XML-Prototyp einer Straßeninformationsbank

Abschlussdokumentation

Version:	1.1
Datum:	19.05.2003
Status:	abgeschlossen
Dateiname:	AP03-2-v1.1.doc
Pfad:	N/a
Verantwortlich:	R. Erstling
	interactive instruments GmbH

Version	Datum	Änderungen	Bearbeiter
0.1	01.04.03	Dokument initialisiert	R. Erstling
0.2	22.04.03	Ausformulierung von Kapitel 3, Integration der Beiträge von C. Vogt und B. Weidner	R. Erstling
0.3	25.04.03	Überarbeitung nach Vorschlägen von C. Portele	R. Erstling
0.4	26.04.03	Überarbeitung nach Vorschlägen von C. Vogt	R. Erstling
1.0	27.04.03	Abschluss	R. Erstling
1.1	19.05.03	Einarbeitung der Änderungswünsche der PG 27	R. Erstling

Inhaltsverzeichnis

1 Kurzfassung

Das Projekt „XML-Prototyp" untersucht die Eignung einer XML-basierten Informationsstrukturierung für eine Straßeninformationsbank (SIB) der Zukunft durch eine prototypische Implementierung ausgewählter fachlicher Prozesse. Der Prototyp einer SIB auf XML-Basis arbeitet mit einer vorgegebenen Teilmenge der SIB-Netz- und Bestandsdaten auf der Grundlage der von der OKSTRA®-Pflegestelle erarbeiteten OKSTRA®-XML-Schemata.

Prototypisch realisiert wird eine mehrschichtige Architektur mit Datenhaltung (Schicht 0), Datenbereitstellung (Schicht 1), Datenaufbereitung (Schicht 2) und Datenpräsentation (Schicht 3). Datenbereitstellung und Datenaufbereitung sind als Web-Services realisiert, wobei zur Datenbereitstellung der OKSTRA®-Objekte „Web Feature Services" (WFS) gemäß den Spezifikationen des Open GIS Consortiums eingesetzt werden. Zur Datenaufbereitung kommen im Projekt definierte, fachspezifische Services zum Einsatz. Die Datenpräsentation erfolgt durch einen Web-Browser, also als „Thin-Client".

Die Ergebnisse des Prototyping ermöglichen eine Bewertung des Einsatzes von XML-Technologien und Web-Services im Bereich einer Straßeninformationsbank.

Der Nachweis, dass eine durch und durch XML-strukturierte SIB in einer mehrschichtigen Dienstearchitektur umsetzbar ist, konnte in vollem Umfang erbracht werden. Der Prototyp realisiert alle geforderten Testfälle und ist demonstrierbar. Er beweist, das eine SIB-Datenhaltung verteilt auf unterschiedlichen Systemen durch Web-Services zu einer Applikation integriert werden kann. Bewährt hat sich der Einsatz von OKSTRA®-XML als Basis sowie der Einsatz des OGC WFS zur Bereitstellung von OKSTRA®-XML-Daten.

Als verbesserungswürdig an der vorliegenden Spezifikation des OGC WFS wurde vor allem das zu einfache Transaktionskonzept erkannt, das die Möglichkeiten zur sicheren Fortführung verteilter Daten nicht genügend berücksichtigt.

In der Bewertung der Ergebnisse zeigt sich, dass die Vorteile einer Architektur von Web-Services besonders durch die Verteilung von Daten und Diensten an viele Nutzer auf dem Wege der „Thin-Clients" erreicht wird. Gleichzeitig erweist sich die Programmierung vom Web-Applikationen mit einer Thin-Client-Benutzerschnittstelle als relativ aufwändig.

Diese Überlegungen führen letztlich zur Schlussfolgerung, dass einfache Web-Applikationen, die von einer Vielzahl möglicher Nutzer benötigt werden, das beste Kosten-/Nutzen-Verhältnis aufweisen. Web-Applikationen dieser Art werden als Auskunftssysteme und einfache fachliche Fortführungssysteme charakterisiert. Die XML-basierten Web-Services, vor allem auf der Basis internationaler Standards, bieten die Chance, kooperative, web-basierte Dienste einzurichten, die funktionieren, obwohl die konkrete Datenhaltung im Straßen- und Verkehrswesen auf unterschiedlichen Lösungen beruht. In den einzelnen Straßenbauverwaltungen eröffnen sie die Möglichkeit, Fachanwendungen über definierte Schnittstellen an die vorhandenen Netzdatenbestände anzuschließen und alle Datenbestände einer umfassenderen Nutzung zuzuführen.

Es wird daher vorgeschlagen, die bestehenden SIBs mit Standardschnittstellen (WFS und WMS/SLD) auszurüsten und diese als Basis für zukünftige Fachanwendungen einzusetzen.

2 Aufgabenstellung

Das Ziel des Projekts war die Untersuchung der Eignung einer XML-basierten Informationsstrukturierung für eine Straßeninformationsbank (SIB) der Zukunft. Zu diesem Zweck wurde ein Prototyp einer Straßeninformationsbank auf XML-Basis zu erstellt. Der Prototyp arbeitet mit einer vorgegebenen Teilmenge der SIB-Netzdaten und SIB-Bestandsdaten und verwendet zu deren Beschreibung XML-Schemata aus dem Objektkatalog für das Straßen- und Verkehrswesen (OKSTRA®).

Die Leistungsfähigkeit und Eignung oder ggf. Nichteignung der XML-Technologien für die verschiedenen Bereiche und Zwecke der SIB war durch Aufbau und Test des Prototyps, durch XML-konforme Ein- und Ausgabe sowie durch Datenaufbereitungen und -auswertungen nachzuweisen.

Die folgenden fachlichen Prozesse wurden prototypisch realisiert:

- Netzknotensystem festlegen inkl. Netzänderungen
- Bestandsdaten übernehmen
- Verwaltungsmaßnahmen durchführen
- Erstellen einer Längenstatistik

Die prototypische Untersuchung der Eignung einer XML-basierten SIB für die Aufgaben der Straßenbauverwaltung wurde besonders vor dem Hintergrund der nachfolgend dargelegten Rahmenbedingungen als erforderlich angesehen:

XML-Technologien[1] finden zunehmend starke Verbreitung im gesamten Bereich der IT. Die XML-Technologien sind allerdings noch jung und vor allem noch im Fluss, direkte Erfahrungen im Umgang mit SIB-Daten auf XML-Basis liegen noch nicht in ausreichendem Maße vor. Um die Eignung einer XML-basierten Informationsstrukturierung in der SIB bewerten zu können, sind jedoch konkrete und praktische Erfahrungen mit diesen Technologien im fachlichen Umfeld der SIB notwendig.

Es besteht die Hoffnung, dass eine konsequente Zugrundelegung von XML-Technologien es ermöglicht, die Bindung von Implementierungen von SIB-Modulen an bestimmte Betriebssysteme, Datenbankmanagementsysteme und Datenverteilungsstrategien aufzuheben.

Darüber hinaus soll durch die Verwendung von **Internet/Intranet-Technologien**[2] der Zugang zu SIB-Informationen signifikant erleichtert werden. Dies beruht neben der Verwendung von XML zur Kodierung von Informationen vor allem auf der Tatsache, dass quasi jeder Computer über entsprechende vorinstallierte Software verfügt und mindestens mit dem Internet verbunden ist (natürlich unter Beachtung der jeweiligen IT-Sicherheitsrichtlinien).

[1] Neben dem Kernstandard XML verstehen wir hierunter die gesamte Familie von weiteren Standards, die in diesem Umfeld – zumeist durch das W3C – entwickelt wurden und werden.

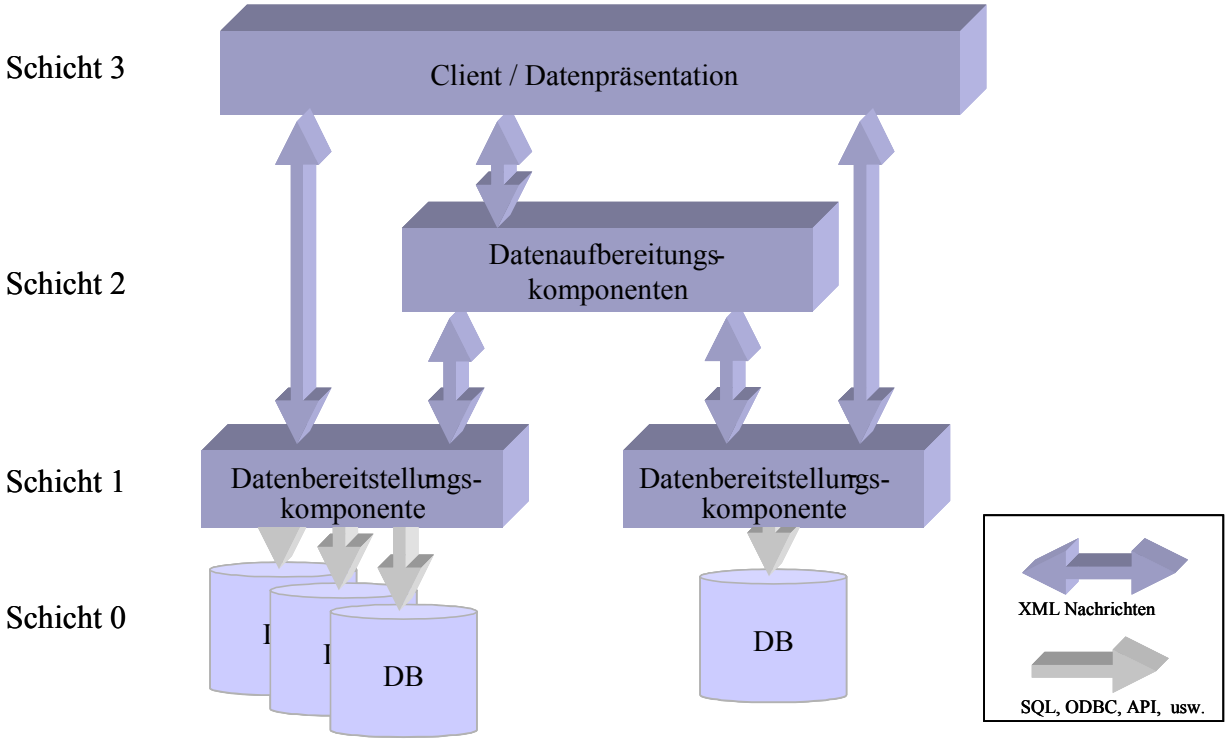
[2] Wir verstehen hierunter gebräuchliche Protokolle und Standards der IETF und darauf aufbauend auch des W3C.

3 Umsetzung des XML-Prototyps

In diesem Kapitel werden die Architektur sowie die implementierten Komponenten des XML-Prototyps vorgestellt. Es dient als Grundlage sowie zum Verständnis der nächsten Kapitel und Diskussionen und wird deshalb in einer stark komprimierten Form dargestellt. Eine ausführliche Behandlung der einzelnen Themen findet sich im Feinkonzept des XML-Prototypen (Dokument „AP02-1-v1.0.doc").

3.1 Architektur

Basierend auf den fachlichen und informationstechnischen Rahmenbedingungen des XML-Prototyps wurde eine mehrschichtige Architektur umgesetzt, deren einzelne Komponenten in den folgenden Unterkapiteln kurz erläutert werden.



Die gewählte Struktur ermöglicht Zugriffe auf verteilte Datenbereitstellungskomponenten. In der Datenaufbereitungs- und der Datenpräsentationsschicht werden die Ergebnisse aus beiden Komponenten kombiniert und integriert ausgewertet.

Bei der konkreten Umsetzung des XML-Prototyps wurden zwei Datenhaltungs- bzw. Datenbereitstellungskomponenten verwendet: Eine für das Straßennetz und eine für die Bestandsdaten (Querschnitts- und Aufbaudaten).

3.2 Datenhaltung (Schicht 0 der Softwarearchitektur)

Bezüglich der Datenhaltung (Schicht 0) wurden aufgrund der DV-technischen Rahmenbedingungen keinerlei Einschränkungen oder Vorgaben bezüglich des Betriebssystems, des Datenbankmanagementsystems oder der Bandbreite zwischen den einzelnen Komponenten des Prototypen gemacht. Durch die Nutzung von Standards bei der Datenbereitstellungskomponente kann die Datenhaltung mit beliebigen Systemen erfolgen, die diese Standards unterstützen. Insoweit war die Art der Datenhaltung aus der Sicht der XML-Strukturierung irrelevant und wurde im Prototypen auch nicht vertieft behandelt.

Für die Datenhaltung der Realisierung des Prototypen wurde die OKSTRA®-Komponente XTRA[3] von interactive instruments eingesetzt.

Die Datenhaltung bedient ein Profil auf Basis der vorliegenden XML-Schema-Beschreibungen der aktuellen, vollständigen OKSTRA®-XML-Entwürfe (Dokumente N0036 und N0037 der OKSTRA®-Pflegestelle, das zugrundeliegende Konzept ist in Dokument N0028 beschrieben).

Für die Zwecke des XML-Prototypen wurden Teile folgender Schemata des OKSTRA® umgesetzt:

1. Strassennetz
2. Administration
3. Verkehr
4. Bauliche_Strasseneigenschaften
5. Geometrieschema
6. Historisierung
7. Allgemeine_Geometrieobjekte

Ferner wurden Typdefinitionen aus dem Schema Allgemeine_Objekte verwendet. Für die Nummernverwaltung wurden zudem die Festlegungen aus der neuen ASB zum TK-Blatt-Verzeichnis in das Profil übernommen.

Die XML-Schema-Beschreibungen zu den OKSTRA®-Objektklassen wurden bis auf geringfügige Anpassungen den auf den OKSTRA®-Webseiten veröffentlichten abgeleiteten OKSTRA®-XML-Schemata entnommen. Die Anpassungen haben sich aus den bisherigen praktischen Erfahrungen mit den abgeleiteten XML-Schemata ergeben und wurden in dieser Form auch in der zu verabschiedenden Fassung von OKSTRA®-XML durchgeführt.

3.3 Datenbereitstellung (Schicht 1)

Aufbauend auf der Datenhaltung werden in dieser Schicht die Daten XML-strukturiert zugänglich gemacht – auch für die Fortführung. Die Komponente nimmt Aufträge/Anfragen über definierte

[3] XTRA ist eine Komponente zur Verfügbarmachung und Speicherung von OKSTRA®-Objekten, die es nach außen in der Form von CTE-Dateien, OKSTRA®-SQL und OKSTRA®-XML entgegennehmen und ablegen kann. Die Programmierschnittstelle von XTRA ist die Basis von XtraServer, dessen Komponente XtraWFS als Datenbereitstellungskomponente eingesetzt wird. XtraWFS addiert zur XML-Bereitstellung durch XTRA noch das standardisierte Diensteprotokoll des „OGC Web Feature Servers".

XML-Nachrichten entgegen und liefert Ergebnisse ebenfalls als XML-Nachrichten zurück. Wo immer Fachobjekte beschrieben werden, wurden die OKSTRA®-XML-Schema-Vorgaben verwendet.

Aus der Sicht der XML-Strukturierung bilden „Datenhaltung" und „Datenbereitstellung" eine Einheit. Die Fachdienste der Datenaufbereitung und die Datenpräsentation (Client) haben keinerlei direkten Kontakt zur Datenhaltung. Alle Informationen werden über die Dienste der Datenbereitstellung ausgetauscht. Damit ist es möglich, jederzeit eine andere Datenhaltung einzusetzen, wenn diese dieselben Datenbereitstellungs-Dienste zur Verfügung stellt wie die in dieser Architektur beschriebenen.

Die Anforderungen an die Datenbereitstellung sind in einer XML-basierten SIB sehr hoch. Neben datenbanktechnischen Funktionen zur Veränderung der Daten (*insert, update, delete*) müssen auch komplexe Abfragen zur Verfügung gestellt werden können. Da es sich bei den Daten einer Straßeninformationsbank fast immer um Daten mit einem räumlichen Bezug handelt, ist es zwingend erforderlich, dass die Datenbereitstellungskomponente auch die Selektion anhand von räumlichen Kriterien ermöglicht.

Der einzige Standard, der diese Anforderungen erfüllt und zudem vollständig auf XML-Strukturen basiert, ist der „Web Feature Server" Standard des Open GIS Consortiums. Ein Web Feature Service ist ein Web-Dienst (d.h. zur Kommunikation wird HTTP benutzt) zur Beschreibung, Abfrage, Erzeugung, Fortschreibung und Löschung von Daten mit geographischem Bezug, die gemäß einem GML-Applikationsschema strukturiert werden können. Für den OKSTRA® liegt ein Vorschlag für eine Strukturierung in Form eines GML-Applikationsschemas vor. OKSTRA®-Daten sind daher für die Abfrage und Pflege durch einen Web Feature Server geeignet.

Für den Web Feature Service gibt es bereits eine Reihe von Produkten von verschiedenen GIS-Herstellern, die diese Schnittstelle realisieren. Es ist zu erwarten, dass in Folge der Veröffentlichung des Standards (September 2002) weitere Produkte hinzukommen werden. Auch die Vermessungsverwaltungen der Länder bauen in ihrem AFIS®-ALKIS®-ATKIS®-Standard auf diese Spezifikation auf.

In der konkreten Realisierung des XML-Prototypen kam das Produkt XtraServer/XtraWFS von interactive instruments zum Einsatz. XtraWFS baut auf XTRA auf und ist ein Web-Feature-Service-Produkt für OKSTRA®-Daten.

Die Wahl von XtraWFS als WFS-Komponente beeinträchtigt nicht die konzeptionelle Produktunabhängigkeit des XML-Prototypen. Bereits die Kapselung der Datenhaltung durch einen geeigneten Web Service stellt diese sicher. Durch die zusätzliche Festlegung auf einen internationalen Standard wird die Chance eröffnet, fertige Lösungen für diese Schicht auf dem Markt vorzufinden. Statt XtraWFS könnte im Prinzip jeder andere Web-Feature-Service zum Einsatz kommen, der eine OKSTRA®-Datenhaltung und -bereitstellung im erforderlichen Umfang realisiert.

Eine ausführlichere Beschreibung des „Web Feature Service" sowie seiner Funktionen erfolgt im Kapitel 4.1.3 des Feinkonzepts.

3.4 Datenaufbereitung (Schicht 2)

Aufbauend auf den Diensten zur Bereitstellung der SIB-Daten in XML auf der Basis der OKSTRA®-XML-Schema-Definitionen werden in dieser Schicht die Daten so aufbereitet, dass spezifische SIB-Funktionen realisiert werden. In der Datenaufbereitungsschicht ist somit ein wesentlicher Teil des Anwendungswissens der Straßeninformationsbank enthalten.

In dieser Schicht liegen Komponenten, die spezifische Dienste im Rahmen der SIB anbieten. Für den Prototypen sind dies:

- „Netzpflege": Operationen, die das Netzknoten-Stationierungssystem verändern wie „Abschnitt/Ast teilen", „Abschnitte/Äste vereinigen", „Stationierungsrichtung umdrehen", „Länge des Abschnitt/Ast ändern", „Verlauf des Abschnitt/Ast ändern" und die jeweils damit einhergehende Fortführung der vorhandenen Punkt-, Strecken und Bereichseigenschaften.
- „Nummernverwaltung": Operationen für die Verwaltung der verfügbaren Netzknotennummern. Damit wird sichergestellt, dass innerhalb eines TK-Blatts Nummern nicht doppelt vergeben werden
- „Verwaltungsmaßnahmen": Operationen wie "Umstufung/Umnummerierung", "Dienststellen- oder Verwaltungsgrenze verlegen", etc.
- „DA-Import": Operationen für die Übernahme von Bestandsdaten.
- „Längenstatistik".

