MMDBMS mit MPEG4 Unterstützung

Stephan Dassow, Ingo Schmitt
Department of Computer Science, University of Magdeburg
P.O. Box 4120, D-39016 Magdeburg, Germany
[sdassow,schmitt]@iti.cs.uni-magdeburg.de

Zusammenfassung

Viele Anwendungen benötigen eine effiziente Verwaltung von Multimedia-Daten. Eine solche Verwaltung kann prinzipiell durch Datenbanksysteme abgedeckt werden, aber leider bieten die heutigen Systeme meist nur Speziallösungen oder einzelne voneinander isolierte Lösungen an. Dieses Papier stellt ein MMDBMS mit MPEG4 Unterstützung vor, welches viele multimediale Datentypen unterstützt und des weiteren Operationen auf diesen ermöglicht.

1 Motivation

Die zunehmende Möglichkeit analoge Informationen zu digitalisieren und mit anderen Informationen zu Multimedia-Dokumenten zu verknüpfen, führt zur einer stetig wachsenden Menge von Multimedia-Daten. Die Verwaltung dieser Daten ist die zentrale Aufgabe eines Multimedia Datenbank Management Systems (MMDBMS). Derzeitige MMDBMS sind meist spezielle DBMS die nur bestimmte Multimediadatentypen (z.B. Bilder bei QBIC von IBM) behandeln, oder die einzelnen Datentypen isoliert voneinander betrachten. So ist in derzeitigen MMDBMS die Verknüpfung von Bild und Video (z.B. Firmenlogo-Einblendungen in Videos) meist nicht möglich. In diesem Papier wollen wir ein MMDBMS mit MPEG4 Unterstützung vorstellen. Dieses System wird insbesondere neue Datenbankoperationen auf Multimediadatentypen ermöglichen. Aber auch die anderen Funktionen eines DBMS werden unterstützt. Das folgende Kapitel wird zur Einführung den MPEG4 Standard erklären. Das Kapitel 3 widmet sich dann einem konkreten Beispiel aus dem Bereich "News and Informations Services". In Kapitel 4 wird dann unser Entwurf für ein MMDBMS mit MPEG4-Unterstützung vorgestellt. Anschließend erfolgt ein Ausblick auf weitere Arbeiten.

2 MPEG4

MPEG4 ist ein ISO/IEC Standard für interaktive Multimedia-Präsentationen. Eine genaue Beschreibung des Standard kann in [MPE98] gefunden werden. Wir werden hier nur kurz eine Einführung in die wichtigsten Begriffe und Verfahrensweisen geben.

Der Standard diskutiert im allgemeinen die kodierte Darstellung von digitalisierten und synthetischen Objekten im Bereich Video und Audio. Diese Objekte werden als AV-Objekte bezeichnet. AV-Objekte sind hierarchisch aufgebaut und können aus einzelnen anderen AV-Objekten gebildet werden. AV-Objekte die sich nicht mehr unterteilen lassen werden als primitive AV-Objekte bezeichnet. Diese Objekte müssen für die Übertragung geeignet zerlegt und kodiert werden. Abbildung 1 zeigt eine typische Übertragungskette und deren Komponenten in MPEG4. Die erste Ebene, Composition and Rendering, stellt dem Anwender alle relevanten Daten einer Szene dar. Im Normalfall wird diese Aufgabe durch einen MPEG4-Renderer übernommen. Die darunter liegende Ebene bereitet die AV-Objekte auf. Dabei werden alle relevanten Informationen für die AV-Objekte entschlüsselt. Diese sind Szenen-Graphen genannt BIFS (BInary Format for Scenes), Objektdeskriptoren und audiovisuelle Objektdaten. Die Objektdeskriptoren beinhalten unter anderem Identifikation-, Assoziation-Informationen, Methoden der AV-Objekte (Kodierung-, Render-Algorithmen) und weitere Attribute wie z.B. Qualtity of Services (QoS). Die audiovisuellen Objektdaten sind ent- oder verschlüsselte Streams, die nur Rohdaten (Bilder, Audio) enthalten. Die Szenen-Graphen enthalten eine Vielzahl von Informationen, die zum Rendern der Szene benötigt werden. Sie werden im Anschluß ausführlicher behandelt. Die Upstream Informationen sind für unsere Betrachtungen nicht von Bedeutung. Die dritte Ebene befaßt sich mit der Synchronisation und dem Buffer-Management. Dagegen ist die vierte Ebene ausschließlich für das Übermitteln der Daten zuständig. Eine genaue Beschreibung der einzelnen Ebenen kann in [Sig99] gefunden werden. Wenden wir uns nun den Szenen-Graphen genau zu.

