-Maske belegt. Es werden Informationen über teiltransparente Bereiche übermittelt. Dies ermöglicht die Reduzierung der durch eine binäre Maske erzeugten Artefakte an den Rändern eines Objektes. Außerdem sind teiltransparente Flächen, z. B. Glas oder Wasser realisierbar.

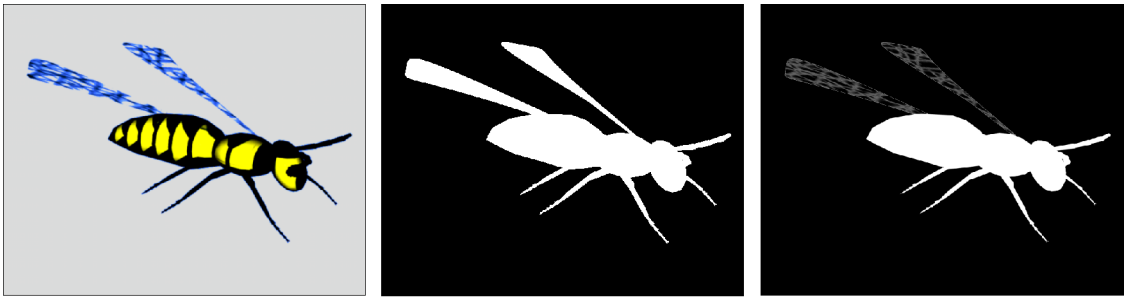

Abbildung 2.3: Geformtes Videoobjekt und zugehörige Masken

Das Video wird zusammen mit der oder den Masken mit entsprechenden Encodern encodiert und später als Medienobjekt in die Szene eingefügt. Auf der Wiedergabeseite wird das Objekt wieder decodiert und kann in einer beliebigen Umgebung platziert werden [Sch02]. Geformtes Video ermöglicht die Integration beweglicher natürlicher Objekte in virtuelle Umgebungen. Diese Objektform wird überwiegend für die Darstellung von Personen eingesetzt. Die Verwendung von geformtem Video in dreidimensionalen Szenen stellt

[2] Opazität: Maß für die Lichtundurchlässigkeit

eine Integration einer zweidimensionalen Fläche in eine dreidimensionale Umgebung dar. Wird dem Nutzer eine freie Navigation im Raum ermöglicht kann es vorkommen, dass der Betrachter von der Seite auf ein solches Videoobjekt schaut, und ein verzerrtes Bild sieht. Eine Möglichkeit dies zu verhindern ist die *Billboardtransformation*. Hierbei wird das Objekt immer automatisch senkrecht in Richtung des Betrachters bzw. der Kamera ausgerichtet [Sch02]. Abbildung 2.4 zeigt die Integration eines beliebig geformten Videoobjektes in eine synthetische Umgebung.

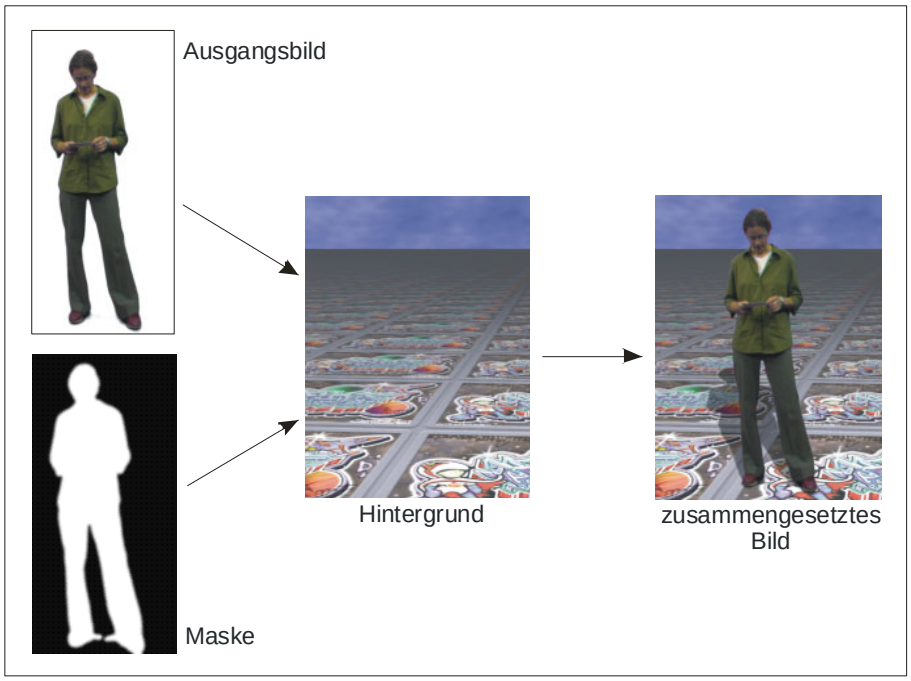

Abbildung 2.4: Natürliches Video in einer synthetischen Umgebung

2.2.3 3D-Videoobjekte

Multimediale Anwendungen können grundsätzlich auch dreidimensional sein. Daher liegt es nahe, auch Videoobjekte dreidimensional zu beschreiben. Der inzwischen häufig verwendete Begriff *3D-Video* wird in vielen verschiedenen Zusammenhängen verwendet. Häufig versteht man darunter lediglich die Erweiterung der visuellen Wahrnehmung um einen räumlichen Eindruck, beispielsweise durch eine *stereoskopische Wiedergabe* [Rit04]. Interessanter für das Authoring ist die Betrachtung der Repräsentation und damit der Nutzungsmöglichkeiten dreidimensionaler Videoobjekte. Innerhalb der MPEG wurde der Bereich 3D-Video wie folgt eingeteilt [ISO03c], [Rit04]:

1. omnidirektionales Video (Panoramaansichten)
2. interaktives stereoskopisches Video
3. Free Viewpoint Video (interactive multiple view video)
4. dreidimensionale Videoobjekte (free viewpoint video objects)

Die letztgenannte Kategorie sind die *3D-Videoobjekte (3DVO)*, die für objektbasierte AV-Anwendungen besonders interessant erscheinen. Eine Eingrenzung des Begriffes *3D-Videoobjekt* wird über seine Eigenschaften vorgenommen:

- 3D-Videoobjekte repräsentieren die visuellen Eigenschaften eines natürlichen (dreidimensionalen), zeitveränderlichen Objektes, z. B. eines Menschen.
- Das 3D-Videoobjekt ermöglicht dem Nutzer eine Ansicht auf das Objekt, die von einem wählbaren Standpunkt zu einem wählbaren Zeitpunkt erfolgen kann. Im Idealfall sind der Betrachterstandpunkt und die Blickrichtung frei wählbar.

Abbildung 2.5 zeigt zwei Ansichten einer virtuellen Szene mit integriertem 3D-Videoobjekt [KR03]. Die Person wird durch ein 3D-Videoobjekt natürlichen Ursprungs repräsentiert. Innerhalb eines limitierten Bereiches erhält der Nutzer während der Navigation eine korrekte Perspektive des Objektes [Rit04].

Abbildung 2.5: Zwei Ansichten auf ein Szenario mit einem 3D-Videoobjekt

2.2.4 Audioobjekte

In klassischen AV-Anwendungen (siehe Abbildung 2.1) werden gewöhnlich alle auditiven Elemente gemeinsam als Mix (z. B. Stereo oder Mehrkanalton) encodiert. Dieser enthält neben den inhaltlichen Informationen auch die jeweilige Raumakustik. Die einzelnen Informationen (z. B. Ort und Ausrichtung einzelner Quellen) lassen sich auf Nutzerseite in der Regel nicht wieder vollständig zurückgewinnen. Für eine optimale Wiedergabe muss bei der Abmischung der Tonspur auch die Art der Wiedergabetechnik beim Nutzer bekannt sein. Üblich sind Tonspuren in Stereo oder Mehrkanalton.

In einer objektbasierten AV-Anwendung können auch die auditiven Elemente, genau wie die visuellen Elemente, als Objekte beschrieben werden (siehe Abbildung 2.2) [SVH99]. Diese werden im Raum platziert und mit akustischen Eigenschaften (z. B. Richtung und Abstrahlcharakteristik) versehen. Der virtuelle Raum muss nun ebenfalls in seinen akustischen Eigenschaften beschrieben werden. Neben natürlichen können auch synthetische Audioobjekte verwendet werden. Die akustische Szene wird beim Nutzer unter Berücksichtigung der Geometrie und der Materialien des virtuellen Raumes für das vorhandene Wiedergabesystem berechnet [RMS01], [DRSD03].

Typische Audioobjekte für objektbasierte AV-Anwendungen sind:

- allgemeine Audiosignale (nicht näher spezifiziert): Musik, Geräusche, Sprache
- Sprache
- synthetisches Audio
- synthetische Sprache

Die objektbasierte Beschreibung ermöglicht die Nutzung separater, angepasster Encoder für die verschiedenen Arten von Audioobjekten und Anwendungen. Beispielsweise lässt sich Sprache mit speziellen Sprach-Encodern wesentlich effizienter encodieren als mit universellen Encodern, die auch Musik und Geräusche verarbeiten können. Weiterhin werden völlig neue Konzepte für die Audiowiedergabe möglich. Ein Beispiel hierfür ist die *Wellenfeldsynthese (WFS)*, auf die im Abschnitt 6.5 noch näher eingegangen wird.

2.3 Szenenbeschreibung

Zentraler Bestandteil einer objektbasierten AV-Anwendung ist die Szenenbeschreibung. Sie stellt die räumlichen und zeitlichen Beziehungen zwischen den Objekten her und beschreibt die Interaktionsmöglichkeiten. Ursprünglich wurden Szenendaten gemeinsam mit Anweisungen für die Verarbeitung (z. B. Rendering) prozedural beschrieben, wodurch oft sehr komplexer und unflexibler Code entstand. Erst durch die konsequente Trennung des Inhaltes einer Szene von Funktionen zu dessen Darstellung sind effektive Szenenbeschreibungen möglich. Im Bereich der Computergrafik hat sich der *Szenengraph* als grundlegende Beschreibungsform durchgesetzt [FvFH96], [Wal02], [WBS01].

2.3.1 Szenengraph

Die Datenstruktur zur Beschreibung zweidimensionaler oder dreidimensionaler Szenen ist für gewöhnlich ein Graph. Ein Graph besteht aus Knoten und Kanten. Jede Kante verbindet zwei Knoten miteinander und repräsentiert die Beziehung der Knoten. In der Computergrafik arbeitet man mit *gerichteten zyklenfreien (azyklischen) Graphen (directed acyclic graph DAG).*

Ein *gerichteter Graph (directed graph)* oder *Digraph* $G = (V, E)$ besteht aus einer endlichen, nicht leeren Menge $V = \{v_1, ..., v_n\}$ von *Knoten (vertices)* und einer Menge $E \subseteq V \times V$ geordneter Paare $e = (u, v)$ von *Kanten (edges)* von G. Jede Kante $e \in E$ hat einen Anfangsknoten u und einen Endknoten v und damit eine Richtung von u nach v. Wenn ein Graph keine Zyklen enthält, ist er *zyklenfrei* oder *azyklisch.*

Ein *Baum* ist ein gerichteter Graph $G = (V, E)$ mit folgenden Eigenschaften: Es existiert genau ein Knoten $v \in V$ (Wurzel des Baumes, *root*) derart, dass für alle Knoten v_i genau ein Weg von v nach $v_i \in V$ existiert. In einer Baum-Struktur darf demzufolge kein Kindknoten mehr als einen Elternknoten besitzen. Abbildung 2.6 zeigt den Aufbau eines gerichteten azyklischen Graphen. Der Szenengraph enthält eine vollständige Beschreibung einer Szene. Die Knoten des Szenengraphen definieren die geometrischen Daten (Objekte und Transformationen), die Attribute (Farbe, Material usw.) und die Kameraparameter.

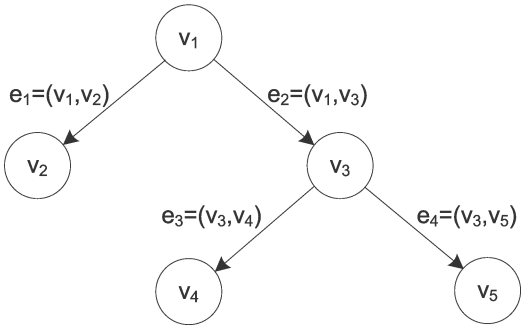

Abbildung 2.6: Gerichteter azyklischer Graph

Die wichtigsten Eigenschaften eines Szenengraphen sind [Wal02]:

- Aus Knoten und Kanten entsteht die Graphenstruktur zur hierarchischen Organisation aller Objekte einer Szene, entsprechend ihrer räumlichen Anordnung.
- Mit Ausnahme des obersten Knotens (*Wurzelknoten*), welcher den Einstiegspunkt in einen Szenengraphen definiert, hat jeder Knoten einer Szene einen Vorgänger (*parent*).
- Knoten, die andere Knoten enthalten, sind Elternknoten (*parent nodes*). Die enthaltenen Knoten sind Kindknoten (*children nodes*) ihrer Eltern.
- Knoten, die keine weiteren Kindknoten enthalten, heißen Blattknoten (*leaf nodes*).
- Knoten, die Kindknoten enthalten, sind Gruppenknoten (*grouping nodes*).
- Zwischen den Knoten treten keine Zyklen auf – man kann den Knoten also nur über seinen parent betreten (*directed acyclic graph DAG*).

Der Szenengraph erlaubt eine einfache und effiziente Beschreibung und Organisation der Elemente einer Szene. Kindknoten erben die Eigenschaften ihrer Eltern. Beliebige Elemente können zu Gruppen zusammengefasst werden. Die Relationen zwischen Szene und Elementen werden durch Transformationsknoten beschrieben. Auch die Erscheinung von Elementen wird als Knoten beschrieben, um das Anlegen von Elementgruppen mit gleicher Erscheinung zu ermöglichen. Elemente oder Elementgruppen können dem Szenengraphen durch Einhängen von Teilgraphen hinzugefügt werden. Bei Manipulationen am Szenengraphen können Teilgraphen gezielt angesprochen werden.

Der Szenengraph ist Kern aller aktuellen *High-Level Grafik-APIs* und Szenenbeschreibungen und bildet die Grundlage vieler Technologien auf dem Gebiet der Computergrafik. Im Vergleich zu *Low-Level Grafik-APIs* wie *OpenGL* oder *Direct3D* bietet diese objektorientierte Beschreibung von Szenen einige Vorteile:

- höhere Abstraktionsebene der Beschreibung
- besser handhabbare Objekte in einer virtuellen Welt
- Objekte werden über eine Menge von Attributen (Fields) definiert, z. B. Menge von 3D-Punkten für ein Geometrieobjekt.
- Objekte sind Elemente von Klassen, die alle Attribute speichern.

Eine der ersten Anwendungen basierend auf einem Szenengraphen war *Open Inventor* [SGIb] der Firma *SGI*. Es handelt sich dabei um ein objektorientiertes 3D-Toolkit für die Erstellung interaktiver Grafikanwendungen auf Basis des 3D-Grafik-API *OpenGL* [SGIa]. Weitere Anwendungen sind die *Virtual Reality Modeling Language (VRML)* sowie deren Nachfolger *Extensible 3D (X3D)* [Weba] und *Java3D* [SUN]. Eine aktuelle Entwicklung auf diesem Gebiet ist der unter Leitung des *OpenSG Forum (ZGDV Darmstadt)* entwickelte Open Source (LGPL) Szenengraph *OpenSG* [Ope].

2.3.2 Spatiale Komposition

Unter spatialer Komposition versteht man die räumliche Anordnung der Medienobjekte in der Szene. Der Autor legt fest, an welchem Ort, mit welcher Ausrichtung und in welcher Größe ein Objekt oder eine Gruppe in der Szene erscheint. Im technischen Sinne beruht die spatiale Komposition auf geometrischen Transformationen innerhalb des Szenengraphen. Zur Beschreibung der Positionen dienen Koordinatensysteme. Diese bilden die mathematische Basis für Anwendungen der Computergrafik. Transformationen sind mathematische Beschreibungen geometrischer Veränderungen von Objekten. Sie werden durch einfache arithmetische Operationen beschrieben und durch Matrizen repräsentiert. Hierfür wird der Vektorraum um eine homogene Koordinate erweitert [FvFH96]. Im Folgenden werden die grundlegenden geometrischen Transformationen im 3D-Raum beschrieben:

1. *Translation:* Die Verschiebung von Punkten im Raum erfolgt durch Addition der Translationswerte zu den Koordinaten (Vektoraddition):

$$x' = x + d_x,\ y' = y + d_y,\ z' = z + d_z \tag{2.1}$$

   Durch Verwendung der homogenen Koordinate lässt sich eine Translation auch durch eine Matrixmultiplikation beschreiben.

   Tranformationsmatrix:

$$T\left(d_x, d_y, d_z\right) = \begin{bmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

2. *Skalierung:* Die Skalierung von Punkten im Raum erfolgt durch Multiplikation der Koordinaten mit den Skalierungsfaktoren (skalare Multiplikation):

$$x' = s_x * x,\ y' = s_y * y,\ z' = s_z * z \tag{2.3}$$

   Tranformationsmatrix:

$$S\left(s_x, s_y, s_z\right) = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

   Wenn $s_x = s_y = s_z$ handelt es sich um eine uniforme Skalierung (Seitenverhältnisse bleiben gleich.).

Einen Sonderfall der Skalierung stellt die Spiegelung dar. Ausgedrückt wird dies durch negative Skalierungsfaktoren.

Beispiel: Spiegelung an der x-Achse

$$S(s_x, s_y, s_z) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.5}$$

3. *Rotation:* Punkte können durch Matrixmultiplikation um eine Achse rotiert werden. Die Drehung im Raum ist um die Hauptachsen definiert.

   Beispiel: Drehung um die z-Achse

$$x' = x * \cos\theta - y * \sin\theta,\ y' = x * \sin\theta + y * \cos\theta \tag{2.6}$$

   Tranformationsmatrizen:

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.7}$$

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.8}$$

$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.9}$$

Verkettung von Transformationen

Sollen mehrere Transformationen hintereinander ausgeführt werden, so sind die einzelnen Matrizen (in umgekehrter Reihenfolge) zu multiplizieren. Dabei ist zu beachten:

- Die Matrixmultiplikation ist assoziativ:
  $A * (B * C) = (A * B) * C = A * B * C$
- Die Matrixmultiplikation ist im Allgemeinen nicht kommutativ!
  $A * B \neq B * A$

Die resultierende Transformationsmatrix enthält nun alle Transformationen:

$$M = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.10}$$

Die Elemente r_{11}, r_{22}, r_{33} enthalten auch die Skalierungskomponente. Transformationen können rückgängig gemacht werden, indem man die Inverse der Transformation auf die Transformationsmatrix anwendet.

2.3.3 Temporale Komposition

Unter temporaler Komposition versteht man die Steuerung des zeitlichen Ablaufes einer Anwendung. Nicht alle Objekte sind zwingend immer sichtbar bzw. hörbar. Ein Objekt hat im Allgemeinen eine Lebensdauer, die durch einen *Anfangzeitpunkt* und einen *Endzeitpunkt* bestimmt wird. Diese Zeitpunkte können entweder absolut (globaler Bezugspunkt, z. B. Start der Anwendung) oder relativ (lokaler Bezugspunkt, z. B. abhängig von der Sichtbarkeit eines anderen Objektes) festgelegt werden. Bei interaktiven Anwendungen gibt es in der Regel keinen durchgängig festgelegten zeitlichen Ablauf, da dieser durch Ereignisse beeinflusst werden kann.

Synchronisation

Man unterscheidet zeitliche Beziehungen zwischen verschiedenen Medienobjekten und zeitliche Beziehungen der Bestandteile eines Medienobjektes [Ste00]:

- *Interobjekt-Synchronisation:* zeitliche Beziehungen zwischen verschiedenen Medienobjekten, z. B. Anfangszeitpunkte der einzelnen Videoobjekte
- *Intraobjekt-Synchronisation:* zeitliche Beziehungen der Bestandteile eines Medienobjektes, z. B. der Einzelbilder eines Videoobjektes

Die Interobjekt-Synchronisation wird während des Authoring-Prozesses vom Autor festgelegt. Die Intraobjekt-Synchronisation hingegen wird von der verwendeten Wiedergabeplattform sichergestellt.

Animationen

Im Sinne multimedialer Anwendungen steht der Begriff *Animation* für ein zeitlich veränderliches Erscheinen von Elementen einer Szene. Es können Position (Bewegungsanimation), Form, Farbe, Transparenz, Struktur und Textur eines Objektes (Objektanimation) zeitabhängig manipuliert werden [FvFH96], [Yas00]. Für gewöhnlich werden Animationen über Parameter beschrieben, d. h. es werden lediglich einige Schlüsselwerte (*Keyframes*) festgelegt, zwischen denen interpoliert wird. Die Bewegungsanimation wird häufig als *Tweening*, die Objektanimation als *Morphing* bezeichnet [Lut94]. Wird eine Animation dennoch manuell Bild für Bild beschrieben, liegt eine Zellanimation vor [Yas00]. Animationen lassen sich nach ihrem Erscheinen unterscheiden in zweidimensionale und dreidimensionale Animationen. Sehr komplexe Formen der Animation sind Körper- und Gesichtsanimationen, die mit Hilfe standardisierter Parameter (*Facial Animation Parameter* [FAP] und *Body Animation Parameter* [BAP]) beschrieben werden [ISO01a], [PE02]. Die Produktion ansprechender Animationen zählt zu den anspruchsvollsten Aufgaben bei der Erstellung multimedialer Anwendungen. Hierfür sind zahlreiche spezialisierte Software-Produkte verfügbar.

2.3.4 Sprachen zur Szenenbeschreibung

Zur textuellen Beschreibung von Szenen für multimediale Anwendungen wurden verschiedene Formate bzw. Sprachen entwickelt. Im Folgenden werden einige dieser Sprachen vorgestellt, die für die Beschreibung objektbasierter AV-Anwendungen von Bedeutung sind.

VRML

Die *Virtual Reality Modeling Language (VRML)* ist eine Szenenbeschreibungssprache für multimediale, interaktive 3D-Szenen und basiert auf *Open Inventor* [SGIb]. VRML wurde ursprünglich für Internet-Anwendungen entwickelt. Im Jahre 1997 wurde die Version 2 unter dem Namen *VRML97* von der ISO/IEC (ISO/IEC 14772) standardisiert [ISO03a]. Mit der Weiterentwicklung befasst sich das *Web3D-Konsortium*[3] [WBS01].

Eine VRML-Szene repräsentiert einen Szenengraphen in textueller Form. Medienobjekte (z. B. Bilder oder Audioobjekte) werden separat gespeichert und durch Verweise in die Szene integriert. Diese Szene kann dynamisch mit verschiedenen Mechanismen beeinflusst werden. Eine VRML-Szene definiert ein *Weltkoordinatensystem*, in das alle Objekte eingebettet werden sowie einen Satz von 3D-Objekten und Medienobjekten. Es können Hyperlinks zu anderen Dateien und Applikationen integriert werden. Die Basis für Interaktionen und Animationen in VRML ist das *Event-Modell*. Felder sind Sender oder Empfänger von Ereignissen, die zwischen Knoten ausgetauscht werden. Ereignisquellen, die Ereignisse bei Aktionen in der Welt produzieren, sind Sensoren. Jeder Sensortyp definiert, wann ein Ereignis generiert wird. Eine weitere Ereignisquelle sind die so genannten Interpolatoren. Dabei handelt es sich um Knoten für die Keyframe-basierte Animation von Objekteigenschaften.

Für die Steuerung von VRML-Szenen zur Laufzeit ist ein Skriptknoten vorgesehen, der eine programmierbare Schnittstelle zum Szenengraphen definiert. Über ihn können komplexe Ereignismodelle aufgebaut werden. Ein Skriptknoten besteht aus frei definierbaren öffentlichen und privaten Feldern und Programmcode. Dieser kann direkt in die Datei eingebettet oder über eine URL verlinkt sein. Mit einem solchen Skript können Werte von Feldern gespeichert, transformiert und logisch miteinander verknüpft werden. Es sind zwei Arten von Skripten vorgesehen. Die erste Möglichkeit ist die Integration von *ECMAScript* [ISO02a] direkt in die VRML-Datei. Mit dieser auf *JavaScript* basierenden Skriptsprache kann Code in einer Sandbox des Browsers ausgeführt werden. Die andere Möglichkeit ist das *External Authoring Interface (EAI)*. Dieses erlaubt Interaktionen zwischen Java-Bytecode und dem VRML-Szenengraphen. Dem Entwickler stehen alle Konstrukte der Java-Plattform zur Verfügung [ISO03b]. Eine weitere wichtige Eigenschaft von VRML ist die Definition von neuen Knotentypen unter Verwendung bereits definierter Knotentypen: das so genannte *PROTO-Typing*. Einmal definiert, können diese neuen Knotentypen ebenso im Szenengraphen verwendet werden wie herkömmliche VRML-Knoten. Diese *PROTOs* werden innerhalb der VRML-Datei oder in einer externen Datei gespeichert (*EXTERNPROTO*).

Die Übertragung einer VRML-Datei erfolgt im ASCII-Format oder komprimiert mit *gzip*. Für die Anzeige kommen spezielle VRML-Betrachter oder Plugins des Webbrowsers zum Einsatz. Die Entwicklung von VRML-Szenen kann prinzipiell mit einem Text-Editor erfolgen. Komfortabler ist die Nutzung von grafischen VRML-Autorenwerkzeugen. VRML ist ein verbreitetes Austauschformat für interaktive 3D-Szenen. Viele 3D-Modellierungswerkzeuge ermöglichen den Im- und Export von VRML-Dateien. Auch das Format zur Beschreibung von MPEG-4-Szenen basiert auf VRML (siehe Unterabschnitt 2.4.2). Listing 2.1 zeigt eine einfache VRML-Szene. Es werden eine Lichtquelle und eine rote Kugel beschrieben.

[3] http://www.web3d.org/

```
#VRML V2.0 utf8
Transform {
 children [ NavigationInfo { headlight FALSE }
  DirectionalLight { direction 0 0 -1 }
  Transform {
   translation 3 0 1
    children [
     Shape {
      geometry Sphere { radius 2.3 }
      appearance Appearance {
       material Material { diffuseColor 1 0 0 }
     }
    }
   ]
  }
 ]
}
```

Listing 2.1: Beispiel für eine Szenenbeschreibung in VRML

X3D

Unter der Bezeichnung *VRML Next Generation (VRML NG)* wurden vom Web3D-Konsortium zunächst die Arbeiten an VRML weitergeführt. Im Jahre 1999 erfolgte die Umbenennung des Projektes in *Extensible 3D (X3D)* [WBS01]. Dieser neue Name sollte die Beziehung der Arbeiten zu XML deutlich machen. Die Standardisierung erfolgte 2004 unter der Bezeichnung *ISO/IEC 19775: Extensible 3D (X3D)* [ISO04a], [ISO04b], [Weba]. X3D ist damit der offizielle Nachfolger von VRML, konnte sich aber als Austauschformat für 3D-Szenen bisher noch nicht durchsetzen.

X3D ist eine XML-konforme Umsetzung des VRML-Standards. Damit wird die Verarbeitung von X3D-Anwendungen mit XML-Werkzeugen möglich. Die Definitionen liegen als *DTD* und *XSD* vor. X3D ist modular aufgebaut und besteht aus einem allgemein gehaltenen, kompakten X3D-Kern, der um spezialisierte Module (z. B. Modellierung von Menschen in *H-Anim*[4] oder die Darstellung geographischer Daten in *GeoSpatial*) erweitert werden kann. Hierdurch werden die Möglichkeiten von VRML deutlich erweitert, beispielsweise ist nun auch die Beschreibung von 2D-Szenen möglich. Der Funktionsumfang ist in *Profiles* unterteilt, um die Interoperabilität mit anderen Anwendungen und Formaten zu erleichtern [Weba]. Die Manipulation des Szenengraphen kann über die Schnittstelle *Scene access interface (SAI)* erfolgen [ISO04b].

Für die Erstellung von X3D-Szenen stehen grafische Autorenwerkzeuge und verschiedene Konverter (z. B. von VRML nach X3D) zur Verfügung. Für die Anzeige kommen spezielle X3D-Betrachter oder Plugins des Webbrowsers zum Einsatz. Listing 2.2 zeigt eine einfache X3D-Szene. Es werden eine Lichtquelle und eine rote Kugel beschrieben.

[4] Humanoid Animation (H-Anim)

```
<X3D version='3.0' profile='Interchange'>
 <head>
  <meta name='filename' content='RedSphere.x3d'/>
 </head>
  <Scene>
   <Transform>
    <NavigationInfo headlight='false'
     avatarSize='0.25 1.6 0.75' type='"EXAMINE"'/>
     <DirectionalLight/>
      <Transform translation='3.0 0.0 1.0'>
       <Shape>
        <Sphere radius='2.3'/>
        <Appearance>
          <Material diffuseColor='1.0 0.0 0.0'/>
         </Appearance>
       </Shape>
   </Transform>
  </Scene>
</X3D>
```

Listing 2.2: Beispiel für eine Szenenbeschreibung in X3D

SMIL

Die *Synchronized Multimedia Integration Language (SMIL)* ist eine XML-basierte Sprache für die Beschreibung interaktiver Multimedia-Anwendungen. Sie wurde vom *World Wide Web Consortium (W3C)*[5] entwickelt und hat den Status einer *W3C Recommendation (REC)* [W3Ce]. Typische Einsatzgebiete sind zweidimensionale Präsentationen mit Streaming-Audio und -Video, Grafik und Text [W3Cd]. SMIL beschreibt nicht die Medienobjekte selbst, sondern lediglich deren spatiale und temporale Komposition auf einer hohen Abstraktionsebene. Somit kann SMIL als Authoring-Format bezeichnet werden (siehe Abschnitte 3.3 und 4.4). SMIL ist unterteilt in zehn funktionelle Einheiten, u. a. für Animation, Layout und Medienobjekte. Jede dieser Einheiten ist wiederum in Module unterteilt. Für die Erstellung von SMIL-Anwendungen sind einige grafische Autorenwerkzeuge verfügbar. Für die Wiedergabe werden spezielle Betrachter oder Plugins des Webbrowsers verwendet. Listing 2.3 zeigt eine einfache SMIL-Szene.

SVG

Scalable Vector Graphics (SVG) ist eine Sprache für die XML-basierte Beschreibung zweidimensionaler Grafiken. Sie wurde ebenfalls vom *World Wide Web Consortium (W3C)* entwickelt und hat den Status einer *W3C Recommendation (REC)* [W3Cc]. SVG erlaubt die Beschreibung von Vektorgrafiken, Bildern und Text [W3Cb]. Auch SVG ist modular aufgebaut und es wurden Profiles für verschiedene Anwendungen definiert, z. B. *SVG-Mobile* für die Anzeige auf Mobiltelefonen und *SVG-Print* für Druckerzeugnisse [W3Cb]. SVG ist in viele Grafikprogramme als Export-Format integriert. Für die Anzeige werden spezielle Betrachter oder Plugins des Webbrowsers verwendet. Listing 2.4 zeigt eine einfache SVG-Szene.

[5] http://www.w3c.org/

```
<smil>
  <head>
    <meta name="title" content="SMIL-Beispiel"/>
     <layout>
       <root-layout width ="250" height="190"/>
       <region id="video" top="5" left="5" width="240" height="180" z-index="1"/>
     </layout>
  </head>
  <body>
    <seq>
      <video src="media/videobeispiel.rm" region="video" />
      <audio src="media/audiobeispiel.rm"/>
    </seq>
  </body>
</smil>
```

Listing 2.3: Beispiel für eine Szenenbeschreibung in SMIL

```
<svg width="300px" height="300px">
   <title>Small SVG example</title>
   <circle cx="120" cy="150" r="60" style="fill: gold;" />
   <polyline points="120 30, 25 150, 290 150"
    stroke-width="4" stroke="brown" style="fill: none;" />
   <polygon points="210 100, 210 200, 270 150"
    style="fill: lawngreen;" />
   <text x="60" y="250" fill="blue">Hello, World!</text>
</svg>
```

Listing 2.4: Beispiel für eine Szenenbeschreibung in SVG

Fazit

Die hier vorgestellten Sprachen wurden mit sehr verschiedenen Zielsetzungen entwickelt. Auf Grund der gemeinsamen Basis XML können X3D, SMIL und SVG sehr gut in Kombination verwendet werden, z. B. SMIL und X3D für interaktive 3D-Anwendungen oder SMIL und SVG für interaktive 2D-Anwendungen. Die Verarbeitung erfolgt mit XML-Werkzeugen, z. B. XSL-Transformationen oder Parsern. In den Unterabschnitten 2.4.2 und 2.4.3 sind die Szenenbeschreibungsformate aus MPEG-4 zu finden. In [CD02] werden die vorgestellten Formate in kompakter Form gegenübergestellt und mit zwei proprietären Formaten verglichen. In [LMBS01] ist ein Verfahren zur Konvertierung von *Macromedia Flash Shockwave* nach MPEG-4-BIFS beschrieben. In [CE02] ist eine ausführliche Gegenüberstellung von SMIL und BIFS zu finden.

2.4 Objekt- und Szenenkonzept von MPEG-4

Das Objekt- und Szenenkonzept ist in dem von der Moving Picture Experts Group (MPEG) entwickelten Standard MPEG-4 (ISO/IEC 14496) erstmals umfassend beschrieben worden. Eine MPEG-4 Szene besteht aus auditiven, visuellen und audiovisuellen

Objekten, welche durch eine Szenenbeschreibung örtlich und zeitlich zu einer audiovisuellen Szene verknüpft werden. Diese Objekte können natürlichen Ursprungs in Form von Mikrofon- bzw. Kameraaufnahmen oder synthetisch mit einem Computer generierte Objekte sein und werden *Medienobjekte* genannt [ISO01a]. Der objektorientierte Ansatz ermöglicht eine separate Verarbeitung einzelner Medienobjekte, wodurch sich Vorteile beim Entwurf, bei der Codierung und beim Transport ergeben. Medienobjekte werden als *Elementary Streams (ES)* codiert, welche einen Teil der MPEG-4-Architektur darstellen und im *Object Description Framework (ODF)* beschrieben sind. Als Terminal ist gemäß MPEG-4 ein Wiedergabegerät zu verstehen, welches sowohl Szenen anzeigt als auch Nutzeraktionen entgegennimmt. Durch Interaktionen ist es möglich, Medienobjekte hinzuzufügen, zu löschen oder ihre Gestalt zu verändern [PE02], [ISO02b], [ISO01a], [WBS02].

2.4.1 MPEG-4: Eine Übersicht

Die Arbeiten am MPEG-4-Standard begannen im Juli 1993 und hatten zunächst die Codierung audiovisueller Inhalte für sehr niedrige Bandbreiten zum Ziel. Diese Fokussierung wurde nach einiger Zeit auf Grund geänderter Anforderungen teilweise fallen gelassen und man wendete sich der Codierung audiovisueller Szenen zu. Ein Schwerpunkt sollte dabei die Kombination von synthetischen und natürlichen Inhalten sein. Es wurde der Begriff *Synthetic/Natural Hybrid Coding (SNHC)* geprägt [PE02]. Der Standard wurde im Januar 1996 schließlich „Coding of audio-visual objects“ genannt. Im März 1999 wurde MPEG-4 ein internationaler Standard unter der Bezeichnung „ISO/IEC 14496: Information technology – Coding of audio-visual objects (MPEG-4)“ [ISO98], [MPE], [PE02]. Im Januar 2000 folgte die Version 2, die Version 3 ist in Arbeit. Die höheren Versionen erweitern jeweils den Standard – stellen also keine Versionen im eigentlichen Sinne dar.

Anders als die zuvor verabschiedeten und sehr erfolgreichen Standards MPEG-1 und MPEG-2 widmet sich MPEG-4 nicht nur der Codierung von Audio- und Videodaten, sondern ermöglicht die Beschreibung audiovisueller Inhalte jeglicher Art. Beispielsweise lassen sich neben klassischen Videodaten auch computergenerierte Grafiken sowie Körper- und Gesichtsanimationen effizient codieren. Eine MPEG-4-Anwendung kann prinzipiell aus vielen verschiedenen Medientypen bestehen. Die Entwicklung von MPEG-4 zielt auf die folgenden Anwendungsgebiete [ISO02b]:

- digitales Fernsehen
- interaktive Grafikanwendungen
- interaktive Multimedia-Anwendungen

MPEG-4 bietet standardisierte Elemente für die integrierte Produktion, Distribution und Nutzung von Inhalten aus den genannten Feldern. Der Standard stellt eine Reihe an Technologien zur Verfügung, welche die Bedürfnisse von Autoren, Anbietern und Konsumenten gleichermaßen befriedigen sollen [ISO02b]:

- *Autoren* ermöglicht MPEG-4 die Erstellung wiederverwendbarer Inhalte. Die gebotene Flexibilität ist weitaus höher, als dies mit heute verbreiteten Einzeltechnologien, wie dem digitalen Fernsehen, animierten Grafiken, Webseiten und deren Erweiterungen möglich ist. Ebenso ist eine bessere Verwaltung der Inhalte und der Schutz von Urheberrechten realisierbar.

- *Anbietern und Service-Providern* stellt MPEG-4 Möglichkeiten einer transparenten Distribution unter Berücksichtigung der Qualität der Dienste (*Quality of Service*) zur Verfügung. MPEG-4 definiert einen allgemeinen *QoS-Deskriptor* für verschiedene MPEG-4-Medien. Die konkrete Umsetzung der QoS-Parameter wird nicht innerhalb des Standards geregelt und wird den Netzwerkprovidern überlassen.

- *Nutzern* bietet MPEG-4 neue Interaktionsmöglichkeiten mit den Inhalten, die völlig neue Rezeptionsmöglichkeiten und mediale Erfahrungen erschließen. Multimediale Anwendungen stehen in neuen Netzwerken zur Verfügung; eingeschlossen derer mit relativ niedrigen Datenraten und mobilen Zugängen.

Die Teile des Standards

Der MPEG-4 Standard ist in verschiedene Teile (*Parts*) unterteilt. Gegenwärtig existieren insgesamt 21 Parts[6]. Die grundlegenden Parts von MPEG-4 sind:

- *ISO/IEC 14496-1 Systems:* Komposition und Codierung audiovisueller Szenen

- *ISO/IEC 14496-2 Visual:* Codierung von natürlichen und synthetischen visuellen Objekten

- *ISO/IEC 14496-3 Audio:* Codierung von natürlichen und synthetischen auditiven Objekten

- *ISO/IEC 14496-6 DMIF (Delivery Multimedia Integration Framework):* Synchronisation von audiovisuellen Inhalten und deren Transport über verschiedene Netzwerke

In der kommenden Version 3 des Standards wird der bisher bekannte Part 1 aufgeteilt in einen neuen Part 1 und *Part 11: ISO/IEC 14496-1 Scene Description and Application Engine*. Die für das Authoring interessanten Szenenbeschreibungen sind dann in Part 11 zu finden. In der vorliegenden Arbeit wird weiterhin die noch gültige Aufteilung (ohne Part 11) verwendet.

Profiles in MPEG-4

Der MPEG-4-Standard deckt sehr viele Anwendungsgebiete ab und hat daher einen enormen Umfang erreicht. Es sind kaum Anwendungen vorstellbar, die den vollen Umfang des Standards nutzen. Um die Entwicklung von Applikationen zu ermöglichen, wurden Untermengen definiert; die *Profiles*. Diese werden für folgende Bereiche spezifiziert:

Scenegraph, Descriptors, Visual, Audio, Graphics und MPEG-J[7].

Ein *Profile* grenzt die nutzbaren Funktionen des Standards ein und erleichtert damit die Implementierung eines Decoders. Die *Profiles* werden in Kombination mit *Levels* verwendet. Diese begrenzen die Parameter des zu decodierenden Signals. Der erforderliche Leistungsumfang eines MPEG-4-Terminals wird durch eine Kombination aus *Profile* und *Level* bestimmt (*Profile@Level*).

Grundlegende Konzepte

Der MPEG-4-Standard spezifiziert Methoden und Werkzeuge für die Beschreibung audiovisueller Szenen. Diese Spezifikation umfasst folgende Bestandteile [ISO01a]:

[6] Stand: Frühjahr 2005

[7] MPEG-J ist eine in MPEG-4 definierte Java-basierte Application Engine.

1. die codierte Repräsentation audiovisueller Szenen, bestehend aus natürlichen und synthetischen Objekten
2. die codierte Beschreibung der räumlichen und zeitlichen Anordnung der Objekte und die Definition von Interaktionsmechanismen
3. die Beschreibung von Informationen zur Verwaltung der Medienströme (IDs, Synchronisation, Prioritäten)
4. eine allgemeine Schnittstelle zur Funktionalität der Transportschicht
5. eine *Application Engine* zur Steuerung des Terminals
6. ein Dateiformat zur Speicherung von MPEG-4-Anwendungen

MPEG-4 Systems

Für die Entwicklung von Autorenwerkzeugen für MPEG-4-Anwendungen ist vor allem der *Part1: Systems* [ISO01a] von Interesse. Er umfasst folgende Teile:

1. ein Terminal-Modell für Zeit- und Puffermanagement
2. die codierte Repräsentation der Szenenbeschreibung *(BIFS - Binary Format for Scenes)*
3. die codierte Beschreibung von Metadaten für Identifikation, Beschreibung und logische Abhängigkeiten von Elementary Streams *(ODF - Object Description Framework)*
4. die codierte Repräsentation inhaltsbeschreibender Daten zu den audiovisuellen Inhalten *(OCI - Object Content Information)*
5. eine Schnittstelle zur Verwaltung von Urheberrechten und entsprechende Schutzmechanismen *(IPMP - Intellectual Property Management and Protection System)*
6. die codierte Beschreibung von Synchronisationsinformationen *(SL - SyncLayer)*
7. die zusammengesetzte Repräsentation mehrerer *ES (Elementary Stream)* in einem einzelnen Stream (FlexMux)
8. eine *Application Engine (MPEG-J)*

Object Description Framework – ODF

Die *Object Descriptors (OD)* stellen das Bindeglied zwischen Szenenbeschreibung und *Elementary Streams (ES)* dar. Sie enthalten Daten über den *Elementarstrom* und dem damit verbundenen Medienobjekt. Dieser indirekte Zugriffsmechanismus dient der Separation zwischen Szenenstruktur, Mediendaten und Transportmitteln und schließt störende Beeinflussungen aus. Das *Object Description Framework (ODF)* regelt die Identifikation und Zuordnung von ES zu Medienobjekten innerhalb einer Szenenbeschreibung. Medienobjekte eines ES sind eindeutig mit einer *Object DescriptorID* gekennzeichnet. Jeder OD bildet eine Kollektion von Deskriptoren, welche einen ES beschreiben. Diese Ströme enthalten sowohl komprimierte als auch unkomprimierte Mediendaten, aber auch Metadaten

in Form der *Object Content Information (OCI)*. Die Metadaten helfen u. a. dem Autor und dem Nutzer Inhalte zu identifizieren. Mehrere ES können die gleichen Medienobjekte enthalten, die sich jedoch in Form oder Ausprägung unterscheiden (z. B. verschiedensprachige oder skalierbare Inhalte). Ein ES-Deskriptor identifiziert einen einzelnen Strom mit einer Nummer (*ES_ID*) und einer Quell-URL. Jeder ES-Deskriptor enthält zudem wichtige Informationen zur Initialisierung und Konfiguration des Decoders. Abbildung 2.7 zeigt die Verknüpfung der OD mit der Szenenbeschreibung.

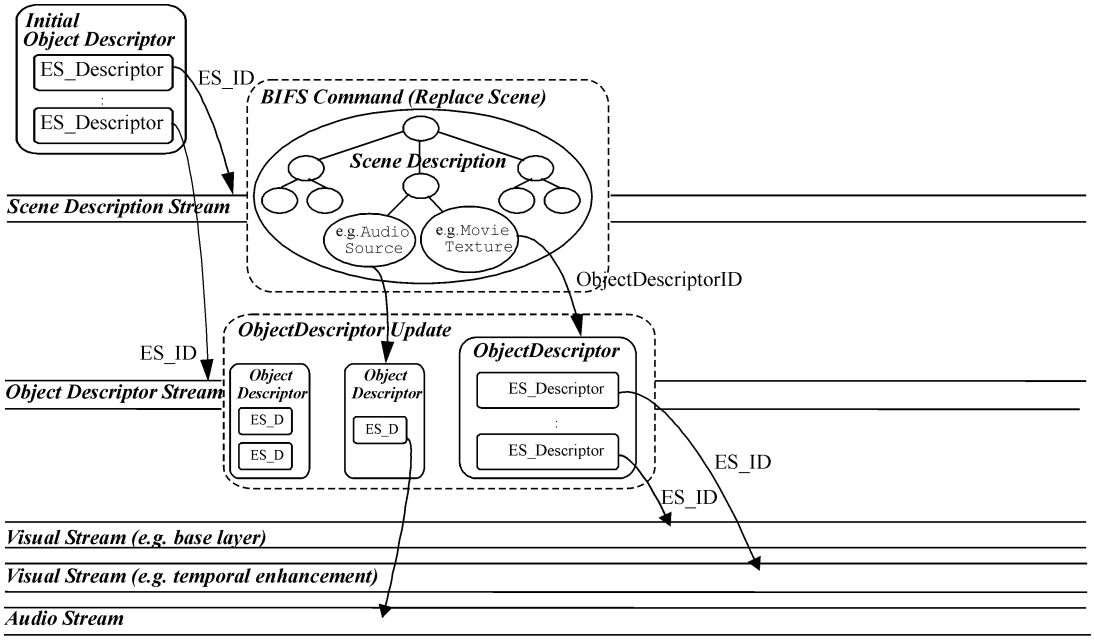

Abbildung 2.7: Object Description Framework [ISO01a]

Der *Initial Object Descriptor (IOD)* ist die einfachste Form eines *OD* und sorgt für die Initialisierung der Decodierung. Nach der Initialisierung ist ausschließlich die Szenenbeschreibung für die Steuerung der Anwendung verantwortlich [ISO01a]. Ein IOD enthält zusätzliche Informationen über die *Profiles*, die das Terminal unterstützen muss, um die beschriebenen Inhalte darstellen zu können.

Scene Description Framework – SDF

Die Szenenbeschreibung legt die spatiale und temporale Komposition aller Medienobjekte zu einer Szene fest. Sie beschreibt die Struktur einer Szene und deren Verhalten [ISO01a]. Abbildung 2.8 zeigt eine typische Szene mit verschiedenen auditiven und visuellen Medienobjekten in einer dreidimensionalen Umgebung. Durch Interaktionen oder Kommandos innerhalb der Szene lassen sich die Objekte manipulieren, z. B. verschieben, löschen oder neu hinzufügen. Auch die Szene selbst kann beeinflusst werden. Die Beschreibung einer solchen Szene erfolgt in einer hierarchischen Baumstruktur, wie sie in Abbildung 2.9 dargestellt ist. Die Grundlage dafür ist die textuelle Beschreibung dieses baumartigen Szenengraphen (siehe Abschnitt 2.3).

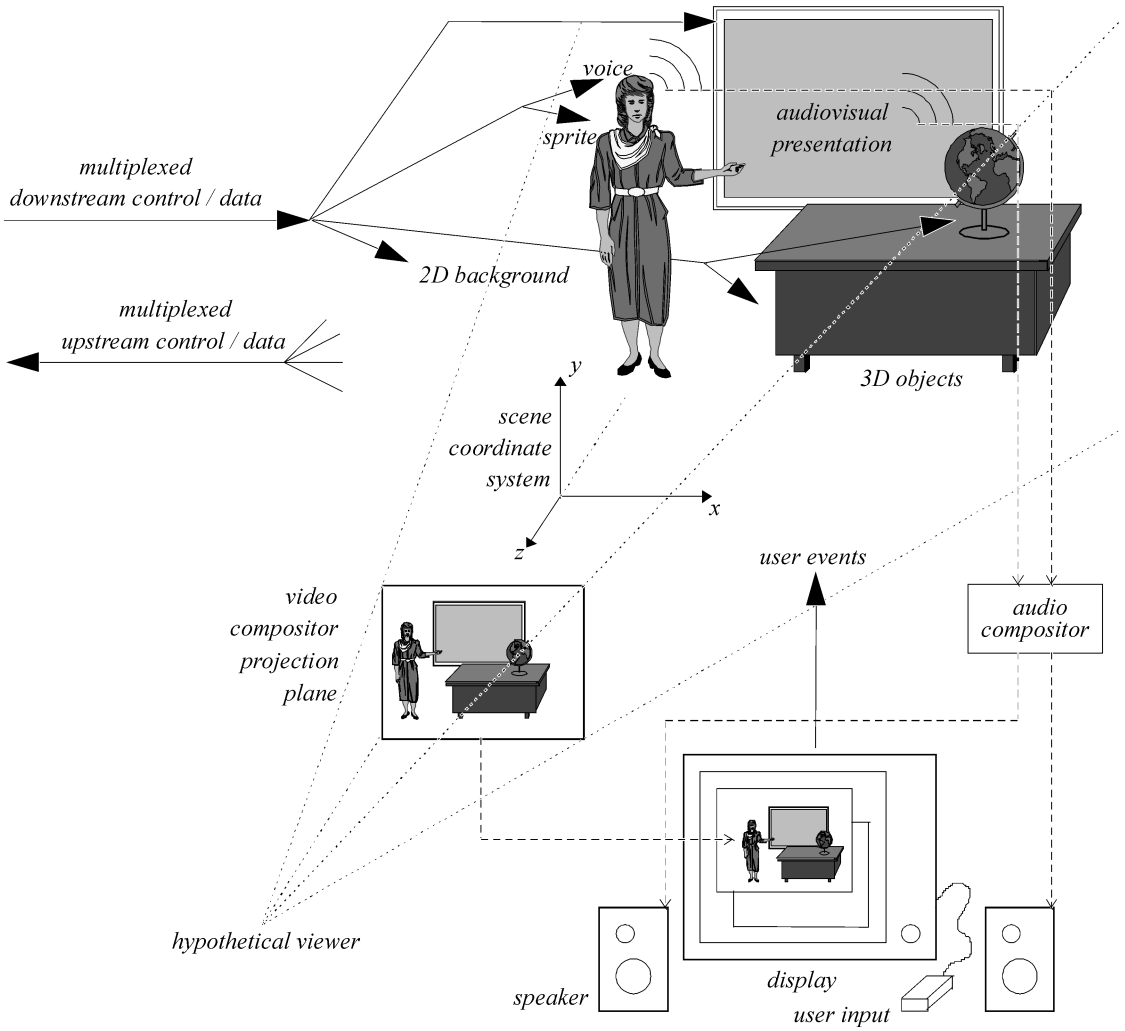

Abbildung 2.8: Objektbasierte Multimedia-Anwendung [ISO01a]

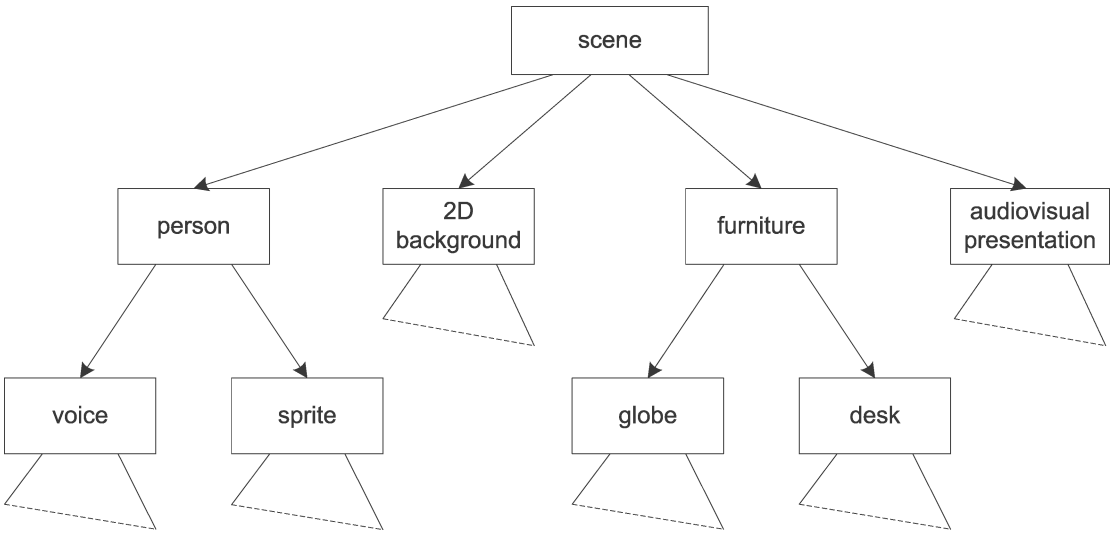

Abbildung 2.9: Logische Struktur der Beipielszene [ISO01a]

2.4.2 Binary Format for Scenes BIFS

Das *Binary Format for Scenes (BIFS)* dient der Beschreibung zweidimensionaler und dreidimensionaler Szenen in MPEG-4. Auch die Kombination von 2D- und 3D-Inhalten ist vorgesehen. Das Format ermöglicht Interaktionen mit den Szeneninhalten und die Einflussnahme auf die Szenendaten. BIFS basiert auf VRML (siehe Unterabschnitt 2.3.4); erweitert dessen Möglichkeiten aber erheblich [WBS01]. Um MPEG-4-Anwendungen effektiv über verschiedene Distributionswege übertragen zu können, sind die Szenendaten in geeigneter Weise zu encodieren. Die textuelle (ebenso wie die gzip-komprimierte) Übertragung von VRML-Dateien ist für eine Distribution über Rundfunk auf Grund der fehlenden Streaming-Fähigkeit nicht geeignet. BIFS verwendet eine binäre Komprimierung, wodurch sich die Datenmenge im Vergleich zu VRML (UTF-8) oft auf weniger als ein Zehntel reduziert [PE02], [WBS01], [ISO02b], [WBS02].

BIFS stellt eine binäre Repräsentation eines vordefinierten Satzes von Szenenobjekten und deren Verhalten bezüglich ihrer räumlichen und zeitlichen Beziehungen dar. Folgende Informationstypen werden beschrieben:

1. Attribute der audiovisuellen Medienobjekte (MO)
2. Struktur des Szenengraphen
3. vordefinierte räumliche oder zeitliche Veränderungen (bzw. Eigenverhalten) dieser MO, unabhängig von Nutzereingaben
4. räumliche und zeitliche Veränderungen abhängig von Nutzereingaben

Neben den umfangreichen visuellen Möglichkeiten ist auch die Beschreibung auditiver Elemente und Szenen vorgesehen. *AudioBIFS* ermöglicht die Komposition von natürlichen und synthetischen Audioobjekten zu einer (interaktiven) auditiven Szene. Eine solche Szene kann mit einem physikalischen (*physical approach*) oder einem wahrnehmungsbezogenen Ansatz (*perceptual approach*) beschrieben werden [SVH99], [PE02]. Mit diesen Möglichkeiten lassen sich in Verbindung mit entsprechenden Wiedergabesystemen neuartige interaktive Anwendungen mit einem hohen Immersionsgrad realisieren [DRSD03], [RMS01].

Ausgangspunkt für die Encodierung in das BIFS-Format ist eine textuelle Beschreibung der Szene. Hierfür gibt es zwei Möglichkeiten:

1. textuelle Szenenbeschreibung basierend auf VRML
2. standardisierte Szenenbeschreibung *Extensible MPEG-4 Textual Format (XMT)* auf Basis von XML

Die VRML-basierte Szenenbeschreibung ist eine direkte Umsetzung des binären BIFS in die textuelle Form und ermöglicht eine Lesbarkeit und eine einfache Darstellung für den Autor. Die textuelle Beschreibung wurde nicht standardisiert, orientiert sich aber stark an VRML. Es haben sich zwei encoderabhängige Ausprägungen herausgebildet. Gebräuchliche Bezeichnungen sind *textuelles BIFS*, *Textual Format for Scenes (TeFS)* oder *BIFS-Text (BT)*. Das textuelle Format wird an einen Encoder übergeben und in das BIFS encodiert [PE02], [WBS02]. Die encodierte Szenenbeschreibung wird als separater Elementarstrom in eine MPEG-4-Anwendung eingebettet. Die XML-basierte Szenenbeschreibung ist Inhalt des nächsten Unterabschnittes.

2.4.3 Extensible MPEG-4 Textual Format XMT

Das *Extensible MPEG-4 Textual Format (XMT)* definiert eine Zwei-Schichten-Architektur für die XML-basierte Beschreibung von MPEG-4-Anwendungen und ist Teil des MPEG-4-Standards [ISO01a][8]. XMT ermöglicht im Vergleich zu den textuellen Szenenbeschreibungen auf Basis von VRML eine bessere Lesbarkeit und gewährleistet die Interoperabilität mit anderen XML-basierten Sprachen (siehe Abbildung 2.10) [KWC00], [PE02]. Die

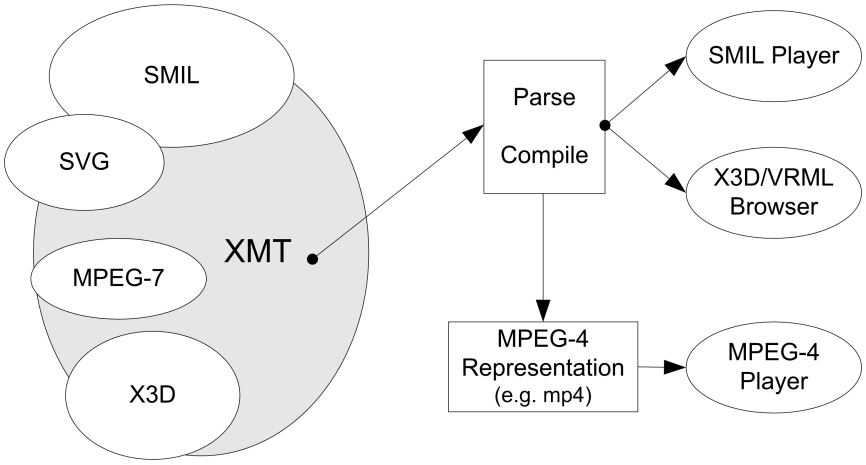

Abbildung 2.10: Interoperabilität von XMT [KWC00]

beiden Schichten der Architektur werden durch die Sprachen XMT-A und XMT-Ω gebildet, die der Beschreibung von Szenen auf verschiedenen Abstraktionsebenen dienen. Ergänzt werden diese durch die XMT-C-Module zur Beschreibung von Metadaten (MPEG-7) und weiterer Zusatzinformationen (z. B. Authoring-Informationen, *encoding hints*, *delivery hints* und *publication hints*). Die Teile von XMT liegen als XML-Schemadefinitionen vor. Abbildung 2.11 zeigt die Architektur von XMT und die Einordnung der Komponenten in die Abstraktionsebenen.

XMT-A

XMT-A ist eine textuelle Repräsentation von BIFS und liegt somit auf derselben semantischen Ebene wie die textuellen BIFS-Formate (siehe Abbildung 2.11). Eine XMT-A-Szene besteht aus einem *Header* und einem *Body*. Im Body ist die eigentliche Szenenbeschreibung abgelegt; der *Header* beinhaltet die Objekt-Deskriptoren. Listing 2.5 zeigt die Struktur einer einfachen XMT-A-Szene. Die XML-basierte Beschreibung ermöglicht eine einfache Verarbeitung mit XML-Werkzeugen. Unter Berücksichtigung einiger Randbedingungen lassen sich z. B. im X3D-Format erstellte Inhalte direkt in das XMT-A-Format überführen. Eine direkte Encodierung in eine binäre MPEG-4-Anwendung ist nicht mit allen Encodern möglich. In diesen Fällen ist eine Konvertierung vom XMT-A-Format in ein textuelles BIFS-Format (z. B. BT) notwendig, welches dann mit den bekannten Encodern verarbeitet werden kann.

