HiQoS: High Performance Multimedia-Dienste mit Quality-of-Service-Garantien

- Abschlussbericht -

Zeitraum vom 1. Mai .1998 bis zum 30. April 2001

Das diesem Bericht zugrunde liegende Vorhaben wurde mit Mitteln des BMBF gefördert.
Förderkennzeichen: 01 IR 803 A bis E

30. September 2001

APE Ptacek Engineering GmbH - München
Axcent Media AG – Paderborn
IEZ AG - Bensheim
Pixelpark AG - Berlin
Siemens SBS C-LAB – Paderborn

HiQoS Abschlussbericht: 1. Mai 1998 - 30. April 2001

EDITOR

Dr. Peter Altenbernd
Siemens SBS C-LAB
Fürstenallee 11
D 33094 Paderborn
+49 5251 60 6123
<peter@c-lab.de>

Autoren:

Peter Altenbernd
Andreas Bartels
Simon Bicskey
Lars Olof Burchard
Michael Holch
Jörg Jensch
Martin Östreicher
Ingo Neumann
Tomas Plachetka
Torsten Prill
Joachim Seiler
Anton Schmitt

Weitere Informationen zum HiQoS-Projekt befinden sich unter:
http://www.c-lab.de/hiqos

Berichtsblatt

1. ISBN oder ISSN	2. Berichtsart
geplant	Schlussbericht

3a. Titel des Berichts

HiQoS: High Performance Multimedia-Dienste mit Quality-of-Service-Garantien

3b. Titel der Publikation

4a. Autoren des Berichts (Name, Vorname(n))

Dr. Peter Altenbernd (Editor)

4b. Autoren der Publikation (Name, Vorname(n))

5. Abschlussdatum des Vorhabens
April 2001

6. Veröffentlichungsdatum

7. Form der Publikation

8. Durchführende Institution(en) (Name, Adresse)

APE Ptacek Engineering GmbH - München
Axcent Media AG – Paderborn
IEZ AG - Bensheim
Pixelpark AG - Berlin
Siemens SBS C-LAB – Paderborn

9. Ber.Nr. Durchführende Institution

10. Förderkennzeichen
01 IR 803 A bis E

11a. Seitenzahl Bericht
112

11b. Seitenzahl Publikation

13. Fördernde Institution (Name, Adresse)

Bundesministerium für
Wirtschaft und Technologie (BMWi)

53107 Bonn

12. Literaturangaben
38

14. Tabellen
3

15. Abbildungen
57

16. Zusätzliche Angaben

17. Vorgelegt bei (Titel, Ort, Datum)

18. Kurzfassung

Hochperformante netzbasierte Multimedia-Dienste standen im Zentrum der Aktivitäten von HiQoS. Darin enthalten ist einerseits das *Rendering* in photo-realistischer Qualität sowie andererseits die Auslieferung von kontinuierlichen Medien mit garantiertem *Quality-of-Service (QoS).*

Rendering wird benötigt, um künstliche Bilder oder Filmsequenzen zu erzeugen, wie sie in der Architektur (z.B. für Gebäuderekonstruktionen) oder in der Filmproduktion benötigt werden. Dazu ist in HiQoS ein Ansatz gewählt worden, der auf der Benutzung von massiv-parallelen Rechnern beruht. Das Ergebnis sind Animationen von beeindruckender Qualität.

QoS-Technologie kommt zum Einsatz, wenn es darum geht, Filme über das Internet, Intranets oder Extranets zu übertragen, um dabei lästige Störungen (z.B. Ruckler) zu vermeiden. In HiQoS ist dabei neben der Integration von modernen Übertragungsstandards besonders auf die Server-Technologie eingegangen worden. Dies kann dann benutzt werden, um per Netzzugang digitalisierte Filme in sehr guter Qualität abzurufen, wie dies z.B. für Schulungen (*Video-on-Demand*) oder für den o.g. Rendering-Service benötigt wird

19. Schlagwörter
Access Management, Content Management, High Performance Computing (HPC), Multimedia, Quality-of-Service (QoS), Rendering, Radiosity, Raytracing, Real-Time Systems, Video Streaming

20. Verlag	21. Preis

Document Control Sheet

1. ISBN or ISSN	2. Type of Report
planned	Final Report

3a. Titel des Berichts

HiQoS: High Performance Multimedia-Dienste mit Quality-of-Service-Garantien

3b. Title of Publication

4a. Author(s) of the Report (Family Name, First Name(s))	5.End of Project
Dr. Peter Altenbernd (Editor)	April 2001
4b. Author(s) of the Publication (Family Name, First Name(s))	6. Publication Date
	7. Form of Publication

8. Performing Organization(s) (Name, Address)	9. Originator's Report No.
APE Ptacek Engineering GmbH - München Axcent Media AG – Paderborn IEZ AG - Bensheim Pixelpark AG - Berlin Siemens SBS C-LAB – Paderborn	10. Reference No. 01 IR 803 A bis E
	11a. No. of Pages Report 112
	11b. No. of Pages Publication
13. Sponsoring Agency (Name, Address) Bundesministerium für Wirtschaft und Technologie (BMWi) 53107 Bonn	12. No. of References 38
	14. No. of Tables 3
	15. No. of Figures 57

16. Supplementary Notes

17. Presented at (Title, Place, Date)

18. Abstract

High-performance net-based multimedia services were the major focus of the HiQoS project. In this context the project addressed both physically-based *Rendering* as well as delivery of streamed media under guaranteed *Quality-of-Service (QoS).*

Rendering is needed in order to generate pictures or videos, as useful for architecture (e.g. for the reconstruction of devastated buildings) or during film production. Therefore, in HiQoS an approach was taken that exploits the use of massive parallel computers. The result is animations of impressing quality.

QoS technology is deployed when videos are submitted via Internet, intranets or extranets, while avoiding annoying disturbances (like jitter). In HiQoS, the integration of modern transmission standard protocols has been addressed including multimedia server technology. This can be used in order to access digital video via network connections, as in distance learning scenarios (video-on-demand) or in the context of the before mentioned rendering service.

19.Keywords
Access Management, Content Management, High Performance Computing (HPC), Multimedia, Quality-of-Service (QoS), Rendering, Radiosity, Raytracing, Real-Time Systems, Video Streaming

20. Publisher	21. Price

Inhaltsverzeichnis

Kurzbeschreibung

Keywords: *Access Management, Content Management, High Performance Computing (HPC), Multimedia, Quality-of-Service (QoS), Rendering, Radiosity, Raytracing, Real-Time Systems, Video Streaming*

Hochperformante netzbasierte Multimedia-Dienste stehen im Zentrum der Aktivitäten von *HiQoS (High Performance Multimedia-Dienste mit Quality-of-Service-Garantien)*. Darin enthalten ist einerseits das Rendering in photo-realistischer Qualität sowie andererseits die Auslieferung von kontinuierlichen Medien mit garantiertem Quality-of-Service (QoS).

Rendering wird benötigt, um künstliche Bilder oder Filmsequenzen zu erzeugen, wie sie in der Architektur (z.B. für Gebäuderekonstruktionen) oder in der Filmproduktion benötigt werden. Dazu ist in HiQoS ein Ansatz gewählt worden, der auf der Benutzung von massiv-parallelen Rechnern beruht. Das Ergebnis sind Animationen von beeindruckender Qualität.

QoS-Technologie kommt zum Einsatz, wenn es darum geht, Filme über das Internet, Intranets oder Extranets zu übertragen, um dabei lästige Störungen (z.B. Ruckler) zu vermeiden. In HiQoS ist dabei neben der Integration von modernen Übertragungsstandards besonders auf die Server-Technologie eingegangen worden. Dies kann dann benutzt werden, um per Netzzugang digitalisierte Filme in sehr guter Qualität abzurufen, wie dies z.B. für Schulungen benötigt wird (Video-on-Demand).

Das HiQoS-Projekt hatte eine Laufzeit von drei Jahren (1. Mai 1998 – 30. April 2001) und wurde mit Mittel des *Bundesministerium für Bildung und Forschung (BMBF)* im Förderschwerpunkt *High-Performance Computing (HPC) - Höchstleistungsrechnen in der Bundesrepublik Deutschland* gefördert. HiQoS wurde vom Projektträger *Deutsches Zentrum für Luft- und Raumfahrt (DLR) e.V.* in Person von Dr. Rüdiger Krahl begleitet. Die Konsortialführung lag bei Siemens SBS C-LAB (Dr. Peter Altenbernd).

1 Einleitung

Internet-Protocol-(IP)-Technologie hat sich in den letzten Jahren so stark verbreitet, dass damit heute dem professionellen oder privaten Benutzer eine Vielzahl von verschiedenen Diensten angeboten werden kann. Basierend auf IP-Technologie ist im Rahmen von *HiQoS (High Performance Multimedia-Dienste mit Quality-of-Service-Garantien)*

- die Realisierung von Plattformen zur Entwicklung hochperformanter netzbasierter Multimedia-Dienste,
- die prototypische Entwicklung dieser Dienste,
- sowie deren Evaluierung in konkreten industriellen Anwendungen

betrieben worden. Die Dienste zeichnen sich vor allem dadurch aus, dass sie hochqualitative kontinuierliche Medien integrieren, photorealistische Animationen erlauben und auf standardisierten Internet-Technologien aufbauen. Zur Realisierung dieser Dienstleistungen wurden in HiQoS insbesondere die zwei folgenden technischen Ansätze vertieft:

1. das Rendering von photorealistischen Animationen auf Parallelrechnern,
2. die Einhaltung von "Quality-of-Service-(QoS)"-Garantien in einem Netzwerk von Multimedia-Servern.

1.1 Wesentliche Projektergebnisse

Rendering wird benötigt, um künstliche Bilder oder Filmsequenzen zu erzeugen, wie sie in der Architektur (z.B. für Gebäuderekonstruktionen) oder in der Filmproduktion benötigt werden. Dazu ist in HiQoS ein Ansatz gewählt worden, welcher durch parallele Datenverarbeitung komplexe Bildern und Animationen in photorealistischer Qualität liefert. Dafür sind im Rahmen des Projekt einzelne Rendering-Verfahren (Raytracing und Radiosity) gezielt fortentwickelt worden. Zur praktischen Einbettung in den Workflow von Endanwendern werden verschiedene Format-Konvertierungen und Erweiterungen (z.B. zum Austausch von Geometriedaten, Materialien und Lichtquellen für Szenen-Beschreibungen) zur Verfügung gestellt. Dabei ermöglicht eine verteilte Service-Broker-Architektur einen transparenten IP-Zugang zu parallelem High Performance Rendering.

QoS-Technologie muss immer dann zum Einsatz kommen, wenn es darum geht, Datenströmen (Video und Audio (V/A)) über IP-Netze zu übertragen, um dabei lästige Störungen (z.B. Ruckler) zu vermeiden. Diese Medien erreichen einerseits ein enormes Datenvolumen und andererseits ist dies mit zeitlichen Restriktionen verbunden.

In HiQoS ist dabei auf die Integration von modernen Übertragungsstandards, wie z.B. RSVP (Resource Reservation Protocol), für eine garantiert ruckelfreie Übertragung von Medien-Strömen eingegangen worden. Des weiteren konnte dabei gezeigt werden, dass Methoden aus der klassischen Realzeitverarbeitung hervorragend dazu geeignet sind, stark variable Medienströme effizient zu übertragen.

Dies umfasst alle beteiligten, z.T. selbst implementierten Netzwerk-Elemente: Medien-Server

(als Open-Source-Modell unter Linux), verschiedene Medien-Clients (unter Windows und basierend auf Java), und Router (Integration).

Neben dem Bereich des QoS ist eine Infrastruktur für Verwaltung von Video-on-Demand-(VoD)-Applikationen geschaffen worden: *Content-Management* (d.h. intelligentes *Verteilen* von Medien (Inhalte) in Netzwerken von Medien-Servern und *Access-Management* (d.h. intelligenter *Zugriff* auf Medien (Inhalte) in Netzwerken von Medien-Servern kommen zum Einsatz.

In den beiden Themen-Schwerpunkten des Projekts sind Anwendungen mit eigenen Ambitionen gestaltet worden, die die Anwendbarkeit der HiQoS-Technologie verdeutlichen sollen. Zum einen sind im Bereich Rendering komplexe Szenen aus der Filmproduktion und der Architektur berechnet worden. Zum anderen sind verschiedene VoD-Szenarien (als Business-TV, zur Lehrerausbildung und zur Mitarbeiter-Info) und ein Live-Szenario (Steuerung eines Miniatur-Fahrzeugs) evaluiert worden. Dabei konnten jeweils positive Erfahrungen gemacht werden.

Die entwickelten HiQoS-Komponenten sind gemacht für Entwickler von Multimedia-Diensten, d.h. ein möglicher Endanwender ist durch HiQoS nicht direkt, sondern erst durch die resultierende Anwendung oder den Service adressiert.

1.2 Beschreibung des Konsortiums

Das HiQoS-Konsortium umfasst folgende Partner und deren Aufgaben:

- *Siemens SBS C-LAB*: Konsortialführung, QoS-Kommunikation, Anwendung „ViLM“ (in Zusammenarbeit mit *AG Didaktik der Informatik* an der Universität Paderborn), Live-Streaming
- *Siemens CT* (als Unterauftragnehmer): Medien-Server, QoS-Kommunikation, Access-Management
- *Universität Paderborn - PC²* (als Unterauftragnehmer): Medien-Server, Content-Management, QoS-Kommunikation, Medien-Client
- *Universität Paderborn - AG Monien* (als Unterauftragnehmer): Rendering-Server und Algorithmen
- *AXCENT Media AG*: Zugang zu Rendering-Server, Medien-Client, Content-Management
- *Pixelpark AG*: Anwendung „Pixelvision“
- *APE Ptacek Engineering GmbH*: Anwendung „CarTV“
- *UPSTART! Filmproduktion GmbH* (als Unterauftragnehmer): Anwendung Rendering in Filmproduktion
- *IEZ AG*: Anwendung Rendering in der Architektur
- *GPO Mülheim mbH* (als Unterauftragnehmer): Anwendung Rendering in der Architektur

1.3 Übersicht über dieses Dokument

In den folgenden Abschnitten wird zunächst das HiQoS-Gesamtszenario (Abschnitt 2.1) erklärt, bevor auf die beiden o.g. Schwerpunkte in den Abschnitten 2.2 und 2.3 eingegangen wird. In Abschnitt 2.4 werden die in HiQoS enthaltenen Anwendungsszenarien beschrieben. Der Ende des Berichts enthält eine Zusammenfassung der Projektplanung (Abschnitt 3) und die Beiträge der einzelnen Partner zu diesem Projekt (Abschnitt 4).

2 Inhaltliche Beschreibung

2.1 Das HiQoS-Gesamtszenario

Das HiQoS-Gesamtszenario (siehe Abbildung 1) ergibt sich durch die Verbindung von Rendering-Services mit Multimedia-Servern.

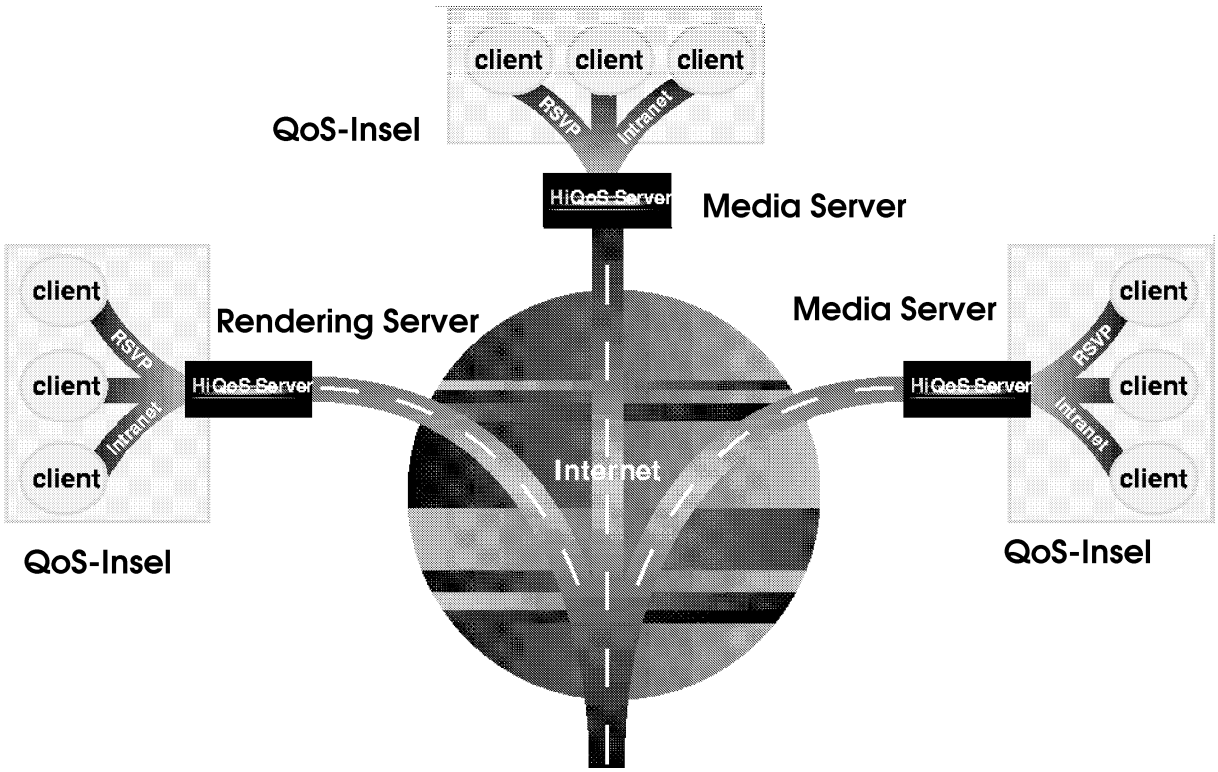

Abbildung 1: Das HiQoS-Gesamt-Szenario

Dabei können Rendering-Aufträge vom Client aus an einen Rendering-Server über das öffentliche Internet vergeben werden. Ist das Rendering erfolgt, wird der entsprechende Datenstrom auf dem nächsten Multimedia-Server abgelegt. Dort kann das Ergebnis, wie gewöhnliche andere Datenströme z.B. bei Video-on-Demand, vom Client abgerufen werden. In einem Subnetz (Intranet) können sowohl Rendering- als auch Multimedia-Server vorkommen. QoS-Garantien können nur innerhalb von Intranets gegeben werden, da realistischerweise gegenwärtig nur dort Bandweitenreservierungen möglich sind („QoS-Inseln“). Um also hochqualitatives Video und -Audio zu ermöglichen, müssen entsprechende Beiträge immer erst im lokalen Intranet des Clients vorliegen. Dazu werden in HiQoS neuartige Content- und Access-

Management-Techniken (siehe Abschnitte 2.3.2 und 2.3.5) zum Einsatz gebracht.

2.2 Rendering in HiQoS

2.2.1 Globale Beleuchtungssimulation

Ein Ziel der Computergrafik ist die Generierung photorealistischer Bilder (photorealistisches Rendering) von 3D-Szenen. *Photorealistisch* bedeutet, dass ein generiertes Bild sich von einem mit einer realen Kamera in einer realen Umgebung aufgenommenen Bild nicht unterscheidet. Um dieses Ziel zu erreichen, müssen mehrere Aufgaben gelöst werden:

1. Ein Modell der Umgebung (Szene) muss erzeugt werden.
2. Ein Modell der Beleuchtungssimulation muss ausgewählt werden.
3. Eine Berechnungsmethode muss ausgewählt werden.
4. Ein Qualitätskriterium muss ausgewählt werden, so dass das Ergebnis der Simulation mit dem optimalen Ergebnis verglichen werden kann.

Diese Schritte sind miteinander verbunden. Jeder der Schritte bringt philosophische als auch technische Probleme mit, die bis jetzt noch nicht vollständig gelöst worden sind. Trotzdem werden heute Bilder berechnet, die man als *photorealistisch* bezeichnen kann. (Dieses Phänomen ist nicht nur in der Computergrafik zu sehen – zum Beispiel, Medikamente heilen Leute, obwohl niemand die Chemie oder den menschlichen Körper ausführlich versteht; Atomkraftwerke produzieren Elektrizität, obwohl die Atomphysik nicht bis zu allen Details verstanden ist; usw.) Die Entwicklung von erfolgreichen Technologien basiert auf dem vertieften Verständnis der Probleme, auf der Erfahrung und auf anwendungsspezifischen Kompromissen.

Einige der Probleme und ihre Lösungen werden hier skizziert:

Ein *Computergrafik-Modell* besteht aus einer Beschreibung der Geometrie aller Objekte, Beschreibung der Materialien, Beschreibung der Lichtquellen und Beschreibung der Kameras. (Für einige Anwendungen wird das Modell um eine Beschreibung der Atmosphäre erweitert.) Schon durch diese Beschreibungen wird die Realität vereinfacht. Beschreibung der *Geometrie* begrenzt sich auf eine geometrische Beschreibung der Oberflächen aller Objekte. Um Flexibilität in weiteren Schritten zu erreichen, werden die Oberflächen oft von Polygonen modelliert. *Materialien* enthalten Reflektions- und Transmissions-Eigenschaften der Objekte. Die reale Reflektion des Lichtes ist eine Funktion der geometrischen Lage, des einfallenden Winkels und des ausfallenden Winkels (und auch der Licht-Wellenlänge und der Temperatur, die meistens nicht berücksichtigt werden). Diese Funktion wird normalerweise durch eine Summe von einer diffusen und einer spekularen Funktion vereinfacht. *Lichtquellen* werden oft als Punkte modelliert, entweder mit einer gleichmäßigen Ausstrahlung in allen Richtungen oder (Spotlights) mit einer richtungsabhängigen Ausstrahlung. Als eine Alternative zu Punktlichtquellen können Flächenlichtquellen (mit einer gleichmäßigen Ausstrahlung in allen Punkten) verwendet werden. *Kameras* werden durch ihre Positionen, Richtungen und optischen Eigenschaften beschrieben. Wenn sich das Modell in der Zeit ändert, spricht man von dynamischen Szenen (ändert sich nur die Kamera, spricht man von Kamera-Animationen).

Die Quantum-Elektrodynamik Theorie erklärt alle bekannte Licht-Phänomene ziemlich gut. Doch wird sie für eine Berechnung von Bildern nicht direkt verwendet, weil die Simulation der Mikrowelt mit heutigen Rechnern zu aufwendig wäre. Die Zielsetzung reduziert sich auf die *Simulation der geometrischen Optik*. Das Licht zwischen den Objekten wird entlang gerader Linien transportiert und einige Interaktionen des Lichtes mit Objekten werden ignoriert. Dieses Problem kann mathematisch exakt durch eine Integral-Gleichung definiert werden (Rendering-Gleichung von Kajiya, 1986). Die Zielsetzung kann noch weiter reduziert werden, indem mehrfache Reflektionen des Lichtes ganz ignoriert werden. In diesem Fall können Bilder durch die Hardware heutiger Grafikkarten in Real-Zeit berechnet werden. Die Bilder werden allerdings nicht mehr realistisch aussehen.

Die Beleuchtungssimulation muss algorithmisch beschrieben werden. Oft verwendete Berechnungsmethoden zu der globalen Beleuchtungssimulation sind *Radiosity* und *Ray-Tracing*. Diese zwei Methoden machen unterschiedliche Vereinfachungen und Annahmen an die vorherige Modellierung und ihre Ergebnisse sind qualitativ unterschiedlich. Radiosity und Ray-Tracing und ihre Vorteile und Nachteile werden später beschrieben. Beide Methoden sind sehr rechenintensiv. Möglichkeiten ihrer Beschleunigung sind Beschleunigung der Hardware, Entwicklung effizienterer sequentieller Algorithmen und Parallelisierung.

Das einzige objektive *Qualitätskriterium* der berechneten Ergebnisse ist ein Vergleich einer physikalischen Messung der Beleuchtung in der realen Szene und mit dem berechneten Ergebnis. Ein solcher Vergleich ist aber meistens nicht möglich – entweder existiert die reale Szene nicht oder die Messungen sind mit großen technischen Schwierigkeiten verbunden. In der Praxis benutzte Qualitätskriterien sind meistens subjektiv und abhängig von der Anwendung. Ein internes Qualitätskriterium ist die Annäherung zu der Lösung der (vereinfachten) Kajiyas Rendering-Gleichung.

2.2.2 Anwendungsszenarien in HiQoS

In der AG Monien an der Universität Paderborn wurden im Laufe des HiQoS-Projektes parallele Radiosity und Ray-Tracing Methoden entwickelt, an zwei Anwendungsszenarien angepasst und durch das *HiQoS Rendering-System* als einen prototypischen e-Commerce Internet-Service angeboten. Das HiQoS Rendering-System ist ein verteiltes System, in dem das parallele Ray-Tracing und das parallele Radiosity als eine *Rendering-Server* Komponente realisiert sind. Die von der Firma Axcent Media AG entwickelten Komponenten *Service-Broker*, *Scheduling-System* kümmern sich um Verwaltung der Benutzer, Verwaltung der Rendering-Aufträge und Verwaltung der Ressourcen.

Ein Anwendungsszenario ist die *Architektur-Visualisierung (Ray-Tracing Szenario)*. Im Programm *speedikon W* der Firma IEZ AG (bzw. im Programm *Arcon-Visuelle Architektur* der Firma mb-Software) werden komplexe Architektur-Modelle erstellt. Ein Anwender dieser Software ist die Firma GPO mbH. Der Rendering-Service wird hierbei benutzt zur Entscheidungsunterstützung während der Bauentwurfs- und Planungsphase sowie zur öffentlichen Präsentation in Form von hoch-qualitativen Einzelbildern und Kamera-Animationen. Das im HiQoS Rendering-System integrierte parallele Ray-Tracing wird für die Berechnung verwendet.

Das zweite Anwendungsszenario ist die *Berechnung der globalen Beleuchtung für die Filmindustrie (Radiosity-Szenario)*. Die Firma Upstart! Filmproduktion GmbH produziert Visual

Effects für Film und Fernsehen im Bereich Werbung, Spielfilm und Dokumentation. Die Radiosity-Methode bietet Flexibilität bei der finalen Komposition eines Films an. Die von der Radiosity-Methode berechnete globale diffuse Beleuchtung kann gespeichert werden und vor dem finalen Rendering weiter bearbeitet werden. Bei größeren Szenen ist die Radiosity-Methode sehr zeit- und speicheraufwendig. Die im HiQoS Rendering-System integrierte parallele Radiosity-Methode adressiert beide Probleme.

2.2.3 Ray-Tracing

Ray-Tracing ist eine oft verwendete Methode, die zur Berechnung der globalen Beleuchtung benutzt wird. Die Methode ist basiert an der Verfolgung von Licht-Partikeln (Photonen), die von den Lichtquellen generiert werden. Bei dem klassischen Ray-Tracing Algorithmus wird die globale Beleuchtung in der 3D-Szene nicht explizit gespeichert. Das Ergebnis des Algorithmus ist ein Bild von einer vorher ausgewählten Perspektive (Kamera). Während des Algorithmus werden Photon-Pfade konstruiert, die immer in einer Lichtquelle anfangen und nach mehreren Reflektionen (oder Refraktionen) von Objekten in der Kamera enden. Weil unendlich viele solche Pfade existieren, müssen einige Vereinfachungen gemacht werden. Das folgende wird angenommen:

- Die Photon-Pfade bestehen aus geraden Linien (Strahlen).
- Die virtuelle Kamera hat eine vorgegebene Auflösung, d. h. sie besteht aus einer endlichen Anzahl von Pixeln. Jedes Pixel kann nur endlich viele Farben haben.
- Die indirekte Reflektion bzw. Refraktion der Photonen von Objekten ist perfekt spekular (d. h., perfekte Spiegelung: auskommender Winkel = herkommender Winkel).

Diese Annahmen erlauben die Formulierung eines deterministischen rekursiven Ray-Tracing Algorithmus. Der Algorithmus verfolgt die Photonen in der umgekehrten Richtung, d. h. er sendet einen Strahl durch ein Pixel der Kamera und konstruiert Pfade zu den Lichtquellen, indem er die Reflektionen und Refraktionen des Strahls rückgängig verfolgt. Es werden also nur die Photonen verfolgt, die das Bild beeinflussen. Bei jeder Interaktion eines Strahls mit einem Objekt wird die direkte (lokale) Beleuchtung in dem Schnittpunkt berechnet, indem Strahlen vom Schnittpunkt zu jeder Lichtquelle gesendet werden, um zu finden, ob der Schnittpunkt von den Lichtquellen beleuchtet ist oder nicht. Das folgende Bild zeigt die Struktur der Rekursion.

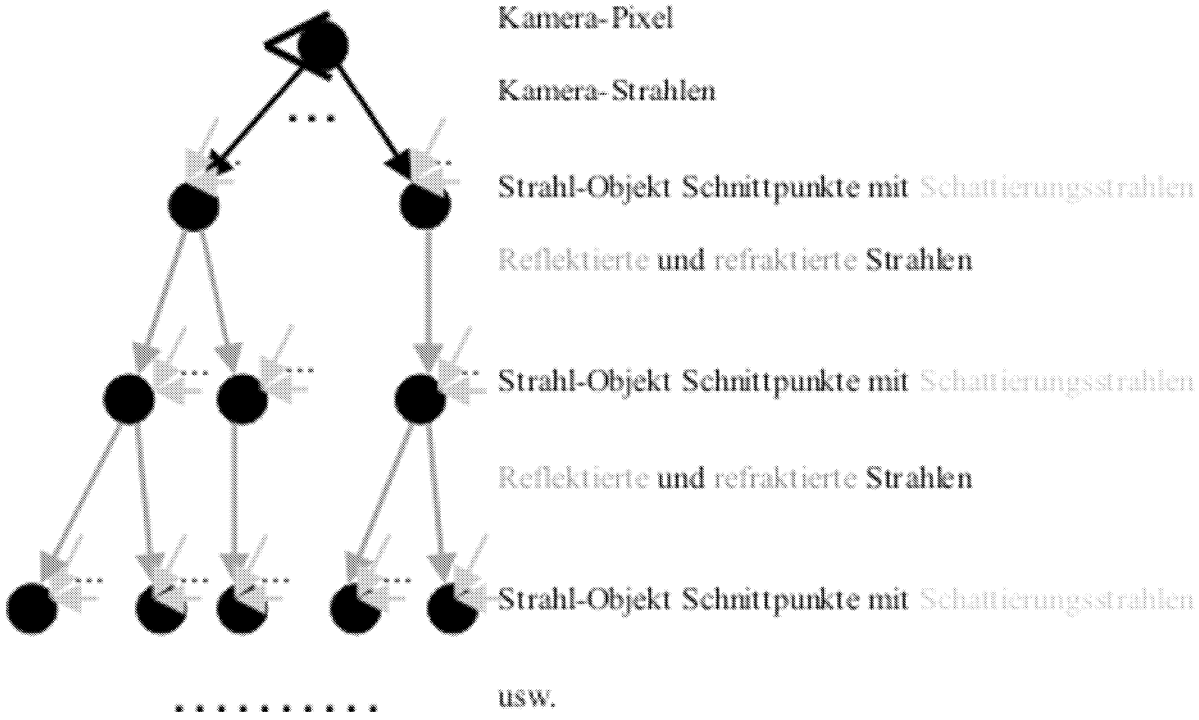

Abbildung 2: Struktur der Rekursion beim Ray-Tracing

Die Berechnungszeit ist abhängig von der Anzahl der Strahlen in dem Baum. Die Anzahl der Strahlen ist abhängig von der Auflösung des berechneten Bildes (sei R die Anzahl der Pixel), von der maximalen erlaubten Rekursionstiefe D, von der Anzahl der Lichtquellen in der Szene L und von den Geometrie- und Materialeigenschaften der Objekte. Die Zeit der Berechnung kann vor der Berechnung nur sehr grob abgeschätzt werden. Wenn N die durchschnittliche Anzahl der reflektierten und refraktierten Strahlen per Schnittpunkt ist, wird die gesamte Anzahl der Strahlen gleich $R * N^{D-1} / (N - 1) * (1 + L)$. In einem realen Fall ist die Anzahl der Strahlen ungefähr $640 * 480 * 1{,}2^5 / (1{,}2 - 1) * (1 + 20) \cong 80.260.000$. Für jeden dieser Strahlen muss während des Laufs des Algorithmus der nächste Schnittpunkt mit den Objekten gefunden werden. Suche nach dem nächsten Schnittpunkt eines Strahls mit den Objekten ist eine teure Operation (auch beim Einsatz von Beschleunigungstechniken wird in diesen Suchen typischerweise mehr als 90% der gesamten Berechnungszeit verbraucht).

Die Ray-Tracing Methode hat mehrere Vorteile, dank welchen sie in vielen Anwendungen integriert wird. Ein wichtiger Vorteil ist, dass sie von einem Benutzer intuitiv als eine Erweiterung der lokalen Beleuchtung (OpenGL-Qualität) betrachtet werden kann. In den Architektur-Visualisierungsprogrammen *speedikon*, bzw. *Arcon*, sieht man in einem 3D-Ansicht-Fenster ein Bild von einer 3D-Szene in der OpenGL Qualität (keine Schatten, keine Spiegelung). Per Maus-Click bekommt der Benutzer ein Bild von der gleichen Perspektive, wobei die Schatten und die spekulare Spiegelung und Transparenz korrekt berücksichtigt werden. Fast keine spezielle Ausbildung ist notwendig für den Benutzer, um realistische Bilder zu erzeugen. Ein weiterer Vorteil ist, dass während der Berechnung nur wenig Speicher gebraucht wird. Der Algorithmus ist robust, kann mit verschiedenen Szene-Repräsentationen arbeiten (nicht nur mit einer Polygon-Repräsentation) und kann an vielen Anwendungen angepasst werden.

Der Nachteil der Methode ist die Abhängigkeit von der Betrachtung. Die globale Beleuchtung wird nicht explizit gespeichert (nur in Laufzeit wird temporär nur ein Teil der Beleuchtungssituation erzeugt, um die Farbe eines Pixels zu bestimmen). Wird eine Animation berechnet, muss man beim Ray-Tracing jedes Bild der Animationssequenz unabhängig berechnen. Fast keine Information von den schon berechneten Bildern kann ausgenutzt werden. Ein weiterer Nachteil ist die hohe Zeit-Komplexität des Algorithmus. Die absoluten Berechnungszeiten für ein Bild in einer Video-Auflösung liegen in Bereichen von Minuten bis Stunden.

Beide obengenannte Nachteile wurden während der Integration des Ray-Tracings im HiQoS Rendering-System adressiert. Die parallele Implementierung ermöglicht die Berechnungszeiten durch die Verwendung von mehreren Prozessoren wesentlich zu reduzieren. Bei der Berechnung einer Animation wird die Zeit für das Einlesen der Szene gespart. Die Szene wird nur am Anfang gelesen und bleibt im Speicher von Prozessoren, die an der Berechnung der Bilder beteiligt sind. So können nicht nur einzelne Bilder, sondern auch hoch-qualitative Ray-Tracing Animationen in einer akzeptierbaren Zeit berechnet werden. Dadurch kann im Kontext der Architektur-Visualisierung der Bauentwurf- und Planungsprozess verkürzt und vereinfacht werden.

2.2.3.1 Paralleles Ray-Tracing

Absolute sequentielle Ray-Tracing Berechnungszeiten können bei komplexen Szenen im Bereich von Minuten bis Stunden liegen (siehe Abschnitt 2.2.3). Eine der Möglichkeiten zur Beschleunigung ist die Parallelisierung der Algorithmus. Die existierenden Parallelisierungsmethoden können in drei Kategorien klassifiziert werden (wobei das verteilte Speichermodel betrachtet wird):

1.Bildschirm-Verteilung

Die Kamera-Strahlen sind voneinander unabhängig und können deshalb von mehreren Prozessoren parallel berechnet werden. Das ist die grundliegende Idee der Bildschirm-Verteilungsmethoden.

Vorteile:

- Relativ einfache Implementierung.
- Sequentielle Beschleunigungstechniken können meistens problemlos verwendet werden.

Nachteile:

- Einfache Implementierungen führen zu einer schlechten Lastbalancierung.
- Die Szene wird an jedem Prozessor repliziert. Bei großen Szenen kann Speicher eines Prozessors nicht ausreichen, um das ganze Modell zu speichern.

2.Objektraum-Verteilung

Ein Nachteil der Bildschirm-Verteilungsmethode ist, dass jeder Prozessor die ganze Szene im

Speicher halten muss. Die maximale Größe der Szene ist also durch Größe eines Prozessors begrenzt. Bei den Objektraum-Verteilungsmethoden wird der 3D-Objektraum an die Prozessoren verteilt. Jeder Prozessor hält nur ein Stück der ganzen Szene. Die klassischen Objektraum-Verteilungsmethoden verteilen die Szene geometrisch. Bei der parallelen Berechnung werden die Strahlen innerhalb des lokalen 3D-Raums eines Prozessors verfolgt. Verlässt ein Strahl den 3D-Raum des Prozessors, übergibt der Prozessor (durch eine Nachricht) Bearbeitung des Strahls seinem Nachbar-Prozessor, in 3D-Raum welchen die Strahl weitergeht.

Vorteile:

- Die Szene wird nicht an Prozessoren repliziert. Die maximale Größe der Szene ist nicht durch den Speicher eines Prozessors beschränkt.

Nachteile:

- Schwierige Implementierung.
- Sequenzielle Beschleunigungstechniken können nicht direkt übernommen werden.
- Die Lastbalancierung ist problematisch.

3.Funktionelle Dekomposition

Bei der funktionellen Dekomposition werden die einzelnen Schritte des Ray-Tracing Algorithmus (z. B. die Suche des ersten Strahl-Objekt Schnittpunkts, Schattierung, usw.) von nur an diese Aufgaben spezialisierten Prozessen behandelt. Die Prozesse werden in einem Client-Server Modell verbunden.

Vorteile:

- Modularität, Kombination mit der vorherigen Methoden.

Nachteile:

- Schwierige Implementierung.
- Sequenzielle Beschleunigungstechniken können nicht direkt übernommen werden.
- Die Lastbalancierung ist problematisch.

Im Rahmen des HiQoS-Projektes wurde von der Universität Paderborn (AG Monien) ein daten-paralleler Ray-Tracer entwickelt, in dem die *Bildschirm-Verteilung mit der Objektraum-Verteilung kombiniert* ist. Diese kombinierte Lösung ermöglicht auch sehr große Szenen zu berechnen (Szenen, bei welchen der Speicher eines Prozessors nicht ausreicht, d.h. bei welchen eine sequenzielle Berechnung nicht möglich ist). Die Software-Realisierung ist sehr flexibel – die Objektraum-Verteilung kann manuell ausgeschaltet werden (um einen kleinen Overhead der Objektverteilung zu sparen), der Algorithmus kann gut mit sequentiellen Beschleunigungstechniken kombiniert werden und ist voll-automatisch. Die daten-parallele Implementierung ist auf dem Code des populären (sequenziellen) POV-Ray Freeware Ray-Tracer

basiert. Die Ideen der Parallelisierung werden im Folgenden skizziert.

Bei einer einfachen Bildschirm-Verteilungsstrategie wird der Bildschirm statisch (vor der Berechnung) zerlegt und die einzelnen Teile werden parallel berechnet. Die Anzahl der Teile entspricht der Anzahl der Prozessoren (siehe Abbildung 3).

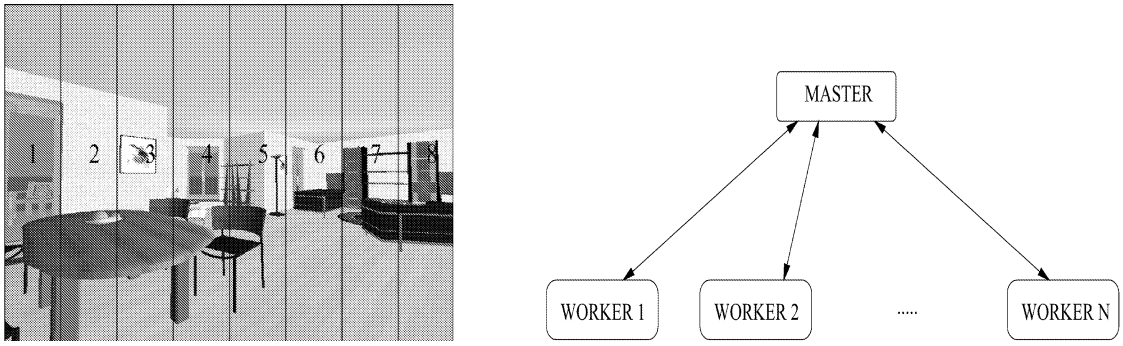

Abbildung 3: Eine statische Bildschirm-Verteilung beim parallelen Ray-Tracing bei 8 Prozessoren. Der Master-Prozess verteilt die Arbeit an Worker-Prozesse und sammelt ein Bild aus den Teil-Ergebnissen.

Das Problem ist, dass die Berechnungszeiten von verschiedenen Gebieten des Bildschirms sehr unterschiedlich sein können. Die Lastbalancierung kann sehr schlecht sein. Wenn zum Beispiel die Berechnung eines Gebiets 1 Minute dauert und die Berechnung der restlichen Gebiete nur 10 Sekunden, wird die gesamte parallele Berechnung 1 Minute dauern. In diesem Beispiel 7 Prozessoren aus 8 haben innerhalb dieser 1 Minute nur 10 Sekunden bearbeitet und 50 Sekunden gewartet.

Eine andere einfache Strategie ist, die einzelnen Spalten (oder Pixel) des Bildschirms an die Prozessoren dynamisch zu verteilen. Ein Worker-Prozess, der keine Arbeit hat, meldet sich beim Master-Prozess und bekommt einen neuen Job zum Berechnen (siehe Abbildung 4).

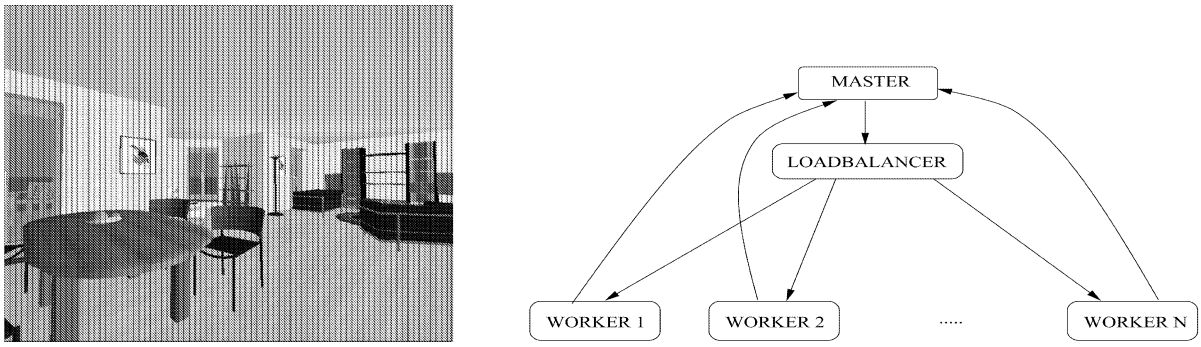

Abbildung 4: Eine dynamische Bildschirm-Verteilung beim parallelen Ray-Tracing. Die Lastverteilung und Sammlung der Ergebnisse wird hier zwischen dem Master- und Loadbalancer-Prozess verteilt. Die Idee ist, den Master-Prozess zu entlasten.

Das Problem dieser Verteilung sind die hohen Kommunikationskosten (die Anzahl der Jobs ist zu hoch).

Die in der AG Monien entwickelte Verteilungsstrategie liegt zwischen den zwei vorherigen Extremen. Die Annahme der Strategie ist, dass sich die Berechnungszeiten T_1, T_2 zwei gleich großer Regionen des Bildschirms nur um eine multiplikative Konstante unterscheiden: $T_1 / T_2 < k$. Wenn eine solche Konstante k existiert (und wenn sie bestimmt werden kann), dann existiert ein optimaler Algorithmus, der eine perfekte Lastbalancierung (bei möglichst wenig Nachrichten) garantiert. Die Idee des Algorithmus ist zuerst große Jobs zu verteilen, später immer kleinere, die kleinste am Ende der Berechnung (siehe Abbildung 5).

