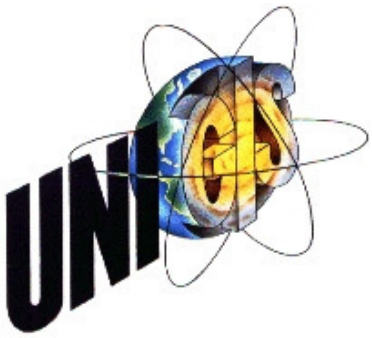

Master Thesis

im Rahmen des
Universitätslehrganges „Geographical Information Science & Systems“
(UNIGIS MSc) am Zentrum für GeoInformatik (Z_GIS)
der Paris Lodron-Universität Salzburg

zum Thema

„3D-WebGIS“
im Umfeld standardisierter Dienste

vorgelegt von

Dipl.-Ing. (FH) Kathrin Schütze
U1301, UNIGIS MSc Jahrgang 2007

Zur Erlangung des Grades
„Master of Science (Geographical Information Science & Systems) – MSc(GIS)”

Gutachter:
Ao. Univ. Prof. Dr. Josef Strobl

Leipzig, 30.06.2009

Vorwort

Dass diese Arbeit sich mit dem Thema WebGIS beschäftigen sollte, stand schon bei Beginn meines Studiums fest. Das Themenfeld um webbasierte GI-Systeme hat mich schon immer interessiert. Die Einbeziehung der dritten Dimension und die Entwicklung von dreidimensionalen WebGIS-Anwendungen finde ich besonders spannend.

Als Beispiel sollte das geografische Informationssystem des Forschungsprojektes „Astronomische und geodätische Untersuchungen der Linien und Geoglyphen auf der Pampa von Nasca/Peru“ der Fakultät Geoinformation der Hochschule für Technik und Wirtschaft Dresden herangezogen werden. Während der Bearbeitung dieser Arbeit stellte ich fest, wie umfangreich das Gebiet der dreidimensionalen GI-Systeme und deren Webanwendungen ist. Aus diesem Grund reichte die Bearbeitungszeit leider nicht aus, für das NascaGIS eine dreidimensionale Webanwendung zu implementieren. Auf der hier gelegten Grundlage kann aber eine entsprechende Applikation entstehen.

Die Fertigstellung dieser Arbeit verdanke ich in erster Linie der Unterstützung meines lieben Mannes und meiner Familie. Besonders bedanke ich mich bei MSc (GIS) Christiane Richter und Prof. Dr.-Ing. Bernd Teichert für die Diskussion vieler Problemstellungen und hilfreichen Anregungen. Ebenso bedanke ich mich bei Frank Markus und Markus Briglmeir von der Firma Autodesk für die Bereitstellung der Software sowie bei Dr. Andreas Poth und Prof. Dr. Stephan Nebiker für die verschiedene Literatur.

Nicht zuletzt möchte ich mich bei Prof. Dr. Josef Strobl bedanken für die vielen interessanten und nützlichen Anregungen vor allem in in der Phase der Themenfindung und die Betreuung der Arbeit.

Eigenständigkeitserklärung

Ich versichere, diese Master Thesis ohne fremde Hilfe und ohne Verwendung anderer als der angeführten Quellen angefertigt zu haben, und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat. Alle Ausführungen der Arbeit, die wörtlich oder sinngemäß übernommen wurden, sind entsprechend gekennzeichnet.

Ort und Datum eigenhändige Unterschrift

Inhaltsverzeichnis

Abkürzungen

3D	Dreidimensional
Abb.	**Abb**ildung
API	**A**pplication **P**rogramming **I**nterface
CityGML	**City G**eography **M**arkup **L**anguage
DB	**D**aten**b**ank
DGM	**D**igitales **G**elände**M**odell
EPSG	**E**uropean **P**etroleum **S**urvey **G**roup
GDI	**G**eo**d**aten**i**nfrastruktur
GIS	**G**eografisches **I**nformations**s**ystem
GML	**G**eography **M**arkup **L**anguage
ISO	**I**nternational **O**rganization for **S**tandardization
KML	**K**eyhole **M**arkup **L**anguage
LoD	**L**evel **o**f **D**etail
OGC	**O**pen **G**eospatial **C**onsortium, Inc.®
OpenGL	**Open G**raphics **L**ibrary
SLD	**S**tyled **L**ayer **D**escriptor
TC 211	**T**echnical **C**ommittee **211**
TIN	**T**riangulated **I**rregular **N**etwork
URL	**U**niform **R**esource **L**ocator
VRML	**V**irtual **R**eality **M**odeling **L**anguage
W3DS	**W**eb **3D S**ervice
WCS	**W**eb **C**overage **S**ervice
WFS	**W**eb **F**eature **S**ervice
WMS	**W**eb **M**ap **S**ervice
WPVS	**W**eb **P**erspective **V**iew **S**ervice

WTS	**W**eb **T**errain **S**ervice
X3D	E**x**tensible **3D**
XML	E**x**tensible **M**arkup **L**anguage

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einführung

1.1 Motivation

„Der Bildschirm ist ein Fenster, durch das man eine virtuelle Welt sieht. Wir müssen nun dafür sorgen, dass die Welt real aussieht, real agiert, real klingt und sich real anfühlt.“ (Ivan Sutherland 1965)

Wie diese Aussage des Zitates von Ivan Sutherland war es schon immer das Bestreben eines geografischen Informationssystems ein Abbild der realen Welt zu geben. Im Laufe der Zeit hat sich dieses Bild immer weiter der Realität angenähert.

Zuerst gab es sehr komplexe Systeme, die klassischen GI-Systeme, die nur von Experten bedient werden konnten. Die weitere Entwicklung brachte eine neue Generation von geografischen Informationssystemen hervor, das WebGIS. An Stelle von komplexen, proprietären Systemen, entwickelten sich Systeme, die auf einem flexiblen Denkansatz basieren. Diese Web-Anwendungen bieten die Möglichkeiten, GIS-Funktionen über ein Intra- oder Internet zu nutzen. Mit der Entwicklung und Verbreitung dieser Web-Applikationen wurde es notwendig, Schnittstellen und Prozesse zu standardisieren, um verschiedene Dienste gemeinsam nutzen zu können. Ein weiteres Ziel der Standardisierung besteht darin, dass der Nutzer möglichst wenige Anforderungen, wie z. B. spezielle Software, erfüllen muss, um eine Anwendung zu nutzen. Im Bereich der Geoinformationen sind für Standards und Normen das Open Geospatial Consortium, Inc. (OGC) und das Technical Committee 211 (TC 211) der International Organization for Standardization (ISO) zuständig. Durch deren Arbeit gibt es heute eine Reihe von Spezifikationen, die eine Interoperabilität von unterschiedlichen Geodaten und Web-Diensten ermöglichen.

Der nächste Entwicklungsschritt ging hin zur dreidimensionalen Visualisierung. Die dreidimensionale Darstellung von Geodaten ist in ihrer realistischen Wirkung faszinierend. Je detaillierter eine dreidimensionale Welt gestaltet wird, umso stärker ist der Eindruck, sich in der realen Welt zu bewegen. 2004 kaufte *Google Inc.* die kalifornische Firma *Keyhole* und begann deren „Keyhole Earth Viewer“ unter dem Namen „Google Earth“ zu vertreiben. Mit diesem Earth Viwer hat jeder die Möglichkeit, sich Geodaten der ganzen Erde in einem dreidimensionalen Raum kostenfrei zu betrachten. In dieser Umgebung kann man auch seine eigenen Geodaten

einbringen. Durch die Verwendung eines digitalen Geländemodells mit darüber gelegten Luft- und Satellitenbildern entsteht der Eindruck einer realen Landschaft. Mit der Verbreitung von Google Earth entstand ein immer größer werdendes Interesse an Geodaten. Die so wachsende Nachfrage nach Geodaten gab dem Bereich der geografischen Informationssysteme Aufschwung. Man begann die GI-Systeme weiterzuentwickeln, so dass die Verarbeitung, Analyse, Speicherung und Visualisierung von 3D-Geodaten möglich wurde.

Großes Potential in der 3D-Visualisierung wird in der Unterstützung beim Vertrieb der jeweiligen Daten gesehen, sowie bei den Präsentationsmöglichkeiten im touristischen Bereich. Um dieses Potential zu nutzen, soll die Visualisierung auch über Web-Dienste zur Verfügung gestellt werden können. Immer mehr Städte bauen dreidimensionale Stadtmodelle auf. Zur Veröffentlichung und Visualisierung dieser Modelle und weiterer 3D-Geodaten über das Internet wird momentan meist Google Earth verwendet. Mit den Zielen der Interoperabilität und Nutzerfreundlichkeit soll die Visualisierung dieser Daten aber auch über Web-Dienste ermöglicht werden. So arbeitet das OGC an entsprechenden Spezifikationen, um diesen Wunsch auf der Basis standardisierter Dienste zu ermöglichen.

Die Erläuterung von standardisierten Diensten und Formaten, die bei einem 3D-WebGIS verwendet werden können, sowie der Aufbau eines solches Systems soll Inhalt der vorliegenden Arbeit sein.

1.2 Überblick

1.2.1 Zielsetzung

Das Ziel dieser Arbeit ist es, einen Überblick über standardisierte Dienste und Formate zu geben, die für die Implementierung eines 3D-WebGIS momentan zur Verfügung stehen. Dabei sollen Möglichkeiten für den Aufbau eines WebGIS erläutert werden, welches dreidimensionale Ansichten auf Daten zur Verfügung stellt. Diese Ansicht soll „on the fly“ aus einem vorhandenen Datenspeicher generiert werden können und die Einbindung von zwei- und dreidimensionalen Daten ermöglichen.

Mit Hilfe der Formulierung von Anforderungen an einen solchen Web-Dienst soll ein Lösungskonzept entwickelt werden, mit dem ein 3D-WebGIS für einen vorhandenen Datenbestand implementiert werden kann.

Der in dieser Arbeit betrachtete Bereich soll nicht die gesamte Thematik dreidimensionaler geografischer Informationssysteme enthalten. Es werden wichtige Ansätze für ein dreidimensionales WebGIS dargelegt. Daher ist es auch kein Ziel der Arbeit, eine komplette 3D-WebGIS-Lösung zu implementieren. Hierbei wird auf dreidimensionale Geodaten im Allgemeinen eingegangen und nicht auf dreidimensionale Stadtmodelle im Speziellen.

1.2.2 Struktur der Arbeit

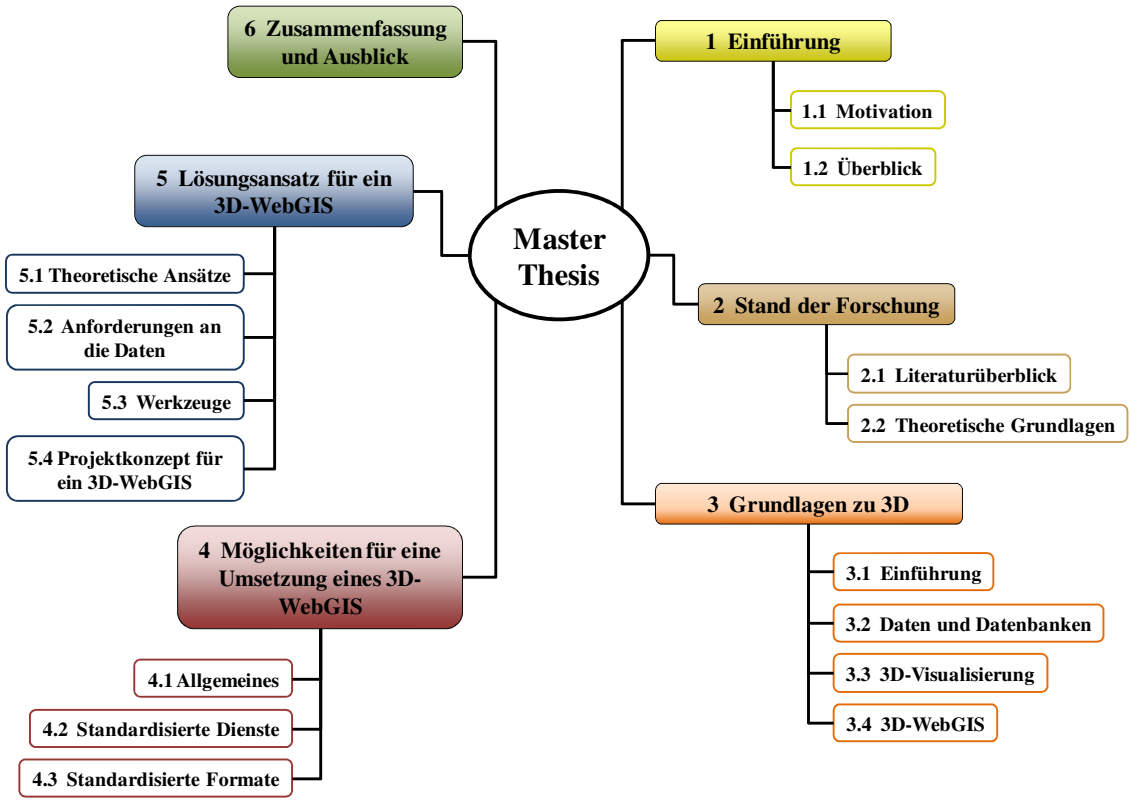

Abb. 1: Überblick

Zu Beginn der Arbeit wird ein Überblick über die Literatur gegeben. Anschließend werden die theoretischen Grundlagen im Bereich WebGIS kurz erläutert.

Kapitel 3 widmet sich den Grundlagen der Thematik. Es werden dreidimensionale Daten, deren Modellierung und Visualisierung erläutert. Anschließend geht man auf Basiswissen, das für ein 3D-WebGIS wichtig ist, ein.

Im nachfolgenden Kapitel stehen die Möglichkeiten für die Implementierung eines 3D-WebGIS im Mittelpunkt. Es werden die zur Verfügung stehenden standardisierten Dienste und Formate im Einzelnen erläutert.

Das Kapitel 5 befasst sich mit der Erarbeitung eines Lösungsansatzes für die Implementierung eines 3D-WebGIS. Anhand von verschiedenen Systemarchitekturen und den beschriebenen Diensten zeigt der Autor unterschiedliche Implementierungsansätze auf. Im nächsten Schritt werden die Anforderungen an die Daten, die visualisiert werden sollen, näher beschrieben, um dann auf die Werkzeuge einzugehen, die für eine Umsetzung vorgeschlagen werden. Anschließend folgt die Erstellung eines Projektkonzeptes. Dabei wird auf die Systemarchitektur für eine mögliche Applikation eingegangen und ein Workflow gezeigt, der den Programmablauf einer fertigen Anwendung beinhaltet.

Die vorliegende Arbeit wird mit einer Zusammenfassung und einem Ausblick abgeschlossen.

2 Stand der Forschung

2.1 Literaturüberblick

In [COORS & ZIPF 2004] wird ein sehr guter Einstieg in das Thema dreidimensionale Geoinformationssysteme gegeben. Die verschiedenen Autoren, die an diesem Buch mitgewirkt haben, vermitteln die Grundlagen zu diesem Thema, wozu auch die Erfassung von dreidimensionalen Daten gehört. In einem weiteren Abschnitt erläutert man die 3D-Datenmodellierung und geht auf topologische Modelle für die Geometriedaten und deren Analyse ein. Operatoren und die Repräsentationsmöglichkeiten, sowie wichtige Datenformate sind ebenfalls Bestandteil der Betrachtungen. Genauso wichtig wie das Speichern der Daten ist deren Visualisierung. Dazu erklärt man in diesem Werk die Grundlagen zum Thema Computergrafik und Echtzeit-Visualisierung via Internet. Im letzten Teil werden verschiedenste Anwendungen als Beispiele beschrieben, in denen 3D-GIS oder 3D-Stadtmodelle zur Anwendung kommen.

Eine weitere wichtige Literaturquelle zum Thema dieser Master Thesis ist [POMASKA 2007]. Er geht auf die Visualisierung von dreidimensionalen Modellen im Internet unter Verwendung von Open Source Produkten ein. So gibt er einen Überblick über die verschiedenen Möglichkeiten, die beim Modellieren von dreidimensionalen Daten Anwendung finden können und welche Formate wichtig sind.

In [KORDUAN & ZEHNER 2007] hingegen werden die grundlegenden Konzepte für ein internetgestütztes geografisches Informationssystem erläutert. Dabei stehen die einzelnen Komponenten, die benötigten Formate und Standards im Mittelpunkt. Am Ende wird ein Überblick über Lösungsansätze mit freier und kommerzieller Software gegeben.

Die Veröffentlichung [BEZEMA, U. MÜLLER et al. 2008] zeigt grundsätzliche Möglichkeiten für eine dreidimensionale Geodateninfrastruktur (GDI) auf. Es werden wichtige Standards erläutert und an einem Beispiel gezeigt, wie man ein Stadtmodell in eine GDI implementieren kann. [KOLBE 2004] befasst sich mit einer interoperablen Visualisierung von 3D-Geodaten im Internet, die auf Standards beruht. Dabei zeigt er Möglichkeiten auf, wie 3D-Modelle aus unterschiedlichen Datenquellen gemeinsam visualisiert werden können. Anhand des Projektes „Pilot 3D“ der GDI Nordrhein-Westfalen beschreibt er eine praktische Umsetzung. Eine dienstbasierende 3D-

Geovisualisierung im Internet wird von [SCHMIDT, MAY et al. 2006] dargestellt. Sie befassen sich mit verschiedene Client-/ Server-Architekturen, die ein 3D-WebGIS auf der Basis von Standards des OGC ermöglichen. Das Beispiel des 3D-Viewers der Landesanstalt für Ökologie, Bodenordnung und Forsten NRW (LÖBF) zeigt, wie eine Realisierung in der Praxis aussieht.

2.2 Theoretische Grundlagen

In den vergangenen Jahren wurde die Entwicklung von WebGIS-Applikationen sehr stark vorangetrieben. Immer mehr Geodatenbestände stellte man über das Internet einem größeren Nutzerkreis zur Verfügung. Es wurde nach Lösungen gesucht, wie man diese Daten verwenden kann, ohne auf proprietäre Anwendungen von speziellen Softwareherstellern angewiesen zu sein. Mit dem immer stärker werdenden Wunsch nach umfassenden Dateninfrastrukturen und der problemlosen Kombination unterschiedlicher Datensätze sind unter dem Dach des Open Geospatial Consortium, Inc. eine Reihe von Standards entwickelt worden, um diesen Wünschen nachzukommen.

Als erstes ist die Web Map Service (WMS) Spezifikation zu nennen. Diese ermöglicht nach einem einheitlichen Verfahren die Visualisierung von Geodaten in Form von Rasterbildern über das Internet. Auf diese Weise können Daten von unterschiedlichen WMS-Servern kombiniert und in einer Karte dargestellt werden. Aber auch Abfragen zu Sachdaten der einzelnen Objekte sind über diesen Service möglich. Um die Darstellung von den verschiedenen WMS zu vereinheitlichen, wurde die Styled Layer Descriptor (SLD) Spezifikation entwickelt. Mit diesem Standard ist es einem Nutzer möglich, die Darstellung der unterschiedlichen Daten selbst zu definieren und für seine eigene Ansicht zu vereinheitlichen.

Ein weiterer wichtiger Standard auf diesem Weg ist der Web Feature Service (WFS). Über den Dienst können komplette Objekte abgefragt und in einem vordefinierten Format übertragen werden. Durch eine Erweiterung dieses Standards ist es auch möglich, Objekte zu ändern und anschließend in die entsprechende Datenbank (DB) zurückzuschreiben.

Um auf multidimensionale Rasterdaten, wie z. B. Digitale Geländemodelle (DGM), zugreifen zu können, wurde der Web Coverage Service (WCS) entwickelt.

Zur Konfiguration und Steuerung der unterschiedlichen Dienste wird eine einheitliche Sprache verwendet, die Extensible Markup Language (XML). Es handelt sich dabei um eine Dokumentenbeschreibungssprache, die 1998 vom World Wide Web Consortium (W3C) zu einem offiziellen Standard gemacht wurde. Für einen standardisierten Austausch von Geodaten wurde auf deren Grundlage eine weitere Beschreibungssprache entwickelt, Geography Markup Language (GML). Sie ermöglicht die Modellierung, den Transport und die Speicherung von räumlichen Daten. Dieses Format findet z. B. bei der Übertragung von angefragten Geo-Objekten über einen WFS Anwendung.

Ein guter Überblick über die angesprochenen Standards und Formate ist in [MITCHELL et al. 2008] und [KORDUAN & ZEHNER 2007] zu finden. Detailliertere Beschreibungen zu den einzelnen Bestandteilen und Parametern sowie deren Verwendung finden sich in den entsprechenden Spezifikationen des OGC.

3 Grundlagen zu 3D

3.1 Einführung

Die Möglichkeiten für eine zweidimensionale WebGIS-Applikation, die eine ganze Reihe an GIS-Funktionalitäten bietet, sind schon sehr ausgereift. Es stellte sich die Frage, ob dreidimensionale Modelle auf die gleiche Art und Weise zur Verfügung gestellt werden könnten. 2003 begannen die ersten Städte dreidimensionale Stadtmodelle aufzubauen, die immer größeren Ansprüchen gerecht werden sollen, wozu auch die Interoperabilität im Rahmen einer Geodateninfrastruktur gehört. So wurde von der Special Interest Group (SIG) 3D der Geodateninfrastruktur in Nordrhein-Westfahlen (GDI NRW) begonnen, aufbauend auf der Beschreibungssprache GML die City Geography Markup Language (CityGML) Spezifikation zu entwickeln. Mit diesem Format ist es möglich, dreidimensionale Stadtmodelle in standardisierter Form zu modellieren, abzuspeichern und auszutauschen. 2008 wurde CityGML vom OGC zum offiziellen Standard erhoben. Heute haben eine Reihe deutscher Städte, wie Berlin, Bonn und Köln, auf dieser Basis ihre 3D-Stadtmodelle aufgebaut. [OGC 2008]

Neben der Entwicklung eines entsprechenden Formates für dreidimensionale Geodaten begann man auch an der Visualisierung der selben über das Internet, in Form eines WebGIS, zu arbeiten. So entstanden die ersten Vorschläge für entsprechende Spezifikationen. Dazu zählen in erster Linie der Web 3D Service (W3DS) und der Web Terrain Service (WTS) sowie der Web Perspective View Service (WPVS), der den WTS ersetzen soll. Diese Vorschläge werden seit einigen Jahren innerhalb des OGC diskutiert, sind aber noch nicht zu einem offiziellen OGC-Standard erklärt worden. Sie sind allerdings soweit ausgereift, dass sie in verschiedenen Anwendungen schon implementiert worden, wie z. B. das Projekt „Pilot 3D“ der GDI Nordrhein-Westfalen [KOLBE 2004] zeigt. Seit 2008 entwickelt man den WPVS verstärkt weiter, da es sich bei diesem Dienst um einen Plug-In-freien Dienst handelt.

3.2 Daten und Datenbanken

3.2.1 3D-Daten

In herkömmlichen geografischen Informationssystemen und Datenbanken modelliert man Daten so, dass Rohdaten von deren Darstellung getrennt werden. Mit dieser Verfahrensweise hat man die Möglichkeit, dieselben Daten mit verschiedenen Darstellungsstilen zu visualisieren. Bei dreidimensionalen Daten sieht die Abbildung der Daten etwas anders aus. 3D-Geodaten werden auf der Basis von 3D-Modellen modelliert und gespeichert. Diese gelten als eine Visualisierung ihrer selbst. Da 3D-Modelle, Daten beinhalten, die für eine Darstellung benötigt werden, wie z. B. Oberflächeneigenschaften von Objekten. Aus diesem Grund ist eine Trennung der Daten und deren Darstellungsart nicht bzw. nur sehr begrenzt möglich.