[8] XMT ist zurzeit noch in Part 1 (ISO/IEC 14496-1:2001) enthalten. In Zukunft wird es Bestandteil von Part 11 (ISO/IEC 14496-11.3:2004) sein.

```
<XMT-A>
  <Header>
    <meta name="title" content="XMT-A-Beispiel"/>
    <InitialObjectDescriptor .../>
  </Header>
  <Body>
    <Replace>
      <Scene>
        <!-- Szenenbeschreibung -->
      </Scene>
    </Replace>
  </Body>
</XMT-A>
```

Listing 2.5: Szenenstruktur von XMT-A

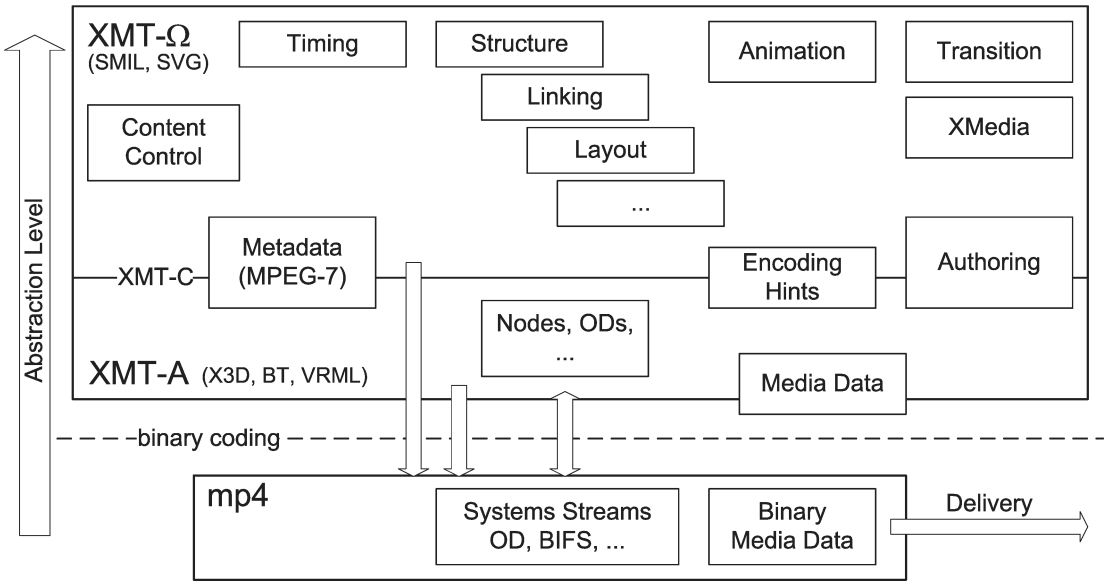

Abbildung 2.11: Architektur von XMT nach [PE02]

XMT-Ω

XMT-Ω (auch XMT-O genannt) ist auf einer höheren Abstraktionsebene angesiedelt und somit intuitiver zu verwenden als XMT-A. Hier steht weniger die Codierung expliziter Knoten im Vordergrund, sondern die Beschreibung audiovisueller Objekte und deren Beziehungen zueinander aus Sicht des Autors. Ausgangspunkt waren die Beschreibungssprachen *SMIL* und *SVG* des W3C (siehe Unterabschnitt 2.3.4). XMT-Ω übernimmt zahlreiche Methoden und Funktionen von SMIL und ergänzt diese. Listing 2.6 zeigt eine einfache Szene in XMT-Ω, in der eine texturierte Box rotiert, sobald sie mit der Maus angeklickt wird. XMT-Ω kann als Austauschformat für Autoren und Autorenwerkzeuge verwendet werden sowie Interoperabilität mit SMIL und SVG gewährleisten. Die Integration von XMT-A-Szenen oder -Elementen in eine XMT-Ω-Szene ist mit Hilfe eines speziellen Mechanismus möglich (*identified low level escape section*). Vor der Erzeugung einer abspielbaren

```
<XMT-O>
  <head>
    <layout metrics="pixel" type="xmt/xmt-basic-layout">
      <topLayout backgroundColor="black" height="400" width="600"/>
    </layout>
  </head>
  <body>
    <par>
      <box begin="0s" dur="indefinite" id="IavasBox" size="2 1 2">
        <material diffuseColor="0.8 0.8 0.8"/>
        <texture src="logo.jpg"/>
        <animate attributeName="rotation" begin="IavasBox.click" calcMode="linear"
                 keyTimes="0; 0.20; 0.5; 0.75; 1" values="0; 90; 180; 270; 360"/>
     </box>
    </par>
  </body>
</XMT-O>
```

Listing 2.6: Beispiel für eine Szenenbeschreibung in XMT-Ω

MPEG-4-Anwendung müssen die Szenenbeschreibungen von XMT-Ω nach XMT-A bzw. BT überführt werden. Prinzipbedingt geht hierbei der Kontext der Anwendung verloren. Es existiert keine umkehrbare Konvertierung zwischen den Abstraktionsebenen. Folglich ist eine Überführung von der niedrigen zur hohen Abstraktionsebene nicht bzw. nur unter Zuhilfenahme von Randbedingungen und Zusatzinformationen möglich.

2.4.4 Modifikationen am Szenengraphen

Der MPEG-4-Standard sieht Möglichkeiten für strukturelle Veränderungen des Szenengraphen zur Laufzeit einer MPEG-4-Anwendung vor. Die Objekte einer bereits encodierten Anwendung können mit Hilfe von Kommandos modifiziert, ersetzt oder gelöscht werden. Weiterhin können die audiovisuellen Eigenschaften eines Medienobjektes geändert, Szenen animiert oder Animationen gesteuert werden [ISO01a], [PE02]. Der MPEG-4-Standard bietet hierfür zwei Möglichkeiten an: *BIFS-Anim* ermöglicht die Änderungen von Feldwerten mit einer konstanten Frame-Rate zur Laufzeit der Anwendung; beispielsweise könnte die Farbe oder die Position eines Objektes gesteuert werden. *BIFS-Commands* ermöglichen dynamische Änderungen an einer Szene zu festgesetzten Zeitpunkten. Die BIFS-Commands können für folgende Szenarien eingesetzt werden:

1. Der Server sendet eine komplette Szene. Zu bestimmten Zeitpunkten wird die Szene durch BIFS-Commands geändert.

2. Der Server sendet eine initiale Szene. Durch mehrere Command-Frames wird diese Szene fortlaufend geändert.

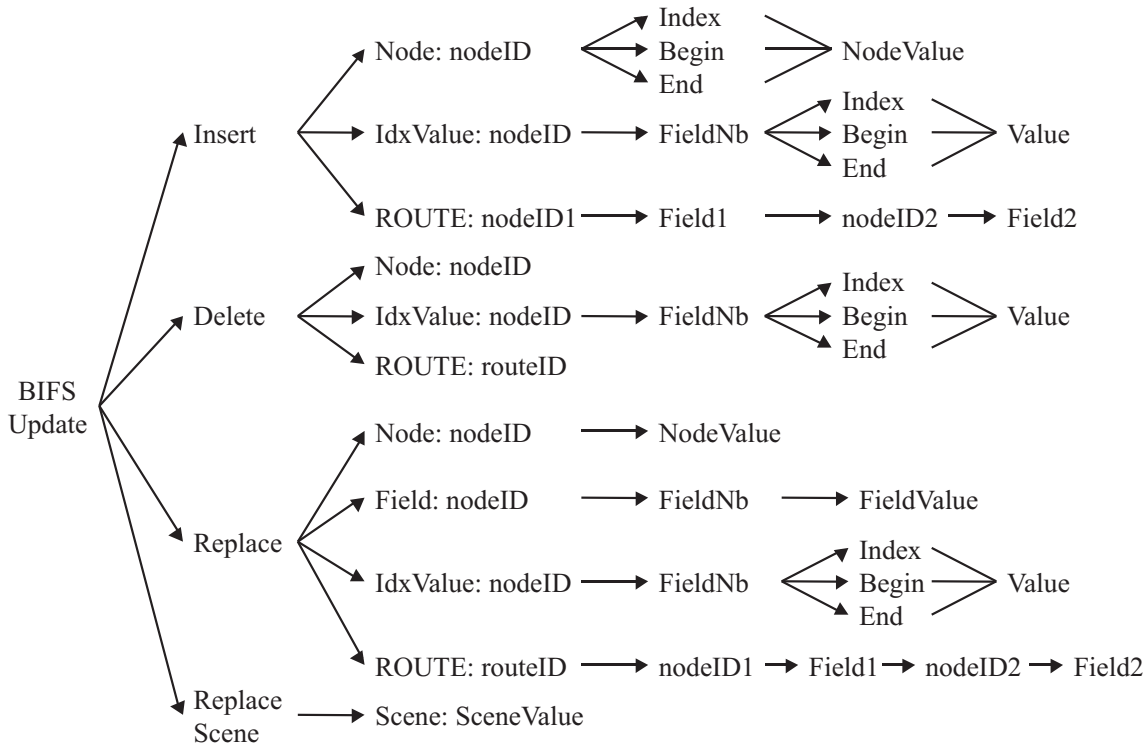

Abbildung 2.12: BIFS-Commands [ISO01a]

Im MPEG-4-Standard werden vier Basis-Kommandos definiert [ISO01a]:

1. Ersetzen einer ganzen Szene (REPLACE SCENE)
2. Einfügen (INSERT)
3. Löschen (DELETE)
4. Ersetzen (REPLACE)

Das erste dieser Kommandos ermöglicht die Ersetzung einer kompletten BIFS-Szene durch einen validen Szenengraphen. Die anderen drei Kommandos können Knoten, indizierte Feldwerte oder ROUTEs modifizieren. Abbildung 2.12 gibt einen Überblick über die BIFS-Commands.

Bei technisch anspruchsvollen Anwendungen werden oft Skript- oder Programmieranweisungen benötigt. Für Skriptanweisungen in MPEG-4-Szenen ist *ECMAScript* vorgesehen, das direkt über einen Skriptknoten in die Szene eingebunden werden kann. Mit Hilfe von ECMAScript eröffnen sich viele Möglichkeiten; größere Skripte werden jedoch unübersichtlich und wirken sich auf Grund der textuellen Darstellung negativ auf den Umfang einer MPEG-4-Anwendung aus. Weitergehende Fähigkeiten bietet das *MPEG-J API*, das den Zugriff auf den Szenengraphen oder dessen Umgebung (Terminal, Decoder, Netzwerkressourcen) ermöglicht. Entsprechende Module werden objektorientiert in Java programmiert, kompiliert und dann als MPEG-J-Ströme (so genannte *MPEGlets*) übertragen. So werden sehr anspruchsvolle Anwendungen möglich und im Gegensatz zu ECMAScript bleibt der Quellcode übersichtlich und die Dateien klein [ISO01a], [PE02].

2.4.5 MPEG-4 Terminal

MPEG-4 definiert eine sehr flexible Architektur für die Verteilung und Nutzung von Anwendungen: das *MPEG-4 Client Terminal* [ISO01a]. Die Quelle überträgt Ströme von komprimierten audiovisuellen Objekten und die zugehörigen Szenen- und Objektbeschreibungen im Multiplex-Verfahren zum Empfänger. Dort werden alle Informationen decodiert und entsprechend der Szenenbeschreibung zusammengesetzt. Der Nutzer kann mit der entstehenden Szene je nach vorgesehenen Möglichkeiten interagieren [PE02]. Abbildung 2.13 zeigt den Aufbau eines solchen Terminals [ISO01a]. Bevor eine Szene beim Benutzer dargestellt wird, durchlaufen die Objekte drei Schichten:

1. Die *Transportschicht (delivery layer)* besteht aus dem Transport-Multiplex (*TransMux*) und dem MPEG-4-Multiplex (*FlexMux*). MPEG-4 definiert keine spezielle Transportschicht, sondern setzt auf vorhandenen Standards auf. Dies können Transportmöglichkeiten nach MPEG-2, Internetprotokolle oder auch Protokolle des digitalen Rundfunks sein. Die zugehörige Schnittstelle heißt *Delivery Multimedia Interchange Format Application Interface (DAI)*. Diese Schicht ist unabhängig vom Medium, aber abhängig vom Transport.

2. Die *Sychronisationsschicht (sync layer)* paketiert die Elementarströme in Zugriffseinheiten (*access units*) und versieht diese mit Zeitmarken, um eine Synchronisation zu ermöglichen. Diese Schicht ist von Medium und Transport unabhängig.

3. Die *Kompressionsschicht (compression layer)* decodiert Daten aus ihrem ursprünglichen codierten Format über das *Elementary Stream Interface* und stellt die notwendigen Operationen bereit, um die Objekte zu rekonstruieren und darzustellen. Die Kompressionsschicht enthält die benötigten Decoder für alle Objekte. Diese Schicht ist abhängig vom Medium und unabhängig vom Transport.

Zwischen den einzelnen Schichten können Rechte ausgehandelt werden, um den Szeneninhalt oder einzelne Objekte zu schützen. *Intellectual Property Rights (IPR)* werden über *IPMP Module* verwaltet und organisiert. Schließlich werden die decodierten Informationen auf dem Terminal des Benutzers dargestellt. Zusammenstellung, Präsentation und Rendering sind hierbei abhängig vom verwendeten System. Dieser Teil des MPEG-4-Terminals wird oft auch als *Präsentationsschicht* bezeichnet. Der Benutzer kann mit der Szene interagieren und Informationen mit dem Programmanbieter über den (optionalen) Interaktionskanal austauschen.

2.4.6 Dateiformat MP4

Für die Distribution von MPEG-4-Anwendungen wurde ein eigenes Dateiformat auf Basis von *Quicktime* spezifiziert [ISO02b]. Eine MP4-Datei enthält alle Medieninformationen der audiovisuellen Szene in binärer Form. Die Medien werden auf einer abstrakten Ebene durch *Tracks* beschrieben. Diese Tracks sind in der Dateistruktur eingebettete (*self-contained MP4*) oder auch externe Medienströme. Es liegt eine objektorientierte Struktur vor, bei der ein Objekt als *Atom* bezeichnet wird. Diese Struktur ermöglicht den gezielten Zugriff auf Tracks sowohl lesend als auch schreibend; aus einer MP4-Datei heraus kann auch auf Tracks in einer anderen MP4-Datei verwiesen werden. Ergänzende Informationen (z. B. Dauer oder Größe) werden in einer separaten Struktur gehalten. Diese Metadaten beschreiben die mit ihnen verbundenen Medienobjekte durch Verweise.

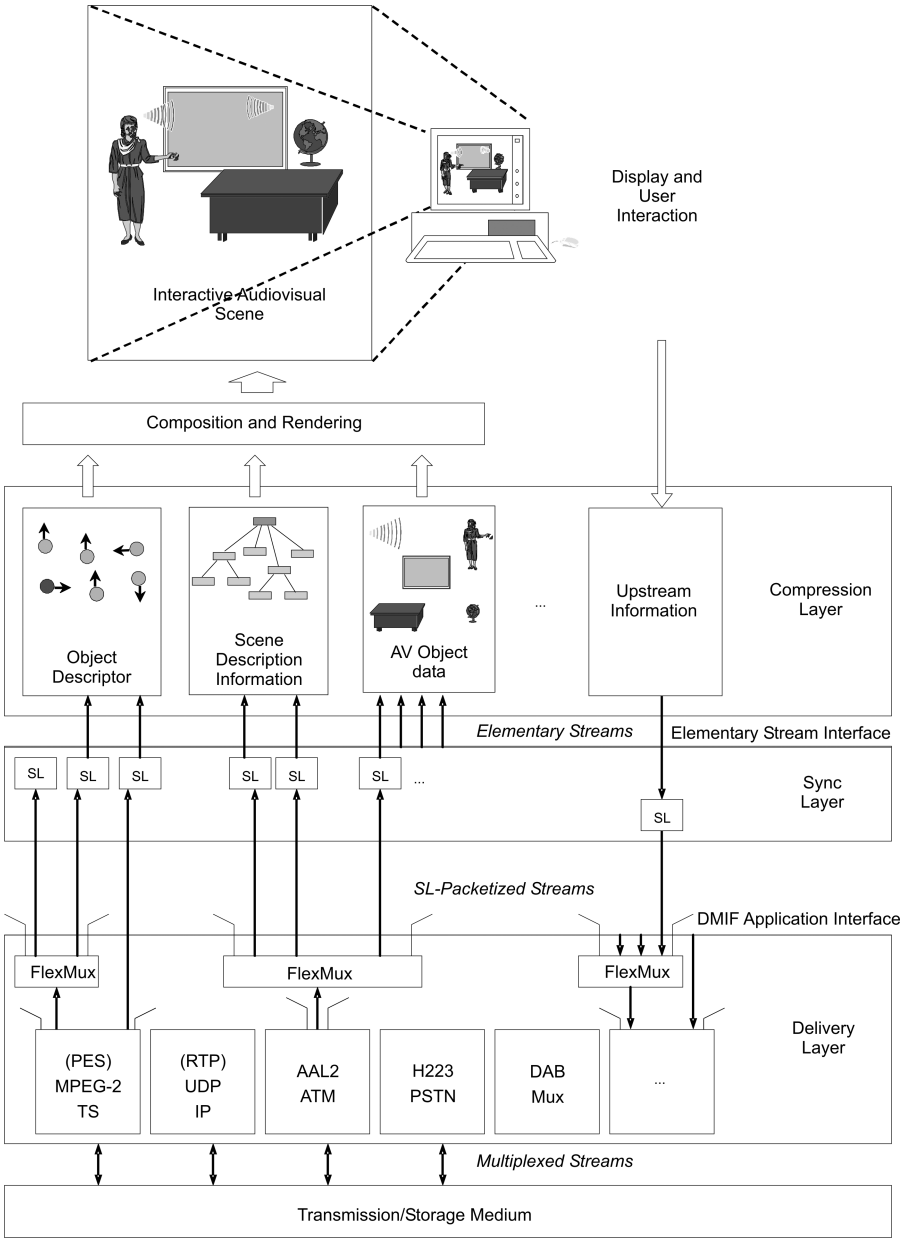

Abbildung 2.13: MPEG-4 Client Terminal [ISO01a]

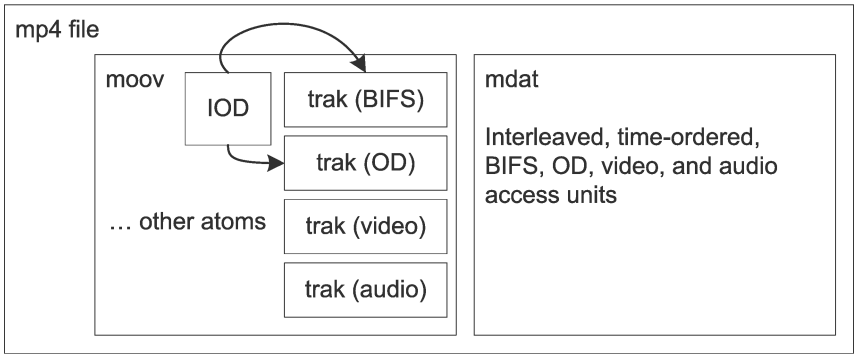

Abbildung 2.14: Struktur einer MP4-Datei [ISO02b]

MP4-Dateien werden mit Encodern aus einer textuellen Szenenbeschreibung und den Medienobjekten erzeugt[9]. Für die Wiedergabe müssen alle Bestandteile wieder vollständig rekonstruiert werden. Abbildung 2.14 zeigt den Aufbau einer MP4-Datei mit integrierten Medienobjekten (*self-contained MP4*). Um eine Datei per Streaming übertragen zu können, sind protokollunabhängige *Hint Tracks* erforderlich. Diese enthalten protokollabhängige Hinweise für den Streaming-Server bezüglich Formatierung und Rahmung der Mediendaten bei der Übertragung. Eine MP4-Datei kann mehrere Streaming-Hint-Tracks für verschiedene Protokolle enthalten. Dies ermöglicht ein Streaming der Datei auf verschiedenen Plattformen. Ein erneutes Aufbereiten der Mediendaten ist dann auch bei Verwendung verschiedener Streaming-Protokolle nicht mehr notwendig.

2.5 Anwendungen und Projekte

Die objektbasierte Beschreibung audiovisueller Inhalte erweitert die Möglichkeiten herkömmlicher Multimedia-Anwendungen erheblich und ist Grundlage vieler neuer Anwendungen und Dienste im AV-Bereich.

Ein herausragendes Einsatzgebiet sind interaktive Anwendungen in verschiedenen Ausprägungen. Auf Basis von 3D-Szenen können dem Nutzer beispielsweise mehrere Perspektiven oder eine freie Perspektivwahl angeboten werden. Interaktive virtuelle Welten können mit Medienobjekten natürlichen Ursprungs kombiniert werden und erhalten so eine neue Qualität. Auf diese Weise können Personen in Form von beliebig geformten Videoobjekten oder 3D-Videoobjekten in eine synthetische Umgebung integriert werden. Ein Beispiel hierfür ist das in Unterabschnitt 3.6.1 vorgestellte Szenario „Interaktiver Messebesuch – iFair“. Die objektbasierte Beschreibung der visuellen und auditiven Elemente einer interaktiven Anwendung ermöglicht vollkommen neue multimediale Erlebnisse. Die so erreichbaren korrespondierenden Eindrücke von visueller und auditiver Wahrnehmung führen zu einer neuartig hohen Immersion. Ein Beispiel für eine solche Anwendung ist die in Unterabschnitt 3.6.2 vorgestellte „Interaktive Musikdarbeitung – Ilmenoke“.

Die verschiedenartigen Verfahren zur Realisierung von Skalierbarkeiten unterstützen die häufig angestrebte Plattformunabhängigkeit von AV-Anwendungen. Einmal produzierte Inhalte sind so auf vielen Plattformen verfügbar, z. B. in geringer Qualität auf mobilen Endgeräten, in mittlerer Qualität in Internet-Applikationen und in hoher Qualität auf einem Fernsehgerät. Auch eine Personalisierung von Inhalten ist realisierbar, indem einzelne Objekte nach individuellen Kriterien ausgewählt und gesteuert werden. So lassen sich beispielsweise die Sprache, die Art der Darstellung oder auch eingebettete Werbung durch individuelle Vorgaben steuern. Objektbasierte AV-Anwendungen erschließen außerdem neue Wiedergabemöglichkeiten für Bild und Ton. Eine 3D-Szene kann beispielsweise ohne erneute Codierung dreidimensional dargestellt werden, z. B. durch stereoskopische Projektion. Dasselbe gilt für die Audiowiedergabe. Neue Wiedergabesysteme wie z. B. die Wellenfeldsysnthese (siehe Abschnitt 6.5) bieten auf Basis objektbasiert beschriebener Inhalte völlig neue Klangerlebnisse.

Ein Anwendungsgebiet mit einem besonders hohen Potenzial ist das interaktive Fernsehen, das im nächsten Abschnitt näher vorgestellt wird.

[9] Dies gilt nur, wenn die Anwendung eine Szene enthält. Einfache Audio- und Videoanwendungen benötigen diese nicht zwingend.

Abbildung 2.15: Zusatzinformationen im Digitalfernsehen

Abbildung 2.16: Interaktive Anwendungen mit Rückkanal

2.5.1 Interaktives Fernsehen

Mit der fortschreitenden Digitalisierung des Rundfunks ergeben sich zahlreiche neue Nutzungsmöglichkeiten für das Medium *Fernsehen*. Neben den ohnehin in den digitalen Signalen enthaltenen Informationen, wie z. B. Sendername und aktuelle Sendung, wurden zunächst verschiedene Zusatzinformationen in die Programme integriert. Beispiele hierfür sind aktuelle Nachrichten und Wetterberichte (siehe Abbildung 2.15). Es folgten einfache Spiele für einen oder mehrere Spieler. Diese Zusatzangebote basieren auf lokaler Interaktivität und benötigen daher keinen Rückkanal.

Die nächste Stufe der Interaktivität erfordert einen individuellen Interaktionskanal, um Informationen vom Nutzer zum Programmanbieter zu übertragen. Verbreitet sind Modem- und ISDN-Verbindungen mit direkter Einwahl beim Anbieter, aber auch Verbindungen über das Internet werden genutzt. Interaktive Spielshows ermöglichen es den Zuschauern, aktiv an einer Rateshow mitzuwirken oder ihre eigene Spielfigur mit der Fernbedienung auf ein Antwortfeld ihrer Wahl zu setzen. Ein anderes Beispiel sind interaktive Magazinsendungen. Über eine Live-Abstimmung können die Zuschauer aktiv in eine Diskussion einbezogen werden. Beispiele sind in Abbildung 2.16 zu finden. Eine Hersteller- und Anbieter-übergreifende Technologie für die Realisierung solcher Anwendungen ist die *Multimedia Home Platform (MHP)*[10].

[10] http://www.mhp.org/

Trotz der Verwendung digitaler Technik wird das eigentliche Fernsehprogramm in gewohnter Weise als zweidimensionales Videobild mit zugehörigem Ton zum Nutzer übertragen. Wenn man bedenkt, dass z. B. ein erheblicher Teil der Nachrichten- und Magazinsendungen seit vielen Jahren mit Hilfe der virtuellen Studiotechnik produziert wird, erscheint dies als wenig sinnvoll. Im Studio liegen virtuelle Sets, Grafiken, Akteure usw. getrennt als Objekte vor. Anschließend werden diese irreversibel zu einem zweidimensionalen Video kombiniert. Viel interessanter wäre es, diese Komposition der einzelnen Elemente erst beim Nutzer zu vollziehen. Die Entwicklung einer interaktiven Fernsehanwendung auf Basis des Objekt- und Szenenkonzeptes von MPEG-4 wird im Unterabschnitt 3.6.3 beschrieben.

2.5.2 Forschungsprojekt SAMBITS

Im Jahr 2000 startete das Forschungsprojekt „System for Advanced Multimedia Broadcast and IT Services (SAMBITS)“ unter der Leitung des *Institutes für Rundfunktechnik (IRT)* mit Partnern aus Industrie und Rundfunk [Sam], [Sam00a], [Sam00c], [Sam00b], [SSI02]. Das Ziel war die Entwicklung einer Systemarchitektur für die Übertragung interaktiver Dienste auf Basis von MPEG-4 und MPEG-7 über Rundfunk und Internet. Das Projekt gliederte sich in vier Aufgabengebiete:

- Rundfunk- und Internetdienste
- Studio- und Serversysteme
- Endgeräte
- Vorschläge zur Standardisierung

Es entstand eine Systemarchitektur für die Produktion, Übertragung und Nutzung integrierter Rundfunk- und Internetdienste sowie mehrere Beispielanwendungen. Das entwickelte System ermöglicht die Übertragung von synchronisierbaren MPEG-4-Inhalten über DVB parallel zu einem Hauptprogramm auf einem hohen Qualitätsniveau. Die Übertragung kann per Streaming oder über ein Objekt-Karussell erfolgen. Der Nutzer kann entscheiden, welche Inhalte er nutzen möchte. Alle Inhalte sind mit Metadaten auf Basis von MPEG-7 versehen. Zur Übertragung werden verschiedene Verfahren unter Nutzung des MPEG-2-Transportstromes verwendet. Abbildung 2.17 gibt einen Überblick über das entwickelte System. Die Endgerätearchitektur basiert auf PC-Technik. Sie besteht aus einem erweiterten DVB-MHP-Terminal mit integrierten Playern für die verschiedenen Inhalte. Implementiert wurde ein eigener *Storage Manager (SM)*, der über *DVB-PES*, *DVB-Carousel*, *DVB private sections*, *DVB IP sections* und Internet über das MHP-API auf Inhalte zugreifen und diese an die zuständigen Applikationen weiterleiten und speichern kann. Weiterhin wurde eine Benutzerschnittstelle entwickelt, die sich an den Gegebenheiten einer Fernsehanwendung (z. B. Bedienbarkeit über Fernbedienung und begrenzte Auflösung) orientiert. Personalisierte Einstellungen werden MPEG-7-konform gespeichert. Für die MPEG-4-Wiedergabe kommen jeweils separate Player für 2D-Szenen und 3D-Szenen zum Einsatz. Die Wiedergabeapplikationen werden von der MHP gesteuert. Weiterhin ist eine MPEG-7-Engine und ein HTML-Browser implementiert worden. Das System wurde 2001 auf der IBC präsentiert [Sam].

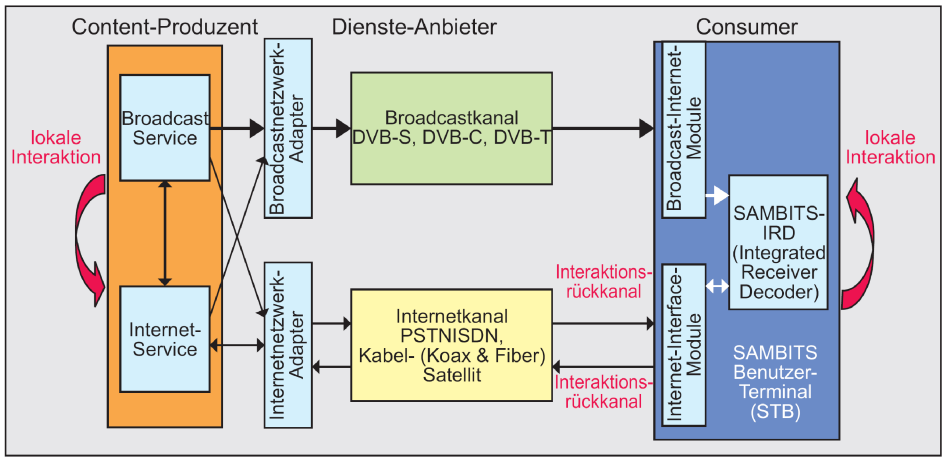

Abbildung 2.17: SAMBITS-System [SSI02]

2.5.3 Forschungsprojekt IAVAS

Das im Herbst des Jahres 2004 abgeschlossene Forschungsprojekt „Interaktive audiovisuelle Anwendungssysteme (IAVAS)“ am *Institut für Medientechnik (IMT)* der TU Ilmenau beschäftigte sich mit der Realisierung der vollständigen digitalen Medienkette auf Basis von MPEG-4 [IAV]. Das Ziel war die Evaluierung und Weiterentwicklung der Möglichkeiten des Objekt- und Szenenkonzeptes von MPEG-4 sowie die Entwicklung von Abläufen und Werkzeugen für die Produktion objektbasierter AV-Anwendungen. Im Mittelpunkt der Arbeiten standen die Erstellung und die Nutzung interaktiver dreidimensionaler Anwendungen mit synthetischen und natürlichen Elementen. Die Encodierung und Übertragung stand hingegen nicht im Fokus [DKR02], [KDRR02].

Neben den visuellen waren auch die umfangreichen auditiven Möglichkeiten von MPEG-4 Gegenstand der Arbeiten. Ein Schwerpunkt war die Untersuchung und Entwicklung neuer Arten von Medienobjekten. So wurden beispielsweise beliebig geformte Videoobjekte, 3D-Videoobjekte und Audioobjekte in interaktive 3D-Anwendungen integriert. Auf der Wiedergabeseite wurden natürliche und technische Effekte realisiert, z. B. Schatten und akustische Verdeckung. Letztere sind Teil der raumakustischen Simulation des virtuellen Raumes. Die entwickelten Anwendungen bieten neue interaktive Nutzungsmöglichkeiten und eine hohe Immersion.

Im Rahmen des Forschungsprojektes wurden mehrere interaktive Beispielszenarien zur Demonstration der Möglichkeiten von MPEG-4 entwickelt. In Abschnitt 3.6 werden drei dieser Anwendungen und deren Erstellung erläutert. Unter dem Namen *i3d* entstand ein leistungsfähiger MPEG-4-Player für die Wiedergabe von interaktiven 2D- und 3D-Szenen mit natürlichen und synthetischen audiovisuellen Inhalten. Einige Ergebnisse dieses Projektes fließen in die Arbeit der MPEG ein. Die Forschungen auf dem Gebiet der audiovisuellen Anwendungen werden u. a. im Rahmen des von der EU-geförderten Projektes „3DTV – Integrated Three-Dimensional Television – Capture, Transmission, and Display“[11] fortgeführt [OSS04]. Die vorliegende Dissertation basiert auf Arbeiten im Rahmen des Forschungsprojektes IAVAS.

[11] https://www.3dtv-research.org/

2.6 Zusammenfassung

Objektbasierte AV-Anwendungen sind eine neue Klasse multimedialer Anwendungen. Sie vereinen die erweiterten Möglichkeiten der digitalen Medien mit den Übertragungsmöglichkeiten der audiovisuellen Medien auf einem definierbaren Qualitätsniveau. Die objektbasierte Produktion, Distribution und Nutzung multimedialer Anwendungen wird die Grundlage vieler neuer Anwendungen und Dienste sein. Die Beschreibung der Inhalte erfolgt weitgehend unabhängig von Distributionsformen und Endgeräten. Dies ist ein wichtiger Schritt in Richtung Medienkonvergenz auf technischer Ebene. Die objektbasierte Beschreibung audiovisueller Anwendungen erhöht deren Komplexität im Vergleich zu klassischen AV-Anwendungen dramatisch. Es kommen Methoden und Verfahren aus vielen verschiedenen Bereichen zur Anwendung, z. B. Computergrafik, Multimedia, Kommunikationstechnik und Rundfunktechnik. Mit MPEG-4 liegt ein Standard für die umfassende Umsetzung des Objekt- und Szenenkonzeptes vor, dessen Potenzial zurzeit nicht annähernd genutzt wird. Die vorgestellten Projekte geben Beispiele für aktuelle und künftige Anwendungen und belegen die Aktualität des Themas.

3 Erstellung objektbasierter AV-Anwendungen

Dieses Kapitel widmet sich den Abläufen bei der Produktion objektbasierter AV-Anwendungen und den dafür erforderlichen Werkzeugen. Zunächst werden wichtige Begriffe erläutert und das Gebiet des Authorings abgegrenzt. Es werden beispielhaft einige Autorensysteme und Forschungsprojekte vorgestellt und bewertet. Abschließend werden die Grundlagen des verteilten Authorings vorgestellt.

3.1 Digitale Medienkette

Die *digitale Medienkette* stellt den Ablauf der Erstellung, Distribution und Nutzung von Multimedia-Anwendungen dar. Abbildung 2.2 zeigt diesen Ablauf für eine objektbasierte AV-Anwendung. Eine oft angewendete Möglichkeit zur Klassifizierung von Anwendungen innerhalb der Medienkette ist die Analogie zur Nahrungskette, wie sie in Abbildung 3.1 dargestellt ist.

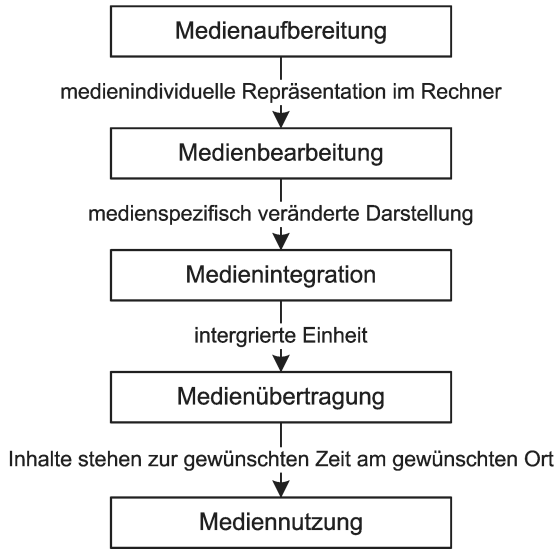

Abbildung 3.1: Multimediale Anwendung als „Nahrungskette“ [Ste00]

Die multimedialen Daten müssen eine Reihe von Transformationen durchlaufen, in denen bestimmte Arten von Anwendungen eingesetzt werden [Ste00]:

1. *Medienaufbereitung:* Im ersten Schritt werden alle Inhalte in eine digitale Repräsentationsform überführt. Bei Elementen natürlichen Ursprungs geschieht dies durch Aufnahme mit Kameras oder Mikrofonen. Elemente synthetischen Ursprungs müssen modelliert werden. Neben der Aufbereitung der Elemente selbst können je nach Anwendungsfall weitere Eigenschaften erfasst werden, z. B. Bewegungen oder Orientierung eines Objektes.

2. *Medienbearbeitung:* Die nun digital vorliegenden Inhalte können separat mit verschiedenen Werkzeugen bearbeitet und erweitert werden.

3. *Medienintegration:* Die unterschiedlichen Elemente werden zueinander in Beziehung gesetzt. Es entsteht eine multimediale Anwendung. Dieser Schritt wird oft auch als *Authoring* bezeichnet. Die ausführenden Personen sind die Autoren. Es kommen Autorenwerkzeuge und -systeme verschiedener Ausprägung zum Einsatz.

4. *Medienübertragung:* Die Anwendung wird dem Benutzer bereitgestellt. Es kommen die vielfältigen digitalen Distributionsformen zur Anwendung, z. B. DVD, DVB und Internet (siehe Unterabschnitt 2.1.4).

5. *Mediennutzung:* Der Nutzer kann die meist interaktiven Anwendungen seinen Wünschen entsprechend anpassen und nutzen.

Der Schritt *Medienintegration* ist Schwerpunkt des Authorings und damit auch der Aufgaben eines Autorensystems. Aus inhaltlicher Sicht kann die Entwicklung einer multimedialen Anwendung als Design-Prozess nach Abbildung 3.2 dargestellt werden [Ste00]. Dieser Ablauf erfolgt nicht zwingend streng chronologisch. Ein Autorensystem sollte den gesamten Prozess mit geeigneten Werkzeugen und Hilfsmitteln unterstützen.

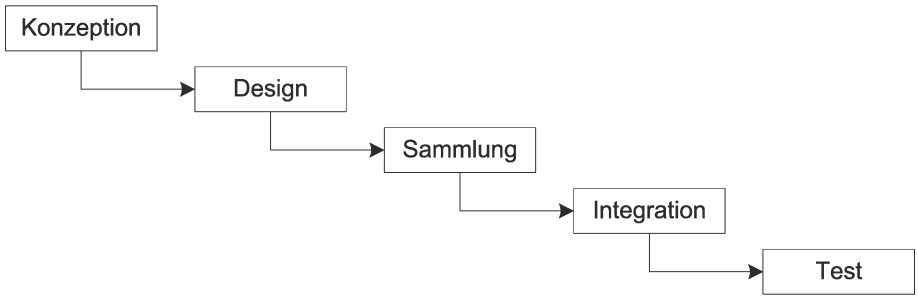

Abbildung 3.2: Phasen des Design-Prozesses einer multimedialen Anwendung [Ste00]

3.2 Produktion von Medienobjekten

Die Produktion von Medienobjekten hat die Bereitstellung von Elementen für audiovisuelle Anwendungen zum Ziel. Natürliche Objekte sind digitale Abbilder ihrer natürlichen Vorlagen und werden mit den bekannten Methoden (Kamera- bzw. Mikrofonaufnahme) erzeugt. Natürliche visuelle Objekte liegen meist in Form von Pixelbildern vor; natürliche auditive Objekte liegen als digitalisierte Wellenformen vor. Je nach Anforderung werden diese digitalen Abbilder in geeigneter Weise weiterverarbeitet und encodiert. Probleme können hinsichtlich der Skalierbarkeit auftreten. Bei komplexen natürlichen Objekten kann der Aufwand bei deren Produktion erheblich sein. Ein Objekt aus beliebig geformtem Video (siehe Unterabschnitt 2.2.2) wird beispielsweise durch eine Kameraaufnahme im Studio mit anschließender Extraktion der Form mit Hilfe des Chroma-Key-Verfahrens erstellt. Um hierbei gute Ergebnisse zu erzielen sind sehr gute und klar definierte Beleuchtungsverhältnisse erforderlich. Weitaus aufwändiger ist die Produktion von 3D-Videoobjekten (siehe Unterabschnitt 2.2.3). Hier werden gleichzeitig viele Ansichten eines Objektes aufgenommen. Vor der Gewinnung der geometrischen Eigenschaften erfolgen zahlreiche Schritte der Vorverarbeitung. Diese Verfahren werden auch innerhalb der MPEG entwickelt und sind Inhalt einiger aktueller Forschungsprojekte (z. B. [3DT]).

Synthetische Objekte werden nicht aufgenommen, sondern mit Hilfe eines Computers generiert. Sie liegen als Beschreibung in Form von Parametern bzw. einer textuellen Objektbeschreibung vor. Auch synthetische Objekte können auf verschiedene Arten encodiert werden. Für das Authoring sind oft uncodierte Beschreibungen von Vorteil. Hierdurch wird eine Skalierbarkeit erreicht und es kann zu jeder Zeit Einfluss auf die Gestalt und die Eigenschaften der Objekte genommen werden.

Neben der Gestaltung künstlicher Objekte gewinnt die Synthese von natürlichen Vorlagen zunehmend an Bedeutung. Hiermit verschwimmt die Abgrenzung zwischen natürlichen und synthetischen Objekten. Ein Ziel dieser Entwicklungen ist oft die extrem effiziente Codierung von Informationen. Dabei werden Einbußen hinsichtlich der Natürlichkeit bzw. der Qualität in Kauf genommen. Ein weiteres Ziel ist die Bereitstellung von Interaktionsmöglichkeiten für synthetisierte Objekte. Beispiele hierfür sind die bekannten *Avatare*.

3.3 Authoring

3.3.1 Aufgaben

Für den Prozess des Zusammenfügens aller Elemente zu einer multimedialen Anwendung hat sich der Begriff *Authoring* etabliert, für den es keine adäquate deutsche Bezeichnung gibt. Nach [EHV93], [FK02] besteht dieser Prozess aus folgenden Schritten:

1. Formulierung und Konzeption eines Drehbuches
2. Sammeln von Medienobjekten (Texte, Audio, Video usw.)
3. Entwurf der Informationsstrukturen und Interaktionstechniken
4. Realisierung der Anwendung

Die Generierung von Medienobjekten gehört demnach nicht zum Authoring. Diese Trennung wird in der Praxis jedoch nicht streng vorgenommen. Viele Autorensysteme enthalten auch Werkzeuge für die Erstellung und Bearbeitung von Medienobjekten. In sehr einfachen Fällen (z. B. beim DVD-Authoring) besteht die Aufgabe des Authorings lediglich darin, den Ablauf einer Präsentation festzulegen, gegebenenfalls Menüs für die Navigation einzufügen und die Daten für die Speicherung auf dem Datenträger vorzubereiten. Auch für solche Anwendungen gibt es spezielle Autorenwerkzeuge. Diese stehen nicht im Fokus dieser Arbeit.

Im Falle objektbasierter AV-Anwendungen ist die Aufgabe des Authorings komplex, da prinzipiell alle in Erscheinung tretenden Inhalte durch separate Objekte repräsentiert sein können. Die resultierenden Szenenbeschreibungen werden dementsprechend sehr umfangreich. Das Authoring objektbasierter AV-Anwendungen auf Basis des Objekt- und Szenenkonzeptes umfasst die folgenden Schritte:

- Integration der Medienobjekte
- logische Strukturierung
- räumliche (*spatiale*) Komposition
- zeitliche (*temporale*) Komposition

- Interaktionsdesign
- Integration von Metadaten
- Vorbereitung zur Distribution
- Encodierung

Die Mehrzahl dieser Schritte hat die Bearbeitung der Szenenbeschreibung zum Inhalt. Die Erstellung dieser stellt somit die zentrale Aufgabe des Authorings dar. Zunächst sollen die Aufgaben der beteiligten Personen bestimmt werden:

- Der *Direktor* ist für die Leitung und Organisation einer Produktion zuständig. Er legt die Abläufe fest, verteilt die anstehenden Aufgaben und überwacht die Ergebnisse.
- *Autoren* sind für Gestaltung, Methodik, Didaktik und Inhalte von multimedialen Anwendungen zuständig. Sie sollen ihre Vorstellungen im Rahmen des Authoring-Prozesses möglichst intuitiv umsetzen können. Autoren können auf einzelne Aufgaben spezialisiert sein, z. B. auf die Bearbeitung von Animationen oder Interaktionen.
- *Designer, Programmierer und Produzenten* von Medienobjekten können als Medienzulieferer tätig werden oder eine Anwendung optimieren. Sie sind nicht mit dem Autor gleichzusetzen, auch wenn meist keine klaren Grenzen gezogen werden können [FK02].
- *Nutzer* sind die Konsumenten der angebotenen Inhalte. Im Gegensatz zu klassischen AV-Anwendungen kann der Nutzer durch Interaktionsmöglichkeiten aktiv in das Geschehen einbezogen werden.

3.3.2 Manuelle Erstellung

Die Erstellung objektbasierter AV-Anwendungen auf Basis von MPEG-4 ohne die Nutzung grafischer Autorenwerkzeuge läuft wie folgt ab:

1. *Produktion der Medienobjekte:* Natürliche Objekte werden mit den bekannten Methoden aufgenommen, z. B. in einem Studio. Die entstehenden Bild-, Audio- und Videoaufnahmen werden bearbeitet und ggf. entsprechend ihrer Art und vorgesehenen Verwendung encodiert und sie liegen dann in binärer Form vor. Synthetische Objekte werden am Computer z. B. mit den Werkzeugen der Textverarbeitung, Computergrafik, Musikproduktion usw. erstellt und ggf. entsprechend ihrer Art encodiert. Sie liegen dann als inhaltliche bzw. parametrische Beschreibungen vor.
2. *Erstellung der Szene mit speziellen Werkzeugen:* Die manuelle Erstellung einer Szene in einem Text-Editor ist nur in sehr einfachen Fällen praktikabel. Komplexe Szenen werden mit Werkzeugen der Computergrafik (z. B. *Maya* oder *3D Studio MAX*) erstellt und liegen dann in proprietären Formaten vor.
3. *Überführung der Szene in ein textuelles Format:* Für die weitere manuelle Bearbeitung muss die Szene in ein textuelles Format überführt werden. Die meisten 3D-Grafikwerkzeuge sehen hierfür einen Export nach VRML vor. Denkbar ist auch ein Export nach X3D bzw. die Konvertierung von VRML nach X3D.

4. *manuelle Bearbeitung der Szene in einem Text-Editor:* Die nun vorliegende textuelle (noch nicht MPEG-4-konforme) Szenenbeschreibung muss auf Quelltext-Ebene bearbeitet werden, um sie später encodieren zu können. Zunächst müssen einige formale Änderungen vorgenommen werden. Anschließend werden weitere Elemente eingefügt. Folgende Schritte sind notwendig:
   - Einfügen des *root*-Knotens (*Group* oder *OrderdGroup*)
   - Einfügen des IOD
   - Einfügen weiterer Medienobjekte
   - Einfügen der OD
   - Einfügen von Metadaten
   - Erstellen von Animationen
   - Erstellen von Interaktionen

   Die Bearbeitung kann auf Basis von VRML/BIFS-Text oder X3D/XMT-A vorgenommen werden. Nach Vervollständigung der Szene erfolgt ggf. eine manuelle Optimierung auf Quelltext-Ebene. Nun liegt eine encodierbare textuelle MPEG-4-Szenenbeschreibung vor.

5. *Encodieren der Szenenbeschreibung:* Nun werden ggf. noch Encodier- und Distributions-Hinweise eingefügt. Die Szenenbeschreibung kann zusammen mit den Medienobjekten encodiert werden. Das Ergebnis ist eine distributionsfähige MPEG-4-Anwendung.

Insbesondere Schritt 4 erfordert tief greifendes Verständnis der Technologie von MPEG-4. Dies ist von Autoren in der Regel nicht zu erwarten. Folglich sollten diese Bearbeitungen durch intuitiv nutzbare grafisch-interaktive Autorenwerkzeuge unterstützt werden.

3.4 Autorensysteme

3.4.1 Begriffe

Die Begriffe *Autorenwerkzeug* und *Autorensystem* werden im Sprachgebrauch und in der Literatur in sehr verschiedenen Zusammenhängen verwendet. Eine klare Definition ist erst bei Einschränkung des Anwendungsgebietes möglich. Folgende Definition ist in [Ste00] zu finden:

> „Autorensysteme oder -werkzeuge sind grafisch-interaktive Tools, die den technischen Entwicklungsprozess multimedialer Anwendungen mit Hilfe von Konzepten der visuellen Programmierung unterstützen. Sie ermöglichen dabei insbesondere eine grafisch-interaktive Spezifikation der Beziehungen zwischen Medienobjekten.“

Autorenwerkzeuge für multimediale Anwendungen sind eine eigenständige Produktkategorie von Software-Werkzeugen. Ein solches Werkzeug soll die Autoren bei der Entwicklung multimedialer Anwendungen unterstützen und die zugrunde liegenden technischen Zusammenhänge weitgehend verbergen. Prinzipiell lassen sich multimediale Anwendungen auch mit herkömmlichen Programmiersprachen realisieren. Naturgemäß kann diese

Aufgabe nur von erfahrenen Programmierern erledigt werden, die meist nicht über die speziellen Kenntnisse eines Autors verfügen. Daher erscheint dieser Weg oft als wenig sinnvoll [BS98].

Weiterhin wurde der Begriff *Autorensystem* geprägt. Nach [Ste00] ist ein Autorensystem eine in eine Autorenumgebung eingebettete Menge von Software-Werkzeugen (Autorenwerkzeugen). Eine klare Unterscheidung dieser Begriffe wird in der Praxis meist nicht vorgenommen bzw. werden die Begriffe synonym verwendet. Im Folgenden wird der Begriff *Autorensystem* für das gesamte System von Werkzeugen und Hilfsmitteln einschließlich der erforderlichen Infrastruktur (z. B. Server und Netzwerk) verwendet. *Autorenwerkzeuge* hingegen sind die einzelnen Werkzeuge selbst.

3.4.2 Entwicklung

Zu Beginn des Multimedia-Zeitalters war die Entwicklung von Multimedia-Anwendungen noch extrem aufwändig und nur von wenigen Experten mit umfangreichem Spezialwissen zu bewältigen. Die Anwendungen wurden ähnlich wie andere Software-Produkte unter Nutzung spezieller Bibliotheken, Funktionen usw. programmiert. Im nächsten Schritt waren grafisch-interaktive Autorenwerkzeuge verfügbar, die es ausgebildeten Multimedia-Autoren erlaubten, ihre inhaltlichen und gestalterischen Ideen umzusetzen, ohne tief in die technischen Zusammenhänge einsteigen zu müssen. Diese Werkzeuge kamen zunächst in spezialisierten Multimedia-Agenturen zum Einsatz. Der Funktionsumfang nahm immer größere Ausmaße an, sodass umfangreiche Schulungen der Autoren erforderlich geworden sind. Parallel dazu wurden in den letzten Jahren aber auch Autorenwerkzeuge für jedermann angeboten, die eine Erstellung multimedialer Anwendungen nahezu ohne spezielle Vorkenntnisse ermöglichen. Neben der zunächst angestrebten Zielgruppe der Heimanwender kommen diese Werkzeuge zunehmend auch in Firmen zum Einsatz. Prinzipiell lassen sich nun einfache Anwendungen von jedermann erstellen. Diese Entwicklung war auch in anderen Bereichen zu beobachten, beispielsweise bei der Textverarbeitung, der Videobearbeitung oder der Entwicklung von Internetpräsenzen. Die Bereitstellung von Autorenwerkzeugen erfolgt in drei Stufen:

1. manuelle Erstellung mit Werkzeugen der klassischen Softwareentwicklung
2. Werkzeuge für Spezialisten
3. Werkzeuge für jedermann

Die einzelnen Stufen sind nicht streng voneinander getrennt und existieren auch parallel. Die Stufe 1 ist bei der Entwicklung und Einführung neuer Technologien immer notwendig und ist die Grundlage für grafisch-interaktive Werkzeuge. Anschließend erfolgt die Entwicklung von Werkzeugen für professionelle Anwender. Aus diesen Erfahrungen heraus werden schließlich einfacher zu nutzende und weniger komplexe Werkzeuge entwickelt. Die Stufen 2 und 3 existieren dann nebeneinander. Oft werden spezielle Versionen professioneller Werkzeuge mit reduziertem Leistungsumfang für Privatanwender auf den Markt gebracht. Im Falle der Entwicklung objektbasierter AV-Anwendungen ist heute der Übergang von Stufe 1 zu Stufe 2 erreicht. Für die Nutzung der umfangreichen Möglichkeiten von MPEG-4 ist weiterhin eine manuelle Bearbeitung der Szenendaten notwendig. Erste Autorenwerkzeuge für abgegrenzte Einsatzgebiete sind verfügbar. Auf Grund der nahen Verwandtschaft zu klassischen Multimedia-Anwendungen profitiert die Entwicklung von

Autorenwerkzeugen stark von den Erfahrungen aus diesem Bereich. Das Zusammengehen der Medienanwendungen wird sich in Zukunft auch auf die Autorenwerkzeuge auswirken. Multimedia-Autorensysteme werden Schritt für Schritt um Möglichkeiten objektbasierter AV-Anwendungen erweitert.

3.4.3 Ansätze

Autorenwerkzeuge basieren meist auf Metaphern, um die teilweise sehr komplexen technischen Zusammenhänge innerhalb einer multimedialen Anwendung übersichtlich darzustellen. Im Folgenden werden die Grundformen von Autorenwerkzeugen vorgestellt [BS98].

Zeitachsen-basierte Autorenwerkzeuge

Dieser Ansatz hat seine Wurzeln im Bereich des digitalen Videoschnittes. Der Autor ordnet Medienobjekte und Aktionen auf einer Zeitachse (Timeline) an, sodass diese leicht und genau synchronisiert werden können und den zeitlichen Ablauf der Anwendung widerspiegeln. Dieser Ansatz ist besonders gut für zeitbezogene Anwendungen mit geringem Interaktionsgrad geeignet, bietet jedoch kaum Unterstützung für logische Strukturen und Interaktionen [FK02], [EHV93]. Dies hat zur Folge, dass die Arbeit bei komplexen Anwendungen und steigendem Interaktionsgrad unübersichtlich wird, weil ein Großteil der Veränderungen in der Szene durch Sprünge auf der Zeitachse repräsentiert wird. Man versucht diesen Nachteil durch Verwendung von speziellen Skriptsprachen zu kompensieren. Abbildung 3.3 zeigt die Darstellung einer Zeitachse in einem kommerziellen Autorenwerkzeug.

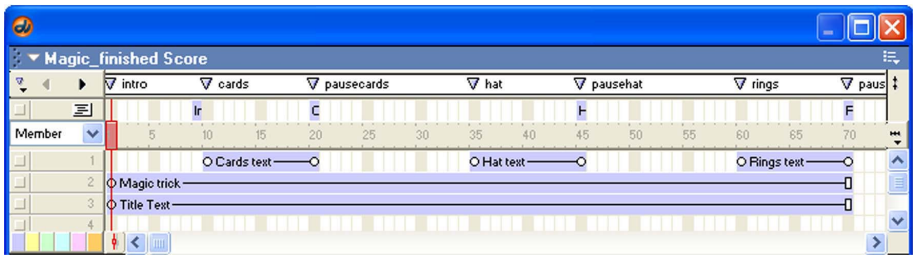

Abbildung 3.3: Zeitachse aus *Macromedia Director 9.0*

Flussdiagramm-basierte Autorenwerkzeuge

Bei diesem Ansatz werden Medienobjekte sowie Kontroll- und Interaktionselemente durch Ikonen repräsentiert, die mit Hilfe von Kanten zu logischen Strukturen verbunden werden. Es entstehen Flussdiagramme, die den möglichen Ablauf einer Anwendung widerspiegeln. Ikonen können für komplette Animationen, Texte oder Kommandos stehen. Bei einfachen Anwendungen ist diese Methode sehr übersichtlich. Der Autor muss lediglich die Flussdiagramme verstehen. Bei komplexen Anwendungen können die Diagramme jedoch sehr unübersichtlich werden. Das Ikonen-basierte Autorenwerkzeug *Macromedia Authorware* bietet daher beispielsweise die Möglichkeit, Teildiagramme als komplexe Medienobjekte in externe Dateien auszulagern. Abbildung 3.4 zeigt korrespondierende Flussdiagramme in einem kommerziellen Autorenwerkzeug.

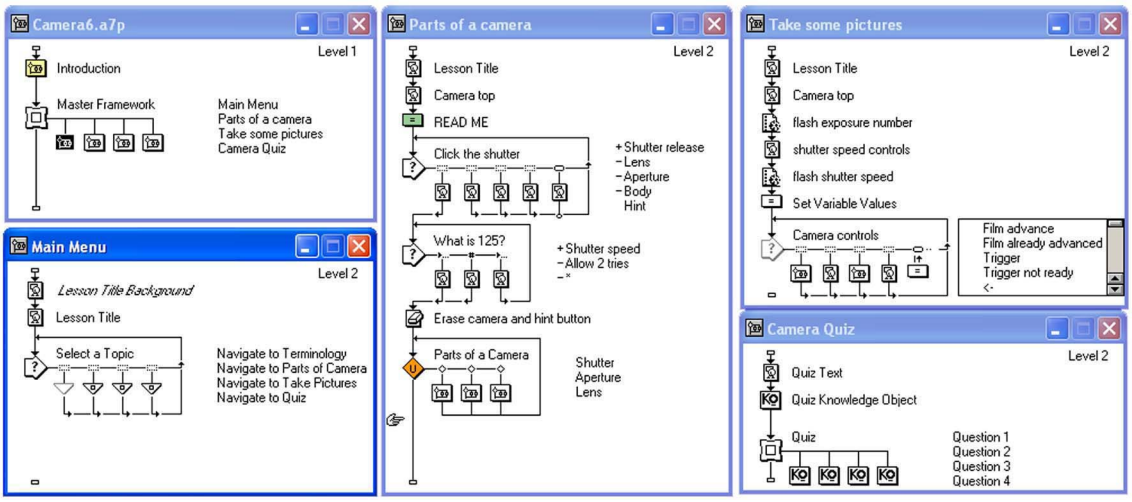

Abbildung 3.4: Korrespondierende Flussdiagramme in *Macromedia Authorware 7.01*

Seiten-basierte Autorenwerkzeuge

Bei Seiten-basierten Autorenwerkzeugen werden (interaktive) Objekte auf Flächen bzw. Seiten oder Karten platziert. Eine Szene besteht aus mehreren Karten, die untereinander verknüpft sind und über Navigationsinteraktionen angesteuert werden können. Diese Herangehensweise ist sehr intuitiv und kommt dem Schreiben eines Buches nahe. In Anlehnung an die Hypertext-Struktur können komplexe Beziehungen zwischen den Seiten hergestellt werden. Ein Seitenwechsel erfolgt meist durch eine Interaktion des Nutzers. Ein verbreitetes Autorenwerkzeug dieser Art ist *click2learn Toolbook*. Auch hier werden komplexere Anwendungen unübersichtlich. Die Seiten-Metapher bietet sich vor allem bei überwiegend statischen Inhalten an. Auch Animations-Editoren werden teilweise angeboten, doch erreichen diese meist nicht die Möglichkeiten Zeitachsen-basierter Autorenwerkzeuge. Abhilfe kann hier die Integration externer Animationen schaffen [FK02], [Bol98].

Skriptsprachen-Ansatz

Weiterhin ist der Skriptsprachen-Ansatz zu finden, der nicht zu den grafisch-interaktiven Bearbeitungen gehört. Der Autor schreibt kleine Skripte für die Steuerung von Interaktionssequenzen. Die verwendeten Skriptsprachen sind meist für einen bestimmten Verwendungszweck optimiert und dadurch leichter verständlich als herkömmliche Programmiersprachen. Dieser Ansatz bietet keine grafische Unterstützung und erfordert genaue Kenntnis der Skriptsprache, zeichnet sich aber durch eine hohe Flexibilität aus [Ste00], [EHV93]. Der Skriptsprachen-Ansatz ist in keinem der betrachteten Autorenwerkzeuge vorherrschend. Er ist aber dennoch häufig elementarer Bestandteil von Autorenwerkzeugen und ergänzt hier andere Ansätze. Ein Beispiel hierfür ist die Skriptsprache *Lingo* der Firma *Macromedia*. Sie ist Bestandteil des Autorensystems *Director*.

Gegenüberstellung

Die vorgestellten Ansätze für die Realisierung von Autorenwerkzeugen haben auf Grund ihrer Eigenschaften jeweils ihre Vor- und Nachteile. In einem universell einsetzbaren Autorenwerkzeug sollten verschiedene Ansätze in Kombination verwendet werden. Tabelle 3.1 stellt die Vor- und Nachteile der Ansätze für Autorenwerkzeuge gegenüber.

Ansatz	Vorteile	Nachteile
Zeitachse	• eingängige Metapher • einfache zeitliche Anordnung und Synchronisation	• erheblicher Aufwand beim Erstellen komplexer Anwendungen • benötigt zusätzliches Interaktionskonzept (z. B. Skripte)
Flussdiagramm	• übersichtlich • einheitliche Metapher	• Verständnis von Flussdiagrammen notwendig • eingeschränkte Flexibilität
Seite	• einfach und intuitiv	• Probleme bei Animationen und Synchronisation von AV-Medien • schlechter Überblick bei komplexen Anwendungen • nicht für 3D-Anwendungen geeignet
(Skript)	• sehr flexibel	• Programmierkenntnisse erforderlich • nicht grafisch-interaktiv

Tabelle 3.1: Vor- und Nachteile der Authoring-Ansätze

3.5 Produkte und Projekte

Es sind verschiedene Autorenwerkzeuge für die Erstellung objektbasierter AV-Anwendungen auf Basis von MPEG-4 verfügbar. Auch einige aktuelle Forschungsprojekte beschäftigen sich mit diesem Thema. Im Folgenden werden exemplarisch drei kommerzielle Produkte und einige interessante Forschungsprojekte vorgestellt.