Alle Operationen und ihre Parameter wurden in Form von XML-Schema-Dokumenten spezifiziert (siehe Feinkonzept). Im Unterschied zur Datenbereitstellungsschicht handelt es sich hierbei um SIB-spezifische Beschreibungen. So ist das Ergebnis einer Längenstatistik-Anfrage ein XML-Dokument, das genau die Informationen der Längenstatistik enthält. Für die Durchführung der Datenaufbereitung nutzen die Komponenten die Datenbereitstellungsdienste der Schicht 1 und andere Dienste aus der Schicht 2. Mit diesem Ansatz wird eine Modularisierung der SIB unterstützt und besonders auch eine schrittweise Realisierung ermöglicht.

Der Umfang der im Prototypen definierten SIB-Operationen ist nur ein Auszug aus der Gesamtheit der für einen Produktionseinsatz benötigten Geschäftsprozesse. Es wurden lediglich diejenigen Operationen umgesetzt, die für die Durchführung der spezifizierten Testfälle notwendig sind. Dabei stellen die gewählten Dienstgruppen („Netzänderung", „DA-Import", etc.) jeweils eigenständige Module dar, die unabhängig voneinander funktionieren, sich aber unter Umständen gegenseitig nutzen (so greift z.B. die Netzänderung häufig auf Nummernverwaltung zurück).

Hervorzuheben ist in diesem Zusammenhang, dass die gesamte Kommunikation, also sowohl von den einzelnen Services zu der Datenbereitstellungsschicht als auch zwischen den Services, mit Hilfe von XML und auf Basis von Internetprotokollen (HTTP) erfolgt.

Für sämtliche Fachdienste fand das OKSTRA®-XML-Schema unmittelbare Anwendung. Es wurde bei den Diensten also intern mit denselben Strukturen gearbeitet, die auch für den Datenaustausch zwischen den einzelnen Diensten Verwendung fand. Somit konnte die Eignung von XML-Strukturen für eine SIB auf Basis des OKSTRA®-XML-Schemas am Prototypen unmittelbar überprüft werden. Die daraus gewonnenen Erkenntnisse beschreibt Kapitel 4.6, eine ausführliche Beschreibung der einzelnen Fachdienste hingegen kann dem Kapitel 4.2 des Feinkonzepts entnommen werden.

3.5 Datenpräsentation (Schicht 3)

Die Schichten 0 bis 2 bieten Komponenten, die ausschließlich über Softwareschnittstellen angesprochen werden. In der Schicht 3, der Datenpräsentation, werden die Ergebnisse so aufbereitet, dass sie im „SIB-XML-Client" dem Anwender zugänglich gemacht werden. Der Client läuft in einem Standard-Web-Browser und realisiert prototypisch (und mit entsprechenden Einschränkungen) die Benutzeroberfläche für die Komponenten aus den Schichten 1 und 2.

Der Client besteht sowohl aus statischem als auch aus in der Datenpräsentation dynamisch generiertem HTML-Code. Dieser wird zum Teil aus den XML-Daten über XSL/XSLT anwenderfreundlich

generiert. Die geografischen Informationen der OKSTRA®-XML-Daten (GML) werden durch einen Mapdienst in Karten gewandelt und im Browser visualisiert.

Der Client wurde vorwiegend mit dem Internet Explorer ab Version 5 unter Windows getestet. Tests mit Netscape wurden durchgeführt, eine Kompatibilität mit alten Netscape-Browsern wurde jedoch wegen der fehlenden Aussagekraft für die Ziele des Prototypings nicht angestrebt. Neue (Mozilla-basierte) Netscape-Browser (ab Version 7) sind erfolgreich getestet worden. Dabei ist anzumerken, dass eine browserseitige Interpretation von XSL nur mit dem Internet Explorer ab Version 6 möglich ist und auch nur bei Einsatz dieses Browsers durchgeführt wurde. Bei der Verwendung von anderen Browsern erfolgt die Interpretation von XSL serverseitig durch den Web-Server.

Obwohl die Ausschreibungsunterlagen eine grafische Visualisierung der Daten nicht verlangen, wurde zur Visualisierung der Daten im Client als Ergänzung der Datenbereitstellung über XtraWFS auch XtraWMS als Web-Map-Server eingesetzt. XtraWMS realisiert die Web-Map-Server-Spezifikation des Open GIS Consortiums, Versionen 1.1.0 und 1.1.1.

Allerdings ergaben sich durch die Ausrichtung der Software auf ihren Zweck als **prototypische Anwendung** zur Ermittlung der Eignung einer XML-Strukturierung auch einige **Einschränkungen**, die sich letztlich auch im Bedienkomfort des Client ausdrücken.

Die wesentlichen Einschränkungen sind:

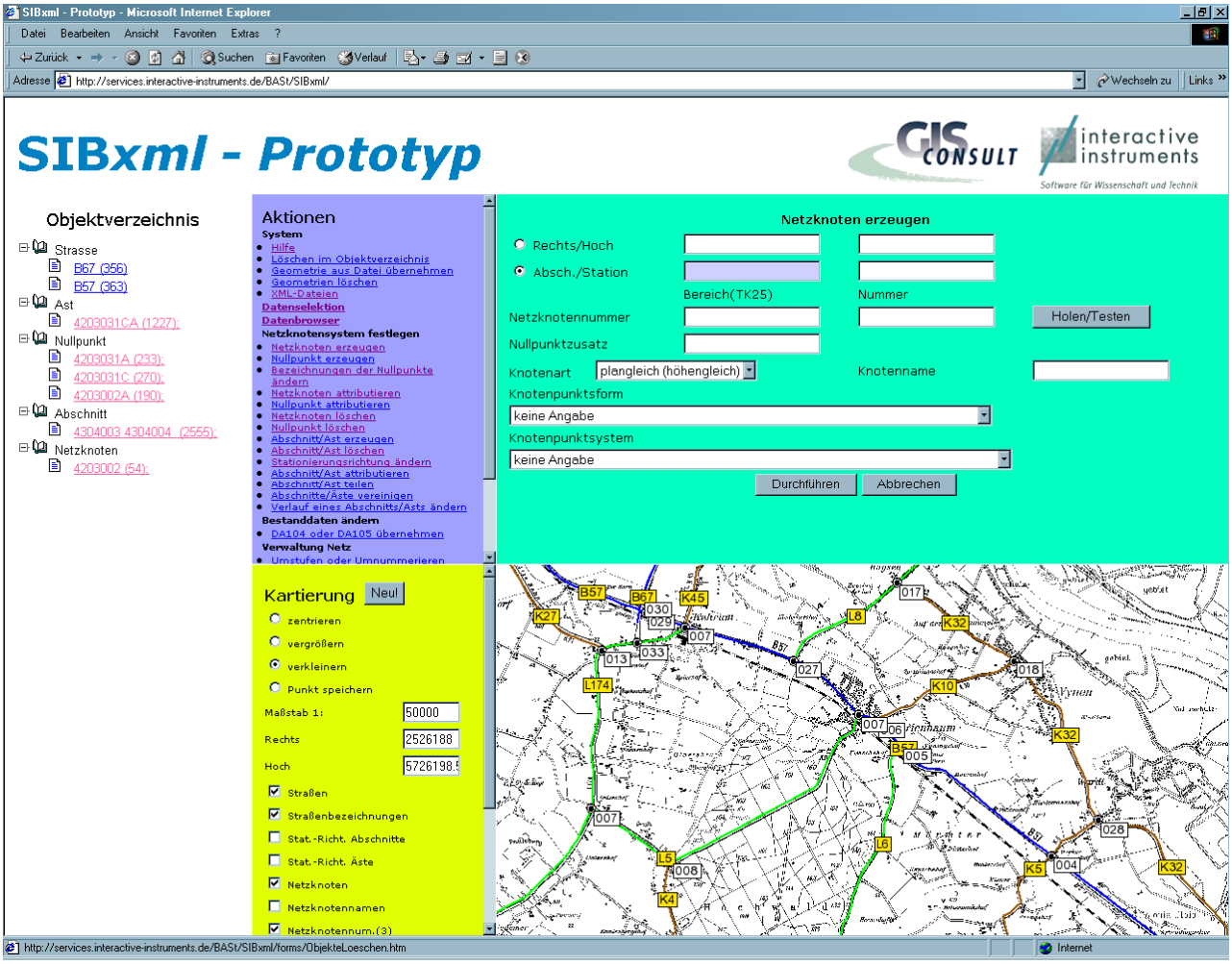
- Fehlende grafische Interaktion: Natürlich ist für viele der Interaktionen eine grafische Identifizierung von Objekten der angemessene Weg für eine Benutzeroberfläche, zumindest unter Produktionsbedingungen. Im Prototypen ist diese Art der Objektidentifizierung generell durch die Bezugnahme auf ein Verzeichnis bekannter Objekte ersetzt, das sog. „Objektverzeichnis". Als zusätzliche Hilfestellung kann außerdem die Position des Cursors im Grafikbereich bestimmt werden.
- Keine grafische Eingabe: Eine Unterstützung für grafische Eingabe von Geometrien war wegen des erforderlichen Aufwands nicht für den Prototypen vorgesehen. Um allerdings nicht nur mit vorgespeicherten Geometrien arbeiten zu müssen, wird auch eine textbasierte Eingabe von Geometrien unterstützt, nämlich in der Form von GML-Geoemtrien. Diese Geometrien werden in der OKSTRA®-Datenhaltung als „allgemeine Geometrieobjekte" abgelegt. Eine besondere Aktion erlaubt die Pflege (Übernahme und Löschen) dieser Geometrie-Objekte.
- Keine durchgängige Benutzung von fachlichen Identifikatoren: Unter Produktionsbedingungen müssten alle Beschreibungen von Fachobjekten die wesentlichen fachlichen Identifikatoren beinhalten (z.B. „Abschnitt 4709003O 4708004O"). Der für den Prototypen umgesetzte Client hingegen ist in erster Linie ein „technischer" Client, der in den einzelnen Aktionen die datenbanktechnischen Identifikatoren anstelle der fachlichen anzeigt (z.B. „Abschnitt.353246"). Lediglich im Objektverzeichnis finden die fachlichen Identifikatoren Verwendung.

Umgesetzt wurde der Client als Internet-Applikation auf der Basis von HTML-Frames[4]. Alle Frames besitzen spezifische Aufgaben und Inhalte, die Aufteilung bleibt während der Laufzeit der Applikation erhalten. Im Allgemeinen werden bei bestimmten Operationen immer nur bestimmte Frames ersetzt, sodass insgesamt der Eindruck des „Surfens im Internet" vermieden wird und vielmehr der einer logisch zusammenhängenden Applikation entsteht.

[4] Teilfenster des nutzbaren Bereichs im Browser. Frames können interakiv in ihrer Größe eingestellt werden.

Neben den beschriebenen Frames werden für zusätzliche Outputs zur Laufzeit noch weitere zusätzliche Browserfenster erzeugt. Dies geschieht immer bei Ausgaben, die nur zum Lesen bestimmt sind und keine weiteren Kontrollelemente enthalten, etwa bei der Ausgabe der „Längenstatistik".

Die folgende Abbildung zeigt den SIB-XML-Client mit den Bereichen „Objektverzeichnis", „Aktionsverzeichnis", „Aktionssteuerung", „Grafiksteuerung" und „Grafikbereich".



Eine ausführliche Beschreibung zum Aufbau des Clients sowie der verfügbaren Aktionen findet sich in Kapitel 4.3 des Feinkonzepts. **Die Beschreibung der Aktionen ist über die Aktion „Hilfe" auch im Client selbst verfügbar.**

Der Client ist über

http://www.okstra.de/Prototyp/SIBxml/

aufrufbar. Es wird ein Microsoft Internet-Explorer ab Version 5.5 benötigt.

Die Umgebung ist so eingerichtet, dass alle Veränderungen an den Daten nachts (um ca. 3 Uhr) auf den Anfangszustand zurückgesetzt werden.

3.6 Implementierung

Für den Prototypen wurden zwei Server eingerichtet (XMLPT1 und XMLPT2), die den Zugriff auf die Daten (mit Hilfe des WMS und WFS), die Client-Anwendung sowie die Fachservices ermöglichen.

Hardware

Die Rechnerkonfiguration besteht aus zwei PCs in einem heute üblichen, leistungsfähigen Ausbau. Wichtigste Kenngrößen:

- CPU Intel P4 2,4 GHz,
- RAM 1GB DDR-RAM,
- Festplatte 80,1 GB IDE,
- Netzwerkkarte,
- Betriebssystem Windows 2000.

Die CPU-Leistung ist für eine performante Arbeit der in Java programmierten, teilweise sehr komplizierten geometrieverarbeitenden Fachdienste erforderlich. WMS und WFS profitieren davon, kommen erfahrungsgemäß aber auch mit ca. 1,5 GHz aus.

RAM und Festplatte kämen für das kleine Testgebiet mit ca. 256MB und 20 GB aus, was im Wesentlichen durch die Bedürfnisse des Betriebsystems bedingt ist. Für Gesamt-NRW würden wir einen Speicherausbau (RAM) auf 1,5GB bis 2GB vorschlagen. (Allerdings ist die im Prototypen eingesetzte Version der Datenhaltung XTRA nicht für eine solche Datenmenge eingerichtet – eine entsprechende Erweiterung ist aber vorhanden.)

Die Datenpräsentationsschicht muss von ihren Bedürfnissen her serverseitig und clientseitig beleuchtet werden. Serverseitig gilt das eben Gesagte: Es ist eine hohe CPU-Leistung und ein Speicherausbau mit über 256 MB RAM und ca. 20 GB Festplatte erforderlich. Clientseitig gelten im Prinzip schwächere Vorgaben. Hier reichen CPUs ab 300 MHz – allerdings ist hierzu festzustellen, dass ein solch schwacher Rechnerausbau heute unüblich und generell behindernd ist, weil die meisten Applikationen (z.B. Office) erst mit mehr CPU-Leistung befriedigend arbeiten.

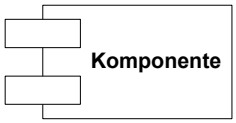
Software-Komponenten

Als Webserver kommt die Open-Source-Software „Apache" (http://www.apache.org) zum Einsatz, über den mittels CGI die XtraServer-Komponenten angesprochen werden. Auf dem XMLPT1 läuft der Web Feature Server für die Straßennetzdaten sowie der Web Map Server für die Visualisierung der Daten. Auf dem XMLPT2 hingegen läuft der Web Feature Service für die Bereitstellung der Bestandsdaten.

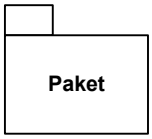
Als Add-on zum Apache-Server wurde auf beiden Computern der Java-Webserver „Tomcat" aus dem Jakarta-Projekt (http://jakarta.apache.org) installiert, bei dem es sich ebenfalls um Open Source handelt. Dieser Webserver stellt die Dienste für die Client-Applikation (auf dem XMLPT1) sowie die Fachdienste (auf dem XMLPT2) zur Verfügung, die als Java-HTTP-Servlets implementiert wurden.

Die folgende Abbildung in UML-Syntax (Komponentendiagramm) veranschaulicht diesen Sachverhalt.

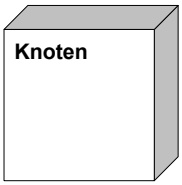
Verwendete UML-Symbole:



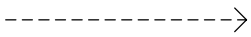
Abgrenzbares Objekt auf Implementierungs-niveau



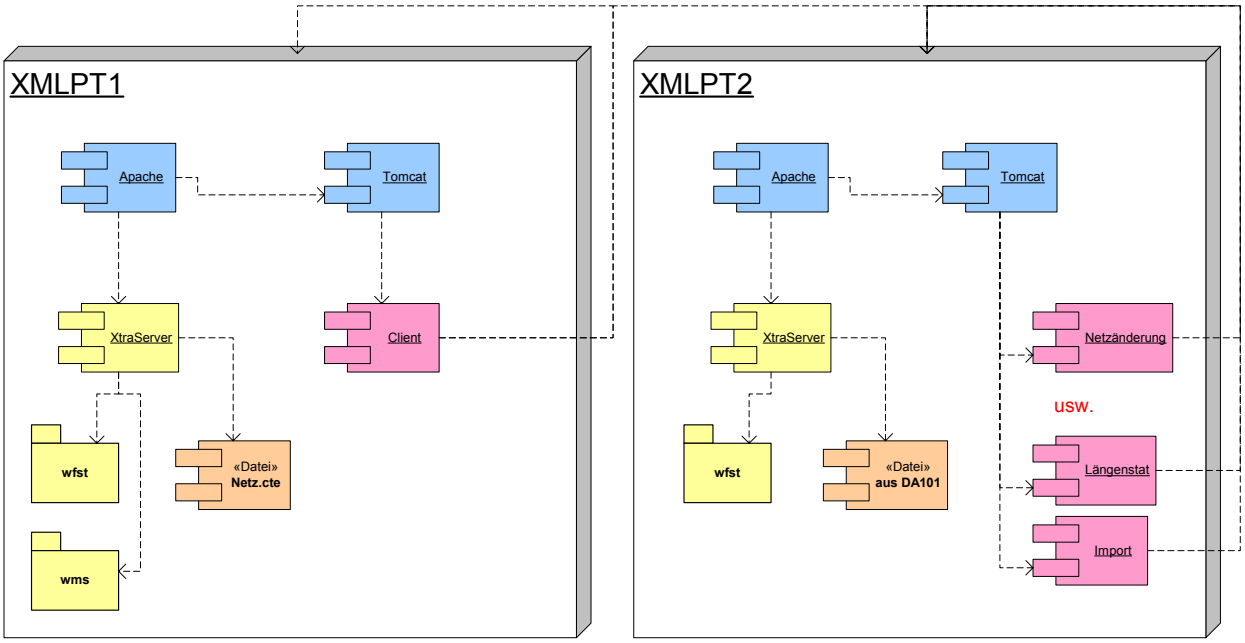
Logisches Objekt eines Funktions-zusammenhangs



Netzwerkknoten



Abhängigkeits-relation



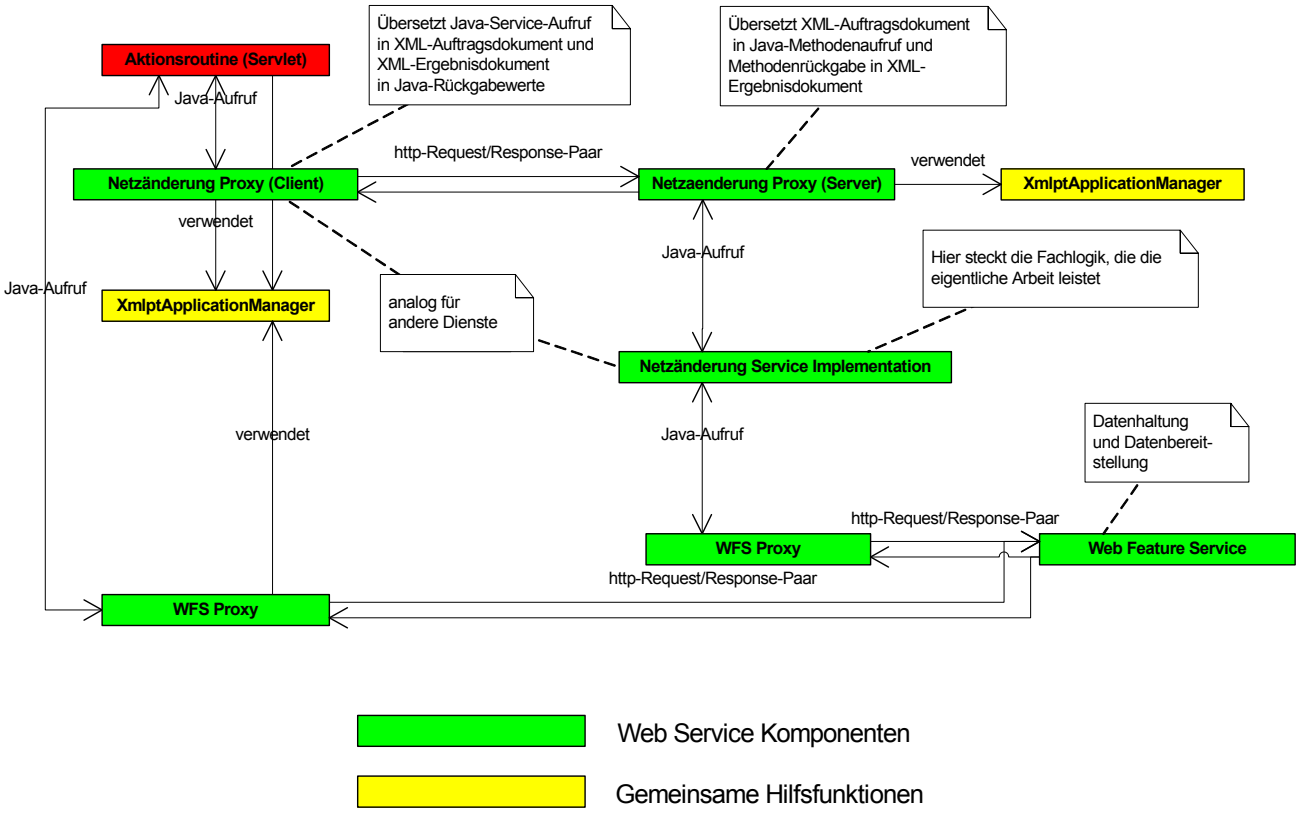
Die Farbgebung im Diagramm wurde wie folgt gestaltet:

- Blau – OpenSource Software
- Gelb – XtraServer
- Orange – Daten
- Rot – In Java für das Projekt implementierte Software

Eine Konfigurationsdatei (im Diagramm nicht sichtbar), die für alle Services sichtbar ist, definiert, auf welchem Server sich welche Daten bzw. Dienste befinden und wie diese zu adressieren sind (siehe auch Kapitel 4.3.1.3 des Feinkonzepts).

