

# **Entwicklung eines Web-Informationssystems mit SVG zur Erschließung von Dokumenten über den Raumbezug**

Master Thesis im Rahmen des Studiengangs UNIGIS MSc 2001 an  
der Universität Salzburg

vorgelegt von  
Martin Jäger (lkz 843)  
am 22. April 2003

## **Erklärung**

Hiermit erkläre ich, dass die vorliegende Arbeit selbständig und nur mit den angegebenen Hilfsmitteln angefertigt wurde und alle Stellen, die dem Wortlaut oder dem Sinn nach anderen Werken entstammen, durch Angabe der Quellen als Entlehnung gekennzeichnet sind.

Stuttgart, den 22. April 2003

## Danksagung

Ganz besonders möchte ich den Personen danken, ohne welche diese Master Thesis in dieser Form nicht möglich gewesen wäre:

- Herrn Ao. Univ.-Prof. Mag. Dr. Josef Strobl für die sehr gewinnbringende Betreuung dieser Arbeit.
- Herrn Dr. Michael Haase, Leiter des Bereichs Umweltinformationssysteme am FAW (Forschungsinstitut für anwendungsorientierte Wissensverarbeitung), der die Zusammenarbeit mit dem FAW ermöglichte und weiterführende Anregungen einbrachte.
- Herrn Dipl.-Ing. Adrian Raiber, dessen Diplomarbeit wichtige Anregungen für diese Abschlussarbeit gab.

## Inhaltsverzeichnis

Erklärung .....	i
Danksagung .....	ii
Inhaltsverzeichnis .....	iii
Abbildungsverzeichnis .....	v
Tabellenverzeichnis .....	vi
Abkürzungsverzeichnis .....	vii
Zusammenfassung .....	viii
Abstract .....	ix
<b>1 Einleitung .....</b>	<b>1</b>
<b>1.1 Umfeld der Arbeit .....</b>	<b>1</b>
<b>1.2 Grundlagen der Arbeit .....</b>	<b>3</b>
<b>1.3 Ziele der Arbeit .....</b>	<b>5</b>
<b>1.4 Problemstellung der Arbeit .....</b>	<b>5</b>
<b>1.5 Methodischer Ansatz .....</b>	<b>7</b>
<b>1.6 Gliederung der Arbeit.....</b>	<b>7</b>
<b>2 Web-Mapping .....</b>	<b>9</b>
<b>2.1 Internet und GIS.....</b>	<b>9</b>
<b>2.2 Kartographische Anforderungen .....</b>	<b>12</b>
2.2.1 Allgemeine Anforderungen an Internetkarten .....	12
2.2.2 Anforderungen an die Gestaltung der Karte des Informationssystems	14
2.2.3 Möglichkeiten der Minimierung internetspezifischer Restriktionen ....	17
<b>2.3 Web-Mapping-Technologien.....</b>	<b>19</b>
2.3.1 Einfache rasterbasierte Lösungen .....	19
2.3.2 Dynamisch generierte Rasterformate .....	22
2.3.3 Java Applets.....	23
2.3.4 Flash.....	25
2.3.5 Offene, XML-basierte Standards .....	29
2.3.5.1 Einführung in das Prinzip von XML .....	29
2.3.5.2 SVG (Scalable Vector Graphics) .....	34
2.3.5.3 GML (Geography Markup Language).....	43
<b>2.4 Hypothesen.....</b>	<b>51</b>

<b>3</b>	<b>Entwicklung des Web-Informationssystems .....</b>	<b>52</b>
<b>3.1</b>	<b>Systemarchitektur .....</b>	<b>52</b>
3.1.1	Webserver .....	53
3.1.2	Datenbank .....	53
3.1.3	Common Gateway Interface .....	54
3.1.4	Clientseitige Komponente.....	55
<b>3.2</b>	<b>Datenaufbereitung.....</b>	<b>57</b>
3.2.1	Generierung der SVG-Karte .....	57
3.2.2	Generierung der Legende .....	61
3.2.3	Konvertieren der Dokumente.....	62
<b>3.3</b>	<b>Prozesse beim Start des Systems .....</b>	<b>63</b>
<b>3.4</b>	<b>Umsetzung der interaktiven Funktionen .....</b>	<b>67</b>
3.4.1	Aufruf von Dokumenten über den Raumbezug.....	67
3.4.2	Ein- und Ausblenden der Layer .....	71
3.4.3	Manipulation der Legendeneinstellungen .....	72
3.4.3.1	Dokumentenbezug .....	74
3.4.3.2	Themenbezüge.....	75
3.4.3.3	Anzahl der Dokumente .....	79
3.4.4	Pan- und Zoom-Funktionen .....	80
3.4.5	Speichern von Benutzereinstellungen .....	84
<b>3.5</b>	<b>Aktualisierung des Systems.....</b>	<b>87</b>
3.5.1	Hinzufügen von Probestellen.....	87
3.5.2	Hinzufügen von Dokumenten .....	88
3.5.3	Inkonsistente Zustände .....	88
<b>4</b>	<b>Überprüfung der Hypothesen .....</b>	<b>90</b>
<b>5</b>	<b>Fazit .....</b>	<b>93</b>
	Literaturverzeichnis .....	96
	URL-Verzeichnis .....	100

