Formate, Technologien und Architekturkonzepte für 3D-Web-Applikationen

Belegarbeit an der
Technischen Universität Dresden
August 2001

Enrico Rukzio

Betreuer: Dipl.-Inform. Raimund Dachselt
Hochschullehrer: Prof. Dr. Klaus Meißner
Fachgebiet Multimediatechnik
Institut für Software- und Multimediatechnik
Fakultät Informatik

Erklärung

Hiermit erkläre ich, Enrico Rukzio, die vorliegende Belegarbeit zum Thema

Formate, Technologien und Architekturkonzepte für 3D-Web-Applikationen

selbständig und ausschließlich unter Verwendung der im Quellenverzeichnis aufgeführten Literatur- und sonstigen Informationsquellen verfasst zu haben.

Dresden, den 31. August 2001

Inhaltsverzeichnis

Inhaltsverzeichnis I
Abbildungsverzeichnis III
Tabellenverzeichnis IV
Abkürzungsverzeichnis IV
Formatverzeichnis V
1 Einleitung 1
2 Grundlagen von Web3D-Grafik 2
2.1 Die Bestandteile einer Web3D-Applikation 2
2.1.1 Grundlegende Client/Server-Architektur 4
2.1.2 Interaktionsmöglichkeiten mit einer Web3D-Applikation 5
2.1.3 Realisierung von Verhaltensweisen 5
2.1.4 Szenenbeschreibung 6
2.2 Anwendungsklassen von Web3D-Applikationen 7
2.3 Notwendigkeit eigener 3D-Formate für das WWW 12
2.4 Hauptprobleme von Web3D-Grafik 13
2.5 Lösungsansätze für bestehende Probleme bei Web3D-Technologien 15
2.5.1 Modulare Ansätze 15
2.5.2 Kompression 16
2.5.2.1 Geometriekompression 16
2.5.2.2 Texturkompression 18
2.5.3 Streaming 18
2.5.3.1 Nachladen von Szenenbestandteilen 18
2.5.3.2 Zeitabhängige Updates der Web3D-Szene 18
2.5.3.3 Progressive Compressed Stream 19
2.5.3.4 Audio- und Video-Streaming 20
2.5.4 Skalierung 20
3 Web3D-Formate 21
3.1 Vergleichskriterien 21
3.1.1 Erstellung 21
3.1.2 Repräsentation 22
3.1.3 Kommunikation 22
3.1.4 Präsentation 23
3.1.5 Bewertung 23
3.2 Standardisierte Formate bzw. Vorschläge für Standards 24
3.2.1 VRML97 25
3.2.1.1 Erstellung 25
3.2.1.2 Repräsentation 27
3.2.1.3 Kommunikation 27
3.2.1.4 Präsentation 27
3.2.1.5 Bewertung 28
3.2.1.6 Rich Media Working Group 29
3.2.1.7 Universal Media Working Group 29
3.2.2 X3D 30
3.2.2.1 Erstellung 30
3.2.2.2 Repräsentation 33
3.2.2.3 Kommunikation 33
3.2.2.4 Präsentation 33

3.2.2.5 Bewertung ... 33
3.2.3 Shout3D und blaxxun3D ... 35
3.2.3.1 Erstellung ... 35
3.2.3.2 Repräsentation ... 38
3.2.3.3 Kommunikation ... 38
3.2.3.4 Präsentation ... 38
3.2.3.5 Bewertung ... 39
3.2.4 MPEG-4/BIFS ... 40
3.2.4.1 Erstellung ... 40
3.2.4.2 Repräsentation ... 42
3.2.4.3 Kommunikation ... 43
3.2.4.4 Präsentation ... 45
3.2.4.5 Bewertung ... 46
3.3 Nichtstandardisierte, proprietäre Formate ... 48
3.3.1 Shockwave3D ... 49
3.3.1.1 Erstellung ... 49
3.3.1.2 Repräsentation ... 51
3.3.1.3 Kommunikation ... 52
3.3.1.4 Präsentation ... 53
3.3.1.5 Bewertung ... 54
3.3.2 Viewpoint ... 55
3.3.2.1 Erstellung ... 55
3.3.2.2 Repräsentation ... 60
3.3.2.3 Kommunikation ... 60
3.3.2.4 Präsentation ... 61
3.3.2.5 Bewertung ... 61
3.3.2.6 Adobe Atmosphere ... 62
3.3.3 Cult3D ... 63
3.3.3.1 Erstellung ... 63
3.3.3.2 Repräsentation ... 66
3.3.3.3 Kommunikation ... 66
3.3.3.4 Präsentation ... 67
3.3.3.5 Bewertung ... 67
3.3.4 Pulse ... 69
3.3.4.1 Erstellung ... 69
3.3.4.2 Repräsentation ... 71
3.3.4.3 Kommunikation ... 71
3.3.4.4 Präsentation ... 71
3.3.4.5 Bewertung ... 72
3.3.5 Weitere Formate ... 73
3.4 Übersicht ... 75
3.5 Kompatibilitätsdiagramm ... 76
3.6 Zusammenfassung ... 78
4 Gedanken zum CONTIGRA-Projekt ... 80
5 Vertiefende praktische Aufgabe ... 81
5.1 Implementation ... 82
5.2 Vergleich der Implementationen in VRML97 und Shockwave3D ... 87
Glossar ... 89
Literaturverzeichnis ... 91
WWW-Adressen - Verzeichnis ... 93

Abbildungsverzeichnis

Abbildung 1: Grundlegende Client/Server-Architektur 4
Abbildung 2: Dreidimensionale Präsentation einer Uhr [Breitling] 7
Abbildung 3: Wohngebiet [iTEConsult] 8
Abbildung 4: Shout3D Tank Game [TankGame] 8
Abbildung 5: CART Auto Racing [Blendo] 9
Abbildung 6: 3D-Fußballsimulation [TOPlay] 9
Abbildung 7: Jay Leno - Sketch [JayLeno] 9
Abbildung 8: International Space Station [ISS] 10
Abbildung 9: 3D-Sitemap des Lehrstuhles MMT an der TU-Dresden [MMT] 10
Abbildung 10: Pariser Platz [e:Berlin] 10
Abbildung 11: Netzvereinfachung [Ross98] 16
Abbildung 12: Generalisierte Dreiecksstreifen 17
Abbildung 13: *Progressive Compressed Stream* [Coh99] 19
Abbildung 14: Erweiterung des *Core-Profile* 31
Abbildung 15: Shout3D Wizard 35
Abbildung 16: blaxxun3D Wizard 36
Abbildung 17: MPEG-4 ToolBox 42
Abbildung 18: MPEG-4 Architektur 43
Abbildung 19: Relation zwischen Szenengraph und den *Streams* 44
Abbildung 20: Streaming von Szenenbeschreibung und Objektdeskriptoren 45
Abbildung 21: Anzeige des Shockwave Players beim Ladevorgang einer 3D-Applikation ... 49
Abbildung 22: Vordefinierte Aktionen 50
Abbildung 23: Vordefinierte Ereignisse 51
Abbildung 24: Durch die MRM-Technologie übertragenes Dreiecksnetz mit 229, 1000 und 8772 Eckpunkten. 52
Abbildung 25: Durch *Subdivision Surfaces* geglättete Oberflächen [Jun00] 53
Abbildung 26: Beispiel für eine Unterblock-Struktur in einer MTS-Datei 56
Abbildung 27: Interne Struktur einer MTX-Datei 56
Abbildung 28: Die Viewpoint-Werkzeuglandschaft 57
Abbildung 29: Viewpoint Scene Builder 59
Abbildung 30: Viewpoint Stream Tuning Studio 60
Abbildung 31: *Variable Tools* 62
Abbildung 32: Die Cult3D-Werkzeuglandschaft 63
Abbildung 33: Cult3D Designer 64
Abbildung 34: Visuelle Verhaltensmodellierung 66
Abbildung 35: Pulse Creator 69
Abbildung 36: Kompatibilitätsdiagramm 76
Abbildung 37: Realisierung der Visualisierungstechnik CCT mittels Shockwave3D (Version 3a) 83
Abbildung 38: Navigation zur Stufe *l=4* (Version 3a) 85
Abbildung 39: Text mit den Werten 1 und 5 (hohe Polygonanzahl) für den Parameter `smoothness` 86

Tabellenverzeichnis

Tabelle 1: Anwendungsklassen und deren Anforderungen an die verwendete Web3D-Technologie 11
Tabelle 2: Anforderungscharakteristik für 3D-Formate 12
Tabelle 3: Beispiele für Knotentypen 25
Tabelle 4: Blockarten in einer MTS-Datei 55
Tabelle 5: Komprimierungsarten im *.CO-Format 65
Tabelle 6: Übersicht über ausgewählte Eigenschaften der behandelten Web3D-Formate 75

Abkürzungsverzeichnis

API	- Application Programming Interface
BIFS	- Binary Format for Scenes
BWG	- Browser Working Group
CCT	- Collapsible Cylindrical Trees
CONTIGRA	- COmponent-orieNted Three-dimensional Interactive GRaphical Applications
DOM	- Document Object Model
DTD	- Document Type Definition
EAI	- External Authoring Interface
HTTP	- Hypertext Transfer Protocol
IEC	- International Electrotechnical Commission
ISO	- International Organization for Standardization
LOD	- Level of Detail
MD	- Macromedia Director
MIME	- Multipurpose Internet Mail Extension
MPEG	- Moving Picture Experts Group
MRM	- Multi-Resolution Mesh
PDA	- Personal Digital Assistant
RTP	- Real-Time Protocol
RTSP	- Real-Time Streaming Protocol
SAI	- Scene Authoring Interface
TCP	- Transmission Control Protocol
TCP/IP	- Transmission Control Protocol over Internet Protocol
SDS	- Subdivision Surfaces
SDK	- Software Development Kit
XML	- Extensible Markup Language
SDL	- Scene Description Language
URL	- Uniform Resource Locator
UTF	- Unicode Transformation Format
VMP	- Viewpoint Media Player
VR	- Virtual Reality
VRML	- Virtual Reality Modeling Language
VSB	- Viewpoint Scene Builder
WWW	- World Wide Web

Formatverzeichnis

Hier werden die wichtigsten Dateiendungen für 3D-Grafik-, Audio-, Bild- und Videoformate aufgeführt, die in dieser Arbeit im Zusammenhang mit den Web3D-Formaten erwähnt werden. Dieses Formatverzeichnis besitzt keinen erklärenden Charakter, soll aber dem Textverständnis und der Unterstützung einer entsprechenden Recherche dienen.

3D-Grafikformate

ASE	- 3ds max ASCII Export Format
DXF	- Drawing Exchange Format (AutoCAD-Format)
NFF	- Neutral ASCII File Format
OBJ	- Alias|Wavefront *.OBJ-Dateiformat

Audioformate

AIFF	- Audio Interchange File Format
AU	- Audio
AVI	- Audio Video Interleave
MID, MIDI	- Musical Instrument Digital Interface
MP3	- MPEG Audio Layer 3
PHN	- Phoneme File
SWA	- Shockwave Audio
WAV	- Wave

Bildformate

CGM	- Computer Graphics Metafile
BMP	- Bitmap
GIF	- Graphics Interchange Format
JPEG, JPG	- Joint Photografic Expert Group
PIC	- Picture
PCT	- Macintosh PICT
PNG	- Portable Network Graphic Format
TGA	- Targa Graphics Adapter Image File

Videoformate

AVI	- Audio Video Interleave
MPEG	- Motion Pictures Expert Group

1 Einleitung

Die interaktive 3D-Grafik im WWW erlebt zur Zeit einen enormen Aufschwung, was insbesondere auf die Neuerscheinungen von Shockwave3D von Macromedia und Adobe Atmosphere zurückzuführen ist. Eine treibende Kraft hierbei ist die Firma Intel, die nach neuen sogenannten „Killer-Applikationen“ für ihre Prozessoren sucht. Der finanzkräftige Prozessorenhersteller ist beispielsweise Partnerschaften mit den Herstellern der Formate Shockwave3D, Viewpoint, Cult3D und Pulse eingegangen und unterstützt diese Technologien dementsprechend. [Fre01, IAL]

Neben diesen proprietären Formaten existieren die standardisierten Formate VRML97 und MPEG-4/BIFS. Für diese werden zur Zeit die zwei Nachfolgeformate X3D und XMT entwickelt. Es ist aber abzusehen, dass X3D und XMT zu einem einzigen Format verschmelzen werden. XMT – welches sich in einem sehr frühen Entwicklungsstadium befindet – wird durch X3D ersetzt werden, wodurch dieses Format im Gegensatz zu seinem Vorgänger VRML97 eine breitere Basis finden wird.
Dieses standardisierte Nachfolgeformat wird trotz einer relativ hohen Entwicklungsdauer eine sehr starke Marktrelevanz erreichen, da es sich nicht wie die proprietären Formate auf einzelne Anwendungsbereiche beschränkt und eine gut strukturierte und mächtige Szenenbeschreibungssprache für interaktive 3D-Grafik im WWW darstellen wird.

Das Ziel dieser Arbeit ist es, bestehende und sich in Entwicklung befindliche Web3D-Technologien zu analysieren und mittels der gewonnenen Erkenntnisse Vorschläge für das CONTIGRA-Projekt zu unterbreiten.
Der relativ neue Begriff „Web3D“ bezeichnet alle offenen, aber auch proprietären Technologien, die interaktive 3D-Grafiken im WWW darstellen können. Diese Definition benutzt bewusst den Begriff „Technologie“ statt „Format“, da zur Repräsentation der Szene durch ein Format auch die entsprechenden Technologien – wie beispielsweise Client/Server-Architektur, Autorenwerkzeuge, Skalierung, Kompression und Streaming – gehören. Diese Sichtweise wurde auch vom Autor verfolgt und äußert sich insbesondere in der Betrachtung des Erstellungsprozesses, der Server-seitigen Repräsentation, der Kommunikation und der Präsentation der Web3D-Applikation.

Die vorliegende Arbeit beschäftigt sich zuerst mit den Grundlagen von Web3D-Grafik. Anschließend erfolgt mit Hilfe eines umfangreichen Kriterienkataloges die Beschreibung und Charakterisierung der einzelnen Web3D-Formate. Hierbei werden die Formate entsprechend der Existenz einer Standardisierung oder entsprechender Bestrebungen in zwei Kategorien unterteilt. Dieser Teil der Arbeit erhält durch die umfangreichen Angaben zu Literaturquellen, WWW-Adressen und Arbeitsgruppen und durch die stark strukturierte Aufführung der Fakten den Charakter eines Handbuches, wodurch mögliche Startpunkte für weitere Recherchen gegeben werden.

Mit Hilfe der gewonnenen Informationen werden dann Vorschläge für das CONTIGRA-Projekt unterbreitet. Abschließend erfolgt die Darstellung der, mit dieser Belegarbeit verbundenen praktischen Aufgabe, die zur Vertiefung in das Format Shockwave3D diente.

Der Aufgabenteil „Konzeption und prototypische Realisierung von Teilaspekten der CONTIGRA-Architektur unter Berücksichtigung von Komponentenansätzen“ wurde durch das Kapitel 5 „Vertiefende praktische Aufgabe“ ersetzt.

2 Grundlagen von Web3D-Grafik

In diesem Kapitel erfolgt zuerst die Betrachtung der Bestandteile einer Web3D-Szene. Anschließend werden die verschiedenen Anwendungsklassen von Web3D-Grafik beschrieben. Danach wird die Notwendigkeit eigener 3D-Formate für das Internet aufgezeigt. Weiterhin werden die drei zur Zeit wichtigsten Probleme bei den Web3D-Technologien charakterisiert und entsprechende Lösungsansätze betrachtet.

2.1 Die Bestandteile einer Web3D-Applikation

Eine Web3D-Szene bezeichnet einen dreidimensionalen Raum, in dem 3D-Objekte angeordnet sind. Die 3D-Objekte können durch Oberflächen- und Volumenmodelle sowie Freiformobjekte repräsentiert werden. Der Begriff „Objekt“ bezeichnet innerhalb dieser Arbeit immer einer 3D-Objekt. In den Web3D-Technologien werden meist folgende Repräsentationsformen verwendet:

Oberflächenmodelle
Meist erfolgt die Repräsentation der Oberfläche durch Polygonnetze, welche oft auch als Geometrie bezeichnet werden. Es werden vorwiegend Dreiecksnetze verwendet, da Grafik-APIs wie DirectX oder OpenGL und die Hardware für das Rendering von Polygonen mit mehr als drei Eckpunkten *(Vertices)* nicht optimiert sind. Man vermeidet durch die Verwendung von Dreiecksnetzen die aufwendige Tesselation beim Client.
Die Beschreibung von Objekten durch bikubische Parameteroberflächen, wie beispielsweise NURBS, wird in den betrachteten Web3D-Technologien nur sehr selten verwendet, da eine clientseitige Tesselation noch nicht ausgereift ist und sehr rechenaufwendig wäre [Jun00].

Volumenmodelle
Volumenmodelle werden meist durch Instanzierung von geometrischen Primitiven und durch *Sweep*-Modelle dargestellt. Primitive Objekte wie zum Beispiel Quader, Kegel, Zylinder und Kugel stellen atomare Grundbausteine für eine 3D-Szene dar. *Sweep*-Modelle sind Formen, wie z.B. Vasen, Schraubendreher, Bananen und Weingläser, die durch Extrusion erzeugt werden können. Extrusion bezeichnet das „Ziehen“ einer zweidimensionalen Geometrie durch den dreidimensionalen Raum. Damit sich beispielsweise aus einem Kreis nicht nur Rohre generieren lassen, kann man durch Veränderung der zweidimensionalen Geometrie und des Extrusionspfades während der Extrusion auch kompliziertere Formen erstellen.

Freiformobjekte
Freiformobjekte sind Formen, wie Wasser, Feuer, Fell u.a., die in der Natur vorkommen und aufgrund Ihrer Komplexität nicht durch Oberflächen- oder Volumenmodelle realisiert werden können. Freiformobjekte werden durch verschiedene mathematische Algorithmen oder Prozeduren, wie z.B. fraktale Generatoren oder Partikelsysteme, erzeugt.

Komplexere und realistischere Objekte entstehen durch die Kombination der verschiedenen Modell- und Objekttypen.

Die verschiedenen Objekte haben in der Realität nicht nur eine bestimmte Form, sondern auch Farb- und Materialeigenschaften, welche unter dem Oberbegriff „Objekterscheinung“ zusammengefasst werden. Die Materialeigenschaften können beispielsweise durch die Grundhelligkeit, den Glanz, die Oberflächenstruktur, die Transparenz oder durch ein Eigenleuchten bestimmt werden. Durch die Verwendung von Texturen – zweidimensionale Bilder, die auf dreidimensionale Objekte projiziert werden – können die Objekte natürlichere und komplexere Oberflächen erhalten.

Die Darstellung von Text kann entweder durch eine entsprechende Funktionalität des Formates oder durch eine Textur, die einen Text in Form eines Bildes enthält, realisiert werden.

Weiterhin muss in der 3D-Szene für eine entsprechende Beleuchtung gesorgt werden, was durch verschiedene Lichtquellen – wie beispielsweise Distanzlicht, Punktlicht oder Spotlicht – erreicht werden kann. Virtuelle Kameras, Video- und Audiosequenzen, Szenenhintergründe, Nebel und Text sind weitere Bestandteile einer 3D-Szene.

Die Übergänge zwischen einer Szene und einem Objekt sind fließend. Falls man eine bestehende Szene in eine andere Szene einfügt, wird diese erstgenannte zu einem Objekt innerhalb der neuen Szene. Die Begriffe „Szene“ und „Applikation“ besitzen in dieser Arbeit dieselbe Bedeutung.

Durch Animation und Interaktion wird eine statische 3D-Szene um dynamische Eigenschaften erweitert. Auf Interaktionen des Benutzers mit der virtuellen Umgebung, wie z.B. bei dem Betreten einer bestimmten Region, bei dem Berühren eines Objektes oder auf Tastatureingaben, kann mittels verschiedener Sensoren reagiert werden. Durch einen Sensor wird ein Ereignis ausgelöst, das in einem Ereignismodell Berücksichtigung findet. Das Ereignismodell verändert anschließend – eventuell unter Berücksichtigung des aktuellen Zustandes – die Szene. Dabei können Objekte und der innere Zustand der Szene verändert oder auch Animationen ausgelöst werden.
Es können auch Ereignisse ohne die Interaktion des Benutzers ausgelöst werden, wie beispielsweise beim Starten der Applikation oder nach dem Ablauf eines bestimmten Zeitintervalls.
Für die Gesamtheit der dynamischen Aspekte einer Web3D-Applikation, die sich durch Animationen, Interaktionen und die Veränderung des inneren Zustandes der Szene definieren, wird in dieser Arbeit die Bezeichnung „Verhaltensweisen“ benutzt.
Die typischen Eingabegeräte für das Auslösen von Ereignissen durch den Nutzer sind die Maus und die Tastatur. Die Präsentation der Web3D-Applikation erfolgt meist durch ein nicht-immersives Ausgabegerät, einen Monitor.

[Wal01, Klo97, Scr01]

2.1.1 Grundlegende Client/Server-Architektur

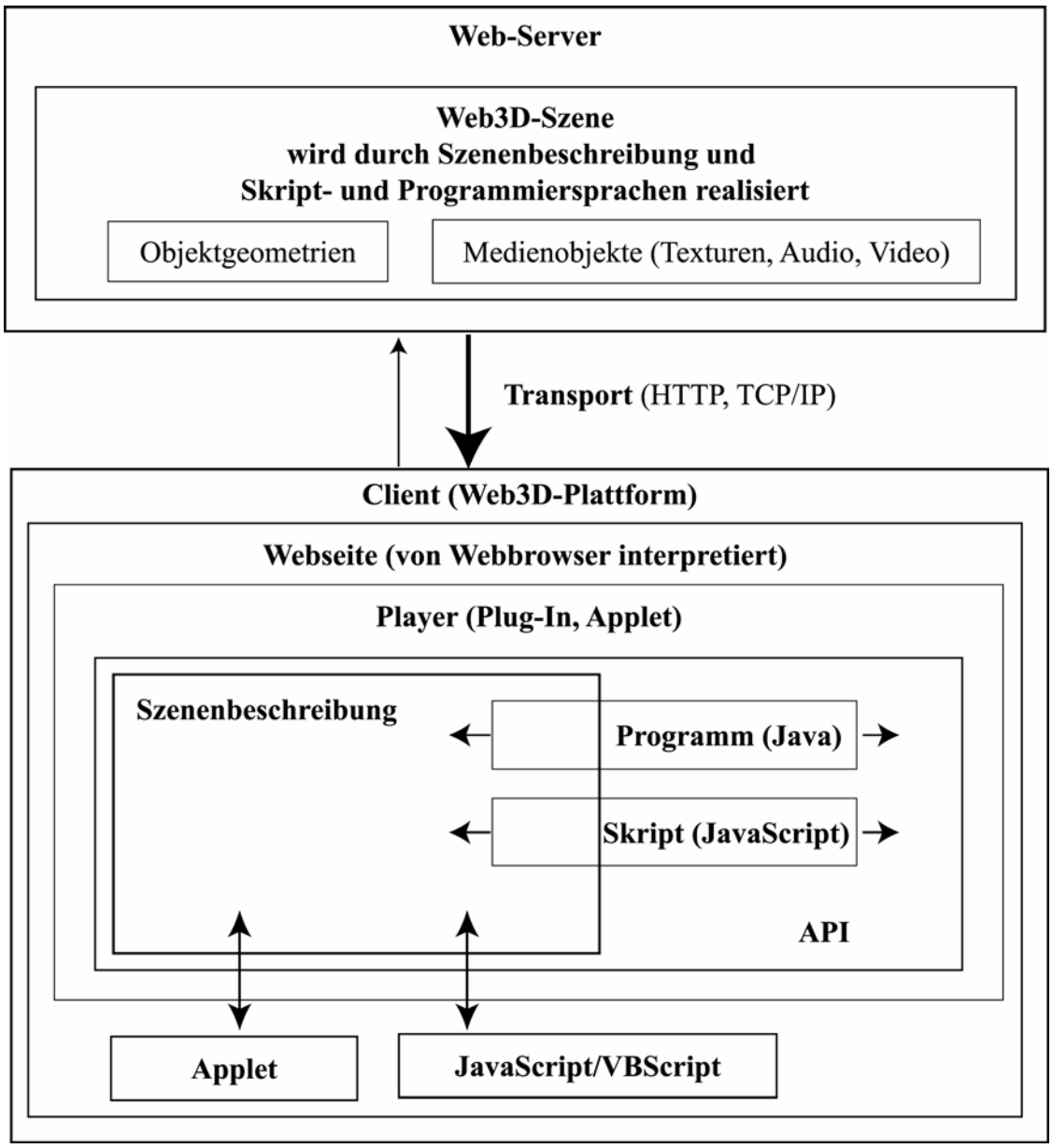

Abbildung 1: Grundlegende Client/Server-Architektur

Die Repräsentation einer Web3D-Applikation erfolgt durch einen Webserver, auf dem die Szenenbeschreibung und die eventuell dazugehörigen Objektgeometrien und Medienobjekte enthalten sind. Die bidirektionale Kommunikation zwischen Client und Server erfolgt meist mittels des HTTP-Protokolls, welches wiederum auf das TCP/IP-Protokoll aufbaut.
Auf dem Client – welcher in diesen Zusammenhang auch als Web3D-Plattform bezeichnet wird – erfolgt dann die Darstellung der Szene durch einen Player innerhalb der Webseite. Der Begriff „Player“ bezeichnet in dieser Arbeit die Software für die Interpretation der Web3D-Szene, welche entweder als Java-Applet oder Plug-In realisiert wurde.

Die Person, die am Client mit der Szene interagiert, wird in dieser Arbeit als „Nutzer“ bezeichnet.

2.1.2 Interaktionsmöglichkeiten mit einer Web3D-Applikation

Die Interaktionsmöglichkeiten mit einer Web3D-Szene lassen sich in die Bereiche Objektmanipulation, Blickpunktmanipulation und Anwendungskontrolle unterteilen.

Objektmanipulation
Die direkten Interaktionen mit den Objekten, wie z.B. auswählen, verschieben, skalieren, rotieren, hinzufügen, verändern und löschen, stellen elementare Interaktionsmöglichkeiten des Nutzers mit der Szene dar.
Hierbei kann zwischen realen Verhaltensweisen wie verschieben und rotieren und in der realen Welt unmöglichen Verhaltensweisen wie skalieren und löschen unterschieden werden. Für Objektmanipulationen wird auch der Begriff „Selektion und Manipulation“ verwendet.

Blickpunktmanipulation
Der Begriff „Blickpunktmanipulation“ beschreibt einerseits das Navigieren durch die Szene durch Bewegung des virtuellen Auges und andererseits die Veränderung der Szene gegenüber eines nichtbeweglichen Betrachters. Das Bewegen innerhalb eines virtuellen Gebäudes wird durch die Bewegung eines virtuellen Auges hervorgerufen. Bei Produktpräsentationen wird meist das entsprechende Objekt rotiert, gezoomt oder innerhalb des Blickfeldes verschoben. Dies wird auch als *Single Object 3D Animation* bezeichnet.

Anwendungskontrolle
Anwendungskontrolle beschreibt die Kommunikation zwischen dem Benutzer und dem Teil der Anwendung, der nicht direkt zur Szene gehört. Die indirekte Interaktion des Benutzers mit der Szene über die Funktionalität einer Webseite gehört zu diesem Bereich. Beispielsweise kann durch Anklicken eines Links innerhalb der Webseite eine Blickpunktmanipulation ausgelöst werden.

[Han97, Bow99]

2.1.3 Realisierung von Verhaltensweisen

Es gibt drei grundsätzliche Ansätze für die Realisierung von Verhaltensweisen:

Realisierung als Teil der Szenenbeschreibungssprache
In der Szenenbeschreibungssprache existieren nicht nur Konstrukte für die Definition von Objekten und deren Eigenschaften, sondern auch die Möglichkeiten zur Definition von einfachen Animationen und Interaktionen.

Integration von Skript- und Programmiersprachen
In die Szenenbeschreibung können Skriptsprachen, wie beispielsweise JavaScript, integriert werden. Weiterhin kann die Möglichkeit bestehen, innerhalb der Szenenbeschreibung auf externe Programme, die z.B. in der Programmiersprache Java erstellt wurden, zuzugreifen.

Externer Zugriff auf die Szene über eine API
Durch entsprechende APIs kann eine Kommunikation zwischen der Webseite und dem Player – beispielsweise durch ein Java-Applet oder durch Skripte innerhalb der Webseite – erfolgen. Mittels dieser Schnittstelle kann auf die Szene zugegriffen werden.

Diese drei Ansätze für die Realisierung von Verhaltensweisen sind innerhalb der verschiedenen Web3D-Technologien unterschiedlich ausgeprägt und mächtig. [Bro98]

2.1.4 Szenenbeschreibung

Eine 3D-Szene wird durch eine hierarchische baumähnliche Struktur, die man als gerichteten azyklischen Szenengraphen bezeichnet, beschrieben. Dieser enthält die Beschreibung für die einzelnen Objekte und definiert die Beziehungen zwischen ihnen. Die Definition des Szenengraphen kann grundsätzlich durch ein deklaratives 3D-Austauschformat oder durch objektorientierte oder imperative Programmierung erfolgen.

Die Austauschformate beruhen auf einer deklarativen Szenenbeschreibungssprache *(Scene Description Language, SDL)*, mit deren Hilfe die Bestandteile einer Web3D-Szene modelliert werden können. Die SDL besitzt meist eine einfache Syntax und ist durch eine Grammatik beschrieben. Falls diese durch eine XML-Grammatik in Form einer DTD oder eines Schemas definiert ist, entstehen beispielsweise folgende Vorteile:

- Verwendung einer standardisierten Syntax
- Vorhandensein einer hierarchischen Struktur
- Komfortables dynamisches Generieren oder Verändern der Szenenbeschreibung
- Existenz komfortabler Schnittstellen – beispielsweise das *Document Object Model* (DOM) – für den Zugriff auf das Dokument
- Erweiterbarkeit und bessere Verarbeitbarkeit des Formates
- Komfortable Konvertierungsmöglichkeiten
- Verwendung von XML-Editoren
- Integration anderer XML-Grammatiken

Die objektorientierte oder imperative Web3D-Programmierung basiert entweder auf einer verbreiteten Programmiersprache wie beispielsweise Java oder auf einer formatspezifischen Skript- oder Programmiersprache. Für den ersten Fall stehen APIs zur Verfügung, mit deren Hilfe Web3D-Applikationen erstellt werden können. Bei den formatspezifischen Skript- oder Programmiersprachen werden entsprechende APIs nicht benötigt, da diese Sprachen für die Erstellung von Web3D-Applikationen konzipiert wurden und deren Sprachumfang somit die benötigten Konstrukte enthält.

Die Definition einer Szene erfolgt beispielsweise durch Konstrukte für die Modellierung von:

- Objekten (grafische Primitive, Dreiecksnetze, Höhengitter, Extrusionen)
- Oberflächeneigenschaften (Farbe, Textur)
- Koordinatentransformationen (Rotation, Translation)
- Beleuchtung
- Szenenumfeld (Hintergrund, Nebel)
- Benutzerführung
- Gruppierung und Instanzierung
- Hierarchien

Es existieren auch binäre Formate für die Beschreibung von Web3D-Applikationen, die ausschließlich durch entsprechende Programme erzeugt und gelesen werden können.

[Scr01]

2.2 Anwendungsklassen von Web3D-Applikationen

Theoretisch gibt es unendlich viele Anwendungsklassen, da jede Szene aus der natürlichen und fiktiven Welt dargestellt werden kann. Es sollen hier die wichtigsten bisher realisierten Anwendungsklassen aufgeführt und ihre Anforderungen an die Web3D-Technologien aufgezeigt werden.

Bisher fanden folgende Anwendungsklassen besonderes Interesse:

E-Commerce
In einer Shopping Mall bewegt sich der Nutzer durch ein dreidimensionales virtuelles Einkaufszentrum, welches als Rahmen für die Produktpräsentation dient. Hierbei werden hohe Ansprüche an die Fenstergröße der Web3D-Applikation und an die Textdarstellung für die Navigation gestellt. Die Detailgenauigkeit kann zugunsten einer hohen Interaktionsgeschwindigkeit vernachlässigt werden.

Bei Produktpräsentationen erfolgt die Darstellung eines einzelnen Gegenstandes. Durch die dreidimensionale Visualisierung eines Produktes sollen nicht nur die Verkaufszahlen gesteigert, sondern auch die Rücksendequote durch die bessere Vermittlung der Funktionalität des Gegenstandes verringert werden. Man kann auch fiktive Produkte präsentieren, die erst bei entsprechendem Kundeninteresse realisiert werden. Hohe Detailgenauigkeit bei der Darstellung der Produkte ist von elementarer Bedeutung. Weiterhin sollten Interaktionsmöglichkeiten – insbesondere zwischen Webseite und Applikation – realisiert werden können.

Abbildung 2: Dreidimensionale Präsentation einer Uhr [Breitling]

Auch die Erweiterung von Bedienungsanleitungen und FAQs durch dreidimensionale Produktvisualisierungen kann zu Kosteneinsparungen im Support-Bereich führen und ist somit dem Bereich E-Commerce zuzuordnen.

Die räumliche Darstellung architektonischer Gegebenheiten von Häusern oder Wohngebieten nimmt immer mehr an Bedeutung zu. Hierbei kann man sich beispielsweise seine Inneneinrichtung oder sogar ein Haus aus einzelnen Bausteinen zusammensetzen, was zu einem gesteigerten Kaufinteresse führen könnte.

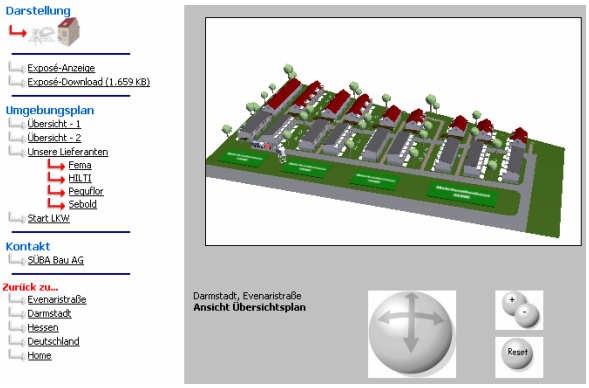

Abbildung 3: Wohngebiet [iTEConsult]

Auch virtuelle Berater, die eine bessere Kommunikation mit dem Nutzer ermöglichen, werden verstärkt realisiert [Charamel, Telopolis].

Unterhaltungsbereich

Spiele – speziell in kooperativen virtuellen Welten – sind von besonderer Bedeutung, da sie zu einer größeren Verbreitung von Web3D-Applikationen führen könnten. Eine sehr beliebte Ausprägung im Spielebereich sind die *First Person 3D Shooter*, in denen man sich aus dem Blickwinkel einer Person durch eine virtuelle Welt bewegt und dort bestimmte Aufgaben löst. Dieser Bereich stellt sehr hohe Anforderungen an die Detailgenauigkeit der Darstellung und somit auch an die verwendete Web3D-Plattform.

