



BACHELORARBEIT

# Räumliche Datenbanken und GML

Zur Erlangung des akademischen Grades  
Bachelor of Science  
(BSc.)

ausgeführt am  
Institut für Rechnergestützte Automation  
Forschungsgruppe Industrial Software

der Technischen Universität Wien

unter der Anleitung von  
Univ.-Prof. Dipl.-Ing. Dr. Thomas Grechenig

durch:  
Johannes Harms  
Josefstädter Straße 14, 1080 Wien

Wien, Februar 2008

## Kurzfassung

Diese Arbeit behandelt das Modellieren, Speichern, Abfragen, Austauschen und Visualisieren von geographischen Daten. Für den Austausch von Geodaten hat sich das XML-basierte Format GML (Geography Markup Language) etabliert. Für die Speicherung räumlicher Daten sind relationale Datenbanksysteme weit verbreitet, haben aber auch Nachteile - insbesondere wenn die zu speichernden Daten im GML Format vorliegen. XML Datenbanken sind eine interessante Alternative, wenn sie räumliche Indizes und Abfragefunktionen bereitstellen.

Im Rahmen dieser Arbeit wird ein Überblick über den Stand der Technik und Forschung gegeben, und es werden Lösungen und offene Probleme beim Umgang mit GML Daten vorgestellt. Für das Projekt "Roncalli" [Roncalli:Website, 2008] mit dem Ziel, eine österreichische, verkehrstelematische Datenbank zu betreiben, wird eine GML basierte Lösung und eine Software-Architektur mit Thin Middleware und geographischer XML Datenbank vorgeschlagen.

## Abstract

This paper presents issues related to the handling of geographic data such as modelling, storage, querying, data exchange, and visualization. The GML (Geography Markup Language) has been become popular for the exchange of geographic data. For the storage of spatial data, relational databases are widely used, but they have disadvantages if used to store data in the GML format. XML databases are an interesting alternative if they provide spatial indexes and querying operators. In this work, an overview of the state of the art is given. Solutions and open problems related to the use of GML data are presented. For the "Roncalli" Projekt [Roncalli:Website, 2008] which aims to establish an Austrian, traffic-related Database and Platform, a GML based solution is proposed and a software architecture featuring a Thin Middleware and a geographic XML database is put forward.

## Schlüsselwörter

Geodateninfrastruktur, Geodatenbanken, räumliche Datenbanken, geographische Middlewares, XML Datenbanken, OGC, SQL, GML Speicherung, GML Abfrage.

## Keywords

Spatial data infrastructure (SDI), Geodatabases, spatial databases, spatial middlewares, GML storage options, GML querying.

# Inhaltsverzeichnis

<b>1. Einleitung</b>	<b>5</b>
1.1. Aufbau der Arbeit . . . . .	5
<b>2. Das vorliegende und andere, ähnliche Projekte</b>	<b>7</b>
2.1. Google Earth und Google Maps . . . . .	7
2.2. INTREST . . . . .	8
2.2.1. Geschäftsmodell . . . . .	8
2.2.2. Austauschplattform . . . . .	9
2.2.3. Datenmodell . . . . .	10
2.3. INSPIRE . . . . .	10
2.3.1. Umsetzung von INSPIRE in Österreich . . . . .	13
2.3.2. Umsetzung von INSPIRE in Deutschland . . . . .	13
2.3.3. Technische Umsetzung . . . . .	14
2.4. EuroRoadS . . . . .	14
2.4.1. Das EuroRoadS Informationsmodell . . . . .	15
2.4.2. Das EuroRoadS Format für den Datenaustausch . . . . .	17
2.5. Das vorliegende Projekt “Verkehrstelematik Datenbank” . . . . .	17
<b>3. Geodatenbanksysteme</b>	<b>21</b>
3.1. Normen und Standards im Bereich Geodatenbanken . . . . .	21
3.1.1. Das OGC Referenzmodell . . . . .	23
3.1.2. Relationale und Objekt-Relationale Geodatenbanken . . . . .	24
3.1.3. Die OGC Implementierungs-Spezifikation für SQL . . . . .	25
3.2. Architekturen von Geodatenbanken . . . . .	25
3.2.1. Middleware als Bindeglied zwischen Geodatenbank und Umgebung . . . . .	26
3.2.2. Thin versus Thick Middlewares . . . . .	27
3.2.3. Einige Produkte räumlicher Middlewares . . . . .	29
3.2.4. Bedeutung für das vorliegende Projekt . . . . .	29
<b>4. XML und GML</b>	<b>31</b>
4.1. Vor- und Nachteile von GML . . . . .	31
4.2. Informationsmodellierung in GML anhand eines Beispiels . . . . .	33
4.2.1. Feature Types im GML Standard . . . . .	33
4.2.2. Die Definition von Features im Applikationsschema . . . . .	35
4.2.3. Feature Instances in GML Dateien . . . . .	36
4.3. Abfragesprachen für GML . . . . .	36

4.3.1.	Geographisch erweitertes XQuery . . . . .	37
4.3.2.	OGC Filter Encoding Spezifikation . . . . .	38
4.4.	Parser für GML . . . . .	38
4.5.	Visualisierung von GML Daten . . . . .	40
4.5.1.	GML Viewer . . . . .	41
4.5.2.	Vektordaten Viewer . . . . .	41
4.5.3.	Bitmap Viewer . . . . .	42
4.6.	GML zur Integration von Geodaten . . . . .	43
4.6.1.	Heterogenität verteilter, autonomer Informationssysteme . . . . .	44
4.6.2.	Ansätze zur Integration von Geodaten aus heterogenen Informationssystemen . . . . .	45
4.6.3.	Geographische Suchmaschinen . . . . .	45
4.6.4.	Mediator-basierte Systeme . . . . .	46
4.7.	Geographische WebServices . . . . .	48
<b>5.</b>	<b>Speicherung von GML</b>	<b>51</b>
5.1.	Speicherplatz und Kompression von GML Daten . . . . .	51
5.2.	Konvertierung von GML zur Speicherung in relationalen Datenbanken . . . . .	52
5.2.1.	Vergleich: Relationale Datenbanken versus XML Datenbanken . . . . .	53
5.3.	Speicherung von GML in Dateien . . . . .	54
5.4.	Speicherung von GML in XML-enabled Datenbanken . . . . .	55
5.4.1.	XML Publishing versus XML Storage Ansätze . . . . .	55
5.4.2.	Data-centric versus Document-centric Anwendungen . . . . .	56
5.4.3.	Schema-based versus Schema-oblivious Ansätze . . . . .	57
5.4.4.	Schema-based Storage: Intelligente Wahl des relationalen Schemas . . . . .	59
5.4.5.	Schema-based Storage: Effiziente XML-zu-SQL Abfrageübersetzung . . . . .	59
5.4.6.	Schema-based Storage: Beispiel-Implementierung für GML Daten . . . . .	60
5.4.7.	Schema-based Storage: Probleme mit der Komplexität des GML Standards . . . . .	60
5.5.	Speicherung von GML in nativen XML Datenbanken . . . . .	61
5.5.1.	Indizierung von GML Daten . . . . .	62
5.6.	Vergleich: Native versus XML-enabled Datenbanken zum Speichern von GML . . . . .	63
<b>6.</b>	<b>Zusammenfassung und Ergebnisse</b>	<b>65</b>
<b>A.</b>	<b>Überblick über ausgewählte Literatur</b>	<b>66</b>
<b>B.</b>	<b>Abbildungsverzeichnis</b>	<b>67</b>
<b>C.</b>	<b>Tabellenverzeichnis</b>	<b>67</b>
<b>D.</b>	<b>Literaturverzeichnis</b>	<b>68</b>

## 1. Einleitung

Einigen geographischen Informationssystemen ist es gelungen, einen weiten Benutzerkreis anzuziehen, und sie sind deshalb nicht nur zum Abrufen und Betrachten geographischer Information weit verbreitet, sondern haben sich auch als Austauschplattform für geographische Information etablieren können. Diese Arbeit ist Teil eines Projekts zur Erstellung einer österreichischen Austauschplattform für verkehrsbezogene Daten. Aus technischer Sicht ist zur Erstellung eines solchen Informationssystems die Möglichkeit effizienter Speicherung und Abfrage von Geodaten notwendig. Diese Aufgaben und Forschung auf diesem Gebiet ist Thema dieser Arbeit.

Für die Aufgabe des Speicherns und Abfragens von geographischer Information können Techniken verwendet werden, wie sie auch in anderen Domänen üblich sind: Es sind dies insbesondere relationale Datenbanksysteme (RDBS) und die Extensible Markup Language (XML). Diese Arbeit beschreibt die beiden Ansätze RDBS und XML und vergleicht sie im Hinblick auf die Nutzung der jeweiligen Technik zum Erstellen einer Verkehrsdatenbank für das österreichische Straßennetz.

Zum Austausch von Geodaten zwischen heterogenen Systemen hat sich das XML-basierte Format GML etabliert. Im Rahmen dieser Arbeit sollen Vorteile und Nachteile dieses Formats beleuchtet werden. Trotz vieler Vorteile, die für die Verwendung von GML sprechen, ist die effiziente Speicherung und Abfrage von GML Daten ein bisher nur ungenügend gelöstes Problem.

Trotzdem wird für das vorliegende Projekt eine GML-basierte Lösung zur Erstellung der österreichischen Verkehrsdatenbank vorgeschlagen. Die Frage nach effizienter Speicherung und Abfrage der GML Daten wird im Rahmen dieser Arbeit diskutiert. Es gibt dafür zwei Möglichkeiten: In Frage kommt (a) eine XML-enabled Datenbank mit schema-based Mapping oder (b) eine native XML Datenbank, die um räumliche Datentypen, Indizes und Abfrage-Operatoren erweitert wurde. Welche der beiden Möglichkeiten für das vorliegende Projekt und für ähnliche Projekte besser geeignet ist, kann im Rahmen dieser Arbeit nicht entschieden werden - dazu fehlen vergleichende Studien und Benchmarks.

Als Software-Architektur wird für das Projekt eine Architektur mit einer Thin Middleware vorgeschlagen, d.h. im Gegensatz zu einer Thick Middleware sollen einfache räumliche Kriterien in Abfrage-Ausdrücken in der Datenbank (und nicht in der Middleware) ausgewertet werden.

### 1.1. Aufbau der Arbeit

Das folgende Kapitel 2 erläutert das vorliegende Projekt zur Erstellung einer österreichischen Verkehrsdatenbank und beschreibt andere Projekte mit ähnlicher Aufgabenstellung.

In Kapitel 3 wird ein Geodatenbanksystem als ein System mit allen Fähigkeiten eines üblichen Datenbanksystems charakterisiert, das jedoch zusätzliche Unterstützung für räumliche Daten hat. In diesem Sinn können relationale Datenbanksysteme für die Verwendung mit geographischen Daten erweitert werden. Normen und Standards, die in diesem Bereich relevant sind, werden vorgestellt.

Abschnitt 3.2 beschreibt mögliche Architekturen bei der Implementierung eines geographischen Datenbanksystems und unterscheidet *thick* und *thin* Middlewares für geographische Daten: Thin Middlewares im Sinne dieser Unterscheidung verwenden einen Datenspeicher, der ein erstes Filtern anhand von einfachen räumlichen Abfragen ermöglicht. Thick Middlewares verwenden einen Datenspeicher, der keine derartige Funktion hat. Das heißt, auch einfache räumliche Abfragekriterien müssen in der Middleware statt im Datenspeicher ausgewertet werden. Für das Projekt einer österreichischen Verkehrsdatenbank wird eine Architektur mit thin Middleware vorgeschlagen.

Kapitel 4 stellt das XML-basierte Format GML vor, das zum Modellieren, Speichern und Austauschen geographischer Information verwendet wird, und geht auf verschiedene Aspekte im Umgang mit GML Daten ein.

Auf den Aspekt der effizienten Speicherung und Abfrage von GML wird in Kapitel 5 ein besonderer Schwerpunkt gelegt. Zur Speicherung von GML Daten können Dateien, relationale Datenbanken, XML-enabled Datenbanken oder native XML Datenbanken verwendet werden. Vor- und Nachteile dieser Ansätze werden verglichen.

Kapitel 6 fasst die Ergebnisse dieser Arbeit zusammen. Anhang A enthält eine Liste ausgewählter Literatur, um dem Leser dieser Arbeit anhand weniger Arbeiten tiefere Kenntnis über das im Rahmen dieser Arbeit diskutierte Themenfeld zu verschaffen.

## 2. Das vorliegende und andere, ähnliche Projekte

Im Folgenden sollen Projekte beschrieben werden, deren Aufgabenstellungen und Herausforderungen auch für das vorliegende Projekt (Schaffung einer geographischen Austauschplattform für österreichische, verkehrsbezogene Daten, siehe Abschnitt 2.5) relevant sind. Im Anschluss daran soll das vorliegende Projekt zur Erstellung einer österreichischen Verkehrsdatenbank näher beschrieben werden.

### 2.1. Google Earth und Google Maps

In Bezug auf die Popularität geographischer Informationssysteme und deren Verwendung als Austausch-Plattform sind Google Earth und Google Maps für das vorliegende Projekt interessant:

Anwendungen wie Google Earth [GoogleEarth:Website, 2007], Google Maps [Google:Maps, 2007] und Nasa World Wind [Nasa:Worldwind, 2007] bilden die Welt in großer Detailtreue ab, und lassen sich über ein einfaches User-Interface navigieren. Im Gegensatz zu vielen anderen geo-informatischen Produkten ist es diesen Anwendungen gelungen, auch nicht-geographische Benutzergruppen anzuziehen, und sich innerhalb dieser Gruppen als geographische Austausch-Plattform zu etablieren.

So sind rund um Google Earth und Google Maps in letzter Zeit (2008) eine Menge neuer Anwendungen entstanden. Der Blog “Cool Google Maps” [CoolGoogleMaps, 2007] listet originelle Beispiele auf - angefangen mit einer Karte mit Hospital Rankings bis hin zum Reiseweg des Santa Claus beim Abliefern der Geschenke. Aber auch Apple’s iPhone [Apple:iPhone, 2007] verwendet Google Maps. Und eine Datenbank über Radarkästen und Verkehrskameras [SpeedCameraDB:GoogleEarthPlugin, 2007] stellt ein Plugin für Google Earth zur Verfügung, das Radarkontrollen in ganz Europa anzeigen kann.

Auch in der Wissenschaft finden Google’s geographische Anwendungen zunehmend Beachtung: Die Autoren W.Ding et.al berechnen aus den Gebäude-Schatten auf den Google Earth Satellitenbildern die Höhe der Gebäude und können so ein 3D Modell einer Stadt erstellen [Ding u. a., 22–24 Aug. 2007]. Die Autoren K.Honda et.al wollen die Popularität und Benutzerfreundlichkeit von Google Earth zum Publizieren geographischer Daten nutzen, und haben deshalb einen Konverter implementiert, der Daten aus OGC Webservices (siehe Abschnitt 4.7) zum Google-Earth basierten Format KML übersetzen kann [Honda u. a., Oct. 2006].

Für die Verwendung als Austauschplattform hat Google folgendes Konzept: Google Earth ermöglicht es auch geographisch ungeschulten Benutzern, eigene Information zum System hinzuzufügen - was aufgrund der Popularität von Google Earth auch laufend geschieht. Dabei ist KML [Google:KML, 2007] (Keyhole Markup Language) das von Google verwendete Format zum Austauschen geographischer Features in den Google Earth und Google Maps Anwendungen. KML basiert auf dem XML Format und ist GML sehr ähnlich, dem vom OGC (Open Geospatial Consortium, [OGC, 2007b]) empfohlenen Format für Modellierung und Austausch geographischer Information. GML wird in den Kapiteln 4 und 5 näher beschrieben.

Außer den Geodaten, die von Benutzern im KML Format erstellt wurden, integriert Google Earth auch noch Geodaten aus sozialen Webanwendungen. Zum Beispiel können Wikipedia [Wikipedia:Website, 2007] Einträge, sofern sie eine Koordinatenangabe enthalten, in einer eigenen Kartenebene in Google Earth angezeigt werden. Ähnliches gilt für georeferenzierte Fotos von Panoramio.com oder Videos von YouTube.com.

## 2.2. INTREST

So wie auch das vorliegende Projekt ist INTREST [Intrest, 2007] als Austauschplattform für verkehrsbezogene Daten konzipiert: INTREST ist eine Plattform für verkehrsaffine Geodaten in Bayern. Im Folgenden soll das Projekt näher beschrieben werden.

### 2.2.1. Geschäftsmodell

Das INTREST Projekt formuliert als sein Geschäftsziel [INTREST:Info5]: “Der INTREST-Betrieb hat das Ziel, verkehrsrelevanten [sic!] Informationen unterschiedlicher Lieferanten in einem geographischen Referenzierungssystem zu sammeln und diese im Auftrag (kommerziell) zu vertreiben.” Zur Verwirklichung dieses Geschäftszieles ist das Projekt auf die Zusammenarbeit mit zwei Benutzergruppen angewiesen:

*Datenabnehmer:* Datenabnehmer sind die Kunden von INTREST. Während ihnen die Infrastruktur des INTREST Projekts frei zur Verfügung gestellt wird, bezahlen Datenabnehmer für die Auslieferung der Geodaten (der Geldbetrag wird über ein Datenbewertungsmodell ermittelt). Auch für das Referenzieren auf INTREST Daten zum Austausch mit Dritten ist eine Pauschale zu bezahlen, es sei denn die referenzierenden Daten werden INTREST zur Verfügung gestellt - dann wird ein Betrag gutgeschrieben. Als Vorteile für den Datenabnehmer werden genannt: INTREST bietet digitales Kartenmaterial und einheitlich referenzierte Geodaten aus verschiedenen Anwendungsgebieten. Eigene Daten können damit ergänzt werden, und INTREST Daten lassen sich um eigene Daten ergänzen. Auch wenn INTREST Daten um eigene Daten ergänzt wurden, bleibt das Aktualisieren der INTREST Daten vergleichsweise mühe-los.

*Datenlieferanten:* Datenlieferanten sind die zweite Benutzergruppe. Als Austauschplattform ist INTREST auf Datenlieferanten angewiesen, die ihre Daten über INTREST verbreiten und vermarkten. Deshalb will INTREST die Lieferanten am Ertrag beteiligen - die zu erwartenden Beträge seien jedoch nicht hoch. Andere Motivatoren für Datenlieferanten sind: Über INTREST zur Verfügung gestellte Daten haben eine einheitliche Referenzierungsbasis und können deshalb leichter zwischen INTREST Nutzern getauscht werden. Der Austausch von Daten soll auch durch die Preisgestaltung unterstützt werden: Nutzer, die Daten zur Verfügung stellen, können im Gegenzug Daten vergünstigt aus INTREST beziehen.



Um, wie im Geschäftsziel formuliert, Information verschiedener Quellen in einem System zu vereinheitlichen, braucht INTREST eine aktuelle Kartenbasis, um neue Daten darin referenzieren zu können. Der folgende Abschnitt beschreibt die dazu notwendige Organisation und Technik.

### 2.2.2. Austauschplattform

Um INTREST als Austauschplattform mit vielen Beteiligten verwenden zu können, ist die Definition offengelegter Schnittstellen und die Nutzung dezentralen Fachwissens wichtig [INTREST:Info1]:

*Offengelegte Schnittstellen:* Um im Sinne einer Austausch-Plattform die Nutzung und Wartung der Geodaten durch mehrere Parteien zu ermöglichen, bietet INTREST offengelegte Schnittstellen und Möglichkeiten zur permanenten Referenzierung von Geo-Objekten.

*Dezentrale Datenerfassung und -Verwaltung:* Die Datenerfassung der geographischen Inhalte des INTREST Systems erfolgt nicht von einer zentralen Stelle. Vielmehr soll das Wissen vieler unterschiedlicher Datenlieferanten genutzt werden, indem Organisationen wie Gemeinden oder Verkehrsverbünde die für sie relevanten Daten selbst bearbeiten können. Aber nicht nur die Daten-Erfassung, sondern auch die Datenhaltung kann dezentral erfolgen: Fachspezifische oder sehr detaillierte Daten (zum Beispiel aus dem Baustellen-Management oder Straßenbau) müssen nicht im INTREST System gespeichert, sondern können von externen Systemen bereitgestellt werden. Dementsprechend beinhaltet auch das Intrest Datenmodell nur eine Auswahl wichtiger Objekte. Abbildung 1 verdeutlicht diesen Zusammenhang und zeigt den von INTREST verwalteten Kern wichtiger Daten und die extern verwalteten zusätzlichen Daten.

### 2.2.3. Datenmodell

*Multimodaler Ansatz:* In INTREST wird nicht nur das Straßennetz (für die Nutzung mit dem PKW) modelliert, sondern auch Transportwege für Fußgänger, Radfahrer, LKW und Schwer- oder Gefahrentransporte, Notfallfahrzeuge, Taxis, öffentlicher Verkehr, Seilbahnen und Wasserwege. Häufig sind Wege im INTREST Verkehrsnetz multimodal, sie können von mehreren Verkehrsmitteln benützt werden. Zur Modellierung der multimodalen Verkehrswege verwendet INTREST ein Netzmodell, in dem jede Kante mehrere Verkehrsmittel unterstützen kann: So kann z.B. ein und die selbe Kante eine Straße, einen Fußweg und Straßenbahnschienen tragen. Als Beispielanwendung wurde eine multimodale Reiseauskunft mit Parkplatzwahl und Fußwegauskunft für den Flughafen München implementiert.

*Zeitlich veränderliche Daten:* Wie bereits erwähnt verwaltet INTREST multimodale Verkehrsdaten, also Verkehrsdaten für mehrere Fortbewegungsmittel, z.B. Auto, öffentlicher Verkehr oder zu Fuß. Diese Daten werden in einem Wegenetz verwaltet. Dabei wird auch auf zeitlich veränderliche Zusammenhänge Rücksicht genommen. Das sind zum Beispiel Verkehrsmeldungen, Baustellendaten oder die Fahrpläne öffentlicher Verkehrsmittel. Zur Berechnung einer voraussichtlichen Reisezeit entlang eines Weges werden Wegzeiten in INTREST gesammelt, und statistisch analysiert.

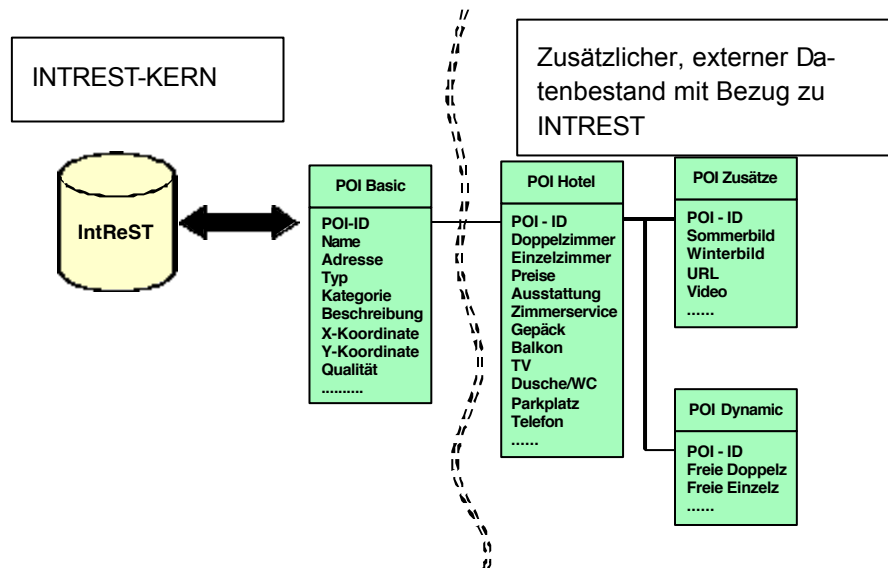


Abbildung 1: INTREST erweitertes Datenmodell und dezentrale Verwaltung: In INTREST wird nur ein Kern einfacher und wichtiger Daten gehalten. Detaillierte oder spezielle Daten werden extern verwaltet. [INTREST:Info2]

*Weitere geographische Daten:* Zusätzliche Information, die für eine optisch ansprechende und benutzerfreundliche Karte benötigt wird, sind Hintergrundschichten mit Wäldern, Seen, Grenzen und bebauten Gebiete je nach Nutzung, Points of Interest und Adressdaten (Postleitzahlen und Hausnummern).

*Relationales Datenmodell:* In der technischen Umsetzung verwendet das INTREST Projekt ein relationales Datenmodell zur Modellierung und Speicherung der geographischen Information. Obwohl das Projekt (wie in Abbildung 1 dargestellt) nur einen Kern der wichtigsten Datenobjekten bereitstellt, ist das Datenmodell sehr kompliziert und umfasst über 100 Relationen. Abbildung 2 vermittelt einen Eindruck dieser Komplexität.

*Erweiterbares Datenmodell:* Das Datenmodell ist auf hohe Erweiterbarkeit hin ausgelegt. Es modelliert generische Basisobjekte, die für jeweilige Anwendungen erweitert werden können. Im Gegensatz zu einer zentralen Datenerfassung können Nutzer des INTREST Systems selbst Daten ergänzend einbringen. Beispielsweise könnte eine Gemeinde Bearbeitungsrechte für die Geodaten in ihrem eigenen Gebiet erhalten. Dann könnte sie Rad- und Wanderwege ergänzen und korrigieren und damit (im eigenen Interesse) sicherstellen, dass die Gemeinde in allen auf INTREST basierenden Anwendungen richtig dargestellt wird.

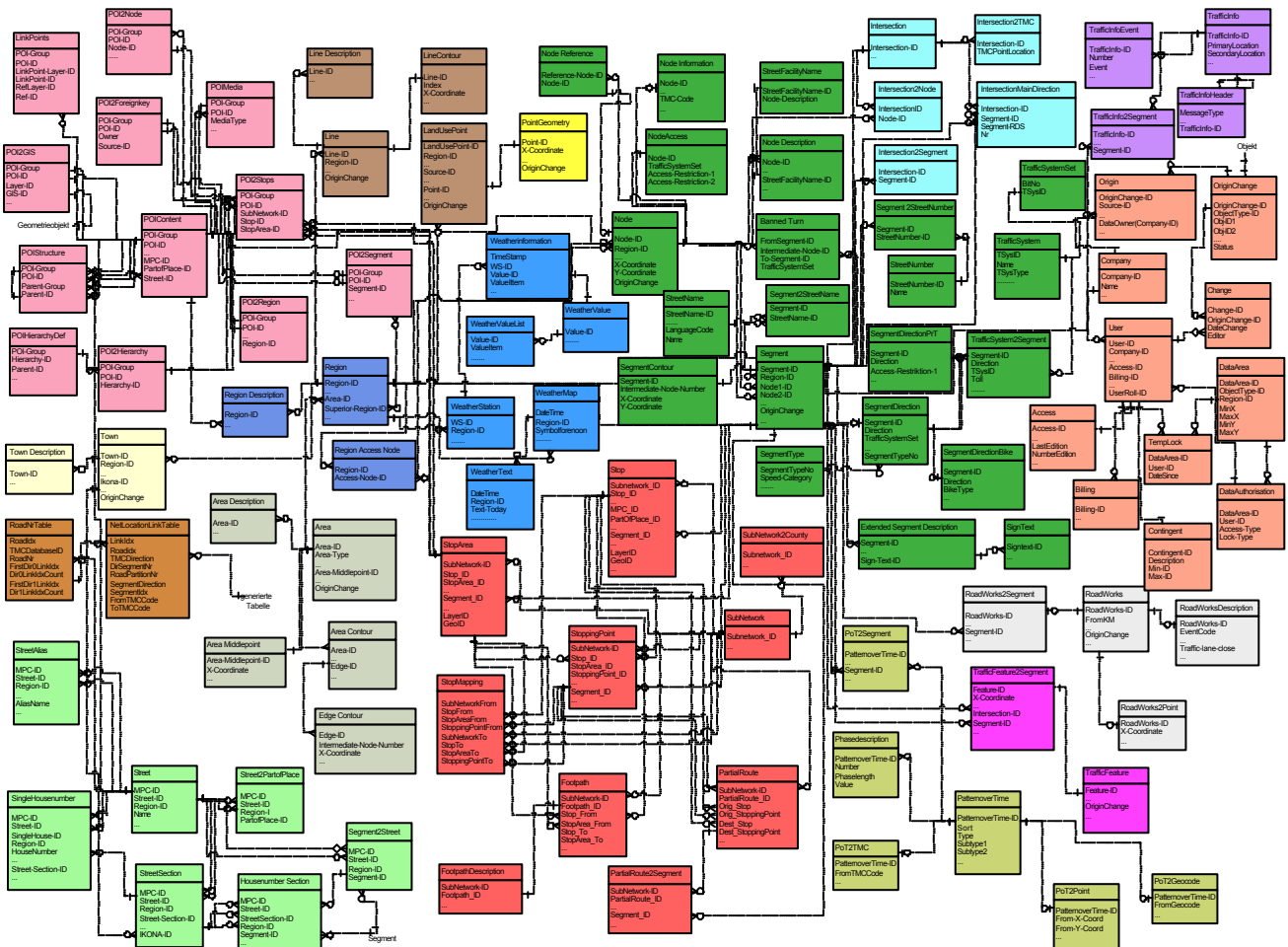


Abbildung 2: Die Komplexität des relationalen Datenmodells von INTREST. Aus [INTREST:Info2]

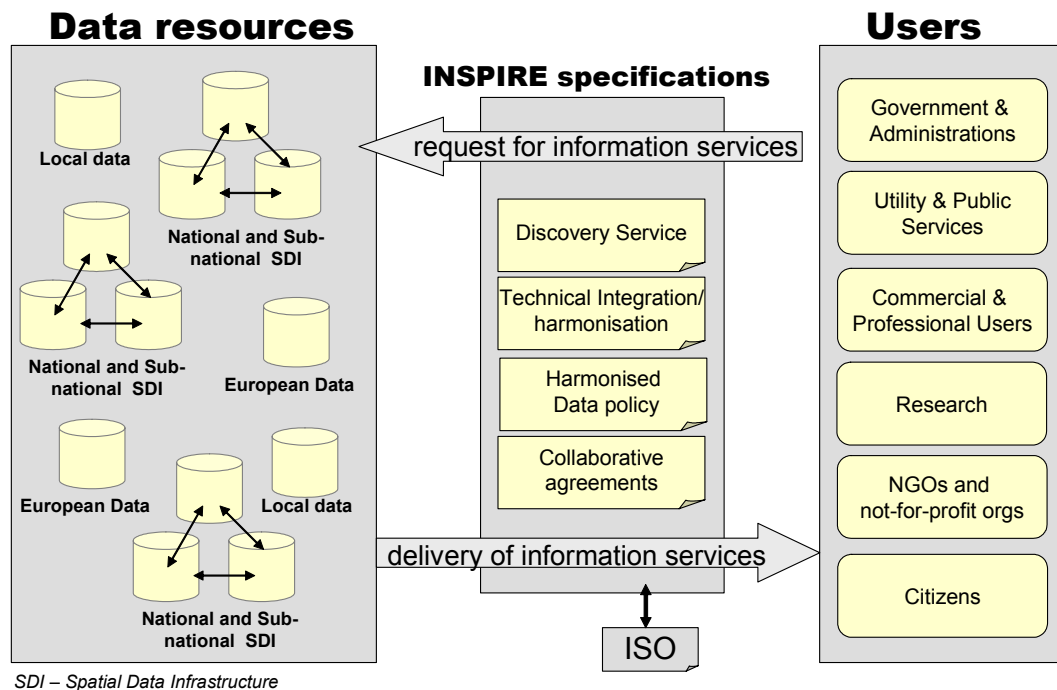


Abbildung 3: INSPIRE Informationsfluss: INSPIRE spezifiziert einheitliche, ISO-konforme Schnittstellen für den Zugriff auf lokal vorhandene Geodaten. Aus [INSPIRE European Geportal, 2002, Abb. 1.1]

### 2.3. INSPIRE

So wie auch das vorliegende Projekt steht INSPIRE [Europäisches Parlament und Rat der europäischen Union, 2007] vor der Herausforderung, Daten aus heterogenen Quellen in ein Gesamtsystem zu integrieren:

INSPIRE hat den Austausch und die gemeinsame Nutzung von Geodaten und Geodatendiensten über verschiedene Verwaltungsebenen und Sektoren der Europäischen Union hinweg zum Ziel. Dadurch soll insbesondere die Umweltpolitik der europäischen Gemeinschaft unterstützt werden. Dieses Ziel soll erreicht werden, indem vorhandene Geodaten-Infrastrukturen der einzelnen Mitgliedsstaaten der Europäischen Union miteinander kompatibel gemacht werden. Technisch gesehen sollen - wie in Abbildung 3 ersichtlich - die Daten der Mitgliedsstaaten über einheitliche, von INSPIRE spezifizierte Schnittstellen auffindbar und abfragbar gemacht werden.