## Abbildungsverzeichnis

Abb. 1: Ansicht der Shapefiles im Informationssystem von RAIBER (2001)..	4
Abb. 2: Geometrische Formen und deren Notation in SVG .....	35
Abb. 3: Systemarchitektur .....	52
Abb. 4: ER-Diagramm der Metadatenbank .....	54
Abb. 5: Einteilung der Oberfläche des Systems nach Funktionseinheiten...	56
Abb. 6: Statusleiste .....	56
Abb. 7: Vergleich der Probestellensymbole .....	57
Abb. 8: Legende des Web-Informationssystems .....	61
Abb. 9: Startseite des Web-Informationssystems .....	63
Abb. 10: Prozesse beim Start des Systems .....	64
Abb. 11: Teilprozess aus Abb. 10 mit Einbeziehung der Benutzererkennung	65
Abb. 12: Oberfläche des Web-Informationssystems.....	66
Abb. 13: Verlinkung von Probestellen mit Dokumenten.....	67
Abb. 14: Prozesse bei Abfrage nach Dokumenten .....	68
Abb. 15: Rückmeldung nach ergebnisloser Suche .....	69
Abb. 16: Suchergebnis der Dokumentenabfrage für Probestelle „19BFÖ“ ....	70
Abb. 17: Sichtbarkeit von Schaltfläche und Layer .....	71
Abb. 18: Toolbox mit Auswahlmenü für Legendeneinstellungen .....	72
Abb. 19: Prozesse bei der Änderung der Legendeneinstellungen.....	73
Abb. 20: Symbole für Legendeneinstellung „Themenbezüge“ .....	77
Abb. 21: Differenzierbarkeit kombinierter Probestellensymbole .....	77
Abb. 22: Kartenausschnitt mit unterschiedlichen Legendeneinstellungen .....	80
Abb. 23: Übersichtskarte mit Auswahlrechteck.....	81
Abb. 24: DB-Tabelle „UserSetting“ .....	85
Abb. 25: DB-Rückmeldung nach erfolgreichem Eintrag .....	86

## Tabellenverzeichnis

Tab. 1: Minimaldimensionierung in der Kartengraphik bei Bildschirmvisualisierung .....	14
---	----

## Abkürzungsverzeichnis

ARPA	Advanced Research Projects Agency
AWT	Abstract Windowing Toolkit
CGI	Common Gateway Interface
DB	Datenbank
DOM	Document Object Model
Dpi	dots per inch
DTD	Document Type Definition
ER-Diagramm	Entity-Relationship-Diagramm
ESRI	Environmental Systems Research Institute
FAW	Forschungsinstitut für anwendungsorientierte Wissensverarbeitung
FTP	File Transfer Protocol
GIS	Geo-Informationssysteme
GIV	Geographische Informationsverarbeitung
GML	Geography Markup Language
HTML	Hypertext Markup Language
JDBC	Java Database Connectivity
LfU	Landesanstalt für Umweltschutz Baden-Württemberg
ODBC	Open Database Connectivity
OGC	OpenGIS Consortium
RMI	Remote Method Invocation
SQL	Structured Query Language
SVG	Scalable Vector Graphics
VML	Vector Markup Language
W3C	World Wide Web Consortium
XML	Extensible Markup Language
XSL	Extensible Stylesheet Language

## Zusammenfassung

Das Ziel dieser Master Thesis besteht in der Entwicklung eines Web-Informationssystems, welches über eine interaktive Karte die räumliche Erschließung verteilter Dokumentenbestände ermöglicht.

Grundlage dafür ist ein auf ArcView basierendes Dokumenteninformationssystem, dessen räumliche Zugriffskomponente in geeigneter Art und Weise in das Internet transferiert werden soll. Die Dokumentenbestände des Systems stehen in Relation zu den Probestellen einer Retentionsfläche. Diese sollen im System kartographisch repräsentiert werden und über die Probestellensymbole der Karte einen räumlichen Zugriff auf die zu ihnen in Bezug stehenden Dokumente bieten.

Bei der Problemlösung werden zwei methodische Ansätze verfolgt. Die Sicherstellung kartographischer Standards und Präzisierung der Anforderungen an das System erforderte eine Auseinandersetzung mit den Erkenntnissen der thematischen Kartographie unter den besonderen Rahmenbedingungen des Internets. Daran schließt eine systematische Untersuchung und Bewertung verschiedener Web-Mapping-Technologien im Hinblick auf das zu entwickelnde System an.

Aufgrund der theoretischen Überlegungen wird die Umsetzung des Systems mit SVG (Scalable Vector Graphics) durchgeführt. In Verbindung mit den Skriptsprachen JavaScript und Perl ermöglicht dieser Ansatz die Gestaltung einer Webkarte mit weitreichenden interaktiven Funktionen bei relativ gutem Antwortzeitverhalten. So konnten u.a. die Durchführung von Datenbankabfragen über die Probestellensymbole der Karte, das Speichern von Benutzereinstellungen und die interaktive Änderung von Punktsignaturen der Karte ermöglicht werden. Letztgenannte Funktion erlaubt es, in die Probestellensymbole zusätzliche, alternativ abrufbare Informationen über die damit verknüpften Dokumente zu integrieren und damit die Suche zu vereinfachen.

Die Übertragbarkeit des gewählten Ansatzes sollte im Wesentlichen dann gegeben sein, wenn statische oder dynamische punktuell verortete Informationen räumlich erschlossen werden sollen.

## **Abstract**

### **Development of a Web Information System with SVG for the Spatial Retrieval of Documents**

The aim of this master thesis is the development of a web information system that enables the spatial retrieval of documents with the help an interactive map. The basis for this work is a document information system that was implemented in ArcView. Its spatial retrieval component has to be transferred adequately into the World Wide Web (WWW). Therefore this paper works on the field where GIS (Geographic Information Systems), the internet and document management meet.

The given documents contain data collected at measuring points within floodable lands at the river Rhine. A metadatabase stores the title and address of the documents. The measuring points serve as spatial keys to the documents to which they are linked to.