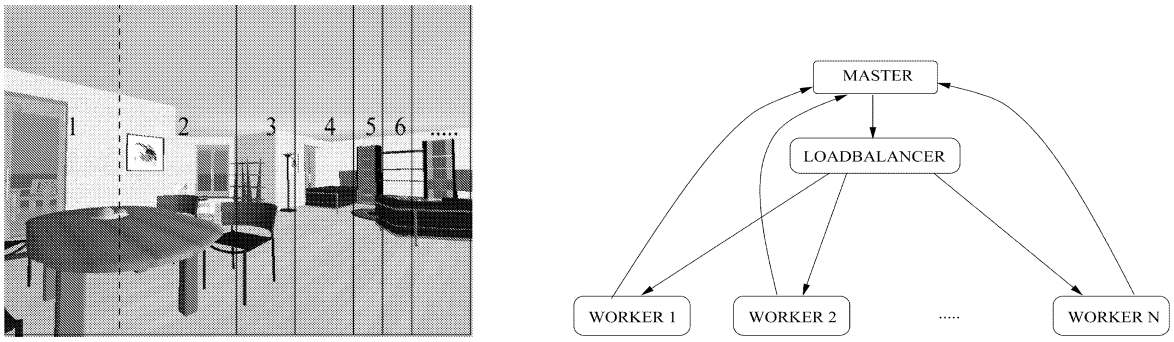

Abbildung 5: Eine Illustration der optimalen Verteilungsstrategie für 2 Worker-Prozesse und k=3. Zuerst werden den 2 Workers die Regionen 1 und 2 zugewiesen. Die weiteren Regionen werden dynamisch zugewiesen, in der beschriebenen Reihenfolge.

Die Annahme der optimalen Verteilungsstrategie wird bei einer konkreten Ray-Tracing Implementierung meistens erfüllt, wobei die Konstante k analytisch oder empirisch bestimmt werden kann.

Das Aliasing-Problem wird meistens in Veröffentlichungen zum parallelen Ray-Tracing nicht diskutiert. Das Aliasing-Problem entsteht durch unzureichendes Sampling eines Pixels. Wenn nur ein Strahl per Pixel verfolgt wird, sieht man stufende Effekte im berechneten Bild. Um das Bild bei der gleichen Auflösung glatter zu machen, wird *Anti-Aliasing* verwendet, d. h. mehrere Strahlen werden durch ein Pixel gesendet (und die durchschnittliche Farbe wird für die Farbe des Pixels verwendet). Bei der Bildschirm-Verteilungsstrategie geht die Effizienz verloren, wenn das Anti-Aliasing nicht speziell berücksichtigt wird – die Pixel an der Grenze zwischen zwei Regionen werden zweimal berechnet (von zwei unterschiedlichen Prozessoren). In der Ray-Tracing Implementierung, die im HiQoS Rendering-System integriert wurde, ist dieses Problem gelöst. Ist das Anti-Aliasing vom Benutzer gewünscht, rechnen die Prozessoren ein Bild mit Anti-Aliasing, allerdings nicht für die Grenz-Pixels. Die Grenz-Pixels werden in einer zusätzlichen Kommunikationsrunde vom zentralen Master-Prozess verteilt. Die Verteilung der Grenz-Pixels kann man sich als eine Generierung zusätzlicher Jobs für die Worker-Prozesse vorstellen. So wird garantiert, dass jedes Pixel nur einmal im gesamten System berechnet wird.

Bei der Animationsberechnung kann die Parsing-Zeit (die Zeit, die für das Einlesen der Szene von der Platte benötigt wird) eine große Rolle spielen, wenn die ganze Szene nach der Berechnung jedes Bilds immer wieder gelesen werden muss. Bei der oben beschriebenen parallelen Lösung wird die Szene nur einmal gelesen. Nach der Berechnung eines Bilds bleibt die Szene im Speicher der Worker-Prozesse behalten. Um das nächste Bild zu berechnen, müssen

nur inkrementelle Änderungen an die Worker-Prozesse gesendet werden. Zum Beispiel, bei Kamera-Animationen muss nur eine neue Kamera-Information gesendet werden. Um die Parsing-Zeiten noch weiter zu reduzieren, wurde das ursprüngliche ASCII-Format durch ein binäres Format ersetzt.

Der größte Nachteil der Bildschirm-Verteilung ist eine ineffiziente Nutzung des Speichers. Die ganze Szene wird in jedem Prozessor gespeichert. Die maximale Große der Szene ist also durch den Speicher eines Prozessors beschränkt. Die entwickelte *daten-parallele Ray-Tracing Methode* löst dieses Problem. Bei der daten-parallelen Lösung wird die maximale Größe der Szene nur von dem gesamten Speicher aller Prozessoren begrenzt. Die Idee ist, die speicherintensive Geometrie zwischen den Prozessoren zu verteilen. Die Verteilung der Geometrie geschieht vollautomatisch beim Einlesen der Szene. Der Master-Prozess kümmert sich um eine gleichmäßige (quasi-zufällige) Verteilung der Objekte vor der Berechnung. Der Master-Prozess hält eine Datenbank mit der Speicher-Auslastung aller Worker-Prozesse und wählt bei der nächsten Objekt-Zuordnung den Worker-Prozess, der in dem Moment die geringste Last hat. Am Ende des Parsings hat jeder Worker-Prozess im Speicher eine Submenge der Objekte und weiß, in welchen Worker-Prozessen die restlichen Objekte gespeichert sind und wie groß sie sind. Für die nichtlokalen Objekte erzeugt jeder Worker-Prozess am Anfang der Berechnung nur leere Boxen, in denen alle Objekt-Informationen neben der Geometrie gespeichert sind. Wenn bei der Berechnung eine fehlende Information gebraucht wird (bei der Suche des nächsten Strahl-Objekt Schnittpunkts), sendet der Worker-Prozess eine *OBJECT REQUEST* Nachricht an den Prozess, der das Objekt hält. Gleichzeitig überprüft er, ob er genug *Cache-Speicher* für das angeforderte Objekt hat (siehe Abbildung 6). Wenn ja, wird das empfangene Objekt im Cache gespeichert und die entsprechende Objekt-Box mit den geometrischen Daten gefüllt. Wenn nein, werden einige Objekte vom Cache-Speicher freigegeben und die entsprechenden Objekt-Boxen werden entleert. Mehrere Cache-Strategien wurden implementiert und getestet (Random, Aging, LRU, usw.) Die LRU-Strategie (***L**ast **R**ecently **U**sed*, d. h., das längst nicht verwendete Objekt wird aus dem Cache-Speicher entfernt) wurde als die effizienteste ausgewählt.

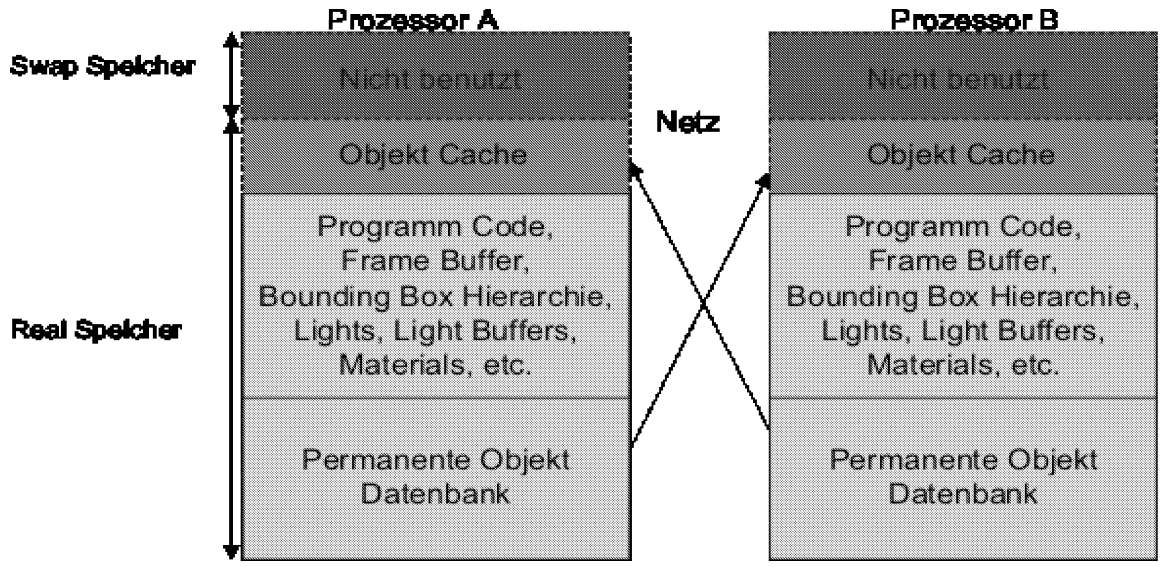

Abbildung 6: Speicherorganisation beim daten-parallelen Ray-Tracing.

Die parallele Implementierung wurde an die Zielarchitektur *hpcLine in PC²* portiert. Die folgenden Diagramme demonstrieren die Effizienz der Implementierung auf dem hpcLine Rechner.

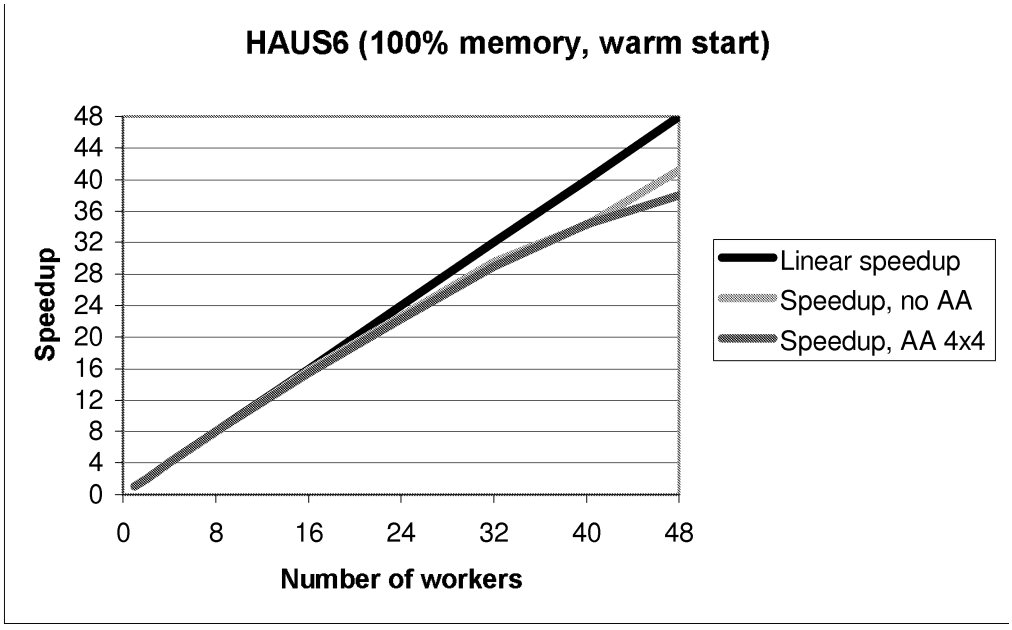

Abbildung 7: Speedup-Messungen für die Arcon Haus6 Szene (hpcLine System, 614 Objekte,22 Punktlichtquellen, Auflösung 640x480). Die Daten-Parallelisierung wurde hier ausgeschaltet (nur die Bildschirm-Verteilung ohne Objekt-Austausch wurde verwendet). „Warm start“ bedeutet, dass die Szene vor der Berechnung im Speicher aller Prozessoren geladen war. „AA 4x4“ bedeutet, dass Anti-Aliasing 4x4 verwendet wurde (durch jedes Pixel wurden 16 Sample-Strahlen gesendet).

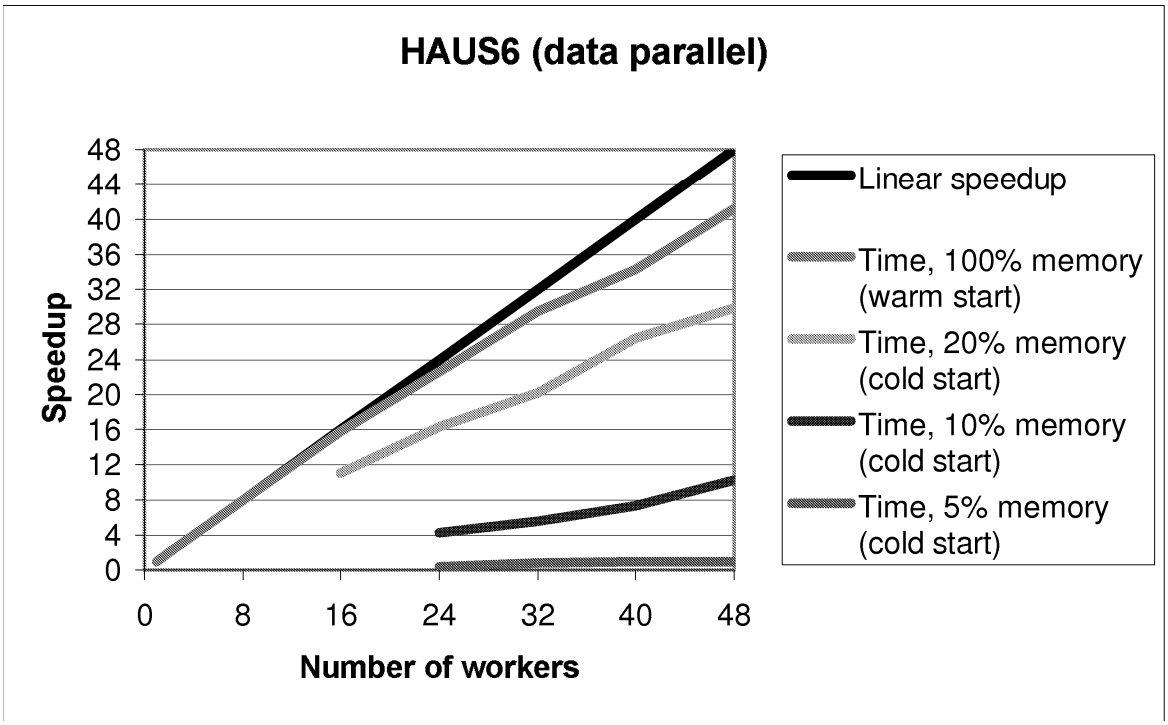

Abbildung 8: Daten-parallele Simulation für die Haus6 Szene. Die Speicherschränke wurden softwaremäßig simuliert. „100% memory warm start“ ist eine Referenz-Messung, wobei jeder Prozessor vom Anfang die gesamte Geometrie schon vor der Berechnung im Speicher hatte. „5% memory“ bedeutet, dass auf jedem Prozessor in jedem Zeitpunkt maximal 5% der gesamten Geometrie gespeichert werden durfte. Die Effizienz fällt ab, wenn die einzelnen Prozessoren wenig Speicher haben (das Bild wird aber berechnet), weil viel Zeit für den Objekt-Austausch zwischen den Prozessoren verbraucht wird. Der 5% Fall ist allerdings sehr extrem und entspricht (nach der Skalierung der Konstanten an die heutige Speichergröße von 500 MB) einer Geometriegröße von mehreren Gigabytes. Mit anderen Worten, auch 1 Terrabyte grosse Szenen sind auf einem parallelen Cluster schon jetzt berechenbar, obwohl in diesem Fall die parallele Berechnungszeit nur der sequenziellen Zeit auf einem (heute allerdings noch hypotetischen) sequenziellen Rechner mit 1 Terrabyte Speicher entspricht.

2.2.4 Radiosity

Radiosity ist auch eine Methode zur Berechnung der globalen Beleuchtung, verwendet aber eine ganz andere Simulationstechnik als Ray-Tracing. Die Radiosity-Methode formuliert das Problem der globalen Beleuchtung als ein System von linearen Gleichungen, das von der Kajiyas Rendering-Gleichung unter bestimmten Annahmen abgeleitet werden kann. Die Methode macht zwei Annahmen:

1. Alle Energie-Transfers in der Szene sind diffuse.

2. Die Szene besteht aus finit vielen Elementen (aus *Polygon-Patches*), wobei die diffuse Ausstrahlung in jedem Punkt eines Patches konstant ist.

Bei der Radiosity-Methode wird nicht ein Bild berechnet, sondern die globale Beleuchtung explizit in dem Polygonnetz gespeichert.

Das System von linearen Gleichungen hat im Kontext der globalen Beleuchtung spezielle Eigenschaften. Vor allem die komplette Matrix ist Aufgrund des extrem hohen Speicher- und Rechenaufwand nicht explizit vorhanden. Die Elemente der Matrix berücksichtigen die diffuse Reflektionseigenschaften der Patches sowie ihre geometrische Lage (relative Sichtbarkeit und Entfernung). Die geometrische Lage zweier Patches wird durch einen *Form-Faktor* repräsentiert. Ein Form-Faktor F_{ij} zwischen zwei Patches P_i und P_j entspricht dem Faktor der vom Patch P_i ausgestrahlten Energie, die vom Patch P_j empfangen wird. Das Speichern aller Form-Faktoren ist bei großen Szenen nicht möglich.

Weil die Radiosity-Matrix nicht vorberechnet werden kann, können direkte Lösungsverfahren zum Lösung des Gleichungssystems nicht verwendet werden. Es müssen iterative Methoden, wie Gauss-Seidel-, Jacobi- oder Southwell-Relaxation verwendet werden, um eine approximierte Lösung zu bestimmen. Die Konvergenz der Methoden ist bei der Radiosity-Methode garantiert. Die Southwell-Relaxation hat eine physikalische Interpretation. Jedes Patch ist eine potentielle Lichtquelle. In jeder Iteration wird die noch nicht verschossene Energie eines Patches in die Umgebung ausgestrahlt und von allen anderen Patches empfangen. Diese *Shootings* werden iteriert, bis die globale unverschossene Energie klein ist. Die (approximierte) Lösung des Gleichungssystems befindet sich dann direkt in den Patches.

Ein wichtiger Vorteil der Radiosity-Methode ist ihre Blickpunktunabgängigkeit. Eine Kamera-Animation kann von der Radiosity-Lösung einfach durch eine Projektion der Szene auf dem Bildschirm und Interpolation der schon berechneten und im Polygonnetz gespeicherten Energie-Werte erzeugt werden. Diese Projektion und Interpolation kann von der Hardware von Grafikkarten schnell berechnet werden. Ein anderer Vorteil sind die Nachbearbeitungsmöglichkeiten. Zu der berechneten diffusen Beleuchtung der Szene können die fehlenden blickpunktabhängigen Effekte in einem unabhängigen Schritt berechnet werden, Texturen können auf die Objekte mappiert werden, usw. Die nach der Bearbeitung berechneten Bilder werden meistens zwar nicht mehr physikalisch korrekt, in der Filmproduktion ist aber die Flexibilität bei der Komposition wichtiger.

Die Radiosity-Methode hat auch ihre Nachteile. Ein Nachteil ist, dass die Szene aus Polygonen modelliert werden muss. Auflösung der polygonalen Repräsentation runder Objekte muss vor der Berechnung bestimmt werden. Ein weiterer Nachteil ist die Zeitkomplexität. Um eine verwendbare Lösung bei großen Szenen zu bekommen, muss die sequenzielle Berechnung Stunden bis Tage lang laufen. Die Speicherkomplexität kann schon bei kleinen Szenen sehr hoch sein, wenn eine sehr gute Qualität gewünscht ist – um die Beleuchtung im Polygonnetz fein genug speichern zu können, muss das Polygonnetz durch neue Punkte und neue Kanten verfeinert werden. Diese Verfeinerung findet immer dann statt, wenn die zweite Annahme der Radiosity-Methode (eine konstante Ausstrahlung in jedem Punkt des Patches) ohne die Verfeinerung nicht erfüllt wäre. Der ursprüngliche Speicherbedarf vervielfacht sich durch die von der Substrukturierung erzeugten Punkte. Bei großen Szenen kann die Speicherkomplexität zu Problemen beim Speichern, bei der Nachbearbeitung und beim Rendering führen. Weiter, um

eine gute Radiosity-Lösung zu bekommen, muss der Benutzer die Prinzipen der Radiosity-Methode kennen. (Selbst die Blickpunktunabgängigkeit kann ein Nachteil sein. Weil es während der Berechnung meistens nicht bekannt ist, welche Gebiete im 3D-Raum bei dem finalen Rendering wichtig werden, muss die gewünschte Qualität für alle Gebiete erreicht werden. Ein großer Aufwand wird bei der Berechnung oft in die Gebiete investiert, die später bei einer Kamera-Animation in keinem Bild sichtbar werden.)

Die Probleme bezüglich der Zeitkomplexität und der Speicherkomplexität während der Berechnung wurden im HiQoS-Projekt durch eine Daten-Parallelisierung der Radiosity-Methode adressiert. Es wurden auch zwei Komprimierungsmethoden entwickelt, welche die Speicherkomplexität der berechneten Radiosity-Lösungen reduzieren und damit eine nachträgliche Bearbeitung der beleuchteten Szene ermöglichen.

2.2.4.1 Paralleles Radiosity

Radiosity ist eine Methode zur Berechnung einer globalen diffusen Beleuchtung. Die Eingabe für die Radiosity-Methode ist ein 3D-Model (Geometrie), Materialien der Oberflächen und Lichtquellen. Eine Annahme der Methode ist, dass das 3D-Model aus finit vielen Polygonen besteht (*Patches*). Einige dieser Polygone sind Lichtquellen. Die Radiosity-Methode löst eine Energie-Balance Gleichung, die unter der Annahme von perfekt-diffusen Licht-Objektinteraktionen als ein lineares Gleichungssystem beschrieben werden kann:

$$\forall 1 \le i, j \le N : B_i = E_i + \rho_i \cdot \sum_{j=1}^{N} B_j F_{ji}$$

wobei

N ist die Anzahl von Patches

B_i ist die diffuse Ausstrahlung des Patches i (Radiosity)

E_i ist die (diffuse) Emissivität des Patches i (ungleich 0 bei Lichtquellen)

ρ_i ist die diffuse Reflektivität des Patches i (nicht absorbierter Teil der empfangenen Energie)

F_{ji} ist der Form-Faktor zwischen den Patches j und i (der Teil der von Patch j ausgestrahlten Energie, die vom Patch i empfangen wird)

Die Unbekannten sind $B_1, ..., B_N$.

Eine explizite Erstellung dieses Gleichungssystems ist extrem teuer (wegen der Berechnung der Form-Faktoren) und deshalb müssen spezielle iterative Lösungstechniken verwendet werden (Progressive Refinement). Eine Iteration entspricht physikalisch einer Ausstrahlung der Energie eines ausgewählten Patches in die Umgebung und dem Empfang der ausgestrahlten Energie von allen anderen Patches. Jedes Patch ist eine potentielle Lichtquelle (die empfangene Energie wird später ausgestrahlt). Das Ergebnis vom Radiosity ist eine mit der 3D-Geometrie verbundene Beleuchtungsinformation. Die existierenden kommerziellen Radiosity-Implementierungen (z. B. Lightscape) versuchen den sequenziellen Radiosity-Algorithmus zu

beschleunigen. Eine Alternative zur Beschleunigung ist die Parallelisierung des Algorithmus.

Das in der AG Monien entwickelte parallele Radiosity ist daten-parallel. Das bedeutet, dass die Geometrie an mehreren Prozessoren verteilt wird (das verteilte Speichermodel wird betrachtet). Die Daten-Parallelisierung ist bei größeren Szenen fast erzwungen. Der Grund ist sowohl die Berechnungszeit als auch die Speicherkomplexität der Radiosity-Lösungen. Die diffuse Beleuchtung wird in den Polygonen der Szene gespeichert (oft in den Endpunkten der Polygone). Eine Annahme der Radiosity-Methode ist, dass die diffuse Ausstrahlung (Radiosity) in allen Punkten eines Polygons gleich ist. Die ursprüngliche Diskretisierung der Szene ist normalerweise grob (d. h., die ursprünglichen Polygone sind groß). Wenn in einer Iteration eine Verletzung der gleichmäßigen Ausstrahlung innerhalb eines Patches entdeckt wird (weil z. B. eine Schattengrenze innerhalb des Patches liegt), wird das Patch in mehrere kleinere Patches zerlegt. Diese dynamische Substrukturierung macht die Radiosity-Methode sehr speicher-intensiv. Der Arbeitsspeicher eines Prozessors reicht bei größeren Szenen nicht aus.

Es wurden mehrere Datenverteilungsstrategien und mehrere Lastbalancierungsverfahren untersucht. Die Virtual-Walls-Strategie, die mit einer Hemicube-Berechnung der Form-Faktoren verbunden war, hatte Probleme mit der Berechnungsgenauigkeit. Die im Lauf des HiQoS-Projekts wurde für die Berechnung von Form-Faktoren eine Monte-Carlo Technik eingesetzt. Diese Technik basiert auf der Berechnung der Visibilität zwischen Sample-Punkten, die zufällig auf dem Shooter- und Empfänger-Patch generiert werden (sogenannte Ray-Tracing Technik zur Form-Faktor Berechnung). Die Objekt-Verteilung kann, aber muss bei dieser Technik nicht geometrisch sein. Für das Lastausgleich ist sogar besser, wenn die einzelnen Patches (Subobjekte) zufällig an die Prozessoren verteilt werden. Am Anfang der Berechnung wird die Szene diskretisiert und jeder Prozessor bekommt eine Menge von Patches. Jede Iteration besteht aus folgenden Schritten:

1. Durch einen zentralen Prozess wird ein Shooting-Patch mit der global höchsten noch nicht ausgestrahlten Energie ausgewählt.

2. Die Information über dem Shooting-Patch wird an alle Prozesse gesendet.

3. Die Energie des Shooting-Patches wird in jedem Prozessor ausgestrahlt, wobei die empfangenen und nicht ausgestrahlten Energien aller Patches in dem Prozessor aktualisiert werden. In diesem Schritt wird auch das Substrukturierungskriterium überprüft und die Empfänger-Patches werden nach dem Kriterium möglicherweise substrukturiert.

4. Die global unverschossene Energie wird in einem zentralen Prozess aktualisiert und das Terminierungskriterium wird überprüft. Wenn das Terminierungskriterium erfüllt wird, terminiert die Berechnung; wenn nicht, die Schritte 1-4 werden wiederholt.

Die (sequentielle) Form-Faktor Berechnung, die im Schritt 3 versteckt ist, wird durch eine zweifache Repräsentation der Szene beschleunigt. Zusätzlich zu der Verteilung einzelner Patches an die Prozessoren bekommt jeder Prozessor eine Repräsentation der gesamten Szene, wobei diese Gesamtrepräsentation aus großen planaren Polygonen besteht und deshalb nicht viel Speicheraufwand bedeutet. Für die Sample-Strahlen, die für die Visibilitätsberechnung verwendet werden, wird zuerst überprüft, ob sie von den großen Polygonen (verdeckt sind.

Durch die Visibilitätstests gegen große Polygonen (anstatt gegen einzelne Patches) wird die Berechnungszeit wesentlich verkürzt.

Um die parallele Berechnung zu beschleunigen, können die Iterationen (alle Schritte 1-4) asynchron laufen. Das bedeutet, dass ein Prozess, der mit seiner Iteration schon fertig ist, wartet an die globale Synchronisation (die im Schritt 2 versteckt ist) nicht, sondern wählt das Shooting-Patch anhand seiner lokalen Information, führt eine weitere Iteration durch und informiert durch eine Nachricht alle andere Prozesse. Diese Nachrichten werden von jedem Prozess empfangen und in einer Message-Queue gespeichert. In diesem asynchronen Szenario überprüft jeder Prozessor am Anfang jeder Iteration, ob seine Message-Queue leer ist. Wenn ja, wird eine lokale Iteration durchgeführt. Wenn nein, wird eine „externe" Iteration durchgeführt, indem die Energie eines Patches aus der Message-Queue lokal verschossen wird. Die Länge der Message-Queue ist eine gute Messeinheit der Last eines Prozesses. Diese Information wird für die Lastbalancierung benutzt [11, 14, 18].

Um eine gleichmäßige Lastbalancierung zu erreichen, wurde eine Technik von lokalen Farmen entwickelt. Wenn ein Prozess überlastet ist, kann er seine „Kopien" an mehreren unterlasteten Prozessoren starten. Dieser Prozess kümmert sich danach um die Verteilung der in seiner Message-Queue wartenden „externen Shootings" zwischen die neu gestarteten Kopien. Abbildung 9 zeigt den Unterschied zwischen der lastbalancierten und nichtbalancierten Berechnung.

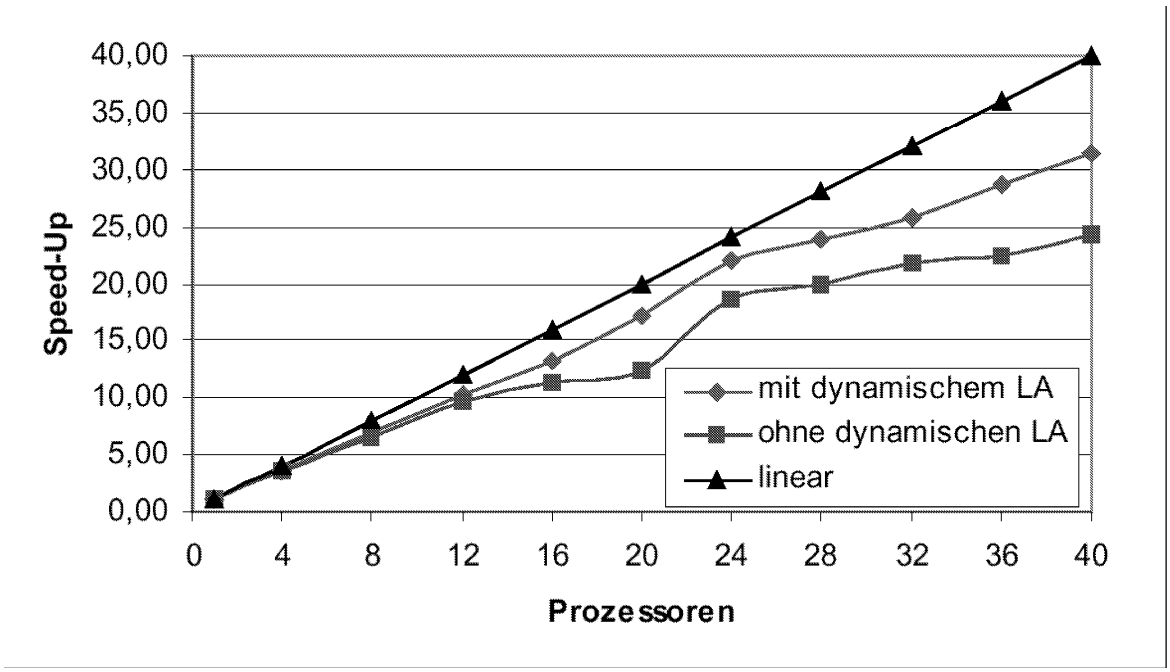

Abbildung 9: Radiosity Speedup-Messungen, mit und ohne dynamischem Lastausgleich. Die sequentielle Berechnungszeit: 1,5 Stunde.

2.2.4.2 Vereinfachung von Radiosity-Lösungen

Eine berechnete und gespeicherte Radiosity-Lösung muss letztendlich visualisiert werden. Vor der finalen Visualisierung wird die beleuchtete Szene oft noch nachbearbeitet. Die aus dem

Radiosity-Programm resultierenden großen Datenmengen stellen ein Problem bei diesen finalen Schritten vor. Die Daten-Parallelisierung des Radiosity-Algorithmus ermöglicht zwar eine Radiosity-Lösung zu berechnen (und im Speicher von vielen Prozessoren zu halten), die Nachbearbeitung und Visualisierung passieren aber typischerweise auf einem PC. Speicher des PCs ist kleiner als der gesamte Speicher der vielen Prozessoren. Die Szene muss ferner im Kontext von HiQoS Rendering-System zum Benutzer transferiert werden. Transfer von Daten von Größe über 100 MB kann problematisch werden.

Um die obenskizzierte Probleme zu lösen, wurden zwei Methoden zur Optimierung (Kompression) von Radiosity-Lösungen entwickelt: Mesh Decimation und Radiosity Textursynthese. Sequentielle als auch parallele Versionen beider Methoden wurden implementiert. Die Textursynthese Methode wurde als die bessere Alternative für das Upstart!-Szenario ausgewählt und im HiQoS Rendering-System integriert.

2.2.4.2.1 Mesh Decimation

Die Idee der Mesh Decimation Methode ist, iterativ Kanten aus dem Polygonnetz zu entfernen durch eine *Vertex-Unify Operation* (siehe Abbildung 10). Der Algorithmus selbst ist ziemlich einfach:

Mesh Decimation

INPUT: Polygonnetz M, gewünschte Kompressionsrate c

BEGIN

WHILE (Kompressionsrate c nicht erreicht)

BEGIN

(1) In M, wähle eine Kante $e=[P_1, P_2]$ zum Löschen

(2) Lösche die Kante e, indem Punkte P_1, P_2 in einen Punkt P zusammengefügt werden

(3) Finde die optimale Platzierung des neuen Punktes P

END

END

OUTPUT: Vereinfachtes Polygonnetz M

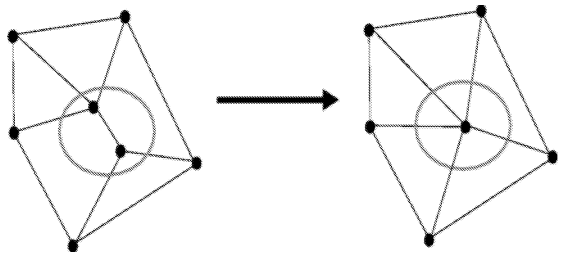

Abbildung 10: Vertex-Unify Operation

Zwei Fragen müssen beantwortet werden um diesen Algorithmus zu implementieren:

1. In welcher Reihenfolge sollen die Kanten gelöscht werden? (Schritt (1) des Algorithmus)

2. Wie berechnet man die optimale Platzierung des von der Vertex-Unify Operation resultierenden Punktes *P*? (Schritt (3) des Algorithmus)

Eine objektive Metrik wird verwendet um beide Fragen in einem Optimierungsschritt zu beantworten. Die Metrik evaluiert die Qualität des aktuellen (bereits vereinfachten) Polygonnetzes. Der Auswahl einer Kante, sowie die Platzierung des neuen Punktes P resultiert aus der Lösung eines Optimierungsproblems. Das Optimierungsproblem minimiert die Qualitätsreduktion über alle Möglichkeiten der Auswahl der Kante *e* und über alle mögliche Platzierungen des neuen Punktes *P*. Es gibt mehrere Möglichkeiten der Definition der objektiven Metrik. Die ausgewählte Metrik beeinflusst die Komplexität des Optimierungsproblems als auch die Visualisierungsqualität des vereinfachten Polygonnetzes. In der Implementierung wurde eine Quadrik-Metrik verwendet, die Vertex-Radiosities berücksichtigt [13, 16].

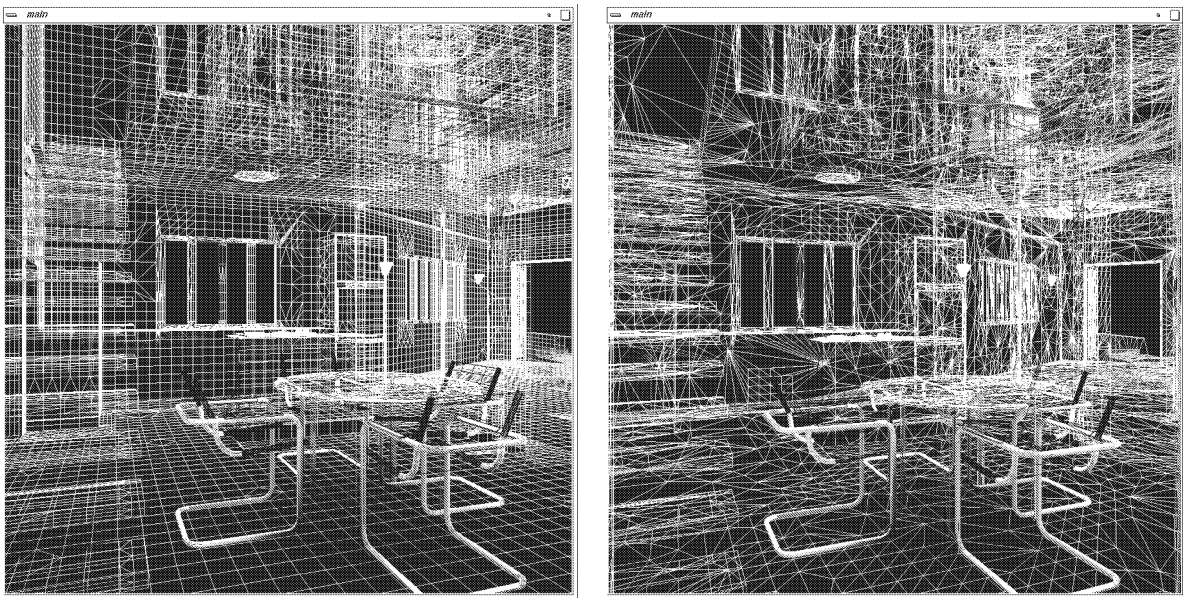

Abbildung 11: Mesh Decimation. Links: ursprüngliches Polygonnetz. Rechts: vereinfachtes Polygonnetz (90% der Punkte wurden entfernt).

Abbildung 12: Mesh Decimation. Links: Gouraud-Schattierung des ursprünglichen Polygonnetz. Rechts: Gouraud-Schattierung des vereinfachten Polygonnetz (90% der Punkte wurden entfernt). Der visuelle Unterschied ist kaum betrachtbar.

Ein Problem der Mesh Decimation Methode ist, dass die *subjektive* visuelle Qualität nicht garantiert werden kann. Die Methode kann man als eine nichtverlustfreie Kompression betrachten. Während bei einigen Szenen können Kompressionsraten von 90% erreicht werden ohne Verlust der visuellen Qualität zu merken (siehe Abbildung 11 und Abbildung 12), bei anderen Szenen und gleicher Kompressionsrate ist der visuelle Unterschied zwischen dem originalen und vereinfachten Polygonnetz zu groß. Die Kompressionsrate muss für jede Szene (Radiosi-

ty-Lösung) experimentell gefunden werden. Wegen dieser Interaktion ist die Methode für einen automatisierten Vorgang in einem Online Rendering-System nicht gut geeignet.

2.2.4.2.2 Radiosity Textursynthese

Die Radiosity Textursynthese ist eine verlustfreie Kompression von Radiosity-Lösungen. Die Kompression wird erreicht, indem eine effizientere Datenstruktur für die Beleuchtungsrepräsentation verwendet wird. Während der Radiosity-Berechnung wird das ursprüngliche Polygonnetz adaptiv substrukturiert um die Beleuchtung genau speichern zu können. Diese Substrukturierung erzeugt neue Punkte, in den die Beleuchtungsinformationen (Vertex-Radiosities) gespeichert sind. Die Idee der Textursynthese ist, die Beleuchtung in Texturen zu speichern und die Substrukturierung zu entfernen [8]. Eine Textur wird für jedes Objekt generiert. In den verbleibenden Punkten des Polygonnetzes werden UV-Koordinaten gespeichert, die das Mapping der Textur auf das Objekt definieren.

Radiosity Textursynthese

INPUT: Polygonnetz M mit Vertex-Radiosities

BEGIN

FOR (alle Objekte obj_k im Polygonnetz M)

BEGIN

FOR (alle Patches p_i im Objekt obj_k)

BEGIN

(1) Finde optimale Auflösung für Radiosity-Textur des Patches p_i

(2) Erzeuge Radiosity-Textur $pmap_i$ für das Patch p_i

(3) Fülle Radiosity-Textur $pmap_i$ für das Patch p_i

END

(4) Anordne Patch-Texturen $pmap_i$ in Objekt-Textur $omap_k$

(5) Entferne Substrukturierung in Patches p_i des Objektes obj_k

(6) Weise UV-Koordinaten den Patches p_i des Objektes obj_k zu

END

END

OUTPUT: Vereinfachtes Polygonnetz M (ohne Substrukturierung) mit Texturen zugewiesen den Objekten

Die optimale Auflösung (Zeile (1)) ist die minimale Auflösung, die eine verlustfreie Repräsentation der Beleuchtung ermöglicht. Diese Auflösung ist von der Tiefe und Form der Substrukturierung abhängig. Der entsprechende Substrukturierungsbaum wird vollständig durchgesucht um die optimale Auflösung einer Patch-Textur zu bestimmen (Abbildung 13).

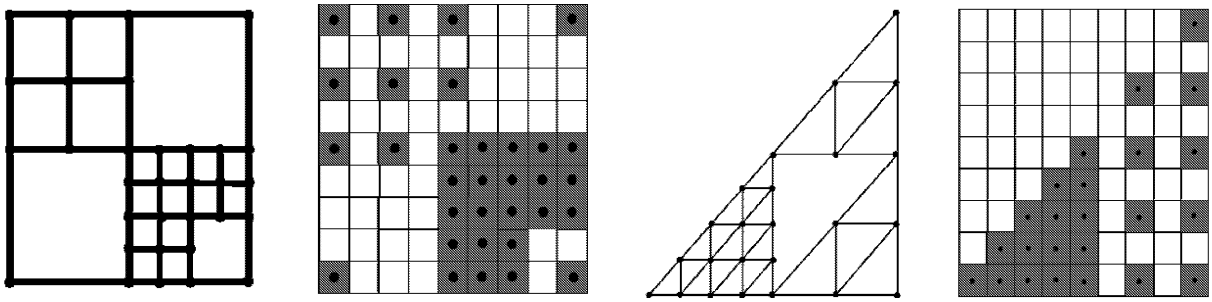

Abbildung 13: Illustration der optimalen Patch-Textur Auflösung und Abbildung der Vertex-Radiosities in die Textur. Links: ein viereckiges Patch. Rechts: ein dreieckiges Patch.

Eine optimale Anordnung der Patch-Texturen in eine Objekt-Textur (Zeile (4) des Algorithmus) ist ein NP-hartes Problem. Eine Heuristik wird in diesem Schritt benutzt. Die Heuristik versucht die Objekt-Textur so klein wie möglich zu halten indem der Platz in der Objekt-Textur mit den Patch-Texturen effizient ausgenutzt wird (Abbildung 14).

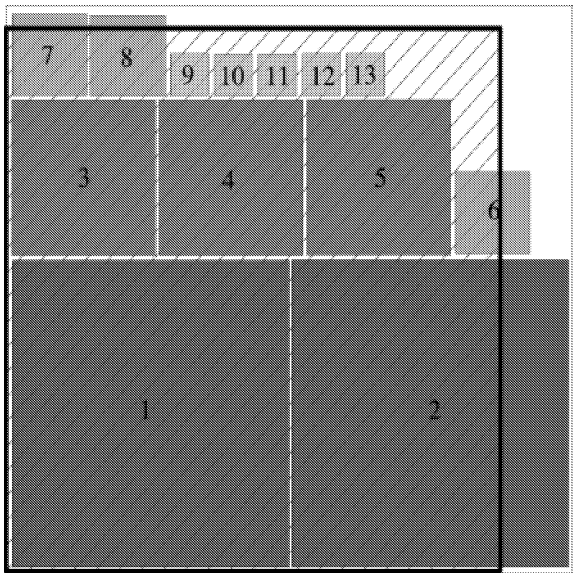

Abbildung 14: Anordnung der Patch-Texturen in eine Objekt-Textur. Der schraffierte Bereich entspricht der gesamten Fläche aller Patch-Texturen. Die gesamte Fläche von einzelnen Patch-Texturen ist eine untere Schranke für die Größe der Objekt-Textur.

2.2.5 Service Broker Architektur

Zum Zwecke der Evaluierung stellte die Universität Paderborn (PC^2) den Zugang zu dem leistungsfähigen Parallelrechner hpcLine zur Verfügung, auf den über das Internet zugegriffen werden kann. Auf diesem Parallelrechner lief die Beleuchtungssimulation für Berechnungen von Bildern (Ray-Tracing) als auch global beleuchteten 3D-Modellen (Radiosity).