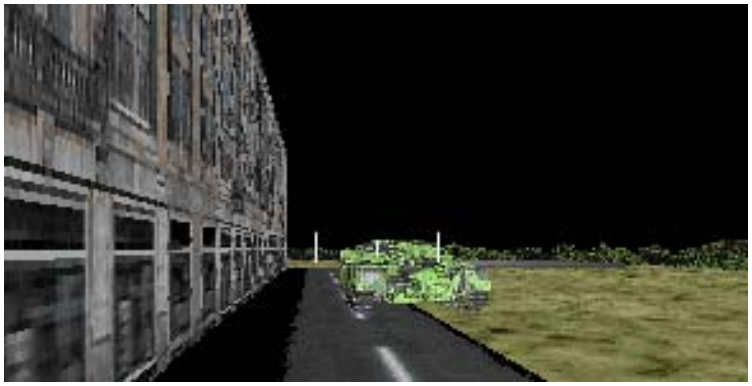

Abbildung 4: Shout3D Tank Game [TankGame]

Bei Sport-Applikationen geht es darum, dass der Nutzer aktiv an dem realen Geschehen teilnehmen kann oder passiv das Sportereignis mit zusätzlichen Informationen und eigener Kameraführung konsumiert. In beiden Fällen muss das beispielsweise durch Sensoren oder Videoaufnahmen gewonnene Datenmaterial der Sportveranstaltung für die Verwendung in der Web3D-Applikation aufbereitet werden.

Im Blendo-Projekt [Blendo] von Sony arbeitet man z.B. daran, dass man mit Hilfe eines virtuellen Cockpits an einem realen Autorennen teilnehmen kann.

Abbildung 5: CART Auto Racing [Blendo]

Durch das von Orad Net in Zusammenarbeit mit Intel zur Zeit entwickelte System ToPlay [TOPlay] soll es möglich sein, sich komplette Fußballspiele in Form einer 3D-Simulation anzuschauen.

Abbildung 6: 3D-Fußballsimulation [TOPlay]

Im Unterhaltungsbereich findet auch die Darstellung von Comics und Sketchen in Form von Charakteranimationen großen Zuspruch. Hier sind Detaillierungsgrad und Interaktionsfähigkeit oft von zweitrangiger Bedeutung. Beispielsweise werden auf den Webseiten des Senders NBC Sketche des Moderators der Tonight Show [JayLeno] in Form einer 3D-Charakteranimation visualisiert.

Abbildung 7: Jay Leno - Sketch [JayLeno]

Informationsvisualisierung
Bei Lehr- und Lernanwendungen ist die Realisierung von umfangreichen Animationen und komplexen Interaktionen oftmals wichtiger als eine hohe Detailgenauigkeit.

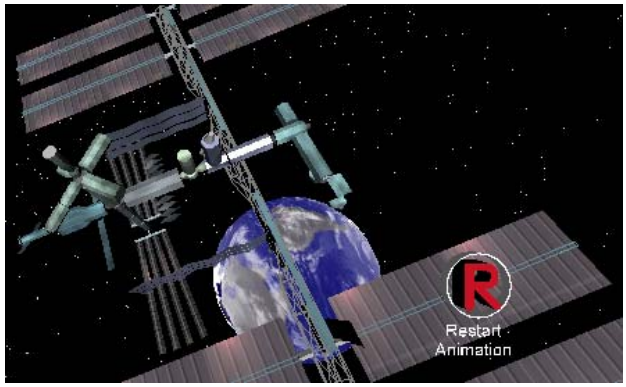

Abbildung 8: International Space Station [ISS]

Durch die Verwendung der dritten Dimension zur Navigation durch umfangreiche und strukturierte Datenbestände kann die Informationsfindung erleichtert werden. Hierbei sind die Textdarstellung und die Interaktionsgeschwindigkeit von elementarer Bedeutung, wobei die Detailgenauigkeit möglicherweise vernachlässigt werden kann.

Abbildung 9: 3D-Sitemap des Lehrstuhles MMT an der TU-Dresden [MMT]

Durch dreidimensionale virtuelle Stadtpläne kann sich der Nutzer von den örtlichen Gegebenheiten einer Stadt oder eines Gebäudes einen Überblick verschaffen.

Abbildung 10: Pariser Platz [e:Berlin]

Anforderungen der Anwendungsklassen an die Web3D-Technologie:

	Detail-genauigkeit	Interaktions-möglichkeiten	Kooperativität	Textdarstellung
E-Commerce				
Shopping Mall	0	++	+	++
Produktpräsentation	++	+	-	++
Kundenbetreuung	++	++	0	++
Architektur	+	++	0	0
Unterhaltungsbereich				
Spiele	++	++	++	0
Sport	++	+	0	+
Charakteranimationen	0	0	--	+
Informationsvisualisierung				
Lehr- und Lernanwendungen	+	++	+	++
Navigation	+	++	-	++
Virtuelle Stadtpläne	+	++	0	++

++ Sehr wichtig, + wichtig, 0 wünschenswert, - unwichtig, -- völlig unwichtig

Tabelle 1: Anwendungsklassen und deren Anforderungen an die verwendete Web3D-Technologie

2.3 Notwendigkeit eigener 3D-Formate für das WWW

Man kann die existierenden 3D-Formate je nach Einsatzzweck und Anforderungscharakteristik in folgende vier verschiedene Kategorien einteilen: CAD/CAM/CAE, Modellierung und Animation, Echtzeitgrafik / VR und Web3D.

	CAD/CAM/ CAE	Modellierung und Animation	Echtzeitgrafik / VR	Web3D
Präzision	Sehr hoch	Hoch	Gering	Sehr gering
Objekt-modellierung	Ja	Ja	Ja	Ja
Szenen-modellierung	Nein	Ja	Ja	Ja
Animation	Nein	Ja	Ja	Ja
Interaktionen	Nein	Nein	Ja	Ja
Echtzeitfähigkeit	Nein	Nein	Ja	Ja
Geringe Dateigrößen	Geringe Bedeutung	Geringe Bedeutung	Geringe Bedeutung	Sehr wichtig
Relation zu anderen Internetstandards	Unbedeutend	Unbedeutend	Unbedeutend	Sehr wichtig
Unterstützung leistungsschwacher Zielplattformen	Geringe Bedeutung	Geringe Bedeutung	Geringe Bedeutung	Sehr wichtig

Tabelle 2: Anforderungscharakteristik für 3D-Formate

Die Notwendigkeit von Web3D-Formaten – die große Gemeinsamkeiten mit den Echtzeitgrafik- und VR-Formaten besitzen – ergab sich durch die Bereitstellung von interaktiven 3D-Szenen im WWW. Erweiterte Anforderungen wie

- Relationen zu anderen Internetstandards (HTML, JavaScript, Java)
- Geringe Dateigrößen (Komprimierung)
- Unterstützung leistungsschwacher Zielplattformen (Skalierung)
- Multiuseranwendungen
- Umfangreiche Möglichkeiten zur Verhaltensmodellierung

sollten durch das 3D-Format erfüllt werden.

[Scr01]

2.4 Hauptprobleme von Web3D-Grafik

Die drei größten Problemfelder der Web3D-Technologien sind zur Zeit:

- leistungsschwache Web3D-Plattformen

  Das Rendering einer Web3D-Applikation stellt hohe Anforderungen an die Web3D-Plattform, welche eine Kombination aus Soft- und Hardware darstellt, die das Visualisieren von und Interagieren mit einer Szene ermöglicht. Die Web3D-Plattform benötigt für die Darstellung einer 3D-Szene einen Webbrowser, der entweder durch ein entsprechendes Plug-In oder durch ein automatisch geladenes Java-Applet erweitert wurde.

  Die Darstellung durch ein Java-Applet ist nur für kleine 3D-Szenen mit einfachen Objektgeometrien und Objekterscheinungen sinnvoll, da ein Java-Applet durch die virtuelle Java-Maschine eine nicht performante Interpretation erfährt. Die Benutzung der Klassenbibliothek Java3D setzt das Vorhandensein einer entsprechenden Laufzeitumgebung voraus, welche bei vielen Clients noch nicht installiert ist und kann somit nur eingeschränkt eingesetzt werden.

  Durch die Installation und Verwendung von Plug-Ins, welche meist die performanten Grafik-APIs DirectX oder OpenGL nutzen, können auch komplexe 3D-Szenen dargestellt werden.

  Ein Applet hat gegenüber einem Plug-In den Vorteil, dass keine Installation notwendig ist. Die Installation eines Plug-Ins stellt einen ungewünschten und deshalb oft vermiedenen Eingriff in das System des Nutzers dar.

  Der Client sollte mindestens folgende Hardwareanforderungen erfüllen
  - Prozessor mit mind. 500 MHz und 3Dnow!- oder ISSE-Erweiterung
    Der Prozessor wird für Transformation (Skalieren, Drehen, Verschieben), für die Beleuchtung der Objekte und für die Anwendungslogik (Interaktionen, Kollisionserkennung) benutzt.
  - Grafikkarte mit 3D-Beschleuniger für das *Texture Mapping* und *Shading*
  - Minimale Arbeitsspeichergröße: 128 MByte

  Eine zu geringe Rendering-Leistung der Web3D-Plattform kann zu hohen Animations- und Interaktionslatenzen und zu einer schlechten Darstellungsqualität führen.

- eine zu geringe Bandbreite zwischen Client und Server

  Die Datenmenge einer 3D-Szene, welche insbesondere durch komplexe Objektgeometrien, Animationen und Texturen und durch Video- und Audiosequenzen erhöht wird, sollte mit einer möglichst hohen Datenübertragungsrate übermittelt werden. Durch die steigende Verbreitung von 56K/V90-Modems, ISDN- und TDSL-Verbindungen und der vermehrten Anwendung von Modularisierungs-, Kompressions-, Streaming- und Skalierungs-Techniken in den Web3D-Technologien verschwindet diese Problematik zunehmend.

- das Fehlen spezialisierter Autorenwerkzeuge

  Es existieren viele semi-professionelle Modellierungstools für die Erstellung von Web3D-Applikationen – wie beispielsweise das nicht mehr vertriebene Cosmo Worlds oder Spazz3D [Spazz3D] – deren Funktionalitäten zur Modellierung von Objekten im Vergleich zu Maya [Maya] oder 3ds max [3ds max] nur ungenügend sind.
  Dieses Problem versucht man dadurch zu entschärfen, dass man für professionelle Modellierungstools – die nicht zur Verhaltensmodellierung geeignet sind – entsprechende Exporter erstellt. Diese generieren dann ein bestimmtes Format, das anschließend durch ein weiteres Modellierungsprogramm zur Verhaltensdefinition weiterbearbeitet wird. Problematisch hierbei ist aber, dass bei einem Export das Objekt oder die Szene dem Zielformat angepasst werden muss, wodurch Informationen konvertiert oder weggelassen werden müssen. Es ist meist dokumentiert, welche Funktionalitäten des Modellierungsprogramms durch den Exporter exportiert werden können. Man kann nun bestehende Objekte für den Export anpassen oder sich bei der Modellierung auf die konvertierbaren Funktionalitäten beschränken.
  Die Wiederverwendung von bestehenden Objekten oder Szenen durch den Import in ein Modellierungsprogramm oder durch die Konvertierung in ein anderes Format ist meist problematisch, da das Zielformat eventuell einige Funktionalitäten nicht unterstützt und dessen vorteilhafte Konzepte möglicherweise nicht genutzt werden.

  Die Verhaltensmodellierung für die Web3D-Applikation wird oft vernachlässigt und muss meist durch den Autor selbst programmiert werden oder bezieht sich nur auf ein bestimmtes Anwendungsgebiet.

Die Kombination aus einer geringen Bandbreite und einer leistungsschwachen Web3D-Plattform führt zu einer hohen Startup-Latenz, da viel Zeit für Abruf, Transport, Decodieren und Rendering der Web3D-Szene in Anspruch genommen wird.

Ein weiteres Problem sind die unterschiedlichen Benutzungsschnittstellen der einzelnen Player, die von jedem Nutzer erneut Einarbeitungsaufwand beanspruchen.

Zur Zeit gibt es bei den Web3D-Technologien kaum Möglichkeiten für die Verwendung von stereoskopischen Ausgabegeräten und 3D-Interaktionsgeräten.

[Wal01]

2.5 Lösungsansätze für bestehende Probleme bei Web3D-Technologien

In der Vergangenheit erfolgte meist die Übertragung von monolithischen Web3D-Szenen, die hohe Anforderungen an die Bandbreite und die zur Verfügung stehende Rechenkapazität hatten.

Das Problem der geringen Bandbreite versucht man durch modulare Ansätze, Kompression und Streaming zu lösen. Auf eine leistungsschwache Web3D-Plattform kann mittels Skalierung – Anpassung der Szene an die verfügbare Rechenleistung – reagiert werden.
Durch Kompression und Streaming erfolgt eine schnellere und sukzessive Übertragung der 3D-Szene und somit kann der Nutzer sofort – wenn eventuell auch mit eingeschränktem Detaillierungsgrad – mit der 3D-Szene interagieren.

2.5.1 Modulare Ansätze

Eine Szene kann in verschiedene Module aufgeteilt werden, deren Zusammengehörigkeit durch entsprechende Referenzen realisiert wird. Falls sich ein Benutzer beispielsweise durch eine virtuelle Stadt bewegt und er ein Haus betritt, kann dieses aus einem neuen Modul bestehen, welches erst zum Zeitpunkt des Betretens geladen wird. Durch diese Modularisierung der Szenen wird eine hohe Startup-Latenz vermieden, die beim Laden einer monolithischen Szene entstehen würde. Oft sind Mediendaten wie Audio, Video und Texturen nicht direkt in der Szenenbeschreibung enthalten und können somit auch als Module angesehen werden.

Zu den modularen Ansätzen kann auch die clientseitige Speicherung von Daten zur wiederholten Verwendung gezählt werden. Diese Daten sind entweder durch eine entsprechende Bibliothek standardmäßig vorhanden oder werden nach einer initialen Übertragung dort gespeichert. Besonders prädestinierte Daten für die Wiederverwendung sind Texturen und parametrisierbare menschliche Modelle. Diese Ansätze werden auch von der Universal Media - und der H-Anim - Arbeitsgruppe des Web3D-Konsortiums verfolgt [UMWG, H-ANIM].
Durch *Bones*-Animationen können Charakter-Animationen mit geringem Bandbreitenbedarf erstellt werden. Es wird zuerst ein initialer Charakter mit definierten Gelenkpunkten übertragen, welcher anschließend nur durch entsprechende Bewegungsinstruktionen gesteuert wird. Adaptive Algorithmen sorgen für die realistische Deformation des Objektes.

Bei komplexen räumlichen Strukturen, wie beispielsweise eine lange Häuserfront, verzichtet man häufig auf eine Realisierung durch eine umfangreiche Geometrie. Es werden hierbei oft Texturen verwendet, die beispielsweise diese Häuserfront nachbilden.

Abhängig von der Entfernung des Benutzers vom Objekt können verschieden detaillierte Objektversionen *(Levels of Detail, LODs)* angezeigt werden. Um Sprungeffekte zwischen den einzelnen Detaillierungsstufen zu verhindern, sollte zwischen den beiden LODs interpoliert werden. Falls die verschiedenen LODs nicht automatisch generiert werden, entsteht für den Modellierer einer Szene ein erheblicher Mehraufwand. Weiterhin ist es manchmal notwendig die Zuordnung zwischen der Entfernung vom Objekt und dem entsprechenden LOD explizit festzulegen. Wünschenswert wäre es, wenn das LOD mit der geforderten Detailstufe zuerst geladen wird und bei Bedarf höhere Detailstufen nachgeladen werden. Hierdurch könnte die Startup-Latenz verringert werden. [Klo97]

2.5.2 Kompression

Zur Komprimierung von Szenen werden meist hybride Verfahren benutzt, wobei verschiedene Kompressionsverfahren miteinander kombiniert werden. Das Hauptinteresse bei der Kompression liegt in der Komprimierung der umfangreichen Geometrie- und Texturdaten der Objekte. Die restlichen Daten, werden meist durch eine verlustfreie Entropiecodierung – beispielsweise durch das Huffmann-Verfahren – komprimiert.

2.5.2.1 Geometriekompression

Bei komplexen Objekten erfolgt die Darstellung der Oberfläche oft durch eine spezielle Datenstruktur, einem Dreiecksnetz *(Triangle Mesh)*. Für jeden Punkt müssen die X-, Y- und Z-Koordinaten gespeichert werden. Diese Punkte werden dann zu Dreiecken zusammengefasst (z.B. Dreieck A (P1,P2,P3), Dreieck B (P2,P3,P4)).
Da die meisten 3D-Grafikbibliotheken und die Rendering-Hardware auf das Verarbeiten von Dreiecksnetzen optimiert sind, hat die Repräsentation der Oberflächen durch bikubische Parameteroberflächen und Polygone mit mehr als 3 Eckpunkten im Web3D-Bereich nur eine relativ geringe Bedeutung. Diese müssten nämlich durch den Client erst in ein äquivalentes Dreiecksnetz (Tesselation) umgewandelt werden.
Das vorhandene umfangreiche Wissen zur Datenkompression, zum Beispiel für Bilder, kann nicht für zufällige Dreiecksnetze angewandt werden, da diese keine reguläre Struktur besitzen. [Coh99, Vel99]

Man kann zwischen verlustbehafteter und verlustloser Geometriekompression unterscheiden.

Verlustbehaftete Geometriekompression
Grundsätzlich können die Koordinaten der Eckpunkte quantisiert werden, um dadurch Speicherplatz zu sparen.

Bei verlustbehafteter Kompression wird das Polygonnetz vereinfacht und somit die Anzahl der Polygone verringert. Für die Vereinfachung von Polygonnetzen wurden viele verschiedene Verfahren entwickelt, die meist aus vier ähnlichen Bearbeitungsschritten bestehen, aber diese in unterschiedlicher Reihenfolge abarbeiten [Ross98].

1. Einteilung der Eckpunkte in disjunkte Mengen. Ein Cluster ist eine Zelle eines tabellenartigen Rasters, welches auf das Polygonnetz „gelegt“ wird. Die Cluster beinhalten die disjunkten Eckpunktmengen.
2. In jedem Cluster wird ein neuer Eckpunkt *(Attractor)* angelegt. Dieser ist am Ende des Verfahrens der einzige Eckpunkt in dem entsprechenden Cluster, der von den übriggebliebenen Dreiecken benutzt wird.
3. Die Position des neuen Eckpunktes wird bestimmt.
4. Alle Dreiecke mit mehr als einem Eckpunkt in einem Cluster werden entfernt, wenn dabei kein anderes Dreieck zerstört wird.

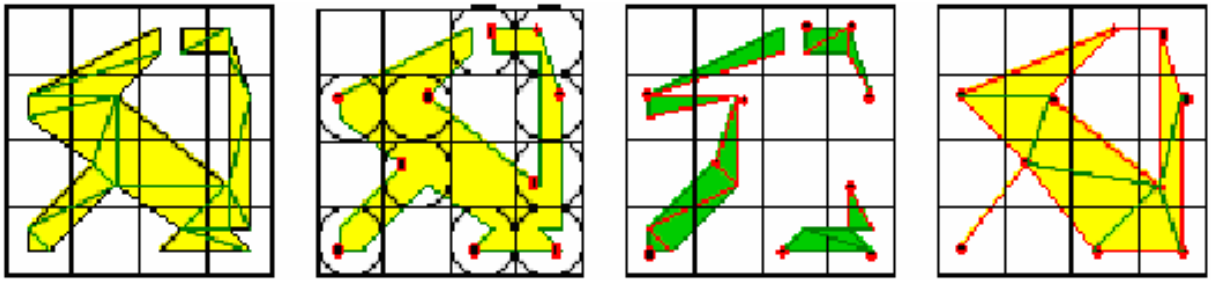

Abbildung 11: Netzvereinfachung [Ross98]

Man versucht weiterhin gleichzeitig zur Reduktion der Polygone auch eine Glättung der Oberfläche zu erreichen. Die verschiedenen LODs eines Objektes können durch verlustbehaftete Geometriekompression erstellt werden.

Verlustlose Geometriekompression
Die verlustlose Kompression versucht den benötigten Speicherplatz für ein Polygonnetz zu verringern [Deer95]. Hierbei gibt es folgende Vorgehensweisen:

Generalisierte Dreiecksstreifen (Generalized Triangle Strips)

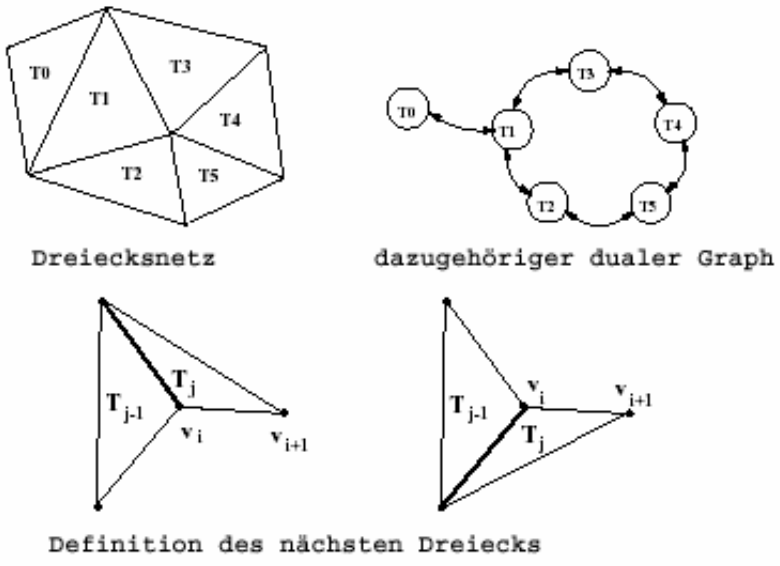

Abbildung 12: Generalisierte Dreiecksstreifen

Als Erstes wird die topologische Struktur des Dreiecksnetzes betrachtet. Hierbei ist es üblich einen dualen Graphen zu erstellen, welcher die Nachbarschaftsrelationen zwischen den Dreiecken verdeutlicht. Das Dreiecksnetz wird so in kleine Teilnetze unterteilt, dass jedes Teilnetz die Hamilton-Eigenschaft besitzt. In einem Hamilton-Dreiecksnetz existiert eine Ordnung $T_1,...,T_N$, so dass zwei aufeinanderfolgende Dreiecke T_j und T_{j+1} sich eine Kante teilen. Ein Dreiecksnetz besitzt die Hamilton-Eigenschaft, wenn ein Pfad in dem dazugehörigen dualen Graphen existiert, der jeden Knoten des Graphen nur einmal durchläuft.

Nehmen wir an, dass ein Dreieck T_{j-1} mit dem Dreieckspunkt v_i existiert, welcher mit dem nächsten Dreieck geteilt wird. Dann reicht es für die Spezifikation des nächsten Dreiecks T_j aus, einen neuen Dreieckspunkt v_{i+1} und die Kante anzugeben, welche das neue Dreieck mit dem alten Dreieck teilt. Somit benötigt man für die Speicherung von N Dreiecken N+2 Eckpunkte und N Bits für die Bestimmung der geteilten Kante (geteilte Kante im Uhrzeigersinn (0) - entgegen dem Uhrzeigersinn (1)).

[Vel99, Ross98, Olb00]

Generalisiertes Dreiecksnetz (Generalized Triangle Mesh)

Das generalisierte Dreiecksnetz stellt eine Kombination aus einem generalisierten Dreiecksstreifen und einem Puffer für mehrfach genutzte Eckpunkte dar. Es passiert häufig, dass Eckpunkte in einem generalisierten Dreiecksstreifen mehrfach auftreten. Bei einem generalisierten Dreiecksnetz werden alte Eckpunkte, die in Zukunft nochmals benötigt werden in einem FIFO-Puffer abgelegt. Bei der nächsten Verwendung des Punktes müssen nicht noch einmal seine Koordinaten abgespeichert werden, sondern nur eine Referenz auf den Puffer. Durch diese Technik kann das Speichern redundanter Punkte größtenteils vermieden werden. [Deer95, Ross98]

Die beiden vorangegangen Verfahren haben zusätzlich den Vorteil, dass eine Erhöhung der Rendering-Rate bis zu einen Faktor 3 erzielt werden kann.

Die Topologische Chirurgie (Topological Surgery)

Bei dieser Topologie-orientierten Kompression wird erst ein Dreieckspunkt entfernt. Anschießend berechnet man anhand eines vorher angelegten *Vertex Spanning Tree* und einem Voraussage-Algorithmus *(Prediction Operator)*, wo dieser Punkt gelegen haben könnte. Danach berechnet man die Differenz aus dem berechneten und dem ursprünglichen Punkt und speichert diese ab. Statt eines kompletten Punktes muss nur noch der errechnete quantisierte Voraussage-Fehler abgespeichert werden. [Ross98, Coh99, Fog01]

2.5.2.2 Texturkompression

Die für die Realitätssteigerung eingesetzten Texturen werden meistens im JPEG-, GIF-, MPEG- oder PNG- Format übertragen. Es existieren aber auch Kompressionsverfahren, die speziell für die Texturkompression entwickelt wurden, wie beispielsweise das Trixels NT - Verfahren welches bei Viewpoint (siehe Gliederungspunkt 3.3.2 „Viewpoint“) verwendet wird.

2.5.3 Streaming

Durch die Verwendung von Streaming können eine verbesserte zeitliche Bandbreitenausnutzung, eine Verringerung der Startup-Latenz, eine sukzessive Verbesserung der Szene und eine Erhöhung der Interaktionsgeschwindigkeit mit der Applikation erreicht werden.

Der Begriff Streaming wird im Zusammenhang mit Web3D-Applikationen in mehreren verschiedenen Bedeutungen verwendet:

2.5.3.1 Nachladen von Szenenbestandteilen

Dies ist die am häufigsten verwendete, aber auch die eingeschränkteste Form des Streamings. Meist erfolgt ein initialer Ladevorgang für die Applikation und im Laufe der Zeit werden die neu benötigten Daten für die Szene nachgeladen. Solche Daten können beispielsweise Szenenbeschreibungen, Medienobjekte oder 3D-Objekte sein. Dieses Vorgehen wird durch Modularisierung der Szenenbestandteile ermöglicht.

2.5.3.2 Zeitabhängige Updates der Web3D-Szene

Hierbei werden Veränderungen in der Szene bzw. eines Teiles daraus mittels Ersetzung durch eine jeweils vollständige, aktualisierte (Teil-)Szenenbeschreibung bewirkt. Solche Updates werden meist durch den Server gesteuert. [Olb00]

2.5.3.3 Progressive Compressed Stream

Die Verwendung von LODs kann zu einem hohen Bandbreitenbedarf führen, wenn jedes LOD eines Objektes unabhängig von dem anderen übertragen wird. Dieses Problem wird durch die *Multiresolution Analysis* gelöst, wobei zuerst die Übertragung eines initialen, detailarmen LODs des Objektes erfolgt und dieses anschließend sukzessiv durch weitere Daten verfeinert wird. Grundlage hierfür bildet die topologische Chirurgie.
Da keine verschiedenen LODs übertragen und angezeigt werden, sondern ein initiales LOD ständig verfeinert wird, können Sprungeffekte zwischen den verschiedenen Detailstufen vermieden werden.
Für die Erstellung eines initialen LODs des Objektes auf der Serverseite wird aus dem Dreiecksnetz des Originalobjektes M_i durch das Entfernen von unabhängigen Eckpunkten u_i eine vereinfachte Version M_{i-1} erstellt. Durch das Entfernen der Eckpunkte entstehen Löcher, welche durch neue Dreiecke *(Patches)* gefüllt werden. Damit man feststellen kann, welche neuen Dreiecke durch das Entfernen eines einzelnen Eckpunktes entstanden sind, werden diese beispielsweise mit einer einheitlichen Farbe markiert. Mit Hilfe der neu entstandenen Dreiecke wird versucht die ursprüngliche Position des entfernten Eckpunktes vorherzusagen. Anschließend werden Verschiebungsvektoren aus der Differenz dieser vorhergesagten Eckpunkte und den dazugehörigen Originalpositionen errechnet. Obwohl man zusätzlich zu dem Verschiebungsvektor noch die Farbinformationen speichern muss, verringert sich dennoch die Größe der insgesamt zu übertragenden Daten für das Originalmodell.
Dieser Vorgang kann beliebig oft wiederholt werden. Dadurch erhält man verschiedene aufeinander aufbauende Detaillierungsstufen des Dreiecksnetzes.

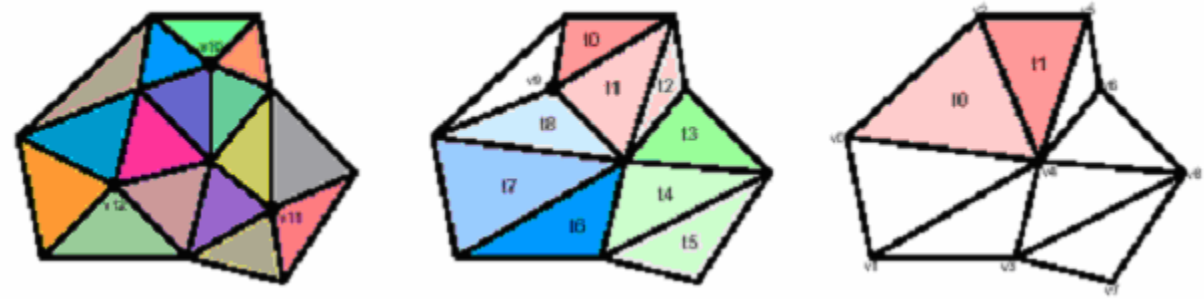

Abbildung 13: *Progressive Compressed Stream* [Coh99]

Nach k Reduktionsschritten wird die Version M_{i-k} zum Client übertragen. Mit Hilfe der Voraussage für einen entfernten Dreieckspunkt und dem entsprechenden Verschiebungsvektor wird die Originalposition rekonstruiert. Somit kann sukzessiv das Dreiecksnetz des Originalobjektes wiederhergestellt werden. Zusätzlich zu der schnellen Darstellung einer detailärmeren 3D-Szene und der fortwährenden Verbesserung der visuellen Qualität, werden auch weniger Daten übertragen, da teilweise nicht alle LODs ausgenutzt werden.
Es gibt auch Verfahren des Geometrie-Streamings, bei denen keine Verringerung der insgesamt zur übertragenden Datenmenge des Objektes erfolgt.

[Coh99, Fog01]

2.5.3.4 Audio- und Video-Streaming

In eine 3D-Applikation können audiovisuelle Medienströme eingebunden werden. Die Videodaten werden meist durch einen virtuellen Videobildschirm – ein mit einer Videotextur versehenes Rechteck in der 3D-Szene – dargestellt. Die Wiedergabe der Audioströme erfolgt entweder mittels räumlich angeordneter Audioquellen oder durch eine nicht räumliche Zuordnung der Audiodaten zur Szene.
Zur Zeit erfolgt nur die Benutzung von klassischem Audio- und Video-Streaming. Streaming-Server, die z.B. Audio- und Videoströme entsprechend der Entfernung des Nutzers skalieren, sind noch nicht realisiert worden. [Olb00]

2.5.4 Skalierung

Unter Skalierung versteht man die dynamische Anpassung der Präsentationsqualität entsprechend der verfügbaren Datenübertragungsrate und der Leistungsfähigkeit der Web3D-Plattform. Diese Skalierung kann beispielsweise durch eine Anpassung der Bildrate, der Objektgeometrien und der Objekterscheinungen erfolgen. Um eine flüssige Animation zu erzielen, sollten aber mindestens 10 Bilder pro Sekunde wiedergegeben werden. [Olb00]
Durch entsprechende Implementierungen mittels der in den Web3D-Technologien einsetzbaren Skript- und Programmiersprachen kann auch oft auf die Umgebung der Szene reagiert werden.

3 Web3D-Formate

Für den Vergleich der einzelnen Formate wurden verschiedene Vergleichskriterien gewählt, die in die Bereiche Erstellung, Repräsentation, Kommunikation, Präsentation und Bewertung unterteilt sind.
In dem Bereich Erstellung werden insbesondere Aussagen über das Dateiformat und den Autorenprozess gemacht. Repräsentation bezeichnet die serverseitige Bereitstellung der Szene und den eventuell möglichen Zugriff des Servers auf die laufende Szene beim Client. Der Bereich Kommunikation beschreibt die Übertragung der Daten zwischen Server und Client. Unter Präsentation versteht man die Anzeige der Szene beim Client.

Die folgenden Kriterien werden für den Vergleich der Web3D-Formate verwendet:

3.1 Vergleichskriterien

3.1.1 Erstellung

Dateiformat
- Ist das Dateiformat standardisiert oder ist es proprietär?
- Handelt es sich um ein deklaratives 3D-Austauschformat oder um objektorientierte bzw. imperative Programmierung?
- Wie erfolgt die Zeichen-Codierung der Szenenbeschreibung?
  Diese kann entweder im Klartext – beispielsweise UTF-8 – oder in binärer Form erfolgen. Falls die Zeichencodierung im Binärformat erfolgt, ist ein Zugriff auf diese Informationen oder deren Erzeugung ohne entsprechende Werkzeuge sehr problematisch.
- Wie lautet die Dateierweiterung?

Internes Modell und interne Struktur
- Wird das interne Modell durch einen Szenengraphen repräsentiert?
- Wie ist die interne Struktur der Datei? Wird die Syntax nur durch ein entsprechendes Dokument spezifiziert oder existiert zusätzlich eine direkt verfügbare XML-Grammatik?

Ereignismodell
Es erfolgt eine Beschreibung, wie Ereignisse ausgelöst werden können und wie ihre anschließende Behandlung erfolgt.

Integration von Skript- und Programmiersprachen
- Welche Skript- und Programmiersprachen können in die Szenenbeschreibung eingebunden werden?
- Wie wird die Manipulation der Szenen- und Objekteigenschaften ermöglicht?

Hierbei soll insbesondere das Zusammenspiel von Szenenbeschreibung und eventuellen Skripten für die Erzeugung von Verhaltensweisen analysiert werden.

Erweiterbarkeit
- Stehen Mechanismen zur Erweiterung der Funktionalität der Szenenbeschreibung zur Verfügung?

Medienintegration
- Welche Art von Medien wie Video, Audio oder Bilder können eingebunden werden?
- Welche verschiedenen Medienformate können eingebunden werden?
  Die explizite Aufführung der unterstützten Medienformate erfolgt unter Gliederungspunkt 3.4 „Übersicht“.
- Wie erfolgt die Einbindung der Medien in die Szene?
- Sind die Medien direkt in der Datei enthalten, die die Szene beschreibt, oder erfolgt eine Referenzierung auf externe Medien?
- Werden verschiedene Detaillierungsstufen der Medien angeboten?
- Wie erfolgt die Behandlung der Medienobjekte?
- Werden die Medienobjekte clientseitig in einem Cache abgelegt?

Werkzeuge
- Was sind die typischen Werkzeuge für die Erstellung einer Web3D-Szene in dem entsprechenden Format?
- Sind nur einfache Texteditoren verfügbar oder existieren visuelle Entwicklungstools für die Erstellung von Web3D-Applikationen? Wie erzeugt man Interaktivität innerhalb dieser Werkzeuge?
- Existieren Exportfunktionen von anderen Tools für das entsprechende Format?

Beispielszene
In einem anschließenden Beispiel soll bei einigen der vorgestellten Formate der grundlegende Aufbau der Datei verdeutlicht werden. Hierbei wird eine Kugel mit einer Textur versehen und nachdem auf die Kugel geklickt wurde, wird diese animiert.