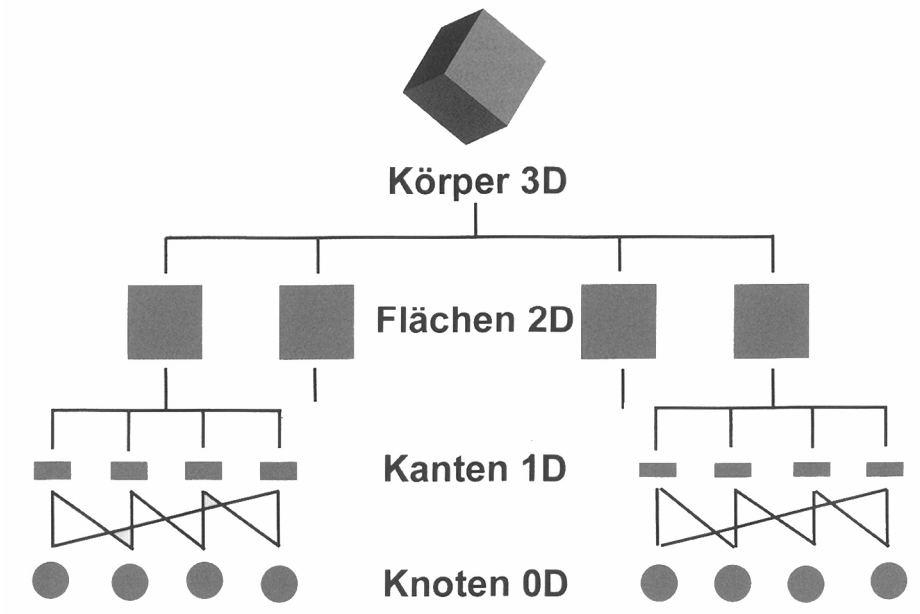

Abb. 2: Hierarchie der Elementarobjekte [POMASKA 2007]

In 3D-Modellen gibt es vier sogenannte Elementarobjekte. Abb. 2 zeigt deren Zusammenhänge und hierarchischen Aufbau. Als dimensionslos wird der Punkt, Knoten, bezeichnet, da er keine Ausdehnung besitzt und in der realen Welt nicht existiert. Die Linie, Kante, gilt als eindimensional und besitzt eine Längeneinheit. Gebildet wird eine Linie aus mindestens zwei Punkten. Das dritte Elementarobjekt ist die Fläche. Sie wird aus mindestens drei Linien gebildet und durch eine Flächeneinheit, welche sich aus der Multiplikation der Längeneinheiten ergibt, beschrieben. Eine Fläche gilt als zweidimensional. Das dreidimensionale Elementarobjekt ist das Solid, oder auch

Körper genannt. Es wird aus Flächen gebildet und trägt eine kubische Einheit, welche sich aus der Multiplikation der Grundfläche mit deren Normalen ergibt. Weiterführende Erläuterungen zu topologischer und geometrischer Datenmodellierung für dreidimensionale Geodaten sind bei [COORS & ZIPF 2004] zu finden.

Für die Repräsentation der Geometrien von dreidimensionalen Objekten gibt es nach [MÜLLER 2004] verschiedene Ansätze. Das am häufigsten verwendete Verfahren ist das der Polygon-Darstellung. Dabei erfolgt die Darstellung der Objektoberflächen mit Hilfe eines Netzes von Polygon-Facetten. Dieses Verfahren ist mathematisch recht einfach und ermöglicht eine effiziente Beschreibung von Algorithmen für die Darstellung und Schattierung der Polygone. Damit eignet es sich besonders für interaktive Anwendungen. Die Polygon-Darstellung gehört der Klasse der „Boundary Representations“ an. Darunter versteht man Darstellungsformen für Flächen- und Volumenmodelle, bei denen die Beschreibung der Objekte durch ihre begrenzenden Oberflächen erfolgt. Hierbei finden geschlossene, planare und einfache Polygone Verwendung. Geschlossen bedeutet in diesem Zusammenhang, dass das Polygon eine zusätzliche Kante besitzt, die die Facette definiert. Desweiteren gilt ein Polygon als planar, wenn alle Eckpunkte in einer Ebene liegen und als einfach, wenn sich das Polygon nicht selbst schneidet und keine Löcher besitzt. Eine weitere Eigenschaft dieser Polygone ist, dass man durch die Definition der Facette eindeutig zwischen Innen und Außen liegend unterscheiden kann. Die einfachste Form für solche Polygone sind Dreiecke, wie sie Abb. 3 als ein Beispiel für ein Polygonnetz zeigt.

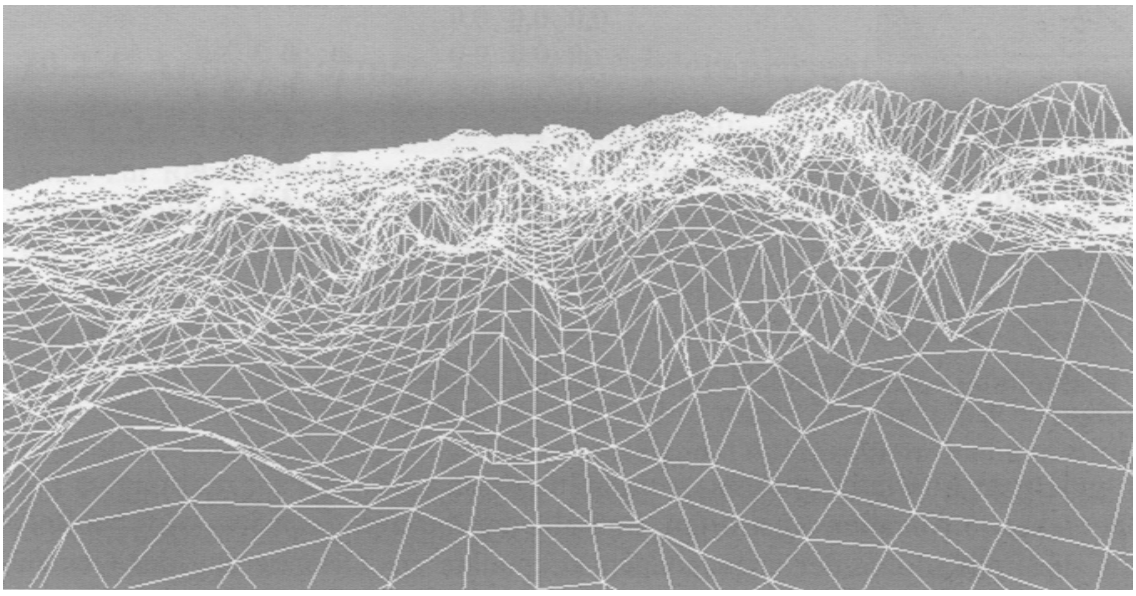

Abb. 3: Beispiel eines Polygonnetzes [MÜLLER 2004]

Möchte man auf diese Weise ein größeres Polygonnetz effizient verwalten und Redundanzen vermeiden, muss man spezielle Datenstrukturen verwenden. Am häufigsten wird die Datenstruktur „Index Face Set“ verwendet. Dabei kommt es zur Trennung der Koordinaten von der Beschreibung der Polygone. Alle Eckpunkte werden in einer Punktliste abgelegt. So brauchen die Eckpunkte, die aneinandergrenzende Polygone gemeinsam haben, nur einmal abgespeichert werden. Die Beschreibung der Polygone besteht aus einer Liste mit Indizes, der Facettenliste, die auf die entsprechenden Eckpunkte des Polygons in der Punktliste verweisen. Die Orientierung der jeweiligen Fläche ergibt sich aus der Reihenfolge der Eckpunkte, wie sie in der Facettenliste aufgeführt sind. In Abb. 4 ist ein Beispiel eines „Index Face Sets“ zu sehen. Werden Änderungen an einem Polygon vorgenommen, können auf diese Weise inkonsistente Daten vermieden werden. Die entsprechenden Änderungen müssen lediglich in der Punktliste vorgenommen werden. Diese Art der Repräsentation von Geometrien ist zentrales Beschreibungselement in den Szenebeschreibungssprachen Virtual Reality Modeling Language 97 (VRML97) und Extensible 3D (X3D), auf die im Laufe dieser Arbeit noch eingegangen wird. [MÜLLER 2004]

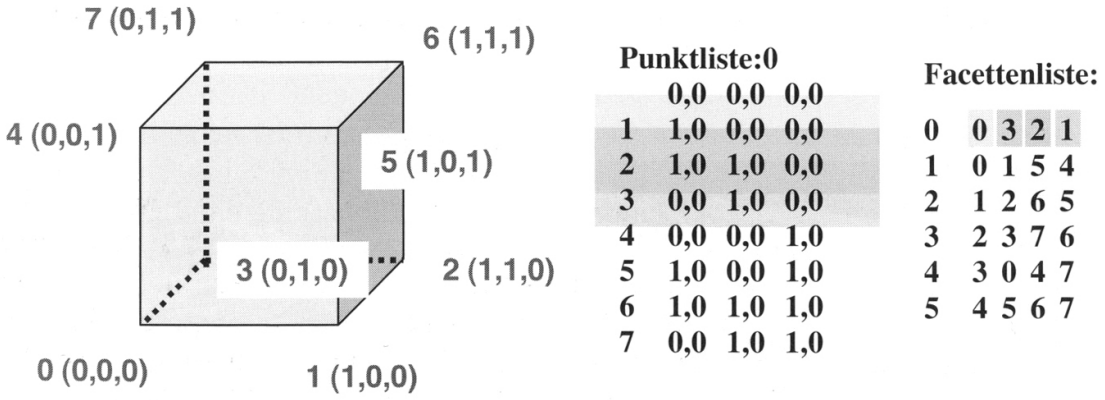

Abb. 4: „Index Face Set" für einen Würfel [MÜLLER 2004]

Neben diesem einfachen Verfahren der Geometrierepräsentation gibt es weitere komplexere Formen, die bei [MÜLLER 2004] nachgelesen werden können.

Um die modellierten 3D-Daten in eigenen Anwendungen nutzen zu können, gibt es entsprechende Application Programming Interfaces (API). Sie ermöglichen die Darstellung, Manipulation und Interaktion mit den 3D-Daten durch unterschiedliche Methoden. Eine wichtige Rolle spielt dabei Open Graphics Library (OpenGL). Dies ist

ein offener und industrieweiter Standard, der fast alle Systemplattformen unterstützt. Er bietet, neben unterschiedlichen Grafikfunktionen für 2D- und 3D-Daten, Funktionen zur Transformation, Beleuchtung und Darstellung von Flächenelementen an. Neben OpenGL gibt es noch weitere APIs im Bereich der 3D-Grafik. Primär für Windows-Plattformen gibt es Direct3D von Microsoft. Diese ist der OpenGL sehr ähnlich, arbeitet aber mit einem objektorientierten Ansatz bei der Verwaltung von 3D-Objekten und -Szenen. Als Plattformunabhängige API gilt die Java3D. Sie bietet vor allem Flexibilität bei der Verwaltung der Ein- und Ausgabegeräte. [MÜLLER 2004]

Eine weitere Modellierungsform von Daten im 3D-Bereich ist das Digitale Geländemodell (DGM). Es liefert die Höheninformationen des dargestellten Geländes und kann in Form von Raster- oder Vektordaten verwendet werden. Im ersten Fall sind die Höheninformationen je Pixel gespeichert und durch einen entsprechenden Grauwert repräsentiert. Im zweiten Fall, dem Vektorformat, handelt es sich um ein Triangulated Irregular Network (TIN). Grundlage für die Berechnung eines TINs sind die Knoten (Punkte mit X-, Y- und Z-Koordinaten). Diese werden mittels Dreiecksvermaschung, auch Triangulation genannt, verbunden, wobei keine Dreiecksseite eine andere schneiden darf. Zudem sollten die Dreiecke möglichst gleichseitig sein. Durch die Vermaschung entstehen Kanten zwischen den einzelnen Knoten und somit auch eine topologische Beziehung. Bei einer dreidimensionalen Visualisierung besteht die Oberfläche eines TINs aus Dreiecksflächen. Dies entspricht einem Polygonnetz, wie es Abb. 3 zeigt. Durch Interpolation kann nun an jeder Stelle ein Höhenwert abgenommen werden. Problem dabei ist die Genauigkeit, da diese sehr stark von den Ausgangspunkten und deren Verteilung über das berechnete Gebiet abhängt. [SCHIEBOLD 2007]

3.2.2 3D-Geodatenbanken

Allgemein werden in Geodatenbanken Geo-Objekte gespeichert. Sie besitzen verschiedene Sachattribute und ein Geometrieattribut. Mit Hilfe des Geometrieattributes können räumliche Anfragen und Operationen durchgeführt werden. Diese Grundaussage trifft natürlich auch auf 3D-Geodatenbanken zu, wobei das Geometrieattribut neben der Lage die dritte räumliche Dimension, die Höhe, mit ablegt.

Bisher konnten allerdings nur zweidimensionale Objekte mit einer Höhe in Datenbanken gespeichert werden, als sogenannte 2,5D-Daten. In einer 3D-Geodatenbank erfasst man die dreidimensionalen Objekte als räumliche Körper. Dadurch sind räumliche Anfragen und Operationen im dreidimensionalen Raum durchführbar.

Ob eine 3D-Geodatenbank eingesetzt werden muss, hängt nach [BRINKHOFF 2008] von unterschiedlichen Gegebenheiten ab. Zum einen ist zu untersuchen, ob Anfragen bearbeitet werden müssen, in denen Höhenangaben enthalten sind. Zum anderen ist es wichtig, zu prüfen, ob für eine zweidimensionale Position mehrerer Höhenwerte abzulegen sind. Ein dritter wichtiger Punkt betrifft die Frage, ob bei den Anfragebedingungen dreidimensionale Funktionen, wie z. B. die Berechnung eines Schnitt‘s zwischen zwei Körpern, erforderlich sind.

Sind 3D-Daten zu managen und zu administrieren, so benötigt man eine sehr leistungsstarke Datenbank, z. B. die objektrelationalen Datenbanken PostGIS oder Oracle Spatial. Allerdings ist momentan Oracle Spatial in der Version 11g die einzige Datenbank, die in der Lage ist, dreidimensionale Operationen zu bearbeiten. Weitere Informationen dazu findet man bei [KOTHURI, GODFRIND, et al. 2007].

3.3 3D-Visualisierung

3.3.1 Allgemeines

Nach [POMASKA 2007] ist die virtuelle Realität eine realitätsnahe Beschreibung einer Welt, die interaktiv betrachtet werden kann und Simulationseigenschaften besitzt. Demnach gehören neben den 3D-Modellen mit deren Geometrie und Attributen noch weitere szenebeschreibende Elemente dazu, um dem Betrachter ein möglichst realitätsnahes Abbild der realen Welt zu zeigen. Wird in der virtuellen Realität ein bestimmter Ausschnitt der realen Welt gezeigt, so spricht man von einer 3D-Szene. Diese definiert sich als eine zweidimensionale Projektion von dreidimensionalen Objekten in eine Sichtebene beliebiger Position und Winkel auf einem Ausgabegerät. Durch die Projektion wird der Eindruck einer räumlichen dritten Dimension beim Betrachter erweckt. [OGC 2003]

Unter 3D-Visualisierung verstehen [SCHMIDT, MAY et al. 2006] die Generierung visuell wahrnehmbarer Darstellungen von raumbezogenen Daten sowie die Prozesse in einem dreidimensionalen Darstellungsraum. Um eine Visualisierung von Objekten zu ermöglichen, müssen diese entsprechend beschrieben werden. Üblicherweise dient dazu ein Szenegraph als Datenstruktur. Er ist hierarchisch aufgebaut und beschreibt die einzelnen Objekte und deren Aussehen in der 3D-Welt. Um die geometrischen Bestandteile der Szene zu beschreiben, verwendet ein Szenegraph eine Baumstruktur, indem er mit Hilfe von Grundkörpern, Transformationen und Gruppierungen die einzelnen Objekte zusammensetzt. Die Abb. 5 zeigt ein Beispiel eines solchen Szenegraphen. In diesem Beispiel besteht die Szene (T_{XZ}) aus einem Hintergrund (S_{XZ}) und zwei Gebäuden (T_{XZ} und $R_y \otimes T_{XZ}$). Das erste Gebäude (T_{XZ}) setzt sich aus zwei Körpern zusammen, aus dem Dach (T_y) und dem Gebäudekörper (T_y). Ebenso das zweite Gebäude. Es besteht aus den gleichen Körpern, wie das erste Gebäude. Die Grundformen wurden allerdings um einen Faktor (R_y) skaliert, so dass sich ein längeres Gebäude ergibt. Das Beispiel zeigt sehr deutlich den hierarchischen Aufbau einer 3D-Szene.

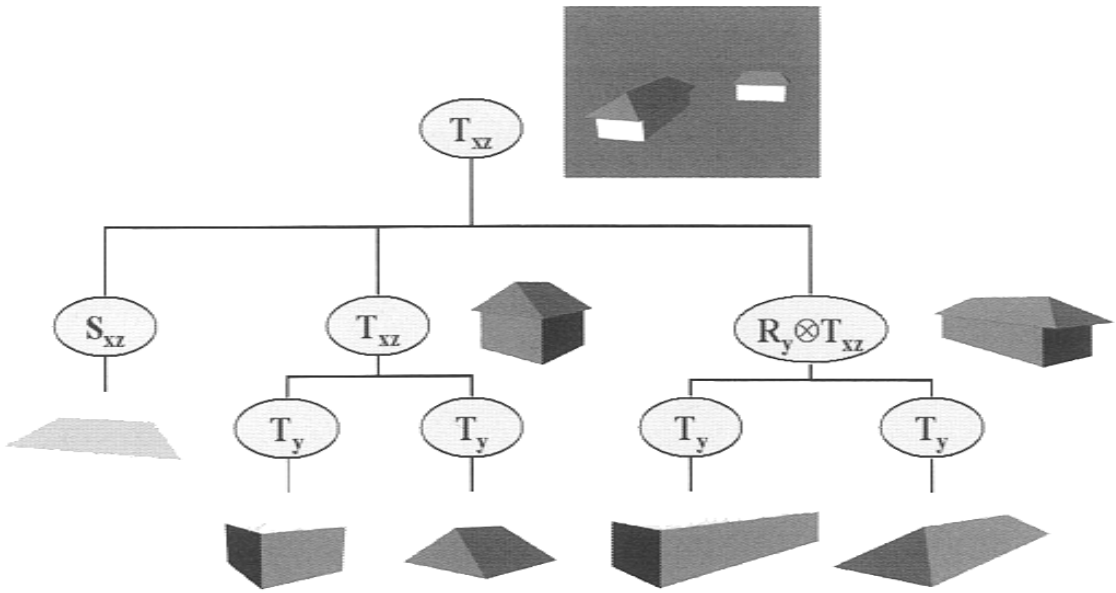

Abb. 5: Beispiel für einen einfachen Szenegraphen einer 3D-Szene [MÜLLER 2004]

Neben den geometrischen Objekten enthält der Szenegraph auch die möglichen Kamerapositionen und die Definitionen der Lichtquellen. Zur Vervollständigung der 3D-Welt werden Informationen zum Hintergrund und zur Ausgestaltung mit hinterlegt.

Soll die Szene Multimediaelemente, wie z. B. Interaktionsmöglichkeiten oder Audio-Quellen enthalten, so werden diese ebenfalls vom Szenegraphen verwaltet.

3.3.2 Texturen und Oberflächeneigenschaften

3D-Modelle stellen komplexe Strukturen und Landschaften dar, die durch individuelle Färbungen und Schattierungen der jeweiligen Polygone eines Objektes ein realistisches Aussehen bekommen. Um einen hohen Detailgrad zu erreichen, ist meist ein hoher Aufwand bei der Erzeugung des Modells nötig. Dabei stoßen die Komplexität sowie der benötigte Speicherbedarf schnell an die Grenzen der Umsetzbarkeit.

Aus diesem Grund werden Texturen verwendet. Nach [MÜLLER 2004] sind Texturen bildbasierende Informationen, die bei 3D-Objekten oder Polygonen für die punktuelle Veränderung globaler Oberflächeneigenschaften verwendet werden. Texturen sind daher Rasterbilder, die auf Objekte „aufgebracht“ werden, um einen höheren Detailierungsgrad zu erhalten. Dies ruft beim Betrachter den Eindruck einer homogenen Struktur hervor. Für den Szenenhintergrund finden Texturen ebenfalls Anwendung und bilden damit eine Art Kulisse, sodass der realistische Eindruck des 3D-Modells noch verstärkt wird.

Texturen kann man für unterschiedliche Effekte in einem 3D-Modell verwenden. Mit Hilfe des sogenannten „Texture Mapping“ wird die Objektoberfläche visualisiert. Dabei kann man auch Einfluss auf die Transparenz und die Beleuchtung eines Objektes nehmen. Mit Hilfe des „Bump Mappings“ verleiht man einem Objekt Reliefstrukturen. Dazu sind in der Textur neben der Basisstruktur, der Textur an sich, Reliefstrukturen hinterlegt. Ähnlich wie bei einem Digitalen Geländemodell stehen die einzelnen Farbwerte des Rasterbildes für Höheninformationen, die die Reliefstruktur ausprägen. [POMASKA 2007] Ein Beispiel für ein „Bump Map“ zeigt Abb. 6.

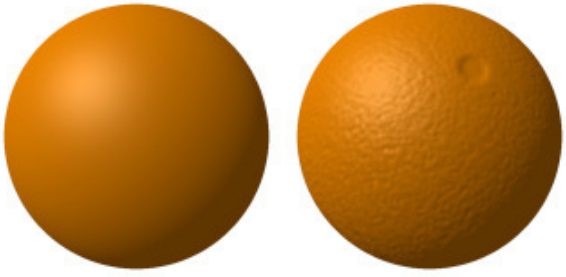

Abb. 6: „Bump Mapping“: li. ohne und re. die gleiche Kugel mit einem „Bump Map“ [Quelle: http://de.wikipedia.org/wiki/Bumpmapping]

Die wohl häufigste Anwendung für Texturen sind Rasterbilder mit realen Strukturen. Zum Beispiel werden Satellitenbilder oder auch thematische Karten als Textur auf Geländemodelle aufgebracht oder ein entzerrtes Foto einer Gebäudefassade auf die Oberfläche des entsprechenden 3D-Modells des Gebäudes. Bei diesem Verfahren stellt sich immer das Problem der begrenzten Ausdehnung und Auflösung der verwendeten Bilder. Eine andere Anwendung sind die künstlichen Texturen. Sie werden jeweils für eine bestimmte Ausdehnung und Auflösung aus einer Vorlage generiert. So können natürliche Strukturen, wie Wolken oder Feuer nachgebildet werden. Aber auch für die Modellierung von Materialien können Texturen verwendet werden, sogenannte Patterns oder Festkörperstrukturen. Sie haben eine feste Größe und werden solange auf einer Oberfläche wiederholt, bis ein Polygon vollständig gefüllt ist. Auf diese Weise sind z. B. Holzstrukturen nachbildbar. [MÜLLER 2004] Abb. 7 zeigt einige Beispiele für die Texturierung von Objekten und Szenen.

Abb. 7: Beispiele für „Texture Mapping“ [POMASKA 2007]

Texturen müssen mit den Oberflächen der Objekte verknüpft sein und die gleiche Größe haben, damit sie in einer 3D-Szene visualisiert werden können. Dazu werden sie auf die entsprechende Oberfläche transformiert. Man überführt dabei die Ortskoordinaten (x, y, z) der Oberfläche über einen Zwischenschritt, die Parameterkoordinaten (u, v), in Texturkoordinaten (r, s), um dann die Textur „auflegen“ zu können, wie Abb. 8 zeigt. Die Transformation kann allerdings nicht ganz spannungsfrei durchgeführt werden, da die Ausdehnung der Textur und der entsprechenden Oberfläche des Objektes nicht exakt übereinstimmen. So ist es nicht möglich die Texturierung ohne Verzerrungen durchzuführen. Ein weiterer Grund für das Auftreten von Verzerrungen kann auch die

Oberflächengestalt des Objektes sein. Dies ist der Fall, wenn gewölbte Oberflächen mit Texturen belegt werden. [MÜLLER 2004]

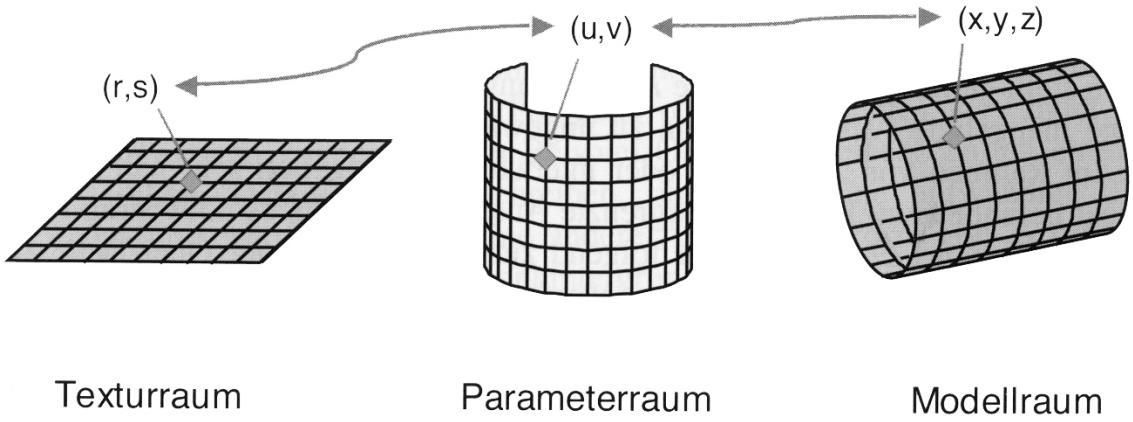

Abb. 8: Transformation der Ortskoordinaten in Texturkoordinaten [MÜLLER 2004]

Damit das Aussehen eines Objektes realistisch wirkt, ist es wichtig, dass die natürlichen Wechselwirkungen zwischen Farbe, Beleuchtung und Oberflächenbeschaffenheit des Objektes in etwa der Realität entsprechen. Dazu werden den einzelnen Oberflächen Eigenschaften, wie Lichtreflexion und -absorption sowie Rauigkeit und Transparenz zugewiesen. Mit Hilfe dieser Eigenschaften können die Objekte auf die eingesetzten Lichtquellen in einer Szene „reagieren“. [POMASKA 2007]

3.3.3 Level of Detail

Der Rechenaufwand für die Darstellung einer 3D-Szene mit hohem Detailierungsgrad ist sehr groß. Um das Laufzeitverhalten beim Generieren der 3D-Szene nach jeder Nutzeraktion zu verbessern, gibt es die Möglichkeit für jedes Objekt verschiedene Abstrahierungsgrade zu verwenden. Diese nennt man Level of Detail (LoD). Grundsätzlich handelt es sich dabei um eine mehrstufige Abstraktion eines realen Objektes. Je geringer die Abstraktion, umso höher der Level of Detail und umso realistischer wirkt das Objekt.