The web information system must meet the following requirements: a high-quality cartographic representation of the floodable lands and its measuring points, the possibility of querying the database via the symbols of the map, interactive functionalities that simplify the querying process, the minimization of user access limitations as well as the fulfilment of basic demands concerning performance and maintenance.

The aspects of this problem definition can be subdivided into two categories that require different methodical approaches. In order to meet the cartographic standards it is required to deal with the level of knowledge of thematic cartography with regard to the particular context of the WWW. How the signatures of the map have to be designed and which interactive functionalities are necessary for an intuitive, high-quality access to the documents will be derived in this discussion.

This follows a systematical examination of different web mapping technologies and examples of their implementation in order to evaluate to what extend these technologies are suitable for the development of the web information system. The discussion of these aspects leads to four hypotheses that state how the requirements of the system can be fulfilled most adequately. Afterwards the

system is realized according to the theoretically developed solution. Based on the results of this case study the hypotheses will be tested.

The discussion of the cartographic issues is about the limitations of web mapping and the demands that arise from these restrictions. The signatures of web maps should be designed as simple as possible and with few details due to the lack of control over the final presentation of internet maps. Interactive functionalities like pan and zoom can help to overcome some limitations. It is expected that integrating information about the documents into the symbols of the measuring points simplifies the querying process. The combination of different simple signatures is considered to be the best solution in providing more information. Because of the additional information (e.g. thematic categories of the documents, number of documents available) the user should be able to retrieve the documents via the measuring points much easier. In order to avoid a non-acceptable increase in information density different pieces of information must not be viewed at the same time. Therefore a functionality is necessary for interchanging the symbols of the measuring points.

The analysis of web mapping technologies considers the following aspects: data model, interactivity and the degree of openness. Starting with simple raster maps, they will turn out to be not suitable for these fields of application. This is especially true for interactivity and performance. Whereas the degree of interactivity can be improved with the use of a map server which dynamically generates maps on demand, the performance of the system can even get worse. In contrast to this Java Applets allow a very good implementation of the system, especially for interactivity. However, it takes a lot of effort to create or customize Java Applets for this purpose and this could also lead to sub-optimal performance. By comparison Flash is expected to have a very good performance as Flash is a binary and highly compressed format. Being proprietary, this format has shortcomings especially concerning data conversion steps. In contrast to this, open, XML-based standards are much more suitable. Using the markup language Scalable Vector Graphics (SVG), one can design high-quality vector-based maps with a high degree of interactivity and – at least for this application – a relatively good performance. Geography Markup Language (GML) is a language for the storage and transportation of

geographical data and therefore it offers more far-reaching possibilities than SVG (e.g. independence of storage and presentation, realization of service chains, individual presentation of the data). On the other hand, the use of GML would lead to an immense extra effort that is comparatively large in relation to its benefits. Therefore the development of the system is based on SVG.

This master thesis documents the development of the system, presenting its architecture, the data processing steps that were carried out, the interactive functionalities and their corresponding processes. This is done by showing code fragments and processing schemes.

As the development of the web information system shows, an open, non-proprietary technology meets the stated requirements most adequately. The designing of an interactive web map with SVG and the scripting languages JavaScript and Perl allows the implementation of far-reaching interactive functionalities like running database queries via the symbols of the measuring points in the map, fading-in and fading-out of map layers, interchanging of point signatures, saving of user settings and providing zoom and pan functionalities with the help of an overview map.

Requirements concerning the simplicity of maintenance and updating are fulfilled, too. One reason for this is the management of the document metadata within a database. Moreover, it turns out that additional measuring point symbols can be easily added to the map.

It should be possible to utilize the suggested architecture for the implementation of other document management applications whenever the spatial reference of the documents can be represented with point signatures. In addition to document management, static and especially dynamic information (e.g. up-to-date meteorological data) can be accessed with this approach most up-to-date. However, for dynamic or distributed vector data this approach is considered to be less suitable.

The main problem that occurred in the process of developing the web information system is that the symbols of the measuring points may cover one another. In that case the access to the linked documents would be blocked. The solution of this problem is considered to be a far-reaching object of research.

# 1 Einleitung

## 1.1 Umfeld der Arbeit

Diese Arbeit ist in den Überschneidungsbereich von Dokumentenmanagement, Internet und Geo-Informationssystemen (GIS) einzuordnen. Die genannten Begriffe sollen im Folgenden kurz definiert werden. Der Begriff Dokumentenmanagement bezeichnet *„the process of managing documents through their lifecycle. From inception through creation, review, storage and dissemination all the way to their destruction“* (DOCUMENT MANAGEMENT AVENUE 2001). Es handelt sich also um einen umfassenden, prozessorientierten Ansatz bei der Verwaltung von Dokumenten.

Unter dem Begriff „Internet“ versteht man eine Vernetzung von Computern. Gemeint ist damit *„nicht ein einziges homogenes Netz, sondern ein Verbund aus vielen kleinen, territorial oder organisatorisch begrenzten Netzen. Diese Netze besitzen eine Anbindung an die Backbones und damit an das Gesamtnetz“* (MÜNZ 1998, Kapitel Entstehung des Internet). Backbones sind Leitungs-Verbundsysteme, an welche kleinere Einzelnetze angeschlossen werden können.

GIS lässt sich definieren als *„a computer-based information system that enables capture, modelling, manipulation, retrieval, analysis and presentation of geographically referenced data“* (WORBOYS 1995, S. 1). Informationssysteme im allgemeinen Sinne sind Systeme zur Aufnahme, Speicherung, Verarbeitung, Auswertung und Wiedergabe von Informationen (vgl. BILL 1999). Dies ist so in der genannten GIS-Definition auch wieder zu finden. Es fällt jedoch auf, dass bei GIS zusätzlich explizit die Analysefunktion erwähnt wird. Die Möglichkeit zur Analyse ist neben dem räumlichen Bezug der Daten eine definierende, charakterisierende Funktion von GIS.