Features
Falls das Format noch weitere Funktionalitäten bietet, die nicht im Text aufgeführt sind und der Autor diese für erwähnenswert hält, erfolgt hier deren stichpunktartige Nennung.

3.1.2 Repräsentation

Repräsentationsformat
- Erfolgt eine Veränderung der Daten vor der Auslieferung an den Client?
- Wie werden die Daten auf dem Server abgelegt?

Softwaretechnologischer Zugriff
- Ist es dem Server möglich auf die laufende Szene beim Client zuzugreifen?

3.1.3 Kommunikation

Skalierung
- Erfolgt eine Skalierung der zu übertragenden Daten entsprechend der zur Verfügung stehenden Datenübertragungsrate?

Streaming
- Wird das Streaming von Daten unterstützt?

Medienauslieferung
- Wie erfolgt die Auslieferung der Medien?

3.1.4 Präsentation

Player
- Mit welchem Player kann die Szene betrachtet werden?
- Wird ein Plug-In oder ein Applet verwendet?

Softwaretechnologischer Zugriff
- Wie kann mittels der Webseite im Webbrowser auf die laufende Szene zugegriffen werden?

Skalierung
- Erfolgt eine Anpassung der Szene an die Leistungsfähigkeit der Web3D-Plattform?

3.1.5 Bewertung

Einsatzgebiet
- Für welche Anwendungsklassen ist das Format prädestiniert?
  Hier werden nur die Bereiche aufgeführt, für die das Format konzipiert ist. Mit den Formaten können oft auch andere Anwendungsklassen abgedeckt werden, wobei hierfür aber ein sehr hoher Aufwand betrieben werden muss.

Nutzungsbedingungen
- Welche Nutzungsbedingungen gibt es für die Verwendung des Formates?

Zukunft des Formates
- Wird das Format weiterentwickelt?
- Gibt es ein Nachfolgeformat?

Fazit
- Wie verbreitet ist das Format?
- Wie ist die Dokumentation des Formates?

Weiterhin erfolgen einige Abschlussbemerkungen zum Format.

Die Art der Kompatibilität oder Konvertierbarkeit der einzelnen Formate zueinander wird im Gliederungspunkt 3.5 „Kompatibilitätsdiagramm“ verdeutlicht.
Falls ein Vergleichskriterium bei dem entsprechenden Format nicht aufgeführt ist, kann dies bedeuten, dass dieses Kriterium oder diese Funktionalität entweder nicht existiert, nicht dokumentiert ist oder schon innerhalb eines anderen Vergleichskriteriums dokumentiert wurde. Es erfolgt keine sture Abarbeitung der einzelnen Kriterien, da die entsprechenden Informationen teilweise nicht verfügbar sind oder im Bezug auf das Format unrelevant sind.
Weiterhin wurde bei dem Vergleich der Formate und deren Charakterisierung ein Abstraktionsniveau eingehalten, das sich auf die grundlegenden Funktionalitäten und den primären Einsatzzweck der Technologie konzentriert.
Die einzelnen Kriterien werden teilweise auch in unterschiedlicher Reihenfolge aufgeführt. Dies dient einer verbesserten Verdeutlichung der Zusammenhänge. Falls beispielsweise ein binäres Format nur durch ein entsprechendes Werkzeug erstellt werden kann, werden innerhalb des Kriteriums „Erstellung“ zuerst Aussagen über das entsprechende Werkzeug gemacht.
Eine tabellenartige Zusammenfassung, die einen schnellen Überblick über die Formate ermöglichen soll, erfolgt im Gliederungspunkt 3.4 „Übersicht“.

3.2 Standardisierte Formate bzw. Vorschläge für Standards

Um Portabilität und Interoperabilität der Web3D-Applikationen zu erreichen, sollten die Formate, Verfahren, Schnittstellen und Protokolle auf der Grundlage einer übereinstimmenden technischen Spezifikation realisiert werden. Nach einem von einer Normungsorganisation wie der ISO/IEC durchgeführten öffentlichen Verfahren wird aus einer Spezifikationen eine Norm.
Bei den sogenannten Gremienstandards besteht aber immer die Gefahr, dass diese erst eine sehr lange Zeit bis zur Fertigstellung benötigen und anschließend so komplex sind, dass eine Umsetzung problematisch ist. [Fey01]

Die in diesem Abschnitt vorgestellten Formate wurden oder werden vom Web3D-Konsortium [Web3D-Konsortium] oder von der Moving Picture Experts Group (MPEG) [MPEG] entwickelt.
Das Web3D-Konsortium ist eine gemeinnützige Organisation, welche das Ziel hat, ein Forum für die Schaffung offener Standards für Web3D-Anwendungen bereitzustellen und die weltweite Verbreitung von darauf basierenden Produkten zu fördern. Die Schaffung dieser Standards erfolgt in den einzelnen Arbeitsgruppen und in Zusammenarbeit mit der ISO. Die Mitglieder des Konsortiums sind verschiedene Firmen (z.B. 3Dlabs, ATI Technologies, Sony Cooperation, ELSA AG, Autodesk/Discreet), universitäre Einrichtungen und Einzelpersonen.
Die Moving Picture Experts Group [MPEG] beschäftigt sich in Zusammenarbeit mit der ISO/IEC mit der Entwicklung von digitalen Repräsentationen für Audio- und Videoinformationen.

Zunächst erfolgt die Charakterisierung von VRML97. Die hier verwendeten grundlegenden Methoden zur Realisierung von interaktiver 3D-Grafik im WWW dienten und dienen als Vorbild für die Entwicklung weiterer Web3D-Technologien. Obwohl die Konzepte des VRML97-Formats innerhalb dieser Arbeit oft als Vergleichskriterium herangezogen werden, erfolgt aufgrund der zahlreich vorhandenen Literatur keine detailliertere Darstellung von VRML97.
Anschließend erfolgt eine Vorstellung des Standes der Arbeit an X3D. Dieses Format stellt eine Weiterentwicklung von VRML97 dar, die möglicherweise zu dessen Nachfolgestandard VRML200x führen soll. Wahrscheinlicher ist aber, dass X3D selbst zum Nachfolgestandard von VRML97 wird [BWG01].
Shout3D und blaxxun3D sind die ersten Implementierungen von X3D. Da diese Formate sich aber nach 1999 von der X3D-Entwicklung abgekoppelt haben und eigene Erweiterungen entwickelten, sind sie im Grunde als proprietär anzusehen.

Danach folgt mit MPEG-4/BIFS ein Format, welches auf VRML97 aufbaut und in eine komplexe Medienarchitektur integriert ist.

3.2.1 VRML97

Mit VRML97 *(Virtual Reality Modeling Language)* [VRML97] können Web3D-Szenen erstellt werden, die statische, animierte und dynamische Objekte enthalten können. VRML97 ist ein gut strukturiertes, plattformunabhängiges 3D-Austauschformat zur Darstellung von dreidimensionalen Welten im WWW.
Die aktuelle Version VRML97 wurde 1997 als ISO/IEC-Standard 14772-1 verabschiedet. VRML97 unterscheidet sich gegenüber VRML 2.0 nur dadurch, dass die Dokumentation von VRML97 ISO/IEC-konform ist und die von VRML 2.0 nicht.

3.2.1.1 Erstellung

Dateiformat
VRML97 ist ein standardisiertes, deklaratives Austauschformat, dessen Zeichencodierung im UTF-8 - Format erfolgt. Es sind die Dateierweiterungen VRML und WRL möglich.

Internes Modell und interne Struktur
Das interne Modell basiert auf einem Szenengraphen. Die interne Struktur ist in die Bereiche Kopf, Szenengraph, Prototypen und Ereignisbehandlung unterteilt. Innerhalb des Szenengraphen können insgesamt 54 verschiedene Knotenarten *(Nodes)* benutzt werden, deren Eigenschaften durch insgesamt 20 verschiedene Feldtypen *(Fields)* bestimmt werden können.

Geometrische Primitive	`Box, Cone, Cylinder, ElevationGrid, Extrusion, IndexedFaceSet, IndexedLineSet, PointSet, Sphere, Text`
Gruppierung	`Anchor, Billboard, Collision, Group, Inline, LOD, Switch, Transform`
Lichtquellen	`DirectionalLight, PointLight, SpotLight`
Erscheinungsbild	`Appearance, Material, MovieTexture`
Sound	`AudioClip, Sound`

Tabelle 3: Beispiele für Knotentypen

Ereignismodell
Sensoren sind die elementaren Knotentypen für die Realisierung von Verhaltensweisen. Sensoren generieren Ereignisse *(Events)*, welche durch Ereignispfade mit anderen Knoten verknüpft werden können und dadurch Änderungen an der Szene hervorrufen.
Sensoren-Knotentypen: `Anchor, Collision, CylinderSensor, PlaneSensor, ProximitySensor, SphereSensor, TimeSensor, TouchSensor, VisibilitySensor`

Jeder Knotentyp definiert Namen und Typen von Ereignissen, die er generieren oder empfangen kann. Durch separate Ereignispfade (`ROUTE`) erfolgt die Verbindung zwischen Ereigniserzeuger und Ereignisempfänger.
Beispiel: `ROUTE NodeName.eventOutName To NodeName.eventInName`

Interpolatoren empfangen und bearbeiteten Ereignisse und generieren daraus neue Ereignisse. Interpolatoren-Knotentypen: `ColorInterpolator`, `CoordinateInterpolator`, `NormalInterpolator`, `OrientationInterpolator`, `PositionInterpolator`, `ScalarInterpolator`

Durch die Verwendung von Ereignispfaden und Interpolatoren können einfache Animationen und Interaktionen realisiert werden.

Integration von Skript- und Programmiersprachen
Für die Skripte können die Sprachen Java und JavaScript benutzt werden. JavaScript-Programme werden direkt in die Datei eingebunden. Die externen Java-Programme werden referenziert. Durch die Verwendung von Java und JavaScript ist es möglich Verhaltensweisen zu realisieren, deren Komplexität nur durch die Mächtigkeit der verwendeten Sprache und der Schnittstelle auf den Szenengraphen begrenzt wird.

Erweiterbarkeit
Prototypen (`PROTO`, `EXTERNPROTO`) ermöglichen die Definition von neuen Knotentypen auf der Basis der bestehenden Knotentypen. Dadurch werden Kapselung und Wiederverwendung ermöglicht.

Medienintegration
Es können Bild-, Audio- und Videodaten in die Szene eingebunden werden. Diese Datei-externen Medien werden innerhalb des Szenengraphen referenziert. Die Anzeige der Szene kann erst dann beginnen, wenn die komplette Szenenbeschreibung übertragen wurde. Das einzelne Medienobjekt wird erst nach vollständiger Übertragung der entsprechenden Daten angezeigt.

Die Integration von Bildern (JPEG, GIF) und Videos (MPEG-1) erfolgt durch die Texturknoten `ImageTexture` und `MovieTexture`. Audioquellen (WAV) werden durch den `Sound`-Knoten eingebunden, bei dem insbesondere Richtung, Intensität und Quelle beschrieben werden können. Die Audio-Ausbreitung erfolgt in die angegebene Richtung in ellipsenähnlicher Form. Die Audiquelle kann entweder durch den `AudioClip`-Knoten oder bei einem MPEG-1 - Video mit Ton durch einen `MovieTexture`-Knoten definiert werden.
Die Textintegration, welche durch den Knoten `Text` realisiert wird, stellt ein großes Problem dar, da die Darstellungsqualität von Text bei den meisten Playern ungenügend ist.

Werkzeuge
VRML97-Dateien können mit einem beliebigen Texteditor erstellt werden. Da dies insbesondere für umfangreichere Szenen sehr umständlich ist, werden semi-professionelle visuelle Modellierungstools wie zum Beispiel Spazz3D [Spazz3D], Internet Space Builder & Internet Scene Assembler [ParallelGraphics] oder Beyond 3D Extreme [Beyond 3D] eingesetzt.
Weiterhin existieren meist Export-Möglichkeiten bei professionellen Modellierungstools wie 3ds max [3ds max] oder Maya [Maya]. Leider ist bei diesen letztgenannten Werkzeugen die Modellierung von Interaktionen stark eingeschränkt oder überhaupt nicht vorhanden. Diese Modellierungstools erlauben meist nur das Definieren von Animationen, welche auch teilweise nach VRML97 exportiert werden können.
Weiterhin gibt es eine große Anzahl von Zusatzwerkzeugen zum Validieren, Bereinigen und Optimieren von VRML97-Dateien. Bekannte Vertreter hierfür sind Chisel [Chisel], NIST VRML97 Parser [NIST VRML97 Parser] und Vorlon [Vorlon].

Beispielszene

```
#VRML V2.0 utf8
Group {
        children [
                DEF SphereSystem Transform {
                        children [
                                DEF Sphere_1 Shape {
                                        appearance Appearance {
                                                texture ImageTexture {
                                                        url "texture.jpg"
                                                }
                                        }
                                        geometry Sphere {}
                                }
                        ]
                }
                DEF Touch TouchSensor {}
                DEF Time TimeSensor {}
                DEF Path PositionInterpolator {
                        key [ 0.0, 0.5, 1.0 ]
                        keyValue [-3 2 0, 0 0 3, -3 2 0,]
                }
        ]
}
ROUTE Touch.touchTime TO Time.set_startTime
ROUTE Time.fraction_changed TO Path.set_fraction
ROUTE Path.value_changed TO SphereSystem.set_translation
```

3.2.1.2 Repräsentation

Repräsentationsformat
Die VRML97-Dateien können mit dem GZIP-Verfahren komprimiert werden, da die meisten Plug-Ins dieses Format akzeptieren. Obwohl dieses Komprimierungsverfahren nicht für VRML97 optimiert ist, erreicht man trotzdem Komprimierungsraten um einen Faktor 5 bis 10.

Softwaretechnologischer Zugriff
Es ist nicht möglich, dass der Server auf die laufende Szene beim Client zugreift. Es gibt aber proprietäre Lösungen für die Realisierung von Multiuser-Applikationen, beispielsweise von blaxxun interactive [blaxxun], mit denen dies möglich ist.

3.2.1.3 Kommunikation

Streaming
Es existiert nur eine eingeschränkte Methode zum dynamischen Nachladen von Szenenbestandteilen. Durch den `Inline`-Knoten ist es möglich die 3D-Szene in mehrere Bestandteile aufzuteilen, die erst bei Bedarf geladen werden. Wie die Realisierung des Nachladens erfolgt, ist in der VRML97-Spezifikation nicht exakt definiert.

3.2.1.4 Präsentation

Player
Clientseitig muss ein Webbrowser - Plug-In für die Darstellung der VRML97-Szene vorhanden sein. Leider gibt es große Unterschiede in dem Verhalten der Plug-Ins, was dazu führen kann, dass eine Szene nur von einem bestimmten Plug-In dargestellt werden kann. Ein solches Verhalten widerspricht grundsätzlich dem Gedanken an eine standardisierte Web3D-Technologie. Es muss also nicht nur das Format exakt spezifiziert werden, sondern auch dessen Interpretation und Anzeige durch den Player.

Die verbreitetsten Plug-Ins sind CosmoPlayer [Cosmo Player], Cortona [Cortona] und blaxxun Contact [blaxxun Contact]. Obwohl diese immer weiter entwickelt werden, besitzen sie dennoch eine Größe zwischen 984 KByte und 3,2 MByte.
Es existieren auch Applets für die Darstellung von VRML97-Applikationen, welche aber meist nicht den gesamten Sprachumfang realisieren und auch nicht die Bedeutung wie die Plug-Ins besitzen.

Softwaretechnologischer Zugriff
Es existiert eine eingeschränkte Schnittstelle *(Browser Script Interface)* zwischen der Szene und dem Webbrowser. Skripte können durch diese einfache Schnittstelle auf den Webbrowser und den Player mit *get*- und *set*-Methoden zugreifen.
Eine Kommunikation zwischen der Szene und der HTML-Seite – beispielsweise mittels JavaScript – ist nicht im VRML97-Standard vorgesehen.
Durch die nichtstandardisierte *External Authoring Interface* (EAI) wird die Kommunikation zwischen einem Applet einer Webseite und einem VRML97 - Plug-In ermöglicht.

Skalierung
Durch die Verwendung des `LOD`-Knotens können Objekte in verschiedenen Detaillierungsstufen eingebunden werden. Entsprechend der aktuellen Performance und dem Abstand des virtuellen Auges vom entsprechenden Objekt können dann verschiedene LODs – die vom Autor selbst erstellt und definiert wurden – vom Player ausgewählt werden. Die Verwendung von LODs in VRML97 stellt eine sehr eingeschränkte Form der Skalierung dar. Besonders problematisch ist, dass alle LODs eines Objektes unabhängig voneinander gespeichert werden, was die zu übertragende Datenmenge erheblich erhöht.

3.2.1.5 Bewertung

Einsatzgebiet
Da VRML das erste Web3D-Format war, wurde es in fast allen Gebieten eingesetzt. Es ist ein allumfassendes und gut strukturiertes Format, mit dem sich theoretisch jede Art von Web3D-Applikationen erstellen lässt. Da sich die proprietären Formate meist auf die Anwendungsbereiche Spiele, Produktpräsentationen und Charakteranimationen konzentrieren und dort detailgenauere, multiuserfähige, Streaming-fähige und skalierbare Lösungen anbieten, werden diese hierfür auch bevorzugt eingesetzt.

Nutzungsbedingungen
Es existieren keinerlei Einschränkungen. Die Dokumentation des VRML97-Standards ist frei verfügbar [VRML97].

Zukunft des Formates
Es gibt Bemühungen innerhalb des Web3D-Konsortiums den VRML97-Standard weiterzuentwickeln. Es existierten Arbeitsgruppen, die an Weiterentwicklungen von VRML97 arbeiten. Die Arbeitsgruppe GeoVRML [GeoVRML] beschäftigt sich mit Möglichkeiten der geografischen Visualisierung und die H-ANIM - Arbeitsgruppe (Humanuid Animation Working Group) [H-ANIM] beschäftigt sich mit der Darstellung von Menschen. Die Arbeitsgruppen Rich Media und Universal Media werden anschließend explizit aufgeführt. Es gibt noch viele weitere Arbeitsgruppen deren letzte Aktivitäten oft Jahre zurückliegen oder die nach einer anfänglichen Arbeit an VRML97 zu X3D [X3D] oder MPEG-4 [MPEG] wechselten. X3D ist die direkte Weiterentwicklung von VRML97, welche letztendlich zu einem VRML200X- oder X3D-Standard führen soll.

Fazit
Für VRML97 spricht die Einfachheit und Flexibilität des Formates, die Standardisierung, die gute Werkzeugunterstützung und die Vielzahl von exzellenten VRML97-Dokumentationen. Mit Hilfe dieses Formates lässt sich fasst jede denkbare Web3D-Applikation erstellen, da sich bei der Entwicklung nicht auf einen einzelnen oder wenige Anwendungsbereiche konzentriert wurde. Die Anpassung an geringe Bandbreiten und an eine geringe Rechenleistung der Web3D-Plattform waren zwar Ziele von VRML97, wurden aber nur sehr eingeschränkt verwirklicht.
VRML hat Maßstäbe für die Entwicklung der Web3D-Technologien gesetzt, aber wenig Akzeptanz und Verbreitung im kommerziellen Bereich gefunden.

[Car97, Klo97, VRML97, Wal01, Scr01]

3.2.1.6 Rich Media Working Group

Die Rich Media - Arbeitsgruppe [RMWG] innerhalb des Web3D-Konsortiums beschäftigt sich mit der Schaffung eines neuen Web3D-Standards für qualitativ hochwertige, interaktive Multimediainhalte. Es sollen 2D- und 3D-Grafik, Video, Bilder und Audio nach einer Übertragung über ein Netzwerk durch Internet-Applikationen, Mobile Endgeräte, Spiele-Konsolen und Set-Top-Boxen in Echtzeit dargestellt werden. Besonderes Interesse gilt dem Erstellungsprozess und der Auslieferung und Synchronisation der Daten.
Die Basis dieser Arbeitsgruppen stellen die Dokumente zu VRML, X3D und MPEG-4 dar. Das Dateiformat soll szenengraphbasiert sein und deklarative und prozedurale Elemente enthalten können. Weiterhin sollen Streaming und Kompression unterstützt werden.
Die Grundlage für das aktuelle *Rich Media 3D Proposal Draft 0.1* bildet das von Sony sehr intensiv betriebene Blendo-Projekt [Blendo], welches sich insbesondere mit interaktivem Fernsehen beschäftigt. Angesichts der Versionsnummer 0.1 wird es noch einige Zeit in Anspruch nehmen, bevor eine genauere Charakterisierung der Ergebnisse der Arbeitsgruppe erfolgen kann.

3.2.1.7 Universal Media Working Group

Die Universal Media - Arbeitsgruppe [UMWG] des Web3D-Konsortiums versucht die Realität von Web3D-Szenen zu verbessern und den Datentransfer zu reduzieren. Dies will man durch das Ablegen von plattformunabhängigen, frei verfügbaren Medienelementen wie Sound, Texturen und 3D-Objekten beim Client erreichen.
Mittels eines *Uniform Resource Name* (URN) - Mechanismus können die Autoren diese Medienelemente in die 3D-Szenen integrieren.
Für Bildtexturen existiert schon eine Bibliothek, die in die Bereiche Materialien, Natur, Stadt und Panorama unterteilt ist.
Für VRML97 existiert eine praktische Lösung des Universal Media - Prinzips. Das Plug-In blaxxun Contact [blaxxun Contact] unterstützt das Verwenden von lokalen Bildtexturen innerhalb von VRML97-Welten. Das Programm Spazz3D [Spazz3D] erlaubt die Erstellung von Szenen, die lokale Bildtexturen verwenden können.

3.2.2 X3D

Die X3D Working Group des Web3D-Konsortiums [Web3D-Konsortium] hat das Ziel das Format E**x**tensible **3D** (X3D) [X3D] zu entwerfen, zu implementieren, zu evaluieren und zu standardisieren. Diese Arbeitsgruppe wird von der neu formierten Browser Working Group und der Source Code Working Group unterstützt.

Die Ziele bei der Entwicklung von X3D sind:

a. Kompatibilität mit VRML97 hinsichtlich bestehender Applikationen, Playern und Werkzeugen
b. Ein – gegenüber VRML97 kompakter – X3D-Kern *(Core X3D)* für grundlegende Funktionalitäten zur Realisierung von Web3D-Applikationen in den Anwendungsbereichen E-Commerce, Unterhaltung und verteiltes Lernen
c. Ein Erweiterungsmechanismus für die Integration neuer Funktionalitäten
   Es werden Komponenten verwendet, um das schlanke *Core-Profile* zu erweitern. Mögliche Erweiterungen stellen beispielsweise die Ergebnisse der Arbeitsgruppen H-ANIM und GeoVRML des Web3D-Konsortiums dar. Der Begriff Komponente hat im Umfeld von X3D eine andere Bedeutung als in der Softwaretechnologie.
d. Die Unterstützung von anderen Codierungen wie beispielsweise XML

Die folgenden Informationen entsprechen dem Stand der X3D-DTD in der Version 0.80 vom 9. Juli 2001 und der FAQ Version 1.17 vom 9. August 2001 [X3D].

3.2.2.1 Erstellung

Dateiformat
X3D ist ein deklaratives Austauschformat, dessen Zeichencodierung im UTF-8 - Format erfolgt. Es wird aber auch an einer binären Codierung der Szenenbeschreibung gearbeitet.

Internes Modell und interne Struktur
Das interne Modell basiert auf einem Szenengraphen. Die interne Struktur des Dateiformates wird durch die X3D-Grammatik bestimmt. Deren Definition erfolgt zur Zeit in Form einer DTD. Für die Zukunft wird aber die Benutzung eines XML-Schemas favorisiert. Aus Kompatibilitätsgründen sollen auch bestehende VRML97-Dokumente ohne Konvertierung übernommen werden können.

Die Basis der X3D-Grammatik stellt das *Core-Profile* mit seinen 26 Knoten dar. Die Eigenschaften der Knoten können durch insgesamt 23 verschiedene Feldtypen *(Fields)* bestimmt werden. Zusätzlich zu den 20 Feldtypen von VRML97 wurden *Booleans*, *Vector3Double* und *Vector3DoubleArray* hinzugefügt.

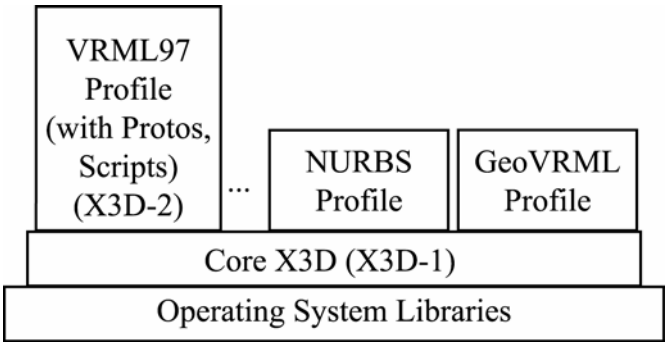

Abbildung 14: Erweiterung des *Core-Profile*

Die Basisfunktionalität des *Core-Profile* kann durch *Profiles* erweitert werden. Diese Erweiterungen erfolgen durch einen oder mehrere X3D-Komponenten, welche beispielsweise aus neuen Knoten mit entsprechend implementierten Funktionalitäten bestehen. Die Komponenten können auch in mehrere, in ihrer Komplexität und Mächtigkeit unterschiedliche Levels eingeteilt werden.
Bei der Einführung von neuen Komponenten wird ein von der OpenGL-Entwicklung bekanntes Verfahren benutzt. Es wird jeder Firma *(Vendor)* ermöglicht eigene Komponenten zu erstellen. Diese können bei hoher Akzeptanz einen *Multi-Vendor* – Status erhalten, der zu einer Aufnahme in eine spätere offizielle Spezifikation führen kann.
Für die vollständige Kompatibilität zu VRML97 steht ein entsprechendes *Profile* zur Verfügung, welches auch die `PROTO`-Funktionalitäten und die Verwendung von Skripten ermöglicht. Deshalb sind bestehende VRML97-konforme Player gleichzeitig auch X3D-2 - Player, da eine XML-Codierung nicht notwendig ist.

Falls man in einer Applikation Streaming-Funktionalitäten benutzen möchte, könnte man sich beispielsweise folgende fiktive Quellcode-Zeile vorstellen:

```
#X3D component:streaming:1
```

Dies teilt dem Player mit, dass man die Streaming-Funktionalitäten des Levels 1 für die Szene benötigt. Diese könnten beispielsweise durch neue Knoten oder durch eine API realisiert werden, die nach einem durch diese Quellzeile initiierten Download zur Verfügung stehen.

Zur Zeit sind folgende *Profiles* in der DTD aufgeführt:

a. VRML97, die Umsetzung von VRML97
b. GeoVRML, zur geografischen Visualisierung
c. H-ANIM, zur Darstellung von Menschen
d. NURBS, Verwendung von NURBS
e. DisJavaVRML, für verteilte interaktive Simulationen
f. LatticeXvl, für die kompakte Übertragung komplexer Geometrien

Durch die niedrige Knotenanzahl des *Core-Profile* und die Komponentenarchitektur können kompakte, erweiterbare und leistungsfähige Player erstellt werden.

Ereignismodell
Hierzu wurden noch keine konkreten Äußerungen bezüglich der Veränderungen zu VRML97 gemacht.

Integration von Skript- und Programmiersprachen
Zur Zeit wird an der X3D - *Scene Authoring Interface* (SAI) gearbeitet. Die X3D-SAI ist eine komfortable API für den Zugriff auf den Szenengraphen. Mittels dieser Schnittstelle werden unter Verwendung des `Script`-Knotens – welcher aber noch in keinem *Profile* enthalten ist – Verhaltensweisen für die Szene definiert. Es können hierfür wahrscheinlich – wie bei VRML97 – die Sprachen Java und JavaScript verwendet werden.

Beispielszene

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "http://www.web3D.org/TaskGroups/x3d/translation/x3d-compact.dtd">