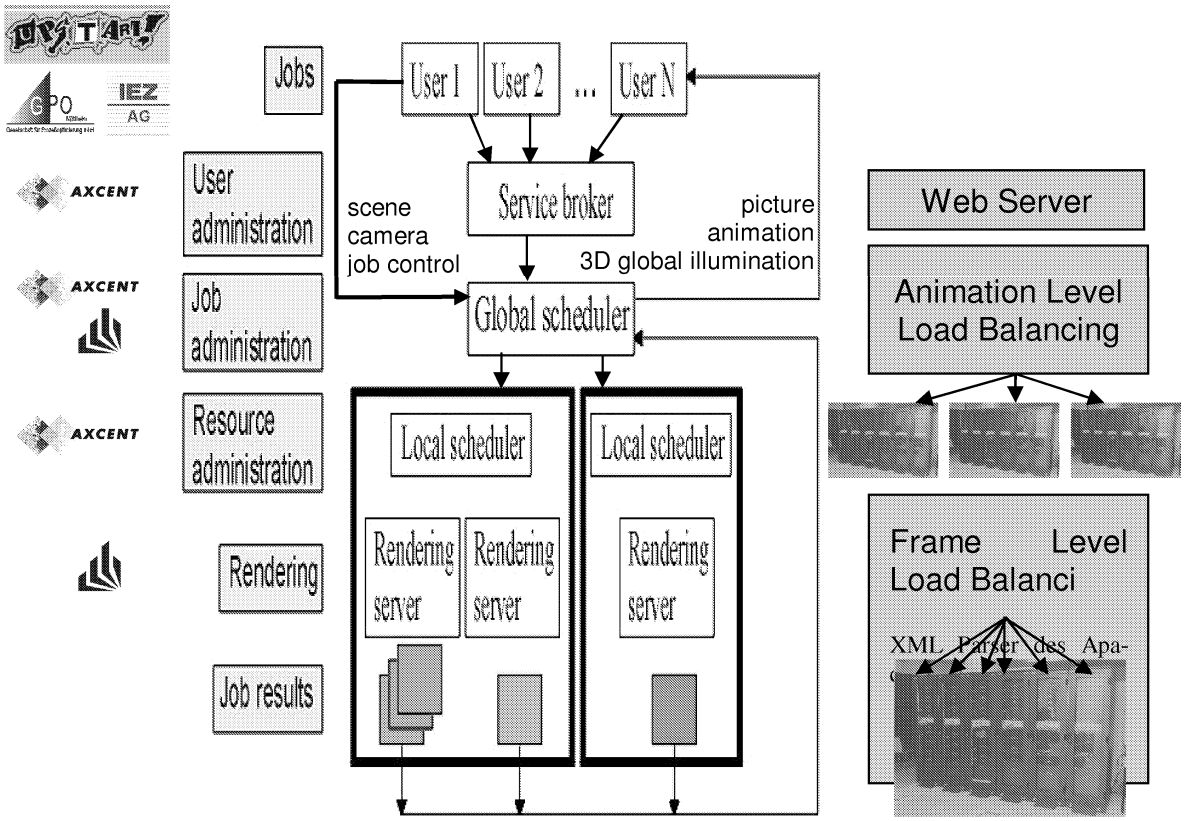

Abbildung 15: Das HiQoS-Rendering-Szenario

Das Rendering-System (siehe Abbildung 15) besteht aus den vier wesentlichen Komponenten Client-System, Service-Broker, Scheduling-System und Rendering-Netzwerk.

2.2.5.1 Client

Bei dem *Client* handelt es sich um ein Anwender Tool zur Anmeldung bei dem Service-Broker, zum Abschicken von Rendering-Aufträgen und zur Übergabe von 3D-Modellen, die Benutzer mit Hilfe einer Modellierungssoftware (Arcon, Speedikon, 3D Studio Max) erstellt haben.

Bei der Evaluierung kam ein web-basiertes Interface (Web Browser) zum Einsatz. Die Nutzung des Renderingdienstes erfordert von einem Anwender die Anmeldung durch Email Adresse und Passwort. Neukunden müssen auf einer separaten Webseite zuvor einige Angaben machen.

Für die Bearbeitung eines Auftrags werden vom Anwender alle erforderlichen Parameter auf den entsprechenden Formularen abgefragt und von Service-Broker in einer Datenbank gespei-

chert werden. Die Übergabe von Szenarien an das Scheduling-System erfolgt über ein IP-basiertes Netzwerk in Form von Dateien, die auf einem Webserver zu platzieren sind.

Das Absenden eines neuen Auftrags ist nur eine Funktionalität des Web Interfaces. Auf einer weiteren Webseite wird dem Anwender die Möglichkeit geboten, sich einen Überblick über alle Aufträge und deren Bearbeitungsstatus erschaffen zu können. Von der Übersichtsseite aus besteht die Möglichkeit, einen bereits berechneten Auftrag mit geänderten Parameter erneut zu stellen, Einsicht in die Ergebnisse zu nehmen oder den Auftrag aus der Datenbank zu löschen.

2.2.5.2 Service-Broker

Der *Service-Broker* ist die Schnittstelle zwischen Client-Systemen und dem Rendering-Service. Seine Aufgabe liegt in der Administration des Rendering-Services und Validierung der Zugriffsberechtigung eines Benutzers.

Als Internet/Intranet-Dienst realisiert, ist der Service-Broker die primäre Schnittstelle zwischen Client-Systemen und Rendering-Dienst. Als Teil einer verteilten Auftragsabwicklung bestehen seine Aufgaben in der Administration des Dienstes und der Auftragsbearbeitung. Als Service für eine geschlossene Benutzergruppe besteht die Administration des Dienstes aus einer Verwaltung der Benutzer und deren Berechtigung für einzelne Bereiche des Rendering Dienstes. Die Auftragsbearbeitung besteht aus den folgenden Teilaufgaben:

- Annahme von neuen Aufträgen,
- der Speicherung von Auftragsdaten und Metainformationen,
- dem Löschen von Aufträgen und
- das Anzeigen von Ergebnissen

Bei der Evaluierung kam als Service-Broker ein Apache Webserver mit PHP3 Erweiterung und eine MySQL Datenbank zum Einsatz. Die Spezifikation eines Auftrags besteht aus der Ausfüllung und Bestätigung eines HTML Formulars. Nach der Bestätigung fragt der Service-Broker die Globalen Scheduler, wie viele Aufträge sich in Ihrer Warteschlange befinden. Informationen über die Länge der kürzesten Warteschlange wird vom Service-Broker dem Benutzer mitgeteilt, der den Auftrag noch einmal bestätigen oder abbrechen muss.

Nach Bestätigung des Auftrags generiert der Service-Broker eine eindeutige Auftrags ID und übergibt den Auftrag dem globalen Scheduler. Nach Fertigstellung eines Auftrags baut der globale Scheduler, zwecks Übertragung von Auftragsdaten und Ergebnissen, eine Verbindung zum Service-Broker auf.

2.2.5.3 Scheduling-System

Das *Scheduling-System* ist ein verteiltes System von lokalen und globalen Schedulern die räumlich getrennt sein können. Die Vorteile dieses Ansatzes liegen in seiner Skalierbarkeit, Ausfallsicherheit und der Lastverteilung.

Die Aufgaben des Scheduling-Systems und deren Aufteilung auf den jeweiligen Schedulertyp wurde in der folgenden Tabelle zusammengefasst:

Aufgabe	Verantwortlich
Verwaltung des Rendering-Netzwerks	lokaler Scheduler
Auftragsbearbeitung	globaler Scheduler
Scheduling von Jobs	globaler Scheduler
Berechnung von Teilaufträgen	globaler Scheduler
Reservierung von Ressourcen	lokaler Scheduler

Bedingt durch die hierarchische Struktur, siehe Abbildung 15, des HiQoS Rendering-Systems meldet sich ein Scheduler bei Systemstart bei der nächst höheren Instanz an. Der globale Scheduler bei einer Menge von Service Broker und der lokale Scheduler bei einer Menge von globalen Scheduler.

Die Struktur des Scheduling-Systems wird durch Konfigurationsdateien beschrieben. Änderungen der Konfiguration und der Struktur sind dynamisch zur Laufzeit möglich. Um eine optimale Ausnutzung des Systems zu garantieren und die Ressourcen zu schonen, protokollieren beide Scheduler im laufenden Betrieb den Status eines Auftrags und wie viele Teilbilder bereits berechnet wurden.

2.2.5.4 Rendering-Server

Der Begriff *Rendering-Server* kennzeichnet ein verteiltes System von Software-Komponenten zur Simulation der globalen Beleuchtung von Radiosity und Ray-Tracing Szenarien für die Zielsysteme SUN Workstation Cluster und dem Parallelrechner hpcLine des PC² der Universität Paderborn.

Die Software-Komponenten werden hinsichtlich ihrer Funktionalität als Frontend- bzw. Backend-Komponente klassifiziert. Ein Beispiel für eine Frontend-Komponente ist der zentrale Prozess zur Synchronisation und Überwachung aller laufenden Prozesse.

Jeder Rendering-Server wird, nach Reservierung der benötigten Ressourcen (Rechner), vom lokalen Scheduler durch Aufruf von Programm start_RendServ gestartet. Die benötigten Auftragsdaten werden, auf mehrere Verzeichnisse verteilt, unterhalb des Wurzelverzeichnisses $(HOME)/execute vom lokalen Scheduler im Filesystem abgelegt. Eine Unterscheidung hinsichtlich Radiosity oder Ray-Tracing gibt es dabei nicht.

Die nachfolgende Tabelle zeigt, welche Daten in welchem Unterverzeichnis abgelegt werden, <i> steht synonym für den i-ten Job und <n> für das n-te Teilbild:

Verzeichnis	Inhalt
job.<i>	allgemeine Auftragsdaten, z.B. die Szenebeschreibung
job.<i>/<n>	auftragsspezifische Daten, z.B. die Kameraposition
job.<i>/result	Berechnungsergebnis (Bild oder Beleuchtungsmodell)

Die vom Rendering-Server gelieferten Ergebnisse werden über den lokalen Scheduler an den globaler Scheduler geschickt und, nur im Ray-Tracing Fall, auf die Daten für die nächste Berechnung gewartet. Sind keine weiteren Bilder zu berechnen, so wird der Rendering-Server durch den lokalen Scheduler terminiert. Aus Performance Gründen wird bei Ray-Tracing Szenario die Szene on Rendering-Server nur einmal eingelesen und bis zur Terminierung im Speicher gehalten.

2.2.6 Datenkonversion

3D-Szenen können mit unterschiedlichen Visualisierungsprogrammen erstellt werden, z.B *speedikon, AutoCAD, SoftImage, Lightwave, 3DS Max*, usw. Je nach Einsatzgebiet der Software werden sehr verschiedene 3D- und Animationsfeatures implementiert. So kann in AutoCAD eine Zeichnung bemaßt werden, es fehlt jedoch an Animationshierarchien wie Inverse Kinematik zwischen Objekten. Jede Software definiert ihr eigens Dateiformat um die jeweiligen Features speichern zu können. Diese Dateiformate sind in der Regel geheim und können von anderen Programmen nicht gelesen werden. Es haben sich zwar inzwischen diverse Austauschformate wie DXF und IGES etabliert, jedoch können damit nie alle Features unterstützt werden.

Die Auswahl für ein geeignetes Dateiformat, um 3D-Szenen im HiQoS Projekt austauschen zu können, wurden folgende Kriterien unterworfen:

- Verfügbarkeit der Dateiformatspezifikation.
- Hoher Verbreitungsgrad
- Plattform-Unabhängigkeit
- Keine Lizenzgebühren
- Möglichst breite Unterstützung von gängigen 3D- , Material- und Animationsfeatures

Die erste Wahl fiel dabei auf das VRML 2.0 (VRML 97) Format, da es sich, ähnlich dem HTML Format, im Internet etabliert hat, es von einer Vielzahl von Programmen unterstützt wird und von einem Export-Konverter des Programms *speedikon* (bzw. *Arcon*) sowohl auch von einem Import-Konverter der Universität Paderborn derzeit schon unterstützt wurde. Die Beschreibung der Geometrie konnte von der VRML 2.0 Formatspezifikation direkt übernommen werden. Die Material- und Lichtbeschreibung ist im VRML 2.0 Format jedoch nur sehr rudimentär. Im Ray-Tracing-Szenario wurde für den Datenaustausch zwischen der Universität Paderborn und den Firmen IEZ AG und GPO mbH ein *erweitertes VRML 2.0* Format defi-

niert, in das alle vom *speedikon* (*Arcon*) unterstützten Material- und Lichteigenschaften exportiert werden können.

Für das Radiosity-Szenario wurde das 3DS (Autodesk 3D Studio Release 3.0) Format als das Austauschformat zwischen der Firma Upstart! Filmproduktion GmbH und der Universität Paderborn definiert, da es im besonderen eine gute Unterstützung für Licht- Material- und Animationsparameter bietet. Die Firma Autodesk Deutschland GmbH hat freundlicherweise das 3D Studio File Toolkit zur Verfügung gestellt, in dem das Format beschrieben ist und der volle Quellcode zum Lesen und Schreiben des 3DS Formates zur Verfügung steht. Für den Austausch der im 3DS-Format fehlenden Features (Vertex-Radiosities, Area-Lights) wurden spezielle Dateiformate definiert.

Für den Austausch der (2D-) Bilddaten (Texturen, gerenderte Bilder) wurden Standartbildformate (BMP, TGA, TIFF, JPEG) verwendet. Im Gegensatz zu den 3D-Szeneformaten sind fast alle Bildformate frei zugänglich. Sie bieten ferner ähnliche bzw. gleiche Features und es existieren Konvertierungsprogramme, die fast jeden Bildformat in jeden anderen konvertieren können.

2.2.6.1 VRML 2 Schnittstelle

Für den Datenaustausch der Szenen zwischen *speedikon* (oder *Arcon*) und dem HiQoS Rendering-System wird das erweiterte VRML 2.0 Format verwendet. Die Geometrie der Objekte wird in *speedikon* durch ein Polygonnetz repräsentiert. Polygonnetze (Meshes) sind im VRML 2.0 Format unterstützt (inklusive Vertex-Normalen und Texturkoordinaten). Viele Material- und Lichtquelleneigenschaften sind im VRML 2.0 allerdings nicht vorhanden. Alle relevante Material- und Lichtquelleneigenschaften werden deshalb von *speedikon* als „Kommentaren“ exportiert. Diese Kommentaren werden zwar von einem üblichen VRML-Browser ignoriert, sie erhalten aber die vollständige *speedikon*-Information aus der das ursprüngliche 3D-Model rekonstruiert werden kann. Die folgenden Kommentaren werden von dem Client-Programm *Arcon* (bzw. *speedikon W*) exportiert und vom Konverter des HiQoS Rendering-Systems interpretiert:

- [WorldInfo] #GlobalIntensityFaktor <float>
  Ein Globaler Faktor zur Skalierung der Intensität aller Lichtquellen.

- [WorldInfo] #Ambientlight <float> <float> <float>
  Intensität des Ambienten Lichts (RGB).

- [WorldInfo] #Sunlight <float> <float> <float>
  Intensität der Sonne (RGB).

- [WorldInfo] #Sundirection <float> <float> <float>
  Richtung der Sonnenstrahlen (3D Vektor).

- [WorldInfo] #Shadelight <float> <float> <float>
  Intensität der „zweiten Sonne” (RGB).

- [WorldInfo] #Shadedirection <float> <float> <float>

Richtung der „Zweiten-Sonnenstrahlen“ (3D Vektor).

- [material] #DiffuseColor <float> <float> <float>
  Diffuse Farbe, d. h. „Pigment” (RGB).

- [material] #AmbientReflection <float>
  Ambienter Reflektionsfaktor.

- [material] #DiffuseReflection <float>
  Diffuser Reflektionsfaktor.

- [material] #TexturedFlag [0 | 1]
  1 wenn eine Textur (Bitmap) verwendet wird, sonst 0.

- [material] #TexturemaskFlag [0 | 1]
  1 wenn die Geometrie durch eine Textur (Bitmap) maskiert wird (entspricht der Alpha-Kanal Technik), sonst 0.

- [material] #MixedtextureFlag [0 | 1]
  1 wenn die Farbe der Textur mit dem Pigment gemischt wird, sonst 0.

- [material] #SpecularColor <float> <float> <float>
  Farbe der spekularen Highlights (direkten spekularen Reflektion).

- [material] #SpecularReflection <float>
  Intensität der spekularen Highlights (direkten spekularen Reflektion).

- [material] #SpecularFlag [0 | 1]
  1 wenn die spekularen Highlights verwendet werden, sonst 0.

- [material] #PhongSize 3
  Glättungskoeffizient, der in der Highlights-Berechnung verwendet wird.

- [material] #TransparencyFlag [0 | 1]
  1 wenn das Objekt transparent ist, sonst 0.

- [material] #Transparency <float>
  Transparenzfaktor.

- [material] #IOR <float>
  Index of refraction (Bruchindex).

- [PointLight] #LightingInfluence <float>
  Einflussbereich einer Lichtquelle.

- [PointLight] #LightingInfluence <float> <float> <float>
  Verschwächung einer Lichtquelle mit der Distanz (wie im OpenGL Model).

2.2.6.2 Kamerabeschreibung

2.2.6.2.1 HiQoS Kamera-Format

Um einem (Ray-Tracing) Benutzer des HiQoS Rendering-Systems zu ermöglichen, Kameras und Kamera-Pfade zu beschreiben, wurde ein *interner Kamera-Format* spezifiziert. Das Format ist ASCII, ist ziemlich einfach und übersichtlich und wird normalerweise automatisch (von einem Programm auf der Seite des Benutzers) generiert. Die Beschreibung einer einzelnen Kamera besteht aus Position, Blickpunkt, Öffnungswinkel und Neigung. Die Beschreibung eines Kamera-Walkthroughs besteht aus einer Sequenz einzelner Kameras (*Keyframe-Kameras*), wobei jeder Kamera ein vom Benutzer definierter Zeitpunkt zugeordnet wird. Eine Animationsrate wird auch vom Benutzer spezifiziert. Die Kameras zwischen Key-Frames werden entsprechend der Animationsrate automatisch durch eine Interpolation erzeugt. Zwei Arte der Interpolation werden unterstützt: lineare Interpolation und Spline-Interpolation. Alle Parameter der Key-Frame Kameras werden interpoliert. Abbildung 16 zeigt die Möglichkeiten der Interpolation.

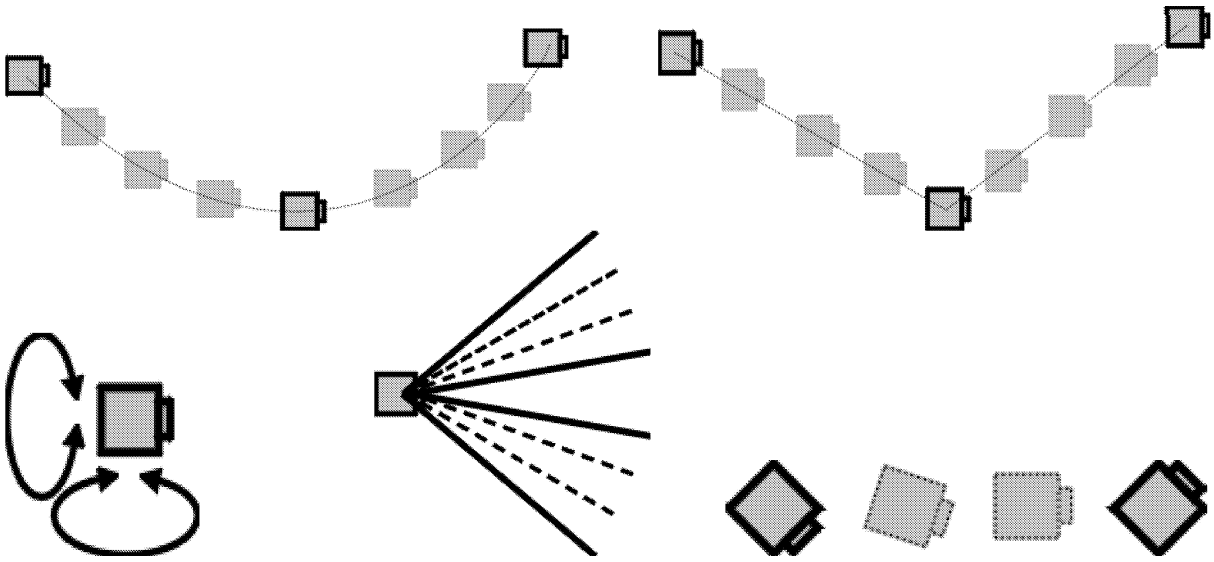

Abbildung 16: Kamera Interpolation.
Oben: Pfad-Interpolation (spline und linear).
Unten: Drehunginterpolation, Öffnungswinkelinterpolation, Neigungsinterpolation.

2.2.6.2.2 Anbindung an *speedikon*

Die Kameras (bzw. Kamera-Pfade) werden in einer separaten Datei gespeichert (also nicht in der VRML Datei mit dem Rest der Szene). Die Kamera-Beschreibung wird im Kamera-Format des HiQoS Rendering-Systems gespeichert (siehe Abschnitt 2.2.6.2.1). Die Kamera-Datei wird von *speedikon* automatisch generiert. Im Visualisierungsmodus des Programms *speedikon* (bzw. *Arcon*) kann der Benutzer die aktuelle Perspektive speichern (Abbildung 17). Mit Hilfe eines externen Programms können aus den in *Arcon* gespeicherten Kameras die Kameras ausgewählt werden, die für einen HiQoS Rendering-Job verwendet werden.

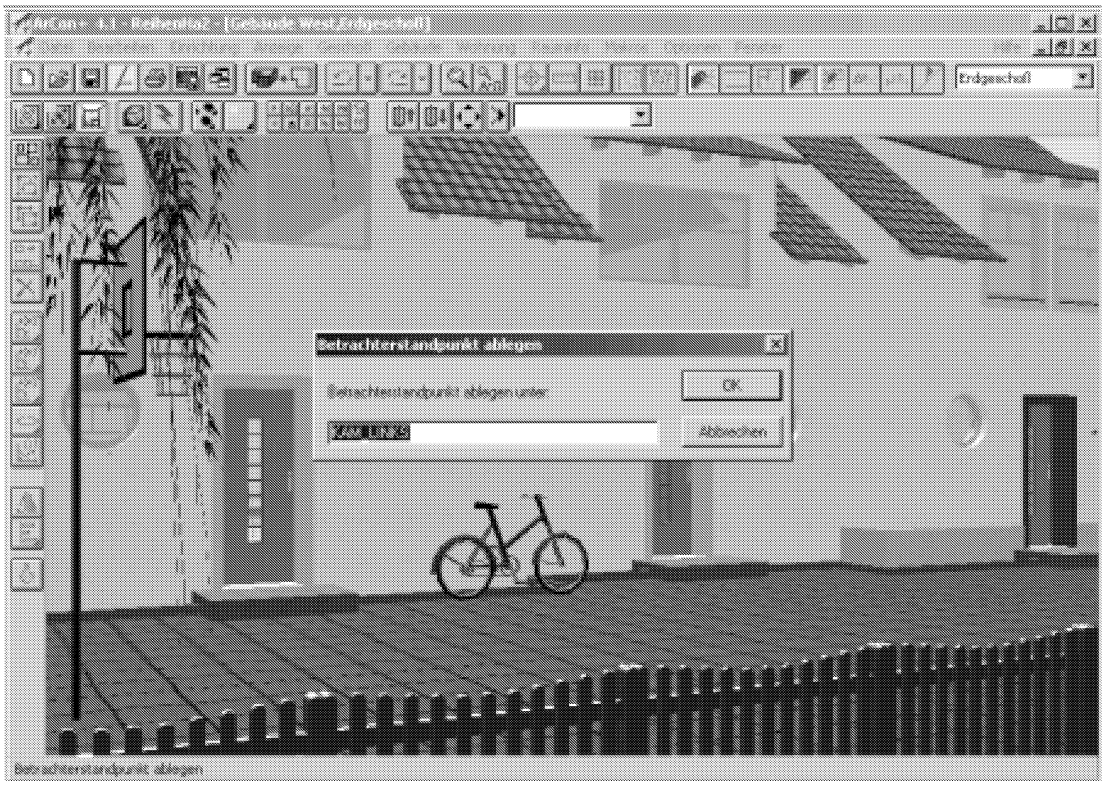

Abbildung 17: Definition einer Kamera in *Arcon*.

Das obenbeschriebene Szenario kann auch für die Definition von Kamerawegen benutzt. Ein Kamera-Pfad wird durch ein paar Kameras (Stütz-Kameras) repräsentiert, die auf dem Pfad liegen. Der Benutzer kann die Stütz-Kameras definieren und bei jeder einen Zeitpunkt eingeben. Das Ergebnis der Kamera-Pfad Spezifikation illustriert die folgende Tabelle:

Zeit	**Kamera-Bezeichnung**
0 Sek.	Kamera_0
2 Sek.	Kamera_1
8 Sek.	Kamera_2
9 Sek.	Kamera_3

Die Tabelle zeigt eine Kamera-Animation, die 9 Sekunden lang ist und durch 4 Stütz-Kameras definiert ist. Der Benutzer muss noch zusätzlich sagen, wie viele Bilder pro Sekunde gerendert werden sollen (z. B., 25 Bilder/Sek.). Die fehlenden Kameras werden automatisch durch Interpolation der Stütz-Kameras berechnet. Zum Beispiel, die Kamera in Zeitpunkt 1 Sek. wurde vom Benutzer nicht spezifiziert. Diese Kamera wird automatisch erzeugt, so dass sie „in der Mitte“ zwischen Kameras Kamera_0 und Kamera_1 liegen wird.

Das Problem dieses Szenarios ist die Steuerung der Kamera-Geschwindigkeit. Die Geschwindigkeit der Kamera ist durch die Zeitpunkte (die linke Spalte in der Beispiel-Tabelle) definiert.

Der Benutzer muss sich den zeitlichen Ablauf sehr gut vorstellen können. Es ist ziemlich schwierig, eine konstante Kamera-Geschwindigkeit in unterschiedlichen Stütz-Kamera-Segmenten (z. B. Kamera_0-Kamera_1 und Kamera_1-Kamera_2) zu erreichen.

Um dem Benutzer eine intuitivere Kontrolle bei der Definition der Kamera-Pfade zu geben, kann das interne „Film-Modul“ des Programms *Arcon* benutzt werden. Der *Arcon* Benutzer klickt auf den „Record“ Knopf, bewegt sich in der 3D-Szene und am Ende klickt auf „Stop“. Der Kamera-Pfad wird von *Arcon* aufgenommen und kann nachher in *Arcon* gespielt werden. Die Aufnahme wird in einer .WLK Datei gespeichert, die der Tabelle aus dem vorherigen Beispiel entspricht. Die Kamerageschwindigkeit ist allerdings konstant (wenn es der Benutzer so will) und eine Vorschau der Kamera-Animation steht dem Benutzer zur Verfügung ohne *Arcon* verlassen zu müssen. Die Beschreibung des .WLK Formats wurde der Universität Paderborn gegeben. Von der Universität Paderborn wurde ein Programm geschrieben, das eine von *Arcon* erzeugte .WLK Datei in das HiQoS Kamera-Format konvertiert.

2.3 Netzwerke von Multimedia-Servern

2.3.1 Medien-Server mit QoS-Support

Das heutige Internet-Kommunikationsparadigma arbeitet nach dem *Best-Effort*-Prinzip. Dies behandelt alle Datenströme gleichrangig. Die Anforderungen isochroner Medienströme können mit diesem Paradigma nicht erfüllt werden. Aus diesem Grund wurden innerhalb des Projektes Protokolle zum Aushandeln von *Quality-of-Service-(QoS)*-Parametern (z.B. Bandbreite, Jitter). in einen Medienserver und verschiedene Clients integriert. Damit können Anwendungen realisiert werden, die die zeitsynchrone Wiedergabe unterschiedlicher Medienströmen ermöglicht. So können beispielsweise in Schulungsfilmen zeitgleich zu audiovisuellen Informationen Texte im HTML-Format angezeigt werden. Dies ermöglicht die dargestellten Inhalte pointiert zusammenzufassen, die aktuelle Szene im Gesamtzusammenhang darstellen zu können und somit das Lernergebnis der Schüler zu verbessern.

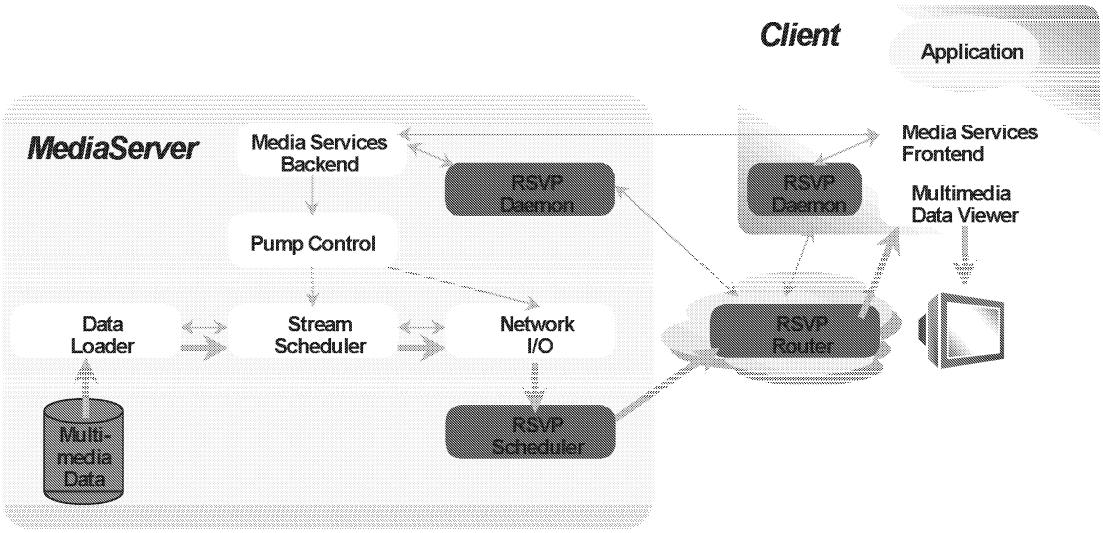

Abbildung 18: HiQoS-Media-Server mit RSVP-Unterstützung

In einem heterogenen Netzwerk, das neben den multimedialen Daten auch andere Daten transportieren soll, müssen Mechanismen für die Einhaltung von QoS umgesetzt werden. Dieser Ansatz wird in HiQoS verfolgt und stützt sich *auf RSVP (Resource ReSer-Vation Protocol)* (vgl. Abbildung 18). RSVP erlaubt es, Ressourcen für Multicast oder Unicast Datenströme zu reservieren. Es handelt sich um ein Signalisierungsprotokoll, das dazu dient, auf jeder Komponente entlang eines Netzwerkpfades Zustandsinformationen über reservierte Ressourcen zu übermitteln.

In HiQoS erfolgt die erste Kommunikation mit dem Client auf Basis von RTSP (Real-Time Streaming Protocol). Sobald ein multimedialer Datenstrom angefordert wird, erhält der Client über RTSP die zur Übertragung notwendigen Ressource-Information (Bandbreite,...). Nach erfolgter Reservierung über das RSVP-Protokoll wird mit Hilfe von RTSP der Datenstrom gesteuert (Start, Stop,...), die Datenpakete mit RTP (Real-Time Transport Protocol) versendet und Statusinformation wie z.B. die Anzahl der Paket-Verluste mit Hilfe von RTCP (Real-Time Control Protocol) kommuniziert.

Die Serverplattform besteht aus weitgehend plattformunabhängigen C++ Code und weist eine Aufteilung der Architektur in zahlreichen Objekten, die entlang eines Datenpfades angeordnet werden, auf. In Abbildung 19 sind die grundlegenden Bausteine dieser Plattform dargestellt.

Eine Datenquelle wird durch ein *Data Source Object* realisiert. Ein solches Objekt besitzt und steuert zu einem beliebigen Zeitpunkt jeweils ein zugehörendes *Source Stream Object* und ein *Data Source Thread Object*. Das *Source Stream Object* entspricht der Abstraktion einer Datenquelle (sei es Festplatte oder Live-Kamera), von der das *Source Thread Object* (ein parallel laufender Unterprozess) Daten lesen kann und in einer dem *Data Source Object* zugeordneten FiFo-Datenschlange puffern kann. Spezielle Memory-Management-Techniken erlauben es, dass, falls ein paralleler Benutzer den selben Strom beinahe zeitgleich benutzen möchte, die Daten bereits im Hauptspeicher vorhanden sind und von anderen Datenquellen nicht noch mal physikalisch von Platte gelesen werden müssen.

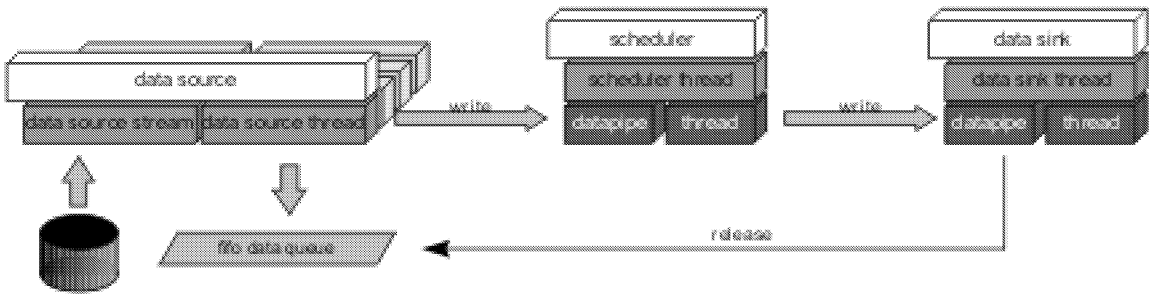

Abbildung 19: Serverplattform, grundlegende Bausteine

Die von einer Datenquelle eingelesenen Daten werden einem Datenpfad entlang von mehreren Objekten (die in der Regel auch parallel ablaufende Threads darstellen) verarbeitet. Es sind dies insbesondere ein Scheduler und ein *Data Sink Object* (siehe Abbildung 19 und Abbildung 20). Da die Verarbeitung durch diese Komponenten asynchron verläuft, werden die Daten in der Warteschlange erst bei expliziter Rückmeldung durch das letzte Objekt des Datenpfades entfernt. Eine Datenquelle erzeugt und verwaltet somit bei jedem Start-/Stop-Befehl eine neues Stream-/Thread-Paar.

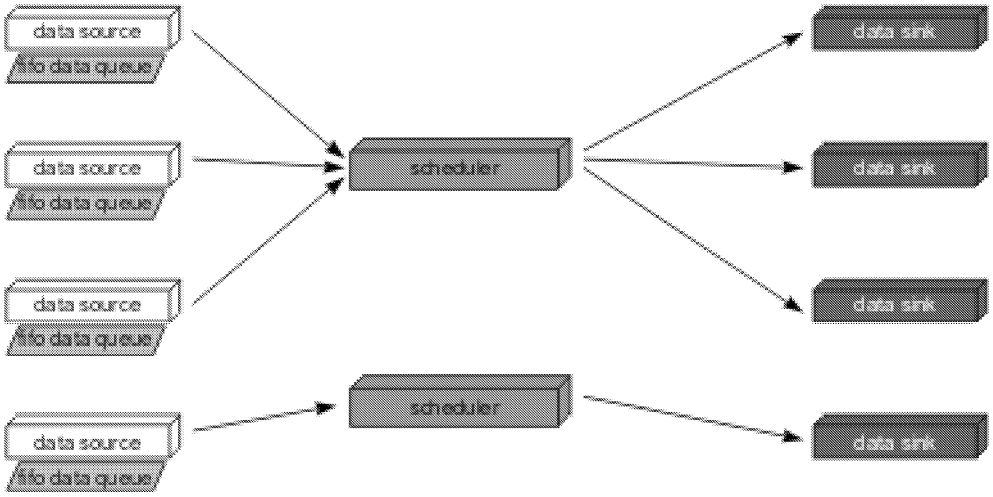

Abbildung 20: Serverplattform, mögliche Konfiguration im SMP-System

In Abbildung 20 ist dargestellt, wie mit der vorgestellten Architektur auf eine verschiedene Anzahl von Benutzer und evtl. auch Systemeigenschaften eingegangen werden kann. Typischerweise laufen die Daten von einer Datenquelle über einem Scheduler zu einer Datensenke. Eine Datenquelle liest Blöcke in für das Dateisystem effizienter Blockgröße ein und übergibt sie mit Zeitstempelinformationen dem Schedulingsystem. Dieser wird bei Eintreffen des Sendezeitpunktes die Daten an eine zugehörige Datenquelle zur Verarbeitung schicken.

Datenquellen lesen größere Blöcke ein und können für verhältnismäßig lange Zeit von einem verarbeitenden in einen wartenden Zustand übergehen und dann kaum Serverlast erzeugen. Die größte Last wird demnach von dem Schedulingsystem und der Datensenke erzeugt. Je nach Systemausstattung und Leistungsmerkmalen kann eine Anpassung des Systems die ge-

wünschten Leistungen erbringen. Idealerweise wird pro Prozessorknoten ein Schedulerobjekt instantiiert, das synchron Datensenken bedient und damit nahezu die gesamte Last verursacht. Gibt es mehrere Netzadapter, so ist es durchaus sinnvoll, auch Datensenken im asynchronen Modus zu betreiben und die Last zwischen Scheduler und Senken aufzuteilen.

Diese Architektur zeichnet sich durch ihre besondere Flexibilität aus. Wie in Abbildung 21 dargestellt können an beliebiger Stelle eines Datenpfades Objekte in der Verarbeitung der Datenpakete eingeschaltet werden. Ein Verschlüsselungsobjekt etwa könnte zur sicheren Übertragung dienen, aber auch multiplexer und Fan-Out-Datensenken zur exakten Replikation der Daten an unterschiedliche Netzsegmente sind denkbar.

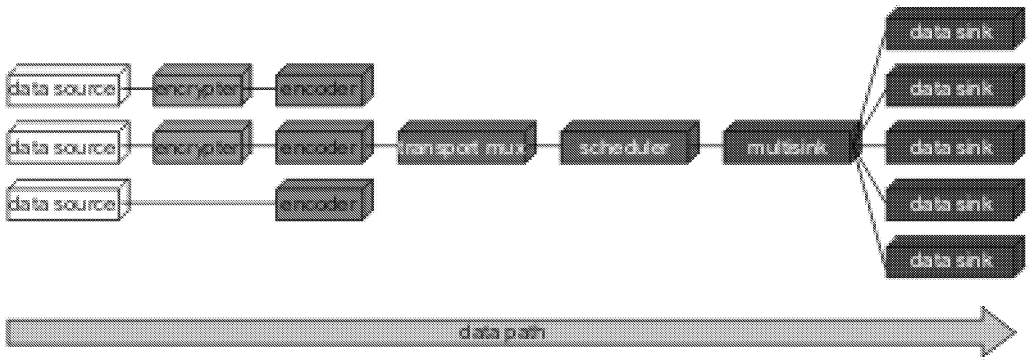

Abbildung 21: Weitere Komponenten im Datenpfad

Die Unterstützung der unterschiedlicher Protokolle (RTCP, RTP RSVP) wird durch die entsprechenden *Data Sink Objects* erreicht.

2.3.2 Content Management

Eine wichtige Problemstellung des HiQoS-Projektes ist die Entwicklung von Algorithmen zum effizienten Management skalierbarer Netzwerke von Media-Servern [1]. Derartige Server-Netzwerke werden relevant, falls breit-bandige Datenströme (Audio und Video hoher Qualität) an eine große Anzahl weit verteilter Client-Systeme ausgeliefert werden sollen. In solchen Szenarien sind zentralistische Ansätze mit nur einem Server aus offensichtlichen Gründen nicht zu realisieren.

Im Servernetzwerk können einzelne Medienobjekte in verschiedenen Qualitätsstufen (Bitraten) zur Verfügung stehen; u.U. ist sogar ein Anpassung der Bitrate (Transcoding) zur Laufzeit des Systems sinnvoll. Dadurch wird das gesamte Serversystem flexibler und ermöglicht so, zum Beispiel von einem Medienobjekt eine Version zu erstellen, die mit geringer Bitrate codiert ist und deshalb auch über schmalbandige Verbindungen übertragen werden kann. Dadurch können unter Umständen Kopiervorgänge eingespart werden, falls keine hohe Qualität beim Abspielen gefordert ist.

Ziel des Content Management ist nun, abhängig von der Popularität eines Medienobjektes dieses in möglichst hoher Bitrate möglichst nahe zu den daran interessierten Nutzern zu speichern. So sind Inhalte für die Benutzer jederzeit in optimaler Qualität abrufbar. Dazu sind

möglicherweise häufige Kopiervorgänge notwendig, zum Beispiel wenn große Mengen neuer Medienobjekte eingespielt werden oder sich das Nutzerverhalten geändert hat.

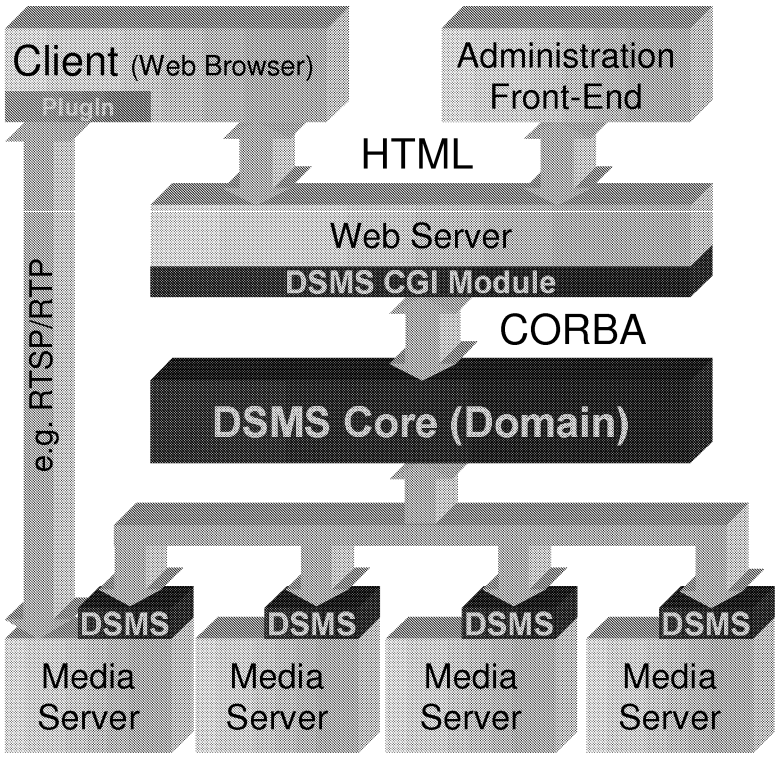

Abbildung 22: DSMS Systemarchitektur

Für die Aufgaben des Content Management in einem solchen Netzwerk von Media-Servern wurde das *Distributed Server Management System (DSMS)* entwickelt, eine Software, die die Verwaltung großer Netzwerke von Medienservern ermöglicht und die automatische Plazierung von Medienobjekten im Servernetzwerk ermöglicht, falls dies erforderlich ist. Das DSMS [2] basiert auf verteilten Modulen (s. Abbildung 22), die untereinander über CORBA-Schnittstellen kommunizieren, und dient zur Verwaltung des Server-Netzwerks. Die Software ermöglicht die Verwaltung beliebig vieler Media-Server und verbindet jeden Client automatisch mit dem am besten für die Übertragung geeigneten Server. In vielen Szenarien ist dies der räumlich am nächsten zum Clienten stehende Media-Server. Das Einspielen und Löschen von Inhalten auf den einzelnen Servern erfolgt über entsprechende CORBA-Schnittstellen, die zum Beispiel aus einer HTML-Anwendung (über CGI oder Java-Applets) aufgerufen werden können. Das DSMS lässt sich durch seine Architektur (standardisiertes CORBA-Protokoll) einfach in andere Anwendungen (z.B. Web-Applikationen) einbinden.

Aufgabe ist es daher, die verfügbaren Medienobjekte derart auf die Server zu verteilen, dass Nutzer möglichst alle für sie interessanten Videoströme von ihrem lokalen Server (d.h. der nächstgelegene Server im LAN) abrufen können.

Es wurden zwei verschiedene Szenarien untersucht. Im ersten Szenario (statisch) wird eine Menge von Medienobjekten auf ein "leeres" Server-Netzwerk aufgespielt. Hier wird untersucht, wie eine möglichst optimale Plazierung der Medienobjekte im Server-Netzwerk aussieht. Im zweiten Szenario (dynamisch) geht es darum, während des Betriebs des Systems

neue Medienobjekte einzuspielen und andere zu löschen und aufgrund der sich dadurch ergebenden Änderungen des Gesamtsystems eine Neuberechnung der optimalen Plazierung durchzuführen. Das in beiden Fällen entstehende Optimierungsproblem ist NP-vollständig und daher ist es (abhängig von der Problemgröße) sehr zeitaufwendig, eine optimale Lösung zu finden [3]. Daher musste auf die Berechnung der exakten Lösung verzichtet werden. Stattdessen wurde auf Heuristiken zurückgegriffen, die für die Lösung anderer Optimierungsprobleme (z.B. File-Allocation Problem) genutzt werden: die sog. Simulated Annealing Methode. Optimierungsziel ist die Qualität eines Medienobjektes (d.h. die Bitrate), auf das Nutzer zugreifen möchten zu maximieren. Dies wird durch Simulated Annealing unter Berücksichtigung der Randbedingungen (verfügbare Bandbreite und Speicherplatz auf den einzelnen Servern) gelöst [4,5].