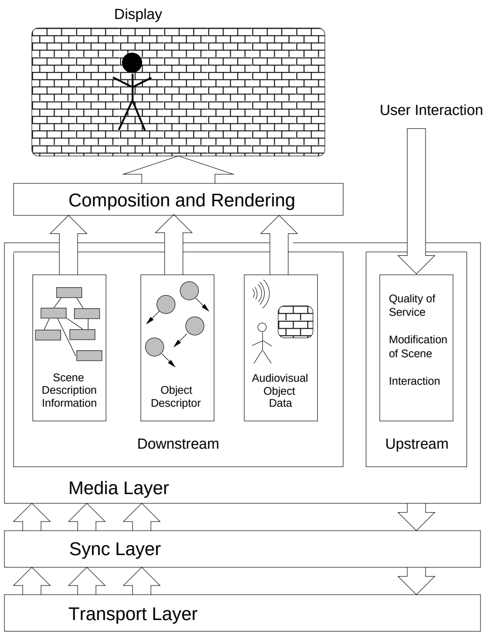

Abbildung 1: MPEG4 Übertragungskette

2.1 BIFS

Eine Szene in MPEG4 besitzt ein hierarchische Struktur, welche als gerichteter, azyklischer Graph interpretiert werden kann. Jeder Knoten des Graphen stellt ein AV-Objekt dar. Die Endknoten (nur ausgehende Kanten) korrespondieren zu speziellen Audio- oder Video-Objekten (primitive AV-Objekte), welche meist direkt aus den audiovisuellen Objektdaten und Objektdeskriptoren oder synthetischen Objekten gebildet werden können. Die anderen Knoten sind Gruppierungs-, Transformations- oder andere Operationsknoten. Folgende Informationen der AV-Objekte können in den BIFS gespeichert sein:

- Positionierung der AV-Objekte in Zeit und Raum,
- Informationen über die Darstellung eines Objektes in der Szene,
- Informationen zur Bildung von zusammengesetzten AV-Objekten und
- weitere Informationen über einzelne synthetische Objekte oder andere MPEG4 typische Informationen.

```
Layer2D { ...
  Transform2D{ ...
    Shape{
      appearance Appearance{
        material Material {     }
        texture ImageTexture { url ``sonystick.jpg'' }
      }
      geometry Box { size 1 2 3 }
    }
  }
}
```

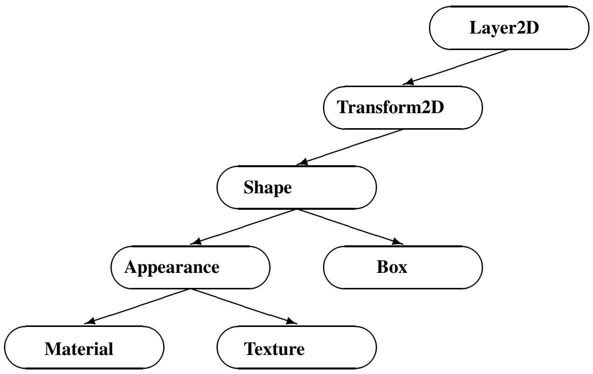

Abbildung 2: BIFS Beispiel und Szenen-Graph

Die Abbildung 2 zeigt einen MPEG-Audio-Player als Sourcecode und als Szenen-Graphen. Die primitiven AV-Objekte sind `Box`, `Texture` und `Material`. Zur Positionierung in Zeit und Raum sind die Knoten `Transform2D` und `Layer2D` notwendig. Weiterführende und detaillierte Ausführungen über BIFS können in [Sig99] nachgelesen werden.

3 Beispiel

In diesem Abschnitt werden wir uns mit einem Beispiel aus dem Bereich “News and Informations Services” beschäftigen. Folgendes Szenario sei gegeben:

Die Firma News & Co vertreibt Nachrichten und allgemeine Informationen übers Internet. Der registrierte Anwender beschreibt als erstes seine Interessen z.B. Sport, genauer Tour de France. Da es sich bei dem Anwender um einen Italiener handelt, bevorzugt dieser Informationen über Marco Pantani. Danach gibt er einen Zeitraum an, z.B. das Jahr 2000. Das Nachrichten System wählt nun alle relevanten Nachrichten und Informationen der Tour de France 2000 aus und sortiert diese nach Nachrichten mit Marco Pantani und ohne ihn. Zusätzlich werden die Daten zeitlich geordnet und geeignet für den Anwender aufgearbeitet, d.h. zusammengehörige Informationen wie Radiosendungen können durch Bilder erweitert werden. Die so ermittelten Multimedia-Daten werden dann als interaktive Nachrichtensendung (Anwender kann bestimmte Abschnitte überspringen oder die Qualität der Daten verändern) mit animiertem Nachrichtensprecher aufbereitet und dem Anwender zur Verfügung gestellt.