*Rechtliches und Organisatorisches:* INSPIRE ist eine von der europäischen Kommission vorgeschlagene Gesetzesrichtlinie, die seit Mai 2007 in Kraft ist. Die Mitgliedsstaaten der europäischen Union sind für die rechtliche und technische Umsetzung der Richtlinie verantwortlich, und müssen bis Mai 2009 Geodaten wie von der Richtlinie gefordert bereitstellen.

*Nutzungs-Bestimmungen:* INSPIRE verlangt, dass die von der Richtlinie betroffenen Geodaten der europäischen Union kostenlos zur Verfügung gestellt werden. Für die Verwendung der Daten durch

andere Nutzergruppen (kommerzielle, wissenschaftliche, gemeinnützige oder private Verwendung; siehe die Nutzergruppen in Abbildung 3) beinhaltet INSPIRE jedoch kein Geschäftsmodell und keine Lizenzbedingungen, sondern ist im Gegenteil sehr unklar: Es wird verlangt, dass Suchdienste und Darstellungsdienste kostenlos der Öffentlichkeit zur Verfügung gestellt werden [Europäisches Parlament und Rat der europäischen Union, 2007, Artikel 14] - diese Forderung wird jedoch noch im selben Artikel wieder eingeschränkt. So wird den Diensteanbietern die Möglichkeit eingeräumt, kostendeckende Gebühren zu verlangen und die Daten in einer Form anzubieten, die kommerzielle Weiterverwendung ausschließt. Eindeutig ist die Richtlinie jedoch in der Forderung, dass die Geodaten und die damit verbundenen Datendienste kostenlos der europäischen Gemeinschaft zur Verfügung gestellt werden müssen [Europäisches Parlament und Rat der europäischen Union, 2007, Artikel 17].

*Inhaltliche Bestimmungen:* INSPIRE schreibt nicht das Erfassen neuer Geodaten vor, sondern nur die Bereitstellung vorhandener Daten. Was für Daten bereitgestellt werden müssen, ist den Anhängen I bis III der Inspire Richtlinie [Europäisches Parlament und Rat der europäischen Union, 2007] zu entnehmen. Bis Mai 2009 müssen die Daten aus Anhang I und II zur Verfügung gestellt werden. Das sind u.a. Ortsnamen und Adressen, Verwaltungseinheiten und Schutzgebiete, Grundbuchdaten, Verkehrs- und Gewässernetze, ein digitales Höhenmodell und Orthofotographien.

### 2.3.1. Umsetzung von INSPIRE in Österreich

In Österreich ist ein Projekt unter der Leitung von Maximilian Pock vom österreichischen Lebensministerium für die Umsetzung der INSPIRE Richtlinie zuständig [Pock, 2007]. Ziele des Projekts sind:

*Vorarbeiten:* Ermittlung der von INSPIRE betroffenen Geodaten und der dafür zuständigen Behörden. In Österreich sind das in erster Linie das Bundesministerium für Land- und Forstwirtschaft, Umwelt und Wasserwirtschaft (BMLFUW), das Umweltbundesamt (UBA), die Bundesländer und Gemeinden, die Statistik Österreich (ÖSTAT) und das Bundesamt für Eich- und Vermessungswesen (BEV) [Pock, 2007].

*Rechtliche Umsetzung:* Das Projekt soll zeitliche Vorgaben für die rechtliche Umsetzung von INSPIRE in Österreich schaffen, und ist dann auch für die konkrete bundesrechtliche Umsetzung zuständig. Die Umsetzung in Landesrecht ist nicht Teil des Projekts, soll aber durch eine Gesetzesvorlage unterstützt werden.

*Technische Vorgaben:* Es sollen Vorgaben und ein konkreter Plan für die technische Umsetzung geschaffen werden. Die technische Umsetzung selbst ist aber nicht Teil des Projekts, sondern soll in Folgeprojekten organisiert werden.

*Lizenzbedingungen:* Im Rahmen des Projekts sollen Lizenzbedingungen für den Datenaustausch zwischen Behörden entworfen werden.

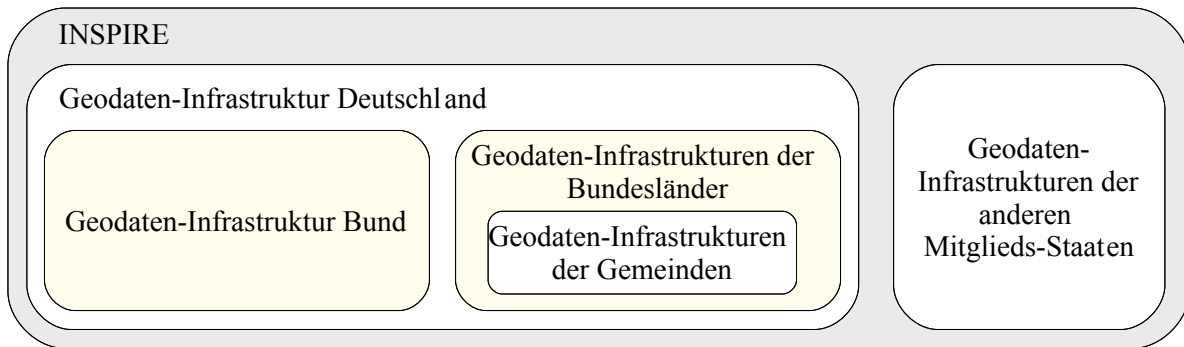


Abbildung 4: INSPIRE und Deutschlands Geodateninfrastruktur GDI-DE im Zusammenspiel [GDI-DE, Abb. 1]

### 2.3.2. Umsetzung von INSPIRE in Deutschland

Die INSPIRE Richtlinie kann Anlass sein, im Zuge ihrer Umsetzung eine nationale Geodateninfrastruktur aufzusetzen, die nicht nur der europäischen Gemeinschaft die geforderten Daten bereitstellt, sondern auch im eigenen Land ein breites Spektrum an Nutzern hat. Eine solche Geodateninfrastruktur könnte der nationalen Regierung und ihren Behörden, der Wissenschaft und der Wirtschaft Geodaten einem Geschäftsmodell entsprechend zur Verfügung stellen.

Deutschland geht diesen Weg, und will die (zeitlich bereits vor INSPIRE initiierte) Geodaten-Infrastruktur “GDI-DE” aufbauen, deren Ziel über das Erfüllen der INSPIRE Richtlinie hinausgeht. Abbildung 4 zeigt, wie Deutschlands Infrastruktur die Geodaten von Bund und Ländern vereinheitlicht. Auf diese Weise kann Deutschland auch die von der INSPIRE Richtlinie geforderten Daten bereitstellen, und ist zugleich über INSPIRE mit den Geodaten-Infrastrukturen der anderen Mitglieds-Staaten verbunden. Österreich hat nach Wissen des Autors zur Zeit keine entsprechenden Pläne für den Aufbau einer nationalen Geodaten-Infrastruktur.

### 2.3.3. Technische Umsetzung

Für die technische Umsetzung macht der INSPIRE Gesetzestext keine Vorgaben. Es arbeiten jedoch “Drafting Teams” und die “Architecture and Standards (AST) Working Group” an technischen Empfehlungen mit dem Ziel, dass diese Empfehlungen in den Gesetzestext aufgenommen werden [INSPIRE European Geoportal, 2002, Kap. 1.5]. Eine aktuelle Version solcher technischer Empfehlungen findet sich im “INSPIRE Technical Architecture” Paper [Drafting Teams “Data Specifications”, “Network Services”, “Metadata”, 2007] auf der INSPIRE Website.

Die geographischen Daten und die dazugehörigen Metadaten der einzelnen Mitgliedsstaaten sollen, wie in Abbildung 5 gezeigt, über Web Services (mehr dazu in Abschnitt 4.7 dieser Arbeit) zu einem Gesamtsystem integriert werden. Auf diese Weise ist es möglich, die Daten dezentral in den einzelnen Ländern zu speichern, und trotzdem über eine einheitliche Schnittstelle abzufragen.

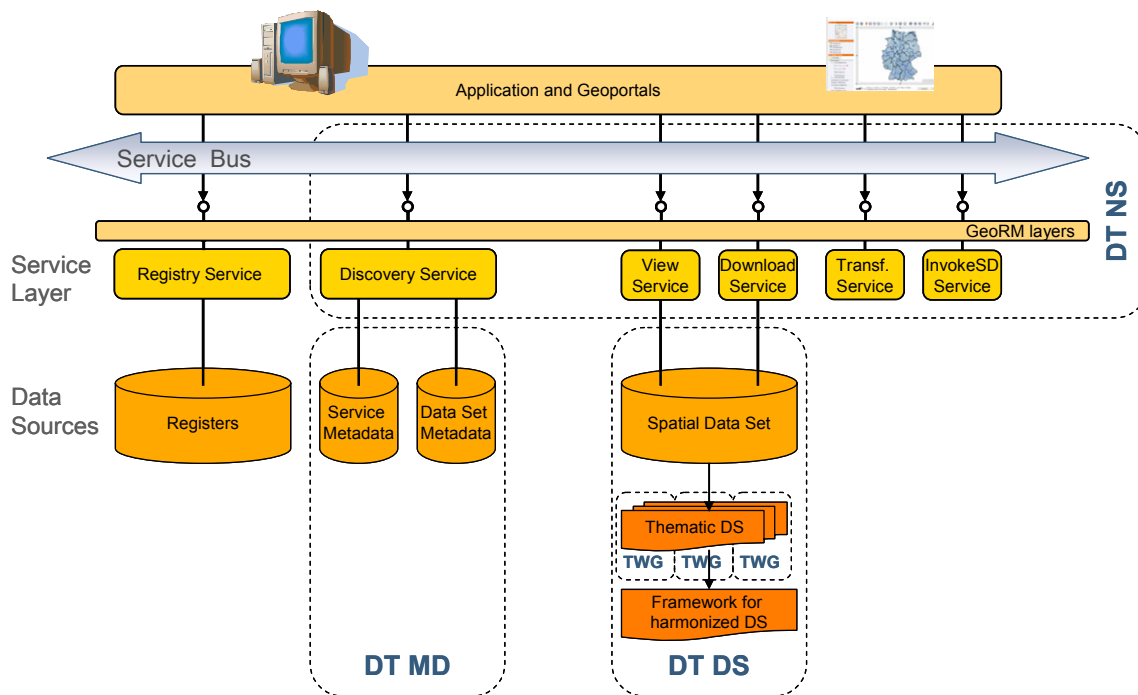


Abbildung 5: Die technische Architektur von INSPIRE: Daten und Metadaten der Mitgliedsstaaten sind über WebServices verbunden. Aus [Drafting Teams “Data Specifications”, “Network Services”, “Metadata”, 2007, Abb. 2.1]

## 2.4. EuroRoadS

So wie auch das vorliegende Projekt steht EuroRoadS vor der Aufgabe, Straßenzüge und andere, mit Straßenverkehr in Zusammenhang stehende Objekte mit Hilfe eines Datenmodells in einem Geoinformationssystem zu modellieren:

Das EuroRoadS Projekt [EuroRoadS:Website, 2007] hat das Erstellen von Vorgaben für den Austausch europäischer Straßendaten zum Ziel. Ziel des Projekts ist es nicht, eine europäische Straßen-Datenbank zu betreiben, sondern eine einheitliche Infrastruktur für den Betrieb nationaler Datenbanken zu spezifizieren. In den letzten Jahren (bis 2006) wurde vom EuroRoadS Projekt eine solche Infrastruktur entworfen, die Erzeugung und Austausch von qualitätsgesicherten geographischen Straßendaten ermöglicht. Teil der EuroRoadS Spezifikationen sind [EuroRoadS:Framework, 2007]:

- Ein Informationsmodell für Straßennetze.
- Ein GML-basiertes Modell für den Austausch von Straßendaten, sowohl für ganze Datensätze wie inkrementelle Updates.
- Spezifikation wichtiger Metadaten über Straßendaten: Was für Eigenschaften von Straßendaten sollen bei der Katalogisierung dieser Daten berücksichtigt werden?



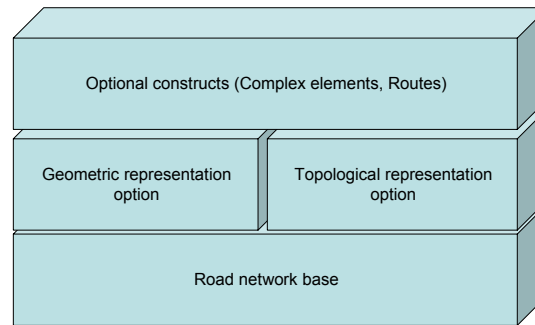


Abbildung 6: Die Modellierung des Verkehrsnetzes in EuroRoadS gliedert sich in drei Ebenen: Grundlegende Konzepte wie Straßen und Straßenknoten in der untersten Ebene. Geometrische oder topologische Repräsentationen dieser Konzepte auf der nächsten Ebene. Und Aggregationen dieser Konzepte zu zusammengesetzten Konstrukten auf der dritten Ebene. Aus [EuroRoadS:InformationModel, Abb. 6-16]

### 2.4.1. Das EuroRoadS Informationsmodell

Das EuroRoadS Informationsmodell [EuroRoadS:InformationModel] ist eine Spezifikation für die Modellierung von Straßendaten. Die Spezifikation bleibt auf einer abstrakten Ebene, d.h. es wird nicht auf konkrete Implementierungen (z.B. in der Form eines Datenformats oder Datenbankschemas) eingegangen. Das Informationsmodell gliedert sich in eine Reihe unterschiedlicher Pakete:

*Basic types Package:* Dieses Paket spezifiziert, was zur Identifikation von Objekten benötigt wird. Alle in EuroRoadS identifizierbare Objekte müssen von der abstrakten Klasse `ER.IdentifiableObject` erben und haben als eindeutige, permanente Identifikationsnummer eine UUID Zeichenkette. (UUIDs sind "Universally Unique Identifiers" entsprechend der ISO Norm für Remote Procedure Calls [ISO/JTC 1, 1996] und der - frei erhältlichen - X.667 Norm [ITU (International Telecommunication Union)] der International Telecommunication Union).

*Network Package:* Dieses Pakete spezifiziert auf drei Ebenen die Modellierung eines Straßennetzes, wie auch in Abbildung 6 gezeigt. Diese Art der Modellierung bringt Flexibilität, um andere Formate wie zum Beispiel GDF ins EuroRoadS Modell abbilden zu können. (GDF, Geographic Data Files, ist ein ISO-Standard [ISO/TC 204, 2004] zur Beschreibung von Straßendaten). Die drei Ebenen sind: (1) Auf unterster Ebene sind die Basis-Objekte modelliert, die ein Verkehrsnetz ausmachen - im Wesentlichen sind das Straßenstücke und Straßenknoten. (2) Auf nächster Ebene sieht EuroRoadS zwei Arten der Repräsentation vor: Geometrisch (z.B durch Spezifikation eines Kurvenverlaufs), oder topologisch (durch Angabe der von einem Straßenstück verbundenen Knoten). Jede topologische Repräsentation muß jedoch ihrerseits wieder eine geometrische Realisation - zum Beispiel einen Kurvenverlauf - enthalten. (3) Auf dritter Ebene können mehrere Elemente des Verkehrsnetzes zu neuen Objekten aggregiert werden, zum Beispiel zu Kreuzungen oder Kreisverkehren oder Verkehrsrouten.



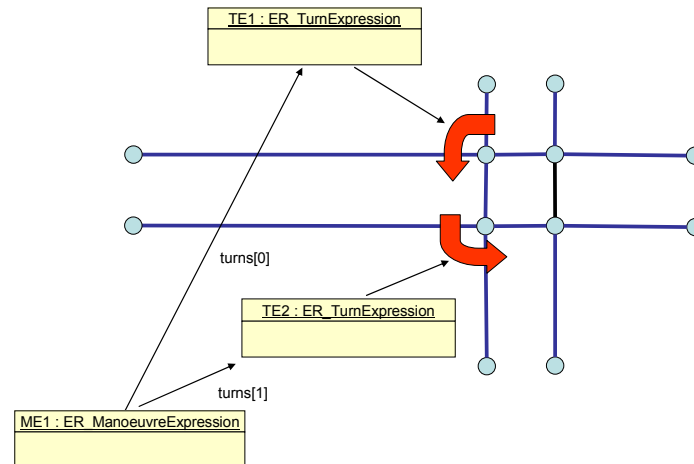


Abbildung 7: Ein Manöver in EuroRoadS - hier als Beispiel ein U-Turn auf einer Kreuzung - ist mit eines der Objekte im Wegenetz, das mit zusätzlicher Information verknüpft werden kann. Aus [EuroRoadS:InformationModel, Abb. 6-29]

*Network Referencing Package:* Dieses Paket definiert Möglichkeiten, zusätzliche Information mit dem im Network Package spezifizierten Wegenetz zu verknüpfen. Das geschieht, indem diese Information Teile des Wegenetzes referenziert. Teile des Wegenetzes sind Punkte auf den Wegstrecken, die Knoten des Wegenetzes, Streckenabschnitte auf dem Wegenetz, oder Manöver, die in dem Wegenetz gefahren werden können. (Abbildung 7 zeigt als ein Beispiel für ein Manöver einen U-Turn, der auf auf einer Kreuzung gefahren wird). Was für Information in der Form von Attributen und Features mit dem Wegenetz verknüpft werden kann, wird im Dokument “Final Specification of Core European Road Data” [EuroRoadS:CoreEuropeanRoadData] festgelegt. Zusätzlich erlaubt EuroRoadS die Definition benutzerdefinierter Daten, die auf gleiche Weise mit dem Wegenetz verknüpft werden können.

*Updates Package:* Dieses Paket definiert die Modellierung von Updates, mit denen vorhandene Straßendaten geändert werden können. Updates in EuroRoadS unterstützen die folgenden Operationen: Insert - Erzeugen eines neuen Objekts. ModifyObject - Bereitstellen einer neuen Version eines Objekts. DeleteObject - Löschen eines Objekts. Split - Aufspalten eines Objekts zu mehreren Objekten. Merge - Zusammenführen mehrerer Objekte. Replace - Ersetzen einer Menge von Objekten durch eine andere. ModifyAttributes - Bearbeiten von Eigenschaften. DeleteAttributes - Löschen von Eigenschaften.

## 2.4.2. Das EuroRoadS Format für den Datenaustausch

EuroRoadS definiert ein Datenmodell [EuroRoadS:ExchangeModel] für den Austausch von Wegenetz-Daten, und ein GML-basiertes Datenformat [EuroRoadS:ExchangeFormat] zur Codierung des Wegenetzes und von inkrementellen Updates. Dieses gemeinsame Format für den Daten-Austausch ist notwendig, weil - so wie auch beim vorliegenden Projekt - mehrere Akteure am Betrieb der Euro-

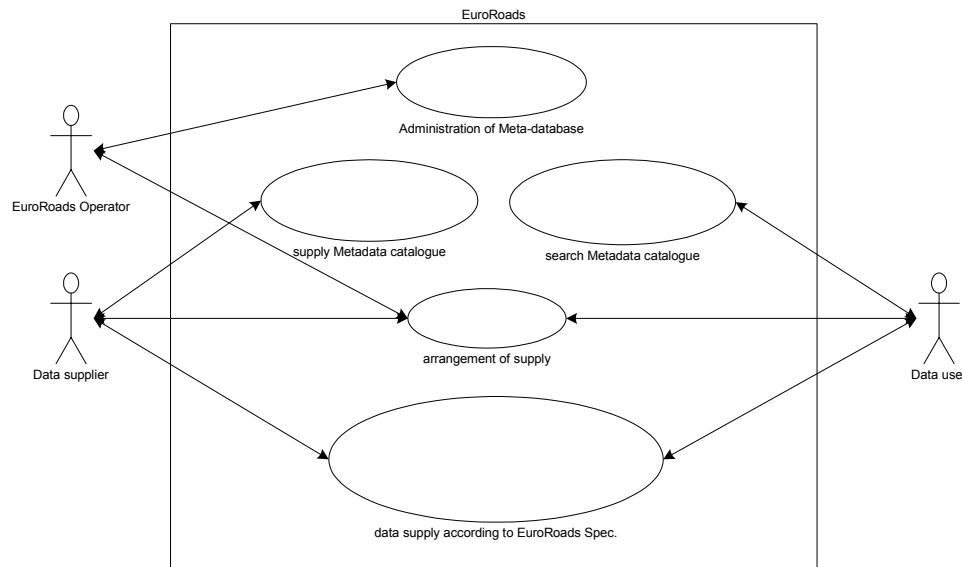


Abbildung 8: Akteure und Use-Cases beim Datenaustausch im EuroRoadS Projekt. Aus [EuroRoadS:ExchangeModel, Abb. 5-1]

RoadS Plattform beteiligt sind. Abbildung 8 zeigt die Rollen und Use-Cases im Hinblick auf den Datenaustausch in EuroRoadS.

Das Datenmodell für den Austausch von Straßendaten ist in vielen Punkten dem im vorigen Abschnitt beschriebenen EuroRoadS Informationsmodell sehr ähnlich, und wird deshalb im Rahmen dieser Arbeit nicht weiter beschrieben. Interessant ist jedoch das für den Austausch verwendete, GML-basierte Datenformat. GML wird auch in den Kapiteln 4 und 5 dieser Arbeit genauer beschrieben. Beispiele von Straßendaten, die der EuroRoadS-Spezifikation entsprechend in GML codiert wurden, finden sich im Anhang der Spezifikation des Austauschformats in [EuroRoadS:ExchangeFormat, Kap. 5.2].

## 2.5. Das vorliegende Projekt “Verkehrstelematik Datenbank”

Diese Arbeit nimmt Bezug auf das *Roncalli*-Projekt [Roncalli:Website, 2008], dessen Ziel der Aufbau einer geographischen Austausch-Plattform und verkehrstelematischen Infrastruktur für das österreichische Straßennetz ist. Mit Hilfe dieser Infrastruktur steht (a) in Echtzeit, (b) auf die Position des Benutzers bezogene und (c) aktuelle verkehrstechnische Information zur Verfügung. In mehreren Services wurde die Nützlichkeit solcher Information demonstriert, beispielsweise um - wie in Abbildung 9 gezeigt - Autofahrer über Verkehrsschilder, Geschwindigkeitsbeschränkungen oder den Zustand der Straße zu informieren. Für eine detailliertere Beschreibung des Projekts sei auf den Roncall\_I2 Endbericht [Klementsitz u. a., 2005] verwiesen.

Das Projekt ist in zwei Teilprojekte “Roncalli” und “Roncall\_I2” gegliedert - beide Teilprojekte sind abgeschlossen (das spätere der beiden Projekte wurde im Februar 2005 fertiggestellt). Das



Abbildung 9: Das Roncalli ISA (Intelligent Speed Adaption) Service unterstützt den Autofahrer durch Anzeige aktueller Geschwindigkeitsbeschränkungen. Foto von [Roncalli:Website, 2008].

Projekt wurde von mehreren österreichischen Projektpartnern durchgeführt und vom österreichischen Bundesministerium für Verkehr, Innovation und Technologie unterstützt. Die im Rahmen des Projekts erstellte Austauschplattform wird von der OeKB Business Services GmbH betrieben.

Abbildung 10 zeigt die Rollen in diesem System: Eine Datenbank mit Information über das österreichische Straßennetz hat die Funktion eines Marktplatzes und bietet Schnittstellen zu zwei Benutzertypen: Einer dieser Benutzertypen sind die *Content Providers* (CP). Sie können über eine Schnittstelle, möglicherweise als Webservice ausgeführt, auf die Datenbank zugreifen. Content Providers pflegen eigene Daten in die Datenbank ein. Der zweite Benutzertyp sind die *Service Providers* (SP). Sie können ebenfalls über eine Schnittstelle auf die Datenbank zugreifen, und auch hier wird die Schnittstelle möglicherweise als Webservice ausgeführt. Die Service Providers nutzen die geographische Information aus der Datenbank, um darauf basierend Dienste anzubieten. Mögliche Dienste wären beispielsweise das Anzeigen aktueller Geschwindigkeitsbeschränkungen, das Hinweisen auf sensible Bereiche wie z.B. vor Schulen oder Altersheimen, oder das Warnen vor Unfallshäufungspunkten in Echtzeit während des Autofahrens.

*Content Service:* Das Content-Service hat die Funktion eines Marktplatzes (in [Klementsitz u. a., 2005] auch als *Clearing Stelle* bezeichnet) für verkehrstechnische Information. Der Marktplatz wird von einer in Bezug auf Content Providers unabhängigen Stelle (der OeKB Business GmbH) betrieben. Dem Content-Service liegt ein gemeinsames, geographisches Referenzsystem zugrunde, auf das sich alle über Roncalli bereitgestellten Daten beziehen müssen.

*Content Providers:* Content Providers können die Roncalli Clearing Stelle als Marktplatz nutzen, um ihre Daten zu verkaufen. Die Vorteile für Content Providers sind (ähnlich wie bei der Konzeption des INTREST Projekts, siehe Abschnitt 2.2): Die Daten selbst gewinnen an Wert, wenn sie

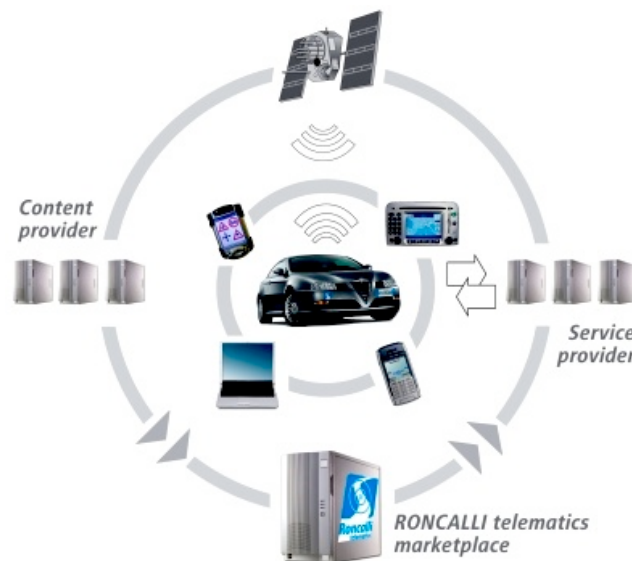


Abbildung 10: Die Roncalli Clearing Stelle: Content Providers stellen Daten am Roncalli Marktplatz zur Verfügung. Diese Daten werden von Service Providers genutzt, um verkehrstelematische Services für mobile Endgeräte anzubieten. Graphik von [Roncalli:Website, 2008].

mit den anderen, über Roncalli zur Verfügung gestellten Daten kombiniert werden. Und der Verwaltungsaufwand beim Verkauf der Daten sinkt, weil der Verkauf und die Verrechnung der verkauften Daten von der Roncalli Clearing Stelle übernommen wird.

*Service Providers:* Service Providers nutzen die über die Roncalli Clearing Stelle verfügbaren Daten, und können mit Hilfe der Daten für den Endkunden interessante Services anbieten. Roncalli-Services sind spezielle Services, die den vom Roncalli Projekt formulierten Konventionen (bezüglich technischer Schnittstellen und der Gestaltung des User-Interfaces) genügen.