Da im HiQoS Szenario kein reales Netzwerk mit einer hinreichend großen Anzahl an Servern und Medienobjekten zur Verfügung stand, wurde die Performance der Algorithmen simulativ ermittelt. Dazu wurden unterschiedliche Anzahlen von Medienobjekten, Servern und verfügbarem Speicherplatz auf den Servern sowie unterschiedliche Zugriffsmuster (normal- bzw. exponential-verteilt) verwendet. Für das statische Szenario erreichte der Simulated Annealing basierte Ansatz eine Abweichung von der optimalen Lösung zwischen 13.34 % bis zu 2,8 %, abhängig von der Anzahl der Medienobjekte und dem zur Verfügung stehenden Speicherplatz.

In [6] wurden verschiedene Batching-Algorithmen basierend auf Multicast-Verfahren daraufhin untersucht, inwieweit eine Steigerung der Performance über die Optimierung der Plazierung hinaus erzielt werden kann. Hierzu wurden unterschiedliche Multicast-Verfahren (lokales und globales Multicast) untersucht. Die Performance der Verfahren wurde ebenfalls simulativ ermittelt. Die Ergebnisse legen nahe, dass es eine signifikante Steigerung der Performance durch Multicast-Verfahren insbesondere in Zeiten hoher Last zu erzielen ist d.h. wenn viele Nutzer auf Inhalte zugreifen. Simulativ konnte eine Steigerung der Performance von bis zu 15% festgestellt werden, was den Erfolg der Multicast-Verfahren deutlich macht.

2.3.3 HiQoS Client (Media Player)

Die Dekodierung von MPEG-Filmen auf Client-Seite ist zunächst mittels einer eigenen Implementierungen erreicht worden, die auf die *Microsoft-ActiveMovie*-Architektur aufgesetzt. Dafür konnte eine vollständig kompatible RSVP-Anbindung für das Betriebssystem Windows 2000 erreicht werden.

Um die Verwendbarkeit der HiQoS-Anwendungen auch mit anderen außer Windows-basierten Client-Systemen sicher zu stellen, wurde ausgehend von der aktuellen *Java Media Framework (JMF)* Distribution eine Java-Version des HiQoS-Clients entwickelt. Der Client ist sowohl als Applet für Browser als auch als Stand-Alone Version verfügbar. In HiQoS wird dieser insbesondere im Live-Szenario (Abschnitt 2.4.6) zum Einsatz gebracht. Seine vollständige Integration mit RSVP stellte sich allerdings als weitaus problematischer dar, als zunächst angenommen. Eine komplette Lösung dieses Problems wird über den Abschluss des Projekts hinausgehen.

Alternativ zu dieser Entwicklung wurde die Anbindung des *Real*-Client-Systems von *Real-Networks* an den HiQoS-Mediaserver untersucht. Der Client konnte angebunden werden (allerdings ohne QoS-Unterstützung), was die Interoperabilität des HiQoS-Systems unter Beweis

stellt.

2.3.4 Advance Reservation

Im Servernetz ist nicht nur eine möglichst günstige Platzierung von Medienobjekten entscheidend, sondern ebenfalls die für die Streaming- und Kopiervorgänge notwendige Bandbreite. Wenn es zum Beispiel notwendig wird, aufgrund veränderten Nutzerverhaltens die vorhandenen Medienobjekte neu zu platzieren, muss auch entsprechende Bandbreite im Netzwerk zur Verfügung gestellt werden, um die u.U. erheblichen Datentransfers bewältigen zu können. Darüber hinaus ist es wichtig, ein Medienobjekt zu einem bestimmten Zeitpunkt an einem bestimmten Server verfügbar zu haben (beispielsweise für eine Präsentation). In diesem Fall muss für den Kopierprozess von einem entfernten Server ausreichende Bandbreite zu Verfügung stehen, um ein Medienobjekt bis zur geforderten Zeit (Deadline) auf den lokalen Server eines Nutzers zu transferieren. Auch falls eine bestimmte Deadline nicht existiert, ist es sinnvoll, Nutzern des Systems für den Fall eines Dateitransfers den Zeitpunkt der Verfügbarkeit am lokalen Server mitzuteilen und diese Zeitvorgabe auch einhalten zu können. Dies kann ein wichtiges Kriterium für die Entscheidung eines Nutzers sein, ein Medienobjekt von einem entfernten Server anzufordern bzw. auf die Anforderung zu verzichten. Hierzu wurden Mechanismen für die Reservierung von Bandbreite im voraus, d.h. bevor die Bandbreite tatsächlich genutzt wird, entwickelt und ins DSMS integriert [2]. Im DSMS steht die Funktionalität jedoch nur bei entsprechender Unterstützung von Reservierungen im Netzwerk zur Verfügung.

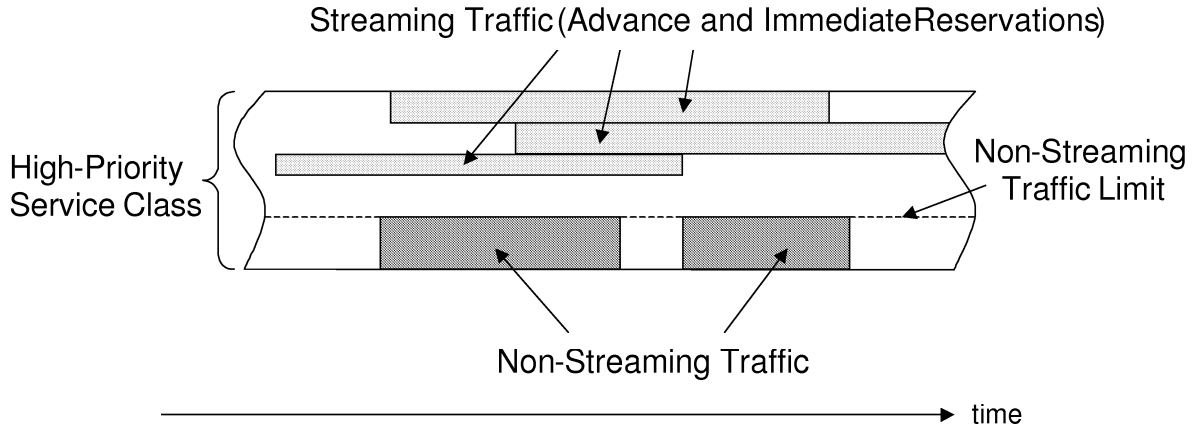

Abbildung 23: Dateitransfer und Streaming in einer Serviceklasse

Falls Bandbreite auch im Backbone reserviert werden kann (ggf. durch Anmieten eigener Leitungen oder durch Bereitstellung einer DiffServ-Serviceklasse, s. Abbildung 23), ist es möglich, die notwendigen Übertragungen von Medienobjekten und die Streaming-Vorgänge derart zu planen, dass stets ausreichend Bandbreite für die Übertragungen bereit steht und alle Deadlines bzw. Zeitvorgaben eingehalten werden können. Hierzu können über eine CORBA-Schnittstelle des DSMS Reservierungen für Dateitransfers vorgenommen werden. Sowohl bei der Reservierung für das Streaming von Videos als auch bei automatisch vorgenommenen Dateitransfers wird durch entsprechendes Scheduling im DSMS sichergestellt, dass die verfügbare Bandweite nicht überbeansprucht wird.

Es sind drei verschiedene Arten der Reservierung zu unterscheiden:

1. Transfer von Medienobjekten durch explizite Nutzeranforderung.

2. Explizite Anmeldung von Streaming Prozessen durch Nutzer.

3. Automatischer Transfer von Medienobjekten aufgrund neu berechneter Platzierung (s. Abschnitt 2.3.2).

Alle Reservierungen werden nach dem First-Come-First-Served-Prinzip behandelt, d.h. für die erste ankommende Anforderung wird die gewünschten Bandbreite bereitgehalten.

Für den Fall, das auf Netzwerkebene keine Bandbreite exklusiv reserviert werden kann, können für das Streaming keine Reservierungen von Bandbreite vorgenommen werden und es können keinerlei Garantien für Übertragungsdauer gemacht werden. Allerdings stellt das DSMS im letzteren Fall sicher, dass nur jeweils ein Kopiervorgang pro Link stattfindet, so dass Dateiübertragungen sich gegenseitig nicht beeinflussen können.

2.3.5 Access Management

Die automatische Verteilung von Informationen birgt das Problem in sich, dass der unberechtigte Zugang zu diesen verteilten Daten verhindert werden muss. Jeden Informationsserver einzeln mit einem alleinigen Zugriffskontrollmechanismus zu schützen stellt dabei einen zu hohen Administrationsaufwand dar und wirft zusätzlich erhebliche Konsistenzprobleme auf. Erforderlich ist hier vielmehr ein einheitlicher Netzwerk-Dienst für die Zugriffskontrollfunktion.

Zusätzlich sollen eine Vielzahl unterschiedlicher Informationsdienste geschützt werden. Da ein und derselbe Benutzer in den unterschiedlichen Informationsdiensten auch unterschiedliche Rollen einnehmen kann ist ein flexibles, rollenbasiertes Zugriffskontrollsystem notwendig.

Das *HiQoS Access Management System* ermöglicht die einheitliche Verwaltung von Zugriffsrechten für alle HiQoS Informationsdienste. Es verwendet ein flexibles, fein-granulares Rollenmodell. Die Client-Server-Kommunikation des sicherheitssensitiven *HiQoS Access Control Service* ist durch Server-Zertifikate und ein geeignetes Verschlüsselungsverfahren bei der Übermittlung der Daten übers offene Netzwerk geschützt.

Das *HiQoS Access Management System* besteht aus folgenden Komponenten:

- ein *Access Directory,* in dem die Zugriffsrechte und -regeln für alle HiQoS Informationsdienste gespeichert werden,
- ein *Access Management Server*, über den die Zugriffsrechte im Access Directory abgefragt und gesetzt werden können,
- Administrationswerkzeuge zur Einstellung und Verwaltung der Zugriffsrechte und
- Module, mit denen die HiQoS-Informationsserver (HTTP-server, RTSP-Server) Zugriffsrechte auswerten und setzen können.

Jeder HiQoS-Informationsserver überprüft mit einem *Access Control Component Module,* ob ein anfragender Benutzer die erforderlichen Zugriffsrechte hat, bevor er die gewünschten Ak-

tionen ausführt. Werden Inhalte eines HiQoS Informationsservers modifiziert, so müssen diese Veränderungen mit Hilfe eines *Content Administration Component* beim *Access Directory* registriert werden.

Abbildung 24 zeigt die Einbettung der Access-Management-Komponenten in die HiQoS-Systemarchitektur.

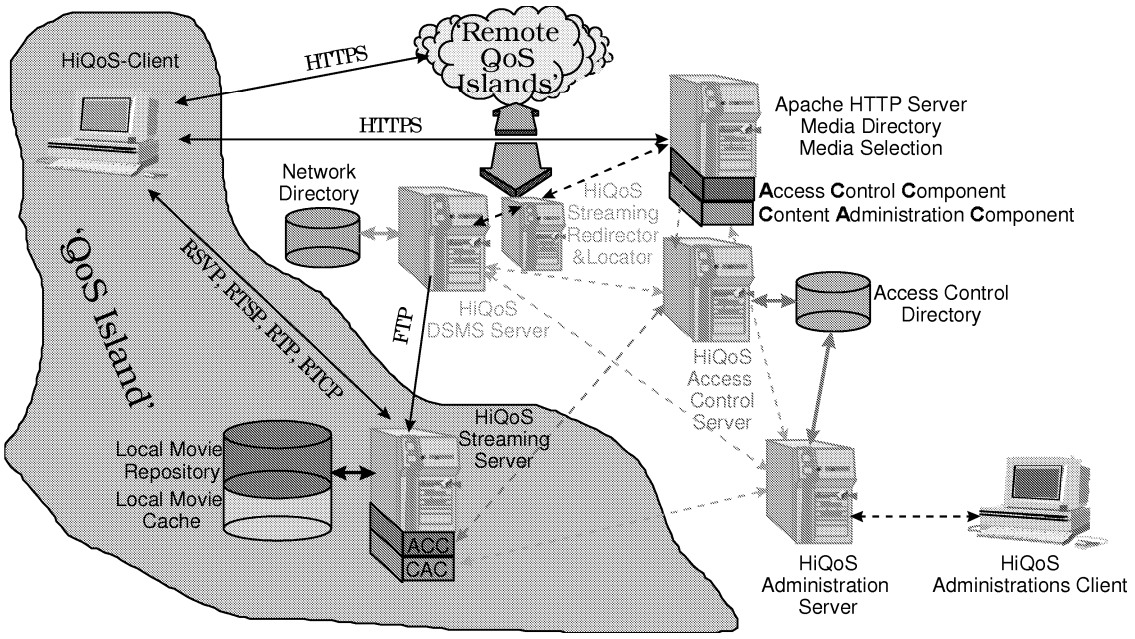

Abbildung 24: Access Management System und HiQoS-Systemarchitektur

2.3.5.1 Funktionale Beschreibung

Das Access Management System von HiQoS verfügt über folgende Eigenschaften:

- *Trennung von Authentisierungs- und Autorisierungsfunktion*, d.h. unterschiedliche Module für Authentisierung und Autorisierung. Für das *HiQoS Access Management System* liegt der Fokus auf der Autorisierung bzw. der Implementierung der Zugriffskontrollfunktion. Die Verifizierung der Benutzer-Identität (Authentisierung) wird durch geeignete bestehende Dienste realisiert. Die Art des verwendeten Authentisierungsdienst ist jedoch nicht fest definiert. Somit kann aus einer Vielzahl bestehender (oder auch neuer) Authentisierungsdienste gewählt werden.

- *Einheitliche Authentisierung des Benutzers möglich*, d.h. ein Benutzer kann den gleichen Prozess zum Nachweis seiner Identität bei allen HiQoS-Informations-Diensten anwenden, wenn ein entsprechender einheitlicher Authentisierungsdienst eingesetzt wird.

- *Einheitliche, zentrale Verwaltung der Zugriffsrechte*, d.h. es wird ein generisches Werkzeug zur Verfügung gestellt, mit dem die Zugriffsrechte für alle Informationsdienste des HiQoS-Szenarios in einer gemeinsamen Zugriffsrecht-Datenbank eingestellt und verändert werden können (d.h. um eine Rolle mit Zugriffsrechten auf *n* Informationsserver zu etablieren, müssen nicht mehr *n* Informationsserver konfiguriert werden sondern nur entspre-

chende Einträge in der Zugriffsrecht-Datenbank)

- *Kombinationsmöglichkeit mehrerer verteilter Server zu einem logischen Informationsdienst*, d.h. verteilte Server eines Informationsdienstes (auch unterschiedlichen Typs) können mit dem *HiQoS Access Management System* integriert verwaltet werden, so dass für sie einheitliche Benutzerrechte gelten.

- *Flexible, Service-spezifische Einstellbarkeit der Zugriffsrechte*, d.h. das Zugriffsrechtschema, das dem *Access Management System* zugrunde liegt, ist flexibel genug, um unterschiedliche Anforderungen der einzelnen Informationsdienste zu erfüllen. Anpassungen an die Spezifika im Zugriffsrechtsschema des jeweiligen Informationsdienstes sind mit entsprechenden Administrationswerkzeugen möglich.

- *Dezentrale, inhaltsbasierte Zugriffskontrolle*, d.h. jeder Informationsserver kontrolliert selbst den Zugang auf die von ihm zur Verfügung gestellten Informations-Ressourcen mit Hilfe eines speziell hierfür vorgesehenen Netzwerkdienstes.

2.3.5.2 AMS Module

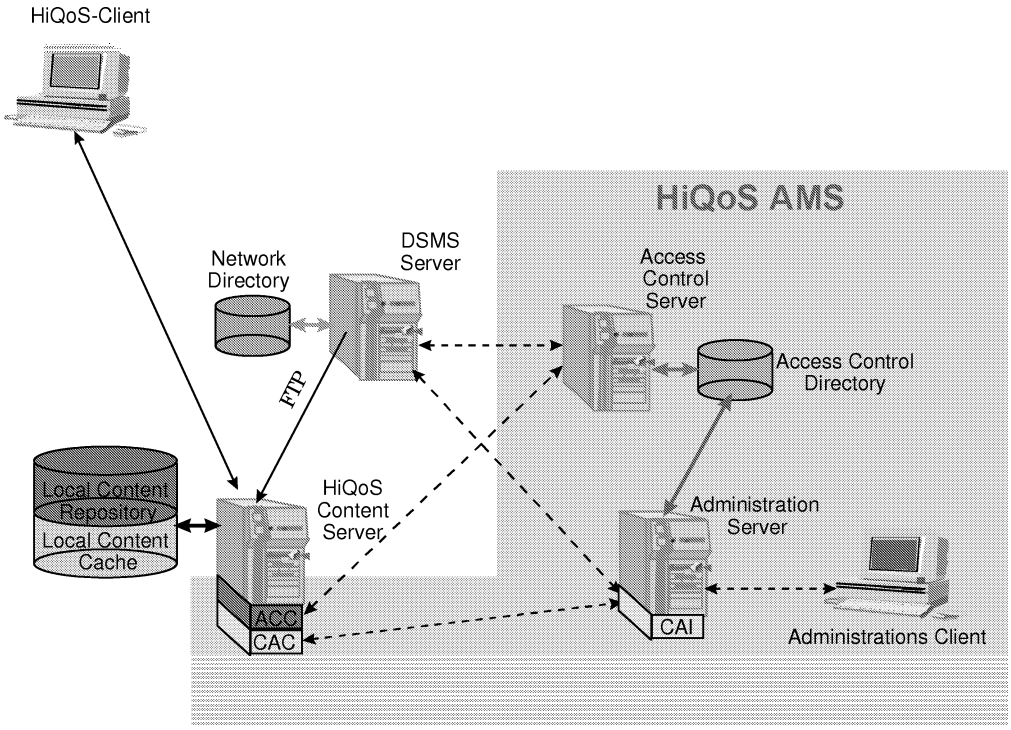

Abbildung 25: AMS Module

Im *HiQoS Access Management Systems (AMS)* sind mehrere (verteilte) Module erforderlich, die im Zusammenspiel die o.g. Eigenschaften gewährleisten können. Die Module des HiQoS AMS und ihre Aufgaben sind im einzelnen:

- *Access Directory*: Alle Zugriffsrechtinformationen der einzelnen Informationsdienste werden im einheitlichen *Access Directory* gespeichert.

- *Access Control Server*: Über den generischen *Access Control Server* kann jeder Informationsdienst Zugriffsrechte im Access Directory für seine Informationsinhalte und seine Benutzer abrufen und modifizieren.

- *Access Control Components*: Jeder Informationsserver überprüft mit einer geeigneten *Access Control Component* die Zugriffsrechte des anfragenden Benutzers. Diese ruft hierfür die Zugriffsrechte des anfragenden Benutzers für den verlangten Informationsdienst und -inhalt ab, vergleicht diese mit der Anfrage und erteilt oder verweigert entsprechend den Zugriff.

- *Administration Tool*: Mit dem *Administration Tool* erfolgt der Eintrag und die Modifikation von Zugriffsrechtdaten im *Access Directory*. Das *Administration Tool* besteht wiederum aus zwei Modulen: Einem *Administrationsserver*, der die zur Administration der Zugriffsrechte erforderlichen Interaktionen mit dem *Access Directory* ausführt, und einer *grafischen Benutzeroberfläche* mit einem entsprechenden *Administration Client*. Die grafische Benutzeroberfläche umfasst folgende funktionale Komponenten:

  User Management, zur Verwaltung der allgemeingültigen Einträge der einzelnen HiQoS Benutzer (z.B. Name, Adresse, Authentisierungsinformation, etc.) ,

  Service Registration, zur Initialisierung eines neuen HiQoS-Informations-Service und zur Modifikation der Charakteristika bestehender HiQoS-Informationsdienste,

  Service Management, zur Definition und Modifikation der spezifischen Zugriffsrechtstrukturen eines einzelnen HiQoS Informationsdienstes, zur Verwaltung von Service-spezifischen Benutzerrechten, und zur Registrierung von zu schützender Informationsinhalte eines Informationsdienstes im Access Directory.

Durch das Zusammenspiel dieser Module wird der Zugriff eines Benutzers auf einzelne Informationsinhalte geschützt. Fordert ein Benutzer bei einem HiQoS-Informationsserver eine Ressource an, so fragt dieser zunächst beim *HiQoS Access Control Server* die aktuellen Benutzerrechte ab, vergleicht diese mit den Forderungen des Benutzers, und gewährt bzw. verweigert daraufhin den Zugriff. Der *HiQoS Access Control Server* stützt sich dabei auf Einträgen im *HiQoS Access Directory* ab. Mit dem *Administration Tool* können die Zugriffsrechte für den Informations-Service definiert und aktualisiert werden.

Abbildung 26 gibt einen Überblick über die Module des *HiQoS Access Management* und ihr Zusammenspiel.

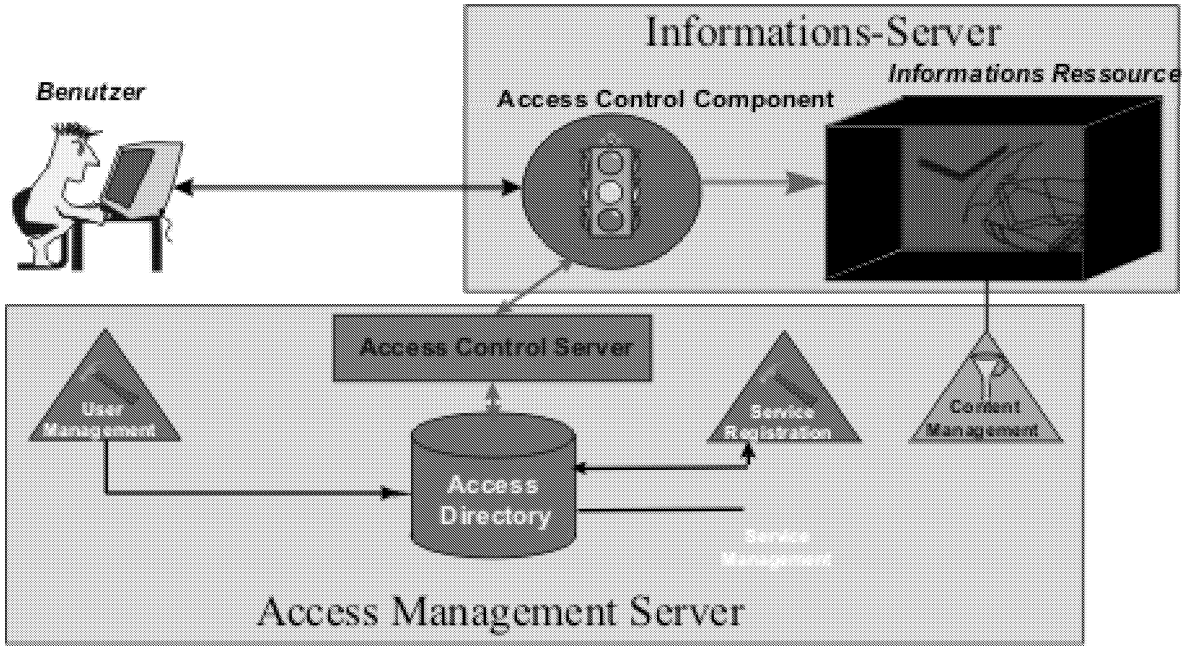

Abbildung 26: Funktionaler Überblick *HiQoS Access Management System*

2.3.6 Open Source

Ziel dieses Teilprojekts war es, einige der im Projekt entstandenen Softwarekomponenten (Media Server, Media Player) als Open Source zu veröffentlichen und so auch nach der Projektlaufzeit die Entwicklung der Komponenten weiter voranzutreiben.

„Die zugrundeliegende Idee für Open Source ist sehr einfach: Programmierer, die Zugriff auf den Quellcode einer Anwendung im Internet haben und diesen ansehen, verändern und weiter verteilen können, entwickeln diesen weiter. Sie verbessern, adaptieren, korrigieren und all das im Verhältnis zur konventionellen Software-Entwicklung mit einer erstaunlichen Geschwindigkeit." [30]

Dieser von der Webseite der *opensource.org*[1] übernommene Beschreibung stellt eine kurze Einführung in die Thematik anhand des für Open Source zugrundeliegenden Gedanken dar. Dass diese Idee durchaus realistisch ist und sich so erfolgreich Produkte im Rahmen von Open-Source-Projekten entwickeln lassen [31, 35], zeigen die folgenden Beispiele [32] :

- Linux (Betriebssystem)

[1] Da es sich bei „Open Source" um keinen eindeutigen und schutzfähigen Begriff handelt, wurde die Open Source Initiative (OSI) gegründet. Diese hat es sich zum Ziel gesetzt, eventuellen Missbrauch damit zu unterbinden. Das geschieht dadurch, dass Projekte, die unter einer OSI-konformen Open-Source-Lizenz verbreitet werden, mit einem OSI-Zertifikat versehen werden (Näheres dazu auch unter www.opensourrce.org).

- Mozilla-Browser (neben MS Internet Explorer der meistbenutzte Browser)
- Apache (läuft auf mehr als 50 % aller Web-Server)
- GNU Software (Entwicklungsumgebung)
- Perl (aktiver Bestandteil zahlreicher Webseiten)
- Sendmail (Mail-Tool)

Der Bekanntheitsgrad und die Verteilung dieser Softwarekomponenten (nicht zuletzt Linux) zeigen die Möglichkeiten, die hinter dieser Idee stecken. Das wirtschaftliche Potenzial von Open Source beweist sich auch am Interesse bekannter Firmen, wie zum Beispiel Sun, IBM, Netscape, 3Com, Ericsson, ..., die ebenfalls in Open-Source-Projekten engagiert sind.

In HiQoS wurden diesbezüglich die folgenden Teilergebnisse erzielt, die im Weiteren näher beschrieben sind:

- Eingehende Untersuchung des Open Source Gedanken und der damit verbundenen Eigenschaften.
- Evaluierung der verschiedenen Open Source Lizenzmodelle.
- Auswahl eines für HiQoS geeigneten Lizenzmodells.
- Erstellen eines Prototypen einer HiQoS Website.
- Diskussion der Struktur und Inhalte des Prototypen
- Überprüfung und gegebenenfalls Überarbeitung der HiQoS-Komponenten, so dass diese als Open Source verteilt werden können.

2.3.6.1 Lizenzmodell

Nach dem Studium verschiedener Lizenzmodelle [33,34] *(GPL - GNU General Public License, LGPL - GNU Library General Public License, MPL - Mozilla Public License, NPL Netscape Public License*) erscheint die MPL für HiQoS am besten geeignet sein. Sie sichert einerseits den Open-Source-Gedanken (OSI Zertifikat), erlaubt aber auf der anderen Seite den industriellen Projektpartnern weiterhin eigene Entwicklungen (ohne diese zu veröffentlichen) auf Basis der erstellten Software

Erarbeitet und eingesetzt wurde die MPL im Zusammenhang mit der Veröffentlichung des *Communicator 5* von Netscape. Abgeleitete Arbeiten müssen nicht veröffentlicht werden, wo hingegen Änderungen am Quell-Code frei verfügbar gemacht werden müssen. Dabei gilt auch, dass Erweiterungen (im Gegensatz zu Modifikationen) unter einer anderen Lizenz verbreitet werden können.

2.3.6.2 Bekanntmachung von HiQoS Open Source

Für das Weiterkommen eines Open-Source-Projektes ist es immens wichtig, viele Benutzer und potenzielle Entwickler zu finden. In HiQoS soll dies folgendermaßen erreicht werden:

- *Mail-Verteiler, Usenet-Foren*: Von Interesse sind Verteiler und Gruppen aus den Berei-

chen MPEG, Media-Streaming, Video-on-Demand

- *Webseiten*: Auch hier bieten sich Seiten aus den obigen Bereichen an. Zusätzlich könnten von den Seiten der Projektpartner Links auf die Open Source Webseite verweisen.
- *Open-Source-Webseiten*: Auf dem Web existieren verschiedene (auch internationale) Seiten, die einen Überblick über bestehende Open-Source-Projekte geben.
- *Online-Magazine, Fachpublikationen*: Neben bekannten Magazinen wie *heise.de* gibt es zahlreiche weitere mit einer immens großen Leserschar. Auch hier könnten die Projektpartner für entsprechende Beiträge sorgen.

2.3.6.3 Verteilung der Software-Komponenten

Das Internet die ideale Plattform für die Verteilung von Open-Source-Projekten. So wurde auch im Rahmen dieser Untersuchung ein Prototyp für eine Website für ein erfolgreiches Open-Source-Projekt HiQoS erarbeitet. Auf dieser Website werden die folgenden Bestandteile enthalten sein:

- *Einführung*: In diesem Bereich findet sich eine Einführung und ein Überblick über das Einsatzszenario der betreffenden Softwarekomponenten. Es wird möglichst umfassend auf die Technologie eingegangen. Die Einführung sollte so ausführlich und verständlich sein, dass potenzielle Anwender in der Lage sind, das Szenario zu verstehen.

- *Software-Download*: Dies ist eines der zentralen Elemente einer Open Source Website. Von hier können die beteiligten Software-Komponenten herunter geladen werden. Es müssen neben den reinen Software-Bestandteilen (Quellcodes, ausführbare Programme) auch Hinweise zur Installation gegeben werden. Benutzer sollten so in der Lage sein, die Programme ohne größere Probleme herunter zu laden, zu installieren und auch erfolgreich einsetzen zu können.

- *Dokumentation*: Hier müssen sich möglichst ausführliche Dokumentationen für das Projekt befinden. Das umfasst Einführungen, Installations-, Anwender- und Programmierdokumentation. Dadurch sollten Anwender in der Lage sein, die Programmkomponenten zu installieren und Programmierer dazu den Quellcode zu verstehen und zu modifizieren. Zur Unterstützung dieses Personenkreises ist eine ausführliche Inline-Dokumentation im Quellcode natürlich ebenfalls unerlässlich.

- *Lizenz*: Die Lizenz, unter der die Software verteilt wird, darf selbstverständlich auch nicht fehlen.

- *Support*: Dies ist ebenfalls ein zentrales Element einer Open Source Website. Bestandteile sind hier ein Forum bzw. eine Mailingliste und ein Ansprechpartner innerhalb des Projekts. Im Forum besteht die Möglichkeit für die Benutzer, sich untereinander über ihre Erfahrungen, Ideen und Entwicklung auszutauschen. So behalten auch die Projektpartner jederzeit Informationen über die aktuellen Wünsche und Entwicklungen der Anwender. Der Ansprechpartner dient dazu, zentral Informationen, Wünsche und Modifikationen zu sammeln.

- *Aktuelles*: In diesem Bereich finden sich ständig aktuelle Informationen über das Projekt (neue Versionen, ToDo-Listen, ...). Ein mögliche Erweiterung wären zusätzliche Informationen aus dem Themengebieten rund um HiQoS. So könnten auch weitere potenzielle Projektinteressierte zufällig auf die Open Source Website treffen.

- *Linkliste*: Hier finden sich Verweise auf Websites aus dem Themengebiet HiQoS.

- *Projektpartner*: Nicht zuletzt müssen auch die Projektpartner, die an der bestehenden Software mitgearbeitet haben, auf der Website ihren Platz finden.

- *Zentrale Leitung*: Obwohl es in einem Open Source Projekt keine klare Führungsinstanz gibt, sollte doch immer ein Gremium über das Projekt wachen. Dieses könnte die verteilte Entwicklung koordinieren, so dass nicht mehrere Anwender an einer Baustelle arbeiten. Mögliche Entscheidungen sind darüber hinaus, ob Versionsnummern aufdatiert werden (aufgrund von Modifikationen oder Erweiterungen von Anwendern) und welche Arbeiten zunächst angegangen werden um so das Projekt gezielt voranzubringen

Auf die gestalterische Ausarbeitung dieser Webseite soll hier nicht näher eingegangen werden. Die oben aufgeführten Bestandteile sind in einem Prototyp realisiert worden. Dieser Prototyp diente als Diskussionsgrundlage und kann als Basis für die HiQoS-Open-Source-Präsenz angesehen werden.

2.3.6.4 Fazit

Ein Open-Source-Projekt ist nicht mit der Veröffentlichung der Quell-Codes bewältigt. Die auszuführenden Tätigkeiten inklusive noch ungeklärter Probleme und die anschließende Betreuung der Benutzer enthalten einiges an Arbeit. Dass und wie erfolgreich auf der anderen Seite die Entwicklung als Open Source sein kann, zeigt sich nicht zuletzt an dem Betriebssystem Linux. Um HiQoS auf einen ähnlichen Weg zu bringen, wurden die Grundlagen von Open Source (insbesondere die vorhandenen Lizenzmodelle und deren Eignung) untersucht. Darüber hinaus wurde der Prototyp einer Website erstellt, dessen Struktur als Grundlage für eine Verteilung von HiQoS über das Internet dienen kann. Zusätzlicher Aufwand muss in jedem Fall noch in die Dokumentation investiert werden: zum einen in Benutzerdokumentation (Bedienung, Installation), zum anderen in Quelltextdokumentation, da von diesen die endgültige Akzeptanz maßgeblich abhängt. Wesentlich ist auch die Klärung der Lizenzfrage, die sich insbesondere bei großen Projektpartner als problematisch erweist. Sollte dies gelingen, die entsprechende Web-Infrastruktur geschaffen sein und das Projekt bekannt gemacht werden können, steht einer Weiterentwicklung der Softwarekomponenten aus HiQoS als Open Source nichts mehr im Wege.

2.3.7 SMIL

Ein geeigneter Demonstrator für die Einhaltung von "Quality-of-Service-(QoS)"-Garantien in einem Netzwerk von Multimedia-Servern, ist die Verknüpfung mehrerer breitbandiger kontinuierlicher Medien, wie sie für die ViLM-Anwendung (vgl. Abschnitt 2.4.1) benötigt wird.

Eine Möglichkeit solche Anwendungen zu implementieren, bietet die *vom World Wide Web Consortium (W3C)* 1998 [36] vorgeschlagene *Synchronized Multimedia Integration Langua-*

ge (SMIL). SMIL ist eine auf XML basierende Sprache, mit der räumliche, zeitliche und inhaltliche Beziehungen zwischen kontinuierlichen und statischen Medien für eine Anwendung beschrieben werden können.

Innerhalb von HiQoS wurde ein funktional eingeschränkter *SMIL-Client* entwickelt, bei dem besonderes Gewicht auf die gleichzeitige Wiedergabe mehrerer Videoströme und Plattformunabhängigkeit gelegt wurde.

Die vom SMIL-Client abgedeckten Funktionalitäten entsprechen der für die Wiedergabe der Multimediadokumente aus der ViLM-Anwendung (siehe Abschnitt 2.4.1) benötigten. Diese ergeben sich aus der Art, Verknüpfung und Anzahl der hierbei darzustellenden Daten, d.h. den darin enthaltenen beiden parallelen Videos, den Materialien, die entweder sequentiell oder exklusiv dargestellt werden, und dem Inhaltsverzeichnis, über das die Materialien referenziert werden.

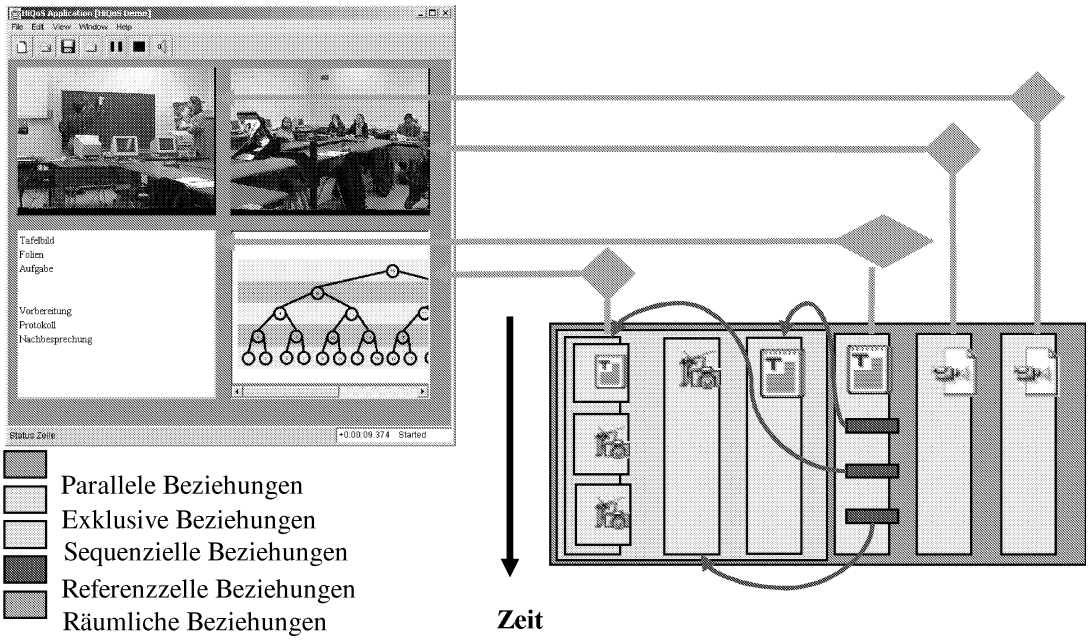

Abbildung 27: Beziehungen bei der SMIL-Applikationen aus ViLM

Dem entsprechend wurden von den SMIL-Elementen für die zeitlichen Beziehungen (siehe

Abbildung 27) SEQ, PAR und EXCL (das Element EXCL ist Bestandteil von SMIL 2.0, das z.Z. als Vorschlag vorliegt [37]) bearbeitet. Das Element SEQ verknüpft die wiederzugebenden Daten zeitlich sequentiell. Durch das Element PAR verknüpfte Daten werden zeitlich parallel wiedergeben und die durch das Element EXCL verknüpften Daten werden bedingungsabhängig exklusiv wiedergegeben.

SMIL–Elemente zur Beschreibung von Referenzbeziehungen wurden nicht implementiert. An dessen Stelle werden Referenzen aus angezeigten HTML–Texten ausgewertet, soweit sie sich auf Inhalte der SMIL–Applikation beziehen.

Die SMIL–Elemente für räumliche Beziehungen sind im vollen Umfang implementiert worden. Von den SMIL-Elementen, die unterschiedliche Datentypen in SMIL integrieren, wurden die Elemente *TEXT, IMAGE, AUDIO* und *VIDEO* realisiert.

Ergänzend zu SMIL wurden die Elemente *SOUND*, *VOLUME* und *SOUNDCHANNEL* implementiert, welche es dem Anwender ermöglichen, während der Wiedergabe die Audioquelle zu wechseln.

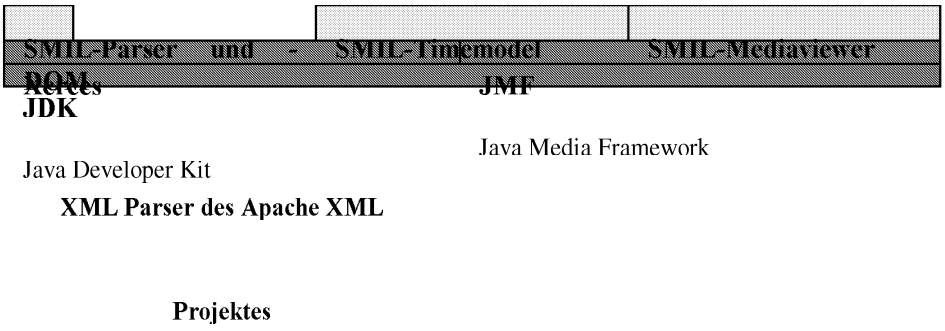

Abbildung 28: Beziehungen bei der SMIL-Applikationen aus ViLM

Um die Plattformunabhängigkeit des Clients weitgehendst zu gewährleisten, wurde dieser in Java implementiert. Als Basispackete wurden das *Java Developer Kit*, der *SAX (Simple API for XML*, d.h. kompatibler Part von Xerces zum Parsen der SMIL-Applicationen) und das *Java Media Framework* zur Wiedergabe der einzelnen Daten verwendet (siehe auch **Fehler! Verweisquelle konnte nicht gefunden werden.**).

2.3.8 Realzeit-Scheduling

Die Anwendung von Methoden aus der klassischen Realzeit-Theorie [38] haben in der Vergangenheit gezeigt, dass für zeitkritische Systeme nicht nur die Einhaltung von Deadlines garantiert gegeben werden kann, sondern dass diese Systeme auch besonders effizient (d. h. besonders gute HW-Ausnutzung) bzw. besonders preiswert sind. Diese Vorteile sind auch in HiQoS im Bereich der Übertragung von Audio/Video-Material in IP-Netzwerken ausgenutzt worden. Dabei wurde gezeigt, dass ein Konzept basierend auf Realzeit-Scheduling sich in Bezug auf *Admission Control*[2] effizienter verhält, als eine von RSVP vorgeschlagenen Vorgehensweise (*Token-Bucket*-Methode). Als Ergebnis muss für einen Strom i.a. weniger Bandweite im Netz reserviert werden als mit der normalen Methode. Dies gelingt dadurch, dass bekannte Varianzen zu verschiedenen Frame-Typen bei MPEG-2-Strömen mit variabler Bitrate ausgenutzt werden und bei der Versendung ggf. bewusst einzelne Pakete zurückgehalten (d.h. gepuffert) werden. Das betrachtete Szenario arbeitet mit dem in HiQoS benutzten RSVP-Protokoll zusammen.

---

[2] Admission Control bezeichnet den Vorgang, zu überprüfen, ob ein weiterer Medien-Strom zur Auslieferung gelangen darf, ohne dabei seine eigene Qualität oder die von anderen zu beeinträchtigen.

2.3.8.1 Einleitung

Das *Resource Reservation Protocol (RSVP)* ist dafür entworfen worden QoS-Anforderungen (d.h. eine bestimmte Bandweite) in einem IP-Netzwerk zu signalisieren (vgl. Abschnitt 2.3.1). Damit eine hohe Ressourcen-Auslastung gewährleistet sein kann, ist es sehr wichtig, dass die jeweilige Anwendung, die benötigte Bandweite möglichst präzise bestimmt. Im Falle von Überschätzungen kann die Netzwerkauslastung erheblich leiden. Des weiteren muss neben der Bandweite auch die akzeptierbare Übertragungsdauer (*End-to-End Delay)* berücksichtigt werden.

Im Rahmen der hier beschriebenen Arbeit ist dabei auf die effiziente Übertragung von MPEG-2-Strömen eingegangen worden. MPEG-2 ist ein weitläufig eingesetztes Video-Kodierungsverfahren, welches hohe Kompressionsraten erzielt, wenn dabei mit variablen Bitraten gearbeitet wird. Hierbei ist allerdings der resultierende Datenstrom großen Schwankungen ausgesetzt, so dass eine Bestimmung der benötigten Bandweite sehr schwierig ist.

Die in diesem Kontext eingesetzte neuartige Analyse-Methode zeichnet sich insbesondere durch eine hohe Präzision und durch große Flexibilität aus. Dabei kann diese Methode sowohl mehrere Netzwerk-Verbindungen als auch mehrere Ströme gleichzeitig analysieren und kann sogar einfach an verschiedene Scheduling-Mechanismen (wie *First-In First-Out* und *Fixed-Priority Scheduling*) angepasst werden.

Die erhaltenen Evaluierungsergebnisse sind dabei mit der *Guaranteed Service Analysis* der RSVP-Vorgehensweise verglichen worden: Es konnten Bandweiten-Ersparnisse bis zu 22% nachgewiesen werden.

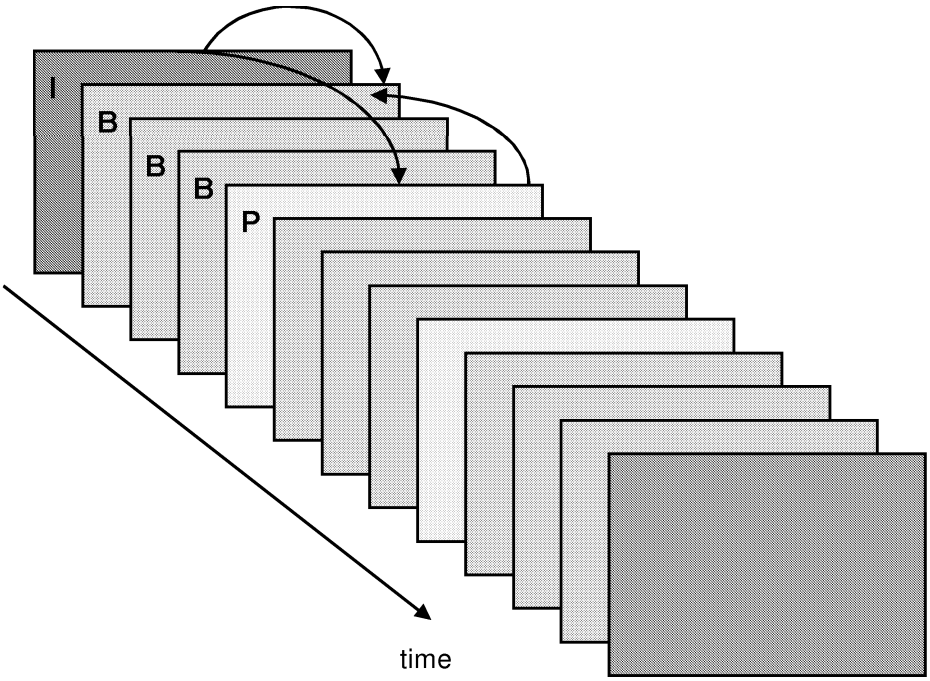

Abbildung 29: Eine MPEG Group-of-Picture

Es ist des weiteren möglich, das hier vorgestellte Verfahren in das RSVP-Szenario zu integrieren, ohne dabei das Protokoll zu verändern.