3.5.1 Kommerzielle Produkte

iVAST Studio Author

Das Produkt *iVAST Studio Author* der Firma *DG2L Technologies*[1] ist ein Seiten-basiertes Autorenwerkzeug für die Erstellung zweidimensionaler Anwendungen, in welches auch Zeitachsen integriert sind. Es handelt sich um eine Modifikation des proprietären Autorenwerkzeuges *iShell 3* der Firma *Tribeworks*[2]. Grundlage einer Anwendung bildet das *Projektfenster*, in dem globale Merkmale (z. B. Größe der Anwendung) definiert werden und in dem die integrierten Seiten verlinkt sind. Die Seiten enthalten den eigentlichen Inhalt und stellen Unterszenen dar, die miteinander verknüpft werden können. Die Seiten können in separaten Dateien encodiert werden. Somit wird eine Verteilung der Anwendung möglich. Repräsentation jeder Seite ist ein *Dokumentenfenster*, das aus einem Bereich für Struktur und Interaktionen sowie einem Bereich mit Zeitachse besteht. Jedem Objekt und jeder Hierarchieebene kann eine eigene Zeitachse (z. B. für Keyframe-Animationen) zugewiesen werden, wobei immer nur die Zeitachse des ausgewählten Objektes sichtbar

[1] http://www.dg2l.com/
[2] http://www.tribeworks.com/

ist. Gerade bei zweidimensionalen Anwendungen hat dies den Nachteil, dass der Autor zeitliche Relationen zu anderen Objekten nicht intuitiv erfassen kann. Die hierarchische Struktur der Unterszene wird in Struktogramm-ähnlicher Form dargestellt, wobei jedem Objekt Eigenschaften, Interaktionen und Kommandos hinzugefügt werden können. Anwendungen können in eine Textdatei exportiert oder in eine MP4-Datei encodiert werden. Die Textdatei entspricht weitgehend dem textuellen MPEG-4-Format BT.

Envivio Broadcast Studio

Das Produkt *Envivio Broadcast Studio* (neue Version: *Envivio 4mation*) der Firma *Envivio*[3] ist ein Zeitachsen-basiertes Autorenwerkzeug. Der Arbeitsablauf wird mit Hilfe von Karteikarten sehr übersichtlich aufgeteilt. Die erste Karteikarte (*Manage Project*) ermöglicht die Zuweisung von allgemeinen Metadaten (Name des Autors, Erstellungsdatum usw.). Die zweite Karteikarte (*Compose*) besteht unter anderem aus einer Liste mit eingebundenen Medien, einer Bühne, der Zeitachse und Objekt-Editoren. Im Bereich *Actions* können einfache Interaktionen mit Hilfe von Pulldown-Menüs intuitiv festgelegt werden. Für die Bearbeitung komplexer Abläufe wird ein Skript-Editor angeboten. Diese Zweiteilung ermöglicht die Erstellung von Interaktionen sowohl für Anfänger als auch für Fortgeschrittene in angepasster Umgebung. Die dritte Karteikarte (*Encode/ Prepare Streaming*) beinhaltet eine grafische Benutzeroberfläche für den integrierten Encoder. Die Einstellmöglichkeiten für die Gesamtszene und die einzelnen Medienobjekte sind sehr umfangreich. Die letzte Karteikarte (*Make MP4 File*) bietet schließlich die Möglichkeit, die Szene im MP4-Format zu speichern. Eine textuelle Ausgabe von Szenendaten ist nicht vorgesehen.

Mindego MPEG Application Authoring Tool

Mit dem *MPEG application authoring tool* hat die Firma *Mindego*[4] bereits Ende 2003 ein OpenSource-Plugin für die Entwicklungsumgebung *Eclipse*[5] angekündigt. Es handelt sich hierbei nicht um ein Autorenwerkzeug im eigentlichen Sinne, da es nicht über eine grafisch-interaktive Oberfläche verfügt. Im Gegensatz zu anderen Entwicklungen steht für die Szenenbeschreibung nicht BIFS im Vordergrund, sondern MPEG-J, das Java-API für MPEG-4. MPEG-J fungiert dabei als Basis von Anwendungen und steuert verschiedene Medienobjekte (u. a. BIFS-Beschreibungen). Dies hat den Vorteil, dass die Verwendung von BIFS nicht Voraussetzung, sondern Möglichkeit wird. Andere, möglicherweise effizientere Objektbeschreibungen, wie sie in den *SNHC Tools* des MPEG-4-Standards definiert sind, können ebenfalls verwendet werden [PE02].

3.5.2 MPEG-4 Toolbox

MPEG-4 Toolbox ist ein grafisches Autorenwerkzeug des *Center for Research and Technology Hellas*[6] in Griechenland für die Erstellung dreidimensionaler MPEG-4-Anwendungen. Die Entwicklung erfolgte in C/C++ unter Verwendung der MPEG-4 Referenzsoftware IM1 [DKRS04]. *MPEG-4 Toolbox* setzt einige spezielle Konzepte von MPEG-4 praktisch um. Neben der Erzeugung einfacher, objektbasierter 3D-Szenen sind dies die Verwendung von *BIFS-Commands* sowie die Integration von *Gesichtsanimation* und beliebig geformtem Video. Die Gesichtsanimation erfolgt über *Facial Animation Parameter (FAP)*, die

[3] http://www.envivio.com/
[4] http://www.mindego.com/
[5] http://www.eclipse.org/
[6] http://www.certh.gr/

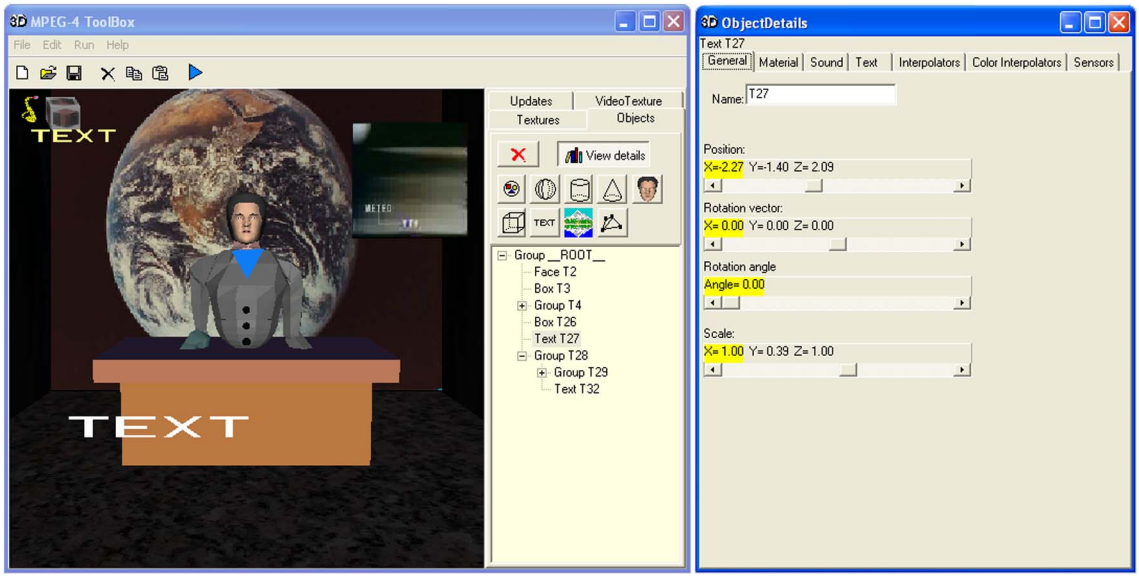

Abbildung 3.5: Benutzeroberfläche von *MPEG-4 Toolbox* [DKRS04]

zuvor festgelegte Bereiche an einem synthetischen Kopfmodell dynamisch ändern [PF02]. *MPEG-4 Toolbox* enthält weiterhin ein Werkzeug für die halbautomatische Segmentierung von Videoelementen. Abbildung 3.5 zeigt die grafische Benutzeroberfläche von MPEG-4-Toolbox, die aus drei größeren Bereichen besteht. Die Bühne bietet eine statische Sicht auf die Szene. Sie ist passiv, d. h. direkte Manipulation ist nicht möglich. Auf der rechten Seite ist u. a. ein Drag&Drop-fähiger Szenenbaum untergebracht. Im Bereich *Object Details* können schließlich die Eigenschaften eines ausgewählten Objektes (z. B. Layout und Animation) festgelegt werden. Dieser Bereich ist nicht konsistent gehalten; es können auch nicht zum Objekt gehörende Attribute verändert werden.

Die Implementierung des Werkzeuges realisiert einen stark Technik-zentrierten Ansatz. Es gibt keine aktive Unterstützung des Autors durch das Werkzeug und es werden keine Authoring-Metaphern verwendet. Die *Low-Level*-Konstrukte des MPEG-4-Szenengraphen werden durch einfache grafische Elemente repräsentiert. Das Werkzeug zeigt die Möglichkeit der Erstellung einer dreidimensionalen Szene mit verschiedensten MPEG-4-Objekten mit Hilfe einer grafischen Benutzeroberfläche. Für Encoder, Multiplexer und Player greift *MPEG-4 Toolbox* auf die Werkzeuge *BifsEnc*, *MP4Enc* und *Player3D* zurück, die aus der MPEG-4-Referenzimplementierung hervorgegangenen sind [ISO04c].

3.5.3 ETRI Interactive Contents Authoring System

Das *Radio & Broadcasting Laboratory* des südkoreanischen *Electronics and Telecommunications Research Institut ETRI*[7] beschäftigt sich unter anderem mit der Entwicklung interaktiver Anwendungen auf Basis von MPEG-4 für das digitale Fernsehen. Hier soll ein unter dem Namen „Interactive Broadcasting Contents Authoring System“ entwickeltes Autorensystem vorgestellt werden [KCKK01], [KLK02], [JK02], [JKK02]. Zielanwendungen des Autorensystems sind interaktive Fernsehanwendungen, z. B. interaktive Wetterberichte. Es werden 2D-Anwendungen generiert, die typische Objekte enthalten, z. B. Audio, rechteckiges und geformtes Video, Standbild, Text und Grafik. Das Authoring erfolgt

[7] http://www.etri.re.kr/

auf Basis von XMT-Ω bzw. XMT-A. Zusatzinformationen werden im Metadatenformat MPEG-7 integriert. Als Ergebnis entstehen encodierte MPEG-4-Anwendungen. Abbildung 3.6 zeigt die Architektur des modular aufgebauten Autorensystems. Die wichtigsten Teile sind:

1. Nutzerschnittstelle (*User Interface*)
2. Verarbeitung (*Information Editor and Processor*)
3. Codec-Bibliothek (*Media Library*).

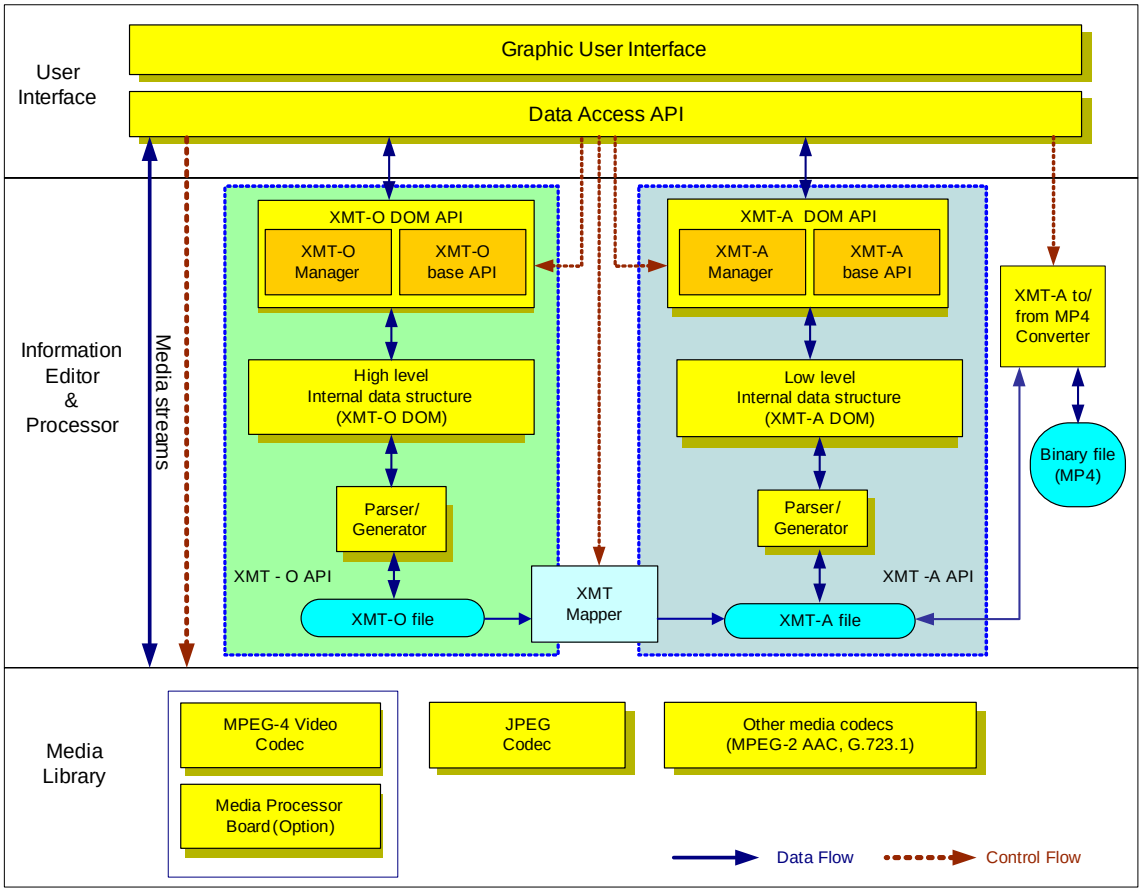

Abbildung 3.6: Architektur des Autorensystems [JKK02]

Benutzerschnittstelle

Die Benutzerschnittstelle besteht aus einer leistungsfähigen grafischen Benutzeroberfläche und einem *Data Access API*. Die Benutzeroberfläche beinhaltet die üblichen Elemente eines grafisch-interaktiven Autorenwerkzeuges und dient der einfachen und komfortablen Erstellung interaktiver Anwendungen. Sie erlaubt die räumliche und zeitliche Komposition zahlreicher Objekttypen zu einer MPEG-4-Szene und die Integration von Interaktionen und Verweisen. Für die Unterstützung des Autors werden vordefinierte, einfach anzupassende Templates angeboten. Für komplexe Bearbeitungen der Szene ist eine direkte Manipulation des Szenengraphen möglich. Außerdem ist ein Tool für die halbautomatische Segmentierung von Videomaterial integriert, um die Verwendung von beliebig geformtem

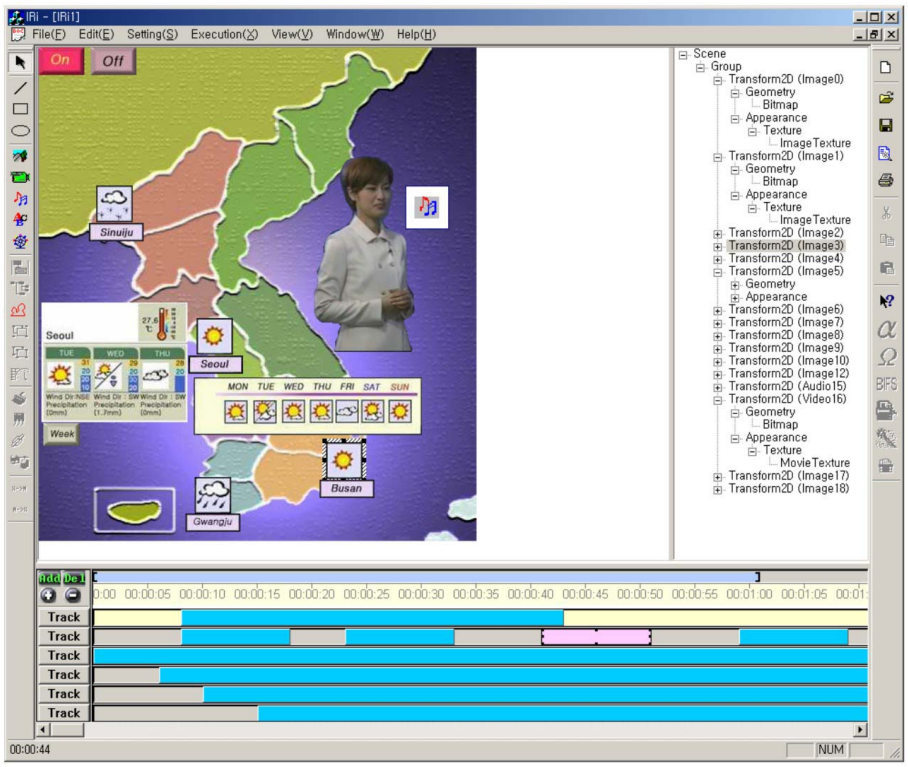

Abbildung 3.7: Benutzeroberfläche des Autorensystems [KCKK01]

Video zu erleichtern. Abbildung 3.7 zeigt die Benutzeroberfläche des Autorensystems bei der Bearbeitung eines interaktiven Wetterberichtes. Die Szene enthält als Hintergrund eine Landkarte mit davor angeordneten Wettersymbolen. Eine Sprecherin ist als geformtes Videoobjekt integriert. Am rechten Rand ist eine Baumdarstellung des Szenengraphen angeordnet. Im unteren Bereich ist ein Zeitachsen-Editor zu sehen.

Die *Data Access APIs (DA APIs)* bieten Schnittstellen für den einfachen Zugriff auf die gesamte Funktionalität des Autorensystems. Sie enthalten beispielsweise die Schnittstelle zur internen Datenhaltung von Szenenbeschreibung, Objektbeschreibung, Routing-Informationen und MPEG-4-Encodierhinweisen (*Encoding Hints*).

Verarbeitung

In diesem Teil des Autorensystems findet die gesamte Verarbeitung der Daten statt. Es stehen verschiedene Werkzeuge (Parser, Mapper und Konverter) für die Erzeugung, Bearbeitung und Konvertierung von XMT-Beschreibungen auf Basis des *Complete2D scene graph profiles* zur Verfügung. Die Implementierung erfolgte in C++ auf Basis des XML-Parsers *Xerces*.

Abbildung 3.8 zeigt die Komponenten und ihr Zusammenwirken. Im Mittelpunkt steht ein DOM der XMT-Ω- bzw. XMT-A-Szene. Der *XMT-A/Ω Manager* ist die Schnittstelle zwischen *Data Access API* und *XMT-A/Ω base API*. Das *XMT-A/Ω base API* besteht aus XMT-A- und XMT-Ω-Elementklassen, welche Funktionen für die Manipulation von Attributen korrespondierender Elemente beinhalten.

Der in Abbildung 3.9 dargestellte *XMT-Ω to A Mapper* übersetzt Szenen von XMT-Ω nach XMT-A. Dafür parst der *XMT-Ω Input Processor* die Eingangsdaten; das *Converter Module* traversiert das XMT-Ω-DOM und ruft die korrespondierenden Funktionen des

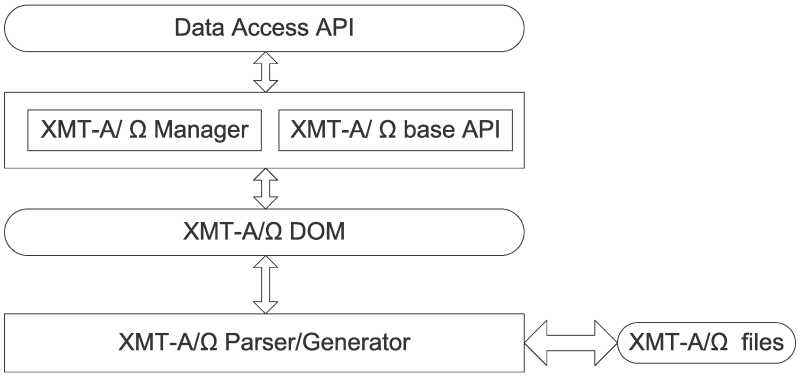

Abbildung 3.8: XML-Werkzeuge [JKK02]

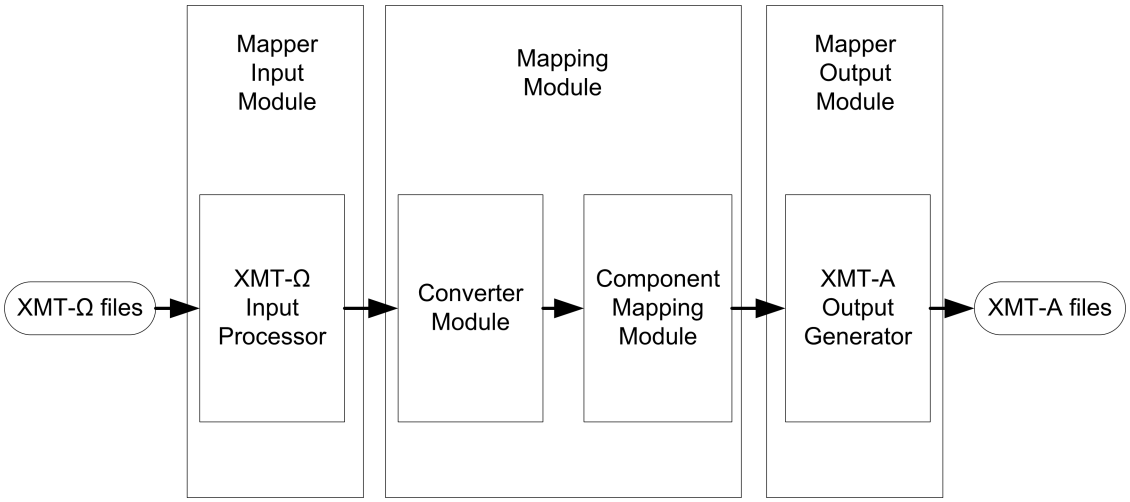

Abbildung 3.9: XMT-Ω to A Mapper [JKK02]

Component Mapping Module auf. Dieses definiert Regeln für die Übersetzung in das Format XMT-A. Der *XMT-A Output Generator* generiert eine XMT-A-Datei aus dem DOM. Die Anbindung der Benutzerschnittstelle erfolgt über das *XMT-A/Ω base API*.

Media Library

Die *Media Library* besteht aus einer Sammlung von Encodern und Decodern für verschiedene Arten von audiovisuellen Objekten, die allen Komponenten des Autorensystems zur Verfügung stehen.

Fazit

Das vorgestellte Autorensystem erlaubt die grafisch-interaktive Entwicklung von zweidimensionalen interaktiven AV-Anwendungen auf Basis von MPEG-4. Die Komponenten bilden eine solide Architektur für die Bearbeitung von Szenenbeschreibungen auf Basis von XMT. Funktionen für ein verteiltes Authoring sind nicht vorhanden und ein Zugriff auf einen Authoring-Server ist ebenfalls nicht vorgesehen.

3.5.4 IBM Toolkit for MPEG-4

Die *Composite Media Group* am *T.J.Watson Research Center*[8] der Firma IBM widmet sich seit langem der Entwicklung und Standardisierung interaktiver Anwendungen. Wichtige Beiträge aus diesem Hause sind beispielsweise die hier maßgeblich vorangetriebene Entwicklung von SMIL und XMT. Im Folgenden wird das Projekt *IBM Toolkit for MPEG-4* und das darauf aufbauende Autorenwerkzeug *IBM MPEG-4 XMT Editor* vorgestellt [IBMb], [IBMa].

IBM Toolkit for MPEG-4

Das *IBM Toolkit for MPEG-4* [IBMb] ist eine Sammlung von Java-Klassen und APIs für die Erstellung, Verarbeitung und Wiedergabe von MPEG-Anwendungen. Zusätzlich werden systemabhängige Bibliotheken für verschiedene Zielplattformen angeboten (Windows, Linux, OS/2 und iPAQ). Zur Demonstration der Leistungsfähigkeit werden einige Beispielapplikationen bereitgestellt, die im Folgenden kurz vorgestellt werden:

AVgen ist ein einfaches grafisches Werkzeug für die Erstellung von AV-Inhalten für ISMA- oder 3GPP-kompatible Geräte. Hiermit können Audio- und Videoinhalte erzeugt und bei Bedarf mit *Hint-Informationen* versehen werden, wie sie für die Übertragung über RTP/RTSP oder zur Nutzung auf ISMA- und 3GPP-kompatiblen Geräten benötigt werden. *XMTBatch* ist ein Werkzeug für die Generierung von binären MPEG-4-Dateien aus Szenenbeschreibungen im XMT-Format. Es können MP4-Dateien für verschiedene Verwendungszwecke generiert werden, z. B. für die Übertragung via RTP/RTSP oder HTTP. Die Überführung von MP4 nach XMT-A ist ebenfalls möglich. *M4Play* ist eine kompakte Applikation für die Wiedergabe audiovisueller Inhalte. Sie unterstützt verschiedene Audio- und Videoformate, z. B. Divx, Xvid, MP3 und AAC Low-Complexity Profile Audio (.aac, .adif, .adts). M4Play ist ISMA-kompatibel. *M4Applet for ISMA* ist ein Java-Applet für die Wiedergabe von ISMA-kompatiblen Inhalten über RTP/RTSP. *M4Applet for HTTP* ist ein Java Applet für die Wiedergabe von MPEG-4-Inhalten über HTTP. Das Applet nutzt das standardisierte Interleave-Format *M4Mux* für den progressiven Download während der Wiedergabe.

Weiterhin ist das *IBM Toolkit for MPEG-4 SDK* erhältlich. Durch die Verwendung von Java sind alle Applikationen plattformunabhängig. Ein Beispiel für die Integration des Authoring-Kernes in fremde Applikationen ist die Implementierung von *AVgen* als Plugin für die Entwicklungsplattform *Eclipse*[9]. Die hier vorgestellten Applikationen sind zu Testzwecken frei erhältlich. Für die Entwicklung eigener Anwendungen kann das Toolkit bei IBM lizenziert werden.

IBM MPEG-4 XMT Editor

Der *IBM MPEG-4 XMT Editor (XMTEdit)* [IBMa] ist ein auf dem *IBM MPEG-4 Toolkit* basierendes grafisches Autorenwerkzeug für die Erstellung von zweidimensionalen MPEG-4-Anwendungen. XMTEdit verwendet das Szenenbeschreibungsformat XMT-Ω. Die Inhalte sind visuell oder direkt im XMT-Quelltext (*XMT tree view*) editierbar und können in XMT gespeichert oder in eine binäre MPEG-4-Datei exportiert werden. Auch eine Voransicht in einem speziellen Fenster ist möglich. Der grafische Editor ermöglicht die räumliche und zeitliche Komposition von Szenen mit Hilfe bewährter Techniken. Das Werkzeug verwendet einen Top-Down-Ansatz und behandelt Medienobjekte als vordefi-

[8] http://www.watson.ibm.com/
[9] http://www.eclipse.org/

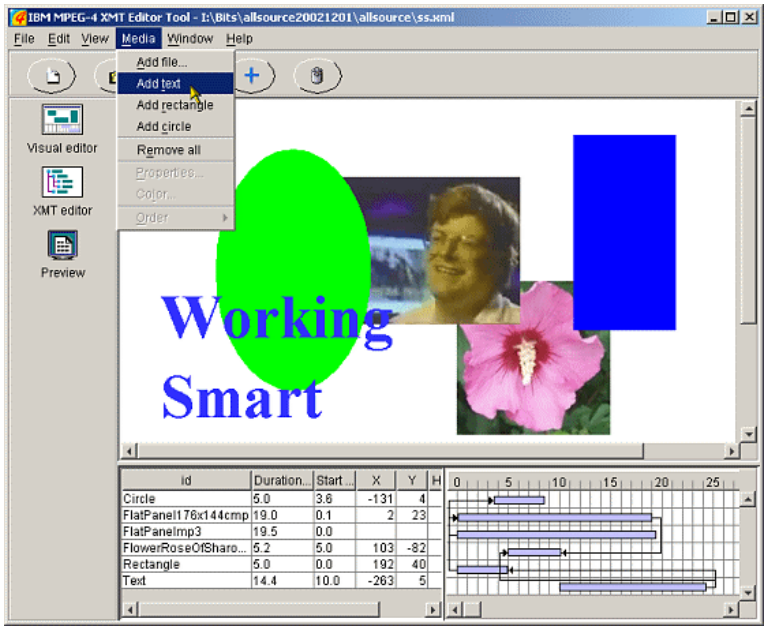

Abbildung 3.10: Benutzeroberfläche von *XMTEdit* [IBMa]

nierte Bausteine. Er bietet einen schnellen Einstieg in die Erstellung von MPEG-4-Anwendungen, ohne dass hierfür technisches Hintergrundwissen erforderlich ist. Abbildung 3.10 zeigt die grafische Benutzeroberfläche von XMTEdit. Es werden drei Sichten angeboten:

- *Spatial View:* Festlegen der räumlichen Anordnung der Objekte und der Interaktionsmöglichkeiten
- *Temporal View:* Festlegen der zeitlichen Abläufe und Übergangseffekte mit einem Zeitachsen-Editor
- *Details View:* Bearbeiten der Eigenschaften des ausgewählten Objektes

Eine weitere wichtige Komponente ist der eigentliche XMT-Editor für die Bearbeitung der Szenen auf Quelltext-Ebene. Alle Aktionen in den grafischen Editoren werden sofort in eine korrespondierende XMT-Ω-Beschreibung überführt. Der XMT-Editor stellt diese Beschreibung als editierbare Baumstruktur dar. Die Softwarearchitektur basiert auf einem vereinfachten *MVC-Konzept*[10] (bestehend aus *Model* und *View*) und nutzt zur internen Datenhaltung ein *DOM*.

Fazit

Das *IBM Toolkit for MPEG-4* stellt zahlreiche Komponenten für die Realisierung von MPEG-4-Systemen auf Basis von Java bereit und gibt praktische Beispiele für Anwendungen. Der Schwerpunkt lag bisher auf der Wiedergabeseite. Das Autorenwerkzeug zeigt die Funktionsweise eines ausschließlich auf XMT aufbauenden Authoring-Konzeptes für zweidimensionale Anwendungen. Die Komponenten des Toolkits können in eigene Entwicklungen einfließen.

[10] Model-View-Controller, siehe Unterabschnitt 4.5.3

3.5.5 Authoring 744

Unter dem Titel „Authoring 744: Writing Descriptions to Create Content“ wird in [MM03] ein sehr ungewöhnlicher Ansatz für die Erstellung von MPEG-4-Anwendungen vorgestellt. Mit Hilfe einer inhaltlichen Beschreibung der Szene auf Basis von MPEG-7 werden MPEG-4-Dateien erzeugt. Die Beschreibung erfolgt auf einer semantisch sehr hohen Abstraktionsebene. Das System besteht aus folgenden Komponenten:

- einem Texteditor (*MPEG-7 ScriptWriter*)
- einem Transcoder nach XMT-A
- einem MPEG-4-Encoder

Abbildung 3.11 zeigt die Software-Architektur von Authoring 744. Als Encoder wird das *mp4tool* von der ENST verwendet. Die vorgestellten Beispiele zeigen einfache multimediale Anwendungen mit Video, Text, Audio, Bildern und Grafik. Die Autoren arbeiten an einer grafischen Benutzeroberfläche für die Erzeugung der MPEG-7-Beschreibungen und an weiteren Modulen.

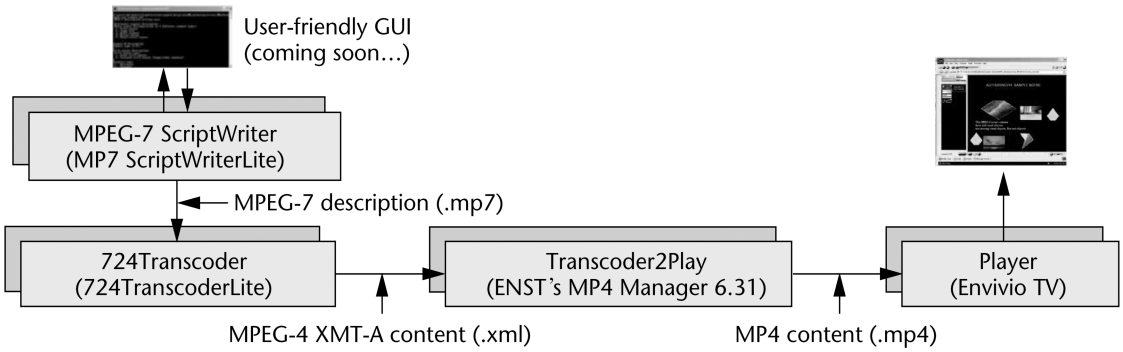

Abbildung 3.11: Software Architektur von *Authoring 744* [MM03]

3.5.6 Fazit

Es existieren inzwischen einige kommerzielle Autorenwerkzeuge für die Erstellung von objektbasierten AV-Anwendungen auf Basis von MPEG-4. Äußerlich unterscheiden sich diese kaum von Autorenwerkzeugen für klassische Multimedia-Anwendungen. So findet man bewährte Elemente, wie z. B. Zeitachse, Bühne, Objektbrowser usw. wieder, wodurch die Einstiegsschwelle für Autoren sehr niedrig ist. Es gelten aber auch die gleichen Einschränkungen, wie mangelnde Flexibilität und mangelnde Interoperabilität. Auch die Unterstützung mehrerer Autoren wurde bisher stark vernachlässigt. Der Schwerpunkt liegt derzeit auf Werkzeugen für die Erstellung interaktiver 2D-Anwendungen, beispielsweise für das interaktive Fernsehen oder für Web-Anwendungen. Zusammenfassend können folgende positive Eigenschaften der vorgestellten Lösungen genannt werden:

- in sich geschlossene Software-Umgebungen
- wiedererkennbare Benutzerschnittstellen und Arbeitsabläufe
- intuitiv nutzbare Elemente, z. B. Zeitachse, Objektbrowser usw.

- integrierte Werkzeuge für die Erstellung und Bearbeitung von Medienobjekten
- schnelles Prototyping möglich

Die Betrachtung zeigte aber auch einige prinzipbedingte Schwächen der untersuchten Autorenwerkzeuge auf:

- eingeschränkter Funktionsumfang (Profiles, MO usw.)
- proprietäre Datenformate
- eingeschränkte Kontrolle über die Inhalte, da meist kein direkter Zugriff auf die Szenendaten möglich ist
- eingeschränkte Austauschbarkeit der Inhalte zwischen Autoren und zwischen Werkzeugen verschiedener Hersteller
- mangelnde Unterstützung von Gruppenarbeit
- mangelnde Erweiterbarkeit
- häufig mangelnde Kompatibilität der erstellten Anwendungen

Die kommerziellen Autorensysteme konnten lediglich hinsichtlich ihres Verhaltens untersucht werden, da in der Regel keine Informationen über die interne Arbeitsweise zur Verfügung standen. Insgesamt konnte keines der untersuchten Autorenwerkzeuge vollständig überzeugen. Die erweiterten Möglichkeiten von MPEG-4, z. B. 3D-Anwendungen und neuartige Medienobjekte werden derzeit kaum genutzt. Lediglich das verwendete Distributionsformat weist auf MPEG-4 hin.

Bei den vorgestellten Forschungsprojekten hingegen ist MPEG-4 der Ausgangspunkt der Entwicklungen und das Objekt- und Szenenkonzept bildet die Grundlage der Architekturen. Der Funktionsumfang ist meist noch eingeschränkt und für den produktiven Einsatz nicht ausreichend. Dennoch lassen sich aus diesen Projekten zahlreiche Ansätze und Komponenten für künftige Autorensysteme ableiten.

3.6 Beispielszenarien

Das vorgestellte Objekt- und Szenenkonzept wurde seit der Verabschiedung des MPEG-4-Standards nur sehr zögerlich praktisch angewendet. Zur Determination und Weiterentwicklung des Konzeptes wurden am IMT im Rahmen des Forschungsprojektes IAVAS [IAV] mehrere Beispielszenarien entworfen und realisiert. Dabei kamen überwiegend Werkzeuge aus der MPEG-4-Community zum Einsatz. Die folgenden Abschnitte beschreiben einige dieser Szenarien, deren Bestandteile, deren Produktion und deren Codierung [DKR02].

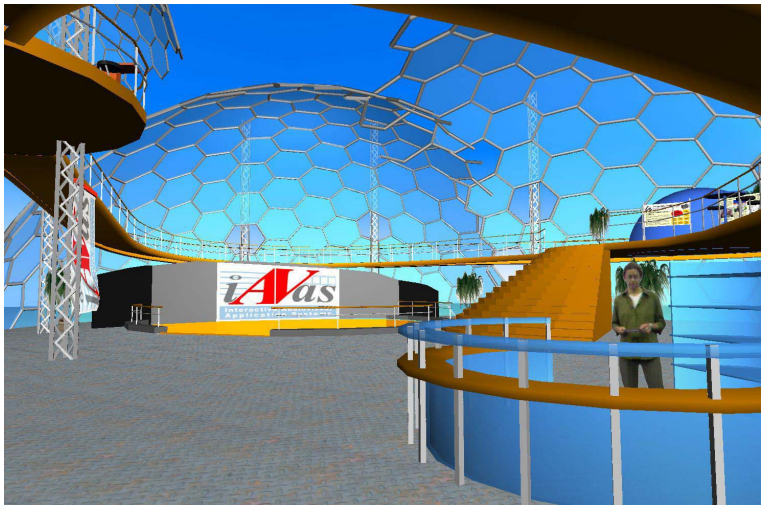

Abbildung 3.12: Virtuelle Messe mit Personal

3.6.1 Interaktiver Messebesuch – iFair

Unter dem Arbeitstitel „iFair“ wurde am IMT ein interaktiver Messebesuch auf Basis von MPEG-4 entwickelt. Das Ziel war die Schaffung einer interaktiven dreidimensionalen Anwendung mit synthetischen und natürlichen Medienobjekten in einer synthetischen Umgebung. Folgende Komponenten sind enthalten:

- Messegelände/Messehalle als dreidimensionales synthetisches Modell
- einzelne Messestände als synthetische Modelle
- Schautafeln/Poster als Bilder
- Standpersonal als (beliebig geformtes) Video inkl. Audio zzgl. örtlicher Beschreibung
- Vorführsaal o. ä. als synthetische Modelle
- Film im Vorführsaal als Video/Audio
- Personal an der Auskunft (Video, Audio, örtliche Beschreibung)
- geführte Rundgänge (vorbereitete Routen zur Auswahl)
- individuelle Erkundung (Navigation durch die 3D-Szene)
- Wunsch nach Erläuterungen zum Stand (mit Interaktionsschaltern zur Aktivierung)
- individuelle Fragen an das Standpersonal über Online-Verbindung (z. B. via Webcam) zum Personal des Unternehmens

Abbildung 3.12 zeigt eine Ansicht der Anwendung. Im Vordergrund ist ein Empfangstresen mit einer Person zu sehen, die als beliebig geformtes Videoobjekt integriert wurde. Für die Realisierung der einzelnen Komponenten kamen unterschiedlichste Hilfsmittel für die Produktion von audiovisuellen Inhalten zum Einsatz. Die synthetischen Komponenten (Messehalle, Stände, Vorführraum) wurden mit Hilfe von Werkzeugen der konventionellen Computergrafik/-animation modelliert. Da es bisher keine geeignete Software für die

direkte Modellierung von MPEG-4-Szenen/Objekten gibt, ist hierfür der Umweg über den Export von VRML-Szenenbeschreibungen gegangen worden. Die natürlichen Komponenten (z. B. Standpersonal) wurden in einem typischen Videostudio unter Verwendung der Blue-Screen-Technik aufgenommen. Der Chroma-Keying-Prozess kann mit Hilfe eines Bildmischers erfolgen, jedoch geschah dies hier in der Postproduktion. Dies hat den Vorteil, dass der Keying-Prozess wesentlich genauer parametrisiert werden kann und somit zu besseren Ergebnissen führt. Weiterhin ist vorteilhaft, dass die resultierenden Bildsequenzen dann bereits auf Dateiebene vorliegen. Die Erzeugung der Visual-ES erfolgte mit den entsprechenden Encodern, deren Nutzung jedoch Expertenwissen voraussetzt [Sch02]. Die Produktion der geformten Videoobjekte schränkte aber mitunter auch die gestalterische Freiheit ein. Ein Problem ergibt sich beispielsweise, wenn der Akteur sich in einem größeren Bereich bewegen soll. Wird hierfür der Kamerabildausschnitt groß gewählt, dann ist die Auflösung meist zu gering. Würde die Kamera bewegt (geschwenkt), müsste man die Bewegung erfassen, in eine Veränderung des Ortes umrechnen und in die Szenenbeschreibung codieren. Während der Aufnahmen im Videostudio wurde auch die Sprache der Akteure aufgenommen. Die Audioaufnahmen erfolgten akustisch trocken, da der spätere raumakustische Eindruck der modellierten Messehalle entsprechen sollte. Die Poster und Plakate an den Ständen wurden als Pixelbilder in die Szene integriert. Der im Vorführraum gezeigte Film lag als Videodatei im AVI-Format vor und wurde in einen MPEG-4 Video-ES konvertiert.

Die verschiedenen MO wurden mit den zugehörigen Knoten im Baum der Szenenbeschreibung verknüpft. Weiterhin war zu jedem Objekt ein OD anzulegen. Auch für diese Arbeitsschritte gab es zum Zeitpunkt der Produktion[11] noch kein Autorenwerkzeug, sodass dies manuell in einem Quelltext-Editor erfolgte und Expertenwissen voraussetzte. Die MO waren mit den geforderten Interaktionen zu verknüpfen (z. B. Touch Sensor), um den Beginn eines Vortrages durch Anklicken des Standpersonals zu erzielen. Letztlich waren markante Punkte in der Messehalle als Standpunkte zu definieren und geeignete Rundgänge als Routen zu erstellen. Die nun vorliegende, vollständige textuelle Szenenbeschreibung war unter Nutzung eines BIFS-Encoders in einen ES zu codieren. Alle ES (Szenenbeschreibung, Video, Audio, Bilder, OD) wurden im Multiplex-Verfahren zu einer MPEG-4-Datei encodiert. Auch für diese letzten beiden Schritte sind die Encoder manuell zu verwenden, sodass wiederum Expertenwissen benötigt wird [ISO01a]. Die vorliegende MPEG-4-Anwendung kann mit einem MPEG-4-Player, welcher alle erforderlichen Decoder aufweist (insbesondere gemäß *Complete Scene-Graph-Profile*), wiedergegeben werden.

3.6.2 Interaktive Musikdarbietung – Ilmenoke

Als ein weiteres Szenario wurde eine interaktive Musikdarbietung gewählt. Hier befinden sich die Musiker in einer virtuellen dreidimensionalen Umgebung (Musiksaal), durch die der Nutzer navigieren kann. Als ein ganz wesentliches Leistungsmerkmal eines solchen Szenarios ist die raumakustisch angepasste Audiowiedergabe hervorzuheben [Bau02], [DRSD03]. Während der Navigation passt sich die Audiowiedergabe dem Standort und der Blickrichtung an. Hierfür wurde die Metapher „Hören mit den Ohren der Kamera“ geprägt [RMS01]. Wie auch für das Szenario Messebesuch (siehe Unterabschnitt 3.6.1) können die erforderlichen Komponenten virtuell nachgebildet werden. Folgende Komponenten sind enthalten:

[11] Frühjahr 2003

- Konzertsaal/Musiksaal als dreidimensionales synthetisches Modell
- Bühne/Kulisse und Einrichtung als synthetische Modelle
- die einzelnen Musiker als (beliebig geformtes) Video inkl. Audio zzgl. örtlicher Beschreibung
- Poster u. ä. als Bilder
- geführte Rundgänge (vorbereitete Routen zur Auswahl)
- individuelle Erkundung (Navigation durch die 3D-Szene)

Diese virtuelle Musikdarbietung weist aber auch zusätzliche Leistungsmerkmale auf:

- nahezu freie Navigation durch den Musiksaal, auch auf der Bühne und in der Nähe der Musiker
- Durch die erzielbare Nähe zu einzelnen Musikern kann deren Spiel intensiv gehört werden.
- Auf Wunsch spielen die einzelnen Musiker Soli.

Für eine erste Version wurde eine fünfköpfige Musikgruppe des Genres Rock 'n' Roll gewählt (Gesang, Schlagzeug, E-Gitarre, E-Bass, Keyboard). Die Musiker wurden in Bild und Ton einzeln in einem Videostudio aufgenommen. Auch hier galt es den Ton akustisch trocken aufzunehmen. Die technischen Schritte für die Erstellung von Szene, Video und Audio entsprachen denen für das Szenarium „Messebesuch". Da es sich um eine Musikdarbietung handelt, ist die raumakustisch korrekte Wiedergabe des Tones wesentlich. Hierfür muss der Musiksaal auch mit seinen raumakustischen Eigenschaften modelliert werden [RMS01]. Die Szenenbeschreibung gemäß MPEG-4 unterstützt im *3D Audio Profile* die Codierung von akustischen Materialien. Die Audiowiedergabe ist z. B. in *5.1 Surround Mehrkanalton* möglich.

Das Szenarium *Musikdarbietung* weist eine neue Qualität einer interaktiven audiovisuellen Anwendung auf. Die Übereinstimmung von visuellem und auditivem Eindruck beim Nutzer während der Navigation durch die 3D-Szene bewirkt eine neuartig hohe Immersion. Künftig ist die Nutzung der Wellenfeldsynthese für die akustische Wiedergabe denkbar (siehe Abschnitt 6.5).

3.6.3 Interaktive Nachrichtensendung – iNews

Unter dem Arbeitstitel „iNews" wurde am IMT eine interaktive Nachrichtensendung auf Basis von MPEG-4 entwickelt [BHJ+04]. Das Ziel war die Untersuchung der Möglichkeiten von MPEG-4 für die Realisierung eines interaktiven Fernsehformates. Neben der Darstellung von klassischen Nachrichtensendungen sollte vor allem gezeigt werden, wie der Nutzwert einer solchen Sendung durch Personalisierung und Interaktionsmöglichkeiten gesteigert werden kann. Alle typischen Elemente einer Nachrichtensendung, z. B. Sprecher, Hintergründe, Grafiken und Einspielungen, werden entsprechend dem Objekt- und Szenenkonzept separat beschrieben, codiert und übertragen. Auch alle benötigten Bedienelemente sind Bestandteil der MPEG-4-Szenenbeschreibung (siehe Abbildung 3.13). Dieser

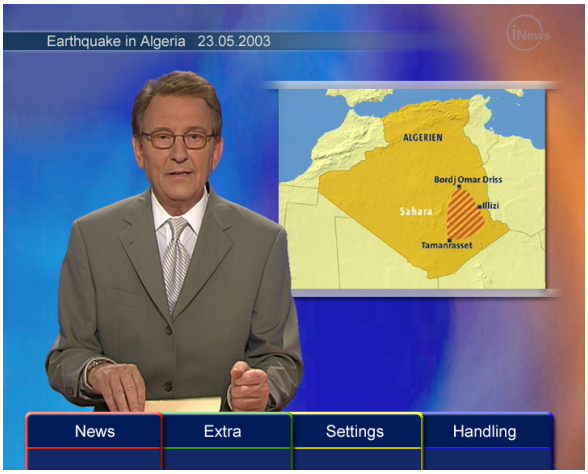

Abbildung 3.13: Nachrichtensendung mit Bedienelementen

Abbildung 3.14: Nachrichtensendung mit Gebärdendolmetscher

Ansatz bietet viele neue Nutzungsmöglichkeiten. Beispielsweise lässt sich der Nachrichtensprecher auf Wunsch durch einen Gebärdendolmetscher ersetzen (siehe Abbildung 3.14) bzw. kann ein solcher eingeblendet werden. Hintergrundelemente, z. B. Grafiken und Karten, können vergrößert oder gespeichert werden. Weiterhin wurde ein personalisierbarer Ticker integriert, der bezüglich seines Inhaltes, seines Aussehens und seines Verhaltens gesteuert werden kann. Es wurden Versuche mit zweidimensionalen und dreidimensionalen Szenenbeschreibungen unternommen – letztlich wurde das Projekt als 2D-Anwendung realisiert. Folgende Funktionalität ist enthalten:

- personalisierbare Zusammenstellung der Nachrichten
- auswählbare Sprecher (Sprache, Gebärdendolmetscher)
- Abruf von Hintergrundinformationen und Grafiken
- zuschaltbare Untertitel zum gesprochenen Nachrichtentext
- personalisierbarer, interaktiver Ticker
- Übersicht über die verfügbaren Beiträge mit Dauer und Kurzbeschreibung
- Navigation durch die Zusammenstellung
- Visualisierung des zeitlichen Ablaufes der Zusammenstellung
- Steuerung über Fernbedienung oder Tastatur
- kontextabhängige Hilfe zu den Einstellungsmöglichkeiten
- Navigationshilfe
- Audio-Buttons (gesprochener Text zum selektierten Oberflächenelement)
- Speicherung der persönlichen Einstellungen

Auch bei dieser Produktion mussten verschiedene Werkzeuge in Kombination verwendet werden. Das Ausgangsmaterial wurde von Sendeanstalten der ARD als MAZ bereitgestellt. Der NDR stellte drei komplette Sendungen der Tagesschau zur Verfügung. Der MDR lieferte zwei Sendungen des Thüringen-Journals sowie Aufnahmen eines Gebärdendolmetschers. Das Material wurde auf einen PC übertragen und dort weiterverarbeitet. Nach dem Chroma-Keying wurden die Sprecher als beliebig geformtes Video und als rechteckiges Video mit Alpha-Kanal codiert. Ergänzend wurden für einen Teil der Beiträge Tonspuren in englischer Sprache aufgenommen und codiert. Die Hintergrund-Grafiken wurden manuell nachempfunden. Es folgte die Aufbereitung der Texte für die Untertitel und den Ticker sowie die Sammlung von Zusatzinformationen. Unter Berücksichtigung der Bedingungen einer Fernsehanwendung wurde eine Benutzeroberfläche entworfen und mit Hilfe eines Demonstrators getestet.

Das Authoring erfolgte ohne die Nutzung grafischer Autorenwerkzeuge. Die Erstellung der Szenenbeschreibungen für die eigentlichen Beiträge gestaltete sich im Vergleich zu den Szenarien *iFair* und *Ilmenoke* (siehe Unterabschnitte 3.6.1 und 3.6.2) auf Grund der wenigen Elemente in einer Szene deutlich einfacher. Aufwändig waren hingegen die Beschreibung der Benutzeroberfläche und des interaktiven Tickers, die ebenfalls Bestandteile der MPEG-4-Szene sind. Aus dem vorliegenden Material wurden zwölf Beiträge nach thematischen Gesichtspunkten zusammengestellt und in die Szenenbeschreibung integriert. Die erforderlichen Skript-Funktionen wurden mit ECMAScript realisiert. Die entstandene Szenenbeschreibung im textuellen Format BT enthält ca. zehntausend Zeilen Quelltext. Diese wurde zusammen mit den Medienobjekten zu einer MPEG-4-Anwendung codiert.

3.6.4 Fazit

Das Objekt- und Szenenkonzept von MPEG-4 wurde für die Beispielszenarien implementiert. Als Resultat sind interaktive audiovisuelle Anwendungen entstanden, die dem Nutzer neue Leistungsmerkmale bieten. Für die Erstellung von Szenarien entsprechend dem Objekt- und Szenenkonzept von MPEG-4 sind Produktionen mit mannigfaltigen Schritten notwendig. Dies setzt Kompetenz auf den unterschiedlichsten Gebieten der Grafik-, Video- und Audioproduktion voraus. Die verfügbaren grafisch-interaktiven Autorenwerkzeuge bieten nicht die benötigte Funktionalität. Es müssen mehrere Werkzeuge in Kombination einschließlich erforderlicher Konvertierungen verwendet werden. Die Anforderungen reichen hin bis zur manuellen Bearbeitung von beispielsweise Szenenbeschreibungen und Objekt-Deskriptoren auf Quelltext-Ebene.

3.7 Computer-Supported Cooperative Work

Bei der Produktion multimedialer Anwendungen ist es oftmals notwendig, dass mehrere Autoren, Designer, Programmierer usw. effektiv zusammenarbeiten. In vielen Bereichen, z. B. bei der Entwicklung von Kiosk-Systemen, Lehranwendungen oder Webpräsenzen haben sich entsprechende Produktionssysteme bereits etabliert. Die Unterstützung der Zusammenarbeit konzentriert sich meist auf die Bearbeitung der Inhalte. Es wird davon ausgegangen, dass sich die grundlegende Struktur einer einmal erstellten Anwendung nur sehr selten ändert, die Inhalte hingegen ständig bearbeitet und aktualisiert werden müssen [Lie01]. Viele etablierte Autorensysteme sehen keinerlei explizite Unterstützung der Zusammenarbeit mehrerer Autoren vor. Die Arbeitsstände werden in Form von Da-

teien weitergegeben; die Abstimmung der Zusammenarbeit müssen die Beteiligten selbst übernehmen. Hierdurch entstehen zwangsläufig viele Varianten und Versionen einer Anwendung.

Die gemeinsame Bearbeitung von Aufgaben ist in vielen Bereichen des Lebens notwendig. Folglich wurden zahlreiche Ansätze und Lösungen entwickelt, die eine solche Zusammenarbeit mit technischen Mitteln unterstützen. *Computer-Supported Cooperative Work (CSCW)* ist die Bezeichnung eines interdisziplinären Forschungsgebietes aus verschiedenen Ingenieur- und Geisteswissenschaften, welches sich mit der Nutzung von Informations- und Kommunikationstechnologie zur Unterstützung von Gruppenarbeit befasst [Gre91], [Gre88], [Dew99]. Einige Aspekte aus diesem breiten Aufgabenspektrum sind:

- Art und Weise der Zusammenarbeit von Menschen in Gruppen
- Koordination von Arbeit
- Anforderungen an die Technologien und deren Auswirkung auf den Menschen
- Anforderungen an Organisationsstrukturen und Auswirkungen auf diese

Inzwischen beschäftigen sich zahlreiche Lehrstühle, Forschungseinrichtungen und Konferenzen ausführlich mit dieser Thematik. Begriffe wie *Groupware*, *Telekonferenz* oder *Telekooperation* entstammen diesem Umfeld. Im Folgenden wird ein kurzer Überblick über CSCW und die wichtigsten Begriffe gegeben.

3.7.1 Kommunikation

In CSCW-Systemen kommen verschiedene Kommunikationsformen zur Anwendung. Diese beschreiben, wie Mitglieder einer oder mehrerer Gruppen Informationen untereinander austauschen. Grundsätzlich sind zwei Arten der Kommunikation möglich:

- *synchrone Kommunikation:* Die Zusammenarbeit erfolgt gleichzeitig (synchron).
- *asynchrone Kommunikation:* Die Zusammenarbeit erfolgt nicht gleichzeitig (asynchron).

Je nach Grad der räumlichen Verteilung unterscheiden sich die nutzbaren Kommunikationsmedien, z. B. direkte Konversation, Telefon oder Verbindung via Internet. Zusätzlich wird die Kommunikation durch die Richtung beschrieben, in der sie ausgeführt wird. Tabelle 3.2 zeigt den Zusammenhang von Art und Richtung anhand von Beispielen.

	Richtung		
Art	1:1	1:n	m:n
synchron	Telefonat	Rundfunk	Telefonkonferenz
asynchron	E-Mail	Mitteilung	Newsgroup

Tabelle 3.2: Richtung und Art der Kommunikation

3.7.2 Datenhaltung

Die Art der Datenhaltung bestimmt maßgeblich die Eigenschaften und die Architektur eines CSCW-Systems. Folgende Ansätze sind zu finden [GC99]:

- *zentralisierte Architekturen:* Alle Daten werden auf einem zentralen Server gehalten und verwaltet. Dieser steuert alle Ein- und Ausgaben der angeschlossenen Clients. Die Arbeiten der Teilnehmer können so sehr einfach synchronisiert und damit die Konsistenz der Daten gewährleistet werden.
- *replizierte Architekturen:* Der gesamte Datenbestand wird auf jedem Client gehalten. Problematisch ist die Gewährleistung der Konsistenz der dann mehrfach vorhandenen Daten.
- *hybride Architekturen:* Diese bilden eine Mischform aus den zuvor genannten Architekturen. Es existiert ein Server für die zentrale Datenhaltung, aber auch replizierte Daten auf jedem Client.

Die Wahl der Architektur hängt stark vom jeweiligen Anwendungsfall ab. In der Praxis sind alle drei Formen der Datenhaltung zu finden.

3.7.3 Klassifizierungen

Klassifizierung nach Ort und Zeit

Die Form der Zusammenarbeit lässt sich nach Ort und Zeit der Bearbeitung klassifizieren:

- Ort: Wo befinden sich die Bearbeiter zum Zeitpunkt der Bearbeitung?
- Zeit: Wann findet die Bearbeitung statt?

Tabelle 3.3 zeigt einige Beispiele für diese Klassifizierung [Gre91]:

	Zeit		
Ort	gleich	verschieden, vorhersehbar	verschieden, nicht vorhersehbar
gleich	gemeinsame Sitzung	Schichtarbeit	schwarzes Brett
verschieden, vorhersehbar	Videokonferenz	E-Mail	gemeinsames Verfassen von Dokumenten
verschieden, nicht vorhersehbar	Mobilfunk-Konferenz	Bulletin Boards	Vorgangsbearbeitung

Tabelle 3.3: Formen der Zusammenarbeit nach [Gre91]

Klassifizierung nach Funktion

Diese Klassifizierung unterscheidet CSCW-Systeme nach ihren Anwendungsbereichen. Die funktionellen Kategorien sind dabei nicht exklusiv, sondern können sich in realen CSCW-Systemen überlappen. Folgende funktionelle CSCW-Systemklassen können unterschieden werden [EGR91]:

- *Nachrichtensysteme:* Sie ermöglichen einen asynchronen Austausch von Nachrichten zwischen den Gruppenmitgliedern, z. B. per E-Mail.

- *Mehrbenutzer-Editoren:* Hier verfassen mehrere Autoren gemeinsam ein Dokument. Dies kann synchron oder asynchron erfolgen.

- *Telekonferenzen:* Sie ersetzen teilweise klassische Konferenzen, indem sie die Bindung der Teilnehmer an einen Ort aufheben. Beispiele hierfür sind Telefonkonferenzen und Videokonferenzen.

- *Intelligente Agenten:* Intelligente Agenten sind Programme, die Aufgaben eines Konferenzteilnehmers übernehmen, z. B. das Aufzeichnen eines Sitzungsprotokolls oder die Überwachung des Sitzungsfortschrittes.

- *Gemeinsame Informationsräume:* Die Gruppeninformationen werden vom System verwaltet, persistent gespeichert und mit geeigneten Zugriffsmechanismen bereitgestellt, z. B. durch spezielle Datenbanken oder verteilte Hypermediasysteme.

- *Koordinierungssysteme:* Koordinierungssysteme dienen der Koordinierung der Tätigkeiten, die von den Gruppenmitgliedern zur Vollendung des gemeinsamen Zieles ausgeführt werden müssen. Das Problem der Koordination tritt verstärkt bei asynchroner Zusammenarbeit auf.

Klassifizierung nach Unterstützungsfunktion

CSCW-Systeme können in Abhängigkeit vom Grad der Unterstützung von Kommunikation, Koordination und Kooperation klassifiziert werden. Diese Unterstützungsfunktionen werden Systemklassen genannt:

- *Kommunikation:* Diese Systemklasse realisiert den expliziten Informationsaustausch zwischen den Gruppenmitgliedern. Sie ermöglicht das Überwinden von Ort und Zeit.

- *Gemeinsame Informationsräume:* Sie erlauben einen impliziten Informationsaustausch zwischen Gruppenmitgliedern. Diese Systemklasse verwaltet ebenfalls die persistenten Gruppeninformationen und stellt geeignete Zugriffsmechanismen zur Verfügung.

- *Workgroup Computing:* Diese Systemklasse stellt die Unterstützung der Kooperation in den Vordergrund. Es werden insbesondere schwach strukturierte und sich selten wiederholende Tätigkeiten unterstützt.

- *Workflow Management:* Diese Systemklasse koordiniert die Aktivitäten und Ressourcen der Gruppenmitglieder und führt zu einem verbesserten Informationsfluss.