Für den Überschneidungsbereich von Internet und GIS lassen sich Bezeichnungen finden wie Online-GIS oder Web-Mapping. Web-Mapping nimmt *„auf die Verbindung von Geographischen Informationssystemen (GIS) mit den Leistungsmerkmalen des Internet bezug“* (STROBL 2001, S. 18). Die für

GIS entscheidende Funktion der Analyse findet jedoch bislang sehr selten Einzug in das Internet.

Während GIS und das Internet mehr und mehr zusammenwachsen (vgl. Kapitel 2.1), kommt auch der Verbindung von Dokumentenmanagement und GIS bzw. Web-Mapping eine größer werdende Bedeutung zu. Dies ist darauf zurückzuführen, dass viele der in Wirtschaft und Verwaltung erhobenen Daten einen Raumbezug aufweisen und der Mehrwert einer räumlichen Perspektive – bei der Darstellung oder Analyse – zunehmend erkannt wird. Geographische Zugangsportale auf Dokumentenbestände können dem Nutzer helfen, sich deren Inhalte durch das Wandern auf Karten intuitiv zu erschließen (vgl. SCHWARTZ und TOCHTERMANN 2000). In gängigen Dokumentenmanagementsystemen spielt dagegen der Raumbezug bislang eine untergeordnete Rolle und kann lediglich in textueller Form angegeben werden.

Am Beispiel von Umweltdokumentenbeständen soll der Mehrwert geographischer Zugangsportale für das Dokumentenmanagement verdeutlicht werden: In Deutschland besteht eine Informationspflicht für Behörden der Umweltverwaltung gegenüber der Öffentlichkeit. Auf Bundesebene gibt es beispielsweise den Bericht „Daten zur Umwelt“, welcher in digitaler Form vorliegt und über ein solches Informationssystem erschlossen werden kann. Gerade bei Umweltdokumenten ist es sehr häufig der Fall, dass sie nur innerhalb eines sehr begrenzten Raumes gültig sind. Dies kann eine administrative Raumeinheit sein, für die ein Dokument, insbesondere ein Gesetz oder eine Verordnung, entworfen wird. Oder es kann sich um eine ökologisch definierte Raumeinheit handeln, z.B. ein Biotop oder eine Retentionsfläche, in der Daten für ein Dokument erhoben werden.

Soll nun beispielsweise nach im Landkreis Rügen gültigen Verordnungen von Abfall- und Bodenschutzbehörden recherchiert werden, so muss in konventionellen Informationssystemen der thematische Suchbegriff und der räumliche Bezug in textueller Form eingegeben werden. Das Ergebnis der Suchanfrage ist dann in erheblichem Maße von der gewählten administrativen Ebene und der verwendeten Bezeichnung für die betroffene Gebietseinheit abhängig. In diesem Beispiel kann die Suche so fälschlicherweise zu einem

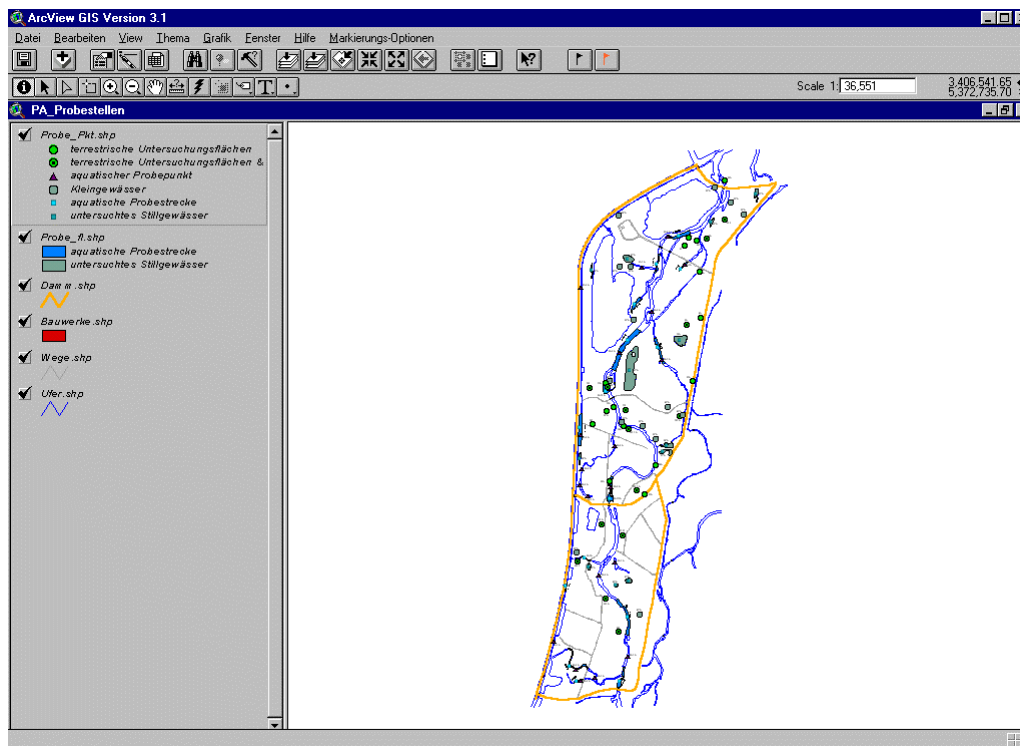
leeren Suchergebnis führen. Der Grund dafür liegt darin, dass solche Verordnungen meist auf Länderebene beschlossen werden, in diesem Fall also für Mecklenburg-Vorpommern. Die Verordnung ist damit zwar auch auf Rügen gültig, aber in aller Regel nicht explizit damit im Dokumenteninformationssystem verlinkt. Wird dagegen die Suche über ein geographisches Zugangsportale gestartet, so ist anhand der anklickbaren administrativen Flächen unmittelbar ersichtlich, auf welchen administrativen Ebenen nach Verordnungen gesucht werden kann. Gleichzeitig entfällt auf diese Weise die Gefahr leerer Suchanfragen aufgrund fehlerhaft eingegebener Gebietsangaben .