2.3.8.2 MPEG-2

MPEG kennt drei verschieden Frame-Typen (siehe Abbildung 29):

1. *I-frames (intracoded-frames)* repräsentieren JPEG-Bilder und enthalten komplette Information zu Vollbildern und sind deshalb die datenintensivsten Bilder

2. *P-frames (inter-frame predicted pictures)* referieren auf vorhergehende I- oder P-Frames. Ihre Größe liegt zwischen denen der I-Frames und den nachfolgend beschriebenen B-Frames.

3. *B-frames (Bi-directional predicted/interpolated pictures)* benutzen sowohl Vorwärts- als auch Rückwärtsreferenzen und sind überlicherweise am kleinsten.

Diese Bilder sind in sogenannten *Group Of pictures (GOPs)* organisiert. Diese enthalten immer mindestens ein I-Frame und optionale P- und B-Frames. Reguläre GOPs erscheinen in festen Zeitabständen.

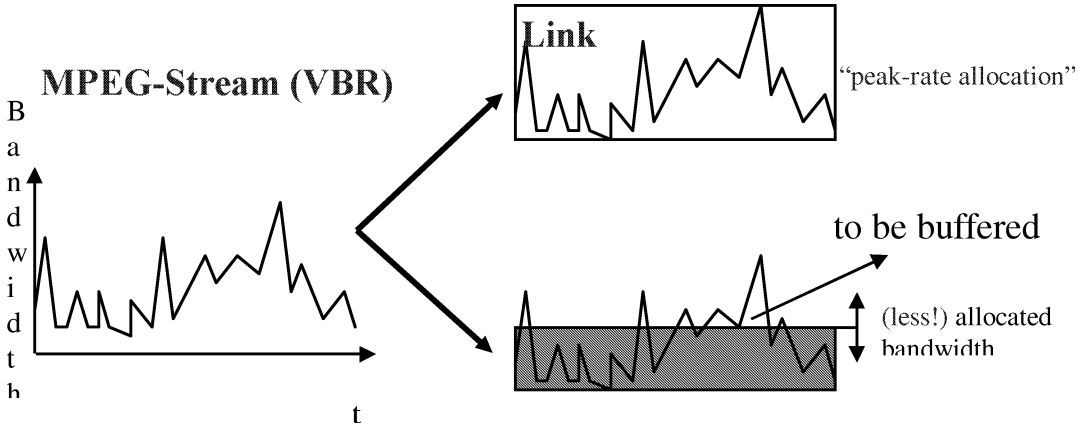

Abbildung 30: Variable Bitrates

Die besondere Herausforderung bei der Übertragung ist dann die stark schwankende Bitrate auf eine Bandweiten-Reservierung abzubilden. Während I-Frames sehr groß sind, braucht eine Reihen von B-Frames nur sehr wenig Bandweite. Ein einfacher Ansatz wäre es, einfach das größte I-Frame herauszufinden (*Peak-Rate Allocation*) und als Basis für die Allokierung zu nehmen. Allerdings würde dabei viel ungenutzte Bandweite verschwendet werden. Für einen typischen Strom kann dann die Auslastung kleiner als 20% sein (siehe auch). Ziel ist es daher die zu allokierende Bandweite unterhalb des Peaks anzusetzen, sofern die zeitlichen Restriktionen dies zulassen. Der hier vorgestellt Ansatz führt dabei zu besseren Ergebnissen als die in RSVP vorgeschlagenen Token-Bucket-Methode [40], weil eine genauere Kenntnis der zu versendenden Daten vorliegt.

2.3.8.3 RSVP Response-Time Analysis

In diesem Abschnitt wird nur kurz auf die prinzipielle Arbeitsweise des neuartigen Ansatzes eingegangen. Die komplette Darstellung ist [39] zu entnehmen.

Realzeit-Systeme

Realzeit-Systeme [38] sind eine besondere Klasse von Computern. Ihre besondere Eigenschaft besteht daraus, dass nicht nur die Funktionalität eines Systems sondern auch zeitlichen Eigenschaften berücksichtigt werden. Dabei muss überlicherweise eine bestimmte Funktion vor einer bestimmten *Deadline* terminieren, andernfalls könnte ein Systemfehler auftreten. Daher ist es besonders wichtig, eine komplette zeitlicher Analyse aller beteiligten Komponenten zu machen.

Traditionell gesehen war die Forschung im Bereich Realzeit eher auf *harte* Systeme (z.B. ABS beim Auto) fokussiert. Jedoch hat in den letzten Jahren auch die Analyse von *weichen* Systemen wie die hier betrachteten Video-Server begonnen. Dabei ist es wichtig, darauf hinzuweisen, dass bei weichen Systemen im Gegensatz zu harten keine katastrophalen Dinge im Falle einer Deadline-Verletzung auftreten, aber jedoch die Qualität stark leiden kann.

Response-Time Analysis

Die erreichbare Übertragungszeit vom Sender zum Empfänger (und damit das Einhalten von Deadlines) ist direkt abhängig von der verfügbaren Bandweite. Wenn diese zu jedem Zeitpunkt garantiert werden kann, kann auch eine obere Schranke für die Übertragungszeit bestimmt werden. Somit lässt sich dann umgekehrt die *wirklich benötigte* Bandweite aus der tolerierbaren Übertragungszeit herleiten. Je genauer diese Analyse dabei vorgeht, um so effizienter ist hinterher die Netzwerkauslastung (um so komplexer und zeitintensiver ist aber auch die Analyse).

In Rahmen dieser Tätigkeit ist dabei die sogenannte *Response-Time Analysis* [38] zum Einsatz gekommen, welche ihre Wurzeln im der Betrachtung von CPU-Scheduling hat. Dieses Verfahren rechnet die *Response Time* einer *Task* aus, welches die Zeitspanne zwischen dem Moment des möglichen Starts und dem Abschluss der Task ist. Berechnen lässt sich diese aus der Summe der *Idle*-Zeit der Task (weil diese auf höher-priorisierte Tasks warten muss) und ihrer eigentlichen Berechnungszeit. Somit ist dieser Wert vom benutzten Scheduling-Verfahren abhängig. Die Response-Time Analysis in Mulitasking-Systemen ist schon sehr lange Gegenstand von Forschungsarbeiten. Dabei werden immer periodische Tasks betrachtet, die mit einer Deadline versehen sind, die am Ende der Analyse mit dem berechneten Wert verglichen wird.

Eine derartige Technik (zugeschnitten auf den Anwendungskontext RSVP und IP-Netzwerke) ist auch immer Rahmen dieser Tätigkeit zum Einsatz gekommen.

2.3.8.4 Resultate

Das neu-entworfene Verfahren ist durch zahlreiche simulative Experimente evaluiert worden. Dabei hat sich gezeigt, dass sich Ressourcen in großem Umfang einsparen lassen (bis zu 22%), wie in Abbildung 31 zu sehen ist.

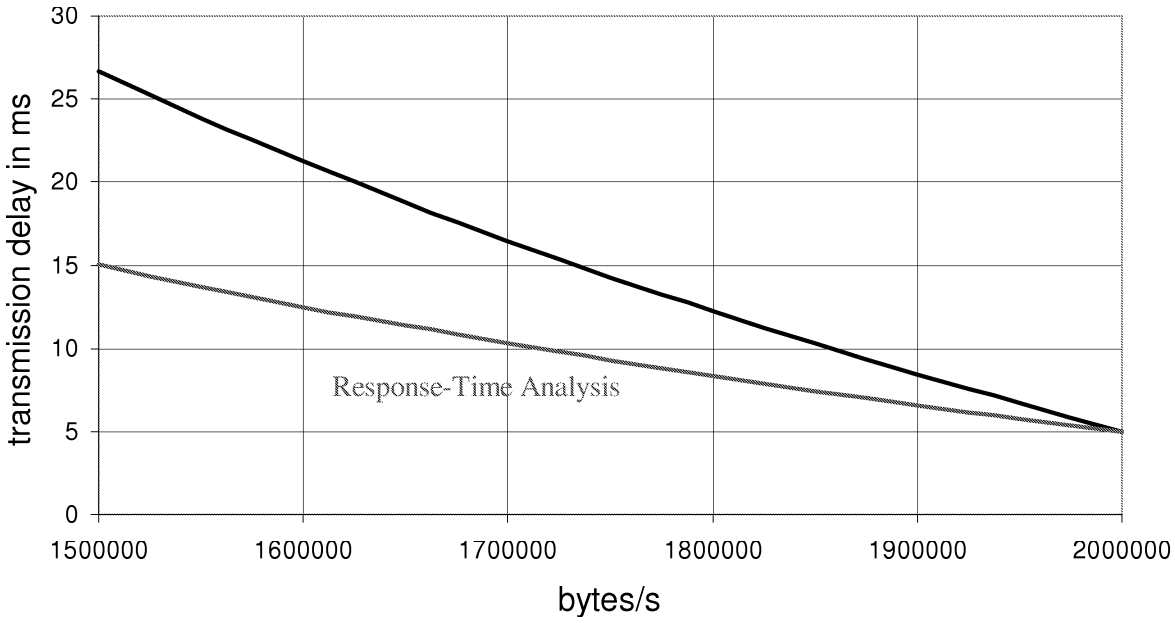

Abbildung 31: Resultate des Realzeit-Schedulings

2.3.8.5 Einbettung in das RSVP-Szenario

Die Einbettung des neuen Verfahrens in das vorgegeben RSVP-Szenario3 ist in Abbildung 32 zu sehen. Dabei ist nur das Problem der Auslieferung eines einzelnen Stroms betrachtet worden.

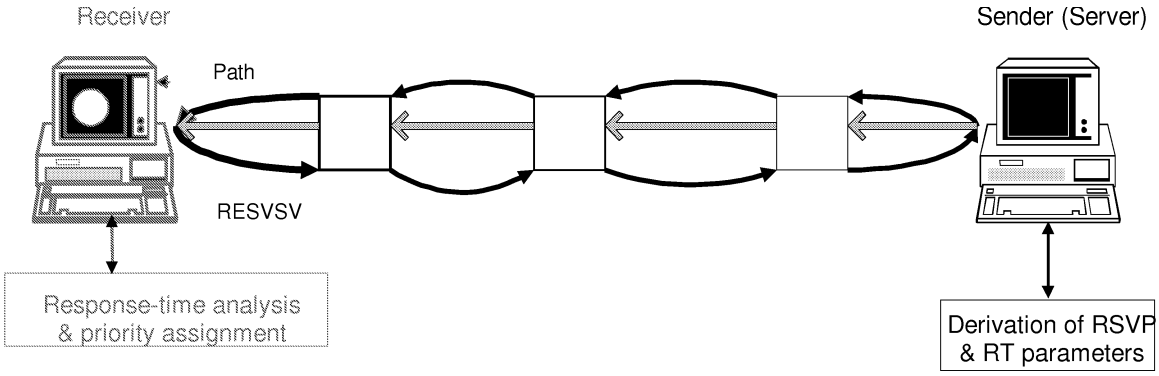

Abbildung 32: RSVP-Szenario mit integrierter Response-Time Analysis

Hierbei werden im wesentlichen die vier folgenden Schritte durchlaufen:

1. Der Sender analysiert das zu übermittelnde Video-Material, so dass die benötigten Para-

[3] Das Szenario ist als Patent eingereicht worden.

meter für die Response-Time Analysis vorhanden sind.

2. Mithilfe der *Path MSG* (einer Message, die im RSVP-Szenario immer zu Beginn verschickt wird, um den Auslieferungspfad festzulegen), werden die ermittelten Parameter bis zum Empfänger geschickt. Dabei werden außerdem die derzeit möglichen Bandweiten und Übertragungsraten der einzelnen Links notiert.

3. Auf der Empfänger-Seite wird basierend auf den übermittelten Daten die Response-Time Analysis eingesetzt, um die benötigte Bandweite zu berechnen. Dieser Wert ist dann überlicherweise geringer als der von der normalen (*Token Bucket*) Methode.

4. Der ermittelte Wert wird mit der an dieser Stelle üblichen *RESV MSG* zurück den Pfad entlang geschickt, um die Reservierung an den einzelnen Knoten umzusetzen

2.3.8.6 Fazit

Es wurde gezeigt, dass ein Konzept basierend auf Realzeit-Scheduling sich in Bezug auf *Admission Control* effizienter verhält als die von RSVP vorgeschlagene Vorgehensweise. Das heißt, es muss für einen Strom i.a. weniger Bandweite im Netz reserviert werden als mit der normalen Methode. Dies gelingt dadurch, dass bekannte Varianzen zu verschiedenen Frametypen bei MPEG-2-Strömen mit variabler Bitrate ausgenutzt werden.

Allerdings war eine praktische Einbindung in die HiQoS-Komponenten aufgrund von fehlender MPEG2-Unterstützung noch nicht möglich. Die Wirksamkeit des neuartigen Ansatzes konnte jedoch anhand von Simulationen gezeigt werden.

2.4 Anwendungen in HiQoS

Im Rahmen von HiQoS wurden eine Reihe von Prototypen multimedialer Dienste entwickelt und in praktischen Anwendungen getestet. Dabei wurden die im Projekt selbst entwickelten Technologien eingesetzt.

2.4.1 ViLM

In Zusammenarbeit mit dem Paderborner Lehrstuhl für Didaktik der Informatik, Prof. Magenheim wird auf Basis von HiQoS-Technologie eine Lehrerausbildung angeboten. Die Inhalte der Lehrerausbildung sind Teil des dortigen Projektes Visualisierung von Lehr- Lernprozessen in verteilten Multimediasystemen *ViLM*, dessen technische Umsetzung auf den Methoden von HiQoS beruht.

Das Anwendungsszenario von ViLM begleitet die Lehramts-Studierenden in einem Seminar, in dem Sie Lehrproben halten müssen. Diese Lehrproben werden durch zwei Videoeinstellungen dokumentiert: einer *Lehreransicht* und einer *Schüleransicht* (siehe Abbildung 33).

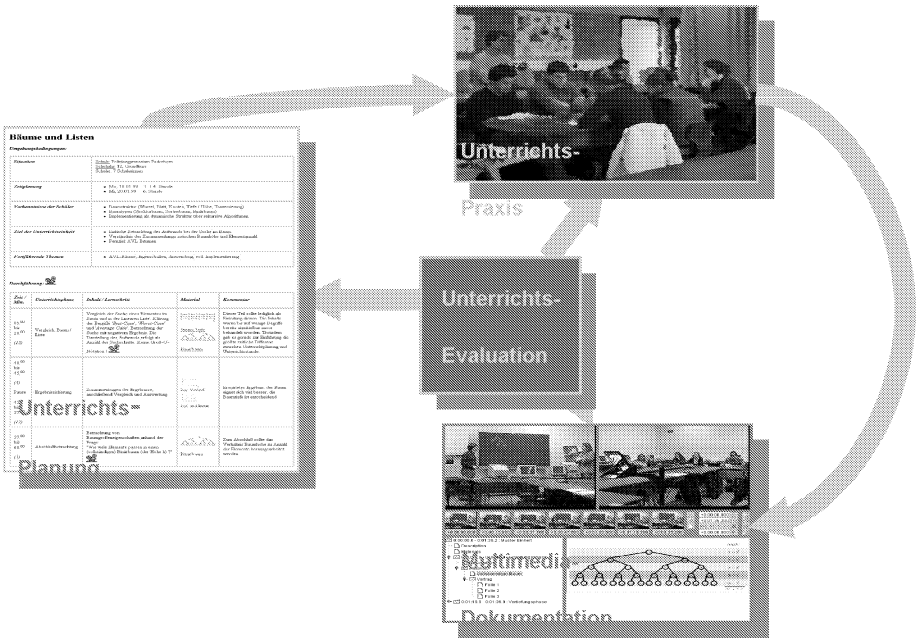

Abbildung 33: Workflow in der ViLM-Anwendung

Aus den Videos und den während der Lehrproben verwendeten Lehrmaterialen sollen die Studierenden anhand des Unterrichtsverlaufsplans eine Multimedia-Dokumentation der Lehrprobe erstellen (*Authoring*). Die Ziele dieses Vorgehens, sind Selbstevaluation durch die nachträgliche visuelle Konfrontation mit der Interaktion zwischen Lehrenden und Schülern und das Erlernen von Fertigkeiten im Umgang mit Multimedia-Dokumenten.

Neben technischer Beratung und Hilfestellung wurde folgendes technische Szenario zur Entwicklung und Wiedergabe von Multimedia-Dokumenten entworfen und zu großen teilen fertiggestellt. Die Systemarchitektur ist in Abbildung 34 dargestellt.

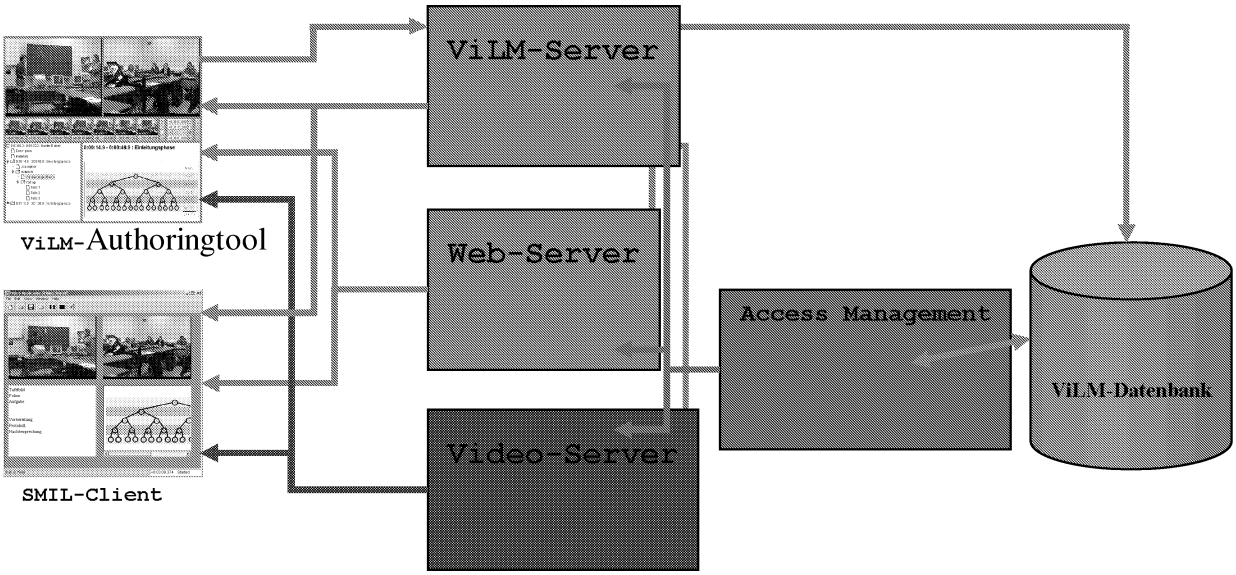

Abbildung 34: Architektur der ViLM-Anwendung

Die Komponenten *ViLM-Authoringtool*, *ViLM-Server* und *ViLM-Datenbank* wurden speziell für den für das ViLM-Szenario entwickelt. Der *SMIL Client*, der *Video Server* und das *Access Management* sind HiQoS-Bestandteile, die nicht speziell für das ViLM-Szenario entstanden sind. Die Komponente *Web-Server* kann durch jeden Web-Server, der sich an das *Access Management* anpassen lässt, zum Einsatz gebracht werden.

Das ViLM-Authoringtool ermöglicht die zeitliche Synchronisation der Lehrer- und Schüleranschicht, die Zerlegung der Videos in markante Unterrichtsphasen und die Verknüpfung der Lehrmaterialen mit den Unterrichtsphasen. Der ViLM-Server ermöglicht das Ablegen der ViLM-Struktur in der Datenbank, das Verteilen der verwendeten Medien auf den entsprechenden Servern und den Zugriff auf die ViLM-Struktur als ViLM-, HTML- oder SMIL-Applikation. In der ViLM-Datenbank wird die ViLM-Struktur abgelegt, um weitere Auswertungen auch über mehrere ViLM-Applikationen machen zu können.

Die entwickelten Tools werden von den Didaktik-Bereichen der Informatik in den Universitäten Paderborn und Dortmund in dem *Projekt Multimediale Evaluation in der Informatiklehrerausbildung (MUE)* eingesetzt. Die dabei bisher gemachten Erfahrung waren überwiegend positiv.

2.4.2 Pixelvision

Computer-Based Training (CBT) und Business TV (BTV) (oder auch Distance Learning) sind in vielen Firmen wesentliche Anwendungsszenarien für Multimedia-Dienste mit QoS-Anforderungen. Sie dienen der Mitarbeiterschulung, der Einführung in neue Produkte und der allgemeinen Weiterbildung. Jedoch ist die fehlende Unterstützung des Transports kontinuierlicher Medien ein wesentliches Hindernis für den Einsatz von CBT-Systemen in Intranets. Daher ist die Integration von kontinuierlichen Medien in CBT-Anwendungen heute nur selten anzutreffen.

Beim Pixelpark-Trial ging es daher um die Integration der HiQoS-Technologie in den Ablauf der *Pixelvision*, einem System für Schulungszwecke und zur Verbreitung technischer und organisatorischer Information. Dabei stand die Nutzbarmachung von Inhalten für die einzelnen Standorte der Pixelpark AG im Vordergrund.

2.4.2.1 Integration HiQoS – Pixelvision

Mittels der HiQoS-Technologie sollen digitale Informationen (Videomitschnitte, Powerpoint-Präsentationen und Textbeiträge) einer breiten Mitarbeiterschaft zur Verfügung gestellt werden. Die HiQoS-Technologie soll es dabei den Mitarbeitern im Hauptsitz Berlin sowie in ausgesuchten Filialen ermöglichen, die Inhalte zu einem qualifizierten Zeitpunkt abzurufen. Dazu wurden in den Standorten entsprechende Videoserver und Clients installiert (siehe Abbildung 35). In der ersten Stufe wurde das System (Videoserver und Client) im Standort Berlin erfolgreich installiert. In der weiteren Ausbauphase kamen die Standorte Hamburg, Dortmund und Stuttgart hinzu. Bedingung für den erfolgreichen Einsatz der HiQoS-Technologie ist eine homogene Netzinfrastruktur im Router-Bereich. Wichtig ist die Unterstützung von RTSP/RSVP durch die aktiven Komponenten (z.B. *Cisco* Router).

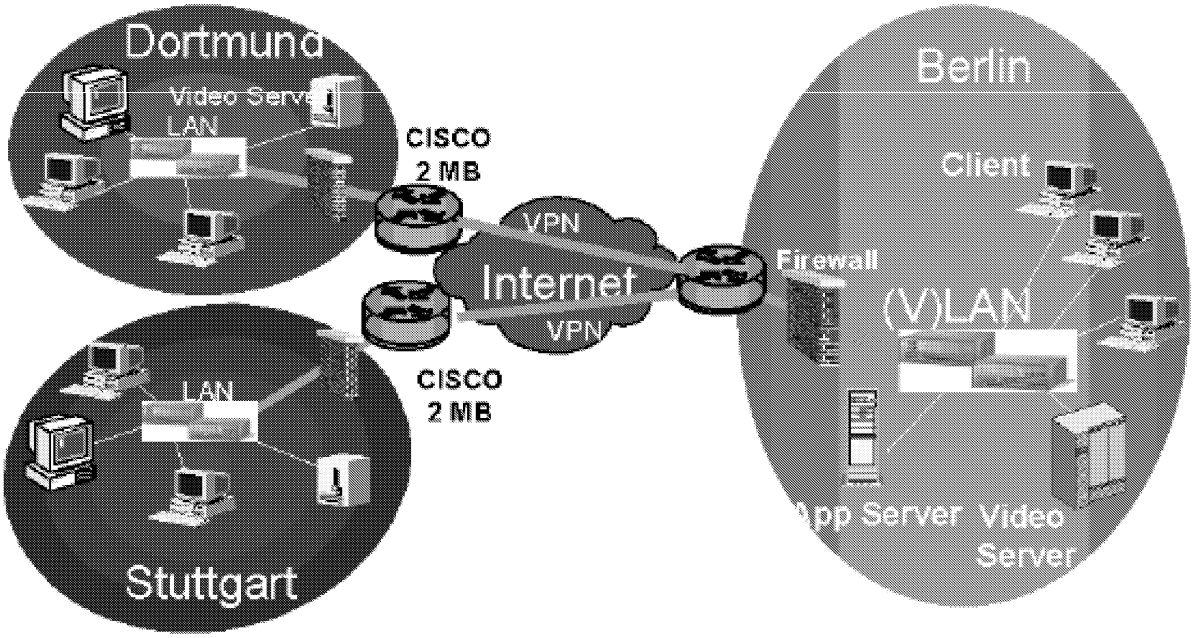

Abbildung 35: Übersicht über die *Pixelvision*

In Berlin sowie in den Standorten stehen jeweils untereinander vernetzte Videoserver. Die Videoserver bedienen ausgewählte Clients und versorgen diese mit Inhalten. Die Mitarbeiter können auf den Servern hinterlegte Inhalte abrufen. Die Clients sind an zentralen Stellen für jedermann zugänglich aufgestellt.

Die wesentlichen Komponenten des Pixelvision Systems in der Zentrale der Pixelpark AG in Berlin wurden in der ersten Projektphase integriert. Dies betrifft den zentralen Mediaserver,

die in dem Gebäude verteilt installierten Clientsysteme, sowie alle notwendigen Softwarekomponenten. In der zweiten Projektphase wurde ein neues QoS-Konzept erarbeitet, was durch die fehlende RSVP-Unterstützung auf den eingesetzten Cabletron-Routern notwendig wurde. Dabei wurden zur Simulation von QoS-Strecken die entsprechenden aktiven Komponenten (Netzwerkhardware wie Router oder Switch) zwischengeschaltet.

Zum Einpflegen der multimedialen Inhalte kommt ein Autorensystem zum Einsatz, welches ein transparentes Management der eingepflegten Inhalte ermöglicht; dabei können die Zugriffsmechanismen differenziert gesteuert werden. Dieses Redaktionssystem basiert auf einem grafischen User-Interface (GUI). Die Videoinhalte werden in einem ersten Schritt digitalisiert und anschließend redaktionell bearbeitet. Das bei den Präsentationen eingesetzte Material in Form von Powerpoint-Slides wird ebenfalls digitalisiert und zeitsynchron mit dem Videomaterial verknüpft. Die Integration der Inhalte erfolgte zuerst manuell und später mit dem innerhalb des Projekts entwickelten *Content Management Systems* (vgl. Abschnitt 2.3.2).

2.4.2.2 Evaluierung und Fazit

Die Hauptaufgabe innerhalb der zweiten Projektphase bestand in der Überführung des lokalen Systems in ein WAN basiertes, verteiltes System.

Das verteilte Pixelvision System wurde in den drei Pixelpark-Filialen vollständig aufgebaut und in einer längeren Betriebsphase innerhalb der Pixelpark AG evaluiert. Zur Evaluierung zählen verschiedene Bereiche sowohl mit als auch ohne QoS-Funktionalität.

Bei den durchgeführten Test zeigte sich, dass die Möglichkeiten von durchgehenden QoS-Verbindungen stark von den dazwischenliegenden lokalen aktiven Elementen abhängig sind. Sobald nichtgerichtete Internetverbindungen nicht QoS-Fähige Komponenten der entsprechenden ISPs nutzen, bricht auf Grund fehlender Protokollunterstützung die QoS-Strecke zusammen.

Es hatte sich gezeigt, dass bei weitem noch nicht alle Provider durchgängig QoS Funktionalität anbieten. Hinzu kommen noch Unterschiede in den sich etablierenden Standards (Diff, RTSP/RSVP etc.). Es konnte aber dennoch gezeigt werden, dass die HiQoS-Technologie im Wesentlichen die Anforderungen der Pixelvision erfüllen konnte.

2.4.3 CarTV

Bei APE wurde in einer späteren Phase das HiQoS-Szenario in einem Trial des erfolgreichen BusinessTV/Fahrzeugbörsen-Systems *CarTV* eingesetzt. Beim CarTV-System werden u. a. in Autohausfilialen und Versicherungsniederlassungen Werbe- und Schulungsfilme sowie Produktinformationen verteilt und den Mitarbeitern und Kunden zur Verfügung gestellt.

CarTV ist eine multimediale, satellitengestützte Plattform, um Fahrzeuge und Fahrzeugteile zu kaufen und zu verkaufen. Es verbindet modernste Computertechnologie mit Satelliten- und ISDN-Kommunikation.

2.4.3.1 Integration HiQoS – Car TV

In diesem komplexen System sind u.a. Autohäuser ein wichtiger Teilnehmer. In den Show Rooms der Autohäuser stehen leistungsfähige Computer mit Satelliten-Empfangsanlagen und ISDN-Leitungen für den Rückkanal. Die Fahrzeugdaten werden den KFZ-Händlern teilweise multimedial präsentiert, in erster Linie handelt es sich um Fahrzeugfotos und –videos. Die Fahrzeugvideos liegen im Echtbetrieb auf einer lokalen Festplatte des Empfangsrechners und werden von dort abgespielt. Für das HiQoS-Projekt wurde diese Schnittstelle so erweitert, dass die Videos nicht mehr lokal, sondern vom HiQoS Video Server über eine Netzwerkverbindung abgespielt werden können.

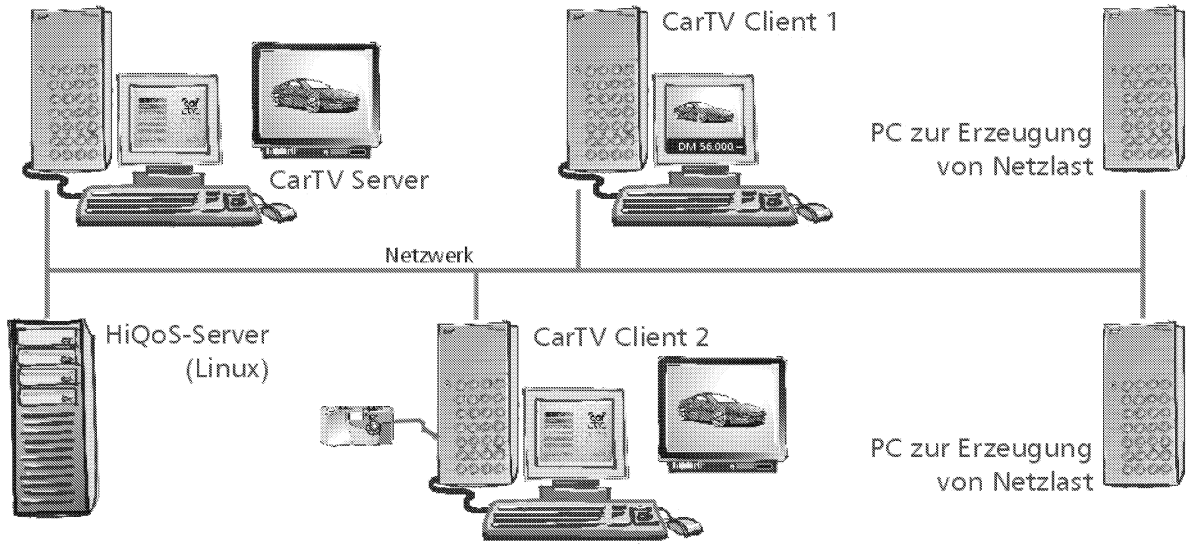

Abbildung 36: Übersicht über CarTV in HiQoS

Eine Übersicht über die in HiQoS integrierte Version von CarTV ist in Abbildung 36 zu sehen. Die Integration umfasste drei Schritte:

1. *Netzwerk-Integration*: Zuerst wurde der HiQoS-Server, der auf dem Betriebssystem Linux läuft, in das firmen-interne APE-Netzwerk integriert (Windows-2000/NT-Netz), welches dabei die Rolle des Autohaus-Netzwerkes spielt. Dies ist eine reale Installation, wie sie in der Praxis vorkommen würde, da die CarTV-Server teilweise ebenfalls in das Autohaus-Netzwerk integriert sind. Außerdem erzeugt der täglicher Verkehr auf dem Netzwerk automatisch Lasten, mit der die HiQoS-Technik dann umzugehen hatte.

2. *Übertragen und Anmelden der Videos auf dem HiQoS-Server:* Dieser Schritt ist aus Aufwands- und Kostengründen nicht automatisiert worden. Der Aufwand für eine automatische Übertragung der per Satellit empfangenen Daten auf den HiQoS-Server und deren Anmeldung im System wäre zu zeit- und kostenaufwendig. Auch das Anmelden der Videos geschieht manuell.

3. *Integration der HiQoS-Technik in die CarTV-Software*: Der größte Aufwand bestand aus der Integration der HiQos-Technik in die CarTV-Händlersoftware. Hierfür wurde der HiQoS-Client (ActiveX-Steuerelement) so in die Händlersoftware integriert, dass diese die

Videofilme nicht mehr von der lokalen Festplatte, sondern über das Netzwerk vom HiQoS-Server abruft.

Dabei war das Ziel, für das HiQoS-Projekt keine "Spezialversion" der Händlersoftware zu schreiben, sondern "nur" den Videoplayer – bei sonst gleichbleibendem Funktionsumfang der Software – durch HiQoS-Technik zu ersetzen. Dieser Umstand ist auch für den Praxisfall wichtig, denn die HiQoS-Software sollte sich leicht in realen Szenarien einsetzen lassen, und nicht nur in speziell für Evaluationszwecke erstellten Testumgebungen.

Nähere Informationen zum CarTV-System in HiQoS können [28] entnommen werden.

2.4.3.2 Fazit

Als Fazit bleibt zu erwähnen, dass sich die HiQoS-Technologie relativ leicht in CarTV integrieren ließ. Die HiQoS-Software lief sehr stabil und führte nie zu größeren Problemen. Die Evaluierungsergebnisse [29] zeigten deutlich, dass diese Technologie eine Alternative zum Video-Play des "normalen" CarTV-System darstellt.

2.4.4 Rendering in der Architektur

In den meisten modernen Architekturbüros zählt der Einsatz von CAD zum täglichen Alltag. Als einer der Pioniere hat die IEZ AG seit 1981 hierfür die Software *speedikon®* entwickelt.

Hierbei handelt es sich um ein modular aufgebautes, objektorientiertes 3D-CAD-System, das mit weiteren Softwareprodukten aus dem Mutterkonzern mb Software AG in allen Bereichen des Bauwesens einsetzbar ist : Architektur, Einrichtungsplanung, Ingenieurbau, Haustechnik, Facility Management.

Dabei erlaubt das Programm aufgrund seiner Datenstruktur und Schnittstellen (bauspezifisches 3D-Gebäudemodell) die durchgängige Bearbeitung der einzelnen Aufgabengebiete.

Das Programm ist lauffähig auf unterschiedlichen Hardware- und Softwareplattformen: auf Workstations mit Unix/Motif sowie auf PCs unter *AutoCAD MicroStation* und der eigenen Grafikentwicklung *ATLANTIS*.

Der für HiQoS interessante Teil von *speedikon®* ist der Bereich Architektur, hierdurch wird der Architekt von Vorentwurf über Baueingabe und Werkplanung bis hin zur Massenermittlung und Kostenberechnung unterstützt. Das Programm verwendet die im Bauwesen üblichen Terminologien und orientiert sich an Bauwerksteilen (Wände, Stützen, Unterzüge, usw.), denen eine „Intelligenz“ mitgegeben ist.

Die Eingabe erfolgt grundrissbezogen, wobei die dritte Dimension automatisch mitgeführt wird. Als Planungsergebnisse liefert *speedikon®* Architektur automatisch die Materialmengen, Wohn-/Nutzflächen sowie alle Arten von Grundrissplänen und darauf basierende Schnitte, Perspektiven und Ansichten in beliebigen Maßstäben.

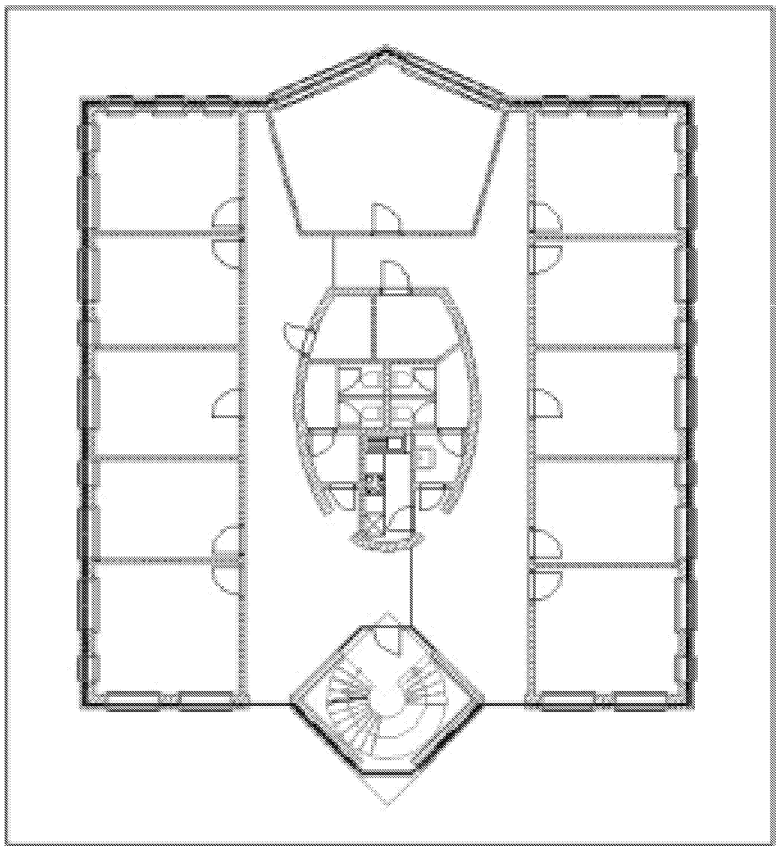

Abbildung 37: Grundriss mit *speedikon®*, ©IEZ AG Bensheim

Abbildung 37 zeigt einen Grundriss, der mit *speedikon®* eingegeben wurde. Aufgrund eines solchen Grundrisses können dann automatisch Perspektiven und Schnitte berechnet werden, wie es in Abbildung 38 zu sehen ist. Abbildung 39 zeigt *speedikon® -M* im Einsatz.

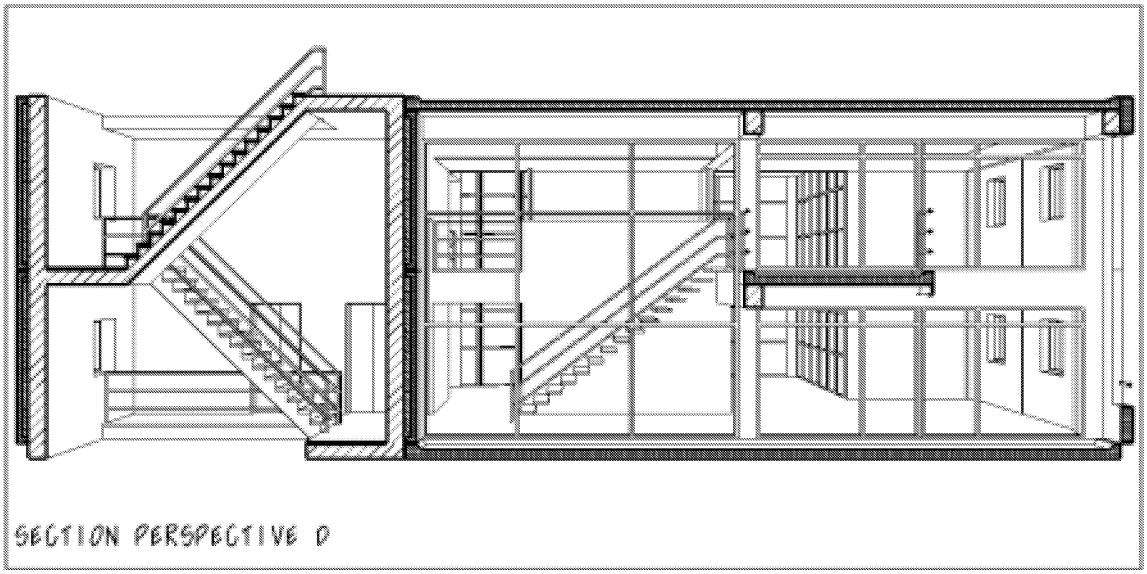

Abbildung 38: Seitenansicht eines Gebäudes, ©IEZ AG Bensheim

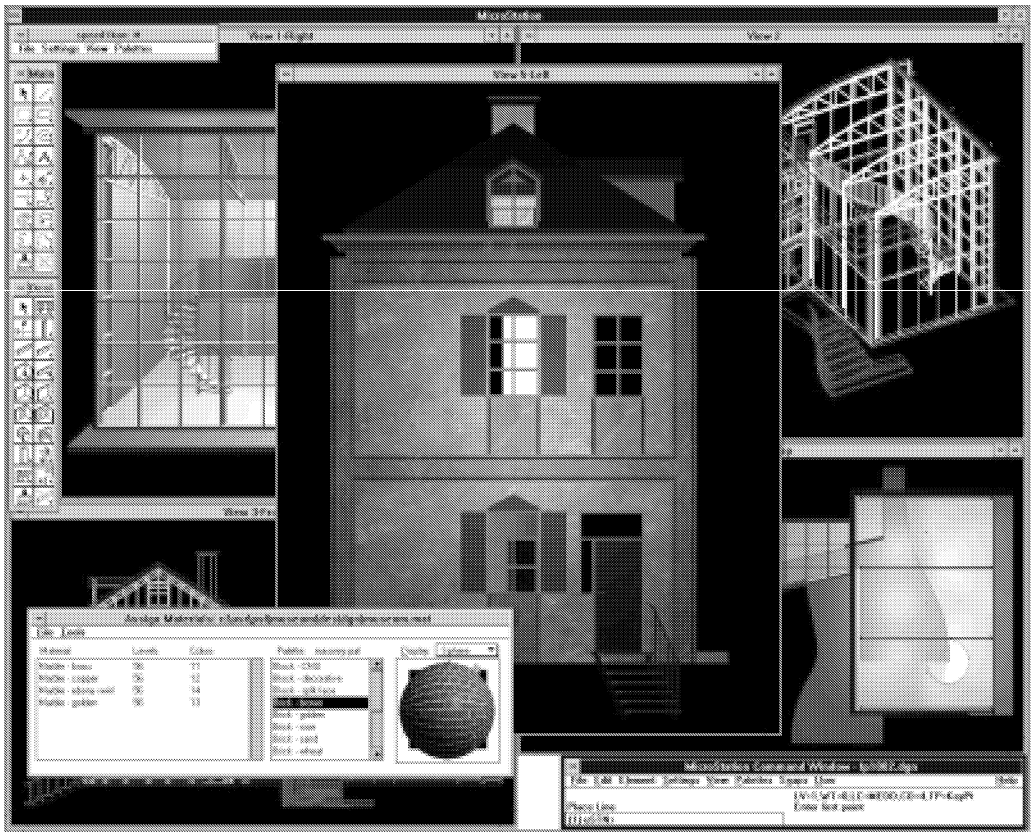

Abbildung 39: speedikon® -M, ©IEZ AG Bensheim

In den letzten Jahren ist der Anspruch von Investoren an die Darstellung ihrer geplanten Gebäude gestiegen, so dass immer mehr wert auf 3D Visualisierung gelegt wird. Es genügt nicht das ein Bild nur gut aussieht, es wird ein photorealistisches Bild erwartet.

Um ein konkretes Beispiel zu zeigen, das mit *speedikon®-M* konstruiert und visualisiert wurde, sieht man in Abbildung den Flughafen in Dubai, welcher sich derzeit im Bau befindet.