<X3D>
 <Scene>
  <Group>
   <Transform DEF="SphereSystem">
    <Shape DEF="Sphere_1">
     <Appearance>
      <ImageTexture url="texture.jpg"/>
     </Appearance>
     <Sphere/>
    </Shape>
   </Transform>
   <TouchSensor DEF="Touch"/>
   <TimeSensor DEF="Time"/>
   <PositionInterpolator DEF="Path"
        key="0.0, 0.5, 1.0"
        keyValue="-3 2 0, 0 0 3, -3 2 0"/>
  </Group>
  <ROUTE fromField="touchTime" fromNode="Touch"
   toField="set_startTime" toNode="Time"/>
  <ROUTE fromField="fraction_changed" fromNode="Time"
   toField="set_fraction" toNode="Path"/>
  <ROUTE fromField="value_changed" fromNode="Path"
   toField="set_translation" toNode="SphereSystem"/>
 </Scene>
</X3D>
```

Erweiterbarkeit
Zusätzlich zu den Möglichkeiten von VRML97 ist es möglich die Szenenbeschreibungssprache um eigene Funktionalitäten – mittels entsprechender Komponenten und der dazugehörigen Implementation – zu erweitern

Medienintegration
Die Medienintegration entspricht zur Zeit der von VRML97.

Werkzeuge
Da die Codierung der Zeichen im UTF-8 - Format erfolgt und die Struktur durch eine XML-Grammatik definiert sein kann, können einfache Texteditoren oder XML-Editoren verwendet werden.
Der zur Zeit beste Editor ist X3D-Edit, welcher das fehlerfreie Editieren, Erstellen und Validieren von X3D ermöglicht. Dieser Editor ist eine Kombination des visuellen XML-Editors Xeena [Xeena] von IBM und einer, auf der X3D-DTD aufbauenden, Xeena-Konfigurationsdatei.

Aufgrund der Abwärtskompatibilität zu VRML97 können auch die entsprechenden Werkzeuge weiterbenutzt werden.

3.2.2.2 Repräsentation

Zu dieser Problematik wurden noch keine konkreten Aussagen gemacht.

3.2.2.3 Kommunikation

Zu dieser Problematik wurden noch keine konkreten Aussagen gemacht.

3.2.2.4 Präsentation

Player
Die X3D-Player sollen einen schlanken Kern besitzen, der das *Core-Profile* unterstützt. Zusätzliche Funktionalität soll durch nachgeladene Komponenten erreicht werden. Dieser schlanke Kern ist insbesondere bei einer leistungsschwachen Zielplattform – wie beispielsweise einer Set-Top-Box – vorteilhaft, da nur die Funktionen bereitgestellt werden müssen, die auch benötigt werden.

Xj3D [Xj3D] ist ein sich in einer frühen Beta-Phase befindlicher, auf Java3D-basierender, Open Source - Player für X3D. Xj3D ist zur Zeit der einzige X3D-Player.
Im August 2001 wurde die Browser Working Group (BWG) [BWG01] von den Firmen blaxxun [blaxxun], Nexternet [Nexternet], OpenWorlds [OpenWorlds] und ParallelGraphics [ParallelGraphics] gegründet. Die Aufgabe dieser Arbeitsgruppe ist es, die Weiterentwicklung von X3D und entsprechenden Implementationen zu unterstützen. Die primären Ziele dieser Arbeitsgruppe sind:

- die Vermeidung von Player-Inkompatibilitäten durch die enge Zusammenarbeit dieser zukünftigen Player-Hersteller
- die beschleunigte Fertigstellung des *Core-Profile*
- die Entwicklung des Erweiterungsmechanismus

Softwaretechnologischer Zugriff
Durch die X3D-SAI soll die Kommunikation zwischen einem externen Applet innerhalb einer HTML-Seite und der Szene ermöglicht werden.

3.2.2.5 Bewertung

Einsatzgebiet
X3D soll wie VRML97 ein Format sein, mit dem sich theoretisch jede Art von Web3D-Applikation realisieren lässt. Dies wird insbesondere durch die Erweiterungsmechanismen für die Integration neuer Funktionalitäten erreicht.

Zukunft des Formates
Es bedarf noch großer Anstrengungen bis zu einer Standardisierung von X3D. Zur Zeit sind noch sehr viele Fragen ungeklärt. Insbesondere sind Fragen zur Geometrie- und Texturbehandlung und zu Streaming, Skalierung und Kompression noch völlig offen.
Das Format wird wahrscheinlich in MPEG-4 integriert werden und dort BIFS ersetzten, wodurch X3D sehr stark an Bedeutung gewinnen wird.

Fazit
Bisher liegen nur einige vorläufige Grammatiken, die sich in Entwicklung befindliche SAI und der Xj3D-Player vor. Nach der Gründung der BWG wurde die Planung für die Weiterentwicklung von X3D stark verändert. Die neuerdings übertriebenen Bestrebungen bezüglich der Kompatibilität zu VRML97 äußern sich darin, dass man die XML-Codierung nicht mehr favorisiert und bestehende VRML97-Player als X3D-2 - Player bezeichnet. Durch die intensiven Bestrebungen der BWG soll die Weiterentwicklung von X3D stark beschleunigt werden, was zu einer baldigen Verfügbarkeit entsprechender Implementationen führen soll. Es besteht aber hierbei die Gefahr, dass durch eine übertriebene VRML97-Kompatibilität kein wirklich neues Format entsteht, welches dann möglicherweise die Nachteile des Vorgängerformates behält. Diese Kompromisse müssen wahrscheinlich eingegangen werden, da die X3D Working Group und das Entwicklungsteam von Xj3D nicht genügend Zeit und Ressourcen für die Realisierung eines selbstständigen Standards haben. Durch die beschleunigte Entwicklung von X3D durch die BWG und die Integration in MPEG-4, wird das Format in Zukunft eine große Bedeutung erlangen.

[X3D, Web3D-Konsortium, Wal01, Scr01, Web3D-MPEG]

3.2.3 Shout3D und blaxxun3D

In diesem Abschnitt werden zwei Formate – Shout3D [Shout3D] von Shout Interactive [Shout Interactive] und blaxxun3D [blaxxun3D] von blaxxun interactive [blaxxun] – gleichzeitig behandelt, da beide erste X3D-Implementierungen darstellen und es nur relativ wenige Unterschiede gibt.
Die grundlegende Philosophie besteht darin, dass beim Client ein Java-Applet innerhalb der Webseite angezeigt wird, welches eine eigene Rendering-Engine enthält. Dieses Applet interpretiert die Szene und zeigt sie an.
Die *Shout3D 2.0 Specification* ist abgeleitet von *Shout Interactive's Core X3D Specification* von 1999 [X3D Shout]. blaxxun3D basiert auf der *Core X3D Specification* von blaxxun interactive [X3D blaxxun], welche ebenfalls 1999 erstellt wurde. Die Firma OpenWorlds [OpenWorlds] bietet auch einen ähnlich gearteten Player an, der aber noch in keinen Webbrowser integriert werden kann und somit nicht behandelt wird.

3.2.3.1 Erstellung

Dateiformat
Beide Formate sind erweiterte und somit veränderte Untermengen von VRML97. Sie sind als Zwischenschritte auf dem Weg zu X3D anzusehen.

Shout3D
Das Java-Applet kann Szenen, die im Shout3D-Format (*.S3D) vorliegen, anzeigen. Der Shout3D-Wizard importiert S3D- und VRML97-Dateien und generiert eine Webseitenvorlage mit entsprechendem Applet für die Publikation der Szene. Die importierten VRML97-Dateien werden analysiert und gegebenenfalls entsprechend der Konventionen in das Shout3D-Format konvertiert. Weiterhin erfolgt durch den Wizard standardmäßig eine Komprimierung des *.S3D-Formates in das *.S3Z-Format.

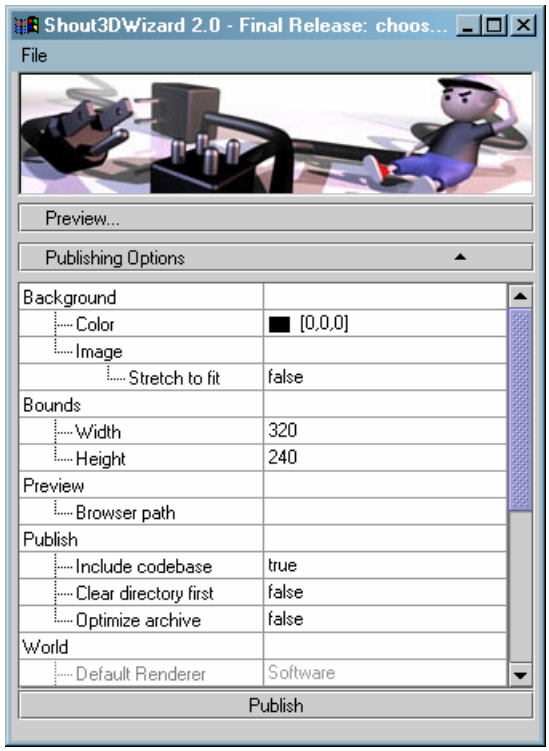

Abbildung 15: Shout3D Wizard

blaxxun3D
Das Java-Applet kann Szenen, die im VRML97-Format (*.WRL) vorliegen und den Konventionen von blaxxun3D entsprechen, anzeigen. Der blaxxun3D-Wizard importiert WRL-Dateien und generiert auch eine Webseitenvorlage mit entsprechendem Applet. Durch den blaxxun3D-Wizard ist es möglich *.BX3D-Dateien zu erstellen, die die Szenenbeschreibung und die referenzierten Medien in komprimierter Form enthalten. Es ist auch eine optionale Komprimierung der Szenenbeschreibung mit GZIP möglich.

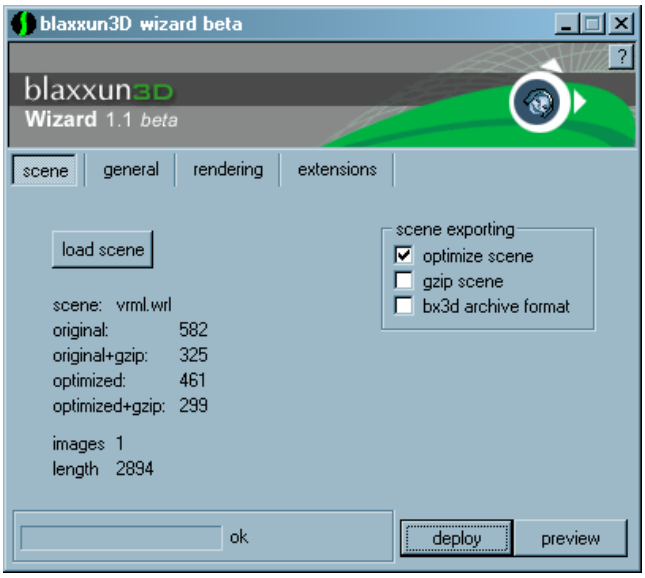

Abbildung 16: blaxxun3D Wizard

Beide Wizards geben Hinweise, welche Bestandteile der Szenenbeschreibung nicht konvertiert werden konnten und somit nicht durch das Applet interpretiert werden.

Internes Modell und interne Struktur
Shout Interactive und blaxxun interactive waren bzw. sind an der Entwicklung von X3D in der entsprechenden Arbeitsgruppe des Web3D-Konsortiums beteiligt. Die beiden Technologien entsprechen nicht dem aktuellen Stand der X3D-Entwicklung. Eine Beschreibung der Szene durch XML ist nicht möglich. Die vollständige Abwärtskompatibilität zu VRML97 wurde nicht gewahrt. Die derzeitige Komponentenphilosophie von X3D ist nur sehr unzureichend realisiert. Es erfolgt keine Unterteilung in *Profiles*.
Die Applets zur Anzeige der Szene können durch den Autor beliebig erweitert werden.

In beiden Formaten dürfen im Gegensatz zu VRML97 keine Skripte und keine PROTO-*Statements* verwendet werden.

Shout3D
Folgende VRML97-Knoten werden durch das Applet nicht interpretiert: `AudioClip`, `Collision`, `Cylinder`, `Fog`, `FontStyle`, `Inline`, `LOD`, `MovieTexture`, `Normal`, `NormalInterpolator`, `PixelTexture`, `PlaneSensor`, `PointLight`, `ProximitySensor`, `Script`, `Sound`, `SphereSensor`, `SpotLight`, `Text`, `TextureTransform` und `VisibilitySensor`. Es gibt keine speziellen *EventIn*- und *EventOut*-Felder. Es können alle Felder für Ereignisketten benutzt werden. Weiterhin muss beachtet werden, dass Veränderungen bei der Definition des Erscheinungsbildes von Objekten vorgenommen wurden. In den übernommenen Knoten wurden teilweise auch Felder entfernt.
Diese Untermenge von VRML97 wurde nun durch neue Felder in den übriggebliebenen Knoten und durch neue Knoten erweitert. Es existieren insgesamt 86 Knoten und 12 Feldtypen in Shout3D.

Die Anzeige der Szene erfolgt durch ein Applet. Es muss ein Objekt der `Shout3Dapplet`-Klasse existieren, das als Container für ein Objekt der `Shout3Dpanel`-Klasse fungiert. Die direkte Anzeige der Szene wird durch das `Shout3Dpanel`-Objekt realisiert. Beide Klassen können beliebig erweitert werden, da alle Methoden `public` sind. Der unlogische Sachverhalt, dass ein `Shout3DPanel`-Objekt immer in einem `Shout3DApplet`-Objekt enthalten sein muss, resultiert daraus, dass der Programmierer nur Applets erstellen soll.

Es steht eine Java-API zur Verfügung, die insbesondere einen komfortablen Zugriff auf den Szenengraphen mit seinen Knoten und Feldern (`Core`-*Package)* ermöglicht. Weiterhin existieren die *Packages* `hanim`, `math` und `sound`.

blaxxun3D
Folgende VRML97-Knoten werden durch das Applet nicht interpretiert: `Box`, `Collision`, `Color`, `Cone`, `Cylinder`, `CylinderSensor`, `ElevationGrid`, `Extrusion`, `Fog`, `FontStyle`, `MovieTexture`, `Normal`, `NormalInterpolator`, `PixelTexture`, `PlaneSensor`, `PointLight`, `Script`, `Sphere`, `SpereSensor`, `SpotLight`, `Text` und `TextureTransform`.
Es werden insgesamt 32 Knoten unterstützt. Bis auf `SFImage` wurden alle Feldtypen von VRML97 übernommen. Insbesondere nahm man Änderungen bei der Beleuchtung und der Texturbehandlung vor. Es erfolgt keine Unterscheidung bei den Feldtypen zwischen *eventIn* und *eventOut*.

Ereignismodell
Die `Route`-Funktionalität wurde beibehalten. Die Ereignisbehandlung kann durch Java, aber nicht durch JavaScript erfolgen.

Integration von Skript- und Programmiersprachen
Die Verwendung von JavaScript innerhalb des Szenengraphen ist nicht möglich. Dieses ist aber kein Problem, da diese Funktionalitäten auch mittels der entsprechenden Java-API realisiert werden können.

Erweiterbarkeit
Die Erweiterungsmöglichkeiten sind nur durch den Sprachumfang von Java und die existierenden Java-Klassen begrenzt.

Medienintegration
Möglichkeiten für die Einbindung von Videotexturen sind in den Formaten nicht vorhanden. Die Behandlung von Bild-Texturen wurde von VRML97 übernommen.

Shout3D
Es existiert ein sehr gutes Texturhandling. Beispielsweise wird das *Bump-Mapping* ermöglicht, wodurch Prägestrukturen realisiert werden können. Es existiert ein *Package* `shout3d.sound`, welches eine rudimentäre Realisierung von Sound ermöglicht.

blaxxun3D
Im Gegensatz zu Shout3D wurden die kompletten Soundfunktionalitäten aus VRML97 übernommen

Werkzeuge
Shout3D
Wichtige Werkzeuge für die Erstellung von Shout3D-Applikationen sind 3ds max [Discreet], Character Studio [Discreet] und Spazz3D [Spazz3D], da entsprechende Exporter für das *.S3D-Format existieren. Charakter-Animationen, die in Character Studio erstellt wurden, können auf diese Weise übernommen werden.

blaxxun3D
Es existiert ein Exporter für 3ds max [Discreet] für das blaxxun3D-Format.

Weiterhin können alle VRML97-Werkzeuge verwendet werden. Es muss beachtet werden, dass die VRML97-Dateien für eine Anzeige durch das Shout3D- oder blaxxun3D-Applet oft stark angepasst werden müssen.

3.2.3.2 Repräsentation

Hierbei gibt es keine Unterschiede zu VRML97.

3.2.3.3 Kommunikation

Hierbei gibt es keine Unterschiede zu VRML97.

3.2.3.4 Präsentation

Player
Die Präsentation der Szene erfolgt durch ein Java-Applet, das durch das Applet-Tag in die entsprechende Webseite eingebunden wird. Der Webbrowser muss die Java-Version 1.1 unterstützen.
Durch die Attribute des Applet-Tags können dem Applet verschiedene vordefinierte Parameter übergeben werden. Hierdurch wird beispielsweise die Größe der Szene oder die Hintergrundfarbe bestimmt.

Softwaretechnologischer Zugriff
Hierbei stehen nahezu unbegrenzte Möglichkeiten zur Verfügung, da eine bidirektionale Kommunikation zwischen JavaScript innerhalb der Webseite und dem Shout3D - oder blaxxun3D - Java-Applet möglich ist. Der Zugriff auf die Szene wird durch eine Erweiterung der EAI von VRML97 ermöglicht.
Weiterhin ist die Einbindung eines weiteren Java-Applets in die Webseite möglich, welches mit dem Applet, das die Szene anzeigt, kommuniziert.

3.2.3.5 Bewertung

Einsatzgebiet
Shout3D
Es können Oberflächen gestaltet werden, die die Umgebung reflektieren. Dies ist besonders für den E-Commerce - Bereich interessant. Die realistische Darstellung von Gesichtern ist durch die *Morphing*-Technologie möglich, die insbesondere für den Import entsprechender Animationen aus 3ds max wichtig ist. Somit ist Shout3D auch für die Realisierung von Charakteranimationen geeignet.

blaxxun3D
Blaxxun3D besitzt nur einen Projekt-Status und wurde bisher sehr wenig für kommerzielle Zwecke eingesetzt.

Es ist immer zu beachten, dass durch das Applet nur relativ einfache Szenen in Echtzeit dargestellt werden können. Ein Applet hat aber den Vorteil, dass kein Plug-In installiert werden muss. Somit sind die beiden Technologien insbesondere für Anwendungsbereiche geeignet, wo die Installation eines Plug-Ins unerwünscht ist. Dies könnte insbesondere bei Produktpräsentationen der Fall sein.

Nutzungsbedingungen
Shout3D
Die komplette Software von Shout3D steht kostenlos zum Download bereit. Es existiert ein frei verfügbares Applet mit dem beliebige Inhalte publiziert werden können. Bei diesem wird aber immer ein verlinktes Shout3D-Logo angezeigt. Dies ist nach dem Erwerb einer entsprechenden Lizenz nicht vorhanden. Eine Lizenz bezieht sich immer auf die URL der entsprechenden Webseite, in dem das Applet enthalten ist. Es erfolgt eine Preisstaffelung entsprechend der Anzahl der Webseiten, die ein Shout3D-Applet enthalten.

blaxxun3D
Das Format blaxxun3D ist ein nichtkommerzielles Produkt, die komplette Software ist kostenlos erhältlich. Es existieren abgesehen vom Start des Applets keine firmenspezifischen Hinweise oder die Darstellung eines Logos während der Präsentation der Szene.

Zukunft des Formates
Beide Formate beziehen sich auf den Stand der X3D-Entwicklung von 1999. Es ist zu bezweifeln, dass eine X3D-konforme Aktualisierung der Formate vor Beendigung der X3D-Entwicklung erfolgt.

Fazit
Beide Formate sind nur sehr wenig verbreitet. Shout3D bietet gegenüber blaxxun3D eine umfangreichere API, die beispielsweise ein verbessertes Texturhandling und Charakter-animationen ermöglicht. Falls man dies mit blaxxun3D realisieren wollte, müsste man die entsprechenden Java-Klassen selbst erstellen. Weiterhin ist Shout3D besser dokumentiert und scheint performanter und detailgenauer als blaxxun3D zu sein. Shout3D und blaxxun3D sind die ersten Realisierungen von X3D. Der Autor kann einerseits das Applet entsprechend seinen Bedürfnissen anpassen, andererseits ist hierfür aber ein umfangreiches Expertenwissen erforderlich. Da die beiden Formate nicht der zukünftigen X3D-Spezifikation genügen werden, sind sie zum jetzigen Zeitpunkt als proprietär anzusehen. Es ist zu hoffen, dass nach der Standardisierung von X3D eine Hinwendung zu diesem Format erfolgt.

[Shout3D, blaxxun3D]

3.2.4 MPEG-4/BIFS

MPEG-4 [MPEG01a] wurde von der Moving Picture Experts Group (MPEG) [MPEG] entwickelt und 1999 standardisiert (ISO/IEC 14496). Es ist ein System für die Erstellung, Übertragung und Betrachtung von komplexen interaktiven multimedialen Szenen bei geringen Bitraten. MPEG-4 ist der erste Standard, der von dieser Arbeitsgruppe entwickelt wurde, bei dem die einzelnen Bestandteile einer Szene unabhängig von den anderen präsentiert, manipuliert und transportiert werden. Diese als audiovisuelle Objekte bezeichneten Bestandteile sind Szenenbeschreibungs-, Audio-, Video-, und Kontrolldaten die entweder durch ein Recording-Verfahren erzeugt oder künstlich generiert werden. Im Gegensatz zu MPEG-1 und MPEG-2, die nur die Präsentation von vorgefertigten Inhalten ermöglichen, kann MPEG-4 clientseitig dynamisch generierte audiovisuelle Objekte realisieren. Weiterhin ist eine Interaktion des Nutzers mit der Szene möglich.

MPEG-4 kann aber auch auf Distributionsmedien wie CD-ROM und DVD gespeichert werden oder durch Broadcast-Systeme übertragen werden.

Die Kombination der einzelnen audiovisuellen Objekte in einer Szene wird durch die, auf VRML97 aufbauende, Szenenbeschreibungssprache BIFS beschrieben. Es werden nur die Änderungen bezüglich VRML97 aufgeführt. Visuelle Objekte können pixelbasierend sein, oder aus 2D- und 3D-Geometrien bestehen.

Der MPEG-4 - Standard ist in die sechs Teile *Systems, Visual, Audio, Conformance Testing, Reference Software* und *Delivery Multimedia Integration Framework* (DMIF) untergliedert. Weiterhin erfolgt eine Unterteilung in die Profile *Visual, Audio, Graphics, Scene Graph, MPEG-J* und *Objekt Descriptor*, welche wiederum in verschiedene *Levels* unterteilt sind. Alle in dieser Arbeit erwähnten Informationen zu MPEG-4 beziehen die Erweiterungen von Version 2 mit ein und geben die maximalen Möglichkeiten der Web3D-Funktionalitäten des Standards wieder. Informationen über Funktionalitäten, die sich nicht auf den 3D-Bereich von MPEG-4 beziehen, werden nicht beschrieben.

3.2.4.1 Erstellung

Dateiformat
Das Dateiformat *Binary Format for Scenes* (BIFS) ist sehr stark an VRML97 orientiert und verfügt fast über die gesamte Funktionalität dieses Standards.
Die Codierung erfolgt im UTF-8 - Format, welches aber serverseitig vor der Auslieferung in ein Binärformat umgewandelt wird.

Internes Modell und interne Struktur
Das interne Modell und die interne Struktur entsprechen dem VRML97-Standard. Es existieren im Gegensatz zu den 54 Knoten von VRML97 in BIFS fast 100 Knoten.

Es existieren zusätzlich Knoten für die Darstellung von zweidimensionalen Objekten. Dadurch können reine 2D-Szenen und Kombinationen von 2D- und 3D-Objekten in einer Szene erstellt werden. Die neuen 2D-Knoten sind meist VRML97-Knoten, bei denen eine Dimension fehlt.

Neue 2D-Knoten: `Background2D`, `Circle`, `CoordinateInterpolator2D`, `Curve2D`, `IndexedFaceSet2D`, `IndexedLineSet2D`, `Material2D`, `PlaneSenor2D`, `PointSet2D`, `PositionInterpolator2D`, `ProximitySensor2D`, `Rectangle`, `Sound2D` und `Transform2D`.

Die beiden Knoten `Layout` und `Form` ermöglichen die kontrollierte Anordnung von 2D-Objekten.

Es existieren Möglichkeiten für die Darstellung von Gesicht und Körper eines Menschen. Durch die *Face Definition Parameter* (FDP) und *Body Definition Parameter* (BDP) können eigene Gesichts- oder Körpermodelle definiert werden. Ansonsten wird das Standardmodell des Players benutzt. Für die Beschreibung des Gesichtes existieren 66 Parameter und für die Beschreibung des Körpers 260 Parameter. Für die Animation des Gesichts stehen *Facial Animation Parameter* (FAP) und für die Animation des Körpers *Body Animation Parameter* (BAP) zur Verfügung. Diese Animationsparameter basieren auf den natürlichen Bewegungsmöglichkeiten des Menschen. Diese Funktionalitäten wurden in Zusammenarbeit mit der H-Anim - Arbeitsgruppe des Web3D-Konsortiums bereitgestellt.

Integration von Skript- und Programmiersprachen
Für die Skripte kann nur die Sprache JavaScript benutzt werden. JavaScript wird – wie in VRML97 – direkt in die Datei eingebunden.

Erweiterbarkeit
Die Möglichkeit zur Definition von Prototypen – wie in VRML97 – wurde nicht übernommen.

Medienintegration
Die Kombination von 2D- und 3D-Objekten kann folgendermaßen erreicht werden:

- Zeichnen der 2D-Objekte in eine lokale Ebene der 3D-Szene
- Durch die Knoten `Layer2D` und `Layer3D` können Ebenen definiert werden, in denen 2D- oder 3D-Unterszenen eingebunden werden können. Dadurch ist es möglich ein 3D-Logo in ein 2D-Video oder eine 2D-Benutzungsschnittstelle in eine 3D-Szene einzubinden.
- Mit Hilfe der Knoten `CompositeTexture2D` und `CompositeTexture3D` ist es möglich 2D- und 3D-Szenen als Texturen zu realisieren.

In VRML97 werden nur die Objektgeometrien und der Blickwinkel für die Berechnung von Sound berücksichtigt. MPEG-4 erweitert dies durch die Beschreibung der physikalischen Oberflächeneigenschaften von Materialien, Echo, Doppler-Effekt, weitere Parameter für die Berechnung des Umgebungssounds und die Möglichkeit zum Mischen von Sound. Bei der Darstellung von Menschen sind die Funktionalitäten zur Lippensynchronisation von besonderer Bedeutung. Es existieren die neuen Audio-Knoten `AudioBuffer`, `AudioDelay`, `AudioFX`, `AudioMix` und `AudioSwitch`.

Werkzeuge
Die MPEG-4 - Datei kann mit einem Texteditor erstellt werden, da die Umwandlung in ein Binärformat erst später erfolgt.
Es existiert ein sehr einfacher visueller MPEG-4 - Editor MPEG-4 ToolBox [M4TB], der aber noch nicht für den praktischen Einsatz geeignet ist.

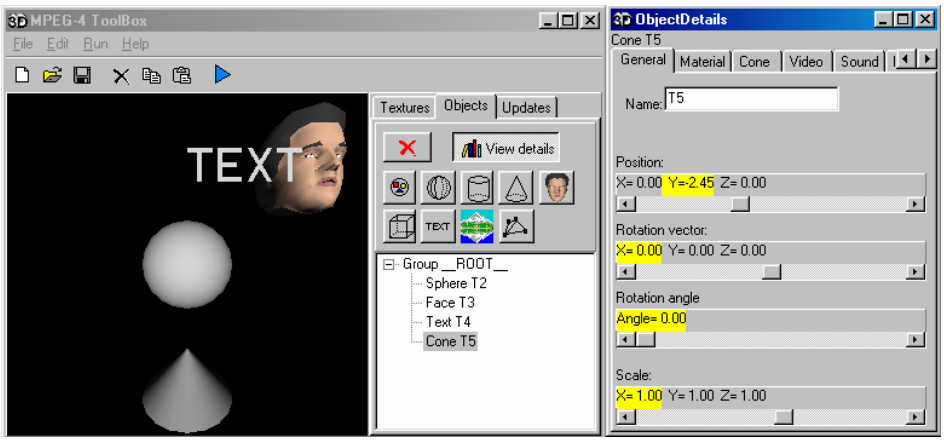

Abbildung 17: MPEG-4 ToolBox

Es existiert weiterhin eine Sammlung von MPEG-4 Tools von der Telekom Paris. [M4To]

Beispielszene
Da kein Standard, keine Knotenbeschreibung, keine Werkzeuge und kein Player vorliegen, kann hier leider kein Beispiel aufgeführt werden.

3.2.4.2 Repräsentation

Repräsentationsformat
Die audiovisuellen Objekte werden durch die im Gliederungspunkt 3.2.4.3 „Kommunikation“ erwähnte *Compression Layer* komprimiert. Ob die Daten in komprimierter Form vorliegen, oder erst bei Bedarf komprimiert werden, ist nicht bestimmt. Wahrscheinlich wird man bestimmte Daten vorkomprimieren und andere nicht.
Die Komprimierung von visuellen Objekten ist einerseits durch pixelbasierende Verfahren und andererseits durch Geometriekompression möglich. Durch die Verwendung von Geometriekompression soll ein Kompressionsfaktor 30 erreicht werden.
Es existiert die Möglichkeit hierarchische 3D-Netze zu spezifizieren. Dadurch ist es möglich sukzessiv 3D-Netze an den Client zu übertragen. Weitere Kompression ist durch Quantisierung der Geometrie möglich.
Durch Sprachkomprimierungsalgorithmen sollen natürliche Sprache durch Datenströme von 2 KBit/s bis 24 KBit/s und synthetische Sprache durch Datenströme von 200 Bit/s bis 1,2 KBit/s repräsentiert werden können. Für die Komprimierung von natürlich erzeugten Audiodaten stehen Datenströme von 2 bis 64 KBit/s zur Verfügung.
Die Komprimierung von audiovisuellen Objekten kann in verschiedenen Qualitätsstufen erfolgen.
Es existieren weiterhin spezielle Kompressionsverfahren für das Komprimieren von Texturen.

Softwaretechnologischer Zugriff
Das BIFS-Befehlsprotokoll ermöglicht eine Kommunikation zwischen Server und Client, auf dem die MPEG-4 - Anwendung ausgeführt wird. Der Server kann die komplette Szene ersetzen oder bestimmte Knoten, Felder und `ROUTE`-Anweisungen hinzufügen, löschen oder ersetzen. Hierdurch können beispielsweise ein Avatar geändert werden oder bei komplexen Animationen die Werte der Felder `key` und `keyvalue` sukzessiv ergänzt werden.

Das BIFS-Anim - Protokoll wird für die kontinuierliche Animation der Szene benutzt. Dadurch können beispielsweise Transformationen, Blickpunkte, Farben oder Licht animiert werden.

3.2.4.3 Kommunikation

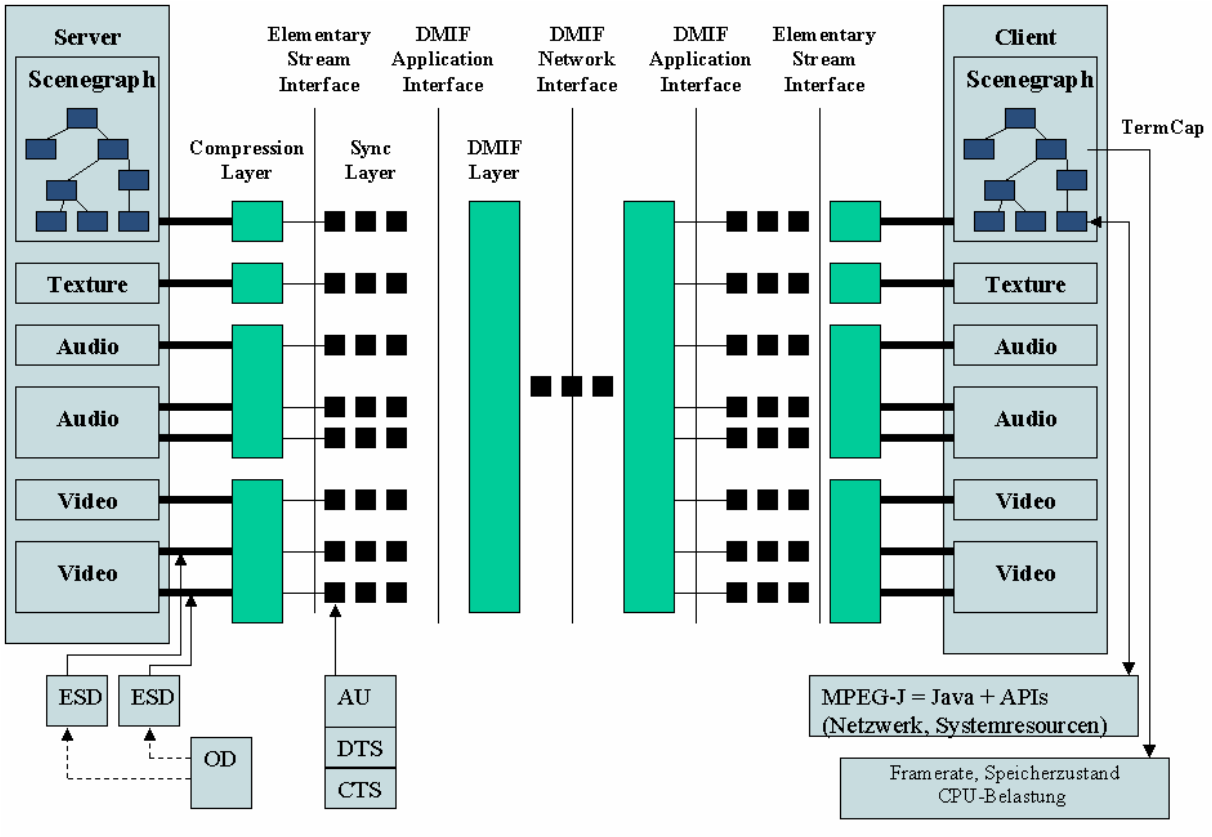

Abbildung 18: MPEG-4 Architektur

Die Übertragung der Szene erfolgt durch die Anwendung der Kompressions-, Synchronisations- und DMIF-Schicht. Das *Delivery Multimedia Integration Framework* (DMIF) ist ein Session-fähiges Protokoll für das Streaming von multimedialen Inhalten.

Die audiovisuellen Objekte, die Szenenbeschreibung und die Kontrollinformationen werden in der *Compression Layer* komprimiert, in ein Binärformat umgewandelt und anschließend als *Streams* bezeichnet. Ein einzelnes audiovisuelles Objekt kann in mehrere *Streams* zerlegt werden.
Jeder einzelne *Stream* wird über einen *Elementary Stream* - Deskriptor (ESD) referenziert. Es wird ein Objekt-Deskriptor (OD) angelegt, in dem die zu einem audiovisuellen Objekt gehörenden *Streams* referenziert werden. Im OD wird weiterhin vermerkt wie die Decodierung zu erfolgen hat und in welchem Format das Objekt vorliegt.
Es existieren verschiedene Encoder für die unterschiedlichen *Streams*. Die Encoder für die Video- und Audiokomprimierung sind in den Teilen *Visual* und *Audio* des Standards beschrieben. BIFS wird in dieser Schicht in ein Binärformat umgewandelt. Dadurch wird eine Komprimierung der Szenenbeschreibung um einen Faktor 10 bis 15 erreicht.

Anschließend erfolgt in der *Snychronisation Layer* das Aufteilen eines *Streams* in einzelne atomare Einheiten, die als *Access Unit* (AU) bezeichnet werden. Durch den *Decoding Timestamp* (DTS) wird der Decodierungszeitpunkt und durch den *Composition Timestamp* (CTS) wird der Präsentationszeitpunkt einer AU bestimmt. Die in der AU enthaltenen Zeitstempel sind für eine spätere Synchronisation von elementarer Bedeutung.

Durch Multiplexing werden anschließend die verschiedenen Ströme zu einem einzelnen Strom zusammengefasst und zum Client gesendet. Da MPEG-4 keine eigene Transportschicht definiert, muss auf RTP, TCP, MPEG-2 Transport System, H-323 oder ATM zurückgegriffen werden. [MPEG01a]
Anschließend erfolgt das Demultiplexing, die Dekompression und die Synchronisation der Daten. Nun kann mittels der vorliegenden Daten das Rendering beginnen.

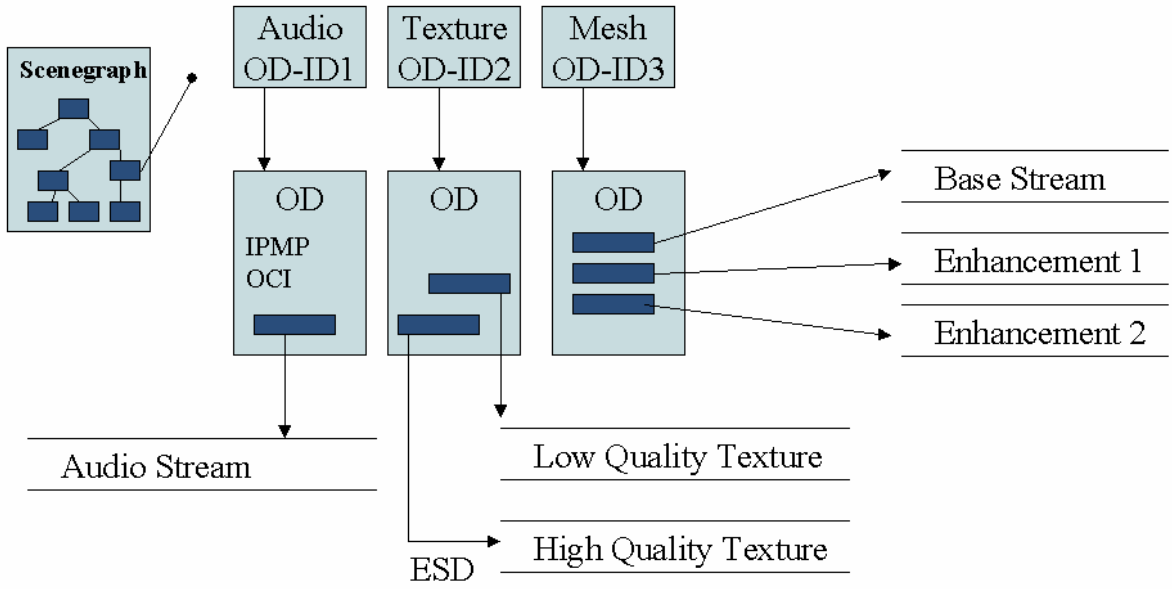

Abbildung 19: Relation zwischen Szenengraph und den *Streams*

Der Zusammenhang zwischen Szenenbeschreibung und Objektdeskriptor (OD) wird durch die Objektdeskriptor-ID (OD-ID) erreicht. Im Szenengraphen werden die externen Referenzen nicht durch URL-Angaben, sondern durch OD-IDs realisiert. Eine OD-ID zeigt auf einen OD, welcher über die *Elementary Stream* - IDs Zugriff auf die einzelnen *Elemetary Stream* - Deskriptoren (ESD) und somit auf den eigentlichen *Stream* hat. Ein OD kann einen einzelnen *Stream*, einen *Stream* in verschiedenen Qualitätsstufen oder mehrere auf sich aufbauende *Streams* referenzieren. Dadurch können beispielsweise auch hierarchische 3D-Netze sukzessiv übertragen werden.
In einem OD sind weiterhin Angaben zum Rechtemanagement *(Intellectual Property Management and Protection, IPMP)* und textuelle Informationen *(Object Content Information, OCI)* zu den einzelnen audiovisuellen Objekten enthalten.

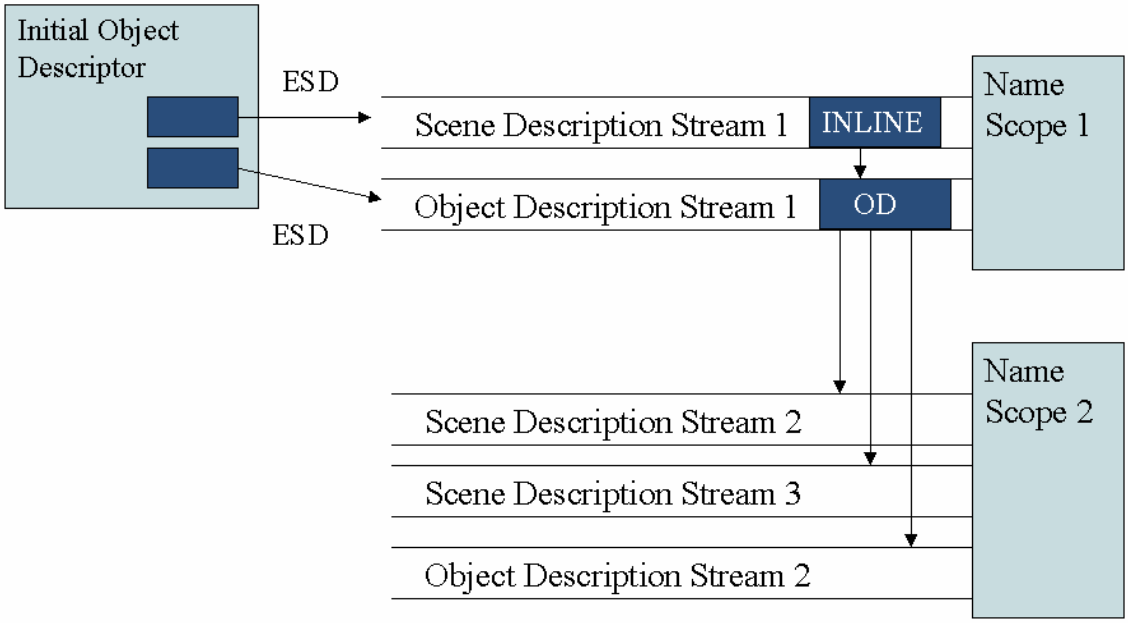

Abbildung 20: Streaming von Szenenbeschreibung und Objektdeskriptoren

Bei der initialen Anforderung einer Szene durch den Client, wird zuerst ein *Stream* mit der Szenenbeschreibung und den darin referenzierten ODs übertragen. Die initiale Szenenbeschreibung muss nicht vollständig sein. Es kann beispielsweise über einen `Inline`-Knoten und einem entsprechenden OD auf weitere Szenenbeschreibungen und ODs verwiesen werden. In einem ES-Deskriptor sind möglicherweise URL-Angaben vorhanden, durch die auf verteilte *Streams* zugegriffen werden kann.