3.7.4 Grundfunktionen

Im Folgenden werden die Grundfunktionen eines CSCW-Systems kurz vorgestellt [GC99]:

- *Kommunikationsmechanismen:* Sie sind für die Kommunikation der Teilnehmer einer Konferenz sowie für die Kommunikation zwischen den verteilten Instanzen des CSCWS-Systems zuständig.

- *Koordination:* Für die Koordination des kollaborativen Arbeitens von Gruppenteilnehmern sind vier verschiedene Systeme anwendbar:
  - *Formular-basiertes System:* Modellierung des Datenflusses innerhalb einer Organisation
  - *Prozedur-basiertes System:* Modellierung der Funktionen und Abläufe innerhalb einer Organisation
  - *Konversations-orientiertes System:* Modellierung der Interaktionen zwischen den Teilnehmern sowie der daraus resultierenden Aktionen
  - *Kommunikations-orientiertes System:* Modellierung komplexer Kommunikationsstrukturen innerhalb einer Organisation

- *Awareness:* Als Awareness wird das Bewusstsein der Kollaborationspartner über die Existenz und die Aktivitäten anderer Konferenzteilnehmer bezeichnet.

- *Konferenzverwaltung:* Verwaltung der Benutzer, der aktiven Dienste und der Softwarekomponenten:
  - Anmeldung und Abmeldung von Teilnehmern
  - Organisation der Nutzer und Arbeitsgruppen
  - Zuweisung von Rechten
  - Identifizierung und Authentifizierung der Teilnehmer
  - Starten und Beenden der Konferenz
  - Vermittlung zwischen den Werkzeugen und den aktiven Benutzern
  - Moderation zwischen verschiedenen Teilnehmern
  - Vermittlung von Awareness

- *Nebenläufigkeitskontrolle:* Kontrolle von parallelen Aktionen unterschiedlicher Nutzer, die möglicherweise nicht miteinander vereinbar sind und somit zu inkonsistenten Daten führen können:
  - *optimistische Nebenläufigkeitskontrolle:* Sie sichert nicht in jedem Fall die Konsistenz der Daten unterschiedlicher Nutzer. Sie erlaubt inkonsistenten Zugriff, was für bestimmte Anwendungsfälle vorteilhaft sein kann.
  - *pessimistische Nebenläufigkeitskontrolle:* Sie verhindert Inkonsistenzen für Anwendungsfälle, in denen eine Konsistenz der Daten erforderlich ist. Dies kann entweder durch eine zentrale oder dezentrale Kontrolle erreicht werden.

CSCW stellt ein sehr inhomogenes und breites Forschungsgebiet dar und beeinflusst direkt oder indirekt viele Anwendungen im täglichen Leben. Auch in einem auf mehrere Autoren verteilten Authoring-Prozess kommen verschiedene Ansätze und Techniken aus diesem Bereich zur Anwendung. Weiterführende Literatur ist beispielsweise zu finden in [Gre88], [Gre91], [Dew99], [CD04].

3.8 Zusammenfassung

Die Umsetzung objektbasierter AV-Anwendungen auf Basis des Objekt- und Szenenkonzeptes von MPEG-4 erfordert neue Methoden, Verfahren und Werkzeuge für deren Erstellung. Es sind Kenntnisse und Fähigkeiten aus den Bereichen Computergrafik, Multimedia-Produktion, Studioproduktion und Programmierung erforderlich. In vielen Fällen werden daher mehrere Autoren benötigt, die mit ihren jeweiligen Spezialisierungen gemeinsam an einer Anwendung arbeiten. Grundlage der Betrachtung waren die Abläufe und Verfahren bei der Produktion herkömmlicher Multimedia-Anwendungen. Die Mehrzahl der bisher verfügbaren Anwendungen wird aufwändig unter Nutzung verschiedener Werkzeuge erstellt, die meist in Kombination angewendet werden müssen. Dies erfordert umfangreiche Kenntnisse der zugrunde liegenden Technologie. Die bisher verfügbaren Werkzeuge und Autorensysteme reichen nicht aus, um die umfangreichen Möglichkeiten des Objekt- und Szenenkonzeptes von MPEG-4 adäquat in objektbasierten AV-Anwendungen zu nutzen. Die Verfügbarkeit leistungsfähiger Autorensysteme ist jedoch eine wichtige Voraussetzung für den künftigen Erfolg solcher Anwendungen.

4 Autorensystem

Nachdem in den Kapiteln 2 und 3 die technischen Grundlagen und der Prozess der Erstellung objektbasierter AV-Anwendungen betrachtet wurden, widmet sich dieses Kapitel dem Entwurf eines universell einsetzbaren Autorensystems für die Erstellung solcher Anwendungen. Das Anwendungsgebiet soll zunächst möglichst wenig eingegrenzt werden. Die konkreten Anforderungen werden aus verschiedenen Szenarien des Forschungsprojektes IAVAS abgeleitet (siehe Unterabschnitt 2.5.3 und Abschnitt 3.6).

4.1 Anforderungen

Die objektbasierte Beschreibung audiovisueller Anwendungen hat tief greifende Auswirkungen auf den gesamten Produktionsprozess. Der Arbeitsablauf aller an einer Produktion Beteiligten ändert sich grundlegend. Es werden Qualifikationen aus dem Bereich des Multimedia-Authorings ebenso benötigt wie aus den Bereichen Film- und Fernsehproduktion. Die Erstellung von Objekten und Szenenbeschreibungen lässt die entstehende Datenflut dramatisch ansteigen. Damit steigen auch die technischen Anforderungen an die Komponenten des Autorensystems.

Der Prozess des Authorings hat die Zusammenführung aller entstandenen Komponenten zu einer funktionsfähigen AV-Anwendung zum Ziel. Die räumliche und zeitliche Komposition der Elemente und die Beschreibung von Interaktionsmöglichkeiten sind mitunter hoch-komplexe Aufgaben, die nur selten von einem einzelnen Autor bearbeitet werden können. Die Unterstützung der Zusammenarbeit mehrerer Autoren mit möglicherweise verschiedenen Qualifikationen ist daher eine wichtige Anforderung an Autorensysteme, die bei aktuellen Lösungen stark vernachlässigt wurde. Ausgehend von den Erkenntnissen aus den Kapiteln 2 und 3 können folgende Anforderungen an ein Autorensystem für die Erstellung objektbasierter AV-Anwendungen formuliert werden:

- Das Autorensystem soll es mehreren Autoren ermöglichen, sich mit ihrer jeweiligen Spezialisierung an einer Produktion aktiv zu beteiligen.

- Das System muss flexibel an verschiedene Bedürfnisse und Randbedingungen anzupassen sein, z. B. Art und Umfang der Produktion, Anzahl der Beteiligten usw.

- Das System muss für alle Arbeitsschritte und Tätigkeiten geeignete Werkzeuge anbieten. Die Nutzung externer Werkzeuge, z. B. Produktionswerkzeuge, Editoren für Medienobjekte oder Animationswerkzeuge muss möglich sein.

- Die erreichten Arbeitsstände sollen zwischen Autoren und zwischen Autorensystemen austauschbar sein. Hierfür müssen alle relevanten Informationen in klar definierten Authoring-Formaten gespeichert werden.

- Die erstellte AV-Anwendung muss umfassend erprobt und an ein Distributionssystem übergeben werden können.

- Das System muss um die Funktionalität eines vollständigen Produktionssystems erweiterbar sein. Dies beinhaltet dann auch die Produktion und Bearbeitung von Medienobjekten.

Diese zunächst sehr allgemein formulierten Anforderungen werden in den weiteren Abschnitten konkretisiert.

4.2 Ansätze

4.2.1 Erweiterung existierender Autorenwerkzeuge

Der Authoring-Prozess objektbasierter AV-Anwendungen weist viele Gemeinsamkeiten mit entsprechenden Abläufen auf dem Gebiet der Multimedia-Produktion auf. Daher liegt es nahe, vorhandene Autorenwerkzeuge um die noch fehlende Funktionalität zu erweitern. Einige kommerzielle Autorenwerkzeuge bieten Schnittstellen für Erweiterungen an, die jedoch in ihren Möglichkeiten häufig begrenzt sind. Eine volle Kontrolle über die Szenendaten ist daher meist nicht erreichbar. Die häufige Verwendung proprietärer Datenformate erschwert einen solchen Weg zusätzlich.

Für Anwendungen, bei denen der Schwerpunkt des Authorings auf der Erstellung synthetischer 3D-Welten liegt, können als Ausgangspunkt etablierte 3D-Grafikwerkzeuge, wie z. B. *Maya* oder *3D-Studio* verwendet werden. Diese müssen dann lediglich um ein Export-Modul für ein encodierbares Datenformat erweitert werden. Als Quasi-Standard hat sich hierfür das Format VRML durchgesetzt (siehe Unterabschnitt 2.3.4), das bereits in vielen Werkzeugen verfügbar ist. Diese Exportfunktion lässt sich prinzipiell erweitern auf den Export einer textuellen MPEG-4-Szenenbeschreibung. In [KVW03] ist eine solche Erweiterung des Grafikwerkzeuges *3D-Studio MAX* beschrieben. Neben der Implementierung einer Exportfunktion wurde auch der Funktionsumfang des Werkzeuges erweitert.

Dieser Ansatz kann verfolgt werden, wenn das vorhandene Autorenwerkzeug lediglich um wenige Funktionen erweitert werden muss. Die entstehenden Lösungen sind meist proprietär und nicht auf andere Werkzeuge übertragbar.

4.2.2 Authoring auf Basis eines Terminals

Eine Möglichkeit zur grafisch-interaktiven Erstellung von Szenenbeschreibungen ist die Nutzung von MPEG-4-internen Funktionen mit einem MPEG-4-Terminal. Der MPEG-4-Standard sieht mehrere Verfahren für strukturelle Änderungen am Szenengraphen zur Laufzeit einer Anwendung vor (siehe Unterabschnitt 2.4.4). Die verfügbaren Funktionen decken die wichtigsten Arbeitsschritte des Authorings ab (*INSERT*, *DELETE*, *REPLACE*). Ein MPEG-4-Terminal müsste um Authoring-Funktionen erweitert werden: z. B. verschiedene Sensoren zur Auswahl von Medienobjekten und spezielle Dialogelemente. Diese können in Form einer MPEG-4-Szene beschrieben werden. Das Terminal wird mit einer solchen Szene gestartet, die zunächst keine sichtbaren Elemente enthält. Mit Hilfe von BIFS-Commands wird die Szene Schritt für Schritt erweitert (*INSERT*, *REPLACE*); die erforderlichen Kommandos werden interaktiv vom Autor ausgelöst. Um beispielsweise Objekte innerhalb der Szene grafisch-interaktiv zu positionieren, werden diese durch einen Sensor (*Plane-Sensor*) ausgewählt und per Drag&Drop an die gewünschte Stelle verschoben. Hierfür wird ein *REPLACE*-Kommando mit den entsprechenden Koordinaten erzeugt. Die erforderlichen Berechnungen (z. B. Transformationen) führt ein Skript-

knoten auf Basis von *ECMAScript* oder *MPEG-J* aus. Alle Kommandos werden durch Aktionen im Terminal ausgelöst und direkt in die MPEG-4-Szene encodiert. Das Terminal ist somit Werkzeug und Betrachter zugleich, wodurch das WYSIWYG-Paradigma erfüllt ist. Nach Fertigstellung der Szene müssen die Authoring-Funktionen wieder aus der Szenenbeschreibung entfernt werden. Auch dies erfolgt mit Hilfe eines BIFS-Commands (*DELETE*). Auf diese Weise lässt sich ein Autorenwerkzeug ausschließlich unter Nutzung interner MPEG-4-Funktionen realisieren. Folgende Komponenten sind erforderlich:

1. MPEG-4-Terminal

2. Szenenbeschreibung der Authoring-Elemente (z. B. Sensoren und Dialoge)

3. BIFS-Encoder zur schnellen Erzeugung der BIFS-Commands

Als Ergebnis liegt eine MPEG-4-Anwendung im BIFS-Format vor. Diese kann mit einem geeigneten Decoder in eine textuelle Szenenbeschreibung überführt werden. Auch ein verteiltes Authoring ist denkbar. Alle an der gemeinsamen Bearbeitung beteiligten Terminals starten mit derselben Szenenbeschreibung. Die Änderungen an der Szene können mit Hilfe eines Streaming-Servers per HTTP oder RTSP an die angeschlossenen Terminals verteilt werden. Abbildung 4.1 zeigt eine entsprechende Architektur.

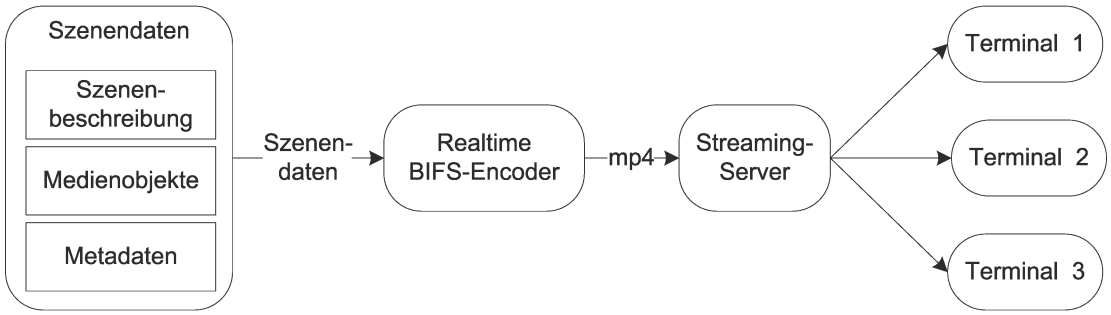

Abbildung 4.1: Verteilung von Voransichten mit BIFS-Streaming

Dieser Ansatz ist auch in Kombination mit anderen Ansätzen anwendbar. So kann auf diese Weise beispielsweise die Verteilung von Voransichten in einer verteilten Umgebung realisiert werden. Hierfür wurden erste erfolgreiche Versuche unter Verwendung des *Darwin Streaming Servers*[1] unternommen.

Eine andere Möglichkeit ist die Erweiterung eines MPEG-4-Terminals um neue Funktionen. Hierfür werden DMIF-Implementierungen auf Client- und Serverseite integriert, die einen direkten Austausch von BIFS-Commands ermöglichen. Ein Streaming-Server ist dann nicht mehr erforderlich. Dieser Ansatz wurde beispielsweise in [LD02b], [LD02a], [Dob01] verwendet. Abbildung 4.2 zeigt eine solche Architektur. Das Authoring auf Basis eines MPEG-4-Terminals findet prinzipbedingt auf der Abstraktionsebene von BIFS statt. Bearbeitungen auf einem höheren semantischen Niveau sind auf diese Weise nicht möglich.

[1] http://developer.apple.com/darwin/projects/streaming/

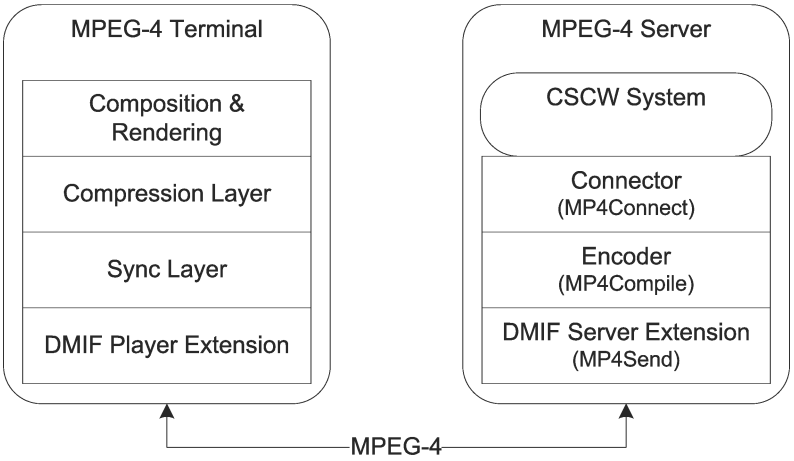

Abbildung 4.2: MPEG-4 CSCW-Architektur nach [LD02b]

4.2.3 Server-basiertes Authoring

Das Server-basierte Authoring ist ein Ansatz, bei dem alle Informationen auf einem zentralen Server (Authoring-Server) gehalten und den beteiligten Autoren unter Berücksichtigung individueller Berechtigungen zur Verfügung gestellt werden. Praktisch alle *Content Management Systeme (CMS)* und *Redaktionssysteme* arbeiten nach diesem Prinzip. Aus dem Bereich des Webpublishing kommen zahlreiche Ansätze und Lösungen für Server-basiertes Authoring.

Eine mögliche Komponente für Server-basiertes Authoring ist *Web-based Distributed Authoring and Versioning (WebDAV)*, eine von der *Internet Engineering Task Force (IETF)*[2] vorgeschlagene Erweiterung des Protokolls HTTP [SW]. Das WebDAV-Protokoll erweitert HTTP um einen Satz neuer Methoden und Header für die Bearbeitung und Speicherung von Daten im Netz. Entfernte Dateisysteme können direkt in den Arbeitsplatz integriert werden. Es sind Mechanismen zur Sperrung und Freigabe von Dateien vorgesehen, um die Zusammenarbeit mehrerer Bearbeiter zu erleichtern. Mit der Erweiterung *DeltaV* können Versionen verwaltet werden. Die Unterstützung von WebDAV ist inzwischen in vielen Produkten integriert. Dateien und Ordner können so direkt auf entfernten Servern bearbeitet werden [GWF+99].

Auf dem Gebiet der Softwareentwicklung sind so genannte Versionierungs-Systeme verbreitet. Der bekannteste Vertreter dieser Art ist das *Concurrent Versioning System (CVS)*[3]. Mehrere Entwickler können gemeinsam an einem Projekt arbeiten, indem sie sich einen Quelltext von einem CVS-Server abholen (*auschecken*), diesen separat bearbeiten und dann wieder auf dem Server ablegen (*einchecken*). Der Server fügt die bearbeiteten Fragmente zusammen und erlaubt einen Zugriff auf die Arbeitsstände der Entwicklung. Logische Konflikte (z. B. gleichzeitige Bearbeitung der gleichen Bereiche im Quelltext durch mehrere Autoren) müssen durch die Anwender vermieden werden. Ein solches System ist prinzipiell für alle textbasierten Formate und somit auch für Szenenbeschreibungen einsetzbar.

Bei der Produktion objektbasierter AV-Anwendungen sind die Anforderungen an die Organisation der Daten deutlich höher als bei vielen anderen Anwendungen. Besondere

[2] http://www.ietf.org/
[3] https://www.cvshome.org/

Anforderungen gelten für die Verwaltung der Szenenbeschreibungen. Für eine sinnvolle Zusammenarbeit mehrerer Autoren ist die Steuerung der Zugriffe auf einzelne Bereiche und Elemente eines Szenengraphen notwendig. Für die Verwaltung umfangreicher, komplexer Daten liegt die Nutzung eines Datenbankmanagementsystems (DBMS) nahe. Solche Ansätze werden im Abschnitt 5.2 näher betrachtet.

4.3 Entwurf eines Autorensystems

4.3.1 Benutzerprofil

Ein Autorensystem soll den Autoren eine grafisch-interaktive Erstellung objektbasierter AV-Anwendungen ermöglichen, ohne dass diese tief greifende Kenntnisse der technischen Zusammenhänge haben müssen. Auf Grund des Entwicklungsstandes der verfügbaren Werkzeuge (siehe Unterabschnitt 3.4.2) sowie der zugrunde liegenden Standards und Technologien (siehe Abschnitt 2.4) ist jedoch davon auszugehen, dass Autoren auf absehbare Zeit Expertenwissen benötigen werden, um anspruchsvolle Anwendungen erstellen zu können. Die Integration neuartiger Medienobjekte (z. B. beliebig geformte Videoobjekte und 3D-Videoobjekte), die Gestaltung von Interaktionskonzepten und die Berücksichtigung der verschiedenen Distributions- und Konsumtionsformen erfordert auch Verständnis der technischen Zusammenhänge. Die vielfältigen zu bearbeitenden Aufgaben sind effektiv nur in einem Team von spezialisierten Autoren zu lösen. Für die Optimierung der Anwendungen werden Kenntnisse der technischen Details des MPEG-4-Standards benötigt. Auch um die zunächst vorhandenen Einschränkungen im Funktionsumfang neuer grafisch-interaktiver Autorenwerkzeuge zu umgehen, müssen spezialisierte Autoren die Szenenbeschreibungen bei Bedarf auch auf Quelltext-Ebene bearbeiten können.

4.3.2 Aufgaben innerhalb des Authoring-Prozesses

Das Authoring einer komplexen AV-Anwendung umfasst vielfältige Aufgaben, die meist von mehreren Personen mit verschiedenen Qualifikationen bearbeitet werden müssen. Eine strenge Abgrenzung der Aufgabenbereiche wird dabei oft nicht vollzogen bzw. ist nicht immer notwendig. Der Authoring-Prozess umfasst folgende Aufgaben:

1. Leitung und Organisation: Direktor
2. Sammeln und Aufbereiten der Inhalte
3. örtliche Komposition (visuell und auditiv)
4. zeitliche Komposition (visuell und auditiv)
5. Interaktionsdesign
6. Optimierung und Test
7. Vorbereitung zur Distribution
8. Pflege

Die Leitung und Organisation bezieht sich meist auf die gesamte Produktion. Dies schließt auch die inhaltliche Leitung und die Erstellung aller Elemente (z. B. der Medienobjekte) ein. Bezogen auf das Authoring ist der Direktor zunächst für das Anlegen des Projektes und die Verteilung der Aufgaben auf Autoren bzw. Autorenteams zuständig. Schritt 2 ist die Schnittstelle zu den Medienproduzenten. Die erstellten Medienobjekte werden übernommen und ggf. an die Erfordernisse des Authorings angepasst. Hierfür ist vor allem technisches Verständnis für Medienformate und deren Integrierbarkeit erforderlich. Die Schritte 3, 4 und 5 sind Kern des Authoring-Prozesses. Hier werden alle Elemente zu einer Anwendung zusammengefügt. Es sind verschiedene Fähigkeiten erforderlich. Die örtliche Komposition erfordert gestalterisches Geschick, aber auch eine genaue Vorstellung über die zeitlichen Abläufe. Die Komposition der zeitlichen Abläufe erzeugt die Dramaturgie der Anwendung. Neben dem globalen Ablauf der Anwendung werden hier auch „lokale“ zeitliche Abläufe (z. B. Animationen und Szenenübergänge) festgelegt. Eine besondere Bedeutung bei der Erstellung objektbasierter AV-Anwendungen hat die Gestaltung von Interaktionskonzepten. Hierdurch kann sowohl die räumliche als auch die zeitliche Anordnung der Elemente einer Anwendung durch den späteren Nutzer beeinflusst werden. Besonders zu beachten ist hierbei die Integration der Interaktionskonzepte in das Szenario der Distribution. Nach Abschluss dieser Arbeiten muss eine Anwendung ausführlich getestet und ggf. optimiert werden. Nun ist die Anwendung bereit für die Distribution. Viele Szenarien erfordern eine kontinuierliche Pflege der Inhalte. Dies kann sowohl eine inhaltliche Pflege (z. B. Aktualisierung von Inhalten) als auch eine weitere technische Optimierung sein.

4.3.3 Authoring-Konzept

Das Ziel des Authorings ist die Erstellung einer encodierbaren AV-Anwendung. Hierfür werden alle Elemente räumlich und zeitlich zusammengefügt und ggf. mit Interaktionsmöglichkeiten versehen. Die Szenenbeschreibung wird dabei schrittweise aufgebaut; auch Zwischenergebnisse können jederzeit encodiert werden. Alle Arbeitsschritte werden direkt in XML-basierten Authoring-Formaten abgebildet und mit Metadaten versehen. Somit ist die vollständige Kontrolle über die Inhalte permanent gegeben. Die Bearbeitung der Szenendaten im Authoring-Format erfolgt mit Hilfe von Editoren, die Bestandteil von Autorenwerkzeugen sein können. Alle Editoren bzw. Autorenwerkzeuge arbeiten auf Basis derselben Szenendaten; somit ist die Konsistenz der Anwendung jederzeit gewährleistet. Der gesamte Authoring-Prozess lässt sich auf wenige grundlegende Funktionen reduzieren: das Einfügen, das Bearbeiten und das Löschen von Elementen im Szenengraphen.

Die Zusammenarbeit mehrerer Autoren wird auf verschiedene Arten unterstützt. Der gesamte Authoring-Prozess basiert auf XML-basierten Authoring-Formaten, die zwischen Autoren und Systemen ausgetauscht werden können. Damit werden Behinderungen der Zusammenarbeit, wie sie bei der Verwendung proprietärer Datenformate auftreten können, vermieden. Die Aufteilung einer Szene auf mehrere Dateien ermöglicht bereits auf Datei-Ebene eine parallele Bearbeitung der Szenendaten durch mehrere Autoren. Eine weitaus komfortable Zusammenarbeit ist unter Nutzung eines Authoring-Servers möglich. Dieser verwaltet alle relevanten Daten und erlaubt die gleichzeitige Bearbeitung der Szenendaten durch mehrere Autoren.

4.3.4 Szenarien der Zusammenarbeit

Es lassen sich drei mögliche Abläufe des Authoring-Prozesses hinsichtlich der Zusammenarbeit mehrerer Autoren unterscheiden. Dabei wird immer davon ausgegangen, dass zu Beginn alle Medienobjekte und Informationen in ihrer endgültigen Form vorliegen.

1. *Einzelbearbeitung:* Ein Autor erstellt die gesamte Anwendung.
2. *asynchrone Bearbeitung:* Mehrere Autoren arbeiten nacheinander (nicht gleichzeitig) an einer Anwendung.
3. *synchrone Bearbeitung:* Mehrere Autoren arbeiten gemeinsam (gleichzeitig) an einer Anwendung. Die Ergebnisse der einzelnen Bearbeiter müssen kombiniert und störende Beeinflussungen vermieden werden. Hierbei sind zwei Fälle zu unterscheiden:
   - *optimistische Annahme:* Störende Beeinflussungen (z. B. gleichzeitige Bearbeitung desselben Elementes) sind unwahrscheinlich und werden daher vom System nicht explizit verhindert (optimistische Nebenläufigkeitskontrolle).
   - *pessimistische Annahme:* Störende Beeinflussungen (z. B. gleichzeitige Bearbeitung desselben Elementes) sind wahrscheinlich und müssen daher vom System explizit verhindert werden (pessimistische Nebenläufigkeitskontrolle).

Ablauf 1 erfordert prinzipbedingt keine gesonderte Koordinierung. Bei Ablauf 2 muss das Autorensystem lediglich eine definierte Übergabe der erreichten Arbeitsstände ermöglichen. Im einfachsten Fall erfolgt dies durch Übergabe von Dateien und Notizen. Die Nutzung eines Authoring-Servers kann hilfreich sein, ist aber nicht zwingend erforderlich. Ablauf 3 hingegen stellt deutlich höhere Anforderungen an das System. Die Ergebnisse der einzelnen Bearbeitungsschritte müssen richtig kombiniert und störende Beeinflussungen ausgeschlossen werden. Dies erfordert eine gute Strukturierung des Authoring-Prozesses und eine konsequente Aufgabenteilung. Die Verwendung eines zentralen Authoring-Servers ist sinnvoll. Die synchrone Bearbeitung kann hinsichtlich der zu erwartenden Beeinflussung von Autoren untereinander (Nebenläufigkeit) in zwei Ausprägungen erfolgen (siehe Abschnitt 3.7). Die optimistische Annahme erleichtert den Aufbau eines Systems erheblich, kann aber oft nicht vorausgesetzt werden. Bei der pessimistischen Annahme wird der gesamte Ablauf des Authoring-Prozesses vom System gesteuert. Dies kann u. U. die Arbeit des Einzelnen beeinträchtigen, wenn z. B. der Zugriff auf Ressourcen verwehrt wird. Das angestrebte Autorensystem soll auch das Authoring unter pessimistischer Annahme ermöglichen. Der Direktor legt die zu schützenden Bereiche fest und entscheidet über die anzuwendenden Kriterien zur Vermeidung von Konflikten. Die gemeinsame Bearbeitung der Szenenbeschreibung durch mehrere Autoren kann prinzipiell auf zwei Arten erfolgen:

1. Die Autoren arbeiten in ihnen zugewiesenen Bereichen des Szenengraphen, der entsprechend unterteilt werden muss. Dies kann z. B. durch Aufteilung der Szene in einzelne Unterszenen geschehen, die als separate Dateien organisiert werden. Die Autoren bearbeiten die ihnen zugewiesenen Dateien exklusiv. Eine Nebenläufigkeitskontrolle ist nicht notwendig. Anschließend werden die Unterszenen wieder zu einer Szene zusammengefügt, z. B. unter Verwendung des *Inline*-Knotens. Gegenseitige Störungen können so leicht vermieden werden. Die strenge Zuweisung von Dateien zu Autoren schränkt eine flexible Zusammenarbeit jedoch stark ein. Diese Art der Zusammenarbeit wird häufig als *kooperative Zusammenarbeit* bezeichnet.

2. Alle Autoren bearbeiten denselben Szenengraphen asynchron oder synchron. Widersprüchliche Bearbeitungen müssen durch Kommunikation der Autoren oder durch technische Mittel vermieden werden (pessimistische Nebenläufigkeitskontrolle). Diese Art der Zusammenarbeit ist Voraussetzung für eine synchrone Bearbeitung und wird häufig als *kollaborative Zusammenarbeit* bezeichnet.

Abhängig vom konkretenen Anwendungsfall sind beide Arten der Zusammenarbeit denkbar. Das Autorensystem sollte daher beide Arbeitsweisen ermöglichen.

4.3.5 Datenhaltung

Die Art der Datenhaltung ist von zentraler Bedeutung für die Entwicklung der Architektur eines Autorensystems. In Unterabschnitt 3.7.2 wurden die grundsätzlichen Formen der Datenhaltung in einem CSCW-System genannt. Prinzipiell sind in einem Autorensystem alle drei Arten denkbar. Bei der Produktion umfangreicher AV-Anwendungen müssen an allen Stellen im Produktionsprozess umfangreiche Daten zwischen den Beteiligten ausgetauscht werden. Hierfür werden für gewöhnlich Server (z. B. Fileserver oder Medienserver) verwendet. Selbst im Falle der Bearbeitung durch nur einen einzelnen Autor ist es oft sinnvoll, alle Daten zentral zu halten. Unter diesen Annahmen liegt es nahe, auch für das Authoring eine Server-basierte Architektur aufzubauen.

Im Falle des Autorensystems ist eine ausschließlich zentralisierte Architektur jedoch nicht realisierbar, da für die Anzeige und Bearbeitung einer objektbasierten AV-Anwendung ein lokaler Szenengraph vorhanden sein muss. Auch die Forderung nach der Möglichkeit einer autarken Bearbeitung (ohne Server) schließt eine zentralisierte Architektur nach [Gre88] aus. Folglich erfüllt nur die hybride Architektur (siehe Unterabschnitt 3.7.2) die gestellten Anforderungen.

Für das Autorensystem wird die Verwendung eines zentralen Authoring-Servers vorgeschlagen. Dieser verwaltet alle Daten der audiovisuellen Anwendung und stellt diese den Autoren in geeigneter Form zur Verfügung. Neben der Verwaltung von Szenendaten soll der Server auch die Gruppenarbeit unter der Annahme einer pessimistischen Nebenläufigkeit ermöglichen. Hierfür sind die Zugriffe auf die Szenenbeschreibungen auf Knotenebene zu steuern.

4.3.6 Architektur

Struktur des Autorensystems

Für die Umsetzung des Authoring-Konzeptes und im Hinblick auf die geforderte Erweiterbarkeit zu einem vollständigen Produktionssystem wird ein modular aufgebautes, Server-basiertes Autorensystem vorgeschlagen. Abbildung 4.3 zeigt die Grobstruktur des Systems. Mit Hilfe der *Produktionswerkzeuge* werden die Medienobjekte erstellt und auf dem Server abgelegt. Die *Medien-Editoren* dienen der Bearbeitung der Medienobjekte. Unter Nutzung der *Autorenwerkzeuge* wird die AV-Anwendung erstellt und schließlich an ein *Distributionssystem* übergeben. Produktionswerkzeuge, Medien-Editoren und Distributionssystem stehen nicht im Fokus dieser Arbeit. Es müssen lediglich geeignete Schnittstellen zum Server bereitgestellt werden. Im einfachsten Fall erfolgt der Datenaustausch auf Datei-Ebene.

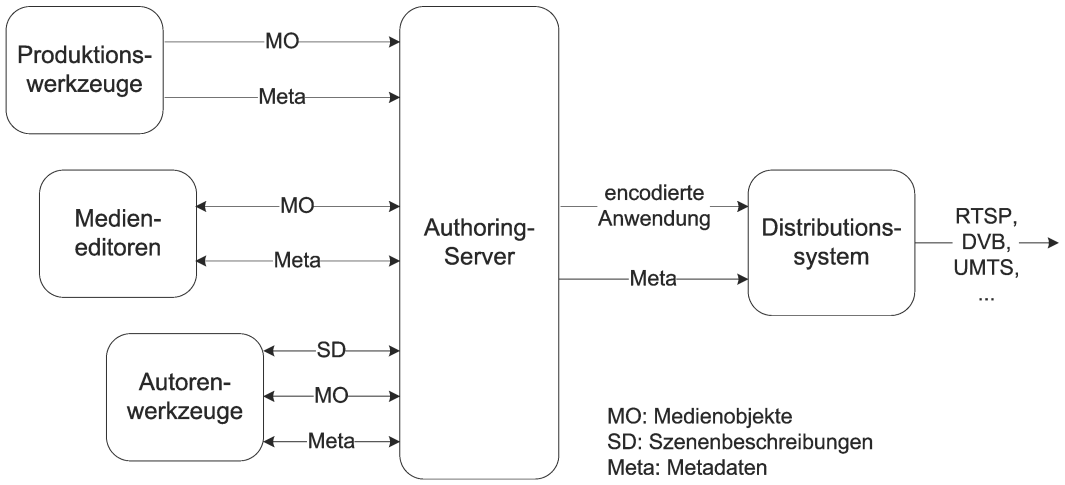

Abbildung 4.3: Struktur des vorgeschlagenen Autorensystems

Autorensystem

Die Kernaufgabe des Autorensystems ist die Erstellung der vollständigen Szenenbeschreibung einer audiovisuellen Anwendung. Ein universell einsetzbares Konzept erfordert eine modulare Architektur mit klar definierten Schnittstellen. Dies ermöglicht eine flexible Anpassung des Systems an verschiedene Anwendungsgebiete sowie die Integration in bestehende Produktionssysteme und Abläufe. Die einzelnen Module können unabhängig voneinander entwickelt und implementiert werden. Eine wichtige Anforderung an die Architektur ist die vollständige Entkoppelung von Präsentation, Verarbeitung und Speicherung der Szenendaten. Im Mittelpunkt des Systems stehen immer die Szenendaten bestehend aus Szenenbeschreibungen, Medienobjekten und Metadaten. Diese werden mit verschiedenen Werkzeugen bearbeitet und können jederzeit zu einer AV-Anwendung zusammengefügt und encodiert werden. Über die eigentlichen Aufgaben eines Autorensystems hinausgehend ermöglicht die Architektur auch eine Anbindung von Werkzeugen für die Erstellung bzw. Bearbeitung von Medienobjekten. Somit ist der Ausbau zu einem vollständigen Produktionssystem möglich.

Abbildung 4.4 zeigt den grundsätzlichen Aufbau des vorgeschlagenen Autorensystems. Im Mittelpunkt steht die Datenhaltung mit den Szenendaten. Zu Beginn eines neuen Projektes wird ein initialer Szenengraph geladen, der das Grundgerüst einer Anwendung enthält. Im einfachsten Fall besteht dieser lediglich aus einem Wurzelknoten und ggf. einem IOD. Sinnvoll ist eine Ergänzung um Prototypen für Elemente, die für die zu erstellende Anwendung typisch sind. Im nächsten Schritt werden Medienobjekte und die mit ihnen verbundenen Metadaten in die Datenhaltung des Autorensystems importiert. Auch der Import von Szenenbeschreibungen oder Fragmenten von solchen ist möglich. Die Datenhaltung enthält nun alle Szenendaten. Im nächsten Schritt können die Medienobjekte in den Szenengraphen integriert werden. Die Bearbeitung der Szenenbeschreibungen und Metadaten erfolgt mit Hilfe spezialisierter Werkzeuge (Editoren). Als Ergebnis liegt eine encodierbare AV-Anwendung vor. Nach dem Encodieren kann diese betrachtet oder an ein Distributionssystem übergeben werden.

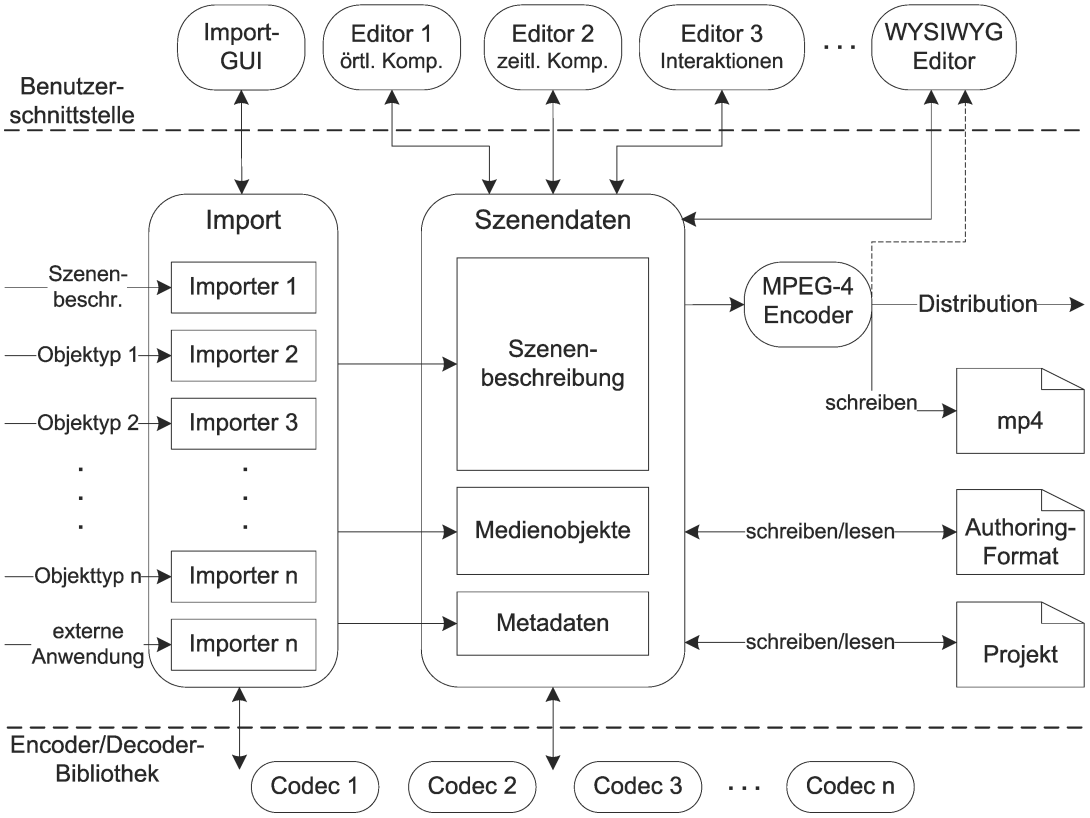

Abbildung 4.4: Architektur des Autorensystems

Verteiltes Authoring

Die Unterstützung eines auf mehrere Autoren verteilten Authoring-Prozesses stellt besondere Anforderungen an die Architektur. Für das vorgeschlagene Autorensystem wird ein Server-basierter Ansatz gewählt. Alle Daten werden in diesem Fall zentral auf einem Authoring-Server verwaltet und den beteiligten Autoren unter Berücksichtigung individueller Berechtigungen zur Verfügung gestellt. Prinzipiell ist ein verteiltes Authoring auch ohne Verwendung eines Servers unter Nutzung eines *Peer-to-Peer*-Ansatzes denkbar; diese Möglichkeit wird hier jedoch nicht verfolgt. Der Entwurf eines Authoring-Servers ist in Kapitel 5 ausführlich beschrieben.

Im Falle eines Server-basierten Authorings werden Teile der Architektur nach Abbildung 4.4 durch den Authoring-Server gebildet, der dann die Datenhaltung und die Steuerung der Abläufe übernimmt. Im einfachsten Fall fungiert der Server lediglich als externer Datenspeicher für den Datenaustausch. Der Import von Elementen kann sowohl zentral auf dem Server als auch lokal durch die angeschlossenen Autorenwerkzeuge erfolgen. Gleiches gilt für die Decoder und Encoder. Zu Beginn der Arbeit werden alle erforderlichen Daten vom Server in die lokale Datenhaltung des Autorenwerkzeuges importiert (replizierte Datenhaltung). Anschließend werden die lokalen Daten bearbeitet und die Ergebnisse wieder auf dem Server abgelegt. Im Falle einer nicht-gleichzeitigen (asynchronen) Bearbeitung einer Szene durch mehrere Autoren greifen diese auf einen gemeinsamen Datenbestand zu und stellen ihre Ergebnisse für die weitere Bearbeitung zur Verfügung. Im Falle einer gleichzeitigen (synchronen) Bearbeitung einer Szene durch mehrere Autoren werden alle Bearbeitungsschritte und Änderungen sofort in den zentralen Szenengraphen auf dem Server eingetragen und stehen somit allen Beteiligten zur Verfügung.

4.3.7 Komponenten

Import

Die Import-Module sind die Schnittstellen am Eingang des Autorensystems. Die zu importierenden Daten müssen an die Erfordernisse der Datenhaltung angepasst und ggf. um Metadaten ergänzt werden. Im einfachsten Fall liegen die Daten bereits in einem für das Authoring geeigneten Format vor und können so direkt weiterverwendet werden, z. B. Bilddateien in den Formaten JPEG oder PNG. Oftmals ist jedoch eine Konvertierung bzw. Encodierung von Daten notwendig. Je nach Art des Objektes und dessen geplanter Verwendung ist eine geeignete Form der Speicherung auszuwählen. Gerade bei den verschiedenen Arten von Videoobjekten ist hierbei sorgfältig vorzugehen, um die Flexibilität bei der Bearbeitung und die Qualität des Ergebnisses zu gewährleisten. Einige Arten von Medienobjekten erfordern die zusätzliche Verwaltung von Voransichten, um die Integration in die Anwendung zu erleichtern. Sinnvoll ist dies beispielsweise bei Videoobjekten, bei denen eine oder mehrere typische Einzelbilder mit abgelegt werden. Dies erleichtert zusammen mit aussagekräftigen Metadaten die Orientierung in großen Objektbeständen erheblich.
Beispiel 1: Geformtes Video kann dem Autorensystem auf verschiedene Arten übergeben werden. Ein Videoobjekt kann repräsentiert sein durch:

1. *Einzelbildfolgen (z. B. TGA-Sequenzen):* Das Videoobjekt besteht aus eine Folge von Einzelbildern, die als separate Dateien vorliegen. Dies erschwert die Verwaltung des Objektes erheblich. Die Anzeige von Voransichten ist hingegen sehr leicht realisierbar.

2. *uncodiertes Video (z. B. YUVA-Dateien):* Die Einzelbilder werden zu einer Datei zusammengefasst. Die Verwaltung des Objektes wird hierdurch vereinfacht. Voransichten müssen aus der Datei extrahiert werden. Der Autor kann die Encodierung des Videoobjektes beeinflussen.

3. *encodiertes Video (z. B. AVC):* Das Video wird mit einem speziellen Encoder encodiert. Das Objekt kann ggf. sehr speicherplatzsparend verwaltet werden. Voransichten müssen extrahiert werden. Bei der Encodierung müssen zahlreiche Parameter festgelegt werden, welche die Eigenschaften und die Qualität des Videoobjektes stark beeinflussen. Hierdurch wird eine flexible Nutzung des Objektes erschwert.

Eine Besonderheit stellen synthetische Objekte dar, die wiederum selbst auf einer Szenenbeschreibung beruhen. Diese können direkt in den Szenengraphen integriert werden und bleiben somit zu jeder Zeit bearbeitbar. Auch eine Speicherung in encodierter Form ist möglich.
Beispiel 2: Ein synthetisches 3D-Objekt wurde mit einem 3D-Werkzeug erstellt und in das VRML-Format exportiert.

1. *direkte Integration:* Die vorliegende VRML-Datei kann mit einem Konverter in das X3D-Format überführt werden und ist somit direkt in einen XMT-A-Szenengraphen integrierbar.

2. *encodierte Integration:* Die VRML-Datei wird in ein encodierbares BIFS-Text-Format überführt und einem MPEG-4-Encoder übergeben. Das encodierte Objekt kann als Inline-Knoten in die Szene integriert werden.

Dem Import-Modul müssen alle angesprochenen Konverter und Encoder zur Verfügung stehen. Diese sind in einer Bibliothek zusammengefasst und können von allen Komponenten des Autorensystems genutzt werden. Import-Module können sowohl Bestandteil von Autorenwerkzeugen als auch des Authoring-Servers sein.

Export
Ergebnisse und Arbeitsstände können auf verschiedene Arten abgelegt werden. Typisch für das Authoring ist die Speicherung der Szenenbeschreibung in einem Authoring-Format, da die Medienobjekte nicht bearbeitet werden. Für den Datenaustausch mit anderen Autoren oder Systemen müssen Projekte auch vollständig exportierbar sein. Dies umfasst neben der Szenenbeschreibung auch die Medienobjekte. Audiovisuelle Objekte, z. B. Videoobjekte können einen beachtlichen Bedarf an Speicherplatz verursachen. In solchen Fällen können Platzhalter oder vereinfachte Repräsentationen verwendet werden.

Szenengraph-Manager
Das eigentliche Authoring findet im Szenengraph-Manager (SGM) statt. Hier werden die Szenenbeschreibungen verwaltet und Änderungen durch die Editoren ausgeführt. Beginnend mit einer initialen Szene werden die Medienobjekte Schritt für Schritt dem Szenengraphen hinzugefügt. Zu jedem Objekt muss ggf. ein OD erzeugt werden. Die örtliche Positionierung erfolgt durch Transformationen, die mit einem geeigneten Editor festgelegt werden. Die Integration komplexer Objekte, wie z. B. beliebig geformter Videoobjekte oder 3D-Videoobjekte, erfolgt oft unter Nutzung von Platzhaltern in Form vereinfachter Darstellungen oder Voransichten. Für die grundlegende Bearbeitung eines Szenengraphen werden folgende Funktionen benötigt:

1. *Einfügen eines Knotens (INSERT)*
2. *Bearbeiten eines Knotens (UPDATE)*
3. *Löschen eines Knotens (DELETE)*

Im Falle einer gemeinsamen Bearbeitung durch mehrere Autoren werden Teile der Funktionalität des SGM auf den Authoring-Server verlagert. Insbesondere bei der synchronen Bearbeitung müssen alle Änderungen unverzüglich für alle Beteiligten verfügbar sein und die Autoren über alle Änderungen informiert werden. Dies kann zentral über den Server oder durch Kommunikation zwischen den Clients erfolgen. Der SGM wird durch eine Objekt-Repräsentation des Szenengraphen dargestellt und ist auch Grundlage für die Visualisierung der Szenendaten.

Editoren
Die Editoren ermöglichen die gezielte und effiziente Bearbeitung von Szenendaten. Sie sind an den SGM angebunden und bearbeiten dessen Daten. Durch diesen modularen Aufbau können bei Bedarf weitere Editoren in das System eingefügt werden. Ein Autorenwerkzeug beinhaltet meist mehrere Editoren.

Für die direkte Bearbeitung von Szenenbeschreibungen und Metadaten sind Quelltext-Editoren geeignet. Sie erfordern Expertenwissen vom Autor, weisen aber dafür die höchste Flexibilität auf. Erleichtert wird die Arbeit durch Funktionen wie z. B. Syntax-Highlighting oder Baumdarstellungen. Auch wenn Quelltext-Editoren nicht unmittelbar zu den grafisch-interaktiven Werkzeugen zählen, erweitern sie durch ihre Flexibilität die Möglichkeiten eines Autorensystems erheblich.

Typisch für Autorensysteme sind jedoch grafisch-interaktive Editoren. Sie verfolgen oft einen WYSIWYG-Ansatz oder bieten spezielle Sichten auf einzelne Eigenschaften der Szene. Es kommen bekannte Techniken, wie z. B. *Drag&Drop* zum Einsatz. Im vorgeschlagenen Autorensystem arbeiten alle Editoren auf einer gemeinsamen Datenbasis, die in einem Authoring-Format abgebildet wird.

4.3.8 Arbeitsgebiete

Die weitere Entwicklung des Autorensystems wurde in drei Arbeitsgebiete unterteilt:

1. *Authoring-Formate:* Je nach Anwendungsgebiet sind geeignete Authoring-Formate auszuwählen oder bei Bedarf zu entwickeln.

2. *Autorenwerkzeuge:* Es sind Konzepte und Architekturen für grafisch-interaktive Werkzeuge zu entwickeln, die den Autoren eine intuitive und effiziente Arbeit ermöglichen.

3. *Authoring-Server:* Für die Unterstützung der Zusammenarbeit mehrerer Autoren ist die Möglichkeit eines zentralen Datenmanagements zu entwickeln.

Der Aspekt des verteilten Authorings kommt in allen drei Aufgabenbereichen zum Tragen.

4.4 Authoring-Formate

In der Vergangenheit sind zahlreiche Datenformate für die Beschreibung multimedialer Anwendungen entstanden [HBS03]. Neben standardisierten Formaten (z. B. VRML) konnten sich vor allem einige proprietäre Formate (z. B. *Flash*) durchsetzen. Solche Datenformate waren Gegenstand mehrerer Untersuchungen. In [CD02] werden beispielsweise mehrere Multimedia-Formate vorgestellt und verglichen: VRML, BIFS (Version 4), SMIL, SVG, *Flash* (*.swf) und *QuickTime*.

4.4.1 Anforderungen

Autoren und Nutzer multimedialer Anwendungen haben sehr verschiedene Anforderungen an Datenformate. Während der Autor beispielsweise großen Wert auf verlustlose Bearbeitbarkeit, Zusatzinformationen und Semantik legt, ist für den Nutzer vor allem eine effektive Komprimierung und schnelle Darstellung von Bedeutung. Tabelle 4.1 zeigt einige typische Anforderungen und Formate.

Diese Diskrepanz ist praktisch nicht zu überbrücken. Daher liegt es nahe, für Authoring und Distribution verschiedene Datenformate zu verwenden. Es erfolgt eine Trennung in *Authoring-Format* und *Distributions-Format*. Die möglichen Distributions-Formate für MPEG-4-Anwendungen sind Teil des MPEG-4-Standards und sollen hier nicht diskutiert werden. Bei der Entwicklung von MPEG-4 konzentrierte man sich zunächst auf eine effektive Beschreibung und Codierung multimedialer Inhalte. Die zu Beginn verwendeten textuellen Szenenbeschreibungssprachen wurden nie standardisiert und sind abhängig vom verwendeten Encoder. Erst mit der Vorstellung von *XMT* (siehe Unterabschnitt 2.4.3) ist eine standardisierte Beschreibung möglich. Durch die XML-basierte Beschreibung der Inhalte ist eine gute Interoperabilität mit anderen XML-basierten Formaten gegeben.

Anforderung	Autor	Nutzer
Bearbeitbarkeit	vollständig	keine
Komprimierung	Kompromiss	hohe
Metadaten	alle	einige
Hardwareanforderungen	evtl. hoch	möglichst gering
Datenformate:		
Bilder	TIFF, PSD	JPEG, PNG
Audio	AIFF, WAV	mp3, AAC
Video	DV, M-JPEG	MPEG-2, DivX

Tabelle 4.1: Anforderungen an Datenformate

Ein Authoring-Format ermöglicht die Beschreibung aller relevanten Informationen der zu erstellenden Anwendung. Die Medienobjekte werden als Verweise entsprechend dem ODF (siehe Unterabschnitt 2.4.1) eingefügt. Alle notwendigen Zusatzinformationen (z. B. Objektbeschreibungen und Namen) werden als Metadaten beschrieben. Die Aufgaben eines Authoring-Formates sind:

- Speicherung der Szenenbeschreibungen
- Speicherung aller Informationen über Medienobjekte
- Speicherung von weiteren Metadaten

Anforderungen an die Form eines Authoring-Formates:

- *Lesbarkeit und Bearbeitbarkeit:* Das Authoring-Format sollte auf Quelltext-Ebene bearbeitbar sein, um dem Autor auch ohne grafische Werkzeuge die Erstellung von Anwendungen zu ermöglichen.
- *Mehrbenutzerfähigkeit:* Mehrere Autoren sollten bereits auf Dateiebene sinnvoll zusammenarbeiten können. Dafür sind beispielsweise Mechanismen zur Freigabe oder Sperrung einzelner Elemente erforderlich, um ein synchrones Editieren einer Szenenbeschreibung durch mehrere Autoren zu ermöglichen. Das Datenformat muss klar strukturiert sein.
- *Erweiterbarkeit:* Neue Anforderungen und Funktionen sollten leicht integrierbar sein, ohne das Format von Grund auf neu entwickeln zu müssen.
- *Plattformunabhängigkeit:* Das Format muss unabhängig von Hard- und Software sein.

Die Authoring-Formate sollen einerseits die Grundlage eines Authorings auf Basis grafisch-interaktiver Autorenwerkzeuge sein, andererseits soll die Erstellung einer AV-Anwendung auch auf Quelltext-Ebene ermöglicht werden. Dies gilt sowohl für den Fall des Authorings durch nur einen Autor als auch für ein auf mehrere Autoren verteiltes Authoring. Weiterhin ist die Interoperabilität mit anderen Beschreibungsformaten zu berücksichtigen, um einen Datenaustausch zwischen Autorenwerkzeugen zu ermöglichen. Schließlich ist die Verarbeitung der Authoring-Formate möglichst einfach zu halten.

4.4.2 Abstraktionsebenen

Um eine intuitive Erstellung einer Anwendung zu ermöglichen, müssen alle Beschreibungen auf einer der jeweiligen Aufgabe angepassten Abstraktionsebene erfolgen. Bereits die im MPEG-4-Standard bereitgestellten Werkzeuge ermöglichen die Arbeit auf verschiedenen Abstraktionsebenen bzw. semantischen Niveaus (siehe Abbildung 2.11):

1. Auf der untersten Ebene befindet sich das encodierte MPEG-4-Dateiformat (*MP4*), welches eine effiziente Codierung und Übertragbarkeit ermöglicht. In diesem binären Format werden lediglich die für die Beschreibung der Szene erforderlichen technisch notwendigen Zusammenhänge gespeichert. Weitere Informationen über den Inhalt (Semantik) sind nicht enthalten. Eine manuelle Bearbeitung zum Zweck des Authorings ist auf dieser Ebene praktisch nicht sinnvoll.

2. Auf der nächst höheren Ebene ist das Ausgangsmaterial vor der Encodierung angesiedelt. Die Szenenbeschreibungen liegen in einem textuellen Format vor. Die Beschreibung der Inhalte orientiert sich auch hier stark an technischen Kriterien. Das Authoring auf dieser Ebene ist möglich, aber es erfordert spezielle Kenntnisse des MPEG-4-Standards.

3. Eine Zwischenstufe bildet die Metaebene XMT-C. Sie stellt das Bindeglied zur nächst höheren Ebene dar.

4. Die höchste im MPEG-4-Standard definierte Abstraktionsebene wird durch XMT-Ω repräsentiert. Diese Ebene umfasst ausgeprägte, universell einsetzbare Authoring-Funktionen und stellt damit ein Authoring-Format im eigentlichen Sinne dar. Vor der Encodierung erfolgt eine Überführung auf die Ebene von XMT-A bzw. BT. Die Semantik der XMT-Ω-Elemente geht hierbei verloren. Diese Überführung ist ohne Informationsverlust nicht umkehrbar.

Naturgemäß sind die Beschreibungen auf niedriger Abstraktionsebene am flexibelsten einsetzbar. Die Definition eines gleichermaßen flexiblen Formates auf hoher Abstraktionsebene müsste deutlich umfangreicher ausfallen und wäre damit kaum beherrschbar. Eine nutzerfreundliche, intuitive Verwendung ist dann nicht mehr vorstellbar. Um die Vorzüge der verschiedenen Abstraktionsebenen zu nutzen, können diese in Kombination verwendet werden.

4.4.3 Entwurf von Formaten

Das Ziel neu zu schaffender Authoring-Formate ist die Bereitstellung angepasster Authoring-Funktionen für definiert abgegrenzte Anwendungen. Sie erweitern und kombinieren vorhandene Formate und weisen ein hohes Abstraktionsniveau auf. Abbildung 4.5 zeigt eine mögliche Einordnung eines neuen Formates. Weiterhin sind einige korrespondierende Im- und Exportformate dargestellt.

Zunächst sind der Inhalt und der Umfang des neuen Formates abzugrenzen. Dafür müssen die zu schaffenden Applikationen genau analysiert werden um zu ermitteln, welche Funktionen für die zu schaffenden Anwendungen typisch sind. Anschließend werden bestehende Formate auf ihre Eignung hin untersucht.

Ausgangspunkt bei der Schaffung neuer Authoring-Formate für objektbasierte AV-Anwendungen ist das Format XMT-Ω. Viele der verwendeten Funktionen wurden von SMIL übernommen. Der Funktionsumfang von XMT-Ω ist in folgende Module aufgeteilt:

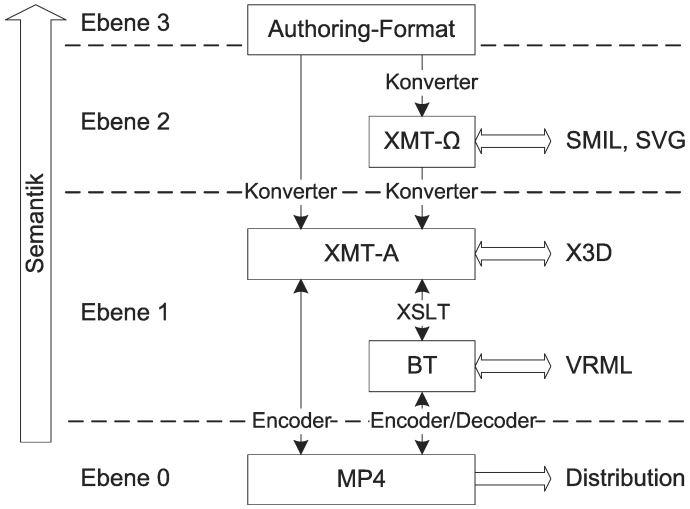

Abbildung 4.5: Semantische Ebenen und Verarbeitung

Animation, Content Control, Layout, Linking, Media, Metadaten, Zeitsteuerung, Synchronisation, Zeitmanipulation und Blenden.

In den meisten Fällen wird nur eine Teilmenge der angebotenen Funktionen benötigt, die durch neu definierte Funktionen ergänzt werden. Diese können aus XMT-Ω bzw. SMIL-Anweisungen bestehen oder Funktionen aus semantisch tiefer liegenden Beschreibungen (z. B. XMT-A oder X3D) verwenden. Auf diese Weise lässt sich eine spezialisierte Authoring-Sprache für konkrete Anwendungen formulieren. Durch die nun nicht mehr geforderte Universalität kann das Format im Vergleich zu XMT-Ω deutlich kompakter gestaltet werden und dem Autor sehr spezielle Funktionen anbieten.

Die Formulierung eines neuen Authoring-Formates erfolgt standardisiert in Form von XML-Schemadefinitionen. Die Einführung zusätzlicher Elemente und Attribute macht es notwendig, auch einen eigenen Namensraum zu definieren, um die neuen Elemente und Attribute von XMT-Ω unterscheiden zu können. In Kapitel 6 sind drei auf dieser Basis realisierte Authoring-Formate beschrieben.