Bei der 3D-Visualisierung in einer Szene werden die Objekte gleichzeitig in verschiedenen LoDs dargestellt. Welches Objekt in welchem LoD dargestellt wird, ist im Wesentlichen abhängig vom Abstand des Beobachters zum Objekt. Dabei gibt es zwei unterschiedliche Konzepte für die Berechnung der verschiedenen Abstrahierungsgrade. Das verbreitetste und älteste Konzept ist der statische Level of

Detail. Es werden jedem Objekt, dem Detailierungsgrad entsprechend, unterschiedliche Repräsentationsdaten zugewiesen, die bei Bedarf geladen werden. Die Daten speichert man explizit, meist in hierarchischer Form. Da bei dieser Methode nur eine relativ geringe Anzahl an Detailstufen verwendet wird, ist auch die Komplexität des Modells nicht so groß. Bei der Visualisierung kann es allerdings zu störenden Sprüngen zwischen den einzelnen Stufen kommen. Die häufigste Methode, diesen Effekt zu verringern, ist die allmähliche Überblendung von zwei unterschiedlichen Stufen. Das zweite Konzept ist der dynamische Level of Detail. Dabei wird eine sehr große Anzahl an Detailstufen verwendet, so dass man von einem stufenlosen Verfahren reden kann. Die unterschiedlichen Detaillierungen der 3D-Objekte werden implizit in einer Datenstruktur abgelegt. Hier kommt es nicht zur Erzeugung unterschiedlicher Repräsentationsformen eines Objektes und deren explizite Speicherung. Es werden eine Reihe von Operatoren in einer festen Struktur abgelegt. Mit diesen Operatoren kann bei jeder Visualisierung die jeweils nächste Detailstufe eines Objektes berechnet werden. [ZACH, GRABNER, et al. 2004]

Im Bereich der digitalen Stadtmodelle verwendet man die statischen LoDs und unterscheidet fünf unterschiedliche Stufen, den Level of Detail der Stufe 0 bis 4. Dabei orientieren sich diese Stufen an der Objektauflösung, dem Umfang der darzustellenden Objekte, und der Genauigkeit. Bei der Visualisierung werden beim Laden der nächst höheren Detailstufe die fehlenden Objekte hinzu geladen. Die unterschiedlichen Levels of Detail sind in Abb. 9 zu sehen.

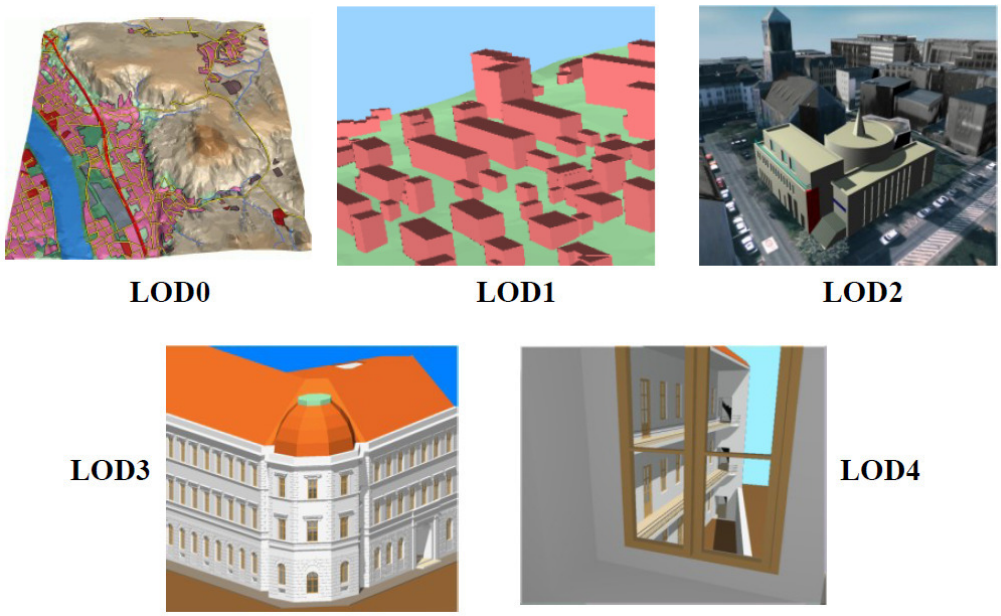

Abb. 9: Levels of Detail [OGC 2008]

Im LoD 0, dem sogenannten Regionalmodell, wird das digitale Geländemodell dargestellt, welches mit Orthophotos oder thematischen bzw. topografischen Karten texturiert werden kann. Die LoD 1 bildet das Blockmodell. Es werden die Gebäudegrundrisse und deren Höhe hinzu geladen und als Blöcke dargestellt. Die nächst höhere Stufe bildet das Strukturmodell, LoD 2. Hier sind vereinfachte Dachformen ausgeprägt und eine thematische Koloration und Texturierung wird vorgenommen, die den Materialeigenschaften der Objekte entspricht. Im LoD 3, dem Architekturmodell, ersetzt man die vereinfachten Dachformen durch reale. Fenster und Türen oder auch Balkone der Gebäude sowie die hochauflösenden Texturen der Objekte werden hinzu geladen. Die höchste Detailierungsstufe, LoD 4, bildet das Innenraummodell. In diesem Modell werden sehr detaillierte Gebäudemodelle und deren ausmodellierte Innenräume hinzugefügt. [POMASKA 2007]

3.4 3D-WebGIS

3.4.1 Allgemeines

Das Internet bietet unendlich viele Möglichkeiten, Informationen zu sammeln und einer sehr breiten Öffentlichkeit zur Verfügung zu stellen. Die Haltung und Pflege der entsprechenden Daten verbleibt beim Anbieter. Aus diesem Grund werden auch Karten und Datenbestände auf diesem Wege zur Verfügung gestellt. Ein weiterer Grund zur Nutzung des Internets für diese Zwecke ist das Vorhandensein funktionierender und standardisierter Technologien für Anfragen eines Nutzers (REQUEST) und deren Beantwortung durch den Server (RESPONSE). Im Laufe der Entwicklung haben sich nach [KORDUAN & ZEHNER 2007] unterschiedliche Begriffe für digitale Kartendarstellung im Internet ergeben. Bei Web-Mapping handelt es sich um Systeme, die einfache Kartendarstellungen mit Verschiebe- und Zoomfunktionen bieten und dem Nutzer ermöglichen, verschiedene Layer auszuwählen. Systeme, die neben einer digitalen Karte auch Sachdaten der abgebildeten Objekte zur Verfügung stellen, werden als WebGIS bezeichnet. Mit einer solchen Anwendung können dem Nutzer Such-, Berechnungs- und verschiedene Analysefunktionen angeboten werden. Ein weiterer Begriff, der sich in der Literatur findet, ist das Internet-GIS. Dabei handelt es sich um eine Anwendung, die einem vollständigen Geoinformationssystem gleichkommt, da

Funktionen für die Eingabe, Verwaltung, Analyse und Darstellung von Daten zur Verfügung stehen.

Sowohl 2D- als auch 3D-WebGIS funktionieren auf der Basis der bekannten Client-Server-Technologie. Auf Anfrage eines Clients generiert der Server mit Hilfe der gesendeten Parameter eine entsprechende Antwort und sendet diese zurück an den Client, bei dem sie visualisiert wird. Direkt aus dem vorhandenen Datenbestand der Vektor- und Rasterdaten einer Datenbank oder einem anderen Datenspeicher erzeugt der Server die angeforderten Daten, z. B. eine 3D-Szene. Mit welchen Verfahren die Generierung der Szenen realisiert werden, wird im folgenden Kapitel noch beschrieben. Der Anspruch eines WebGIS ist es, dass an das System des Clients möglichst wenige Anforderungen gestellt werden müssen, um ein fehlerfreies Funktionieren der Anwendung zu garantieren. Dazu zählt zum Beispiel die Installation von speziellen Programmen, die zur Nutzung eines WebGIS notwendig wären.

Allgemein betrachtet, hat eine Webseite als Hauptaufgabe die Verarbeitung der Kommunikation mit dem Nutzer. Daraus ergeben sich für ein WebGIS nach [ZIPF 2000] die folgenden Charakteristika und Anforderungen:

- Passivität:
  Die gewünschten Informationen eines Systems sollen möglichst einfach und schnell von einem Nutzer gefunden werden können.
- Betrachtungsdauer:
  Eine Internetseite wird im Vergleich zu Printmedien relativ kurz betrachtet. Dadurch müssen die enthaltenen Informationen, vor allem der Karteninhalte, übersichtlich und schnell erfassbar gestaltet werden und sich auf die wichtigen Aussagen beschränken. Auch die Ladezeiten von Daten sind hier sehr wichtig und sollten möglichst kurz sein.
- Aktualisierbarkeit:
  Bei einem WebGIS ist es wichtig, dass die Karteninhalte dynamisch aus dem aktuellen Datenbestand geladen werden. Dadurch wird sichergestellt, dass der Nutzer auf die aktuellsten Daten zugreifen kann.
- Individualisierbarkeit:
  Der Vorteil gegenüber Printmedien, dass jeder Nutzer den Karteninhalt individuell, seinen Ansprüchen entsprechend, zusammensetzen kann, sollte genutzt werden.

- Anonymität:

  Zwischen Anbieter und Nutzer herrscht im Bereich der elektronischen Informations- und Kommunikationssysteme weitgehend Anonymität. Da kein Kontakt zwischen beiden zustande kommt, wie es beim Kauf einer analogen Karte im Handel der Fall wäre.

- Inkassofähigkeit:

  Da es sich bei einem WebGIS auch um den Vertrieb von Daten handeln kann, müssen entsprechende Abrechnungsverfahren genutzt werden.

Dieser letzte Punkt ist nicht Bestandteil dieser Arbeit und wurde der Vollständigkeit halber mit aufgeführt. Für weitergehende Informationen sei auf die entsprechende Literatur verwiesen.

3.4.2 Visualisierungspipeline

Damit man in einem WebGIS unterschiedliche Datenbestände gemeinsam nutzen kann, ist man auf Standards angewiesen. Diese werden im Umfeld von Geoinformationssystemen vom Open Geospatial Consortium, Inc. verabschiedet. Für die Realisierung von interoperabler Geovisualisierung greift das Consortium auf die vierstufige, sogenannte Visualisierungspipeline nach [DOYLE & CUTHBERT 1998] zurück, wie die Abb. 10 sie in leicht abgeänderter Form nach [KOLBE 2004] zeigt.

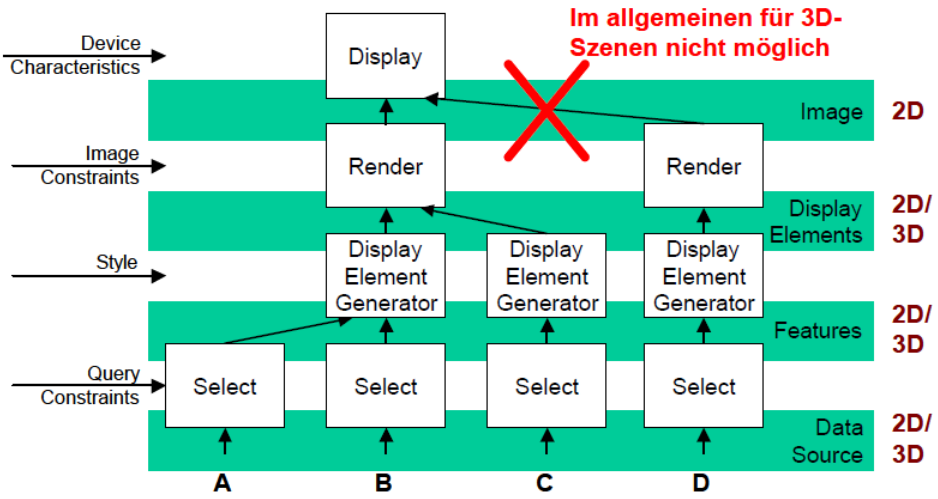

Abb. 10: Visualisierungspipeline [KOLBE 2004]

Danach verläuft eine Visualisierung grundsätzlich in vier Schritten ab. Sie beginnt auf der untersten Ebene mit der Objektauswahl in den unterschiedlichen Datenbeständen. Im folgenden Schritt werden diese Elemente, entsprechend der geforderten Darstellungsvorschrift, in grafische Elemente transformiert. Der dritte Schritt rendert aus diesen Grafikelementen ein Bild, welches im letzten Schritt am Ausgabegerät des Clients angezeigt wird. Handelt es sich dabei um zweidimensionale Visualisierungen, so können auf jeder der einzelnen Ebenen unterschiedliche Datensätze zusammengefügt werden. Bei dreidimensionaler Visualisierung ist dies nicht so ohne weiteres möglich. Grundvoraussetzung für die Kombination unterschiedlicher Datensätze mit 3D-Geometrien ist, dass alle im selben räumlichen Referenzsystem vorliegen. Eine weitere Voraussetzung ist, dass die Daten vor dem Renderingprozess zusammengefügt werden. Dabei erzeugt der Renderer aus dem Modell bzw. der 3D-Szene ein Bild, welches ein Ausgabegerät anzeigen kann. Einmal gerenderte 3D-Szenen können nicht übereinander gelegt werden, wie das bei zweidimensionalen Daten der Fall ist. 3D-Daten müssen ineinander integriert werden, so dass z. B. Objekte, die näher an einer Kameraposition sind, andere überdecken.

Die einzelnen Komponenten, die für den gesamten Visualisierungsrozess notwendig sind, müssen sich nicht zwangsläufig auf ein und demselben Server befinden. Eine Verteilung, auch über das Internet, ist umsetzbar. Dies gilt selbstverständlich ebenso für die Datenbestände. Neben der Visualisierung ist es bei einem 3D-WebGIS möglich, den Renderingprozess auf den Client zu verlagern. Dies ist abhängig von der verwendeten Systemstruktur. Auf diesen Punkt wird später in der Arbeit eingegangen. [KOLBE 2004]

3.4.3 Perspektive

Allgemein versteht man unter Perspektive die Möglichkeit, auf einer zweidimensionalen Fläche ein dreidimensionales Objekt so abzubilden, dass ein räumlicher Eindruck entsteht. Die Definition einer Szenenperspektive erfolgt durch einen dreidimensionalen Zielpunkt, die Entfernung, das Azimut und den Neigungswinkel einer Kameraposition. Diese und einige weitere Angaben zur Definition einer Perspektive müssen, neben den Parametern zu den gewünschten Daten selbst, bei einer Anfrage nach einer 3D-Szene bei einem Server mit übermittelt werden. Die Angabe der einzelnen Parameter erfolgt meistens in der Uniform Resource Locator

(URL). Mit diesen Informationen kann das Bild in der richtigen Perspektive auf die gewünschten Daten erzeugt werden, um es am Bildschirm des Clients anzuzeigen. Die Abb. 11 veranschaulicht die Möglichkeiten für die Definition einer Perspektive.

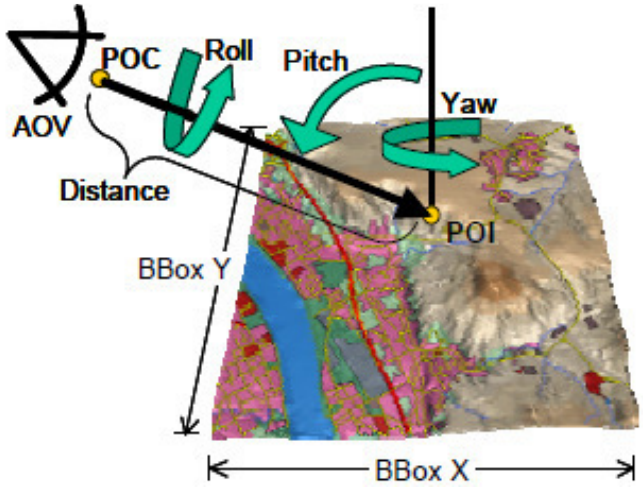

Abb. 11: Parameter für eine Perspektive [KOLBE 2004]

POI (Point of Intrest) stellt in dieser Abbildung den Zielpunkt und POC (Point of Camera) die Position der Kamera bzw. des Beobachters der Szene dar. Beide werden als dreidimensionale Punkte angegeben und definieren den Abstand (Distance) und den Blickwinkel (AOV – Angle of View) des Beobachters zur Szene selbst. Anstelle der Kameraposition können daher auch diese bei der Definition einer Perspektive angegeben werden. Ein weiterer wichtiger Parameter ist die Angabe des sichtbaren Bereichs (Bounding Box – BBOX Y, BBOX X), der mindestens zu sehen sein soll. Da es sich um Perspektiven handelt, ist die Grundfläche der sichtbaren Daten ein Trapez und kein Rechteck. Um die Definition trotzdem mit Hilfe einer Bounding Box zu ermöglichen, wird der Bereich angegeben, der mindestens sichtbar sein soll. Zur Anpassung der Szene sind noch drei weitere Winkel wichtig. Yaw gibt an, wie die Szene in Bezug auf die Nordrichtung ausgerichtet ist. Der Neigungswinkel des Beobachters zur Szene wird durch Pitch bezeichnet und die Verdrehung um die Blickrichtung als Roll. [OGC 2005a]

4 Möglichkeiten für eine Umsetzung eines 3D-WebGIS

4.1 Allgemeines

In den heutigen Systemlandschaften ist Interoperabilität von Systemen und Daten ein sehr wichtiger Punkt. Dies gilt vor allem für Anwendungen, die Dienste über das Internet bereitstellen. Da muss es möglich sein, verschiedene Dienste und Daten miteinander zu kombinieren und verlustfrei auszutauschen, unabhängig von den Systemkomponenten, die ein Nutzer verwendet. Dies ist machbar, indem die Anbieter von Diensten und Daten mit Standards arbeiten. Wie schon erwähnt, sind im GIS-Umfeld für die Spezifikation von Standards und Normen, das Open Geospatial Consortium, Inc. und das Technical Committee 211 der International Organization for Standardization zuständig. Die unter dem Dach des OGC laufenden Standardisierungsprozesse tragen als Markennamen OpenGIS und haben die Interoperabilität geodatenverarbeitender Software zum Ziel.

In [MÜLLER 2005] wird Interoperabilität wie folgt definiert: „Interoperabilität ist die Fähigkeit möglichst vieler Systeme oder Komponenten, Daten elektronisch auszutauschen und sie mit möglichst wenig Aufwand, insbesondere ohne manuelle Bearbeitung, weiter zu verwenden.“ Zu diesem Zweck hat das OGC eine Reihe von Standards veröffentlicht. Ein Teil dieser Standards sollen im Folgenden betrachtet werden. Im Mittelpunkt stehen dabei die Standards, die für ein 3D-WebGIS am wichtigsten sind. Neben den Standards, die vom OGC verabschiedet wurden, werden auch Dokumente betrachtet, die innerhalb des OGC noch bearbeitet und diskutiert werden.

Die meisten von ihnen implementieren Entwickler schon in ihre Anwendungen, wenn sie sich noch in der Diskussion befinden. Daher wird im Folgenden generell von Standards gesprochen, auch wenn sie noch nicht vom OGC offiziell bestätigt wurden.

Die vom OGC definierten Web-Services dienen der Kommunikation zwischen Anwendungen über definierte Schnittstellen. Mit Hilfe dieser Schnittstellen ist es den Diensten möglich, auf komplexe Funktionen anderer Anwendungen zuzugreifen. Jeder Dienst ist für sich autonom, allerdings können sie zu einem Netz kombiniert werden. Sie basieren alle auf der bereits erwähnten Client-Server-Struktur. Auf jede Anfrage

(REQUEST) des Clients folgt eine Antwort (RESPONSE) des Servers. Wie im Web üblich, erfolgt der Aufruf einer Anfrage über die Methoden POST oder GET. Dabei setzt sich die URL für die Anfrage aus dem Teil, der den Server und die auszuführende Applikation anspricht und dem standardisierten Teil, der die zu übermittelnden Parameter und Auszuführenden Methoden enthält, zusammen. Die einzelnen Parameter zählt man mit ihren Schlüsselwörtern und den zugehörigen Werten nach dem Aufruf der Methode hintereinander auf und verbindet sie mit dem Zeichen „&“. Der Aufruf eines WPVS könnte wie folgt aussehen:

```
„http://localhost:8080/deegree-
wpvs/services?request=GetView&BOUNDINGBOX=423750,4512700
,425500,4513900&DATASETS=Buildings,satellite_images&ELEV
ATIONMODEL=saltlake_dem&ROLL=0&AOV=60&CRS=EPSG:26912&WID
TH=800&HEIGHT=600&SCALE=1.0&STYLES=default&EXCEPTIONFORM
AT=INIMAGE&VERSION=1.0.0&OUTPUTFORMAT=image/jpg&backgrou
nd=cirrus&POI=424750.0,4513400.0,1350&YAW=20&PITCH=15&DI
STANCE=2500“
```

Mit „localhost:8080/deegree-wpvs/services“ wird der Server und der Dienst angesprochen. Der Aufruf der auszuführenden Operation erfolgt mit „request=GetView“. Im Anschluss folgt eine Reihe von Parametern mit ihren Werten.

Wichtig ist, dass auf jede Anfrage eine richtige Antwort oder eine entsprechende Fehlermeldung erfolgt. Wie eine solche Fehlermeldung auszusehen hat, ist in den einzelnen Spezifikationen festgelegt.

Jeder Web-Service besitzt mindestens eine Schnittstelle. Über diese können mit dem Aufruf GetCapabilities die Fähigkeiten des Dienstes selbst und die Metadaten der zur Verfügung stehenden Daten abgefragt werden. Die Antwort erhält der Client in einem XML-Dokument, welches bei jedem Dienst gleich aufgebaut und durch die ISO 19119 festgelegt ist. [KORDUAN & ZEHNER 2007]

Ein Beispiel dazu zeigt Abb. 12. Die entsprechende Anfrage für dieses Dokument lautet:

```
„http://localhost:8080/deegree-
wpvs/services?REQUEST=GetCapabilities&version=1.0&service
=WPVS“
```

Der Server wird auch hier mit „localhost:8080“ angesprochen und soll die Operation „GetCapabilities“ ausführen. Als Parameter gibt man die Version „1.0“ und den betreffenden Service „WPVS“ mit, über den man die Metadaten wünscht.


```
− <wpvs:WPVS_Capabilities updateSequence="0" version="0.00.01"
  xsi:schemaLocation="http://www.deegree.org/wpvs /home/rutger/eclipse-projekte/bonn_berlin_3D/resources
  /schema/deegreewpvs/1.0.0/deegree_wpvs.xsd">
  − <ows:ServiceIdentification>
      <ows:ServiceType>WPVS</ows:ServiceType>
      <ows:ServiceTypeVersion>1.0.0</ows:ServiceTypeVersion>
      <ows:Title>WPVS for Salt Lake City, Utah -USA</ows:Title>
    − <ows:Abstract>
        A WPVS which creates a perspective view of the Downtown region of Salt Lake City, Utah - USA
      </ows:Abstract>
    − <ows:Keywords>
        <ows:Keyword>WPVS</ows:Keyword>
        <ows:Type>String</ows:Type>
      </ows:Keywords>
      <ows:Fees>none</ows:Fees>
      <ows:AccessConstraints>none</ows:AccessConstraints>
    </ows:ServiceIdentification>
  + <ows:ServiceProvider></ows:ServiceProvider>
  − <ows:OperationsMetadata>
    − <ows:Operation name="GetCapabilities">
      − <ows:DCP>
        − <ows:HTTP>
```


Abb. 12: Ausschnitt aus einem XML-Dokument nach einem GetCapabilities Aufruf

Im Folgenden werden unterschiedliche Dienste vorgestellt, die für ein 3D-WebGIS relevant sind. Sie sind noch nicht vom OGC als offizieller Standard bestätigt worden, sondern befinden sich in unterschiedlichen Bearbeitungsstadien.