Geographische Zugangsportale bieten also ein großes Potenzial, Dokumente mit räumlich begrenzter Gültigkeit gezielter und intuitiver aufzusuchen als dies über Informationssysteme ohne räumlichen Zugang möglich ist. In der Verbindung von Internettechnologien, GIS und Dokumentenmanagement besteht die Chance, derartige Zugänge einer breiten Öffentlichkeit anzubieten.

## **1.2 Grundlagen der Arbeit**

Grundlage der Master Thesis ist ein Informationssystem, das von Adrian RAIBER im Rahmen seiner Diplomarbeit entwickelt wurde. Die Arbeit mit dem Titel „Archivierung und Analyse von geoökologischen Untersuchungsergebnissen mit Hilfe eines GIS und Erhebung der Daten mittels DGPS“ (vgl. RAIBER 2001) beschäftigte sich u.a. mit dem Zugriff, der Verwaltung und Analyse von heterogenen und verteilten Dokumentenbeständen mit vorwiegend räumlichem Bezug. Die Implementierung wurde unter Einsatz der GIS-Software ArcView 3.1 beispielhaft für die Retentionsfläche „Polder Altenheim“ durchgeführt. Dieses Gebiet liegt am Oberrhein in der Nähe der Stadt Lahr im Südwesten Deutschlands. Die Dokumente mit räumlichen Bezügen zu diesem Untersuchungsgebiet wurden von verschiedenen Dienstleistern und Sachverständigen für die Gewässerdirektion Südlicher Oberrhein/Hochrhein, Projektgruppe Lahr, erhoben und liegen als Word- oder Excel-Dateien, z.T. auch als ESRI (Environmental Systems Research Institute) Shapefiles vor. Für die Verwaltung der Dokumente wurde eine MS Access-Datenbank aufgebaut, die Metadaten über die Dokumentenbestände enthält.

Die geometrische Information des Untersuchungsgebietes liegt in Form von Vektordaten vor und ist in verschiedene Layer untergliedert. Einige davon dienen ausschließlich der Orientierung. Dazu gehören beispielsweise Uferlinien, Wege und Dämme, welche mit Hilfe von Polygonzügen dargestellt werden. Von größerer Bedeutung sind die punktuellen Objekte. Sie repräsentieren Probestellen des Untersuchungsgebietes und liefern als Erhebungsquellen die inhaltliche Grundlage für zahlreiche Dokumente. Daher weisen einige der im Datenmodell umgesetzten Probestellen Bezüge zu diesen Dokumenten auf. Die Verknüpfung der Dokumente wurde bislang nur für die in der Legende mit „terrestrische Untersuchungsflächen & LfU Dauerbeobachtungsflächen“ und „terrestrische Untersuchungsflächen“ betitelten Probestellen realisiert.



**Abb.1: Ansicht der Shapefiles im Informationssystem von RAIBER (2001)**

Quelle: RAIBER (2001)

Diese Master Thesis verwendet die Metadatenbank sowie grundlegende ESRI Shapefiles des Informationssystems von RAIBER (2001) (vgl. Abb. 1) als Grundlage für die Entwicklung des Web-Informationssystems.

### 1.3 Ziele der Arbeit

Das Ziel dieser Arbeit liegt darin, die räumliche Zugriffskomponente des in Kapitel 1.2 vorgestellten proprietären Dokumentenmanagementsystems in ein internetbasiertes Auskunftssystem zu transferieren. Analysefunktionen sowie Funktionen des Systems für die Dokumentenverwaltung sollen nicht übernommen werden. Demzufolge soll es sich dabei um ein reines Frontend-Auskunftssystem handeln, welches den Zugang auf die Dokumente im Internet über eine Karte des Untersuchungsgebietes ermöglichen und damit einem breiteren Publikum zur Verfügung stellen soll. Es wird also nicht das Ziel verfolgt, ein vollständiges GIS oder ein umfassendes Dokumentenmanagementsystem zu entwickeln.

### 1.4 Problemstellung der Arbeit

Eine Besonderheit des zu entwickelnden Web-Informationssystems ist sicherlich, dass die Probestellen und damit die geometrischen Grundlagen als relativ statisch zu bezeichnen sind, wohingegen die damit verknüpften Informationen bzw. Dokumente einer wesentlich höheren Dynamik unterliegen. Daraus und aus der im vorangegangenen Kapitel beschriebenen Zielsetzung ergeben sich spezifische Anforderungen an das System, welche sich von dem in ArcView implementierten System zum Teil beträchtlich unterscheiden:

- **Kartographische Darstellung des Untersuchungsgebietes**

Das System muss dem Benutzer eine räumliche Repräsentation des Untersuchungsgebietes bereitstellen, welche die einfache und intuitive Erschließung des vorliegenden Dokumentenbestandes ermöglicht.

- **Durchführen von Datenbankabfragen über Symbole in der Karte**

Eine der wichtigsten Anforderungen liegt darin, dass Datenbankabfragen über die Symbole der Probestellen gestartet und damit die dazu in Bezug stehenden Dokumente aufgerufen werden können. Diese Forderung ist elementar, da sie sich auf die grundlegendste Funktion des Systems bezieht.