Sowohl ein Großteil der Client-Applikation sowie sämtliche Fachdienste wurden in Java implementiert und benutzen ein gemeinsames Framework, das die Exemplardefinitionen der verwendeten OKSTRA®-Schemata beinhaltet. Die Kommunikation zwischen den Services erfolgt mittels HTTP, unter Verwendung der im Feinkonzept definierten XML-Requests (siehe Kapitel 4.2 des Feinkonzepts).

Für den Aufruf der Fachdienste aus der Client-Oberfläche heraus wurden zudem weitere Dienste und Proxy-Klassen in Java definiert, die aus den HTML-Eingabemasken des Browsers die entsprechenden XML-Requests formulieren und das Ergebnis für den Browser aufbereitet zurückliefern. Das Zusammenspiel der einzelnen Komponenten zeigt die folgende Abbildung.



Zugriff auf die Quellprogramme

Alle Quellprogramme der für das Projekt XML-Prototype erstellten Software stehen zum Nachvollziehen technischer Einzelheiten im Internet zur Verfügung.

Die Struktur ist wie folgt:

http://www.okstra.de/Prototyp/SIBxml/forms/	enthält die HTML-Dateien und in den Unterverzeichnissen
Unterverzeichnisse ...	
css/	CSS-Stylesheets
script/	Javascript-Code für das Objektverzeichnis
xslt/	die XSLT-Stylesheets
Schemata/	alle benötigten XML-Schemata
http://www.okstra.de/Prototyp/SIBclientjava/	die Servlet-Klassen für den Client

http://www.okstra.de/Prototyp/SIBservicejava/	die Servlet-Klassen für die Applikations-Web-Services
http://www.okstra.de/Prototyp/SIBssharedjava/xmlpt/	alle Klassen für die Geschäftslogik der Dienste
Unterverzeichnisse ...	
altova/	Hilfsklassen für das XML-Java-Binding
gml/	Klassen für das XML-Java-Binding des GML-Schemas
importda/	Klassen für den Importdienst für die Querschnitts- und Aufbaudaten
laengenstatistik/	Klassen für den Längenstatistik-Dienst
netzaenderung/	Klassen für den Netzaenderungsdienst
nummernverwaltung/	Klassen für den Nummerbverwaltungsdienst
ogc/	Klassen für das XML-Java-Binding der OGC-Basistypen
okstra/	Klassen für das XML-Java-Binding des OKSTRA-Schemas
sib/	Klassen für die Geschäftslogik der Netzobjekte
verwaltungsmassnahme/	Klassen für den Verwaltungsmassnahmen-Dienst
wfs/	Klassen für die Java-Implementation des WFS-Interfaces

Um den Java-Code anzuzeigen, bindet man zweckmäßig den Dateityp .java im Windows-Explorer auf einen intelligenten Programmeditor, der Java formatieren kann. (Zur Not tut es auch der Notepad). Aus dem Internet-Explorer kann man dann die Java-Dateien unmittelbar öffnen.

3.7 Überprüfung am vorgegebenen Testfall

Der vorgegebene Testfall wurde mit Hilfe des SIB-XML-Clients durchgeführt und die korrekte Behandlung der Fachdaten kontrolliert. Die Funktionsweise des Prototypen wurde während der Abschlusspräsentation vor der PG 27 am 10.04.2003 in der BASt live über das Internet demonstriert.

In Kapitel 5 des Feinkonzepts werden darüber hinaus die Abhängigkeiten zwischen den einzelnen Services anhand des Testfalls ausführlich erläutert.

4 Gewonnene Erkenntnisse

Das Ziel des Projekts war der prototypische Nachweis der Eignung einer XML-Strukturierung für eine Straßeninformationsbank der Zukunft. Dieser ist in vollem Umfang gelungen, der Prototyp realisiert alle geforderten Testfälle, ist demonstrierbar und setzt durch und durch eine XML-basierte Dienstearchitektur um.

4.1 Übersicht

Die Frage, die in diesem Kapitel beantwortet werden soll, ist: Was haben wir aus Konzeption und Durchführung des Projekts XML-Prototyp gelernt?

Die im vorhergehenden Kapitel 2 beschriebenen Ergebnisse weisen die Einsatzfähigkeit der zu untersuchenden Technologien auch im Umfeld der Straßeninformationsbanken nach. Alle Ziele des Prototypen konnten voll erfüllt werden, die Testfälle ließen sich auf der Basis der Prototypen durchführen.

Über diese pauschale Erkenntnis hinaus haben sich im Detail aber eine Reihe weiterer Feststellungen ergeben, die in der nachfolgenden Tabelle in der Übersicht dargestellt und anschließend ausführlich diskutiert werden. In Kapitel 4 leiten wir aus diesen Erkenntnissen eine Reihe von Empfehlungen ab.

Positive Erkenntnisse / erreichte Ziele:

+ Nachweis der Umsetzbarkeit einer SIB in einer mehrschichtigen Dienstearchitektur
+ Verteilung der SIB-Datenhaltung auf unterschiedliche Systeme
+ Bestätigung der Tragfähigkeit von web- und internetbasierten Technologien und Produkten
+ Erarbeitung von XML-basierten Informationsflüssen in einer SIB
+ Erfolgreicher Einsatz von OKSTRA®-XML als Basis einer XML-basierten SIB
+ Bestätigung der Tragfähigkeit des internationalen Standards „Web Feature Service" als Basis für eine Datenbereitstellung von XML-Daten in einer SIB
+ Realisierung eines browserbasierten Thin-Clients[5] zu Inspektion und Pflege der SIB-Daten
+ Integration von Kartendarstellungen unter Verwendung des „Web Map Service"-Standards (einschließlich der Einbindung von extern angebotenen TK50-Karten)

Erkannte Nachteile / Probleme:

- Aufwändige Client-Entwicklung beim Einsatz von Thin-Clients
- Die verfügbaren Standards unterstützen die Datenintegration nur mit Einschränkungen (verteilte Abfragen über mehrere Datenbereitstellungskomponenten, verteilte Transaktio-

[5] Ein „Thin-Client" umfasst nur die Präsentationsschicht einer Applikation. Die Bezeichnung wird im Gegensatz zum „Fat-Client" benutzt, bei dem im Client auch andere, umfangreiche Teile der Applikation enthalten sind, z.B. die Applikationslogik oder sogar Teile der Datenbereitstellung. GIS-Arbeitsplätze mit zentraler Datenhaltung sind typische „Fat-Clients".

nen)

Offene Punkte / Anregungen:

- Entwicklung von definierten OKSTRA-Profilen für Anwendungsbereiche
- Beobachtung und ggf. Einflussnahme auf die Entwicklung relevanter Basisstandards
- Zukünftige Beachtung der grafischen Interaktion in Thin-Clients
- Die Auswirkungen der Historisierung sind in einer Produktionsversion zu beachten.
- Die Frage der Protokolle ist zu klären (z.B. SOAP)
- Authentifizierung, Autorisierung, E-Commerce-Aspekte

4.2 Bewertung des Web-Dienste- und XML-basierten Ansatzes

Der Prototyp realisiert alle geforderten Testfälle, ist demonstrierbar und setzt durch und durch eine XML-basierte Dienstearchitektur um. Es ist damit unter anderem gezeigt, dass auch das technisch anspruchsvolle Gebiet der Netzfortführung durch XML-basierte Dienste geleistet werden kann. Dabei wurde auch deren wesentliches Potential realisiert, nämlich die Verteilung des Systemzugriffs über einen Thin-Client-Ansatz und die Integration von Daten über das Netz.

4.2.1 XML-basierte Dienste

Kern der Architektur des XML-Prototypen ist ein dienstebasierter Ansatz. Ein Dienst („Service") bietet eine klar abgegrenzte Funktionalität, die über Schnittstellen angeboten wird. Eine Schnittstelle umfasst eine Menge von Operationen. Eine Operation besteht aus einem Namen und einer Anzahl von Parametern und realisiert eine verändernde oder eine abfragende Funktion im Rahmen der Schnittstelle. Die Angaben zum Aufruf einer Operation (d.h. die Eingabeparameter) werden dabei als XML-Dokument kodiert und ebenso auch das Ergebnis. Somit erhält man für jede Operation zwei XML-Dokumente, einmal den Auftrag („Request"), der an den Server geht, und zum anderen das Ergebnis („Response").

Wesentlich ist hierbei der Punkt, dass man sich auf einheitliche Strukturen für die XML-Dokumente einigt und somit die semantische Grundlage für eine Kommunikation schafft[6]. Die Struktur von XML-Dokumenten wurde im XML-Prototypen konsequent unter Verwendung von XML Schema beschrieben. Entsprechend gibt es i.d.R. zwei XML-Schema-Dokumente pro Operation, eines für den Aufruf und eines für das Ergebnis.

Die Modellierung der XML-Schema-Strukturen zu den Fachobjekten basiert im XML-Prototyp auf OKSTRA®-XML. Dieses wird durch die Datenbereitstellungskomponente erzeugt, für deren Schnittstelle eine international standardisierte Dienstebeschreibung eingesetzt wurde, die genau auf den beschriebenen Prinzipien beruht: Dem OGC Web Feature Service. Tatsächlich festzulegen waren daher im XML-Prototypen nur die fachlichen Dienste der Datenaufbereitungsschicht.

[6] Das Forschungsprojekt zum objektorientierten OKSTRA® zielt u.A. auf die Bestimmung der zu diesem Zweck aufzudeckenden oder definierenden Sachverhalte ab.

Man beachte das auf diese Weise erreichte Ergebnis: Die Funktionen einer XML-basierten SIB werden vollständig definiert – ohne dass man sich auf ein konkretes Protokoll zur Realisierung festgelegt hat. Dies ist ein wichtiger Punkt, da zurzeit noch unklar ist, welche Web-Service-Übertragungsprotokolle sich tatsächlich in der Praxis durchsetzen werden (http/POST, SMTP, SOAP über http, SOAP über SMTP, usw.). Mit dem dargestellten Ansatz ist man flexibel gegenüber Anpassungen, für das Prototyping musste man sich allerdings für jeden realisierten Dienst auf ein Protokoll festlegen – es wurde http/POST verwendet[7].

Der beschriebene Ansatz macht massiven Gebrauch von den Hauptvorteilen des XML, nämlich der durch XML realisierten Trennung von

- Struktur,
- Inhalt,
- Format.

Die Dienste wurden als reine Struktur-Beschreibungen in XML Schema festgelegt. Der konkret beim Aufruf der Dienste transportierte Inhalt besteht aus XML-Dokumenten, die stets gegen die festgelegten Schema-Beschreibungen validiert werden.

Im Verkehr der Dienste untereinander spielt nur der Inhalt eine Rolle. Der Aspekt der Formatierung wird nur bei der Datenpräsentation (im Client) wichtig, wenn die Inhalte gegenüber Benutzern dargestellt werden müssen. Hier werden die entsprechenden XML-Werkzeuge wie XSL/XSLT eingesetzt, um menschlesbare Formatierungen in der gewohnten Darstellung zu erzeugen. Teilweise (wenn die Browser das zulassen) werden die XML-Dokumente sogar bis in den Browser übertragen, der dann anhand der Formatierungsbeschreibung (XSL-Stylesheet) die Darstellung vornimmt.

4.2.2 Die Vorteile von Web-Diensten

Durch Web-Dienste können im Wesentlichen zwei Vorteile realisiert werden:

1. Die einfache Verteilung von Daten und Diensten
2. Die Integration von Daten und Diensten aus verschiedenen Quellen

Beide Vorteile erwachsen aus den Eigenschaften und dem allgegenwärtigen Vorhandensein des Internet/Intranet und seiner Infrastruktur wie Web-Browsern. Sie werden beide im XML-Prototypen demonstriert.

Die Verteilung der Daten und Dienste macht sich üblicherweise am Begriff des *Thin-Client* fest, worunter i.A. eine im Browser abrufbare Applikation verstanden wird, die als Datenpräsentationsschicht auch im XML-Prototypen implementiert wurde. Um das Potential richtig einzuschätzen, muss man sich aber zusätzlich vergegenwärtigen, dass in Wirklichkeit alle implementierten Dienste an der Verteilung im Netz teilnehmen. Besonders die über OGC Web Feature Server bereitgestellten datennahen Dienste können infolge der Standardisierung des Interface ohne weiteres die Basis anderer Verfahren sein und möglicherweise sogar Nutzungen außerhalb der Straßenbauverwaltungen zulassen.

Die Integration von Daten kann räumlich und fachlich vollzogen werden. Bei räumlicher Integration werden gleichartige Daten verschiedener Gebiete aus mehreren Datenquellen integriert, bei fachlicher Integration hat man es mit einander ergänzenden Daten desselben Gebiets zu tun.

---

[7] http/POST wurde gewählt, weil es das einfachste der in Frage kommenden Protokolle ist und weil auch die OGC-Spezifikationen z.Z. auf dieses Protokoll ausgerichtet sind.

Im XML-Prototypen wurde nur die fachliche Integration erprobt, die allerdings auch die technologisch schwierigere Variante darstellt. **Grundvoraussetzung für beide Formen der Integration ist ein einheitliches und durchgängiges Datenschema, im Falle des XML-Prototypen war dies der OKSTRA®.** Zur Repräsentierung von Daten wurde daher konsequent OKSTRA®-XML eingesetzt.

Ein wichtiger Aspekt ist auch die umgesetzte Mehrebenenarchitektur, wobei hier vor allem die Ebene der Datenaufbereitung zu nennen ist.

Diese Ebene implementiert Dienste, die die eigentlichen fachlichen Prozesse einer SIB umsetzt. Sie bezieht ihre Information ausschließlich über XML-basierte Dienste und stellt diese fachlich aufbereitet („veredelt") wieder durch einen Dienst zur Verfügung. Auf diese Weise wird aus geeigneten Abfragen an die Basisdaten z.B. eine Längenstatistik, also ein Informationsprodukt.

In Wirklichkeit ist die Datenaufbereitung keine einheitliche Ebene, sondern sie besteht aus einer Reihe von Web-Diensten, die sich (über das Web) gegenseitig nutzen. Dies zeigt prototypisch den Nutzen von „Mehrwertdiensten", die auf den bereitgestellten Daten aufsetzend veredelte Information und Dienstleistungen anbieten. Tatsächlich wären einige der implementierten fachlichen Dienste auch für Nutzungen geeignet, die nicht die Netzfortführung betreffen, wie sie im XML-Prototypen schwerpunktmäßig realisiert wurde.

4.3 Erfahrungen mit OKSTRA® und OKSTRA®-XML

Die XML-Repräsentierung des OKSTRA® ist als GML-Anwendungsschema entwickelt worden, weil die Geography Markup Language (GML) die leistungsfähigste und am Markt am meisten akzeptierte zur Zeit existierende standardisierte XML-Sprache zur Darstellung von Sachverhalten mit Raumbezug ist. GML ist auch die bevorzugte Repräsentierung von Information in einem Web Feature Server (WFS). Daher war OKSTRA®-XML für den Einsatz im WFS prädestiniert.

Im Prototyp bestätigte sich das gute Zusammenwirken von OKSTRA®-XML und WFS.

OKSTRA®-XML kann problemlos durch einen WFS nachgewiesen werden und eignet sich für sinnvolle Abfragen und Fortführungsoperationen.

Neuland wurde auch bei der Programmierung von Datenaufbereitung und Datenpräsentation beschritten. Hier wurde der OKSTRA® nicht nur als Datenaustauschdefinition eingesetzt sondern direkt als Objektmodell, in dem die Repräsentierung der Daten für die „innere Applikationslogik" vollzogen wurde.

Auch hier hat sich der OKSTRA® grundsätzlich bewährt. Allerdings ist der OKSTRA® ein flexibler und deshalb komplexer Standard. Software-Erstellung in diesem Modell kann fallweise kompliziert sein. Hier bietet sich die Verwendung eines Profils[8] an, das einige „besonders flexible" Lösungen des OKSTRA® wieder vereinfacht in dem es bestehende Freiheitsgrade einschränkt. Z.B. ist der Umgang mit dem Beziehungsgeflecht bei Strecken- und Bereichseigenschaften durch Normalisierungen stark vereinfachbar (eine Streckeneigenschaft besitzt immer eine Strecke, diese immer einen oder mehr Teilabschnitte und diese kennen ihre Abschnitte).

Durch die Abbildung des OKSTRA®-Schemas entsprechend fester Regeln auf GML/XML-Schema, das bekanntlich nur einfache Vererbung zulässt, sind auf der anderen Seite einige praktische Strukturierungen des OKSTRA® verloren gegangen. Beispielweise wurde die gemeinsame Abstraktion Abschnitt_oder_Ast, die im OKSTRA®-XML-Schema entfallen ist, vermisst, weil sich an dieser Stelle

[8] Unter einem „Profil" wird eine genau definierte Teilmenge eines Schemas verstanden.

praktisch das gesamte Verhalten von Abschnitt und Ast versammelt. Hier sollten aus dem XML-Prototypen Rückflüsse in OKSTRA®-XML erfolgen.

4.4 Datenbereitstellung durch OGC Web Feature Server

Die Datenhaltung unterliegt der Vorgabe, dass sie konzeptionell keinerlei Einschränkung bezüglich der Alternativen bei Betriebssystemen, Datenbankmanagementsystemen oder der Bandbreite zwischen verteilter und zentraler Datenhaltung aufweisen darf. Eine solche Vorgabe lässt praktisch nur die vollständige Kapselung der Datenhaltung zu, d.h. die anderen Komponenten „reden" mit der Datenhaltung nur über die Datenbereitstellung.

Für die Datenbereitstellung wurde kein neuer Web Service festgelegt, sondern es wurde eine bereits existierende Definition herangezogen. Dabei handelt es sich um die Spezifikation des Open GIS Web Feature Service (WFS) Version 1.0.0. Die Erkenntnisse aus dieser Entscheidung und die Erfahrungen mit der Anwendung dieses Standards sind das Thema dieses Abschnitts.