4.2 Standardisierte Dienste

4.2.1 Web 3D Service

Der Web 3D Service ist beim OGC momentan als „Discussion Paper“ gelistet. Zur Diskussion wurde er von der Special Interest Group (SIG) 3D der Geodateninfrastruktur in Nordrhein-Westfahlen (GDI NRW) eingereicht. Dieser Dienst ermöglicht die Visualisierung von 3D-Geodaten, indem er diese auf Anfrage als 3D-Szenegraphen (vgl. Abb. 5) liefert. Der Graph enthält visuell illustrierte 3D-Geodaten, aber nicht die Daten selbst. Die Originaldaten mit den semantischen Eigenschaften und Beziehungen werden nicht übertragen. Zur Visualisierung muss der Client ein entsprechendes Plug-In installiert haben, um aus dem Szenegraphen eine bildhafte Darstellung zu erzeugen bzw. zu rendern. Dadurch ermöglicht er eine anwenderspezifische Symbolisierung der Daten. Für die Umsetzung der Symbolisierung kann der bekannte Standard Styled Layer Descriptor verwendet werden. Um ihn für die Darstellung von dreidimensionalen Objekten verwenden zu können, müssen die Beschreibungen entsprechend für 3D-Daten erweitert werden. Nähere Informationen zu diesem Thema sind in [NEUBAUER & ZIPF 2007] zu finden. Der Szenegraph kann in Formaten wie Virtual Reality Modeling Language, welcher obligatorisch ist, oder Extensible 3D geliefert werden und bietet dadurch die Möglichkeit, einzelne Objekte auszuwählen und mehrere Szenegraphen vor dem Rendern zusammenzufügen. Der Zugriff auf ausgewählte Objekte wird mit Hilfe des bekannten Web Feature Service umgesetzt.

Da von dem Web-Service 3D-Szenegraphen geliefert werden, erfolgt die Anfrage mit dem Aufruf REQUEST=GetScene. Diesem fügt man in der bereits beschriebenen Weise unterschiedliche Parameter hinzu. Sie beziehen sich auf Angaben zum räumlichen Bezugssystem (Parameter SRS), die Begrenzung des Gebietes, welches dargestellt werden soll (umschließendes Rechteck, Parameter BBox), dem gewünschten Ausgabeformat (Parameter FORMAT) sowie die Definition der Kameraeinstellung. Dabei finden die Parameter (POI, PITCH, YAW, ROLL, DISTANCE, POC, AOV und BBOX) Anwendung, die im Punkt 3.4.3 beschrieben wurden. Die Definition der gewünschten Perspektive kann über die Angabe des Zielpunktes (POI) und die Winkel erfolgen oder das Projektionszentrum der Kamera (POC) und deren Blickrichtung. [KOLBE 2004]

Ein weiterer wichtiger Parameter betrifft die Angabe der Layer, die dargestellt werden sollen. Dafür sind die 3D-Objekte im Datenbestand entsprechenden Layern zugeordnet, um diese Auswahl, ähnlich dem Web Map Service, zu ermöglichen. Die Reihenfolge der Aufzählung der einzelnen Layer soll die Darstellungspriorität der Objekte festlegen. Dies wurde von dem WMS mit übernommen, da der W3DS eine größtmögliche Ähnlichkeit mit ihm haben soll. Die Darstellungsreihenfolge von Objekten macht aber für 3D-Szenen keinen rechten Sinn, da sie sich durch die Position des Beobachters in der Szene ergibt. Die vorgesehenen Parameter für die Operation GetScene zeigt Tab. 1 entsprechend der Spezifikation zum Web 3D Service.

Tab. 1: Parameterliste für die Operation GetScene des W3DS [OGC 2005a]

URL parameter	Required/ Optional/ Conditional	Description
VERSION=<version> version	R	requested
REQUEST=GetSceneoperation	R	requested
SRS=namespace:identifier	R	spatial reference system
POI=<point_of_interest>	C	x,y,z point coordinates according to SRS
PITCH=<pitch>	C	angle of inclination [degree]
YAW=<yaw>	C	azimuth [degrees]
ROLL=<roll>	O	rotation around viewing vector [degree]
DISTANCE=<distance>	C	distance POI to POC [meter]
POC=x, y, z	C	x,y,z coordinates of camera according to SRS
AOV=<angle_of_view>	C	angle of view [degree]
BBOX=xmin,ymin,xmax,ymax	R	2d bounding box
MINHEIGHT=<lower_limit>	O	displaying objects with height ≥ lower_limit according to SRS
MAXHEIGHT=<upper_limit>	O	displaying objects with height ≤ upper_limit according to SRS
LAYERS=<layer list>sets	O	comma separated list of 3D object sets
STYLES=<style list>	O	comma separated list of styles for each layer
FORMAT=<format>	R	MIME type of output
TIME=<date_and_time>	O	date and time
EXCEPTIONS=<excepttype>	O	exception format
TRANSLATE=x,y,z	C	translation vector that is applied to all 3D coordinates
ENVIRONMENT=on / off	O	switch on/off background elements like sky or light source
BGCOLOR=<color>	O	background color
BGIMAGE=<image url>	O	URL of background image

Ein W3DsS könnte wie folgt, aufgerufen werden:

```
„http://localhost:8080/CityServer3D/WMS?SERVICE=w3ds&VERS
ION=0.3.0&REQUEST=GetScene&SRS=epsg:16263&FORMAT=model/vr
ml&layers=dgm,buildings&layers=dgm,buildings&BBOX=62574.9
1,36549.58,63023.33,37102.43&PITCH=45&POI=62800,0,-
36800&YAW=0&POC=62300,200,-37505“
```

Der Server selbst wird in diesem Beispiel mit „localhost:8080//CityServer3D/WMS“ angesprochen. Um den richtigen Dienst anzufragen, ist er mit einer Version aufzuführen, „SERVICE=w3ds&VERSION=0.3.0“. Nun folgt die Angabe der Operation, die eine 3D-Szene liefern kann, „REQUEST=GetScene“. Bei einem Web 3D Service ist dafür die Operation GetScene verantwortlich. Daran schließen sich alle Parameter an, die die Daten oder deren Darstellung betreffen. So erfolgt die Angabe des räumlichen Referenzsystems, „SRS=epsg:16263“ und des Formates, indem die 3D-Szene gesendet werden soll, „FORMAT=model/vrml“. In diesem Beispiel wird eine VRML-Datei erzeugt, in der die Ausgangsdaten im Bezugssystem Gauß-Krüger, Zone 3 vorliegen sollen. Anschließend werden die darzustellenden Daten aufgezählt, „layers=dgm,buildings“. Die letzten Parameter betreffen die Definition des sichtbaren Bereichs über eine Boundig Box, „BBOX=62574.91,36549.58,63023.33,37102.43“, sowie die Definition der Perspektive mit Hilfe des Neigungswinkels „PITCH=45“, des Zielpunktes „POI=62800,0,-36800“, der Orientierung der Szene zur Nordrichtung „YAW=0“ und der Kameraposition „POC=62300,200,-37505“.

Die meisten räumlichen Referenzsysteme arbeiten mit großen Koordinatenwerten. Um eine Genauigkeit im Submeterbereich zu erreichen, muss der Ursprung der angeforderten Daten in den Ursprung der Bounding Box verschoben werden. Dabei wird eine Transformation in ein Bildkoordinatensystem vorgenommen, dass heißt, das georeferenzierte, linkshändige, kartesische Koordinatensystem wird in ein rechtshändiges, kartesisches Koordinatensystem überführt, wie es Abb. 13 zeigt. Diese Berechnung führt jede W3DS-Implementierung unterschiedlich durch, so dass man Szenegraphen von unterschiedlichen W3DS nicht ohne weiteres ineinander integrieren kann.

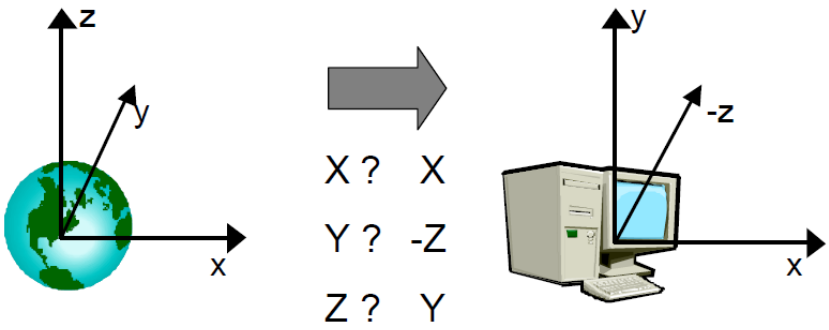

Abb. 13: Transformation innerhalb des W3DS [OGC 2005a]

Die Spezifikation des Web 3D Service basiert auf dem bekannten Standard Web Map Service, dem Web Terrain Service und der OpenGIS Web Service Commen Implementation Spezifikation. Im Gegensatz zu den ersten beiden Diensten bietet allerdings der W3DS volle Unterstützung der dritten Dimension. Dadurch soll ermöglicht werden, den W3DS mit relativ wenig Aufwand in bestehende Geodateninfrastrukturen zu integrieren. [KOLBE 2004]

Ein Beispiel für eine Anwendung, die den Web 3D Service bereits implementiert hat, ist der „Pilot 3D“. Er wird von der Initiative GDI-Nordrhein-Westfalen entwickelt.

4.2.2 Web Terrain Service

Der Web Terrain Service ist in vielen Aspekten gleich, wie der WMS konzipiert. Dadurch soll ermöglicht werden, beide Dienste so zu implementieren, dass sie Parameter des jeweils anderen Dienstes mit verwenden können. Das heißt, dass bei einer Anfrage an einen WTS Parameter einer vorangegangenen Anfrage an einen WMS der Server mit verarbeitet.

Ähnlich wie der WMS erstellt der WTS statische Bilder von dreidimensionalen Landschaften. Bei einer Anfrage werden daher nicht die Daten selbst, sondern ein zweidimensionales Rasterbild, in Formaten wie Portable Network Graphics (PNG), Graphics Interchange Format (GIF) oder Joint Photographic Expert Group (JPEG), von einer 3D-Szene an den Client gesendet. Es handelt sich hier um Ansichten auf georeferenzierte Daten. Sie fragt man mit dem Aufruf REQUEST=GetView ab.

Da es sich um Bilder handelt, kann sich der Nutzer nicht direkt in der Szene bewegen. Er kann nur indirekt mit der Anwendung interagieren. Bei jeder Bewegung des Nutzers,

die er in der Szene durchführt, muss ein neues gerendertes Bild an ihn gesendet werden, welches den geänderten Gegebenheiten entspricht Der Server übernimmt hier das Rendern der Daten. Dies hat den Vorteil, dass keine weiteren Installationen seitens des Clients notwendig sind. Ein Ziel dieser Spezifikation ist die Interoperabilität mit anderen vorhandenen Web Services. Damit soll der WTS ohne großen Aufwand in vorhandenen Systemlandschaften integriert werden können. [OGC 2003]

Tab. 2: Parameter der Operation GetView des WTS [OGC 2003]

URL parameter	Required/ Optional	Description
http://server_address/path/script?	R	URL prefix of server
VERSION=version	R	Request version
REQUEST=GetView	R	Request name
SRS=namespace:identifier	R	Spatial Reference Systems
POI=point_of_interest	R	x, y, z point in SRS units
PITCH=pitch	R	Angle of inclination
YAW=yaw	R	Azimuth
DISTANCE=dist_to_poi	R	Distance between the viewer and the POI in meters
AOV=angle_of_view intersecting the POIrepresenting the breadth of	R	The angle representing the breadth of landscape in the viewer's scene. Specifying AOV=0
BBOX=xmin,ymin,xmax,ymax	O	Minimum geographic extent in SRS units of the view (alternative to POI + DISTANCE)
Layers=layer_list	O	Comma-separated list of one or more map layers. Optional if SLD parameter is present.
Syles=style_list	O	Comma-separated list of one rendering style per requested layer. Optional if SLD parameter is present.
FORMAT	O	Mime type of requested response format
DEM=digital elevation model	O	Name expressing an available combination of elevation data and terrain model used to render a terrain view
TERRAIN=terrain base layerbase image layer to	O	Name expressing an available base image layer to underly any other layers in the terrain view
TRANSPARENT	O	Yes / No
BGCOLOR	O	Color to render any response pixel without terrain content
*SLD=styled_layer_descriptor document URL	O	Web-accessible URL of the SLD document
*SLD_Body=SLD text	O	text of StyledLayerDescriptor (subject to HTTP Get size limitations)
EXCEPTION=application/vnd.ogc.se_xml	O	The format in which exceptions are to be reported by the Map Server
QUALITY [0...100]the quality	O	An integer between 0 and 100 that specifies the quality (e.g. data resolution, rendering accuracy) of the view to be returned.
*SLD_Body parameter is used only with Web Terrain Services that support the Styled Layer Descriptor specification		

Die Hauptoperation ist GetView. Tab. 2 zeigt die Parameter, die bei einer Anfrage mit dieser Operation verwendet werden können. Über sie wird festgelegt, welche Daten wie angezeigt werden sollen. Für die Darstellung wird die SLD Spezifikation verwendet und für die Symbolisierung von 3D-Elementen entsprechend erweitert, wie es auch bei dem W3DS der Fall ist. Die gewünschte Perspektive legt man mit den Parametern fest, die in Punkt 3.4.3 beschrieben wurden (POI, PITCH, YAW, ROLL, DISTANCE, POC, AOV und BBOX). Sollte der WTS mit den geforderten Parametern kein Bild generieren können, da z. B. der gewünschte Sichtabstand nicht umsetzbar ist, so ist es dem Service möglich, angenäherte Werte zu verarbeiten. Der Server verwendet dann mögliche Werte, die den gewünschten Werten am nächsten kommen, ohne einen Fehler zu verursachen. [OGC 2003]

4.2.3 Web Perspective View Service

Die Spezifikation zum Web Perspective View Service befindet sich noch in der Entwurfsphase und soll den Web Terrain Service später ersetzen. Er soll, wie auch der WTS, das Pendant zum bekannten Web Map Service bilden. Genau wie der WMS liefert der WPVS fertig gerenderte Bilder, allerdings von 3D-Szenen, an den Client, so dass dieser keine weitere Software installieren muss. Jede Interaktion des Nutzers löst eine neue Anfrage (REQUEST=GetView) an den Server aus, und es wird eine neue Ansicht der Szene entsprechend den Parametern gerendert und an den Client gesendet. Bei jeder Anfrage bestimmt der Nutzer die perspektivische Ansicht auf die Daten, welche Daten er angezeigt haben möchte und wie deren Symbolisierung sein soll. Die Perspektive wird mit Hilfe der Parameter definiert, die in Punkt 3.4.3 (POI, PITCH, YAW, ROLL, DISTANCE, POC, AOV und BBOX) beschrieben wurden.

Der WPVS stellt neben GetCapabilities die Operationen GetView, GetDescription und GetLegendGraphic zur Verfügung, wobei die letzten beiden nicht zwingend implementiert werden müssen. GetView liefert ein gerendertes Bild der Ansicht, die sich aus den angegebenen Parametern ergibt. Das Bild wird in dem Format als Antwort zurückgesendet, wie es der Nutzer gefordert hat, z. B. als Portable Network Graphics (PNG) oder Joint Photographic Expert Group (JPEG).

Eine Anfrage nach einer dreidimensionalen Ansicht an einen WPVS könnte wie folgt aussehen:

```
„http://localhost:8080/deegree-
wpvs/services?request=GetView&VERSION=1.0.0&BOUNDINGBOX=
423750,4512700,425500,4513900&DATASETS=Buildings,satelli
te_images&ELEVATIONMODEL=saltlake_dem&CRS=EPSG:26912&STY
LES=default&WIDTH=800&HEIGHT=600&SCALE=1.0&OUTPUTFORMAT=
image/jpg&EXCEPTIONFORMAT=INIMAGE&background=cirrus&ROLL
=0&AOV=60&POI=424750.0,4513400.0,1350&YAW=20&PITCH=15&DI
STANCE=2500“
```

Der Server wird in diesem Beispiel mit „localhost:8080/deegree-wpvs/services“ aufgerufen und der Dienst angesprochen. Als nächstes gibt man die Operation an, die der Service ausführen soll und deren Version, „request=GetView&VERSION=1.0.0“. Bei dem WPVS handelt es sich um die GetView-Operation. Mit Hilfe einer Bounding Box ist im Anschluss der Bereich anzugeben, der aus dem Datenbestand geladen werden soll, „BOUNDINGBOX=423750,4512700,425500,4513900“. Nun schließt sich die Angabe der einzelnen Datensätze bzw. Layer an, die man visualisieren möchte, „DATASETS=Buildings,satellite_images“. Hier werden Gebäude und ein Luftbild dargestellt. Desweiteren ist das Digitale Geländemodell aufzuführen, „ELEVATIONMODEL=saltlake_dem“. Der Parameter CRS gibt das zu verwendende räumliche Bezugssystem an, „CRS=EPSG:26912“. Dieser ist für die WPVS-Spezifikation vorgesehen, aber in ihr noch nicht mit aufgeführt. Mit Hilfe von STYLES kann man vom Server vorgegebene Stile für die Daten verwenden, „STYLES=default“. Nun folgen Informationen, die das zu rendernde Bild betreffen. Es werden dessen Breite und Höhe sowie dessen Maßstab und das Dateiformat angegeben, “WIDTH=800&HEIGHT=600&SCALE=1.0&OUTPUTFORMAT=image/jpg“. Somit bekommt der Nutzer ein JPEG-Bild mit einer Breite von 800 Pixeln und einer Höhe von 600 Pixeln geliefert. Falls der Server das geforderte Bild nicht erstellen kann und einen Fehler verursacht, muss er wissen, in welchem Format er die Fehlermeldung ausgeben muss. Im Beispiel wird diese als Bild ausgegeben, „EXCEPTIONFORMAT=INIMAGE“. Um der Szene einen Hintergrund hinzu zu fügen, wird dieser ebenfalls mit angegeben, „background=cirrus“. Die Definition der Perspektive erfolgt, wie auch beim W3DS und WTS, „ROLL=0&AOV=60“ und „POI=424750.0,4513400.0,1350&YAW=20&PITCH=15&DISTANCE=2500“. Das Ergebnis der Anfrage dieses Beispiels ist in Abb. 14 zu sehen.

Abb. 14: Ergebnis einer Beispielanfrage an einen WPVS

Tab. 3 zeigt die Parameter, die mit der GetView-Operation an den Server übergeben werden können. Auch der WPVS ist in der Lage, Näherungswerte zu den angegebenen Parametern zu verwenden und mit diesen das Bild zu generieren. [OGC 2005b]

Auf die Anfrage mit der Operation GetDescription erhält man eine Beschreibung der Datensätze, die man angefordert hat, um in erster Linie deren Darstellung zu kontrollieren. Momentan ist es noch strittig, ob die Bezeichnung für diese Operation beibehalten wird. Mit Hilfe der GetLegendGraphic-Operation kann eine Legende entsprechend der Daten und deren Darstellung angefordert werden.

Tab. 3: Parameter der Operation GetView des WPVS nach [OGC 2005b]

URL parameter a **with example**	**Required/ Optional**	**Description**
service=WPVS	R	Service type or profile identifier
request= GetView	R	Operation name
version=1.0	R	Specification profile and schema version
OutputFormat=TBD	R	MIME type of format in which output data should be encoded
Layers=TBD,TBD or Datasets=TBD,TBD	R	Ordered list of identifiers of desired layers or datasets
Styles=TBD,TBD	R	Ordered list of identifiers desired styles for corresponding layers or datasets b
BoundingBox=TBD	R	BoundingBox surrounding desired part of data, in desired CRS
OtherDimensions=TBD	O	Name(s) and value(s) of other dimensions desired, selecting from types and values are specified in service metadata (Capabilities) (TBR)
store=true	O	Boolean (true or false value), indicating if output data to be stored on a remote resource or returned directly in the response.
Width=480	R	Width of desired output, in pixels
Height=640	R	Height of desired output, in pixels
Transparent=false	O	Boolean (true or false) indicating if background transparency desired
BackgroundColor=TBD	O	Background color desired TBD
ExceptionFormat= INIMAGE	O	Reference to format in which operation exceptions should be returned
SLD=TBD	O	URL reference to Styled Layer Descriptor to be used for TBD
SLD_Body=TBD	O	Text of Styled Layer Descriptor to be used for TBD
POI=TBD,TBD,0	R	Position of point of interest in centre of desired perspective view, in CRS of BoundingBox
Pitch=60	R	Pitch angle inclination from horizontal at POI, of centre of desired view, in degrees
Yaw=90	R	Yaw or azimuth angle from North at POI, of centre of desired view, in degrees
Roll=10	O	Roll angle from vertical at POI, around desired viewing direction, in degrees
Distance=300	R	Distance from POI of view perspective centre, in metres
AOV=45	R	Angle at centre of projection of width of desired portrayal view, centred on POI, in degrees
ElevationModel=TBD	O	Identifier of available digital elevation model to be used in desired portrayal view
Quality=80	O	Integer from 0 and 100 that specifies desired quality of portrayal view (e.g. data resolution, rendering accuracy)
VendorSpecific Parameters=TBD	O	Values of vendor specific parameters (TBR)

a All parameter names are here listed using mostly lower case letters. However, any parameter name capitalization shall be allowed in KVP encoding, see Subclause 11.5.2 of [OGC 05-008r1].

b If all layers are to be shown using the default style, either the form "STYLES=" or "STYLES=,,," may be used.

Im Zuge des Entwurfs des WPVS werden noch weitere Spezifikationen, die eng mit ihm verknüpft sind, entwickelt. Es handelt sich dabei um sogenannte Module, die es ermöglichen sollen, Teile des WPVS in vorhandene Strukturen zu integrieren.

Das Basic Portrayal Additions (BPA) Modul ermöglicht es, perspektivische Ansichten auf Daten aus verschiedenen Quellen in Form einer Rastergrafik zu rendern und an den Client zu senden, ähnlich der GetView-Operation. Dabei kann man aber nur auf die perspektivische Ansicht und die Auswahl der Daten Einfluss nehmen. [OGC 2005c]

Mit dem Client Styling Additions (CSA) Modul kann der Nutzer die Darstellung bzw. Symbolisierung der angeforderten Daten selber bestimmen. Er basiert auf der bekannten Spezifikation der Styled Layer Descriptor. [OGC 2005d]

Um nur Teile eines Datensatzes anzufordern, kann man das Get Subset Additions (GSA) Modul verwenden. Das Perspective View Additions (PVA) Modul ermöglicht einem anderen Dienst, z. B. einem WMS, perspektivische Ansichten auf Daten anzubieten. Die Definition der Perspektive erfolgt mit Hilfe der im Punkt 3.4.3 erläuterten Parameter. [OGC 2005e, 2005f]

4.3 Standardisierte Formate

4.3.1 Allgemeines

Der Anspruch der Interoperabilität von Systemen hängt eng mit den jeweils verwendeten Datenformaten zusammen. Dabei ist es in erster Linie wichtig, Daten so auszutauschen, dass keine Datenverluste auftreten. Um dies zu gewährleisten, sollen standardisierte Datenformate verwendet werden. Dies führt neben dem verlustfreien Datenaustausch auch dazu, dass man Daten unterschiedlicher Quellen gemeinsam in einer Anwendung visualisieren kann.