- **Variabler Maßstab**

Es sollte die Möglichkeit bestehen, in verschiedenen Maßstabsebenen zu arbeiten. Retentionsflächen an Flüssen sind natürlicherweise langgestreckte Formen, die in voller Ansicht die Bildschirmfläche nur zu einem geringen Teil ausfüllen. Ein Überblick über die gesamte Fläche ist bei eher kleinen Maßstäben möglich, wohingegen für das Auffinden und die Auswahl der Probestellen ein größerer Maßstab vorteilhafter wäre. Daher sollten Möglichkeiten zur Navigation und zur Veränderung des Maßstabes gegeben sein.

- **Veränderung des Karteninhalts**

Dem Benutzer sollte die Möglichkeit gegeben werden, den Inhalt der Karte interaktiv an seine Bedürfnisse anzupassen. Der notwendige bzw. zielführende Grad an Interaktivität ist im Folgenden noch zu diskutieren (vgl. Kapitel 2.2.2 und 2.2.3).

- **Einfacher und kostengünstiger Zugang**

Um eine Vielzahl an Benutzern an diesem Web-Informationssystem teilhaben zu lassen, sollte der Zugang mit möglichst geringen Kosten in Form von Zeit und Geld verbunden sein. Es ist dafür Sorge zu tragen, dass mögliche Hürden und Restriktionen für die Benutzung des Systems minimiert werden.

- **Akzeptable Performance**

Der Zugang über das Internet erfordert zudem geringe Ladezeiten, um einen zügigen Aufbau und ein gutes Antwortzeitverhalten des Systems auch über Modems mit durchschnittlichen Übertragungsraten zu gewährleisten. Diese Anforderung ist für eine breite Akzeptanz im Internet von nicht zu unterschätzender Bedeutung.

- **Einfache Wartung und Aktualisierung**

Schließlich besteht die Notwendigkeit, das System so zu gestalten, dass ein möglichst geringer Wartungsaufwand besteht. Das Hinzufügen von Probestellen, aber insbesondere von Dokumenten sollte daher möglichst rasch und unter Vermeidung von Inkonsistenzen durchgeführt werden können. Dem kommt der bereits bestehende Datenbank-Ansatz sehr

entgegen. Diese Forderung impliziert auch, dass die Architektur möglichst langfristig ausgelegt sein sollte, um gravierende Umstellungen des Systems zu vermeiden.

## **1.5 Methodischer Ansatz**

Für die Entwicklung des Web-Informationssystems erfolgt eine Auseinandersetzung mit grundlegenden kartographischen Anforderungen sowie mit speziellen Anforderungen an interaktive Internetkarten. Die dabei gewonnenen Erkenntnisse sind auf die konkrete Problemsituation bei der Entwicklung des Web-Informationssystems anzuwenden. Dies ist unbedingt erforderlich, um eine qualitativ gute Internetkarte zu erstellen, die kartographischen Standards entspricht und ein geeignetes Maß an Interaktivität aufweist.

Zudem werden die bedeutendsten Web-Mapping-Technologien untersucht und in ihrer Eignung für die Umsetzung des geplanten Systems bewertet. Dies erfolgt auf theoretischer Basis und mit Bezugnahme auf bereits implementierte Systeme. Dabei kann eine umfassende oder repräsentative Auswahl angesichts der Vielzahl an Technologien bzw. Lösungen und der gegenwärtigen Dynamik in diesem Bereich kaum beansprucht werden. Stattdessen wird angestrebt, Technologien nach Kriterien wie Datenmodell, Grad der Offenheit und Interaktivität zu differenzieren und ihre grundlegenden Vor- und Nachteile für das zu entwickelnde System herauszuarbeiten. Die Technologie, welche den genannten Anforderungen am ehesten gerecht wird, ist für die Entwicklung heranzuziehen und geeignet umzusetzen.

## **1.6 Gliederung der Arbeit**

Nach der einführenden Erläuterung von Umfeld, Grundlage und Ziel der Arbeit werden die grundlegenden Anforderungen an das zu implementierende System als die zentrale Problemstellung der Arbeit genannt.

Anschließend soll in Kapitel 2.1 die Bedeutung des Zusammenwachsens von Internet und GIS näher beleuchtet und in Kapitel 2.2 Forderungen diskutiert werden, die eine kartographisch gute Qualität von interaktiven Internetkarten

gewährleisten. In Kapitel 2.3 erfolgt eine Auseinandersetzung mit den technischen Grundlagen verschiedener Internet-Technologien und Beispielen bereits implementierter Lösungen. Die Technologien werden im Hinblick auf ihre Eignung für das zu entwickelnde System bewertet.

In Kapitel 3 wird die Entwicklung des Systems beschrieben. Dabei sollen die Architektur, die durchgeführten Datenaufbereitungsschritte, die interaktiven Funktionen sowie die dahinter liegenden Prozesse des Systems vorgestellt werden.

Kapitel 4 nimmt eine Überprüfung der aufgestellten Hypothesen auf Basis der bei der Entwicklung des Informationssystems erlangten Erfahrungen und Erkenntnisse vor.

Abschließend werden in Kapitel 5 die Ergebnisse der Arbeit zusammengefasst und deren Übertragbarkeit für die Entwicklung weiterer kartenbasierter Web-Informationssysteme bewertet. Aus den festgestellten Unzulänglichkeiten des Systems werden weiterführende Forschungsfragen abgeleitet.

## 2 Web-Mapping

### 2.1 Internet und GIS

An dieser Stelle soll zunächst in groben Zügen die Entwicklung des Internet und die Entwicklung von GIS skizziert werden, um anschließend auf die Verbindung dieser Technologien und der daraus resultierenden Chancen und Möglichkeiten einzugehen.