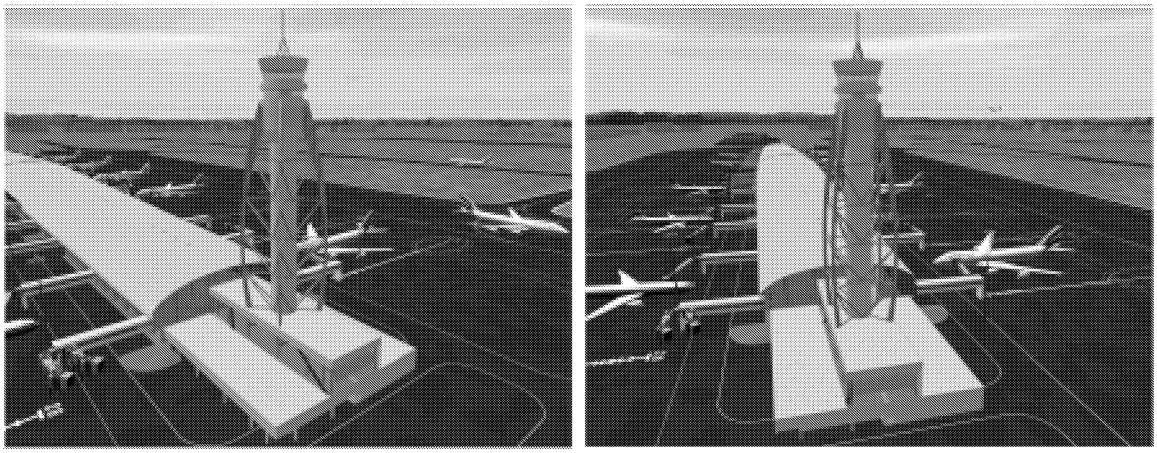

Abbildung 40: Flughafen Dubai mit *speedikon®-M* außen, ©IEZ AG Bensheim

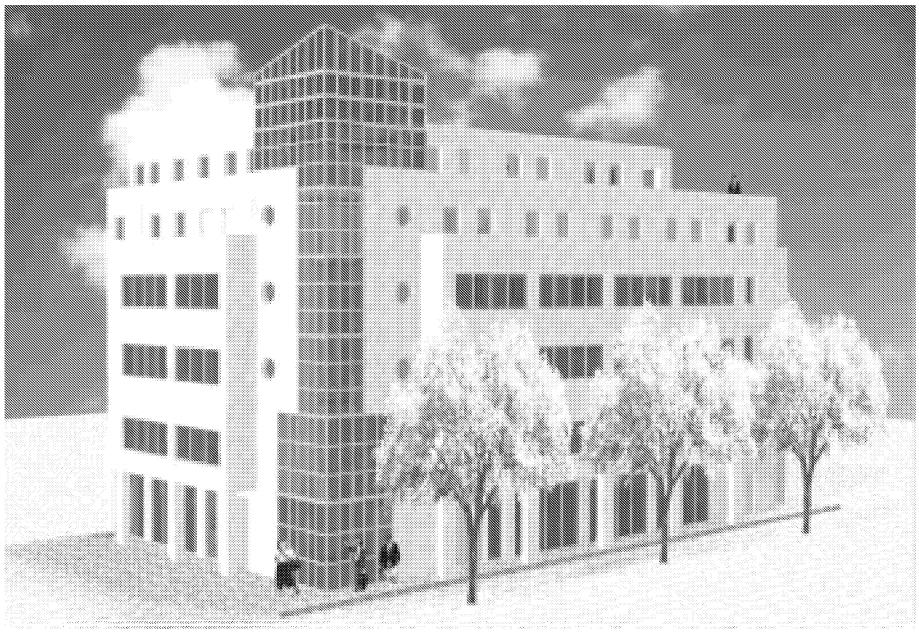

Abbildung 41: Bürogebäude, ©Institut für EDV Anwendung Worms

Wie Abbildung 40 und Abbildung 41 zeigen, kann der Architekt dem Investor schon heute mit *speedikon*® eindrucksvoll demonstrieren, wie das Bauwerk später aussehen wird.

Die Visualisierung ist längst zu einem wichtigen Teil von *speedikon*® geworden, und ständig ist die IEZ AG dabei, diesen Teil weiter zu verbessern und zu verfeinern. Jedoch stieß man in der Vergangenheit oft an die Grenzen der zur Verfügung stehenden Hardware. Im Forschungsprojekt HiQoS steht ein parallel Rendering-Service zur Verfügung. Mit dessen Rechenleistung können auch komplexe Szenen schnell photorealistisch dargestellt werden.

2.4.4.1 Workflow

2.4.4.1.1 Modellhafter Workflow

Der typische Arbeitslauf eines *speedikon* Anwenders ist schwer zu beschreiben, da *speedikon*® sehr universell eingesetzt wird. So ist beispielsweise der Architekt ein typischer Kunde, ebenso können auch Bauämter und Fertigteilhersteller *speedikon* effektiv einsetzen, um nur 3 Kundentypen zu nennen.

Abbildung 42 zeigt einen Workflow zwischen Bauherr, Architekt und Bauamt, wie er als typisch für ein Architekturbüro bezeichnet werden kann.

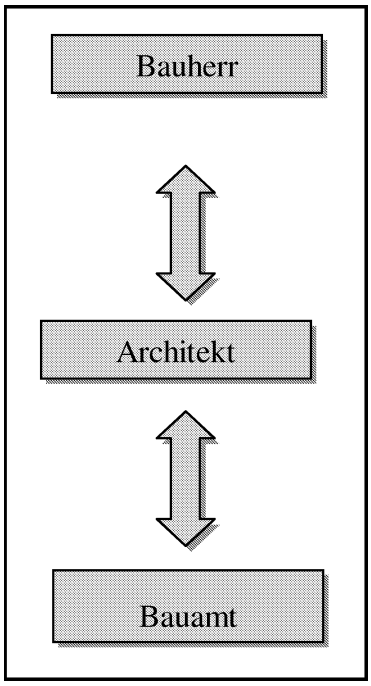

Abbildung 42: Workflow Bauherr, Architekt und Bauamt

Eingeleitet wird der in Abbildung 42 gezeigte Vorgang von dem Bauherren, der ein Bauvorhaben hat. Er kommt mit seinen Vorstellungen zum Architekten und beschreibt ihm seine Wünsche und Nutzungsvorstellungen.

Der Architekt erstellt daraufhin gemeinsam mit dem Bauherren grobe Skizzen, aus denen dann ein Plan im Maßstab 1:100 mit Hilfe von CAD Software erstellt wird.

Dieser Plan geht zur Bauvoranfrage zum Bauamt. Es folgen dann möglicherweise mehrere Iterationsschritte zwischen Architekt und Bauamt bis das Bauvorhaben dem Flur Bebauungsplan und dem Gesetz entspricht.

Sobald die Bauvoranfrage genehmigt ist, entwirft der Architekt in Abstimmung mit dem Bauherren den detaillierten Bauplan und stellt den Bauantrag.

Bei diesem Arbeitsschritt kann es mehrere Iterationsschritte zwischen Architekt und Bauherr geben, hierbei können 3D Visualisierungen des Bauvorhabens wertvolle Dienste leisten um Missverständnisse zu vermeiden.

2.4.4.1.2 Workflow Analyse

In Abschnitt 2.4.4.1.1 wurde ein denkbarer Workflow eines *speedikon®* Anwenders beschrieben. Daraus geht hervor, dass die Vorstellungen des Architekten bei den Bauherrn oft Probleme aufwerfen.

Für den Architekten ist das Planen und Konstruieren von Gebäuden Alltag, für ihn sollte es

kein Problem darstellen, sich einen Grundriss, den er selbst entworfen hat, 3-dimensional vorzustellen.

Bei den Bauherren verhält sich das anders, er baut in der Regel im Laufe seines Lebens nur ein Haus, entsprechend schwer hat er es sich einen Grundriss 3-dimensional vorzustellen.

Abgesehen davon hat der Bauherr im Laufe der Zeit oft neue Ideen, die er in sein Haus einfließen lassen möchte. So ist es Standard, dass der Plan oft geändert werden muss. Da solche Änderungen Räume völlig anders wirken lassen können, als es beabsichtigt war (es können sich Türen und Fenster verschieben), ist es von großer Bedeutung, sich jederzeit ein 3-dimensionales Bild des Bauwerks erzeugen zu können.

Um Enttäuschungen vorzubeugen, müssen demnach 3D Graphiken und Kameraflüge eingesetzt werden, um die Zusammenhänge zu verdeutlichen.

Hier ist der Ansatzpunkt von HiQoS. Hierdurch wird es ermöglicht, komplexe Zusammenhänge in der Gebäudeplanung hochwertig und schnell darzustellen.

Hieraus könnte sich ein Dienst etablieren, den Architekturbüros nutzen, um ihren Kunden in relativ kurzer Zeit photorealistische Bilder und Animationen zur Verfügung stellen zu können.

2.4.4.2 Einsatz und Erstellung von 3D Gebäudemodellen

Es gibt prinzipiell zwei Wege, um zu einem dreidimensionalem Gebäudemodell zu gelangen:

1. Bestandserfassung eines vorhandenen Gebäudes in einem 3D CAD System wie *speedikon*.

2. Neuplanung eines Gebäudes mit einem 3D CAD System wie *speedikon*.

2.4.4.2.1 3D Gebäudemodell für die Bestandserfassung eines vorhandenen Gebäudes

Ziel einer Bestandserfassung ist die effektive Bewirtschaftung des Gebäudes. Aus dem Gebäudemodell lassen sich nach der Erfassung beliebig Massen ermitteln, Flächen auswerten, Perspektiven und Ansichten berechnen. Ein solches Gebäudemodell bietet zudem eine hervorragende Grundlage für Umbau- und Erweiterungsmaßnahmen.

Grundlage einer Bestandserfassung sind alte Baupläne und Schnitte oder 2D CAD Pläne. Gibt es keine Pläne mehr, oder sind solche Pläne zu veraltet, wird ein örtliches Aufmaß gemacht.

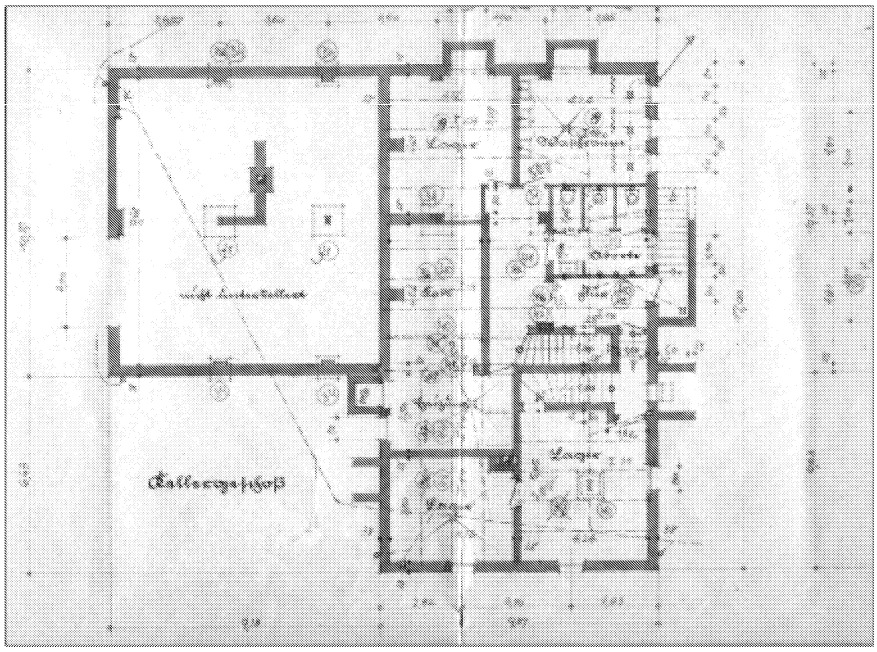

Abbildung 43: Gescannter Plan von 1952 (siehe auch Abbildung 50)

Ziel der 3D Bestanderfassung sind insbesondere Innenraumperspektiven, um zu verdeutlichen, wie bei einer Umgestaltung der neue Raumeindruck ist. Von solchen Bildern werden Entscheidungen abhängig gemacht, welche Materialien für Wand-, Boden und Deckenbekleidungen und Mobiliar verwendet wird.

Abbildung 44: Innenraum Bild aus *Arcon*

2.4.4.2.2 3D Gebäudemodell für die Neuplanung eines Gebäudes

Im Idealfall plant der Architekt oder ein beteiligtes CAD Büro schon während des Bauantrags mit einem Gebäudemodell.

Man hat dann sehr gute Möglichkeiten, das Bauamt und den Bauherren anhand von Visualisie-

rungen von dem geplanten Vorhaben zu überzeugen. Auch die Fortführung der Planung erfolgt in diesem Gebäudemodell, so dass Redundanzen (z. B. Schnitte und Ansichten, die nicht zum Grundriss „passen“) vermieden werden.

Weiterhin verkürzt sich die Entscheidungsfindung für Alternativplanungen deutlich, da man die Möglichkeit hat, jede Alternative in fotorealistischen Bildern vorzulegen. Das ist deshalb so wichtig, weil der normale Bauherr selten Baupläne bewertet und sich aus einem Plan nicht das Gebäude vorstellen kann. Es kommt dann zu immer neuen Entwürfen, da der Bauherr keine Fehlentscheidung treffen möchte. Hat er jedoch ein Bild, das ihm zusagt, wird er die Entscheidung sehr schnell fällen.

Das Gebäudemodell selbst wird mit der CAD Software *speedikon*® erstellt, indem Wände, Öffnungen, Treppen, Fußböden, Decken usw. als dreidimensionale Objekte eingegeben werden. Jedem Objekt können seine Abmessungen in der Länge, Breite und Höhe und u.a. sein Material zugeordnet werden.

Das Programm verknüpft die Objekte automatisch und platziert sie ihrer Höhe entsprechend im Raum.

Das Resultat ist ein dreidimensionales Modell des Gebäudes, das im Rechner im Maßstab 1:1 gespeichert ist. Aus diesem lassen sich alle anderen Maßstäbe und Darstellungsformen, wie z. B. ein Drahtmodell zur schnellen Kontrolle, erstellen (Abbildung 45).

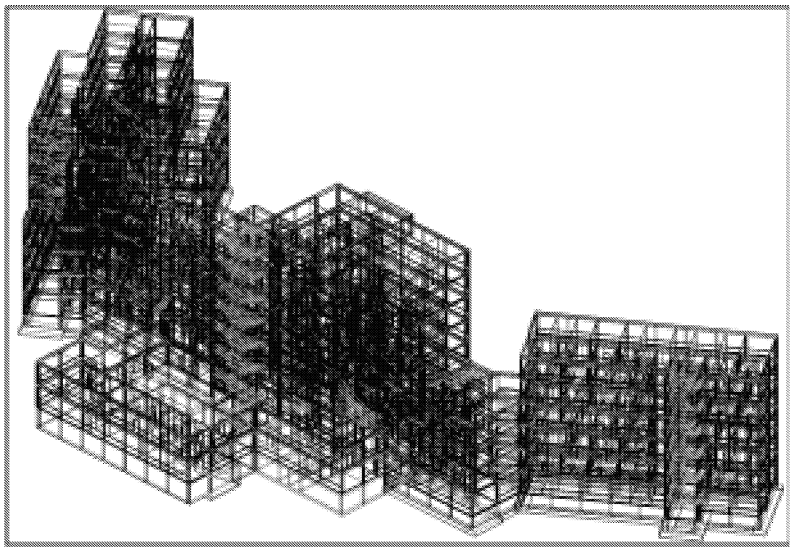

Abbildung 45: Drahtmodell des Bürogebäudes (siehe auch Abbildung 52)

Aus den Modelldaten kann *speedikon*® zu diesem Zeitpunkt Massen und Flächen ermittelt werden, sowie Grundrisse, Schnitte und Ansichten darstellen.

Fotorealistische Bilder sind hier noch nicht möglich.

Dazu werden die CAD-Daten über eine integrierte Schnittstelle an das Visualisierungspro-

gramm *Arcon*®[4] übergeben.

Aus den mit übergebenen Materialinformationen generiert *Arcon* ein Gebäudemodell, das mit den Materialien entsprechenden Texturen belegt ist. Schatten und Lichtquellen werden in dieser Phase noch nicht berücksichtigt.

In *Arcon* können Lichtquellen in Form von Lampen und der Sonne positioniert werden. Diese Lichtquellen können in Ihrer Farbe, Intensität und Einflussbereich verändert werden.

Dies ist die Grundlage für die Erzeugung von fotorealistischen Bildern. Diese werden durch das Ray-Tracing (Strahl Rückverfolgung) Verfahren berechnet. Die daraus resultierenden Bilder beinhalten Schatten und Spiegelungen, was das Modell erst natürlich aussehen lässt.

Diese Berechnung ist sehr rechenintensiv und hängt von

- der Größe des Gebäudemodells
- der Größe der Bilder
- der Antialiasing Stufe
- Anzahl der Lichtquellen

Das bedeutet, dass die Berechung auf dem CAD-PC sehr zeitintensiv ist. Die Dauer der Berechnung einen guten Bildes (mittleres Antialiasing) von einem Mittelgroßen Gebäudes (hier als Beispiel ein Verwaltungsgebäude) in einer Auflösung von 600x800 Pixel (entspricht einer ½ DIN A4 Seite) kann bis zu 8 Stunden betragen. Während dieser Zeit kann der Rechner kaum für andere Tätigkeiten benutzt werden.

2.4.4.3 Erstellung eines 3D Gebäudemodells am Beispiel der Rekonstruktion der Synagoge Mülheim

Im Rahmen des Forschungsprojekts HiQoS sollte die Synagoge Mülheim, die während des Krieges zerstört wurde, im Rechner dreidimensional rekonstruiert werden.

Die GPO Mülheim hat dann alle verfügbaren Informationen über die Synagoge zusammengetragen. Leider sind während des Krieges alle Unterlagen und Pläne verbrannt, so dass zur Rekonstruktion lediglich

- die Gebäudekontur auf Grundlage alter Katasterpläne
- Fotos von außen und innen
- mündliche Informationen durch die jüdische Gemeinde in Mülheim

---

[4] Visualisierer der mb Software AG

- Beschreibungen

zur Verfügung stand.

Auch bei den Nachkommen des Architekten, der den Bau der Synagoge leitete, waren keine Unterlagen mehr vorhanden.

Mit einer dermaßen schlechten Ausgangslage hatte die GPO Mülheim nicht gerechnet, denn auf der Basis dieser Unterlagen ließ sich in der zur Verfügung stehenden Zeit des Forschungsprojekts kein realistisches Modell der Synagoge erstellen. Trotzdem wurden die Arbeiten daran fortgesetzt, so dass man an diesem Beispiel die Erstellung eines solchen Gebäudemodells sehr gut darstellen kann.

1. Schritt: Erarbeitung eines zweidimensionalen Grundrisses

Aus Katasterplänen und Beschreibungen wurden die äußeren Konturen der Synagoge möglichst genau erfasst, so dass auf dieser Grundlage und auf Grundlage der Fotos Abmessungen für den Grundriss erarbeitet werden konnten.

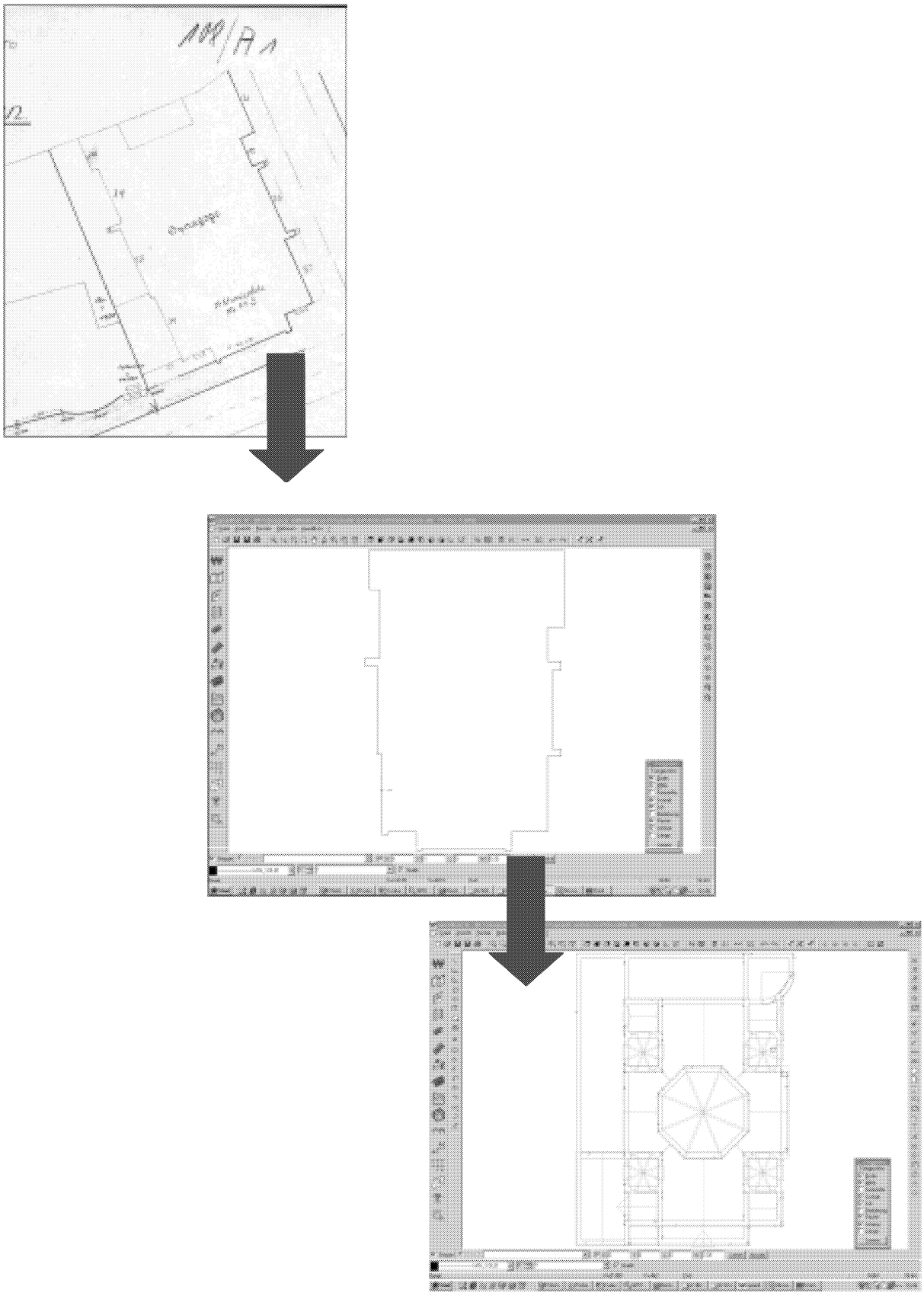

Abbildung 46: Bild Scan Katasterplan und CAD Grundriss

2. Schritt: Ermittlung der Höhen aus den Fotos

Leider kam für die Ermittlung der genauen Maße aus den Fotos die Methode der Fotogrammetrie nicht in Frage, da weder die Brennweite noch der genaue Standpunkt des Fotographen bekannt waren.

Deshalb mussten aus bekannten Maßen (Höhe der Synagoge oder Länge und Breite, die im Schritt 1 ermittelt wurden) Faktoren berechnet werden, mit denen die gemessenen Längen korrigiert wurden, um so auf Realmaße zu kommen.

3. Schritt: Erstellung eines Drahtmodells

Hat man die Höhen und Längen, kann man mit der CAD Eingabe beginnen, ein dreidimensionales Modell aufzubauen. In diesem Modell fehlen noch sämtliche Fenster, Türen und Öffnungen. Dieses Modell ist eine Arbeitsgrundlage, die zeigt, ob die gemessenen Maße stimmig sind.

Aus diesem Hilfsmodell resultiert ein dreidimensionales Raster, in das die im folgenden zu modellierenden Bauteile eingepasst werden.

Erst wenn die Relationen aus Bildern perfekt zu den Relationen des „Hilfsmodells“ passen, kann mit dem nächsten Schritt begonnen werden.

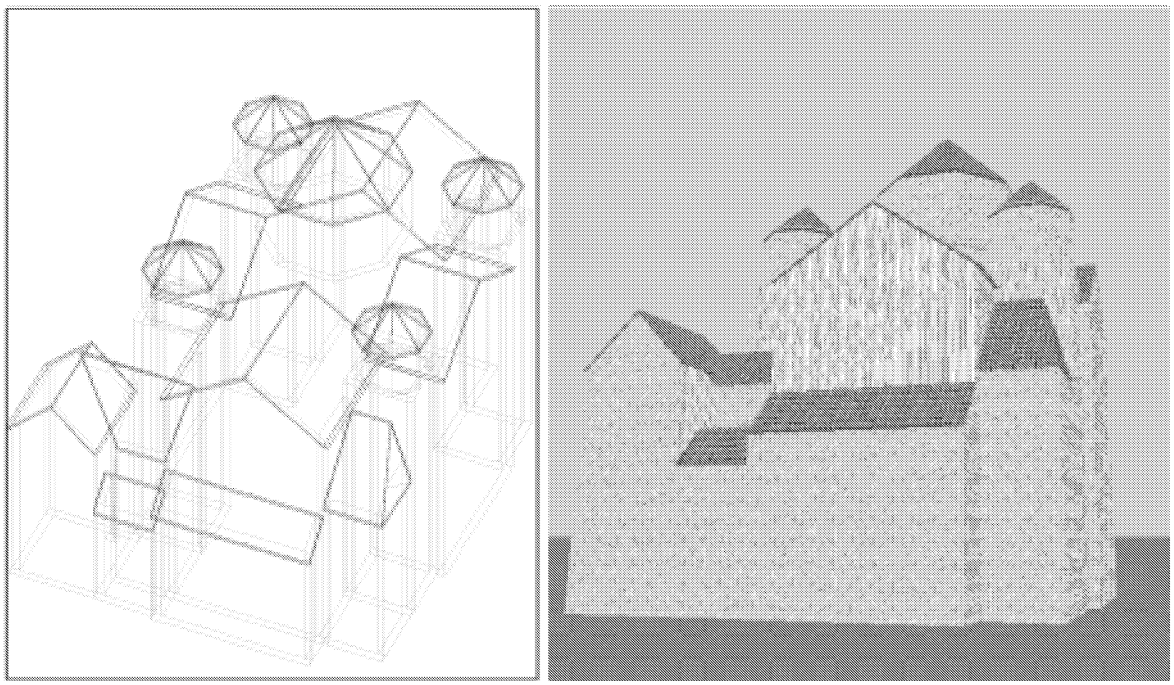

Abbildung 47: Drahtmodell der Synagoge und gerendertes Hilfsmodell

4. Schritt Erstellung der Einbauelemente wie Türen, Säulen usw.

Nun muss das Bauwerk in einzelne Bauteile „zerlegt“ werden. Das heißt, es wird versucht, möglichst gleiche Bauteile zu finden, die dann einmal erstellt, und dann kopiert werden können. Dazu eignen sich besonders Säulen, Öffnungen, Inneneinrichtungen, Verzierungen usw.

Mit Hilfe der Ergebnisse der vorangegangenen Schritte können nun diese Bauteile modelliert werden. Die Vorgehensweise ist dieselbe wie die für die Erstellung des „Hilfsmodells“:

- 2D Skizzen und Zeichnungen
- Ermittlung von Höhen und Materialien
- Erstellung des dreidimensionalen Baukörpers mit Materialdefinition

Die GPO Mülheim arbeitet zur Zeit an diesem Schritt, da die Erstellung der einzelnen Baukörper durch das Fehlen von Plänen und Skizzen sehr aufwendig ist

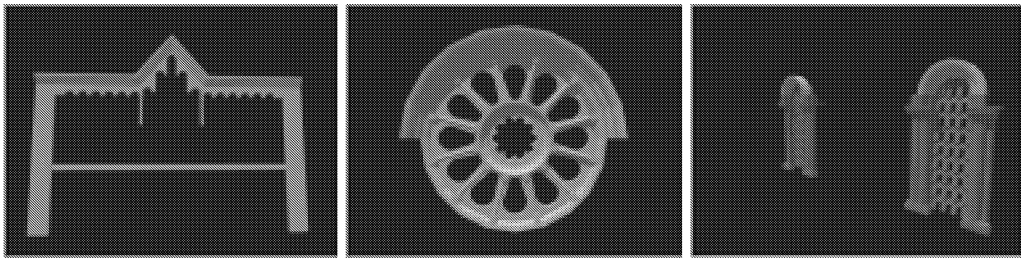

Abbildung 48: Bilder einzelner Bauteile der Synagoge Mülheim

5. Schritt Zusammenfügen der Bauteile

Wenn alle Bauteile erstellt sind, werden sie entsprechend des dreidimensionalen Rasters im Raum platziert. Als Resultat erhält man das fertige Bauwerk.

6. Schritt Vorbereitung der Visualisierung

Den verwendeten Materialien müssen nun möglichst originalgetreue Texturen und Materialeigenschaften gegeben werden, um Bilder zu erhalten, die der Realität nahe kommen. Die Materialeigenschaften müssen aus Erfahrung und Tests bestimmt werden. Die Texturen erhält man durch Fotos, die man von der Oberfläche noch existierender Gebäude aus der entsprechenden Zeit macht und in den Rechner einliest.

Weiterhin müssen Lichtquellen in dem Gebäude mit Ihren Eigenschaften definiert werden, um eine gleichmäßige Ausleuchtung des Gebäudes zu erhalten.

7. Schritt Berechnung der Bilder

Der letzte Schritt ist reine Rechenarbeit des Computers. Bei großen Modellen mit zahlreichen Lichtquellen und vielen Kanten ist diese Berechung sehr zeitaufwendig und kann trotz moderner PCs mehrere Tage in Anspruch nehmen.

Wegen des unerwartet großen Aufwands durch fehlendes Planmaterial hat die GPO Mülheim folgende Alternativen zur Evaluierung des HiQoS Systems zur Verfügung gestellt:

- Kurzfristig waren erste Tests der Software nur mit Demoprojekten von *Arcon* möglich, da diese eine geringe Größe hatten und die Berechnung entsprechend schnell lief und

Fehler leicht gefunden werden konnten

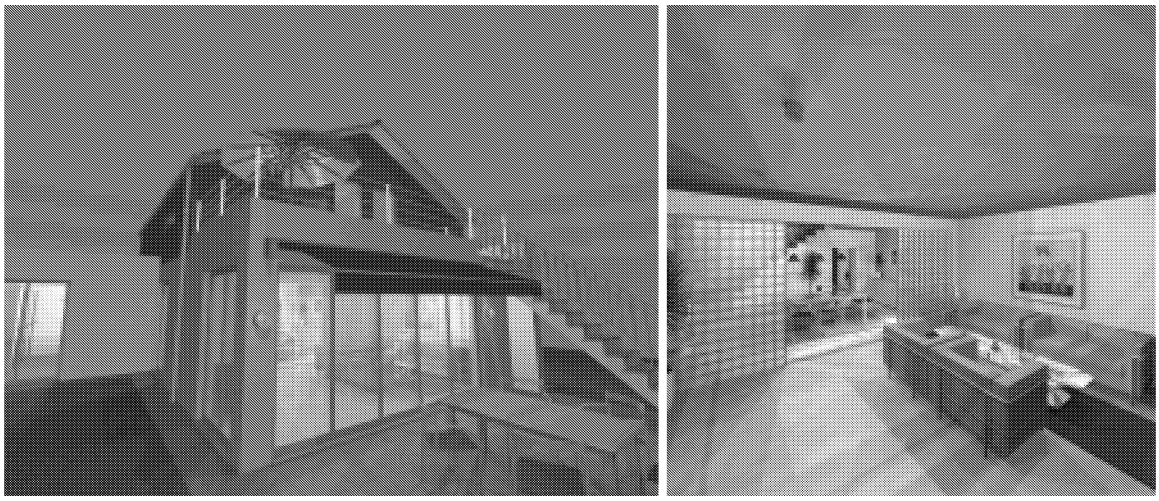

Abbildung 49: Modell *Ferienhaus*

- Aufnahme des Bürogebäudes der GPO Mülheim in Mülheim an der Ruhr. Damit steht ein Testobjekt zur Verfügung, das in der Größe einem Mehrfamilienhaus entspricht. Objekte dieser Größe werden sehr viel von kleinen und mittleren Architekturbüros gebaut, die eine Zielgruppe darstellen.

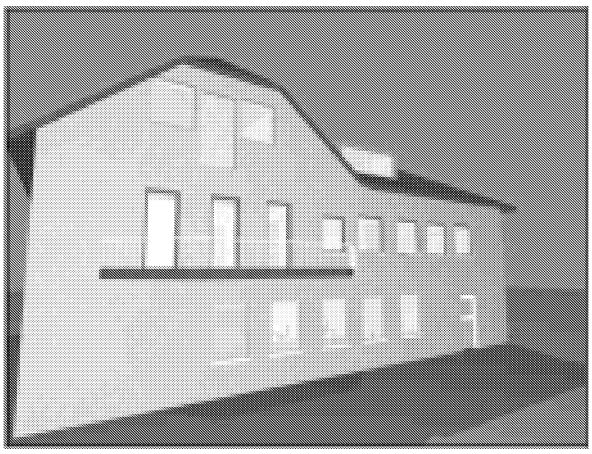

Abbildung 50: Modell *GPO* (siehe auch Abbildung 43)

- Ein Verwaltungsgebäude, das uns als Datensatz von einem Kunden zur Verfügung gestellt wurde. Dieses Gebäude wurde durch die GPO für die Visualisierung vorbereitet und möbliert. Weiterhin wurden Lichtquellen eingefügt. Dieses Modell zeichnet sich durch die Anzahl der Lichtquellen und die Inneneinrichtung aus. Es entspricht von der Größe her einem normalen Gewerbebau.

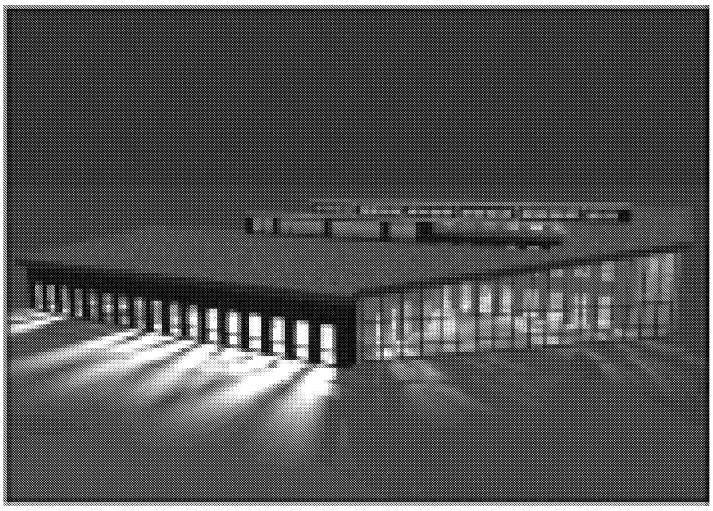

Abbildung 51: Modell *Verwaltung*

- Ein Bürogebäude, das uns ebenfalls durch einen Kunden zur Verfügung gestellt wurde. Dieses Gebäude hat ca. 15.000 m² BGF und entspricht damit einem großen Gebäude, das meist in städtischer Lage anzutreffen ist, und Wohn- und Büroraum zur Verfügung stellt. Dieses Gebäude wurde durch die GPO Mülheim ebenfalls mit Texturen und Materialeigenschaften versehen. Dieses Modell ist von der Größe, dass man als Drahtmodell gerade noch mit einem normalen PC bearbeiten kann. Rendering ist bei dieser Größe in einer akzeptablen Zeit nicht möglich.

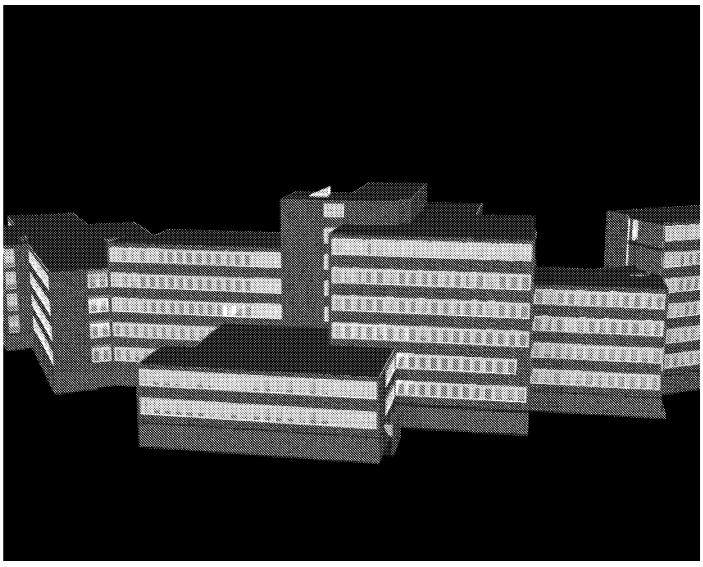

Abbildung 52: Modell *Bürogebäude* (siehe auch Abbildung 45)

Durch diese Auswahl an Gebäuden konnte das System mit der Art von Gebäuden getestet werden, mit denen es auch später in der Praxis genutzt werden soll.

2.4.4.4 Datenfluss in Ray-Tracing Szenario

Die Abbildung 53 zeigt den Datenfluss im Ray-Tracing Szenario. Der Benutzer speichert das in *Arcon* (oder *speedikon*) erzeugte Modell als VRML 2.0 Format. Dann wählt er einzelne Kameras oder einen Kamera-Pfad. Die VRML 2.0 Datei und die Kamera-Datei platziert er auf Web und teilt dem HiQoS Rendering System ihre Adresse mit. Der Rest läuft automatisch. Am Ende werden die Ergebnisse nach Wunsch entweder dem Benutzer per E-Mail geschickt oder auf dem HiQoS Server gespeichert, so dass sie später über *http* zugegriffen werden können.

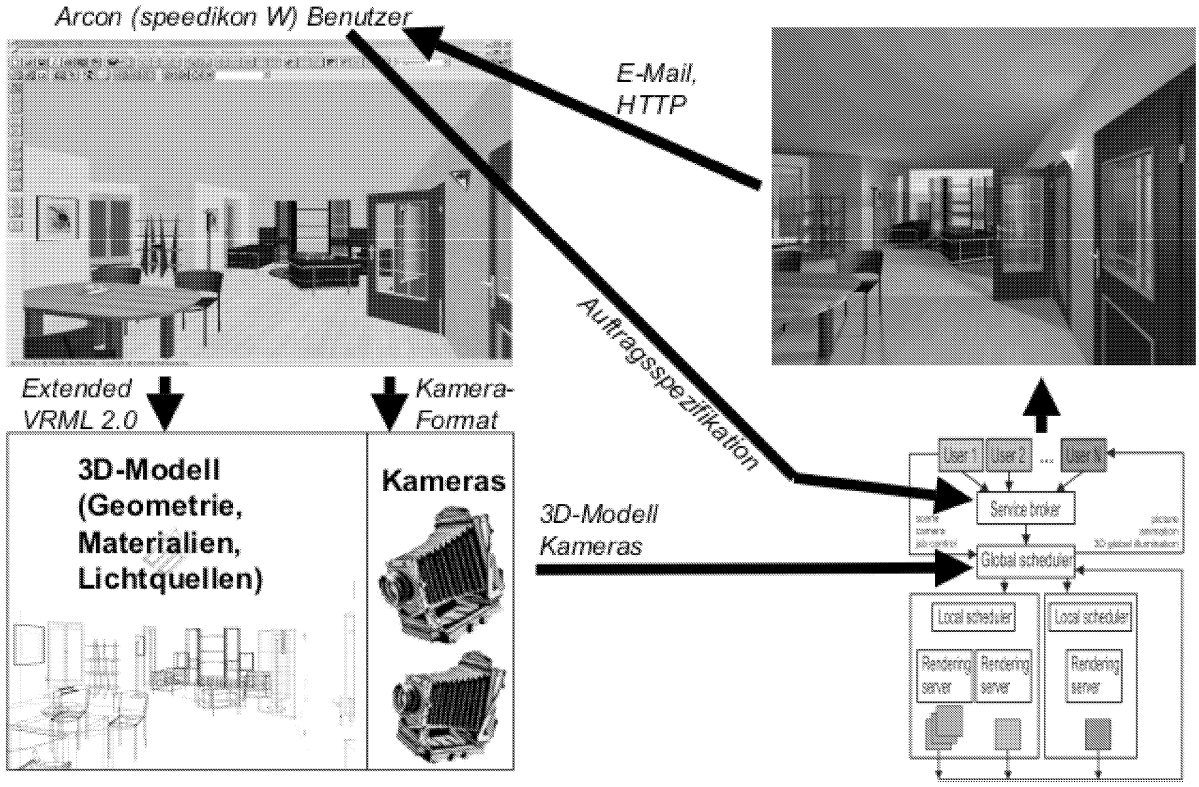

Abbildung 53: HiQoS Rendering System: Datenfluss im Ray-Tracing Szenario

2.4.5 Rendering in der Filmproduktion

UPSTART! benutzt HiQoS für den Bereich Visual-Effects bei Filmproduktionen. Während für die IEZ AG die Berechnung fertiger Bilder oder Animationen mittels Ray-Tracing interessant ist, liegt bei UPSTART! das Hauptinteresse bei der Berechnung der globalen Beleuchtung in komplexen Szenen basierend auf der Radiosity-Methode für eine möglichst photorealistische Lichtsituation, die mit herkömmlichen Renderingverfahren nur schwierig zu simulieren ist.

Haupteinsatzgebiete für Visual-Effects sind Film- und Fernsehproduktionen. Aber auch im Multimedia- und Computerspielmarkt kommen sie zum Einsatz.

2.4.5.1 Workflow

Der im folgenden abgebildete Prozessablauf zeigt die wichtigsten Arbeitsschritte während einer Filmproduktion.

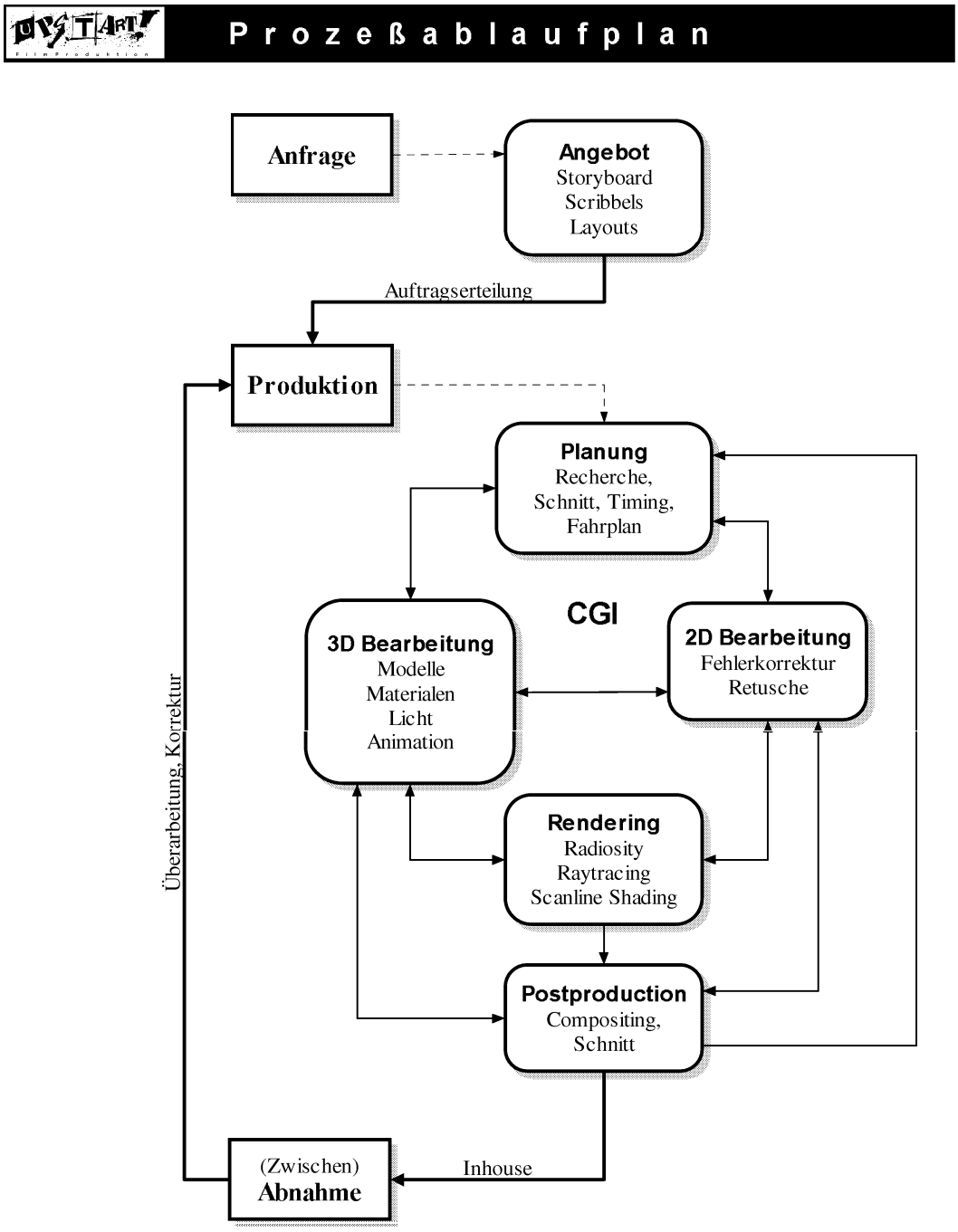

Abbildung 54: Prozessablaufplan einer Filmproduktion

Am Anfang des Prozesses erfolgt die Anfrage eines potentiellen Kunden. Dieser hat eine mehr oder weniger klare Vorstellung davon, was er gerne möchte. Zunächst werden die Vorstellungen des Kunden durch Storyboards, Scribbles und Layouts präzisiert, um auf dieser Basis ein

Angebot zu erstellen.