4.4.1 Aspekte der Kapselung durch den WFS

Durch die vollständige Kapselung der Datenbasis wird die Unabhängigkeit von der Datenhaltungsumgebung erreicht. Die tatsächliche Datenhaltung wird dadurch bezüglich der Architektur irrelevant.

Hierzu eine Bemerkung, um nicht missverstanden zu werden: Durch die Kapselung wird die tatsächlich eingesetzte Datenbasis natürlich nicht unwichtig. Irrelevant ist diese nur in Bezug auf das Design und in Gesamtsicht. Für einen speziellen Fall gibt es meist gute Gründe, ein bestimmtes DB-system einzusetzen, z.B. Vorhandensein, Strategiefestlegungen, Vertrauen, usw. Gerade deshalb muss das Konzept so geartet sein, dass es mit allen Datenhaltungen „auskommt".

Dadurch, dass für die Kapselung ein internationaler Standard (der WFS) eingesetzt wurde, wird ein zweifacher Zusatznutzen erzielt:

1. Die Chancen stehen gut, dass für bestimmte Datenhaltungsumgebungen bereits fertige oder anpassbare Lösungen zur Verfügung stehen und daher der kapselnde Dienst nicht für jede Datenhaltungsumgebung eigens entwickelt werden muss. Beispiele für bestehende WFS-Lösungen:
   - XtraServer/XtraWFS, siehe unten
   - ESRI bietet den OGC WFS Connector for ArcIMS
   - Ionic Software bietet einen WFS für ORACLE 8i/9i spatial an
   - Intergraph bietet einen Adaptor for GeoMedia an
   - Galdos bietet einen WFS auf Basis von X-Hive (einem XML-DBMS) an
2. Es wird Kompatibilität mit „Geo-Daten-Infrastrukturen" hergestellt, die fast alle auf den OGC-Standards basieren. Auch die AFIS®-ALKIS®-ATKIS®-Lösung der Vermessungsverwaltungen der Länder basiert auf GML und ist daher WFS darstellbar.

Eingesetzt für den XML-Prototypen wurde XtraServer/XtraWFS von interactive instruments. Der Vorteile dieser Lösung für den XML-Prototypen lagen vor allem darin, dass hier bereits ein fertiger, für den OKSTRA® geeigneter WFS vorlag und dass dieser von seinem Leistungsumfang her durch den AN dem Projekt angepasst werden konnte. Da das Projekt ja unter anderem auf die Ermittlung des erforderlichen Leistungsprofils des WFS abzielte, hätte ein Verfehlen eines wichtigen Leistungsmerkmals bei einem Dritt-WFS das Scheitern der Implementierung bedeutet. Tatsächlich

mussten im Verlauf des Projekts einige kleinere, nicht vorhergesehene Leistungsmerkmale in XtraServer/XtraWFS integriert werden.

4.4.2 Das erforderliche Eigenschaftsprofil

Es zeigt sich, dass die Eigenschaften, die ein WFS mitbringen muss, im Wesentlichen durch die Struktur des OKSTRA® bestimmt sind und nicht durch besondere Anforderungen der im XML-Prototypen zu schaffenden Funktionalität. Es wirkt sich aus, dass der OKSTRA® ein komplexer, objekt-basierter Standard mit einem sehr hohen Grad an relationaler Verlinkung ist.

Zuallererst ist das positive Resultat festzuhalten, dass der WFS in der Tat die Funktionalität bereitstellt, die von den ihn verwendenden fachlichen Diensten und vom Client abverlangt wird. Hierzu zählen die erforderlichen Datenselektionen und Veränderungen, wobei besonders aus der Sicht des Clients die geometrischen Selektionsfähigkeiten wichtig sind.

Plakativer ausgedrückt: Der WFS ist als „OKSTRA-Web-GIS-Engine" geeignet.

Es ist allerdings auch festzuhalten, dass das Konzept bis an die Grenze beansprucht wird und teilweise (zumindest im heutigen Stand der WFS-Spezifikation) darüber hinaus.

Netzwerklaufzeiten und Verlangsamungen durch die „gesprächige" XML-Repräsentierung sind natürlich zu beobachten, sie führen aber zumindest in der Kommunikation mit dem WFS zu keinen unakzeptablen Effekten. Wie auch später nochmals deutlich werden wird, ist es dafür erforderlich, dass „dienste-nutzende" Software „kommunikationsbewusst" erstellt wird und die Kommunikation in möglichst großen Blöcken zusammenfasst. Die WFS-Spezifikation unterstützt solche Zusammenfassungen sehr wirkungsvoll.

Für den vollen Funktionsumfang des XML-Prototypen wird nahezu ein Vollausbau[9] des WFS benötigt. Das bedeutet:

- Transactional-WFS mit Locks, d.h. es werden nicht nur die schreibenden Zugriffe benötigt sondern auch die von der Spezifikation vorgesehenen Mittel zur Serialisierung.
- Das Filter-Encoding muss nahezu komplett realisiert sein. Fehlen kann nur redundant spezifizierte ausgelegte Funktionalität und möglicherweise ein Teil der Geometriefilterung.

Für eine Funktionalität unterhalb des Niveaus des XML-Prototypen gelten die obigen Aussagen selbstverständlich nur eingeschränkt. Für ein reines Auskunftssystem sind z.B. die schreibenden Funktionen nicht erforderlich, sodass man hier mit einem Basic-WFS auskommt.

Die Geometriefilterung wird im XML-Prototypen ausschließlich durch den Client benutzt. Dies dürfte sich beim Übergang zu einer Produktionsversion ändern, da der OKSTRA® auch die Nutzung geometrischer Beziehungen vorsieht. Auch dürfte durch einen Client in Produktionsreife gerade die Ausnutzung der geometrischen Suchfunktionalität einen Schwerpunkt bilden.

Es werden einige Eigenschaften benötigt, die den Standard über die in ihm benannten Funktionalitätsoptionen hinaus ausdehnen. Sie sind in der Spezifikation z.Z. in keiner Ausbaustufe gefordert.

Unterstützung des OKSTRA®-XML Schemas

[9] Die WFS-Spezifikation lässt spezifikationskonforme Implementierungen unterhalb des Vollausbaus zu. Ein durch eine Operation (GetCapabilities) abrufbarer Eigenschaftsnachweis gibt Auskunft darüber, was ein WFS kann und was nicht. Typische Profile des WFS haben auch eigene Namen, z.B. Basic-WFS, Transactional-WFS.

Die Spezifikation des WFS verlangt nicht, dass jede konforme Implementierung des Standards alle möglichen durch GML-Anwendungsschemata definierte Datenstrukturierungen bedienen kann.

Für die Anwendung des WFS für die Zwecke der hier vorgeschlagenen Architektur muss gefordert werden, dass der WFS wenigstens das OKSTRA®-Schema unterstützt.

In diesem Zusammenhang muss auch noch auf ein kleineres Versionsproblem hingewiesen werden:

Der aktuelle WFS-Standard Version 1.0.0 referiert die immer noch gültige aber inzwischen veraltete Version 2.1.2 des GML. Anfang 2003 wurde jedoch die Version GML 3.0 verabschiedet und OKSTRA®-XML basiert bereits auf dieser.

Es ist davon auszugehen, dass zukünftige Versionen der WFS-Spezifikation auch auf dieser GML-Version beruhen werden. In der Praxis sollte das Versionsproblem nach unserer Einschätzung kein Problem darstellen.

XLink-bewusste Eigenschaftsadressierung

Da der OKSTRA® zwischen seinen Features eine Vielzahl relationaler Verknüpfungen aufweist, muss es möglich sein, bei der Eigenschaftsadressierung über Relationen „von Feature zu Feature" zu springen. Im GML-Anwendungsschema des OKSTRA® sind diese Relationen (GML-gemäß) durch XLink-Referenzen dargestellt.

Ohne eine XLink-bewusste Eigenschaftsadressierung wäre es z.B. nicht möglich, Abschnitte auszuwählen, die zu Bundesstraßen gehören, da hierfür vom Abschnitt der Sprung zur Straße erfolgen muss, die dann ihrerseits (entweder eingebettet oder wiederum durch eine Relation) die Straßenklasse kennt.

Man sieht: Diese Erweiterung ist unabweisbar – ohne sie ist der OKSTRA® in einem Web Feature Server nicht darstellbar.

Tatsächlich ist es so, dass hier einfach die WFS-Spezifikation nicht auf der Höhe der Möglichkeiten von GML ist und in der Tat steht die Lösung dieses Problems als erster Punkt unter „vi. Future Work" in der laufenden WFS-Spezifikation OGC 02-058. Eine Lösung innerhalb des Standards ist also in einer der nächsten Versionen zu erwarten; es gibt bereits konkrete Änderungsvorschläge, wie eine entsprechende Unterstützung spezifiziert werden kann.

In XtraServer wurde das Problem (nicht erst für den XML-Prototypen) so gelöst, dass in den eigenschaftsadressierenden XPath-Ausdrücken XLink-Verfolgungen einfach so behandelt werden, als stünde das referierte Objekt eingebettet an der Stelle der Referenz, d.h. wir unterscheiden nicht zwischen Einbettung (die in XPath wohldefiniert ist) und Durchlaufen einer Referenz.

Der Zugriff auf die Straßennummer vom Abschnitt aus sieht in XtraServer deshalb so aus:

```
Abschnitt/gehoert_zu_Strasse/
Strasse/hat_Strassenbezeichnung/Strassennummer
```

Der zweite (rote) Schrägstrich markiert den Übergang entlang der Relation.

Wir erwarten, dass die Lösung in der Spezifikation in diesem Sinne erfolgen wird, zumal die meisten uns bekannten WFS-Implementierungen ebenfalls diesem Ansatz folgen.

Behandlung multipler Zusammenhänge

Diese Problematik geht eigentlich auf dieselbe Ursache zurück, nämlich dass der WFS-Standard noch nicht ganz zu den Möglichkeiten von GML aufgeschlossen hat.

GML schränkt in der Modellierung von Applikationsschemata die Möglichkeiten von XML-Schema nur unwesentlich ein, sodass Mengen und Listen von geschachtelten Attributwerten und relationalen Beziehungen ausgedrückt werden können. Ohne diese Möglichkeiten wäre die komplexe Welt des OKSTRA® auch nicht darstellbar.

Bei der Eigenschaftsadressierung ist daher mit dem Vorhandensein mengenhafter Relationen und Attributwerte umzugehen. Dies ist im heutigen WFS nur unzureichend der Fall. Es können in der jetzigen Spezifikation nur einzelne, durch Konstante bezeichnete Mengenelemente ausgewählt werden, z.B. durch `Strasse/hat_Abschnitt_oder_Ast[3]` der dritte Abschnitt oder Ast.

Auch dieser offene Punkt ist unter „vi. Future Work" in der laufenden Spezifikation angesprochen und wir erwarten daher eine (standardisierte) Lösung in absehbarer Zeit.

Unerwarteterweise hat diese Problematik im XML-Prototypen keine wirkliche Rolle gespielt. Die Zugriffe liefen überall entlang eindeutiger Relationen. Wir erwarten aber, dass in einem Produktionssystem diese Problematik alleine wegen Beachtung der Historisierung eine Rolle spielen wird.

Es ist sehr wahrscheinlich, dass zukünftige Spezifikationen des WFS, die dieses Problem lösen, die hierfür vorgesehene XPath-Syntax anwenden werden. Um z.B. von einem Nullpunkt zu dem Netzknoten zu navigieren, zu dem er erst seit dem 10.04.2003 gehört, um dessen Knotenpunktsform zu ermitteln, würde man sagen:

```
Nullpunkt/in_Netzknoten[gueltig_von>="2003-04-10"]/
Netzknoten/Knotenpunktsform/@Kennung
```

In XtraServer sind diese Vorgriffe bereits implementiert. Sie wurden aber im XML-Prototypen nicht eingesetzt.

Andere Konsequenzen aus der Behandlung mengenwertiger Relationen und Attribute sind heute weniger präzise vorhersagbar, z.B. der Vergleich von mengenhaften Werten. Hier ist geboten, die Standardisierungsprozesse zu beobachten und im Sinne der Einsetzbarkeit für den OKSTRA® zu beeinflussen.

Automatische Referenzgenerierung beim Insert

Durch die starke relationale Verknüpfung des OKSTRA® müssen bei praktisch allen Features Referenzen zu weiteren Features eingefügt werden. Diese Referenzen werden durch XLink-Verweise ausgedrückt, die die Feature-Ids der referierten Objekte enthalten.

In einer Welt, in der Netzwerklaufzeiten keine Rolle spielen, kann man hierbei so vorgehen, dass zuerst das referierte Objekt erzeugt wird und, nachdem dann seine Feature-Id bekannt ist, das referierende Objekt. Dazu sind zwei Transaktionen gegenüber dem WFS abzusetzen. Alternativ können die beiden Objekte auch zusammen erzeugt werden und in einer zweiten Transaktion die Relation explizit überschrieben werden. Auch in diesem Fall sind zwei Transaktionen nötig.

Man kann sich leicht vorstellen, was passiert, wenn an diesem Szenario mehr als zwei Features beteiligt sind. Fazit: Ein solches Vorgehen führt zu unakzeptablem Laufzeitverhalten – die Objekte sind alle durch eine Transaktion zu erzeugen.

Hierfür braucht man die Fähigkeit des WFS „lokale Verweise" innerhalb einer Transaktion zu berücksichtigen. Diese Fähigkeit wird von der gegenwärtigen WFS Spezifikation nicht explizit gefordert, werden aber durch GML und dessen Verwendung des XLink-Standards nahegelegt.

Vereinfachtes Beispiel:

```
<wfs:Transaction>
  <wfs:Insert>
    <okstra:Netzknoten fid="marke1">
```

```
        ...
      </okstra:Netzknoten>
      <okstra:Nullpunkt>
        ...
        <okstra:in_Netznoten xlink:href="#marke1"/>
      </okstra:Nullpunkt>
    </wfs:Insert>
  </wfs:Transaction>
```

Die Relation „in_Netzknoten" vom Nullpunkt zum Netzknoten wird im Beispiel durch einen lokalen Verweis ausgedrückt. Lokale Verweise werden durch den WFS in „wirkliche" Relationen zwischen „wirklichen" Feature-Ids übersetzt. Fragt man die beiden Features des Beispiels später ab, würde das Ergebnis wie folgt aussehen:

```
<wfs:FeatureCollection>
  <okstra:okstraObjekt>
    <okstra:Netzknoten fid="Netzknoten.7350">
      ...
    </okstra:Netzknoten>
    <okstra:Nullpunkt fid="Nullpunkt.4488625">
      ...
      <okstra:in_Netznoten xlink:href="Netzknoten.7350"/>
    </okstra:Nullpunkt>
  </okstra:okstraObjekt>
</wfs:FeatureCollection>
```

4.4.3 Wünschenswerte Verbesserungen beim Query-Konzept

Die Filterung von Features zur Abfrage (GetFeature) und Fortführung (Transaction/Update und Transaction/Delete) erfolgt auf der Basis des sog. „Filter Encoding", ein OGC Standard für eben diese Selektionsaufgaben.

Das Konzept des Filter-Encoding erwies sich im XML-Prototypen als ausreichend, aber nicht sonderlich komfortabel. Der Grund für diese Beurteilung ist, dass ein Filter, so wie er im WFS eingesetzt wird, immer nur genau einen Feature-Type selektieren kann. Dieses geschieht durch die logische Kombination räumlicher und skalarer Vergleiche auf den Attributen des zu filternden und damit relational verbundener Features. Nicht möglich sind verbundene Abfragen, wie sie z.B. in SQL in der Form von „Views" und „Sub-Queries" gestattet sind.

Im XML-Prototyp werden daher in diesen Fällen mehrere Abfragen gestartet, die dann im Client bzw. in den fachlichen Diensten aufwändig kombiniert werden müssen.

Es ist allerdings zu erwarten, dass das Filter Encoding in Zukunft durch den Standard XQuery ergänzt oder sogar ersetzt wird. XQuery befindet sich im Standardisierungsprozess beim W3C (Status: Working Draft). Der zukünftige Standard ist in seiner Abfragekraft gut mit SQL zu vergleichen. Insofern werden die genannten Einschränkungen der WFS-Spezifikation in absehbarer Zeit überwunden werden.

XQuery wird für die OGC-Standards nicht *genauso wie spezifiziert* eingesetzt werden können, weil in XQuery der Bereich der räumlichen Vergleiche fehlt. Hier sind Ergänzungen im Rahmen des vorgesehenen Erweiterungsmechanismus von XQuery erforderlich. Die Einrichtung einer Arbeitsgruppe für *Spatial/Temporal Query Languages* ist für Juni 2003 vorgesehen.

4.4.4 Wünschenswerte Verbesserungen bei Transaktionskonzept und Serialisierung

Die WFS-Spezifikation besitzt nur relativ schwach ausgeprägte Hilfsmittel zur Serialisierung konkurrierender Fortschreibungsoperationen, nämlich:

- Die atomare (also ganz-oder-gar-nicht) Ausführung zusammengesetzter Insert/Update/Delete-Operationen. (Deshalb heißt diese WFS-Operation auch „Transaction".)
- Die explizite Isolation der Zugriffe auf einzelne Features durch „Locks"

Mit diesen Hilfsmitteln ist die gesicherte Fortführung von Datenbeständen im Große und Ganzen möglich.

Programmiertechnisch ist die Nutzung der vorhandenen Serialisierungshilfsmittel sehr aufwändig. Tatsächlich wurden sie deshalb im Prototypen auch nur exemplarisch (beim Import-Dienst) eingesetzt.

Durch das Fehlen operationsübergreifender Transaktionen muss die Programmierung eines transaktionssicher ändernden Dienstes so erfolgen, dass zuerst alle beteiligten Features unter Vergabe eines (!) Locks (also durch ein GetFeatureWithLock oder ein LockFeature) geholt und gelockt werden müssen. Sobald dies erfolgreich geschehen ist, können die vorgenommenen Veränderungen durch eine (!) Transaction-Operation unter Bezugnahme auf dieses Lock zurückgeschrieben werden.

Man sieht sehr leicht, dass dies eine recht aufwändige Art der Programmierung darstellt, obwohl natürlich die Beschränkung auf zwei Operationen wegen der Netzwerklaufzeiten auch wünschenswert ist. Auch können so programmierte Dienste nur unter Aufgabe der Transaktionssicherheit miteinander zu neuen Diensten kombiniert werden. Sobald die erste aufgerufene Operation erfolgreich abgeschlossen wurde, gibt es bei einem späteren Auftreten eines Fehlers (z.B. ein entscheidendes Feature ist durch einen anderen Prozess gesperrt) keine Möglichkeit mehr die Wirkungen der ersten Operation zurückzurollen.

Das Vorhandensein operationsübergreifende Transaktionen ist deshalb für die effektive Umsetzung von fachlichen Diensten „über" dem WFS sehr wünschenswert. Auch hier ist es geboten, die künftige Entwicklung des Konzepts WFS zu beobachten und zu beeinflussen.

4.4.5 Mögliche Alternativen

Da die Entscheidung für den OGC Web Feature Service im Projekt XML-Prototyp eine zentrale Rolle einnimmt, sollen hier nochmals mögliche Alternativen diskutiert werden, die evtl. geeignet wären, an die Stelle des Web Feature Service zu treten.

Selbstdefinierte Datenbereitstellungsdienste

Natürlich wäre es immer möglich, eigene Definitionen für einen geeigneten Abfrage- und Fortführungsdienst zu bilden. Es ist allerdings zweifelhaft, ob ein solcher Dienst substantiell anders aussehen kann als der WFS, wenn man davon ausgeht, dass dieser „freie" Abfragen unter Einschluss von Geometrie und Fortführung leisten soll. Man könnte bei einer eigenen Definition die Probleme, die die WFS-Spezifikation in ihrer heutigen Form aufweist, vermeiden.