Die Anfänge des Internet gehen bis in die 60er Jahre des 20. Jahrhunderts zurück. Es entstand als elektronisches Datennetz zur verteilten Speicherung militärischer Daten. Die zum US-Militär gehörende Advanced Research Projects Agency (ARPA) zeichnete sich für diese Entwicklung verantwortlich. Daher wurde das Internet zunächst ARPA-Net bezeichnet. Seither hat das Internet eine rasante Entwicklung durchlaufen. Es entstanden zahlreiche Dienste wie FTP, E-Mail, Telnet, Newsgroups oder das World Wide Web (WWW). Erst letzteres brachte dem Internet den Durchbruch in der Öffentlichkeit. Zuvor war es vorwiegend auf Nutzungen im militärischen und wissenschaftlichen Bereich beschränkt gewesen. Das WWW hatte zunächst das Ziel, wissenschaftliche Dokumente online verfügbar zu machen. Diese sollten in Form von Texten und Bildern abgerufen werden können.

Als Hauptgrund für die schnelle Verbreitung und Akzeptanz des WWW ist das Dateiformat HTML (Hypertext Markup Language) zu nennen. Die Besonderheit daran ist, dass von einem Dokument Verweise (Links) auf andere Dokumente gesetzt werden können. Damit ist eine Vernetzung von Dokumenten möglich. Von großer Bedeutung war auch, dass HTML nicht als binäres Format, sondern als textbasiertes Dateiformat konzipiert wurde. Die Einfachheit der Kodierung und die einfache Lesbarkeit (auch für Nicht-Programmierer) hatte sicherlich einen beträchtlichen Anteil am Erfolg des WWW (vgl. MÜNZ 1998).

Die Geschichte von GIS reicht – wie auch die des Internet – bis in die 1960er Jahre zurück. Als erstes GIS gilt heute das Canada Geographic Information System. Dieses System diente der Inventarisierung und Auswertung natürlicher Ressourcen in Kanada und wurde von einem Team um Roger Tomlinson

entwickelt. Systeme dieser Art liefen vorwiegend auf UNIX-Workstations. GIS blieb lange Zeit weitgehend abgeschottet von anderen Bereichen der EDV. Eine etwas weitere Verbreitung – besonders in öffentlichen Einrichtungen, vereinzelt aber auch in privaten Unternehmen – konnte erst erreicht werden, als die führenden Hersteller der GIS-Branche ihre Software nicht nur für Workstations, sondern auch für PCs mit Windows-Betriebssystemen entwickelten (Desktop-GIS). Damit war bereits eine Kopplung mit Büro-Anwendungen und eine Annäherung an andere EDV-Welten möglich.

Während das Internet den Durchbruch in der Öffentlichkeit in den 1990er Jahren erfuhr, steht der endgültige Durchbruch von GIS noch bevor. Interessanterweise kann gerade das Internet diesen Prozess anstoßen bzw. beschleunigen. Denn erst mit der Integration von Funktionen der geographischen Informationsverarbeitung (GIV) in das Internet kann Geoinformation und die darauf bezogenen Analysen einer breiten Öffentlichkeit zugänglich gemacht werden und auf diese Weise endgültig den Status einer sehr speziellen, z.T. von anderen Technologien abgekapselten Fachrichtung verlassen. Dann erst kann GIS, über Behörden und private Unternehmen hinaus, auch für private Aktivitäten eine gewisse Bedeutung erlangen. Man denke nur an die zunehmende Verbreitung und Nutzung von Routenplanern im Internet oder die Möglichkeiten, die GIS-Dienstleistungen im Bereich der Location Based Services versprechen.

An dieser Stelle soll anhand einiger Aspekte zusammenfassend dargestellt werden, welche Bedeutung der Kombination von GIS und Internet zukommt:

- **Verbreitung von Geoinformation sowie von Verarbeitungs- und Darstellungsmöglichkeiten**

Dieser Gesichtspunkt wurde bereits deutlich herausgestellt. Die Verfügbarkeit von Karten und kartenbasierten Analysewerkzeugen steigt zunehmend. Damit verbunden ist auch ein Wandel der Anforderungen an die GIV, insbesondere an die Web-Kartographie. Darauf wird im folgenden Kapitel näher eingegangen.

- **Plattformunabhängigkeit**

Können GIS-Anwendungen in einen Browser oder ein Plugin integriert werden, so sind sie unabhängig von Hardware und Betriebssystem einsetzbar. Dies ist für eine weite Verbreitung von GIS von grundlegender Bedeutung.

- **Integration von Geoinformation**

Geoinformation kann im Internet mit anderen, nicht-räumlichen Informationen, mit anderen Technologien und Medien integriert werden. Ein Beispiel dafür ist die Verknüpfung von administrativen Geodaten mit unternehmenseigenen Sachdaten, z.B. im Geomarketing.

- **Interoperabilität**

Die Zusammenführung von Internet und GIS eröffnet neue Möglichkeiten, nicht nur verteilte Datenhaltung, sondern auch verteilte Datenverarbeitung zu realisieren. Wenn Softwarekomponenten flexibel zusammengesetzt werden und miteinander kommunizieren können, dann können nach Bedarf des Kunden verschiedene Dienstleistungen in Form einer Dienstleistungskette ausgeführt werden, bei welcher jede Komponente einen Teil der geforderten Dienstleistung erfüllt (z.B. Koordinatentransformation, Änderung der Legendenklassifikation).