Dieses soll als kurze Beschreibung genügen. Wenden wir uns nun den Daten zu, die von einer derartigen Anwendung benötigt werden. Die Tabelle 1 zeigt eine mögliche Auswahl von Daten und klassifiziert diese in statische und dynamische Daten. Des weiteren nimmt die Menge der Informationen die in den Daten ausgedrückt wird nach unten zu.

Beispiel	Dokumenttyp	stat./dyn.	AV-Objekte
Pressemitteilungen	textuelle Information	stat	text
Pressebilder	Graphik	stat	image
virtuelle Modelle	Virtual Reality	stat	synthetisches Model, images
Animierter News-Sprecher	Animation	dyn	Animationsdaten
Radiomitteilungen	Audio	dyn	audio
Nachrichtenfilme	Video	dyn	video
kpl. News-Sendung	interaktives Dokument	dyn	alle

Tabelle 1: Datenanalyse

Es ist zu beachten, daß eine Nachrichtensendung im Gegensatz zu Nachrichtenfilmen (primitives AV-Objekt) aus verschiedenen AV-Objekten zusammengesetzt wird und Interaktion ermöglicht. Es ist des weiteren zu beachten, das sich die Daten laufend ändern. So muß zum Beispiel der Inhalt einer aktuellen Nachrichtensendung geändert werden, sobald ein neues, den Anwender interessierendes Ereignis, eintritt. Dabei muß die innere Struktur der Nachrichtensendung geändert werden, da die Nachricht sowohl zeitlich als auch thematisch in die bestehende Nachrichtensendung eingepaßt werden muß. Aber auch inhaltliche Änderungen wie zum Beispiel die Änderung des Führenden der Gesamtwertung der Tour de France, muß in allen relevanten Dokumenten geändert werden. Hierbei wird die Struktur aber nicht geändert. Im folgenden Abschnitt diskutieren wir einige Anwendungsfälle und beschreiben, wie diese durch die Basis-Funktionalität eines DBMS unterstützt werden können.

3.1 Datenbank Funktionalität

Der vorherige Abschnitt beschrieb eine multimediale Anwendung, die Nachrichten aller Art umfaßt. In diesem Abschnitt soll nun die Verwendung eines MMDBMS mit MPEG4-Unterstützung für eine derartige Anwendung motiviert werden.

Ein traditionelles Datenbanksystem unterstützt im Normalfall alle Funktionen nach Codd [Cod82] (Integration, Benutzersichten, Katalog, Operationen, Konsistenzüberwachung, Datensicherung, Datenschutz, Transaktionen, Synchronisation). Diese sollten möglichst auch für ein MMDBMS gelten. In diesem Abschnitt werden wir einige dieser Funktionalitäten in Zusammenhang mit MMDBMS und unserer Anwendung diskutieren.

Wenden wir uns als erstes der *Datenintegration* zu. In traditionellen Datenbanksystemen sollen möglichst alle zur Applikation gehörenden Daten zentral gespeichert sein und über eine einheitliche Schnittstelle zugänglich gemacht werden. Dieses sollte natürlich auch für unser System gelten, da hierdurch Redundanz vermieden wird. Des weiteren wird dem Anwender eine einheitliche Schnittstelle zum System gegeben. Hier sei kurz angemerkt, daß das MMDBMS alle multimedialen Daten im MPEG4-Format speichern könnte, oder alle multimedialen Daten über einen MPEG4-Schnittstelle ausgeben könnte.