4.3.2 VRML

Virtual Reality Modeling Language 97 wurde von der ISO 1997 spezifiziert und löste damit den bis dahin geltenden Standard VRML 1.0 ab. Hierbei handelt es sich weniger um ein Dateiformat im herkömmlichen Sinne, sondern eher um eine Beschreibungssprache für statische dreidimensionale Objekte und dynamische,

interaktive virtuelle Welten. Sie bietet neben der geometrischen Beschreibung der Objekte auch die Möglichkeit, das Laufzeitverhalten von Objekten festzulegen. VRML 97 gilt als Vorreiter für die 3D-Technologie im Internet. Sie ist plattformunabhängig und ein reines Textformat. Mit dieser Version ist es möglich, komplexe und dynamische 3D-Szenen zu beschreiben. Auf diese Weise können Animationen in einer Szene genauso mit abgelegt werden, wie Texturen für geometrische Objekte, Referenzen zu externen Elementen, Lichtquellen und dreidimensionale Klänge. [FREIWALD & JANY 2004]

Als Grundlage zur Beschreibung der einzelnen Bestandteile einer 3D-Szene dient das Konzept des Szenegraphen, der in Punkt 3.3.1 schon kurz erläutert wurde. Zur Visualisierung des Graphen ist allerdings ein Plug-In für den Internetbrowser erforderlich. Das Plug-In rendert den Szenegraphen in zweidimensionale Bilder, die der Bildschirm dem Nutzer anzeigen kann. Bei vielen Anwendungen muss dabei auf die richtige Kombination von Plug-In und Browser geachtet werden, um ein fehlerfreies Funktionieren zu erreichen. Wie auch bei dem W3DS verwendet VRML ein dreidimensionales, rechtshändiges kartesisches Koordinatensystem. Für die Abbildung von georeferenzierten Geometrien muss vorher eine entsprechende Transformation der Daten durchgeführt werden (vgl. Abb. 13).

Eine solche VRML-Datei besteht im Wesentlichen aus drei Teilen. Dem Header, der Informationen wie Version und Codierung enthält, den Knoten, die die unterschiedlichen Elemente der Szene beinhalten, und den Kommentaren. Eine einfache VRML-Datei, die ein Bild anzeigt [POMASKA 2007], hat den folgenden Inhalt:

```
#VRML V2.0 utf8
NavigationInfo {type ["EXAMINE"]
}
Viewpoint {position 0 0 5
}
DEF World Transform {
  children [
    Shape {
      geometry Sphere { }
      appearance Appearance {
     material Material { diffuseColor 1.0 0.0 0.0 }
        texture ImageTexture { url "earth-topo.jpg"}
      }
    }
  ]
}
```

Öffnet man diese Datei mit einem VRML-Plug-In so erhält man das Bild einer Weltkugel, wie es Abb. 15 zeigt.

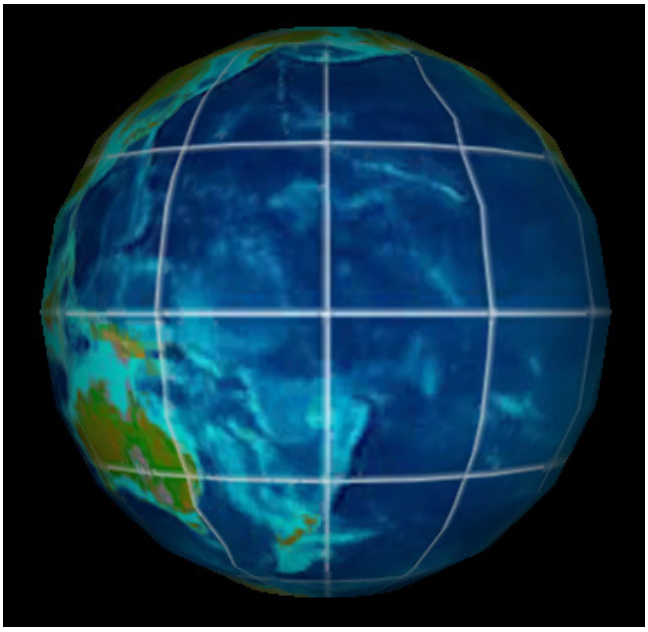

Abb. 15: Anzeige einer VRML-Datei [POMASKA 2007]

Um die Geometrien der Objekte zu beschreiben, verwendet VRML Punkte, Linien, Flächen und Körper. Diese werden mit Knotenpunkten (Vertices) und entsprechenden Index-Listen auf diese Vertices definiert. Die Verfahrensweise entspricht dem „Index Face Set“, wie es im Punkt 3.2.1 beschrieben wurde und Abb. 4 zeigt. Durch dieses Konzept können auch topologische Beziehungen zwischen den Objekten abgebildet werden. Für bestimmte Objekte stehen von dem Standard vordefinierte Knoten zur Verfügung, wie z. B. für die Abbildung von Höhenmodellen. [FREIWALD & JANY 2004]

Neben der Notwendigkeit von Plug-Ins hat VRML noch einen weiteren Nachteil. So können VRML-Dateien unkontrolliert gespeichert und weitergegeben werden, sodass Urheberrechte und Datensicherheit gefährdet sein können.

Das Web3D-Konsortium wurde zur Etablierung und Entwicklung von freien Web3D-Technologien gegründet. Eine Arbeitsgruppe des Konsortiums begann 1998 GeoVRML zu entwickeln. Aufbauend auf VRML soll es die Georeferenzierung von Objekten, sowie eine interaktive Darstellung komplexer Geländemodelle ermöglichen und somit die Schwächen von VRML bezüglich Geodaten ausgleichen. Im VRML-Standard kann man lediglich ein kartesisches Koordinatensystem als Bezugsraum verwenden und für die Angabe von Koordinaten Fließkommazahlen mit einfacher Genauigkeit. Seit dem Jahr 2000 gilt GeoVRML 1.0 als optionale Komponente des VRML97-Standards. Mit

ihm können geografische und projektive Koordinatensysteme verwendet werden. Allerdings sind für die Visualisierung in einem VRML-Viewer spezielle Java-Klassen (GeoTransform-Java Package) notwendig. [FREIWALD & JANY 2004]

4.3.3 X3D

Extensible 3D basiert auf dem bekannten Standard Extensible Markup Language. Es ist ein Standard für dreidimensionale Grafiken und beinhaltet frei navigierbare Szenebeschreibungen. X3D ist der Nachfolger von VRML97, der als Untermenge des X3D weiterhin besteht. So soll die Funktionsfähigkeit bestehender Anwendungen nicht beeinträchtigt werden. Er ist ebenso plattformunabhängig und ein reines Textformat, wie sein Vorgänger, sowie ähnlich im Aufbau. Kreiert vom Web3D-Konsortium wurde er 2004 zum ISO-Standard erklärt.

Hauptmerkmale des X3D sind der modulare Aufbau, wodurch er erweiterbar ist, und die XML-Notation, die eine bessere Integration von 3D-Szenen in multimediale Webseiten ermöglicht. Eine einfache X3D-Datei, die ein Bild anzeigt [POMASKA 2007], hat den folgenden Inhalt:

```
<?xml version="1.0" encoding="UTF-8"?>
<X3D profile="Full">
 <head>
    [. . .]
 </head>
 <Scene>
   <NavigationInfo type="&quot;EXAMINE&quot; "/>
   <Viewpoint description="" position="0.0 0.0 5.0"/>
   <Transform DEF="World">
     <Shape>
       <Appearance>
         <Material diffuseColor="1.0 0.0 0.0"/>
         <ImageTexture url="&quot;earth-topo.jpg&quot; "/>
       </Appearance>
       <Sphere/>
     </Shape>
   </Transform>
 </Scene>
</X3D>
```

Das Ergebnis einer Anzeige mit einem entsprechenden Plug-In enthält ebenfalls das Bild einer Weltkugel, wie es Abb. 15 zeigt.

Die unterschiedlichen Funktionalitäten, die X3D bietet, sind in verschiedene Komponenten gruppiert, welche in Support-Levels zusammengefasst werden. Weiterhin sieht die Spezifikation Profile vor, die Komponenten spezieller Levels vereinigt.

Dadurch ist es möglich, nur Teilmengen des Standards zu implementieren, aber dennoch konform mit ihm zu sein. Durch den modularen Aufbau können immer weitere Support-Levels hinzugefügt werden und der Standard bleibt so flexibel einsetzbar. [FREIWALD & JANY 2004]

X3D definiert Systeme, die 3D-Grafiken und Multimedia integrieren können. Wie auch VRML, ordnet X3D die Objekte in einer Baumstruktur, dem Szenegraphen, an Er benutzt an Stelle von Knoten Elemente und statt Datenfeldern Attribute. Neben der Geometrie der Objekte wird deren Erscheinungsbild mit abgelegt sowie der Hintergrund und die Umgebung der dargestellten Objekte in der 3D-Szene. Ein weiterer Vorteil gegenüber VRML ist, dass die Daten komprimiert werden und sich so die Datenübertragung zum Client verbessert. [POMASKA 2007]

4.3.4 CityGML

City Geography Markup Language ist sowohl ein Austauschformat als auch ein semantisches Objektmodell für 3D-Objekte im städtischen Raum. Es basiert auf dem XML-Datenmodell und entspricht der Spezifikation Geography Markup Language in der Version 3.1.1. Somit wird in den CityGML-Dateien die GML-Notation verwendet. Der folgende Auszug aus einer solchen Datei [OGC 2008] zeigt, wie aus zwei unterschiedlichen Flächen ein Gebäude zusammengesetzt wird, wie es in Abb. 16 zusehen ist:

```
<bldg:Building>
  [. . .]
  <bldg:lod2Solid>
    [. . .]
    <gml:surfaceMember>
      <gml:Polygon gml:id="wallSurface4711">
        <gml:exterior>
          <gml:LinearRing>
            <gml:pos srsDimension="3">32.0 31.0
              2.5</gml:pos>
            [. . .]
          </gml:LinearRing>
        </gml:exterior>
      </gml:Polygon>
    </gml:surfaceMember>
  </bldg:lod2Solid>
  [. . .]
</bldg:Building>
[. . .]
<bldg:Building>
  [. . .]
  <bldg:lod2Solid>
```

```
        [. . .]
        <gml:surfaceMember xlink:href="#wallSurface4711"/>
        [. . .]
      </bldg:lod2Solid>
      [. . .]
    </bldg:Building>
```

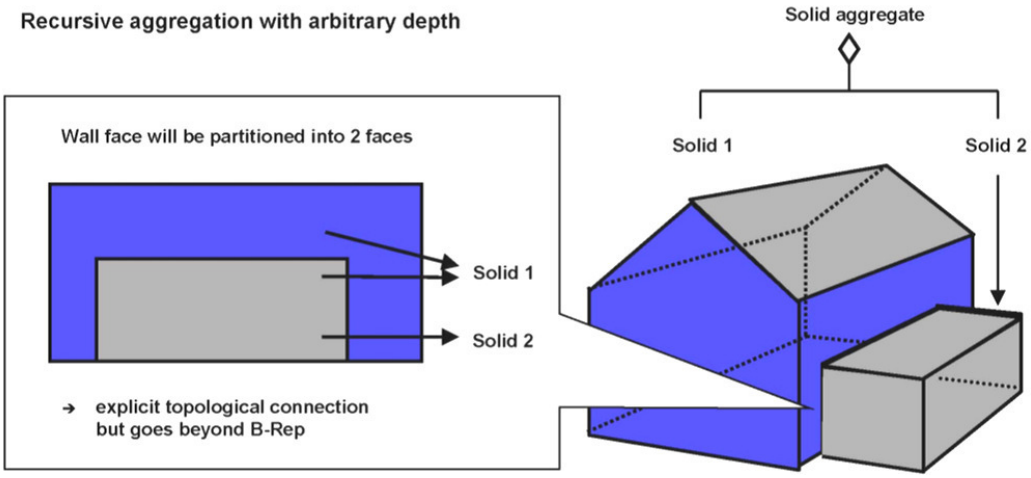

Abb. 16: Zusammensetzen eines Gebäudes aus Flächen nach der CityGML-Spezifikation [OGC 2008]

CityGML ermöglicht die Modellierung von Gebäuden, Straßen, Vegetation und weiterer semantischer und topologischer Informationen, die zu einem Stadtmodell gehören, sowie von Höhenmodellen. Auf diesem Wege sollen Stadtmodelle standardisiert werden, so dass ein verlustfreier Austausch von Daten gewährleistet werden kann. Weiterhin wird dadurch die Verwendung von einheitlichen Anwendungen zur Visualisierung und Verarbeitung der Stadtmodelle möglich. Natürlich kann auf diesem Weg der Mehrwert dieser Datenbestände gesteigert werden. [OGC 2008]

Der Standard CityGML verfolgt einen generellen Modellierungsansatz, so dass er mit seinen verschiedenen Modulen einen breiten Anwendungsbereich abdeckt. Darunter finden sich die Bereiche der Stadt- und Landschaftsplanung, Architektur, Tourismus, Umwelt, Telekommunikation, Katastrophenmanagement und Immobilienwirtschaft sowie Navigation und 3D-Simulationen. Für die einzelnen Anwendungsgebiete sind die definierten Klassen und Relationen in verschiedene Module gruppiert, wie es Abb. 17 zeigt.

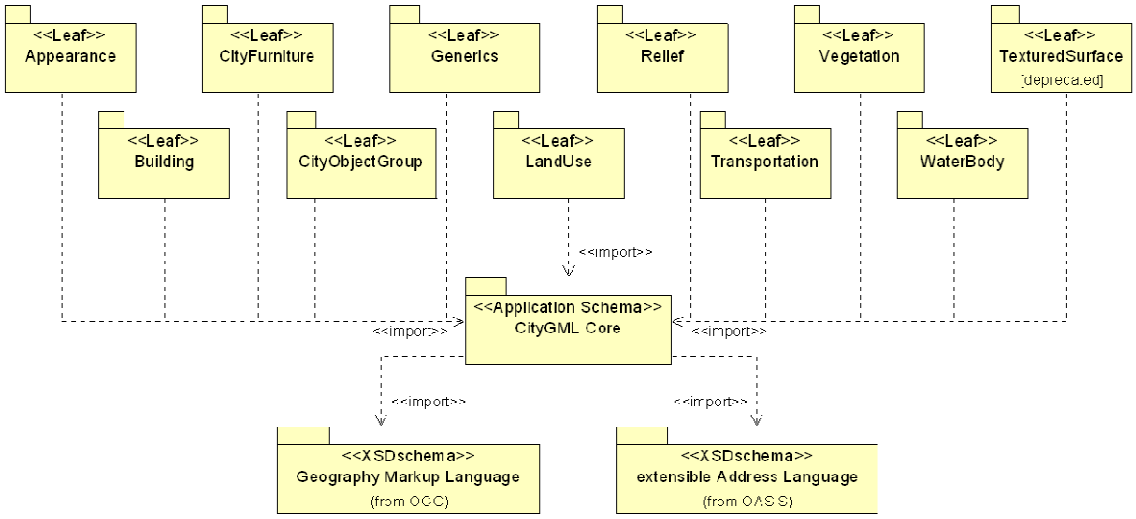

Abb. 17: UML-Diagramm der einzelnen Module von CityGML [OGC 2008]

Mittels dieser Module können die jeweils notwendigen topografischen Objekte modelliert werden. Die einzelnen Klassen und Relationen in den Modulen ermöglichen die Beschreibung der geometrischen, semantischen und topologischen Eigenschaften von Objekten. Aber auch das Aussehen (Darstellungseigenschaften) und Verhalten (Materialeigenschaften, z. B. Absorptionsverhalten) der Objekte kann gestaltet werden. Durch diese modulare Struktur ist CityGML sehr flexibel einsetzbar. So ist es für kleine Modelle ohne Topologie und geringer Semantik genauso geeignet, wie für sehr komplexe und große Modelle, die volle Topologie und Semantik enthalten. [OGC 2008]

Die Spezifikation zu CityGML umfasst ein Geometriemodell und ein thematisches Modell, welche jeweils auf die Elemente des anderen zugreifen. Dies ermöglicht eine homogene und konsistente Definition der topologischen und geometrischen Eigenschaften von Objekten. Wie schon erwähnt, werden diese Modelle aus einzelnen Klassendefinitionen gebildet. Basierend auf einem objektorientierten Ansatz, kommt die Vererbung zum Einsatz. Die Basisklasse, CityObject, vererbt ihre Eigenschaften und Methoden an alle Klassen, die von ihr abstammen. Somit müssen für die unterschiedlichen Objektgruppen nur die für sie spezifischen Elemente modelliert werden. Die allgemeinen Eigenschaften werden von der Basisklasse übernommen.

Für Standardobjekte eines Stadtmodells, wie z. B. Gebäude, sind in der Spezifikation bereits fertige Klassen vorgesehen. Dabei nutzt das thematische Modell das Geometriemodell für seine unterschiedlichen thematischen Felder. Das bedeutet, dass alle Module das gleiche Geometriemodell nutzen. Die anderen Attribute der Objekte

entsprechen aber den jeweiligen thematischen Modellen der einzelnen Module, wie es die Abb. 17 zeigt. Das Geometriemodell ist in dem „CityGML Core“-Modul enthalten. Für noch nicht vordefinierte Objekte ist ein allgemeines Objekt („Generics“) vorgesehen, welches zu deren Definition verwendet werden kann.

Für eine effiziente Visualisierung und Datenanalyse bietet CityGML die Möglichkeit, verschiedene Levels of Detail für die einzelnen Objekte zu definieren. Dies geschieht auf der Basis der fünf statischen LoDs, die in Punkt 3.3.3 erläutert wurden. Die Verbindung und Zergliederung der einzelnen Objekte für die unterschiedlichen LoDs werden von CityGML unterstützt, indem diese Verbindungen explizit abgelegt werden. Eine Aufstellung der einzelnen Detailstufen zeigt Tab. 4. Sie entsprechen den einzelnen Beispielbildern von Abb. 9.

Tab. 4: Generalisierung und Genauigkeiten der LoDs in CityGML [OGC 2008]

	LOD0	**LOD1**	**LOD2**	**LOD3**	**LOD4**
Model scale description	regional, landscape	city, region	city districts, projects	architectural models (out-side), landmark	architectural models (interior)
Class of accuracy	lowest	low	middle	high	very high
Absolute 3D point accuracy (position / height)	lower than LOD1	5/5m	2/2m	0.5/0.5m	0.2/0.2m
Generalisation	maximal generalisation (classification of land use)	object blocks as generalised features; > 6*6m/3m	objects as generalised features; > 4*4m/2m	object as real features; > 2*2m/1m	constructive elements and openings are represented
Building installations	-	-	-	representative exterior effects	real object form
Roof form/structure	no	flat	roof type and orientation	real object form	real object form
Roof overhanging parts	-	-	n.a.	n.a.	Yes
CityFurniture	-	important objects	prototypes	real object form	real object form
SolitaryVegetationObject	-	important objects	prototypes, higher 6m	prototypes, higher 2m	prototypes, real object form
PlantCover	-	>50*50m	>5*5m	< LOD2	<LOD2
… to be continued for the other feature themes					

4.3.5 KML

Keyhole Markup Language (KML) ist eine, auf XML basierende, Beschreibungssprache für Datenmodelle zur Betrachtung im „Keyhole Earth Viewer“. Die kalifornische Firma *Keyhole*, auf die der Name von KML zurückgeht, vermarktete Satellitenbilder und entwickelte diesen Viewer zur Betrachtung von Satellitenbildern via Internet. 2004 kaufte *Google Inc*. die Firma und vertreibt den Viewer seitdem unter dem Namen „Google Earth“. Diese, für private Nutzer kostenlose Anwendung bietet jedem die Möglichkeit, sich Zugang zu Geodaten zu verschaffen. Seit 2007 zählt KML zu den offiziellen Standards des OGC. [POMASKA 2007]

KML ermöglicht die Beschreibung und Speicherung von geografischen Daten für die Darstellung in einem sogenannten Earth Viewer, wie Google Earth. Objekte können mit Hilfe von Icons, Markern, Links oder ähnlichem beschrieben und auf der Erdoberfläche dargestellt werden. Wichtigste Anwendung ist dabei die Visualisierung von dreidimensionalen Daten. Somit bietet dieses Dateiformat auch die Möglichkeit, Informationen zu Blickwinkeln, Kamerapositionen usw. abzuspeichern. Der Raumbezug der darzustellenden Daten wird über orientierte Bounding Boxen und geografische Koordinaten realisiert. Für die Geometriebeschreibung stehen die Basiselemente Punkt, Linie und Polygon zur Verfügung, die in jeglicher Kombination verwendet und zu Collections gruppiert werden können. Die Einbindung von Bildern ist ebenfalls möglich.

Die Originaldaten müssen nicht zwangsläufig in einer KML-Datei abgelegt werden. In diesem Fall werden in der KML-Datei Elemente vom Typ „Model“ verwendet, welche Links zu externen sogenannten COLLADA-Dateien enthalten. COLLAborative Designe Activity (COLLADA) wurde ursprünglich als Format für die Spielkonsole „Playstation 3“ von Sony Entertainment entwickelt. Inzwischen gilt es aber als offenes und offizielles Austauschformat für Assets. Assets sind im 3D-Bereich Modelle, Texturen, Sounds usw. und neben weiterem Programmcode in COLLADA-Dateien enthalten. Durch die Verweise auf externe Dateien wird beim Laden einer KML-Datei das Laden der gewünschten Daten veranlasst, ohne das der Nutzer die Daten selbst erhält. [POMASKA 2007]

Das folgende Beispiel [POMASKA 2007] zeigt eine solche KML-Datei:

```
<?xml version='1.0' encoding='UTF-8'?>
<kml xmlns='http://earth.google.com/kml/2.1>
  <Placemark>
    <name>Martinikirche</name>
    <Style id='default'></Style>
    <Model>
      <altitudeMode>relativeToGround</altitudeMode>
      <Location>
        <longitude>8.9155102</longitude>
        <latitude>52.2883201</latitude>
        <altitude>0.000000000000</altitude>
      </Location>
      <Orientation>
        <heading>0</heading><tilt>0</tilt><roll>0</roll>
      </Orientation>
      <Scale>
        <x>1.0</x><y>1.0</y><z>1.0</z>
      </Scale>
      <Link>
        <href>models/martinikirche.dae</href>
      </Link>
    </Model>
  </Placemark>
</kml>
```

Das Objekt wird zuerst verortet und anschließend der Link für die externe Datei (eine COLLADA-Datei) aufgeführt, in der das darzustellende Modell abgelegt ist. In „Google Earth“ ergibt sich die Ansicht, wie sie Abb. 18 zeigt.

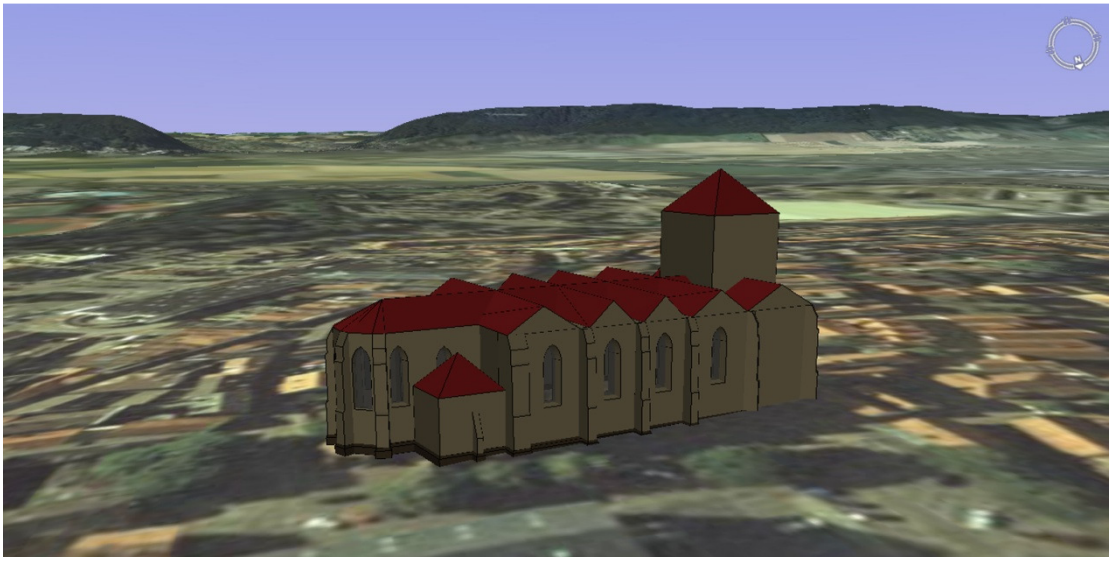

Abb. 18: Martinikirche in Minden in "Google Earth" [POMASKA 2007]

Viele Städte bieten inzwischen ihre 3D-Stadtmodelle als KML-Dateien an. Durch die weite Verbreitung von „Google Earth“ kann sich somit fast jeder die Daten anschauen. Dabei müssen die Städte keinen eigenen Webdienst für die Visualisierung der Modelle anbieten.