Skalierung
Die Komprimierungsalgorithmen sind durch die DMIF-Layer parametrisierbar und lassen sich somit der aktuell möglichen Datentransferrate anpassen.

Streaming
Es gibt zwei Möglichkeiten für die Übertragung einer Szene zum Client. Einerseits kann die gesamte Szene übertragen werden und andererseits kann erst die Übertragung einer detailarmen Szene erfolgen, die aber additiv verbessert wird.
Durch das BIFS-Befehlsprotokoll und das BIFS-Anim - Protokoll kann auch eine sukzessive Übertragung von Szenendaten erfolgen.

3.2.4.4 Präsentation

Player
Es existieren schon einige Player, welche aber nur das Abspielen von Audio- und Videoinformationen erlauben. Die Realisierung von Funktionalitäten zur Darstellung von interaktiver 3D-Grafik in einem Player ist noch nicht erfolgt.

Skalierung
Es ist möglich, dass der Player selbst vereinfachte Objektgeometrien erzeugt, um somit die Echtzeitfähigkeit der Präsentation aufrecht zu halten.
Durch den `TermCap`-Knoten ist es möglich auf Daten wie die aktuelle Framerate, den Speicherzustand und die CPU-Belastung zuzugreifen und entsprechend darauf zu reagieren.

Bilder und Texturen können in maximal elf und Videos in drei Auflösungsstufen durch den Decoder bereitgestellt werden. Bei Audiodaten ist auch eine Parametrisierung der Kompression möglich.

Softwaretechnologischer Zugriff
MPEG-J ist eine Erweiterung der EAI von VRML97. Es existiert eine Java-API, durch die auf den MPEG-4 - Player und die darin enthaltene Szene Einfluss genommen werden kann. Hierdurch werden das Hinzufügen, Löschen und Ändern von Knoten und Feldern des Szenengraphen und der Zugriff auf verschiedene Systemressourcen, das Netzwerk und die Decoder ermöglicht.
Der Java-Code wird in einem eigenen *Stream* übertragen und innerhalb der MPEG-J - Laufzeitumgebung ausgeführt.

3.2.4.5 Bewertung

Einsatzgebiet
MPEG-4 wurde für den Einsatz im digitalen Fernsehen, für interaktive Grafikapplikationen mit synthetischem Inhalt und für interaktive multimediale Anwendungen im WWW konzipiert.

Verbreitung
Es fehlt zur Zeit noch an der praktischen Umsetzung der 3D-Funktionalitäten des MPEG-4 - Standards.

Nutzungsbedingungen
Der MPEG-4 Standard ISO/IEC 14496 ist bei der ISO käuflich erwerbbar. Die Referenzsoftware [RefSW] für MPEG-4 Version 1 ist durch Patente geschützt. Sie unterliegt keinen Copyright-Restriktionen wenn sie für die Implementierung von MPEG-4 - fähiger Technologie benutzt wird. Die Dokumentation des kompletten Standards kostet ca. 1100 US$.

Zukunft des Formates
Die Weiterentwicklung von MPEG-4 soll zu den Versionen 3 und 4 führen, wobei aber schon an MPEG-7 und MPEG-21 gearbeitet wird.

Das MPEG-4 Industry Forum [M4IF] arbeitet daran, dass MPEG-4 ein akzeptierter und oft benutzter Standard wird.

Die im Dezember 2000 gegründete Internet Streaming Media Alliance (ISMA), an der Cisco Systems, Kasenna, Philips und Sun Microsystems beteiligt sind, arbeitet an der praktischen Umsetzung von MPEG-4 im Hinblick auf das Internet. [ISMA]

Das OpenDivX Streaming - Projekt arbeitet an der Entwicklung von Streaming-Technologien und Anwendungen für MPEG-4. [OpenDivX]

Das Lab for Advanced Networking [LAN] in Vancouver arbeitet an der Implementation eines MPEG-4 Streaming Systems.

Das Prometheus-Projekt [Prometheus] der britischen Medienanstalt BBC, beschäftigt sich mit der Erstellung, Verbreitung und Darstellung von interaktiver 3D-Grafik mittels MPEG-4 im TV-Bereich.

Man arbeitet weiterhin an einer XML-Beschreibung von BIFS, dem *Extensible MPEG-4 Textual Format* (XMT) [XMT].

Fazit
MPEG-4 ist ein komplizierter und umfangreicher Standard, dessen Dokumentation leider nicht frei verfügbar ist. Zur Zeit existieren noch keine Implementierungen für die Realisierung von interaktiver 3D-Grafik mittels BIFS.
Der MPEG-4 Standard bietet umfassende Lösungen für die effiziente und skalierbare Übertragung der Objektdaten über das Internet und für die Skalierung der Szene beim Client. MPEG-4 enthält viele neue, leistungsfähige Konzepte und Ideen. Leider fehlt es zur Zeit noch an deren praktischer Umsetzung. Die bestehenden Implementierungen konzentrieren sich meist auf den reinen Audio- und Videobereich von MPEG-4.

BIFS wird wahrscheinlich nie praktisch eingesetzt werden, da die Funktionalitäten für interaktive 3D-Grafik in MPEG-4 zukünftig auf X3D aufbauen werden.

[MPEG01a, MPEG01b, Wal01, Web3D-MPEG]

3.3 Nichtstandardisierte, proprietäre Formate

Die nichtstandardisierten proprietären Formate sind meist nur für wenige Anwendungsbereiche konzipiert und erreichen dort aufgrund ihrer Spezialisierung ein höheres Niveau als VRML97. Man konzentriert sich insbesondere auf die Produktvisualisierung, Charakteranimationen und den Unterhaltungsbereich. Diese Spezialisierung hat aber den Nachteil, dass man ein Format nicht oder nur sehr schwer für andere Anwendungsbereiche einsetzen kann.

Das grundlegende Vorgehen bei der Erstellung einer Web3D-Applikation mittels eines proprietären Formates besteht darin, dass man zuerst die Objekte in einem professionellen Modellierungswerkzeug erstellt, anschließend werden diese exportiert und mittels eines Format-spezifischen Werkzeugs weiterbearbeitet. Hierbei erfolgen die Definition von Verhaltensweisen, die Nachbearbeitung der importierten Objekte und die Vorbereitungen der Szene für die Publikation im WWW.
Diese Konzentration auf den Export von Objekten aus ausgereiften Modellierungswerkzeugen – wie beispielsweise Maya oder 3ds max – liegt darin begründet, dass man hierdurch in relativ kurzer Zeit qualitativ hochwertige Ergebnisse erzielen kann. Dieser Weg ist natürlich auch bei den standardisierten Formaten möglich, wird aber nicht in dem Maße favorisiert.

Es wurden viele fortschrittliche Lösungen für Problembereiche von Web3D-Grafik entwickelt. Beispielsweise finden Intel-Technologien für die Geometriebehandlung in den Formaten Shockwave3D, Viewpoint und Pulse Verwendung. Weiterhin wurden in den teilweise umfangreichen Autorenwerkzeugen bezüglich der Verhaltensdefinition viele Fortschritte gemacht. Grundsätzlich muss aber gesagt werden, dass diese Werkzeuge meist auf den angestrebten Anwendungsbereich der Web3D-Technologie beschränkt sind. Falls man über den Funktionsumfang des Autorenwerkzeuges hinausgehende Applikationen erstellen will, muss man sich eventuell vorhandener Skriptsprachen bedienen, die im Vergleich mit VRML97 oft große Nachteile bezüglich ihrer Mächtigkeit oder Einfachheit besitzen.

3.3.1 Shockwave3D

Der Begriff „Shockwave3D“ [Shockwave3D] bezeichnet die 3D-Funktionalitäten des Autorenwerkzeuges Macromedia Director (MD) und des Shockwave Players, die seit der Version 8.5 eine vollständige 3D-*Engine* enthalten. Diese auf der *Intel Internet 3D Graphics* - Technologie [I3DSWT] beruhende 3D-*Engine* wurde von den Intel Architecture Labs [IAL] entwickelt.

Abbildung 21: Anzeige des Shockwave Players beim Ladevorgang einer 3D-Applikation

Die grundlegende Philosophie bei der Erstellung einer Shockwave3D-Applikation besteht darin, dass vorhandene 3D-Objekte in den MD importiert und dort gesteuert werden. Die Benutzung von 3D-Objekten im MD hat keine einschränkende Wirkung auf die restliche Funktionalität des Autorenwerkzeuges. Nach der Fertigstellung der Szene wird diese in das Shockwave-Format exportiert und durch den Shockwave Player – einem Webbrowser - Plug-In – beim Client angezeigt.

Es existieren keine Unterschiede zwischen Macromedia Director und dem Macromedia Director Shockwave Studio. Es werden nur die Elemente der Shockwave-Produktfamilie besprochen, die eine direkte Beziehung zur Web3D-Grafik besitzen.

3.3.1.1 Erstellung

Werkzeuge
Die Erstellung einer Applikation erfolgt durch den MD. Das zugrunde liegende Konzept besteht aus einer Bühne *(Stage)*, auf der die attributierten Instanzen *(Sprites)* der Darsteller *(Cast Member)* arrangiert werden. Darsteller sind beispielsweise 3D-Objekte, Texte, Bilder und Sound. Im Eigenschafts-Inspektor *(Property Inspector)* können die Eigenschaften der einzelnen Darsteller oder der Instanzen verändert werden. Die *Score* ist eine Anordnung von Kanälen, welche die Organisation, Anzeige und Kontrolle der Objekte einer Szene über die Zeit ermöglicht.

Die Steuerung von komplexen 3D-Szenen erfolgt aber meist durch die im MD enthaltene Skriptsprache Lingo und nicht mittels der *Score*-Funktionalität des Autorenwerkzeuges. In Lingo existieren insgesamt 300 neue Funktionen und Eigenschaften für die 3D-Darsteller. Dadurch steht eine Sprache zur Erstellung von Web3D-Applikationen zur Verfügung, deren Mächtigkeit mit VRML97 vergleichbar ist.

Durch die Verwendung von Lingo wird somit kein deklarativer, sondern ein objektorientierter oder imperativer Ansatz zur Szenenbeschreibung verfolgt. Lingo war anfänglich eine prozedurale Sprache, entwickelt sich aber immer mehr zu einer objektorientierten Skriptsprache.

Durch den MD und dessen umfangreiche Skriptsprache Lingo ist bezüglich der Generierung von Szenenbestandteilen das Erstellen von grafischen Primitiven, Dreiecksnetzen und Partikelsystemen möglich. Leider können auf diese Objekte nicht alle Methoden angewendet werden, die für ein importiertes Objekt zur Verfügung stehen.

Es existieren Exporter für die gängigsten Modellierungstools [ShockPb] wie beispielsweise für Maya [Maya], 3ds max 4 [3ds max] und trueSpace [trueSpace], die das *.W3D-Format erzeugen können. Animationen können teilweise mittels dieses Formates exportiert werden. Nach dem Import einer W3D- oder OBJ-Datei in den MD steht deren Inhalt als Darsteller zur Verfügung. Dieser Darsteller ist eine eigene Unter-Szene, die 3D-Objekte, Beleuchtung, Texturen, Oberflächeneigenschaften, Animationen, Gruppierungen und Kamerapositionen enthält.

Die 3D-Darsteller können anschließend in eine neue Szene eingefügt werden. Die Steuerung der 3D-*Sprites* erfolgt meist durch entsprechende Lingo-Skripte. Im Eigenschaftsinspektor sind gegenüber der vorhergehenden MD-Version neue Funktionalitäten für die Steuerung der 3D-Objekte vorhanden. Beispielsweise können die Position und die Farbe des Lichtes sowie die Eigenschaften der Kamera bestimmt werden.

Dateiformat
Es sind nur wenige Informationen bezüglich des Dateiformates veröffentlicht. Der Import von 3D-Objekten in den MD erfolgt bevorzugt in dem proprietären, szenengraphbasierten *.W3D-Format. Die MD-Applikation wird im *.DIR-Format gespeichert. Die Publikation der Shockwave3D-Applikation erfolgt im *.DCR-Format. Die Codierung des *.DIR-,*.DCR- und *.W3D-Formates erfolgt nicht im Klartext.

Ereignismodell
Mit dem MD ist eine visuelle Modellierung des Verhaltens mittels der vordefinierten Aktionen möglich. Bei diesen können die Auslöser Maus- und Tastaturereignisse *(Trigger)* sein. Hierdurch wird beispielsweise das Verändern des Blickpunktes, das Starten von Animationen und die Ansteuerung von Lingo-Skripten *(Handler)* ermöglicht.

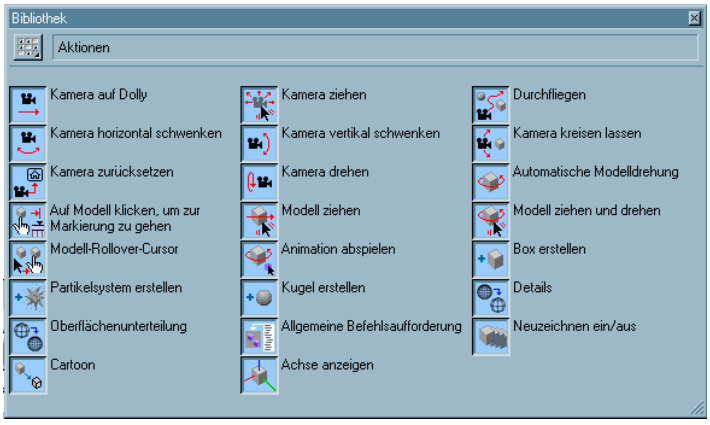

Abbildung 22: Vordefinierte Aktionen

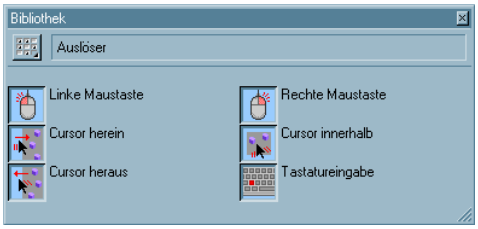

Abbildung 23: Vordefinierte Ereignisse

Umfangreichere Ereignisketten können durch die Elemente der Skriptsprache Lingo realisiert werden.

Integration von Skript- und Programmiersprachen
Durch Lingo können neben grundlegenden 3D-Funktionen insbesondere umfangreiche Animationen (*Keyframe- and Bones-Animation*, Netzverformungen), Kollisionserkennung und *Non-Photorealistic Rendering* (NPR) gesteuert werden. Der gesamte Sprachumfang von Lingo umfasst 800 Befehle.
Die Scriptsprache Lingo enthält objektorientierte Ansätze. Dies äußert sich beispielsweise in dem Klassenkonzept, dass durch die *Parent-Scripts* (Klassen) und *Child-Objects* (Instanzen) repräsentiert wird.

Erweiterbarkeit
Es ist möglich *Xtras* für den MD und den Shockwave-Player zu erstellen, um deren Funktionalitäten zu erweitern. [TN13611]

Besonders hilfreich bei der Erstellung von Shockwave3D-Applikationen ist das *Xtra* 3DPI [3DPI], durch das ein komfortabler Eigenschaftsinspektor zur Verfügung steht. Es existiert auch eine prototypische Implementierung eines *Xtras*, das es ermöglicht bestehende X3D-Szenen als Darsteller zu importieren [X3D File Converter/Importer].

Medienintegration
Es nicht möglich Audioquellen im dreidimensionalen Raum anzuordnen. Das Einfügen von Videotexturen ist durch Lingo möglich.

Features
Bones-Animation, Mesh Deformation, Partikelsysteme, 3D-Text, Keyframe-Animationen, Kollisionserkennung, *Non-photorealistic Shading,* dynamische Generierung von Texturen zur Laufzeit, Animation und Generierung von Schattierungen zur Laufzeit *(Reflection Maps, Light Maps, Specular Light Maps, Gloss Maps)*

3.3.1.2 Repräsentation

Durch den Shockwave Multiuser Server ist das Erstellen von Multiuser-Welten möglich. Durch serverseitiges Lingo-*Scripting* existiert die Möglichkeit, eigene Serverfunktionen zu erstellen.

3.3.1.3 Kommunikation

Streaming
Eine Shockwave-Applikation kann in mehrere DCR-Dateien zerlegt werden. Der Shockwave-Player verwaltet das dynamische Nachladen der einzelnen Objekte einer DCR-Datei und der DCR-Dateien selbst.
Es kann festgelegt werden, ob eine Präsentation erst nach dem vollständigen Download der Szene erfolgt oder ob schon nach der Übertragung initialer Daten mit der Darstellung begonnen wird. Vor dem Beginn des Renderings einer Szene müssen die *Score,* die Skripte, die Größe der Objekte und die Objekte für den ersten Frame übertragen werden.
Es kann eine Anpassung der Szene entsprechend des Ladezustandes der einzelnen Objekte erfolgen. Hierfür gibt es Optionen im MD und entsprechende Lingo-Befehle.
Das Streaming von 3D-Objekten der Shockwave-Applikation kann aber auch durch die Lingo-Befehle `bytesStreamed`, `preLoad`, `state` und `streamSize` gesteuert werden. Hierdurch kann festgelegt werden, wie bestimmte Objekte nachgeladen werden und ab welchem Übertragungszustand diese Objekte durch andere Lingo-Befehle bearbeitet werden können. Diese Befehle sind eine Erweiterung für 3D-Objekte, zusätzlich zu den grundlegenden Netzwerk- und Streaming-relevanten Lingo-Befehlen.
Bei Sprüngen in der Szene, die zu noch nicht geladenen Objekten führen, können entweder Platzhalter eingesetzt oder die Szene gestoppt werden
Für das Streaming von Geometrien, die durch einen Exporter erzeugt wurden, wird die *Multi-Resolution Mesh* (MRM) - Technologie [MRM] verwendet. MRM ähnelt sehr stark der im Gliederungspunkt 2.5.3.3 vorgestellten *Progressive Compressed Stream* - Technologie. Es werden aber keine Aussagen gemacht, ob durch den Einsatz der MRM-Technologie eine Verringerung der Gesamtdatenmenge der Geometrie eines Objektes erreicht wird.
Durch die MRM-Technologie ist es möglich die Geometrie sukzessiv zu übertragen und anzuzeigen. Dies erfolgt vollkommen automatisch. Der Autor muss die Geometrie für die Verwendung der MRM-Technologie nicht überarbeiten. Das Streaming erfolgt automatisch entsprechend der zur Verfügung stehenden Übertragungsbandbreite.

Abbildung 24: Durch die MRM-Technologie übertragenes Dreiecksnetz mit 229, 1000 und 8772 Eckpunkten.

Medienauslieferung
Es erfolgt keine Trennung von Szenenbeschreibung, 3D-Darstellern und den verschiedenen Medien. All diese Daten sind in einer oder mehreren DCR-Dateien enthalten, deren Übertragung aber gesteuert werden kann. Die DCR-Datei muss nicht vollständig zum Client übertragen werden, bevor eine Präsentation der Szene erfolgen kann. Das Nachladen der Medien kann durch den Shockwave Player selbst oder durch Festlegungen des Autors gesteuert werden.

Durch den Lingo-Befehl `preloadNetThing` ist es möglich, Daten die nicht in einer DCR-Datei enthalten sind, entsprechend den Anfoderungen des Programmes zu übertragen und mit Hilfe dieser Daten neue Darsteller und Objekte zu erstellen.
Für das Streaming von Shockwave Audio (SWA), MP3 und QuickTime-Videos stehen spezielle Lingo-Befehle zur Verfügung. Bei der Verwendung von QuickTime-Videos kann auf den RTSP-Server zurückgegriffen werden.

3.3.1.4 Präsentation

Player
Der Shockwave Player ist ein sehr verbreitetes Plug-In für den Internet Explorer und den Netscape Navigator. Einerseits hat die Größe des Plug-Ins mit 3,4 MByte eine abschreckende Wirkung für den potentiellen Nutzer, aber andererseits ist dieses Plug-In in einigen Windows-Betriebssystemen und Webbrowsern standardmäßig – wenn auch noch nicht in der Version 8.5 – enthalten.

Softwaretechnologischer Zugriff
Die bidirektionale Kommunikation zwischen Webbrowser und der Shockwave-Anwendung ist über die Lingo-Befehle `externalEvent` und `on evalScript` möglich. Mittels `externalEvent` können beliebige Skripte innerhalb der Webseite aufgerufen werden. Durch `on evalScript` ist es möglich die Shockwave-Anwendung mittels JavaScript oder VBScript innerhalb der Webseite zu steuern. Durch diese Schnittstelle können beliebige Befehle ausgetauscht werden. [TN14617]

Skalierung
Durch die *Subdivision Surfaces* (SDS) - Technologie werden nicht vorhandene Informationen eines Dreiecksnetzes durch Interpolation ergänzt. Dadurch erreicht man eine natürlichere, glättere Oberfläche. Der Begriff „*Subdivision*“ – Unterteilung – wird deshalb verwendet, da große Polygone, die zu unnatürlichen Oberflächen führen würden, in viele kleine Polygone unterteilt werden. Die Zerlegung der Polygone erfolgt unter Berücksichtigung der Lage der Nachbarpolygone, wodurch neue Scheindetails entstehen können. Es ist eine Unterteilung für das gesamte Objekt und eine dem Blickwinkel entsprechende adaptive Unterteilung möglich. Die individuelle Parametrisierung der SDS-Technologie kann durch den `sds-modifier` von Lingo erfolgen.

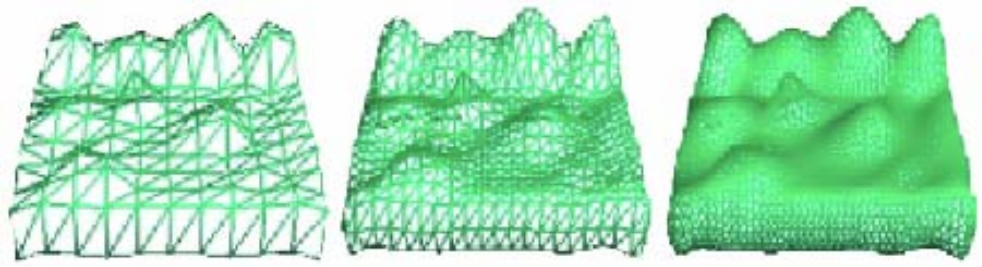

Abbildung 25: Durch *Subdivision Surfaces* geglättete Oberflächen [Jun00]

Die Tiefe der Unterteilung wird automatisch an die gegebene Rechenleistung angepasst. Somit kann bei hoher Rechenleistung eine verfeinerte Darstellung der Oberfläche erfolgen und bei niedriger Rechenleistung möglicherweise die Echtzeitfähigkeit aufrecht erhalten werden. Der Modellierer kann einzelnen Zonen der Objekte Gewichtungsfaktoren zuordnen, wodurch in diesen Bereichen die Tiefe der Unterteilung gesteuert wird. Beispielsweise ist das Gesicht einer Person wichtiger als die Detailgenauigkeit seiner Schuhe. [Jun00, Fre01]

Auch durch die MRM-Technologie ist eine Anpassung der Objektgeometrien an die verfügbare Rechenleistung möglich. Es werden nur neue Geometrieinformationen zum Objekt hinzugefügt und angezeigt, wenn dies auch durch den Player bewältigt werden kann. Natürlich ist auch das Reduzieren der Objektgeometrie entsprechend der Rechenleistung möglich. Die individuelle Parametrisierung der MRM-Technologie erfolgt über den `lod-modifier` von Lingo.

Falls in Lingo nicht anders definiert, sind die MRM- und SDS-Technologie standardmäßig aktiviert.

Durch Lingo kann auch eine Anpassung der Applikation entsprechend der verfügbaren Rechenleistung erfolgen, indem man beispielsweise auf die aktuelle Framerate reagiert.

3.3.1.5 Bewertung

Einsatzgebiet
Shockwave3D ist insbesondere für Spiele- und E-Commerce - Applikationen aufgrund seiner hohen Detailgenauigkeit und der Multiuserfunktionalitäten geeignet. Diese hohe Detailgenauigkeit kann auch bei geringen Bandbreiten und leistungsschwachen Web3D-Plattformen erreicht werden. Durch die Möglichkeiten der *Bones*-Animation und des *Non-Photorealistic Rendering* ist Shockwave3D auch für die Darstellung von Charakteranimationen und insbesondere von Cartoons prädestiniert.

Nutzungsbedingungen
Es muss nur der MD gekauft werden. Es werden keine zusätzlichen Gebühren für das Bereitstellen von 3D-Applikationen erhoben.

Zukunft des Formates
Macromedia ist bezüglich Shockwave3D viele Partnerschaften mit Firmen eingegangen, die Modellierungsprogramme und MD Add-Ons erstellen sowie 3D-Modelle vertreiben. [ShockPb]

Weiterhin wurde durch die Kooperation mit Intel ein Partner mit umfangreichem Werbeetat gewonnen, der nach neuen rechenintensiven Anwendungen für seine Prozessoren sucht. Ob durch eine mögliche Optimierung für Intel-Prozessoren ein signifikanter Vorteil gegenüber anderen Prozessoren besteht, müsste evaluiert werden.

Fazit
Der Shockwave Player besitzt einen sehr hohen Verbreitungsgrad. Laut Macromedia existieren zur Zeit weltweit 200 Millionen Installationen. Da die Version 8.5 des MD erst seit kurzem verfügbar ist, existieren erst wenige Shockwave-Applikationen, die die neuen 3D-Funktionalitäten nutzen.
Shockwave3D stellt eine Kombination aus exzellenten Autorenwerkzeugen und modernsten 3D-Technologien wie SDS und MRM dar. Weiterhin kann auf eine große Anzahl von Autoren zurückgegriffen werden, die mit dem MD und Lingo vertraut sind.

[Mac01, HSW, IPRO]

3.3.2 Viewpoint

Die Viewpoint-Technologie [Viewpoint] – welche vielen noch unter dem Namen Metastream geläufig sein wird – ist ein offenes, XML-basiertes und Streaming-fähiges Web3D-Format. Die grundlegende Philosophie besteht darin, in einem Modellierungsprogramm 3D-Objekte zu erstellen und diese anschließend in das *.OBJ-, *.ASE- oder *.NFF-Format zu exportieren. Danach werden diese Objekte in den Viewpoint Scene Builder (VSB) importiert und mittels der Szenenbeschreibung in eine 3D-Applikation integriert.

3.3.2.1 Erstellung

Dateiformat
Der Grundgedanke ist, dass die deklarative Szenenbeschreibung von der Geometrie und den Medienobjekten getrennt wird. Für die Beschreibung einer Viewpoint-Applikation sind somit meist 2 Dateien notwendig. Eine enthält die Szenenbeschreibung in XML (*.MTX) und eine die verschiedenen Medienobjekte (*.MTS). Diese stellen die Grundbausteine für die Szene dar und können beispielsweise Daten für 3D-Objekte, Sound und Texturen enthalten.
Die XML-Grammatik für die Szenenbeschreibung ist nur durch ein Textdokument und nicht in Form einer DTD oder eines Schemas dokumentiert. Die MTX-Datei kann komprimiert werden und besitzt danach die Dateiendung MTZ.
Die verschiedenen Medienobjekte – hier als Medienatome bezeichnet – sind in binärer, Streaming-geeigneter und komprimierter Form in der MTS-Datei enthalten. Diese Dateien sind in einzelne Blöcke unterteilt, welche wiederum in einzelne Unterblöcke aufgeteilt werden können. Jeder einzelne Block besteht aus einem *Header* und einem Daten-Segment. Es gibt 2 verschiedene Blockarten. Einerseits die Transportblöcke, die für die Speicherung und Organisation anderer Blöcke verantwortlich sind und andererseits die Datenblöcke, welche die Textur- und Geometriedaten enthalten.

	Name	Kürzel	Beschreibung
Transportblöcke			
	Enclosure	mts	Jede MTS-Datei muss einen *Enlosure*-Block enthalten, welcher grundlegende Informationen über die Datei enthält.
	Group	gp	Hierdurch können hierarchische Blockstrukturen innerhalb der MTS-Datei realisiert werden. Hier kann auch festgelegt werden wie die einzelnen Blöcke, die wiederum in Unterblöcke unterteilt werden können, innerhalb der Datei angeordnet sind.
Datenblöcke			
	JPEG Image	jpeg	Bildinformationen für Texturen im JPEG-Format
	Texture Map	txm	Hier werden zusätzlich zu den Bildinformationen auch Informationen für die Benutzung als Textur angegeben.
	Bounding Box	bbox	Daten für die Art der Hüllenstruktur eines 3D-Objektes, welche für einen sukzessiven Bildaufbau wichtig sind
	UMF	umf	Geometriedaten und Oberflächeneigenschaften von 3D-Objekten

Tabelle 4: Blockarten in einer MTS-Datei

Die Informationen zum Aufbau der MTS-Dateien wurden aus [Aba99] entnommen. Leider sind keine aktuellen Informationen über die neuen Blockarten oder Veränderungen innerhalb einer MTS-Datei verfügbar.

Die Blöcke werden in der MTS-Datei so angeordnet, dass ein gleichmäßiger sukzessiver Szenenaufbau möglich ist. Es wäre zum Beispiel unvorteilhaft erst den gesamten Block für die Geometrie A und anschließend die Geometrie B zu übertragen. So wäre nämlich für eine bestimmte Zeit nur A zu sehen, obwohl B möglicherweise für die Szene wichtiger wäre.

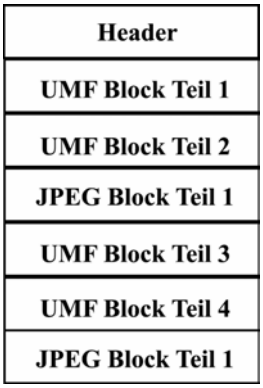

Abbildung 26: Beispiel für eine Unterblock-Struktur in einer MTS-Datei

In diesem Beispiel wird eine JPEG-Textur und ein 3D-Objekt übertragen, wobei die Geometrieinformationen einen höheren Umfang haben. Beide Blöcke erhalten durch diese Anordnung eine faire Behandlung. Durch die Anordnung der Blöcke entsprechend dem einzelnen Datenvolumen wird erreicht, dass jedes Medienobjekt entsprechend der Blockgröße gleichberechtigt neue additive Daten erhält.

Internes Modell und interne Struktur

Die interne Struktur der szenengraphbasierten MTX-Datei ist in die Bereiche Kopf *(MTX File Header)* und die eigentliche Szene `(MTSScene)` untergeteilt. Diese ist wiederum in Szenenparameter `(MTSSceneParms)`, Kameraeinstellungen `(MTSCameraMode)`, Szenengraph `(MTSInstance)` sowie Animationen und Ereignisbehandlung `(MTSTimeElem)` aufgeteilt.

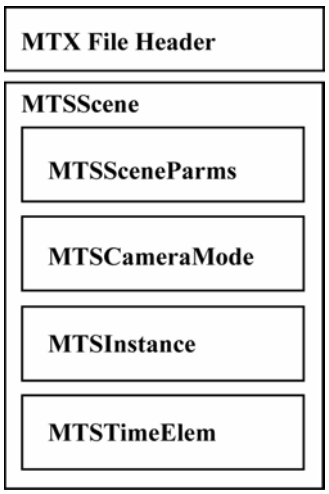

Abbildung 27: Interne Struktur einer MTX-Datei

Ereignismodell
Durch `OnClick`, `OnDrag`, `OnEnter` und `OnExit`-Ereignisse können Animationen gestartet, die Kameraposition verändert, Attribute verändert und kopiert und JavaScript-Methoden aufgerufen werden.

Integration von Skriptsprachen
Es kann innerhalb der MTX-Datei JavaScript-Code definiert werden. Welche Funktionalitäten hierdurch realisiert werden können, ist nur unzureichend dokumentiert. Meist werden die JavaScript-Methoden der umgebenden HTML-Datei aufgerufen, welche wiederum auf die Szene zugreifen können.

Erweiterbarkeit
Es existiert ein SDK [Viewpoint Developer] für die Viewpoint-Technologie, welches den Mitgliedern des Viewpoint Alliance Networks zur Verfügung steht. Hierdurch ist es möglich eigene Erweiterungen an den Dateiformaten und die entsprechende Umsetzung innerhalb des Players durch Komponenten zu realisieren.

Medienintegration
Audiodateien, die im WAV-Format vorliegen, können eingebunden werden. Das direkte Einbinden von Video in eine Szene ist nicht möglich. Die Texturen müssen im JPEG-, BMP- oder Trixels NT - Format vorliegen.
Neben der Benutzung der MTS-Dateien zur Speicherung der Medienobjekte ist es auch möglich JPEG-, BMP- und WAV-Dateien direkt in die Szenenbeschreibung einzubinden.
Weiterhin können in eine Szene Flash-Animationen, die im *.SWF-Format vorliegen, eingebunden werden. Hierdurch ist es möglich 3D-Objekte mit Flash-Animationen zu kombinieren. Beispielsweise lassen sich Texturen als Flash-Animationen realisieren.
Es können auch die Formate QuickTime VR und iPIX für die Erstellung von Panoramen eingebunden werden.

Werkzeuge

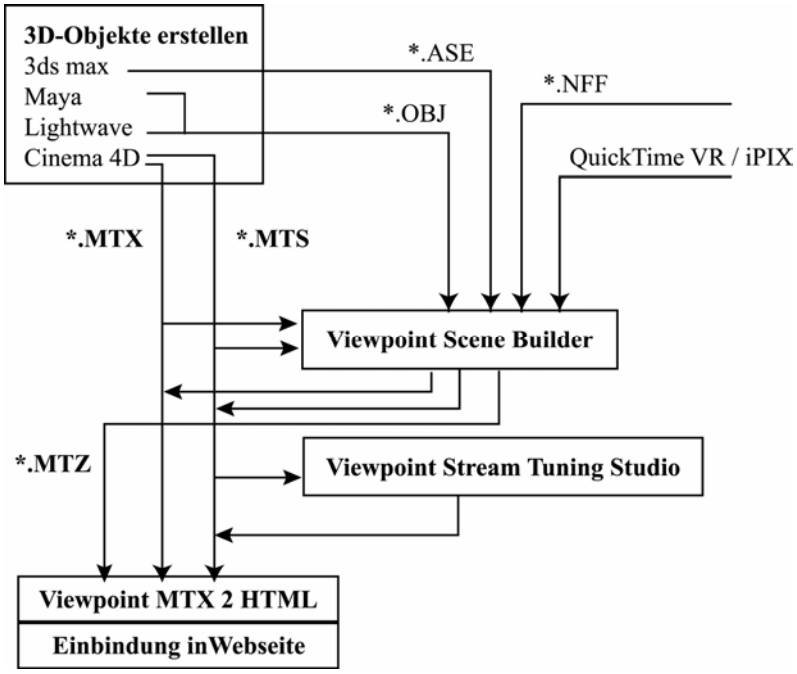

Abbildung 28: Die Viewpoint-Werkzeuglandschaft

Es existieren viele Werkzeuge, die aber nicht in eine entsprechende Entwicklungsumgebung integriert sind.
Die Erstellung von 3D-Objekten kann beispielsweise durch die Modellierungswerkzeuge 3ds max [3ds max], Maya [Maya], Lightwave [Lightwave] oder Cinema 4D [Cinema 4D] erfolgen.
Der Viewpoint Scene Builder (VSB) ist das zentrale Werkzeug innerhalb der Viewpoint-Technologie. Es können Objekte im *.MTS/*.MTX-, *.OBJ-, *.ASE-, *.NFF-, iPIX- und QuickTime VR - Format importiert werden. Durch das *.ASE-Format ist es beispielsweise möglich, Animation aus 3ds max zu übernehmen.

Beispielszene