Auf der anderen Seite verliert man so die durch den WFS vorhandene Kompatibilität mit Geo-Dateninfrastrukturen und den Zugriff auf Fertigprodukte (sowohl client- als auch serverseitig!). D.h. selbstdefinierte Dienste und Applikationen wären immer für alle Datenbankumgebungen in eigener Regie zu erstellen.

XML-Databases mit XQuery als Abfragesprache

Dies wäre eine radikale Hinwendung zur XML-Technologie, die aber in offensichtlichem Widerspruch[10] zu den Zielen des Projekts steht, die eine völlige Unabhängigkeit von Datenbankumgebung und Verteilungsstrategie fordern. Darüber hinaus werden zurzeit in den Straßenbauverwaltungen in der Regel RDBMS-Produkte eingesetzt, und es ist nicht zu erwarten, dass sich dies in absehbarer Zeit ändern wird.

Es ist dabei auch zu bedenken, dass durch XQuery nur die Abfrageseite und nicht die Fortschreibung gelöst wird. Auch gibt es in XQuery bisher keinerlei Unterstützung für geometrische Abfragen, was bei einem so stark raumbezogenen Schema wie dem OKSTRA® unerlässlich erscheint.

SQL/MM spatial

Der SQL99 Standard spezifiziert in seinem „Spatial"-Teil Operationen, die die geforderte Funktionalität erfüllen, also die Kombination von räumlichen und skalaren Queries und die erforderliche Unterstützung für die Fortschreibung. Auch steht bekanntlich der OKSTRA® in einem SQL-Schema zur Verfügung.

Nur – es handelt sich weder um XML noch um einen Web-Service. Beides müsste erst definiert werden, was beim Umfang der SQL99-Definition sicher aufwändiger als die WFS-Spezifikation würde. Weiterhin fehlen zumindest im Augenblick die verfügbaren Produkte, die diesen Standard realisiert haben.

LDAP (Lightweight Directory Access Protocol)

LDAP definiert einen Verzeichnisdienst, der aufgrund seiner flexiblen Anlage im gewissen Umfang auch zur Speicherung von Objekt-Information (also als Datenbank) geeignet ist. Dies geht besonders dann, wenn die Daten eine hierarchische Struktur aufweisen. LDAP erlaubt auch Queries nach inhaltlichen Kriterien.

Für den OKSTRA® scheint LDAP nicht besonders gut geeignet zu sein, da der OKSTRA® stark objekt-relational (und eben nicht hierarchisch) organisiert ist und weil es in LDAP keine Unterstützung für die Darstellung und Abfrage geometrischer Eigenschaften gibt. Hinzu kommt, dass LDAP keine Transaktionsunterstützung bietet.

CORBA (Common Object Request Broker Architecture)

Dies hat nun nichts mehr mit XML und Web-Services zu tun.

Dieser Middleware-Standard ist sicher eine mögliche Basis für eine „SIB der Zukunft", aber eben auf der Basis einer völlig anderen, sehr komplexen Technologie, die zunehmend Boden gegenüber den vergleichsweise einfachen Web-Services verliert.

4.5 Problembereich Datenintegration

Die Vorgaben des XML-Prototypen verlangten eine Bereitstellung der Daten auf wenigstens zwei Servern. Dies wurde in der Architektur so umgesetzt, dass zwei Datenbereitstellungsserver (ausgebildet als OGC WFS) eingesetzt wurden, die sich fachlich (nicht räumlich) ergänzende Teile der Daten beinhalteten. Diese zwei Datenquellen werden durch die darüber angesiedelten Schichten (Datenaufbereitung und Client) integriert. Dies ermöglicht die gewünschte und für den Umfang des Prototypen benötigte Integration.

[10] Völlige Unabhängigkeit von der Datenbankumgebung heißt: Jede Datenbankumgebung, also auch herkömmliche Datenbanken, die nicht XML-basiert sind, müssen durch das Konzept unterstützt werden. Dagegen verstoßen wir, wenn wir nur XML-Datenbanken zulassen.

Transparentes, serverübergreifendes Navigieren über XLink-Verweise

Wir möchten aber ergänzen, dass wir eigentlich einen anderen Ansatz für sinnvoller halten, als die Aufgabe der Integration vollständig in die „höheren" Schichten zu verlagern. Wir haben im Prototyp darauf verzichtet, um nicht Erweiterungen an XtraWFS vornehmen zu müssen, die zwar in Übereinstimmung mit dem WFS-Standard gewesen wären, aber dennoch eine produktspezifische Erweiterung dargestellt hätten.

Auch bei einer Verteilung der OKSTRA®-Objekte auf verschiedene Server ist für viele Operationen eine durchgängige Sicht auf den integrierten OKSTRA®-Datenbestand nötig:

- Die Sicht vom Netz zu den Eigenschaften ist erforderlich, sobald Netzänderungen am Verlauf von Abschnitten oder Ästen durchgeführt werden. Allein schon das Umkehren der Stationierungsrichtung an einem Abschnitt führt dazu, dass alle Eigenschaften, die daran partizipieren, aufgesucht werden müssen und neue Stationen erhalten.
- Die Sicht von den stationierten Eigenschaften zum Netz ist immer dann nötig, wenn der Zugriff auf Netzdaten erfolgen muss. Das ist insbesondere immer dann der Fall, wenn eine Eigenschaft geometrisch ausgewertet werden soll, den die Geometrien der Eigenschaften sind ja grundsätzlich im „lokalen Stationierungskoordinatensystem" des zugehörigen Abschnitts oder Asts definiert.

Wünschenswert wäre nun der transparente Durchgriff von einem Objekt in der einen Datenbereitstellungskomponente auf die Objekte in der anderen Datenbereitstellungskomponente mit denen das Objekt in Beziehung steht. Hierfür müsste dieser jedoch „wissen", dass und wie es auf diese entfernt liegenden Objekte zugreifen kann.

Dies ist zwar bereits theoretisch mit dem aktuellen WFS-Standard möglich, allerdings hat sich noch keine produktübergreifend einheitliche Herangehensweise an das Problem entwickelt. Dennoch ist trotz einiger offener, technischer Punkte eine Umsetzung, eben auf noch nicht standardisierter Basis, bereits heute möglich.

Verteilte Transaktionen

Bei serverübergreifenden Fortführungen weiten sich die bereits in 4.4.4 beschriebenen Sachverhalte zu ernsthaften und nicht-trivialen Problemen aus, die durch Nutzung der WFS-Schnittstelle alleine nicht in den Griff zu kriegen sind.

Bei Durchführung einer komplexen Operation auf mehreren Servern kann nämlich die sichere Logik des Holens und Sperrens und anschließenden Veränderns der gesperrten Features durch eine Transaction-Operation nicht mehr funktionieren, weil für jeden Server eine extra Fortschreibungs-Operation durchgeführt werden muss.

Das Problem zeigt sich besonders auf zwei Arten:

1. Geht die erste Transaction-Operation auf dem ersten Server glatt und versagt die zweite, so bleibt das Gesamtsystem in einem inkonsistenten Zustand, der nicht zurückgefahren werden kann. Dies ist ein schlimmer Fall, weil die Datenbasis dauerhaft in einem insgesamt inkonsistenten und deshalb evtl. unbrauchbaren Zustand verbleibt.
2. Verbunden damit ist auch eine „unsaubere Sichtbarkeit" der nach der erfolgreichen Transaction auf dem ersten Server fortgeführten Features. Denn die dazu gehörenden Fortführungen auf Server 2 sind noch nicht ausgeführt, auch wenn dies schließlich erfolgreich geschehen sollte. Dies erzeugt bis zur Fertigstellung der Operation auf Server 2 temporär denselben inkonsistenten Zustand wie im ersten Fall.

Durch explizit kontrollierbare, also operationsüberspannende Transaktionen (wie in 4.4.4 vorgeschlagen) würden sich diese Effekte vermeiden lassen. Man würde in diesem Falle die Transaktion auf Server 1 offen lassen und „zurückrollen", falls die Transaktion auf Server 2 fehlschlägt.

Die für die Durchführung von verteilten Transaktionen erforderlichen Steuerungs- und Kommunikationsaufgaben sind leider i.d.R. kein standardisierter Bestandteil von Datenbankmanagementsystemen, sodass sich das Konzept des WFS einfach dieser Fähigkeiten bedienen könnte.

Von den existierenden Standards in diesem Umfeld käme für eine XML-basierte SIB vermutlich nur das von der IETF entwickelt Transaction Internet Protocol (TIP) in Frage, da er als einziger von vorneherein als Web-Service konzipiert wurde.

Eine Lösung dieses Problems der verteilten Transaktionen lag natürlich außerhalb der Möglichkeiten des XML-Prototypen. Tatsächlich handelt es sich ja auch um ein Problem der Datenbank-Technologie und nicht der XML-Strukturierung einer SIB. Eine Lösung für die Zukunft wird von der weiteren Entwicklung abhängen. Als Empfehlung für heute kann nur geraten werden, abzuwarten und zunächst den Schwerpunkt auf lesende oder nur lokal verändernde Web-Applikationen zu legen. Wie später ausgeführt werden wird, bilden diese tatsächlich auch den wirtschaftlich interessantesten Teil.

4.6 Fachliche Dienste

Für den XML-Prototypen wurden alle im Feinkonzept spezifizierten Services umgesetzt, es gab keine nennenswerten Abweichungen.

Die Dienste der Datenaufbereitung wurden in Java programmiert. Eingesetzt wurde das Java-Framework Apache Jakarta Tomcat 4.1.18, das als OpenSource erhältlich und damit lizenzfrei ist. Für Java als Entwicklungsplattform sprach weiter die Tatsache, dass Java bezüglich der Unterstützung der XML-Technologien einfach am weitesten ist. Z.B. steht mit Xerces 2 ein umfangreiches Werkzeug zur syntaktischen Prüfung und Verarbeitung von XML zur Verfügung.

Es muss allerdings auch gesagt werden, dass die gesamte verwendete Technologie noch so „frisch" ist, dass es zu einigen unerwarteten Problemen kam. Beispielsweise war das Data-Binding, das als „automatischer" Service des eingesetzten XML-Editors XML-Spy eingesetzt wurde, so fehlerhaft, dass es fast komplett von Hand überarbeitet werden musste. Auch wurde die DOM-2-Schnittstelle des Xerces-Parsers als sehr uneinheitlich empfunden, insbesondere was die Verarbeitung von Namespaces anging.

Im Java-Umfeld traten auch Performanz-Probleme zu Tage. Hier trifft die sehr gesprächige Form von XML (Klartext!) auf die konstruktionsbedingt verlangsamte Interpretation von Java. Dies macht sich durchaus bemerkbar, wenn Geometrien mit Hunderten von Koordinatenpaaren zerlegt und konvertiert werden müssen, wie das bei vielen fachlichen Diensten der Fall ist. Diese Probleme waren jedoch erwartet worden und es war ihnen mit der Auswahl geeignet ausgebauter Server-Rechner begegnet worden.

Bei der Erstellung einiger fachlicher Dienste machte sich auch bemerkbar, dass durch die strikte Nutzung der Datenbereitstellungsschicht die Möglichkeit entfällt, Datenstrukturierungen (z.B. geeignete Indexe) in der Datenbank direkt anzulegen. Zwar kann man über die Datenbereitstellung alle erforderlichen Daten mehr oder weniger problemlos beschaffen – man ist jedoch in der Realisierung des Dienstes dazu gezwungen, die erforderlichen Strukturierungen „on-the-fly" zu schaffen, die in einem Datenbankumfeld von vorneherein vorhanden wären.

Diese Feststellung wurde besonders bei den Längenauswertungen gemacht. In solchen Fällen kann man überlegen, die Auswertung im Datenbankumfeld zu machen und die errechnete Information durch den WFS zur Verfügung zu stellen. Dies liefe dann auf eine OKSTRA®-Modellierung dieser Informationsobjekte hinaus, ähnliche Anregungen wurden auch im Forschungsprojekt zum objektorientierten OKSTRA® entwickelt.

4.7 Erkenntnisse aus der Client-Entwicklung

Bei der Client-Entwicklung für den XML-Prototypen hat sich erneut gezeigt, dass die Konstruktion ästhetischer, funktioneller und ergonomischer Bedienungsoberflächen für den Einsatz in Web-Browsern ein schwieriges und teures Unterfangen ist. Die Ursachen dafür sind folgende:

Zur Entwicklung von Web-Services kommt man i.d.R. mit einigen standardisierten Kerntechnologien aus, die gut dokumentiert sind. Im vorliegenden Fall wurden Java 2 Servlets, XML (samt Namespaces und XLink), XML Schema sowie HTTP genutzt. Die Integration dieser Technologien zu einem Framework für die Anwendungsentwicklung ist dank vorhandener Java-Klassenbibliotheken ein gelöstes Problem. Es gibt eine Reihe von integrierten Entwicklungssystemen (IDEs, z.B. JBuilder von Borland, WebSphere von IBM, Sun ONE Studio von Sun oder das offene Eclipse, um nur einige zu nennen), die in der jeweils umfangreichsten Ausstattung server-seitige Entwicklung samt Debugging über physische Servergrenzen hinweg ermöglichen (wie das auch Microsofts Visual Studio.NET für C# leistet).

Die Entwicklung von sog. Clients, also Web-Anwendungen mit sichtbarer Darstellung der Informationen und der Möglichkeit der Benutzerinteraktion erfordert jedoch zusätzlich den Einsatz weiterer browser-naher Technologien, im vorliegenden Fall HTML, ECMAscript, CSS und XSLT.

Zunächst bedeutet das, dass der Entwickler oder die Entwicklerin auch hierin detaillierte Kenntnisse benötigt, und ganz besonders über ihr Zusammenwirken. Leider ist zudem die Unterstützung für diese Technologien sogar von Browser-zu-Browser-Version desselben Herstellers unterschiedlich oder sogar widersprüchlich.

Die oben genannten Entwicklungswerkzeuge bieten zwar auch die Nutzung der genannten Technologien an, Komfort und Testmöglichkeiten sind jedoch sehr begrenzt. Ein weiteres Problem mit den IDEs ist, dass unter Kontrolle eines IDE entwickelte und getestete Clients sich in der späteren Produktionsumgebung ohne IDE nicht selten anders verhalten als mit und deshalb zusätzliche Testläufe erfordern.

Für die Client-Entwicklung im vorliegenden Fall wurde zunächst die Nutzung eines IDE ins Auge gefasst. Geprüft wurden IBMs Visual Age, Sun ONE Studio 4 und Borlands JBuilder 8. Die Unterstützung für die genannten browser-nahen Technologien wurde als unbefriedigend eingestuft. Stattdessen wurde ein Entwicklungsarbeitsplatz mit folgenden Komponenten eingerichtet:

- Macromedia Dreamweaver 4 für die grafische HTML-Entwicklung
- Xidicone Zeus als sprachsensitiver Editor sowohl für Java, ECMAscript, HTML/CSS und XML
- Apache 2.0.43 und Tomcat 4.1.18 als Web Server (dieselben Versionen wie für die Produktionsumgebung vorgesehen, um ständig produktionsnahe Tests durchführen zu können)
- Suns Java SDK Dokumentation und die SELFHTML Dokumentation

Die Entwicklung der für den Client benötigten Java-Komponenten, die den Anschluss an die Web Services (Fachdienste und WFS) erwies sich auch ohne IDE als sehr unproblematisch, dank der gewählten Rapid-Prototyping[11]- und Unit-Testing[12]-Entwicklungsstrategie.

Auch der Entwurf der Layouts für die Oberfläche war technisch und zeitlich problemlos.

Die Hauptschwierigkeiten lagen in der Nutzung von browser-seitigem Scripting (ECMAscript) und XSLT Stylesheets.

Auf Scripting konnte nicht komplett verzichtet werden, da anders eine reaktive und ergonomische Oberfläche nicht herzustellen ist, will man einen häufigen Bildneuaufbau und damit den Eindruck des „Surfens" vermeiden.

Die Entscheidung, welche Funktionen besser server-seitig und welche besser durch Scripting im Browser erledigt werden, verlangt vom Entwickler viel Erfahrung und Fingerspitzengefühl, vor allem aber ständige Überprüfung des Verhaltens der Oberfläche bei Interaktion.

XSLT-Stylesheets sind eine mächtige, aber auch komplexe, Technik, um das Aussehen von Oberflächenelementen datengesteuert flexibel zu gestalten. Das Testen der XSLT-Stylesheets erwies sich jedoch als schwierig. Ursache hierfür ist, dass die Transformation von Eingabe in Ausgabe durch ein Stylesheet nicht schrittweise nachvollzogen werden kann. Hier ist unbedingt der Einsatz eines entsprechenden Testwerkzeugs zu empfehlen. Der Auswahl eines Produktes muss jedoch zunächst eine gründliche Eignungsprüfung vorausgehen. Die XSLT-Funktionalität der marktgängigen Produkte wurde nach einem durchgeführten ersten Überblick trotz aller Beteuerungen der Hersteller zunächst als unbefriedigend empfunden.

Fazit:

Das Entwickeln von Thin-Clients mit Web-Technologie setzt vielseitige und erfahrene Entwickler voraus, die zudem einen Blick für Ästhetik und Ergonomie haben müssen. Es sind zudem während der Entwicklung sehr häufige Tests unter Produktionsbedingungen und mit verschiedenen Browserversionen erforderlich, was die Entwicklungszeiten entsprechend verlängert und damit die Entwicklungskosten erhöht.

4.8 Offengebliebene Fragen

Dieser Abschnitt enthält eine Sammlung von Punkten, die bei der Erstellung des XML-Prototypen bewusst beiseite gelassen wurden, weil sie für die Fragestellung, um die es beim Prototyping schwerpunktmäßig geht, als nachrangig eingestuft wurden.

Da die hier zu behandelnden Fragen nicht untersucht wurden, stehen auch keine umfassenden Antworten sondern eher skizzenhafte Stichworte zur Verfügung. Für eine Produktionsversion des XML-Prototypen oder für eine ähnlich gerichtete Umsetzung der Erfahrungen aus diesem Projekt sind die genannten Punkte alle in unterschiedlichem Maße relevant und müssen bedacht werden.

4.8.1 Grafische Interaktion

Erst eine grafische Repräsentierung der Information liefert dem Anwender den nötigen Überblick und die nötige Sicherheit im Umgang mit den Daten. Für eine Produktionsversion ist daher unbedingt eine komfortable interaktive grafische Bearbeitung der Daten vorzusehen.

---

[11] Die Funktionalität wird in ganz kleinen Schritten eingebaut, nach jedem Schritt wird getestet.

[12] Es wird eine Methode nach der anderen zu Ende entwickelt und getestet.

Im XML-Prototypen wurde (entsprechend den Vorgaben) Grafik nur auf eine sehr einfache Art berücksichtigt. Es wurde ein OGC Web Map Server (XtraServer/XtraWMS) auf dem Netzdatenbestand eingesetzt, der die jeweilige Fortschreibungssituation auf dem Netzserver als Bild generiert und der die üblichen Funktionalitäten zur Steuerung von Bildausschnitten bietet.

Für eine Produktionsversion wird dies nicht reichen, obwohl das Konzept des Web Map Servers (WMS) in dieser Hinsicht unserer Erfahrung nach erstaunlich weit trägt. Einer der Hauptvorteile des Konzepts ist die mögliche Integration von Bildern, die oft viel einfacher möglich ist als die Integration der dahinter liegenden Daten. (Im XML-Prototypen ist beispielsweise als Hintergrundbild die TK50 integriert, die über einen WMS des Landesamts für Datenverarbeitung und Statistik in NRW bezogen wird.)

Die Funktionalität der WMS ist keineswegs im Abrufen fester Bildebenen erschöpft. Bei WMS in der Ausbaustufe SLD (Styled-Layer Descriptor) kann auf die hinter dem WMS liegende Datenquelle (einen WFS) zugegriffen werden, um die Grafik entsprechend den dort vorgefundenen Features individuell zu steuern. Dies kann von der Client-Software eingesetzt werden, um interaktiv ermittelte Sachverhalte (z.B. eine Selektion) in der Grafik sichtbar zu machen. Auch besitzt das WMS-Konzept die Möglichkeit, Informationen zu Features abzurufen, die durch Koordinateneingaben identifiziert sind.