5 Lösungsansatz für ein 3D-WebGIS

5.1 Theoretische Ansätze

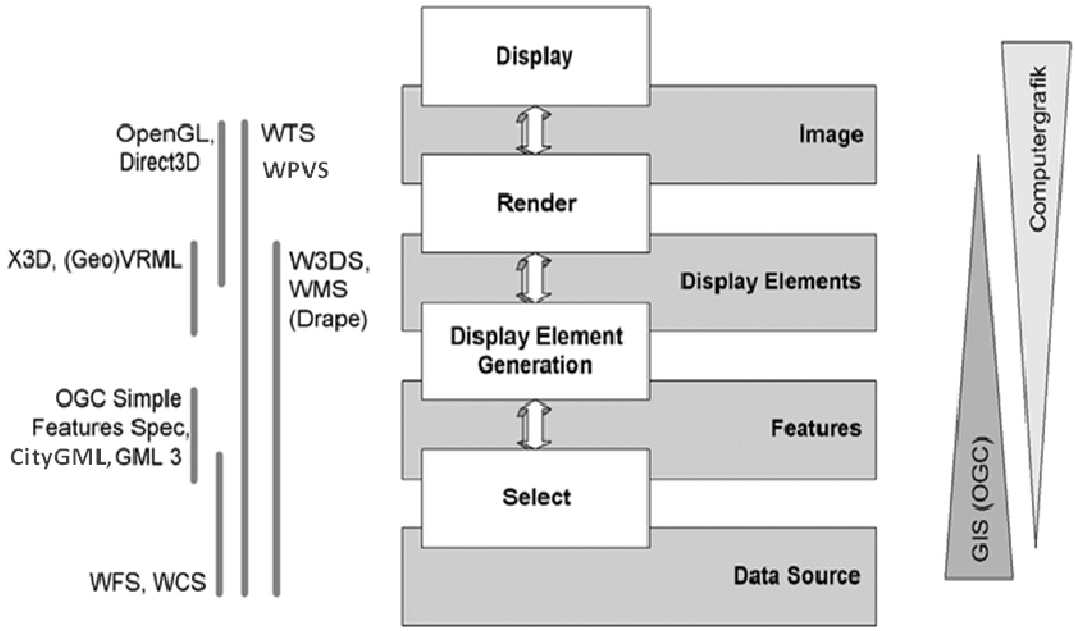

Abb. 19: OGC-Modell des Visualisierungsprozesses nach [SCHMIDT, MAY et al. 2006]

Ordnet man die unterschiedlichen Dienste und Formate den einzelnen Schritten in der Visualisierungspipeline (siehe Punkt 3.4.2) zu, so ergibt sich nach [SCHMIDT, MAY et al. 2006] ein Modell des Visualisierungsprozesses, wie es Abb. 19 zeigt. Die Grafik belegt, dass alle Bereiche des Visualisierungsprozesses von standardisierten Diensten und Formaten abgedeckt werden. Somit ist es möglich, Anwendungen zu entwickeln, die auf diesen basieren.

Je nach Verteilung der einzelnen Verarbeitungsschritte auf den Server oder den Client entstehen verschiedene Systemarchitekturen. Grundlegend kann man drei verschiedene Fälle unterscheiden. In der Abb. 20 sind die unterschiedlichen Verteilungen der Prozesse dargestellt. Welche Möglichkeit für ein 3D-WebGIS Verwendung findet, hängt immer von den jeweiligen Anforderungen an die Anwendung und von den darzustellenden Daten ab.

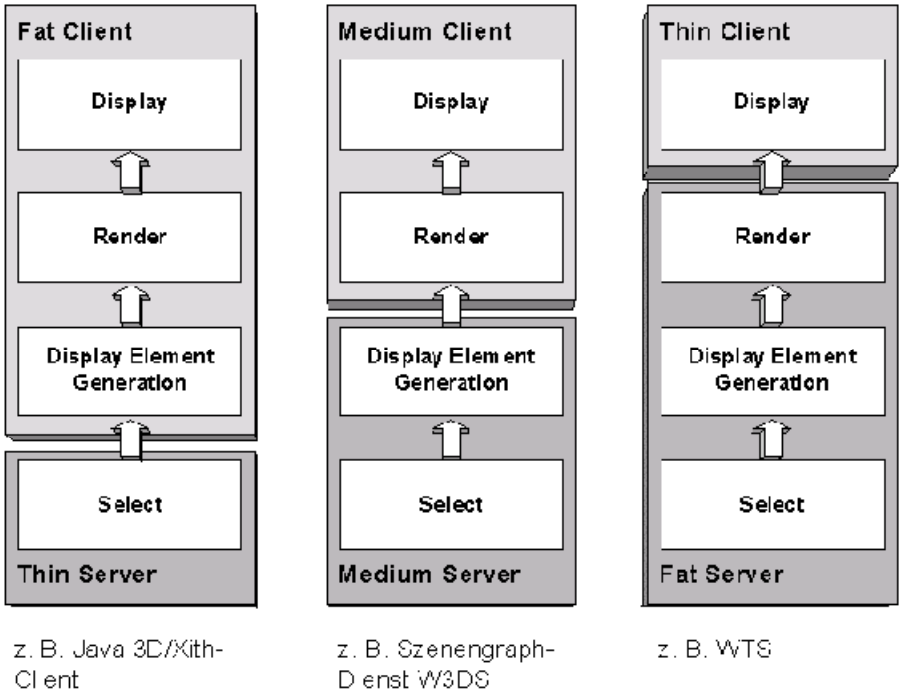

Abb. 20: Unterschiedliche Verteilungsvariante der Visualisierungsprozesse [SCHMIDT, MAY et al. 2006]

Die erste Möglichkeit einer Umsetzung besteht aus einem „Fat Client“ und einem „Thin Server“. Der Server liefert lediglich die selektierten Daten, z. B. mit Hilfe eines Web Feature Services. Die Daten können in einem XML-basierenden Format, z. B. CityGML, an den Client überliefert werden. Der Client erarbeitet die Szenebeschreibung und das anschließende Rendern mit einer geeigneten Anwendung, einem Viewer, selbst und zeigt dem Nutzer das Ergebnis an. Die Vorteile dieser Möglichkeit bestehen darin, dass die Umsetzung von Visualisierung sowie Interaktionen vom Anwender sehr frei gestaltet werden können und es die Darstellung sehr komplexer Modelle ermöglicht. Nachteilig ist für den Anwender die Installation von spezieller Software, die diese Aufgaben bearbeiten kann und dass meist die Originaldaten dem Nutzer zur Verfügung gestellt werden müssen.

Im zweiten Fall wird von einem „Medium Client“ und einem „Medium Server“ gesprochen. Der Server erledigt dabei neben der Selektion auch die Generierung der 3D-Szene, z. B. mit Hilfe eines W3DS. Den fertigen Szenegraphen übermittelt der Server anschließend an den Client, in Form einer X3D- oder VRML-Datei. Der Szenegraph muss nun vom Client gerendert und angezeigt werden. Dafür verwendet man meistens Plug-Ins für Webbrowser. Allerdings können hier, im Gegensatz zum ersten Fall, Standard Plug-Ins verwendet werden. Vorteil dieser Architektur ist, dass das

Real-Time-Rendering dem Nutzer ein hohes Maß an Interaktionen bietet und die Originaldaten beim Dienstanbieter verbleiben. Der Nachteil, die Installation von Software, in diesem Fall Plug-Ins, bleibt allerdings.

Bei der letzten Variante werden die meisten Aufgaben vom Server bearbeitet und man spricht von einem „Thin Client“ und einem „Fat Server“. Nach der Auswahl der Daten erstellt der Server eine 3D-Szene und rendert diese entsprechend der übermittelten Parameter für die perspektivische Ansicht der Daten. An den Client wird lediglich ein Bild dieser Szene, z. B. als JPEG- oder PNG-Datei übermittelt. Dieses Verfahren bietet ein WPVS bzw. WTS an. Der Vorteil dieser Variante ist, dass der Client ohne weitere Software auskommt. Allerdings muss für jede Interaktion des Anwenders ein neues Bild vom Server gerendert und an den Client übermittelt werden, was die Interaktionsmöglichkeiten des Nutzers beschränkt.

Betrachtet man die drei Varianten unter dem Aspekt, welche Möglichkeiten der Nutzer mit der entsprechenden Anwendung hat und welche technischen Voraussetzungen dafür erforderlich sind, so ergibt sich folgendes.

Mit einer Systemarchitektur, die einen „Fat Client“ und einen „Thin Server“ vorsieht, hat ein Nutzer die meisten Möglichkeiten. Er kann mit der Anwendung in direkte Interaktion treten, da er sich direkt in einer 3D-Welt bewegen kann. Durch die gegebene Struktur ist es machbar, unterschiedliche Datensätze zu kombinieren und deren Darstellung selber zu bestimmen. Allerdings muss er Datenanbieter finden, die ihre Daten für solche Anwendungen abgeben. Viele Datenanbieter geben aus urheberrechtlichen Gründen und dem Datenschutz meist keine Originaldatensätze ab oder verlangen dafür meist erhebliche Gebühren. Für die Verwendung eines „Fat Client“ sind relative gute Systemvoraussetzungen des Clients notwendig, da er die Verarbeitung der Datensätze und deren Rendering für die Visualisierung übernehmen muss. Es ist natürlich auch notwendig, entsprechende Software zu installieren.

Setzt sich die Systemarchitektur aus einem „Medium Client“ und einem „Medium Server“ zusammen, so verteilen sich die Ansprüche an die jeweiligen Systeme. Den größeren Teil, die Verarbeitung der Daten, übernimmt der Server. Die Installation von Software beim Client bleibt allerdings nicht aus. Entsprechende Programme, die das Rendern von 3D-Szenen umsetzen, sind notwendig. Dies ermöglicht dem Nutzer in direkte Interaktion mit der Anwendung zu treten, da er sich in der 3D-Szene bewegen

kann. Durch Verwendung von eigenen Stilen kann er die Darstellung der Daten zum Teil beeinflussen. Hierbei muss der Aufbau der Datensätze genau beachtet werden und die Stildefinition entsprechend darauf abgestimmt sein. Die gemeinsame Darstellung von unterschiedlichen Datensätzen ist mit dieser Systemarchitektur nur zum Teil möglich. Es ist nur machbar, wenn die Transformation von dem Koordinatensystem der Daten in das Bildkoordinatensystem (vgl. Abb. 13 in Punkt 4.2.1) bei allen verwendeten Datensätzen exakt gleich ist.

Bei Verwendung der letzten Variante, „Thin Client“ und „Fat Server“, als Systemarchitektur hat der Nutzer die wenigsten Möglichkeiten, auf die Anwendung und die Daten Einfluss zunehmen. So kann es keine direkte Interaktion mit der Anwendung geben, da sich der Anwender nicht direkt in der 3D-Szene bewegt, sondern lediglich ein Bild der aktuellen Ansicht der Daten erhält. Ähnliches gilt auch für die Darstellung der Daten. So kann der Nutzer nur die Darstellungsstile nutzen, die von der Anwendung selber angeboten werden. Da alle Schritte der Visualisierung, außer der Ansicht der Daten, auf dem Server ablaufen, können unterschiedliche Datensätze aus verschiedenen Quellen nicht kombiniert und gemeinsam dargestellt werden. Der große Vorteil dieser Systemarchitektur liegt darin, dass der Nutzer keinerlei Software installieren muss. Ein einfacher Browser ist ausreichend. Somit sind die Ansprüche an das System des Nutzers entsprechend gering. Da eine solche Anwendung ohne weitere Software beim Client auskommt, ist sie zugleich plattformunabhängig und kann jederzeit ausgeführt werden.

Tab. 5 fasst die Eigenschaften der einzelnen Systemarchitekturen noch einmal kurz zusammen.

Tab. 5: Eigenschaften der Systemarchitekturen bezogen auf den Nutzer

	Notwendigkeit einer Installation von Software	**Direkte Interaktion des Nutzers mit der Anwendung**	**Einfluss des Nutzers auf die Darstellung der Daten**	**Gemeinsame Darstellung unterschied-licher Datensätze**
Fat Client / Thin Server	Ja	Ja	Hoch	Ja
Medium Client / Medium Server	Ja	Ja	Mittel	Teilweise
Thin Client / Fat Server	Nein	Nein	Niedrig	Nein

5.2 Anforderungen an die Daten

Um einen Datenbestand in einem 3D-WebGIS zu visualisieren, müssen die Daten grundsätzlich keine speziellen Anforderungen erfüllen. Ist ein hinreichend genaues Digitales Geländemodell vorhanden, reicht es aus, wenn alle Daten zweidimensional vorliegen. Ist das nicht der Fall, so müssen die Daten natürlich Höheninformationen haben.

Für das Ablegen der Geometrien sollte das OpenGIS Geometry Model des OGC verwendet werden. Legt man die Daten in einer Datenbank der Firma *Oracle* ab, so ist es in deren MDSYS-Schema implementiert. Als Geometrieattribut wird dann einheitlich GEOM mit dem Datentyp SDO_GEOMETRY verwendet. Um die Bearbeitung von Anfragen zu beschleunigen, sollten alle Daten in demselben räumlichen Bezugssystem vorliegen. Da dies bei jedem Objekt mit angegeben werden kann, können auch verschiedene Systeme verwendet werden. Wie schon erwähnt, würde dies die Bearbeitung einer Anfrage etwas verlängern, da die Daten vor der Verarbeitung transformiert werden müssten.

Die Gestaltung des Datenmodells ist ebenfalls nicht zwingend relevant. Es ist aber zu empfehlen ein Datenmodell zu verwenden, welches dem CityGML Standard entspricht. Dies hat mehrere Gründe. In einem solchen Datenmodell sind eine ganze Reihe von wichtigen Aspekten für ein dreidimensionales Modell berücksichtigt worden. So kann man für jedes Objekt verschiedene Levels of Detail in der Datenbank ablegen, um eine flüssigere Visualisierung zu gewährleisten und die Ladezeiten der Daten zu verringern. Desweiteren ist das Abspeichern von verschiedenen Darstellungseigenschaften für die Objekte vorgesehen. Auch auf die Interoperabilität der Anwendung würde sich dies positiv auswirken, da man mit einem offiziellen OGC Standard schon in der Datenhaltung beginnt. Gegen die Verwendung eines Datenmodells nach dem CityGML Standard spricht allerdings der erhebliche Aufwand, der für den Aufbau eines solchen Systems erbracht werden müsste. Es ist daher für jedes Projekt individuell abzuwägen, wie das Verhältnis zwischen Aufwand und Nutzen aussehen soll.

Neben Vektordaten sind auch Rasterdaten für ein 3D-WebGIS wichtig. Dazu zählen alle Bilder, die als Texturen für die einzelnen Objekte Verwendung finden ebenso wie Luft- oder Satellitenbilddaten. Diese sollen für die Gestaltung des Geländes auf das Digitale Geländemodell aufgelegt werden. Um eine fehlerfreie Visualisierung zu ermöglichen, sollten die Luft- oder Satellitenbilddaten als georeferenzierte Bilddaten vorliegen. Das räumliche Koordinatensystem muss dabei dem des Digitalen Geländemodells entsprechen. Gleiches gilt für das DGM.

5.3 Werkzeuge

5.3.1 LandXplorer

Der LandXplorer der Firma *Autodesk* wurde ursprünglich von der Firma *3D Geo* entwickelt. Mit diesem Softwarepacket wird das Management komplexer zwei- und dreidimensionaler Geodaten auf der Basis von virtuellen 3D-Stadt- und 3D-Landschaftsmodellen abgedeckt. Neben der Visualisierung großer Mengen von Geodaten in Echtzeit, ist die Exploration, Analyse und Editierung sowie die Präsentation dieser Daten möglich. [DÖLLNER 2006] Nach tiefgreifenden Recherchen ist dies momentan die einzige Software, die diese Möglichkeiten für dreidimensionale Datenbestände, die in einer Datenbank vorliegen, bietet.

Als Grundlage für jedes Raummodell verwendet der LandXplorer ein Geländemodell, welches als georeferenziertes Raster oder als TIN eingelesen werden kann. Als sogenannte Layer können sowohl zwei- als auch dreidimensionale Vektordaten geladen werden. Die ersteren projiziert die Software auf das Geländemodell und die letzteren passt er daran visuell an. Damit soll verhindert werden, dass Objekte über dem Gelände „schweben“ oder in dieses „einsinken“. Auch im LandXplorer werden als grundlegende Gestaltungsmittel Rasterdaten als georeferenzierte Texturen verwendet. Dies gilt aber nicht nur für die einzelnen Geometrieobjekte sondern auch für das Geländemodell selbst, welches z. B. mit Luftbildern texturiert werden kann. Das Hauptanwendungsgebiet der LandXplorer-Technologie sind 3D-Stadtmodelle. Daher unterstützt er CityGML und deren LoD-Strukturen, die jeweils sicht- und distanzabhängig geladen werden.

Neben der Visualisierung und Analyse der Daten können Raummodelle mit dem LandXplorer Studio in vollem Umfang erstellt, bearbeitet und in einem Dateisystem

oder einer Datenbank abgelegt werden. Hier ist es möglich, lesend und schreibend auf eine Datenbank der Firma *Oracle* oder eine ArcSDE der Fima *ESRI* zuzugreifen. [DÖLLNER 2006]

Mit Hilfe der „Pack & Go“ – Funktion hat man die Möglichkeit, das erstellte Raummodell zu exportieren. Dabei wird eine programmspezifische Dateistruktur erstellt, die man einem Nutzerkreis zur Verfügung stellen kann, ohne die Originaldaten weitergeben zu müssen. Mit dem kostenlosen LandXplorer Xpress Viewer kann das Raummodell dann betrachtet werden. Die Daten können auf diesem Weg auch über das Internet zur Verfügung gestellt werden. [DÖLLNER 2006]

5.3.2 Oracle Spatial

Zur Verwaltung von großen Datenbeständen ist eine leistungsstarke Datenbank notwendig. Für die Überarbeitung der Daten soll der LandXplorer eingesetzt werden. Wie schon erwähnt, kann man mit diesem Werkzeug nur auf eine Datenbank der Firma *Oracle* oder die ArcSDE der Firma *ESRI* zugreifen. Daher wird als Datenbank eine Oracle Spatial Datenbank vorgeschlagen.

Oracle Spatial ist eine Teilkomponente des Datenbankmanagementsystems eines der größten Softwarehersteller, *Oracle Corporation*, mit dessen Hilfe Geometriedaten in einer objektrelationalen Datenbank abgelegt und verwaltet werden können. Die aktuell verfügbare Version ist die Oracle 11g Release 1 (11.1). Mit der Version 11g veröffentlichte *Oracle* die erste echte 3D-Datenbank. Mit dieser ist es möglich, dreidimensionale Geodaten als komplexe Objekte abzulegen, zu verwalten und zu analysieren. Die folgenden 3D-Datentypen werden dabei unterstützt: 3D-Punkte, 3D-Linien, 3D-Polygone, 3D-Oberflächen, TIN’s, 3D-Punktwolken und zusammengesetzte 3D-Oberflächen, sowie 3D-Körper. Im Gegensatz zu zweidimensionalen Objekten, die eine Höhe im 3D-Raum besitzen, handelt es sich bei dreidimensionalen Objekten um Volumenkörper. Während z. B. ein Rechteck eine Höhe im Raum besitzt und von Linien begrenzt wird, bildet sich ein Volumenkörper aus mehreren Flächen.

Um diese Daten effizient speichern und verwalten zu können, bietet das Datenbankmanagementsystem eine Indexierung der Daten, die sich auch auf die dritte Dimension, in Form der Höhe, erstreckt. Dabei bildet der minimal umgebende Quader die Grundlage für eine geometrische Approximation. Räumliche Analysen und

Abfragen werden ebenfalls unterstützt. Natürlich bietet Oracle Spatial 11g auch die Möglichkeit, aus zweidimensionalen Objekten und der Zuweisung einer Höhe dreidimensionale Objekte zu erzeugen und unter Verwendung des entsprechenden 3D-Datentyps abzulegen.

Die Datenbank wird in logische Einheiten gegliedert. Die oberste Strukturebene bilden die sogenannten Tablespaces, die alle logisch zusammengehörigen Datenbankbestandteile zusammenfassen. Den Tablespaces können nun verschiedene Schemata zugewiesen werden, welche die Tabellen, Indexe, Prozeduren usw. enthalten. Mit Hilfe dieser Struktur kann man schon vordefinierte Schemata für den Aufbau eigener Datenbanken verwenden. So wurde das „3D-Citybase Schema“ von der Universität Bonn für den Aufbau von 3D-Stadtmodellen, die auf der CityGML-Spezifikation des OGC basieren, entwickelt. Natürlich kann von jedem Schema aus auf ein anderes zugegriffen werden. [BRINKHOFF 2008]

Zum Ablegen der Geometrie wurde die OGC-Spezifikation „Simple Features“ und die ISO-Norm „SQL/MM Spatial“ implementiert. Dazu ist standardmäßig das Schema MDSYS angelegt, was die generelle Geometrieklasse SDO_GEOMETRY bereitstellt. Mit Hilfe dieser Klasse können alle Geometrien abgelegt werden. Durch Angabe einer Schlüsselnummer wird der Geometrietyp entsprechend spezifiziert. Auf diesem Weg ist es möglich, in einer Tabelle Objekte mit unterschiedlichen Geometrietypen, allerdings der gleichen Dimension, zu speichern. Wichtig ist es, immer das entsprechende räumliche Bezugssystem anzugeben. Dazu wird das Element „Spatial Reference ID“ (SRID) verwendet. Die dabei anzugebenden Schlüsselnummern entsprechen den European Petroleum Survey Group-Codes (EPSG). [BRINKHOFF 2008]

5.3.3 Deegree

Deegree ist ein Open Source System, das auf Java basiert und zur Erstellung von Geodateninfrastrukturen entwickelt wird. Seine Wurzeln gehen auf das Projekt EXSE (GIS-Experimental server at the Internet) zurück, welches an der Universität Bonn 1997 ins Leben gerufen wurde. Dessen Ziel war die experimentelle Verbindung von GIS-Funktionalitäten und Internettechnologien. Heute wird das deegree-Projekt in einer Kooperation zwischen der Firma *lat/lon* und der Arbeitsgruppe Geografische Informationssysteme der Universität Bonn entwickelt.

Deegree setzt sich aus unterschiedlichen Komponenten zusammen, wodurch flexible Lösungsmöglichkeiten für die unterschiedlichsten Anwendungsfälle gefunden werden können. Es beinhaltet zum einen verschiedene Dienste, die deegree Web Services, und zum anderen Lösungsmöglichkeiten für Portale (deegree iGeoPortal), Mechanismen für die Handhabung von Sicherheitsfragen (deegree iGeoSecurity) sowie zur Speicherung und Visualisierung von 3D-Geodaten (deegree iGeo3D). [LUPP 2008]

Alle Komponenten basieren auf den Standards des OGC und der ISO/TEC 211 und wurden unter der GNU Lesser General Public License veröffentlicht. Aus diesem Grund nutzt deegree auch die Konzepte des OGC für seine internen Strukturen. So sind alle Konfigurationsdateien XML-basierend und entsprechen den aktuellen OGC-Spezifikationen.