Als zweites diskutieren wir die *Operationen* eines MMDBMS. In traditionellen RDBMS gibt es meisten Operation wie Insert, Delete, Update und weitere um den aktuellen Datenbestand zu ändern, sowie komplexe Anfrageoperationen. Genau hierbei sind normale MMDBMS meist überfordert. Sie stellen zwar noch die Grundoperationen zum Ändern des Datenbestands zur Verfügung, aber für komplexe Datenbankanfragen auf multimedialen Daten gibt es meist keine Unterstützung z.B. einfaches Verbinden von Audio- und Video-Daten (nur visuelle Information im Video). Mittels eines MMDBMS mit MPEG4-Unterstützung wäre es möglich auch diesen Operationen zu unterstützen. Das Zusammensetzen von AV-Objekten stellt in MPEG4 kein Problem dar, da diese einfach gruppiert werden können. Des weiteren ist eine inhaltsbasierte Suche über den AV-Objekten möglich, was auch eine Erweiterung zu traditionellen DBMS darstellt.

Ein weiterer wichtiger Punkt in traditionellen Datenbanken sind die *Benutzersichten*. Hiermit wird es dem Anwender ermöglicht, bestimmte Daten in seiner Sichtweise zusammenzustellen. Dieses ist natürlich auch in unserer Anwendung von zentraler Bedeutung. So sollen Nachrichten z.B. nach Thematik (Sport, Kurioses) betrachtet werden. Des weiteren ist es möglich eine QoS bezogene Sicht zu erzeugen. Diese QoS bezogene Sicht zeigt dem Anwender alle für Ihn darstellbaren Nachrichten.

Auch weitere Punkte wie z.B. *Datenschutz* (Watermarking Algorithmen usw), *Datensicherung* und *Konsistenzüberwachung* werden von einem MMDBMS auf Basis von MPEG4 unterstützt, werden hier aber aus Platzgründen nicht weiter diskutiert. Wie aber an diesen Beispielen zu sehen ist, kann eine solche Anwendung sehr gut durch ein MMDBMS mit MPEG4 unterstützt werden.

4 MMDBMS mit MPEG4

In diesen Abschnitt wollen wir uns als erstes der Architektur eines MMDBMS mit MPEG4-Unterstützung widmen. Danach wenden wir uns der Speicherung der einzelnen Komponenten des Media Layers zu. Der Aufbau unseres

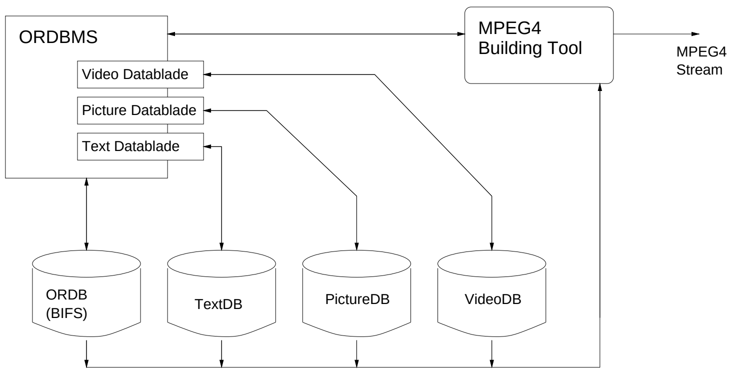

Abbildung 3: MMDBMS Architektur

MMDBMS orientiert sich stark an der Universal Server Architektur, wie in Abbildung 3 zu sehen ist. Das MPEG4-Building-Tool generiert hierbei einen MPEG4-Stream aus den vom System bereitgestellten Daten. Wie in [BDB00] beschrieben, benötigt ein solches System die Upstream-Daten (Daten die von der Anwendung zum Anwender gehen) des Media Layers. Im MMDBMS werden also nicht die kompletten MPEG4-Datenstreams gespeichert sondern nur deren Komponenten. Es ist aber zu beachten, daß das MPEG4-Building-Tool auch Bestandteil des MMDBMS sein kann.

Wenden wir uns nun der Speicherung der Komponenten des Media Layers zu. In MPEG4 wird zwischen flexible und non-flexible Terminals unterschieden. Damit werden im allgemeinem interaktive Dokumente mit standardisierten (non-flexible) und nicht-standardisierten (flexible) Datentypen bezeichnet. Bei non-flexible Terminals wird davon ausgegangen, daß die Methoden zum Rendern und Kodieren der einzelnen Streams auf dem Client vorhanden sind. Genaueres kann in [MPE98] nachgelesen werden. Wir wollen uns im folgenden auf non-flexible Terminals beschränken.

Die Beschränkung auf non-flexible Terminals hat zur Folge, das alle Objektdeskriptoren zum größten Teil (Methoden, Typdefinitionen) standardisiert sind. Die fehlenden, nicht standardisierten Attribute (Assoziations- und Identifikationsinformationen) werden erst bei der Generierung berechnet. Dieses wiederum hat zur Folge, daß ein Abspeichern der Objektdeskriptoren in der Datenbank nicht erforderlich ist.