Extrem wichtig bei der grafischen Interaktion ist Performanz. Nach unserer Erfahrung sind WMS hier durchaus konkurrenzfähig. Das liegt daran, dass die erzeugten und an den Client gesendeten Bilder in Pixelformaten (z.B. als PNG) eher klein sind und keine ernst zu nehmenden Download-Zeiten erzeugen. Große Bilder in Vektorformaten, etwa als „Scalable Vector Graphics"-Format (SVG), haben hier durchaus mehr Probleme. Dafür sind sie natürlich auch ungleich komfortabler und erlauben im Thin-Client interaktive Operationen, die mit einer reinen Pixelgrafik nicht möglich sind.

Deshalb ist nach unserer Erfahrung eine Kombination der Verfahren angezeigt, wo der Hauptteil des Bildes von einem WMS in Pixelgrafik erzeugt wird und der aktive zu verändernde Teil in SVG (welches natürlich auch von einem WMS erzeugt werden kann).

4.8.2 Auswirkungen der Historisierung

Im XML-Prototypen blieb die im OKSTRA® vorgesehene Historisierung unbeachtet. In einer Produktionsversion muss sie jedoch einbezogen werden.

Relativ einfach gestaltet sich die Einführung einer „Stichtags-Sicht" auf einen historisierten Datenbestand. Dies kann durch Einführen entsprechender Prädikate bei der Relationsverfolgung in Queries sehr leicht umgesetzt werden. Voraussetzung sind allerdings bestimmte Erweiterungen bei der Eigenschaftsadressierung im WFS. siehe hierzu 4.4.2 Teil „Behandlung multipler Zusammenhänge".

Für Abfragesysteme ist die „Stichtags-Sicht" die normale Sichtweise. Insofern dürfte in diesem reduzierten Zusammenhang kein wirkliches Problem existieren.

In einer Vollversion ist dies nur ein keiner Teilaspekt. Dies wird zu einer neuen Qualität der Komplexität bei den fachlichen Diensten und beim Client führen. Insbesondere müssen auch einer Reihe von Zusatzobjekten beachtet werden, die bisher keine Rolle spielten, wie etwa Ereignis, identisches_Netzteil, usw.

4.8.3 SOAP?

XML-kodierte Web-Services können auf Basis verschiedener unterschiedlicher Protokolle abgewickelt werden. Eines davon ist das „Simple Object Access Protocol" (SOAP), ein Spezifikationsent-

wurf des W3C. SOAP ist ein Protokoll für die Durchführung von Funktionsaufrufen (RPC = Remote Procedure Call) über das Internet. Es nimmt daher eine Stellung ähnlich zu CORBA ein - Middleware für das Internet. SOAP kann über verschiedene Internetprotokolle transportiert werden, z.B. http oder SMTP.

Der Standardisierungsprozess von SOAP 1.2 beim W3C ist trotz großer Ressourcen noch nicht abgeschlossen. Tatsächlich ist das Protokoll in der Zwischenzeit alles andere als „simple" und entwickelt sich immer mehr zu einem Ersatz für CORBA.

Wegen der noch bestehenden Instabilität klammert der XML-Prototyp SOAP zunächst aus und verwendet direkt http/POST. Dies tut er in Übereinstimmung mit den OGC Spezifikationen, die aus demselben Grund erst mal den Weg gegangen sind, Request/Response-Paare zu definieren, die auch direkt auf den bestehenden http-Protokollen eingesetzt werden können.

Letztlich ist es möglich, die vereinbarten Request/Response-Paare über alle in Frage stehende Protokolle zu übertragen. D.h. hier kann man abwarten. Falls sich SOAP durchsetzt, ist es natürlich sinnvoll, diesen Weg zu gehen. Zur Zeit bedeutet er nur zusätzlichen Overhead.

4.8.4 Authentifizierung, Autorisierung

Ein Produktionssystem muss die Aspekte der Authentifizierung und Autorisierung (AA-Dienste) einbeziehen. Dies ist besonders dringlich, falls das System über das Internet zur Verfügung steht. Aber auch ein auf das Intranet beschränkte System wird nicht jedem Benutzer alles erlauben wollen.

Authentifizierung ist die Voraussetzung für die Autorisierung. Durch die Authentifizierung wird die Frage beantwortet: Ist der Diensteanforderer derjenige, der er vorgibt zu sein?

Man muss hier das Problem der Authentifizierung eines Benutzers von einer automatisierten Diensteinanspruchnahme durch einen anderen Dienst unterscheiden, der natürlich ebenfalls eine Authentifizierung benötigt. Bei einem Benutzer kommen im einfachen Fall „Passwords" in Frage oder auch Signaturkarten oder biometrische Verfahren. Auch kann natürlich die Authentifizierung eines Benutzers nicht laufend erfolgen sondern nur einmal in einer Applikation. Ein Dienst muss sich gegen verschiedene andere Dienste authentifizieren, wobei in diesem Falle ein einheitliches technisches Verfahren erforderlich ist, das möglichst homogen mit den eigentlichen (technischen) Web-Services zusammenarbeitet.

Zur Authentifizierung kommen in erster Linie zwei Verfahren in Frage:

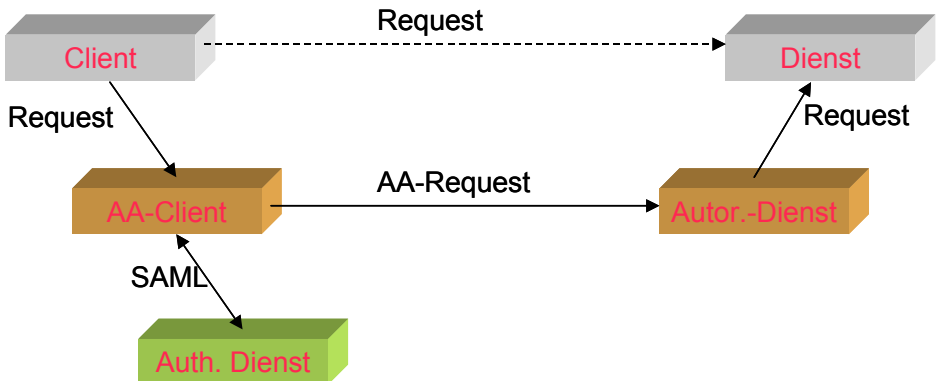
1. SAML (Security Assertion Markup Language): Es handelt sich um einen XML-basierten Standard, der von OASIS entwickelt wurde. Das Grundelement sind sogenannte „Assertions", die nach erfolgter Authentifizierung an den Client übermittelt werden und ihm als Ausweis für folgende Operationen dienen.
2. XML Signature: Vom W3C entwickelter Standard zur Darstellung digitaler Signaturen und zu deren Berechnung und Verifizierung. Die digitale Signatur kann vom Server sicher überprüft werden und beweist die Authentität des Requests.

Die Autorisierung beantwortet die Frage: Darf der Diensteanforderer das, was er will?

Dies ist normalerweise Aufgabe von Datenbanken, die dies auch leisten könnten, wenn die Autorisierungsinformation zu ihnen gelangen würde. Durch einen Dienst, wie den Web Feature Server, gelangt jedoch nur unter Zusatzanforderungen außerhalb des WFS-Standards Authentisierungsinformation bis in die Datenbank.

Es ist deshalb in einer offenen Infrastruktur günstiger (dies gilt nicht für geschlossene Applikationen), Autorisierung durch zusätzliche Dienste abzuwickeln, die automatisch das „Autorisierungsgeschäft" außerhalb des WFS vornehmen und diesen nur mit bereits autorisierten Requests versorgen. Eine solche Logik kann durch eine „Dienste-Fassade" völlig transparent gemacht werden. Siehe hierzu auch folgende Darstellung:

Quelle: Gartmann, Fraunhofer ISST, Dortmund



SAML kann auch Assertions für den Zugriff auf die Eigenschaften von Objekten erzeugen, sodass auf dieser Basis auch ein Autorisierungsdienst erstellt werden kann.

4.8.5 E-Commerce

Bei einem kontrollierten Nutzerkreis von Web-Diensten reichen zumeist die soeben diskutierten AA-Dienste für eine evtl. erforderliche Abrechnung der Inanspruchnahme von Web-Diensten aus.

Anders wird dies, wenn sich die Straßenbauverwaltungen öffnen wollen und ihre Daten und Dienste zur freien Verwendung in Geodaten-Infrastrukturen anbieten. In diesem Falle wird man die Möglichkeit benötigen, die angebotenen Dienste zu verkaufen, was insbesondere auch das Anbieten und Bepreisen der Dienste einschließt.

Möglich ist dies natürlich über eigens dafür erstellte Portallösungen. Angemessener würde dies jedoch über spezialisierte Web-Services erfolgen, die Preismodell, nutzungsbezogene Bepreisung, Vertragsabschluss und Lieferung der Dienste durch einen spezialisierten Web-Service durchführen. Ein solcher Vorschlag wurde im Rahmen der Initiative GDI NRW vom Fraunhofer ISST Dortmund gemacht und zusammen mit interactive instruments erprobt. Der Dienst heißt „Web Pricing & Ordering Service" (WPOS). Er wurde inzwischen von der OGC als „Discussion Paper" veröffentlicht.

5 Empfehlungen für das weitere Vorgehen

5.1 Nutzen und Kosten

5.1.1 Nutzen

Die XML-Strukturierung einer Straßeninformationsbank ist die Voraussetzung für die Errichtung und das Funktionieren einer offenen Architektur von Web-Services. Wie schon in 4.2.2 dargelegt, bietet eine solche Konstruktion zwei wesentliche Vorteile:

- Verteilung von Daten und Diensten zu vielen Nutzern
- Integration von Daten und Diensten aus verschiedenen Quellen

Beide Vorteile erwachsen aus den Eigenschaften und dem allgegenwärtigen Vorhandensein des Internet/Intranet und seiner Infrastruktur wie Web-Browsern.

Der Verteilungsaspekt ist eng mit dem Begriff des Thin-Client verknüpft, worunter üblicherweise eine im Internet-Browser abrufbare Applikation verstanden wird. An dieser Stelle ergibt sich aus der Technologie zuallererst ein offensichtliches und konkretes Einsparpotential, das aus zwei Quellen schöpft:

1. Es entfallen die Kosten für die Lizenzierung spezieller Client-Software.
2. Es entfallen die Kosten für Installation, Wartung und Pflege der Applikation in der Fläche.

Der zweite der Punkte ist wirklich essentiell. Denn die durchgängige Pflege einer Applikation an vielen Arbeitsplätzen ist vor allem ein sehr personalintensives Unterfangen. Eine Thin-Client-Applikation wird dagegen zentral gepflegt.

Weniger gut zahlenmäßig erfassen lassen sich die strukturellen Vorteile aus Thin-Client-Lösungen, als da sind:

- Da im Gegensatz zu herkömmlichen PC-Applikationen die Daten nicht mehr mit der Applikation verteilt, sondern zentral über das Netz abgerufen werden, stehen sie in einer aktuelleren Qualität zur Verfügung.
- Durch die beliebige Verfügbarkeit der Applikation im Netz kann ein erheblich höherer Nutzungsgrad der Applikation und der zugrunde liegenden Daten erreicht werden.

Durch diese beiden Punkte ergibt sich vor allem das Potential für eine Verbesserung der Dienstleistungsqualität bei den Straßenbauverwaltungen. Durch den verbesserten Informationszugang kann natürlich auch ein Rationalisierungseffekt erreicht werden.

Aus den Möglichkeiten zur Integration von Daten und Diensten ergibt sich vor allem ein struktureller und organisatorischer Nutzen.

Das Einsparpotential kommt dadurch zustande, dass ein Wildwuchs im Bereich der Straßeninformationsbanken verhindert wird, die ohne die Einbeziehung der Datenintegration über das Netz wegen des Bedarfs an sie angeschlossenen Applikationen mit immer mehr zusätzlichen Aufgaben belastet werden.

Durch die Möglichkeit, verteilte Daten online zu integrieren, wird es möglich,

- gleichartige Daten, die in verschiedenen Gebieten liegen, gemeinsam zu verarbeiten (räumliche Integration) und
- einander ergänzende Daten im selben Gebiet zusammen zu sehen (fachliche Integration).

Besonders der letzte Punkt hat einen hohen organisatorischen Nutzen, da die Daten in der Abteilung verbleiben können, die sich um deren Pflege kümmert.

5.1.2 Kosten

Die Erstellung von Web-Lösungen ist aufwändiger als die Erstellung funktional vergleichbarer konventioneller Lösungen, was sich natürlich in den Kosten dafür niederschlägt.

Besonders die Erstellung von Web-Clients ist ein sehr aufwändiger Prozess, weil man es mit heterogenen und meist unausgereiften Programmierumgebungen zu tun hat. Auch werden gut ausgebildete Programmierer benötigt, die sich in einem weitem Feld verschiedener Umgebungen zu Hause fühlen. Die Problematik wurde in Abschnitt 4.7 bereits dargelegt.

Zur Beurteilung des Aufwands und als Vergleichsmaßstab diene der Hinweis, dass die Erstellung des vergleichsweise einfachen Client für den XML-Prototypen über 30 Personentage gekostet hat. Es ist daher davon auszugehen, dass die Entwicklung eines Client für volle SIB-Lösung weit mehr Ressourcen benötigen wird.

Auch die Entwicklung fachlicher Dienste ist aufwändiger als es z.B. bei Einsatz eines homogenen GIS-Systems als Basis wäre. Dies liegt nicht nur daran, wie man vordergründig meinen könnte, dass die in XML formulierten Request/Response-Paare zu analysieren sind. Das tatsächliche Problem ist es, sich die benötigten Daten zu beschaffen und darauf die erforderlichen Indizierungen „on-the-fly" zu generieren.

Die Kosten für die Entwicklung von Thin-Clients für ein proprietäres Umfeld (also z.B. eine vorhandene SIB) sind zwar auch nicht zu unterschätzen, sind aber weit niedriger als im umgebungsunabhängigen Fall, der durch den XML-Prototypen angegangen wurde. Das liegt daran, dass moderne DBMS/GIS-Systeme heute bereits meist eine gute Unterstützung für den „Web-Export" der bereits programmierten interaktiven Funktionalität mitbringen, sodass das „Web-fähig-Machen" einer vorhandenen SIB-Lösung relativ einfach ist.

Auf der Kostenseite darf auch nicht vergessen werden, dass vor einer Durchführung einer allgemeinen, verteilten und fortschreibenden Lösung auch noch technologische Probleme zu lösen sind, siehe hierzu 4.5.

5.2 Bewertung und Empfehlung

5.2.1 Das Optimum

Das wirtschaftliche Optimum liegt wie immer bei einer größtmöglichen Nutzung der Vorteile bei Minimierung der Kosten. Nach unserer Auffassung erfüllen vor allem spezialisierte Fach-Auskunftssysteme und einfache fachbezogene Fortführungssysteme wie „Unfälle", „Baumkataster", „Baustellen", „Zustand" usw. diese Voraussetzungen.

Wie oben (5.1.1) dargelegt wurde, besteht der wirtschaftliche Nutzen vor allem aus der Verteilung von Daten und Diensten zu vielen Nutzern. Der Nutzen wird deshalb am größten, wenn möglichst viele Nutzer von der Verteilung profitieren.

Gleichzeitig sind die Entwicklungskosten von Web-Applikationen relativ hoch (5.1.2). Sie liegen üblicherweise über denen konventioneller Lösungen. Die geringsten Entwicklungskosten entstehen bei kleinen, wenig komplexen Anwendungen.

Die genannten Fach-Auskunftssysteme und einfachen fachbezogene Fortführungssysteme entsprechen beiden Gedankengängen. Sie werden von einer Vielzahl von potentiellen Nutzern benötigt und sind gleichzeitig relativ klein und übersichtlich. Bei ihnen wird also der größtmögliche Nutzen bei geringsten Kosten bewirkt.

Bei den genannten Systemen wird überdies die Mehrzahl der technischen Probleme vermieden, die zur Zeit noch im Bereich der Datenbereitstellung existieren, siehe 4.5.

Die Datenbereitstellung sollte über standardisierte Dienste erfolgen. Zur Zeit kommt hierfür nur der OGC Web Feature Server Frage, weil es keinen alternativen verwendbaren Standard gibt. Die Eigenentwicklung eines solchen Standards wäre aufwändig, weil für die Realisierung nicht auf vorhandene Software zurückgegriffen werden kann. Außerdem würde so die Möglichkeit vertan, kompatibel zu den im Aufbau befindlichen Geodaten-Infrastrukturen zu werden.

Es ist dabei auch zu beachten, dass einfache Auskunfts-Anwendungen auch oft nur mit Bildern und Auskünften zu den Objekten auskommen. In diesem Falle geht unsere Empfehlung ganz klar in Richtung des Einsatzes des OGC Web Map Servers. Web Map Server mit Styled-Layer Descriptor liefern alle nötige Funktionalität, um auch die für die Dialoge nötigen grafischen Echos zu generieren.

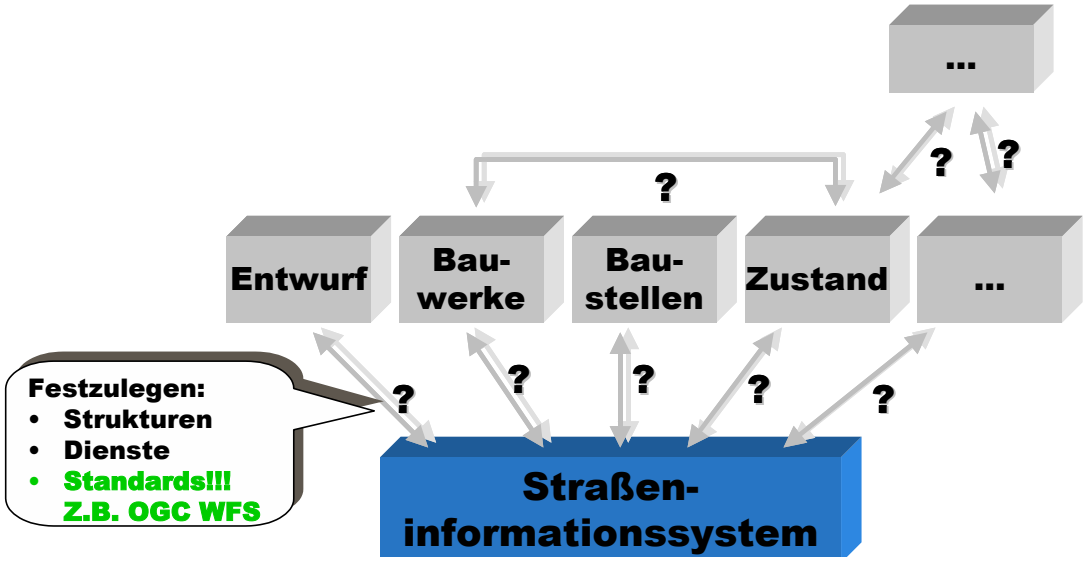
5.2.2 Empfohlene Architektur

Für die Architektur der beschriebenen Auskunftsanwendungen schlagen wir trotz der teilweise bestehenden technischen Schwierigkeiten (diese sind immerhin im Prototypen gelöst worden) eine verteilte Lösung vor, in der die Fachdaten nicht in ein zentrales Straßeninformationssystem integriert werden, das die Netzdaten enthält, sondern dezentral vorgehalten werden (zumindest sofern organisatorisch unterschiedliche Stellen für die Daten zuständig sind). Die Integration würde dann in den Applikationen erfolgen.

Eine zentrale Lösung ist natürlich auch möglich. Von ihr geht die Gefahr aus, dass das zentrale System immer mehr anwächst, bis es nicht mehr handhabbar ist.