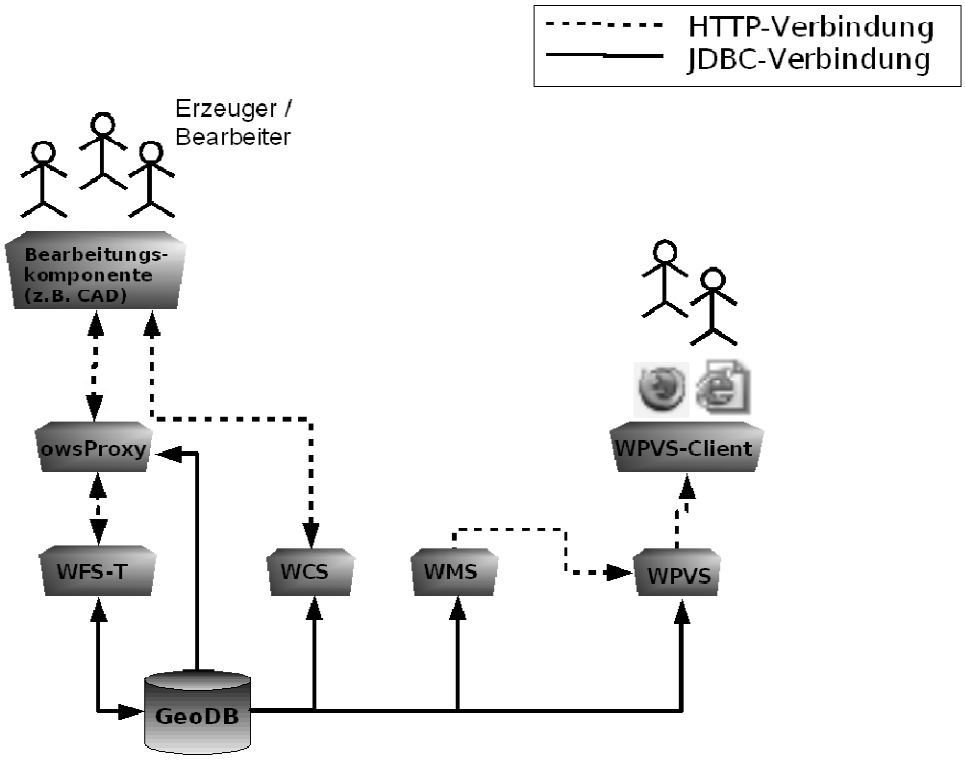

Abb. 21: Architektur einer 3D-Geodateninfrastruktur mit deegree-Komponenten [BEZEMA, U. MÜLLER et al. 2008]

Seit der Veröffentlichung von deegree2 2005 werden dreidimensionale Datenstrukturen unterstützt. In Abb. 21 ist zu sehen, wie eine Geodateninfrastruktur für dreidimensionale Daten mit deegree Komponenten aussehen kann. Dabei sind die Daten in einer Datenbank (GeoDB), z. B. Oracle Spatial, oder in Dateien abgelegt. Die Visualisierung des Geländemodells wird mit Hilfe eines Web Coverage Services umgesetzt, während alle anderen Objekte mit Hilfe des Web Perspective View Services

direkt von der Datenbank abgefragt und dargestellt werden. Zur Abbildung der Texturen, z. B. von Luftbildern, bindet man einen Web Map Service in die Visualisierung ein. [BEZEMA, U. MÜLLER et al. 2008]

Die Recherchen für die vorliegende Arbeit haben ergeben, dass momentan der deegree-WPVS als einzige Anwendung zur Implementierung eines 3D-WebGIS zur Verfügung steht, wenn man eine Plug-In freie Lösung anstrebt.

5.4 Projektkonzept für ein 3D-WebGIS

5.4.1 Anforderungen

Bevor eine Anwendung implementiert werden kann, müssen die Anforderungen an diese klar formuliert sein.

In erster Linie soll das 3D-WebGIS zur Information dienen. Nutzer sollen die Möglichkeit haben, sich in einem dreidimensionalen Modell zu bewegen. Dabei wird das Modell „on the fly“ aus der Datenbank generiert und dem Nutzer einige Abfragemöglichkeiten, wie die Suche nach z. B. Adressen oder anderen interessanten Objekten zur Verfügung gestellt. Daher ist das erste Ziel der Anwendung die Visualisierung eines Datenbestandes in einer 3D-Szene.

Die Applikation soll die aus dem zweidimensionalen Bereich bekannten Zoomfunktionen (Pan, hinein und heraus zoomen) zur Verfügung stellen. Dazu kommen noch weitere Interaktionsmöglichkeiten. So soll der Nutzer den Sichtabstand zu den Objekten, den Neigungswinkel der Sichtachse sowie die Ausrichtung der Szene bezüglich der Nordrichtung selbst bestimmen können. Zur besseren Orientierung in der gesamten Szene ist es wichtig, eine zweidimensionale Übersichtskarte anzubieten, in der der Nutzer seinen momentan angezeigten Ausschnitt sehen kann.

Wie auch bei jeder anderen digitalen Karte, sollte eine Legende angeboten werden, die die einzelnen Objektklassen und deren Darstellung anzeigt. Darüber hinaus soll der Nutzer selbst bestimmen können, welche dieser Objektklassen in der Szene geladen werden.

Neben den einzelnen Funktionen, die die Anwendung beinhalten soll, ist es wichtig eine Plug-In freie Lösung umzusetzen. Somit muss der Nutzer keine weitere Software

installieren, um das 3D-WebGIS zu verwenden. Ein normaler Browser wäre ausreichend und der Nutzerkreis wird nicht begrenzt.

Ein weiterer wesentlicher Punkt ist die Verwendung von Standards. Zudem ist es wichtig, sich an den Grundsatz der Interoperabilität zu halten, um sich mit anderen Projekten austauschen zu können. Eine Konformität mit Standards ermöglicht später auch eine Kombination unterschiedlicher Datensätze und somit eine gemeinsame Auswertungen von Daten.

5.4.2 Das Konzept im Überblick

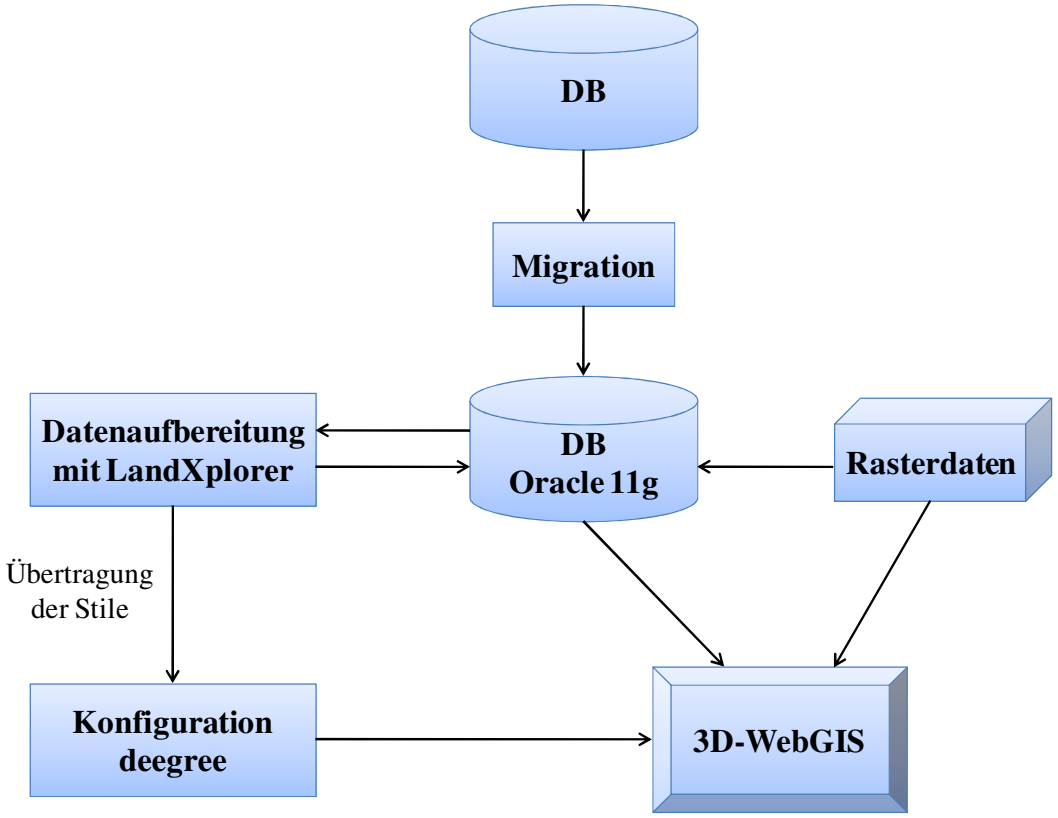

Abb. 22: Projektkonzept im Überblick

Eine Implementierung eines 3D-WebGIS muss in mehreren Schritten durchgeführt werden. Nachdem die Anforderungen an die Applikation formuliert wurden und die Werkzeuge ausgewählt, müssen die Daten entsprechend aufbereitet werden. Dazu sollte eine Migration des Datenbestandes von der vorhandenen Datenbank in eine Oracle Spatial Datenbank der Version 11g durchgeführt werden. Anschließend kann mit der Datenüberarbeitung im LandXplorer Studio begonnen werden. Die entsprechenden

Änderungen an den Geodaten sind zurück in die Datenbank zu speichern. Nun kann die Konfiguration des deegree-WPVS vorgenommen werden. Neben den Einstellungen für den Zugriff auf die Rasterdaten und die Datenbank, werden die Stile aus dem LandXplorer Studio übertragen. Aus diesen drei Komponenten, Datenbank, Rasterdaten und Konfiguration, bildet sich schließlich die Applikation für das 3D-WebGIS.

5.4.3 Datenaufbereitung

Als Grundlage für das 3D-Modell wird ein Digitales Geländemodell verwendet. Die Genauigkeit eines vorhandenen DGMs sollte für das WebGIS ausreichend sein, da in der Darstellung alle Objekte auf das DGM angepasst werden. Dadurch verhindert man, dass Objekte über dem Gelände „schweben“ oder darin „einsinken“. Dies kann bei der Verwendung der richtigen Objekthöhe passieren, da diese meist eine höhere Genauigkeit aufweisen als ein Geländemodell. Für Objekte, die nur mit ihren Lagedaten vorliegen, ist die Anpassung an das Geländemodell von vornherein nötig.

Zur Verfügung stehende Satelliten- oder Luftbilder werden als Texturen auf das Geländemodell gelegt. Für ein fehlerfreies Funktionieren sollten sie als georeferenzierte Rasterdaten im Tagged Image File Format (TIFF) vorliegen.

Es ist sinnvoll den vorhandenen Datenbestand in eine Oracle Spatial Datenbank der Version 11g zu überführen, da diese Version die Möglichkeit zur Verwaltung und Analyse von 3D-Objekten bietet und dem Stand der Technik entspricht. Bei der Überführung des Datenbestandes werden die Objektgeometrien allerdings nicht automatisch in 3D-Geometrien umgewandelt. Der jeweilige Geometrietyp, zweidimensionales Objekt mit einer Z-Koordinate, bleibt erhalten. Zur Erzeugung von 3D-Objekten in der Datenbank müsste für jedes Objekt der Geometrietyp geändert werden. Für den Aufbau eines 3D-WebGIS im Rahmen dieser Arbeit ist dies nicht notwendig, da meistens nicht alle Objekte eine dritte Dimension besitzen und zum anderen vorerst die dritte Dimension lediglich für die Visualisierung benötigt wird. Die 3D-Visualisierung ermöglicht das Digitale Geländemodell.

Für ein vollständiges 3D-Modell sollten die Vektordaten überarbeitet werden. Dazu verwendet man die Software LandXplorer Studio. Mit Hilfe dieses Werkzeugs können alle Daten des Modells geladen, entsprechend verändert und wieder in der Datenbank

abgelegt werden. Wie in Punkt 5.3.1 beschrieben, bietet es die dafür notwendigen Funktionen.

Die Gebäude müssten z. B. so aufbereitet werden, dass sie mindestens der LoD1, dem Blockmodell, entsprechen. Dazu wird allen Gebäuden vorerst eine einheitliche Gebäudehöhe zugewiesen, falls noch keine richtigen Höhen oder Stockwerkzahlen vorhanden sind. Sollten diese zur Verfügung stehen, sind sie natürlich auch zu verwenden. Zweidimensionale Objekte, die auf Grund ihrer Gestalt, als dreidimensionale Objekte dargestellt werden können, z. B. Steinhaufen oder Mauern, werden mit einem entsprechenden Körper, z. B. einem Kegel oder Quader, visualisiert und mit einer Textur versehen. So kann mit dem LandXplorer das gesamte 3D-Modell erstellt und gestaltet werden. Die hier festgelegten Konfigurationen bezüglich der Darstellung der Daten werden später in die Konfiguration der WebGIS-Anwendung übernommen. Abb. 23 zeigt ein Blockmodell im LandXplorer Studio mit einheitlichen Gebäudehöhen.

Abb. 23: Erzeugtes Blockmodell mit dem LandXplorer

5.4.4 Systemarchitektur

Da das 3D-WebGIS ohne Plug-In auskommen soll, erfolgt die Umsetzung mit Hilfe des Web Perspective View Service. Zur Realisierung wird des Open Source Projekte deegree verwendet. Zum einen handelt es sich um kostenfreie Software und zum anderen basiert es auf den Standards des OGC und hat den WPVS bereits implementiert. Dies würde eine interoperable Zusammenarbeit mit anderen Projekten ermöglichen.

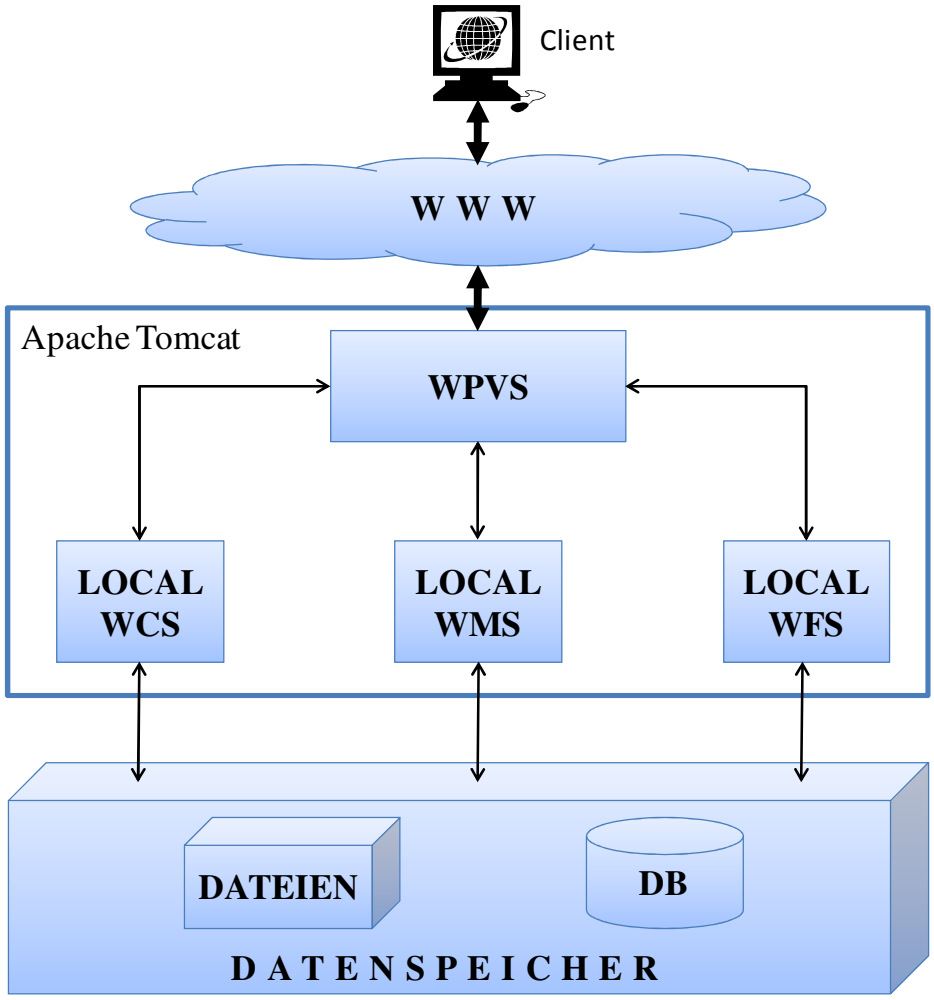

Abb. 24: Architektur des deegree-WPVS

In der Abb. 24 ist der Aufbau des deegree-WPVS zu sehen, wie er für das WebGIS verwendet werden soll. Die Datengrundlage des 3D-WebGIS besteht aus der Oracle Datenbank mit allen Vektordaten und einem Dateisystem, welches alle zur Verfügung stehenden Rasterdaten, wie DGM und Satellitenbilder, enthält. Der WPVS-Dienst wird in einen Apache Tomcat Server integriert. Er ist für das Erstellen und Rendern der 3D-Szene verantwortlich. Um eine angeforderte 3D-Szene zusammenzusetzen, verwendet der deegree-WPVS weitere, sogenannte lokale Dienste. Das DGM wird mit Hilfe eines WCS geladen, während die Texturen, wie Satelliten- und Luftbilder, über einen WMS hinzugefügt werden. Die Vektordaten kommen über einen WFS hinzu, der die entsprechenden Daten aus der Datenbank abfragt.

Die Anpassung des WPVS an den entsprechenden Datenbestand erfolgt über mehrere Konfigurationsdateien in XML-Notation. Dabei werden in allen Dateien jeweils der zur Verfügung stehende Bereich in Form eines umgebenden Rechteckes und das

verwendete Koordinatensystem angegeben, genauso wie die Informationen zur Herkunft der Daten und die Urheberrechte. Die einzelnen Datensätze, die über den Dienst zur Verfügung gestellt werden sollen, sind in der WPVS-Konfigurationsdatei (wpvs_configuration.xml) aufzuführen und mit einem Verweis auf den jeweils zu verwendenden lokalen Dienst und dessen Einstellungsdatei versehen. In diesen werden nun die Zugriffsparameter auf die entsprechenden Daten hinterlegt. Die Konfigurationsdatei des WPVS ist als Hauptkonfiguration anzusehen. Diese legt weitere Rahmenparameter fest, wie Darstellung der Legende, Übersichtskarte, Hintergrund der Szene usw. Bei dem WCS ist für jeden einzelnen Rasterdatensatz eine Konfigurationsdatei notwendig. Sie beinhaltet die Eigenschaften der Rasterdaten und es ist sinnvoll, diese direkt bei den entsprechenden Daten abzulegen. In der Konfigurationsdatei für den lokalen WCS selbst (LOCALWCS_capabilities.xml) werden diese dann für jeden Datensatz mit aufgeführt. Die Datei für den lokalen WFS (LOCALWFS_capabilities.xml) enthält neben den Angaben zu den möglichen Operatoren und Filterfunktionen des Dienstes, Verweise auf die einzelnen Konfigurationsdateien, die den Datenzugriff ermöglichen. Durch die Verwendung von Konfigurationsdateien für jede Objektklasse können Daten aus unterschiedlichen Datenbanken verwendet werden. Alle Objektklassen, die dem Nutzer zur Verfügung gestellt werden sollen, sind hier mit ihren einzelnen Attributen und Datentypen sowie deren Darstellungseigenschaften in Form von GML-Elementen aufzuführen. Das bedeutet, dass man sie in Form von XML-Schemata beschreibt und die Parameter für den Zugriff auf die jeweilige Datenbank mit angibt.

Startet ein Nutzer die Anwendung, so wird ein erster Aufruf mit dem Operator GetView an den WPVS-Server geschickt. Dabei werden die für die Startansicht festgelegten Parameter übermittelt. Die hier verwendeten Parameter sind in der richtigen Reihenfolge in Tab. 6 aufgeführt. Mit diesen fragt der Server die benötigten Daten über die lokalen Dienste ab, setzt sie zu einer 3D-Szene zusammen und rendert die Szene. Anschließend kann das erstellte Pixelbild mit der gewünschten Perspektive zur Ansicht an den Nutzer gesendet werden. Diesen Vorgang führt der Server bei jeder Interaktion des Nutzers aus, wobei die entsprechenden Parameter jeweils mit zu senden und zu verarbeiten sind. Aus diesem Grund kann man nur von einer indirekten Interaktion des Nutzers sprechen.

Tab. 6: verwendete Parameter des deegree-WPVS

Parameter	Beschreibung	Beispiel
BOUNDINGBOX	Minimal umschließendes Rechteck	423750,4512700,425500,4513900
DATASETS	Anzuzeigende Datensätze	Buildings,satellite_images
ELEVATIONMODEL	Zu verwendendes DGM	saltlake_dem
ROLL	Verdrehung um die Blickrichtung	0
AOV	Blickwinkel	60
FARCLIPPINGPLANE	Begrenzung des Sichtweite	10000
CRS	räumlichen Referenzsystems als EPSG-Code	EPSG:26912
WIDTH	Breite des Ausgabebildes in Pixel	800
HEIGHT	Höhe des Ausgabebildes in Pixel	600
SCALE	Maßstab	1.0
STYLES	Zu verwendende Darstellungsstile der einzelnen Datensätze	default
DATETIME	Aktualität der Anwendung	2007-03-21T12:00:00
EXCEPTIONFORMAT	Format für eventuelle Fehlermeldungen	INIMAGE
SPLITTER	Begrenzung des sichtbaren Bereichs	BBOX
VERSION	Version der Anwendung	1.0.0
OUTPUTFORMAT	Rasterformat, indem das Ergebnis angezeigt werden soll	image/jpg
BACKGROUND	Hintergrundgestaltung	cirrus
POI	Zielpunkt	424750.0,4513400.0,1350
YAW	Ausrichtung der Szene in Bezug zur Nordrichtung	20
DISTANCE	Sichtabstand zu den Objekten	2500

Mit Hilfe des Parameters BOUNDINGBOX gibt man ein Rechteck an, welches für die Selektion der darzustellenden Daten verwendet wird. So selektiert der Server alle Objekt, die sich in diesem Bereich befinden. Angegeben wird dieser Gebietsausschnitt in dem räumlichen Koordinatensystem, welches für die Daten verwendet werden soll. FARCLIPPINGPLANE gibt an, wie weit der Nutzer sehen kann. Damit kann unter Umständen die Menge der abzufragenden Daten reduziert werden, um die Antwortgeschwindigkeit zu erhöhen. Sie sollte allerdings nicht zu gering gewählt sein. SPLITTER ist für die Begrenzung des sichtbaren Bereichs verantwortlich. Er kann einen eigenen Wert haben oder den gleichen wie die BOUNDINGBOX.

Mit Hilfe von DATASETS zählt man alle Datensätze auf, die von der Datenbank abgefragt und später angezeigt werden sollen. Das zu verwendende Digitale Geländemodell legt man über den Parameter ELEVATIONMODEL fest. CRS definiert

das räumliche Koordinatensystem, in welchem die Daten angezeigt werden sollen. Dazu werden die EPSG-Codes verwendet.

Der Parameter STYLES gibt die Stile an, die für die Darstellung der verschiedenen Objekte Verwendung finden. Hier kann man aber nur die angeben, die vom Server zur Verfügung gestellt werden. Der Nutzer hat keine Möglichkeit eigene Stildefinitionen einzusetzen. Bei BACKGROUND gibt man ein Rasterbild an, welches für die Gestaltung des Hintergrundes genutzt wird.

ROLL, AOV, POI, YAW, DISTANCE definieren die Perspektive auf die Daten. Sie wurden in Punkt 3.4.3 beschrieben und werden hier in gleicher Weise verwendet.

Mit den folgenden Parametern definiert man das Rasterbild, welches als Ergebnis beim Nutzer angezeigt wird. WIDTH und HEIGHT bestimmen die Größe des Rasterbildes in Pixeln. Über SCALE kann man noch eine Skalierung des Bildes vornehmen und OUTPUTFORMAT gibt das Dateiformat des Rasterbildes an.

Der Parameter EXCEPTIONFORMAT legt fest, in welcher Form auftretende Fehlermeldungen an den Nutzer gesendet werden, während DATETIME und VERSION Aussagen zur Anwendung selbst treffen.

In der Abb. 25 ist der deegree-WPVS mit Beispieldaten, die vom deegree-Projekt zum Testen zur Verfügung gestellt werden, zu sehen.