```
<?xml version="1.0"?>
<MTSScene Version="3">

    <!—Szenenparameter →
     <MTSSceneParms EdgeBias="1"/>

    <!—Kameraeinstellungen →
     <MTSCamera OrbitDist="4">
          <Translate x="-0.01" y="0.33" z="0"/>
     </MTSCamera>

    <!—Definition der Kugel →
     <MTSInstance Name="sphere">
          <MTSGeometry Name="Sphere" Type="MTSSphere"/>
          <MTSMaterial Name="sphere_Mat_0" ID="0"  RenderMode="Texture">
               <MTSTextureMap Type="Diffuse" Name="sphere_TEXTURE"/>
          </MTSMaterial>
          <OnClick Action="Trigger" ID="0" Target="anim"/>
     </MTSInstance>

     <!—die Zuweisung der Textur wird mittels einer Animation realisiert →
     <MTSTimeElem Type="MTSImageStream" On="1" Path="texture.jpg">
          <Target Name="MTSTexture.sphere_TEXTURE"/>
     </MTSTimeElem>

    <!—Die Animation der Position der Kugel →
     <MTSTimeElem Type="Keyframe" Name="location" On="0" Clamp="10">
          <Target Name="MTSInstance.sphere" Property="loc_" Timeline="T1"/>
          <Time>
               0 0.5 1.0
          </Time>
          <Timeline Name="T1" Type="R3D">
               [-3 2 0]   [0 0 3]    [-3 2 0]
          </Timeline>
     </MTSTimeElem>