Die hier vorliegende Arbeit untersucht aus technischer Sicht Möglichkeiten zur Realisierung des Content Services in Abbildung 11, also einer geographischen Datenbank mit Schnittstellen zur Erstellung, Wartung und Nutzung geographischer Information. Das folgende Kapitel 3 gibt eine Einführung in Geodatenbanksysteme und Standards, die in diesem Bereich relevant sind. In den darauf folgenden Kapiteln wird insbesondere auf XML-basierte Lösungen eingegangen.

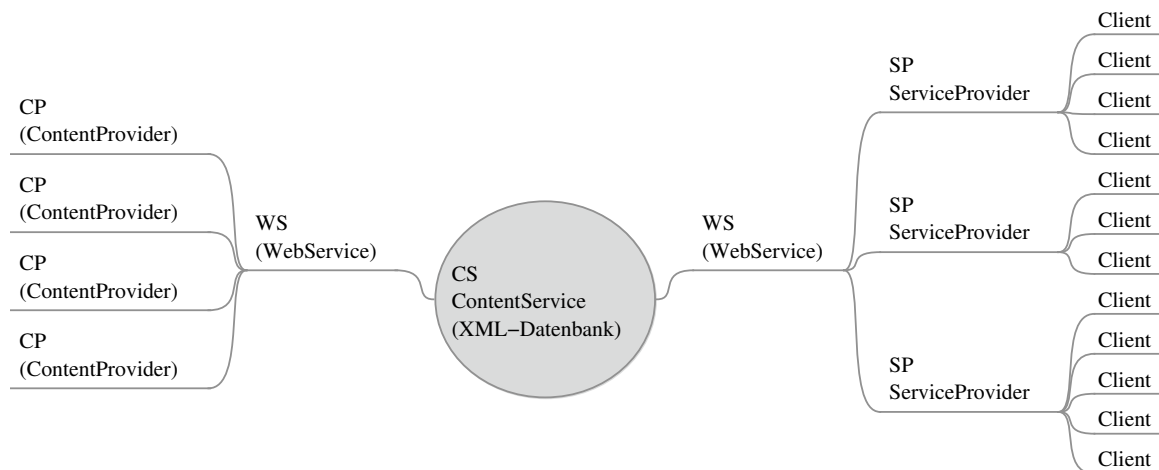


Abbildung 11: In dieser Arbeit wird als technische Umsetzung der Roncalli Clearing Stelle eine XML-basierte Lösung vorgeschlagen: Die von der Clearing Stelle verwalteten Daten sollen in einer XML Datenbank gespeichert werden. Die Datenbank soll als Content Service (CS) über WebService Schnittstellen für die Content Providers (CP) und Service Providers (SP) zugänglich gemacht werden.

### 3. Geodatenbanksysteme

Geodatenbanksysteme (oder auch bezeichnet als räumliche Datenbanksysteme) unterstützen die Speicherung, Abfrage und Verwaltung von raumbezogenen Daten. Häufig werden sie zur Unterstützung von Geoinformationssystemen (GIS) eingesetzt, die eine Benutzerschnittstelle zu den geographischen Daten bereitstellen. Um den Zugriff auf die Daten auch anderen Anwendungen als GIS zu ermöglichen gibt es Bemühungen, Datenbanksysteme zu verwenden, wie sie bereits in anderen Domänen weite Verbreitung gefunden haben [Brinkhoff, 2005, S.5]. Wo notwendig muss ein solches Datenbanksystem für den Umgang mit geographischen Daten angepasst werden. Entsprechend wird von R.Güting in [Güting, 1994] ein geographisches (räumliches) Datenbanksystem als Erweiterung eines herkömmlichen Datenbanksystems charakterisiert:

- a) Ein räumliches Datenbanksystem ist ein vollständiges Datenbanksystem mit zusätzlichen Fähigkeiten für raumbezogene Daten.
- b) Es bietet *räumliche Datentypen* in seinem Datenmodell und *räumliche Funktionen* in seiner Abfragesprache.
- c) Und es unterstützt räumliche Abfragen zumindest durch *räumliche Indizierung* und effiziente Algorithmen für räumliche Joins.

Es existiert eine große Menge an Datenbanksystemen mit räumlichen Erweiterungen. Häufig handelt es sich bei der zugrundeliegenden Technologie um *Objekt-Relationale Datenbanksysteme* (OR-DBMS). Viele Produkte existieren aus der kommerziellen wie aus der Open-Source Welt, zum Beispiel (kommerziell) Oracle Spatial [Oracle:SpatialDB, 2007] und (open-source) PostGIS [PostGIS:Website, 2007]. Die Internetseite FreeGis.org [Wagner, 2008] bietet eine Liste mit weiteren frei erhältlichen Produkten. Das Kapitel 3.1.2 dieser Arbeit beschreibt die Verwendung von OR-DBMS für geographische Anwendungen.

Ein neuerer Ansatz versucht, *XML Datenbanken* (XML-DB), das sind Datenbanksysteme zur Speicherung von XML Daten, zu erweitern und zur Speicherung und Abfrage geographischer Daten zu verwenden. Dieser Ansatz wird in Kapitel 4 dieser Arbeit beschrieben.

Der folgende Abschnitt 3.1 gibt einen Überblick über Normen und Standards im Bereich Geodatenbanken, die für beide Ansätze relevant sind.

#### 3.1. Normen und Standards im Bereich Geodatenbanken

Herkömmliche Datenbanksysteme sind typischerweise auf Geschäfts- und Verwaltungsaufgaben fokussiert und können effizient und sicher eine große Anzahl relativ einfacher Transaktionen durchführen. Räumliche Daten sind jedoch komplexer als traditionelle Geschäftsdaten [Shekhar und Chawla, 2003, S.22] und stellen neue Anforderungen an entsprechende Systeme. Deshalb wurden historisch viele auf Geoinformation spezialisierte Lösungen gefunden. Das ist insbesondere im Bereich der Datenmodel-

lierung und -speicherung der Fall [Brinkhoff, 2005, S.5] - hier existiert eine Vielfalt unterschiedlicher Formate und Lösungen.

Die aus dieser Vielfalt resultierende mangelnde Interoperabilität war ein Problem, das keiner der GIS Hersteller allein bewältigen konnte. Die Standardisierungs- und Normungsbemühungen des Open Geospatial Consortiums (OGC) [OGC, 2007b] und des Technischen Komitees 211 der International Organization for Standardization (ISO/TC211) [ISO:Website, 2007] wollen diesem Problem entgegen und haben größere Interoperabilität geographischer Anwendungen in verteilten Computersystemen zum Ziel. Seit 1998 gibt es ein Abkommen zwischen OGC und ISO: Die abstrakten Spezifikationen der ISO werden vom OGC übernommen, und im Gegenzug werden die Implementierungsspezifikationen des OGC als Normvorschläge bei der ISO eingereicht [Brinkhoff, 2005, Kap. 3.2]. Die Standards des OGC sind - im Gegensatz zu den Normen der ISO - frei erhältlich und können von der OGC Website [OGC, 2007b] heruntergeladen werden.

Eine Einteilung der ISO und OGC Standards im Hinblick auf ihre Relevanz für sieben verschiedene Tätigkeiten im Umgang mit geographischer Information findet sich im Architecture and Standards Position Paper [INSPIRE European Geoportal, 2002, Kap. 3.4] der INSPIRE Initiative der europäischen Union. Im Folgenden soll ein Überblick über für das vorliegende Projekt bedeutsame Standards gegeben werden.

Das *OGC Referenzmodell* ist eine Grundlage für alle anderen Normen des OGC. Es wird in Abschnitt 3.1.1 dieser Arbeit beschrieben. Die *Abstract Specifications* [OGC, e] des OGC arbeiten das Referenzmodell detailliert aus, bleiben aber so wie das Referenzmodell auf einer abstrakten, implementierungsunabhängigen Ebene. Die *Implementation Specifications* des OGC bauen auf dem Referenzmodell und den eben genannten Abstract Specifications auf und definieren Austausch und Speicherung geographischer Daten unter Verwendung *konkreter* Computersprachen oder -Systeme: Unter Verwendung der Structured Query Language (SQL) [OGC, h], unter Verwendung der Common Object Request Broker Architecture (CORBA) [OGC, i], und unter Verwendung der proprietären OLE/DCOM Architektur [OGC, j]. In dieser Arbeit wird in Kapitel 3.1.3 auf SQL eingegangen.

Die eben genannten Spezifikationen für SQL, CORBA und OLE/DCOM können nur mit "einfachen" geographischen Merkmalen umgehen. Vom OGC werden diese als *Simple Features* bezeichnet und wie folgt definiert [OGC, 2004, Kap. 6.1]: Simple Features sind einfach in dem Sinn, dass die geographischen Eigenschaften dieser Merkmale auf einfache Geometrien beschränkt sind, d.h. die Koordinaten sind in nur zwei Dimensionen spezifiziert, und der Verlauf von Kurven wird über lineare Interpolation berechnet.

*GML* ist eine auf XML basierende Spezifikation zur Modellierung geographischer Merkmale. In der aktuellen Version 3 unterstützt GML mehr als nur Simple Features, also auch Koordinatenangaben in 3D und Kurven mit nicht-linearem Verlauf. Während GML zur Modellierung und zum Austauschen von Geodaten vielfach verwendet wird, ist das effiziente Speichern und Abfragen von GML Daten noch eine Herausforderung. Kapitel 4 ist ein Schwerpunkt dieser Arbeit, und geht näher auf GML ein.



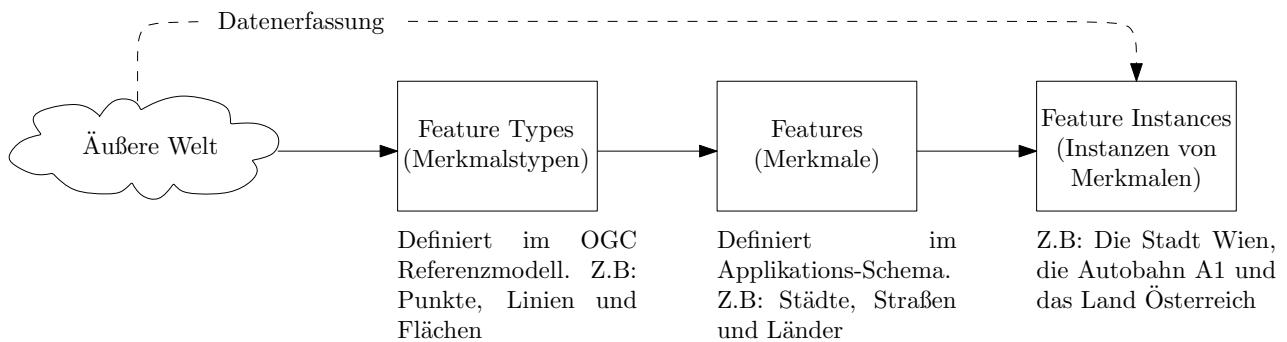


Abbildung 12: Features und Feature Types bei der geographischen Informations-Modellierung. Aus [Inc., 2003, Fig. 4]

Weiters hat das OGC eine Reihe geographischer WebServices (*OGC Web Services*) mit unterschiedlichen Aufgaben definiert: Das WebMapService (WMS) [OGC, d] rendert aus geographischer Information Landkarten als Bitmap-Bild. Im Gegensatz dazu liefern das WebFeatureService (WFS) [OGC, c] und das WebCoverageService (WCS) [OGC, l] die Daten selbst (z.B. im GML Format), ohne sie zu einem Bild zu rendern. Das WebCatalogService (WCS) [OGC, a] gibt Auskunft über andere Services und deren Daten und Fähigkeiten. Das WebProcessingService (WPS) [OGC, m] übernimmt auf Anfrage geographische Berechnungen und liefert das Ergebnis zurück. Abschnitt 4.7 geht näher auf diese WebServices ein.

### 3.1.1. Das OGC Referenzmodell

Das OGC Referenzmodell [Inc., 2003] bietet einen Rahmen für vorhandene OGC Standards und für die laufende Arbeit des OGC. Es geht nicht auf die Bedürfnisse konkreter geographischer Anwendungen oder auf die Umsetzung in einer konkreten Computersprache ein, sondern bleibt auf einer abstrakten Ebene und beschreibt unterschiedliche Aspekte eines offenen, verteilten Systems: Etwa aus der Sichtweise von Firmen die Wertschöpfungskette bei der Arbeit mit geographischer Information, oder aus technischer Sicht die Zusammenarbeit geographischer Services. Für die hier vorliegende Arbeit ist insbesondere der Aspekt der Informationsmodellierung bedeutsam:

Das Referenzmodell nimmt eine Klassifikation von Feature Types (Merkmalstypen) vor. Features sind Abstraktionen von Phänomenen der äußeren Welt. Geographische Features sind mit einer Ortsangabe in einem Koordinatensystem versehen. Wie in Abbildung 12 dargestellt und auch im folgenden Text erläutert, treten geographische Features in mehreren Ebenen auf. Ein konkretes Beispiel über den Zusammenhang dieser drei Ebenen (Feature Types, Features und Feature Instances) wird in Abschnitt 4.2 anhand der Sprache GML gebracht.

*Feature Types (Merkmals-Typen):* Das Referenzmodell kategorisiert und beschreibt Feature Types, und ist damit eine Basis für alle anderen Standards des OGC. Feature Types können diskret sein,



zum Beispiel Punkte, Linienzüge oder Polygone im Raum, oder kontinuierlich zur Modellierung von sich gleichmäßig ändernden Phänomenen wie Luftdruck oder Temperatur.

*Features (Merkmale):* Das Referenzmodell definiert keine Features, sondern stellt dem Entwickler einer Anwendung die Feature Types zur Definition von Features zur Verfügung. Je nach Anwendung können Features zum Beispiel Städte, Straßen, Länder oder Flughäfen sein. Zum Beispiel könnte in einer Anwendung das Feature “Stadt” mit einem Punkt (Stadtzentrum) und einem Polygon (Stadtgrenze) modelliert werden.

*Feature Instances (Instanzen von Merkmalen):* Objekte der äußeren Welt werden in einem geographischen Informationssystem (GIS) auf Instanzen von Features abgebildet. Zum Beispiel könnte die Stadt Wien als eine Instanz des Features “Stadt” gespeichert werden, mit passenden Koordinaten für das Stadtzentrum und die Stadtgrenze.

Im soeben genannten Beispiel mit der Stadt Wien wurden Punkte und Polygone als Merkmalstypen verwendet. Das entspricht einem von zwei grundlegend verschiedenen Modellen um geographische Information zu repräsentieren, nämlich dem Objektmodell. Neben dem Objektmodell sieht das OGC Referenzmodell auch noch Feature Types vor, die dem Feldmodell entsprechen.

*Objektmodell:* Das Objektmodell bildet den Raum durch einzelne, diskrete, identifizierbare und wohl-unterschiedene Objekte (Feature Instances) ab. Es eignet sich zur Repräsentation geographischer Phänomene mit klar definierten Grenzen. Jedes Objekt hat eine Menge von Eigenschaften, die räumlicher oder nicht räumlicher Natur sein können. Beispielsweise könnte ein Objekt zur Repräsentation einer Waldfläche als räumliches Attribut ein Polygon besitzen, der Name des Waldes hingegen wäre ein textuelles (nicht-räumliches) Attribut. Weil die räumlichen Attribute mit geometrischen Primitiven wie Punkten, Linien und Flächen modelliert sind, spricht man bei dieser Art von Daten auch von *Vektordaten*.

*Feldmodell:* Im Unterschied dazu geht das Feldmodell von einer kontinuierlichen Funktion  $f : \text{Punkte in Raum und Zeit} \mapsto \text{Wertemenge}$  aus, die Punkten in Raum und Zeit Werte aus einer Wertemenge zuweist [Shekhar und Chawla, 2003, Kap.2]. Das Feldmodell eignet sich zur Repräsentation geographischer Phänomene, die sich kontinuierlich im Raum verändern, die aber selbst kein fixe Ausdehnung haben. Beispiele für Daten dieser Art sind Temperaturkarten, digitale Höhenmodelle, Satellitenbilder oder eingescannte Karten. Wenn der Raum zu einem Raster diskretisiert ist spricht man von Rasterdaten. Weil der Raster nicht die einzige derartige Datenstruktur ist, wird im OGC Referenzmodell der allgemeinere Begriff *Coverage* geprägt.

### 3.1.2. Relationale und Objekt-Relationale Geodatenbanken

Relationale und objekt-relationale Datenbanken gehören zu den meist verbreiteten Möglichkeiten, geographische Daten zu speichern und abzufragen. Mit ein Grund dafür sind die vorteilhaften Eigenschaf-

ten dieser Systeme wie Transaktionen, Sicherheit, Crash Recovery und Optimierte Anfrageauswertung [Zhu u. a., 2006, Kap. 1][Li u. a., 2004]. Zur Verwendung für geographische Daten haben viele relationale Datenbankprodukte (kommerziell wie open-source) Erweiterungen für räumliche Indizierung der Daten und räumliche Abfragemöglichkeiten.

Zu den oft genannten und von vielen Seiten unterstützten Produkten gehören die kommerzielle *Oracle Spatial* Datenbank [Oracle:SpatialDB, 2007] und das Open-Source Projekt *PostGIS* [PostGIS:Website, 2007], eine Erweiterung der PostgreSQL Datenbank um räumliche Datentypen und Funktionen. Beide Produkte werden auch vom Persistenz-Framework *Hibernate Spatial* [Hibernate:Spatial:Website, 2007] unterstützt. Hibernate ist ein weit verbreitetes Framework, um Objekte der Java oder .Net Programmiersprachen in relationalen Datenbanken zu Speichern und wieder auszulesen. Hibernate Spatial ist eine Erweiterung von Hibernate für den Umgang mit geographischen Daten. Ein anderes Framework, *Geo Tools* [GeoTools:Website, 2007], eine in Java geschriebene Library für geographische Daten, unterstützt neben diesen beiden relationalen Datenbanken auch noch MySQL [MySQL:Website, 2008] und die räumlichen Erweiterungen [MySQL:SpatialExtensions, 2007] der MySQL Datenbank.

Für die Implementierung einer relationalen Datenbank mit geographischen Datentypen und Funktionen - oder auch für den Zugriff auf entsprechende Datenbanken - ist der im folgenden Kapitel beschriebene OGC Standard Simple Features for SQL wichtig.

### 3.1.3. Die OGC Implementierungs-Spezifikation für SQL

Im Gegensatz zum abstrakten Referenzmodell (siehe Kap. 3.1.1) beschreiben die Implementierungsspezifikationen des OGC konkrete Computersprachen oder -Systeme. So definiert das OGC Dokument *Simple Feature Access with SQL Option* [OGC, h] den Zugriff auf geographische Information bei Verwendung der Structured Query Language (SQL) für relationale Datenbanken. Wie schon der Name sagt, wird in dem Standard nur die Modellierung ‘einfacher’ geographischer Merkmale spezifiziert (siehe dazu Abschnitt 3.1 dieser Arbeit über das OGC Referenzmodell). Technisch sieht der Standard zwei Möglichkeiten zur Repräsentation räumlicher Eigenschaften vor: Entweder unter Verwendung benutzerdefinierter Datentypen in SQL oder ohne benutzerdefinierte Datentypen. Im zweiten Fall werden geographische Eigenschaften als Verweise auf Tupel in einer Geographie-Tabelle gespeichert.

Etliche relationale Datenbank-Produkte haben Erweiterungen zur Speicherung und Abfrage räumlicher Daten, und in den meisten Fällen sind diese Erweiterungen konform zur OGC Spezifikation. Das Speichern von Geodaten in solchen Datenbanken ist weit verbreitet und effizient.

Das Austauschen relationaler Daten in verteilten, heterogenen Systemen ist jedoch schwierig. Deshalb werden die Daten häufig zum Austausch vom relationalen Schema in ein anderes Datenmodell konvertiert, das für den Datenaustausch besser geeignet sind. Ein solches Datenmodell, das auch im Bereich geographischer Daten an Bedeutung gewonnen hat, ist XML, und wird in Kapitel 4 beschrie-

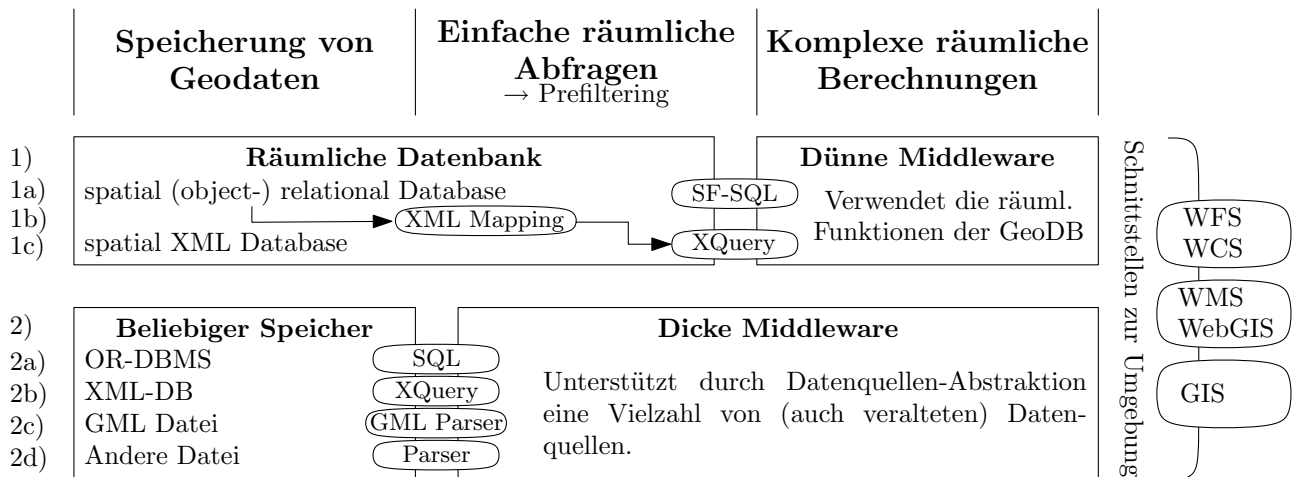


Abbildung 13: Mögliche Architekturen für die Verkehrstelematik Datenbank

ben. Zuvor sollen jedoch noch verschiedene Software-Architekturen für geographische Datenbanken beleuchtet und in Hinblick auf das vorliegende Projekt verglichen werden.

### 3.2. Architekturen von Geodatenbanken

Ein Beitrag dieser Arbeit ist die folgende Analyse und Einteilung möglicher Software-Architekturen für geographische Datenbanksysteme. Wie in Abbildung 13 ersichtlich, unterstützen Geoinformationssysteme das Speichern und Abfragen geographischer Daten, und erlauben die Durchführung mehr oder weniger komplexer Berechnungen. Dabei werden zum Speichern der Daten meist vorhandene Datenbanksysteme weiterverwendet, die um zusätzliche Funktionalität erweitert werden können. Komplexere geographische Berechnungen werden zumeist in einer eigenen Softwareschicht implementiert, die als (räumliche) *Middleware* bezeichnet wird. Diese Middleware besitzt Schnittstellen nach innen (zur Datenbank hin) und nach außen (zu anderen Systemen hin), und ist damit ein Bindeglied zwischen der Datenbank und der äußeren Umgebung. Diese Bindeglied-Funktion der Middleware und ihre Schnittstellen nach innen und außen sollen im folgenden Abschnitt beschrieben werden.

#### 3.2.1. Middleware als Bindeglied zwischen Geodatenbank und Umgebung

Wie auch in Abbildung 13 ersichtlich, hat eine räumliche Middleware Schnittstellen nach innen zur Datenbank hin, und nach außen zu anderen Systemen.

*Schnittstellen nach außen:* Eine räumliche Middleware hat Schnittstellen einerseits zur eigentlichen Datenbank und andererseits nach außen zu der jeweiligen Anwendung. Als Schnittstellen nach außen kommen unter anderen die Webservice Spezifikationen des OGC (siehe Kapitel 4.7) in Frage. Das WebFeatureService (WFS) und das WebCoverageService (WCS) erlauben dabei den Zugriff auf die Geodaten selbst - das WebMapService (WMS) hingegen macht aus den Geodaten

Bilder (Landkarten), und stellt diese statt der ursprünglichen Daten zur Verfügung. Auch WebGIS (webbasierte Geoinformationssysteme) können als Schnittstelle nach außen gesehen werden - in dem Fall als Schnittstelle nicht zu einem Softwaresystem, sondern direkt zu einem menschlichen Benutzer. WebGIS sind über HTTP erreichbare, interaktive Oberflächen zum Betrachten von Geodaten. Beliebige weitere (auch proprietäre und/oder produktspezifische) Schnittstellen zu anderen Geoinformationssystemen (GIS) sind möglich.

*Schnittstellen nach innen:* Die Schnittstelle einer Middleware nach innen, zur Datenbank hin, hängt in erster Linie vom Typ der verwendeten Datenbank ab. Zur Abfrage relationaler und objekt-relationaler Datenbanken hat sich SQL etabliert (siehe Abschnitt 3.1.3), und zur Abfrage von XML Datenbanken wird häufig XQuery verwendet (siehe Abschnitt 4.3).

Wenn die Schnittstelle nach innen, also zur Datenbank hin, räumliche Abfragefunktionen unterstützt, muß die Middleware diese Funktionen nicht selbst implementieren. Im Rahmen dieser Arbeit sollen solche Middlewares als Thin Middlewares bezeichnet werden. Der folgende Abschnitt beschreibt den Unterschied zwischen Thin Middlewares und Thick Middlewares, und geht dabei auch auf Möglichkeiten bei der Wahl einer geeigneten Datenbank ein.

### 3.2.2. Thin versus Thick Middlewares

Ein wesentlicher Unterschied in der Konzeption eines geographischen Datenbanksystems liegt darin, ob die Funktionalität einfacher, räumlicher Abfragen in der Datenbank oder in der Middleware implementiert wird. Einfache und häufig vorkommende Abfragen sind von der Art: "Bitte alle Objekte, die innerhalb eines bestimmten rechteckigen Suchfensters liegen". Diese Art der Abfrage entsteht beispielsweise, wenn ein bestimmter Kartenausschnitt gezeichnet werden soll. Oft bleibt nach Auswertung solcher Abfragen nur ein Bruchteil der in der Datenbank gespeicherten Geo-Objekte für weitere Berechnungen übrig, weshalb die Auswertung auch als *Pre-Filtering* vor der Auswertung komplexerer räumlicher Berechnungen gesehen werden kann.

Tabelle 1 und der folgende Text sollen den Unterschied zwischen Thin und Thick Middlewares weiter verdeutlichen. Dabei entspricht die Gliederung der folgenden Abschnitte (1, 1a, 1b...) den in Abbildung 13 aufgezeigten möglichen Varianten.

#### 1) *Räumliche Datenbank und Thin Middleware*

Bei Architekturen mit einer räumlichen Datenbank und einer Thin Middleware unterstützt die Datenbank zumindest einfache räumliche Abfragen, indem sie räumliche Datentypen und Abfragefunktionen zur Verfügung stellt und räumliche Indizes verwaltet, die das Auswerten räumlicher Abfragen beschleunigen. Eine thin Middleware verwendet diese Funktionalität auch und muß sie in der Folge nicht selbst implementieren.

Eine thin Middleware ist sinnvoll, wenn ihre Schnittstelle nach außen von der restlichen System-Umgebung selbst als Datenquelle wahrgenommen wird - dann nämlich ist eine schlanke und effiziente

Thin Middlewares	Thick Middlewares
+ Filtern anhand von räumlichen Kriterien schon in der Datenbank	- Hoher Datenverkehr zwischen Datenquelle und Middleware
- Inkompatibel mit veralteten oder sonst unzulänglichen Datenquellen	+ Kompatibel mit einer Vielzahl an - auch veralteten - Datenquellen
Beispiel-Anwendung: Ein Web Feature Service (WFS), dessen Middleware als Schnittstelle zu den Geodaten fungiert.	Beispiel-Anwendung: Ein Desktop GIS, dessen Middleware mit möglichst vielen Datenquellen kompatibel sein soll.

Tabelle 1: Vergleich: Thin versus thick Middlewares

Architektur wichtiger als die Frage, ob die Middleware eine Vielzahl verschiedener Datenbanken oder Dateiformate unterstützt.

Für die Verwendung mit einer räumlichen Thin Middleware kommen nur Datenbanken in Frage, die um räumliche Datentypen, Indizes und Abfragefunktionen erweitert sind. Insbesondere sind das die folgenden, auch in Abbildung 13 gezeigten Datenbanken:

**1a) Räumliche Relationale Datenbank:** Räumliche Erweiterungen für relationale Datenbanksysteme (RDBS) sind üblicherweise konform zum Simple Features für SQL Standard [OGC, h] des OGC (siehe Abschnitt 3.1.3). Eine Thin Middleware kann unter Nutzung dieses Standards auf die relationale Geodatenbank zugreifen, und deren räumliche Abfragemöglichkeiten nutzen. Dieser Ansatz (d.h. die Verwendung räumlicher RDBS) wird in Hinblick auf die Speicherung von Daten im GML Format in Abschnitt 5.2 dieser Arbeit genauer beschrieben.

**1b) Räumliche XML-enabled Datenbank:** Weil das XML-basierte Format GML gut für den Austausch von Geodaten zwischen heterogenen Systemen geeignet ist, liegen die zu speichernden Daten häufig im GML Format vor. Zur Speicherung dieser XML-basierten Daten können RDBS verwendet werden, allerdings müssen sie um zwei Aspekte erweitert werden: Erstens muss das RDBS entsprechend dem Simple Features für SQL Standard um räumliche Funktionen erweitert werden. Zweitens muss das RDBS für die Verwendung mit XML Daten erweitert werden. Dabei bildet ein XML-Mapping das Schema der XML Daten ins relationale Modell ab. Mit Hilfe dieses Mappings können Abfragen über den XML Daten zu SQL Abfragen übersetzt werden. Auf diese Weise lässt sich eine XML-enabled räumliche Datenbank über XML Abfragesprachen wie z.B. XQuery abfragen, und so auch zur Speicherung von geographischen Daten im GML Format nutzen. Dieser Ansatz wird in Abschnitt 5.4 dieser Arbeit genauer beschrieben.