4.4.4 Verarbeitung

Die konsequente Verwendung XML-basierter Formate ermöglicht die Nutzung zahlreicher XML-Werkzeuge für die Verarbeitung. Die Realisierung der kompletten Verarbeitungskette – vom neuen Format bis hin zu einer encodierten MPEG-4-Anwendung – wird dadurch erheblich vereinfacht. Die typischen Schritte bei der Verarbeitung von XML-Dokumenten innerhalb des Authoring-Prozesses sind:

- Einlesen des XML-Dokumentes
- Parsen der Struktur des Dokumentes
- Bereitstellung der Inhalte in strukturierter Form, z. B. als Objekt-Modell
- Bearbeitung der Inhalte
- Ausgabe der Inhalte in der gewünschten Form

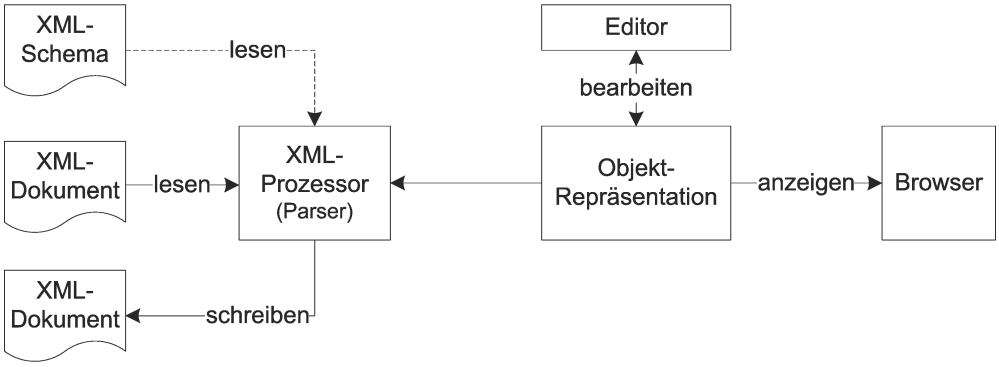

Abbildung 4.6: Typische Verarbeitungskette für XML-Dokumente

Abbildung 4.6 zeigt den typischen Ablauf einer XML-Verarbeitung. Das XML-Dokument und ggf. das XML-Schema werden vom XML-Prozessor eingelesen. Der Parser analysiert und zerlegt den Inhalt entsprechend den Erfordernissen der Objekt-Repräsentation. Diese stellt den Inhalt in Form von Objekten zur Verfügung, auf die lesend und schreibend zugegriffen werden kann. Auf dieser Basis können die Daten direkt bearbeitet werden. Im vorgeschlagenen Autorensystem werden Parser, Schnittstellen und Objekt-Repräsentation zu einem *Data Interface (DI)* zusammengefasst. Alle Bearbeitungen werden auf Basis der Objekt-Repräsentation auf der Abstraktionsebene des Authoring-Formates durchgeführt. Zur Überführung der Daten in Formate auf anderen Abstraktionsebenen werden Konverter benötigt. Diese führen die in Abbildung 4.5 dargestellten Konvertierungen durch. Abbildung 4.7 zeigt die Verarbeitung eines Authoring-Formates. Das *Data Interface* kann

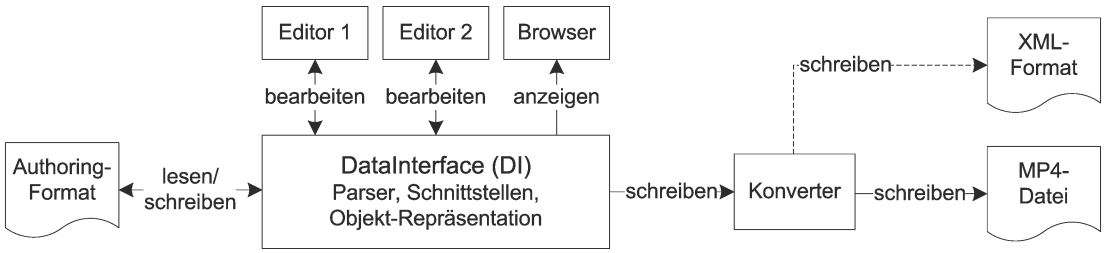

Abbildung 4.7: Bearbeitung einer Szene auf Basis des Authoring-Formates

je nach Anwendung auf verschiedenen Abstraktionsebenen arbeiten. Für die Erzeugung einer MPEG-4-Anwendung im binären MP4-Format müssen dem Konverter zusätzlich die Medienobjekte übergeben werden. Der Konverter enthält auch einen MPEG-4-Encoder, z. B. *mp4tool*. Diese dateibasierte Verarbeitung wird verwendet, wenn die aktive Bearbeitung auf nur einer Abstraktionsebene erfolgt.

Erfolgt die Bearbeitung der Szene hingegen parallel auf verschiedenen Abstraktionsebenen, müssen die Szenengraphen auf allen Ebenen durch Objekt-Repräsentationen abgebildet werden. Die Synchronisation der Szenengraphen übernimmt ein *Mediator*. Bei jeder Änderung eines Szenengraphen wird eine korrespondierende Änderung des anderen Szenengraphen vorgenommen. Abbildung 4.8 zeigt dies an einem Beispiel. Die beiden beschriebenen Arten der Verarbeitung von Authoring-Formaten bilden die Grundlage für die Entwicklung von Autorenwerkzeugen.

Eine weit verbreitete Objekt-Repräsentation ist das *Document Object Model (DOM)* [W3Ca]. Das DOM ist eine vom W3C definierte Programmierschnittstelle für die objekt-

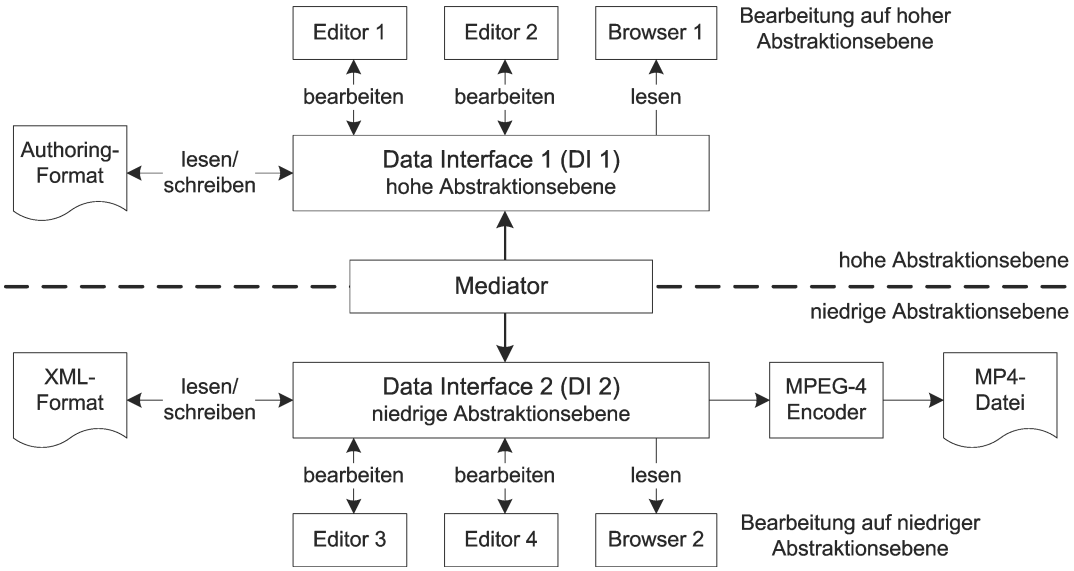

Abbildung 4.8: Bearbeitung einer Szene auf verschiedenen Abstraktionsebenen

basierte Repräsentation strukturierter Daten (z. B. HTML- oder XML-Dokumente). Es sind Implementierungen für verschiedene Programmiersprachen (z. B. C++ und Java) verfügbar. Eine andere Möglichkeit einer Objekt-Repräsentation von XML-Dokumenten ist das *Data Binding* [Bou].

4.5 Autorenwerkzeuge

Die Autorenwerkzeuge sind die Schnittstellen zwischen den Autoren und den Szenendaten. Je nach Aufgabenstellung sind hier viele Ausprägungen denkbar (siehe Abschnitt 3.4). Im Fokus stehen grafisch-interaktive Werkzeuge zur Unterstützung eines intuitiven Arbeitens während des Authoring-Prozesses. Autorenwerkzeuge müssen die zugrunde liegenden technischen Zusammenhänge der AV-Anwendung möglichst weitgehend vor dem Autor verbergen.

4.5.1 Anforderungen

Ein Autorenwerkzeug soll dem Autor die Bearbeitung folgender Arbeitsschritte ermöglichen:

- Import von Medienobjekten
- räumliche und zeitliche Komposition der Medienobjekte
- Erstellung von Interaktionsmöglichkeiten
- Bearbeitung von Metadaten
- Bereitstellung von Voransichten
- Erzeugung und Erprobung einer encodierten Anwendung
- Organisation des Authoring-Prozesses

- Im- und Export von Szenendaten auf Basis eines Authoring-Formates

Eine weitere Spezifizierung ist erst mit der Eingrenzung des Anwendungsgebietes sinnvoll.

4.5.2 Elemente

Die Konzeption der grafischen Benutzeroberfläche orientiert sich zunächst an bekannten Systemen und übernimmt bewährte Prinzipien und Elemente. Nach [Shn02] sollten folgende Punkte beim Entwicklungsprozess beachtet werden:

- fortlaufende Darstellung interessanter Objekte, Aktionen und Werkzeuge mit bedeutungsvollen visuellen Metaphern
- intuitiver und schneller Zugriff auf wichtige Elemente
- schnelle, inkrementelle Operationen, deren Auswirkungen sofort sichtbar sind

Im Folgenden werden Elemente und Funktionen beschrieben, die bei den meisten Autorenwerkzeugen benötigt werden. Die Betrachtungen setzen als Authoring-Format eine Beschreibungssprache wie XMT-Ω voraus (siehe Abschnitt 4.4). Neben den grafisch-interaktiven Editoren soll auch eine Bearbeitung des Authoring-Formates auf Quelltext-Ebene möglich sein, um die Flexibilität der Autorenwerkzeuge weiter zu erhöhen.

Bühne

Die *Bühne* ist das zentrale Element der meisten Autorenwerkzeuge. Sie erlaubt die grafisch-interaktive Bearbeitung einer audiovisuellen Anwendung und setzt ansatzweise das WYSIWYG-Prinzip um. Der Autor bekommt damit bereits während der Bearbeitung einen visuellen und auditiven Eindruck von der zu erstellenden Anwendung. Hierfür ist eine schnelle Darstellung der Szene in adäquater Qualität notwendig. Auf der Bühne erfolgt das Einfügen, das Positionieren und das Verknüpfen der einzelnen Elemente. Der Szenengraph wird auf diese Weise grafisch-interaktiv manipuliert. Auditive Elemente werden durch grafische Symbole repräsentiert und können somit ebenfalls grafisch-interaktiv positioniert und angehört werden. Je nach Anwendungsfall kann die Bühne eine zweidimensionale Fläche oder die Projektion eines dreidimensionalen Raumes sein.

Editoren

Die Editoren dienen der gezielten Bearbeitung von Elementen und deren Eigenschaften. Sie stellen Sichten auf das Datenmodell dar und verbergen dessen Komplexität vor dem Autor. Die Darstellung und Bearbeitung kann auf verschiedenen Abstraktionsebenen erfolgen. Im Folgenden werden einige häufig verwendete Editoren vorgestellt.

Struktur-Editor: Der Struktur-Editor visualisiert die hierarchische Struktur der Szene und macht damit die Beziehungen innerhalb des Szenengraphen sichtbar. Er ermöglicht eine anschauliche Darstellung und interaktive Bearbeitung der Strukturen einer Szene und zeigt alle Elemente an. Zur übersichtlichen Darstellung werden lediglich die Knoten angezeigt, die für die logische Struktur notwendig sind (Gruppenknoten und Objektknoten). Die zugehörigen Attribute werden in dafür vorgesehenen Editoren (Eigenschaften-Editor, Zeitachsen-Editor usw.) dargestellt. Abbildung 4.9 zeigt den Entwurf eines Struktur-Editors [Zie04].

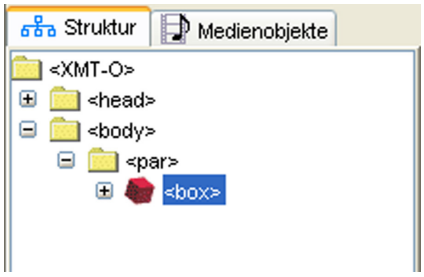

Abbildung 4.9: Struktur-Editor

Zeitachsen-Editor: Für die Bearbeitung der zeitlichen Zusammenhänge bietet sich die bekannte Zeitachsen-Metapher an (siehe Unterabschnitt 3.4.3). Es werden *globale* und *lokale* Zeitabläufe unterschieden. Eine Anwendung enthält stets einen globalen und optional mehrere lokale Zeitstränge, die visuell in lokalen und globalen Zeitachsen dargestellt werden. Diese geben einen schnellen Überblick, welche Objekte zu welcher Zeit aktiv sind. Als zusätzliche Funktion sollte die Zeitachse das Animieren von Element-Eigenschaften über Schlüsselwerte unterstützen. Abbildung 4.10 zeigt den Entwurf einer Zeitachse [Zie04].

In vielen Autorenwerkzeugen ist die globale Zeitachse die vorherrschende Metapher für die zeitliche Komposition. Im Falle interaktiver dreidimensionaler Anwendungen erscheint dies als weniger sinnvoll, da die Mehrzahl der Objekte dauerhaft in der Szene vorhanden ist. Lokale Zeitachsen bieten eine sinnvolle Möglichkeit für die Darstellung des zeitlichen Verlaufes innerhalb einzelner Hierarchie-Ebenen der Szene oder einzelner Medienobjekte. Die Auswahl der anzuzeigenden Bereiche erfolgt über den Struktur-Editor. Die Zeitachse passt sich an die aktuelle Auswahl an und kann somit sowohl lokale Beziehungen zwischen Objekten als auch Animationen übersichtlich darstellen. Bei der Erstellung von Animationen soll der Autor möglichst schon während der Erstellungsphase einen Eindruck vom Resultat bekommen. Hierzu wird ein verschiebbarer Positionszeiger in der Zeitachse benötigt, dessen Position den Zustand des Objektes auf der Bühne steuert. Um Schlüsselwerte von Animationen verändern zu können, ist die Zeitachse mit der Bühne und dem Eigenschaften-Editor verbunden. Der Autor wählt zunächst die zu animierende Eigenschaft eines Objektes aus. Daraufhin erscheint ein Zeitstrang zur Platzierung von Schlüsselbildern für die gewählte Eigenschaft. Nach Auswahl eines Schlüsselbildes können dessen Werte im Eigenschaften-Editor oder auf der Bühne definiert werden.

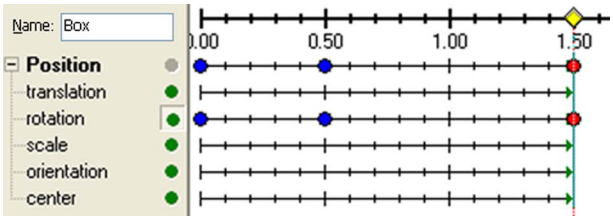

Abbildung 4.10: Entwurf einer Zeitachse

Interaktions-Editor: Gerade die Bearbeitung von komplexen Interaktionsmöglichkeiten ist eine sehr anspruchsvolle Aufgabe. Die textuelle Definition von komplexen Interaktionen ist für einen Autor nicht praktikabel. Der Seiten-basierte Authoring-Ansatz (siehe Unterabschnitt 3.4.3) ist für die Entwicklung komplexer Anwendungen zu unübersichtlich, doch kann er für die Festlegung von Interaktionen nützlich sein. Wie schon im Falle

des Zeitachsen-Editors soll es möglich sein, Interaktions-Beziehungen nur für bestimmte Teilbereiche zu berücksichtigen, wodurch die Interaktionsdiagramme in ihrer Komplexität skalierbar werden. Ist beispielsweise ein bestimmtes Objekt im Struktur-Editor oder auf der Bühne ausgewählt, werden nur diejenigen Objekte angezeigt, die mit dem Objekt in einer direkten Interaktions-Beziehung stehen. Sollen weitläufigere Zusammenhänge visualisiert werden, können mit Hilfe des Struktur-Editors auch Gruppen-Elemente (bis hin zur ganzen Szene) ausgewählt werden. Die Interaktions-Beziehungen werden durch ein Quellobjekt, einen Auslöser (ein Ereignis), ein Zielobjekt und ein Kommando beschrieben. Diese können mit Hilfe von Drop-Down-Menüs festgelegt werden und erscheinen dann als grafische Repräsentation. Die Verwendung von Drop-Down-Menüs stellte sich bereits bei *Envivio Broadcast Studio* (siehe Unterabschnitt 3.5.1) als eine sehr intuitiv zu handhabende Möglichkeit dar, Interaktionen zu definieren. Eine andere Möglichkeit bietet das Drag&Drop von Interaktionsobjekten und -beziehungen. Objekte werden dabei aus dem Struktur-Editor oder der Bühne in den Interaktions-Editor gezogen und dort mit Hilfe von Pfeilen verbunden. Die Pfeilenden wiederum enthalten Auswahlmöglichkeiten für den Auslöser und das Resultat der Interaktion (siehe Abbildung 4.11).

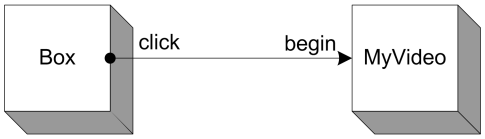

Abbildung 4.11: Darstellung einer einfachen Interaktions-Beziehung

Quelltext-Editor: Quelltext-Editoren dienen der direkten Bearbeitung des Authoring-Formates. Die Editoren können alternativ zur Bühne dargestellt werden. Somit hat der Autor stets die volle Kontrolle über die Szene, unabhängig von der Leistungsfähigkeit des grafischen Autorenwerkzeuges. Neben der reinen Textdarstellung sind auch Baumdarstellungen des Szenengraphen sinnvoll.

Eigenschaften-Editor: In Eigenschaften-Editoren kann der Autor die Eigenschaften eines ausgewählten Objektes direkt bearbeiten. Die Darstellung und Bearbeitung der Parameter kann textuell (z. B. in einer Tabelle mit editierbaren Feldern) oder grafisch-interaktiv erfolgen.

Metadaten-Editor: Die Zuweisung von Metadaten zu Objekten und Szenenelementen erfolgt auf Basis von MPEG-7-Elementen bzw. XMT-C-Elementen. Die Darstellung und Bearbeitung von Metadaten wird in einfachen Dialogboxen oder in einer XML-Darstellung vorgenommen.

Medienobjekt-Bibliothek
In der Medienobjekt-Bibliothek werden alle Mediendateien gruppiert nach ihrer Art namentlich aufgelistet. Neben Bild-, Audio- und Videoobjekten zählen hierzu beispielsweise auch bereits encodierte MPEG-4-Szenen und integrierte XMT-A-Szenen. Ergänzt wird die Medienobjekt-Bibliothek durch ein Vorschau-Modul für die übersichtliche Darstellung der Mediendateien. Je nach Art des Objektes werden hierfür separat gespeicherte Voransichten verwendet.

Externe Elemente

Während die Editoren verschiedene Sichten auf das Datenmodell ermöglichen, soll ein zweiter Bereich der Nutzerschnittstelle den externen Bereich darstellen. Zum externen Bereich gehören häufig benötigte Werkzeuge, die keine direkte Beziehung zum Datenmodell haben. Zum einen sind dies Browser für Elemente auf dem Authoring-Server oder des lokalen Dateisystems, zum anderen eine Bibliothek mit vorgefertigten Elementen.

Die beiden Browser haben ein einheitliches Erscheinungsbild, da es für den Nutzer unerheblich ist, ob er auf das lokale Dateisystem oder auf den Authoring-Server zugreift. Sowohl Dateisystem als auch Server halten die Dateien in hierarchischer Form. Die Szenen selbst liegen ebenfalls in einer hierarchischen Struktur vor. Somit eignen sich Baumstrukturen für die Darstellung von Dateien und Daten in den Browsern. Verbunden mit den Browsern sind Vorschau-Möglichkeiten für die Objekte und deren Metadaten. Für die Vorschau der Objekte sind geeignete Betrachter zu integrieren; die Metadaten können beispielsweise in Tabellen dargestellt werden. Die programminterne Bibliothek enthält vordefinierte Objekte, die mit Hilfe von Assistenten oder dem Eigenschaften-Editor angepasst werden. Dies können beispielsweise Standardobjekte (z. B. 3D-Primitive und Lichtquellen), Platzhalter für Medienobjekte (z. B. Videos und Bilder) oder Kontrollfunktionen (z. B. für Animationen oder Interaktionen) sein. Der Autor soll die Bibliothek auch um komplexe oder häufig benötigte Elemente erweitern können. Abbildung 4.12) zeigt einen Entwurf für eine interne Bibliothek [Zie04].

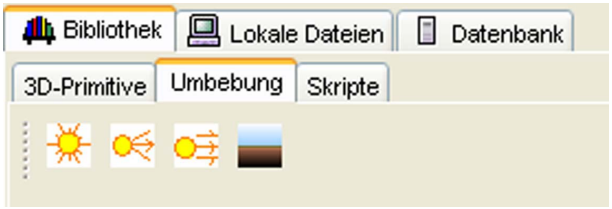

Abbildung 4.12: Entwurf einer internen Bibliothek

4.5.3 Entwurfsmuster

Entwurfsmuster (engl. design pattern) sind schematisierte Lösungen, mit denen wiederkehrende Aufgaben gelöst werden können. Auf dem Gebiet der Softwareentwicklung sind zahlreiche solcher Lösungen zu finden. Diese erleichtern die Entwicklung von Applikationen erheblich. Im Folgenden wird eine Architektur vorgestellt, auf deren Basis Autorenwerkzeuge für die verschiedensten Anwendungen umgesetzt werden können. Zwei Beispiele finden sich in den Unterabschnitten 6.1.4 und Abschnitt 6.3. Grundlage bilden Authoring-Formate auf Basis von XMT (siehe Abschnitt 4.4). Die Autorenwerkzeuge sollen sowohl autark (ohne Authoring-Server) als auch in einem verteilten Autorensystem (mit Authoring-Server) verwendet werden können. Daher ist eine optionale Serveranbindung vorzusehen.

Die geforderte hohe Flexibilität der Architektur lässt sich durch einen konsequent modularen Aufbau erreichen. Der Funktionsumfang ist damit leicht erweiterbar. Dabei sollen durchaus auch vorhandene und bewährte Komponenten zum Einsatz kommen. Um Elemente der Benutzerschnittstelle (z. B. Editoren und Betrachter) austauschen zu können, müssen Darstellung und Verarbeitung getrennt werden. Ein in der Softwaretechnik weit

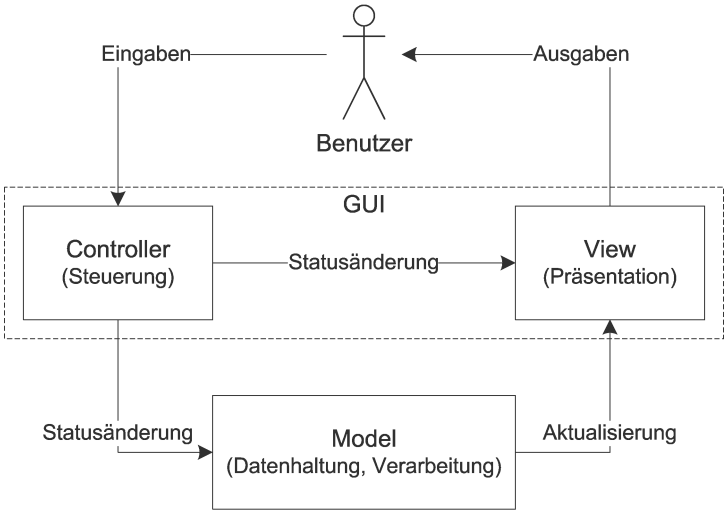

Abbildung 4.13: MVC-Architektur

verbreiteter Ansatz zur modularen Softwareentwicklung ist das *Model-View-Controller-Konzept (MVC)* nach Trygve M. H. Reenskaug [Ree]. Das MVC-Konzept wurde ursprünglich für Benutzeroberflächen in *Smalltalk* entwickelt. Mittlerweile ist es zu einem Quasi-Standard für komplexe Softwareprojekte geworden. Die Teile einer Software werden in Module getrennt, die in drei Gruppen eingeteilt sind:

1. *Model:* Das *Model* dient der internen Repräsentation der Anwendungsdaten (Datenschicht) und beinhaltet Methoden zum Abfragen und Ändern dieser Daten. Eine Änderung der Daten löst eine Benachrichtigung der *Views* aus, welche am *Model* als unabhängige Objekte registriert sind.

2. *View:* Der *View* präsentiert die Daten (Präsentationsschicht). Er ist beim *Model* registriert und aktualisiert sich automatisch nach einer Änderung in der Datenschicht. Ein Model kann durch mehrere Views präsentiert werden.

3. *Controller:* Der *Controller* beinhaltet die Logik zur Eingabe von Daten (Steuerungsschicht). Er empfängt Signale (z. B. Maus- oder Tastatur-Ereignisse und Ereignisse anderer Controller) und entscheidet über die weitere Verarbeitung.

Abbildung 4.13 zeigt die grundlegende Architektur des MVC-Konzeptes. Im Falle des Autorensystems bedeutet dies, dass die Szenendaten im *Model* gehalten und *View* sowie *Controller* durch die Benutzeroberfläche dargestellt werden. Eine strenge Unterteilung in *View* und *Controller* eines Oberflächenelementes ist nicht immer notwendig, da die grafischen Elemente und deren Steuerung meist eng miteinander verbunden sind [Krü02]. Für die Komponenten des Autorensystems kann an vielen Stellen diese vereinfachte Form der MVC-Struktur verwendet werden. Die Unterscheidung zwischen View und Controller entfällt dann, sodass sich die Architektur auf eine reine Model und View/Controller (View) Beziehung reduziert. Diese Struktur senkt den Implementierungsaufwand erheblich und kann häufig ohne merkliche Nachteile angewendet werden. Abbildung 4.14 zeigt die vereinfachte Architektur mit mehreren View/Controller-Modulen.

Die Autorenwerkzeuge werden auf Grundlage der in Unterabschnitt 4.4.4 vorgeschlagenen Architektur zur Bearbeitung von Szenen auf Basis eines Authoring-Formates (siehe Abbildung 4.7) entworfen. Das dort beschriebene *Data Interface* enthält u. a. eine

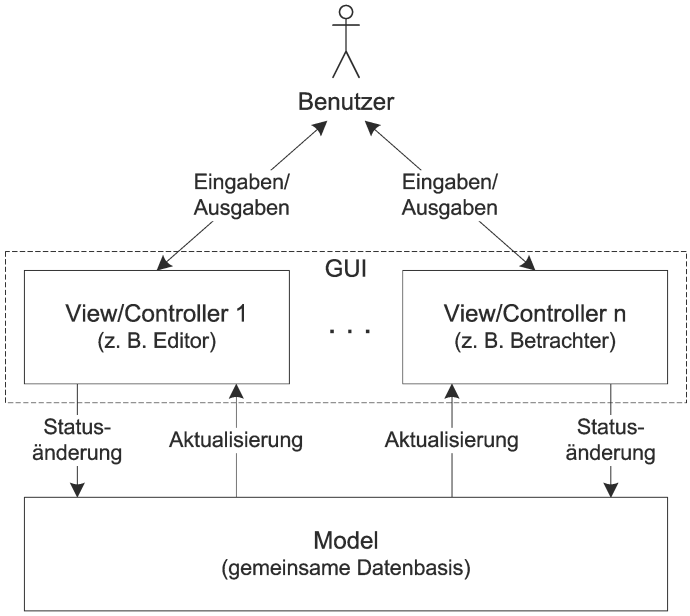

Abbildung 4.14: vereinfachte MVC-Architektur mit mehreren View/Controller-Modulen

Objekt-Repräsentation der Szenenbeschreibung. Aus dieser wird auch das *Model* der MVC-Architektur gebildet. Die Editoren werden durch View/Controller-Module dargestellt, mit denen die Daten des *Data Interface* bearbeitet werden.

4.5.4 Visualisierung

Eine grafisch-interaktive Bearbeitung multimedialer Anwendungen erfordert eine angemessene Visualisierung der Szenendaten zu verschiedenen Zwecken. Hierbei sind drei Fälle zu unterscheiden:

1. *statische Visualisierung:* Visualisierung eines Zustandes bzw. Zeitpunktes der Anwendung

2. *interaktive Visualisierung:* Visualisierung der funktionsfähigen Anwendung mit Interaktionsmöglichkeiten, aber ohne Bearbeitungsmöglichkeit

3. *interaktive Visualisierung mit Bearbeitungsmöglichkeit:* Visualisierung der funktionsfähigen Anwendung mit Interaktionsmöglichkeiten und mit der Möglichkeit der Bearbeitung

Nicht immer steht dabei die beste visuelle Qualität im Vordergrund; häufig genügt eine aussagekräftige Darstellung des erreichten Arbeitsstandes. Komplexe Objekte und Szenen können oft durch vereinfachte Darstellungen oder Platzhalter ersetzt werden, um die Darstellungsgeschwindigkeit oder die Übersichtlichkeit zu verbessern. Aus dem Gebiet Computergrafik sind verschiedene Darstellungsformen mit reduzierter visueller Qualität bekannt, z. B. das Drahtgitter-Modell und die einfach schattierte Schnellansicht (*Flatshading*). Insbesondere bei Visualisierungen mit Bearbeitungsmöglichkeit können solche reduzierten Darstellungen vorteilhaft sein. Grundsätzlich sind zwei Möglichkeiten der Visualisierung denkbar, die auch in Kombination angewendet werden können:

1. *Visualisierung auf Basis des Distributionsformates:* Die Anwendung muss für die Ansicht vollständig encodiert werden und liegt dann im Distributionsformat vor. Dies kann bei umfangreichen Anwendungen zu erheblichen Verzögerungen führen. Für die Darstellung kann jeder MPEG-4-Player mit entsprechender Leistungsfähigkeit genutzt werden. Damit entspricht die Darstellung dem später zu erwartenden Ergebnis.

2. *Visualisierung auf Basis des Authoring-Formates:* Die Ansicht wird aus dem Authoring-Format ohne vorherige Encodierung erzeugt. Für die Darstellung werden speziell entwickelte Betrachter eingesetzt. Die Darstellung entspricht dabei nicht zwingend exakt dem später zu erwartenden Ergebnis.

Die Voransicht kann auf Basis der lokalen oder der zentralen Szenendaten erfolgen. Im Falle der synchronen Bearbeitung einer Anwendung durch mehrere Autoren sind insbesondere die Arbeitsstände der Beteiligten zu beachten. Für den Einsatz im Autorensystem sind aus technischer Sicht mehrerer Ansätze für die Visualisierung von Szenendaten möglich. Im Folgenden werden drei dieser Ansätze kurz diskutiert.

Visualisierung mit Low-Level-Grafik-APIs

Ein Low-Level-Grafik-API (z. B. *OpenGL*) stellt Hardware-nahe Funktionen zur Visualisierung bereit und orientiert sich dabei stark an den Erfordernissen des Rendering-Prozesses. Die geometrische Modellierung von Objekten ist daher mitunter sehr aufwändig. *OpenGL* kapselt beispielsweise lediglich einige Basisprimitive (z. B. Punkt, Linie, Bitmap und Polygone) und unterstützt geometrische Transformationen. Eine Unterstützung von Eingabegeräten ist nicht vorhanden und muss vom Entwickler implementiert werden. Ebenso fehlen Mechanismen zur Organisation der Szene in einem Szenengraphen. Für die Visualisierung einer MPEG-4-Szene muss für jeden BIFS-Knoten eine entsprechende Klasse erzeugt werden, die den Knoten auf Basis von Polygonen in *OpenGL* modelliert. Hierfür sind Regeln zur Überführung des lokalen Szenengraphen und Mapping-Funktionen von XMT-A auf die entsprechenden Klassen notwendig. Um eine Interaktion des Autors mit Medienobjekten zu realisieren, müssen eigene Routinen für Maus- und Tastatur-Ereignisse entwickelt werden. Die Entwicklung eines Betrachters auf Low-Level-Basis ist damit sehr aufwändig.

Visualisierung mit High-Level-Grafik-APIs

Auf einer höheren Abstraktionsebene arbeiten High-Level-APIs, z. B. *OpenInventor* oder *Java3D*. Sie enthalten geometrische Basis-Objekte, Möglichkeiten zur Organisation von Szenen in Szenengraphen und Interaktionsmöglichkeiten über verschiedene Eingabegeräte. Auch hier ist die Definition von Mapping-Funktionen vom lokalen Szenengraphen nach *Java3D* notwendig.

Visualisierung mit einem MPEG-4-Terminal

Nahe liegend erscheint die Nutzung eines MPEG-4-Terminals zur Visualisierung einer multimedialen Szene. Hierfür muss die gesamte Anwendung zunächst nach MPEG-4 encodiert werden, was bei umfangreichen Szenen mit einem erheblichen Zeitaufwand verbunden sein kann. Das MPEG-4-Terminal decodiert die Szene wieder und stellt sie in der gleichen Weise dar, wie dies auch später das Wiedergabegerät des Konsumenten tun wird. Somit ist eine Beurteilung des zu erwartenden Ergebnisses möglich. Um die vollständige Encodierung der Szene bei kleinen Änderungen zu vermeiden, können die Mechanismen des MPEG-4-Standards für die Änderung des Szenengraphen zur Laufzeit genutzt werden

(siehe Unterabschnitt 2.4.4). Hierfür ist eine Encodierung und Übertragung der *BIFS-Commands* mit geringer Verzögerung erforderlich.

Fazit

Die Verwendung eines Low-Level-API bietet die höchste Flexibilität bei der Implementierung und lässt gute Ergebnisse bei der Visualisierung erwarten. Auf Grund der niedrigen Abstraktionsebene der Beschreibung ist dieser Ansatz mit dem höchsten Implementierungsaufwand verbunden. Daher wird dieser Weg bei der Umsetzung des Autorensystems nicht weiterverfolgt. Bei Verwendung eines High-Level-API können viele vordefinierte Funktionen genutzt werden. Die Beschreibung auf der höheren Abstraktionsebene erleichtert die Anwendungsentwicklung erheblich. Daher erscheint dieser Ansatz für die Bereitstellung von Ansichten bei der interaktiven Bearbeitung von Szenen geeignet. Möglicherweise auftretende Einschränkungen hinsichtlich der visuellen Qualität können hingenommen bzw. durch Kombination mit Low-Level-Elementen vermieden werden. Die Verwendung eines MPEG-4-Terminals zur Visualisierung erscheint aus technischer Sicht viele Vorteile zu bieten. Die erreichbare visuelle Qualität ist vergleichbar mit der der potenziellen Endgeräte und der Implementierungsaufwand ist gering. Voransichten auf Basis eines Terminals werden daher im vorgeschlagenen Autorensystem genutzt, wenn keine Bearbeitungsfunktion benötigt wird. Für Visualisierungen mit Bearbeitungsfunktion wird bei der Entwicklung des Autorensystems das High-Level-API *Java3D* verwendet. Dieser Ansatz bietet einen guten Kompromiss zwischen zu erwartendem Aufwand, Möglichkeiten und visueller Qualität.

4.6 Zusammenfassung

Die in diesem Kapitel entwickelten Konzepte und Komponenten sind Grundlage für die Schaffung von Autorensystemen für verschiedene Anwendungsgebiete. Es können sowohl universell einsetzbare als auch spezialisierte Autorensysteme für die Erstellung objektbasierter AV-Anwendungen realisiert werden. Aus den ermittelten allgemeinen Anforderungen wurden drei Schwerpunkte für die weitere Entwicklung abgeleitet: Authoring-Formate, Autorenwerkzeuge und Authoring-Server. Die Grundlage des Authorings bilden XML-basierte Authoring-Formate. Diese nehmen alle erforderlichen Daten auf und erlauben die Beschreibung multimedialer Anwendungen auf einer an die Aufgabe angepassten Abstraktionsebene. Die Bearbeitung kann Quelltext-basiert oder grafisch-interaktiv erfolgen. Somit lassen sich die Stärken beider Ansätze miteinander verbinden. Es wurde eine Architektur für die Verarbeitung von Authoring-Formaten vorgeschlagen. In den Unterabschnitten 6.1.2, 6.2.1 und 6.5.1 werden drei spezialisierte Authoring-Formate und deren Verarbeitung vorgestellt. Autorenwerkzeuge sind die Schnittstellen der Autoren zu den Szenendaten. Im Fokus standen grafisch-interaktive Werkzeuge für eine intuitive Bearbeitung während des Authoring-Prozesses. Es wurde eine modulare Architektur unter Verwendung etablierter Komponenten vorgeschlagen. Diese ermöglicht die Realisierung sowohl von universellen als auch von spezialisierten Werkzeugen. Die Anbindung an einen Authoring-Server kann auf verschiedenen Ebenen erfolgen; ein autarker Betrieb ist ebenfalls möglich. In Abschnitt 6.3 und in Unterabschnitt 6.1.4 werden prototypische Implementierungen von Autorenwerkzeugen auf Basis der hier entwickelten Konzepte und Komponenten vorgestellt.

5 Authoring-Server

Für eine effektive Zusammenarbeit mehrerer Autoren während der Erstellung einer komplexen AV-Anwendung ist eine zentrale Verwaltung der anfallenden Daten erforderlich. In einem Server-basierten Autorensystem übernimmt ein Server die Speicherung, Verwaltung und Bereitstellung aller Produktions-relevanten Daten. In diesem Kapitel wird ein solcher Authoring-Server entsprechend den Anforderungen aus Kapitel 4 entwickelt.

5.1 Anforderungen

Der Authoring-Server ist ein in ein Netzwerk eingebundener zentraler Datenspeicher, der alle relevanten Daten einer Produktion aufnehmen kann. Die beteiligten Medienlieferanten und Autoren greifen mit ihren Werkzeugen auf den zentralen Datenbestand zu und legen die Ergebnisse ihrer Arbeit dort ab. Im einfachsten Fall dient der Authoring-Server lediglich dem Datenaustausch zwischen den Beteiligten einer Produktion. Die weit reichende Unterstützung eines verteilten Authoring-Prozesses erfordert die Steuerung der Zugriffe auf die abgelegten Elemente. Ein verbreiteter Ansatz für die Koordination der Zusammenarbeit ist beispielsweise das Sperren von Elementen, die gerade von einem Autor bearbeitet werden. Einfache WCMS arbeiten nach diesem Prinzip. In solch einem Fall beschränkt sich der Zugriffsschutz auf Dateien. Daher ist eine synchrone Bearbeitung einer Datei nicht möglich.

Bei der Produktion objektbasierter AV-Anwendungen sind die Anforderungen an die Organisation der Daten deutlich höher als bei vielen anderen Anwendungen. Besondere Anforderungen gelten für die Verwaltung der Szenenbeschreibungen. Für eine sinnvolle Zusammenarbeit mehrerer Autoren ist die Steuerung der Zugriffe auf einzelne Bereiche und Elemente eines Szenengraphen notwendig.

5.1.1 Aufgaben

Die wichtigsten Aufgaben eines Authoring-Servers im Prozess der verteilten Erstellung multimedialer Anwendungen sind:

- zentrale Datenhaltung und Datenmanagement
- Projektverwaltung
- Bereitstellung von Schnittstellen zu Werkzeugen und Encodern
- Anbindung an ein Distributionssystem

Bei der Produktion objektbasierter AV-Anwendungen entstehen zahlreiche Informationen verschiedenster Art. Dies erfordert ein sehr leistungsfähiges und flexibles Datenmanagement. Die Zusammenarbeit in einer Gruppe von Autoren muss mit technischen Mitteln organisiert werden. Der Authoring-Server muss entsprechende Werkzeuge zur Projektverwaltung bereitstellen. Mit ihnen werden Projekte angelegt und verwaltet, sowie Nutzer,

Gruppen und Berechtigungen definiert. Der Authoring-Server muss in eine Netzwerkstruktur zur Kommunikation mit den verbundenen Autorenwerkzeugen eingebunden sein. Dabei ist ein Schutz vor unberechtigten Zugriffen vorzusehen. Das Authoring-Konzept ist so auszulegen, dass einzelne Autoren auf Wunsch auch ohne Verbindung zum Server arbeiten können. Hierfür müssen Projektdaten vom Server für eine lokale Verwendung bereitgestellt und die bearbeiteten Daten in den zentralen Datenbestand integriert werden. Schließlich müssen fertig gestellte Anwendungen an ein Distributionssystem übergeben werden. Hierfür sind entsprechende Schnittstellen vorzusehen.

5.1.2 Betriebsarten

Ein Authoring-Server kann prinzipiell in drei Betriebsarten verwendet werden:

1. *Retrieval-Modus:* Der Server dient lediglich als zentraler Speicher und stellt Szenendaten bereit. Ein Autor holt sich die Szenendaten, um diese entweder in einem eigenen Projekt oder lokal weiterzuverarbeiten.

2. *Mehrbenutzer-Modus, asynchrone Bearbeitung:* Ein Autor kann jederzeit die komplette Szene vom Server herunterladen, darf aber nur bestimmte Knoten bearbeiten. Dem muss nicht nur der Server, sondern auch der Client Rechnung tragen.

3. *Mehrbenutzer-Modus, synchrone Bearbeitung:* Die Autoren bearbeiten die Szene direkt auf dem Server. Jede Veränderung an der Szene korreliert mit einer Veränderung auf dem Server und den angeschlossenen Clients. Bei mehreren aktiven Autoren entsteht so ein erheblicher Datenverkehr.

Eine weitere Möglichkeit ist die unidirektionale Übertragung der veränderten Daten zum Server in festgelegten Zeitintervallen. Im Gegensatz zur Echtzeitbearbeitung verringert sich das Datenaufkommen dadurch spürbar. Die Veränderungen auf dem Server haben keine Auswirkungen auf die Szenengraphen der angeschlossenen Autorenwerkzeuge, wodurch die Autoren in ihrer Arbeit nicht gestört werden. Parallel zur Aktualisierung des Szenengraphen kann auf dem Server eine entsprechende MP4-Datei generiert werden, die von den Autoren mit Hilfe von MPEG-4-Terminals jederzeit wiedergegeben werden kann.

5.2 Datenmanagement

Das Datenmanagement des Authoring-Servers muss verschiedenste Arten von Daten verwalten können. Die Szenenbeschreibungen sind beispielsweise komplexe baumartige Strukturen, wogegen die Medienobjekte meist durch sehr große binäre Daten repräsentiert werden. Zentrale Aufgabe des Authoring-Servers ist die Realisierung eines leistungsfähigen Datenmanagements. Tabelle 5.1 zeigt die zu verwaltenden Arten von Daten und deren Speicherung. Diese inhomogenen Daten erfordern jeweils spezielle Lösungen. Die anspruchsvollste Aufgabe hierbei ist ohne Zweifel die Verwaltung von hierarchisch strukturierten Daten (z. B. Szenenbeschreibungen) und die Steuerung der Zugriffe und Berechtigungen auf deren Elemente.

Datenart	Beschreibung	Speicherung
Szenen-beschreibungen	baumartige Datenstruktur, meist sehr komplex; z. B. VRML, X3D, XMT, SMIL usw.	strukturiert (XML)
natürliche Medienobjekte	meist große binäre Dateien, uncodierte und codierte Audio- und Videodateien	unstrukturiert (BLOB)
synthetische Medienobjekte	parametrische Beschreibung	strukturiert (XML, Text)
Metadaten	baumartige Datenstruktur; z. B. MPEG-7, XML usw.	strukturiert (XML)
administrative Daten	Nutzerdaten und Knotenrechte, Projektverwaltung	strukturiert (XML)
Ergebnisse	encodierte Anwendungen; z. B. MP4	unstrukturiert (BLOB)

Tabelle 5.1: Arten von Daten und deren Speicherung

Medienobjekte

Das Authoring hat maßgeblich die Erstellung der Szenenbeschreibung zum Ziel; die Medienobjekte werden dabei nicht aktiv bearbeitet. Daher stellt die Verwaltung der Medienobjekte im Vergleich zu den Szenenbeschreibungen weitaus geringere Anforderungen an das Datenmanagement. Die Medienobjekte werden von den Produzenten auf dem Server abgelegt und stehen den Autoren ohne Einschränkung zur Verfügung. Die Produzenten bleiben Eigentümer der Medienobjekte und haben vollen Zugriff. Die Autoren haben zunächst lediglich lesenden Zugriff. Ausnahmen bilden Medienobjekte, die wiederum selbst auf einer Szenenbeschreibung aufbauen, z. B. synthetische 3D-Objekte. Aus Sicht des Autors der Szene bilden diese ebenfalls Objekte, ohne dass dieser Rücksicht auf deren Beschreibungsform nehmen muss. In die Szenenbeschreibung werden solche Medienobjekte beispielsweise als *Inline*-Knoten integriert, wodurch die Bearbeitbarkeit erhalten bleibt. Für die Verwaltung von Medienobjekten muss das Datenmanagement folgende Funktionen bereitstellen:

- *Speicherung von Medienobjekten:* Natürliche Medienobjekte werden als unstrukturierte binäre Dateien, so genannte *Binary Large Objects (BLOB)* verwaltet. Diese sind für das Datenmanagement nicht interpretierbar und können nur mit entsprechenden Applikationen betrachtet werden. Die parametrischen Beschreibungen der synthetischen Medienobjekte können ggf. strukturiert verwaltet werden.

- *Speicherung von Voransichten:* Zur schnellen Sichtung großer Objektbestände sind gut handhabbare Ansichten, z. B. Szenenausschnitte, Thumbnails und Hörproben zu integrieren.

- *Speicherung von Metadaten:* Inhaltliche und technische Zusatzinfomationen sind in geeigneter Weise zu verwalten, z. B. als XML-Beschreibungen bzw. MPEG-7-Beschreibungen.

Szenenbeschreibungen
Szenenbeschreibungen sind vergleichsweise komplexe hierarchisch strukturierte Daten. Sollen mehrere Autoren gemeinsam an einer Szene arbeiten, so erfordert dies eine feingranulare Steuerung der Zugriffe auf einzelne Elemente des Szenengraphen. Bei der Erstellung komplexer Anwendungen kann es beispielsweise erforderlich sein, Rechte für verschiedene Gruppen von zugriffsberechtigten Personen zu vergeben. Denkbar wäre eine Unterteilung in einen Szenen-Administrator (Direktor) mit vollen Zugriffsrechten, einen Autor für Interaktionen oder einen Autor für Kamerafahrten. Hierfür reicht eine einfache Steuerung von Schreib-, Lese- und Änderungsrechten auf eine Szene nicht aus. Die Zugriffsrechte müssen für einzelne Elemente des Szenengraphen gesteuert werden. Besondere Bedeutung erlangt diese Forderung bei gleichzeitiger Bearbeitung der Szenenbeschreibung durch mehrere Autoren. Eine weitere Forderung ist die Versionskontrolle. Hiermit wird sichergestellt, dass Änderungen an der Szene nachvollziehbar sind. Ansätze für die Versionskontrolle für Dokumente bietet die Technik *Web Document Authoring and Versioning (WebDAV)*, welche nicht auf bestimmte Medientypen beschränkt ist (siehe Unterabschnitt 4.2.3). Eng mit der Versionskontrolle verbunden ist die Funktion des *Rollback*, d. h. die Möglichkeit, beliebige Bearbeitungsstufen zurückzusetzen. Vergleichbar mit dem *UNDO* Befehl in herkömmlichen Programmen soll das Rollback für knotenbasierte Dokumente Funktionen bieten, die auf einzelne Objekte, Objektgruppen und Teilbäume angewendet werden können. Für die Verwaltung der Szenenbeschreibungen muss das Datenmanagement folgende Funktionen bereitstellen:

- Speicherung der Szenenbeschreibungen entsprechend ihrer Struktur
- Steuerung der Zugriffe auf einzelne Elemente des Szenengraphen
- Versionskontrolle und *Rollback*

Eine mögliche Art der Speicherung von XML-Dokumenten ist die Speicherung als Datei im Dateisystem. Eine automatisierte Verarbeitung (z. B. die Suche nach Elementen) ist begrenzt möglich, wird aber durch die fehlende Trennung von Struktur und Inhalt erschwert. Die Steuerung der Zugriffe auf einzelne Elemente des Szenengraphen ist ebenfalls sehr aufwändig. Um die zuvor gestellten Forderungen an das Datenmanagement zu erfüllen, müssten alle Funktionen separat implementiert werden. Daher erscheint diese Art der Speicherung von Szenenbeschreibungen für das Autorensystem nicht sinnvoll.

5.2.1 Szenendaten in Datenbanken

Die Forderung nach kontrollierten Zugriffsmöglichkeiten auf Elemente des Szenengraphen für verschiedene Nutzer legt die Verwendung einer Datenbank-Lösung zur Verwaltung von MPEG-4-Anwendungen nahe. Folgende Punkte sprechen für eine datenbankbasierte Speicherung der Szenendaten:

- zentrale Haltung großer Datenbestände
- Organisation der Daten
- definierte Zugriffsmöglichkeiten
- Bearbeitung von Teilstrukturen

- komplexe Suchmöglichkeiten

Ein Datenbankmanagementsystem bietet gewöhnlich folgende grundlegende Funktionen an [Cod82]:

> Integration, Benutzersichten, Katalog, Operationen, Konsistenzüberwachung, Datensicherung, Datenschutz, Transaktionen und Synchronisation.

In Datenbanksystemen sollen möglichst alle zu einer Anwendung gehörenden Daten zentral gespeichert und über einheitliche Schnittstellen zugänglich gemacht werden. Es werden Operationen wie *Insert*, *Delete* und *Update* angeboten, um den aktuellen Datenbestand zu ändern sowie Operationen, mit denen komplexe Anfrageoperationen ausgeführt werden können. Ein weiterer wichtiger Punkt in traditionellen Datenbanken ist die Bereitstellung von Nutzersichten. Hiermit wird es dem Anwender ermöglicht, bestimmte Daten nach seinen Wünschen zusammenzustellen und darzustellen [DS01].

Die Speicherung multimedialer Daten in Datenbanken wird mit verschiedenen Zielstellungen betrieben. Oftmals steht lediglich die Verwaltung von Audio- und Videodateien im Fokus, z. B. bei der Distribution von Filmen mit so genannten *Video-Servern* oder *Multimedia-Servern*. Die einzelnen Inhalte (z. B. Filme, Musik oder Bilder) werden als Ganzes betrachtet und organisiert. Komplexe strukturierte Daten, wie sie beispielsweise Szenenbeschreibungen darstellen, werden hierbei nicht gesondert berücksichtigt. Solche Datenbanken werden oft als *Multimedia-Datenbanken* oder *MMDBMS* bezeichnet.

Anspruchsvoller und für das Authoring multimedialer Anwendungen interessanter ist die Verwaltung der Bestandteile einer Anwendung [DS01]. Ein Ansatz für die Speicherung von MPEG-4-Szenen unter Nutzung eines objektrelationalen Datenbankmanagementsystems (*ORDBMS*) nach [DS02] ist in Abbildung 5.1 dargestellt. Medienobjekte werden als unstrukturierte Daten betrachtet und als *BLOB* gespeichert. Diese Daten sind demzufolge für die Datenbank nicht direkt interpretierbar. Die einzelnen Objektarten werden durch so genannte *Extender* in das objektrelationale Modell integriert.

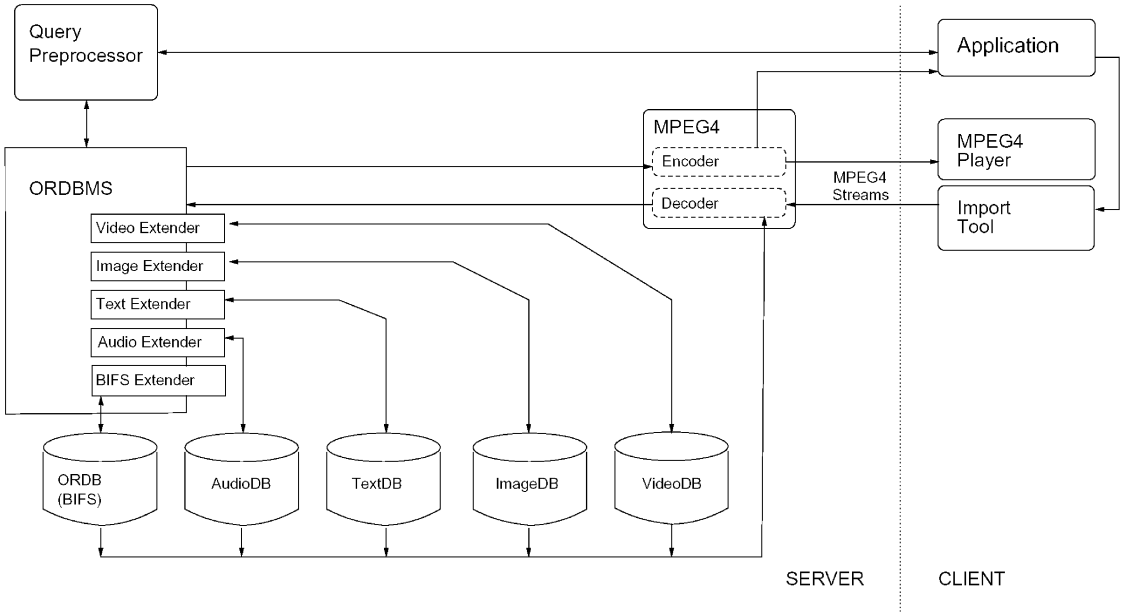

Abbildung 5.1: DBMS für MPEG-4-Anwendungen [DS02]

Für das Authoring ist insbesondere die Verwaltung der Szenenbeschreibungen von Interesse. Diese stellen hierarchisch strukturierte Daten dar, die sich entsprechend ihrer Struktur verwalten lassen. Im einfachsten Fall können Szenenbeschreibungen als gewöhnlicher Text behandelt und unstrukturiert als *Character Large Object (CLOB)* in einer Datenbank abgelegt werden. Eine Suche im Szenengraphen ist nur durch ein aufwändiges Parsen möglich. Eine Trennung in Struktur und Inhalt erfolgt nicht. Die Modellierung der Beziehungen zwischen den Medienobjekten und der Szenenbeschreibung ist daher unzulänglich und die Erzeugung von Abfragen ist sehr komplex in der Verarbeitung [DS02].

Im Abschnitt 4.4 wurde die Verwendung XML-basierter Authoring-Formate behandelt. Ein Argument für XML-basierte Formate sind die vielfältigen Möglichkeiten der Verarbeitung mit XML-Werkzeugen. Dies gilt auch für die Speicherung und Verwaltung der Szenendaten in Datenbanken. Im Folgenden werden einige Ansätze für die Verarbeitung von XML-Daten in Datenbanken beschrieben.

5.2.2 Abbildung von XML-Dokumenten in Datenbanken

Für Szenenbeschreibungen auf Basis von XMT bzw. der in Abschnitt 4.4 beschriebenen Formaten liegen Schemas in Form von *XML-Schemadefinitionen (XSD)* vor. Damit sind die Regeln für den Aufbau der jeweiligen Beschreibung vollständig bekannt. Anders sieht dies bei Szenenbeschreibungen im textuellen BIFS-Format oder in VRML aus. Für diese existieren bislang keine vollständigen Schemadefinitionen.

Die meisten verfügbaren Datenbanksysteme arbeiten mit einem relationalen oder einem objektrelationalen Datenmodell. Die hierarchisch strukturierten Daten der Szenenbeschreibungen müssen zunächst in das Datenmodell der Datenbank überführt werden. Hierfür existieren verschiedene *Mapping-Verfahren* für die Transformation eines Datenmodells in ein anderes durch festgelegte Vorschriften. Dies kann z. B. die Abbildung eines XML-Dokumentes in eine relationale Datenbank oder umgekehrt sein. Mapping-Verfahren lassen sich in *unidirektionale* und *bidirektionale* Verfahren unterteilen. Ein unidirektionales Verfahren ist das *Template Driven Mapping*. Unter der Bezeichnung *Model Driven Mapping* werden Verfahren beschrieben, die in beide Richtungen einsetzbar sind. Hierzu gehören das *Table Based Mapping* und das *objekt-relationale Mapping*. Abbildung 5.2 gibt eine Übersicht über die verschiedenen Mappingverfahren.

Template Driven Mapping

Dieses unidirektionale Verfahren erlaubt die Erzeugung von XML-Dateien mit Hilfe geeigneter Schablonen (*Templates*) aus einer Datenbank heraus. Als Technik kommt hier häufig XSL zum Einsatz, mit der aus einer relationalen Datenbank heraus dynamisch strukturierte Dokumente (z. B. Webseiten) erzeugt werden können. In den Templates sind Datenbankanweisungen in speziellen XML-Tags eingebettet, die nach der Ausführung XML-konforme Daten beinhalten. So können sehr einfach und flexibel Informationen aus Datenbanken in XML-Dokumente eingefügt werden. Die Templates sind leicht editierbar und lassen sich über *Style Sheets* anpassen [Bou].

Model Driven Mappings

Die bidirektionalen Verfahren ermöglichen die Abbildung sowohl von einer Datenbank in ein XML-Dokument als auch umgekehrt. Hierfür wird ein festes Modell für die Struktur bzw. die Daten in jedem XML-Dokument vorausgesetzt, das abgebildet werden soll. Zwei Verfahren dieser Art sind verbreitet [Bou]:

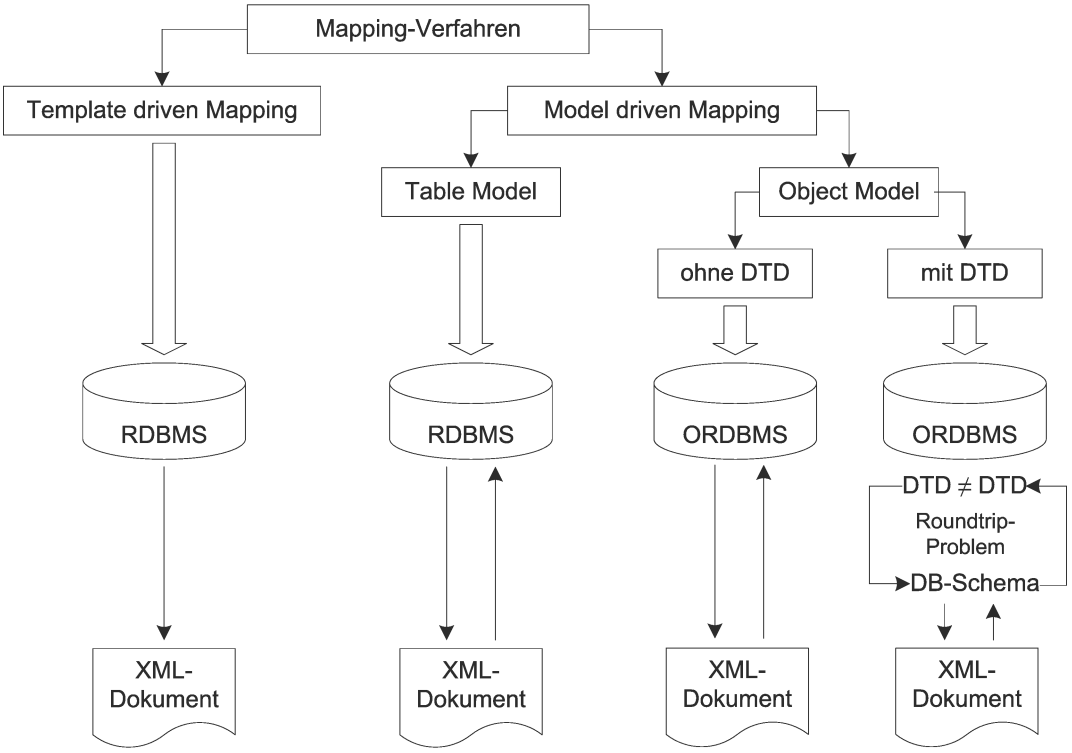

Abbildung 5.2: Einordnung von Mapping-Verfahren

1. *Tabellen-basiertes Mapping:* Dieses Verfahren bildet die Struktur von XML-Dokumenten auf Basis eines zwingend erforderlichen XML-Schemas (DTD oder XSD) in einem relationalen Datenmodell ab. Besonderes eignet es sich für die Verarbeitung von Dokumenten mit flacher Struktur und gleichförmigen Inhalten, z. B. in Bestell- oder Buchungssystemen. Diese Abbildungsform ermöglicht einen schnellen und effektiven Datenaustausch zwischen relationalen Datenmodellen und ist einfach zu implementieren. Nicht geeignet ist es hingegen für tief verschachtelte XML-Dokumente und für komplexe Datenmodelle [Bou]. In [FK99] wird ein auf diesem Modell basierendes System vorgestellt.