</MTSScene>
```

Die Funktionen des VSB beschränken sich auf

- Bearbeitung der Geometrien und Oberflächeneigenschaften
- Transformationen
- Bearbeitung importierter Animationen
- Erstellen von primitiven Objekten
- Generierung von *Rollover*-Effekten, die bei der Berührung der Maus mit primitiven Objekten erzeugt werden
- Festlegen von globalen Einstellungen für die Szene

Es können leider keine weiteren Interaktionen oder Animationen mit dem VSB erstellt werden. Weiterhin kann die Konvertierung von JPEG in das Trixels NT - Format innerhalb des VSB realisiert werden.

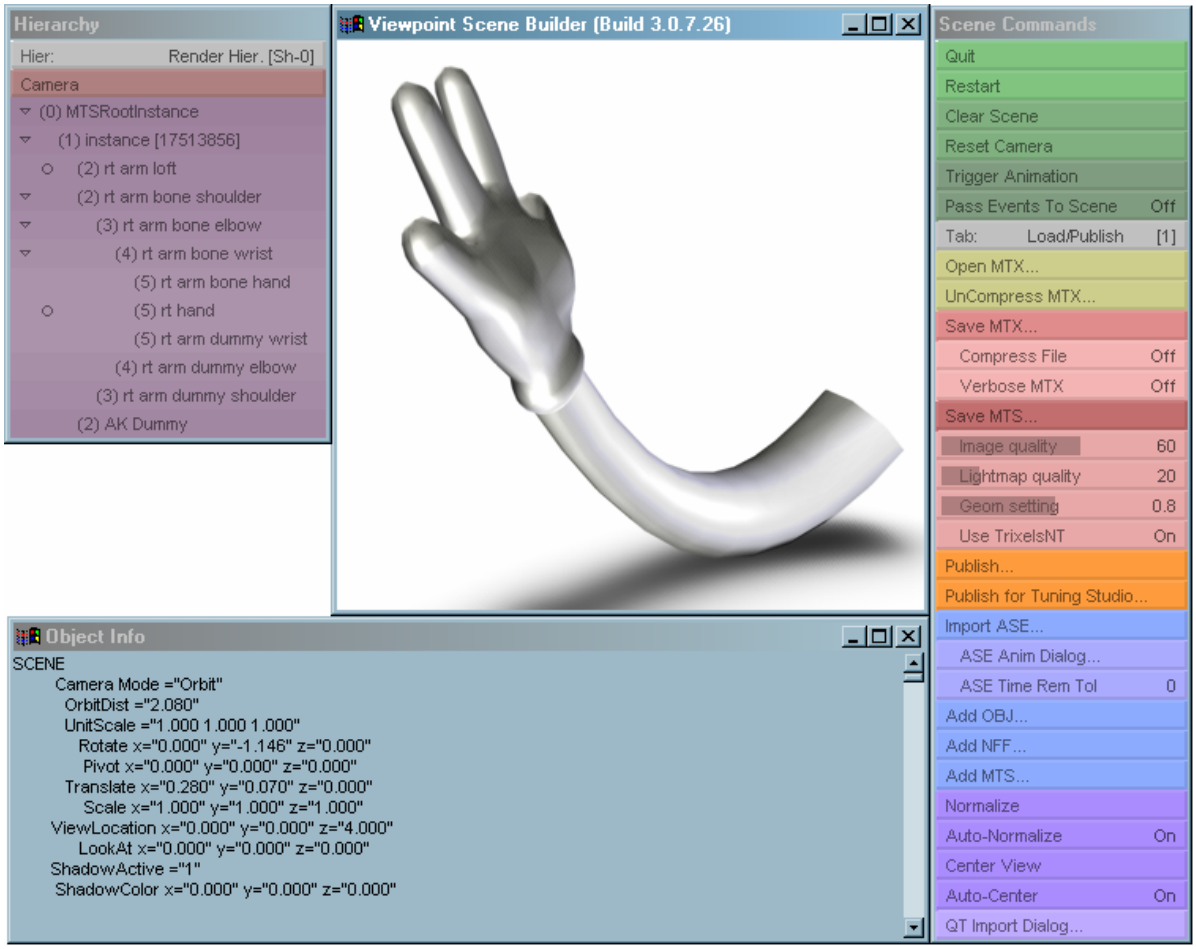

Abbildung 29: Viewpoint Scene Builder

Durch das Viewpoint Stream Tuning Studio können die Medienobjekte in den MTS-Dateien optimiert werden. Dies bezieht sich meist auf eine Verringerung der Polygonanzahl der Objektgeometrien und auf eine Komprimierung der Texturen.

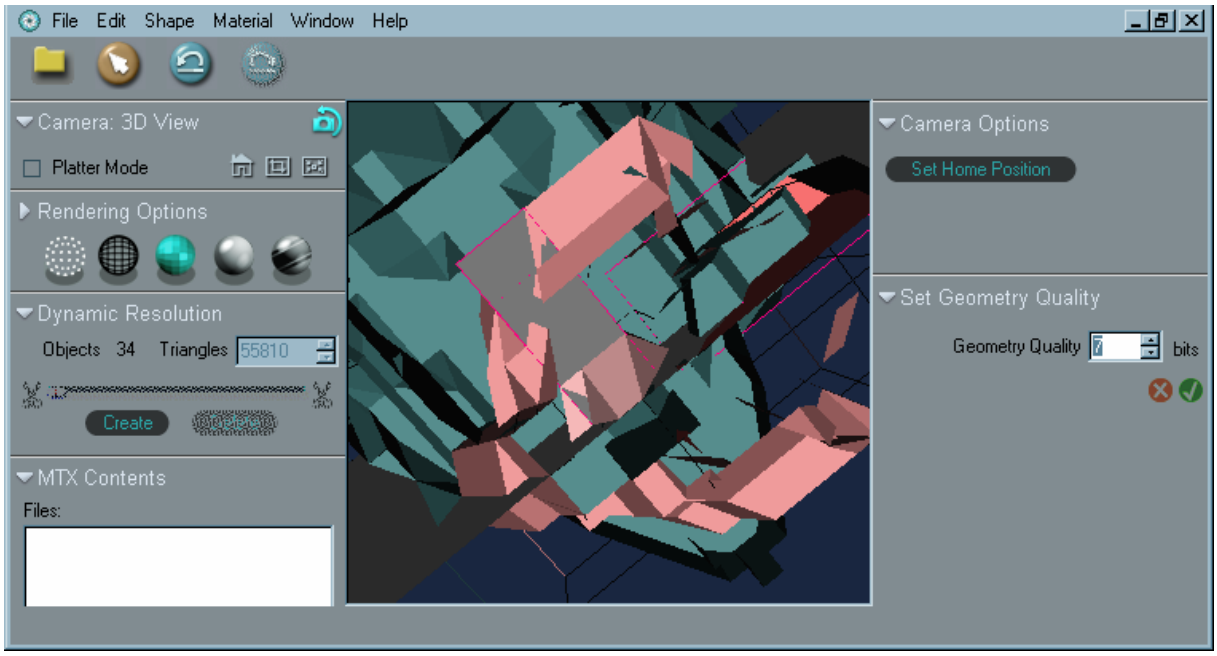

Abbildung 30: Viewpoint Stream Tuning Studio

Das Werkzeug Viewpoint MTX 2 HTML generiert ein entsprechendes HTML-*Template* für die Anzeige der Viewpoint-Applikation.

3.3.2.2 Repräsentation

Repräsentationsformat
Durch VSB kann die Szenenbeschreibung in das komprimierte *.MTZ-Format umgewandelt werden. Weiterhin ist es in diesem Werkzeug möglich, den Speicherplatzbedarf der MTS-Datei zu verringern. Es kann angegeben werden, in welcher Qualität die Texturen und die Geometrien abgespeichert werden. Die Kompression der Texturen erfolgt entweder durch das JPEG- oder durch das Trixels NT - Verfahren [ViewpointWP]. Das letztgenannte Verfahren basiert auf einer *Wavelet*-Kompression und erreicht gegenüber der diskreten Kosinustransformation von JPEG bessere Komprimierungsraten und eine höhere Bildqualität.

Softwaretechnologischer Zugriff
Da die Szenenbeschreibung in XML erfolgt, wäre eine komfortable dynamische Generierung von MTX-Dateien denkbar. Leider ist dies durch das Nichtvorhandensein einer entsprechenden DTD oder eines Schemas problematisch. Es kann beispielsweise durch eine JavaScript-Methode eine URL auf dem Webserver aufgerufen werden, welche wiederum eine Datenbank anspricht und aus den Ergebnissen eine MTX-Datei generiert, welche dann in die Szene beim Client eingefügt wird.

3.3.2.3 Kommunikation

Streaming
Nachdem die Szenenbeschreibung vollständig übertragen wurde, kann mit der Darstellung der Szene durch den Player begonnen werden.
Es wird die *Multi-Resolution Mesh* (MRM) - Technologie [MRM] verwendet, welche auch in Shockwave3D (siehe Gliederungspunkt 3.3.1.3 „Kommunikation“) integriert ist.

Medienauslieferung
Die Auslieferung der Medien erfolgt im *.MTS-Format. Zu einer Szene können mehrere verschiedene MTS-Dateien gehören.

3.3.2.4 Präsentation

Player
Die Anzeige der Szene erfolgt durch den modularen Viewpoint Media Player (VMP), einem Webbrowser - Plug-In. Das Grundgerüst stellte der updatefähige und erweiterbare Viewpoint Hub dar, der eine Größe von 349 KByte besitzt. Durch diesen werden zusätzliche und neue Player-Komponenten nachgeladen, installiert und verwaltet. Dadurch sind keine manuellen Updates oder Erweiterungen notwendig und die initiale Downloadgröße des Plug-Ins ist relativ klein. Es können nicht nur Viewpoint-Komponenten dynamisch nachgeladen werden, sondern auch eigene proprietäre Erweiterungen.
Der Viewpoint-Player erlaubt es auch die Szene über anderen Webseiteninhalten – wie beispielsweise Text oder Bilder – darzustellen.

Softwaretechnologischer Zugriff
Es ist möglich mittels JavaScript auf die Szene zuzugreifen. Die Schnittstellen hierfür sind vorgegeben. Es ist beispielsweise möglich Attribute des Szenegraphen abzufragen und zu verändern. Auch die Steuerung von Animationen ist durch JavaScript möglich.

Skalierung
Durch die Verwendung der MRM-Technologie von Intel (siehe Gliederungspunkt 3.3.1.4 „Kommunikation“) ist auch eine Anpassung der Geometrie der Objekte an die verfügbare Rechenleistung möglich.

3.3.2.5 Bewertung

Einsatzgebiet
Das Viewpoint-Format wurde für die Realisierung von Produktpräsentationen entwickelt und beschränkt sich in seiner Funktionalität auch auf diesen Anwendungsbereich.
Durch die Verwendung von *ZoomView* [ViewpointWP] kann parallel zu einem 3D-Modell auch eine detaillierte zweidimensionale Darstellung des Objektes angeboten werden. Die Daten für dieses Bild werden entsprechend des aktuellen Zoom-Status zum Client übertragen. Dies ist für Applikationen, die einen sehr hohen Detaillierungsgrad erfordern – wie beispielsweise für die Darstellung der Maserung eines Möbelstückes – sehr vorteilhaft.
Besonders für E-Commerce - Anwendungen ist das Nutzer-Tracking durch *DataView* [ViewpointWP] interessant. Hierdurch kann verfolgt werden, wie der Nutzer durch die Szene navigiert und wie er mit ihr interagiert.
Die Integration von Flash-Animationen ist vor allem für professionelle Produktpräsentationen interessant.

Nutzungsbedingungen
Die Software für die Erstellung von Viewpoint-Applikationen ist frei verfügbar. Mitglieder des Viewpoint Alliance Networks erhalten Zugang zu dem Viewpoint-SDK. Es existiert eine Lizenz für die nichtkommerzielle Nutzung, bei der aber der Schriftzug „Viewpoint“ durch das Plug-In angezeigt wird. Die Kosten für eine kommerzielle Lizenz liegen zwischen 2.500 und 100.000 US$.

Fazit
Viewpoint ist eine exzellente Nischenlösung für anspruchsvolle Produktpräsentationen. Durch diese Technologie können 3D-Objekte in sehr hoher Qualität und großer Realitätsnähe dargestellt werden. Ursache hierfür sind hauptsächlich die exzellente Texturenbehandlung und die Anwendung von Textur- und Geometrie-Streaming.
Es ist eines der ersten Formate, das die Szenenbeschreibung durch ein XML-Dokument realisiert. Leider werden die meisten Vorteile einer XML-Codierung durch das Fehlen einer Viewpoint-Grammatik in Form einer DTD oder eines Schemas zunichte gemacht. Zudem sind die Dokumentation für das Format und die meisten Tutorials meist oberflächlich und fehlerhaft, was ein Erstellen einer Szenenbeschreibung durch einen XML-Editor erschwert. Wünschenswert wäre auch ein Autorenwerkzeug, in dem die verschiedenen Werkzeuge integriert sind.

[Viewpoint, Viewpoint Developer, ViewpointWP, Aba99]

3.3.2.6 Adobe Atmosphere

Adobe Atmosphere [Atmosphere] basiert auf der Viewpoint-Technologie und besteht aus dem Modellierungsprogramm Atmosphere Builder und dem Webbrowser - Plug-In Atmosphere Browser. Adobe Atmosphere befindet sich zur Zeit noch in der Beta-Phase.
Die Atmosphere-Technologie ist für die Realisierung von Multiuser-Welten mit entsprechenden Avatar-Funktionalitäten prädestiniert. Hierfür wird beispielsweise auch eine Chat-Funktionalität angeboten, die mittels des Adobe Community Servers realisiert wird.
Der Grundgedanke besteht darin, dass die Objekte der zu erstellenden Welt mittels der Viewpoint-Werkzeuge realisiert und in das Viewpoint-Format exportiert werden. Diese werden dann in den Atmosphere Builder importiert. Mittels dieses intuitiven Werkzeuges können in sehr kurzer Zeit anspruchsvolle Multiuser-Welten erstellt werden. Besonders hilfreich bei der Erstellung der virtuellen Welten sind die *variable Tools*. Durch diese können beispielsweise Wände, Tore, Treppen und Steigungen erstellt werden.

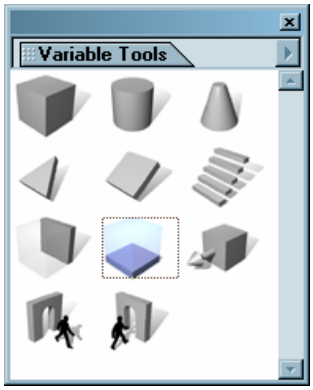

Abbildung 31: *Variable Tools*

Die Verhaltensmodellierung erfolgt mit Hilfe einer JavaScript-API, die aber noch nicht veröffentlicht wurde. Die Publikation der Szene erfolgt in dem binären *.AER-Format welches durch ein 4,53 MByte großes Webbrowser - Plug-In angezeigt werden kann. Dessen Größe könnte eine abschreckende Wirkung auf einige potentielle Nutzer haben.

Die auf die Viewpoint-Technologie aufbauende Atmosphere-Technologie ermöglicht die intuitive Erstellung von qualitativ hochwertigen Multiuser-Welten.

3.3.3 Cult3D

Das Hauptanwendungsgebiet von Cult3D [Cult3D] der Firma Cycore [Cycore] ist der E-Commerce - Bereich. Der Grundgedanke ist, dass die Szene in einem Modellierungsprogramm erstellt und aus diesem exportiert wird. Anschließend erfolgt die Aufbereitung für die Publikation im WWW.

3.3.3.1 Erstellung

Werkzeuge

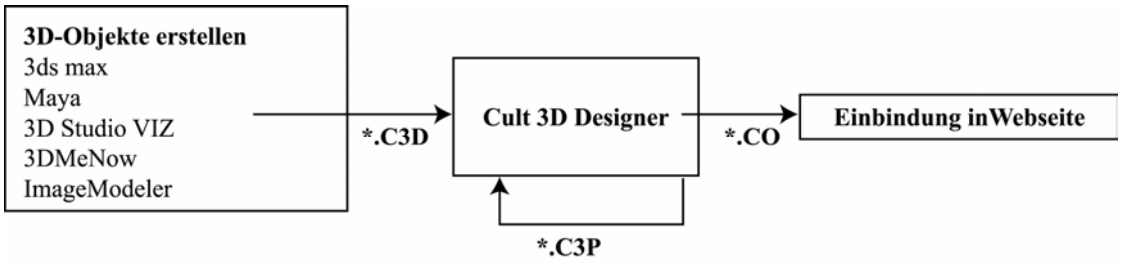

Abbildung 32: Die Cult3D-Werkzeuglandschaft

Die Erstellung der 3D Objekte kann beispielsweise durch die Modellierungsprogramme Maya [Maya] oder 3ds max [3ds max] erfolgen. Hierfür existieren sehr komfortable und konfigurierbare Exporter, die ein binäres Format (*.C3D) erzeugen. Auch durch 3D Studio VIZ [Autodesk] – einem 3D-Modellierungsprogramm für Architekten, Landschaftsplaner und Konstrukteure –, 3DMeNow [3DMeNow] – welches aus 2 Photographien einer Person ein dreidimensionales Modell erzeugen kann – und dem ImageModeler [ImageModeler] – ein Programm zur Erzeugung eines 3D-Objektes aus 2D-Photographien – kann dieses Format mittels eines Exporters erzeugt werden. Leider können noch keine Animationen in das *.C3D-Format exportiert werden.

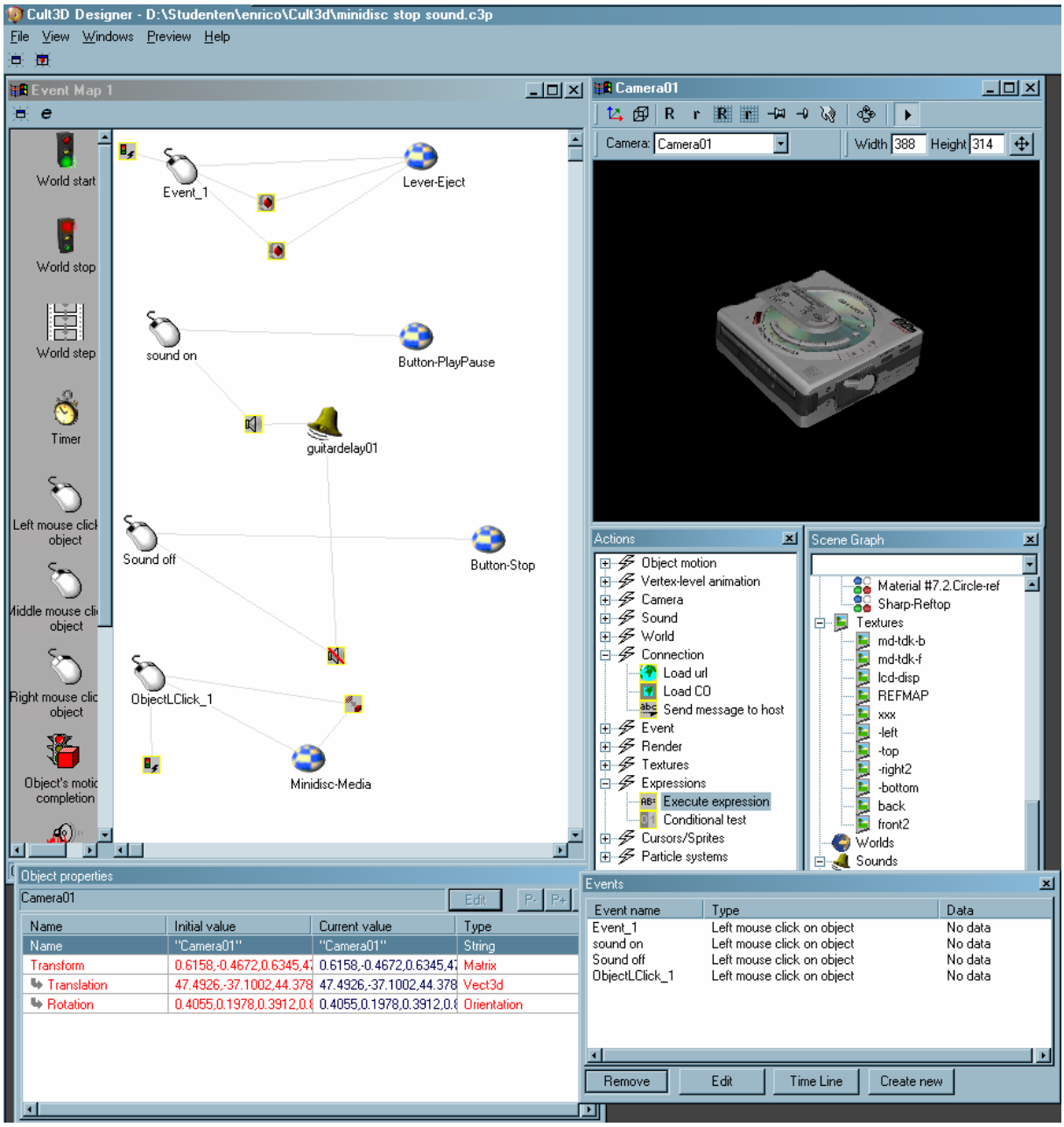

Abbildung 33: Cult3D Designer

Die Weiterbearbeitung der Objekte erfolgt anschließend durch den intuitiven Cult3D Designer. Durch dieses Programm ist es neben der Nachbearbeitung der Szene möglich, die importierte Szene um Verhaltensweisen zu erweitern. Die Szene wird für eine spätere Verwendung durch den Cult3D Designer im *.C3P-Format abgespeichert. Die Publikation der Cult3D-Applikation erfolgt in einem binären, Streaming-fähigen, komprimierten, verschlüsselten und szenengraphbasierten Dateiformat (*.CO). Dieses kann nicht durch den Cult3D Designer geladen werden, was den Vorteil hat, dass sich niemand den Inhalten der Applikation ungewollt bemächtigen kann. Für jede einzelne Animation, Geometrie, Textur und Audioinformation kann die Art und der Grad der Komprimierung festlegt werden.

Datenart	Komprimierung durch
Animationen	Wavlet
Audio	Wavlet, Metasound, Metavoice
Geometrien	Mesh Level 2
Texturen	Wavlet, Graustufen, Farbpalette

Tabelle 5: Komprimierungsarten im *.CO-Format

Für jede Szene gibt es nur eine monolithische Datei (*.C3P,*.CO), in der alle Daten wie die Szenenbeschreibung und die Medienobjekte enthalten sind.

Internes Modell und interne Struktur
Das interne Modell basiert auf einem Szenengraphen. Zur internen Struktur können keine Aussagen gemacht werden, da es sich um nicht-offene und binäre Formate handelt.

Ereignismodell
Durch den Cult3D Designer können Verhaltensweisen intuitiv visuell modelliert werden. Auslöser für ein Ereignis können Anwendungsbeginn und -ende, Maus- und Tastatureingaben und das Abspielende eines Audiosignals sein. Es können auch Ereignisketten definiert werden, bei denen ein Ereignis ein anderes auslöst. Die Behandlung der Ereignisse erfolgt durch die *Actions*, durch die zum Beispiel

- Objektbewegungen und -veränderungen, wie beispielsweise Rotation, Translation oder Skalierung *(Object Motion)*
- Morphing-Effekte *(Vertex-level Animation)*
- Veränderung der Kameraposition *(Camera)*
- Steuerung von Audio *(Sound)*
- Auswechseln der kompletten Szene *(Connection)*
- Aufruf von JavaScript-Methoden in der anzeigenden HTML-Seite *(Connection)*
- Definieren von bedingten Ereignissen *(Expressions)* und Ereignisketten *(Event)*
- Steuerung von Partikelsystemen
- Verändern der Objekterscheinung

realisiert werden können.

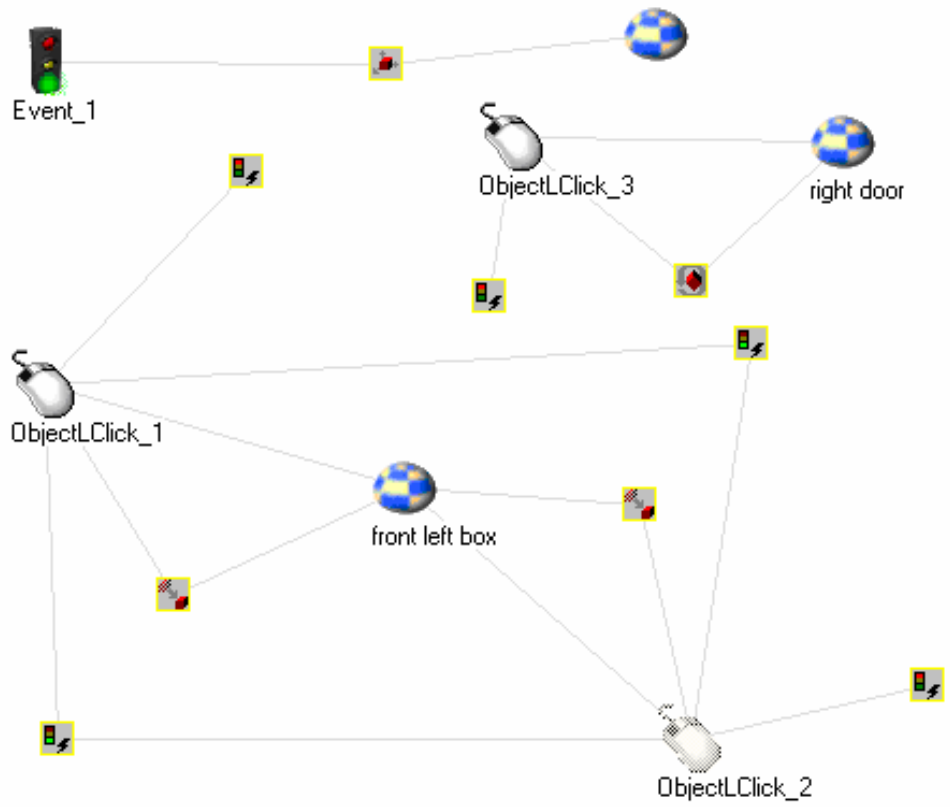

Abbildung 34: Visuelle Verhaltensmodellierung

Integration von Skriptsprachen
Es können zusätzliche *Actions* mit Hilfe von Java programmiert werden. Hierzu muss die entsprechende Java-Klasse in den Cult3D Designer importiert werden. Diese muss die Schnittstelle `Cult3Dscript` implementieren, welche in der umfangreichen Cult3D-API enthalten ist.

Medienintegration
Die Einbindung der Texturen sollte schon durch das Modellierungsprogramm erfolgen. Sie können aber auch nachträglich verändert und ausgewechselt werden. Insbesondere kann für jede einzelne Textur deren Behandlung durch den Renderer festgelegt werden.
Die Audioinformationen werden generell der Szene und nicht einem Punkt innerhalb dieser zugeordnet.

3.3.3.2 Repräsentation

Es wird kein spezieller Webserver benötigt.

3.3.3.3 Kommunikation

Streaming
Die Anzeige der Szene kann schon erfolgen, bevor die gesamte Datei heruntergeladen wurde. Weiterhin soll das Dateiformat auch Streaming unterstützen. Leider sind hierfür keine weiteren Informationen verfügbar.
Durch die *World*-Aktionen können Szenenbestandteile, ähnlich der `Inline`-Funktionalität in VRML97, nachgeladen werden.

Medienauslieferung
Da alle Szenenbestandteile in einer CO-Datei abgelegt werden, kann von einem monolithischen Verfahren gesprochen werden, welches aber aufgrund der Unkenntnis der inneren Struktur dieses Formates nur ungenügend beurteilt werden kann. Dieses Verfahren hat den Vorteil, dass die Anzahl der Anfragen an den Server dadurch verringert wird.

3.3.3.4 Präsentation

Player
Der Cult3D Viewer ist für erstaunlich viele Plattformen verfügbar, was auf eine portable Architektur schließen lässt. Leider besitzt der Player keinen modularen Aufbau und schreckt durch seine Größe von 1,1 MB möglicherweise potentielle Nutzer vom Download des Plug-Ins ab.

Softwaretechnologischer Zugriff
Mittels JavaScript oder VBScript innerhalb der Webseite kann auf die Cult3D-Applikation über eine vordefinierte Schnittstelle zugegriffen werden. Hierdurch können beispielsweise Ereignisse ausgelöst, die Ereignisbehandlung gesteuert oder Anzeigeoptionen verändert werden.

Skalierung
Durch das *World Step* - Ereignis kann festgestellt werden, mit welcher Framerate die Applikation angezeigt wird. Mit Hilfe dieser Information und einer entsprechenden Verwendung in einem Java-Programm kann eine Anpassung der Szene an die Rechenleistung erfolgen.

3.3.3.5 Bewertung

Einsatzgebiet
Cult3D ist ein direkter Konkurrent von Viewpoint, da beide für Produktpräsentationen prädestiniert sind. Hierfür spricht einerseits die exzellente Texturbehandlung und andererseits die vordefinierten produkterklärenden Animations- und Interaktionsfähigkeiten von Cult3D.
Die Cult3D-Applikationen können außerdem nicht nur im WWW, sondern auch in Microsoft Office - und Adobe Acrobat - Dokumenten eingesetzt werden. Hierfür sind entsprechende Plug-Ins zu installieren. Somit existieren mehrere Verwendungsmöglichkeiten für die erstellte Applikation.
Weiterhin ist es möglich eine Cult3D-Applikation in den Macromedia Director einzubinden und dort mittels Lingo zu steuern.

Nutzungsbedingungen
Die Werkzeuge und die Software für die Erstellung von Cult3D-Applikationen sind frei verfügbar. Für die Publizierung muss eine Lizenz erworben werden. Diese ist für die nichtkommerzielle Nutzung kostenlos. Ansonsten existiert eine jährliche Grundgebühr von 500 US$ zu der, entsprechend der Frequentierung der Cult3D-Applikation, noch zusätzliche Kosten hinzukommen.

Zukunft des Formates
Man arbeitet an der Unterstützung von Cult3D für Spiele-Konsolen, Set-Top-Boxen, PDAs und Mobiltelefonen.

Fazit
Die Besonderheit der Cult3D-Technologie ist, dass durch die komfortablen, konfigurierbaren Exporter, den intuitiven und leistungsfähigen Cult3D-Designer und die exzellente Dokumentation in kurzer Zeit einfache und hochwertige Applikationen im E-Commerce - Bereich erstellt werden können. Für andere Anwendungsbereiche ist die Technologie noch nicht eingesetzt worden und ist für diese auch nicht konzipiert. Für die Visualisierung der Funktionalität der Produkte stehen entsprechende vorgefertigte Verhaltensweisen zur Verfügung.

[Cult3D]

3.3.4 Pulse

Pulse [Pulse] ist „Das Format“ für Charakteranimationen im WWW. Für den Bereich Produktpräsentation, welcher auch als Anwendungsgebiet von Pulse genannt wird, ist das Format aufgrund der nicht mehr zeitgemäßen Darstellungsqualität nicht geeignet.

3.3.4.1 Erstellung

Werkzeuge
Der Pulse Producer ist ein 3ds max - Exporter, der die Szenenbeschreibung in das *.BLD-Format exportiert. Diese referenziert auf die Geometrie (*.MDL), Animationen (*.BVR) und Texturen (*.JPG, *.TGA). Die szenengraphbasierten BLD-, MDL- und BVR-Dateien liegen in einem nichtspezifizierten Klartextformat vor und dienen gleichzeitig als Format für die Speicherung des Projektes.
Für die Modellierung der Pulse-Applikation wird der Pulse Creator verwendet. Dieses komfortable und gut strukturierte Modellierungsprogramm kann neben dem *.BLD-Format auch DXF- und VRML97-Dateien importieren. Mit Hilfe dieser Daten wird dann eine Szene mit Animationen, Interaktionen, Beleuchtung, Sound und Kameraeinstellung erstellt.

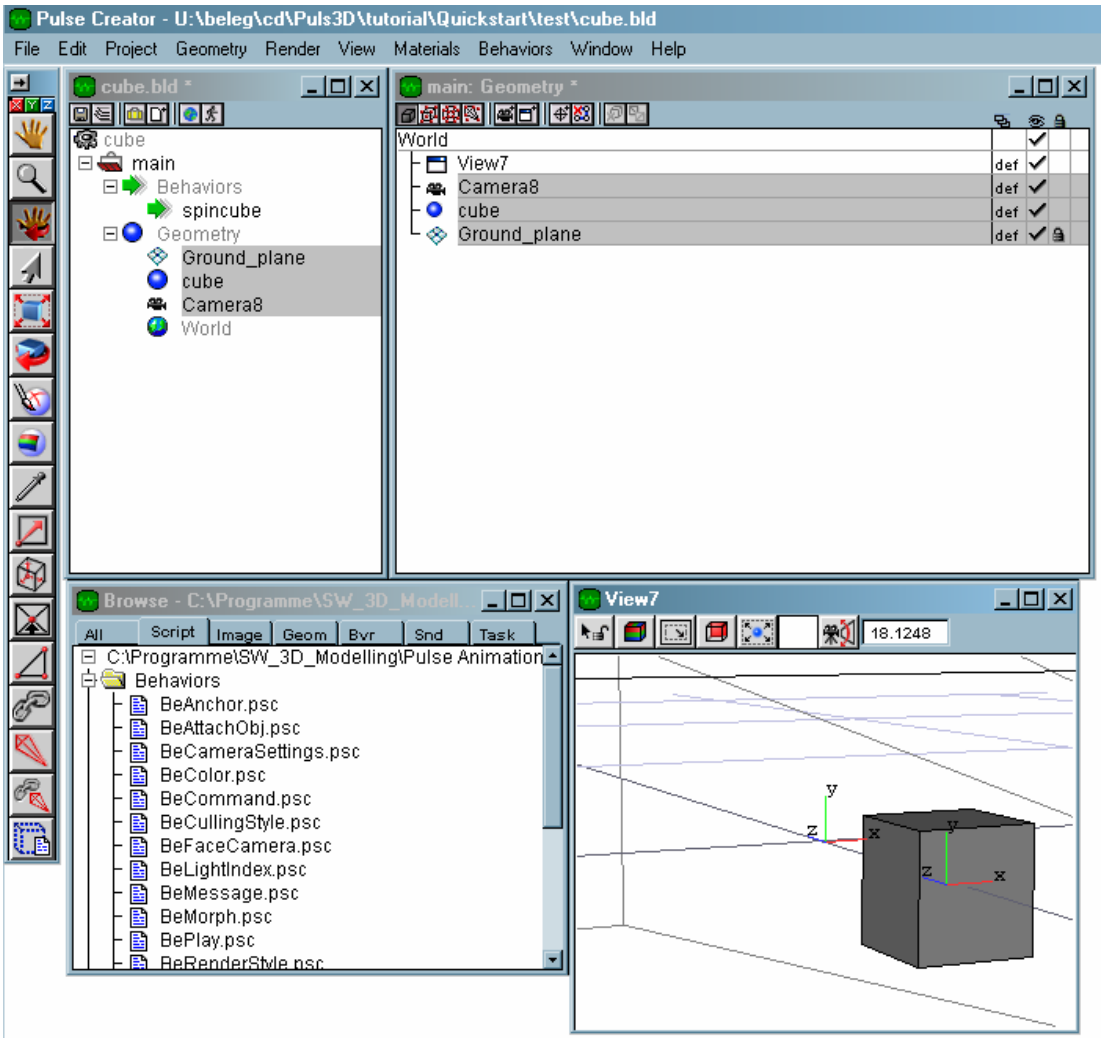

Abbildung 35: Pulse Creator

Dateiformat
Die Publikation der Szene erfolgt in den binären, verschlüsselten und komprimierten Formaten *.PWC und *.PWS. Durch die Verschlüsselung soll eine unerlaubte Weiterverwendung der Szenendaten verhindert werden. Die Komprimierung der Geometriedaten erfolgt in einem nicht publizierten Verfahren. Für die Texturkompression kann neben der Verwendung von JPEG auch eine Wavlet-Kompression eingesetzt werden. Für Soundinformationen, die als WAV- oder AIFF-Dateien importiert wurden, wird die aus dem Mobilfunkbereich bekannte GSM-Kompression benutzt, welche eine Datenrate von 20-25 KBit/Sekunde erzeugt. Bei dem Import von MP3-Dateien wird dieses Format beibehalten.
PWC-Dateien enthalten beispielsweise die Szenenbeschreibung, Geometrien und Texturen. In den PWS-Dateien sind alle Daten enthalten, die gestreamt werden sollen. Dies betrifft insbesondere die Audio- und Verhaltensinformationen, wodurch Charakteranimationen, die über einen langen Zeitraum ablaufen, ohne eine hohe Startup-Latenz realisiert werden können. Für den Fall, dass Unregelmäßigkeiten beim Streaming auftreten, wird bei dem Player ein Puffer von 10 Animations-Sekunden eingerichtet.
Durch die Aufteilung der Szene in verschiedene *Packages* erfolgt die automatische Unterteilung der Szene in verschiedene PWC- und PWS-Dateien. Hierdurch kann eine komfortable Aufteilung der Web3D-Applikation in einzelne Bestandteile erfolgen, die erst bei Bedarf übertragen werden.

Internes Modell und interne Struktur
Die szenengraphbasierten BLD-, MDL- und BVR-Dateien liegen in einem nichtspezifizierten Klartextformat vor.

Ereignismodell
Den einzelnen Objekten der Szene werden *Tasks* zugeordnet, welche dann für die Ereignisbehandlung zuständig sind. Im Pulse Creator existieren ca. 100 dieser *Tasks*, die primär Charakteranimationen und Produktpräsentationen ermöglichen. Eigene Verhaltensweisen können mit Hilfe von PulseScript erstellt werden.

Integration von Skriptsprachen
Die objektorientierte, proprietäre und Java-ähnliche Skriptsprache PulseScript (*.PSC) und die Vorgängerskriptsprache Joescript (*.JOE) können in die Szene eingebunden werden. Hierdurch wird der Zugriff auf nahezu alle Szenenbestandteile ermöglicht, wodurch Verhaltensweisen realisiert werden können. Weiterhin ist es möglich JavaScript-Methoden in der umgebenden HTML-Seite anzusprechen.
Im Pulse Creator sind schon einige vorgefertigte Skripte *(Behavior, Camera, Character, Poly, Task, World)* enthalten. Die Veränderung dieser oder das Erstellen eigener Skripte kann mit Hilfe des enthaltenen Texteditors Pulse Script Editor und einem entsprechenden Debugger erfolgen. PulseScript befindet sich zur Zeit noch im Beta-Stadium.

Medienintegration
Für Texturen *(Transparent and Greyscale Lightmaps, Bump-Mapping*, Transparenzen) können viele verschiedene Formate (*.PIC, *.PCT, *.BMP, *.JPG, *.TGA) verwendet werden. Die Audioinformationen (*.AIF, *.WAV, *.MP3, *.MID, *.AVI) werden nur der Szene im Allgemeinen zugeordnet und nicht einem einzelnen Objekt im Raum. Für das Einbinden von Sprachinformationen, für die eine Lippensynchronität erreicht werden soll, steht eine Importfunktion für das *.PHN-Format zur Verfügung.

Features
Kollisionserkennung, *Morphing*, Inverse Kinematik, *Bones*, Key-Frame - Animationen, *Dynamic Lights, Lightmaps, Adaptive Subdivision Smoothing*

3.3.4.2 Repräsentation

Es wird kein spezieller Webserver benötigt.

3.3.4.3 Kommunikation

Streaming
Die initiale Szenenbeschreibung besteht immer aus einer main.pwc, die durch den *Package*-Mechanismus sehr klein gehalten werden kann. Beispielsweise können verschiedene Animationen in separaten Streaming-fähigen PWS-Dateien und die restlichen Daten in der main.pwc enthalten sein, wodurch die Start-Up - Latenz verringert werden kann.

Es können Audio- und Verhaltensinformationen durch HTTP-Streaming übertragen werden. Durch den Pulse Creator ist es auch möglich allen anderen Szenenbestandteilen Streaming-Fähigkeiten zuzuordnen, was aber laut Dokumentation eigentlich nicht möglich sein soll.

3.3.4.4 Präsentation

Player
Der Pulse Player ist ein sehr kompaktes Webbrowser - Plug-In, welches auch auf leistungsschwachen Clients eine echtzeitfähige Interaktion mit der Szene erlaubt. Es existiert ein, durch den Nutzer in seiner Größe definierbarer, Cache für die Bestandteile der Szene. Welche Daten im Cache abgelegt werden sollen, kann durch den Autor innerhalb des Pulse Creators festgelegt werden. Die Verdrängung der Daten des Cache erfolgt nach einem modifizierten LRU-Verfahren *(Least Recently Used)*.
Der Cache wird nach dem Beenden der Pulse-Applikation nicht geleert. Dies ist insbesondere für Charakteranimationen interessant, da bei Mehrfachverwendung des Charakters in verschiedenen Anwendungen dessen Daten noch im Cache vorhanden sind und somit nur noch die entsprechenden Verhaltensweisen und Sprachinformationen gestreamt werden müssen.
Es ist auch möglich Pulse-Applikationen mit Hilfe des RealPlayer oder des QuickTime-Players abzuspielen.

Softwaretechnologischer Zugriff
Durch JavaScript können Methoden der in der Szene enthaltenen PulseScripte aufgerufen werden. Durch diese unbeschränkte Schnittstelle können komplexe Funktionalitäten realisiert werden, die nur durch den Sprachumfang der benutzten Skriptsprachen beschränkt sind.

Skalierung
Der PulsePlayer passt die Framerate der aktuellen Animation an die Möglichkeiten des Clients an. Für das Hauptanwendungsgebiet von Pulse ist es beispielsweise so, dass bei einem leistungsschwachen Client die Framerate der Charakteranimationen verringert wird, wodurch dann eine unterbrechungsfreie Sprachausgabe gewährleistet wird.
Weiterhin wird das SDS-Verfahren (siehe Gliederungspunkt 3.3.1.4 „Präsentation“) angewandt.

3.3.4.5 Bewertung

Einsatzgebiet
Pulse ist insbesondere für Charakteranimationen geeignet, da natürliche Bewegungen *(Skeletal Deformation)*, animierte Mimik, realisitische Augenbewegungen und audiosynchrone Lippenbewegungen unterstützt werden. Weiterhin können die Audio- und Verhaltensinformationen gestreamt werden, wodurch auch komplexe Charakteranimationen realisiert werden können.

Nutzungsbedingungen
Die Software für die Erstellung von Cult3D-Applikationen kostet 3.000 US$ pro Jahr. In diesem Betrag ist auch eine Lizenz für die Bereitstellung von Pulse-Anwendungen enthalten. Es existiert auch eine kostenlose Lizenz, wobei aber ein entsprechender Hinweis innerhalb des Players erfolgt.

Fazit
Pulse ist die zur Zeit beste Lösung für die Realisierung von Charakteranimationen. Es existieren mit dem Pulse Producer und dem Pulse Creator ausgereifte und komfortable Werkzeuge. Nachteilig hierbei ist aber die Realisierung der Interaktionen durch die proprietäre Skriptsprache PulseScript. Hierdurch wird ein sehr hoher Einarbeitungsbedarf für die Realisierung eigener Verhaltensweisen bedingt. Eine Lösung, die auf einer verbreiteten Sprache wie Java und einer entsprechenden API basiert, wäre PulseScript auf jeden Fall vorzuziehen. Weiterhin existieren eine exzellente Dokumentation und umfangreiche Tutorials für die Pulse-Technologie.
Das größte Problem ist aber die nicht mehr zeitgemäße Darstellungsqualität von Pulse-Applikationen. Hierdurch ist das Format für Produktpräsentationen nicht mehr geeignet und behauptet seine Vormachtstellung im Bereich der Charakteranimationen nur durch Features wie Lippensynchronisation, Inverse Kinematik und entsprechenden Streaming- und Caching-Fähigkeiten.

[Pulse]

3.3.5 Weitere Formate

Java3D
Java3D [Java3D] ist eine szenengraphbasierte 3D-Grafik-API für die objektorientierte Programmiersprache Java. Das Konzept dieser Schnittstelle ist stark an VRML97 orientiert. Die mit Hilfe dieser API erstellten Applets können in eine HTML-Seite integriert und angezeigt werden. Trotzdem ist Java3D nicht als Web3D-Technologie anzusehen. Die Gründe hierfür sind, dass

- für die Betrachtung eines Java-Applets, welches mit Hilfe der Java3D-API erstellt wurde, eine entsprechende Laufzeitumgebung vorhanden sein muss, was bei vielen Nutzern nicht der Fall ist.
- die Erstellung einer Web3D-Applikation durch die Verwendung von Java zur Szenenbeschreibung sehr aufwendig ist.
- Java3D nicht als eine Technologie für Web3D-Applikationen von Sun konzipiert wurde.

Java3D besitzt kein eigenes Dateiformat, sondern entsprechende Importmechanismen `(runtime-loader)` für existierende Formate wie beispielsweise VRML97 oder das Alias|Wavefront *.OBJ-Dateiformat. Diese Funktionalität wird auch für die Implementation des Xj3D-Players [Xj3D] benutzt. Java3D wird im Zusammenhang mit Web3D eher als Rendering-Engine verwendet, die ein 3D-Format interpretiert und anzeigt. Durch die Benutzung von OpenGL und DirectX durch die Java3D-API können auch komplexe Szenen durch Java dargestellt werden.

[Java3D, Braig00, Scr01, Stün00]

3Space
Die 3Space-Technologie [3Space] von TGS besteht aus dem Webbrowser - Plug-In 3Space Player, welcher die XML-basierte Szenenbeschreibung interpretiert. Für die zugrunde liegende XML-Grammatik ist ein entsprechendes XML-Schema vorhanden, das als 3Space SDL bezeichnet wird. Die Szenenbeschreibung kann auf Geometrieinformationen, die im kompakten *.ZAP-Format vorliegen, referenzieren und diese somit in die Szene integrieren. Dieses Format kann beispielsweise durch das Modellierungsprogramm AMAPI 3D [AMAPI 3D] erzeugt werden.
Die Besonderheit an dem *.ZAP-Format ist, dass die Oberflächen der Objekte durch bikubische Parameteroberflächen (NURBS) beschrieben werden und bei der Abspeicherung der Objekte deren Entstehungshorizont betrachtet wird *(Reconstructive Geometry)*.
Leider befindet sich die 3Space-Technologie noch im Entwicklungsstadium, was an der fehlenden Dokumentation und den bisher realisierten Anwendungsbeispielen zu erkennen ist. Man sollte sich aber aufgrund der Verwendung eines XML-Schemas für die Grammatikbeschreibung und der Verwendung von NURBS zur Oberflächenrealisierung von Objekten weiterhin mit dieser Technologie beschäftigen.

[3Space, Scr01]

Es existieren noch weitere proprietäre Web3D-Technologien wie beispielsweise:

- 3DAnywhere [3DAnywhere], bei dem die Anzeige der Szene durch ein Java-Applet erfolgt und die Szenenerstellung durch das Modellierungsprogramm Designer realisiert wird. Diese kompakte Technologie ist insbesondere für hochwertige, Plug-In - freie Produkt-präsentationen geeignet.
- Open SPX [OpenSPX], zur Realisierung von kompakten Produktpräsentationen. Das Format soll zur WWW-Präsentation von Objekten dienen, die mit den 3D-Scanner-Lösungen dieser Firma erstellt wurden.
- Die Firma Janet [Janet] ermöglicht die Visualisierung von 3D-Szenen durch ein 22 KByte großes Java-Applet. Es werden insbesondere Lösungen zur Visualisierung und Konfiguration von komplexen Maschinen und Anlagen und zur Produktvisualisierung angeboten.

3.4 Übersicht

	Dateiformat								Medienintegration				Streaming	Präsentation									
	SDL	**Szenengraph**	**Klartextformat**	**Binärformat**	**XML**	**Dateierweiterung**	**Standardisiert**	**Integration von Skript- und Programmier-sprachen**	**Audio**	**Video**	**Bilder (Texturen)**	**Weitere Medien**		**Plug-In**	**Applet**	**Netscape Navigator**	**Internet Explorer**	**Windows**	**Macintosh**	**Linux/Unix**	**Solaris**	**Skalierung**	**Zugriff**
VRML97	X	X	X			VRML, WRL	ISO/IEC - Standard (1997)	JavaScript Java	WAV, (MIDI)	MPEG 1	JPEG, GIF, (PNG), (CGM)		(xx)	0,96 - 3,2 MByte	(x)	X	X	X	X	(x)	(x)	(xx)	(x)
X3D	X	X	X	V	X	(XML)	Angestrebt	JavaScript Java	(J)	(J)	(J)		V	NI	NI							V	NI
MPEG-4	X	X	X	X		MP4	ISO/IEC - Standard (1999)	JavaScript (Java)	NI	NI	NI		NI	NI	NI							NI	NI
Shout3D	X	X	X			WRL, S3D, S3Z		Java	WAV, AU		JPEG, GIF				150 KB	X	X	X	X	X	X		X
blaxxun3D	X	X	X			WRL, BX3D		Java	WAV		JPEG, GIF		(xx)		55 KB	X	X	X	X	X	X	(xx)	X
Shockwave 3D	oo	X	X	X		W3D, DIR, DCR		Lingo	WAV, MP3, AIFF, SWA	AVI, iPIX, QuickTime	GIF, BMP		X	3,4 MByte		X	X	X	X			X	X
Viewpoint	X	X	X	X	X	MTX, MTZ, MTS		JavaScript	WAV		JPEG, BMP, (Trixels NT)	QuickTime VR, Flash	X	0,34-1,85 MByte		X	X	X	X			X	X
Cult3D	(?)	(?)		X		C3D, C3P, CO		Java	WAV, MIDI		(JPEG, GIF)		(x)	1,09-1,2 MByte		X	X	X	X	X	X	(x)	X
Pulse	X	X	X	X		BLD, MDL, BVR, PWC, PWS		PulseScript Joescript	WAV, MP3, AIF, MID, AVI, PHN		PIC, PCT, BMP, JPG, TGA		X	370 KByte		X	X	X	X			X	X

Tabelle 6: Übersicht über ausgewählte Eigenschaften der behandelten Web3D-Formate

Legende: **X**- ist vorhanden, wird unterstützt; **(x)** - teilweise unterstützt, indirekt vorhanden; **(xx)** - sehr eingeschränkt vorhanden; **(?)** - unbekannt, nicht dokumentiert, nicht zugänglich; **(J)** - Ja, wie VRML97 aber Änderungen möglich; **NI** - noch nicht implementiert; **V**- vorgesehen; **oo**- objektorientierte/imperative Programmierung

3.5 Kompatibilitätsdiagramm

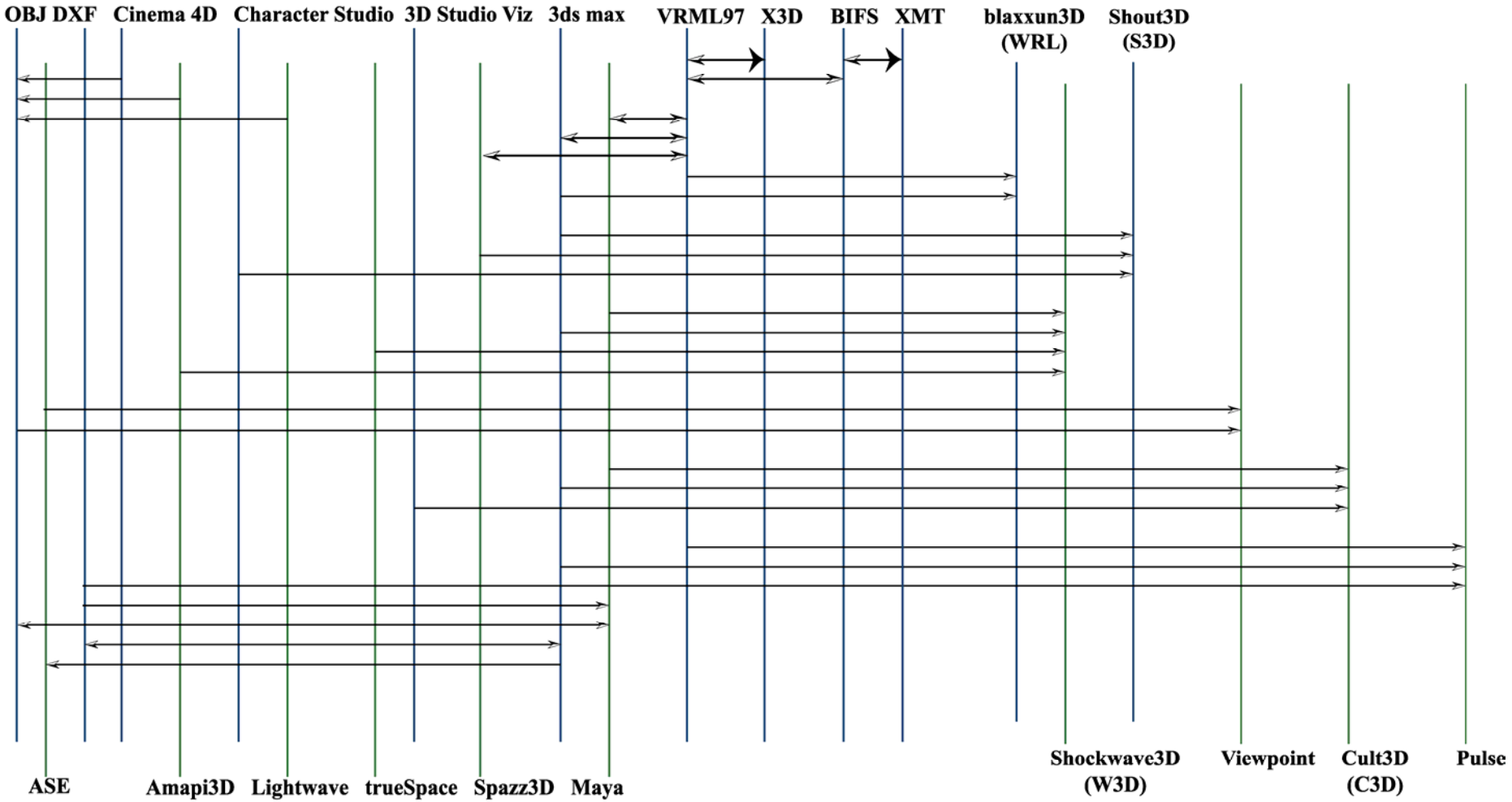

Abbildung 36: Kompatibilitätsdiagramm

In diesem Diagramm erfolgt die Darstellung der Kompatibilitätsbeziehungen zwischen 3D-Grafikformaten, 3D-Modellierungsprogrammen und den Web3D-Formaten. Es erfolgt keine exakte Darstellung, wie die Konvertierung einer Szene von einem Programm oder Format in ein anderes Programm oder Format erfolgt. Meist erfolgt eine diesbezügliche Erklärung bei der entsprechenden Beschreibung der Web3D-Formate.
Ein vollständig ausgefüllter Pfeil wurde nur bei verlustlosen Konvertierungsvorgängen benutzt. In allen anderen Fällen müssten die Einschränkungen, die sich durch die Konvertierung ergeben, evaluiert werden.

Auf der linken Seite des Kompatibilitätsdiagramms befinden sich drei 3D-Grafikformate OBJ, DXF und ASE, die im Zusammenhang mit den Web3D-Formaten als Austauschformate fungieren. Anschließend erfolgt die Aufzählung der wichtigsten 3D-Modellierungsprogramme. Danach werden die behandelten Web3D-Formate entsprechend der Reihenfolge ihrer Nennung in dieser Arbeit aufgeführt.

3.6 Zusammenfassung

Das Nachfolgeformat von VRML97 – X3D – wird noch einige Zeit bis zur Standardisierung und dem Erscheinen entsprechender marktreifer Player und Werkzeuge benötigen. Die auf VRML97 aufbauende Szenenbeschreibungssprache BIFS in MPEG-4 ist sehr umfangreich und bis jetzt noch nicht praktisch realisiert worden und wird dies wahrscheinlich auch nie werden.
Sehr interessant sind die aktuellen Bestrebungen bezüglich der Ersetzung von BIFS und XMT durch X3D. Durch eine diesbezügliche Zusammenarbeit zwischen dem Web3D-Konsortium und der Moving Picture Experts Group könnten Synergieeffekte auftreten, die zu einer beschleunigten Entwicklung von X3D führen. Da X3D nicht nur der Nachfolger von VRML97, sondern auch die Grundlage für interaktive 3D-Grafik in MPEG-4 darstellen wird, erhält dieses Format exzellente Zukunftsaussichten.
Es bleibt abzuwarten, wie die neu formierte Browser Working Group innerhalb des Web3D-Konsortiums auf die weitere Entwicklung von X3D einwirken wird. Nach den ersten Pressemitteilungen zu urteilen, soll die Weiterentwicklung des Formates durch diese Arbeitgruppe beschleunigt werden, was durch eine starke Anlehnung an VRML97 erreicht werden soll.
Es bleibt zu hoffen, dass die Bestrebungen der X3D Working Group bezüglich der Favorisierung von XML und der Überarbeitung der Szenenbeschreibung auf der Basis der Erfahrungen mit VRML97 zu entsprechenden Ergebnissen in dem zukünftigen Standard und den entsprechenden Implementierungen führen.

Shout3D und blaxxun3D sind als Zwischenschritte der Entwicklung von VRML97 zu X3D anzusehen und haben demonstriert, dass auch ein Applet als Player geeignet sein kann.

Das neue Format Shockwave3D könnte der Web3D-Grafik aufgrund der enormen Marktdurchdringung des Shockwave-Players zum Durchbruch verhelfen. Durch die Kombination aus neuen Technologien, exzellenten Autorenwerkzeugen und einer entsprechenden Dokumentation können insbesondere anspruchsvolle Spiele - und E-Commerce - Applikationen entwickelt werden. Da die Web3D-Funktionalitäten in Shockwave noch relativ neu sind, bleibt abzuwarten wie sich diese Technologie mit weiteren Produktversionen verbessert und sich somit zum direkten Konkurrenten von X3D entwickeln könnte.

Viewpoint ist für die Realisierung von hochwertigen Produktpräsentationen prädestiniert. Durch zusätzliche Funktionalitäten – wie beispielsweise die Möglichkeiten zum Nutzer-Tracking und der Einbindung von Flash-Animationen oder der *ZoomView*-Funktionalität – verschafft sich Viewpoint große Vorteile gegenüber anderen Formaten.

Cult3D ist ein weiteres Format, das auf den Anwendungsbereich E-Commerce spezialisiert ist. Als Vorteil gegenüber Shockwave3D oder Viewpoint ist das intuitive Autorenwerkzeug anzusehen, welches insbesondere bei einfachen Applikationen zu einem geringen Einarbeitungs- und Entwicklungsaufwand führt. Andererseits ist der Cult3D-Player nicht im entferntesten so weit verbreitet, wie der Shockwave3D-Player und das Format besitzt nicht die gleiche Mächtigkeit wie Shockwave3D.

Pulse ist für die Realisierung von Charakter-Animationen prädestiniert. Das Format könnte aber aufgrund der nicht mehr zeitgemäßen Darstellungsqualität seine Führung auf diesem Gebiet verlieren.

Grundlegend kann gesagt werden, dass die nichtstandardisierten, proprietären Formate sich meist auf wenige Anwendungsgebiete spezialisiert haben und dort teilweise über exzellente Autorenwerkzeuge und innovative Detaillösungen verfügen. Mittels dieser Technologien ist es aber meist nicht – wie bei den standardisierten Formaten – auf einfache Weise möglich, jede beliebige Web3D-Applikation zu realisieren.

4 Gedanken zum CONTIGRA-Projekt

Das CONTIGRA-Projekt (COmponent-orieNted Three-dimensional Interactive GRaphical Applications) [Contigra] beschäftigt sich mit der Unterstützung der Erstellung von interaktiven Web3D-Applikationen. Mit Hilfe von implementationsunabhängigen 3D-Komponenten soll ein einfaches, deklaratives und interdisziplinäres Erstellen dieser Anwendungen möglich sein.

Player
Für die Präsentation einer CONTIGRA-Szene gibt es zwei verschiedene Möglichkeiten. Einerseits könnte ein eigener CONTIGRA-Player geschrieben werden und andererseits könnte eine Konvertierung in ein bestehendes Format erfolgen. Da die CONTIGRA-Grammatiken sich stark an X3D orientieren, wird dieses auch das primäre Zielformat sein.
Die erste Möglichkeit ist zu aufwendig, da die Implementation eines eigenen Players sehr aufwendig ist. Außerdem müsste man auch aktuelle Technologien wie beispielsweise Streaming, Komprimierung und Skalierung berücksichtigen und ein eigenes Client/Server-Konzept realisieren.
Der andere Weg besteht darin, existierende Formate und somit deren Player zu benutzen. Dafür müsste eine Konvertierung in das entsprechende Format erfolgen. Dies ist prinzipiell nur dann komfortabel möglich, wenn das Zielformat szenengraphbasiert ist und eine entsprechende Dokumentation existiert.

Konvertierung in andere Formate
Eine Konvertierung in ein Format, welches grundlegende Konzepte von X3D und somit CONTIGRA nicht unterstützt, ist nicht trivial. Bisherige Erfahrungen zeigen, dass entsprechende Formatumwandlungen zwischen zwei Formaten mit unterschiedlichen Konzepten meist nur suboptimale Ergebnisse erzielen.
Falls die Codierung eines Formates in binärer Form erfolgt und es somit nur durch entsprechende Werkzeuge erstellt werden kann, ergeben sich weitere Probleme, da die Schnittstellen zu dem internen Werkzeug-Format meist nicht offen gelegt sind.
Ein weiteres Problem ist die unterschiedliche Ausprägung der Medienverwaltung. Die Medien werden beispielsweise bei Viewpoint in dem binären MTS-Format abgelegt. Die Erzeugung dieses Formates bei der Konvertierung einer CONTIGRA-Szene in eine Viewpoint-Applikation wäre relativ problematisch.
Durch die Skriptsprache Lingo und die vielen Erweiterungsmöglichkeiten des ausgereiften Autorenwerkzeuges Macromedia Director ist der Import einer X3D-Szene beispielsweise in Form eines Darstellers oder als Lingo-Skript denkbar. Für Ersteres existiert auch schon eine prototypische Implementierung [X3D File Converter/Importer].
Für VRML97, X3D, MPEG-4/BIFS, XMT, Shout3D, blaxxun3D und Viewpoint ist eine Konvertierung einer CONTIGRA-Szene prinzipiell am besten möglich, da die entsprechenden Werkzeuge außer Acht gelassen werden können.
Die Formate Shout3D und blaxxun3D sind erweiterte Untermengen von VRML97, die bisher nur sehr wenig für die Realisation von professionellen Anwendungen genutzt wurden und deshalb eher als unbedeutend anzusehen sind.
Letztendlich kann gesagt werden, dass Cult3D und Shockwave3D als Zielformat für CONTIGRA relativ ungeeignet ist, die Konvertierung in Viewpoint und Pulse möglich ist, dass Shout3D und blaxxun3D relativ unbedeutend sind und dass die Entwicklungen bei VRML97, X3D, MPEG-4/BIFS und XMT auf **ein** Format hinführen, welches als Zielformat -prädestiniert ist.

5 Vertiefende praktische Aufgabe

Der Gegenstand der vertiefenden praktischen Aufgabe ist die Implementation der bereits in VRML97 realisierten *Collapsible Cylindrical Trees* (CCT) – einer schnellen hierarchischen Visualisierungs- und Navigationstechnik für mittelgroße hierarchische Strukturen – in Shockwave3D. [Dach01]
Eine solche baumähnliche Anordnung ist oft bei Sitemaps zu finden, die einen Überblick über die Struktur einer Webseite geben sollen und wurde deshalb auch von Dachselt et al. als Grundlage für erste Implementationen genutzt [Dach01].

Das Format Shockwave3D wurde deshalb für die Implementierung gewählt, da durch die Verwendung von Vorschlägen für Standards oder standardisierter Formate entweder keine neuen Informationen aufgrund der Ähnlichkeit zu VRML97 gewonnen werden könnten oder diese hierfür noch kein adäquates Entwicklungsstadium erreicht haben. Bei den proprietären Formaten wurde Viewpoint aufgrund seiner mangelhaften Dokumentation nicht verwendet. Bei Pulse und Cult3D wird ein erheblicher Erstellungsaufwand vermutet, da diese in ihrem Funktionsumfang auf die Bereiche Produktpräsentationen und Charakteranimationen konzentriert sind, wozu CCT nicht gehört. Weiterhin ist von Shockwave3D zu erwarten, dass es in kurzer Zeit durch seine Funktionalität und durch seine starken Partner eine relativ starke Marktdurchdringung erfährt. Deshalb sind Erfahrungen mit diesem Format von besonderem Interesse.

Die nachfolgenden Erläuterungen sollen einerseits zur Dokumentation der implementierten Applikation und andererseits als Einblick in die Scriptsprache Lingo des Autorenwerkzeuges Macromedia Director (MD) dienen. Für die Bewertung von Shockwave3D wurde VRML97 herangezogen, da der Autor mit diesem Format die meisten Erfahrungen besitzt und auch das entsprechende Nachfolgeformat X3D ein direkter Konkurrent von Shockwave3D sein wird.

Neben der obligatorischen Speicherung der Projektdaten auf der beiliegenden CD, erfolgt die Veröffentlichung der implementierten Applikation auf der Webseite http://www.rukzio.de/Web3D/results.html (Login: Web3D, Passwort: CCT310801). Die verschiedenen Versionen der Applikation, die im Text erwähnt werden, befinden sich sowohl auf der CD als auch auf der Webseite.

Einführung in die Navigationstechnik CCT
Gegeben sei eine baumartige Datenstruktur, die durch eine XML-Datei repräsentiert wird. In diesem Baum existieren verschiedene Knoten mit Attributen, die Informationen zum Text, der Farbe und einer dazugehörigen Aktion liefern. Jeder Eltern-Knoten p besitzt ein Set von Kindern C. Die Tiefe des Baumes ist d und l die Ebene in der Hierarchie. Der Wurzel-Knoten wird mit $l=0$ bezeichnet und die Kinder dieses Knotens mit $l=1$.

Beim Start der Applikation präsentieren sich dem Nutzer horizontal angeordnete drehbare Zylinder, welche die erste Stufe *(l=1)* der Hierarchie darstellen und auf denen die entsprechenden Kinder C in Form von horizontal ausgerichteten Streifen enthalten sind. Diese realisieren entweder Verweise auf die nächste Ebene *(l=2)* oder enthalten einen Hyperlink. Das Aufrufen eines Webbrowsers mit dem entsprechenden Hyperlink wird durch einen Maus-Klick ausgelöst. Die Navigation zu der tiefer gelegenen Hierarchiestufe erfolgt durch einen *MouseOver*-Effekt. Diese Ebene wird mit Hilfe eines Teleskop-Effektes visualisiert. Hierbei werden die Zylinder der Stufe $l=1$ so skaliert und nach rechts oder links verschoben, dass

genügend Platz für den neuen Zylinder der Stufe *l=2* existiert. Auf diesem neuen Zylinder können wiederum Hyperlinks oder Verweise für eine niedrigere Ebene *l=3* vorhanden sein.
Mit Hilfe dieser Visualisierungs- und Navigationstechnik soll eine Balance zwischen Detailinformationen und dem Überblick auf die Gesamtstruktur gefunden werden. Deshalb erfolgt eine Skalierung der Zylinder der Stufe *l=1* und nicht deren bloße Richtungstransformation nach links oder rechts. Dies würde nämlich zu einem Verschwinden dieser Zylinder aus dem sichtbaren Bereich der Applikation führen.

5.1 Implementation

Die Grundlage für die Datenstruktur der Implementation stellt eine XML-Datei dar, in der die baumartige Navigationsstruktur abgelegt ist.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE LINK SYSTEM "../import.dtd">
<LINK color="0.6719 0.0625 0.0937" name="MMT" target="http://www-mmt.inf.tu-dresden.de/home">
  <LINK color="0.6719 0.0625 0.0937" name="Home"  target="http://www-mmt.inf.tu-dresden.de/home">
        <LINK color="0.6719 0.0625 0.0937" name="Lageplan" target="http://www-mmt.inf.tu-dresden.de/home/Lage/index.htm"/>
        <LINK color="0.6719 0.0625 0.0937" name="Offene Stellen"  target="http://www-mmt.inf.tu-dresden.de/home/">
            <LINK color="0.6719 0.0625 0.0937" name="Wissl. Mitarbeiter">
                <LINK color="0.6719 0.0625 0.0937" name="Virtuelle Gemeinschaft"
                        target="http://www-mmt.inf.tu-dresden.de/home/stellen_II_1.htm"/>
                <LINK color="0.6719 0.0625 0.0937" name="Intell. Benutzerschittstellen"
                        target="http://www-mmt.inf.tu-dresden.de/home/stellen_II_2.htm"/>
            </LINK>
            <LINK color="0.6719 0.0625 0.0937" name="Doktorandenstipendium"
                    target="http://www-mmt.inf.tu-dresden.de/home/Graduiertenkolleg.htm"/>
            <LINK color="0.6719 0.0625 0.0937" name="Stud. Hilfskraft CONTIGRA"
                    target="http://www-mmt.inf.tu-dresden.de/home/HiWi_2001.htm"/>
            <LINK color="0.6719 0.0625 0.0937" name="Stud. Hilfskraft CHAMELEON"
                    target="http://www-mmt.inf.tu-dresden.de/home/HiWi_2001_2.htm"/>
        </LINK>
        <LINK color="0.6719 0.0625 0.0937" name="Diplom- &amp;amp; Belegarbeiten"
                target="http://www-mmt.inf.tu-dresden.de/lehre/neue_Themen/"/>
  </LINK>
</LINK>
```

Auszug aus der XML-Datei

Für das Verständnis der nachfolgenden Angaben sind die Informationen des Gliederungspunktes 3.3.1 „Shockwave3D“ unerlässlich. Die Implementation und der Test der Applikation erfolgte unter Verwendung der 3D-Grafik-API DirectX auf einer hochwertigen Grafikworkstation, einer HP Visualise (X-Class).