**1c) Räumliche XML Datenbank:** Eine Alternative zur Verwendung von XML-enabled Datenbanken sind Datenbanken, die von Haus aus auf das Speichern von XML Daten spezialisiert sind - man spricht in dem Zusammenhang von 'nativen' XML Datenbanken. Solche Datenbanken müssen um räumliche Funktionen erweitert werden, damit sie zum Speichern geographischer Daten verwendet werden können. Sie können dann über XML Abfragesprachen (z.B: XQuery) abgefragt werden. Dieser Ansatz wird in Abschnitt 5.5 genauer beschrieben.

## 2) *Beliebiger Datenspeicher und Thick Middleware*

Bei Architekturen mit einer thick Middleware unterstützt der Datenspeicher räumliche Abfragen auf keine besondere Weise. Alle räumlichen Abfragen und Berechnungen sind in der Middleware implementiert - deshalb wird in dieser Arbeit eine solche Middleware als “thick” bezeichnet.

Eine thick Middleware stellt kaum funktionale Anforderungen an den von ihr verwendeten Datenspeicher. Produkte mit dieser Architektur können daher häufig mit vielen verschiedenen Datenquellen umgehen, und unterstützen auch veraltete oder wenig gebräuchliche Dateiformate. Das ist besonders dann wichtig, wenn die Middleware ein System unterstützt, wo das Laden, Bearbeiten und Integrieren von Geodaten verschiedener Art, Herkunft und Qualität eine essenzielle Rolle spielt. Das ist zum Beispiel bei Desktop GIS Anwendungen der Fall.

Als Datenbanken für die Verwendung mit einer Thick Middleware kommen alle Möglichkeiten in Frage, die schon für die Verwendung mit Thin Middlewares genannt wurden. Zusätzlich kann jedoch eine Thick Middleware auch einfachere oder veraltete Speichermöglichkeiten unterstützen:

**2a) (Objekt-) relationale Datenbanken:** Thick Middlewares können mit relationalen oder objekt-relationalen Datenbanken betrieben werden. Diese Datenbanken können, müssen aber nicht, mit räumlichen Erweiterungen ausgestattet sein.

**2b) XML Datenbanken:** Für die Speicherung von in XML codierten Daten (z.B. im GML Format) können XML Datenbanken verwendet werden. Nachdem alle räumlichen Abfragen und Berechnungen in der Thick Middleware ausgeführt werden, muß eine solche XML Datenbank auch keine spezielle Unterstützung für räumliche Daten haben.

**2c) GML Dateien:** Geodaten im GML Format können - wie Daten anderer Formate auch - in Dateien gespeichert werden. Dass Dateien keine Unterstützung für die Abfrage nach räumlichen oder sonstigen Kriterien bieten, macht bei der Verwendung einer Thick Middleware nichts aus. Dieser Ansatz (die Verwendung von Dateien zur Ablage von GML Daten) wird in Abschnitt 5.3 dieser Arbeit beschrieben.

**2d) Andere Dateien:** Viele Thick Middlewares unterstützen eine Reihe verschiedener Dateiformate als Datenquellen. Zu den unterstützten Formaten gehören häufig auch wenig gebräuchliche oder veraltete Formate.

### 3.2.3. Einige Produkte räumlicher Middlewares

Tabelle 2 zeigt eine Liste von großteils frei erhältlichen räumlichen Middlewares. Viele Middlewares lassen sich nicht genau einer der beiden Kategorien (thin oder thick Middleware) zuordnen, weil sie einerseits räumliche Datenbanken unterstützen und deren räumliche Abfrage-Funktionalität nützen, aber andererseits auch andere Datenquellen unterstützen, die diese Funktionalität nicht haben. Die Unterscheidung thin/thick Middleware ist trotzdem sinnvoll, weil auch diese Middlewares in Bezug auf eine bestimmte Datenquelle genau entweder als thin oder thick Middleware agieren, d.h. sie nutzen die räumlichen Funktionen der Datenquelle oder nutzen sie nicht.

Projektname	Beschreibung
OpenJump [OpenJump:Website, 2007], früher JUMP Project [JumpProject, 2007]	Frei, Open-source, Java. Wird im OpenJUMP Desktop GIS verwendet. Wurde für das Deegree Projekt [Deegree:Website, 2007] zu DeeJUMP erweitert.
Deegree [Deegree:Website, 2007]	Frei, Open-source, Java. Erweitert das OpenJUMP Projekt um OGC WebServices, webbased Clients, Sicherheitsmechanismen und ein eigenes Desktop GIS.
GRASS [GRASS:Website, 2007]	Frei, Open-source, ANSI-C. Binaries für Windows, Linux und Mac OSX. Wird für das GRASS Desktop GIS verwendet. Das QuantumGIS [QuantumGIS:Website, 2007] Desktop GIS kann auch als Frontend für die GRASS Infrastruktur verwendet werden.
GeoTools [GeoTools:Website, 2007]	Frei, Open-source, Java. Eine Code-Bibliothek für den Umgang mit geographischen Daten. Wird u.a. für das uDig Desktop GIS [uDig, 2007] und für das GeoServer WMS [GeoServer, 2007] verwendet.
ArcSDE [Esri:ArcSDE, 2007]	Kommerziell, closed-source. Eine Middleware für den Zugriff auf Geodaten in verschiedenen relationalen Datenbanken. Wird sowohl für Esri's Desktop wie Server-Produkte eingesetzt.

Tabelle 2: Einige räumliche Middlewares

### 3.2.4. Bedeutung für das vorliegende Projekt

Wie auch in Abschnitt 2.5 beschrieben, soll im vorliegenden Projekt ein *Content Service* als Austauschplattform für Geodaten dienen. Das Content Service soll einerseits von Content Providers verwendet werden, die ihre Geodaten in die Austauschplattform einspielen. Andererseits soll es von Service Providers verwendet werden, die mit den Daten Services beliebiger Art zur Verfügung stellen.

Das Content Service hat für seine Umgebung die Funktion eines Datenspeichers: Effizienter lesender und schreibender Zugriff hat Priorität gegenüber der automatisierten Integration heterogener Datenquellen zu einem Gesamtsystem. Deshalb ist die Architektur einer thin Middleware in Kombination mit einer Datenbank mit räumlichen Erweiterungen für das vorliegende Projekt besser geeignet.

Das folgende Kapitel 4 behandelt das XML-basierte Format GML, das beim Austausch von Geodaten zunehmend Bedeutung gewonnen hat, und beschreibt verschiedene Aspekte im Umgang mit GML Daten. Im darauf folgenden Kapitel 5 wird auf Speichermöglichkeiten für GML Daten eingegangen.



## 4. XML und GML

Das Aufkommen des Internets brachte das Bedürfnis nach Sprachen, die den Informations-Austausch zwischen verschiedenartigen Systemen ermöglichen [Sripada u. a., 2004, Kap.1]. Eine solche Sprache ist die eXtensible Markup Language (XML). Eine auf XML basierende Sprache ist die *Geographic Markup Language* (GML) des Open Geospatial Consortiums (OGC, [OGC, 2004]), die auch als ISO Standard adoptiert werden soll [ISO/TC 211, 2007]. GML findet weite Unterstützung in der Industrie und in der Open Source Welt [OGC, 2007a]. Mit als Grund dafür nennt [Lu u. a., 2007, Kap.2], dass GML ein gutes Trade-Off zwischen Effizienz und Interoperabilität darstellt.

GML dient zur Modellierung, Speicherung und zum Austausch von geographischer Information. Die aktuelle Version 3 des Standards umfasst circa 600 Seiten, es gibt jedoch um den Einstieg zu erleichtern das wesentlich kürzere GML Simple Feature Profile [Inc., 2005], das sich auf einfache geometrische Objekte beschränkt. Außerdem ist es möglich und vorgesehen, dass Benutzer durch Weglassen von Teilen des GML Standards ein eigenes Profil erzeugen.

Der GML Standard verwendet XML Schema Definitions (XSD) zur Definition von geometrischen Merkmalstypen wie z.B. Punkte, Linien, und Polygone. Darüber hinaus umfasst der Standard Aspekte der Topologie, Rasterdaten (Coverage), zeitliche und räumliche Referenzsysteme, XLink um auf Objekte in anderen GML Dokumenten zu verweisen, und die Möglichkeit, durch Style Descriptors die Darstellung geometrischer Features festzulegen. Abbildung 14 zeigt eine Hierarchie von Merkmalstypen in GML und macht die Vielzahl an berücksichtigten Aspekten ersichtlich (Coverages entsprechend dem Feldmodell, Features entsprechend dem Objektmodell, Geometrie und Topologie, Koordinatensysteme, Zeit, visuelle Darstellung, Datentypen und Maßeinheiten...).

Im Folgenden seien die Vor- und Nachteile bei der Verwendung von GML kurz umrissen.

### 4.1. Vor- und Nachteile von GML

Vorteile bei der Verwendung von GML sind:

*Standardisierung, Normung, Verbreitung:* GML ist ein OGC Standard [OGC, 2004] und soll zur ISO Norm gemacht werden [ISO/TC 211, 2007]. GML wird von vielen kommerziellen wie freien Softwareprodukten unterstützt. Auf der Website des OGC findet sich eine lange Liste registrierter Produkte [OGC, 2007a], die dem GML Standard sowie anderen OGC Standards genügen.

*XML Kompatibilität:* GML ist eine XML basierte Sprache. Daher können XML-kompatible Anwendungen prinzipiell auch mit GML Daten umgehen. Oft jedoch müssen diese Anwendungen für den Umgang mit räumlichen Daten angepasst werden, z.B. durch Erweiterung von XML Abfragesprachen um räumliche Funktionen [Lu u. a., 2007, Kap. 3.2][Boucelma und Colonna, 2004], oder durch Hinzufügen einer Filterung nach räumlichen Kriterien noch vor dem eigentlichen Parsen [Huang u. a., 2006].



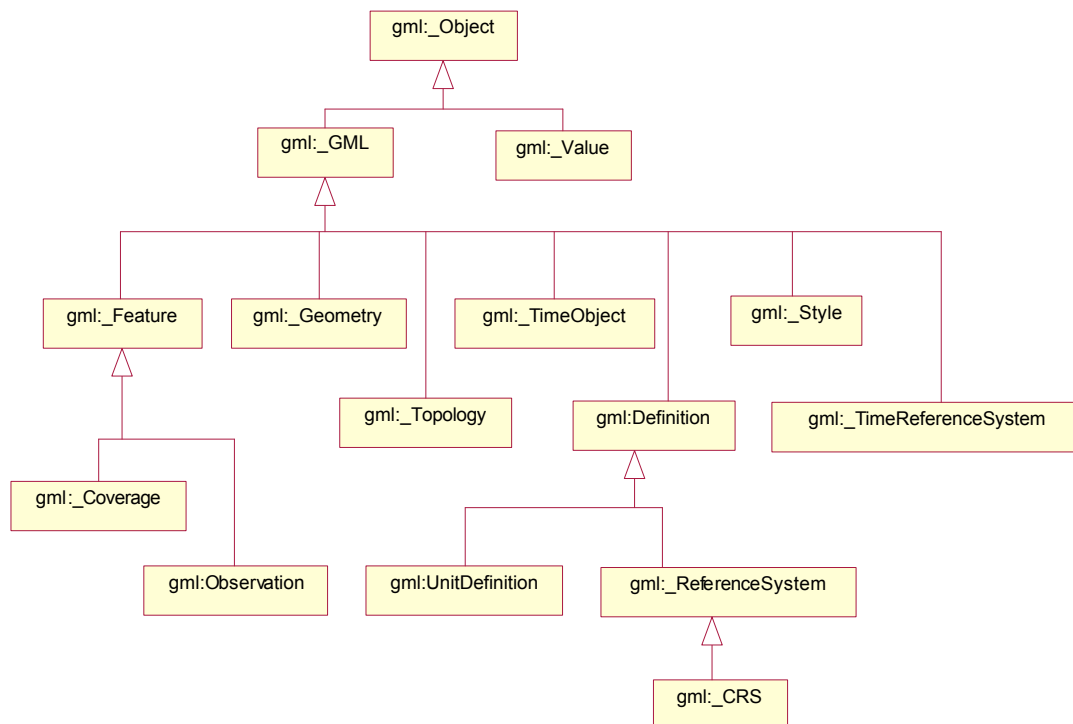


Abbildung 14: Die Hierarchie von Objekten in GML zeigt die Vielfalt von Aspekten, die in der Modellierungssprache GML berücksichtigt werden. Aus [OGC, 2004, Fig. 2]

*Transformabilität:* XML Daten im allgemeinen, und im Speziellen auch GML Daten, lassen sich zu anderen Datenformaten hin transformieren. Diese anderen Formate können, aber müssen nicht, wieder auf XML aufbauen. Zum Beispiel können GML Daten zu Bildern im SVG Format transformiert werden (siehe Kapitel 4.5).

*Interoperabilität:* Viele geographische Datenformate sind proprietär oder für eine bestimmte Anwendung entworfen. Das erschwert den Austausch geographischer Information. Im Gegensatz dazu ist der GML Standard frei erhältlich und für eine Vielzahl an Anwendungen geeignet. Insbesondere wird GML in geographischen WebServices verwendet, die als Schnittstelle zwischen heterogenen Systemen ein hohes Maß an Interoperabilität bieten (siehe Abschnitt 4.7).

*Integration heterogener Daten:* GML dient in etlichen Arbeiten zur Integration von Geodaten aus verschiedenartigen Datenquellen. Dazu werden unter anderem Ontologien und Semantic-Web Techniken verwendet (mehr dazu in Abschnitt 4.6).

*Flexibilität:* Wie in Abschnitt 4.2 veranschaulicht, ermöglicht GML seinen Benutzern die Definition eines passenden Applikations-spezifischen Datenschemas. Für verschiedene Anwendungsgebiete (u.a. Bauindustrie, Luftfahrt, Seefahrt, militärische Anwendungen, Klimaforschung...) wurden

solche Applikationsschemas definiert. Eine Liste solcher Domänen-spezifischer, auf GML basierender Formate findet sich in [OGC, g; wikipedia.org].

*Nicht nur räumliche Daten:* GML eignet sich zur Modellierung von räumlicher sowie auch von nicht-räumlicher Information. Beispielsweise könnten die folgenden (nicht nur geographischen) Zusammenhänge in GML modelliert werden [Inc., 2005, Annex C]: “Zu jedem Reporter wird Kontaktinformation und ein Foto gespeichert. Jedem Reporter können Nachrichten (bestehend aus Datum, Titel und Text) zugeordnet werden, die sich auf einen bestimmten geographischen Ort beziehen”. Das heißt, GML stellt dem Entwickler eines Applikationsschemas nicht nur vorgerfertigte räumliche und zeitliche Eigenschaften zur Verfügung, sondern ermöglicht die Definition beliebiger anderer Eigenschaften im Applikationsschema.

Nachteile oder Schwierigkeiten im Umgang mit GML sind:

*Effizienz:* Der GML Standard geht nicht auf Aspekte der Leistung und Effizienz ein. Die Aufgabe, GML Daten effizient zu speichern, abzufragen und zu übertragen, bzw. allgemein effizient mit GML Daten umzugehen, bleibt beim Benutzer des GML Standards. Zur Zeit (2008) ist GML noch nicht das effizienteste Format für den Umgang mit geographischer Information [Lu u. a., 2007, Kap. 2]

*Komplexität des GML Standards:* . Auf Grund seiner Komplexität unterstützen viele Produkte nur einen Bruchteil der Möglichkeiten, die der GML Standard in der Version 3 bietet. Viele Produkte unterstützen nur das wesentlich einfachere GML2 oder das Simple Features Profile von GML3.

*Speicherung von GML:* Für das Speichern und Abfragen von Daten im GML Format gibt es mehrere Ansätze, jeweils mit Vor- und Nachteilen. Viele dieser Ansätze sind Stand der Forschung und es fehlen ausgereifte Produkte und vergleichende Benchmarks. Ein Schwerpunkt dieser Arbeit ist das Aufzeigen und vergleichen dieser Ansätze (siehe Kapitel 5).

## 4.2. Informationsmodellierung in GML anhand eines Beispiels

Das folgende Beispiel macht die Informationsmodellierung geographischer Information in GML deutlich: Mit Hilfe der im GML Standard bereitgestellten Feature Types können anwendungsspezifische Features in einem Applikationsschema definiert werden. GML Daten, die konform zu dem Applikationsschema sind, bestehen aus Instanzen dieser im Applikationsschema definierten Features. (Die Unterscheidung von Feature Types, Features und Feature Instances wurde im Abschnitt 3.1.1 über das OGC Referenzmodell beschrieben, und ist in Abbildung 12 übersichtlich dargestellt.)

### 4.2.1. Feature Types im GML Standard

Der GML Standard definiert die Repräsentation geometrischer Eigenschaften (Feature Types), aber nicht deren applikations-spezifische Verwendung. Der Entwickler einer Applikation verwendet die be-

reitgestellten Feature Types und schreibt so ein für seine Anwendung geeignetes Applikationsschema. Zu beiden Zwecken (zur Definition der Feature Types und zur Definition des Applikationsschemas) wird die Sprache XML Schema [W3C, e] verwendet.

Listing 1 zeigt einen exemplarischen Ausschnitt aus der GML Simple Feature Specification [Inc., 2005], wo der Feature Type eines Punktes definiert wird: Der Inhalt des Elements Point wird durch seinen Typ festgelegt, nämlich gml:PointType (Zeile 1). Die Definition von gml:PointType (Zeile 4) sagt, dass jeder Punkt ein pos-Element beinhalten muss (Zeile 8). Das pos-Element wiederum hat den Typ gml:DirectPositionType, dessen Inhalt (Zeile 17) eine gml:doubleList ist, also eine Liste von Double-Werten als Koordinaten des Punktes.

---

```
1 <element name="Point" type="gml:PointType"
2 substitutionGroup="gml:_GeometricPrimitive" />
3
4 <complexType name="PointType">
5   <complexContent>
6     <extension base="gml:AbstractGeometricPrimitive">
7       <sequence>
8         <element ref="gml:pos" />
9       </sequence>
10    </extension>
11  </complexContent>
12 </complexType>
13
14 <element name="pos" type="gml:DirectPositionType" />
15 <complexType name="DirectPositionType">
16   <simpleContent>
17     <extension base="gml:doubleList">
18       <attributeGroup ref="gml:SRSReferenceGroup" />
19     </extension>
20   </simpleContent>
21 </complexType>
22
23 <simpleType name="doubleList">
24   <list itemType="double" />
25 </simpleType>
```

---

Listing 1: GML Feature Type eines Punktes

Auf gleiche Weise wie bei der Definition eines Punktes definiert GML eine Reihe weiterer Feature Types zur Repräsentation von Kurven und Flächen. Wie im nächsten Abschnitt beschrieben stehen die Feature Types dem Entwickler einer Anwendung beim Schreiben seines Applikationsschemas zur Verfügung.

#### 4.2.2. Die Definition von Features im Applikationsschema

Der Entwickler einer Anwendung definiert Features, die in seiner Anwendung relevant sind. Diese Objekte können - aber müssen nicht - eine oder mehrere geometrische Eigenschaften haben. Zum Beispiel könnte der Entwickler wie in Listing 2 festlegen, dass eine Stadt durch ihren Namen sowie durch einen Punkt im Raum repräsentiert werden soll, also durch eine geometrische und eine nicht-geometrische Eigenschaft.

```
1 <?xml version="1.0" encoding="ISO-8859-1"?>
2 <xsd:schema targetNamespace="http://www.tuwien.ac.at/City"
3   xmlns:city="http://www.tuwien.ac.at/City"
4   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
5   xmlns:gml="http://www.opengis.net/gml/profile/sfgml/1.0"
6   elementFormDefault="qualified" version="1.0">
7
8   <xsd:import namespace="http://www.opengis.net/gml/profile/sfgml/1.0"
9     schemaLocation="gmlSimpleFeaturesProfile.xsd" />
10
11   <xsd:element name="City" type="city:CityType"
12     substitutionGroup="gml:_Feature" />
13   <xsd:complexType name="CityType">
14     <xsd:complexContent>
15       <xsd:extension base="gml:AbstractFeatureType">
16         <xsd:sequence>
17           <xsd:element name="location"
18             type="gml:PointPropertyType" minOccurs="1" maxOccurs="1" />
19           <xsd:element name="name" minOccurs="1"
20             maxOccurs="1">
21             <xsd:simpleType>
22               <xsd:restriction base="xsd:string">
23                 <xsd:maxLength value="9" />
24               </xsd:restriction>
25             </xsd:simpleType>
26           </xsd:element>
27         </xsd:sequence>
28       </xsd:extension>
29     </xsd:complexContent>
30   </xsd:complexType>
31 </xsd:schema>
```

Listing 2: Applikations-Schema city.xsd

In der Spezifikation des GML Simple Feature Profile [Inc., 2005, Annex C] findet sich noch ein weiteres Anwendungsbeispiel. Dort wird ein Datenschema zur Repräsentation von Straßenzügen definiert. Der nächste Abschnitt zeigt, wie vom Applikationsschema einzelne Instanzen von GML Dokumenten abgeleitet werden können.

#### 4.2.3. Feature Instances in GML Dateien

Listing 3 zeigt eine Instanz eines GML Dokuments, das von city.xsd abgeleitet wurde. Seinem Applikationsschema entsprechend beschreibt das Dokument eine Stadt, und gibt dazu den Namen der Stadt an, und als Ortsangabe einen Punkt.

---

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <city:City
3   gml:id="idvalue0"
4   xmlns:city="http://www.tuwien.ac.at/city"
5   xmlns:gml="http://www.opengis.net/gml/profile/sfgml/1.0"
6   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
7   xsi:schemaLocation="http://www.tuwien.ac.at/city
8     city.xsd http://www.opengis.net/gml/profile/sfgml/1.0
9     gmlSimpleFeaturesProfile.xsd ">
10  <city:location>
11    <gml:Point gml:id="point123">
12      <gml:pos>49 -123.55</gml:pos>
13    </gml:Point>
14  </city:location>
15  <city:name>Wien</city:name>
16 </city:City>
```

---

Listing 3: Beispieldaten City.xml

### 4.3. Abfragesprachen für GML

Da GML ein XML-basiertes Datenformat ist, können generische XML-Abfragesprachen auch zur Abfrage von GML Daten verwendet werden. Im Allgemeinen müssen diese Abfragesprachen jedoch für die Verwendung mit GML um räumliche Abfrage-Operatoren erweitert werden. Mittels dieser Operatoren können räumliche Abfragen (über die Lage von geographischen Features) und topologische Abfragen (über die relative Lage mehrerer Features zueinander) formuliert werden. Zur effizienten Auswertung solcher Abfragen muß ein Datenbanksystem räumliche Indizes verwalten (siehe Abschnitt 5.5.1).

Im Folgenden sollen zwei Abfragesprachen für GML verglichen werden: XQuery und die OGC Filter Encoding Spezifikation. Tabelle 3 gibt einen Überblick über die beiden Methoden.

XQuery	Filter Encoding
Technisch	Intuitiv
Schema-Information wird zum Formulieren einer Abfrage benötigt.	Formulieren einer Abfrage auch ohne Schema-Information möglich.
Wird von XML Datenbanken verstanden.	Wird von OGC Web Services verstanden.

Tabelle 3: Vergleich: XQuery versus Filter Encoding

```

1 for $var in document('city.xml')/City/CityHall where $var/@number = '1'
2 return geo:getCentroid($var)
3
4 #liefert als Ergebnis:
5 <wfs:Point>
6 <wfs:coordinates>17.333333333333332, 84.66666666666667</wfs:coordinates>
7 </wfs:Point>

```

Listing 4: Beispiel-Abfrage mit räumlich erweitertem XQuery

### 4.3.1. Geographisch erweitertes XQuery

XQuery [W3C, i] und XPath [W3C, h] sind vom W3C (World Wide Web Consortium, [W3C, g]) empfohlene Abfragesprachen für XML, und werden als solche von vielen XML Datenbanken unterstützt.

XPath ermöglicht das Selektieren von Dokumentteilen eines XML Dokuments mittels Pfadausdrücken mit strukturellen oder wertbasierten Kriterien. XPath wird von fast allen XML Datenbanken unterstützt. XQuery verwendet und erweitert XPath und erlaubt das Formulieren komplexerer Abfragen. Wichtig für geographische Anwendungen ist die von XQuery gebotene Möglichkeit, benutzerdefinierte Funktionen einzubinden, und so den regulären Sprachumfang von XQuery zu erweitern. In mehreren Arbeiten [Boucelma und Colonna, 2004; Gusheng und Xiaohua, 2004] wurde auf diese Art XQuery um räumliche Abfragemöglichkeiten erweitert, unter anderem auch für die Abfrage von bewegten Objekten [Chung u. a., 2004]. Auch die kommerzielle XML-Datenbank “Tiger Logic XDBMS” [RainingData:TigerLogicXDBMS, 2007] bietet ein um räumliche Funktionen erweitertes XQuery als Abfragesprache.

In Listing 4 zeigt ein einfaches Beispiel aus der Arbeit von Boucelma et.al [Boucelma und Colonna, 2004] eine Abfrage in XQuery, das für Geodaten erweitert wurde. In dem Beispiel wird nach dem Ort des Rathauses einer Stadt gesucht. (Das Datenschema des city.xml Dokuments in diesem Beispiel weicht jedoch von dem in Kapitel 4.2 vorgestellten Beispiel ab). Die Abfrage sucht (Zeile 1) nach XML-Elementen im Dokument city.xml, auf die Folgendes zutrifft: Das Element ist ein Kind des City-Elements und trägt den Namen CityHall. Weiters soll dieses Element ein Attribut number mit Wert 1 haben. Für alle Elemente dieser Art soll (Zeile 2) von der Funktion geo:getCentroid der Schwerpunkt berechnet werden. Das Ergebnis der Funktion wird als Ergebnis der Abfrage zurückgegeben.

Aus dem obigen Beispiel (Listing 4) wird ersichtlich, dass sich der Abfrage-Ausdruck in XQuery sehr stark an der Struktur des jeweils abgefragten GML Dokuments (bzw. des dazugehörigen Schemas)

orientiert. Zum Formulieren einer Abfrage ist daher Kenntnis über das Datenschema notwendig, und ohne dieser Kenntnis ist auch eine Abfrage der Art “Ich suche alle Features, egal welchen Typs, die innerhalb meines rechteckigen Suchfensters liegen” schwierig zu formulieren.

Im Gegensatz zu XQuery wirkt die im Folgenden beschriebene Abfragesprache *OGC Filter Encoding Specification*, die speziell für die Abfrage geographischer Daten entworfen wurde, weniger technisch, und näher der Denkweise eines menschlichen Benutzers.

#### 4.3.2. OGC Filter Encoding Spezifikation

Die Filter Encoding Spezifikation [OGC, f] des OGC ist eine Abfragesprache für geographische Merkmale, und wird in erster Linie zur Abfrage von OGC Web Services unterstützt und verwendet. Sie erlaubt es, einen Filterausdruck mit Einschränkungen für die Eigenschaften von (geographischen) Features zu formulieren. Als Ergebnis der Abfrage werden alle Objekte zurückgegeben, auf welche die in der Abfrage formulierten Einschränkungen zutreffen.

Filterausdrücke werden entsprechend der Spezifikation in XML formuliert. Zur Auswertung durch eine Datenbank können sie zu SQL, XQuery oder zu anderen Abfragesprachen übersetzt werden. In der Arbeit von Vidal et.al [Vânia Vidal, 2005] werden Anfragen an ein Web Feature Service (WFS) mit Filterausdrücken zu SQL/XML Abfrageausdrücken übersetzt. (SQL/XML [SQL/XML:Website, 2007] ist eine ISO Norm für den Umgang mit XML Daten, die in relationalen Datenbanken gespeichert sind. Unter anderem erlaubt SQL/XML auch das Einbetten von XQuery Abfragen in SQL Ausdrücken).

Listing 5 zeigt ein Beispiel für einen Abfrage-Ausdruck entsprechend der Filter-Encoding Spezifikation: Zwei Filter-Ausdrücke werden mit dem logischen Operator And verknüpft. Der erste Filter-Ausdruck schränkt eine nicht-geographische Eigenschaft ein: Die Eigenschaft DEPTH soll einen Wert unter 30 haben. Der zweite Ausdruck schränkt eine geographische Eigenschaft ein: Er verlangt, dass das gesuchte Merkmal mit einem rechteckigen Suchfenster überlappen soll.

Aus dem Beispiel in Listing 5 wird ersichtlich, dass Filter Encoding Ausdrücke intuitiv sehr gut verständlich sind. Ein weiterer Vorteil ist, dass zum Formulieren der Abfragen kein (oder kaum ein) Wissen über das Datenschema der zugrundeliegenden Daten notwendig ist. In der Tat müssen die Daten, die mit Filterausdrücken abgefragt werden auch nicht im GML Format gespeichert sein: Viele Implementierungen von Web Feature Services [OGC, c] unterstützen verschiedenartige Datenquellen, z.B. auch relationale Geodatenbanken. Oder auch die weit verbreitete, kommerzielle, objekt-relationale Geodatenbank Oracle Spatial [Oracle:SpatialDB, 2007] hat in der aktuellen Version 11 ein Web Feature Service Interface, und wird daher über die Filter Encoding Spezifikation abgefragt.