Kommt es zur Auftragserteilung beginnt Phase 1 der Produktion, die Planung. Es werden Rohmaterialien recherchiert bzw. In-house generiert. Im Grobschnitt wird das Timing festgelegt und daraus ein Fahrplan erzeugt, welcher genau beschreibt wie lange eine bestimmte Einstellung dauert.

Steht der Fahrplan, fällt für das CGI-Department der eigentlichen Startschuss, die Produktion kann beginnen. Gegebenenfalls vorhandene Filmsequenzen werden digitalisiert und 2D bearbeitet. 3D Modelle werden gebaut, Licht, Material und Animation werden erzeugt und schließlich mit unterschiedlichen Verfahren gerendert. In der Postproduction werden die einzelnen Bildebenen schließlich zusammengesetzt.

Alle erzeugten Bilder bzw. Bildsequenzen aus der 2D und 3D Bearbeitung, sowie die der Postproduction fließen je nach Aufgabenstellung wieder in die einzelnen Produktionsprozesse ein. Um die Produktionskosten zu minimieren werden wichtige Teilergebnisse dem Kunden bei Zwischenabnahmen präsentiert. So wird die Produktion schrittweise über mehrere Zyklen verfeinert, bis das gewünschte Endergebnis vorliegt und es erfolgt schließlich die Endabnahme durch den Kunden.

2.4.5.2 Rendering + Photorealismus

Bei komplexen 3D Szenen steht oft nicht genügend Rechenzeit zur Verfügung um optimale Ergebnisse bei Lichtgestaltung und somit Photorealismus zu erzielen.

Dies liegt nicht an der insgesamt zur Verfügung stehenden Rechenleistung In-house, sondern daran, dass die photorealistische Ausleuchtung einer Szene oft nur durch Radiosity-Berechnung zu realisieren ist.

Bei UPSTART! sind über 30 Grafik-Workstations, meistens mit zwei Prozessoren, im Einsatz. Es stehen somit über 50 Prozessoren zur Verfügung. Insgesamt eigentlich genügend Rechenleistung zur Radiosity-Berechnung. Da es z. Z. jedoch keine kommerzielle Software gibt, welche eine Radiosity-Solution auf mehreren Prozessoren und Computern parallel im Netzwerk berechnen kann, wird oft auf eine Radiosity Berechnung verzichtet und herkömmliche Renderingverfahren verwendet.

Um bei komplexen 3D Szenen eine Radiosity-Berechnung In-house durchführen zu können, müsste es also eine kommerzielle Software geben, die den Berechnungsprozess auf mehrere Prozessoren und Computer verteilt. Es ist jedoch nicht zu erwarten, dass in absehbarer Zeit eine solche Software auf den Markt kommen wird, da es immer noch keinerlei Produktankündigungen der namhaften Hersteller für kommerzielle Radiosity-Software gibt. HiQoS ermöglicht die parallele Radiosity-Berechung mit den Algorithmen des Paragraph Forschungsprojektes.

Das Rendering ist zwar nur ein Teil des Produktionsprozesses innerhalb einer Filmproduktion, sie ist aber meist einer der zeitaufwendigste Teile. Pro Bild kann es zu Rechenzeiten von mehreren Stunden auf modernen Grafik-Workstations kommen, wobei die Hauptzeit bei der Berechnung der Lichtsimulation liegt.

2.4.5.3 Datenfluss in Radiosity Szenario

Die in 3D Studio Max erstellte 3D Geometrie wird mit Beleuchtungs- und Materialparameter im 3DS Format exportiert und per Web-Interface an den HiQoS Rendering-System gesendet. Dieser berechnet die Radiosity-Solution mit den vom Benutzer eingestellten Parametern. Um die Datenmenge zu reduzieren wird die Radiosity-Solution vereinfacht und die globale Beleuchtungsinformationen als Textur gespeichert. Das Ergebnis wird dem Benutzer wie beim Ray-Tracing Verfahren auf Wunsch per E-Mail geschickt oder auf dem HiQoS Server gespeichert, so dass sie über *http* abgerufen werden können.

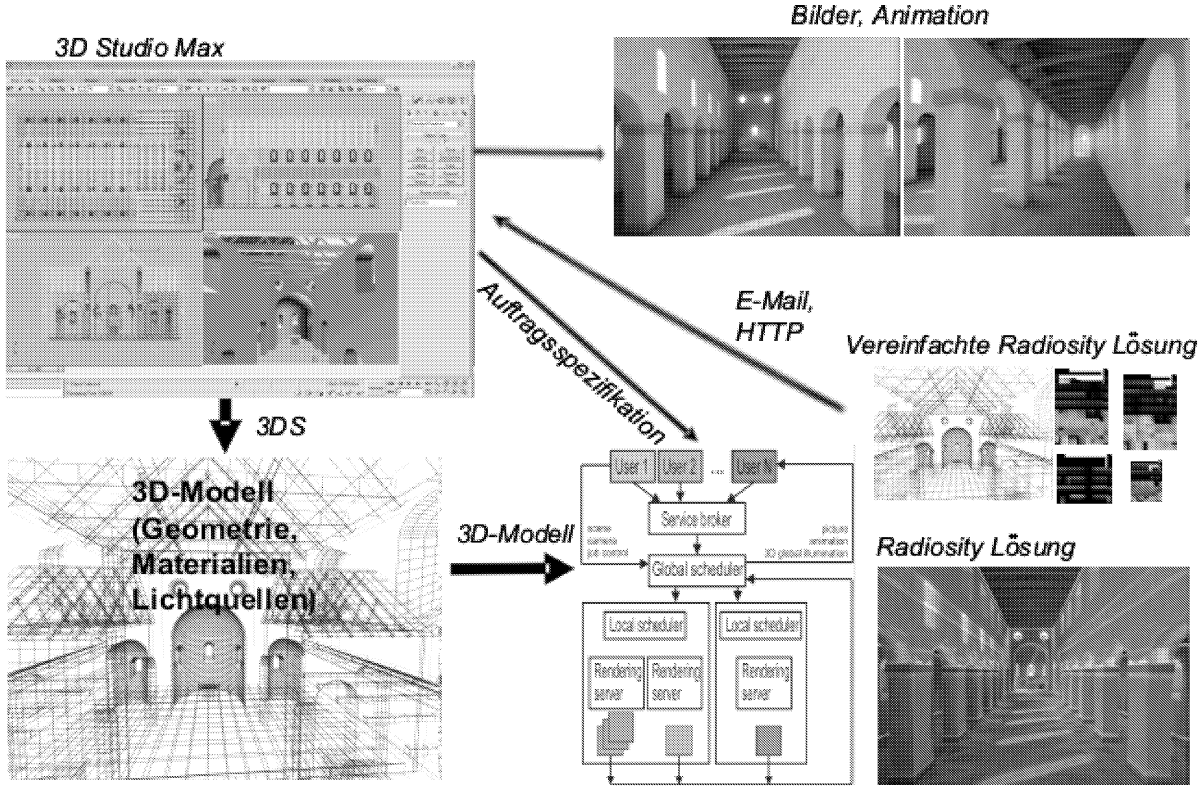

Abbildung 55: HiQoS Rendering System: Datenfluss im Radiosity Szenario

2.4.6 Live-Streaming-Demonstrator

Mithilfe einer Beispiel-Anwendung [41] aus dem Bereich der eingebetteten Systeme sollte gezeigt werden, dass die in HiQoS entwickelten Verfahren zur QoS-gerechten Video-on-Demand-Abwicklung auch für Live-Streaming geeignet sind. Dazu wird ein von mit einer Kamera versehenes Miniatur-Fahrzeug (*Pathfinder*) (siehe Abbildung 56) von Benutzern über ein IP-Netz gesteuert. Der so gelieferte Video Stream wird in ein QoS-Netz eingespeist.

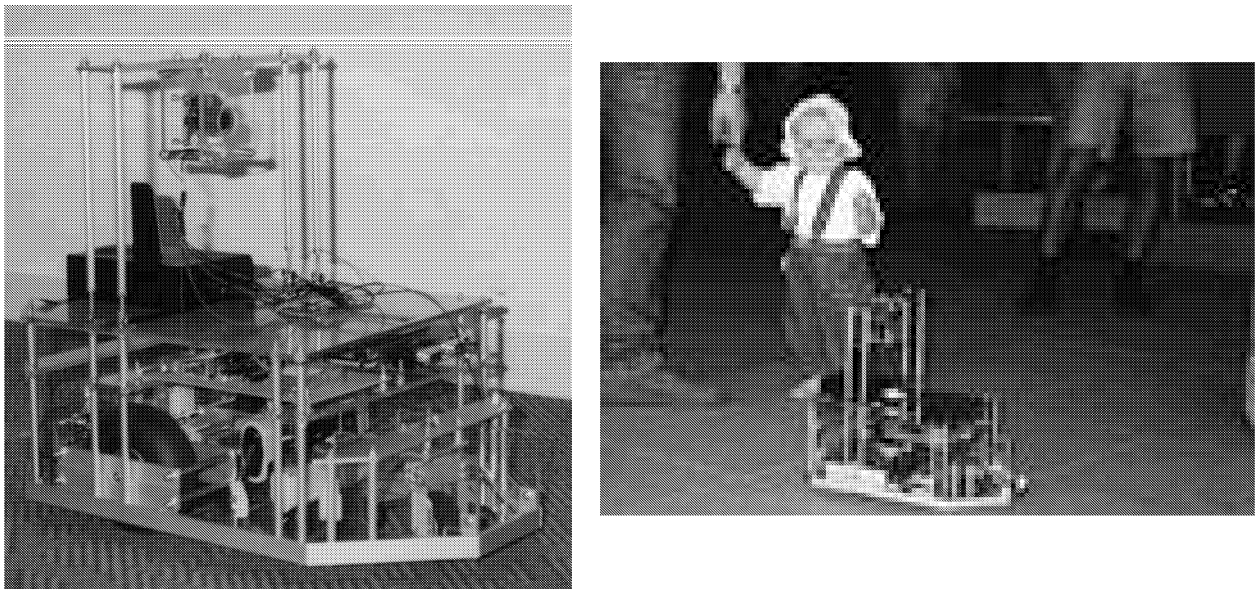

Abbildung 56: Der C-LAB Pathfinder

Die mechanische Steuerung erfolgt benutzerseitig über ein Java-Applet, das gleichzeitig mit den Videobildern in einem Browser dargestellt wird. Die im Applet generierten Steuersignale werden über einen Steuer-Server und eine Funkübertragungsstrecke dem Fahrzeug übermittelt und dort ausgewertet. In diesem Trial gilt es, dem Benutzer unter zeitlichen Gesichtspunkten eine vernünftige Steuerung des Fahrzeugs zu gewährleisten.

2.4.6.1 Bisherige Lösung (ohne HiQoS)

Schwerpunkt bei dieser Anwendung ist die Realisierung des Capturing und Präsentation der von den Kameras aufgenommenen Bilder. Hierbei wurde zunächst im Pathfinder-Projekt auf herkömmliche Streaming-Technologie wie dem *Real Player* von *Real Networks* aufgebaut. Dieser besteht aus einem Video-Enkoder, einem Server und einer Client-Lösung mittels Browser-Plugin. Das von der Kamera des Fahrzeugs aufgenommene Videosignal wurde über eine Funk-Übertragungstrecke einer Grabberkarte zugeführt und dort digitalisiert. Im nächsten Schritt wird das digitalisierte Videosignal dem Video-Enkoder übergeben und anschließend auf dem Video-Server zwischengespeichert. Die Kommunikation zwischen Video-Server und Video Plugin (Client) lief über eine proprietäre Schnittstelle. Allerdings waren die Ergebnisse für eine erfolgreiche *Human-Computer Interaction* nicht zufriedenstellend, da bei einer ISDN-Anbindung des Clients (64 Kb/s) eine Ende-zu-Ende-Verzögerungszeit der Videosignale von bis zu 1s entstand, was sich bei einer Verringerung der Bandbreite für die Anbindung an das Intra- / Internet noch bis auf fünf Sekunden erhöhen kann. Dieser Wert ist inakzeptabel, da eine annehmbare Steuerung der Fahrzeuge so nicht erreicht werden kann. Des weiteren kann t.w. nur eine Bildwiederholrate von 5 Bildern/Sekunde erreicht werden. Dieser Wert entspricht einer durchschnittlichen Darstellungszeit von über 100ms pro Bild.

In HiQoS war daher das Ziel im Rahmen dieser Anwendung, die Verzögerungszeit beim Übertragen des Videosignals auf 300ms zu senken, so dass anhand der auf dem Client dargestellten Videobilder eine bessere Steuerung des Fahrzeugs möglich ist. Sofern technisch machbar, sollte eine Bildwiederholrate von 25 Frames/Sec. angestrebt werden. In jedem Fall aber muss das

Erreichen solcher Eckdaten konzeptionell garantiert sein, sofern eine entsprechende Hardware vorliegt. Deswegen wurde eine eigene Lösung realisiert, um diese Ziele zu treffen.

2.4.6.2 Neue Lösung (mit HiQoS)

Jeder Pathfinder wird über eine Funkstrecke an einen dedizierten PC verbunden. Dort wird das analoge Video-Signal durch eine Steckkarte („MPEG Encoder“) in einen MPEG-1- (oder wahlweise MPEG-2) Strom umgewandelt. Der Video-Strom wird über einen oder mehrere Router zunächst an den HiQoS-Server weitergeleitet. Dort werden die Ströme aller Fahrzeuge zentral verwaltet (ggf. besteht dort die Möglichkeit, Sequenzen aufzuzeichnen). Der Server leitet die Ströme an die jeweiligen Clients via der entsprechenden Router weiter. Die einzelnen Komponenten sind in der dargestellt.

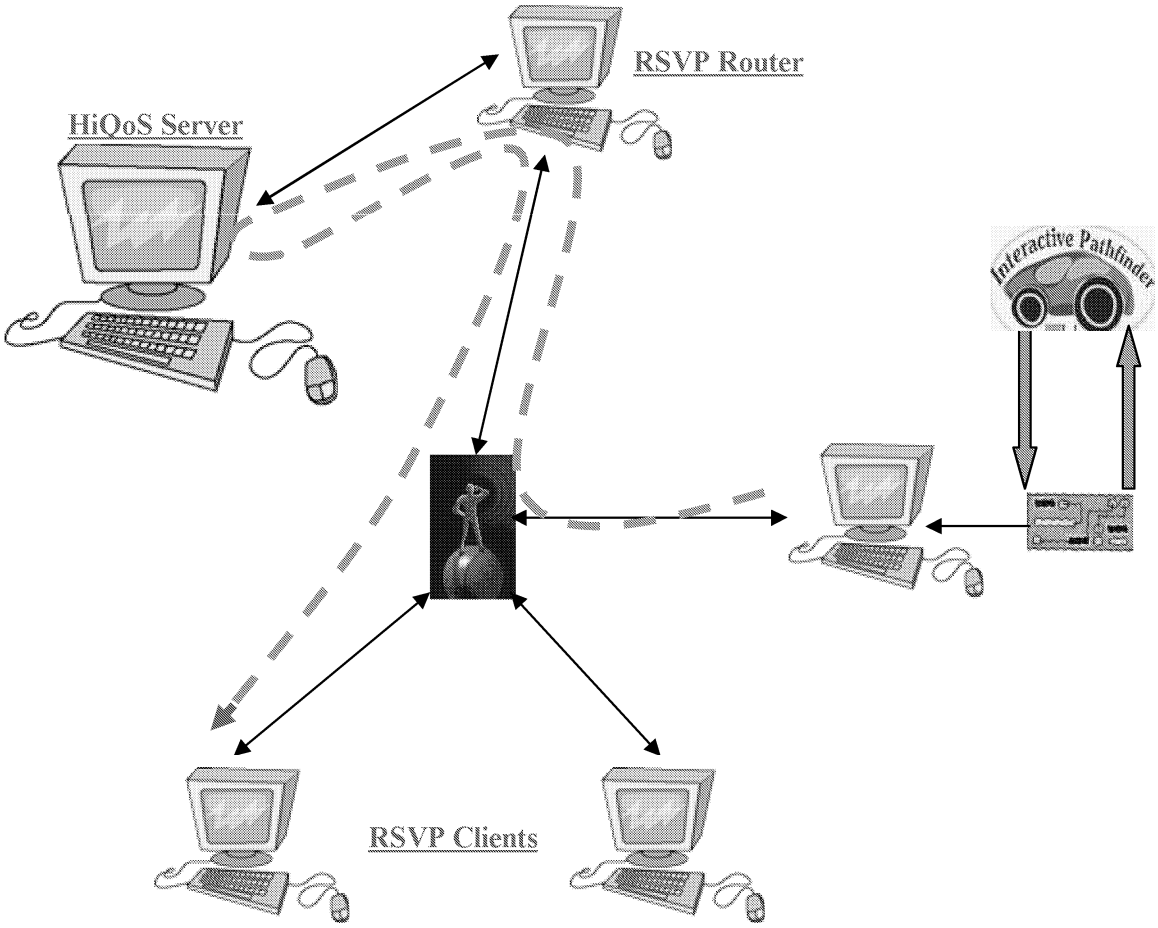

Abbildung 57: Das Live-Streaming-Szenario

Die zur Steuerung der Fahrzeuge nötigen Signale werden in umgekehrter Richtung vom Client zum HiQoS-Server verschickt. Von dort aus werden in zentraler Art und Weise alle Steuersignale per Funkstrecke an die einzelnen Fahrzeuge weitergeleitet.

Im Folgenden werden die Aufgaben der einzelnen Komponenten im Live-Streaming-Szenario beschrieben.

1. PC an der Kamera

Die Aufgabe dieses mit einer Grabber-Karte ausgestatteten PCs besteht ausschließlich aus der Entgegennahme des analogen Video-Signals der Kamera und dessen Umwandlung in einen MPEG-1-Strom, welcher über eine RSVP-Schnittstelle in das HiQoS-Netz eingespeist wird. Eine Hardware-MPEG-Encoder-Karte der Marke *MPEGRich* sorgt für einen Videostream, der über das Netz weitergeleitet wird. Beim Transport werden gängige Protokolle wie z.B. RTP und RTSP eingesetzt.

Um unser Ziel der Benutzer-zufriedenstellenden Videoqualität zu erreichen, ist ein Realtime-Verfahren notwendig. Die MPEG-Video Stream wird nach den entsprechenden *Internet Requests for Comments (RFCs) Drafts* paketisiert und bearbeitet. Beim MPEG Stream handelt es sich entweder um einen MPEG1 oder einen MPEG2 Stream. Der entscheidende Punkt dabei ist die eventuelle Verfügbarkeit eines MPEG2-fähiges Player auf der Benutzerseite. Da die Realisierung dessen mit dem heutigen Rechnerleistung nur mit Hilfe eines Hardware-unterstützten Dekoders möglich ist, bleibt für Allgemeinzwecke alleine MPEG1 als die Wahl.

Es wurden die folgenden Softwarekomponenten für die Win32-Plattform implementiert:

- Eine zentrale Anwendungskomponente für das Einbinden der anderen Komponenten
- Eine Komponente für die Steuerung der Enkoder-Karte
- Eine Komponente für MPEG Stream Parsing, RTP-Header-Erstellung und Paketisierung
- Eine Komponente für RTP-Datentransfer, die eine RTP-Implementierung namens *Rtplib* verwendet
- Eine Komponente für die Client-Server-Kommunikation via RTSP
- Eine Komponente für die RSVP-Protokoll-Steuerung. Diese benutzt die QoS-Dienste des Winsock API bzw. eine Version der PC-RSVP Schnittstelle der Intel-RSVP-Implementierung.

Die zur Verfügung stehende Enkoder-Karte ist durch die *mitgelieferten Software Development Kit (SDK)* zu betreiben. Dieses SDK wurde für DVD und Filmproduktionszwecken entwickelt und unterstützt im Moment nur *Constant Bitrate Encoding*. Deswegen ist das entwickelte fortgeschrittene Realzeit-Scheduling für *MPEG2 Variable Bit Rate* (siehe Abschnitt 2.3.8) noch nicht einsetzbar. Ein weiterer Nachteil des SDKs ist der asynchrone Polling-Mechanismus, welcher beim Datenaustausch zwischen der Treiber der Karte und der Anwendung eingerichtet worden war. Dieses Mechanismus enthält einen Call auf den Device-Treiber, um zu überprüfen, ob enkodierte Videodaten der Anwendung zur Verfügung stehen. Da dieser Call ein blockierender Call ist, verhindert er die Implementation eines Schedulingalgorithmus. Diese wäre wichtig um einen „Network-Flooding“ beim Abschicken der während des blockierenden Calls in der Karte akkumulierten Daten zu vermeiden. Dieses Problem ist momentan durch ein gezwungenes Warten von 1 Millisekunde beim Abschicken überbrückt, was im Zusammenhang mit Bandbreitenreservierung einigermaßen ausreichend ist. Dem zufolge sind aber leider ab und zu Bildstörungen zu sehen, was sich nur durch Anwendungs-gesteuertes Scheduling korrigieren lässt.

2. RSVP-Router

Die Aufgabe des Routers besteht darin, den Netzwerkverkehr mit QoS-Garantien zu ermöglichen. Dabei ist eine Software Distribution von *Sun Microsystems* zum Einsatz gekommen.

3. HiQoS-Media-Server

Das Szenario bietet zwei Lösungsmöglichkeiten an. Zu betrachten ist die Haupterwartung, dass der Stream mit möglichst kleiner Verzögerung und Qualitätsschwankung beim Benutzer ankommen muss.

1. Der Rechner, an dem der Stream generiert wird, kann als ein Client betrachtet werden, und der Stream kann an einen Streaming-Server weitergeleitet werden: Hier wird der Rechner, an dem der Benutzer sitzt, den Server kontaktieren, um von ihm den Stream zu bekommen (wie in Abbildung 57 dargestellt).

2. Der Rechner kann den Stream direkt an den Rechner des Benutzers leiten. Diese Version ist für den praktischen Einsatz optimiert, der Video Stream wird direkt an den Rechner des Roboter-Steuerers geleitet.

Anhand im Internet vorhandenen Libraries die RTP implementieren ist die RTP Protokoll für beide Lösungen als Mittel der Übertragung einsetzbar. Falls - nach der ersten Variante - der Stream bei einem Streaming Server angemeldet wird, kommt bei der Anmeldung des Streams an den Server RTSP in Einsatz. Die benötigte Bandbreite wird mittels RSVP entlang der Route zwischen Client und Server reserviert.

4. HiQoS-Client

Die Aufgabe des HiQoS-Clients ist das Präsentieren der Bilder und das Absetzen der Steuersignale über die vorhandene RSVP-Schnittstelle.

Der Stream ist mit einem RTP-fähigen Player (z.B. *Java Media Framework*) auf Java-fähigen Plattformen abspielbar, deshalb ist die Integration mit dem Fahrzeugsteuerungskomponent möglich. Die Zeitverzögerung liegt zwischen 1 und 2 Sekunden. Die könnte man verbessern, indem die einzelnen MPEG „*I-*“ oder „*P-*“ Frames von der Karte sofort nach der Enkodierung erreichbar wären. Auf die Verwendung von „*B-*“Frames wurde verzichtet.

2.4.6.3 Fazit

Bisher konnte gezeigt werden, dass das ursprünglich erarbeitete Szenario prinzipiell lauffähig ist. Allerdings ergaben sich noch ein paar Probleme, besonders beim Einsatz der ausgewählten Encoder-Karte. Die angestrebte Lösung wird über das Projektende weitergeführt.[5]

Hoffnung dabei bietet eine vor kurzer Zeit für die Karte erschienene SDK für Linux, wo die Größe des asynchron-abfragbaren Datentransferbuffers sich beliebig klein einstellen lässt, und

[5] Inzwischen konnte ein erster Demonstrator entwickelt werden.

dadurch schon auf die einzelne Frames Zugriff bietet.

Die Hindernisse stammen also aus der technischen Bedingungen in Zusammenhang mit der Enkoder-Karte und der Player Software. In der Zukunft wird das Programm auf Linux portiert und ein Hardware-unterstützter RSVP-fähiger MPEG2-Client ebenfalls für Linux entwickelt. Versuche auf einer Software-betriebene MPEG2-Reproduktion hatten soweit keine zufriedenstellende Ergebnisse, aber in der letzten Zeit sind im Netz auch in diesem Bereich Vorschritte gemacht worden.

Falls zwischendurch die benötigte VBR-unterstützende SDK vorhanden sein würde, wäre der Einsatz von VBR und das verbesserte Scheduling ebenfalls möglich.

Bisher hat sich die Enkoderkarte und die RSVP-Unterstützung der Transport Layer als Erfolg erwiesen. Die anderen Komponenten sind generell genug ausgereift, um eine flexiblen Anwendung zu ermöglichen. Bei der RSVP-Implementationen der TCP/IP-Providers für z.B. Router Soft- oder Hardware, High-Level API-s, Client-Anwendungen usw. ist die eventuelle Kompatibilität zu beachten und überprüfen.

Mit einer sichergestellten Zusammenstellung der Endpoint- und Hop-Komponenten halten wir eine Video- und Tonübertragung mit Realtime-Garantien weiterhin für möglich.

3 Projektplanung und –verlauf

Das Projekt hatte eine Laufzeit von drei Jahren (1. Mai 1998 – 30. April 2001) und wurde durch Siemens SBS C-LAB geleitet. Es war in drei planerische Phasen aufgeteilt, die jeweils mit einer Projekt-Review und einer Planung für die jeweils nächste Phase einher gingen. Die Phasen hatten eine Länge von 15, 12 bzw. 9 Monaten. Das Konzept der jeweils kurzfristigen Neuplanung war gewählt worden, um auf die sich rapide wandelnden Technologien im betrachteten Umfeld eingehen zu können.

Die im Abschnitt 2 behandelten inhaltlichen Themen waren auf die folgenden Arbeitspakete aufgeteilt:

- **AP 0**: Anforderungsanalyse und technische Spezifikation: In diesem Arbeitspaket wurde die grundlegende technische Ausrichtung der nachfolgenden Arbeitspakete und somit von HiQoS insgesamt spezifiziert. Dies beinhaltete eine Re-Definition nach einem Jahr.

- ***AP 1**: QoS-Services auf IP-Netzwerken*: Hier wurden neben grundlegenden Fragen der QoS-Verarbeitung die in den Abschnitten 2.3.1 und 2.3.8 präsentierten Ergebnisse erarbeitet.

- ***AP 2**: Netzwerke paralleler Rendering Server*: Hier wurden die in Abschnitt 2.2 dargestellten Arbeiten vollendet.

- ***AP 3**: Netzwerke von Multimedia Servern*: Hier wurden die in den Abschnitten 2.3.2, 2.3.3 und 2.3.5 präsentierten Ergebnisse erarbeitet.

- ***AP 4**: Integration* diente der Integration der Anwendungsprototypen (Abschnitt 2.4) und der Open-Source-Aktivität (Abschnitt 2.3.6).

- ***AP 5**: Anwendungs-Prototypen*: diente der Evaluierung der Anwendungs-Prototypen (Abschnitt 2.4) über einen Zeitraum hinweg.

Im Projektverlauf kam es zu einzelnen Verzögerungen, die allerdings in keiner Weise problematisch waren oder gar die Ziele des Projektes in Frage gestellt hätten.

Wie erwartet (und durch die jeweils kurzfristigen Planungen Rechnungen getragen) kam es zu Re-Definitionen und auch zur Verwerfung einzelner Tasks. So erwiesen sich beispielsweise beim Rendering die Teilaufgaben *Billing* und *Bildkompression* als wenig relevant und wurden daher durch wichtigere Arbeiten ersetzt. Ebenso konnte die anfangs geplante *CBT-Anwendung* nicht wie beabsichtigt im Siemens-eigenen Intranet angeboten werden. Statt dessen wurde eine Zusammenarbeit mit dem Paderborner Lehrstuhl für Didaktik der Informatik zur Unterstützung eine Lehrerausbildung eingegangen (*ViLM*, siehe Abschnitt 2.4.1). Veränderungen wie diese stellten ebenfalls die Ziele des Projektes nie in Frage.

Die Details dazu sind den Zwischenberichten [42] und den Projektplanungen der einzelnen Phasen [43] zu entnehmen.

4 Ergebnisse der Projektpartner

4.1 APE Ptacek Engineering GmbH

4.1.1 Einordnung in das Verbundprojekt

Die Firma APE Ptacek Engineering GmbH war am HiQoS-Projekt hauptsächlich beteiligt, um die HiQoS Mediaservertechnologie in das CarTV-System zu integrieren und den Videoserver in einem lokalen CarTV-Netzwerk zu installieren (siehe Abschnitt 2.4.2). Außerdem wurde innerhalb dieses HiQoS/CarTV-Trialnetzwerk über einen längeren Zeitraum betrieben, evaluiert und die Ergebnisse in einem ausführlichen Evaluationsreport beschrieben [29].

4.1.2 Ergebnisse

Die Firma APE Ptacek Engineering GmbH arbeitete innerhalb des Projektes HiQoS an folgenden Arbeitspaketen:

- Arbeitspaket 4: Integration
- Arbeitspaket 5: Anwendungs-Prototypen
- Arbeitspaket 6: Projektmanagement

Die Arbeiten in allen drei Arbeitspaketen konnten innerhalb des Projektes erfolgreich abgeschlossen werden. Aufgrund von Mitarbeiter-Weggängen kam es allerdings zu zeitlichen Verzögerungen, da die 12 MM für die Task 4.5 sowie die 6 MM für die Task 5.5 mit reduziertem Mitarbeiterstamm durchgeführt werden mussten. Dadurch hat sich der erfolgreiche Abschluss des Arbeitspaketes 4 um 11 Monate von 01/00 auf 12/00 verschoben. Dies hatte wiederum ein Verschieben des erfolgreichen Abschlusses der Arbeiten im Arbeitspaket 5 von 07/00 auf Mitte 04/01 zur Folge. Die Zeitverzögerungen hatten jedoch keinerlei Auswirkung auf Tasks anderer Projektpartner oder auf den erfolgreichen Abschluss des Gesamtprojektes (siehe hierzu auch den Halbjahresbericht des Zeitraums 07/00 bis 12/00 der Firma APE [27]).

4.1.2.1 Ergebnisse im Arbeitspaket 4 (Task 4.5)

Wie bereits im vorigen Abschnitt erwähnt, konnten die Arbeiten (siehe auch Abschnitt 2.4.2) im Arbeitspaket 4 erfolgreich abgeschlossen werden:

1. Integration des HiQoS-Servers (Linux) in ein Autohaus- bzw. CarTV-Netzwerk (Windows 2000/NT 4.0)

2. Integration des HiQoS-Clients (ActiveX-Steuerlement) in die CarTV-Händlersoftware – damit werden die Videos nicht mehr von der lokalen Festplatte, sondern über das Netzwerk vom HiQoS-Mediaserver abgespielt.

Damit waren die Voraussetzung für die Arbeiten im Arbeitspaket 5 gegeben, in welchem diese integrierten Techniken einer ausführlichen Evaluierung unterzogen wurden.

4.1.2.2 Ergebnisse im Arbeitspaket 5 (Task 5.5)

Nachdem die Integrationsphase technisch abgeschlossen war, konnte mit Task 5.5 begonnen werden.

Die Firma APE begann noch im Dezember 2000 mit der Entwicklung und dem Aufbau eines Evaluationsszenarios für die ausführliche Evaluation und den HiQoS-Testbetrieb. Zeitgleich wurde mit der Entwicklung eines Fragenkataloges als Basis für den Evaluationsreport begonnen, welcher das Ergebnis der Task 5.5 darstellt.

Bis Mitte April 2001 wurden auch sämtliche Aufgaben (siehe auch Abschnitt 2.4.2) des Arbeitspaketes 5 erfolgreich abgeschlossen:

1. Entwicklung und Aufbau eines Evaluationsszenarios für die ausführliche Evaluation und den HiQoS-Testbetrieb

2. Entwicklung des Fragenkataloges als Basis für den Evaluationsreport

3. Ausführliche Evaluation des CarTV-Trials mit diversen Probanten

4. Erstellung des Evaluationsreports

Nähere Informationen und auch Schaubilder zum CarTV-Trial können dem Dokument "Beschreibung des APE-Trials innerhalb des Forschungsprojektes HiQoS" vom Februar 2001 entnommen werden [28].

4.1.3 Zusammenfassung

Trotz der zeitlichen Verzögerung konnten alle Arbeiten erfolgreich bis zum Ende des Projektes abgeschlossen werden. Die Integration der HiQoS-Technik in unsere CarTV-Software war relativ problemlos möglich. In der Anwendung innerhalb des CarTV-Trials erwies sich die HiQoS-Software als sehr stabil. Das haben alle Probanten unserer Evaluationstests auch bestätigt.

Würde die HiQoS-Technik noch etwas funktionsoptimiert (bzw. funktionell erweitert) und vor allem für die integrierende Partei und den Endbenutzer ausführlicher dokumentiert, wäre sie für einen Echtbetrieb-Einsatz innerhalb eines BusinessTV-Systems wie CarTV gar nicht so weit entfernt. Für unseren Zweck wäre es allerdings von Vorteil, wenn der Mediaserver nicht nur unter Linux laufen würde, sondern auch eine Windows NT bzw. Windows 2000 Version existent wäre.

Die detaillierten Ergebnisse unserer Evaluation sind unserem Evaluationsreport zu entnehmen [29].

Abschließend lässt sich sagen, dass die Mitarbeit am Projekt HiQoS für die Firma APE Ptacek Engineering GmbH ein interessantes und lohnendes Unterfangen war. Das Gebiet der hochqualitativen Multimediadienste mit Quality-of-Service-Garantien ist ein wichtiges Thema im Zusammenhang mit BusinessTV-Lösungen. Die APE wird dieses Thema mit Sicherheit auch in Zukunft weiter verfolgen.

Das CarTV-Trial-Szenario wurde im Rahmen des HiQoS-Abschluss-Workshops am 8. Mai 2001 der Öffentlichkeit präsentiert.

4.2 Axcent Media AG

4.2.1 Einordnung in das Verbundprojekt

Die AXCENT Media AG war im Rahmen des HiQoS Projektes an zwei wesentlichen Arbeitspaketen beteiligt:

- Der Entwicklung einer skalierbaren Rendering-Server Architektur
- Der Realisierung eines verteilten Server Management Systems für die Bereitstellung von Breitbanddaten insbesondere Audio/Video Streams hoher Qualität

Die Entwicklung einer Renderings-Server Architektur wurde in Zusammenarbeit mit der Universität Paderborn (Unterauftrag) und den Anwendungspartnern UPSTART! und IEZ durchgeführt. Der Rendering-Server wird durch eine verteilte Architektur von Softwaresystemen realisiert. Der hier gewählte verteilte Ansatz erlaubt es das System zu skalieren und somit eine große Anzahl von Rendering-Servern zu verwalten und für die Renderingsaufgaben einzusetzen. Die Architektur wurde in sehr enger Zusammenarbeit mit der Arbeitsgruppe Prof. Monien der Universität Paderborn konzeptioniert und realisiert.

Die Realisierung eines verteilten Server Management Systems (Distributed Server Management System DSMS) wurde von der Universität Paderborn (PC^2), der Siemens AG und AXCENT durchgeführt. AXCENT hat hier die Grundlage des DSMS Systems mit eingebracht und Teile der Konzeptionierung und Realisierung durchgeführt. Die wesentlichen Arbeiten zur Realisierung wurden hier in Zusammenarbeit mit der Universität Paderborn realisiert.

4.2.2 Ergebnisse

Die Ergebnisse wurden im wesentlichen bereits in den ersten Abschnitten des Berichts beschrieben. Folgende Teilergebnisse wurden von AXCENT erzielt:

4.2.2.1 Verteilte Rendering-Server Architektur

Die in Abschnitt 2.2.5 vorgestellte Service Broker Architektur wurde von AXCENT realisiert und anhand eines parallelen Rendering-Services implementiert. Diese Service Broker Architektur kann dabei nicht nur für derartige Rendering-Services eingesetzt werden, sondern auch für andere verteilte Rechenservices genutzt werde.

4.2.2.2 Verteiltes Server Management System

Auf Basis einer Anzahl verteilt installierter Video Server wurde im Rahmen des HiQoS Projektes eine Software zur Verwaltung der Inhalte auf einem Netzwerk verteilter Media-Server realisiert (siehe Abschnitt 2.3). An diesem Ergebnis hat AXCENT durch die Bereitstellung von DSMS Komponenten beigetragen.

4.2.2.3 Zusammenfassung

Das HiQoS Projekt war aus Sichtweise der AXCENT Media AG ein sehr erfolgreiches Projekt, da hier wesentliche Arbeiten geleistet wurden die für das aktuelle und zukünftige Geschäft der AXCENT Media AG relevant sind. Insbesondere die Algorithmen zur verteilten Koordination von Rechenaufgaben (Rendering-Server), sowie zur Datenverteilung (Netzwerke verteilter Mediaserver) sind hier hervorzuheben.

4.3 GPO mbH

4.3.1 Einordnung in das Verbundprojekt

Der Gegenstand der Firma GPO mbH ist die Analyse und Verbesserung von Planungsprozessen und Prozessen zum Betreiben von Gebäuden, technischen Anlagen und Fabriken u.a. mit Hilfe von EDV-Verfahren wie CAD und CAFM und deren Einführung und Realisierung:

- CAD - Einsatz bei der Planung und Bestandserfassung von Großbauwerken mit Expertensystemen für Hochbau, Anlagenbau, Haustechnik, Fabrikplanung, usw.

- CAFM - Computergestützte Gebäudebewirtschaftung zum effizienten Betreiben von Großbauwerken (Liegenschaftsverwaltung, Bau- und Anlagenunterhaltung, Instandhaltungsmanagement, Flächenmanagement, Reinigungsmanagement, Besiedelungsplanung, Umzugsmanagement usw.

Im Auftrag der IEZ AG lieferte GPO mbH Architekturmodelle, von denen für Test- und Präsentationszwecke Visualisierungen durch Ray-Tracing berechnet wurden. Die Visualisierungen enthielten Standbilder, Kamera-Animationen und interaktive Panoramas von in *Arcon*, bzw. *Speedikon W* modellierten Gebäuden.

4.3.2 Ergebnisse

4.3.2.1 Architekturmodelle

Im Laufe des Projektes hat GPO mbH zu Test- und Evaluierungszwecken mehrere Architektur-Modelle an die Universität Paderborn geliefert. Einige der Modelle wurden in Abschnitt 2.4.4.3 („Schritt 7: Berechnung der Bilder“) präsentiert.

4.3.2.2 Evaluierung

Während der Evaluierung des HiQoS Rendering-Systems wurden die folgenden Architektur-Modelle benutzt:

Szene	VRML Größe, Bytes	Texturen Größe, Bytes	Anzahl Texturen	Anzahl Objekte	Anzahl Dreiecken	Anzahl Lichtquellen	Anzahl Kameras
Kugel-Tisch	58778	630	1	14	1547	2	2

BWHAUS2	4247862	427234	67	8380	82748	35	4
Ferienhaus	6376545	392530	80	5677	121362	53	4
Gesamtansicht	1329639	66993	11	9063	22760	5	4
Erdgeschoss	4417871	315786	15	5828	69668	120	11
Verwaltung	5418081	42077	9	14391	74526	233	6

Alle Szenen sind in *Arcon* oder *speedikon* modelliert. Diese Szenen wurden in VRML 2 Format exportiert und auf einen Web-Server platziert. Jede Szene wurde mit verschiedener Beleuchtungs- und Qualitätseinstellungen an das HiQoS Rendering-System gesendet (siehe Abschnitt 2.4.4.4). Die berechneten Bilder sowie Statistiken vom HiQoS Rendering-System sind in [19] gesammelt.

4.3.3 Zusammenfassung

Der Anwender bedient sich bei der Visualisierung von Modellen verschiedener Softwareprodukte (z. B. *Arcon*) in denen Visualisierungsalgorithmen hinterlegt sind, die die Realität möglichst genau abbilden.

Die Visualisierung von Objekten ist ein komplexer Vorgang, der unter anderem von folgenden Faktoren abhängt

- Materialeigenschaften (Absorption, Reflexion usw.) aller umgebenden Körper
- direktes Licht
- indirektes Licht
- diffuses Licht
- Farbe des Lichts
- Größe des zu berechnenden Bildes

Da eine mathematische Darstellung dieser Beleuchtung unendlich komplex wird, versucht die Software mit Visualisierungsalgorithmen die Berechnung zu vereinfachen und trotzdem realistische Bilder zu erzeugen.

Das Problem bleibt jedoch, dass dieser Berechungsvorgang extrem kompliziert und damit langwierig ist, da

- jeder Lichtstrahl mit Reflexionen berechnet werden muss, das heißt je mehr Lichtquellen und je mehr Kanten das Objekt hat, desto länger dauert die Berechung
- der Speicherbedarf normaler PC für solche Berechnungen nicht ausreicht, und deshalb

Daten auf die Festplatte „zwischengelagert“ werden müssen. Diese Schreib- und Lesevorgänge kosten sehr viel Zeit

- Der Prozessor des PC nicht nur für diese Aufgabe reserviert ist, sondern auch andere Aufgaben erledigen muss. Es steht also nicht 100% der eher schon schwachen Prozessorleistung zur Verfügung sonder u.U. nur 70%

Bei Berechnungen, die mehrere Stunden oder gar Tage dauern, ist es schwer abzuschätzen, wann der Rechner noch arbeitet oder wann er abgestürzt ist. Es kommt vor, dass man einen Rechner mehrere Stunden oder gar Tage arbeiten lässt, jedoch kein Ergebnis erhält.

Vorhandene Methoden zur Visualisierung komplexer Szenen brauchen extrem große Rechnerleistungen. Kleinere Szenen und kleine Bilder lassen sich auf einem normalen PC berechnen. Steigt jedoch der Anspruch an die Qualität und Größe der Bilder oder möchtet man große, komplexe Szenen oder Kamerafahrten erstellen, muss man sich entsprechend leistungsfähige Hardware anschaffen, die nur für Visualisierungen benutzt wird.

Das HiQoS Rendering System adressiert das Problem der Visualisierung komplexen Szenen. Diese werden nicht auf einem PC, sondern auf Parallelrechnern der Universität Paderborn berechnet. Die Modelle, Texturen und Lichtquellen werden korrekt übertragen. Die Beleuchtung Nachts macht einen sehr realistischen Eindruck. Leider fehlen bei Tagbeleuchtung sämtliche Schatten, was dann im Bild sehr leblos aussieht. Das kann man aber umgehen, indem die vom Benutzer definierten Lichtquellen auch eingeschaltet werden.