Es existiert ein *Parent-Script* `ClassCylinder` welches die Realisation der Darstellung und der Verhaltensweisen jedes einzelnen Zylinders übernimmt. Diese Klasse wird von den Prozeduren

- `startMovie` - Initialisierung der Applikation
- `createCylinder` - Dynamische Generierung neuer Zylinder

genutzt, welche wiederum mit den Prozeduren

- `createParseList` – Rekursive Analyse der XML-Datei
- `exitFrame` – Starten von Verhaltensweisen, Realisation dynamischer Aspekte
- `mouseDown` – Realisierung der Hyperlink-Funktionalität
- `createImage` – Erstellung einer Bild-Textur
- `transformColor` – Umwandlung von VRML97-Farbangaben in RGB-Werte
- `stopMovie` – Beenden des Filmes
- `cleanup` – Löschen von Modellen, Modellressourcen, *Shadern* und Texturen

interagieren.

Dies Klasse `ClassCylinder` enthält neben den Funktionalitäten zum Anlegen eines neuen Zylinders folgende Prozeduren:

- `addHead` – Hinzufügen der Beschriftung über den Zylindern der Stufe *l=1*
- `addText` – Aufrufen von `createText` und das Anordnen der Text-Labels
- `createText` – Erstellen der Text-Labels
- `rotateUp, rotateDown` – Rotation des Zylinders
- `OScale` – Skalieren des Zylinders
- `toTheLeft, toTheRight` – horizontales Verschieben des Zylinders
- `ODelete` – Löschen des Zylinders

Start der Applikation

Abbildung 37: Realisierung der Visualisierungstechnik CCT mittels Shockwave3D (Version 3a)

Nach dem Start der Applikation erfolgt zunächst das Anlegen der Modellressourcen. Diese definieren die 3D-Geometrien, die entweder durch den Export aus einem Modellierungsprogramm oder durch Lingo erzeugt werden können. Die Modellressourcen werden von den Modellen genutzt. Diese Beziehung ist mit der von Darstellern und *Sprites* zu vergleichen. In dem folgenden Quellcode-Beispiel erfolgt das Anlegen der Hauptzylinder-Modellressource, für die auch ein *Shader* und eine entsprechende Textur definiert wird.

```
MainCylinderModelResource = TheMember.newModelResource("MainCylinderModelResource", #cylinder)
MainCylinderModelResource.height = 200
MainCylinderModelResource.bottomRadius=170
MainCylinderModelResource.topRadius=170
MainCylinderModelResource.topCap = false
MainCylinderModelResource.bottomCap = false

MainCylinderShader = TheMember.newShader("MainCylinderShader", #standard)

MainCylinderTexture = TheMember.newTexture("MainCylinderTexture" ,#fromImageObject, createImage(rgb(230,230,242)))
```

Anlegen der Modellresource für den Grundzylinder

Das Generieren des Hauptzylinder-Modells erfolgt in der Klasse `ClassCylinder.`

```
gMainCylinderModel = TheMember.newModel(gCylinderName)

gMainCylinderModel.resource = MainCylinderModelResource
gMainCylinderModel.shader = MainCylinderShader
gMainCylinderModel.shader.texture = MainCylinderTexture
gMainCylinderModel.shader.blend = 50
```

Erzeugen eines Modells mit *Shader* und Textur in der Klasse `ClassCylinder`

Für die Generierung des Dreieckes, welches das Vorhandensein von niedrigeren Hierarchiestufen signalisiert, wurde ein eigenes Dreiecksnetz als Modellressource erstellt.

```
TriangleModelResource = TheMember.newMesh("TriangleModelResource",1,3,0,3,0)

TriangleModelResource.vertexList = [vector(0,0,0), vector(10,10,0), vector(0, 20, 0)]
TriangleModelResource.face[1].vertices = [1,2,3]
TriangleModelResource.colorList = [rgb(255,255,255), rgb(255, 255, 255), rgb(255,255,255)]
TriangleModelResource.face[1].colors = [3,2,1]
TriangleModelResource.generateNormals(#smooth)
TriangleModelResource.build()
```

Generierung eines Dreiecksnetzes

Anschließend erfolgt das Parsen der XML-Datei in der die Navigationsstruktur enthalten ist. Für das Parsen steht das *XML Parser Xtra* zur Verfügung. Bei der Anzeige der Applikation muss dieses *Xtra* vorhanden sein. Falls dies nicht der Fall ist, wird es automatisch nachgeladen. Dies würde die Ladezeit erhöhen, da das *Xtra* im unkomprimierten Zustand eine Größe von 164 KByte besitzt.

```
-- Creates the one and only ParserObject
ParserObject = new(xtra "xmlparser")
-- Parse (Analyse) the XML-Data (It is a non-validate Parser)
errCode = ParserObject.parseString(member("XMLtext").text)
```

Anlegen des Parser-Objektes

Anschließend erfolgt die Generierung der Zylinder der Stufe *l=1* und das Ablegen der Parsing-Ergebnisse in eine Listenstruktur.

Verhaltensweisen der Applikation

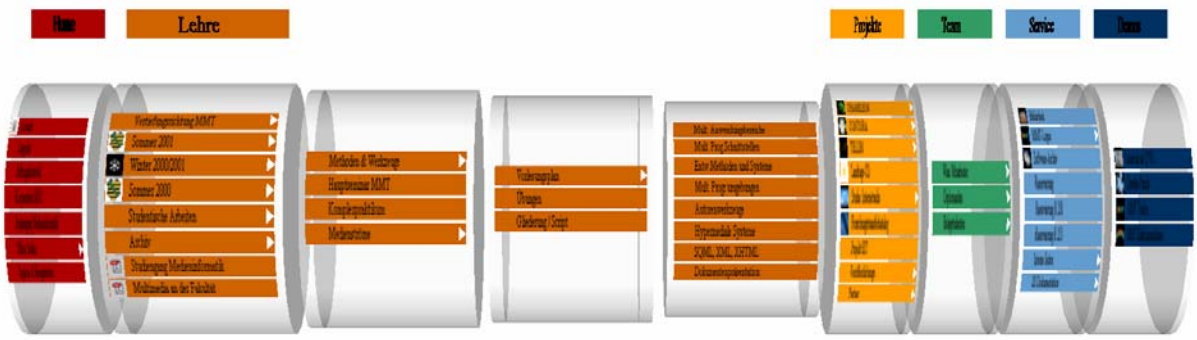

Abbildung 38: Navigation zur Stufe *l=4* (Version 3a)

Nach einem Klick auf einen URL-Eintrag wird dieser in dem Webbrowser angezeigt. Hierfür wird in VRML97 der `Anchor`-Knoten benutzt. Eine ähnliche Funktionalität steht in Lingo nicht zur Verfügung. Mittels des Befehles `modelsUnderLoc` erhält man eine Liste der Modelle, die ein fiktiver Laserstrahl durchdringt, der von der Kameraposition aus auf den Mauszeiger gerichtet ist. In dem Namen des Modells für die Darstellung des Streifens ist die URL enthalten, die durch verschiedene *String*-Operationen extrahiert wird und für den Aufruf des Webbrowsers benötigt wird.

```
on mouseDown
  tOffset = point(sprite(1).left, sprite(1).top)
  inModels = sprite(1).camera.modelsUnderLoc(the mouseLoc - tOffset)
  if not(inModels = []) then
    repeat with j=1 to inModels.count()
      if inModels[j].name contains "http://" then
        LinkEnd=offset("Link",inModels[j].name)
        goToNetPage chars(inModels[j].name,0,LinkEnd-2), "_new"
      end if
    end repeat
  end if
end mouseDown
```

Nachbilden der `Anchor`-Funktionalität und Starten des Browsers

Die Realisierung der Funktionalität des `TouchSensor`-Knotens von VRML97 – für die Navigation zwischen den verschiedenen Hierarchiestufen – ist in Lingo nicht trivial. Hierzu muss die `on exitFrame` - Prozedur genutzt werden, die beim Verlassen des aktuellen Frames ausgelöst wird. Diese Frame-Metapher ist Bestandteil der Bühne und ist für die implementierte Applikation sehr nachteilig. Um auf allen Zielsystemen eine gleiche Interaktionsgeschwindigkeit zu bestimmen steht die Sprite-Eigenschaft `targetFrameRate` zur Verfügung, die mittels der MRM-Technologie die Objekte der verfügbaren Rechenleistung anpasst und somit das Erreichen der gewünschten Framerate und der Interaktionsgeschwindigkeit garantiert. Leider ist die MRM-Technologie nur auf importierte Objekte anwendbar und nicht auf Objekte, die durch Lingo erzeugt wurden. Allgemein kann gesagt werden, dass durch Lingo generierte Modellprimitive bei der Erstellung von Verhaltensweisen stark benachteiligt werden, da für sie viele Befehle nicht verfügbar sind.

Die Funktionsweise des Mechanismus zur Anzeige neuer Zylinder in einer tieferen Hierarchiestufe bei gleichzeitiger Skalierung und Transformation der anderen Zylinder soll aufgrund der Komplexität hier nicht aufgezeigt werden.

Grundsätzlich werden nur die Zylinder-Objekte im Speicher gehalten, die in der Applikation sichtbar sind. Falls man in eine niedrigere Hierarchiestufe navigiert, wird der neue Zylinder mit Hilfe einer Referenz auf das Parser-Objekt dynamisch erzeugt. Falls ein Zylinder nicht mehr sichtbar ist, wird er aus dem Speicher entfernt und bei wiederholter Anzeige erneut erzeugt. Dieses Vorgehen wird auch von der Tatsache unterstützt, dass die Zeit für das Erstellen eines neuen Zylinders bei Verwendung von Texturen zur Textdarstellung vernachlässigt werden kann. Man könnte auch die nichtsichtbaren Zylinder-Objekte für eine spätere Wiederverwendung aufheben, was aber zu einem linearen Speicherzuwachs und somit zu einer möglichen Verlangsamung der Animations- und Interaktionsgeschwindigkeit führt.

Probleme bei der Textdarstellung

Das größte Problem bei der Implementation stellt die Darstellung von Text dar. Die übliche Erstellung von 3D-Text in Lingo beruht darauf, das man zuerst einen 2D-Textdarsteller `(TextMember)` erzeugt, ihn anschließend in einen 3D-Textdarsteller `(DMemberText)` umwandelt und mit dessen Hilfe dann ein 3D-Text-Modell `(TextLabel)` erzeugt (Versionen 1-1c).

```
TextMember = new (#text)
TextMember.fontSize=14
TextMember.tunnelDepth=1
TextMember.smoothness = 5
TextMember.name="TextMember"
TextMember.text = "aText"
DMemberText = member(TextMember).extrude3d(member("Scene"))
TextLabel = TheMember.newModel("anOtherName", DMemberText)
TextLabel.shaderList = TheMember.shader("TextShader")
TextLabel.shaderList.texture = TheMember.texture("TextTexture")
```

Erstellung eines 3D-Text-Modells

Das Problem hierbei ist, dass der Text intern durch Polygone realisiert wird, was zu einer Polygonanzahl von ca. 82.000 nach dem Start der Applikation führt. Für das Generieren eines 3D-Text-Modells wird sehr viel Zeit benötigt. Dies deutet darauf hin, dass der Algorithmus für die Erzeugung von 3D-Text nicht sehr performant ist. Die Animations- und Interaktionsgeschwindigkeit ist durch die hohe Polygonanzahl nicht akzeptabel.
Die Polygonanzahl und somit die Performance der Applikation wird entscheidend von dem Parameter `smoothness` beeinflusst, der die Anzahl der Dreiecke und somit die Detailgenauigkeit des Textes bestimmt. Falls der Wert des Parameters sehr niedrig gewählt wird, kann die Polygonanzahl bis auf ca. 29.000 verringert werden, was aber zu einer sehr schlechten Lesbarkeit des Textes führt.

Abbildung 39: Text mit den Werten 1 und 5 (hohe Polygonanzahl) für den Parameter `smoothness`

Die Alternative zur Verwendung von 3D-Text ist die Darstellung durch eine entsprechende dynamisch generierte Textur, die den Text enthält. Die entsprechend realisierte Lösung (in den Versionen 2-2a) benötigt nur noch ca. 19.000 Polygone, was die Startup-Latenz verringert und die Animations- und Interaktionsgeschwindigkeit deutlich erhöht.

```
-- 2D TextMember
TextMember = new (#text)
TextMember.fontSize=14
TextMember.smoothness = 10
TextMember.antiAlias = true
TextMember.fontStyle = [#outline]
TextMember.name="TextMember"
TextMember.text = "aText"
TextMember.text = TextMember.text
TextMember.font="Times New Roman"
TextMember.width = 150
TextMember.height= 12

-- steal the alpha chanel
myImage = image(150, 12, 4)
myImage=TextMember.image.extractAlpha()

TextLabelShader =TheMember.newShader("aShaderName", #standard)
TextLabelTexture=TheMember.newTexture("aTextureName",#fromImageObject, myImage)

TextLabel = TheMember.newModel("aTextLabelName", TheMember.ModelResource("StripeModelResource"))
TextLabel.shaderList[2] = TextLabel Shader
TextLabel.shaderList[2].textureList[1] = TheMember.Texture(chars(gCylinderName,0, 1))
TextLabel.shaderList[2].textureList[2] = TextLabelTexture
TextLabel.shaderList[2].textureList[2]= #wrapSpherical
TextLabel.shaderList[2].textureRepeatList[1] =true
TextLabel.shaderList[2].textureRepeatList[2] =false
TextLabel.shaderList[2].textureTransformList[2].scale(0.7,1,1)
```

Dynamisches Erstellen einer Text-Textur, welche sich auf einer farbigen Ebene befindet

Die Texturbehandlung weist aber auch einige Probleme auf. Beispielsweise entstehen bei dem Verschieben von Texturen ungewollte Streifen-Effekte (Version 2b).

Streaming der Icons
Die Icons werden durch den Befehl `preloadNetThing` in maximal 6 parallelen Streams geladen. Der Effekt des dynamischen Nachladens der verschiedenen Icons kann in den Versionen 3-3a sehr gut nachvollzogen werden.

5.2 Vergleich der Implementationen in VRML97 und Shockwave3D

Die Lesbarkeit des Textes stellt auch in der existierenden VRML97-Implementation ein großes Problem dar. Letztendlich kann aber gesagt werden, dass mit beiden Möglichkeiten zur Realisation von Text in Shockwave3D ein besseres Ergebnis als bei der VRML97-Implementation erzielt wurde.

Die Problematik der möglichen falschen Mausposition, die entsteht, wenn man sich beispielsweise in der Hierarchiestufe *l=4* befindet und zu der Hierarchiestufe *l=1* in der nächsten rechts gelegenen Hierarchiestufe *l=1* wechselt, wurde in den Versionen 1-1c der Applikation aufgrund der schlechten Animationsgeschwindigkeit gelöst. Der Nutzer kann nun den Skalierungs- und Translations-Prozeß verfolgen und sich ihm mittels der Bewegung der Maus anpassen.

Durch das *XML Parser Xtra* ist es möglich, die XML-Datei erst beim Start zu analysieren und daraus die Informationen für die Applikation abzuleiten.

Letztendlich kann gesagt werden, dass beide Formate für diese Anwendung gleich gut geeignet sind.

Die Vorteile von VRML97 bei dieser Applikation sind:
- der deklarative Ansatz,
- die einfache Texturenbehandlung
- der einfache Zugriff auf externe Medien (Texturen)
- das einfache Erzeugen von Animationen durch Sensoren und Interpolatoren
- die Erweiterungsmöglichkeiten durch `PROTOS`

Die Vorteile von Shockwave3D (Lingo) bei dieser Applikation sind:
- die Realisation von komplexen dynamischen Aspekten zur Laufzeit
- die Möglichkeit zum Streaming von Medienobjekten
- das Erstellung und Löschen von visuellen Objekten zur Speicherverwaltung
- die Verwendung objektorientierter Ansätze
- die höhere Textqualität

Glossar

3DNow!	ist eine Erweiterung der x86-Prozessorarchitektur von AMD, die deutlich gesteigerte Gleitkomma-Rechenleistung für dreidimensionale Grafik- und Multimedia-Anwendungen bietet. [ARCHmatic]
Animation	Eindruck eines Bewegungsablaufes, der durch das sequentielle Anzeigen oder Abspielen von Bildern mit einer Rate von 15 bis 20 Bilder pro Sekunde entsteht.
Avatar	Bezeichnung für eine fiktive Identität, die ein Teilnehmer innerhalb virtueller Welten annehmen kann. [ARCHmatic]
Bones	dienen zur Animation komplexer Objekte mit Hilfe simpler Skelette (*Bones* - engl. für "Knochen"). Am Skelett lassen sich die gewünschten Posen wie an einer Gliederpuppe einstellen - unterstützt durch die inverse Kinematik. [ARCHmatic]
Extrusion	„Ziehen“ einer zweidimensionalen Geometrie durch den dreidimensionalen Raum
Interoperabilität	sichert die Fähigkeit der Anwendungsprogramme zur verteilten Zusammenarbeit auf einer Kommunikationsinfrastruktur
Interaktive 3D-Grafik	Bildsequenzen werden zum Zeitpunkt des Abspielens in Echtzeit und in Abhängigkeit von Interaktionen des Benutzers erzeugt
Inverse Kinematik	Finger, Arme und Beine sind aus unzähligen Gliedern zusammengesetzt. Um eine Pose einzustellen (oder zu animieren), müsste man alle Körperglieder einzeln in die gewünschte Lage bringen (Vorwärts-Kinematik). Inverse Kinematik vereinfacht das: mit ihrer Hilfe verhält sich das 3D-Modell wie eine Gliederpuppe. Es genügt zum Beispiel, an einer Fingerkuppe zu ziehen, um den Finger zu strecken oder sogar den gesamten Arm. [ARCHmatic]
ISSE	*(Internet Streaming SIMD Extensions)* Erweiterung des CPU-Befehlssatzes bei Intel-Prozessoren
Lightmap	ist eine Textur, die zur Laufzeit berechnet wird und die durch die Beleuchtung resultierenden Farb- oder Helligkeitswerte beinhaltet. [GP]
NURBS	*(Non-Uniform Rational B-Spline und Surfaces)* sind mathematische Kurven oder Flächen, die beliebige Formen von einfachen 2D Linien, Bögen oder Rechtecken bis zu organischen 3D-Freiformflächen und Volumenkörper darstellen können. NURBS Modelle werden wegen ihrer Genauigkeit und der hohen Verbreitung im Grafik- und Industriedesign, in der mechanischen Konstruktion und Fertigung und bei Animationsprojekten eingesetzt. [ARCHmatic]

Morphing	Durch Überblenden einer geometrischen Struktur oder eines Bildes kann man z.B. eine Teekanne in eine Kugel verwandeln oder mit der Mimik von Gesichtern spielen. [ARCHmatic]
Portabilität	gewährleistet die Lauffähigkeit von Anwendungssoftware auf unterschiedlichen Systemen
Rendering	fasst alle bei der Umsetzung der Szenendaten zum Bild anfallenden Rechenvorgänge zusammen
Startup-Latenz	Zeitraum zwischen der Anforderung einer Web3D-Szene durch den Player und dem Zeitpunkt der ersten Interaktionsmöglichkeit
Texture Mapping	Überlagerung einer Fläche mit einem Muster (Textur) inklusive perspektivischer Korrektur, z.B. eine Holzmaserung auf einem Möbelstück oder eine Wand mit Tapete in perspektivischer Ansicht. Auch ein Video kann als Textur benutzt werden. [ARCHmatic]
Shading	Schattierung von gekrümmten Flächen, damit diese realitätsnäher aussehen. Dazu werden die gekrümmten Flächen in kleine Dreiecke aufgeteilt. Die drei wichtigsten 3D-Shading-Methoden (*Flat-Shading, Gouraud-Shading, Phong-Shading*) unterscheiden sich darin, wie genau die Farbverläufe innerhalb dieser Dreiecke dargestellt werden. [ARCHmatic]
Szene	(auch 3D-Szene, 3D-Welt) bezeichnet einen dreidimensionalen Raum, in dem 3D-Objekte angeordnet sind.
Szenengraph	hierarchische baumähnliche Struktur
Tesselation	Umwandlung eines Objektes in Polygone (meist Dreiecke), die eine Geometrie- und Rendering-Engine verarbeiten können.
Vertex	Knotenpunkt, Scheitelpunkt, Referenzpunkt Dieses geometrische Element verbindet die Kanten (*Edges)* der Gittermodelle.
Vertices	Plural von Vertex
Web3D	bezeichnet alle offenen, aber auch proprietären Technologien, die interaktive 3D-Grafiken im WWW darstellen können. [ARCHmatic]

Literaturverzeichnis

[Aba99] Vadim Abadjev ; Miguel del Rosario ; Alexei Lebedev ; Alexander Migdal ; Victor Paskhaver: Metastream. Proceedings of the fourth symposium on the virtual reality modeling language, VRML99 Paderborn, 1999

[Bow99] Doug A. Bowman, Larry Hodges: Formalizing the Design, Evaluation and Application of Interaction Techniques for Immersive Virtual Environments. In: *Journal of Visual Languages and Computing* 10 (1999)

[Braig00] Andreas Braig: *Entwicklung eines Java-basierten Komponentenansatzes für interaktive, dreidimensionale Benutzungsschnittstellen.* Dresden, Technische Universität, Fakultät Informatik, Institut für Software- und Multimediatechnik, Lehrstuhl Multimediatechnik, Dipl.-Arb., 2000

[Bro98] Wolfgang Broll: *Ein objektorientiertes Interaktionsmodell zur Unterstützung verteilter virtueller Umgebungen: GMD - Research Series* (1998) GMD - Forschungszentrum Informationszentrum GmbH

[BWG01] Neil Trevett: *Browser Working Group Charter*, 2001 http://www.web3d.org/x3d/browserwg_charter.html

[Car97] Rikk Carey ; Gavin Bell: *The Annotated VRML 97 Reference,* 1997 http://www.best.com/%7erikk/Book/

[Coh99] Daniel Cohen-Or ; David Levin ; Offir Remez: Progressive Compression of Arbitrary Triangular Meshes. IEEE Visualization ‘99, 1999

[Dach01] Raimund Dachsel ; Jürgen Ebert: *Collapsible Cylindrical Trees: A Fast Hierarchical Navigation Technique*. Dresden, Technische Universität, Fakultät Informatik, Institut für Software- und Multimediatechnik, Lehrstuhl Multimediatechnik, 2001

[Deer95] Michael Deering: Geometry Compression. Proceedings of the 22nd annual ACM conference on Computer graphics, 1995

[Fey01] Jürgen Fey: XML nach Schema: W3C präsentiert Nachfolger der XML-Dokumententyp-Definition. In: c *’t* 11/01, S. 102

[Fog01] Efi Fogel ; Daniel Cohen-Or ; Revital Ironi ; Tali Zvi: A web architecture for progressive delivery of 3D content. Proceedings of the 3D technologies for the World Wide Web, Web3D 2001, 2001

[Fre01] Frank Fremerey: Nicht nur als Spielzeug: 3D-Anwendungen erobern das Web. In: *c’t* 08/01, S. 100-106

[Han97] Chris Hand: *A Survey of 3D Interaction Techniques.* The Eurographics Association, 1997

[Jun00] Stephen Junkins ; Allen Hux: *Subdividing Reality / Employing Subdivision Surfaces for Real-time Scalable 3D.* Intel Architecture Labs, Intel Corporation, 2000
http://www.intel.com/ial/3dsoftware/doc.htm

[Klo97] Jörg Kloss ; Robert Rockwell ; Kornél Szabó ; Martin Duchrow: *VRML97 - Der neue Standard für interaktive 3D-Welten im World Wide Web*, Addison Wesley Longman Verlag, 1997

[Mac01] *Macromedia Director 8.5 Shockwave Studio with Intel® Internet 3D Graphics Software : Powerful, Scalable 3D for the Web Mainstream*, Intel Architecture Labs, Intel Corporation, 2001
http://www.intel.com/ial/3dsoftware/doc.htm

[MPEG01a] *Tutorial Issue on the MPEG-4 Standard*, Image Communication Journal, 2001
http://www.cselt.it/leonardo/icjfiles/mpeg-4_si/

[MPEG01b] Rob Koenen: *Overview of the MPEG-4 Standard,* 2001
http://www.cselt.it/mpeg/standards/mpeg-4/mpeg-4.htm

[Olb00] Stephan Olbrich: *Ein leistungsfähiges System zur Online-Präsentation von Sequenzen komplexer virtueller 3D-Szenen*. Hannover, Universität, Fachbereich Elektrotechnik und Informationstechnik, Diss., 2000

[Ross98] Jarek Rossignac: Interactive exploration of distributed 3D datasets over the Internet. Computer Graphics International Congress (CGI '98 Congress), 1998

[Scr01] Klaus Meißner ; Raimund Dachselt: *3D-Grafikformate: Modellierung, Animation, Interaktion (Vorlesung 6) innerhalb der Lehrveranstaltung Medienströme: Formate, Werkzeuge und Standards.* Technische Universität, Fakultät Informatik, Institut für Software- und Multimediatechnik, Lehrstuhl Multimediatechnik, 2001

[Stün00] Jens Stündel: *Interaktive 3D-Grafik im Netz: Anwendungsklassen, Analyse und Vergleich von Internet-basierten 3D Formaten und Toolkits.* Dresden, Technische Universität, Fakultät Informatik, Institut für Software- und Multimediatechnik, Lehrstuhl Multimediatechnik, Beleg-Arb., 2000

[Vel99] Luiz Velho ; Luiz Henrique de Figueiredo ; Jonas Gomez: Hierachical generalized triangle strips, In *The Visual Comuter 15 (1999) 1*, Pages 21-35, Springer-Verlag

[VRML97] The Virtual Reality Modeling Language
International Standard ISO/IEC 14772-1:1997
http://www.web3d.org/technicalinfo/specifications/vrml97/index.htm

[Wal01] Aron E.Walsh, Mikael Bourges-Sévenier: *core WEB3D*, Prentice Hall PTR, 2001

WWW-Adressen - Verzeichnis

[3DAnywhere] 3DAnywhere
http://www.3danywhere.com/

[3DMeNow] 3DMeNow
http://www.cult3d.com/3Dmenow

[3DPI] 3DPI
http://www.ullala.at/3DPI/

[3ds max] 3ds max von Discreet
http://www2.discreet.com/products/d_products.html?prod=3dsmax

[3Space] 3Space von TGS
http://www.tgs.com/3space/

[AMAPI 3D] AMAPI 3D
http://www.eovia.com/amapi

[ARCHmatic] ARCHmatic-Glossar und -Lexikon
http://www.glossar.de

[Atmosphere] Adobe Atmosphere
http://www.adobe.com/products/atmosphere/

[Autodesk] Autodesk
http://www.autodesk.com

[Beyond 3D] Beyond 3D Extreme
http://www.beyond-3d.com/beyond/

[blaxxun] blaxxun interactive
http://www.blaxxun.de

[blaxxun Contact] blaxxun Contact von blaxxun interactive
http://www.blaxxun.de/c/s?cat=9&sub=2&url=/products/contact/index.html

[blaxxun3D] blaxxun3D von blaxxun interactive
http://www.blaxxun.de/blaxxun3d

[Blendo] Blendo von Sony
http://www.blendomedia.com

[Breitling] Breitling - Uhr
http://www.cult3d.com/gallery/breitling.asp#gallery

[Charamel] Charamel
http://www.charamel.com/

[Chisel] Chisel
http://www.trapezium.com/ChiselPage.htm

[Cinema 4D] Cinema 4D
http://www.cinema4d.com

[Contigra] Contigra
http://www.contigra.de

[Cortona] Cortona von ParallelGraphics
http://www.parallelgraphics.com/products/

[Cosmo Player] Cosmo Player von Cosmo Software
http://www.cai.com/cosmo/

[Cult3D] Cult3D
http://www.cult3d.com

[Cycore] Cycore
http://www.cycore.com

[Discreet] Discreet
http://www.discreet.com

[e:Berlin] e:Berlin
http://www.e-berlin.de

[GeoVRML] GeoVRML Working Group des Web3D-Konsortiums
http://www.geovrml.org/

[GP] Glossar bei G&P Computer
http://www.gpcomputer.de/sg3d/g_gloss.htm

[H-ANIM] Humanuid Animation Working Group des Web3D-Konsortiums
http://h-anim.org/

[HSW] HowStuffWorks - How Shockwave 3D Technology works
http://www.howstuffworks.com/shockwave.htm

[I3DSWT] Intel® 3D Software Technologies - Intel Architecture Labs
http://www.intel.com/ial/3dsoftware/

[IAL] Intel Architecture Labs
http://www.intel.com/ial/

[ImageModeler] ImageModeler
http://www.realviz.com/products/im

[IPRO] Internet Professionell - Macromedia Director 8.5 & Shockwave 3D
http://www.zdnet.de/internet/artikel/wdm/200106/director_00-wc.html

[ISMA] Internet Streaming Media Alliance
http://www.isma.tv

[ISS] International Space Station (CNN)
http://www.cnn.com/SPECIALS/multimedia/vrml/iss/

[iTEConsult] iTEConsult GmbH
http://www.iteconsult.de

[Janet] Janet
http://www.janet.de

[Java3D] Java3D API
http://java.sun.com/products/java-media/3D/index.html

[JayLeno] Jay Leno - Tonight Show
http://nbctv.nbci.com/tonightshow/virtualjay/special.html

[Lightwave] Lightwave von Newtek
http://www.newtek.com

[M4IF] MPEG-4 Industry Forum
http://www.m4if.org

[M4TB] MPEG-4 Toolbox
http://uranus.ee.auth.gr/pened99/Demos/Authoring_Tool/authoring_tool.html

[M4To] MPEG-4 Tools von ENST (Telekom Paris)
http://www.comelec.enst.fr/~dufourd/mpeg-4/

[Maya] Maya von Alias|Wavefront
http://www.aliaswavefront.com/en/WhatWeDo/maya/see/solutions/soln_intro.shtml

[LAN] Lab for Advanced Networking, Vancouver
http://www.ece.ubc.ca/~yasserp/research/

[MMT] Lehrstuhl für Multimediatechnik (TU-Dresden)
http://www-mmt.inf.tu-dresden.de/

[MPEG] Moving Picture Experts Group
http://www.cselt.it/mpeg/

[MRM] Multi-Resolution Mesh - Technologie der Intel Architecture Labs
http://www.intel.com/ial/3dsoftware/mrm.htm

[Nexternet] Nexternet
http://www.nexternet.com/

[NIST VRML97 Parser] NIST VRML97 Parser
http://sdk.web3d.org/summer2001disk2/tools/NIST/index.html

[OpenDivX] OpenDivX Streaming
http://www.projectmayo.com/projects/subproject.php?projectID=5&subprojectID=8

[OpenSPX] Open SPX
http://www.dimension-3d.com/products/spx_plugin.html

[OpenWorlds] OpenWorlds
http://www.openworlds.com

[ParallelGraphics] ParallelGraphics
http://www.parallelgraphics.com

[Prometheus] Prometheus-Projekt des BBC
http://www.bbc.co.uk/rd/projects/prometheus/

[Pulse] Pulse
http://www.pulse3d.com

[RefSW] Referenzsoftware für MPEG-4
www.iso.ch/ittf [unter Menüpunkt „Public Available ISO/IEC Standards“]

[RMWG] Rich Media Working Group des Web3D-Konsortiums
http://groups.yahoo.com/group/rm3d

[ShockPb] Director 8.5 Shockwave Studio - 3D Partners List
http://www.macromedia.com/software/director/3d/partners/

[Shockwave3D] Macromedia Shockwave3D
http://www.macromedia.com/shockwave/

[Shout Interactive] Shout Interactive
http://www.shoutinteractive.com

[Shout3D] Shout3D
http://www.shout3d.com

[Spazz3D] Spazz3D von Virtock
http://spazz3d.com

[TankGame] Shout3D Tank Game
http://www.ismarvr.cz/g2/

[Telopolis] Zeigt her eure Pixel - Telopolis
http://www.heise.de/tp/deutsch/inhalt/te/7697/1.html

[TN13611] Macromedia Director Support Center TechNote (ID:13611)
Information and Support for Xtras Development
http://www.macromedia.com/support/director/ts/documents/xtras_dev.htm

[TN14617] Macromedia Director Support Center TechNote (ID:14617)
How can a Shockwave movie pass information to and from a browser?
http://www.macromedia.com/support/director/ts/documents/sw_comunic.htm

[TOPlay] TOPlay von Orad Net
http://www.oradnet.com

[trueSpace] trueSpace von Caligari
http://www.caligari.com

[UMWG] Universal Media Working Group des Web3D-Konsortiums
http://www.web3d.org/WorkingGroups/media/

[Viewpoint] Viewpoint
http://www.viewpoint.com

[Viewpoint Developer] Viewpoint Developer
http://developer.viewpoint.com/

[ViewpointWP] Viewpoint Whitepapers
http://developer.viewpoint.com/developerzone/5-22.html

[Vorlon] Vorlon - Web3D Validation Tool by Trapezium
http://sdk.web3d.org/summer2001disk1/tools/Vorlon/index.html

[Web3D-Konsortium] Web3D-Konsortium
http://www.web3d.org

[Web3D-MPEG] Web3D-MPEG - Arbeitsgruppe
http://web3dmedia.com/web3d-mpeg/

[X3D] Extensible 3D Working Group
http://www.web3d.org/x3d/

[X3D blaxxun] Core X3D Specification by blaxxun interactive
http://www.blaxxun.de/products/blaxxun_3d/spec/contents.html

[X3D File Converter/Importer] X3D File Converter/Importer von Score Multimedia
http://www.score.de/3D/

[X3D Shout] Core X3D Specification by Shout Interactive
http://www.web3d.org/TaskGroups/x3d/shout/CoreX3D_Proposal_20/contents.html

[Xeena] Xeena - visueller XML-Editor von IBM
http://www.alphaworks.ibm.com/tech/xeena

[Xj3D] Xj3D
http://hypermultimedia.com/Xj3D/

[XMT] XMT
http://www.classica.org/mike/mpeg4/