```
1 <Filter>
2   <And>
3     <PropertyIsLessThan>
4       <PropertyName>DEPTH</PropertyName>
5       <Literal>30</Literal>
6     </PropertyIsLessThan>
7     <Not>
8       <Disjoint>
9         <PropertyName>Geometry</PropertyName>
10        <gml:Envelope srsName="http://www.opengis.net/gml/srs/epsg.xml#63266405">
11          <gml:lowerCorner>13.0983 31.5899</gml:lowerCorner>
12          <gml:upperCorner>35.5472 42.8143</gml:upperCorner>
13        </gml:Envelope>
14      </Disjoint>
15    </Not>
16  </And>
17 </Filter>
```

Listing 5: Beispiel-Abfrage mit Filter Encoding Spezifikation OGC [f]

#### 4.4. Parser für GML

GML Daten können, weil GML auf XML basiert, mit generischen XML Werkzeugen geparkt werden. Weit verbreitete APIs zum Parsen von XML Daten sind DOM [W3C, a] und SAX [Megginson], es existieren aber auch Parser speziell für Geodaten im GML Format [Galdos:GmlSDK, 2007; GeoTools].

DOM-kompatible Parser müssen das gesamte XML Dokument in den Arbeitsspeicher laden, und sind deshalb für große Datenmengen nicht gut geeignet. Das ist besonders bei Geodaten ein Problem, weil Geodaten sehr umfangreich sein können - eine Landkarte kann aus vielen tausenden Objekte bestehen. Dazu kommt noch, dass Geodaten im GML Format mehr Speicherplatz brauchen, als bei anderen, kompakteren Formaten (mehr über den Speicherplatz-Bedarf von GML-Daten in Abschnitt 5.1). Nach dem Parsen kann über die DOM API das XML Dokument als Baumstruktur abgefragt werden.

SAX-kompatible Parser müssen nicht das gesamte XML Dokument in den Arbeitsspeicher laden, um es zu parsen und sind daher besser für Geodaten geeignet. SAX Parser operieren auf einem Datenstrom von XML Daten und liefern über die SAX API Ereignisse, wann immer der Parser auf XML Elemente und Attribute stößt. Dem Vorteil, auch mit großen Datenmengen umgehen zu können, steht der Nachteil gegenüber, dass SAX Parser nur sequenziellen Zugriff auf XML Daten erlauben (d.h. keinen wahlfreien Zugriff).

Die Dokumentation des open-source Projekts GeoTools [GeoTools:Website, 2007] enthält eine gute Übersicht über Vor- und Nachteile bei der Verwendung von DOM, SAX und anderen, zum Teil auf GML spezialisierte Parser. Das Projekt hat ausserdem einen XML Parser speziell für GML Daten implementiert, der GML Applikationsschemas zur Unterstützung des Parsing Vorgangs verwenden