Durch die vielfältigen Einstellmöglichkeiten in dem Web-Interface sind dem Anwender des Systems viele Möglichkeiten gegeben, das Modell gut auszuleuchten und gute Bilder als Ergebnis in sehr kürzer Zeit zu erhalten. Anmerken muss man aber, dass das System zur Zeit einen technisch versierten Anwender braucht, denn

- der VRML-Export der *Arcon* Szenen wurde in der neuen *speedikon Version (4.5)* nicht mit aktualisiert. Die Datei accvisu.dll führt unter der neuen Version zu Problemen, so dass zur Zeit nur die Version *speedikon 4.0* unterstützt wird

- das Upload der VRML Datei, Texturdateien und Kameradatei auf einen Webserver nicht Anwenderfreundlich genug ist. Dieser Vorgang ist für den Benutzer ziemlich komplex, und dadurch sehr fehleranfällig

- das direkte Eingeben der RGB Werte für Ambient Intensität, Flashlight Intensität und Hintergrundfarbe funktioniert, die Benutzer-Schnittstelle für das Farbmischen ist aber nicht zeitgemäß

4.4 IEZ AG

4.4.1 Einordnung in das Verbundprojekt

Die IEZ AG gehört mit den *speedikon* Produkten zu den führenden europäischen Softwarehäusern im Bereich Architektur. Vertretungen in 25 Ländern machen die *speedikon* Programme und die damit verbundenen (Service) Dienstleistungen weltweit verfügbar. Die IEZ AG

wurde von dem niederländischen Marktforschungsinstitut EFER (European Foundation for Entrepreneurship Research) 1993 zu einem der 500 dynamischsten Unternehmen Europas gewählt. Seit 1998 ist die IEZ AG ein Mitglied der mb Software AG.

Das technologische Grundkonzept der IEZ AG, „dreidimensional, ganzheitlich, modell- und objektorientiert", ist durch den rapiden Fortschritt der Hardwaretechnologie nicht mehr nur an hochwertige Workstationrechner gebunden, sondern kann heute auf jedem gut ausgestatteten PC problemlos installiert werden.

Im Rahmen des HiQoS-Projektes war die IEZ AG für die Software-Entwicklung bezüglich der Schnittstelle zwischen dem Client-Software *speedikon* bzw. *Arcon-Visuelle Architektur* und dem HiQoS Rendering-System zuständig. Ein Teil dieser Schnittstelle ist ein Konverter, der Architekturmodelle aus den *speedikon* bzw. *Arcon-Visuelle Architektur* Programmen in das im Lauf des Projektes spezifizierte erweiterte VRML 2.0 Format exportiert. Der zweite Teil der Schnittstelle ist ein Programm, das die Kamera-Beschreibung von *speedikon* (oder *Arcon-Visuelle Architektur*) in das im Lauf des Projektes spezifizierte HiQoS Kamera-Format exportiert.

4.4.2 Ergebnisse

4.4.2.1 Arcon VRML 2 Export

Arcon und *speedikon* Architekturmodellierungsprogramme verwenden ein eigenes binäres Format (ACP) um 3D-Modelle zu speichern. Dieses Format ist nicht öffentlich und wird mit *Arcon*, bzw. *speedikon* stark gekoppelt. Um 3D-Modelle zwischen dem Benutzer und HiQoS Rendering-System austauschen zu können, wurde in Kooperation mit der Universität Paderborn ein erweitertes VRML 2.0 Format spezifiziert (diese Erweiterung ist in Abschnitt 2.2.6.1 ausführlich beschrieben). IEZ AG hat der Universität Paderborn eine Lizenz von *Arcon* sowie *speedikon* mit einem integrierten VRML 2 Export-Modul zu Verfügung gestellt. Die für die Visualisierung relevanten Daten werden fast 1:1 von Arcon exportiert und von dem Konverter und parallelen Ray-Tracer des HiQoS Rendering-Systems korrekt interpretiert.

4.4.2.2 Arcon Kamera Export

Beschreibung von Kameras (Perspektiven) wird aus praktischen Gründen von der restlichen Beschreibung des Modells getrennt. Von der Universität Paderborn wurde Kamera-Format spezifiziert, in dem einzelne Kameras, sowie Kamera-Pfade gespeichert werden können (siehe Abschnitt 2.2.6.2.1). Von der IEZ AG wurde ein eigenständiges Programm implementiert, das einzelne *Arcon*-Kameras in HiQoS Kamera-Format exportiert. Um Kamera-Pfade erzeugen zu können, wurde eine bessere Alternative gesucht. In *Arcon* hat der Benutzer die Möglichkeit, Kamera-Pfade interaktiv zu definieren (Walkthrough). Die vom Benutzer definierten Kamera-Pfade werden in einem binären WLK Format gespeichert. Von der Universität Paderborn wurde ein Konverter implementiert, der die binären WLK Dateien des *Arcon* Video-Moduls in HiQoS Kamera-Format übersetzt.

4.4.3 Zusammenfassung

Als das 3D-Austauschsformat in dem HiQoS Architektur-Visualisierungsszenario wird VRML 2.0 verwendet. In der Kooperation mit der Universität Paderborn wurde eine Erweiterung des VRML 2.0 Formats definiert um *speedikon* (bzw. *Arcon*) Szenen ohne Informationsverlust speichern zu können. Von der IEZ AG wurde einen Exporter geschrieben, der *speedikon* Szenen in dem vereinbarten Format speichert. Alle notwendige Geometrie-, Material- und Lichtquellen-Eigenschaften werde korrekt exportiert und vom HiQoS Rendering-System korrekt interpretiert. Es wurde fast 100% Kompatibilität zwischen dem *Arcon* Ray-Tracer und dem parallelen Ray-Tracer der Universität Paderborn erreicht.

Ein Programm zur Definition einzelner Kameras wurde entwickelt. Das Programm ermöglicht dem *speedikon* (bzw. *Arcon*) Benutzer einzelne Kameras für das HiQoS Rendering System zu definieren. In diesem Programm kann der Benutzer auch Kamera-Pfade für Walkthroughs definieren, dieser Vorgang ist allerdings nicht intuitiv. Um dem Benutzer mehr Komfort bei der Definition von Kamera-Pfaden zu geben, wird das interne *Arcon*s „Video-Modul“ benutzt und die resultierende .WLK Datei automatisch (von einem externen Konverter der Universität Paderborn) in das HiQoS Kamera-Format konvertiert. Die Beschreibung des binären .WLK Formats wurde der Universität Paderborn zu Verfügung gestellt.

Das Gesamtszenario wurde auf mehreren Szenen erfolgreich getestet und evaluiert.

4.5 Pixelpark AG

4.5.1 Einordnung in das Verbundprojekt

Im Rahmen des HiQoS-Verbundprojektes hatte die Pixelpark AG die Aufgabe, die gelieferte Technologie in entsprechenden Trails zu testen. Bei der von Pixelpark ausgewählten *Pixelvision* handelt es sich um die digitale Bereitstellung interner Veranstaltungen für alle Mit-arbeiter. Um einen kontinuierlichen Fluss der Informationen (Audio-, Video-, Text- also Metadaten) zu gewährleisten wurden entsprechende QoS-Elemente (RSVP, RTSP etc.) eingesetzt.

Im Rahmen eines Unterauftrags beauftragte Pixelpark die Firma UPSTART! (vgl. Abschnitt 4.10) mit der Generierung digitaler Inhalte und deren Berechnung im Bereich Rendering-Server. So kamen bei UPSTART! alle HiQoS-Elemente im Bereich der digitalen Film Produktion zum Einsatz.

4.5.2 Ergebnisse

Die Pixelvision wurde in das HiQoS-System überführt. Innerhalb *des Pixelpark Corporate Networks* wurde ein System verteilter Multimediaserver (Videopumpen und Access Server bzw. Applikation Server) installiert. Integriert wurden auch die dazugehörigen HiQoS Clients unter Windows2000 Professional.

Die während einer Pixelvision aufgenommenen Inhalte wurden nach Ihrer Digitalisierung mittels des *Access Managements* (siehe Abschnitt 2.3.5) zusammen mit allen zeitbasierenden Informationen wie Präsentationen etc. in das System integriert.

Anschließend wurden die Zugriffe der Clients auf entsprechende Inhalte sowohl mit als auch ohne QoS Funktionalität untersucht.

Das System bestand in der Anlaufphase aus einem separaten Netzwerk mit wenigen Clients. Im weiteren Verlauf wurde die Anzahl der Clients erhöht. Ein weiterer Punkt war die Integration des HiQoS-Systems in das inzwischen neu aufgebaute Netzwerk und die Überführung in ein VLAN zur Vermeidung von internen Belastungen durch große Videodatenströme.

Zum Ende des Projektes wurden funktionsfähige Prototypen eingesetzt.

Die Details sind Abschnitt 2.4.2 zu entnehmen.

4.5.3 Zusammenfassung

Es konnte gezeigt werden, dass die HiQoS-Technologie im Wesentlichen die Anforderungen der Pixelvision erfüllen konnte. Dennoch wurden einige Probleme erkannt:

- Unterschiedliche Hardware bei den ISP-Internetstrecken werfen große Probleme auf
- Strenge Firewall-Regeln verhindern unter Umständen den Einsatz der QoS-Funktionalität
- Eine Macintosh-Integration der Clients (MacOS X) wäre wünschenswert
- QuickTime-5-Funktionalität wäre wünschenswert
- Das Zusammenspiel der einzelnen Komponenten gestaltete sich recht komplex

4.6 Siemens CT

4.6.1 Einordnung in das Verbundprojekt

Die von Siemens CT gelieferten Komponenten dienten dazu, es den Projektpartner zu ermöglichen, entsprechende multimediale Informationssysteme aufzubauen. Diese Aufgabenstellungen wurden im Rahmen eines Unterauftrags von Siemens SBS C-LAB durchgeführt.

Ausgehend von der Anforderungsanalyse an Netzwerke von Mediaservern wurde von Siemens CT ein Mediaserver entwickelt, der es basierend auf Internet-Technologie ermöglicht multimediale Anwendungen mit Quality-of-Service-Garantien zu realisieren.

Das in der ersten Projektphase realisierte System zur Übertragung von multimedialen Daten mit Quality-of-Service-Parametern ist auf Netzbereiche begrenzt, die eine entsprechende Unterstützung in allen Netzelementen (z.B. Routern) bieten, deshalb wurde in den folgenden Projektphasen ein System entwickelt, das diese Anforderungen nicht über das gesamte Netz hinweg stellt. Als Lösungsansatz wurde die automatische Verteilung der Inhalte gewählt.

Siemens CT entwickelte für dieses System einen Redirector der eine automatische Umleitung der Benutzer-Zugriffe auf den dem Anwender am nächsten gelegenen Cache Server ermöglicht.

Informationen vor dem unberechtigtem Zugriff zu schützen wird in heutigen Systemen immer wichtiger. Dies liegt darin begründet das entweder eine Bezahlung für den Abruf der Informationen gewünscht wird oder aber der Nutzergreis begrenzt werden soll. Siemens CT entwickelte ein Access Management System das die Zugriffskontrolle für verteilte HiQoS Informationsdienste ermöglicht.

4.6.2 Ergebnisse

4.6.2.1 Medien-Server mit QoS Support

Das Entwicklungsergebnis ist ausführlich in Abschnitt 2.3.1 beschrieben.

4.6.2.2 Redirect Server

Benutzerzugriffe auf global bereit gestellte Media Assets müssen in verteilten Multimedia Server System auf lokale Multimedia Server umgelenkt werden. Dabei sind die Wünsche des Benutzers nach Qualität der Darstellung und die Möglichkeiten des Netzwerkes (z.B. Bandbreite) zu berücksichtigen. Das Umleiten der Benutzerzugriffe erfolgt für die Anwendung transparent und konform zu Standard-Protokollen.

Zur Bereitstellung der prinzipiellen Funktionalität wurde von Siemens CT eine Redirector-Komponente entwickelt, die bei jedem Zugriff auf ein globales Media Asset, eine Anfrage an das zentrale Content Management System (DSMS) stellt. Der Benutzer wendet sich normalerweise an den Video Server über das RTSP-Protokoll. Hier setzt die von Siemens CT entwickelte Komponente an. Sie ‚spricht' auf der Benutzer Seite das RTSP-Protokoll und fragt zunächst über eine CORBA-Schnittstelle beim DSMS nach, auf welchen Servern der Inhalt vorhanden ist. Dann leitet der Redirector den Client zu dem am besten geeigneten Server um.

4.6.2.3 Access Management

Das Entwicklungsergebnis ist ausführlich in Abschnitt 2.3.5 beschrieben.

4.6.3 Zusammenfassung

Die angestrebten Komponenten-Entwicklungen konnten alle erfolgreich abgeschlossen werden. Diese werden heute von Siemens genutzt, um Anwendungen in zukünftigen QoS-basierten mobilen Netzen zu demonstrieren.

Innerhalb der Projektlaufzeit zeigte sich, dass die Integration von Systemen mit einem solchen Komplexitätsgrad sich innerhalb des Projektes als schwierig erwiesen. Einerseits stellt die Netzinfrastruktur eine nicht zu unterschätzendes Problem dar, was dazu führt das bis zu einer flächendeckenden Einführung einer neuen Technologie (z.B. QoS-Unterstützung) oft mehrere Jahre vergehen. Andererseits erfordert die Erstellung und Verwaltung von multimedialen Inhalten und darauf beruhender Informationsdienste einen Aufwand, der von den meisten Anwendern unterschätzt wird. Die Problematik zukünftiger Systeme wird also vorrangig darin liegen, eine effiziente Integration der Systeme zu ermöglichen und die Schritte zur Erstellung und Verwaltung multimedialer Informationsdienste zu vereinfachen.

4.7 Siemens SBS C-LAB

4.7.1 Einordnung in das Verbundprojekt

Neben der Projekt-Koordination (siehe Abschnitt 3) lagen die technischen Aufgaben von Siemens SBS C-LAB vor allem im Bereich QoS, wobei ein spezielles Test-Netzwerk zur experimentellen Erprobung der HiQoS-Technologie zur Verfügung gestellt worden ist, so dass dem Konsortium besonders bei der Integration der einzelnen RSVP-Komponenten geholfen werden konnte. Wissenschaftliche Untersuchungen wurden auf den Gebiet der Kombination von klassischen Methoden der Realzeitverarbeitung mit denen des QoS durchgeführt (siehe Abschnitt 2.3.8).

Darüber hinaus war Siemens SBS C-LAB verantwortlich für den Ablauf und die technische Unterstützung (durch SMIL (siehe Abschnitt 2.3.7)) der ViLM-Anwendung (siehe Abschnitt 2.4.1), die in Kooperation mit *der AG Didaktik der Informatik* an der Universität Paderborn durchgeführt wurde und wird. Ein weiterer Anwendungsprototyp wurde mit dem *Pathfinder* (siehe Abschnitt 0) für das Live-Streaming-Szenario gestellt. Schließlich wurde ein Konzept für ein HiQoS-Open-Source-Projekt erarbeitet (siehe Abschnitt 2.3.6).

Darüber hinaus hat Siemens SBS C-LAB weitere Aufgabenstellungen von den Projektpartner Siemens CT (siehe Abschnitt 4.6) und PC^2 (siehe Abschnitt 4.6) durch entsprechende Unteraufträge unterstützt.

4.7.2 Ergebnisse

4.7.2.1 Projekt-Management

Siemens SBS C-LAB war verantwortlicher Projektkoodinator. Einzelheiten dazu sind in Abschnitt 3 zu finden.

4.7.2.2 Realzeit-Scheduling

Das Ergebnis dieser Tätigkeit ist ausführlich in Abschnitt 2.3.8 beschrieben.

4.7.2.3 SMIL und ViLM

Zur Unterstützung der ViLM-Anwendung, für die Siemens SBS C-LAB verantwortlich gewesen ist (siehe Abschnitt 2.4.1), war besondere SMIL-Unterstützung (siehe Abschnitt 2.3.7)) notwendig. Beides ist ausführlich in den entsprechenden Abschnitten beschrieben.

4.7.2.4 Open Source

Das Entwicklungsergebnis ist ausführlich in Abschnitt 2.3.6 beschrieben.

4.7.2.5 Live-Streaming-Demonstrator

Das Entwicklungsergebnis ist ausführlich in Abschnitt 2.4.6 beschrieben.

4.7.3 Zusammenfassung

Die von Siemens SBS C-LAB angestrebten inhaltlichen Ziele konnten überwiegend erfolgreich abgeschlossen werden. Diese werden in Zukunft für weitere wissenschaftliche Zwecke fortentwickelt werden. Bei einzelnen Themen (wie der ViLM-Anwendung (siehe Abschnitt 2.4.1) ist auch ein kommerzielle Verwertung in der Diskussion.

4.8 Universität Paderborn - PC^2

4.8.1 Einordnung in das Verbundprojekt

Die an der Universität Paderborn - PC² bearbeiteten Aufgabenstellungen wurden im Rahmen eines Unterauftrags von Siemens SBS C-LAB durchgeführt. Der Schwerpunkt der Arbeiten lag auf dem Management von Server-Netzwerken, der Plazierung von Medienobjekten und der Realisierung von Mechanismen für Advance Reservation auf Netzwerkebene. Diese Arbeiten wurden aufbauend auf der Implementierung des Medienservers durch Siemens CT (s. Abschnitt 2.3.1) durchgeführt. Die entwickelte Software zum Server Management wurde an die in HiQoS entwickelten Server und Clients angepasst. Die Unterstützung für QoS auf Netzwerkebene, die in die HiQoS Server und Clients implementiert wurde, ist eine notwendige Voraussetzung für die vorgesehenen Advance-Reservation-Mechanismen (s. Abschnitt 2.3.3). Die Software DSMS muss zusammen mit dem in HiQoS entwickelten Access Management System eingesetzt werden, um unberechtigte Zugriffe auf die im Servernetzwerk gespeicherten Inhalte zu verhindern.

4.8.2 Ergebnisse

4.8.2.1 Distributed Server Management System

Die Umgebung für die zu entwickelnden Algorithmen und Verfahren stellt die für HiQoS implementierte Management Software *Distributed Server Management System (DSMS)* (s. Abschnitt 36) dar. Diese für die Administration von Servern (auch verteilten Servern) entwickelte Software erlaubt die Verwaltung großer Servernetzwerke und die darauf gespeicherten Medieninhalte. Wegen hohen Bandbreite der zu übertragenden Medienobjekte ist für einen Client in einer solchen Umgebung kein Streaming von einem entfernten (d.h. nicht dem lokalen Server innerhalb der Breitbandinsel) möglich. Der Zugriff auf lokal nicht vorhandene Medienobjekte kann daher nur zeitversetzt vom lokalen Server erfolgen, nach das gewünschte Objekt dorthin kopiert wurde.

Das DSMS verwaltet die Videos auf allen Videoservern im Netzwerk. Die Videos werden über ein Administration Interface dem DSMS bekannt gemacht und dann auf Servern im Netzwerk abgelegt. Bei der Anforderung eines Videos von einem Client werden alle Kopien dieses Videos im Netzwerk lokalisiert und dem Client zusammen mit dem Datum der Verfügbarkeit am nächstliegenden Server dem Client zur Verfügung gestellt. Bei Anforderung eines Videos von einem anderen, nicht lokalen Server wird das Video vom DSMS auf den lokalen Server kopiert. Die dazu notwendigen Berechnungen (Routing, Start und geschätzte Dauer des Kopiervorgangs) werden im DSMS vorgenommen.

Um Absicherung der im Servernetzwerk gespeicherten Inhalte zu gewährleisten, wird zusätzlich zum DSMS das Access Management System (s. Abschnitt 2.3.5) benötigt. Die Kommunikation zwischen beiden Programmen wird über CORBA-Schnittstellen realisiert.

Für die Einbettung des DSMS in die verteilte 'Pixelvision' Anwendung (s. Abschnitt 2.4.2) wurden CGI-Skripten entwickelt, die aus der Anwendung heraus aufgerufen werden, um das Aufspielen und Löschen von Medienobjekten zu ermöglichen. Diese Skripte greifen über die CORBA-Schnittstellen auf die entsprechenden DSMS Module zu.

4.8.2.2 Algorithmen zum Content Management

Die zentrale Funktionalität des DSMS ist die Möglichkeit, Medienobjekte automatisch auf die Server des Netzwerkes in einer Weise zu verteilen, dass Nutzer des Systems die Medienobjekte, die für diese von großen Interesse sind, möglichst ohne Verzögerung zur Verfügung haben. Das bedeutet, dass Kopiervorgänge zwischen Servern möglichst zu vermeiden sind, insbesondere da diese eventuell nur zu bestimmten Zeiten erlaubt sind, um das Netzwerk nicht zu überlasten (s. Abschnitt 2.3.2).

4.8.2.3 Advance Reservation

Das in Abschnitt 2.3.3 beschriebenen Verfahren zur Reservierung von Bandbreite in voraus für Streaming zwischen Clients und Servern bzw. für Dateiübertragungen zwischen Servern wurden am PC² entwickelt und in das DSMS integriert. Da die Reservierung von Bandweiten im Internet heutzutage nicht üblich ist, musste auf die konkrete Implementierung und Evaluierung verzichtet werden. Allerdings sind alle in Abschnitt 2.3.3 beschriebenen Mechanismen im DSMS integriert und müssen nur aktiviert werden.

4.8.3 Zusammenfassung

Für die umfangreichen Aufgaben des Content Management innerhalb großer, skalierbarer Netzwerke von Medienservern wurde die Management Software DSMS entwickelt.

Die zentrale Funktionalität des DSMS ist die automatische Verteilung von Medienobjekten innerhalb des Servernetzwerks. Hierzu wurden Algorithmen entwickelt, mit deren Hilfe eine möglichst optimale Verteilung der Daten auf den zur Verfügung stehenden Server sowie die jeweils optimale Qualität (Bitrate) berechnet wird. Wegen der Komplexität der Problemstellung wurden hierzu Heuristiken verwendet. Es konnte durch Simulationen verschiedener großer Servernetzwerke ermittelt werden, dass die Resultate der Heuristiken nur unwesentlich von der optimalen Lösung abweichen und daher erfolgreich in realen Umgebungen eingesetzt werden können.

Abgesehen von der automatischen Verteilung der Medienobjekte stellt das DSMS alle weiteren zur Verwaltung des Servernetzwerkes benötigten Funktionen zur Verfügung. Dazu gehören Datenbanken zur Speicherung der Netzwerktopologie und der Speicherorte der einzelnen Medienobjekte, Schnittstellen zum Abonnement von Content-Kategorien und die Administration, d.h. das Aufspielen und Löschen von Medienobjekten, das unabhängig an jedem einzelnen Server möglich ist.

4.9 Universität Paderborn - AG Monien

4.9.1 Einordnung in das Verbundprojekt

Die Arbeitsgruppe von Prof. Monien im Fachbereich Mathematik/Informatik an der Universität-GH Paderborn arbeitet seit mehr als 10 Jahren im Bereich des parallelen und verteilten Rechnens. In diesem Arbeitsgebiet hat die Arbeitsgruppe eine Vielzahl von Projekten durchgeführt und ist auch zur Zeit an mehreren nationalen und internationalen Projekten beteiligt.

Die Aufgaben der AG Monien innerhalb von HiQoS konzentrierten sich auf die Forschungs- und Entwicklungsarbeiten in Aufträgen der Axcent Media AG und der Pixelpark AG. Im Auftrag der Pixelpark AG war die AG Monien für die Entwicklung und Integration von datenparallelen Radiosity-Algorithmen zuständig. Im Auftrag der Axcent Media AG war die AG Monien für die Entwicklung und Integration des parallelen Ray-Tracings, sowie für die Spezifikation der Gesamtarchitektur des HiQoS Rendering-Systems zuständig.

4.9.2 Ergebnisse

4.9.2.1 Paralleles Ray-Tracing

Ein daten-paralleler Ray-Tracer wurde implementiert und an das Beleuchtungsmodell *von Arcon* Architekturvisualisierungsprogramm angepasst (siehe Abschnitt 2.2.3).

4.9.2.2 Paralleles Radiosity

Daten-paralleles Radiosity wurde implementiert und an ein 3DS-Szenario angepasst (siehe Abschnitt 2.2.4).

4.9.2.3 Vereinfachung von Radiosity-Lösungen

Zwei Methoden zur Vereinfachung von Radiosity-Lösungen wurden implementiert (sequentielle sowie parallele Versionen): Mesh Decimation und Radiosity Textursynthese. Die Methode der Textur-Synthese bietet eine verlustfreie Kompression an und aus diesem Grund wurde sie im HiQoS Rendering-System integriert (siehe Abschnitt 2.2.4.2).

4.9.2.4 Datenkonversion

Ein VRML 2 Import-Konverter und ein 3DS Import- und Export-Konverter wurden implementiert um 3D-Modelle von *Arcon* und 3DS Max bearbeiten zu können sowie Import-Konverter von HiQoS Kamera-Format und *Arcon* WLK Format (siehe Abschnitt 2.2.6).

4.9.2.5 Integration des HiQoS Rendering-Systems

Die zweistufige Architektur des HiQoS Rendering-System wurde vorgeschlagen, in Kooperation mit der Firma Axcent Media AG verfeinert und die Schnittstellen zwischen einzelnen Software-Komponenten wurden definiert (siehe Abschnitt 2.2.5). Die Architektur ermöglicht dem Benutzer einen transparenten Zugriff zu hochperformanten Rechnern über ein Web-

Interface um eine globale Beleuchtung von 3D-Szenen zu berechnen. Das System kümmert sich um eine effiziente Ausnutzung der verfügbaren Ressourcen. [12,7]

Die Integrationsarbeiten in der Universität Paderborn (AG Monien) konzentrierten sich auf die Spezifikation der Parameter, die dem Benutzer durch das Web-Interface des Systems angeboten werden, auf die Implementierung der Daten-Konverter, die vom globalen Scheduler angerufen werden, auf die Implementierung des parallelen Rendering-Servers und auf die Portierung des Rendering-Servers an die verfügbaren parallelen Systeme. In Zusammenarbeit mit Firma Axcent Media AG wurde ein Prototyp des Rendering-Systems integriert und in zwei Anwenderszenarien (das Ray-Tracing-Szenario von den Firmen IEZ AG und GPO mbH im Kontext von der Architektur-Visualisierung, das Radiosity-Szenario von der Firma Upstart! Filmproduktion GmbH im Kontext von einer globalen Beleuchtung für Filmproduktion) getestet und evaluiert.

4.9.3 Zusammenfassung

Im Rahmen des HiQoS-Projektes wurden von der Universität Paderborn (AG Monien) die folgenden Ergebnisse erreicht:

- *Ein daten-paralleler Ray-Tracer* wurde entwickelt, in dem die Bildschirm-Verteilung mit dem Objektraum-Verteilung kombiniert ist. Diese kombinierte Lösung ermöglicht auch sehr große Szenen zu berechnen (Szenen, bei welchen der Speicher eines Prozessors nicht ausreicht, d.h. bei welchen eine sequenzielle Berechnung nicht möglich ist). Diese Lösung ist sehr flexibel – die Objektraum-Verteilung kann jederzeit manuell ausgeschaltet werden, kann gut mit sequentiellen Beschleunigungstechniken kombiniert werden und ist voll-automatisch.

- *Eine Erweiterung des VRML2 Formats* wurde spezifiziert und ein *VRML2 Import-Konverter wurde implementiert. Ein binäres POV-Format* wurde spezifiziert um den Speicherplatz für die konvertierten Szenen zu reduzieren und Lese-Zeiten zu verkürzen. (Die Lese-Routinen wurden dementsprechend im parallelen Ray-Tracer implementiert.)

- *Das Beleuchtungsmodel des daten-parallelen Ray-Tracers wurde an das Beleuchtungsmodel vom Programm Arcon-Visuelle Architektur von mb-Software angepasst,* wobei eine fast volle Kompatibilität mit *Arcon* erreicht wurde.

- *Ein 3DS Input-Konverter als auch 3DS Export-Konverter* wurden implementiert. In Kooperation mit der Firma Upstart! Filmproduktion GmbH wurde eine Lösung für Export von Vertex-Radiosities im 3DS Export implementiert (von Upstart! wurde ein 3DS-Plugin für Import der Vertex-Radiosities implementiert).

- *Neue Lichtquellen-Typen* (Flash-Light, Directional-Light) wurden *im parallelen Ray-Tracer* implementiert. Neue Lichtquellen-Typen (Point-Light, Spot-Light) wurden *im parallelen Radiosity* implementiert.

- *Ein semi-automatisches Programm zur Reduktion der Anzahl von Lichtquellen* wurde implementiert, weil einige der GPO Testszenen mit mehreren Hunderten zueinander

nahliegender Punktlichtquellen modelliert wurden, was auch bei einer parallelen Ray-Tracing Berechnung mit vielen Prozessoren zu unakzeptierbar hohen Berechnungszeiten fuhr.

- *Ein Konverter vom Arcons WLK-Format* (in dem in *Arcon* erzeugte Kamera-Wege gespeichert werden) wurde implementiert.

- *Neue Lastbalancierungsverfahren für daten-paralleles Radiosity, sowie eine neue Methode zur Berechnung der Radiosity Form-Faktoren* wurden entwickelt.

- *Zwei Methoden zur Optimierung von Radiosity-Lösungen* wurden entwickelt und als sequentielle sowie parallele Programme implementiert: die Mesh Decimation Methode und die Methode der Radiosity-Textursynthese. Damit wurde die Netzwerk-Übertragung und Nachbearbeitung der durch die Radiosity-Methode diffuse-beleuchteten Szenen ermöglicht.

- Beide *daten-paralleles Ray-Tracing und daten-paralleles Radiosity*, sowie weitere notwendige Software-Komponenten wurden *auf das Ziel-System Siemens-hpcLine in PC^2 portiert.*

- In Kooperation mit der Firma Axcent Media AG wurde die *Software-Architektur des HiQoS Rendering-Systems spezifiziert.*

- Das daten-parallele Ray-Tracing und daten-parallele Radiosity wurden im *Rendering-Server* integriert und die Anbindung an das von der Firma Axcent entwickelte Scheduling-System spezifiziert und implementiert. Das HiQoS Rendering-System wurde lokal in der Universität Paderborn getestet und später in zwei Anwenderszenarien evaluiert.

- Viele *Standbilder, Animationssequenzen und Radiosity-Lösungen* wurden auf dem parallelen Rechner *hpcLine* in PC^2 berechnet. Von dem ausgewählten Material wurde von der Firma Upstart! Filmproduktion GmbH eine Videosequenz geschnitten und in der Universität Paderborn wurde die finale Version des Films komplettiert (und von APE Ptacek Engineering GmbH digitalisiert).

- *Web-Seiten* des Teilprojekts Rendering mit einer Demonstration der Ergebnisse durch Bilder, QTVR Panoramas, Animationssequenzen und erklärenden Texten wurden geschrieben und veröffentlicht (http://www.c-lab.de/hiqos).

Einige der obenbeschriebenen Technologie-Entwicklung- und Forschungsergebnisse wurden in der Form von Web-Seiten, mehreren Konferenz-Publikationen, drei Diplomarbeiten und einer Dissertation veröffentlicht.

4.10 UPSTART!

4.10.1 Einordnung in das Verbundprojekt

Die UPSTART! Filmproduktion GmbH war innerhalb des HiQoS Projekts als Kooperationspartner der Pixelpark AG im Bereich Rendering tätig.

Die Hauptaufgabe von UPSTART! bei HiQoS war die Evaluierung des Rendering-Systems, also die Bereitstellung von praxisnahen 3D Szenerien, das Auffinden von Berechungsfehlern, die Mitarbeit bei der Spezifikation und Auswahl der Datenformate und Schnittstellen sowie die Integration der Rendering-Server in ein realistisches Produktionsumfeld.

4.10.2 Ergebnisse

Neben den Ergebnissen, die in Zusammenarbeit mit der Universität Paderborn und der Firma Axcent entstanden sind (siehe 4.9.3), wurden folgende Ergebnisse erreicht:

- Spezifikation und Design des Webinterfaces der Clientanwendung. Dieses für den Benutzer transparente Interface, wurde in seiner Grundversion von UPSTART! spezifiziert und gestaltet. Es wurde im Verlauf von HiQoS von Axcent und der Universität Paderborn erweitert und an den aktuellen Entwicklungsstand angepasst.

- Spezifikation und Implementierung der Schnittstellen zwischen Renderingserver und Anwendungsprogramm (3D Studio Max).

    - Dies umfasst die Wahl des Datenformates (3DS), die Spezifikation eines Zusatzformates in welchem die Farbinformationen der Punkte der Radiosity Solution (Vertex Colors) gespeichert sind und die Implementierung von Scripts und Plugins in 3DS Max, um dies Format lesen zu können. Die Ergebnisse wurden im Rahmen von HiQoS als Open Source der Allgemeinheit zur Verfügung gestellt.

    - Durch die Verwendung des 3DS Formates, konnten nicht nur die Ladezeiten drastisch verkürzt werden (je nach Szene Faktor 10–500), auch beim gesamten Datenfluss konnten die Wege vereinfacht werden.

    - Viele Fehler des Rendering-Servers wurden während der Evaluierung von UPSTART! erkannt und durch intensive Kommunikation mit der Universität Paderborn durch diese beseitigt sowie die Algorithmen des Rendering-Server optimiert werden.

4.10.3 Zusammenfassung

Während der Projektlaufzeit von HiQoS wurden viele 3D Szenen an der Universität Paderborn berechnet. Die anfänglichen Schwierigkeiten (Datenaustausch, Inkompatibilitäten) wurden im Laufe der Zeit überwunden. Es kam, sozusagen als Nebeneffekt, auch zu einer kommerziellen Zusammenarbeit im Bereich des herkömmlichen Scanline-Renderings zwischen der Universität Paderborn und UPSTART!.

Die Qualität der HiQoS Radiosity-Berechnung ist mit den Ergebnissen von kommerzieller Software wie Lightscape zu vergleichen, auch wenn bei der reibungslosen Integration in ein professionelles Produktionsumfeld noch Wünsche offen sind. Kommerzielle Software rechnet zwar langsamer, doch der Datenaustausch funktioniert präziser und zuverlässiger.

Der entwickelte HiQoS Rendering-Service hat in seinem jetzigen Stadium ein frühes Beta-Stadium erreicht. Dies liegt im wesentlichen an den nicht veröffentlichten Dateiformaten kommerzieller Software wie 3D Studio Max und an der Natur des Radiosity-Berechnungs-

verfahrens, bei dem der Anwender viele wesentliche Parameter zu berücksichtigen hat, die bei jeder Szene unterschiedlich gewichtet sein können. Es gibt ferner keine normierte Implementierung der Radiosity-Berechnung. Jede Implementierung, auch die kommerzieller Software, hat ihre speziellen Eigenheiten.

Intelligente Ratgeber, die auf dem Erfahrungsschatz des HiQoS Rendering-Systems aufbauen könnten diese Problematik in Zukunft, wenn nicht ganz verhindern, aber doch stark verringern und somit das HiQoS Rendering-System für einen größeren Benutzerkreis zugänglich machen.

5 Zusammenfassung

Basierend auf IP-Technologie ist im Rahmen von HiQoS die Realisierung von Plattformen zur Entwicklung hochperformanter netzbasierter Multimedia-Dienste, die prototypische Entwicklung dieser Dienste, sowie deren Evaluierung in konkreten industriellen Anwendungen betrieben worden.

Die Dienste zeichnen sich vor allem dadurch aus, dass sie hochqualitative kontinuierliche Medien integrieren, photorealistische Animationen erlauben und auf standardisierten Internet-Technologien aufbauen. Zur Realisierung dieser Dienstleistungen wurden in HiQoS insbesondere die zwei folgenden technischen Ansätze vertieft:

1. das Rendering von photorealistischen Animationen auf Parallelrechnern,
2. die Einhaltung von "Quality-of-Service-(QoS)"-Garantien in einem Netzwerk von Multimedia-Servern.

Beim Rendering ist in HiQoS ein Ansatz gewählt worden, welcher durch parallele Datenverarbeitung komplexe Bildern und Animationen in photorealistischer Qualität liefert. Im Projekt wurden dabei einzelne Verfahren (Raytracing und Radiosity) gezielt fortentwickelt, Format-Konvertierungen und Erweiterungen zur Verfügung gestellt und ein transparenter IP-Zugang zu parallelem High Performance Rendering geschaffen.

Zur Bereitstellung von QoS-Technologie ist in HiQoS auf die Integration von modernen Übertragungsstandards, wie z.B. RSVP (Resource Reservation Protocol), für eine garantiert ruckelfreie Übertragung von Medien-Strömen eingegangen worden. Dies umfasst alle beteiligten, z.T. selbst implementierten Netzwerk-Elemente. Desweiteren konnte dabei gezeigt werden, dass Methoden aus der klassischen Realzeitverarbeitung hervorragend dazu geeignet sind, stark variable Medienströme effizient zu übertragen. Neben dem Bereich des QoS ist mit Content und Access Management eine Infrastruktur für Verwaltung von Video-on-Demand-(VoD)-Applikationen geschaffen worden.

In den beiden Themen-Schwerpunkten des Projekts sind Anwendungen mit eigenen Ambitionen gestaltet worden, die die Anwendbarkeit der HiQoS-Technologie verdeutlichen.

6 Literatur

[1] Lars-Olof Burchard, Reinhard Lüling: Management of Broadband Media Assets on Wide Area Networks. 5th Int. Web Caching and Content Delivery Workshop, , Lisbon, Portugal, May 2000

[2] Lars-Olof Burchard, Reinhard Lüling: An Architecture for a Scalable Video-on-Demand Server Network with Quality-of-Service Guarantees. Proc. 7th International Workshop on Interactive Distributed Multimedia Systems and Telecommunication Services, Springer, LNCS, pp. 132-143, 2000.

[3] Lars-Olof Burchard, Reinhard Lüling: Management of Media Assets in Large Scale Server Networks. International Conference on Internet and Multimedia Systems and Applications (IMSA). Las Vegas, Nevada, USA. 2000.

[4] Reinhard Lüling, Xiaobo Zhou:. Dynamic Adaptive Mapping of Videos in a Hierarchical TV-Anytime Server Network. Proc. of the 9th International Conference on Computer Communication and Networks, Las Vegas, USA, 2000.

[5] Xiaobo Zhou, Reinhard Lüling: Heuristic solutions for a mapping problem in a hierarchical TV-anytime server network. International Parallel and Distributed Processing Symposium (IPDPS), 2000.

[6] Xiaobo Zhou, Lars-Olof Burchard, Reinhard Lüling: A Video Replacement Policy based on Revenue to Cost Ratio in a Multicast TV-Anytime System. Workshop on Parallel and Distributed Computing in Image Processing, Video Processing, and Multimedia (PDIVM). San Francisco, USA, 2001.

[7] T. Plachetka, O. Schmidt, F. Albracht: The HiQoS Rendering System, (to appear in) Proc. of the 28th Annual Conference on Current Trends in Theory and Practice of Informatics (SOFSEM 2001), Piestany, Slovakia, Lecture Notes in Computer Science, Springer-Verlag, 2001

[8] F. Albracht: Radiosity-Texturen: Eine sequentielle und parallele Methode zur verlustfreien Vereinfachung von Radiosity-Polygonnetzen, Diplomarbeit, Universität Paderborn, 2001

[9] Computergrafik: Parallele Photorealistische Bildgenerierung als Dienstleistung in e-Commerce Szenarios, HNI Jahresbericht 2000, Heinz Nixdorf Institut, Universität Paderborn, 2001, 67-67

[10] R. Lüling, B. Monien, T. Plachetka, O. Schmidt: HiQoS: High Performance Multimedia Services with Quality of Service Guarantees, PC^2 Annual Report 2000, Paderborn Center for Parallel Computing, Universität Paderborn, 2001

[11] O. Schmidt: Parallele Simulation der globalen Beleuchtung in komplexen Archi-

tekturmodellen, Dissertation, Fachbereich Mathematik/Informatik, Universität Paderborn, 2000

[12] O. Schmidt, T. Plachetka: High Performance Computing Rendering System supporting Electronic Commerce Applications in IP-Networks, Proc. of the Int. Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'00), Las Vegas, Nevada, H.R. Arabnia (ed.), CSREA Press, 4, 2000, 1801-1807

[13] M. Rasch, O. Schmidt: Parallel Mesh Simplification, Proc. of the Int. Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'00), Las Vegas, Nevada, H.R. Arabnia (ed.), CSREA Press, 3, 2000, 1361-1367

[14] A. Junklewitz: Effiziente dynamische Lastverteilung für datenparalleles Radiosity mittels Latency-Hiding, Diplomarbeit, Universität Paderborn, 2000

[15] T. Plachetka: HiQoS Rendering System: Performance Messungen der ersten Evaluierung, Anlage zu IEZ AG: HiQoS Evaluierungsbericht, 2000

[16] M. Rasch: Parallele Optimierung von Polygonnetzen zur Beschleunigung der interaktiven Walkthrough-Animation, Diplomarbeit, Universität Paderborn, 1999

[17] P. Altenbernd, F. Cortes, M. Holch, J. Jensch, O. Michel, C. Moar, T. Prill, R. Lüling, K. Morisse, I. Neumann, T. Plachetka, M. Reith, O. Schmidt, A. Schmitt, A. Wabro: BMBF-Projekt HiQoS: High-Performance-Multimedia-Dienste mit Quality-of-Service Garantien, Statustagung des BMBF, HPSC'99, Höchstleistungsrechnen in der Bundesrepublik Deutschland, R. Krahl (ed.), Bonn, 1999

[18] L. Reeker, O. Schmidt: New Dynamic Load Balancing Strategy for Efficient Data-Parallel Radiosity Calculations, Proceeding of the Int. Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'99), Las Vegas, Nevada, H.R. Arabnia (ed.), CSREA Press, 1, 1999, 532-538

[19] IEZ AG: HiQoS Evaluierungsbericht, 2000

[20] M. Handley, V. Jacobsen: SDP: Session Description Protocol, RFC 2327

[21] H. Schulzrinne, A. Rao, R. Lanphier: Real Time Streaming Protocol (RTSP), RFC 2326

[22] H. Schulzrinne, S. Casner, R. Frederik, V. Jacobson: RTP: A Transport Protocol for Real-Time Applications, RFC 1889

[23] H. Schulzrinne: RTP Profile for Audio and Video Conferences with Minimal Control, RFC 1890

[24] R. Braden, L. Zhang, S. Berson, S. Herzog, S. Jamin: Resource ReserVation Protocol (RSVP), RFC 2205

[25] J. Wroclawski: The Use of RSVP with IETF Integrated Services, RFC 2210

[26] S. Shenker, J. Wroclawski: General Characterization Parameters for Integrated Service Network Elements"RFC 2215

[27] Michael Holch: Halbjahresbericht für den Zeitraum 1. Juli 2000 bis 31. Dezember 2000, (APE GmbH), München, Februar 2001

[28] Michael Holch: Beschreibung des APE-Trials innerhalb des Forschungsprojektes HiQoS, (APE GmbH), München, Februar 2001

[29] Michael Holch: Evaluationsreport des APE-Trials innerhalb des Forschungsprojektes HiQoS, (APE GmbH), München, April 2001

[30] www.opensource.org

[31] The Halloween Documents - Where will Microsoft try to drag you today? Do you really want to go there?, www.opensource.org/halloween

[32] Open Source Stars, www.engr.csufresno.edu/Personal/CSci/Faculty/Read/OpenSource/index.html

[33] Snoopy & Martin Müller: Open Source kurz & gut, O'Reilly & Associates, April 1999

[34] Chris DiBona, Sam Ockman, Mark Stone: Open Sources, O'Reilly & Associates, Januar 1999

[35] Eric S. Raymond: The Cathedral & the Bazar, O'Reilly & Associates, Oktober 1999

[36] Synchronized Multimedia Integration Language (SMIL) 1.0 Specification ; W3C Recommendation 15 - June -1998; http://www.w3.org/TR/REC-smil/

[37] Synchronized Multimedia Integration Language (SMIL 2.0) Specification;W3C Proposed Recommendation 05 - June - 2001; http://www.w3.org/TR/smil20/

[38] A. Burns, A. Wellings: Real-Time Systems and Programming Languages, Addison Wesley, 1997

[39] Simon Schneider: Combining Multimedia Response-Time Analysis and the Resource Reservation Protocol for Efficient Network Scheduling of Media Streams. Diplomarbeit, Universität Paderborn, 2001

[40] S. Shenker, C. Partridge, R. Guerin: Specification of Guaranteed Quality of Service, RFC, 2212, IETF, Sept. 1997

[41] Simon Bicskey: Task 5.6 – Abschlussbericht für das Live-Streamining Szenario

[42] P. Altenbernd (Editor): HiQoS-Zwischenberichte: 1. Juli – 31. Dezember 2000, 1. Januar – 30. Juni 2000, 1. Juli – 31. Dezember 1999, 1. Januar – 30. Juni 1999, 1. Mai – 31. Dezember 1998; http://www.c-lab.de/hiqos/intern/reports.html

[43] P. Altenbernd, R. Lüling (Ed.): HiQoS-Projektantrag, Detailplanung der ersten Projektphase Monat 1 – 15, Detailplanung der zweiten Projektphase Monat 16 – 27, Detailplanung der dritten Projektphase Monat 28 – 36; http://www.c-lab.de/hiqos/intern/reports.html