2. *Objekt-relationales Mapping:* Bei diesem Verfahren erfolgt die Abbildung der XML-Daten in einem objektrelationalen Datenmodell. Die modellierten Daten werden auf Objekte abgebildet und in einer Baumstruktur angeordnet. Es entsteht ein datenspezifischer Objektbaum, in dem Objekte durch Elemente dargestellt werden und Attribute (bzw. Subelemente) die Eigenschaften dieser Objekte repräsentieren. Mit dieser Modellierung können alle Arten von XML-Dokumenten verarbeitet werden, ohne dass Einschränkungen auf Grund der Verschachtelungstiefe auftreten. Die logische Struktur bleibt in beiden Richtungen erhalten. Das Mapping kann mit und ohne Verwendung eines XML-Schemas (DTD oder XSD) erfolgen.

Es folgen einige Beispiele für Systeme auf Basis des objektrelationalen Mappings. In [HSS03] wird ein Ansatz zur Speicherung von Szenengraphen in einer objektrelationalen Datenbank beschrieben. Ein XML-Schema ist nicht erforderlich. Der Szenengraph wird zunächst in Inhalte und Struktur zerlegt und die einzelnen Knoten in Tabellen abgelegt (siehe Abbildung 5.3). Die Struktur wird als Liste von Kanten gespeichert. Der Schwerpunkt dieses Ansatzes liegt auf der redundanzfreien Speicherung von Szenengraphen.

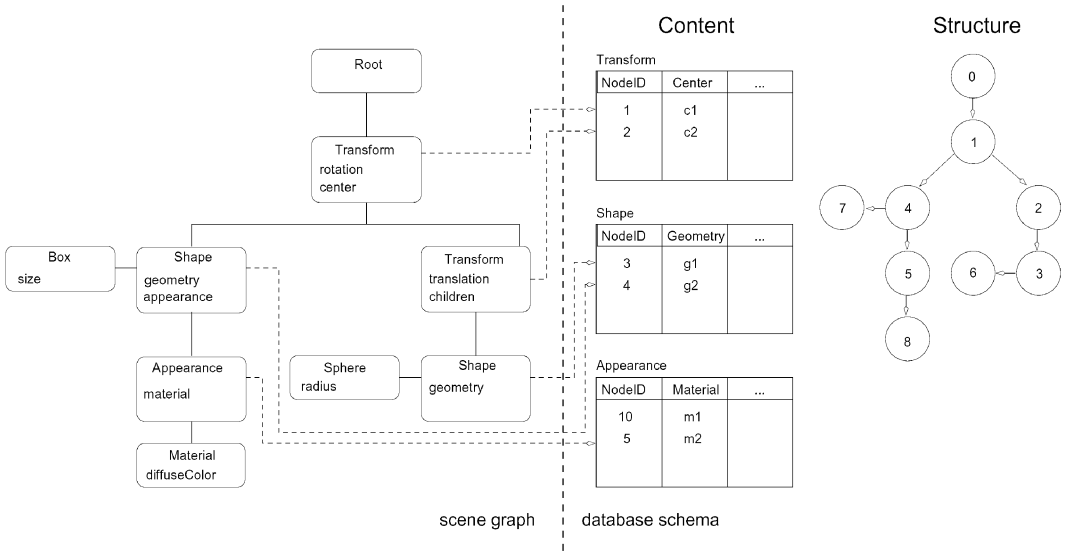

Abbildung 5.3: Dekomposition eines Szenengraphen [HSS03]

Das in [STZ+99] vorgestellte Verfahren benötigt eine DTD und bildet die Daten in einer relationalen Datenbank ab. Ein Problem dieser Ansätze ist die nicht eindeutige Konvertierung des XML-Schemas in das Datenbank-Schema, bei deren Rückwandlung nicht zwingend exakt das gleiche XML-Schema entsteht. Dieses Problem wird als *Roundtripping-Problem* bezeichnet [Neu02].

Das Prinzip des Objekt-Modells wird für die Integration von XML-Fähigkeiten in vorhandene Datenbanksysteme angewendet. Die dabei entstehenden Systeme heißen *XML Enabled Databases (XEDB)* [STZ+99]. Zahlreiche kommerzielle Produkte wurden auf diese Weise um Funktionalitäten für die Verarbeitung von XML-Dokumenten erweitert, z. B. *IBM DB2*, *Oracle8i*, *Informix* und *Microsoft SQL Server*. Dieser Ansatz ist praktikabel, wenn die XML-Dokumente sehr gleichmäßig und nicht zu tief strukturiert sind. Beispiele hierfür sind Produktkataloge und Verzeichnisse. Ist dies nicht der Fall, entstehen im relationalen Modell entweder sehr viele Tabellen oder sehr viele Spalten mit Null-Werten. Dies führt zu einer sehr ineffizienten Speicherung der Daten [Bou]. Der Inhalt eines XML-Dokumentes ist im Datenmodell je nach Komplexität der Daten immer auf mehrere Tabellen verteilt (siehe Abbildung 5.3), was sich negativ auf die Zugriffsgeschwindigkeit auswirkt. Einige OODBMS bieten Sicherheitsrichtlinien für ihre Systeme an, beispielsweise ein Dokument-basiertes Sicherheitssystem. Damit können Rechte für Dokumente vergeben werden. Dies entspricht nicht den Anforderungen an das Autorensystem. Hier sollen mehrere Benutzer gleichzeitig an einer Szene arbeiten. Dafür ist ein Zugriffskonzept auf Knotenebene erforderlich.

5.2.3 Native XML-Datenbanken

Die so genannten *nativen XML-Datenbanken* oder *Native XML Databases (NXD)* sind eine relativ junge Gattung von Datenbanken, die XML-Dokumente direkt in ihrer hierarchischen Struktur speichern und verwalten. Die logische Struktur ist ein XML-Dokument. Genau wie alle anderen Datenbanken bieten diese Funktionen wie z. B. Transaktionen, Sicherheitsmechanismen, Mehrbenutzer-Betrieb, Programmierschnittstellen und Anfragesprachen [Bou]. Ein aufwändiges Mapping entfällt und ein vollständiges Roundtripping ist

gewährleistet. Für die Abfrage von Daten stehen spezielle XML-Abfragesprachen (*XML Query Languages*) zur Verfügung.

Auf Grund der Speicherung von XML-Dokumenten werden Abfragen auf Dokumentebene sehr schnell abgearbeitet. Werden bei einer Abfrage mehrere Dokumente benötigt, steigt die Bearbeitungszeit. Bei den zuvor beschriebenen *XML-enabled* Datenbanken sind die XML-Daten immer auf mehrere Tabellen verteilt und die Zugriffe auf einzelne XML-Dokumente daher sehr langsam. Zur Abfrage von Daten können XML-spezifische Abfragesprachen wie *XQuery* genutzt werden [Webb]. Die Abfrage einer XML-Datenbank liefert als Ergebnis immer ein XML-Dokument. In vielen Fällen können die Daten so direkt verarbeitet oder über XML-basierte Protokolle übertragen werden. Wird eine abweichende Form der Daten benötigt, müssen diese zunächst einen Parser durchlaufen, um die benötigten Informationen auslesen zu können. Für lokale Anwendungen kann dies z. B. im Vergleich zu Anwendungen auf Basis eines RDBMS mit ODBC ein Nachteil sein. Bei verteilten Anwendungen, die ohnehin XML für den Datentransport nutzen, ist dies nicht von Bedeutung [Bou]. Eine weitere wichtige Eigenschaft nativer XML-Datenbanken ist das Roundtripping von XML-Dokumenten; ein gespeichertes XML-Dokument kann exakt (Struktur und Inhalt) wiederhergestellt werden. Dies ist auf dem Gebiet der XML-basierten Datenverarbeitung keineswegs selbstverständlich. Oft ist eine Reproduzierbarkeit der Inhalte ausreichend, insbesondere bei datenzentrierten Anwendungen. Im Falle dokumentzentrierter Anwendungen ist ein vollständiges Roundtripping obligatorisch.

Im Zuge der allgemeinen XML-Euphorie entstanden in den letzten Jahren zahlreiche Lösungen für die native Speicherung von XML-Dokumenten. Der Schwerpunkt der Anwendungen liegt momentan in den Bereichen *E-Commerce* und *Content Management Systeme (CMS)*. Eines der ersten kommerziellen Produkte war der *Tamino XML-Server*[1] der Firma *Software AG*[2]. Neben vielen kommerziellen Produkten gibt es inzwischen auch zahlreiche Projekte aus dem Open-Source Bereich, z. B. *Xindice* der *Apache Software Foundation (ASF)*[3] oder *ozone*[4].

Die nativen XML-Datenbanken stehen in Konkurrenz zu den XML-enabled Datenbanken, die für die oben genannten typischen Anwendungen (einfach strukturierte und wenig variable Dokumente) gut anwendbar sind. Eine umfangreiche Auflistung aller XML-fähigen Datenbanken ist in [Bou] zu finden.

5.2.4 XML:DB

Um die Integration von XML-fähigen Datenbanken (NXD und XEDB) in IT-Systeme zu erleichtern und die Interoperabilität zu fördern, wurde die *XML:DB Initiative*[5] ins Leben gerufen. Ein Ziel ist die Entwicklung von Spezifikationen für die Verwaltung von XML-Dokumenten in Datenbanken. Die Referenzimplementierungen stehen unter Open Source Lizenz. Es entstand eine Community aus Anbietern und Nutzern von XML-Datenbanken. Die XML:DB Initiative bearbeitet folgende Projekte [XMLc]:

- *XML Database API* [XMLa]: Entwicklung einer einheitlichen Programmierschnittstelle für XML-Datenbanken

[1] http://www.tamino.de/
[2] http://www.softwareag.com/
[3] http://xml.apache.org/xindice/
[4] http://ozone-db.org/
[5] http://xmldb-org.sourceforge.net/

- *XUpdate – XML Update Language* [XMLd]: Entwicklung einer Sprache zur Manipulation von Daten in XML-Datenbanken

- *SiXDML – Simple XML Manipulation Language* [XMLb]: Entwicklung einer einheitlichen Sprache zur Definition *Data Definition Language (DDL)* und Manipulation *Data Modification Language (DML)* von Daten ähnlich SQL und eines API basierend auf der *XML:DB Database API*

Das *XML:DB Database API* bietet allgemein formulierte Mechanismen für den Zugriff auf XML-Datenbanken an und ist vergleichbar mit *ODBC* oder *JDBC* aus der Welt der relationalen Datenbanken. In Zukunft könnten die Ergebnisse dieser Arbeiten die Entwicklung von Anwendungen ermöglichen, die weitgehend unabhängig von der zugrunde liegenden Datenbank sind. Damit wären die Datenbanken austauschbar.

5.2.5 Auswahl einer Datenbank

Die Speicherung und die Verwaltung von Szenendaten stellen auf Grund der inhomogenen Daten hohe Anforderungen an eine Datenbank. Insbesondere die hierarchisch strukturierten Szenenbeschreibungen sind nur unter erheblichem Aufwand mit herkömmlichen Datenbanken zu verwalten. Die genannten Mapping-Verfahren weisen verschiedene Nachteile auf. Die Auswahl eines Verfahrens für die Speicherung von XML-Dokumenten hängt stark von der Art der zu speichernden Daten ab. Die zu erwartenden XMT-basierten Szenenbeschreibungen sind weder rein datenzentriert noch rein dokumentzentriert. Datenzentrierte Dokumente lassen sich gut mit den zuvor beschriebenen Mapping-Verfahren verwalten. Dokumentzentrierte XML-Daten hingegen können als Ganzes (z. B. als CLOB oder BLOB) gespeichert und verarbeitet werden. Szenenbeschreibungen fallen in Umfang und Struktur sehr unterschiedlich aus, sodass eine eindeutige Zuordnung nicht möglich ist. Eine denkbare Nutzung des Objekt-Modells lässt die oben genannten Probleme erwarten. Insbesondere die Ineffizienz bei tief verzweigten Szenengraphen spricht gegen den Einsatz einer solchen Lösung. Auch das Problem des Roundtrippings kann nicht zufriedenstellend gelöst werden. Ein weiteres Kriterium ist die Steuerung der Zugriffe auf Szenendaten. Die bekannten *XML-enabled* Datenbanken ermöglichen eine Steuerung auf der Ebene eines Dokumentes. Der Schutz einzelner Bereiche und Elemente des Dokumentes ist hingegen nicht möglich. Diese Überlegungen motivierten zur Nutzung einer nativen XML-Datenbank für die weitere Entwicklung des Authoring-Servers.

Anforderungen an das DBMS

Unter Berücksichtigung der Anforderungen an das Autorensystem (siehe Abschnitt 4.1) werden folgende Anforderungen an das *Datenbank-Managementsystem (DBMS)* auf Basis einer nativen XML-Datenbank gestellt:

1. Speicherung der Szenendaten:
   - strukturierte Speicherung von XML-Dokumenten
   - Speicherung von Nicht-XML-Daten, z. B. Video, Audio usw.
   - Bearbeitung von Teilbäumen
   - Unterstützung der XML-Schemaspezifikation

2. Kommunikationsmöglichkeiten mit der Datenbank:
   - integrierbar in eine Netzwerkumgebung
   - Übertragungsprotokolle auf Basis von HTTP
   - standardisierte Abfragen, z. B. mit *XQuery*
3. Datenzugriff und -sicherheit:
   - Verwaltung von Nutzern und Nutzergruppen
   - Zugriffsrechte auf Knotenebene unter Berücksichtigung von Attributen
   - Einbindung in eine Webserver-Umgebung
   - Möglichkeit der Fernadministration

Zum Zeitpunkt der Entscheidung[6] war die Auswahl an nativen XML-Datenbanken sehr begrenzt. Viele der Entwicklungen waren noch relativ jung und unausgereift. Inzwischen sind ca. 40 native XML-Datenbanken verfügbar. Neben den kommerziellen Produkten gibt es auch einige interessante Entwicklungen aus dem Open Source Bereich [Bou]. Tabelle 5.2 zeigt eine Vorauswahl aus vier verfügbaren XML-Datenbanken und einige wichtige Eigenschaften: Unterstützung von Schemadefinitionen (XSD oder DTD), Zugriffssteuerung mit Zugriffskontrolllisten (ACL)[7], Speicherung von binären Daten (BLOB) und die Unterstützung durch XML:DB.

Produkt	XSD	DTD	ACL	BLOB	XML:DB
Tamino	×	×	×	×	×
XStreamDB	×	×	×	×	
dbXML			×	×	×
eXist			×	×	×

Tabelle 5.2: Vorauswahl der XML-Datenbank

Tamino XML Server

Für die prototypische Implementierung des Authoring-Servers wurde der *Tamino XML-Server* ausgewählt. Entscheidend für die Auswahl war das leistungsfähige Knotenrechtemanagement, die Speicherung aller Datenformate, die umfangreichen Kommunikationsmöglichkeiten und der gute Online-Support. *Tamino* speichert alle Arten von Daten (XML und Nicht-XML). XML-Dokumente werden strukturiert in ihrer ursprünglichen Form abgelegt. Der Zugriff auf Daten erfolgt über verschiedene Protokolle (z. B. WebDav oder SOAP) und über direkte API-Aufrufe aus Programmen heraus. Auf alle Daten kann direkt über einen Webbrowser oder durch die Verwendung von *XQL* zugegriffen werden. Das Zugriffsmanagement des XML-Servers erlaubt die Steuerung der Zugriffe bis hinab auf Knotenebene. Durch die Integration in eine Webserver-Umgebung lässt sich der XML-Server fernadministrieren. Die *Tamino* XML-Server-Software besteht aus dem *Tamino XML-Server*, der *Tamino Schema Definition (TSD)*, den *Tamino Tools* und den *Tamino Application Programming Interfaces (API)*. Folgende Funktionen werden unterstützt [Sof03]:

[6] Frühjahr 2002
[7] auch Befugnislisten (Capabilities), siehe 5.3.1

- Der XML-Verarbeitungsteil des XML-Servers verwendet die Protokolle HTTP, TCP/IP, SOAP und Webdav für den Datenaustausch. Der XML-Server arbeitet mit verschiedenen Webservern zusammen, z. B. mit *Apache*, *IIS* und *iPlanet*.

- Zu XML-Dokumenten können Elemente oder Knoten hinzugefügt werden, ohne die Dokument- bzw. Schemastruktur verändern zu müssen.

- *Tamino* verwendet für die Abfrage von XML-Dokumenten und Teilen aus XML-Dokumenten die Abfragesprache *X-Query*, die eine Umsetzung der *XPath*-Spezifikation des W3C darstellt. Diese Basistechnik unterstützt die Abfrage von hierarchisch strukturierten XML Daten.

- XML-Schema ist die Basis für die *Tamino* Schemadefinition (TSD). Sie bietet eine Abbildung der XML-Schemadefinition auf die interne Struktur von *Tamino*. Die aktuelle Version TSD4 bietet noch nicht den kompletten Sprachumfang von XML-Schema, soll in Zukunft aber dahingehend erweitert werden.

Die Entwicklung des Authoring-Servers soll weitestgehend unabhängig von der konkret verwendeten XML-Datenbank erfolgen. In Zukunft könnten verschiedene Datenbanken zum Einsatz kommen. Interessant erscheinen vor allem die Entwicklungen aus dem Umfeld des Projektes *XML:DB* [XMLc].

5.3 Datenzugriff

Ein auf mehrere Autoren verteilter Authoring-Prozess erfordert ein Konzept für den Datenzugriff zur Vermeidung störender Beeinflussungen zwischen den beteiligten Autoren. Hierfür wird eine Szene in Verantwortungsbereiche aufgeteilt, die den Autoren zugewiesen werden. Diese haben dann (bearbeitenden) Zugriff auf diese Bereiche der Szene. Für jede Produktion wird ein *Direktor* festgelegt. Er legt neue Projekte an, legt Autoren, Gruppen und deren Rechte fest und hat die Verantwortung für den Schutz der Daten vor unbefugten Zugriffen. Für die Steuerung der Zugriffe auf die Elemente des Szenengraphen wird ein *Zugriffskonzept* erarbeitet, das eine Zusammenarbeit mehrerer Autoren an einer Szene ermöglicht.

5.3.1 Zugriffskonzept

Ausgangspunkt für das Zugriffskonzept bildet ein Modell zur Formulierung von Zugriffsrechten nach Lampson [Lam71]. Aus der Menge der Nutzer (Subjekte), der Menge der Objekte und einer endlichen Menge von Zugriffsrechten wird eine Zugriffsfunktion gebildet, die durch eine Zugriffsmatrix dargestellt wird (siehe Tabelle 5.3). Die Tabelleneinträge bestehen aus Rechten (Lesen oder Schreiben), die es dem Subjekt ermöglichen, eine bestimmte Operation auf das Objekt anzuwenden. Die Matrix wird schnell sehr groß, unübersichtlich und unhandlich. Eine direkte Umsetzung des Modells in ein zweidimensionales Feld ist aus mehreren Gründen nicht sinnvoll. Die Zugriffsmatrix ist in der Regel nur sehr dünn besetzt, d. h. viele Felder sind leer und die Subjekte und Objekte ändern sich häufig. Daher ist eine vollständige Speicherung nicht effizient. Alternative Ansätze entstehen durch die Teilung der Zugriffsmatrix in Zeilen und Spalten.

	Objekt 1	Objekt 2	Objekt 3	...
Subjekt 1	lesen	lesen	schreiben	
Subjekt 2		schreiben	lesen	
Subjekt 3	lesen	lesen	schreiben	
...				

Tabelle 5.3: Zugriffsmatrix nach Lampson [Lam71]

Zugriffskontrolllisten (Access Control Lists, ACL): Es werden die nicht-leeren Spalten der Zugriffsmatrix gespeichert. Diese beschreiben die Rechte von Subjekten an einem Objekt. Kommt ein Subjekt in der Liste nicht vor, hat es auch keine Rechte auf das betreffende Objekt. Nutzer mit denselben Rechten werden oft in Gruppen zusammengefasst.

Beispiel:
$acl(Objekt1) = \{(Subjekt1, \{lesen\}), (Subjekt3, \{lesen\})\}$
$acl(Objekt2) = \{(Subjekt1, \{lesen\}), (Subjekt2, \{schreiben\}, (Subjekt3, \{lesen\})\}$
$acl(Objekt3) = \{(Subjekt1, \{schreiben\}), (Subjekt2, \{lesen\}), (Subjekt3, \{schreiben\})\}$

Zugriffslisten finden beispielsweise Verwendung bei der Beschreibung von Dateizugriffsrechten in den meisten Betriebssystemen (z. B. UNIX, Linux und Windows).

Befugnislisten (Capability Lists): Es werden die nicht-leeren Zeilen der Zugriffsmatrix gespeichert. Diese beschreiben die Rechte eines Subjektes an den Objekten. Zugriffsrechte an einem Objekt lassen sich leicht ablesen.

Beispiel:
$cap(Subjekt1) = \{(Objekt1, \{lesen\}), (Objekt2, \{lesen\}), (Objekt3), \{schreiben\})\}$
$cap(Subjekt2) = \{(Objekt2, \{schreiben\}), (Objekt3), \{lesen\})\}$
$cap(Subjekt3) = \{(Objekt1, \{lesen\}), (Objekt2, \{lesen\}), (Objekt3), \{schreiben\})\}$

Für das Zugriffskonzept auf Szenenbeschreibungen ist es erforderlich, einzelne Elemente vor unbefugten Zugriffen zu schützen. Szenenteile werden dabei durch Knoten innerhalb der XMT-Dokumente repräsentiert. Bei Zugriffen auf Knoten muss schnell feststellbar sein, welche Rechte ein Autor besitzt. Dies entspricht der Funktionsweise von Befugnislisten.

5.3.2 Zugriff auf Szenenbeschreibungen

Die Steuerung der Zugriffe auf Elemente des Szenengraphen erfolgt auf Basis der Knoten in der Baumstruktur eines XMT-Dokumentes. Alle Nutzer sind durch Namen eindeutig identifizierbar und können zu Gruppen zusammengefasst werden. Jeder Autor muss einer Gruppe zugewiesen werden. Zugriffsrechte werden in Befugnislisten gespeichert. Die Knoten bilden die Menge der Objekte, denen Rechte zugewiesen werden. Diese Listen besitzen einen eindeutigen Namen. Es können mehrere Listen in einem System existieren; gleiche Knoten können auch unterschiedliche Rechte erhalten.

Mindestens eine Befugnisliste muss einer Nutzergruppe zugewiesen werden, um eine Wirkung zu erzielen. Hierdurch ist die Strukturierung von Zugriffsrechten in Gruppen möglich. Im Falle einer mehrfachen Zuweisung von Knotenrechten zu einem Nutzer wird

die höchste Einschränkung wirksam. Grundsätzlich ist die Vergabe von Knotenrechten für jeden einzelnen Knoten durchführbar. Knoten, die nicht in einer Befugnisliste beschrieben sind, unterliegen keinen Beschränkungen. Ist ein Knoten nicht erreichbar, sind auch alle untergeordneten Knoten nicht erreichbar. Damit lassen sich die Zugriffe für ganze Zweige des Szenengraphen mit einem Eintrag steuern. Da sich in einem Dokument Knoten befinden können, die nur durch ihre Attribute voneinander zu unterscheiden sind, ist die Berücksichtigung der Attributwerte bei der Rechtevergabe zu beachten. Die Realisierbarkeit eines solchen Zugriffskonzeptes ist eine wichtige Bedingung bei der Auswahl eines Datenbankmanagementsystems. Die Abbildung 5.4 zeigt das Prinzip der Zuordnung von Zugriffsrechten.

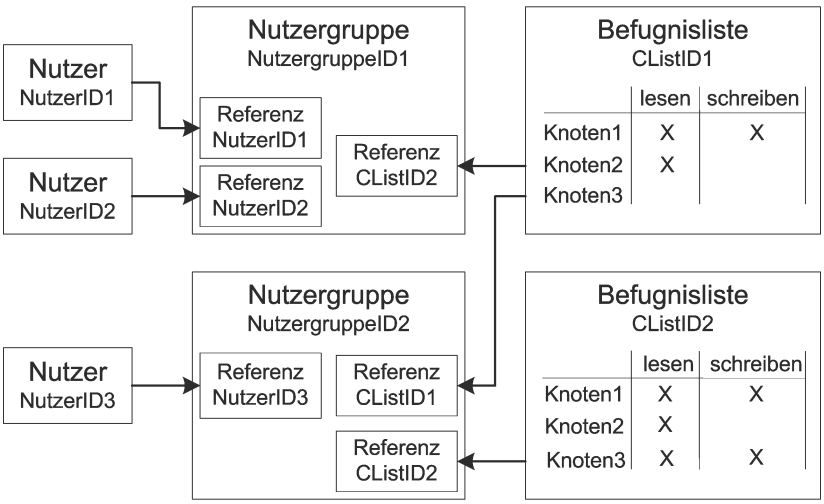

Abbildung 5.4: Knotenzugriffskonzept

Vor Beginn des Authoring-Prozesses legt der Direktor die Berechtigungen der Autoren auf Bereiche der Szene fest, wobei sich die Bereiche auch überlappen dürfen. Während des Authorings können gerade bearbeitete Elemente vorübergehend für andere Autoren gesperrt werden, um Kollisionen bei der Bearbeitung zu vermeiden.

5.3.3 Zugriff auf Medienobjekte

Die Steuerung der Zugriffe auf Medienobjekte während des Authorings kann wesentlich einfacher gestaltet werden als bei den Szenenbeschreibungen. Gewöhnlich haben alle Autoren lesenden Zugriff auf alle Objekte, um diese in Szenen zu integrieren. Lediglich objektbezogene Metadaten werden von Autoren erzeugt bzw. bearbeitet. Für Objekte genügt eine einfache Zuordnung von Lese- und Schreibrechten. Soll der Authoring-Server auch den Medienproduzenten zur Verfügung stehen, sind zusätzlich Mechanismen zum Sperren bzw. Freigeben von Objekten vorzusehen. Hierfür sind beispielsweise die Möglichkeiten von *WebDAV* ausreichend.

Der Zugriff auf Medienobjekte und andere nicht strukturierte Daten kann in gleicher Weise wie bei Szenenbeschreibungen erfolgen. Objekte werden immer als Ganzes behandelt, d. h. es erfolgt eine Zugriffssteuerung auf Objektebene. Ausnahmen sind hierbei Objekte, die selbst auf einer Szenenbeschreibung beruhen. Diese können je nach Anwendung sowohl als Objekte als auch als strukturierte Daten behandelt werden.

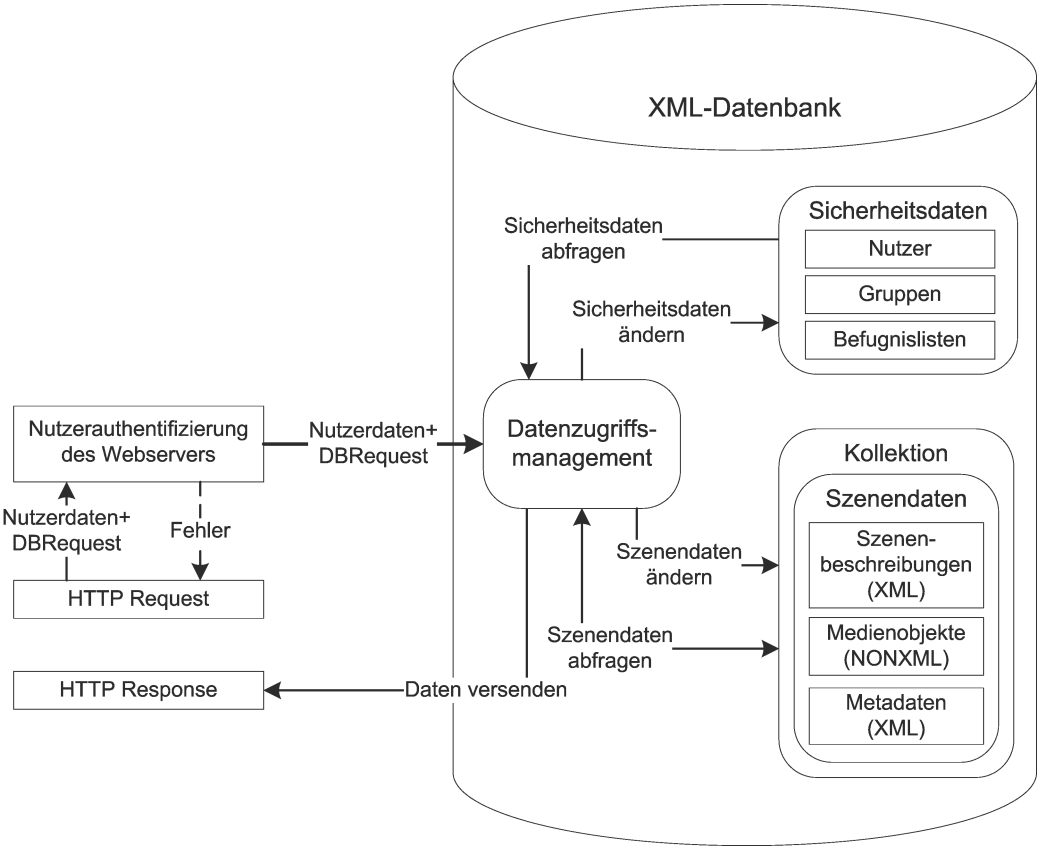

Abbildung 5.5: Datenbankmanagement des Authoring-Servers

5.4 Architektur

5.4.1 Datenbank

Die anfallenden Daten müssen in der Datenbank organisiert werden. Alle zu einem Projekt gehörenden Daten werden in einer so genannten *Kollektion* zusammengefasst, die als Datenordner betrachtet werden kann. Im Falle des Authoring-Servers besteht eine Kollektion aus XML-Dokumenten (Szenenbeschreibungen und Metadaten) und Nicht-XML-Daten (Medienobjekte), die gemeinsam verwaltet werden. Jede Kollektion benötigt Schemadefinitionen aller verwendeten Dokumente, auf deren Basis die Validierung der Daten durchgeführt wird. Es werden nur Dokumente verarbeitet, die einem Schema entsprechen. Auch die Eigenschaften von Nicht-XML-Daten werden mit einem einfachen Schema beschrieben. Abbildung 5.5 zeigt den Aufbau der Datenbank.

Tamino verwendet eigene Schemadefinitionen: die *Tamino-Schemadefinitionen (TSD)*. Eine direkte Unterstützung von XML-Schema ist momentan nicht möglich. Bevor Daten in der Datenbank abgelegt werden können, müssen entsprechende TSD erstellt werden. Diese legen die Regeln für die zu verwaltenden Dokumente fest. Das Prinzip ist vergleichbar mit der Schemadefinition von XML-Dokumenten. Die Syntax der TSD stammt von der Schemaspezifikation des W3C, unterstützt aber nur einen Teil deren Methoden. Ein vorhandenes XML-Schema (z. B. das Schema für XMT-A) muss zunächst in eine TSD überführt werden. Hierfür sind zahlreiche manuelle Anpassungen notwendig.

5.4.2 Webserver-Umgebung

XML-Datenbanken werden auf Grund ihrer typischen Verwendung für gewöhnlich in einer Webserver-Umgebung betrieben. Alle Datenbankanfragen werden zunächst vom Webserver entgegengenommen; damit stehen dessen Autorisierungsmechanismen zur Verfügung. Bei einem Zugriff auf die Datenbank muss sich der Nutzer gegenüber dem Webserver authentifizieren. Die Daten werden nun an die Datenbank weitergegeben, welche daraufhin die Berechtigungen prüft. Abhängig von den Zugriffsrechten wird die Abfrage durchgeführt. Für die Zugriffskontrolle wird das Knotenzugriffskonzept (siehe Unterabschnitt 5.3.2) eingesetzt. Die Sicherheitsdaten (Nutzer, Gruppen und Befugnislisten) sind in Form von XML-Dokumenten in der Datenbank abgelegt und können somit wie jedes XML-Dokument bearbeitet werden.

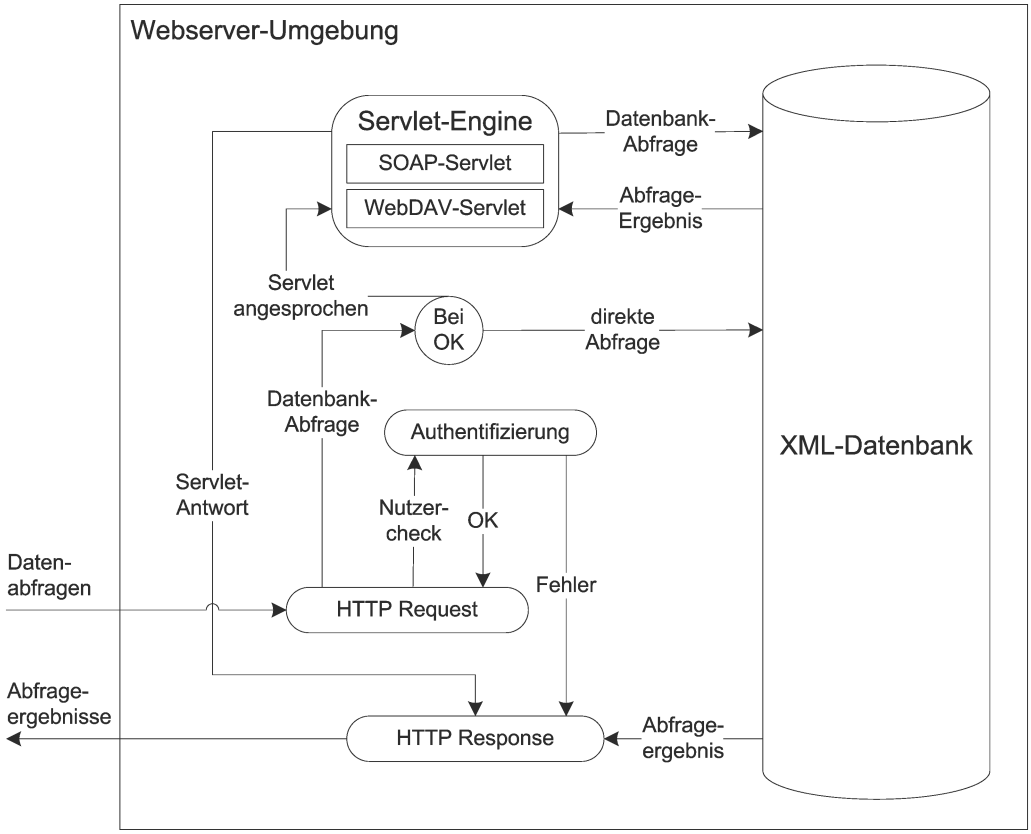

Abbildung 5.6: XML-Datenbank in einer Webserver-Umgebung

5.4.3 Kommunikation und Schnittstellen

HTTP

Die Kommunikation mit der Datenbank basiert auf HTTP. Damit ist der Server leicht in die vorhandene Infrastruktur einzubinden. Auf dem Server abgelegte Daten können über eine URL direkt angesprochen werden:

```
http://<hostname>/tamino/<dbname>/<collection>/<datei>
```

Die Kommunikation über HTTP wird in der Regel nicht durch Firewalls behindert. Aus einem Programm heraus kann durch API-Aufrufe auf die Datenbank zugegriffen werden. Diese Aufrufe erfordern die oben genannte Autorisierung.

WebDAV

Für die Übertragung von Dateien (XML und Nicht-XML) wird WebDAV verwendet (siehe Unterabschnitt 4.2.3). Zugriffe auf Dateien sind mit den Möglichkeiten von WebDAV steuerbar (z. B. Sperren und Freigeben). Jede Datei kann mit einer Versionskontrolle belegt werden; Änderungen werden in einer separaten Kollektion gespeichert. WebDAV ist über Servlets in verschiedene Webserver integrierbar und auch in vielen Programmen bereits enthalten, z. B. in *XMLSpy* und *Windows-Explorer*. Diese können somit direkt an den Authoring-Server angebunden werden.

SOAP

Eine weitere Schnittstelle für den Datenaustausch bietet das *Simple Object Access Protocol (SOAP)* [Mit03]. Es handelt sich hierbei um ein XML-basiertes Protokoll, mit dessen Hilfe Daten zwischen Systemen ausgetauscht und *Remote Procedure Calls (RPC)* durchgeführt werden können. Es eignet sich gut für den Austausch von XML-Daten. SOAP ist über Servlets in verschiedene Webserver integrierbar. In Verbindung mit der *Web Services Description Language (WSDL)* können XML-Editoren (z. B. *Altova XMLSpy*[8]) über SOAP direkt mit der Datenbank kommunizieren. Somit ist eine direkte Bearbeitung von XML-Dokumenten auf dem Server möglich.

APIs

Für die Entwicklung von Anwendungen für den *Tamino* XML-Server stehen zahlreiche Programmierschnittstellen (API) zu Verfügung. Es werden APIs für verschiedene Programmiersprachen (Java, C, PHP, .NET) und eine Umsetzung der XML:DB-API angeboten (siehe Unterabschnitt 5.2.4). Für die Realisierung des Autorensystems wurde das *Tamino API* für Java verwendet. Es können direkte Aufrufe aus einem Programm heraus getätigt werden. Die Datenbankantwort ist ein XML-Dokument, welches abgefragte Daten und Metadaten enthält. Metadaten sind z. B. Anzahl der gefundenen Dokumente oder die datenbankinterne *SpeicherID*. Das *Tamino API* für Java ist eine Java-Klassenbibliothek, die bei der Programmierung von Datenbankapplikationen verwendet wird.

5.5 Zusammenfassung

Der in diesem Kapitel entwickelte Authoring-Server ist die technische Grundlage für die gemeinsame Erstellung objektbasierter AV-Anwendungen durch mehrere Autoren. Er verwaltet alle bei einer Produktion anfallenden Daten und stellt diese den Beteiligten unter Berücksichtigung ihrer individuellen Berechtigungen zur Verfügung.

Die bei der Produktion objektbasierter AV-Anwendungen anfallenden Daten werden entsprechend ihrer Art und ihrer Verwendung verwaltet. Der Schwerpunkt der Entwicklung lag auf der flexiblen Verwaltung komplexer Szenenbeschreibungen. Es wurden verschiedene Verfahren für die Speicherung strukturierter Daten in Datenbanken untersucht und bewertet. Für das vorgeschlagene Autorensystem wurde eine native XML-Datenbank ausgewählt. Diese nimmt die Szenenbeschreibungen auf und verwaltet diese strukturiert

[8] http://www.altova.com/

in Form von XML-Dokumenten. Die Steuerung der Zugriffe auf Knotenebene ermöglicht es mehreren Autoren, gleichzeitig (synchron) an einem Szenengraphen zu arbeiten. Hierfür wurde ein Zugriffskonzept mit der Zuweisung von Nutzer- und Gruppenrechten auf Knoten erarbeitet. Dies eröffnet vollkommen neue Möglichkeiten der Server-basierten Zusammenarbeit mehrerer Autoren bei der Bearbeitung komplexer Aufgaben. Die Medienobjekte werden je nach Art als binäre Objekte oder ebenfalls strukturiert verwaltet; die Metadaten werden strukturiert als XML-Dokumente gespeichert.

Der vorgeschlagene Authoring-Server besteht aus einer Webserver-Umgebung und einem darin eingebetteten XML-Server. Der Zugriff erfolgt unter Nutzung verschiedener Protokolle über ein Netzwerk.

Der Authoring-Server ist die Schnittstelle zwischen den Produzenten der Medienobjekte und den Autoren und ist damit Grundlage eines vollständigen Produktions-Systems. Weiterhin ermöglicht er eine Wiederverwendung von Szenen und Szenenelementen über Produktionsgrenzen hinweg. Die strukturierte Speicherung von Szenenbeschreibungen und die Steuerung der Zugriffe auf Knotenebene kann auch Grundlage für ein künftiges Distributionssystem sein. Eine exemplarische Umsetzung ist in Abschnitt 6.4 beschrieben.

6 Anwendungen

In diesem Kapitel werden beispielhaft einige Anwendungen vorgestellt, die auf den zuvor erarbeiteten Konzepten beruhen. Die Realisierungen erfolgten mit Ausnahme von Abschnitt 6.5 am Institut für Medientechnik (IMT) im Rahmen des Forschungsprojektes IAVAS (siehe Unterabschnitt 2.5.3). Die Anwendung aus Abschnitt 6.5 wurde am Fraunhofer-Institut für Digitale Medientechnologie (IDMT) in Ilmenau entwickelt.

6.1 Präsentation von Vorträgen

Unter dem Arbeitstitel „MPEG-4 Presentation Application (MPA)“ wurde am IMT ein System zur Präsentation von Vorträgen entwickelt. Computergestützte audiovisuelle Präsentationen werden seit vielen Jahren zur Vermittlung von Sachverhalten im Rahmen von Vorträgen eingesetzt. Die verwendeten Softwaresysteme basieren meist auf einem Folien-Paradigma mit aufeinander folgenden Folien eines bestimmten Formates mit vorwiegend visuellen, aber auch akustischen Inhalten. Die Darstellung erfolgt für gewöhnlich mit einem Projektor auf einer Leinwand. Der bekannteste Vertreter solcher Lösungen ist derzeit *Microsoft PowerPoint*. Existierende Systeme haben mehrere prinzipbedingte Nachteile. Sie arbeiten meist mit proprietären Datenformaten, die untereinander inkompatibel sind und besitzen eine geringe Flexibilität bezüglich der Wiedergabe auf verschiedenen Endgeräten. Die rein visuelle Erstellung schränkt die Kontrolle über die Inhalte stark ein. Auch die Unterstützung eines auf mehrere Autoren verteilten Authoring-Prozesses wurde bisher vernachlässigt.

Der MPEG-4-Standard bietet Verfahren und Werkzeuge an, auf deren Basis ein System zur Präsentation von audiovisuellen Vorträgen realisiert werden kann. Die wesentlichen Nachteile vorhandener Lösungen können so behoben werden. Es stehen alle Möglichkeiten der Distribution einer MPEG-4-Anwendung zur Verfügung. Neben der Speicherung als MP4-Datei ist u. a. eine Streaming-Übertragung via Internet oder DVB denkbar. Nach einer umfassenden Analyse der Anforderungen wurde ein spezielles Authoring-Format auf Basis von XMT-Ω und SMIL sowie ein Werkzeug für dessen Verarbeitung entwickelt [Kno03], [Wei03], [WK03]. Dieses Format bildet die Grundlage für ein grafisches Autorenwerkzeug zur interaktiven Erstellung multimedialer Präsentationen auf Basis von MPEG-4 [RS04].

Bevor die digitale Technik in diesem Bereich Einzug hielt, wurden Vorträge und Präsentationen durch Dias, Overhead-Projektionen, Filme oder Audioeinspielungen unterstützt. Daran angelehnt ist die Informationseinheit einer Präsentation die *Folie* (Slide). Die Folie bildet den visuellen Rahmen für die Darstellung der Inhalte und enthält alle zu präsentierenden Objekte. Eine Präsentation kann prinzipiell alle Arten von Medienobjekten enthalten. Typische Elemente sind Texte, Tabellen, Grafiken, Fotos, Animationen und Videos. Einige Besonderheiten unterscheiden eine solche Präsentation von anderen multimedialen Anwendungen. Der zeitliche Ablauf ist für gewöhnlich linear, d. h. es wird mit der ersten Folie begonnen und alle weiteren folgen in einer vorher festgelegten Reihenfolge. Zwischen den Folien sind animierte Folienübergänge möglich. Innerhalb einer Folie können

Objekte in einer vorher definierten Reihenfolge verzögert oder durch Nutzereingabe erscheinen. Die Interaktionsmöglichkeiten sind damit stark eingeschränkt. Die abgeleiteten Anforderungen orientieren sich an den Erfordernissen einer Vortragspräsentation.

Die Steuerung der Präsentation auf der Wiedergabeseite ist möglichst einfach und intuitiv zu gestalten. Die Wiedergabe der erstellten Präsentationen soll auf verschiedenen Endgeräten unterschiedlicher Art und Leistungsfähigkeit möglich sein. Neben dem PC sind hier beispielsweise Set-Top-Boxen für das digitale Fernsehen oder mobile Geräte denkbar. Die dafür benötigten MPEG-4-Terminals sollen hier nicht näher betrachtet werden. Für Demonstrationszwecke wird der am IMT entwickelte MPEG-4-Player *i3d* eingesetzt.

6.1.1 Ereignis- und Aktionsmodell

Der typische Ablauf und der erforderliche Interaktionsgrad von Präsentationen weist gegenüber anderen Multimediaanwendungen einige Besonderheiten auf. Die für eine solche Präsentation benötigten Ereignisse und Aktionen bilden die Grundlage eines *Ereignis- und Aktionsmodells (EAM)* [Wei03]. Es wurde unabhängig von den Möglichkeiten von MPEG-4 entworfen und eignet sich somit auch für weitere Umsetzungen.

Eine Präsentation besteht aus einer oder mehreren Folien, die während der Präsentation in einer festgelegten Reihenfolge dargestellt werden. Sprünge zwischen Folien die nicht benachbart sind, sind weder vorwärts noch rückwärts möglich. Der Übergang zwischen zwei Folien kann durch eine Animation erfolgen. Jeder Folie sind Objekte unterschiedlichen Typs zugeordnet, die sich während der Präsentation dynamisch verändern können. Der Übergang von Folien und das Verändern von Objekten wird während des Authorings festgelegt und während der Präsentation durch *Ereignisse* gesteuert, die *Aktionen* auslösen. Innerhalb einer Präsentation gibt es drei Arten von Ereignissen zur Steuerung des Ablaufes durch den Nutzer:

- vorwärts Navigieren durch die Präsentation (*FWD-Ereignis*)
- rückwärts Navigieren durch die Präsentation (*BWD-Ereignis*)
- Auslösen einer Objektinteraktivität (*OI-Ereignis*)

Diese Ereignisse können durch definierbare Nutzereingaben ausgelöst werden, z. B. Mausklick oder Tastendruck. Ein Ereignis löst eine oder mehrere Aktionen aus. Aktionen werden auf Folien und Objekte angewendet, um deren Erscheinen oder Verhalten zu beeinflussen. Aktionen werden immer im Kontext einer einzelnen Folie betrachtet. Alle Aktionen einer Folie befinden sich in einer festgelegten zeitlichen Reihenfolge, in der sie ausgeführt werden. Folgende Aktionen sind vorgesehen:

vorwärts Betrachten, Richtungswechsel und rückwärts Betrachten.

Weiterhin sind zwei Arten von Folienübergängen vorgesehen: animierte Folienübergänge einer bestimmten Dauer und einfache Folienübergänge durch Umblendung ohne Animation.

Objekte

Die Objekte sind wie in allen multimedialen Anwendungen Träger der Information. Im vorgestellten System werden unter Berücksichtigung der Anforderungen und der Möglichkeiten von MPEG-4 folgende Objekte verwendet:

- *Textobjekt:* Das Textobjekt stellt formatierten Text dar. Es hat zusätzlich zu den allgemeinen Objekteigenschaften folgende spezielle Attribute: Schriftart, Schriftgröße, Schriftstil, horizontale Ausrichtung und Zeilenabstand.
- *Grafikobjekt:* Grafikobjekte dienen der Darstellung von rechteckigen Bildern und Grafiken. Neben Bitmap-Bildern können auch Vektor-Grafiken verwendet werden. Diese werden direkt in den Szenengraphen integriert.
- *Audioobjekt:* Das Audioobjekt dient der Wiedergabe eines Audioclips und hat keine visuellen Komponenten. Das Abspielen kann durch eine dem Objekt zugeordnete Aktion geschehen. Es ist auch eine Steuerung des Abspielens durch ein interaktives Objekt möglich. Die Steuerungsmöglichkeiten sind *Abspielen, Pause* und *neu Starten*.
- *Videoobjekt:* Das Videoobjekt dient der Wiedergabe eines Videoclips und ist immer rechteckig. Das Abspielen des Videoclips kann durch eine dem Objekt zugeordnete Aktion geschehen. Die Steuerungsmöglichkeiten sind *Abspielen, Pause* und *neu Starten*. Sie sind keine Aktionen im Sinne des EAM. Aktionen des EAM werden jedoch ungeachtet des durch Objekt-Interaktivität erzeugten Zustandes des Objektes ausgeführt.
- *Kombiobjekt:* Das Kombiobjekt fasst einzelne Objekte zusammen, um diese im EAM als ein Objekt steuern zu können. Die elementaren Objekteigenschaften der Einzel-Objekte bleiben erhalten. Alle dem Kombiobjekt zugeordneten Objekte behalten ihre Interaktivität.

Objekt-Interaktivität (OI)

Neben seinen statischen Eigenschaften kann ein Objekt auch dynamische Eigenschaften besitzen, wie z. B. Auslösen eines Hyperlinks oder Steuern eines Audio- oder Videoobjektes. Das Ziel der Hyperlink-Interaktion kann beispielsweise eine URL oder eine Datei sein. Steuert die Interaktion ein Audio- oder Videoobjekt, ist dessen Verhalten von der zugewiesenen Art der Steuerung abhängig.

Objektreihenfolge

Folien und Objekte werden in einem zweidimensionalen Raum abgebildet. Die Sichtbarkeit und Verdeckung von Objekten wird über ein Ebenenmodell bestimmt. Jede Folie besitzt ein eigenes Ebenenmodell mit durchnummerierten Ebenen. Die Nummer bestimmt die Objektreihenfolge und damit die Verdeckung.

6.1.2 Authoring-Format XMT-MPA

Die Grundlage des Präsentationssystems ist ein eigens für diesen Zweck entwickeltes Authoring-Format. Das Ziel war die Schaffung einer textbasierten Beschreibungssprache auf hoher Abstraktionsebene, die das im vorhergehenden Abschnitt beschriebene Ereignis- und Aktionsmodell (EAM) umsetzt. Das Authoring-Format wurde unter Verwendung von Elementen aus SMIL und XMT-Ω entwickelt. Somit ist eine Konvertierung in ähnliche Auszeichnungssprachen mit geringem Aufwand möglich (siehe Abschnitt 4.4). Die Syntax ist bewusst einfach gehalten, um eine manuelle Erstellung von Präsentationen zu ermöglichen. Eine Präsentation wird durch drei Arten von Dokumenten beschrieben (siehe Abbildung 6.1).

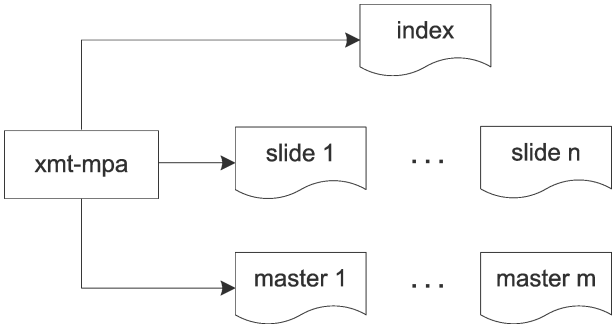

Abbildung 6.1: Dokumenttypen in XMT-MPA

Index-Dokument: Das Index-Dokument organisiert die gesamte Präsentation. Es enthält eine Liste aller Folien und legt die Reihenfolge während der Präsentation fest. Weiterhin werden Übergänge zwischen den Folien definiert. Listing 6.1 zeigt ein Beispiel für ein Index-Dokument.

```
<xmt-mpa>
 <index>
  <head>
   <meta id="author" content="author" name="Hans Muster"/>
   <meta id="name" content="name" name="Vortrag"/>
   <topLayout size="1024 768"/>
   <transition id="t1" type="barWipe" subtype="rightToLeft" dur="0.3s"/>
  </head>
  <body>
   <inline src="slide01.xml"/>
   <inline src="slide02.xml" transIn="t1"/>
   ...
   <inline src="slide27.xml"/>
  </body>
 </index>
</xmt-mpa>
```

Listing 6.1: Beispiel für ein Index-Dokument

Folien-Dokument: Die Folien-Dokumente beschreiben die Inhalte der Folien. Sie enthalten alle sichtbaren und hörbaren Objekte und deren Attribute. Es kann eine Hintergrundfarbe und evtl. eine Master-Folie angegeben werden. Jede Folie wird durch eine separate Datei repräsentiert. Eine Präsentation kann beliebig viele Folien enthalten. Listing 6.2 zeigt ein Beispiel für ein Folien-Dokument.

Master-Dokument: Die Master-Dokumente sind Vorlagen für die Folien. Sie beschreiben immer wiederkehrende Elemente, z. B. Kopf- und Fußzeilen, Logos und Randbereiche. Eine Präsentation kann mehrere Master enthalten, aber eine Folie kann nur auf einem Master aufbauen. Listing 6.3 zeigt ein Beispiel für ein Master-Dokument.

```
<xmt-mpa>
 <slide>
  <head>
   <topLayout>
    <region id="rList" size="880 540" translation="100 205"/>
   </topLayout>
  <inline src="master.xml"/>
  </head>
  <body>
   <use id="title" href="headline">Overview</use>
   <list id="list" region="rList">
    <li id="li1">Examples of applications with object based content</li>
    <li id="li2">MPEG-4 short overview</li>
    <li id="li3">Fields of research in the IAVAS project</li>
    <li id="li3_1" indent="40" fontsize="29">Authoring</li>
     ...
    <li id="li5">Demonstrations</li>
   </list>
  </body>
 </slide>
</xmt-mpa>
```

Listing 6.2: Beispiel für ein Folien-Dokument

Abbildung 6.2 zeigt den Aufbau einer Präsentation bestehend aus mehreren Folien und mehreren Master-Dokumenten. Das Authoring-Format wurde *XMT-MPA* genannt und liegt als XML-Schemadefinition (XSD) vor. Präsentationen können auf Quelltext-Ebene erstellt werden. Diese Arbeit ist mit der manuellen Erstellung von HTML-Dokumenten vergleichbar und somit für viele Benutzer zu bewältigen.

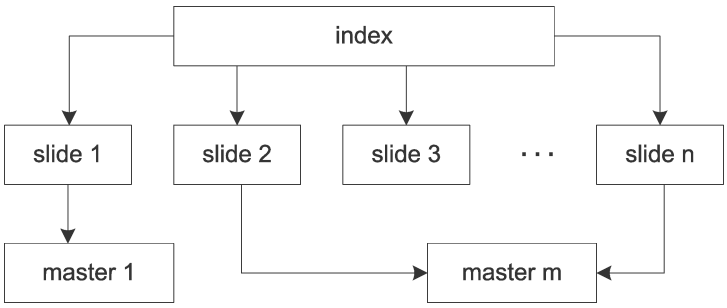

Abbildung 6.2: Dokumentenarchitektur von XMT-MPA

6.1.3 Verarbeitung

Die Verarbeitung von XMT-MPA-Dokumenten basiert auf dem in Unterabschnitt 4.4.4 entwickelten Ablauf. Zunächst wurde ein Werkzeug für die Verarbeitung der bereits als Dateien vorliegenden Präsentationen entwickelt [Wei03], [Kno03]. Abbildung 6.3 zeigt die Verarbeitungskette von XMT-MPA nach MP4. Im ersten Schritt wird das Index-

```
<xmt-mpa>
 <master>
  <head>
   <meta id="author" content="author" name="Hans Muster"/>
    <topLayout backgroundColor="#FFFFFF">
     <region id="rLinkerRahmen" size="75 768" translation="0 0"/>
     <region id="rHeader" size="960 60" translation="76 12"/>
     <region id="rTitleLine" size="950 39" translation="75 250"/>
     <region id="rTitlePicture" translation="410 290" size="280 160"/>
     <region id="rSubTitleLine1" size="950 39" translation="75 510"/>
     <region id="rFootLine" size="950 39" translation="75 500"/>
    </topLayout>
   <defs>
    <img id="linkerRahmen" src="rand_links.jpg" region="rLinkerRahmen"/>
    <img id="iavas_logo" src="iavas_logo.jpg" region="rTitlePicture"/>
    <text id="titleLine" region="rTitleLine"
    fontfamily="Arial" fontcolor="#333399" fontsize="50" align="middle"/>
    ...
    <text id="footLine" region="rFootLine"
    fontfamily="Arial" fontsize="40" fontcolor="#333399" align="middle"/>
   </defs>
  </head>
 </master>
</xmt-mpa>
```

Listing 6.3: Beispiel für ein Master-Dokument

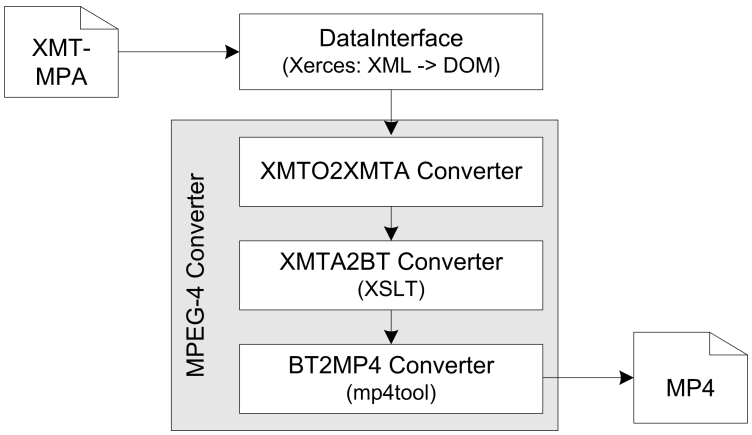

Abbildung 6.3: Verarbeitungskette für XMT-MPA

Dokument mit einem XML-Parser eingelesen. Anschließend werden alle Daten in eine Objekt-Repräsentation überführt, die als *Data Interface (DI)* bezeichnet wird. Die interne Darstellung ist ein DOM. Dieses bildet die Basis für einen Konverter zur Erzeugung der MP4-Datei. Die Konvertierung erfolgt in mehreren Schritten. Die erweiterten Elemente von XMT-MPA werden zunächst nach XMT-Ω überführt. Im nächsten Schritt wird hieraus eine Szenenbeschreibung im Format XMT-A gebildet, die anschließend mit Hilfe einer XSL-Transformation in das textuelle BIFS-Format BT überführt wird. Dieser Schritt war erforderlich, da zum Zeitpunkt der Entwicklung[1] keine geeigneten Encoder zur Verfügung standen, die XMT-A direkt nach MP4 encodieren können. Die erforderlichen XSL-Transformationen stammen aus der Referenzsoftware der MPEG-4-Standardisierung [ISO01b]. Im letzten Schritt wird diese Szenenbeschreibung zusammen mit den Ressourcen (Bilder, Videos usw.) einem Encoder übergeben, der eine binäre MPEG-4-Anwendung generiert[2]. Es liegt eine Applikation zur Erzeugung einer encodierten MPEG-4-Anwendung aus einer XMT-MPA-Präsentationen vor. Die Implementierung erfolgte in Java und beinhaltet eine einfache grafische Benutzeroberfläche. Die Architektur berücksichtigt bereits Möglichkeiten zur Realisierung grafisch-interaktiver Autorenwerkzeuge.

6.1.4 Autorenwerkzeug MPA

Aufbauend auf dem Authoring-Format XMT-MPA und dem zuvor beschriebenen Werkzeug zu dessen Verarbeitung wurde ein Autorenwerkzeug für die grafisch-interaktive Erstellung von Präsentationen entwickelt [RS04]. Die allgemeinen Anforderungen an ein Autorenwerkzeug aus Unterabschnitt 4.5.1 wurden entsprechend konkretisiert. Abbildung 6.4 zeigt die Aufgaben des Autorenwerkzeuges.

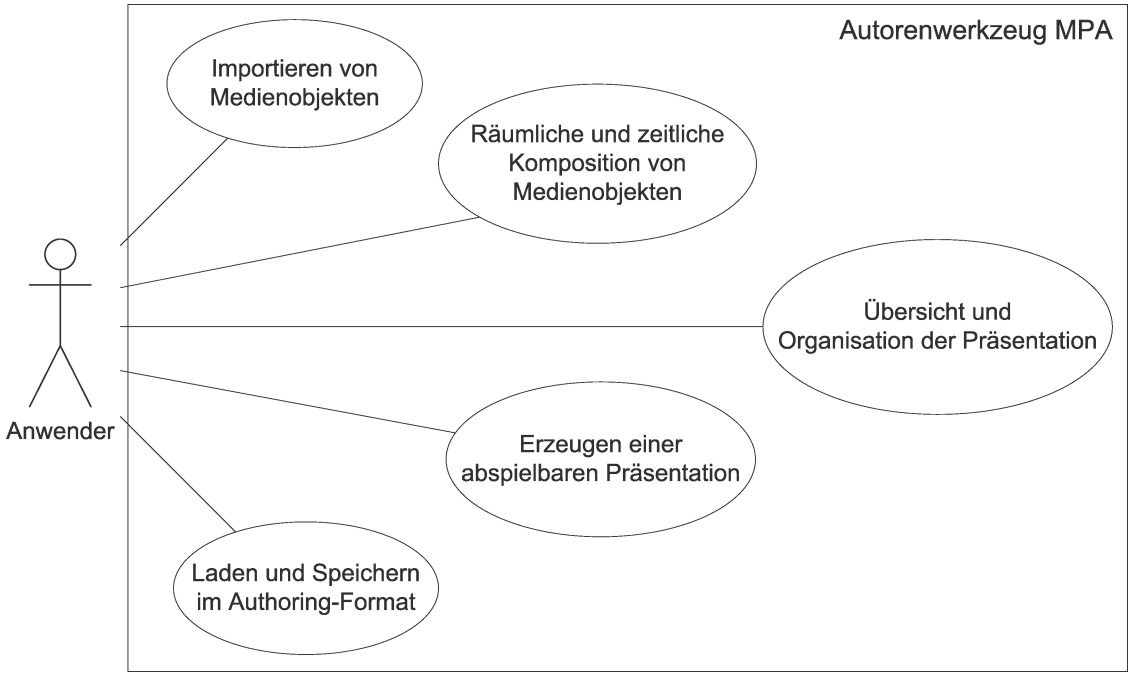

Abbildung 6.4: Aufgaben des Autorenwerkzeuges

[1] Herbst 2002
[2] Als Encoder kommt das *mp4tool* der ENST zum Einsatz.