```
1 <exp:City>
2   <gml:name>Belgrade</gml:name>
3   <exp:size>1,700,000</exp:size>
4   <gml:extentOf>
5     ...
6   </gml:extentOf>
7 </exp:City>
8
9 <gml:FeatureStyle featureType="exp:City">
10  <gml:LabelStyle>
11    <gml:style>font-family:Verdana;font-size:18;fill:red</gml:style>
12    <gml:label>
13      City:
14      <gml:LabelExpression>//City/name</gml:LabelExpression>
15      , Size:
16      <gml:LabelExpression>//City/size</gml:LabelExpression>
17    </gml:label>
18  </gml:LabelStyle>
19 </gml:FeatureStyle>
```

Listing 6: GML Map Styling

kann: Der schema-assisted Parser GTXML [GeoTools] nützt beim Parsen die Information aus dem Applikationsschema, um zu wissen, was für ein Binding zu Java Objekten verwendet werden soll.

#### 4.5. Visualisierung von GML Daten

Das GML Format hat eine strikte Trennung von Inhalt und Aussehen: Die Codierung geographischer Merkmale in GML ist unabhängig davon, wie diese Merkmale dargestellt werden sollen. Der GML Standard erlaubt jedoch, über einen *DefaultStyle* festzulegen, wie Merkmale (bzw. Merkmals-Typen aus dem Applikationsschema) dargestellt werden sollen. Listing 6 zeigt ein Beispiel aus dem GML Standard [OGC, 2004, Kap. 20.6], wo festgelegt wird, wie Merkmale vom Typ `exp:City` dargestellt werden sollen. Insbesondere wird ein *LabelStyle* festgelegt, sodass das Label der Stadt Belgrad auf der Landkarte den Text `City: Belgrade, Size: 1,700,000` zeigen wird. Über ähnliche Angaben lassen sich in GML auch andere graphische Symbole gestalten. Das Gestalten ist auch in Abhängigkeit von Eigenschaften des Merkmals möglich, um z.B. verschieden große Punkte je nach Größe einer Stadt anzuzeigen. Weiters kann die graphische Repräsentation über SMIL [W3C, c] animiert werden.

Die OGC Web Services WMS, WFS und WCS (siehe Kapitel 4.7) verwenden eine andere Sprache, nämlich *Styled Layer Descriptor (SLD)* [OGC, k], zur Spezifikation der Darstellung von Features. Der Client des WebServices kann über SLD angeben, welche Ebenen einer Landkarte angezeigt werden sollen, und auf welche Weise einzelne Merkmalstypen dargestellt werden sollen.

Um in einer Anwendung geographische Daten im GML Format bildhaft darzustellen, müssen die im Folgenden genannten drei Funktionen implementiert werden [Lu u. a., 2007, Kap. 4]:

1. Merkmals-Extraktion: Extrahieren der geographischen Merkmale (z.B. Brücken, Flüsse, Seen und Straßen) aus den Geodaten.
2. Map Styling: Festlegen, auf welche Art die geographischen Merkmale visuell repräsentiert werden sollen (durch z.B. gefärbte Linien und Flächen).
3. Rendern: Umwandeln der Daten zu Bildern (z.B. im Jpeg Format).

In einer Client / Server - Architektur kann jede dieser drei Funktionen jeweils beim Client oder beim Server implementiert werden. Die folgenden Abschnitte vergleichen die daraus resultierenden Architekturen.

#### 4.5.1. GML Viewer

Ein GML Viewer (als Client zu einem Server mit GML Geodaten) muß alle drei oben genannten Funktionen (Merkmals-Extraktion, Styling und Rendern) implementieren, um Geodaten im GML Format vom Server zu laden und für den Benutzer zu visualisieren.

Desktop-GIS Systeme bieten häufig diese Funktionalität und können GML Daten aus Dateien oder von WebFeatureServices laden und darstellen. Im Unterschied zu Desktop GIS Systemen haben GML Viewers üblicherweise keine Editierfunktion, sondern dienen bloß zum Betrachten der Daten. Die Firmen Snowflake Software [Snowflakesoftware:GMLViewer, 2007] und TatukGis [TatukGis:Viewer, 2007] bieten gratis erhältliche GML Viewers als Demo für ihre ansonsten kommerziell vertriebenen Produkte an. Manche GML Viewers können als Plugin im Webbrowser des Benutzers ausgeführt werden, so zum Beispiel das open-source Produkt MapBuilder GML Viewer [MapBuilder:Website, 2007].

Charakteristisch für eine Architektur mit einem GML Viewer als Client sind die folgenden Punkte:

- Der Client implementiert Funktionen eins bis drei (Merkmals-Extraktion, Styling, Rendering)
- Der Server gibt die Original Geodaten aus der Hand. Oft wird das bei proprietären Geodaten, die teuer verkauft werden, vermieden - dann bekommt der Client nur Zugriff auf fertig gerenderte Bilder.
- Dem Client bleibt die Semantik der Daten (z.B. Merkmale wie Straßen, Brücken und Flüsse) erhalten.
- Der Client ist selbst für das Map Styling verantwortlich, und kann es selbst beeinflussen.
- Stufenloses Hinein- und Herauszoomen ohne Qualitätsverlust ist möglich.

#### 4.5.2. Vektordaten Viewer

Ein Vektordaten-Viewer bekommt vom Server keine geographischen Merkmale, sondern nur deren graphische Repräsentation in der Form von Vektordaten (d.h. in einer Darstellung aus Punkten, Linien und Flächen). Um diese Daten darstellen zu können, muß ein Vektordaten-Viewer kein Verständnis für geographische Daten haben - er muß nur die Vektordaten auf den Bildschirm des Benutzers rendern.

XML-basierte Vektordatenformate sind gut für die Verwendung mit GML geeignet, weil sich GML über XML Transformationen in andere XML Formate übersetzen läßt. Mehrere Arbeiten [Ye u. a., 2005; Peng und Zhang, 2004; Guo u. a., 2003] verwenden XSLT Transformationen, um GML Daten (zusammen mit separaten Angaben für das Styling) in das XML-basierte Vektorgraphikformat SVG [W3C, d] zu übersetzen. Gegenüber anderen Vektorgraphik-Formaten hat SVG den Vorteil, dass SVG ein weit verbreiteter und frei erhältlicher Standard des W3C ist. Allerdings unterstützt SVG nur 2-dimensionale Koordinaten.

Im Gegensatz zu SVG unterstützt der zur ISO Norm erhobene X3D [X3D:About, 2008] Standard auch dreidimensionale Daten. X3D ist der Nachfolger von VRML, und verwendet nach wie vor die Syntax von VRML - zusätzlich jedoch können X3D Szenen auch in einer XML-basierten Syntax codiert werden. Mit Hilfe der GeoVRML Erweiterung [GeoVRML:Website, 2007] können VRML-Daten mit einer Ortsangabe versehen werden können. Auch in X3D wird diese Möglichkeit geboten; die entsprechende Erweiterung heißt "Geospatial component" und wird von der X3D Earth Working Group [X3DEarth:Website, 2008] entwickelt.

Ein Nachteil vieler vektorbasierter Formate ist die mangelnde Unterstützung durch Webbrowser: Häufig muß der Benutzer Plugins oder andere Software installieren, um Vektorgraphiken darstellen zu können. Diese Plugins sind in der Regel nicht für alle Plattformen und Browser verfügbar. Für das SVG Format existiert jedoch eine Java-basierte und damit Betriebssystem-unabhängige Lösung: Das open-source Projekt Apache Batik [ApacheBatik:Website, 2008] kann verwendet werden, um SVG Daten in einem Java Applet im Browser zu rendern. Entsprechendes ist für das Rendern von X3D Daten mit der ebenfalls Java-basierten Implementierung Xj3D [Xj3D:Website, 2008] möglich.

Vorteile bei der Verwendung von Vektorgraphiken gegenüber Bitmap Graphiken sind der geringere Speicherbedarf und die Möglichkeit, ohne Verpixelung stufenlos hinein und hinaus zu zoomen [Peng und Zhang, 2004, Kap. 1].

Charakteristisch für eine Architektur mit einem Vektordaten-Viewer sind die folgenden Punkte:

- Der Server implementiert Funktionen eins und zwei (Merkmals-Extraktion und Styling), der Client ist nur für das Rendern zuständig.
- Der Server gibt nicht die original-Daten aus der Hand, sondern nur eine Abbildung.
- Der Client kennt die Semantik der Daten nicht. (z.B: Ob es sich bei einer Linie um eine Straße, einen Fluß, oder eine Landesgrenze handelt)

- Stufenloses Hinein- und Herauszoomen ohne Qualitätsverlust ist möglich.

#### 4.5.3. Bitmap Viewer

Jeder Webbrowser, aber auch andere, auf Geodaten spezialisierte Anwendungen, sind in der Lage, pixelbasierte Bilddaten zu lesen und darzustellen. Ein entsprechender Client bekommt fertige, zu Bildern gerenderte Kartenausschnitte vom Server zugeschickt. Um diese Bilder darstellen zu können, muß er keine speziellen Fähigkeiten für den Umgang mit Geodaten haben. Als Server-Komponente ist das OGC WebMapService (WMS, siehe Abschnitt 4.7) geeignet, weil es Kartenausschnitte in pixelbasierten Bildformaten ausgeben kann.

Wegen der geringen Anforderungen an die Client Software sind Bitmaps eine weit verbreitete Möglichkeit, um Geodaten über das Internet zu veröffentlichen. Beispielsweise können auf diese Weise moderne Webbrowser mit JavaScript-Unterstützung als Client zum Betrachten von Geodaten verwendet werden. Viele Produkte folgen diesem Ansatz. Eine Liste frei erhältlicher Produkte findet sich auf der FreeGIS.org Website [Wagner, 2008] in der Kategorie WebGIS.

Pixelbasierte Bildformate, die von vielen Webbrowsern dargestellt werden können, sind PNG, JPEG und GIF. Das GeoTIFF [GeoTiff:Website, 2007] Format und der OGC Standard zum Einbetten von GML in JPEG 2000 [OGC, 2007c] ermöglichen das Einfügen geographischer Referenzen (Koordinaten des Bildausschnitts und die verwendete Kartenprojektion) direkt in die Bilddatei.

Obwohl die zu Bitmaps gerenderten Karten den Vorteil haben, dass viele Anwendungen mit dieser Art von Daten umgehen können, hat der Ansatz auch Nachteile: Beim Hineinzoomen in einen Kartenausschnitt entstehen unscharfe und verpixelte Bilder, wenn der Kartenausschnitt nicht neu vom Server geladen wird. Weiters ist der hohe Speicherplatzbedarf der Bilder von Nachteil und führt (bei langsamen Internet-Verbindungen) zu schlechten Reaktionszeiten des Systems.

- Der Server implementiert Funktionen eins bis drei (Merkmals-Extraktion, Styling, Rendering).
- Der Server gibt nicht die Original-Daten aus der Hand, sondern nur eine Abbildung.
- Geringe Anforderungen an den Client - jeder moderne Webbrowser ist gut genug.
- Der Client kennt die Semantik der Daten nicht.
- Probleme mit Unschärfe und Verpixelung beim Hineinzoomen in Kartenausschnitte.
- Probleme mit langsamen Internet-Verbindungen wegen des hohen Speicherplatzbedarfs von pixelbasierten Geodaten.

#### 4.6. GML zur Integration von Geodaten

Moderne Computeranwendungen wie Sensor Networks oder soziale Webanwendungen führen dazu, dass im Internet eine große Menge an Information in heterogenen Systemen zur Verfügung steht.

Potenziell kann diese Information genutzt werden, um Echtzeit- oder auch historische Modelle unserer Welt zu erstellen und zugänglich zu machen. Damit wäre es möglich, die Welt virtuell zu erkunden - die Autoren Castelli et.al [Castelli u. a., 2007] nennen diese Möglichkeit "Browsing the World".

Die Herausforderung bei der Realisierung solcher Anwendungen ist, Daten aus heterogenen Quellen so zu integrieren, dass sie on-demand über eine zentrale Schnittstelle abfragbar sind. Häufig werden zur Integration heterogener Daten XML-basierte Datenformate verwendet, und im Fall geographischer Daten das auf XML basierende Format GML.

Der folgende Abschnitt beschreibt, worin sich heterogene Datenquellen unterscheiden können, und nennt die aus der Heterogenität resultierenden Probleme.

#### 4.6.1. Heterogenität verteilter, autonomer Informationssysteme

Geoinformation (sowie beliebige andere, nicht-geographische Information) wird im Internet in verteilten, autonomen und heterogenen Informationssystemen zur Verfügung gestellt. Die Heterogenität dieser Systeme folgt aus ihrer Autonomie, die nach der Doktorats-Arbeit von Goh [Goh, 1997] wie folgt klassifiziert werden kann:

*Design Autonomie:* Die Möglichkeit, ein eigenes Informations- und Datenmodell und eine beliebige Implementierung zu wählen. Design-Autonomie führt zu *Daten-Heterogenität* (unterschiedliche Arten, Daten zu organisieren und zu interpretieren) und zu *System-Heterogenität* (betreffend Unterschiede im Datenmodell, in der Datenbearbeitungs-Sprache, in Concurrency Control...). Die beiden soeben genannten Arten der Heterogenität erschweren die semantische Interoperabilität, das ist die Fähigkeit zum sinngemäßen Informationsaustausch zwischen Systemen.

*Kommunikations-Autonomie:* Die Möglichkeit zu entscheiden, mit welchen anderen Systemen was für Information ausgetauscht werden soll.

*Ausführungs-Autonomie:* Die Möglichkeit zu entscheiden, wie und wann Anfragen von anderen Systemen ausgeführt werden sollen. Führt zusammen mit der Kommunikations-Autonomie zu Schwierigkeiten bei der Auswertung und beim Optimieren von Abfragen.

In Bezug auf die Daten, die zwischen heterogenen Systemen ausgetauscht oder zu einem neuen System integriert werden sollen, unterscheidet Goh drei Arten von Heterogenität, und nennt die daraus folgenden Konflikte:

*Schematische Heterogenität.* Unterschiede in der Struktur, wie Daten organisiert sind, führen zu den folgenden Konflikten: Datentyp-Konflikte (Unterschiede in der Repräsentation primitiver Datentypen, e.g. Datumswerte). Namensgebungs-Konflikte (Synonyme und Homonyme in der Benennung von Attributen und Entitäten, können durch Umbenennen aufgelöst werden). Aggregations-Konflikte (Unterschiede in der Gruppierung von Daten: Z.B. können Lagerbestände zu jeder Ma-

schine gespeichert werden, oder alle gemeinsam in chronologischer Ordnung). Generalisierungs-Konflikte (Unterschiede in den Beziehungen der Entitäten untereinander: Z.B. getrennte Modellierung von ‘Manager’ und ‘Ingenieur’ versus gemeinsame Modellierung als ‘Angestellter’).

*Semantische Heterogenität.* Konflikte in der Namensgebung (Synonyme und Homonyme in den Attributwerten). Konflikte bezüglich Maßeinheiten. Konflikte durch Verwechslung (Verwecheln zweier Konzepte, die eigentlich verschieden sind).

*Intentionale Heterogenität.* Domänen-Konflikte (Zwei Systeme bilden jeweils ein anderes “Universe of Discourse” ab). Konflikte in den Integritätsbedingungen (Zum Beispiel ist das Attribut ‘Name’ in einem System eindeutig, in einem anderen nicht).

#### 4.6.2. Ansätze zur Integration von Geodaten aus heterogenen Informationssystemen

Die im vorigen Abschnitt angeführten Heterogenitäten und die daraus resultierenden Konflikte stehen der Integration von Daten aus verschiedenen Quellen im Internet entgegen. Die Autoren Guan et.al [Guan u. a., 2003] beschreiben verschiedene Ansätze zur Lösung dieses Problems in Hinsicht auf die Integration von Geodaten. Eine Alternative zur Integration von Daten ist die Verwendung von Suchmaschinen.

*Kataloge:* Das Katalogisieren von verschiedenen Datenquellen und deren Eigenschaften ist Voraussetzung für die Integration heterogener Daten.

*Schnittstellen zwischen Datenbanken:* Definieren eines globalen Schemas und damit Erstellen einer globalen Sicht auf Daten unterschiedlicher Quellen.

*Data Warehousing:* Sammeln von Daten aus unterschiedlichen Quellen in einen zentralen Speicher.

*Mediator-basierte Systeme:* Daten aus unterschiedlichen Quellen über eine Schnittstelle (einen Mediator) abfragbar machen, ohne die Daten selbst an eine zentrale Stelle zu kopieren.

*Suchmaschinen:* Eine auf geographische Daten spezialisierte Suchmaschine kann bei der Suche nach passender geographischer Information helfen. Die Integration der auf diese Weise gefundenen, heterogen Daten bleibt jedoch Aufgabe des Benutzers.

Von diesen Ansätzen sollen im Folgenden Mediator-basierte Systeme und Suchmaschinen genauer beschrieben werden.

#### 4.6.3. Geographische Suchmaschinen

Die Integration heterogener Daten zu einem homogenen Gesamtsystem ist schwierig und für den Betreiber des Systems sehr aufwändig sein [Gibotti u. a., 2005]. Ein möglicher Ausweg ist das

Anbieten spezialisierter Suchmaschinen: In diesem Fall ist die Arbeit des Informationsanbieters darauf beschränkt, seine Daten im Internet zur Verfügung zu stellen.

Die Autoren Gibotti et.al [Gibotti u. a., 2005] nennen als Beispiel für spezialisierte Suchmaschinen CiteSeer [CiteSeer:Website, 2007] und Google Scholar [Google:Scholar, 2007], das sind Suchmaschinen, die auf die Suche nach wissenschaftlichen Artikeln spezialisiert sind. Spezialisierte Suchmaschinen wissen über die Struktur des von ihnen gesuchten Inhalts Bescheid (zum Beispiel bestehen wissenschaftliche Arbeiten aus dem Titel, der Liste mit Autoren, einer Zusammenfassung, dem Text und einem Literaturverzeichnis), und sie nützen dieses Wissen bei der Indizierung der von ihnen durchsuchten Inhalte.

Auch Geodaten haben eine typische Struktur, und können von spezialisierten Suchmaschinen gefunden werden. Gibotti et.al haben eine solche Suchmaschine “GeoDiscover” gebaut, die auf Geodaten im ESRI Shape File Format spezialisiert ist. Im Gegensatz zur GeoDiscover Suchmaschine ist die ebenfalls auf geographische Daten spezialisierte Suchmaschine Geometa.info [Geometa.info:Website, 2007] von Keller und Kälin [Keller und Kälin, 2005] frei im Internet verfügbar und läuft als Forschungs-Prototyp im Testbetrieb. Die Suchmaschine findet Geodaten aus Deutschland, Österreich und der Schweiz und hat derzeit (Jänner 2008) mehr als 4600 Dokumente indiziert.

Weil Suchmaschinen das Problem der Daten-Integration eher umgehen als lösen (die Suchende Person muss sich selbst um die Integration der gefundenen Daten kümmern) wird im Rahmen dieser Arbeit nicht weiter auf geographische Suchmaschinen eingegangen.

#### 4.6.4. Mediator-basierte Systeme

In etlichen Arbeiten wurden Mediator-basierte Systeme zur Integration von Geodaten vorgeschlagen. Solche Systeme bieten eine homogene Sicht auf Daten aus heterogenen Datenquellen. Diese homogene Sicht läßt sich über eine Schnittstelle abfragen. Bei der Auswertung der Abfrage übernimmt das System die Rolle eines Mediators, d.h. es leitet die Abfragen in jeweils geeigneter Form an die einzelnen Datenquellen weiter, sammelt und integriert die Ergebnisse und gibt sie an den Benutzer zurück.

Die Autoren An und Zhao [An und Zhao, 2006] beschreiben die drei-schichtige *Architektur* eines Mediator-Systems (siehe Abbildung 15): Eine “Foundation Layer” bindet die Datenquellen ein. Eine “Mediation Layer” unterstützt den Austausch von Anfragen und Ergebnissen mit den einzelnen Datenquellen. Und eine “Application Layer” bietet eine Benutzerschnittstelle zur Bedienung des Systems.

Das vorhin genannte System von An und Zhao verwendet GML zur Integration der Geodaten. Das gilt auch für das “VirGIS” System der Autoren Boucelma et.al [Boucelma u. a., 2003]. Eine vereinfachte Darstellung dieses Systems findet sich in Abbildung 16. VirGIS dient zur Integration von Daten aus verschiedenen WFS (Web Feature Service, siehe Abschnitt 4.7) Servern. Das System läßt sich über ein WFS Interface auf einem globalen GML Schema abfragen. Die Abfrage-Abarbeitung geschieht in mehreren Schritten wie folgt: WFS Anfragen an VirGIS werden zu GQuery Abfragen übersetzt (GQuery ist eine Erweiterung von XQuery um räumliche Operatoren [Boucelma und Colonna, 2004], siehe



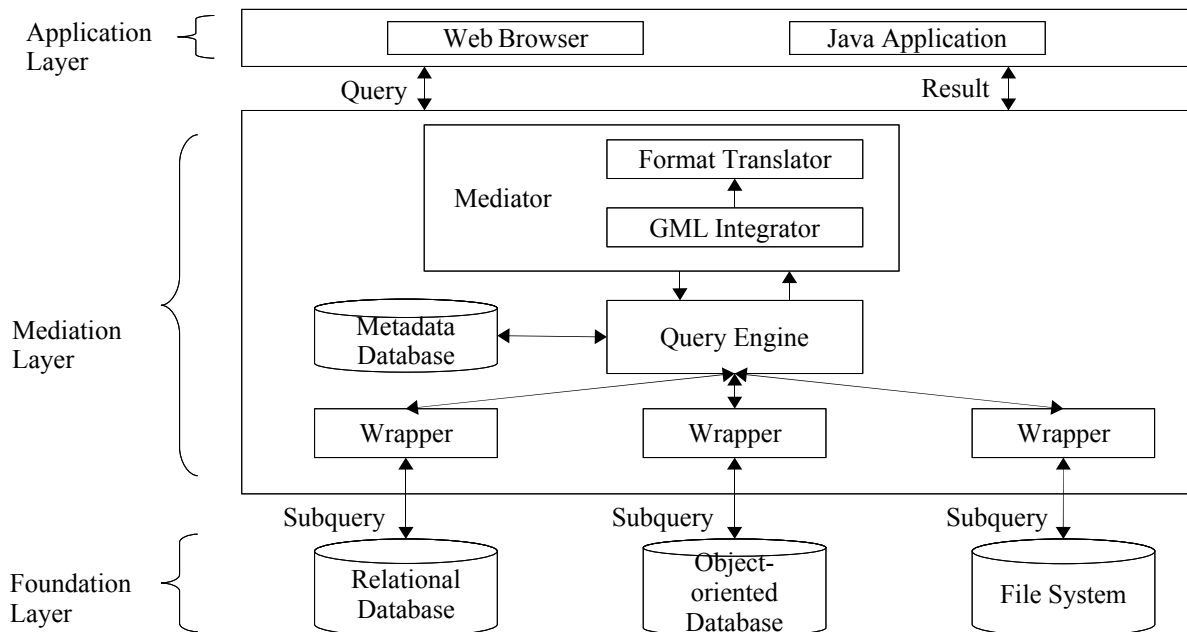


Abbildung 15: Die dreischichtige Architektur eines Mediator-Systems besteht aus einer Foundation Layer, Mediation Layer und Application Layer. Aus [An und Zhao, 2006, Abb.1]

auch Abschnitt 4.3.1). Der resultierende GQuery Ausdruck ist eine Abfrage über dem globalen GML Schema, welches die Daten der einzelnen WFS Datenquellen vereinheitlicht. Der GQuery Ausdruck wird nun zu einem Ausführungsplan mit Abfragen über den lokalen Schemas der einzelnen WFS Server übersetzt. Entsprechend dem Ausführungsplan werden dann die Anfragen abgearbeitet, die Ergebnisse werden zusammengesetzt und über die WFS Schnittstelle des VirGIS Systems zurückgegeben.

Auch die in Abschnitt 2.3 vorgestellte INSPIRE Initiative der europäischen Union verwendet WebServices zur Integration von Geodaten und schlägt eine Service-orientierte, europäische Geodateninfrastruktur vor. Im nächsten Abschnitt 4.7 wird die dabei verwendete Technik geographischer WebServices näher beschrieben.

Eine weitere Möglichkeit, Daten mit verschiedenen Schemas zu integrieren, ist die Verwendung von Ontologien. Wenn die Daten in XML Form vorliegen, und neben XML noch andere Techniken des W3C zum Einsatz kommen, wird dieser Ansatz mit "Semantic Web" [W3C, b] assoziiert. Etliche Arbeiten verwenden den Semantic Web Ansatz, um Geodaten aus heterogenen Datenquellen zu integrieren:

Die Autoren Córcoles und González [Córcoles und González, 2004] verwenden RDF (das Resource Description Framework [W3C, b] des W3C) zur Definition eines virtuellen, globalen Datenschemas. Ein Mediator-basiertes System kann über diesem globalen Schema abgefragt werden. Das Faktum, dass es sich bei den zu integrierenden Daten um geographische Daten handelt, bringt zusätzliche Herausforderungen: (1) Die Semantik der räumlichen Operatoren in der Abfragesprache muß mit den Datenquellen übereinstimmen. (2) Um räumliche Abfragen auf GML Daten effizient abarbeiten zu



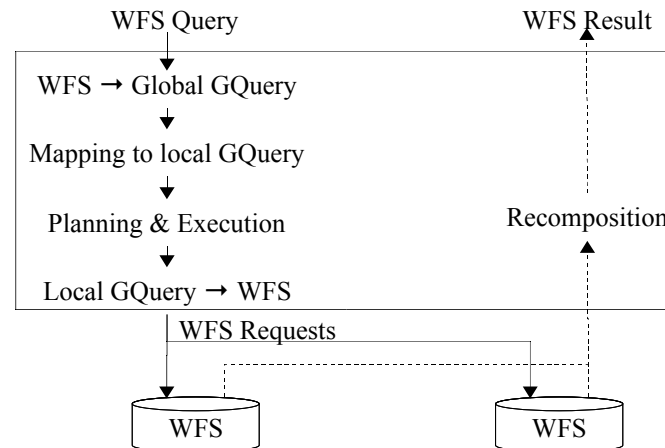


Abbildung 16: Architektur des VirGIS Mediator-Systems: Hier sind die Datenquellen Web Feature Services (WFS), und das Mediator-System wird selbst über das WFS Interface abgefragt. [Boucelma u. a., 2003, Abb.1]

können, müssen die Datenquellen räumliche Indizes verwalten (mehr über die Indizierung von GML Daten in Abschnitt 5.5.1). (3) Zusätzlich zu den Datenschemas der einzelnen Datenquellen soll der Mediator auch Bescheid wissen, was für Information in welcher Datenquelle tatsächlich verfügbar ist - sonst müsste der Mediator eine Abfrage an alle Datenquellen mit passendem Schema verschicken, auch wenn die gesuchte Information nur in einem Bruchteil dieser Datenquellen vorhanden ist. Die Autoren schlagen zu diesem Zweck in RDF formulierte “Descriptions” für die Datenquellen vor. Anhand dieser Description weiß der Mediator, dass z.B. Datenquelle XY Information über Madrid enthält.

Auch die Autoren Guan et.al [Guan u. a., 2003] verwenden Ontologien zur Integration heterogener GML Daten. Jedem lokalen GML Applikationsschema wird eine eigene, lokale Ontologie zugeordnet. Eine globale Ontologie verknüpft die lokalen Ontologien und schafft so ein gemeinsames Vokabular. Die Verknüpfung (englisch: ‘matching’, ‘mapping’) zwischen den einzelnen Datenquellen geschieht auf Schema-Ebene und nicht auf Daten-Ebene, d.h. nicht die Daten, sondern deren Applikationsschemas werden übersetzt. Die Granularität dieses Schema Mappings kann Element-weise sein (d.h. einzelne Elemente oder Attribute aus den Applikationsschemas werden aufeinander abgebildet), aber auch für ganze Strukturen von Elementen.

## 4.7. Geographische WebServices

Geographische WebServices bilden eine standardisierte Schnittstelle für den Zugriff auf geographische Information über das Internet. Grundlegend für die geographischen WebServices des OGC ist die Spezifikation “OpenGIS Service Architecture” [OGC, b]. Sie bietet einen Rahmen für die Entwicklung von Software, die es ihren Benutzern erlaubt, auf geographische Daten verschiedener Quellen über ein generisches Interface zuzugreifen. Die Spezifikation geht nicht auf konkrete Implementierungen

ein, nennt jedoch eine Reihe möglicher Plattformen für die Umsetzung. Eine solche Plattform sind WebServices.

WebServices werden in [W3C, f] vom World Wide Web Consortium (W3C, [W3C, g]) als ein Softwaresystem zur Unterstützung interoperabler Maschinen-zu-Maschinen Interaktion über ein Netzwerk definiert: Ein WebService bietet eine Beschreibung seiner Schnittstelle in einem maschinenlesbaren Format (Web Services Description Language, WSDL) und kommuniziert mit anderen Systemen typischerweise mittels XML Daten über HTTP.

Das OGC hat für den Austausch geographischer Daten mehrere WebService Standards spezifiziert. Diese Standards erfreuen sich großer Beliebtheit, und viele kommerzielle und auch viele freie / open-source Produkte (sowohl webbasierte wie Desktop-Anwendungen) sind damit kompatibel. Die Anzahl kompatibler Produkte wird auch aus einer vom OGC geführten Liste registrierter Produkte [OGC, 2007a] ersichtlich.

Die OGC WebService Standards lassen sich in die folgenden drei Kategorien einteilen [Lu u. a., 2007, Kap. 5.2]:

**Data Services** stellen ihren Nutzern geographische Information zur Verfügung.

Das Web Map Service (WMS, [OGC, d]) rendert geographische Information zu Bildern. Es läßt sich im Wesentlichen durch die Angabe eines rechteckigen Kartenausschnitts abfragen. Das Web Feature Service (WFS, [OGC, c]) liefert geographische Features im GML Format. Es läßt sich über in XML formulierte Filterausdrücke [OGC, f] (siehe Abschnitt 4.3) abfragen. Das Web Coverage Service (WCS, [OGC, l]) liefert geographische Daten, die nicht als Objekte sondern als Coverages modelliert sind.

**Processing Services** übernehmen geographische Berechnungen.

Das Web Processing Service (WPS, [OGC, m]) übernimmt für seine Benutzer geographische Berechnungen. Die dazu benötigten Daten können am WPS Server verfügbar sein oder vom Client zur Verfügung gestellt werden. Der Standard sagt nichts über Art der Berechnung noch über die Art des Rückgabewerts, sondern beschreibt die Schnittstelle, über die Rechenaufgaben formuliert und an den Server geschickt werden können, und über welche die Art des Rückgabewerts beschrieben werden kann.

**Catalogue Services** stellen einen Katalog von Metadaten zur Verfügung.

Das Catalogue Service [OGC, a] unterstützt die Möglichkeit, Metadaten zu veröffentlichen und mittels einer Abfragesprache zu durchsuchen. Metadaten sind im Rahmen dieses Standards beschreibende Information über geographische Daten, Services und damit in Zusammenhang stehende andere Informationsobjekte.

Unter den verschiedenen OGC Standards für geographische WebServices erscheinen für den Rahmen dieser Arbeit besonders die Standards WFS und WMS interessant, weil sie eine mögliche Schnittstelle

für den Zugriff auf Geodatenbanken darstellen: WFS bietet eine standardisierte Schnittstelle für lesen- und schreibenden Zugriff auf die geographischen Merkmale, und WMS bietet eine standardisierte Schnittstelle, um geographische Daten in Bildform als Landkarten-Ausschnitte darzustellen.

Die Nützlichkeit der OGC WebServices als Schnittstellen für den Zugriff auf Geodatenbanken wird unter anderem auch von den Herstellern der Datenbank Oracle Spatial erkannt: In der neuen Version 11g [Oracle:SpatialDB, 2007] wird die Datenbank mit einer WFS Schnittstelle ausgeliefert.

WebServices können zur *Integration von Geodaten* unterschiedlicher Herkunft verwendet werden. Dabei werden lokal verwaltete, heterogene Datenbanken zu einem Gesamtsystem integriert. Jede einzelne Datenbank wird mit einer WebService Schnittstelle ausgestattet, sodass auf die selbe Art auf alle Datenquellen zugegriffen werden kann. Mehr über die Integration von Geodaten findet sich in Abschnitt 4.6 dieser Arbeit.

Weiters können WebServices zum Aufbau einer dezentralen *Geodaten-Infrastruktur* (GDI, oder englisch SDI, Spatial Data Infrastructure) verwendet werden. Eine entsprechende Architektur ist als technische Lösung für die INSPIRE Initiative der europäischen Union vorgesehen (mehr über INSPIRE in Abschnitt 2.3).

## 5. Speicherung von GML

Die zunehmende Verwendung von GML insbesondere zum Austausch von Geodaten hat auch die Nachfrage nach effizienter Speicherung und Abfrage von Daten im GML Format gesteigert. Statt die Daten nur für den Austausch mit anderen Systemen zu GML zu übersetzen, kann GML auch intern zur Speicherung und Abfrage verwendet werden. Es gibt dazu mehrere Ansätze, die jeweils Vor- und Nachteile haben.

In der Arbeit von Chang-Tien Lu et.al [Lu u. a., 2007] wird ein systematischer Überblick über Speicherung und Abfrage von GML Daten gegeben, und auch im Rahmen dieser Arbeit sollen verschiedene Methoden zum Speichern von GML verglichen werden. Tabelle 4 gibt einen Überblick über Speichermöglichkeiten für GML, und die folgenden Abschnitte beschreiben diese Möglichkeiten im Detail. Zuvor soll jedoch noch auf den Aspekt des Speicherplatz-Bedarfs von GML Daten eingegangen werden.

### 5.1. Speicherplatz und Kompression von GML Daten

Aufgrund seiner Codierung in XML braucht GML mehr Speicherplatz als andere, kompaktere Formate. Das ist auch aus dem Beispielcode in Abschnitt 4.2 ersichtlich. Es läßt sich jedoch - und das ist ein häufig gesehener Ansatz - Datenkompression zur Kompression von GML Daten verwenden:

Die Firma Google verwendet .zip-Kompression um ihre in XML codierten Geodaten zu .kmz Dateien zu komprimieren [Google Inc.]. Das proprietäre Produkt “GML Viewer” der Firma Snowflakesoftware [Snowflakesoftware:GMLViewer, 2007] folgt dem gleichen Ansatz und kann GML Dateien lesen, die mit Gzip oder Winzip komprimiert wurden.

Die Kompression von GML Daten ist jedoch nicht nur des Speicherbedarfs wegen relevant, sondern ist besonders für die Übertragung von GML Daten über das Internet von großer Bedeutung. In der Arbeit von Yingwei et.al [Yingwei u. a., 2004] wird Datenkompression für die Datenübertragung zwischen geographischen WebServices genutzt.

Die Autoren Min, Park und Chung [Min u. a., 2006] schlagen den Kompressionsalgorithmus XPRESS für XML Daten vor, der Abfragen (lesend und schreibend) direkt auf den komprimierten Daten unterstützt und unter anderem auch für GML Daten geeignet ist.

Im Gegensatz zu den bisher genannten Ansätzen ist der GPress Kompressor der Autoren J.Guan und S.Zhou [Guan und Zhou, 15–20 April 2007] auf die Kompression von GML Daten spezialisiert. Bei diesem Ansatz werden räumliche Daten (Koordinaten) getrennt von den restlichen Daten mit einem eigens dafür entwickelten Algorithmus komprimiert. Die damit erzielten Kompressionsraten liegen deutlich über denen von generischen Tools wie XMill.

Ein einfacherer Ansatz der Autoren R. Demontis et.al [Demontis u. a., 2005] verwendet Abkürzungen statt der oft recht langen GML-Elementnamen und erreicht so um bis zu mehr als die Hälfte kleinere Dokumente. Während Kompressions-Algorithmen bessere Ergebnisse erzielen, hat dieser Ansatz den

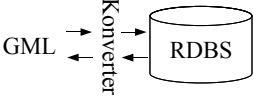
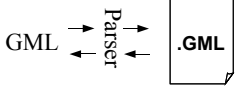

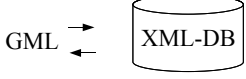
Speichermethoden für GML	Vorteile	Nachteile
Relationale Datenbank 	Erprobte Technik, viele Produkte	GML wird nur als Import/Export Format gesehen und läßt sich nicht über XML Abfragesprachen abfragen.
Dateisystem 	Einfach und für kleine Datenmengen gut geeignet	Synchroner Zugriff ist schwierig. Keine Indizierung. Abfragemöglichkeit muß getrennt implementiert werden. Speicherplatz bei großen Datenmengen
Relationale, XML-enabled Datenbank 	Erprobte relationale Datenbanktechnik, räumliche Indizes	Schwierigkeit des Mappings ins relationale Datenmodell. Abfragesprache muß um räumliche Funktionen erweitert werden.
Native XML Datenbank 	Standard XML Tools und APIs. Für XML passendes Datenmodell.	Weniger erprobte Technik. Datenbank muß um räumliche Indizes und die Abfragesprache um räumliche Funktionen erweitert werden.

Tabelle 4: Speichermöglichkeiten für GML Daten

Vorteil, dass zur Kompression und Dekompression nicht viel Rechenleistung benötigt wird. In der Folge ist der Ansatz auch für die Anwendung in mobilen Geräten geeignet.

Wie viel Speicherplatz zur Speicherung von GML Daten benötigt wird hängt auch von der Speichermethode (e.g. als Dateien auf einer Festplatte oder in einer Datenbank) ab: In der Diplomarbeit von B. Shrestha [Shrestha, 2004] wird beobachtet, dass GML Dokumente, die in der Oracle XML DB [Oracle] Version 9i gespeichert werden (egal ob strukturiert in objekt-relationalen Tabellen oder unstrukturiert als CLOB Textfeld) mehr Speicherplatz benötigen als die ursprünglichen GML Dateien.

Die folgenden Abschnitte vergleichen verschiedene Speichermethoden für GML Daten.

## 5.2. Konvertierung von GML zur Speicherung in relationalen Datenbanken

Eine uneigentliche Art der Speicherung von GML ist häufig anzutreffen: Viele geographische Informationssysteme und deren Middlewares verwenden GML als Format für den Datenaustausch, aber intern werden die Geodaten nicht in GML repräsentiert. Oft ist der Grund für eine solche Architektur die Neuheit des GML Formats: Historisch gewachsene Systeme wurden um GML als ein weiteres Datenformat erweitert, und können Geodaten von GML importieren und zu GML hin exportieren