Die Speicherung der audiovisuellen Objektdaten soll in klassischen Erweiterungen der Datenbank erfolgen, da es sich hier "nur um einfache" Datentypen handelt. Vorteil dieser Vorgehensweise ist die Nutzung der einzelnen

Feature jeder Datenbank Erweiterung. So können zum Beispiel zur Suche ähnlicher einzelner Objekte die Indexe sowie Featurewerte der einzelnen Erweiterungen benutzt werden.

Im Gegensatz zu den eben behandelten Daten sind die BIFS recht stark strukturiert. Ihre Form erinnert in unkomprimierter Form stark an XML Dateien oder allgemeiner an semi-strukturierte Daten [Abi97]. Des weiteren können BIFS durch eine DTD beschrieben werden. Hieraus wiederum folgt, daß eine Speicherung in Relationalen DBMS oder Objekt Relationalen DBMS möglich ist. In [FK99], [STH+99] und [DFS99] werden einige Ansätze zur Speicherung solcher Daten diskutiert. Es sei hier aber bemerkt, daß diese Speicherung nicht frei von Problemen ist. Deshalb sollten hier auch Ansätze aus dem Bereich objektorientierter Datenbanken und Graphendatenbanken untersucht werden. Wichtig hierbei ist, daß die Datenbank Operationen wie das Generieren von neuen BIFS Knoten unterstützt, um die Gruppierung von AV-Objekten zu unterstützen. Vorteile diese Architektur sind in folgender Übersicht zusammengefaßt:

- inhaltsbasierte Suche in MPEG4 Daten
- Suche nach ähnlichen Dokumenten durch Vergleich der verwendeten primitiven AV-Objekte und deren Anordnung (siehe [FGT97])
- Bereitstellung neuer Operationen zur Gruppierung
- Unterstützung von zeitbezogenen Integritätsbedingungen

5 Ausblick

In folgenden Arbeiten sollen insbesondere die Aspekte der Speicherung von BIFS genauer betrachtet werden. Dabei werden die unterschiedlichen Speichermöglichkeiten diskutiert und eventuell prototypisch auf Basis von VRML Daten umgesetzt. Des weiteren ist der kurz beschriebene Aspekt, daß alle Multimedia-Daten intern als MPEG4 Daten gespeichert werden, genauer zu untersuchen. Auch eine genauere Untersuchung von zeitlichen und räumlichen Integritätsbedingungen im Zusammenhang mit MPEG4 ist angedacht.

Literatur

[Abi97] Serge Abiteboul. Querying semi-structured data. In Foto N. Afrati and Phokion Kolaitis, editors, *Database Theory - ICDT '97, 6th International Conference, Delphi, Greece, January 8-10, 1997, Proceedings*, volume 1186 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 1997.

[BDB00] Souhila Boughoufalah, Jean-Claude Duford, and Frederic Bouilhaguet. MPEG-Pro, an Authoring System for MPEG-4 with Temporal Constraints and Template Guided Editing. *IEEEE*, 2000.

[Cod82] E. F. Codd. Relational Database: A Practical Foundation for Productivity. *Communications of the ACM*, 25(2):109–117, February 1982.

[DFS99] Deutsch, M. Fernandez, and D. Sucio. Storing Semi-strucured Data with STORED. *Proceedings of the ACM SIGMOD Conference*, Mai 1999.

[FGT97] A. Müifit Ferman, Bilge Günsel, and A. Murat Tekalp. Object-Based Indexing of MPEG-4 Compressed Video. *Proceedings VCIP 97*, pages 953–963, Feb 1997.

[FK99] Daniela Florescu and Donald Kossmann. A Performance Evaluation of Alternative Mapping Schemes for Storing XML Data in Relational Databases. *INRIA*, Rapport de Recherche No. 3680, 1999.

[MPE98] MPEG-4 ISO/IEC 14496. *Standard, http://www.cselt.it/mpeg*, 1998.

[Sig99] Julien Signés. Binary Format for Scene (BIFS): Combining MPEG4 media to build rich multimedia services. *Proceedings of SPIE*, 3653, 1999.

[STH+99] J. Shanmugasundraram, Kristin Tufte, Gang He, Chung Zhang, David DeWitt, and Jeffrey Naughton. Relational Databases for Quering XML Documents: Limitations and Opportunities. *Proceedings of the 25th VLDB Conference*, 1999.