Das folgende Diagramm zeigt den Gesamtansatz im Überblick. Die einzelnen grauen Kästchen stehen für Fachsysteme (Entwurf, Bauwerke, Baustellen, Zustandserfassung und –bewertung, Pavement-Management, etc.). Das blaue Kästchen steht für das vorhandene zentrale Straßeninformationssystem. Es enthält mindestens das Straßennetz gemäß ASB und die Geometriebezüge für die zentralen Objekte wie Abschnitte, Äste, Nullpunkte usw.

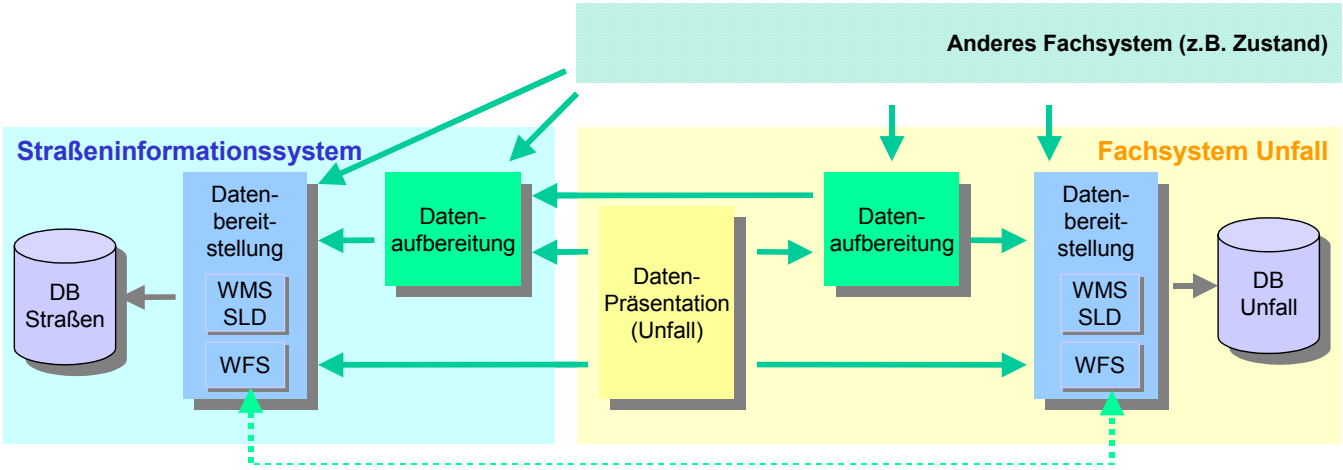
Die Schnittstellenproblematik zwischen den Fachsystemen und dem zentralen Straßeninformationssystem sowie bei den Fachsystemen untereinander ist im Diagramm durch Pfeile symbolisiert. Die Natur dieser Schnittstellen ist festzulegen. Wir schlagen im Sinne der Ergebnisse aus dem Projekt XML-Prototyp für die Bereitstellung des Basisdaten den Transport in der Form OKSTRA®-XML durch den standardisierten OGC-Service „Web Feature Service" vor. Falls für Anwendungen die Bereitstellung von Bildern sinnvoll ist, kommt zusätzlich oder alternativ der „Web Map Service" in Frage, am besten in seiner Ausbaustufe „Styled-Layer Descriptor", die eine Gestaltung der erzeugten Karten durch den Client erlaubt.



Auch die Fachsysteme können (und sollten) ihre Daten durch einen WFS und/oder WMS/SLD anbieten. Dies ist natürlich im Prinzip nur dann erforderlich, falls die Daten für andere Fachsysteme von Interesse sein können, was zumindest rein lesend meist der Fall sein dürfte. Dies gilt natürlich auch für den Fall, dass einzelne Fachsysteme bereits jetzt integriert im zentralen Straßeninformationssystem vorliegen.

Wie bildet sich der im obigen Überblicks-Diagramm dargestellte Ansatz auf die Komponenten-Strukturierung des XML-Prototypen ab?

Dazu nehmen wir uns im Detail das Beispiel „Unfälle" heraus und stellen den inneren Aufbau dieser Applikation im Zusammenhang zum zentralen Straßeninformationssystem dar. Siehe dazu die nächste Grafik.



In der Grafik wurde die Software des bereits existierenden Straßeninformationssystems auf Vereinfachungsgründen weggelassen. Diese greift parallel auf die Datenbasis „Straßen" zu und leistet deren interaktive Pflege. Ob der Einbau der Datenbereitstellungsdienste nun direkt auf der Datenbasis erfolgt oder ob dafür auf Softwareschichten des bereits bestehenden Systems zugegriffen wird, ist von der Struktur dieses Systems abhängig und bleibt deshalb offen.

Die grünen Pfeile (Pfeile immer in Aufrufrichtung) stehen für den Austausch von Standard-Information über Web-Services, meist in der Form von XML-Dokumenten - anderenfalls Bilder aus den WMS. Die beiden grauen Pfeile (zwischen der Datenbereitstellung und Datenbasis „Straßen", bzw. zwischen Datenbereitstellung und Datenbasis „Unfall") stehen für eine von der jeweiligen Datenbasis abhängige Form der Kommunikation. Der gebrochene grüne Pfeil zwischen den beiden Datenbereitstellungen (WFS) steht für eine (versteckte) Kommunikation zwischen den WFS, wie sie in 4.5 zur Vereinfachung der Integration zwischen den Diensten gefordert wurde.

Die Datenbereitstellungsdienste (WFS), die über das Straßeninformationssystem und die Fachdatenbank geschichtet sind, stellen die Information in OKSTRA®-XML zur Verfügung. Die WMS/SLD liefern Bilder mit zu spezifizierenden Inhalten.

Die Datenaufbereitungsdienste wurden für die Bereiche Straßennetz und Verwaltungsmaßnahmen im XML-Prototyp analysiert – einige der ermittelten Dienste werden auch in einer Produktionsumgebung sinnvoll sein, darunter mit Sicherheit die Dienste zur Geometrieermittlung von Teilabschnitten und Straßenpunkten. Welche Datenaufbereitungsdienste für eine Unfall-Applikation erforderlich sind, muss noch bestimmt werden, möglicherweise sind diese so spezifisch, dass sie der Datenpräsentationsschicht zugeschlagen werden können.

Die Datenbereitstellungsdienste und Datenaufbereitungsdienste sowohl des Straßeninformationssystems als auch der Fachdatenbasis stehen weiteren Fachsystemen zur Integration mit deren Diensten zur Verfügung.

5.2.3 Eine vollständige „Web-Straßeninformationsbank"?

Eine komplette „Web-SIB", wie im XML-Prototyp (prototypisch) realisiert wurde, befindet sich nach unserer Auffassung nicht in einem günstigen Preis/Leistungsverhältnis. Für eine „Web-SIB" wird ein sehr aufwändiger Client programmiert werden müssen und die Zahl der Arbeitsplätze, die tatsächlich für Netzänderungen eingerichtet werden wird, dürfte vergleichsweise klein sein.

Wir denken, dass das Verhältnis so ungünstig ist, dass vermutlich sogar eine proprietäre Thin-Client-Lösung kaum begründet werden kann.

Tatsächlich sind für die wenigen Fortführungsarbeitsplätze die vorhandenen Straßeninformationsbanken nach unserer Einschätzung kostengünstiger einzusetzen.

Wir verstehen dabei unter einer kompletten „Web-SIB" ein web-basiertes System im Sinne des XML-Prototypen, das die zentralen fachlichen Aufgaben, die durch den Prototypen zu leisten waren (also Netzfortführung, Verwaltungsmaßnahmen) in produktionsreifem Zustand zur Verfügung stellt. Das bedeutet, dass eine Thin-Client-Applikation zusammen mit der darunter liegenden Dienstearchitektur entstehen muss, die im Umfang und Bedienbarkeit mit einer der heutigen Straßeninformationsbanken konkurrieren kann. Grafische Bedienbarkeit, Historisierung, Zugangskontrolle usw. müssen in vergleichbarem Umfang zur Verfügung stehen.

Dies ist mit Sicherheit ein sehr aufwändiges Unterfangen. Es wird leider nicht aufgewogen durch einen entsprechend breiten Bedarf für diese Leistung. Netzfortführung ist die Arbeit einiger weniger Spezialisten bei den Straßenbauverwaltungen.

5.2.4 Festlegung fachlicher Dienste

Im XML-Prototypen enthielt die Schicht „Datenaufbereitung" Dienste, die aufbauend auf den Diensten zur Bereitstellung der SIB-Daten in XML auf der Basis von OKSTRA®-XML die Daten so aufbereitet, dass spezifische SIB-Funktionen realisiert werden. In der Datenaufbereitungsschicht ist somit ein wesentlicher Teil des Anwendungswissens der Straßeninformationsbank enthalten.

Wird man solche Dienste auch unter der oben empfohlenen Systemarchitektur benötigen?

Ja, aber nicht unbedingt die im Prototypen entwickelten. Wenn darauf verzichtet wird, eine webbasierte Gesamtlösung einer SIB einschließlich Netzfortführung zu entwickeln, werden die netzändernden Operationen sicherlich entfallen. Lesende fachliche Funktionen sind aber trotzdem wichtig, z.B. die Ermittlung der Geometrie eines Teilabschnitts oder eines Straßenpunkts bzw. umgekehrt die Verortung einer Koordinate im Straßennetz. Hierunter fällt auch der Dienst für die Ermittlung der Längenstatistik. Bei der Ausdehnung der Betrachtung auf Fachsysteme werden sich möglicherweise weitere fachliche Dienste ergeben.

Die Implementierung dieser Dienste kann an drei möglichen Stellen geschehen:

1. Als separate Dienste, die sich die Daten über die Datenbereitstellungen (WFS) holen. Dies ist die architektonisch saubere und universell einsetzbare Lösung, die auch im XML-Prototypen so durchgeführt wurde.
2. Falls ein vorhandenes Straßeninformationssystem die Funktionalität bereits beinhaltet, können die Dienste direkt auf dieses abbilden. Eine solche Lösung ist natürlich nicht universell einsetzbar, aber leicht realisierbar und hat vielleicht sogar Laufzeitvorteile.
3. Versteckt im Fachclient. Hier würden die Dienste nicht offen zu Tage treten und möglicherweise auch nicht über das Netz angesprochen.

In allen Fällen werden Definitionen für diese Dienste benötigt und wir empfehlen diese durchzuführen. Ausgangspunkt sind dabei die für die jeweiligen Fach-Clients geplanten Arbeitsabläufe.

Diese Arbeit sollte am besten formalisiert vollzogen werden, sodass klare und weiterverwendbare Definitionen entstehen. Einer der besten Wege hierfür stellt die Festlegung von Request/ Response-Paare in XML (XML Schema) dar, wie sie im XML-Prototypen vollzogen wurde. Diese Vereinbarungen sollten möglichst im OKSTRA® festgeschrieben werden, siehe auch die Vorschläge zur objektorientierten Weiterentwicklung des OKSTRA®.

5.2.5 Öffnung zu Geodaten-Infrastrukturen

Straßen- und Verkehrsdaten besitzen das Potential, im Zusammenspiel mit den Daten aus anderen Quellen neue und vorher nicht erreichbare Informationen abzuleiten. Sie sind deshalb von allgemeinem Interesse.

Deshalb ist eine Teilnahme der Straßenbauverwaltungen an Geodaten-Infrastrukturen ein sinnvolles Vorhaben. Die Geodaten-Infrastrukturen basieren meist auf den OGC-Standards wie GML, WFS und WMS sowie den ISO-Normen der Serie ISO 19100.

Eine Öffnung über den Bereich der Straßenbauverwaltungen hinaus, kann einem Informationsbedürfnis interessierter (und evtl. sogar zahlender) Abnehmer dienen oder im öffentlichen Bereich helfen, Mehrfacherfassungsaufwand zu vermeiden oder die Aktualität zu verbessern.

5.3 Beispielrechnungen

Im Folgenden wollen wir die soeben ausgesprochenen Empfehlungen aufgreifen und beispielhaft drei aus unserer Sicht sinnvolle Applikationen von ihren Kosten her beleuchten.

Es handelt sich um:

- Auskunft Straßennetz und Bestand
- Auskunft Verkehrsstärken

- Baumschadenskataster

Die Basis für die Applikationen bilden entsprechend der vorgeschlagenen Architektur OGC Web Feature Server und Web Map Server.

Die Kosten für diese Produkte sind zur Zeit noch sehr uneinheitlich und bewegen sich nach unserer Beobachtung zwischen 0 und 50.000 EUR für die Einzellizenz.

Für die drei Beispielapplikationen wurden jeweils Aufwandsabschätzungen durchgeführt und in „Personentagen" angegeben. Zur Ermittlung der Kosten sind diese mit marktüblichen Preisen für den Personentag zu multiplizieren. Die aufgeführten Abschätzungen beziehen sich auf die Erstellung der jeweils angegebenen Konzeption oder Software. Aus dem XML-Prototypen übernehmbare Anteile wurden in den Aufwänden bereits berücksichtigt.

5.3.1 Auskunft Straßennetz und Bestand

Hierbei handelt es sich im Wesentlichen um eine Auskunftsversion des XML-Prototypen, d.h. alle fortschreibenden Operationen auf Netz und Bestand würden fehlen, ebenso die Abfrage der Längenstatistik. Statt dessen würden grafische Darstellung und Bedienbarkeit auf ein produktionsreifes Niveau angehoben, sodass Selektion und Objektbrowser leicht und intuitiv bedienbar sind.

Eigenschaften:

- Grafische Darstellung der Netz- und Bestandsobjekte im Grundriss[13]
- Grafische Selektion durch Anklicken der Objekte und durch Aufziehen einer Box
- Bildnavigation durch Pfeiltasten und Box
- Intelligente Auswahl von Bildebenen (logische Zusammenhänge und Maßstäbe werden beachtet)
- Verallgemeinerte und leicht bedienbare Selektion
- Echo der selektierten Objektmenge in der Grafik
- Durchgängige Verwendung fachlicher Bezeichnungen
- Verbesserte Gestaltung des Gesamteindrucks

Aufwandsabschätzung:

Arbeitspaket	Personentage
Konzeption, Abstimmung mit dem AG	6
Konfiguration der Datenbereitstellung (WFS, WMS/SLD)	12
Dienste in der Datenaufbereitung (durch Prototyp bereits vorhanden)	0
Erstellung des Client	37
Dokumentation	5
Projektleitung, QS, KM	17
Gesamt	**77**

[13] Querprofildarstellungen (Aufbauschichten etc.) sind im Prinzip auch möglich, bedeuten aber zusätzliche Aufwände.

Eine unserer Auffassung nach praktischer Zusatz zu einem solchen Auskunftssystem wäre die Möglichkeit, an den Objekten anwenderspezifische *Anmerkungen* anbringen zu können. Hierbei würde es sich natürlich um schreibende Operationen handeln. Die Anmerkungen in Form von Text („Post-It") oder Grafik („Red-Lining") würden georeferenziert gespeichert und dem Anwender oder einer Anwendergruppe zugeordnet. Die Anmerkungen würden natürlich separat in einer eigenen Datenhaltung/Datenbereitstellung abgelegt.

Geschätzter Aufwand für einen solchen Zusatz: ca. 18 Personentage.

5.3.2 Auskunft Verkehrsstärken

Hier handelt es sich um ein Auskunftssystem eher am unteren Rand der Komplexitätsskala. Es liegen konkrete Erfahrungen zugrunde, weil das System in der beschriebenen Form (beauftragt durch den Landesbetrieb Straßenbau NRW) als Prototyp durch interactive instruments im Sommer 2002 erstellt wurde. Das System kommt nur mit Web Mapping Technologie aus, d.h. es benutzt nur einen WMS/SLD (XtraWMS). Die Daten (Straßennetz und Verkehrsmengen) standen als OKSTRA®-Exportdateien zur Verfügung bzw. wurden in solche umgewandelt.

Eigenschaften:

- Darstellung des Straßennetzes, einschränkbar auf Autobahnen, Bundesstraßen, Landesstraßen/Staatsstraßen, Kreisstraßen
- Darstellung der Verkehrsmengenkarte 2000
- Darstellung der Zählstellenorte mit wahlweiser Angabe von Nummer und KFZ-Gesamt
- Hintergrund TK50 oder DTK10
- Objektinformation zu Zählstellenbereichen und Zählstellen mit Angabe der DTV-Werte zu 1995 und 2000.

Aufwandsabschätzung:

Arbeitspaket	Personentage
Konzeption, Abstimmung mit dem AG	3
Konfiguration der Datenbereitstellung (WMS/SLD)	9
Dienste in der Datenaufbereitung (keine)	0
Erstellung des Client	5
Dokumentation	1
Projektleitung, QS, KM	2
Gesamt	**20**

5.3.3 Baumschadenskataster

Hierbei handelt es sich um eine spezialisierte, fortschreibende Applikation auf der Architektur des XML-Prototypen. Thema ist die Erfassung und Fortführung eines Baumkatasters (punktbezogen) im Straßennetz.

Es kann als Beispiel für die Lösung ähnlicher Erfassungsaufgaben gelten, wie z.B. Unfälle, etc.

Eigenschaften:

- Darstellung des Straßennetzes
- Wahlweise geeignete Hintergrundkarte, z.B. DTK10
- Geometrische Identifikation von Straßenpunkten zur Netzzuordnung der Bäume
- Erfassung und Änderung der Daten zu den Bäumen
- Suchfunktion zur Selektion von Bäumen nach Eigenschaften und per Maus
- Hervorheben des Suchergebnisses in der Grafik

Aufwandsabschätzung:

Arbeitspaket	Personentage
Konzeption, Abstimmung mit dem AG	3
Konfiguration der Datenbereitstellung (WFS, WMS/SLD)	6
Dienste in der Datenaufbereitung (teilweise durch Prototyp vorhanden)	2
Erstellung des Client	9
Dokumentation	2
Projektleitung, QS, KM	7
Gesamt	**29**

Ein solches System würde sinnvoll durch einen GPS-Anschluss erweitert, der ca. 6 PT in Anspruch nehmen würde. Eigenschaften des GPS-Anschlusses:

- Identifikation der Netzzuordnung über GPS
- Identifikation von Bäumen per GPS
- Automatische Bildnachführung

6 Umsetzung der Empfehlungen

Wir empfehlen in Evaluation der Ergebnisse aus dem XML-Prototypen, zunächst zu einer Meinungsbildung über den vorgeschlagenen Weg hin zu einem „Informationsmanagement" im Bereich der Straßeninformationsbanken zu gelangen.

Die Meinungsbildung sollte in einer Studie und einem abgestimmten Stufenplan für die Durchführung münden. Die Studie sollte das Potential der Lösung für die einzelnen Akteure ausleuchten und einen Mindestumfang der über Standardschnittstellen bereitzustellenden Daten und Dienste festlegen.

Die XML-basierten Web-Services, möglichst unter Nutzung internationaler Standards, bieten die Chance, kooperative, web-basierte Dienste einzurichten, die funktionieren, obwohl die konkrete Datenhaltung im Straßen- und Verkehrswesen auf unterschiedlichen Lösungen beruht. In den einzelnen Straßenbauverwaltungen eröffnen sie die Möglichkeit, Fachanwendungen über definierte Schnittstellen an die vorhandenen Netzdatenbestände anzuschließen und alle Datenbestände einer umfassenderen Nutzung zuzuführen.

Die bestehenden Straßeninformationsbanken sollten wenigstens im durch die Studie festgelegten Umfang mit Standardschnittstellen versehen werden (WFS, WMS/SLD), die dann die technischen Grundvoraussetzungen für den Anschluss von Fachsystemen und (sofern dies gewünscht wird) von länderübergreifenden Systemen bilden. Fachanwendungen, die auf denselben Standard-Schnittstellen aufsetzen, sind unabhängig von den zugrundeliegenden Straßeninformationsbanken und können deshalb in Kooperation der Länder entwickelt oder beschafft werden.