Relationale Datenbank	XML Datenbank
+ Ausgereift und effizient	- Neuartige Produkte, große Unterschiede in der Effizienz
+ Räumliche Indizes und Abfragefunktionen	- Nur wenige Produkte bieten räumliche Indizes und Abfragefunktionen.
+ Wird von fast allen GIS Produkten und Middlewares als Datenspeicher unterstützt.	- Findet kaum Out-of-the-Box Unterstützung bei GIS Produkten oder Middlewares.
- Genormt nur für einfache Merkmale (“Simple Features”)	+ Beliebiges GML, auch mit komplizierten Merkmalen
- Schema Mismatch zwischen dem relationalen Datenmodell und dem XML Datenmodell macht Mapping notwendig.	+ GML Daten können als solche gespeichert werden, und müssen nicht in ein anderes Datenmodell konvertiert werden.
- Bei evolvierenden GML-Applikations-Schemas muss zusätzlich noch das relationale Schema angepasst werden.	+ Es gibt nur ein Schema

Tabelle 5: Vergleich: Relationale versus XML Datenbanken

- beim internen Umgang mit den Daten hat sich aber nichts geändert. Es ist bei solchen Systemen in der Folge auch nicht notwendig, die Geodaten im GML Format zu speichern, sondern es können die konventionellen Speichermethoden beibehalten werden - üblicherweise sind das relationale Geo-Datenbanken. Im Folgenden dieser Ansatz mit der Alternative verglichen werden, die GML Daten nicht zu konvertieren, sondern in speziell für XML und GML geeigneten Datenbanken zu speichern.

### 5.2.1. Vergleich: Relationale Datenbanken versus XML Datenbanken

Die oben genannte Vorgehensweise, GML als Import- und Exportformat zu unterstützen, aber intern eine bereits existierende Speichermethode (üblicherweise eine relationale Datenbank) weiterzuverwenden, hat Vor- und Nachteile. Die alternative Vorgehensweise ist, die GML Daten *als solche* in einer dafür geeigneten XML-Datenbank zu speichern. Tabelle 5 vergleicht die beiden Ansätze.

Systeme, die als interne Speichermethode eine **relationale Datenbank** verwenden, profitieren von den Vorteilen beider Seiten: Zur Speicherung werden die effizienten und ausgereiften relationalen Datenbanksysteme verwendet, und zum Austausch das für diesen Zweck sehr gut geeignete XML Format.

Dieser Ansatz birgt aber auch beträchtliche Nachteile: Während GML weiterentwickelt wurde und in der aktuellen Version mehr als nur “Simple Features” (siehe Abschnitt 3.1) unterstützt, ist das beim OGC Standard für SQL nicht der Fall. In der Folge können - auf standardisierte Weise - in relationalen Datenbanken nur einfache Merkmale gespeichert werden.

Ein weiterer Nachteil für Systeme, die GML Daten in relationalen Datenbanken ablegen, entsteht aus dem Faktum, dass die GML Daten einem Schema entsprechen, die relationalen Daten aber einem anderen Schema. Wenn eines der beiden Schemas angepasst werden muss, so auch das zweite

Schema, und zusätzlich noch das Mapping zwischen den beiden Schemas - das kann bei evolvierenden Datenschemas zu einem erheblichen Arbeitsaufwand führen.

Die wachsende Bedeutung von GML als Format für den Datenaustausch hat deshalb zu den bereits erwähnten Überlegungen geführt, GML Daten im GML Format zu speichern, anstatt jedes Mal von und zu GML konvertieren zu müssen. Diesem Ansatz entsprechend können ***XML Datenbanken*** (d.h.: XML-enabled Datenbanken oder native XML Datenbanken, mehr darüber in den folgenden Abschnitten 5.4 und 5.5 ) zur Speicherung von GML Daten verwendet werden.

Die Nachteile bei der Verwendung von XML Datenbanken betreffen in erster Linie die Neuartigkeit dieses Ansatzes. Damit zusammenhängend gibt es nur wenige XML Datenbanken mit Unterstützung für räumliche Daten, und die vorhandenen räumlichen XML Datenbanken werden nicht oder nur kaum von dritten Produkten unterstützt.

Trotz dieser Nachteile ist im Sinne einer schlanken Architektur eine XML Datenbank vorteilhaft, weil sie Geodaten direkt im GML Format verwalten kann (und die Daten nicht in ein anderes Format konvertieren muss). Prototypen solcher Systeme wurden gebaut [Corcoles und Gonzalez, 2002; Huang u. a., 2006] und werden in den folgenden Abschnitten beschrieben, aber ein umfassendes Benchmarking entsprechender Systeme ist dem Autor dieser Arbeit nicht bekannt.

Im Folgenden werden Ansätze beschrieben, wie GML Daten als solche, d.h. in einer XML-basierten Form, gespeichert und abgefragt werden können. Insbesondere ist es möglich, GML Daten in Dateien abzulegen (auch wenn das mit beträchtlichen Nachteilen verbunden ist), oder in XML-enabled oder in nativen XML Datenbanken zu speichern.

### 5.3. Speicherung von GML in Dateien

GML Daten können (so wie allgemein auch XML Daten) im Dateisystem in Dateien gespeichert werden. Von Vorteil ist die Gewöhnlichkeit und Einfachheit dieses Ansatzes. Der Ansatz hat jedoch auch etliche Nachteile:

*Schwierig abzufragen:* GML Daten, die als einfache Dateien im Dateisystem liegen, bieten keine Unterstützung bei der Abfrage: Das Parsen der XML Daten sowie das Filtern nach räumlichen und anderen Kriterien muß erst implementiert werden. Außerdem bieten GML Daten in einfachen Dateien keine Indizes, um die Auswertung von Abfragen zu beschleunigen.

*Probleme beim gleichzeitigen Zugriff:* Dateien im Dateisystem unterstützen nur sehr schlecht den gleichzeitigen Zugriff durch mehrere Prozesse: Die Sperren bei Schreibzugriff sind nur auf Datei-Ebene möglich (statt z.B. auf der Ebene einzelner Features eines GML Dokuments). Eine klare und fein strukturierte Transaktionssemantik fehlt.

Trotz dieser Nachteile wird dieser Ansatz von den Autoren C.H. Huang et.al [Huang u. a., 2006] bei der Erstellung eines Prototypen für ein webbasiertes GIS gewählt. Allerdings liegt der Fokus dieser Arbeit



primär auf der Implementierung effizienter Parser für GML und nicht auf dem Finden effizienter Speichermethoden.

Für das vorliegende Projekt “Verkehrstelematik Datenbank” kommt nach Meinung des Autors der Ansatz, GML Daten in Dateien zu speichern, nicht in Frage, weil die Aspekte der effizienten Abfrage und des gleichzeitigen Zugriffs zu wichtig sind.

#### 5.4. Speicherung von GML in XML-enabled Datenbanken

XML-enabled Datenbanken dienen der Speicherung von XML Daten, und können daher auch zur Speicherung von GML Daten verwendet werden. XML-enabled Datenbanken übertragen die zu speichernden Daten in ein anderes, intern verwendetes Datenmodell und speichern die Daten in einer für dieses Datenmodell geeigneten Datenbank. Obwohl die Daten in einem anderen Datenmodell gespeichert sind, lassen sie sich wieder als XML Daten abfragen. Um das zu ermöglichen, übersetzen XML-enabled Datenbanken XML-Abfragen zu Abfragen über dem intern verwendeten Datenmodell. Das Ergebnis dieser Abfragen wird zurück ins XML Datenmodell übersetzt und dann erst zurückgegeben.

Eines der weitest verbreiteten Datenmodelle ist das relationale Datenmodell. Etliche relationale Datenbanksysteme (RDBS) wurden aufgrund ihrer vorteilhaften Eigenschaften [Zhu u. a., 2006, Kap. 1][Li u. a., 2004] (Transaktionen, Sicherheit, Crash Recovery, Optimierte Anfrageauswertung, räumliche Indizierung und räumliche Abfragemöglichkeiten) weiterverwendet und zu XML-enabled Datenbanken erweitert. Auch andere als relationale Datenbanksysteme (zum Beispiel objekt-orientierte Datenbanken) lassen sich zu einer XML-enabled Datenbank erweitern. Es soll jedoch wegen der Popularität relationaler Datenbanken im Folgenden der Ausdruck ‘XML-enabled Datenbank’ für eine relationale Datenbank stehen, die zum Speichern und Abfragen von XML Daten erweitert wurde.

Anfragen an eine XML-enabled Datenbank müssen zum intern verwendeten, relationalen Datenmodell hin übersetzt werden. Weil jedoch das Datenmodell von XML anders ist als das relationale Datenmodell, kann es Schwierigkeiten bei der Übersetzung der Anfrage geben. In der englischen Literatur wird das zugrundeliegende Problem *Schema Mismatch* genannt. Ein Symptom des Schema Mismatch Problems ist die Schwierigkeit der Übersetzung von rekursiven XPath oder XQuery Anfragen (mit der XPath ‘descendant’-Achse // ) zu einer Anfrage in SQL [Krishnamurthy u. a., 2003]. Diese Schwierigkeit tritt auf, weil sich rekursive Anfragen in XPath und XQuery leicht formulieren lassen, nicht aber in SQL.

Die Forschung an XML-enabled Datenbanken gliedert sich in verschiedene Ansätze, die in den folgenden Abschnitten unterschieden und beschrieben werden sollen.

##### 5.4.1. XML Publishing versus XML Storage Ansätze

Die Autoren R. Krishnamurthy et.al geben in [Krishnamurthy u. a., 2003] einen hervorragenden Überblick über Forschung an XML-enabled Datenbanken: Die Forschungsarbeiten lassen sich zwei



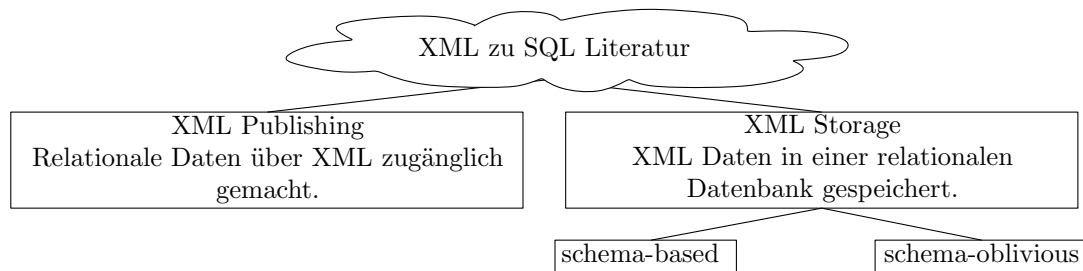


Abbildung 17: Einteilung der Forschungsliteratur über XML-enabled Datenbanken: Bei XML Publishing geht es um das Abfragen relationaler Daten über eine XML-basierte Schnittstelle. Bei XML Storage um das Speichern und Abfragen von XML-Daten, die jedoch in einer relationalen Datenbank gespeichert sind. Aus [Krishnamurthy u. a., 2003]

XML Publishing	XML Storage
<p>Macht relationale Daten über XML zugänglich.</p> <ul style="list-style-type: none"> <li>• Definition einer XML Ansicht über den relationalen Daten.</li> <li>• Auswertung von XML Abfragen liefert als Ergebnis eine modifizierte XML Ansicht.</li> </ul>	<p>Verwendet eine relationale Datenbank, um XML Daten abzuspeichern.</p> <ul style="list-style-type: none"> <li>• Wahl eines geeigneten relationalen Schemas</li> <li>• Import der XML Daten ins relationale Schema</li> <li>• Übersetzung von XML Abfragen zu SQL. Rekonstruktion eines XML Dokuments aus den Ergebnissen der SQL Abfrage.</li> </ul> <p>Siehe Tabelle 7 für einen Vergleich der beiden XML-Storage Ansätze, Schema-based und Schema-oblivious Storage.</p>

Tabelle 6: XML-enabled Datenbanken: XML Publishing und XML Storage im Vergleich [Krishnamurthy u. a., 2003].

verschiedenen Aufgabenstellungen zuordnen, *XML Publishing* und *XML Storage*. Abbildung 17 zeigt die beiden Ansätze im Überblick, Tabelle 6 vergleicht sie im Detail. Bei XML Publishing sollen im relationalen Modell vorliegende Daten über XML zur Verfügung gestellt werden. Bei XML Storage hingegen soll zur Speicherung und Abfrage von im XML Format vorliegenden Daten eine relationale Datenbank zu verwendet werden. Dazu müssen die XML Daten ins relationale Datenmodell hinein abgebildet werden (englisch: XML-to-relational mapping). Diese Abbildung kann auf das Datenschema der jeweiligen XML Daten Rücksicht nehmen, dann wird sie als *schema-based* bezeichnet. Andernfalls, wenn die Abbildung ohne Rücksicht auf ein spezifisches XML Datenschema beliebige XML Daten ins relationale Datenmodell hinein abbildet, wird sie als *schema-oblivious* bezeichnet. Für die Frage, wann welcher der beiden Ansätze (schema-based oder schema-oblivious) besser geeignet ist, ist die im folgenden Abschnitt erläuterte Unterscheidung wichtig.

### 5.4.2. Data-centric versus Document-centric Anwendungen

Eines der wichtigsten Kriterien bei der Auswahl einer XML-enabled Datenbank ist, ob Datensätze oder Dokumente gespeichert werden sollen - der IT-Berater R. Bourret erklärt diesen Unterschied auf seiner Website [RpBourret:XMLAndDatabases, 2007]. Davon abhängig sind schema-based oder schema-oblivious Ansätze für die gegebene Anwendung besser geeignet.

*Data-centric:* Wenn XML zur Speicherung oder zum Transport von *Datensätzen* verwendet wird, dann entsprechen die XML Daten meist einem genau definierten und wohl bekannten Schema, und die Struktur der Daten ist typischerweise sehr regelmäßig und repetitiv. Häufig werden solche Daten zwischen Programmen oder Maschinen ausgetauscht. Zum Speichern von Datensätzen werden häufig XML-enabled Datenbanken mit schema-based Mapping verwendet, weil das relationale Datenmodell für Daten dieser Art gut geeignet ist.

*Document-centric:* Gänzlich anders sind in XML gespeicherte *Dokumente*: Deren Struktur ist meist sehr unregelmäßig und enthält viel gemischten Inhalt (das ist Fließtext mit eingebetteten XML Tags), und fast immer kommt es auf die Reihenfolge der Elemente an. Zum Speichern von Dokumenten sind native XML Datenbanken oder XML-enabled Datenbanken mit schema-oblivious Mapping ins relationale Datenmodell besser geeignet, weil diese Datenbanken mit der unregelmäßigen Struktur der Dokumente gut umgehen können.

GML Daten können je nach Anwendung einer der beiden Kategorien (data-centric oder document-centric) zugehören. Zum Beispiel ließe sich in GML ein Roman schreiben, dessen Ortsnamen jeweils mit einer geographischen Angabe versehen ist - das wäre ein extremes Beispiel für eine document-centric Anwendung. Häufig aber sind GML Daten "data-centric" und bestehen aus langen Listen mit Datensätzen geographischer Merkmale. Auch die verkehrsbezogenen Daten, die im vorliegenden Projekt verwaltet werden sollen, werden von dieser Art sein.

### 5.4.3. Schema-based versus Schema-oblivious Ansätze

Die Unterscheidung von schema-based und schema-oblivious XML-enabled Datenbanken betrifft das intern verwendete Datenschema: Wenn das intern verwendete (relationale) Schema vom jeweiligen XML Schema der zu speichernden Daten abgeleitet wird, spricht man von schema-based Storage. Wenn hingegen das intern verwendete, relationale Datenmodell ein generisches ist, das für XML Daten mit beliebigem Schema geeignet ist, spricht man von schema-oblivious Storage. Tabelle 7 zeigt die Eigenschaften beider Ansätze im Vergleich.

Wie bereits im vorigen Abschnitt 5.4.2 geschildert, ist der schema-based Ansatz für data-centric Anwendungen besser geeignet, während der schema-oblivious Ansatz für document-centric Anwendungen der bessere ist. Diese Behauptung wird auch von vergleichenden Benchmarks gestützt, die im Folgenden vorgestellt werden sollen:

Schema-oblivious	Schema-based
Verwendet ein generisches relationales Schema zur Speicherung beliebiger XML Dokumente.	Verwendet das XML Datenschema, um davon ein spezifisches, relationales Schema abzuleiten.
<p>Äquivalente Bezeichnungen:</p> <ul style="list-style-type: none"> <li>• Generische Speicherung</li> <li>• Document-independent</li> <li>• Model Mapping</li> </ul>	<p>Äquivalente Bezeichnungen:</p> <ul style="list-style-type: none"> <li>• Strukturierte Speicherung [Brinkhoff, 2005]</li> <li>• Document-dependent [Lu u. a., 2005]</li> <li>• Structure Mapping [Corcoles und Gonzalez, 2002]</li> </ul>
<p>Vorteile:</p> <ul style="list-style-type: none"> <li>• Für <i>document-centric</i> Anwendungen: Gut geeignet für unregelmäßig strukturierte Daten, z.B. XHTML Seiten.</li> <li>• Effizient bei Nutzung von Indizes.</li> <li>• Kann Dokumente beliebigen Schemas speichern.</li> </ul>	<p>Vorteile:</p> <ul style="list-style-type: none"> <li>• Für <i>data-centric</i> Anwendungen: Gut geeignet für regelmäßig strukturierte Daten mit striktem Schema.</li> <li>• Effizient bei passender Wahl des relationalen Schemas und bei Berücksichtigung des XML Schemas zur Optimierung der SQL Abfragen.</li> </ul>
<p>Ansätze (Möglichkeiten bei der Wahl des relationalen Schemas):</p> <ol style="list-style-type: none"> <li>1. <i>Id-basiert</i>: Jedes XML Element bekommt eine eindeutige Id. Ein Fremdschlüssel verweist auf das Eltern-Element.</li> <li>2. <i>Intervall-basiert</i>: Jedem Element entspricht ein Intervall, in dem all seine Kind-Elemente (und deren Kinder) liegen.</li> <li>3. <i>Pfad-basiert</i>: Zu jedem Element wird der Pfad ausgehend von der Wurzel gespeichert.</li> </ol>	<p>Ansätze:</p> <ol style="list-style-type: none"> <li>1. Unter Verwendung vorhandener <i>XML-Publishing Techniken</i>: Erst werden die XML Daten in die relationale Datenbank importiert. Dann werden XML-Publishing Techniken verwendet, um die relational gespeicherten Daten abzufragen. Nachteil: Aufgrund der Natur von XML Publishing Techniken kann bei der Abfrage-Übersetzung (von einer Abfrage über XML Daten zu einer SQL Abfrage über den relationalen Daten) das ursprüngliche XML Schema nicht mehr verwendet werden. Viele der so generierten SQL Abfragen sind ineffizient [Krishnamurthy u. a., 2004].</li> <li>2. <i>Spezielle schema-based Technik</i>: Wenn bei der Abfrage-Übersetzung das ursprüngliche XML Datenschema verwendet wird, um die SQL Abfrage zu optimieren (siehe Abschnitt 5.4.5).</li> </ol>

Tabelle 7: XML-enabled Datenbanken: Schema-based versus Schema-oblivious Speicherung. Vergleiche mit [Krishnamurthy u. a., 2003]

In einer letztes Jahr (2007) erschienenen Arbeit vergleichen die Autoren C.Lu, R.Dos Santos et.al [Lu u. a., 2007] Produkte beider Ansätze (d.h. schema-based und schema-oblivious Speicherung von XML Daten in RDBSs) in einem umfassenden Benchmark, sie gehen dabei jedoch nicht speziell auf geographische Daten ein. Im Ergebnis schneidet der schema-based Ansatz besser ab.

In einer älteren Arbeit von J.Córcoles und P. González [Corcoles und Gonzalez, 2002] geht es speziell um geographische Daten im GML Format. Die Autoren vergleichen drei Arten, GML in einer relationalen Datenbank zu speichern und mit einer eigenen, um räumliche Operatoren erweiterten Sprache [Corcoles und Gonzalez, 2001] abzufragen: LegoDB [Bohannon u. a., 2002] (schema-based), MonetDB [Schmidt u. a., 2001; MonetDB:Website, 2007] (schema-oblivious) und XParent [Jiang u. a., 2002] (schema-oblivious). Auch hier schneidet im Ergebnis der Arbeit der schema-based Ansatz besser ab.

*Bedeutung für das vorliegende Projekt:* Nach Meinung des Autors ist bei der Verwendung von XML-enabled Datenbanken der schema-based Ansatz für die im vorliegenden Projekt zu verwaltenden GML Daten besser geeignet, weil diese Daten typischerweise ‘data-centric’ sind, d.h. es sind maschinenlesbare Daten mit regelmäßiger Struktur, wo es auf die Reihenfolge der Elemente häufig nicht ankommt. Der schema-based Ansatz kann diese Eigenschaften ausnützen und die Daten somit in geeigneter Weise ins relationale Datenmodell übertragen.

#### 5.4.4. Schema-based Storage: Intelligente Wahl des relationalen Schemas

Bei allen Ansätzen zum Speichern von XML Daten in XML-enabled Datenbanken (sowohl schema-oblivious wie schema-based) spielt die Wahl des relationalen Schemas eine zentrale Rolle. Bei schema-oblivious Ansätzen ist jedoch das verwendete relationale Modell generisch, und damit invariant gegenüber den XML Schemas der zu speichernden Daten. Daher ist beim schema-oblivious Ansatz die Wahl eines passenden relationalen Modells eine Entscheidung der Datenbank-Entwickler, und nicht der Datenbank-Benutzer. Beim schema-based Ansatz muß das relationale Datenmodell aus dem XML Datenmodell der zu speichernden Daten abgeleitet werden. Dieser Vorgang wird auch als *Mapping* oder *Shredding* bezeichnet. Das kann entweder manuell durch den Benutzer geschehen, der in dem Fall das Mapping vom einen Datenmodell ins andere spezifizieren muß, oder aber automatisch oder halbautomatisch mit Techniken, die das XML Datenschema zu einem relationalen Schema übersetzen. Automatische oder halbautomatische Techniken zur Wahl eines passenden relationalen Schemas, die auch die zu erwartende Abfragelast und statistische Eigenschaften der zu speichernden Daten mit einbeziehen, heißen *adaptive Shredding* Techniken. Die Autoren Bohannon, Freire et.al [Freire und Simeon, 2003; Bohannon u. a., 2002] haben eine solche adaptive Shredding Technik für die Datenbank LegoDB [LegoDB:WebSite, 2008] implementiert.

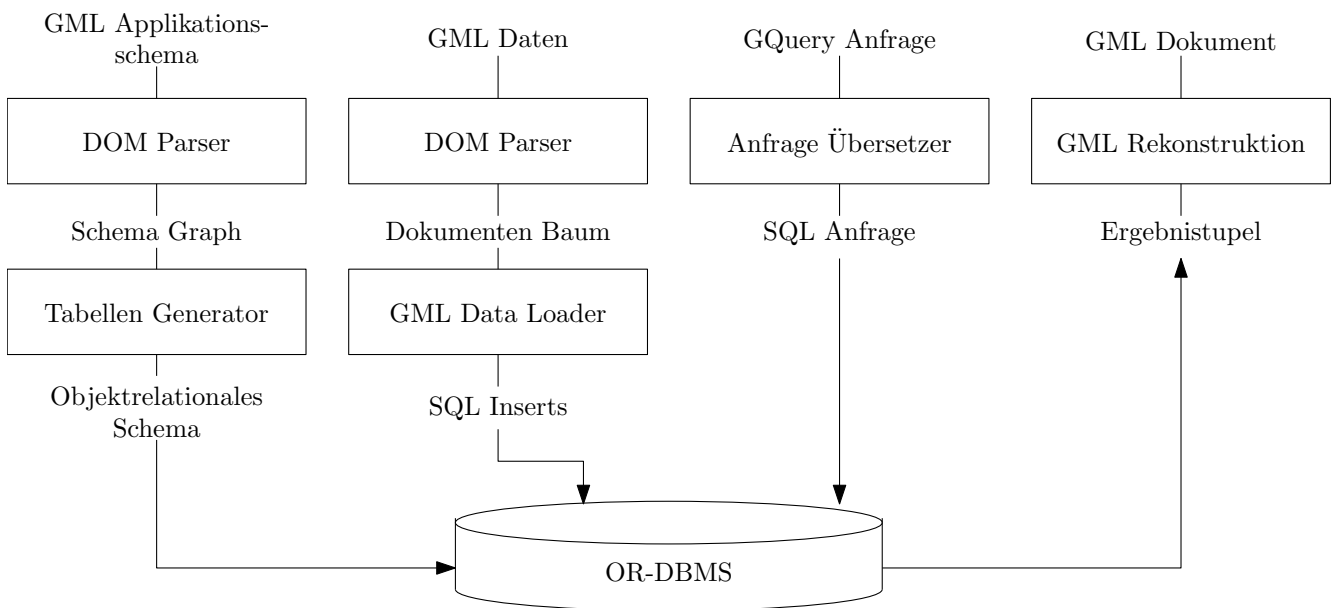


Abbildung 18: Architektur des Prototypen von Zhu, Guan, Zhou et.al [Zhu u. a., 2006]

#### 5.4.5. Schema-based Storage: Effiziente XML-zu-SQL Abfrageübersetzung

Neben der Wahl eines passenden relationalen Schemas ist beim schema-based Storage Ansatz die effiziente XML-zu-SQL Abfrage-Übersetzung ein offenes Problem. Den Autoren Krishnamurthy et.al [Krishnamurthy u. a., 2003] war im Jahr 2003 kein Algorithmus bekannt, der zur Optimierung der Abfrage-Übersetzung Information aus dem ursprünglichen XML Schema verwendet. Erst letztes Jahr (2007) haben sie einen entsprechenden Prototypen gebaut [Krishnamurthy u. a., 2004]. Es sind daher bei vielen Produkten die generierten SQL Abfragen ineffizient und unnötig komplex, wenn bei der XML-zu-SQL Abfrage-Übersetzung das ursprüngliche XML Schema nicht verwendet wird.

#### 5.4.6. Schema-based Storage: Beispiel-Implementierung für GML Daten

Von den Autoren Zhu, Guan, Zhou et.al [Zhu u. a., 2006] wurde ein Prototyp eines schema-based Systems gebaut, das GML Daten in einer relationalen Datenbank speichert. Abbildung 18 zeigt die Architektur des Systems. Eine ähnliche Implementierung wird auch von Li, Zhou et.al [Li u. a., 2004] beschrieben.

Das GML Applikationsschema, dem die Daten entsprechen, wird mit einem DOM Parser [W3C, a] zu einem Schema Graphen umgewandelt. Ein Algorithmus erzeugt aus dem Schema Graphen eine Menge an Relationen für die Oracle Spatial Datenbank [Oracle: SpatialDB, 2007]. Der nächste Schritt ist das Importieren der GML Daten ins relationale Datenschema: Dazu werden die GML Daten von einem DOM Parser eingelesen. Ein Algorithmus erzeugt aus jedem Knoten im DOM Baum SQL Anweisungen, welche die entsprechenden Daten in die Datenbank schreiben. Zur Abfrage der in der Datenbank gespeicherten Daten verwenden die Autoren GQuery [Boucelma und Colonna, 2004], eine Erweiterung

von XQuery um räumliche Operationen (mehr dazu in Abschnitt 4.3). Die GQuery-Abfragen werden zu SQL Abfragen übersetzt. Die SQL Abfragen werden vom relationalen Datenbanksystem ausgeführt, und aus dem Ergebnis der Abfrage wird ein GML Dokument rekonstruiert.

#### 5.4.7. Schema-based Storage: Probleme mit der Komplexität des GML Standards

Beim Speichern von GML Daten in relationalen Datenbanken macht die Komplexität von GML in der aktuellen Version 3 Probleme. Das ist bei schema-based XML-enabled Datenbanken der Fall (weil die GML Daten in der Datenbank zum relationalen Modell konvertiert werden), als auch bei Middlewares mit relationaler Datenbank (weil die GML Daten in der Middleware zum relationalen Modell konvertiert werden). In beiden Fällen ist die Ursache der Problematik, dass das Applikationsschema der GML Daten in ein relationales Schema übersetzt werden muss. Es wird also ein mitunter sehr komplexes GML Applikationsschema in ein relationales Schema übersetzt, das in der Folge ebenfalls sehr kompliziert ist - unter Umständen so kompliziert, dass der Umgang damit zu schwierig wird.

Manche Arbeiten und Produkte umgehen das Problem, indem sie sich auf einfache XML Schemas beschränken, und verwenden GML in der alten Version 2: Die Autoren Zhu, Guan et.al [Zhu u. a., 2006] räumen ein, dass ihre Methode des schema-based Speicherns von GML nur mit einfachen GML Dokumenten in der GML Version 2 funktioniert. Ähnlich verhält es sich auch mit dem kommerziellen Produkt GO Loader [Snowflakesoftware:GoLoader, 2008], welches GML Daten in die Oracle Spatial Database [Oracle:SpatialDB, 2007] importiert - es kann auch nur mit GML Daten in der Version 2 umgehen.

In einer anderen Arbeit [Huang u. a., 2006, Kap. 2] haben sich die Autoren Huang et.al von vornherein gegen die Verwendung relationaler Datenbanken entschieden, weil die relationalen Schemas ihrer Anwendung zu kompliziert waren, um nützlich zu sein. Wie komplex ein relationales Datenschema einer größeren Anwendung werden kann, wird beispielsweise aus dem INTREST Projekt (siehe Abbildung 2 in Abschnitt 2.2) ersichtlich.

Ein weiteres Hindernis beim Speichern von GML Daten in relationalen Datenbanken ist, dass in GML Version 3 kompliziertere geographische Merkmale modelliert werden können als in SQL bei Verwendung des Simple Features for SQL Standards (siehe Abschnitt 3.1 dieser Arbeit).

Zusammenfassend läßt sich sagen, dass die Verwendung von XML-enabled Datenbanken (insbesondere bei Verwendung des schema-based Ansatzes) eine gute Möglichkeit ist, um GML Daten zu speichern. Schwierigkeiten machen jedoch Unterschiede zwischen dem relationalen Datenmodell und dem von XML. Diese Schwierigkeiten fallen bei Verwendung sogenannter nativer XML-Datenbanken weg. Dieser Ansatz soll im folgenden Abschnitt beschrieben werden.

### 5.5. Speicherung von GML in nativen XML Datenbanken

GML Daten können in nativen XML Datenbanken gespeichert werden. Native XML-DBs sind Datenbanksysteme, die auf die Speicherung von XML Dokumenten spezialisiert sind und werden von *XML-enabled Datenbanken* unterschieden. Die XML:DB Initiative [Initiative] beschreibt eine native XML Datenbank wie folgt:

- Eine native XML Datenbank definiert ein logisches Datenmodell (logisch im Gegensatz zu physisch) für das Speichern und Abfragen von XML Dokumenten. Dieses Modell muß zumindest XML Elemente, Attribute, PCDATA und die Reihenfolge von Elementen berücksichtigen.
- Die logische Speichereinheit einer nativen XML Datenbank sind XML Dokumente (ähnlich wie ein Tupel in einer Relation die logische Speichereinheit einer relationalen Datenbank ist).
- Eine native XML Datenbank kann eine beliebige Art der physischen Speicherung verwenden. Das inkludiert (objekt-) relationale, objekt-orientierte und hierarchische Datenbanken, Dateien im Dateisystem sowie beliebige andere Methoden.
- Der Zugriff auf eine native XML Datenbank erfolgt (im Gegensatz zu XML-enabled Datenbanken) nur über XML und darauf bezogene Technologien. Wenn ein Datenbanksystem zusätzlich noch eine zweite Schnittstelle direkt zur physischen Art der Speicherung zur Verfügung stellt (z.B. SQL bei Verwendung einer relationalen Datenbank), so spricht man von einer XML-enabled Datenbank.

Typischerweise bieten native XML Datenbanken standardisierte Zugriffsmethoden auf die von ihnen verwalteten XML Daten: DOM (Document Object Model) [W3C, a] und SAX (Simple API for XML) [Megginson] als APIs für das Parsen von XML, XSLT [W3C, j] für die Transformation von XML Daten zu anderen Formaten, sowie XPath [W3C, h] und XQuery [W3C, i] als Abfragesprachen.

So wie auch bei der Speicherung von XML Daten in relationalen Datenbanken (siehe Abschnitt 5.4) gibt es auch bei nativen XML Datenbanken schema-based und schema-oblivious Ansätze. Die meisten nativen XML-DBs unterstützen das Speichern von XML Daten, deren Schema unbekannt ist. Die Verwendung von Schemas kann jedoch Abfragen beschleunigen [Vakali u. a., 2005]. Für einen Vergleich von schema-based und schema-oblivious Ansätzen siehe Tabelle 7 bzw. Abschnitt 5.4.3.

Ein Problem bei vielen nativen XML Datenbankprodukten ist die mangelhafte Unterstützung für räumliche Daten: Für die Abfrage nach räumlichen und topologischen Kriterien müssen die Abfragesprachen für räumliche Abfragen erweitert werden (siehe Abschnitt 4.3). Weiters muss für die effiziente Auswertung solcher räumlicher Abfragen die Datenbank um räumliche Indexstrukturen erweitert werden (siehe Abschnitt 5.5.1).



### 5.5.1. Indizierung von GML Daten

Zur effizienten Abarbeitung von Abfragen verwalten Datenbanksysteme Indizes über den von ihnen gespeicherten Daten. Dabei können zur Indizierung von GML Daten generische Methoden für XML Daten verwendet werden. Zusätzlich müssen GML Daten nach räumlichen Kriterien indiziert werden, um Abfragen mit räumlichen Kriterien effizient zu beantworten. Es können somit zur Indizierung von GML Daten drei Methoden unterschieden werden können [Lu u. a., 2007, S.142]:

*Strukturelle Indizes:* Um Abfragen zu beschleunigen, die nach einer bestimmten Struktur in einem XML Dokument suchen, können Indizes über Pfade zu den Elementknoten oder Indizes über Eltern-Kind-Beziehungen angelegt werden.

*Wertbasierte Indizes:* Um Abfragen zu beschleunigen, die nach bestimmten Element-Inhalten oder nach bestimmten Attributwerten suchen.

*Räumliche Indizes:* Um räumliche Abfragen (nach der räumliche Position eines Features oder topologisch nach der relativen Lage zweier Features zueinander) zu beschleunigen, können die gleichen räumlichen Indextechniken angewendet werden, wie sie bereits für relationale Datenbanksysteme implementiert wurden.

Während das Implementieren struktureller und wertbasierter Indizes ein wichtiges Anliegen aller XML Datenbanken ist, gibt es nur wenige Produkte mit räumlicher Indizierung.

Die kommerzielle XML Datenbank TigerLogic XDMS [RainingData:TigerLogicXDBMS, 2007] hat jedoch diese Fähigkeit: Sie läßt sich über um räumliche Funktionen erweitertes XQuery abfragen. Zur Unterstützung dieser räumlichen Abfragen können räumliche Indizes über den GML Daten angelegt werden.

Für die open-source XML Datenbank eXist [eXist:Website, 2007] wurde letztes Jahr (2007) eine neue, modulare Architektur für Indizes fertiggestellt, und ein Prototyp eines räumlichen Indexing-Moduls fertiggestellt.

## 5.6. Vergleich: Native versus XML-enabled Datenbanken zum Speichern von GML

Die Effizienz und Skalierbarkeit von nativen XML Datenbanken im Vergleich zu XML-enabled Datenbanken mit einer relationalen Datenbank im Hintergrund ist ein umstrittenes Thema: Vakali et.al [Vakali u. a., 2005] nennen als Vorteile nativer XML Datenbanken Skalierbarkeit, Zugriffsgeschwindigkeit und Zuverlässigkeit. Jedoch in der Arbeit von Zhu, Guan et.al [Zhu u. a., 2006] sind ähnliche Argumente Grund für die Verwendung von relationalen XML-enabled Datenbanken. Die Autoren Hongjun Lu et.al [Lu u. a., 2005] haben einen ausführlichen Vergleich von acht verschiedenen XML-DBs durchgeführt:



*Ansatz 1:* Relationale Datenbank XML-enabled mit einem schema-dependent Mapping. Dieser Ansatz lieferte die besten Ergebnisse.

*Ansatz 2:* Relationale Datenbank XML-enabled mit einem schema-oblivious Mapping. Der Ansatz lieferte ähnliche Ergebnisse, waren aber allesamt schlechter als native XML-DBs.

*Ansatz 3:* Native XML-DBs können nur dann gute Ergebnisse liefern wenn sie alle möglichen Indizes voll ausnutzen.

Ein ähnlich umfassendes Benchmarking speziell für GML Daten ist dem Autor dieser Arbeit nicht bekannt. Im Folgenden sollen die drei genannten Ansätze in Bezug auf das vorliegende Projekt verglichen werden:

Der oben genannte Ansatz 1 hat den Vorteil, dass zur Speicherung der Geodaten die vielen positiven Eigenschaften der mittlerweile sehr ausgereift relationalen Datenbankprodukte zum Tragen kommen. Schema-dependent Mapping für Geodaten wurde bereits mehrfach in der Forschung implementiert (siehe Abschnitt 5.4.6). Ein Nachteil des Ansatzes ist, dass für relationale Datenbanken kein Standard für den Umgang mit nicht-”einfachen” geographischen Merkmalen (z.B. mit Koordinaten-Angaben in drei Dimensionen) existiert (siehe Abschnitt 5.4.7).

Für das vorliegende Projekt kommt Ansatz 2 sicher nicht in Frage. Erstens schon wegen der schlechteren Leistung, weiters aber auch weil der schema-oblivious Ansatz gut für document-centric, nicht aber für data-centric Anwendungen geeignet ist (siehe Abschnitt 5.4.2).

Wie sehr Ansatz 3 für das vorliegende Projekt sinnvoll ist, hängt in erster Linie von der Entwicklung nativer Geodatenbanken ab. Wichtig ist die Implementierung von Indizes für verschiedene Datentypen (textuelle, numerische, und räumliche Angaben) und von geographischen Abfrageoperatoren (siehe Abschnitte 4.3 und 5.5.1).

## 6. Zusammenfassung und Ergebnisse

GML ist als Sprache für die Modellierung und den Austausch geographischer Information nützlich. Trotz existierender Prototypen und Produkte ist der Aspekt effizienter Speicherung und Abfrage von GML Daten derzeit (2008) jedoch noch eine Herausforderung.