Architektur

Die grundlegende Architektur des Autorenwerkzeuges wurde bereits in [Kno03] entwickelt. Auch hier kommt das Entwurfsmuster MVC (siehe Unterabschnitt 4.5.3) zur Anwendung. Es werden vier Editoren definiert: jeweils ein grafischer Editor und ein Quelltext-Editor für Präsentationen und Folien. Diese kommunizieren über das *Data Interface (DI)* miteinander, welches eine Objekt-Repräsentation des zu bearbeitenden Szenengraphen beinhaltet. Alle Editoren greifen auf denselben Datenbestand zu, um die Konsistenz der Ansichten zu gewährleisten. Abbildung 6.5 zeigt die Kommunikation zwischen den verschiedenen Editoren.

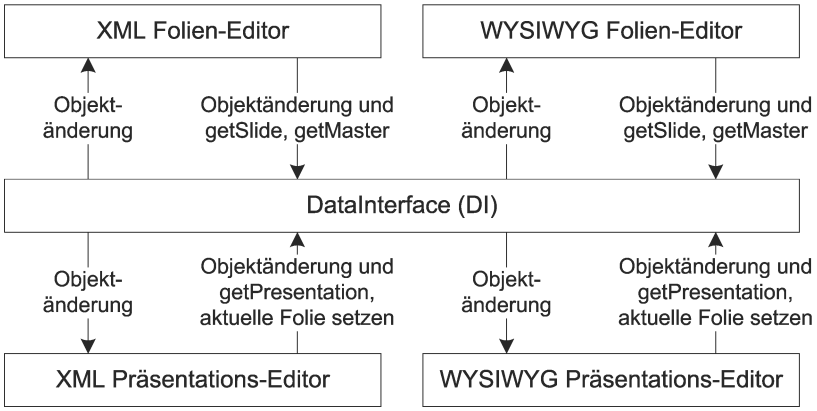

Abbildung 6.5: Kommunikation zwischen den Editoren [Kno03]

Das in [Kno03] und [Wei03] entwickelte System wurde um notwendige Module erweitert, die für ein grafisch-interaktives Authoring erforderlich sind. Das Autorenwerkzeug ermöglicht die Erstellung und Verwaltung von Präsentationen, die Verwaltung von Ressourcen und das Editieren von Folien. Weiterhin wurde ein Modul zum Abspeichern der Präsentationsdaten im Authoring-Format XMT-MPA implementiert. Im Laufe der Entwicklung wurden Erweiterungen und Verbesserungen am Authoring-Format XMT-MPA vorgenommen.

Abbildung 6.6 zeigt die Beziehungen der Module des Autorenwerkzeuges. Die Folien-Editoren haben Zugriff auf den *Ressourcen Manager*, der die Medienobjekte verwaltet. Das *Import Export Module (IEM)* dient dem Datenaustausch und der Anbindung an einen optionalen Authoring-Server. Das Modul *Converter* ist die in Unterabschnitt 6.1.3 beschriebene Verarbeitungskette zur Konvertierung von XMT-MPA nach MP4.

Benutzeroberfläche

Die Gestaltung der grafischen Benutzeroberfläche orientiert sich an etablierten Autorenwerkzeugen und ist entsprechend der Konzeption aus Kapitel 4 modular aufgebaut. Die grundlegenden Komponenten nach Abschnitt 4.5 wurden entsprechend den speziellen Anforderungen implementiert [RS04]. Abbildung 6.7 zeigt eine Ansicht der Applikation mit einer geöffneten Präsentation. In der Mitte ist der WYSIWYG-Folien-Editor (4) angeordnet. Auf der linken Seite befindet sich der *Ressourcen-Manager* (7) zur Verwaltung von Bildern, Grafiken, Videos usw. Auf der rechten Seite ist der *WYSIWYG-Präsentations-Editor* (1) zu sehen. Der Eigenschaften-Editor (6) ist unterhalb der dargestellten Folie angeordnet. Die Statusleiste zeigt Meldungen und Informationen. Alle Module sind frei positionierbar; der Nutzer kann das Layout der Benutzeroberfläche selbst bestimmen. Eine Menüleiste stellt die Funktionen der Anwendung in gewohnter Weise zur Verfügung.

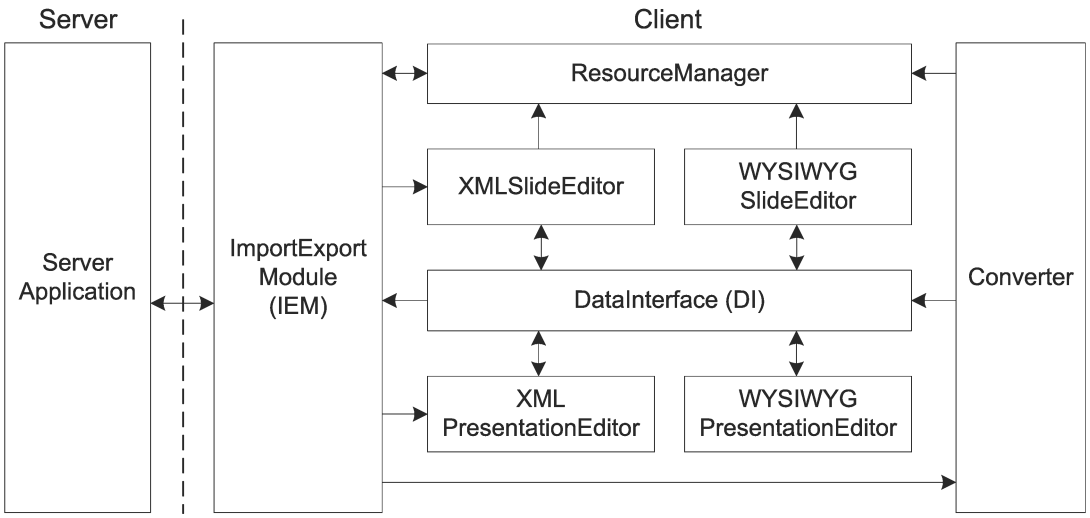

Abbildung 6.6: Beziehungen der Module des Autorenwerkzeuges MPA

Der *Präsentations-Editor* (1) erlaubt das Arrangieren der Folien zu einer Präsentation. Außerdem können die Folienübergänge festgelegt werden (3). Das Ergebnis der Bearbeitung wird durch die Index-Datei repräsentiert (siehe Unterabschnitt 6.1.2). Folgende Funktionen stehen zur Verfügung:

- Darstellung einer Übersicht aller in der Präsentation verwendeten Folien und deren Reihenfolge
- Anlegen und Entfernen von Folien
- Ändern der Reihenfolge der Folien
- Anzeigen und Editieren der Attribute für Folienübergänge
- Zuweisen von Master-Folien
- Erstellen und Entfernen von Master-Folien

Die *Folienübersicht* (2) zeigt alle momentan verwendeten Folien in einer Liste. Auf Wunsch können die verfügbaren Master-Folien und die definierten Folienübergänge aufgelistet werden. Vier Funktionen sind direkt aufrufbar:

> Folie hinzufügen, Folie entfernen, Folie in ihre Reihenfolge nach vorn verschieben und Folie in ihre Reihenfolge nach hinten verschieben.

Der *Folien-Editor* (4) dient der Erstellung und Bearbeitung von Folien nach dem WYSIWYG-Prinzip. Elemente können zu einer Folie hinzugefügt und bearbeitet werden. Es lassen sich beispielsweise Grafiken positionieren oder Texte eingeben. Bearbeitungen im WYSIWYG-Folien-Editor lösen Signale aus, die das *Data Interface* informieren. Dieses entscheidet über die Weiterleitung der Signale an den XML-Folien-Editor (Quelltext-Editor) (5) oder bei Netzwerkanbindung auch an das *Import Export Modul (IEM)*.

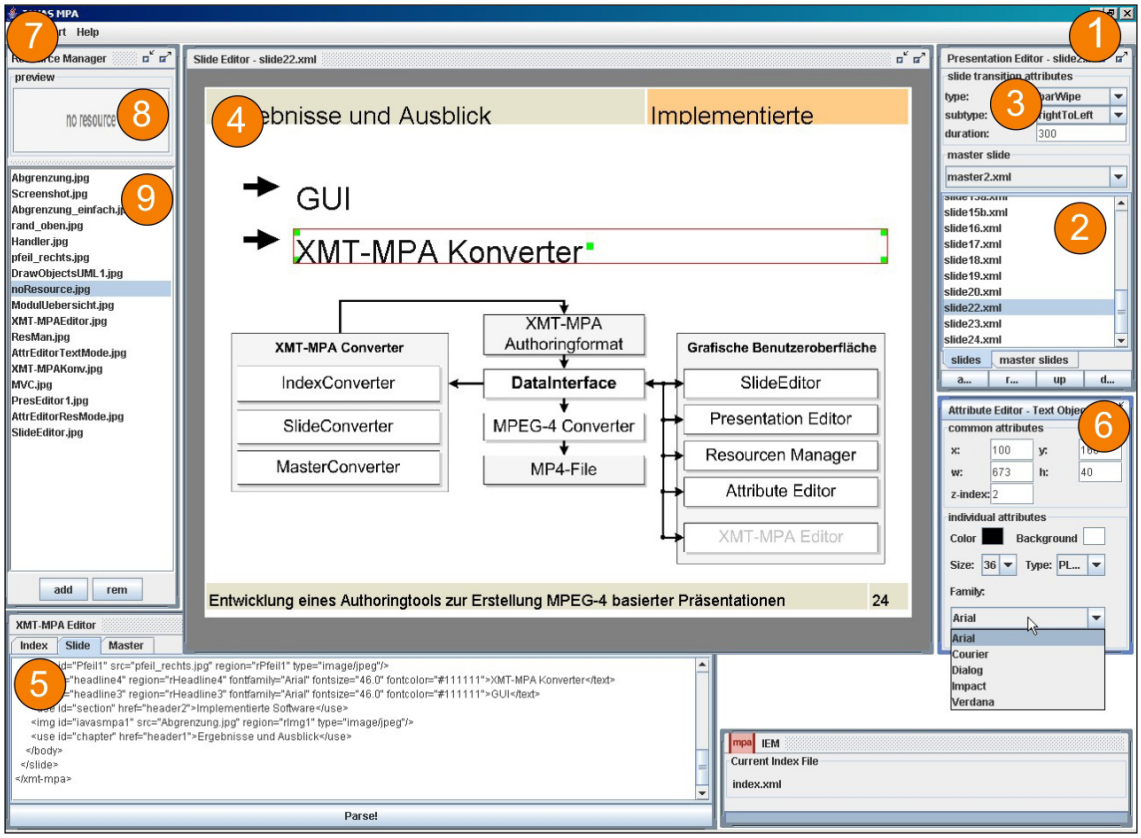

Abbildung 6.7: Benutzeroberfläche des Autorenwerkzeuges MPA

Die *Eigenschaften-Editoren* (6) dienen der Bearbeitung einzelner Objekte. Da die verschiedenen Folienobjekte auch unterschiedliche Eigenschaften besitzen, sind die angebotenen Bearbeitungsmöglichkeiten abhängig vom Typ des selektierten Objektes. Es wurden Eigenschaften-Editoren für Textobjekte und für Ressourcen-Objekte implementiert.

Der *Ressourcen-Manager* (7) verwaltet alle Medienobjekte einer Präsentation, wie z. B. Bild-, Audio-, und Videoobjekte. Die Objekte können aufgelistet (9) oder als Voransicht (8) dargestellt werden. Ressourcen können zu einer Präsentation hinzugefügt werden, ohne deren endgültige Zuordnung zu einer Folie vornehmen zu müssen.

Implementierung

Für die Implementierung der grafischen Benutzeroberfläche kam *Java Swing* zum Einsatz. Die Darstellung der Inhalte im grafischen Editor wurde mit der Grafikbibliothek *Java2D* realisiert. Weiterhin wurde ein Modul zum Abspeichern der Präsentationsdaten im Authoring-Format XMT-MPA entworfen. Die Umsetzung dieses Moduls erfolgte mit dem *XML Data Binding Framework XMLBeans* [RS04].

6.1.5 Fazit und Ausblick

Die entwickelten Komponenten Authoring-Format, Verarbeitungskette und Autorenwerkzeug demonstrieren die Möglichkeiten der in Kapitel 4 vorgeschlagenen Konzepte für das Authoring von MPEG-4-Anwendungen. Die Erstellung von Präsentationen auf Basis des Authoring-Formates XMT-MPA wird bereits praktisch am IMT genutzt. Die Bearbeitung erfolgt auf Quelltext-Ebene unter Verwendung von Vorlagen. Das grafische-interaktive Au-

torenwerkzeug ermöglicht die Erstellung und Bearbeitung von Präsentationen auf Grundlage des Authoring-Formates XMT-MPA. Die Umsetzung zeigt einige Vorzüge des Konzeptes; beispielsweise kann zu jeder Zeit der Quelltext der Präsentation bearbeitet werden und es ist eine Distribution über verschiedene digitale Kanäle möglich. Die Repräsentation der Folien durch separate Dateien erleichtert die Zusammenarbeit mehrerer Autoren auch ohne technische Lösungen. Der Funktionsumfang des Autorenwerkzeuges erreicht noch nicht die Möglichkeiten kommerzieller Produkte.

In folgenden Arbeiten muss die Anbindung an den Authoring-Server realisiert werden, um alle Vorzüge des vorgeschlagenen Authoring-Konzeptes nutzen zu können. Eine Erweiterung des grafischen Autorenwerkzeuges um einen leistungsfähigen XML-Quelltext-Editor wurde bereits in [Kno03] konzipiert. Die umfangreichen Möglichkeiten zur Objektanimation sind bisher lediglich für Folienübergänge implementiert. Künftige Versionen des Werkzeuges sollten diese Funktionalität z. B. mit Hilfe eines Zeitachsen-Editors auf Folienobjekte erweitern.

6.2 Interaktives Fernsehen

6.2.1 Authoring-Format XMT-ITA

Unter dem Arbeitstitel „Interactive Television Authoring (XMT-ITA)“ wurde am IMT ein Authoring-Format zur Beschreibung von Anwendungen für das interaktive Fernsehen entwickelt [BW04]. Der Bedarf für ein solches Format wurde bei der Bearbeitung des Projektes *iNews* (siehe Unterabschnitt 3.6.3) deutlich. Die dabei entstandenen Szenenbeschreibungen im textuellen BIFS-Format waren so umfangreich, dass sich die Bearbeitung auf Quelltext-Ebene sehr aufwändig gestaltete. Insbesondere die Steuerung der Anwendung durch Menüelemente und immer wiederkehrende Elemente (z. B. Ticker) ließen den Umfang der Szenenbeschreibungen extrem anwachsen. Durch eine Beschreibung der benötigten Elemente auf einer hohen Abstraktionsebene wurde die Grundlage für ein kompaktes, intuitiv handhabbares Authoring-Format geschaffen. Der Entwurf des Formates beschränkt sich zunächst auf die Beschreibung von 2D-Szenen. Als Ausgangspunkt diente das Format XMT-Ω mit den von SMIL übernommenen Elementen. Zur Demonstration der Möglichkeiten des Formates wurde ein prototypischer Konverter implementiert, der ausgewählte Anwendungen von XMT-ITA nach MP4 überführt. Die Entwicklung weist viele Parallelen zu den Arbeiten an XMT-MPA auf (siehe Unterabschnitt 6.1.2). Die allgemeinen Anforderungen an ein Authoring-Format aus Abschnitt 4.4 wurden entsprechend den speziellen Anforderungen einer interaktiven Fernsehanwendung konkretisiert. Ausgangspunkt war das Szenario *iNews* mit seinen typischen Elementen. Die folgenden Elemente werden beschrieben:

- Hintergrund
- Nachrichtensprecher (auch alternative Sprecher)
- Grafiken und Bilder
- Video- und Audioelemente
- Sender-Logo
- interaktiver Ticker

- Zugriff auf weitere Informationen
- Benutzerschnittstelle für eine TV-Umgebung (Auflösung, Fernbedienung)
- Möglichkeiten zur Personalisierung

Die gesamte Benutzerschnittstelle (Einstellungen, Menüs usw.) wird durch eine Szenenbeschreibung realisiert und ist somit Bestandteil der MPEG-4-Szene. Die benötigten Objekte wurden in Gruppen eingeteilt:

- *Textobjekte:* Überschriften, Untertitel, Einblendung der Uhrzeit, Menüeinträge, Nachrichtenticker
- *Grafikobjekte:* Logos, Hintergründe, Landkarten, Wetterkarten
- *Videoobjekte:* Nachrichtensprecher (beliebig geformtes Video), Videoeinblendungen (rechteckiges Video)
- *Audioobjekte:* Stimmen eines Sprechers, Audio-Einspielungen
- *Menüobjekte:* Elemente für die Benutzeroberfläche, z. B. Buttons und Checkboxen

Ein Nachrichtensprecher wird beispielsweise durch ein beliebig geformtes Videoobjekt und ein Audioobjekt repräsentiert. Beide Objekte sind unabhängig voneinander austauschbar. Weiterhin haben die meisten Objekte Ebeneneigenschaften wie Hintergrund, Rahmen, Position oder Z-Index. Abbildung 6.8 zeigt die typischen Elemente einer interaktiven Nachrichtensendung. Vor einem farbigen Hintergrund (2) sind folgende Elemente zu sehen: die Überschrift der Meldung (1), eine Nachrichtensprecherin (3) als beliebig geformtes Videoobjekt, Datum und Uhrzeit (4), ein Sender-Logo (5), eine Landkarte (6) als Grafikobjekt, die Benutzerschnittstelle mit der Menüleiste (8) und einem geöffneten Menü (7) sowie ein interaktiver Nachrichtenticker (9). Um das Authoring auf Quelltext-Ebene zu vereinfa-

Abbildung 6.8: Elemente einer interaktiven Nachrichtensendung

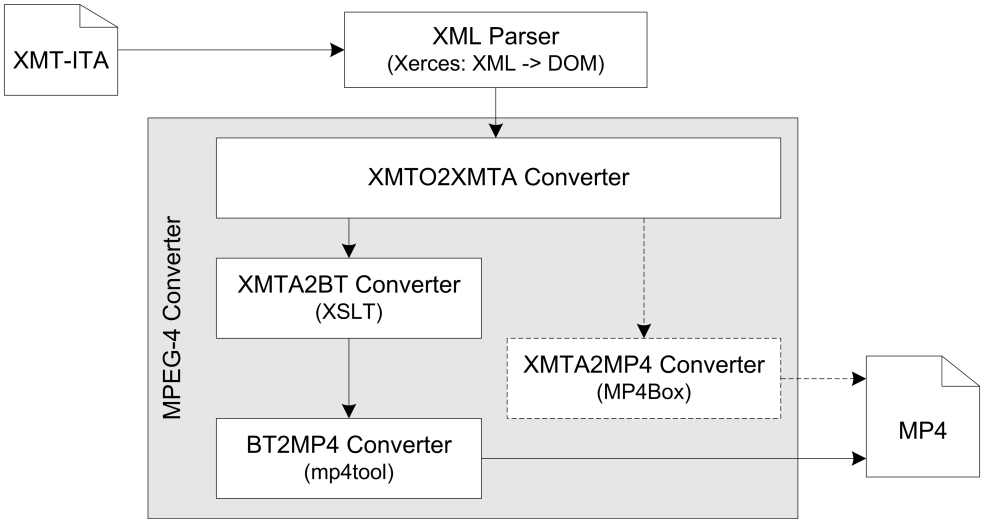

Abbildung 6.9: Verarbeitungskette für XMT-ITA

chen, wurden verschiedene Vorlagen (Templates) entworfen. Es sind z. B. fünf Vorlagen für Nachrichtenticker verfügbar. Der Autor wählt eine Vorlage aus und muss dann nur noch den darzustellenden Inhalt (Text) angeben. Dieser kann aus einer Datei ausgelesen oder von einem Server kontinuierlich empfangen werden (Streaming oder DSM-CC).

6.2.2 Verarbeitung

Die Verarbeitung des Authoring-Formates XMT-ITA erfolgt in der gleichen Weise wie die Verarbeitung von XMT-MPA (siehe Unterabschnitt 6.1.3). Ergänzend wurden hier Versuche mit dem Encoder *MP4Box* aus dem GPAC-Projekt[3] unternommen, der XMT-A direkt nach MP4 encodieren kann. Die Umwandlung von XMT-A nach BT durch eine XSL-Transformation kann so entfallen. Einige Inkompatibilitäten des Encoders veranlassten jedoch zur Nutzung des bereits bewährten Encoders *mp4tool*. Abbildung 6.9 zeigt die Verarbeitungskette von XMT-ITA nach MP4.

6.2.3 Fazit und Ausblick

Das entwickelte Authoring-Format XMT-ITA ermöglicht die Beschreibung interaktiver Nachrichtensendungen und anderer objektbasierter Fernsehanwendungen auf einer hohen Abstraktionsebene. Die grundlegenden Elemente wurden bereits implementiert. Das Element mit der höchsten Komplexität ist der interaktive Ticker. Im Vergleich zu den Szenenbeschreibungen aus iNews wird deutlich, dass die Verwendung von XMT-ITA eine wesentlich kompaktere und damit leichter zu bearbeitende Beschreibung einer Anwendung ermöglicht. Die benötigten Funktionen wurden basierend auf XMT-Ω und SMIL implementiert. Für das Authoring einfacher Anwendungen kann ein XML-Quelltext-Editor genutzt werden.

Im nächsten Schritt wird ein grafisch-interaktives Autorenwerkzeug entwickelt. Hierbei kann auf die Erfahrungen bei der Entwicklung des Autorenwerkzeuges für Präsentationen zurückgegriffen werden (siehe Unterabschnitt 6.1.4). Auch weitere Funktionen sind denkbar. Eine direkte Integration von Vektorgrafiken würde beispielsweise den Import

[3] http://gpac.sourceforge.net/

von SVG-Grafiken in die XMT-Repräsentation ermöglichen. Hiermit könnten interaktive Landkarten, Wetterkarten usw. realisiert werden. Eine grundlegende Erweiterung wäre die Beschreibung von dreidimensionalen Szenen.

6.3 Autorenwerkzeug 3dAuthor

Unter dem Arbeitstitel „3dAuthor“ wurde am IMT ein universell einsetzbares grafisch-interaktives Autorenwerkzeug zur Erstellung dreidimensionaler AV-Anwendungen auf Basis der in Kapitel 4 entwickelten Konzepte und Architekturen entworfen und partiell umgesetzt [Zie04]. Im Gegensatz zu dem in Unterabschnitt 6.1.4 vorgestellten Autorenwerkzeug wurde der Anwendungsbereich nicht eingegrenzt. Das Ziel der Arbeiten an 3dAuthor ist die umfassende Umsetzung der in Abschnitt 4.5 beschriebenen Ansätze. Als Authoring-Format wird zunächst XMT-Ω mit eingebetteten XMT-A-Elementen verwendet. Die Nutzung spezieller XML-basierter Authoring-Formate (siehe Unterabschnitte 6.1.2, 6.2.1 und 6.5.1) ist vorgesehen.

6.3.1 Benutzeroberfläche

Die grafische Gestaltung der Benutzeroberfläche orientiert sich an etablierten Autorenwerkzeugen. Abbildung 6.10 zeigt die grafische Benutzeroberfläche des Autorenwerkzeuges. Die logische Struktur der Szene wird im Struktur-Editor (1) dargestellt und kann dort verändert werden. Die Medienobjekt-Bibliothek (2) listet alle Medienobjekte einer Szene auf; für jedes Medienobjekt kann eine Vorschau (5) betrachtet werden. Die Eigenschaften von Szenen-Knoten werden im Eigenschaften-Editor (3) verändert, der durch eine editierbare Tabelle dargestellt wird. Gleiches gilt für den Metadaten-Editor (4). Der Interaktions-Editor (6) besteht aus einem Grafikfenster mit 2D-Ansicht und erlaubt die Festlegung der Beziehungen zwischen Szenenobjekten über ein Drop-Down-Menü oder durch Drag&Drop. Mit Hilfe des Zeitachsen-Editors (7) können zeitliche Abfolgen festgelegt oder Animationen erstellt werden. Zeitachsen-Editor und Interaktions-Editor repräsentieren jeweils nur einen lokalen Bereich der Szene, der durch den Struktur-Editor ausgewählt wird. Durch diese begrenzte Sicht auf die Daten wird ein besserer Überblick erreicht. Im unteren Mittelteil befinden sich schließlich die Bühne (8) und der Quelltext-Editor (9). Die Bühne gibt dem Autor einen unmittelbaren Eindruck von der erstellten Szene und kann zur direkten Manipulation von Objekten verwendet werden. Der Quelltext-Editor ermöglicht die direkte Bearbeitung der Szene auf Quelltext-Ebene. Während der linke und der mittlere Teil Sichten auf das interne Datenmodell darstellen, befinden sich im rechten Teil externe Elemente. Dazu gehören eine Objektbibliothek (b), ein Datei-Browser für das lokale Dateisystem (a) und ein Datenbank-Browser für den Authoring-Server (c). Um das Auffinden von Elementen zu erleichtern sind die Browser mit einer Vorschau auf die Medienobjekte (d) und deren Metadaten (e) verbunden.

Elemente

Das *Hauptfenster* der Benutzeroberfläche ist in die Bereiche Menüleiste, Werkzeugleiste, interner Bereich für die Editoren und externer Bereich aufgeteilt. Die Strukturierung der letzten beiden Bereiche erfolgt über *Split Panes*, die ineinander geschachtelt sind. Diese Art der Darstellung ermöglicht es dem Autor, die Größe der einzelnen Unterbereiche über Trennbalken an die eigenen Bedürfnisse anzupassen. Alternativ wurde eine Variante mit schwebenden Fenstern getestet. Hierbei legt der Nutzer die Positionierung der Fenster

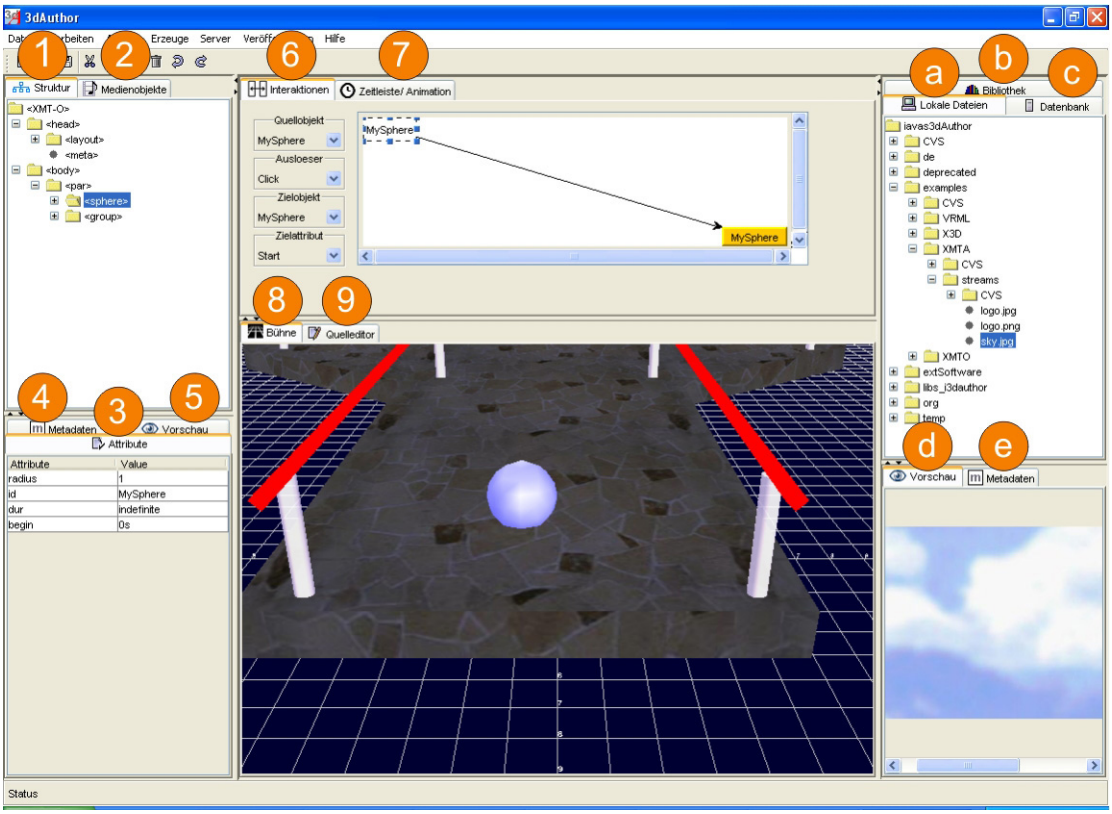

Abbildung 6.10: Benutzeroberfläche von *3dAuthor*

individuell fest. Die Fenster können sich überlappen, wodurch die Darstellung jedoch unübersichtlich werden kann.

Der *Struktur-Editor* visualisiert den XMT-Ω-Szenengraphen als Baumstruktur. Beim Einlesen der XML-Datei in das Werkzeug wird das zugehörige Baummodell mit den Daten des XML-Dokumentes gefüllt. Der *Datei-Browser* stellt die Verzeichnisstruktur des lokalen Dateisystems als Baum dar. Selektiert der Nutzer eines der Objekte im Datei-Browser, wird eine Voransicht in der Medienvorschau dargestellt. Der *Quelltext-Editor* dient der manuellen Bearbeitung von Szenendaten auf Quelltext-Ebene. Die Schlüsselelemente einer XMT-Ω- bzw. einer XMT-A-Szene werden farbig hervorgehoben.

Die *Einstellungsdialoge* dienen der Festlegung von Einstellungen für die Kommunikation mit dem Server und für die Distribution der Szene. Mit den Dialogen korrespondieren XML-Dateien, die eine Speicherung der Eingaben erlauben und beim Aufruf der Dialoge geladen werden. Das *Hilfesystem* basiert auf dem *JavaHelp*-API. Es zeigt ein Inhaltsverzeichnis, Stichwortverzeichnis und die jeweils ausgewählten Hilfe-Dokumente an. Abbildung 6.11 zeigt eine Ansicht des Hilfe-Fensters.

Die *Bühne* wurde in *Java3D* realisiert und bietet verschiedene Hilfsmittel, die den Autor bei der Bearbeitung der Anwendung unterstützen sollen. Mit Hilfe eines beschrifteten Gitternetzes wird ein Koordinatensystem in der x-z-Ebene erstellt. Es können zusätzlich zu den Eigenschaften der Basisklasse auch Screenshots der Bühne erstellt werden. Gemeinsam mit der Szene können somit auch Ansichten gespeichert werden. Dies kann bei der späteren Suche nach Szenen hilfreich sein, da durch eine geeignete Verknüpfung von Szenen- und Screenshot-Dateien eine Vorschau im Browser möglich ist. Die Sicht auf die Szene kann mit Hilfe der Maus gesteuert werden.

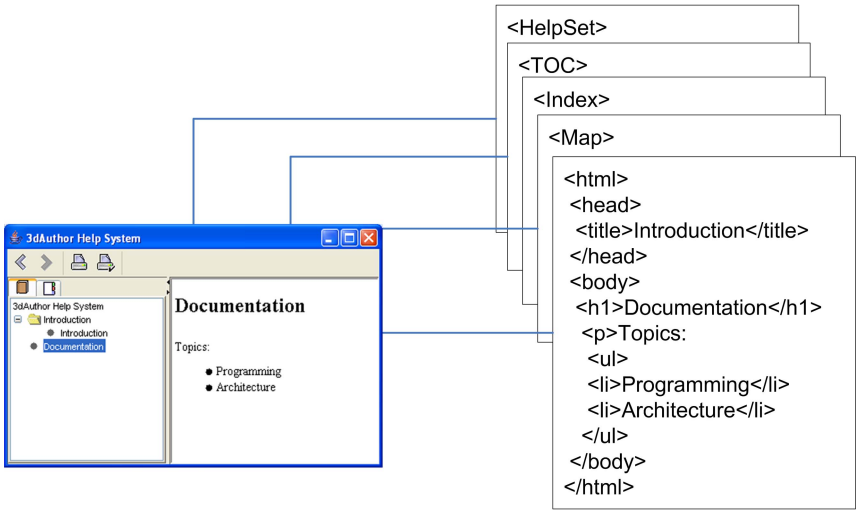

Abbildung 6.11: Hilfesystem von *3dAuthor*

Die entwickelten Konzepte sind weitgehend visuell umgesetzt, in Teilen jedoch noch ohne Funktion. Die Konzepte für den Interaktions-Editor und den Zeitachsen-Editor wurden visuell umgesetzt. Der Interaktions-Editor zeigt Objekte, die über einen Pfeil miteinander verbunden sind. Für den Eigenschaften-Editor und die Metadaten-Editoren sind Klassen und Tabellen angelegt. Am unteren Bildrand der Anwendung befindet sich die Statusleiste.

6.3.2 Architektur

Abbildung 6.12 zeigt eine abstrakte Beschreibung der Software-Architektur des Autorenwerkzeuges. Der Nutzer interagiert über Tastatur oder Maus mit der Anwendung bzw. deren Benutzeroberfläche. Wird beispielsweise ein Knoten im Struktur-Editor ausgewählt, so wird dies dem *Mediator* mitgeteilt, der daraufhin die Attribute des ausgewählten Knotens aus dem DOM ausliest und an das Datenmodell des Eigenschaften-Editors übergibt. Die Veränderung eines spezifischen Datenmodells löst die Aktualisierung des korrespondierenden *View*-Moduls des Eigenschaften-Editors aus. Ändert nun der Nutzer einen Wert im Eigenschaften-Editor und somit dem zugeordneten Datenmodell, so muss diese Veränderung auch den anderen angeschlossenen Editoren und dem eigentlichen Datenmodell, der DOM-Repräsentation des XML-Dokumentes, mitgeteilt werden. Diese Funktionen übernimmt der *Mediator* als abstrakte Funktionseinheit.

Um Daten auch auf der Bühne anzeigen zu können, muss der Szenengraph des DOM mit dem von *Java3D* synchronisiert werden. Dies wird mit Hilfe des *DOM2J3DMediators* erreicht. Wird beispielsweise ein Objekt auf der Bühne durch direkte Manipulation verschoben, so ändern sich die Translationswerte des *Java3D*-Knotens. Dies hat wiederum die Änderung des korrespondierenden DOM-Knotens zur Folge. Somit sind beide Datenmodelle synchron, aber relativ unabhängig voneinander. Auf den *Java3D*-Szenengraphen können verschiedene Sichten festgelegt werden, die der Nutzer mit den Eingabegeräten steuern kann. Durch diese Trennung wird die Austauschbarkeit der Rendering-Technologie Bühne ermöglicht.

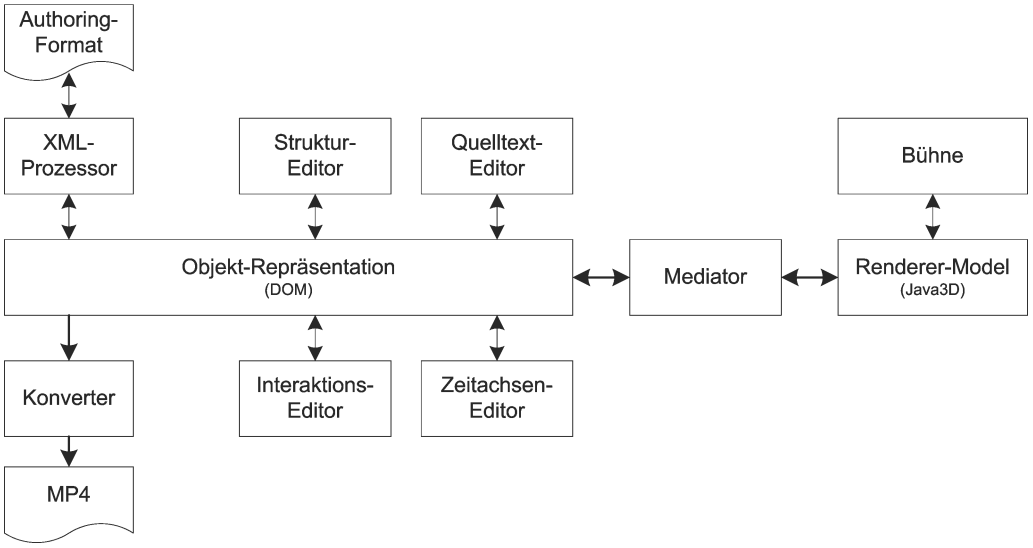

Abbildung 6.12: Software-Architektur von *3dAuthor*

6.3.3 Fazit und Ausblick

Das Projekt *3dAuthor* setzt das in Kapitel 4 entwickelte Authoring-Konzept und die vorgeschlagene Architektur prototypisch um. Hierfür wurden verschiedene technische Möglichkeiten evaluiert. Es wurde eine Benutzeroberfläche bestehend aus verschiedenen Komponenten entwickelt und verschiedene Technologien hinsichtlich ihrer Eignung untersucht. Der erreichte Stand kann als *horizontaler Prototyp* bezeichnet werden. Auf dieser Basis können nun einzelne Komponenten (z. B. Editoren) weitgehend unabhängig voneinander entwickelt und implementiert werden. Ein wichtiger Schritt hin zu einem einsatzfähigen Autorenwerkzeug ist die Erweiterung der Darstellungsmöglichkeiten und Funktionen der Bühne. Ein Schwerpunkt ist die Entwicklung von weiteren Komponenten für den Mehrbenutzerbetrieb. Die Ansätze hierfür wurden in Kapitel 5 erarbeitet.

6.4 Authoring-Server

Zur Evaluierung der in Kapitel 5 erarbeiteten Konzepte wurde ein Authoring-Server auf Basis von PC-Technik aufgebaut. Alle Komponenten wurden so ausgewählt bzw. implementiert, dass prinzipiell keine Abhängigkeit von einem Betriebssystem oder einer Plattform vorliegt. Für die Umsetzung wurde das Betriebssystem *Microsoft Windows 2000 Server* ausgewählt. Im Folgenden werden die einzelnen Komponenten beschrieben [Neu02], [Fri03], [Pöt04].

6.4.1 Komponenten

Webserver

Die Komponenten des Authoring-Servers sind in eine Webserver-Umgebung eingebettet. Ein Webserver bildet die Schnittstelle zwischen den Nutzern und dem Datenbestand. Hierfür wurde der weit verbreitete *Apache Webserver* der *Apache Software Foundation (ASF)*[4]

[4] http://www.apache.org/

ausgewählt. Der Apache Webserver verfügt über ein Sicherheitsmanagement, das Zugriffe auf Dateien und Verzeichnisse durch Passwörter schützt. Für die prototypische Umsetzung des Authoring-Servers wurde die Methode *basic authentification* eingesetzt, da hier weder hohe Zugriffszahlen auftreten, noch brisante Daten übertragen werden. Bei einer großen Anzahl zu erwartender Nutzer sollte die Methode *database authentification* zum Einsatz kommen. Der Apache Webservers ist modular aufgebaut und somit gut erweiterbar. Neue Funktionen können durch Einbindung von Modulen hinzugefügt werden, ohne den Server neu installieren zu müssen. Bei der Arbeit mit Servlets ist dies sehr hilfreich, da diese eine spezielle Arbeitsumgebung benötigen, die sich durch Module leicht realisieren lässt.

Servlet-Engine

Die Servlet-Engine ist ein Container zur Ausführung von Servlets und JSP-Skripten[5]. Servlets sind in Java programmierte, serverseitige Anwendungen, die den Funktionsumfang eines Webservers erweitern; JSP sind Serverskripte, die zur Laufzeit in Servlets übersetzt und ausgeführt werden. Ein weit verbreiteter Servlet-Container ist *Apache Jakarta Tomcat*[6], ebenfalls ein Open Source Projekt der ASF. Tomcat ist im Alleinbetrieb (als WebServer) oder auch in Verbindung mit dem Apache Webserver einsetzbar. Alle Anfragen werden zunächst vom Webserver entgegengenommen. Identifiziert dieser eine Servlet-Anfrage, leitet er diese an die Servlet-Engine weiter. Es gelten globale Sicherheitseinstellungen, d. h. diese müssen nur einmal für den Webserver vorgenommen werden und gelten dann auch für den Servlet-Container. Die Servlet-Engine wird für die Ausführung des *SOAP-Servlets* und des *WebDAV-Servlets* benötigt. Diese ermöglichen den Zugriff auf die Datenbank unter Nutzung der Protokolle SOAP und WebDAV.

XML-Server

Für den Aufbau des Authoring-Servers wurde der *Tamino XML Server* in der Version 4.1.6 unter *Windows 2000 Server* verwendet. Als weitere Plattformen kommen *Linux*, *Solaris*, *AIX* und *HP-UX* in Frage [Sof03].

Bevor Daten in der Datenbank gespeichert werden können, müssen verschiedene *Tamino*-Schemadefinitionen (TSD) erstellt werden. Diese legen die Regeln für die zu speichernden Dokumente fest. Es wurden Definitionen für Dokumente in den Formaten X3D, XMT-A, XMT-Ω erstellt. Hierfür mussten zahlreiche manuelle Anpassungen vorgenommen werden. Eine vollständige Überführung der Schemadefinition für XMT-Ω war nicht möglich, weil das entstehende Schema auf Grund seines Umfanges vom XML-Server nicht verarbeitet werden konnte. Erst durch eine Reduzierung des Funktionsumfanges der *Tamino*-Schemadefinition für XMT-Ω konnten entsprechende Szenen verarbeitet werden. Dieses Problem tritt nicht auf, wenn die verwendete XML-Datenbank den Standard für XML-Schemadefinitionen des W3C unterstützt. Die vorliegenden *Tamino*-Schemas werden den Kollektionen zugewiesen. Nun können alle XML-Dokumente gespeichert werden, deren Schemas der Datenbank bekannt sind. Für Nicht-XML-Daten muss ebenfalls eine Schemadefinition zugewiesen werden. Für diese Dateien verhält sich die Kollektion wie ein Dateiordner. Mit Hilfe folgender URL kann mit einem Webbrowser auf die gespeicherten Daten zugegriffen werden:

```
http://<hostname>/tamino/<dbname>/<collection>/<datei>
```

[5] http://java.sun.com/products/jsp/
[6] http://jakarta.apache.org/tomcat/

Die Implementierung des Knotenzugriffskonzeptes (siehe Abschnitt 5.3) setzt sich aus zwei Teilen zusammen. Der Webserver besitzt einen Autorisierungsmechanismus, der Nutzername und Passwort bei jeder Datenabfrage verlangt. Diese Daten werden vom Sicherheitsmechanismus der Datenbank übernommen. Das Autorisierungskonzept von *Tamino* erlaubt die Vergabe von Rechten auf Knotenebene. Eine Berücksichtigung von Attributen sieht das Knotenrechtekonzept von *Tamino* bisher nicht vor.

Innerhalb einer speziellen Kollektion sind drei Dokumenttypen definiert: jeweils ein Dokument für Nutzer, Gruppen und Knotenrechte. Letzteres wird als ACL bezeichnet, obwohl Befugnislisten (*Capabilities*) gespeichert werden. Abbildung 6.13 zeigt die Beziehungen zwischen diesen Sicherheitsdokumenten.

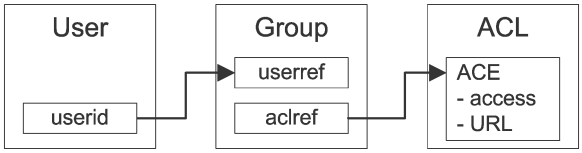

Abbildung 6.13: Dokumente der Sicherheits-Kollektion [Sof03]

Folgende Prinzipien gelten für die Autorisierung:

- Jeder zu schützende Knoten muss in einem *Tamino*-Schema definiert oder als XML-Inhalt in der Datenbank gespeichert sein.
- Jeder Knoten (und seine Kindknoten) kann individuell geschützt werden. Ist der Elternknoten durch Einschränkungen nicht erreichbar, kann auch sein Kindknoten nicht erreicht werden.
- Ist ein Knoten nicht als *Access Control Element (ACE)* definiert, existiert auch kein Schutz.
- Jeder Nutzer muss Mitglied einer Gruppe sein.
- Nutzer, Nutzergruppe und ACL dürfen jeweils nur einmal vorkommen.
- Integrität kann nicht garantiert werden (Auch gelöschte Knoten können Rechte besitzen. Rechte können bestehen bevor es den Knoten in der Datenbank gibt.).
- Geschützte Knoten werden nicht an unautorisierte Nutzer gegeben.
- Nutzernamen werden vom Webserver verwaltet.
- Beinhaltet ein *Tamino*-Update-Konstrukt geschützte Knoten, wird es nicht ausgeführt.
- Neue ACE überschreiben vorhandene gleichnamige ACE.

Die Dokumente der Sicherheits-Kollektion lassen sich wie jedes andere XML-Dokument bearbeiten. Nach der Speicherung tritt die geänderte Rechtesituation sofort in Kraft. Die Berechtigungen werden vom Direktor vor Beginn des Authoring-Prozesses festgelegt. Autoren können einzelne Elemente temporär sperren, wenn sie diese gerade bearbeiten. Eine Produktion wird durch eine Kollektion in der Datenbank repräsentiert. Es gibt ein ACL-Dokument in einer Datenbank. Darin werden Regeln in folgender Form gespeichert:

```
[Collection]/[DocumentRoot]/[Element]/[ChildElement]/...
```

Neben den Nutzern existiert im Sicherheitskonzept der Datenbank ein Administrator. Dieser darf systemspezifische Einstellungen vornehmen und wird für die Pflege des Datenbanksystems benötigt. Die Passwortverwaltung für Administratoren übernimmt das Betriebssystem.

6.4.2 Programmierschnittstellen

Um die Anbindung von Autorenwerkzeugen an den Authoring-Server zu erleichtern, wurden zwei Programmierschnittstellen (API) entwickelt [Pöt04]. Sie stellen oft benötigte Funktionen bereit und verbergen spezifische Eigenschaften des XML-Servers vor dem Entwickler. Hiermit soll eine Austauschbarkeit der XML-Datenbank ermöglicht werden, ohne dass eine Anpassung der Autorenwerkzeuge erforderlich ist. Abbildung 6.14 zeigt eine 3-Ebenen-Architektur aus Werkzeugen, APIs und Server. Die APIs abstrahieren die Funktionalität des Servers und stellen allgemein formulierte Funktionen für das Authoring und das Projektmanagement bereit. Bei einem Austausch des Servers bzw. der Datenbank muss lediglich die Implementierung der APIs angepasst werden; die Werkzeuge selbst bleiben unberührt. Die implementierten APIs nutzen das *Tamino Java API* und realisieren die Kommunikation der Werkzeuge mit dem Server über ein Netzwerk.

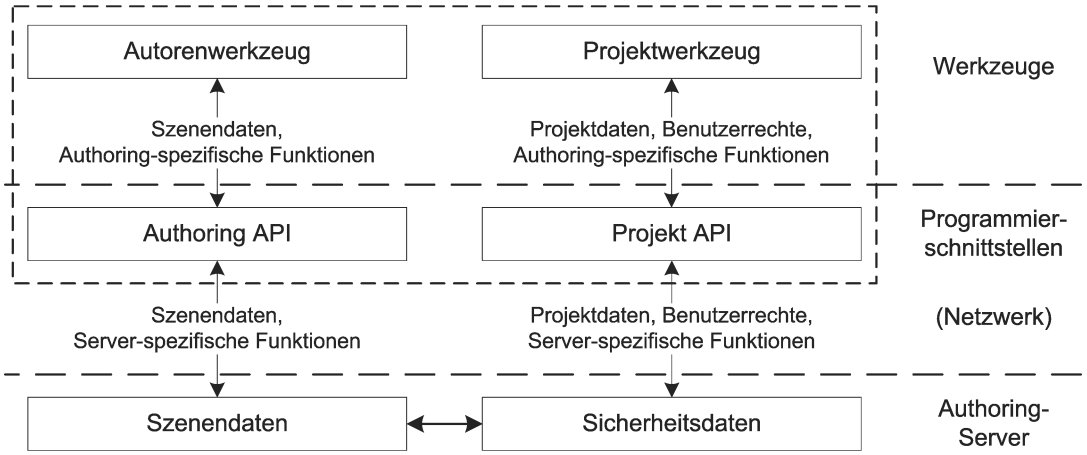

Abbildung 6.14: Programmierschnittstellen des Authoring-Servers

Authoring-API

Das Authoring-API stellt grundlegende Funktionen für die Bearbeitung von Szenen auf dem Server bereit. Folgende grundlegende Funktionalität wird angeboten:

- Verbindungsaufbau zum Authoring-Server
- Anmeldung am Authoring-Server
- Auflisten der Projekte
- Auflisten der Szenen eines Projektes
- Herunterladen von Szenen, Elementen, Medienobjekten vom Server

- Einfügen und Löschen von Szenen, Elementen und Metadaten
- Hochladen von Szenen, Elementen, Medienobjekten auf den Server
- Verbindungsabbau

Das Authoring-API stellt Funktionen für die Kommunikation mit dem Authoring-Server bereit. Vor einer Transaktion muss sich der Autor am Server mit Login und Passwort anmelden. Anschließend können Projekte aufgelistet und ausgewählt sowie die zu bearbeitende Szene oder Szenenelemente heruntergeladen werden. Beim Hochladen der bearbeiteten Elemente auf den Server wird überprüft, ob der Autor über die erforderlichen Rechte verfügt. Das Einfügen von Elementen in eine Szene umfasst die Bearbeitung von Szenenbeschreibungen und Metadaten sowie das Hinzufügen von Medienobjekten. Es können beispielsweise neue Knoten zur Szenenbeschreibung hinzugefügt oder vorhandene Knoten bearbeitet werden.

Projekt-API

Das Projekt-API stellt grundlegende Funktionen für die Verwaltung von Projekten auf dem Server bereit. Projekte können angelegt und gelöscht sowie Zugriffsrechte bearbeitet werden. Entsprechend den Anforderungen an die Projektverwaltung stellt das Projekt-API folgende Funktionalität bereit:

- Anlegen von Projekt und Basis-Szene
- Anzeigen von Projekten und Inhalten
- Löschen von Projekten und Inhalten
- Zuweisen der Nutzer- und Gruppenrechte für Szenen und Elemente (Knoten)
- Anzeigen der Nutzer- und Gruppenrechte von Szenen und Elementen (Knoten)
- Ändern der Nutzer- und Gruppenrechte von Szenen und Elementen (Knoten)
- Löschen von Szenen, Elementen und Medienobjekten aus einem Projekt

Die Funktion zur Anzeige von Projekten gibt eine Liste aller vorhandenen Projekte aus. Auch hierfür ist zunächst ein Verbindungsaufbau und eine Anmeldung am Server erforderlich. Bei der Erstellung eines Projektes durch den Direktor wird die Datenstruktur der Kollektion angelegt und mit einem Namen versehen. Es wird eine initiale Szene (Basisszene) angelegt und der Direktor kann Rechte auf einzelne Elemente festlegen. Für diese Aufgaben können Vorlagen (Templates) genutzt werden. Die Erstellung einer Basis-Szene ist für die anschließende Vergabe von Rechten notwendig. Beim Löschen von Szenen werden zugleich die Einträge für Nutzerrechte gelöscht. Die Abfrage der Nutzer- und Gruppenrechte dient der Anzeige bereits festgelegter Rechte. Diese Daten können vom Autorenwerkzeug in einem Dialog dargestellt werden. Das Einfügen der Nutzer- und Gruppenrechte unterteilt sich in drei Aufgaben:

- Festlegen der Nutzer-Zuordnungen
- Festlegen der Gruppen-Zuordnungen
- Festlegen der Befugnislisten

Diese Daten können mit Hilfe eines Nutzer- und Gruppenrechte-Dialoges festgelegt werden und sind auf dem Authoring-Server gespeichert.

6.4.3 Werkzeug zur Projektverwaltung

Zur Erprobung der Möglichkeiten des Authoring-Servers wurde ein Werkzeug für die Projektverwaltung entwickelt. Es besteht aus zwei Modulen und nutzt die zuvor beschriebenen Programmierschnittstellen Authoring-API und Projekt-API. Abbildung 6.15 zeigt schematisch die Funktionsweise des Werkzeuges. Für die Implementierung wurde die Programmiersprache Java und das DOM API von Xerces verwendet [Fri03], [Pöt04].

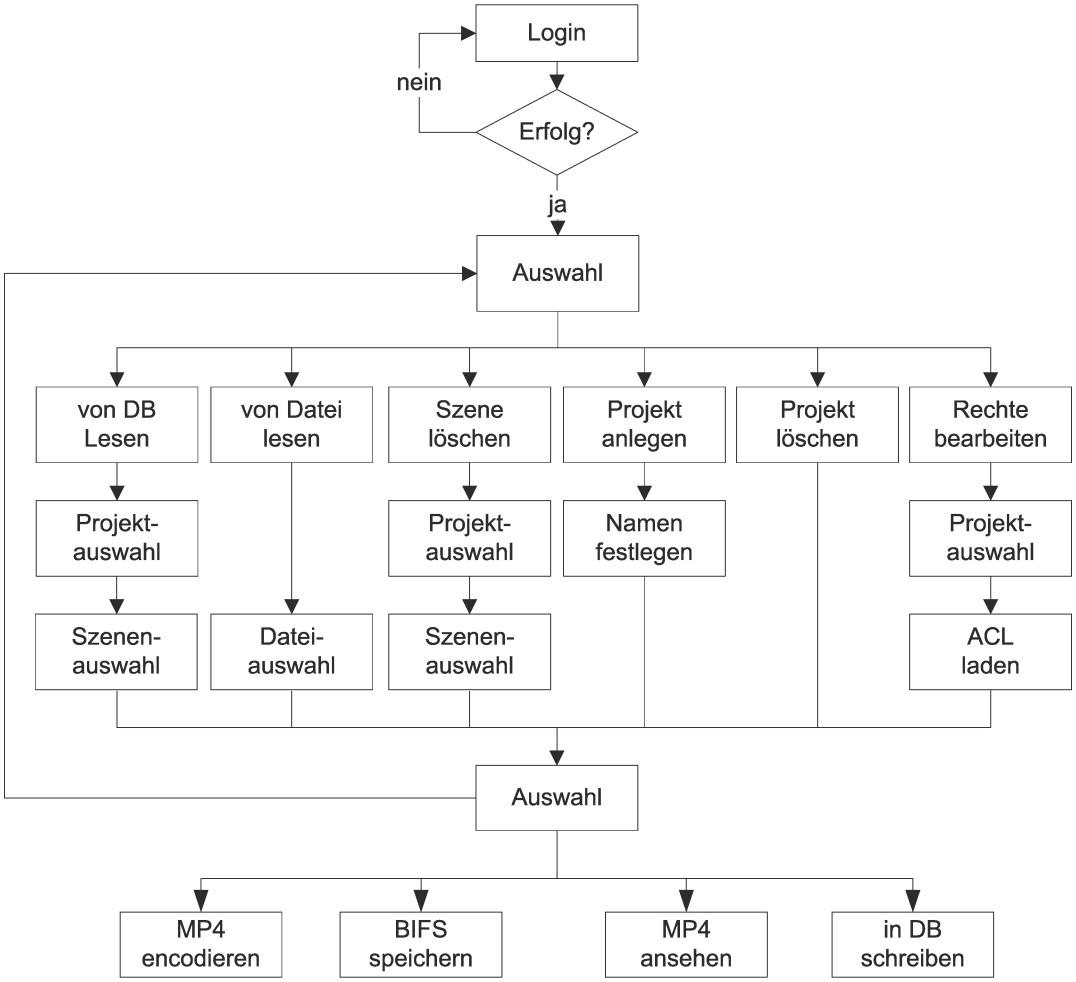

Abbildung 6.15: Funktionen des Projekt-Moduls

Projekt-Modul

Das Projekt-Modul dient der Verwaltung von Projekten und Szenendaten mit folgender Funktionalität:

- Anlegen und Löschen von Projekten
- Speichern und Löschen von Szenen
- Speicherung aller Szenendaten in der Datenbank
- Umwandlung von XMT nach textuellem BIFS (BT)
- Codierung zu einer MPEG-4-Datei

Zu Beginn einer Produktion erzeugt der Direktor ein neues Projekt. Anschließend wird eine neue Kollektion erstellt, unter der das Projekt-Modul alle zu einem Projekt gehörenden Szenendaten speichert. Nun können dem Projekt Szenendaten hinzugefügt werden. Abbildung 6.16 zeigt die grafische Benutzeroberfläche mit einer geöffneten XMT-A-Szene. Die Baumstruktur auf der linken Seite enthält ein XMT-A-Dokument, die rechte Seite das

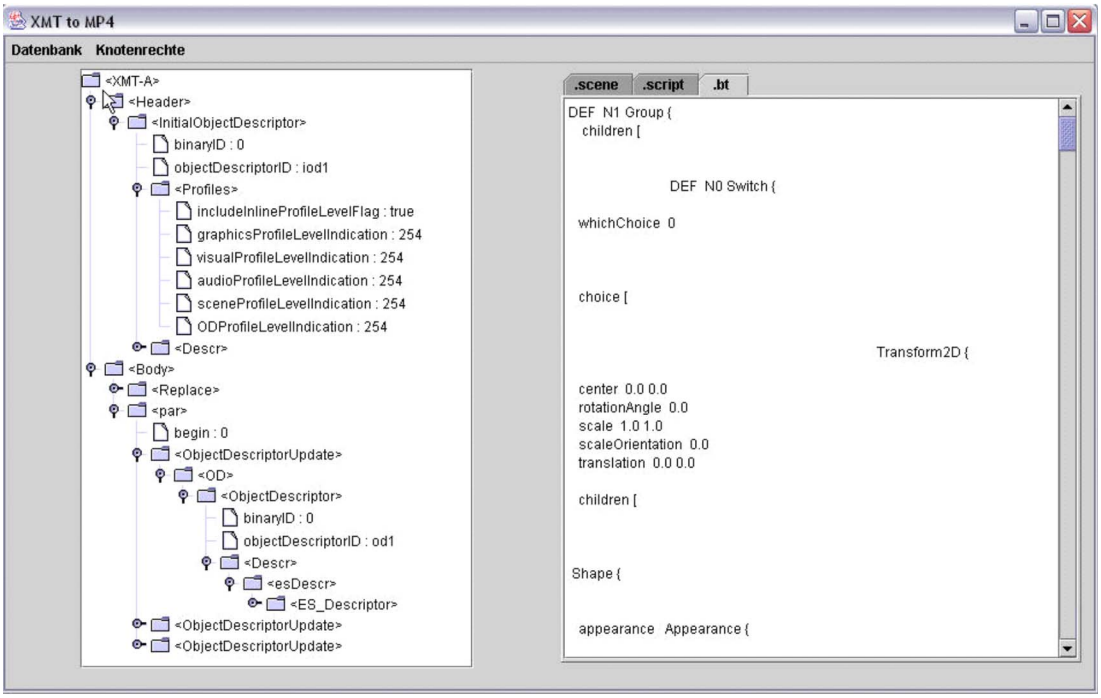

Abbildung 6.16: Projektmodul mit Szenendaten

Resultat der Transformation in das textuelle BIFS-Format BT. Alle Szenendaten können auf Wunsch auf dem Server gespeichert werden.

Rechte-Modul
Das Rechte-Modul ermöglicht dem Produktionsdirektor die Bearbeitung von Zugriffsrechten auf Knoten gespeicherter Szenendaten. Das Modul stellt folgende Funktionen bereit:

- Anlegen von Knotenzugriffslisten
- Bearbeiten von Knotenzugriffslisten
- Ändern der Beziehungen von Rechten und Nutzergruppen
- Ablegen der Knotenrechte in der Datenbank

Der Direktor hat die Möglichkeit, Knotenzugriffsrechtelisten für Szenen zu erstellen oder zu ändern. Hierfür steht eine einfache grafische Benutzeroberfläche zur Verfügung.