Anhang A – Erläuterung der Kurzbegriffe

AdV	Arbeitsgemeinschaft der Vermessungsverwaltungen der Länder der Bundesrepublik Deutschland. Die für die Landesvermessung und das Liegenschaftskataster zuständigen Verwaltungen der Länder und verschiedene Bundesministerien wirken zusammen, um fachliche Angelegenheiten von grundsätzlicher und überregionaler Bedeutung mit dem Ziel einer einheitlichen Regelung zu behandeln. Siehe http://www.adv-online.de/
CGI	Common Gateway Interface. Schnittstelle zur dynamischen Erzeugung von Web-Inhalten durch Programme oder Skipts auf der Seite des Servers. Typischerweise wird ein CGI-Programm/Skript via http wie eine HTML-Seite durch den Client aufgerufen und erzeugt dynamisch HTML, das dann an den Client als Response zurückfließt. Natürlich kann durch ein CGI statt HTML auch anderes Daten-Material, z.B. XML erzeugt werden. Für CGI gibt es keinen Standard, der dem Status von HTML, CSS oder XML vergleichbar wäre. Das W3C, bei dem es zwar auch eine Arbeitsgruppe gibt, die sich um das HTTP-Protokoll kümmert, klammert die CGI-Schnittstelle aus den HTTP-Dokumenten aus. Die CGI-Schnittstelle wurde mit einem der ersten und seinerzeit erfolgreichsten Web-Server-Produkte eingeführt, nämlich mit dem NCSA-Web-Server Die Original-Spezifikation für die CGI-Schnittstelle von NCSA ist auf den Web-Seiten von NCSA noch verfügbar: http://hoohoo.ncsa.uiuc.edu/cgi/interface.html
Client/Server	Asymmetrisches Kommunikationsmuster, bei dem eine Komponente für die Erbringung einer spezialisierten Leistung verantwortlich ist (Server) und die andere für die Inanspruchnahme (Client). Der Vorteil liegt vor allem in der mehrfachen Nutzung eines Servers durch viele Clients.
CORBA	Common Open Request Broker Architecture. Standard der Object Management Group (OMG), einem vorwettbewerblichen Konsortium mit mehr als 700 Mitgliedern. CORBA wurde 1991 in der ersten Version definiert. Der Standard definiert eine sogenannte „Middleware", d.i. Software, die das Bindeglied zwischen Client und Server darstellt. Der Anspruch von CORBA ist die Definition einer „objektorientierten" Middleware, welche eine orts-, plattform- und implementations-unabhängige Kommunikation zwischen Applikationen erlaubt. Wirklich interessant geworden ist CORBA seit der Verabschiedung der Version 2.0 im Dezember 1994. Diese Version brachte das Kommunikationsprotokoll IIOP, welches den Meldungsaustausch zwischen Object Request Brokern (ORB) verschiedener Hersteller und vor allem auch über das Internet ermöglicht. ORBs sind die technischen Implementationen des Standards CORBA. Ein ORB ermöglicht es einem Client, eine Meldung transparent an ein

	Serverobjekt zu senden, wobei das Serverobjekt auf derselben oder einer anderen Maschine laufen kann. Der ORB ist dafür zuständig, das Serverobjekt zu finden, dort die Funktion aufzurufen, die Parameter zu übergeben und das Resultat an den Client Bei CORBA-basierenden besteht eine strikte Trennung zwischen der Schnittstellendefintion eines Objektes und deren Implementation. Beim CORBA-Vorgehen wird zunächst die öffentliche Schnittstelle eines Objektes (d.h. die Funktionen) in der Interface Definition Language (IDL) definiert. IDL ist eine implementations-unabhängige Beschreibungssprache. In einem zweiten Schritt erst wird diese Definition ausprogrammiert, und zwar sowohl für den Client als auch für den Server-Teil. Dabei kann der Client beispielsweise in Java implementiert werden, während der Server in C++ programmiert wird. Einem umfassenden Erfolg von CORBA stehen einerseits die hohe Komplexität der Implementierungen und vor allem die Kollision mit dem Sicherheitsanforderungen von Web-Anbindungen entgegen. Web-Services stehen in direkter Konkurrenz zu CORBA. Sie sind einfacher und nutzen zur Kommunikation das http-Protokoll, für das Firewalls im Allgemeinen durchlässig ist.
Feature	Abstraktion von Erscheinungen der Welt, beschrieben durch wertebehaftete Eigenschaften skalarer und geometrischer Natur. Jede Eigenschaft besitzt Namen und Datentyp. Der einer Eigenschaft im Feature zugeordnete Wert entspricht diesem Datentyp (z.B. Text, Zahl, logischer Wert, Geometrie). Alle Features eines Feature Type liegt dieselbe Beschreibung durch Eigenschaften, also durch Namen und Datentyp zugrunde.
Filter Encoding	Durch das Open GIS Consortium festgelegte XML-Sprache für die Filterung von Features. Die Spezifikation liegt mehreren anderen OGC Spezifikationen zugrunde.
GML	Geography Markup Language. Durch das Open GIS Consortium festgelegte XML-Sprache für die Repräsentierung geographischer Sachverhalte beliebiger fachlicher Anwendungsbereiche. Ziel ist Zugriff und Austausch der Informationen über das Internet – vor allem durch web-basierte Dienste. GML spezifiziert, wie XML-Schema zu verwenden ist, um einen offenen, herstellerunabhängigen Rahmen für die Festlegung von raumbezogenen Anwendungsschemata und Objekten zur Verfügung zu stellen. Es ermöglicht die Beschreibung von Anwendungsschemata für Fachbereiche oder Anwendergruppen und lässt die Bildung von Profilen zu, die echte Untermengen des vollen GML-Schemaumfangs sind. Ziele sind die Erstellung und Pflege verteilter Anwendungsschemata und Datenbasen und die Speicherung und den Austausch von Anwendungsschemata und Daten zu unterstützen. Generell soll die Fähigkeit von Organisationen erhöht werden, geographische Informationen und deren Strukturbeschreibungen für Dritte verfügbar zu machen. Aktuelle Version ist die Version 3.0. Allerdings wird aus den relevanten GML-nutzenden Standards (z.B. Web Feature Server) noch auf deren Vorgängerversion 2.12 Bezug genommen.

HTML	HyperText Markup Language. Es handelt sich dabei um eine Sprache, die mit Hilfe von SGML (ISO-Norm 8879) definiert wird. Mittlerweile gibt es einen Ableger von HTML namens XHTML. In der Sprachversion 1.0 ist XHTML eine Redefinition von HTML mit Hilfe von XML. HTML ist eine Auszeichnungssprache (Markup Language). Sie hat die Aufgabe, die logischen Bestandteile eines textorientierten Dokuments zu beschreiben. Als Auszeichnungssprache bietet HTML daher die Möglichkeit an, typische Elemente eines textorientierten Dokuments, wie Überschriften, Textabsätze, Listen, Tabellen oder Grafikreferenzen, als solche auszuzeichnen. Web-Browser, die HTML-Dateien am Bildschirm anzeigen, lösen die Auszeichnungsmarkierungen auf und stellen die Elemente dann in optisch gut erkennbarer Form am Bildschirm dar. Eine der wichtigsten Eigenschaften von HTML ist die Möglichkeit, Verweise zu definieren. Verweise ("Hyperlinks") können zu anderen Stellen im eigenen Projekt führen, aber auch zu beliebigen anderen Adressen im World Wide Web und sogar zu Internet-Adressen, die nicht Teil des Web sind. Durch diese einfache Grundeigenschaft eröffnet HTML völlig neue Welten. Das Bewegen zwischen räumlich weit entfernten Rechnern wird bei modernen graphischen Web-Browsern auf einen Mausklick reduziert. Auf dieser Grundidee beruht letztlich das gesamte World Wide Web, und dieser Grundidee verdankt es seinen Namen.
http	Hypertext Transfer Protocol. Protokoll auf der Applikationsebene (des Netzwerkstacks) für verteilte, hypermedia-basierte Informationssysteme. http ist seit 1990 das Protokoll des World Wide Web. Die gültige Version von http ist HTTP/1.1, definiert durch RFC 2616. (http://www.ietf.org/rfc/rfc2616.txt?number=2616&) http definiert eine Reihe von Methoden zur Kommunikation von http-Client und http-Server. Die bekanntesten und in Web-Services benutzten sind GET, welche nur aus der Adresse der Netzwerk-Ressource und angehängten Argumenten besteht, und POST, in welcher Nutzdaten vom Client zum Server übertragen werden können. In beiden Fällen werden Nutzdaten an den Client zurückgeliefert. Browser als http-Clients benutzen i.A. GET zum Ansprechen der gewünschten Seite auf dem Server.
IETF	Internet Engineering Task Force IETF ist eine Aktivität der Internet Society ISOC (http://www.isoc.org/), einer internationalen, nichtstaatlichen Non-Profit-Organisation. Ziele sind die Entwicklung von Standards, öffentliche Einflussnahme, Ausbildung und Training im Umfeld des Internet. IETF (http://www.ietf.org/) befasst sich mit der Entwicklung der Internet-Architektur und dem störungsfreien Betrieb des Internet.

	Die Entwicklung von Standards wird in der Form eines Konsensprozesses auf der Basis von sog. RFCs vorgenommen.
ISO/TC211 191xx	Das Technical Committee 211 der ISO (International Organsisation for Standardization) befasst sich mit Standardisierung im Bereich digitaler geographischer Information. Die Arbeit zielt darauf ab, relevante Standards für Informationen bezüglich Objekte und Phänomene zu entwickeln, die direkt oder indirekt mit Punkten auf der Erdoberfläche verbunden sind. Die Standards (aus der Serie 191xx) spezifizieren Methoden, Werkzeuge, und Dienste zur Beschaffung, Verarbeitung, Analyse, Zugriff, Darstellung und Übertragung geographischer Information zwischen verschiedenen Benutzern, Systemen und Orten.
Java	Moderne, objektorientierte, plattformunabhängige Programmiersprache von Sun Microsystems. Die Plattformunabhängigkeit wird erreicht durch Übersetzung der Java-Programme in eine binäre Zwischensprache (Byte-Code genannt), die interpretiert wird. Nur die Programme zur Interpretation (die Java Virtual Machine) sind plattformabhängig. Die ursprüngliche Bestimmung von Java war die Programmierung von aktivem Code in HTML-Seiten in der Form sog. Java-Applets. Die Bedeutung dieser Applets geht jedoch zurück, da der meistverwendete Browser (der Microsoft Internet Explorer) nur eine stark veraltete Version von Java unterstützt und allgemein befürchtet wird, dass Microsoft die Unterstützung für Java in einiger Zeit ganz fallen lassen wird. Ungebrochen ist jedoch die Bedeutung von Java im Server-Umfeld (sog. Java-Servlets) und für das Erstellen plattformunabhängiger Applikationen, da in diesen Fällen die genannten Einschränkungen nicht gelten.
JavaScript	Hat nichts mit Java zu tun! Es handelt sich um eine prototypenbasierte (ein besonderes Prinzip der Objektorientierung) Programiersprache, die ursprünglich von Netscape für die Erstellung aktiven Codes in HTML-Seiten entwickelt wurde (also zum selben Zweck wie Java-Applets). Die Weiterentwicklung wurde jedoch vom W3C vorgenommen, inzwischen von ECMA. Die neueren Versionen der verbreitesten Browser (also MS IE und Netscape) unterstützen JavaScript in einem gewissen weitgehend kompatiblen Umfang.
lock	Sperre Im Kontext des Web Feature Servers ein Objekt, das den schreibenden Zugriff auf ein oder mehrere Features für alle Zugriffe verhindert, die nicht in Kenntnis des LockIds sind, die das Lock identifiziert.
NAS	Normbasierte Austauschschnittstelle. XML-basiertes Austauschformat für AFIS®-ALKIS®-ATKIS®. Es definiert die Operationen, die von AFIS®, ALKIS® oder ATKIS®-Datenservern unterstützt werden müssen. Die einzelnen Operationen sind jeweils als Request/Response-Paare in XML Schema beschrieben. Die AFIS®-ALKIS®-ATKIS®-Features sind hierbei in einem GML 3.0

	Anwendungsschema definiert, das aus dem zugrundeliegenden konzeptuellen, ISO 19100-konformen UML-Modell abgeleitet wurde. Das GML-Anwendungsschema ist ein zentraler Bestandteil der NAS und wird zur Repräsentierung der fachlichen Informationen verwendet.
OKSTRA®	Objektkatalog für das Straßen- und Verkehrswesen Mit dem OKSTRA®, entsteht in Deutschland zum ersten Mal ein umfassender Standard, der alle Bereiche vom Straßenentwurf über die Bestandsdokumentation bis zur Erfassung von Verkehrsdaten umfasst. Die Modellierung des OKSTRA® ist in Form von NIAM-Diagrammen gegeben, einer sehr intuitiven grafischen Darstellung von Objekten und ihren Beziehungen untereinander. Das aus den NIAM-Diagrammen abgeleitete Referenz-Datenschema des OKSTRA® ist in EXPRESS formuliert, einem lexikalischen ISO-Standard zur Modellierung von Objekten und deren Beziehungen untereinander. Das EXPRESS-Schema bildet den eigentlichen OKSTRA®-Standard. Die Schemabeschreibung in EXPRESS führt direkt zu einem ISO-definierten Austauschformat, genannt CTE (Clear Text Encoding). Für einen direkten Einsatz des OKSTRA® in relationalen Datenbanksystemen wurde eine Abbildung des EXPRESS-Referenz-Datenschemas nach SQL vorgenommen. Seit 09/2002 liegt als Entwurf ein weiteres abgeleitetes OKSTRA-Schema vor, nämlich OKSTRA®-XML. Hierbei handelt es sich um ein GML-Applikationsschema.
Open GIS Consortium (OGC)	OGC ist ein internationales Industriekonsortium aus mehr als 230 Unternehmen, Regierungsstellen und Universitäten. Die Mitglieder nehmen in einem Konsensprozess teil, der das Ziel verfolgt, öffentlich verfügbare Spezifikationen für den Umgang mit Geoinformation zu erzeugen. Die offenen Schnittstellen und Protokolle, die durch OpenGIS®-Spezifikationen definiert werden, unterstützen interoperable Lösungen die das Web, drahtlose und ortsbasierte Lösungen und die gesamte IT-Welt „geo-befähigen". Es wird Technologie geschaffen, die komplexe räumliche Information verfügbar und nützlich für die verschiedensten Applikation macht. Sehr wichtige (und erfolgreiche) OCG-Spezifikationen sind der Web Map Service (WMS), Web Feature Service (WFS) und Geography Markup Language (GML).
PHP	Hypertext Preprocessor PHP ist eine weit verbreitete, allgemeine Skriptsprache, die speziell für Web-Entwicklungen geeignet ist. Sie kann in HTML eingebettet werden. PHP ist ein Projekt der Apache Software Foundation.
SIB	Straßeninformationsbank
SMTP	Simple Mail Transfer Protocol Applikationsprotokoll für das Versenden von Mail. Die gültige Version wird definiert durch IETF RFC 821.

	(http://www.ietf.org/rfc/rfc0821.txt)
SOAP	Simple Object Access Protocol XML-basiertes Protokoll zum Austausch von strukturierten und typisierten Informationen. SOAP ist RPC-Middleware für das Internet, Funktion ähnlich zu CORBA. Transport kann sein HTTP, SMPT und andere. SOAP 1.2 ist ein Spezifikationsentwurf des W3C (2002). Es handelt sich inzwischen um ein relativ kompliziertes Protokoll. Der Standardisierungsprozess ist trotz großer Ressourcen noch nicht abgeschlossen.
URN, URI, URL	Uniform Ressource Name, Uniform Resource Identifier, Uniform Ressource Locator Siehe http://www.w3.org/Addressing/ URIs sind generische Bezeichner, die sich aus einem Adressierungsschema-Präfix und einem persistenten Namen zusammensetzen. URLs sind spezielle URIs mit dem Schemapräfixes: http, ftp, mailto, usw., z.B. http://www.w3.org/ URNs sind spezielle URIs mit dem Präfix urn: und weiteren Namespace-Qualifizierungen.
W3C	World Wide Web Consortium. Das W3C wurde 1994 gegründet. Es handelt sich um ein Konsortium aus Unternehmen, Regierungsstellen und Universitäten. Zur Zeit sind ca. 500 Stellen beteiligt. Ziel ist das Schaffen von Spezifikationen zur Weiterentwicklung des World Wide Web. Siehe http://www.w3.org/
Web Feature Server (WFS)	Standard des OpenGIS Consortiums. Web-Service zum Erfragen und Verändern von geometriebehafteten Features.
Web Map Server (WMS)	Standard des OpenGIS Consortiums. Web-Service zum Abrufen georeferenzierter Karten.
XLink	XML Linking XML-Standard des W3C. XML-Sprache zur Spezifikation von Links zwischen und innerhalb von XML-Dokumenten. Siehe http://www.w3.org/XML/Linking
XML	Extensible Markup Langunge XML ist ein einfaches, sehr flexibles Textformat des W3C, das als Profil von SGML (ISO 8879) definiert wurde. Ursprünglich als Hilfsmittel für das „"Elctronic Publishing" entworfen, hat sich die Bedeutung von XML inzwischen im Bereich des Datenaustausches im Web verfestigt. Siehe http://www.w3.org/XML/
XML Namespace	XML-Standard des W3C, der die Namen von XML-Elementen und –Attributen eindeutig macht. Dadurch wird die Wiederverwendung von XML-Strukturen in verschiedenen Zusammenhängen möglich, ohne dass es zu Kollisionen im Vokabular und zu Erkennungsproblemen

	kommt. Weltweit eindeutige Namensräume werden auf der Basis von URIs definiert, die innerhalb des Dokuments durch kurze, frei wählbare Namespace-Präfixe vertreten werden. Siehe http://www.w3.org/TR/REC-xml-names/
XML Schema	XML Schema ist eine Schemabeschreibungssprache für XML-Sprachen. XML Schema ist selbst XML, d.h. XML Schema ist selbst durch XML Schema definierbar. Es beinhaltet einen umfangreichen Vorrat an vordefinierten, einfachen Datentypen, das explizites Gruppieren von Attributen, die Konstruktion von Datentypen - auch durch Modifikation bereits bestehender, die Definition neuer Elemente auf Basis vorangegangener Definitionen. In XML Schema sind Multiplizitäten genau angebbar und Namesräume werden voll unterstützt. Siehe http://www.w3.org/XML/Schema
XPath	XML Path Language XPath dient zur Selektion von Teilen eines XML-Dokuments. XPath wird in mehreren ergänzenden Standards angewandt, z.B. XSLT, XPointer, XQuery. Es handelt sich um keine Markup-Sprache, d.h. XPath wird in Textbereichen, Attributwerten usw. eingesetzt. Xpath navigiert in einem logischen Modell der „Knoten" in einem XML-Dokument. Bewegung durch Schritte (XPath !) von einem „aktuellen Knoten" (Kontext) zu einem oder mehreren nächsten entlang logischer „Achsen". Die wichtigsten Achsen sind die child-Achse, die zu untergeordneten Elementen führt und die attribute-Achse, die zu den den Attributwerten führt. Siehe http://www.w3.org/TR/xpath
XSL/XSLT	eXtensible Stylesheet Language Dieser Standard besteht aus zwei Teilen, nämlich der Transformationssprache XSLT und der Formatierungssprache XSL (auch XSL-FO genannt). Eines der Hauptziele von XML ist die Trennung von Struktur, Inhalt und Format. D.h. XML-Dokumente enthalten keine Hinweise über das Aussehen als lesbares Dokument. Diese werden ihm durch „Stylesheets" hinzugefügt, sodass im Ergebnis ein lesbares Dokument entsteht. Zielvorstellung: „Multi-Channel Publishing" Siehe http://www.w3.org/Style/XSL/