Für das Speichern von GML können relationale Datenbanken verwendet werden, wenn sie Erweiterungen für den Umgang mit geographischen Daten, und Erweiterungen für das Speichern von XML Daten haben. Ein Nachteil dieses Ansatzes ist, dass nicht alle von GML unterstützten, komplizierteren geographischen Merkmale auch im OGC Standard für relationale Datenbanken vorgesehen sind. Ein weiterer Nachteil bei der Verwendung relationaler Datenbanken ist, dass beim Importieren und beim Abfragen der GML Daten zwischen dem XML Datenschema und dem relationalen Schema konvertiert werden muss. Bei geographischen Daten eignet sich für dieses Konvertieren ein schema-based Mapping besser als ein generisches, schema-oblivious Mapping.

Dieser Nachteile wegen ist das Speichern von GML Daten in nativen XML Datenbanken eine interessante Alternative. Derzeit haben jedoch nur wenige native XML Datenbanken Erweiterungen für geographische Daten - insbesondere wären räumliche Indizes zur Beschleunigung geographischer Abfragen und räumliche Funktionen in der Abfragesprachen notwendig. Weil es nur wenige native XML Datenbanken mit solchen Erweiterungen gibt, haben auch die meisten räumlichen Middlewares keine Unterstützung für derartige Datenbanksysteme. Trotz des Mangels an fertigen Produkten dieser Art ist die Verwendung nativer XML Datenbanken eine interessante Möglichkeit zur Realisierung GML basierter GIS Systeme.

Für das Projekt der Erstellung einer österreichischen Verkehrsdatenbank wird eine Architektur mit einer räumlichen XML Datenbank vorgeschlagen, sodass die über GML ausgetauschten Geodaten gleich auch in dem selben Format gespeichert werden. Aus Effizienzgründen soll eine thin Middleware zumindest einfache räumliche Abfragen im Sinn eines 'Pre-Filtering' an die Datenbank weiterreichen.

## A. Überblick über ausgewählte Literatur

Im Folgenden soll versucht werden, dem Leser dieser Arbeit anhand einiger weniger ausgewählter Artikel einen tieferen Einblick in den Themenkomplex von Geodatenbanken und den Umgang mit GML Daten zu gewähren:

- [Brinkhoff, 2005] Geodatenbanken: Dieses Buch gibt eine Einführung in die Verwendung relationaler Geodatenbanken, speziell im Hinblick auf die Verwendung der Oracle Spatial Datenbank.
- [Lu u. a., 2007] GML: Ein Überblick über den Stand der Forschung im Umgang mit GML Daten: Speichermethoden, Parser, Abfragesprachen, Visualisierung, GML für mobile Anwendungen, GML in WebServices.
- [Lu u. a., 2005] XML storage: Ein vergleichender Leistungstest verschiedener XML Datenbanken (native und XML-enabled).
- [Krishnamurthy u. a., 2003] XML storage: Ein Überblick über verschiedene Ansätze zur Speicherung von XML Daten in relationalen Datenbanken: XML Publishing und XML Storage, schema-based und schema-oblivious Mapping.
- [An und Zhao, 2006] Integration heterogener Geodaten: Diese Arbeit gibt einen Überblick über Möglichkeiten zur Integration von Geodaten aus heterogenen Datenquellen. Vorhandene Integrationsansätze werden beschrieben, und eine Mediator-basierte Architektur unter Verwendung von GML wird vorgeschlagen.
- [Wagner, 2008] GIS Produkte: Die Freegis.org Website bietet Zugriff auf eine umfassende Datenbank freier GIS Produkte (u.a. Web GIS, Desktop GIS und Datenbanken).

## B. Abbildungsverzeichnis

1.	INTREST Datenmodell mit Kern wichtiger Daten . . . . .	9
2.	Komplexität des relationalen Datenmodells von INTREST . . . . .	11
3.	INSPIRE Informationsfluss . . . . .	12
4.	INSIRE und Deutschlands Geo-Infrastruktur . . . . .	14
5.	Technische Architektur von INSPIRE . . . . .	15
6.	Modellierung des Verkehrsnetzes in EuroRoadS . . . . .	16
7.	Manöver in EuroRoadS . . . . .	17
8.	Akteure und Use-Cases in EuroRoadS . . . . .	18
9.	Das Roncalli Intelligent Speed Adaption Service . . . . .	19
10.	Roncalli Clearing Stelle . . . . .	19
11.	Roncalli Clearing Stelle: Technisches Konzept . . . . .	20
12.	Features und Feature Types . . . . .	23
13.	Architekturen für die Verkehrstelematik-Datenbank . . . . .	26
14.	GML Objekthierarchie . . . . .	32
15.	Architektur eines Mediator-Systems . . . . .	47
16.	Architektur des VirGIS Mediator-Systems . . . . .	47
17.	Einteilung der Forschungsliteratur über XML-enabled Datenbanken . . . . .	56
18.	Architektur des Prototypen in [Zhu u. a., 2006] . . . . .	60

## C. Tabellenverzeichnis

1.	Vergleich: Thin versus thick Middlewares . . . . .	27
2.	Einige räumliche Middlewares . . . . .	30
3.	Vergleich: XQuery versus Filter Encoding . . . . .	37
4.	Speichermöglichkeiten für GML Daten . . . . .	52
5.	Vergleich: Relationale versus XML Datenbanken . . . . .	53
6.	XML Publishing und XML Storage im Vergleich . . . . .	56
7.	Schema-based versus Schema-oblivious Speicherung . . . . .	58

## D. Literaturverzeichnis

- [EuroRoadS:InformationModel] : *EuroRoadS Road Network Information Model*. – URL <http://www.euroroads.org/php/Reports/D6.3%20Road%20Network%20Information%20Model.pdf>. – Zugriffsdatum: 08/12/2007
- [EuroRoadS:CoreEuropeanRoadData] : *Final Specification of Core European Road Data*. – URL <http://www.euroroads.org/php/Reports/D6.5%20Final%20specifikation%20of%20core%20European%20road%20data.pdf>. – Zugriffsdatum: 08/12/2007
- [EuroRoadS:ExchangeFormat] : *Final Specification of Road Network Exchange Format*. – URL <http://www.euroroads.org/php/Reports/D6.11%20Road%20Network%20Exchange%20Format.pdf>. – Zugriffsdatum: 11/01/2008
- [EuroRoadS:ExchangeModel] : *Final Specification of Road Network Exchange Model*. – URL <http://www.euroroads.org/php/Reports/D6.10%20Road%20Network%20Exchange%20Model.pdf>. – Zugriffsdatum: 11/01/2008
- [INTREST:Info5] : *INTREST als kommerzieller Betrieb*. – URL [http://intrest.org/downloads/Intrest\\_Info\\_5.PDF](http://intrest.org/downloads/Intrest_Info_5.PDF). – Zugriffsdatum: 19/11/2007
- [INTREST:Info2] : *Das INTREST-Objektmodell*. – URL [http://intrest.org/downloads/Intrest\\_Info\\_2.PDF](http://intrest.org/downloads/Intrest_Info_2.PDF). – Zugriffsdatum: 19/11/2007
- [INTREST:Info1] : *Was unterscheidet INTREST von anderen GIS-Datenbanken?*. – URL [http://intrest.org/downloads/Intrest\\_Info\\_1.PDF](http://intrest.org/downloads/Intrest_Info_1.PDF). – Zugriffsdatum: 19/11/2007
- [Apple:iPhone 2007] : *Apple iPhone Website*. 11 2007. – URL <http://www.apple.com/iphone/>. – Zugriffsdatum: 24/11/2007
- [Esri:ArcSDE 2007] : *ArcSDE Website*. 10 2007. – URL <http://www.esri.com/software/arcgis/arcscde/>. – Zugriffsdatum: 23/10/2007
- [CiteSeer:Website 2007] : *CiteSeer Website*. 03 2007. – URL <http://citeseer.ist.psu.edu/>. – Zugriffsdatum: 07/03/2007
- [CoolGoogleMaps 2007] : *Cool Google Maps*. 11 2007. – URL <http://coolgooglemaps.blogspot.com/>. – Zugriffsdatum: 24/11/2007
- [Deegree:Website 2007] : *Deegree Website*. 10 2007. – URL <http://www.deegree.org/>. – Zugriffsdatum: 23/10/2007
- [EuroRoadS:Framework 2007] : *EuroRoadS Specification Framework*. 11 2007. – URL <http://www.euroroads.org/php/Presentations/framework.pdf>. – Zugriffsdatum: 25/06/07

- [EuroRoadS:Website 2007] : *EuroRoadS Website*. 2007. – URL <http://www.euroroads.org/>. – Zugriffsdatum: 25/06/07
- [eXist:Website 2007] : *eXist Website*. 10 2007. – URL <http://exist.sourceforge.net/>. – Zugriffsdatum: 23/10/2007
- [Galdos:GmlSDK 2007] : *Galdos Inc. GML SDK Website*. 10 2007. – URL <http://www.galdosinc.com/products/gml-sdk/>. – Zugriffsdatum: 22/10/2007
- [Geometa.info:Website 2007] : *Geometa.info Website*. 12 2007. – URL <http://geometa.info/>. – Zugriffsdatum: 07/12/2007
- [GeoServer 2007] : *GeoServer Website*. 11 2007. – URL <http://geoserver.org/>. – Zugriffsdatum: 03/11/07
- [GeoTiff:Website 2007] : *GeoTiff Website*. 10 2007. – URL <http://www.remotesensing.org/geotiff/geotiff.html>. – Zugriffsdatum: 31/10/2007
- [GeoTools:Website 2007] : *GeoTools Website*. 10 2007. – URL <http://xircles.codehaus.org/projects/geotools>. – Zugriffsdatum: 23/10/2007
- [GeoVRML:Website 2007] : *GeoVRML Website*. 10 2007. – URL <http://www.ai.sri.com/geovrml/>. – Zugriffsdatum: 31/10/2007
- [GoogleEarth:Website 2007] : *Google Earth Website*. 11 2007. – URL <http://earth.google.com/>. – Zugriffsdatum: 24/11/2007
- [Google:KML 2007] : *Google KML Documentation*. 11 2007. – URL <http://code.google.com/apis/kml/documentation/>. – Zugriffsdatum: 24/11/2007
- [Google:Maps 2007] : *Google Maps Website*. 11 2007. – URL <http://maps.google.com/>. – Zugriffsdatum: 24/11/2007
- [Google:Scholar 2007] : *Google Scholar Website*. 12 2007. – URL <http://scholar.google.com>. – Zugriffsdatum: 07/12/2007
- [GRASS:Website 2007] : *GRASS Website*. 10 2007. – URL <http://grass.itc.it>. – Zugriffsdatum: 23/10/2007
- [HibernateSpatial:Website 2007] : *Hibernate Spatial Website*. 10 2007. – URL <http://www.hibernate.org/>. – Zugriffsdatum: 11/10/2007
- [Intrest 2007] : *INTREST - Moderne Austauschplattform für verkehrsaффine Daten*. 2007. – URL <http://www.intrest.org>. – Zugriffsdatum: 25/06/07

- [ISO:Website 2007] : *ISO (International Organization for Standardization) Website*. 10 2007. – URL <http://www.iso.org/>. – Zugriffsdatum: 10/10/2007
- [JumpProject 2007] : *JUMP Project Website*. 2007. – URL <http://www.jump-project.org/>. – Zugriffsdatum: 30/04/2007
- [MapBuilder:Website 2007] : *MapBuilder Website*. 10 2007. – URL <http://communitymapbuilder.org/>. – Zugriffsdatum: 31/10/2007
- [MonetDB:Website 2007] : *MonetDB Website*. 06 2007. – URL <http://monetdb.cwi.nl/projects/monetdb/>. – Zugriffsdatum: 13/06/07
- [MySQL:SpatialExtensions 2007] : *MySQL Spatial Extensions*. 2007. – URL <http://dev.mysql.com/doc/refman/5.0/en/spatial-extensions.html>. – Zugriffsdatum: 17/01/2008
- [Nasa:Worldwind 2007] : *Nasa World Wind Website*. 11 2007. – URL <http://worldwind.arc.nasa.gov/>. – Zugriffsdatum: 24/11/2007
- [OpenJump:Website 2007] : *OpenJump Website*. 10 2007. – URL <http://openjump.org/>. – Zugriffsdatum: 23/10/2007
- [Oracle:SpatialDB 2007] : *Oracle Spatial Database*. 2007. – URL <http://www.oracle.com/technology/products/spatial/>. – Zugriffsdatum: 15/05/2007
- [PostGIS:Website 2007] : *PostGIS Website*. 2007. – URL <http://postgis.refrations.net/>. – Zugriffsdatum: 17/01/2008
- [QuantumGIS:Website 2007] : *Quantum GIS Website*. 10 2007. – URL <http://www.qgis.org/>. – Zugriffsdatum: 23/10/2007
- [Snowflakesoftware:GMLViewer 2007] : *Snowflakesoftware GML Viewer Website*. 10 2007. – URL <http://www.snowflakesoftware.co.uk/products/gmlviewer/index.htm>. – Zugriffsdatum: 31/10/2007
- [SpeedCameraDB:GoogleEarthPlugin 2007] : *Speed Camera Database Plugin für Google Earth*. 11 2007. – URL <http://www.scdb.info/en/plugin-google-earth/>. – Zugriffsdatum: 24/11/2007
- [SQL-XML:Website 2007] : *SQL/XML Website*. 10 2007. – URL <http://www.sqlx.org/>. – Zugriffsdatum: 24/10/2007
- [TatukGis:Viewer 2007] : *TatukGis Viewer Website*. 10 2007. – URL <http://www.tatukgis.com/products/viewer/viewer.aspx>. – Zugriffsdatum: 31/10/2007
- [RainingData:TigerLogicXDBMS 2007] : *TigerLogic XDBMS*. 2007. – URL <http://www.rainingdata.com/products/tl/overview.html>. – Zugriffsdatum: 13/01/2008

- [uDig 2007] : *uDig Website*. 2007. – URL <http://udig.refractions.net/confluence/display/UDIG/Home>. – Zugriffsdatum: 30/04/2007
- [Wikipedia:Website 2007] : *Wikipedia.org Website*. 12 2007. – URL <http://wikipedia.org/>. – Zugriffsdatum: 09/12/2007
- [RpBourret:XMLAndDatabases 2007] : *XML and Databases*. Mar 2007. – URL <http://www.rpbouret.com/xml/XMLAndDatabases.htm>. – Zugriffsdatum: 26/03/07
- [ApacheBatik:Website 2008] : *Apache Batik Website*. 01 2008. – URL <http://xmlgraphics.apache.org/batik/>. – Zugriffsdatum: 08/01/2008
- [Snowflakesoftware:GoLoader 2008] : *GoLoader Website*. 2008. – URL <http://www.snowflakesoftware.co.uk/products/goloader/index.htm>. – Zugriffsdatum: 17/01/2008
- [LegoDB:WebSite 2008] : *LegoDB Website*. 01 2008. – URL <http://www-db-out.bell-labs.com/legodb/>. – Zugriffsdatum: 06/01/2008
- [MySQL:Website 2008] : *MySQL Website*. 2008. – URL <http://mysql.com/>. – Zugriffsdatum: 16/01/2008
- [Roncalli:Website 2008] : *Roncalli Telematics Project Website*. 2008. – URL <http://www.roncalli-telematics.com/>. – Zugriffsdatum: 17/01/2008
- [X3D:About 2008] : *What is X3D?* 01 2008. – URL <http://www.web3d.org/about/overview/>. – Zugriffsdatum: 08/01/2008
- [X3DEarth:Website 2008] : *X3D Earth Website*. 01 2008. – URL <http://www.web3d.org/x3d-earth/>. – Zugriffsdatum: 08/01/2008
- [Xj3D:Website 2008] : *Xj3D Website*. 01 2008. – URL <http://www.xj3d.org/>. – Zugriffsdatum: 08/01/2008
- [An und Zhao 2006] AN, Yang ; ZHAO, Bo: Integration of Distributed Geographical Information Based on Mediation and GML. In: *Computer Supported Cooperative Work in Design, 10th International Conference on* (2006), S. 1–6
- [Bohannon u. a. 2002] BOHANNON, P. ; FREIRE, J. ; ROY, P. ; SIMEON, J.: From XML schema to relations: a cost-based approach to XML storage, 2002, S. 64–75
- [Boucelma und Colonna 2004] BOUCELMA, Omar ; COLONNA, François-Marie: GQuery: A Query Language for GML. In: ELFRIEDE FENDEL, Massimo R. (Hrsg.): *24th Urban Data Management Symposium*. Chioggia, Italie, 2004, S. 23–32



- [Boucelma u. a. 2003] BOUCELMA, Omar ; GARINET, Jean-Yves ; LACROIX, Zoé: The virGIS WFS-based spatial mediation system. In: *CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management*. New York, NY, USA : ACM Press, 2003, S. 370–374. – ISBN 1-58113-723-0
- [Brinkhoff 2005] BRINKHOFF, Thomas: *Geodatenbanksysteme in Theorie und Praxis*. Herbert Wichmann Verlag, Heidelberg, 2005. – ISBN 978-3-87907-433-4
- [Castelli u. a. 2007] CASTELLI, Gabriella ; ROSI, Alberto ; MAMEI, Marco ; ZAMBONELLI, Franco: A Simple Model and Infrastructure for Context-Aware Browsing of the World. In: *Pervasive Computing and Communications, 2007. PerCom '07. Fifth Annual IEEE International Conference on* (2007), 03, S. 229–238
- [Chung u. a. 2004] CHUNG, W. ; PARK, Soon-Young ; BAE, Hae-Young: An extension of XQuery for moving objects over GML. In: *Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004. International Conference on* 2 (2004), S. 142–147 Vol.2
- [Corcoles und Gonzalez 2001] CORCOLES, J. E. ; GONZALEZ, P.: A specification of a spatial query language over GML. In: *GIS '01: Proceedings of the 9th ACM international symposium on Advances in geographic information systems*. New York, NY, USA : ACM Press, 2001, S. 112–117. – ISBN 1-58113-443-6
- [Corcoles und Gonzalez 2002] CORCOLES, J. E. ; GONZALEZ, P.: Analysis of different approaches for storing GML documents. In: *GIS '02: Proceedings of the 10th ACM international symposium on Advances in geographic information systems*. New York, NY, USA : ACM Press, 2002, S. 11–16. – ISBN 1-58113-591-2
- [Córcoles und González 2004] CÓRCOLES, J.E. ; GONZÁLEZ, P.: Studying an Approach to Query Spatial XML. In: BELLAHSENE, Zohra (Hrsg.) ; MCBRIEN, Peter (Hrsg.): *DIWeb2004, Third International Workshop on Data Integration over the Web*, 2004, S. 22–33
- [Demontis u. a. 2005] DEMONTIS, Roberto ; VITA, Emanuela D. ; PIRAS, Andrea ; SANNA, Stefano: The CGML: a XML Language for Mobile Cartography. / CRS4, Center for Advanced Studies, Research and Development in Sardinia. Cagliari, Italy, apr 2005 (05/14). – Technical Report. Submitted to MOWEIT2005
- [Ding u. a. 22–24 Aug. 2007] DING, Weili ; ZHU, Feng ; HAO, Yingming: Interactive 3D City Modeling using Google Earth and Ground Images. In: *Image and Graphics, 2007. ICIG 2007. Fourth International Conference on* (22–24 Aug. 2007), S. 849–854
- [Drafting Teams “Data Specifications”, “Network Services”, “Metadata” 2007] DRAFTING TEAMS “DATA SPECIFICATIONS”, “NETWORK SERVICES”, “METADATA”: *INSPIRE Technical Architecture*

- *Overview*. 11 2007. – URL [http://www.ec-gis.org/inspire/reports/ImplementingRules/network/INSPIRETechnicalArchitectureOverview\\_v1.2.pdf](http://www.ec-gis.org/inspire/reports/ImplementingRules/network/INSPIRETechnicalArchitectureOverview_v1.2.pdf). – Zugriffsdatum: 07/12/2007
- [Europäisches Parlament und Rat der europäischen Union 2007] EUROPÄISCHES PARLAMENT ; RAT DER EUROPÄISCHEN UNION: Richtlinie 2007/2/EG des europäischen Parlaments und des Rates vom 14. März 2007 zur Schaffung einer Geodateninfrastruktur in der Europäischen Gemeinschaft (INSPIRE). In: *Amtsblatt der europäischen Union* L 108/1 (2007), 04. – URL <http://eur-lex.europa.eu/JOHtm1.do?uri=OJ:L:2007:108:SOM:DE:HTML>. – ISSN 1725-2539
- [Freire und Simeon 2003] FREIRE, Juliana ; SIMEON, J.: Adaptive XML Shredding: Architecture, Implementation, and Challenges. In: *Efficiency and Effectiveness of XML Tools and Techniques and Data Integration over the Web: VLDB 2002 Workshop EEXTT and CAiSE 2002 Workshop DIWeb. Revised Papers* (2003), S. 104–116
- [GDI-DE ] GDI-DE, Bundesamt für Kartographie und Geodäsie Deutschland: *Architektur der Geodateninfrastruktur Deutschland*. – URL [http://www.gdi-de.org/de/download/GDI\\_ArchitekturKonzept\\_V1.pdf](http://www.gdi-de.org/de/download/GDI_ArchitekturKonzept_V1.pdf). – Zugriffsdatum: 01/12/2007
- [GeoTools ] GEOTOOLS: *GeoTools User Guide: XML Parsing*. – URL <http://docs.codehaus.org/display/GEOTDOC/13+XML>. – Zugriffsdatum: 23/10/2007
- [Gibotti u. a. 2005] GIBOTTI, Fernando R. ; CÂMARA, Gilberto ; NOGUEIRA, Renato A.: GeoDiscover – a specialized search engine to discover geospatial data in the Web. In: *GEOINFO, Brazilian Symposium on GeoInformatics* (2005). ISBN 85-17-00022-6
- [Goh 1997] GOH, Cheng H.: *Representing and reasoning about semantic conflicts in heterogeneous information systems*, Massachusetts Institute of Technology, Sloan School of Management, Dissertation, 1997. – URL <http://hdl.handle.net/1721.1/10713>. – Supervisor-Stuart E. Madnick
- [Google Inc. ] GOOGLE INC.: *Google KMZ*. – URL <http://code.google.com/support/bin/answer.py?answer=55211&topic=10427>. – Zugriffsdatum: 03/10/2007
- [Guan u. a. 2003] GUAN, Ji-Hong ; ZHOU, Shui-Geng ; CHEN, Jun-Peng ; CHEN, Xiao-Long ; AN, Yang ; YU, Wei ; WANG, Rong ; LIU, Xu-Jun: Ontology-based GML schema matching for spatial information integration. In: *Machine Learning and Cybernetics, 2003 International Conference on* 4 (2003), S. 2240–2245 Vol.4
- [Guan und Zhou 15–20 April 2007] GUAN, Jihong ; ZHOU, Shuigeng: GPress: Towards Effective GML Documents Compression. In: *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on* (15-20 April 2007), S. 1473–1474

- [Guo u. a. 2003] GUO, Zhimao ; ZHOU, Shuigeng ; XU, Zhengchuan ; ZHOU, Aoying: G2ST: a novel method to transform GML to SVG. In: *GIS '03: Proceedings of the 11th ACM international symposium on Advances in geographic information systems*. New York, NY, USA : ACM Press, 2003, S. 161–168. – ISBN 1-58113-730-3
- [Gusheng und Xiaohua 2004] GUSHENG, Xu ; XIAOHUA, Tong: GML and XQuery based cadastral spatial object query model description and implementation. In: *Geoscience and Remote Sensing Symposium, 2004. IGARSS '04. Proceedings. 2004 IEEE International 5* (2004), S. 2908–2911 vol.5. ISBN 0-7803-8742-2
- [Güting 1994] GÜTING, R.H.: An Introduction to Spatial Database Systems. In: *The VLDB Journal* 3 (1994), Nr. 4, S. 357–399. – ISSN 1066-8888
- [Honda u. a. Oct. 2006] HONDA, Kiyoshi ; HUNG, Nguyen D. ; SHIMAMURA, Hiroshi: Linking OGC Web Services to Google Earth. In: *SICE-ICASE, 2006. International Joint Conference* (Oct. 2006), S. 4836–4839
- [Huang u. a. 2006] HUANG, Chia-Hsin ; CHUANG, Tyng-Ruey ; DENG, Dong-Po ; LEE, Hahn-Ming: Efficient GML-native processors for web-based GIS: techniques and tools. In: *GIS '06: Proceedings of the 14th annual ACM international symposium on Advances in geographic information systems*. New York, NY, USA : ACM Press, 2006, S. 91–98. – ISBN 1-59593-529-0
- [Inc. 2003] INC., Open Geospatial C.: *OGC Reference Model*. 2003. – URL [http://portal.opengeospatial.org/files/?artifact\\_id=3836](http://portal.opengeospatial.org/files/?artifact_id=3836). – Zugriffsdatum: 07/03/2007
- [Inc. 2005] INC., Open Geospatial C.: *GML Simple Features Profile*. 2005. – URL [http://portal.opengeospatial.org/files/?artifact\\_id=11266](http://portal.opengeospatial.org/files/?artifact_id=11266). – Zugriffsdatum: 07/03/2007
- [Initiative ] INITIATIVE, XML:DB: *What is an XML database?*. – URL <http://xmldb-org.sourceforge.net/faqs.html#faq-1>. – Zugriffsdatum: 20/06/07
- [INSPIRE European Geoportal 2002] INSPIRE EUROPEAN GEOPORTAL: *INSPIRE Architecture and Standards Position Paper*. 10 2002. – URL [http://inspire.jrc.it/reports/position\\_papers/inspire\\_ast\\_pp\\_v4.3\\_en.pdf](http://inspire.jrc.it/reports/position_papers/inspire_ast_pp_v4.3_en.pdf). – Zugriffsdatum: 25/06/07
- [ISO/JTC 1 1996] ISO/JTC 1: Information technology – Open Systems Interconnection – Remote Procedure Call (RPC) / International Organization for Standardization, Genf, Schweiz. 1996 (ISO/IEC 11578:1996). – Veröffentlicht als
- [ISO/TC 204 2004] ISO/TC 204: Intelligent transport systems – Geographic Data Files (GDF) – Overall data specification / International Organization for Standardization, Genf, Schweiz. 2004 (ISO 14825:2004). – Veröffentlicht als

- [ISO/TC 211 2007] ISO/TC 211: Geographic information – Geography Markup Language (GML) / International Organization for Standardization, Genf, Schweiz. 2007 (ISO 19136:2007). – Veröffentlicht als
- [ITU (International Telecommunication Union) ] ITU (INTERNATIONAL TELECOMMUNICATION UNION): *Information technology - Open Systems Interconnection - Procedures for the operation of OSI Registration Authorities: Generation and registration of Universally Unique Identifiers (UUIDs) and their use as ASN.1 Object Identifier components.* – URL <http://www.itu.int/ITU-T/studygroups/com17/oid/X.667-E.pdf>. – Zugriffsdatum: 11/01/2008
- [Jiang u. a. 2002] JIANG, Haifeng ; LU, Hongjun ; WANG, Wei ; YU, Jeffrey X.: Path materialization revisited: an efficient storage model for XML data. In: *ADC '02: Proceedings of the 13th Australasian database conference*. Darlinghurst, Australia, Australia : Australian Computer Society, Inc., 2002, S. 85–94. – ISBN 0-909925-83-6
- [Keller und Kälin 2005] KELLER, Stefan ; KÄLIN, André: Geometa.info – Eine Spezial-Suchmaschine als Ergänzung zu Geo-Katalogen und allgemeinen Suchmaschinen. In: STROBL, J. (Hrsg.) ; BLASCHKE, T. (Hrsg.) ; GRIESEBNER, G. (Hrsg.): *Proceedings of Angewandte Geographische Informationsverarbeitung XVII*, Wichmann, 2005. – URL <http://geometa.info/>
- [Klementsitz u. a. 2005] KLEMENTSCHITZ, R. ; NEUMANN, A. ; SAMMER, G. ; ET AL.: *Die Clearing-Stelle als Marktplatz für multi-modale Verkehrstelematik-Information, Endbericht (RONCALLI2)*. 2005
- [Krishnamurthy u. a. 2003] KRISHNAMURTHY, R. ; KAUSHIK, R. ; NAUGHTON, J. F.: XML-to-SQL Query Translation Literature: The State of the Art and Open Problems. In: *Database and XML Technologies* (2003), S. 1–18
- [Krishnamurthy u. a. 2004] KRISHNAMURTHY, Rajasekar ; KAUSHIK, Raghav ; NAUGHTON, Jeffrey F.: Efficient XML-to-SQL Query Translation: Where to Add the Intelligence? In: NASCIMENTO, Mario A. (Hrsg.) ; ÖZSU, M. T. (Hrsg.) ; KOSSMANN, Donald (Hrsg.) ; MILLER, Renée J. (Hrsg.) ; BLAKELEY, José A. (Hrsg.) ; SCHIEFER, K. B. (Hrsg.): *VLDB*, Morgan Kaufmann, 2004, S. 144–155
- [Li u. a. 2004] LI, Yuzhen ; LI, Jun ; ZHOU, Shuigeng: GML Storage: A Spatial Database Approach. In: *Conceptual Modeling for Advanced Application Domains* (2004), S. 55–66. – ISBN 978-3-540-23722-8
- [Lu u. a. 2007] LU, Chang-Tien ; SANTOS, Raimundo F. D. ; SRIPADA, Lakshmi N. ; KOU, Yufeng: Advances in GML for Geospatial Applications. In: *GeoInformatica* 11 (2007), Nr. 1, S. 131–157

- [Lu u. a. 2005] LU, Hongjun ; YU, Jeffrey X. ; WANG, Guoren ; ZHENG, Shihui ; JIANG, Haifeng ; YU, Ge ; ZHOU, Aoying: What makes the differences: benchmarking XML database implementations. In: *ACM Trans. Inter. Tech.* 5 (2005), Nr. 1, S. 154–194. – ISSN 1533-5399
- [Megginson ] MEGGINSON, David: *SAX Website*. – URL <http://www.saxproject.org/>. – Zugriffsdatum: 12/06/07
- [Min u. a. 2006] MIN, Jun-Ki ; PARK, Myung-Jae ; CHUNG, Chin-Wan: A compressor for effective archiving, retrieval, and updating of XML documents. In: *ACM Trans. Inter. Tech.* 6 (2006), Nr. 3, S. 223–258. – ISSN 1533-5399
- [OGC a] OGC: *OpenGIS® Catalogue Services Specification*. – URL <http://www.opengeospatial.org/standards/cat>. – Zugriffsdatum: 18/04/2007
- [OGC b] OGC: *The OpenGIS Service Architecture*. – URL <http://www.opengeospatial.org/standards/as>. – Zugriffsdatum: 22/05/07
- [OGC c] OGC: *OpenGIS® Web Feature Service (WFS) Implementation Specification*. – URL <http://www.opengeospatial.org/standards/wfs>. – Zugriffsdatum: 25/04/2007
- [OGC d] OGC: *OpenGIS® Web Map Service (WMS) Implementation Specification*. – URL <http://www.opengeospatial.org/standards/wms>. – Zugriffsdatum: 25/04/2007
- [OGC 2004] OGC: *Geography Markup Language*. 2004. – URL [http://portal.opengeospatial.org/files/?artifact\\_id=4700](http://portal.opengeospatial.org/files/?artifact_id=4700). – Zugriffsdatum: 19/01/2008
- [OGC 2007a] OGC: *All Registered Products*. 2007. – URL <http://www.opengeospatial.org/resource/products>. – Zugriffsdatum: 13/01/2008
- [OGC e] OGC, Open Geospatial Consortium Inc.: *OGC Abstract Specifications*. – URL <http://www.opengeospatial.org/standards/as>. – Zugriffsdatum: 25/04/2007
- [OGC f] OGC, Open Geospatial Consortium Inc.: *OpenGIS® Filter Encoding Implementation Specification*. – URL <http://www.opengeospatial.org/standards/filter>. – Zugriffsdatum: 18/05/2007
- [OGC g] OGC, Open Geospatial Consortium Inc.: *OpenGIS® Geography Markup Language (GML) Encoding Specification*. – URL <http://www.opengeospatial.org/standards/gml>. – Zugriffsdatum: 18/06/07
- [OGC h] OGC, Open Geospatial Consortium Inc.: *OpenGIS® Implementation Specification for Geographic information - Simple feature access - Part 2: SQL option*. – URL <http://www.opengeospatial.org/standards/sfs>. – Zugriffsdatum: 19/01/2008

- [OGC i] OGC, Open Geospatial Consortium Inc.: *OpenGIS® Simple Features Implementation Specification for CORBA*. – URL <http://www.opengeospatial.org/standards/sfc>. – Zugriffsdatum: 25/04/2007
- [OGC j] OGC, Open Geospatial Consortium Inc.: *OpenGIS® Simple Features Implementation Specification for OLE/COM*. – URL <http://www.opengeospatial.org/standards/sfo>. – Zugriffsdatum: 25/04/2007
- [OGC k] OGC, Open Geospatial Consortium Inc.: *OpenGIS® Styled Layer Descriptor (SLD) Implementation Specification*. – URL <http://www.opengeospatial.org/standards/sld>. – Zugriffsdatum: 2007
- [OGC l] OGC, Open Geospatial Consortium Inc.: *OpenGIS® Web Coverage Service (WCS) Implementation Specification*. – URL <http://www.opengeospatial.org/standards/wcs>. – Zugriffsdatum: 25/04/2007
- [OGC m] OGC, Open Geospatial Consortium Inc.: *Web Processing Service (WPS): Request for Public Comments*. – URL <http://www.opengeospatial.org/standards/requests/28>. – Zugriffsdatum: 25/04/2007
- [OGC 2007b] OGC, Open Geospatial Consortium Inc.: *OGC Website*. 2007. – URL <http://www.opengeospatial.org/>. – Zugriffsdatum: 17/01/2008
- [OGC 2007c] OGC, Open Geospatial Consortium Inc.: *OpenGIS® GML in JPEG 2000 for Geographic Imagery Encoding Specification*. 11 2007. – URL <http://www.opengeospatial.org/standards/gmljp2>. – Zugriffsdatum: 03/11/2007
- [Oracle ] ORACLE: *Oracle XML DB*. – URL <http://www.oracle.com/technology/tech/xml/xmlldb/>. – Zugriffsdatum: 14/06/07
- [Peng und Zhang 2004] PENG, Zhong-Ren ; ZHANG, Chuanrong: The roles of geography markup language (GML), scalable vector graphics (SVG), and Web feature service (WFS) specifications in the development of Internet geographic information systems (GIS). In: *Journal of Geographical Systems* 6 (2004), Nr. 2, S. 95–116
- [Pock 2007] POCK, Maximilian: *Koordination Umsetzung Inspire-Österreich*. Vortrag. Nov 2007. – URL [http://www.ageo.at/aktuelles/archiv/28112007/Pock\\_AGE0\\_28Nov2007.pdf](http://www.ageo.at/aktuelles/archiv/28112007/Pock_AGE0_28Nov2007.pdf)
- [Schmidt u. a. 2001] SCHMIDT, Albrecht ; KERSTEN, Martin ; WINDHOUWER, Menzo ; WAAS, Florian: Efficient Relational Storage and Retrieval of XML Documents. In: *The World Wide Web and Databases: Third International Workshop WebDB 2000, Dallas, TX, USA, May 2000. Selected Papers* (2001), S. 137–



- [Shekhar und Chawla 2003] SHEKHAR, Shashi ; CHAWLA, Sanjay: *Spatial Databases: A Tour*. Prentice Hall, 2003
- [Shrestha 2004] SHRESTHA, Bikram B.: *XML Database Technology and its use for GML*, International Institute for Geo-Information Science and Earth Observation, Enschede, Netherlands, Diplomarbeit, 2004
- [Sripada u. a. 2004] SRIPADA, L. N. ; LU, Chang-Tien ; WU, Weili: Evaluating GML support for spatial databases. In: *Computer Software and Applications Conference, 2004. COMPSAC 2004. Proceedings of the 28th Annual International* 2 (2004), S. 74–77 vol.2. ISBN 0730-3157
- [Vakali u. a. 2005] VAKALI, A. ; CATANIA, B. ; MADDALENA, A.: XML data stores: emerging practices. In: *Internet Computing, IEEE* 9 (2005), Nr. 2, S. 62–69. ISBN 1089-7801
- [Vânia Vidal 2005] VÂNIA VIDAL, Fábio F.: Translating WFS Query to SQL/XML Query. In: FONSECA, Frederico (Hrsg.) ; CASANOVA, Marco A. (Hrsg.): *VII Brazilian Symposium on Geoinformatics - GEOINFO*. Campos do Jordão, 2005
- [W3C a] W3C, Word Wide Web Consortium: *Document Object Model*. – URL <http://www.w3.org/DOM/>. – Zugriffsdatum: 12/06/07
- [W3C b] W3C, Word Wide Web Consortium: *W3C Semantic Web Website*. – URL <http://www.w3.org/2001/sw/>. – Zugriffsdatum: 19/01/2008
- [W3C c] W3C, Word Wide Web Consortium: *W3C SMIL Website*. – URL <http://www.w3.org/AudioVideo/>. – Zugriffsdatum: 19/01/2008
- [W3C d] W3C, Word Wide Web Consortium: *W3C SVG Website*. – URL <http://www.w3.org/Graphics/SVG/>. – Zugriffsdatum: 19/01/2008
- [W3C e] W3C, Word Wide Web Consortium: *W3C XML Schema*. – URL <http://www.w3.org/XML/Schema>. – Zugriffsdatum: 18/06/07
- [W3C f] W3C, Word Wide Web Consortium: *Web Services Architecture*. – URL <http://www.w3.org/TR/ws-arch/>. – Zugriffsdatum: 22/05/07
- [W3C g] W3C, Word Wide Web Consortium: *World Wide Web Consortium Website*. – URL <http://www.w3.org>. – Zugriffsdatum: 22/05/07
- [W3C h] W3C, Word Wide Web Consortium: *XML Path Language (XPath) Version 1.0 W3C Recommendation*. – URL <http://www.w3.org/TR/xpath>. – Zugriffsdatum: 19/01/2008
- [W3C i] W3C, Word Wide Web Consortium: *XQuery 1.0: An XML Query Language W3C Recommendation*. – URL <http://www.w3.org/TR/xquery/>. – Zugriffsdatum: 19/01/2008

- [W3C j] W3C, Word Wide Web Consortium: *XSLT*. – URL <http://www.w3.org/TR/xslt>. – Zugriffsdatum: 18/06/07
- [Wagner 2008] WAGNER, Jan-Oliver: *FreeGIS.org Website*. 2008. – URL <http://freegis.org/>. – Zugriffsdatum: 24/01/2008
- [wikipedia.org ] WIKIPEDIA.ORG: *GML Application Schemas*. – URL [http://en.wikipedia.org/wiki/GML\\_Application\\_Schemas](http://en.wikipedia.org/wiki/GML_Application_Schemas). – Zugriffsdatum: 18/06/07
- [Ye u. a. 2005] YE, Sheng ; XUEZHI, Feng ; YUAN, Shaotao ; JULIANG, Li: Visualization GML with SVG. In: *Geoscience and Remote Sensing Symposium, 2005. IGARSS '05. Proceedings. 2005 IEEE International* 5 (2005), S. 3648–3651
- [Yingwei u. a. 2004] YINGWEI, Luo ; XINPENG, Liu ; WENJUN, Wang ; XIAOLIN, Wang ; ZHUOQUN, Xu: Web service and geographical information integration. In: *Computer Software and Applications Conference, 2004. COMPSAC 2004. Proceedings of the 28th Annual International* 2 (2004), S. 130–133 vol.2. ISBN 0730-3157
- [Zhu u. a. 2006] ZHU, Fubao ; GUAN, Jihong ; ZHOU, Jiaogen ; ZHOU, Shuigeng: Storing and querying GML in object-relational databases. In: *GIS '06: Proceedings of the 14th annual ACM international symposium on Advances in geographic information systems*. New York, NY, USA : ACM Press, 2006, S. 107–114. – ISBN 1-59593-529-0