6.4.4 Fazit und Ausblick

Der prototypisch realisierte Authoring-Server erlaubt die Speicherung aller Arten von Szenendaten. Insbesondere die Verwaltung von Szenenbeschreibungen und die Steuerbarkeit der Zugriffe bis hinab auf Knotenebene stellt eine neue Qualität der Datenhaltung dar. Das

umgesetzte Zugriffskonzept ist die Grundlage für eine effektive Zusammenarbeit mehrerer Autoren an einem Projekt. Die entwickelten Programmierschnittstellen sind erste Schritte in Richtung eines universell einsetzbaren und erweiterbaren Autorensystems. Auf dieser Basis können Autorenwerkzeuge unabhängig vom verwendeten Server entwickelt werden. Der Funktionsumfang muss schrittweise erweitert werden. Der Authoring-Server dient als Plattform für die weitere Entwicklung des Autorensystems. Mit Hilfe der bereitgestellten Schnittstellen können Szenendaten auf dem Server sowohl mit universellen als auch mit eigens entwickelten Werkzeugen gearbeitet werden.

6.5 Wellenfeldsynthese

Das Fraunhofer-Institut für Digitale Medientechnologie (IDMT)[7] in Ilmenau beschäftigt sich mit der Entwicklung von neuartigen Systemen für die Audiowiedergabe. Die Wellenfeldsynthese (WFS) ist ein Verfahren zur realistischen räumlichen Audiowiedergabe ohne Einschränkung auf einen Sweetspot. Sie ermöglicht durch die Reproduktion eines Klangfeldes die exakte Lokalisation von Schallquellen. Die Grundlage bildet das aus der Wellentheorie bekannte Huygens´sche Prinzip sich überlagernder Wellen. Sehr viele Lautsprecher werden um den Wiedergaberaum herum platziert. Durch eine gezielte Ansteuerung der einzelnen Lautsprecher werden Wellenfronten erzeugt und es entsteht ein realistisches Klangfeld. Die theoretischen Grundlagen der WFS wurden an der TU-Delft in den Niederlanden erforscht und erstmals Ende der 80er Jahre vorgestellt. Später wurde die Technologie in einem europäischen Forschungsprojekt unter der Leitung des IDMT zur Marktreife entwickelt. Eine Umsetzung der WFS wird unter dem Namen *IOSONO Soundsystem*[8] vom IDMT vermarktet. Typische IOSONO-Installationen arbeiten mit 64 bis zu einigen hundert Lautsprechern. Anwendungsgebiete sind zunächst das Kino und die Beschallung von Live-Events. Für eine optimale Nutzung eines solchen Wiedergabesystems ist eine objektbasierte Beschreibung der Inhalte in Form einer auditiven Szene notwendig. Alle Schallquellen werden im Raum positioniert und ggf. mit einer Ausrichtung versehen. Die so erzeugte Szene kann durch eine MPEG-4-Anwendung repräsentiert werden. Die erforderlichen Autorenwerkzeuge werden ebenfalls am IDMT entwickelt [MRB+03]. Für die Distribution der Szenendaten ist ein MPEG-4-basiertes Format bestens geeignet.

6.5.1 Authoring-Format XMT-SAW

Unter dem Arbeitstitel „XMT-SAW“[9] wurde am IDMT ein Authoring-Format für die objektbasierte Beschreibung von auditiven Szenen für die Wellenfeldsynthese (WFS) entwickelt [Mün03]. Bei der Entwicklung des Formates galten zunächst die in Unterabschnitt 4.4.1 genannten Forderungen, die durch die folgenden Merkmale weiter konkretisiert wurden:

- Darstellung der virtuellen statischen und bewegten Audioquellen und deren Eigenschaften
- Definition von globalen Parametern

[7] http://www.idmt.fhg.de/
[8] http://www.iosono-sound.de/
[9] SAW - Spatial Audio Workstation

- Definition von zeitlichen Abläufen in der Szene
- Verweise auf die Audiosignale
- Darstellung von virtuellen Räumen
- Gruppierungen von Audioobjekten
- Darstellung der vom Benutzer definierten Hierarchien für die Organisation und Verwaltung einer Szene

In [MRB+03] wird die Architektur eines Autorensystems für die Produktion von Inhalten für die Wellenfeldsynthese vorgestellt. Das Werkzeug *Spatial Audio Workstation (Spamix)* erlaubt eine objektbasierte Produktion von auditiven Szenen. Aus diesem Projekt heraus wurden die Anforderungen an das Authoring-Format abgeleitet. Eine besondere Bedeutung erlangten quellenspezifische Eigenschaften, deren Speicherung im Authoring-Format erforderlich ist:

- Position der virtuellen Audioquelle im virtuellen Raum
- Verweis auf das Audiosignal
- Repräsentation der virtuellen Audioquelle durch eine ebene Welle oder durch eine Punktquelle
- objektspezifische Lautstärken der virtuellen Audioquellen
- Abstrahlcharakteristiken
- Lebenszeiten der Audioobjekte, in denen diese aktiv sind

Ausgangspunkt der Entwicklung war das Format XMT-Ω, welches um die zusätzlich erforderlichen Funktionen und Parameter erweitert wurde. Die Definition des Formates XMT-SAW wurde als XML-Schemadefinition (XSD) ausgeführt.

Aufbau einer XMT-SAW-Szene

Das konzipierte Format dient der Speicherung von auditiven Szenen und den damit verbundenen zusätzlichen Informationen in einer XML-basierten Szenenbeschreibung für die Anwendung in der Wellenfeldsynthese. Listing 6.4 zeigt die grundsätzliche Struktur einer auditiven Szene in XMT-SAW. Es erfolgt die Definition von zwei Audioobjekten, die zu einer Gruppe zusammengefasst sind. Die zweite Audioquelle verfügt zusätzlich über einen Animations-Knoten, um das Objekt in der Szene bewegen zu können. Weiterhin gibt es einen Knoten für die Beschreibung der akustischen Raumparameter. Die Objekte sind zu einem *Layer* zusammengefasst, um sie auf der grafischen Benutzeroberfläche des Autorenwerkzeuges anordnen zu können.

Globale Parameter

Zunächst werden einige globale Parameter definiert. Im Kopf der XMT-SAW-Szene werden Szenenlautstärke (für alle Audioobjekte dieser Szene gültig) und die Lautstärke eines Subwoofers festgelegt. Im Rumpf wird ein Timecode-Element für die Synchronisation aller Objekte der Szene angegeben.

```
<XMT-SAW>
 <head>
  <layout>
   <topLayout>
    <region ... />
   </topLayout>
  </layout>
 </head>
 <body>
  <timecode ... />
  <layer id="layer0">
   <room .../>
   <group ... >
    <audio id="1" ... >
     <sound .../>
    </audio>
    <audio id="2" ... >
     <sound ... />
     <animateMotion .../>
    </audio>
   </group>
  </layer>
 </body>
</XMT-SAW>
```

Listing 6.4: Struktur einer Szenenbeschreibung in XMT-SAW [Mün03]

Audioquellen

Alle Audioobjekte müssen innerhalb der Szene positioniert und ggf. bewegt werden. Der Autor kann festlegen, ob die Audioquelle durch eine ebene Welle oder durch eine Punktquelle repräsentiert wird. Die von XMT-Ω angebotenen Funktionen mussten um einige Parameter erweitert werden. Prinzipiell werden zeitunabhängige und zeitabhängige Eigenschaften unterschieden. Audioquellen werden durch die Angabe von Koordinaten positioniert. Aktuelle Systeme für die Wellenfeldsynthese erlauben eine Anordnung von Quellen nur in einer Ebene. Daher ist die z-Koordinate momentan nicht von Interesse und wird für alle Audioobjekte auf einen festen Wert gesetzt (z=0). Ein weiterer Parameter ist die Lautstärke eines Audioobjektes.

Raumdaten

Für eine realistische Audiowiedergabe ist eine Beschreibung der akustischen Eigenschaften des Raumes notwendig. Entsprechend dem wahrnehmungsbezogenen Ansatz werden hierfür Parameter wie Nachhallzeit, Absorptions- und Transmissionskoeffizient verwendet. Die Umsetzung eines physikalischen Modells wurde bisher nicht realisiert.

Animationskonzept

Das Animationskonzept ermöglicht die Bewegung der Audioobjekte in der Szene und deren Gruppierungen in Abhängigkeit von der Zeit für die Darstellung auf der grafischen Benutzeroberfläche. Das vorhandene Autorenwerkzeug generiert für bewegte Objekte ca.

alle 50 ms neue Positionsdaten, unabhängig davon, ob das Objekt wirklich seine Position ändert. Die so entstehenden Daten werden sehr umfangreich und sind auf Quelltext-Ebene nur schwer editierbar; dennoch wurde eine Speicherung innerhalb der XMT-SAW-Datei gewählt. Die Positionsdaten werden in Tupeln gespeichert: x-, y-, z-Koordinate und ein Zeitstempel. Der Zeitstempel liefert die zeitliche Referenz, zu welchem Zeitpunkt das animierte Objekt die neuen Koordinaten annehmen soll.

6.5.2 Verarbeitung

In [Mün03] wurden verschiedene Lösungen für die Verarbeitungen von XML-Daten untersucht und hinsichtlich ihrer Leistungsfähigkeit verglichen. Für die prototypischen Implementierungen wurde der XML-Parser Xerces der ASF ausgewählt. Das Authoring-Format XMT-SAW wurde prototypisch in das Autorenwerkzeug *Spamix* mit folgenden Funktionen integriert:

- *Speichern der Szenenbeschreibung im Format XMT-SAW:* XMT-SAW wurde als Alternative zum bisherigen Ausgabeformat in Spamix integriert. Beide Formate sollten parallel existieren und auswählbar sein, um Kompatibilität zu bereits realisierten Lösungen zu gewährleisten. Daher werden die Szenendaten parallel in beiden Formaten ausgegeben.

- *Einlesen der XMT-SAW-Szenenbeschreibung:* Die XML-Daten werden durch den XML-Parser eingelesen und anschließend in die interne Datenstruktur des Autorenwerkzeuges überführt.

Eine zweite Implementierung erfolgte in den Renderer des WFS-Systems. Da hier das Abspeichern von Dateien nicht gefordert ist, wurde nur das Einlesen von XMT-SAW integriert. Bevor eine Szene für die Wellenfeldsynthese abgemischt wird, erfolgt das Vor-Rendern. Hier werden die Szenendaten eingelesen und die Szenenbeschreibung initialisiert. Im Falle des bisherigen Authoring-Formates werden die Positionsdaten bewegter Audioobjekte erst während des Rendering-Prozesses aus separaten Textdateien eingelesen.

In [Gat03] wurde eine Encoder-Schnittstelle entwickelt, die u. a. das Authoring-Format XMT-SAW an einen MPEG-4-Encoder anbindet. Die Kommunikation aller Module (Autorenwerkzeuge, Encoder, Renderer usw.) erfolgt über einen Szenenbus, an dem alle relevanten Szenendaten anliegen. Um die Schnittstelle möglichst flexibel auszulegen, wurde ein duales API konzipiert: ein textuelles API und ein numerisches API. Das textuelle API ist an das DOM angelehnt; das numerische API wurde weitgehend aus der Referenzsoftware IM1 übernommen. Ergänzend zum XML-Reader und XML-Writer für den Szenenbus wird auch eine Dateischnittstelle bereitgestellt. Folgende Anwendungen wurden implementiert [Gat03]:

- *Dateischnittstelle:* Der XML-Reader parst die textuellen Informationen aus der XML-Datei und schreibt die Daten direkt über das textuelle API in den Szenengraphen. Der XML-Writer liest die Daten über das textuelle API aus dem Szenengraphen, formatiert diese und schreibt das Ergebnis in eine XML-Datei.

- *XMT-SAW nach XMT-A Konverter:* Der XMT-SAW/XMT-A-Konverter konvertiert eine Szene aus dem Format XMT-SAW nach XMT-A. Hierfür wird die XMT-SAW-Datei mit Hilfe der Dateischnittstelle in die Szenengraphenstruktur eingelesen, elementweise ausgewertet und mit Hilfe des numerischen API ein zweiter

XMT-A-Szenengraph aufgebaut. Das Ergebnis wird anschließend über die Dateischnittstelle in eine XMT-A-Datei geschrieben.

- *XMT-A nach BIFS-Text Konverter:* Mit Hilfe des BIFS-Text-Konverters kann eine Szene aus dem Format XMT-A in das BIFS-Text-Format konvertiert werden. Die XMT-A-Datei wird zunächst in die Szenengraphenstruktur eingelesen. Der BIFS-Text-Konverter liest die Szenendaten anschließend über das textuelle API aus, formatiert die Daten und schreibt das Ergebnis in die BIFS-Text-Datei.

Es liegen Lösungen vor, mit denen eine durch das Autorenwerkzeug Spamix erstellte Szene nach MPEG-4 encodiert werden kann.

6.5.3 Fazit und Ausblick

Das vorgestellte Projekt beschreibt ein Authoring-Format für die Wellenfeldsynthese sowie eine prototypische Implementierung der notwendigen Schnittstellen in ein Autorenwerkzeug und den Wellenfeldsynthese-Renderer. Somit steht ein XML-basiertes Austauschformat für das Authoring objektbasierter Audio-Anwendungen zur Verfügung. Bei der Vertonung von Kinofilmen entstehen umfangreiche Szenendaten. Auch hier ist eine Bearbeitung durch mehrere Autoren sinnvoll. In folgenden Arbeiten sollte die Verwaltung der Szenendaten mit einem Authoring-Server realisiert werden (siehe Kapitel 5 und Abschnitt 6.4). Ein wichtiges Anwendungsgebiet der Wellenfeldsynthese ist die Beschallung von Live-Events. Die Übertragung der Szenendaten an den Renderer muss mit möglichst geringer Zeitverzögerung erfolgen, um eine zeitnahe Wiedergabe zu realisieren. Hierfür muss das Authoring-Format erweitert werden, um dynamisch generierte Daten effizient abbilden zu können.

7 Ergebnisse und Ausblick

7.1 Zusammenfassung der Ergebnisse

Die vorliegende Dissertation befasst sich mit dem Authoring objektbasierter AV-Anwendungen auf Basis des Objekt- und Szenenkonzeptes von MPEG-4. Es wurden Konzepte und Komponenten für Autorensysteme entwickelt und deren Funktionsfähigkeit mit Hilfe exemplarischer Umsetzungen nachgewiesen. Zunächst wurde diese moderne Beschreibungsform für audiovisuelle Anwendungen und ihre Möglichkeiten vorgestellt und in das Umfeld eingeordnet. Die Medienobjekte wurden nach verschiedenen Kriterien eingeteilt und einige neuartige, bisher wenig verwendete Arten vorgestellt. Anschließend wurden die Grundlagen der Szenenbeschreibung erläutert, auf denen der gesamte Authoring-Prozess beruht. Eine umfassende Beschreibung des Objekt-und Szenenkonzeptes ist der Standard MPEG-4. Der *Part 1 (Systems)* des Standards stellt die technischen Grundlagen für objektbasierte AV-Anwendungen bereit. Das Potenzial von MPEG-4 wird derzeit bei weitem nicht ausgeschöpft.

Die Verwandtschaft zu herkömmlichen Multimedia-Anwendungen veranlasste zu einer näheren Betrachtung der bei deren Erstellung anzutreffenden Abläufe und Werkzeuge. Ergänzt werden diese durch Elemente der Studioproduktion. Es wurden verschiedene kommerzielle Produkte und Forschungsprojekte untersucht. Dies sind die wichtigsten Erkenntnisse:

1. Objektbasierte AV-Anwendungen sind eine neue Klasse multimedialer Anwendungen, die Grundlage vieler neuer Anwendungen und Dienste sein wird. Die objektbasierte Beschreibung audiovisueller Anwendungen erhöht deren Komplexität im Vergleich zu anderen multimedialen Anwendungen deutlich.

2. Für die Erstellung objektbasierter AV-Anwendungen sind Kenntnisse und Fähigkeiten auf vielen verschiedenen Gebieten notwendig, z. B. aus den Bereichen Computergrafik, Multimedia-Produktion, Studioproduktion und Programmierung. Dies macht die Verteilung der Aufgaben auf mehrere Personen erforderlich.

3. Die bisher verfügbaren Werkzeuge und Autorensysteme reichen nicht aus, um die umfangreichen Möglichkeiten des Objekt- und Szenenkonzeptes von MPEG-4 adäquat in objektbasierte AV-Anwendungen umzusetzen. Die Verfügbarkeit leistungsfähiger Autorensysteme ist jedoch eine wichtige Voraussetzung für den künftigen Erfolg solcher Anwendungen.

Hieraus ergab sich die Motivation, den Produktionsprozess objektbasierter AV-Anwendungen genauer zu untersuchen und geeignete Konzepte, Werkzeuge und Systeme für deren Erstellung zu entwickeln. Bei der Produktion mehrerer interaktiver MPEG-4-Anwendungen im Rahmen des Forschungsprojektes IAVAS konnten Erfahrungen mit verschiedenen Abläufen und Werkzeugen gesammelt werden. Zusammen mit den zuvor genannten Erkenntnissen wurden folgende Anforderungen an ein Autorensystem formuliert:

1. Ein Autorensystem muss den Autoren ein intuitives Arbeiten bei der Erstellung einer objektbasierten AV-Anwendung ermöglichen. Hierfür sind angepasste Abstraktionen der zugrunde liegenden inhaltlichen und technischen Zusammenhänge auf verschiedenen Ebenen notwendig.

2. Für die Realisierung komplexer Anwendungen werden oft die Fähigkeiten mehrerer Autoren benötigt. Ein Autorensystem muss daher die Zusammenarbeit von Autoren aktiv unterstützen.

3. Für eine effektive Zusammenarbeit mehrerer Autoren und die Interoperabilität mit anderen Werkzeugen und Systemen werden geeignete Datenformate benötigt, die alle relevanten Informationen zur Beschreibung einer Anwendung beinhalten.

4. Die große Menge und Vielfalt der bei einer Produktion anfallenden Daten erfordert eine zentrale Datenhaltung auf einem Server. Es muss dem einzelnen Autor aber auch möglich sein, bei Bedarf ohne Verbindung zum Server zu arbeiten.

5. Die konkreten Anforderungen an ein Autorenwerkzeug sind von den zu erstellenden Anwendungen abhängig. Neben universell einsetzbaren Werkzeugen werden auch solche benötigt, die auf ein bestimmtes Anwendungsgebiet spezialisiert sind.

Ausgehend von diesen Anforderungen wurde ein Konzept für ein modular aufgebautes, erweiterbares Autorensystem für die Erstellung objektbasierter AV-Anwendungen auf Basis des Objekt- und Szenenkonzeptes von MPEG-4 entwickelt. Besonderes Augenmerk lag auf der Unterstützung eines auf mehrere Autoren verteilten Authoring-Prozesses und der Berücksichtigung der Erfordernisse eines vollständigen Produktionssystems. Bewährte Abläufe und Werkzeuge aus dem Bereich der Multimedia-Produktion wurden berücksichtigt bzw. integriert. Die Entwicklung erfolgte mit drei Schwerpunkten:

Authoring-Formate, Authoring-Server und Autorenwerkzeuge.

Der Aspekt des verteilten Authorings kommt in allen drei Aufgabenbereichen zum Tragen. Einer der schwerwiegendsten Nachteile vieler etablierter Autorenwerkzeuge ist die Verwendung proprietärer Datenformate während des Authoring-Prozesses. Dies schränkt die Kontrolle über die Inhalte und die Austauschbarkeit der Daten erheblich ein. Auf Grund der divergierenden Anforderungen an die Datenhaltung innerhalb der digitalen Medienkette erfolgt eine Unterteilung in Authoring-Formate und Distributions-Formate. Die Grundlage des Authorings bilden XML-basierte Authoring-Formate auf Basis von XMT. Diese nehmen alle erforderlichen Daten auf und erlauben die Beschreibung multimedialer Anwendungen auf einer der jeweiligen Aufgabe angepassten Abstraktionsebene. Ein Authoring-Format enthält Szenenbeschreibungen und Metadaten; die Medienobjekte werden hingegen nicht gespeichert, sondern als Verweise integriert. Neben universell einsetzbaren können auch sehr spezielle Formate für definierte Anwendungsgebiete formuliert werden. Dies ermöglicht die Beschreibung von Anwendungen auf einer hohen Abstraktionsebene. Die Bearbeitung kann Quelltext-basiert oder grafisch-interaktiv erfolgen. Somit lassen sich die Vorzüge beider Ansätze miteinander verbinden. Die XML-basierte Beschreibung erlaubt die Nutzung zahlreicher bewährter Werkzeuge und Verfahren aus dem Umfeld von XML. Auf dieser Basis wurde eine Architektur für die Verarbeitung von Authoring-Formaten vorgeschlagen und exemplarisch umgesetzt. In den Unterabschnitten

6.1.2, 6.2.1 und 6.5.1 werden drei spezialisierte Authoring-Formate und deren Verarbeitung vorgestellt.

Für eine effektive Zusammenarbeit mehrerer Autoren und Produzenten von Medienobjekten an einer komplexen AV-Anwendung ist eine zentrale Verwaltung der anfallenden Daten erforderlich. Im vorgeschlagenen Autorensystem übernimmt ein Authoring-Server die Speicherung, Verwaltung und Bereitstellung aller Produktions-relevanten Daten entsprechend ihrer Art und ihrer Verwendung. Es wurden verschiedene Möglichkeiten für die Verwaltung und Speicherung von Elementen einer objektbasierten AV-Anwendung untersucht und gegenübergestellt. Der Schwerpunkt lag dabei auf der effizienten und flexiblen Verwaltung der Szenenbeschreibungen. Als Grundlage für die weitere Entwicklung des Authoring-Servers wurde eine native XML-Datenbank ausgewählt. Diese nimmt die Szenenbeschreibungen auf und verwaltet diese strukturiert in Form von XML-Dokumenten. Die Steuerung der Zugriffe auf Knotenebene ermöglicht es mehreren Autoren, asynchron oder synchron gemeinsam an einem Szenengraphen zu arbeiten. Hierfür wurde ein Zugriffskonzept mit der Zuweisung von Nutzer- und Gruppenrechten auf Knoten erarbeitet. Dies eröffnet vollkommen neue Möglichkeiten der Server-basierten Zusammenarbeit mehrerer Autoren bei der Bearbeitung komplexer Aufgaben. Die Medienobjekte werden je nach Art als binäre Objekte oder strukturiert verwaltet; die Metadaten werden strukturiert als XML-Dokumente gespeichert. Der prototypisch umgesetzte Authoring-Server besteht auf einer Webserver-Umgebung und einem darin eingebetteten XML-Server. Der Zugriff erfolgt unter Nutzung verschiedener Schnittstellen über ein Netzwerk und ermöglicht die Bearbeitung von Szenendaten mit verschiedenen Werkzeugen. Mit Hilfe der entwickelten APIs (siehe Unterabschnitt 6.4.2) können Autorenwerkzeuge an den Server angebunden werden. Der Authoring-Server ist nicht nur die Schnittstelle zwischen den Autoren, sondern ermöglicht auch den Produzenten der Medienobjekte die Bereitstellung ihrer Ergebnisse und ist damit Grundlage eines vollständigen Produktions-Systems. Weiterhin ermöglicht er eine Wiederverwendung von Szenen und Szenenelementen über Produktionsgrenzen hinweg. Die strukturierte Speicherung von Szenenbeschreibungen und die Steuerung der Zugriffe auf Knotenebene kann auch Grundlage für ein künftiges Distributionssystem sein.

Die Autorenwerkzeuge sind die Schnittstellen zwischen den Autoren und den Szenendaten. Im Fokus standen grafisch-interaktive Werkzeuge für eine intuitive Bearbeitung der Szenendaten während des Authoring-Prozesses. Bewährte Ansätze und Komponenten aus dem Bereich der Multimedia-Produktion werden übernommen. Es wurde eine modulare Architektur auf Grundlage des Entwurfsmusters MVC entwickelt. Die Datenbasis sind XML-basierte Authoring-Formate, die mit Hilfe der Autorenwerkzeuge erstellt bzw. bearbeitet werden. Die eigentliche Bearbeitung erfolgt mit Hilfe von Editoren, wobei ein Autorenwerkzeug gewöhnlich mehrere Editoren (z. B. Zeitachsen-Editor, Interaktions-Editor und Metadaten-Editor) enthält. Diese stellen die Szenendaten je nach Anwendung vollständig oder teilweise auf verschiedenen Abstraktionsebenen dar. Damit werden die Szenenbeschreibungen entflochten und sind somit intuitiv zu bearbeiten. Dies ermöglicht die Realisierung sowohl von universellen als auch von spezialisierten Werkzeugen. Die Anbindung an einen Authoring-Server kann auf verschiedenen Ebenen erfolgen; ein autarker Betrieb ist ebenfalls möglich. In Abschnitt 6.3 wird die Entwicklung eines universell einsetzbaren Autorenwerkzeuges für interaktive 3D-Anwendungen beschrieben. In Unterabschnitt 6.1.4 wird ein spezialisiertes Autorenwerkzeug für die Erstellung von Vortrags-Präsentationen vorgestellt.

Die erarbeiteten Konzepte und Komponenten orientieren sich stark an den Möglichkeiten von MPEG-4, sind aber auch auf andere multimediale Anwendungen übertragbar, die auf einem Szenengraphen basieren. Es können sowohl universell einsetzbare als auch spezialisierte Autorensysteme und Werkzeuge entwickelt werden. Die in Kapitel 6 vorgestellten Anwendungen belegen die Funktionsfähigkeit der entwickelten Konzepte.

7.2 Künftige Arbeiten

Aus der vorliegenden Dissertation ergeben sich viele Ansatzpunkte und Aufgaben für künftige Arbeiten auf dem Gebiet. Neben den technischen Fragestellungen müssen die Abläufe und Tätigkeiten auch unter nichttechnischen Gesichtspunkten weiter untersucht werden. Wichtigstes Ziel sollte die Erweiterung des Autorensystems zu einem vollständigen Server-basierten Produktionssystem sein. Dies beinhaltet u. a. die Integration von Produktionswerkzeugen für Medienobjekte, von Werkzeugen zur Projektplanung- und Steuerung sowie von Hilfsmitteln für die Drehbuch-Entwicklung. In der vorliegenden Arbeit standen die Anforderungen der Autoren im Vordergrund. Bei einem weiteren Ausbau in Richtung eines vollständigen Produktionssystems sind die Bedürfnisse der anderen beteiligten Personen, z. B. der Produzenten von Medienobjekten ebenfalls zu berücksichtigen.

Nach Fertigstellung weiterer Autorenwerkzeuge und Produktionswerkzeuge ist eine weitere Evaluierung der vorgeschlagenen Arbeitsweise in einer Server-basierten Umgebung notwendig. Das Server-basierte Authoring in einer Gruppe von Autoren ist bislang im Multimedia-Bereich nicht verbreitet. Zunächst sollte der Nutzen des vorgestellten Systems und seiner Komponenten im produktiven Einsatz evaluiert werden. Dies erfordert umfangreiche Untersuchungen der Akzeptanz bei den Autoren. Wichtige Fragestellungen sind hierbei:

1. Wie teilt sich der Prozess der Erstellung auf verschiedene Bearbeiter in der Praxis auf?

2. Wie kann die Kreativität und die Produktivität des Einzelnen durch das System unterstützt werden?

3. Wann ist eine synchrone Bearbeitung einer Anwendung durch mehrere Autoren sinnvoll bzw. notwendig?

4. Wie kann die inhaltliche und technische Qualität einer Anwendung gesichert werden?

Die gewonnen Erkenntnisse fließen dann in die Gestaltung der Abläufe und Werkzeuge des Autorensystems bzw. des Produktionssystems ein.

Ein Aufgabengebiet ist die Entwicklung weiterer Authoring-Formate für konkrete Anwendungsgebiete, z. B. für 3D-Spiele, Live-Shows oder Musikanwendungen. Dabei ist eine mögliche Interoperabilität mit anderen Entwicklungen zu beachten. XML-basierte Authoring-Formate, insbesondere auf hoher Abstraktionsebene werden sehr wahrscheinlich weiter an Bedeutung gewinnen.

Aus Sicht eines Autors ist die Verfügbarkeit leistungsfähiger Autorenwerkzeuge mit intuitiv nutzbaren Editoren wichtig. Das modulare Konzept des vorgeschlagenen Autorensystems erlaubt die autarke Entwicklung neuer Werkzeuge und Editoren. Weitere

Arbeiten sind u. a. erforderlich an grafisch-interaktiven Bühnen, grafischen Interaktions-Editoren und adaptiven Zeitachsen-Editoren. Die Gestaltung der Autorenwerkzeuge beeinflusst maßgeblich die Benutzerfreundlichkeit (*Usability*) des Gesamtsystems. Neue Anforderungen an die Autorenwerkzeuge sind mit der zunehmenden Verbreitung neuartiger Medienobjekte zu erwarten, z. B. 3D-Videoobjekte oder Avatare.

Auf der Seite des Authoring-Servers ist insbesondere die Integration in vorhandene Infrastrukturen erforderlich. Um die Austauschbarkeit der zugrunde liegenden XML-Datenbank zu erleichtern, ist eine Verallgemeinerung der Datenbank-Schnittstelle vorzunehmen. Das im Unterabschnitt 5.2.4 vorgestellte Projekt *XML:DB* könnte hierfür eine Grundlage sein. Bei sehr umfangreichen Szenenbeschreibungen und deren Bearbeitung durch viele Autoren besteht die Gefahr, dass der Szenengraph zunehmend fragmentiert wird. Eine Aufgabenstellung könnte daher sein, den Szenengraphen innerhalb der Datenbank zu optimieren, z. B. durch Entfernen von mehrfachen Beschreibungen desselben Inhaltes oder durch automatisierte Zusammenfassung von Elementen. Es sind weitere Funktionen auf Grundlage der Datenbank zu implementieren, z. B. Produktions-übergreifende Suche nach Inhalten und Schutz der Urheberrechte von Medienobjekten und Szenen.

In komplexen multimedialen Anwendungen erlangen Metadaten eine immer größere Bedeutung. Ein künftiges Produktionssystem sollte daher die Erfassung und Verarbeitung von Metadaten aktiv unterstützen. Es müssen Werkzeuge für deren automatisierte Erfassung und manuelle Bearbeitung bereitgestellt werden. Die verbreitete XML-konforme Beschreibung von Metadaten (z. B. MPEG-7) ermöglicht deren effiziente Verwaltung in einer XML-Datenbank.

Schließlich muss die Produktionskette an Distributionssysteme angebunden werden, z. B. durch Integration in die Infrastrukturen des digitalen Rundfunks. Ein weiteres Einsatzgebiet für objektbasierte AV-Anwendungen sind Präsentationen im Live-Betrieb. Erste Versuche zur Anbindung eines virtuellen Studios an eine MPEG-4-Distribution verliefen am IMT bereits erfolgreich.

7.3 Schlussbetrachtung

Die Produktion objektbasierter AV-Anwendungen auf Basis des Objekt- und Szenenkonzeptes von MPEG-4 stellt neue Anforderungen an alle Komponenten der digitalen Medienkette. Eine wichtige Voraussetzung für den künftigen Erfolg solcher Anwendungen ist die Verfügbarkeit geeigneter Werkzeuge und Systeme für deren Produktion. Diese müssen den Ablauf der Erstellung einer Anwendung vollständig abdecken und sich in vorhandene Strukturen integrieren lassen. Der im Rahmen dieser Dissertation bearbeitete Teil „Authoring“ erfährt auf Grund der Beschreibungsform dramatische Veränderungen. Nie zuvor waren die notwendigen Bearbeitungsschritte so vielfältig und komplex. Damit steigt ebenfalls der Anspruch an die fachlichen Fähigkeiten der beteiligten Autoren. Die Zusammenarbeit in der Gruppe wird dabei immer wichtiger. Die in dieser Arbeit beschriebenen Ansätze, Konzepte und Komponenten sind wichtige Schritte in Richtung eines vollständigen Produktionssystems für objektbasierte AV-Anwendungen.

Abbildungsverzeichnis

Tabellenverzeichnis

Verzeichnis der verwendeten Abkürzungen

2D	zweidimensional
3D	dreidimensional
3GPP	3rd Generation Partnership Project
3DVO	dreidimensionales Videoobjekt
ACE	Access Control Element
ACL	Access Control List
API	Application Programming Interface
ASF	Apache Software Foundation
AV	audiovisuell
AVC	Advanced Video Coding
AVO	audiovisuelles Objekt
BAP	Body Animation Parameter
BIFS	Binary Format for Scenes
BLOB	Binary Large Object
BT	BIFS Text
CLOB	Character Large Object
CMS	Content Management System
CSCW	Computer-Supported Cooperative Work
CVS	Concurrent Versioning System
DAB	Digital Audio Broadcasting
DAG	Directed Acyclic Ggraph
DAI	DMIF Application Interface
DBMS	Database Management System
DDL	Description Definition Language
DMIF	Delivery Multimedia Integration Framework
DOM	Document Object Model
DSM-CC	Digital Storage Media Command and Control
DTD	Document Type Definition
DVB	Digital Video Broadcasting
DVD	Digital Versatile Disc
EAI	External Authoring Interface
EAM	Ereignis- und Aktionsmodell
EBU	European Broadcasting Union
ECMA	European Computer Manufacturers Association
ENST	école nationale supérieure des télécommunications
EPG	Electronic Program Guide
ES	Elementary Stream
FAP	Facial Animation Parameter
GPL	General Public License
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communication

GUI	Graphical User Interface
HDTV	High Definition Television
HTTP	Hypertext Transfer Protocol
IAVAS	Interaktive Audiovisuelle Anwendungssysteme
IDMT	Fraunhofer-Institut für Digitale Medientechnologie
IETF	Internet Engineering Task Force
IMT	Institut für Medientechnik
IOD	Initial Object Descriptor
IPMP	Intellectual Property Management and Protection System
IRT	Institut für Rundfunktechnik
ISDN	Integrated Services Digital Network
ISMA	Internet Streaming Media Alliance
ISO	International Organization for Standardization
ITA	Interactive Television Authoring
JDBC	Java Database Connectivity
JSP	Java Server Pages
KZK	Knotenzugriffskonzept
MHP	Multimedia Home Platform
MIDI	Musical Instrument Digital Interface
MMDBMS	Multimedia Datenbank Management System
MO	Medienobjekt
MP4	MPEG-4 Dateiformat
MPA	MPEG-4 Presentation Application
MPEG	Moving Picture Experts Group
MVC	Model-View-Controller
NXD	Native XML Database
OCI	Object Content Information
OD	Object Descriptor
ODBC	Open DataBase Connectivity
ODF	Object Description Framework
OODBMS	Objektorientiertes Datenbank Management System
ORDBMS	Objektrelationales Datenbank Management System
PES	Packetized Elementary Stream
POI	Point of Information
POS	Point of Sale
QoS	Quality of Service
RDBMS	Relationales Datenbank Management System
RPC	Remote Procedure Call
RTP	Real Time Transport Protocol
RTSP	Real Time Streaming Protocol
SAW	Spatial Audio Workstation
SDF	Scene Description Framework
SGM	Szenengraph-Manager
SMIL	Synchronized Multimedia Integration Language
SNHC	Synthetic/Natural Hybrid Coding
SOAP	Simple Object Access Protocol
SQL	Structured Query Language

SVG	Scalable Vector Graphics
TeFS	Textual Format for Scenes
TGA	Truevision Targa
TSD	Tamino Schema Definition Language
UML	Unified Modeling Language
UMTS	General Packet Radio Service
VRML	Virtual Reality Modeling Language
W3C	World Wide Web Consortium
WCMS	Web Content Management System
WebDAV	Web-based Distributed Authoring and Versioning
WFS	Wellenfeldsynthese
WMF	Windows Meta File
WSDL	Web Service Description Language
WYSIWYG	What you see is what you get
X3D	Extensible 3D
XEDB	XML Enabled Database
XML	Extensible Markup Language
XMT	Extensible MPEG-4 Textual Format
XSD	XML Schema Definition
XSL	Extensible Stylesheet Language
XSLT	Extensible Stylesheet Language Transformations
XQL	XML Query Language

Quellenverzeichnis

[3DT] 3DTV: *Integrated Three-Dimensional Television – Capture, Transmission, and Display.* `https://www.3dtv-research.org/`. – Online–Ressource, Abruf: 12.04.2005

[Bau02] Baum, Oliver: *Konzeption und Demonstration von hardwareunterstützter Virtueller Akustik innerhalb einer 3D-Visualisierung zur Integration in eine MPEG-4-Umgebung*, Technische Universität Ilmenau, Diplomarbeit, 2002

[BHJ+04] Brehm, Carsten; Hagenbring, Melanie; Jacques, Roland; Laubach, Tobias; Nguyen, Thuong H.; Ziegler, Helmut: *Entwicklung einer interaktiven Nachrichtensendung auf Basis der IAVAS-Technologie*, Technische Universität Ilmenau, Medienprojekt, 2004

[Bol98] Boles, Dietrich: *Begleitbuch zur Vorlesung Multimedia-Systeme.* Universität Oldenburg, 1998

[Bou] Bourret, Ronald: *Papers about XML.* `http://www.rpbourret.com/xml/`. – Online–Ressource, Abruf: 12.02.2005

[Bra88] Brand, Stewart: *The Media Lab: Inventing the Future at MIT.* Penguin Books, 1988. – ISBN 0–14–009701–5

[BS98] Boles, Dietrich; Schlattmann, Marco: Multimedia-Autorensysteme - Grafisch-interaktive Werkzeuge zur Erstellung multimedialer Anwendungen. In: *LOG IN (Informatische Bildung und Computer in der Schule)* 18 (1998), Nr. 1, S. 10–18

[BW04] Büchl, Bastian; Wunderlich, Claas: *Entwicklung eines Authoring-Formates zur Erstellung objektbasierter AV-Anwendungen auf Basis von XMT*, Technische Universität Ilmenau, Medienprojekt, 2004

[CD02] Concolato, C.; Dufourd, J.-C.: Comparison of MPEG-4 BIFS and some other Multimedia Description Languages. In: *Proc. Third Workshop and Exhibition on MPEG-4 (WEMP 02)*, ACM Press, 2002

[CD04] Chung, Goopeel; Dewan, Prasun: Towards dynamic collaboration architectures. In: *CSCW '04: Proc. 2004 ACM conference on Computer supported cooperative work*, ACM Press, 2004. – ISBN 1–58113–810–5, S. 1–10

[CE02] Cheok, Lai-Tee; Eleftheriadis, Alexandros: SMIL vs. MPEG-4 BIFS / Columbia University, Department of Electrical Engineering, New York. 2002. – Forschungsbericht

[Cod82] Codd, E. F.: Relational database: a practical foundation for productivity. In: *Communications of the ACM* 25 (1982), Nr. 2, S. 109–117. – ISSN 0001–0782

[Dew99] Dewan, Prasun: Architectures for Collaborative Applications. In: Beaudouin-Lafon, Michel (Hrsg.): *Computer Supported Co-operative Work* Bd. 7. John Wiley & Sons, 1999, S. 169–193

[DKR02] Drumm, Helge; Kühhirt, Uwe; Rittermann, Marco: Anwendungssysteme für MPEG-4. In: *Jahrestagung der Fernseh- und Kinotechnischen Gesellschaft*, FKTG, Juni 2002

[DKRS04] Daras, Petros; Kompatsiaris, Ioannis; Raptis, Theodoros; Strinzis, Michael G.: An MPEG-4 Tool for Composing 3D Scenes. In: *IEEE Multimedia* (2004), 04, S. 58–71

[Dob01] Dobermann, Falk: *Synchrone Telekooperation im CAD-Umfeld auf der Basis von MPEG-4*, Universität Rostock, Diplomarbeit, 2001

[DRSD03] Dantele, Andreas; Reiter, Ulrich; Schuldt, Michael; Drumm, Helge: Implementation of MPEG-4 Audio Nodes in an Interactive Virtual 3D Environment. In: *Proc. 114th Audio Engineering Society Convention*, 2003

[DS01] Dassow, Stephan; Schmitt, Ingo: MMDBMS mit MPEG4 Unterstützung. In: *Grundlagen von Datenbanken*, 2001, S. 23–27

[DS02] Dassow, Stephan; Schmitt, Ingo: Managing MPEG4 Scenes in Databases. In: *International Workshop on Multimedia Information Systems MIS 2002*, 2002, S. 96–102

[EGR91] Ellis, Clarence A.; Gibbs, Simon J.; Rein, Gail: Groupware: some issues and experiences. In: *Communications of the ACM* 34 (1991), Nr. 1, S. 39–58. – ISSN 0001–0782

[EHV93] Encarnação, José; Hübner, Wolfgang; Väänänen, Kaisa: Autorenwerkzeug für multimediale Informationssysteme. In: *Informationstechnik und Technische Informatik* 35 (1993), Nr. 2, S. 31–38

[FK99] Florescu, Daniela; Kossmann, Donald: Storing and Querying XML Data using an RDMBS. In: *Bulletin of the IEEE Computer Society Technical Commitee on Data Engineering* 22 (1999), Nr. 3, S. 27–34

[FK02] Freibichler, Hans; Kerkau, Florian: Werkzeuge zur Entwicklung von Multimedia. In: Issing, Ludwig J. (Hrsg.); Klimsa, Paul (Hrsg.): *Information und Lernen mit Multimedia und Internet*. 3. Auflage. Verlagsgruppe Beltz, 2002, S. 197–225. – ISBN 3–621–27449–9

[Fri03] Frischmuth, Sven: *Entwicklung eines Authoringkonzeptes für MPEG-4-Szenen und exemplarische Umsetzung ausgewählter Komponenten*, Technische Universität Ilmenau, Diplomarbeit, 2003

[FvFH96] Foley, James D.; van Dam, Andries; Feiner, Steven K.; Hughes, John F.: *Computer Graphics - Principles and Practice*. Second edition in C. Addison-Wesley, 1996 (The Systems Programming Series). – ISBN 0–201–12110–7

[Gat03] Gatzsche, Gabriel: *Konzeption und Umsetzung einer Encoder-Schnittstelle für MPEG-4 Szenenbeschreibungen*, Technische Universität Ilmenau, Diplomarbeit, September 2003

[GC99] Greenberg, S.; Cockburn, A.: Groupware Toolkits for Synchronous Work. In: Beaudouin-Lafon, Michel (Hrsg.): *Computer-Supported Cooperative Work*. John Wiley & Sons Ltd, 1999, S. 135–168

[Gre88] Greif, Irene: *Computer-supported cooperative work: a book of readings*. Morgan Kaufmann Publishers Inc., 1988. – ISBN 0–934613–57–5

[Gre91] Greenberg, Saul: Computer-Supported Cooperative Work and Groupware: An Introduction to the Special Issues. In: *International Journal of Man-Machine Studies* 34 (1991), Nr. 2, S. 133–141

[GWF+99] Goland, Y.; Whitehead, E.; Faizi, A.; Carter, S.; Jensen, D.: *HTTP Extensions for Distributed Authoring – WEBDAV, RFC 2518*. Version: 1999. `http://www.webdav.org/specs/`. – Online–Ressource, Abruf: 14.04.2005

[HBS03] Heyna, Arne; Briede, Marc; Schmidt, Ulrich: *Datenformate im Medienbereich*. Carl Hanser Verlag, 2003. – ISBN 3–446–22542–0

[HSS03] Herstel, Thomas; Schulz, Nadine; Schmitt, Ingo: Storing Scene Graphs in Object Relational Databases. In: *MIS 2003, 9th International Workshop on Multimedia Information Systems*, 2003, S. 123–132

[IAV] IAVAS Project: *Interactive AudioVisual Application Systems*. `http://www.iavas.de/`. – Online–Ressource, Abruf: 10.03.2005

[IBMa] IBM: *IBM MPEG-4 Technologies: Authoring*. `http://www.research.ibm.com/mpeg4/Projects/authoring.htm`. – Online–Ressource, Abruf: 12.02.2005

[IBMb] IBM: *IBM Toolkit for MPEG-4*. `http://www.alphaworks.ibm.com/tech/tk4mpeg4`. – Online–Ressource, Abruf: 12.02.2005

[ISM] ISMA – Internet Streaming Media Alliance: *ISMA Homepage*. `http://www.isma.tv/`. – Online–Ressource, Abruf: 16.02.2005

[ISO98] ISO/IEC JTC 1/SC 29/WG 11: *ISO/IEC 14496: Information technology – Coding of audio-visual objects (MPEG-4)*. 1998

[ISO01a] ISO/IEC JTC 1/SC 29/WG 11: *ISO/IEC 14496-1:2001: Information technology – Coding of audio-visual objects – Part 1: Systems*. 2001

[ISO01b] ISO/IEC JTC 1/SC 29/WG 11, Lyndon J. B. Nixon: *N7392: XSLT implementation of XMT to MPEG4 conversion*. Juli 2001

[ISO02a] ISO/IEC JTC 1/SC 22: *ISO/IEC 16262:2002: Information technology - ECMAScript language specification*. 2002

[ISO02b] ISO/IEC JTC 1/SC 29/WG 11: *N4668: MPEG-4 Overview – (V.21 - Jeju Version)*. März 2002

[ISO03a] ISO/IEC JTC 1/SC 24: *ISO/IEC 14772-1:1998: Information technology – Computer graphics and image processing – The Virtual Reality Modeling Language, - Part 1: Functional specification and UTF-8 encoding.* 2003

[ISO03b] ISO/IEC JTC 1/SC 24: *ISO/IEC 14772-2:1997: Information technology – Computer graphics and image processing – The Virtual Reality Modeling Language – Part 2: External authoring interface (EAI).* 2003

[ISO03c] ISO/IEC JTC 1/SC 29/WG 11, Smolić, A, Kimata, H.: *N5877: Applications and Requirements for 3DAV.* Juli 2003

[ISO04a] ISO/IEC JTC 1/SC 24: *ISO/IEC 19775-1:2004: Information technology – Computer graphics and image processing – Extensible 3D (X3D) – Part 1: Architecture and base components.* 2004

[ISO04b] ISO/IEC JTC 1/SC 24: *ISO/IEC 19775-1:2004: Information technology – Computer graphics and image processing – Extensible 3D (X3D) – Part 2: Scene Access Interface (SAI).* 2004

[ISO04c] ISO/IEC JTC 1/SC 29/WG 11: *ISO/IEC 14496-5: Information technology – Coding of audio-visual objects – Part 5: Reference Software.* 2004

[JK02] Joung, YeSun; Kim, Kyuheon: An XMT API for generation of the MPEG-4 scene description. In: *Proc. 10th ACM International Conference on Multimedia 2002*, ACM Press, 2002. – ISBN 1–58113–620–X, S. 307–310

[JKK02] Joung, YeSun; Kim, Hyun-Cheol; Kim, Kyuheon: XMT Tools for Interactive Broadcasting Contents Description / ETRI - Electronics and Telecommunications Research Institute, Korea. 2002. – Forschungsbericht

[KCKK01] Kim, Kyuheon; Cheong, Won-Sik; Kim, Hyun-Cheol; Kim, Jinwoong: Interactive Broadcasting Contents Authoring System / ETRI - Electronics and Telecommunications Research Institute, Korea. 2001. – Forschungsbericht

[KDRR02] Kühhirt, Uwe; Drumm, Helge; Reiter, Ulrich; Rittermann, Marco: Application Systems for MPEG-4. In: *Proc. IEEE International Symposium on Consumer Electronics ISCE'02*, IEEE CES, September 2002

[KLK02] Kim, Kyuheon; Lee, Injae; Ki, Myungseok: Interactive Contents Authoring System based on XMT and BIFS. In: *Proc. 10th ACM International Conference on Multimedia 2002* Bd. P-35, ACM Press, 12 2002. – ISBN 1–58113–620–X, S. 275–278

[Kno03] Knorr, Michael: *Entwicklung eines Authoringkonzeptes für ein MPEG-4-basiertes System zur Präsentation von Vorträgen*, Technische Universität Ilmenau, Diplomarbeit, 2003

[Krü02] Krüger, Guido: *Handbuch der Java-Programmierung.* 3. Auflage. Addison-Wesley, 2002. – ISBN 3–8273–1949–8

[KR03] Kühhirt, Uwe; Rittermann, Marco: Implementierung des Objekt- und 3D-Szenenkonzeptes von MPEG-4. In: *48. Internationales Wissenschaftliches Kolloquium*, Technische Universität Ilmenau, September 2003. – ISSN 1619–4098

[KVW03] Kallenbach, Jan; Voiß, Simone; Welter, Mareike: *Entwurf und Implementierung von Grundfunktionalitäten eines Autorensystems als Prototyp auf Basis eines vorhandenen MPEG-4 Referenzplayers*, Technische Universität Ilmenau, Medienprojekt, 2003

[KWC00] Kim, Michelle; Wood, Steve; Cheok, Lai-Tee: Extensible MPEG-4 textual format (XMT). In: *Proc. 2000 ACM workshops on Multimedia*, ACM Press, 2000. – ISBN 1–58113–311–1, S. 71–74

[Lam71] Lampson, Butler W.: Protection. In: *Proc. 5th Annual Princeton Conference on Information Sciences and Systems*, 1971, S. 437–443

[LD02a] Lukas, Uwe von; Dobermann, Falk: Accessing a Collaborative CAD System via MPEG-4. In: Haßinger, Stefan (Hrsg.); Urban, Bodo (Hrsg.): *COMPUTER GRAPHIK topics - Reports of the INI-GraphicsNet* Bd. 15. Fraunhofer-Institut für Graphische Datenverarbeitung (IGD), 2002, S. 19–20

[LD02b] Lukas, Uwe von; Dobermann, Falk: Using MPEG-4 in the engineering domain. In: Baake, U. (Hrsg.); Herbst, S.a (Hrsg.): *Concurrent Engineering: System Integration for Profit.* SCS Publication, 2002

[Lie01] Liepert, Michael: *Rechte, Benutzerrollen und Inhaltsversionierung für verteilte Multimedia-Autorensysteme*, Technische Universität Darmstadt, Dissertation, Dezember 2001

[LMBS01] Lee, Aaron; Mikael Bourges-Sevenier, Ganesh R.: Converting Macromedia Flash Shockwave to MPEG-4 BIFS. In: *Proc. of Workshop and Exhibition on MPEG-4*, 2001, S. 9–12

[Lut94] Luther, Arch C.: *Authoring Interactive Multimedia.* 1. Auflage. Publisher: Morgan Kaufmann Pub, 1994. – ISBN 0–124–60430–7

[Mit03] Mitra, Nilo: *Simple Object Access Protocoll 1.2 (SOAP 1.2).* W3C (Synchronized Multimedia Working Group) `http://www.w3.org/TR/soap/`

[MM03] Martínez, José M.; Morán, Francisco: Authoring 744: Writing Descriptions to Create Content. In: *IEEE MultiMedia* 4 (2003), S. 94–99

[Mün03] Münnich, Katrin: *Untersuchung und Implementierung von Speicher- und Übertragungsformaten für Mischwerkzeuge der Klangfeldsynthese*, Technische Universität Ilmenau, Diplomarbeit, 2003

[MPE] MPEG: *The MPEG Home Page.* `http://www.chiariglione.org/mpeg/`. – Online–Ressource, Abruf: 20.03.2005

[MRB+03] Melchior, Frank; Röder, Thomas; Brix, Sandra; Wabnik, Stefan; Riegel, Christian: Authoring System for Wave Field Synthesis Content Production. In: *Proc. 115th Audio Engineering Society Convention*, 2003

[Neu02] Neundorf, Volker: *Konzeption eines Datenbanksystems zur Organisation audiovisueller Szenen und Realisierung eines Prototypen am Beispiel „MPEG-4 VRML-Profile“*, Technische Universität Ilmenau, Diplomarbeit, 2002

[Nex] NexTV: *NexTV project public homepage.* `http://www.hitech-projects.com/euprojects/nextv/`. – Online–Ressource, Abruf: 12.03.2005

[OJSC97] Ostermann, J.; Jang, E.S.; Shin, Jae-Seob; Chen, T.: Coding of arbitrarily shaped video objects in MPEG-4. In: *Proc. of ICIP, International Conference on Image Processing*, 1997, S. 496–499

[Ope] OpenSG Forum: *OpenSG.* `http://www.opensg.org/`. – Online–Ressource, Abruf: 22.02.2005

[OSS04] Onural, L.; Sikora, T.; Smolic, A.: An Overview of a New European Consortium: Integrated Three-Dimensional Television – Capture, Transmission and Display (3DTV). In: *Proc. of European Workshop on the Integration of Knowledge, Semantics and Digital Media Technology (EWIMT)*, 2004

[PE02] Pereira, Fernando C.; Ebrahimi, Touradj: *The MPEG-4 Book.* Prentice Hall PTR, 2002. – ISBN 0–13–061621–4

[PF02] Pandzic, Igor (Hrsg.); Forchheimer, Robert (Hrsg.): *MPEG-4 Facial Animation - The standard, implementations and applications.* 1. Auflage. John Wiley & Sons, 2002

[Pre99] Preim, Bernhard: *Entwicklung interaktiver Systeme.* 1. Auflage. Springer-Verlag, 1999

[Pöt04] Pötzsch, Gregor: *Entwicklung eines Systems zur datenbankgestützten Erstellung interaktiver A/V-Anwendungen auf Basis von XMT*, Technische Universität Ilmenau, Diplomarbeit, 2004

[Ree] Reenskaug, Trygve: *The Model-View-Controller (MVC) – Its Past and Present.* `http://heim.ifi.uio.no/~trygver/themes/mvc/mvc-index.html`. – Online–Ressource, Abruf: 10.04.2005

[Rit04] Rittermann, Marco: *Zur Qualitätsbeurteilung von 3D-Videoobjekten*, Technische Universität Ilmenau, Dissertation, Oktober 2004

[RMS01] Reiter, Ulrich; Melchior, Frank; Seidel, Christoph: Automatisierte Anpassung der Akustik an virtuelle Räume. In: *Proc. 46. Internationales Wissenschaftliches Kolloquium*, Technische Universität Ilmenau, September 2001

[RS04] Rehs, Jörg; Schübel, Peter: *Entwicklung eines Authoringtools zur Erstellung von MPEG-4 basierten Präsentationen*, Technische Universität Ilmenau, Medienprojekt, 2004

[Sam] Sambits Project: *System for Advanced Multimedia Broadcast and IT Services.* `http://www.irt.de/sambits/`. – Online–Ressource, Abruf: 12.02.2005

[Sam00a] Sambits Project: SAMBITS – Deliverable 1: Project Description and Plan / Sambits Project. 2000. – Forschungsbericht

[Sam00b] Sambits Project: SAMBITS – Deliverable 3: Demonstrator service scenario and technical specifications of multimedia studio and terminal systems / Sambits Project. 2000. – Forschungsbericht

[Sam00c] Sambits Projekt: SAMBITS – Deliverable 2: Advanced Services Market Survey/ Deployment Strategies and Requirement/Specification of Integrated Broadcast and Internet Multimedia Services / Sambits Projekt. 2000. – Forschungsbericht

[Sch02] Schuldt, Michael: *Implementierung von shaped Video in 3D-Szenen*, Technische Universität Ilmenau, Diplomarbeit, 2002

[SGIa] SGI: *Open Graphics Library (OpenGL) 1.5*. `http://www.opengl.org/`. – Online–Ressource, Abruf: 01.08.2004

[SGIb] SGI: *Open Inventor*. `http://oss.sgi.com/projects/inventor/`. – Online–Ressource, Abruf: 22.02.2005

[Shn02] Shneiderman, Ben: *User Interface Design*. 3. Auflage. mitp-Verlag, 2002. – ISBN 3–8266–0753–8

[Sof03] Software AG: *Tamino XML Server Documentation*. Software AG `http://www.softwareag.com/`

[Son] Song: *SoNG (portalS Of Next Generation)*. `http://www.octaga.com/SoNG-Web/`. – Online–Ressource, Abruf: 12.03.2005

[SSI02] Stammnitz, Peter; Stoll, Gerhard; Illgner, Klaus: SAMBITS: Studio- und Terminaltechnik für interaktives Multimedia-Broadcast. In: *Fernseh- und Kino-Technik* 8-9 (2002), S. 446–475

[Ste00] Steinmetz, Ralf: *Multimedia-Technologie: Grundlagen, Komponenten und Systeme*. 3. Auflage. Springer Verlag, 2000. – ISBN 3–540–67332–6

[STZ+99] Shanmugasundaram, Jayavel; Tufte, Kristin; Zhang, Chun; He, Gang; DeWitt, David J.; Naughton, Jeffrey F.: Relational Databases for Querying XML Documents: Limitations and Opportunities. In: *The VLDB Journal*, 302-314

[SUN] SUN Microsystems: *Java 3D API*. `http://java.sun.com/products/java-media/3D/`. – Online–Ressource, Abruf: 01.09.2004

[SVH99] Scheirer, Eric; Väänänen, Riitta; Huopaniemi, Jyri: AudioBIFS: Describing Audio Scenes with the MPEG-4 Multimedia Standard. In: *IEEE Transactions on Multimedia* 1 (Spetember 1999), Nr. 3, S. 237–250

[SW] Stein, Greg; Whitehead, Jim: *Distributed Authoring Protocol (WebDAV)*. `http://www.webdav.org/`. – Online–Ressource

[W3Ca] W3C: *Document Object Model (DOM)*. `http://www.w3.org/DOM/`. – Online–Ressource, Abruf: 10.04.2005

[W3Cb] W3C: *Scalable Vector Graphics (SVG).* http://www.w3.org/Graphics/SVG/. – Online–Ressource, Abruf: 20.02.2005

[W3Cc] W3C: *Scalable Vector Graphics (SVG) 1.1 Specification.* http://www.w3c.org/TR/SVG11/. – Online–Ressource, Abruf: 20.02.2005

[W3Cd] W3C: *Synchronized Multimedia.* http://www.w3.org/AudioVideo/. – Online–Ressource, Abruf: 20.02.2005

[W3Ce] W3C: *Synchronized Multimedia Integration Language (SMIL 2.0) - [Second Edition].* http://www.w3.org/TR/SMIL/. – Online–Ressource, Abruf: 20.02.2005

[Wal02] Walsh, Aaron E.: Understanding Scene Graphs – Using graph-based data structures to organize and manage scene contents. In: *Dr. Dobbs Journal* July (2002)

[WBS01] Walsh, Aaron E.; Bourges-Sevenier, Mikael: *Core Web3D.* Prentice Hall PTR, 2001. – ISBN 0–13–085728–9

[WBS02] Walsh, Aaron E.; Bourges-Sevenier, Mikael: *MPEG-4 Jump-Start.* Prentice Hall PTR, 2002. – ISBN 0–13–060036–9

[Weba] Web3D Consortium: *Open Standards for Real-Time 3D Communication.* http://www.web3d.org/. – Online–Ressource, Abruf: 20.02.2005

[Webb] Web3D Consortium: *XQuery 1.0: An XML Query Language.* http://www.w3.org/TR/xquery/. – Online–Ressource, Abruf: 20.02.2005

[Wei03] Weigel, Christian: *Entwicklung eines MPEG-4-basierten Systems zur Präsentation von Vorträgen*, Technische Universität Ilmenau, Diplomarbeit, 2003

[WK03] Weigel, Christian; Knorr, Michael: XMT Based Authoring System for MPEG-4 Slide Shows. In: *Proc. IEEE International Symposium on Consumer Electronics ISCE'03*, IEEE CES, Dezember 2003

[XMLa] XML:DB: *Application Programming Interface for XML Databases.* http://xmldb-org.sourceforge.net/xapi/. – Online–Ressource, Abruf: 16.02.2005

[XMLb] XML:DB: *SiXDML – Simple XML Data Manipulation Language.* http://xmldb-org.sourceforge.net/sixdml/. – Online–Ressource, Abruf: 16.02.2005

[XMLc] XML:DB: *XML:DB Initiative for XML Databases.* http://xmldb-org.sourceforge.net/. – Online–Ressource, Abruf: 16.02.2005

[XMLd] XML:DB: *XUpdate – XML Update Language.* http://xmldb-org.sourceforge.net/xupdate/. – Online–Ressource, Abruf: 16.02.2005

[Yas00] Yass, Mohammed: *Entwicklung multimedialer Anwendungen.* 1. Auflage. dpunkt Verlag, 2000

[Zie04] Ziegler, Helmut: *Konzeption und Implementierung eines Autorenwerkzeuges zur Erstellung interaktiver A/V-Anwendungen*, Technische Universität Ilmenau, Diplomarbeit, 2004