Abb. 25: Deegree-WPVS-Applikation mit Beispieldaten

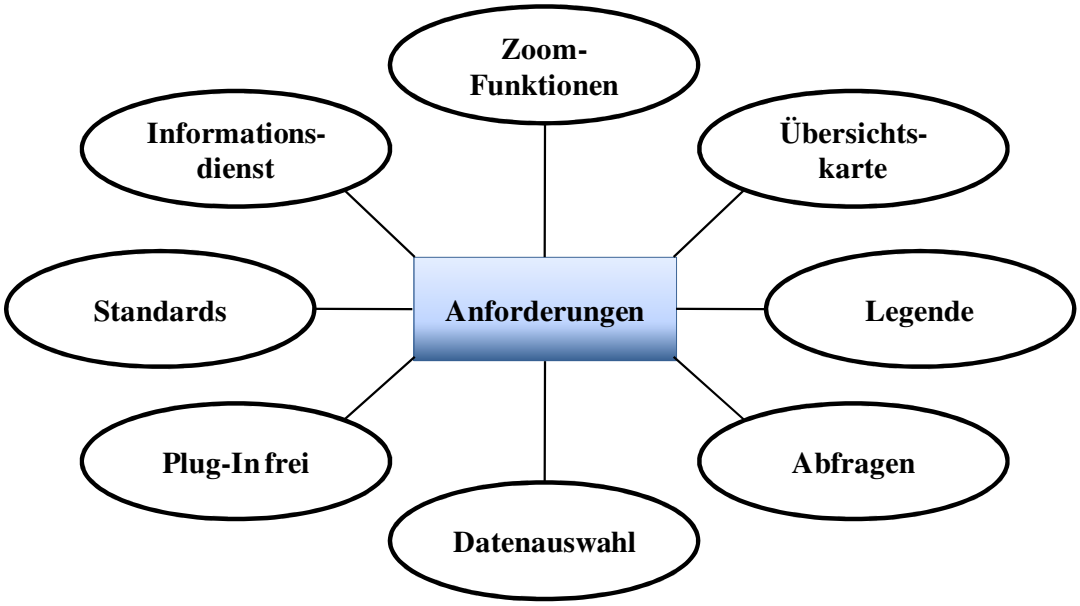

Abb. 26: Anforderungen an das 3D-WebGIS

Mit der vorgeschlagenen Systemarchitektur können die gestellten Anforderungen an die Applikation erfüllt werden. Der verwendete deegree-WPVS bietet dem Nutzer die geforderten Navigationsmöglichkeiten. Zur besseren Orientierung in dem 3D-Modell wird eine Übersichtskarte zur Verfügung gestellt. Ebenso kann er aus den angebotenen Layern die frei auswählen, die angezeigt werden sollen. Auch die Darstellung einer Legende ist machbar.

Die Forderungen nach einer Plug-In freien Lösung und der Verwendung von Standards werden eingehalten. Durch die Wahl des Web Perspective View Service ist die Installation von zusätzlicher Software beim Client nicht notwendig. Die Applikation kann durch einen einfachen Webbrowser genutzt werden. Gleichzeitig stellt man mit diesem System die Verwendung von Standards sicher. Auch wenn die WPVS-Spezifikation noch nicht offiziell vom OGC verabschiedet wurde, so kann man von einem Standard sprechen. Weiterhin verwendet der deegree-Dienst schon bestehende Standards, wie den Web Coverage Service und den Web Feature Service.

Der geforderte Datenzugriff „on the fly“ ist mit einer Anwendung, die der vorgeschlagenen Systemarchitektur folgt, ebenfalls umsetzbar. Alle Daten werden direkt aus dem vorhandenen Datenspeicher generiert. An dieser Stelle sollte man aber noch Überlegungen zur Datensicherheit machen.

Das gesamte 3D-Modell baut sich auf der Grundlage eines Digitalen Geländemodells auf. Um unschöne Effekte, wie das „Schweben“ oder „Einsinken“ von Objekten in das

Gelände, zu vermeiden, werden alle Elemente in ihrer Höhe an das Gelände angepasst. Dies wird bei der Generierung der 3D-Szene vom Server berechnet, so dass die Originaldaten nicht verändert werden müssen. Dieses Verfahren ermöglicht dabei auch, dass zweidimensionale Objekte ebenso mit dargestellt werden können, wie dreidimensionale.

Zusammenfassend lässt sich sagen, dass mit dem vorgeschlagenen Lösungskonzept alle gestellten Anforderungen an ein 3D-WebGIS erfüllt werden können.

6 Zusammenfassung und Ausblick

Zielsetzung der vorliegenden Arbeit war die Beschreibung des Aufbaus eines 3D-WebGIS. Dafür sollte ein Überblick über standardisierte Dienste und Formate, die für eine solche Anwendung wichtig sind, gegeben werden. Die Erarbeitung eines Lösungskonzeptes zum Aufbau eines 3D-WebGIS sollte die Arbeit abschließen. Dabei war es wichtig die Generierung der Datenansicht „on the fly“ aus einem Datenspeicher zu ermöglichen.

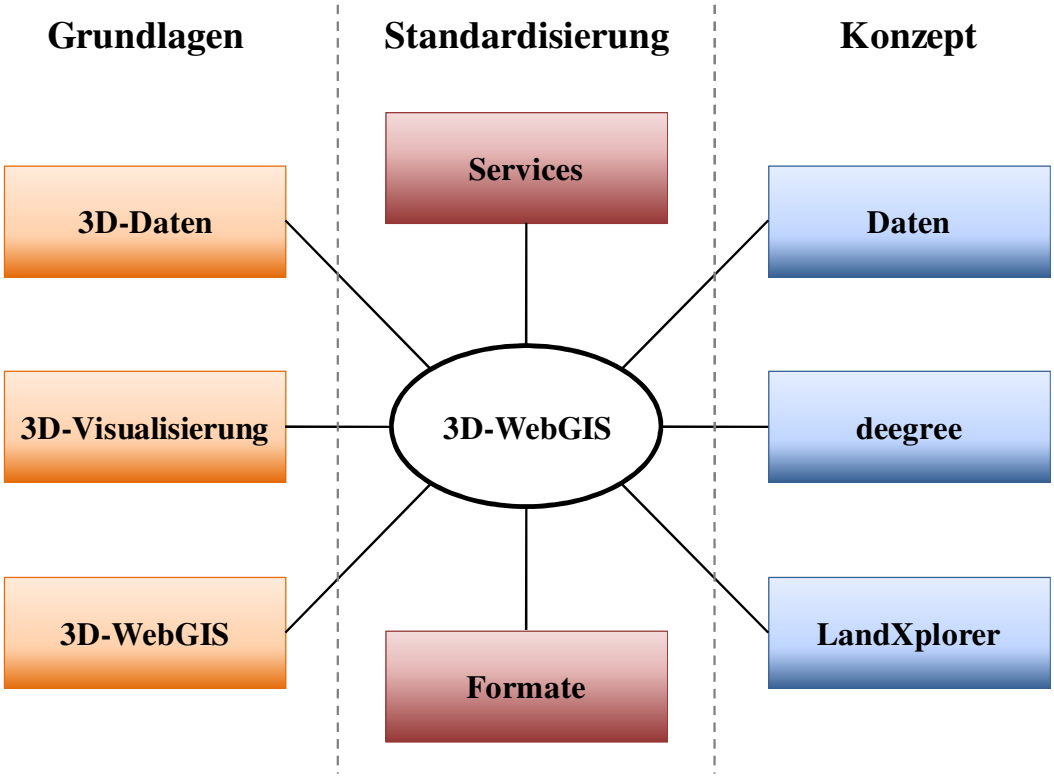

Abb. 27: Zusammenfassung im Überblick

Nachdem ein Überblick über den Stand der aktuellen Forschungen auf dem Gebiet der Visualisierung von Geodaten über das Internet sowie über die Literatur gegeben wurde, erläuterte man die Grundlagen zu 3D-Geodaten, deren Modellierung und Visualisierung. Als wichtigste Beschreibungsmöglichkeit für dreidimensionale Daten stellte sich der Szenegraph heraus. Mit diesem können zum Einen die Geodaten selbst beschrieben werden und zum Anderen deren Darstellung. Um eine 3D-Szene so realistisch wie möglich zu gestalten und gleichzeitig das Datenvolumen relativ gering zu halten, kommen Texturen zur Anwendung. Als Rasterbilder auf die Objekte aufgebracht, verleihen sie diesen ein realitätsnahes Aussehen.

Der grundlegende Aufbau und die Funktionsweise eines dreidimensionalen WebGIS wurden im Anschluss erläutert. Bei einer solchen Applikation handelt es sich um eine klassische Client-/Server-Architektur, deren Aufbau von den Anforderungen an die Endanwendung bestimmt wird. Anhand der Visualisierungspipeline wurde die Funktionsweise des 3D-WebGIS beschrieben. Sie erstreckt sich von der Selektion der Daten über deren Darstellung, das Zusammensetzen und Rendern der 3D-Szene bis hin zur Anzeige des Ergebnisses beim Client. Bei einer 3D-Szene handelt es sich immer um perspektivische Sichten auf Daten. Aus diesem Grund wurde auf die verschiedenen Parameter eingegangen, die für die Definition einer Perspektive wichtig sind. Mit Hilfe dieser Parameter kann der Nutzer seine Sicht auf die Daten selbst festlegen.

An die Grundlagen für ein 3D-WebGIS schloss sich die Beschreibung der unterschiedlichen Möglichkeiten zum Aufbau eines 3D-WebGIS unter Anwendung von standardisierten Diensten und Formaten an. Für die Entwicklung eines solchen Systems sind grundsätzlich drei Standards wichtig, die sich alle in der Diskussion innerhalb des OGC befinden und von diesem noch nicht als offizieller Standard bestätigt wurden. Der erste ist der Web 3D Service. Er erlaubt die Visualisierung von 3D-Geodaten in einem dreidimensionalen Raum. Unter Verwendung eines Szenegraphen, der die Daten selbst und deren Darstellung enthält, ermöglicht der Dienst die Darstellung beim Client. Der Nutzer kann so mit Hilfe eines Plug-Ins direkt mit der Anwendung interagieren. Der zweite Standard, der vorgestellt wurde, ist der Web Terrain Service. Mit dessen Hilfe können ebenfalls 3D-Geodaten in einem dreidimensionalen Raum visualisiert werden. Über diesen Dienst erhält der Nutzer als Ergebnis ein Pixelbild angezeigt. Mit den von ihm gesendeten Parametern wird die Perspektive auf die Daten definiert und das Pixelbild entsprechend erstellt. Der dritte Dienst ist der Web Perspective View Service. Er soll den WTS später ersetzen und funktioniert daher auf der gleichen Basis wie er. Mit dem WPVS wird, im Gegensatz zum WTS, eine größere Bandbreite an Anwendungsmöglichkeiten abgedeckt.

Neben den Diensten sind auch standardisierte Formate zum Ablegen und Austauschen von dreidimensionalen Geodaten wichtig. Für ein 3D-WebGIS können die Formate VRML, X3D, CityGML und KML in Frage kommen. Virtual Reality Modeling Language ist das älteste von diesen Formaten und von der ISO standardisiert. Es verwendet den Szenegraphen und beinhaltet neben den Daten selbst und deren Darstellung auch Informationen über Beleuchtung und Kamerapositionen. Extensible 3D ist eine Weiterentwicklung von VRML und ebenfalls ein ISO-Standard. Als

Dokumentenbeschreibungssprache basiert X3D auf dem bekannten XML. Im Gegensatz zu VRML ist es modular aufgebaut und daher flexibel einsetzbar. City Geography Markup Language ist ein Austauschformat und semantisches Objektmodel für 3D-Objekte. Es basiert ebenfalls auf XML und wurde speziell für 3D-Stadtmodelle entwickelt. CityGML zählt zu den offiziellen Standards des OGC genauso wie KML. Keyhole Markup Language ist ebenfalls eine Beschreibungssprache allerdings für Datenmodelle, die in Google Earth visualisiert werden können und basiert auf XML.

Mit Hilfe dieser Dienste und Formate gibt es verschiedene Möglichkeiten ein 3D-WebGIS aufzubauen, wie im weiteren Verlauf der Arbeit beschrieben wurde. Die Architektur des Systems ist zum Einen von den Anforderungen an die Applikation abhängig und zum Anderen davon, wie die einzelnen Prozesse auf den Client und den Server aufgeteilt werden. Daraus ergeben sich die beschriebenen Aufgabenverteilungen und Systemanforderungen der beiden Komponenten. Dieser Aufteilung entsprechend können die Dienste und Formate eingesetzt werden.

An die Betrachtung dieser theoretischen Ansätze für den Aufbau eines 3D-WebGIS schließt sich die Erarbeitung eines Lösungskonzeptes an. Zuerst wurden dazu die Anforderungen an die Daten, die Datenbank und das Datenmodell formuliert.

Neben der Datenbank Oracle Spatial wurden die Werkzeuge LandXplorer und das deegree-Projekt vorgestellt, die bei der Umsetzung Anwendung finden sollen. Der LandXplorer der Firma *Autodesk* ermöglicht das Management komplexer Datenmodelle mit zwei- und dreidimensionalen Geodaten. Er soll für die Datenaufbereitung zur Darstellung der Daten in einer 3D-Szene verwendet werden. Da der LandXplorer nur in der Lage ist auf eine Datenbank von *Oracle* oder die ArcSDE von *ESRI* zuzugreifen, ist es wichtig, den Datenbestand in eine Oracle Spatial Datenbank zu migrieren. Die Implementierung des 3D-WebGIS soll mit Hilfe des deegree-Projektes durchgeführt werden. Es handelt sich dabei um ein Open Source System, welches auf Java basiert und unterschiedliche Komponenten zur Erstellung verschiedener Web Services beinhaltet.

Nach der Beschreibung der Datengrundlage und den Werkzeugen folgte die Beschreibung des Projektkonzeptes. Zuerst wurden die Anforderungen an die Anwendung definiert und anschließend die Datenaufbereitung erläutert. Daran schloss sich der Entwurf einer Systemarchitektur an. Dabei handelt es sich um den deegree-WPVS, der den gestellten Anforderungen am Besten entspricht. Er generiert „on the

fly“ die Ansicht der 3D-Szene aus dem Datenspeicher, indem er über lokale Dienste darauf zugreift.

Wenn man die in Punkt 3.4 betrachteten Charakteristika und Anforderungen für ein WebGIS nach [ZIPF 2000] erneut betrachtet und dabei das vorgeschlagene Lösungskonzept mit einbezieht, so ergeben sich für eine 3D-WebGIS die folgenden Charakteristika:

- Passivität:
  Die Anwendung ist intuitiv und einfach bedienbar. Durch die Ähnlichkeit mit den bekannten Web Map Services, fällt es einem Nutzer leicht, sich zu Recht zu finden.
- Betrachtungsdauer:
  Die Daten müssen so aufbereitet werden, dass sich eine übersichtliche 3D-Szene ergibt. Durch eine Übersichtskarte wird die Orientierung für den Nutzer zusätzlich vereinfacht. Für die Betrachtungsdauer ist auch die Ladezeit der Daten wichtig. Dies müsste noch getestet werden, kann aber über die Auflösung der Daten geregelt werden.
- Aktualisierbarkeit:
  Durch den „on the fly“-Zugriff auf den Datenbestand stehen dem Nutzer immer die aktuellsten Daten zur Verfügung.
- Individualisierbarkeit:
  Der Nutzer kann entscheiden, welche Daten er sehen möchte. Allerdings hat er keinen Einfluss auf die Darstellung der Daten. Somit ist nur ein Teil der Individualisierbarkeit erfüllt.
- Anonymität:
  Der Nutzer muss keine eigenen Daten von sich preisgeben.

Trotz dass die vorgeschlagene Lösung den Anforderungen an ein WebGIS entspricht, ist sie dennoch ausbaufähig. So wäre es sinnvoll, die Anwendung so weiter zu entwickeln, dass eine dynamische Navigation in der Grafik erfolgen kann. Neben den eher etwas sporadischen Navigationsmöglichkeiten, fehlt der Anwendung aber noch ein entscheidender Punkt, die Abfrage von Sachdaten zu einem bestimmten Objekt, so wie man es von vielen WMS-Diensten kennt. Dies ist bis jetzt noch nicht möglich. Dazu

müssen die Standards und Softwareprodukte für die Umsetzung von Plug-In freien Anwendungen noch weiterentwickelt werden.

In dem vorgeschlagenem Lösungskonzept wurde davon abgesehen, die Daten als dreidimensionale Objekte, also Volumenkörper, in der Datenbank abzulegen. Oracle Spatial bietet Tools für eine entsprechende Migration der Daten an, allerdings ist es fraglich, ob die betrachteten Komponenten solche Objekte verarbeiten können. So wird momentan von der Universität Bonn ein Datenbankschema für die Oracle Spatial 11g entwickelt, welches der CityGML Spezifikation entspricht. Es ist aber noch nicht bekannt, wie solche Datenbestände über die vorgestellten standardisierten Dienste visualisiert werden können. Dieses Datenbankschema sieht nach wie vor die explizite Abspeicherung unterschiedlicher Levels of Detail für die einzelnen Objekte vor, da die Version 11 von Oracle Spatial keine Unterstützung für Detailstufen bietet.

Nach intensiven Recherchen hat sich ergeben, dass momentan der deegree-WPVS die einzige Anwendung ist, die ein 3D-WebGIS ohne Plug-In ermöglicht. Allerdings ist die Abstimmung der Anwendung auf einen bestimmten Datenbestand sehr aufwendig. In verschiedensten Konfigurationsdateien müssen viele Parameter angepasst werden. Eine ganze Reihe davon, wie z.B. die Ausdehnung des gesamten Datenbestandes, legt man fast in jeder Datei fest. Dies soll sich allerdings ändern. Die Entwickler des deegree-Projektes haben bereits deegree3 angekündigt. In dieser Version soll vor allem die Konfiguration der Dienste vereinfacht werden.

Zusammenfassend lässt sich sagen, dass mit dem vorgeschlagenen Lösungskonzept eine einfache, aber zufriedenstellende 3D-WebGIS Anwendung erstellt werden kann, die auf einem standardisierten Dienst basiert. Betrachtet man die momentane Bandbreite an zur Verfügung stehenden Diensten und Formaten für den Bereich der Visualisierung von dreidimensionalen Daten über das Internet, muss man feststellen, dass es noch nicht viele Lösungsvarianten gibt. Die Forschung in diesem Gebiet steht erst am Anfang und man darf gespannt sein, was sich in den nächsten Jahren diesbezüglich tun wird.

Literaturquellen

BEZEMA, R., U. MÜLLER, M., RUBACH, H., POTH, A. & TADDEI, U. (2008). Interoperabilität für 3D-Geodaten – Erfahrungen mit CityGML und OGC Web Services. In: STROBL, J., BLASCHKE, T. & GRIESEBNER, G. (Hrsg.), Angewandte Geoinformatik 2007. Beiträge zum 19. AGIT- Symposium Salzburg, S. 75-84. Heidelberg: Wichmann. ISBN: 978-3-87907-451-8

BRINKHOFF, T. (2008). Geodatenbanksysteme in Theorie und Praxis: Einführung in objektrelationale Geodatenbanken unter besonderer Berücksichtigung von Oracle Spatial. 2., überarbeitete und erweiterte Auflage. Heidelberg: Wichmann. ISBN: 978-3-87907-472-3

COORS, V. & ZIPF, A. (2004). 3D-Geoinformationssysteme, Grundlagen und Anwendungen.1. Auflage. Heidelberg: Wichmann. ISBN: 3-87907-411-9

DÖLLNER, J. (2006). LandXplorer – ein Werkzeug für komplexe Geoinformationen auf Grundlage virtueller 3D-Stadt- und 3D-Landschaftsmodelle. http://gutachterausschuss-bb.de/GeoPortal1/produkte/verm_bb/pdf/1_06_Doellner_32-39.pdf (20.12.2008)

DOYLE, A. & CUTHBERT, A. (1998): Essential Model of Interactive Portrayal. OpenGIS Project Document 98-061. http://www.opengis.org

FREIWALD, N. & JANY, R (2004). Dateiformate für vektorbasierte 3D-Geodaten. In: COORS, V. & ZIPF, A. (Hrsg.). 3D-Geoinformationssysteme, Grundlagen und Anwendungen. S. 142-158. 1. Auflage. Heidelberg: Wichmann. ISBN: 3-87907-411-9

KOLBE, T. H. (2004). Interoperable 3D-Visualisierung („3D Web Map Server“). von http://www.ikg.uni-bonn.de/uploads/tx_ikgpublication/Koenigslutter_2004_Kolbe.pdf (22.11.2008)

KORDUAN, P. & ZEHNER, M. L. (2007). Geoinformation im Internet. 1. Auflage. Heidelberg: Wichmann. ISBN: 978-3-87907-456-3

KOTHURI, R.V., GODFRIND, A. & BEINAT, E. (2007). Pro Oracle Spatial for Oracle Database 11g. 2. Auflage. Berkeley: Apress. ISBN: 978-1-59059-899-3

LUPP, M. (2008). deegree Free Software. In: SHEKHAR, S. & Xiong, H. (Hrsg.). Encyclopedia of GIS. S. 235-239. New York: Springer. ISBN: 978-0-387-359753-1

MITCHELL, T., EMDE, A., CHRISTL, A. & Lang, J. (2008). Web-Mapping mit Open Source-GIS-Tools. 1. Auflage. Köln: O'Reilly. ISBN: 978-3-89721-723-2

MÜLLER, W. (2004). Grundlagen der 3D-Computergrafik. In: COORS, V. & ZIPF, A. (Hrsg.). 3D-Geoinformationssysteme, Grundlagen und Anwendungen. S. 184-201. 1. Auflage. Heidelberg: Wichmann. ISBN: 3-87907-411-9

MÜLLER, W. (2005). Interoperabilität: Nicht nur eine Frage der Technologie. von http://www.gis.ethz.ch/Interoperability2005/Text/Interop_16_DE.pdf (06.06.1009)

NEUBAUER, S. & ZIPF, A. (2007). Vorschläge zur Erweiterung der OGC Styled Layer Descriptor (SLD) Specification in die dritte Dimension – eine Analyse möglicher Visualisierungvorschriften für 3D Stadtmodelle. In: STROBL, J., BLASCHKE, T. & GRIESEBNER, G. (Hrsg.), Angewandte Geoinformatik 2007. Beiträge zum 19. AGIT- Symposium Salzburg, S. 505-510. Heidelberg: Wichmann. ISBN: 978-3-87907-451-8

OGC (2003). Web Terrain Service. OpenGIS Imlementation Specification. Document Stage: Request for Comment. Version 0.5. Reference Number:OGC 03-081r2 (07.11.2003)

OGC (2005a). Web 3D Service. Document Stage: Approved Discussion Paper. Version 0.3.0. Reference Number: OGC 05-019 (02.02.2005)

OGC (2005b). OpenGIS Web Perspective View Service (WPVS) Implementation Specification. Document Stage: Draft proposed version 1.0. Reference Number: OGC 05-0XX (20.10.2005)

OGC (2005c). Basic Portrayal Additions (BPA) module. Document Stage: Draft proposed version 0.0. Reference Number: OGC 05-0XX (17.10.2005)

OGC (2005d). Client Styling Additions (CSA) module. Document Stage: Draft proposed version 0.0. Reference Number: OGC 05-0XX (18.10.2005)

OGC (2005e). Get Subset Additions (GSA) module. Document Stage: Draft proposed version 0.0. Reference Number: OGC 05-0XX (19.10.2005)

OGC (2005f). Perspective View Additions (PVA) module. Document Stage: Draft proposed version 0.0. Reference Number: OGC 05-0XX (19.09.2005)

OGC (2008). OpenGIS City Geography Markup Language (CityGML) Encoding Standard. Document Stage: Final. Version 1.0.0. Reference Number: OGC 08-007r1 (20.08.2008)

POMASKA, G. (2007). Web-Visualisierung mit Open Source: Vom CAD-Modell zur Real-Time Animation: Vom CAD-Modell zur Real-Time-Animation. 1. Auflage. Heidelberg: Wichmann. ISBN: 978-3-87907-450-1

SCHIEBOLD, M. (2007). Konzeption eines virtuellen Stadtmodells. unveröffentlichte Master Thesis. Paris Lodron-Universität Salzburg bei Ao. Univ. Prof. Dr. Josef Strobl

SCHMIDT, B., MAY, M. & UHLENKÜKEN, C. (2006). Service-basierte 3D-Geovisualisierung – Umsetzungsstrategien und praktische Anwendung. In: STROBL, J., BLASCHKE, T. & GRIESEBNER, G. (Hrsg.), Angewandte Geoinformatik 2005. Beiträge zum 17. AGIT- Symposium Salzburg, S. 629-638. Heidelberg: Wichmann. ISBN: 978-3-87907-422-8

ZACH, C., GRABNER, M., SORMANN, M. und KARNER, K. (2004). Internet-basierte Echtzeitvisualisierung von 3D-Geodaten. In: COORS, V. & ZIPF, A. (Hrsg.). 3D-Geoinformationssysteme, Grundlagen und Anwendungen. S. 202-216. 1. Auflage. Heidelberg: Wichmann. ISBN: 3-87907-411-9

ZIPF, A. (2000). Deep Map, GIS - ein verteiltes raumzeitliches Touristeninformationssystem. Dissertation. Ruprecht-Karls-Universität Heidelberg

Software

Apache Tomcat 6.0, ©The Apache Software Foundation

Autodesk LandXplorer Studio Professional 2009, ©Autodesk Inc.

deegree - Free Software for Spatial Data Infrastructures, Open Source Geospatial Foundation (OSGeo)-Projekt

Oracle Spatial 10g und Oracle Spatial 11g, ©Oracle Corporation