- **Aktualität**

In diesen Bereich fällt v.a. der Begriff „Echtzeit-GIS“. Eine hohe Aktualität kann über das Internet einfacher geschaffen bzw. aufrechterhalten werden als über andere Medien, wie z.B. Papierkarten oder CD-ROM. So können aktuelle Daten „on-the-fly“ bereitgestellt und evtl. in Form einer Karte ausgegeben werden.

- **Höherer Nutzen des Internet**

Nachdem das Internet Ende der 1990er Jahre einen regelrechten Hype erlebt hatte, haben sich die in das Internet projizierten wirtschaftlichen und gesellschaftlichen Erwartungen etwas gemäßigt. In der gegenwärtigen Konsolidierungsphase sind umso mehr Dienstleistungen und Inhalte gefragt, die einen wirklichen Nutzen bieten können. Hier liegt für die GIV eine große Chance, einen solchen Mehrwert zu schaffen.

## 2.2 Kartographische Anforderungen

Autoren konventioneller, auf Papier gedruckter Karten haben nahezu uneingeschränkte Kontrolle auf deren Darstellung. Das Erscheinungsbild von über das Internet publizierten Karten hängt dagegen von mehreren Faktoren ab, die nicht im Einflussbereich eines Kartenautors liegen. Einige Faktoren sind technischer Art und auf die Restriktionen der Darstellung am Bildschirm zurückzuführen. Andere resultieren aus der Forderung nach einem hohen Maß an Interaktivität von Internetkarten.

Im Folgenden sollen einige dieser Faktoren diskutiert werden und daraus in Kapitel 2.2.2 Forderungen an die Gestaltung der Karte des zu erstellenden Informationssystems abgeleitet werden. Dies ist auch insofern von großer Bedeutung, da die allgemeinen Anforderungen an das Informationssystem mit den kartographischen Anforderungen in wechselseitiger Beziehung stehen. Es ist also nicht nur die kartographische Darstellung von den Anforderungen an das Informationssystem abhängig, vielmehr haben die Restriktionen der Darstellung auch unmittelbare Folgen für den notwendigen Grad an Interaktivität. Dieser soll auf Basis der folgenden Ausführungen in Kapitel 2.2.3 konkretisiert werden.

### 2.2.1 Allgemeine Anforderungen an Internetkarten

*„Wie jede Kartenerstellung beruht auch die netzbasierte Verarbeitung und Visualisierung raumbezogener Informationen auf den Grundsätzen der konventionellen Kartenbearbeitung“* (ASCHE 2001, S. 13). Denn auch Internetkarten sind im Allgemeinen eine maßstäblich verkleinerte, auf eine Ebene projizierte Repräsentation (von Teilen) der Erdoberfläche unter Wahrung der Lagebeziehung der dargestellten Objekte. Es müssen also allgemeine kartographische Prinzipien wie die Angabe des Maßstabes, eine angemessene Inhaltsdichte und Konventionen für kartographische Signaturen beachtet und auf den Anwendungskontext des Internet bezogen werden.

Im Unterschied zur konventionellen Kartographie müssen bei Internetkarten Abstriche in der Darstellungsqualität akzeptiert werden. Für gedruckte Karten

gilt heute ein Standard von 1.000 dpi (dots per inch), es können aber sogar bis zu 2.400 dpi erreicht werden (vgl. DICKMANN 2001). Gängige Monitore ermöglichen dagegen heute nur Auflösungen zwischen 72 und 80 dpi. Eine Folge davon ist, dass bei Linien, die von der Bildpunktmatrix des Monitors abweichen, ein gezackter Verlauf entsteht (Treppeneffekt oder Aliasing). Diverse Formate erlauben eine Abmilderung dieses Effekts über eine weichere, mitunter etwas unscharfe Darstellung (Antialiasing).

Eine weitere Folge der verminderten Auflösung ist, dass die Inhaltsdichte gegenüber konventionellen Karten verringert werden sollte. Zu beachten ist in diesem Zusammenhang auch die Anhebung kartographischer Mindestgrößen. Lassen sich bei einer Auflösung von 2.400 dpi noch Linien mit einer Strichstärke von unter 0,1 mm erkennen, so ist am Bildschirm die Wahrnehmung von Linien unter 0,35 mm kaum möglich (vgl. DICKMANN 2001). Nach DICKMANN (2001) sollte u.a. auf die Verwendung von Flächenmustern, gerissener Linien und auf Symbole mit einem hohen Ikonizitätsgrad verzichtet werden. Stattdessen sollen Signaturen möglichst generalisiert und in einer Mindestflächengröße von 3x3 mm dargestellt werden. Schrift sollte in einer Schriftgröße von mindestens 12 Punkt, serifenlos und nur in horizontaler Ausrichtung dargestellt werden. Insgesamt verringern sich damit die Möglichkeiten kartographischer Darstellung beträchtlich. Insbesondere die Gestaltung sprechender Signaturen dürfte so kaum geeignet zu realisieren sein. Der Begriff Auflösung hat über die Anzahl der Pixel pro Flächeneinheit eine weitere Bedeutung: die horizontale und vertikale Anzahl der Pixel am Bildschirm. Diese Auflösung kann für verschiedene Bildschirme unterschiedlich eingestellt sein, was wiederum erheblichen Einfluss auf die Darstellungsgröße und Erkennbarkeit von Signaturen hat. Dies verdeutlicht Tabelle 1 mit der Auflistung der Minimalgrößen verschiedener Kartengraphiken in Abhängigkeit von der Bildschirmauflösung. Hinzu kommt, dass bei der Wahl einer geringeren Auflösung Karten insgesamt größere Ausmaße erreichen. Dieser Umstand macht es sehr schwierig, Karten zu erstellen, die von einer breiten Nutzerschicht in einer optimalen Größe (nicht zu viele Leerflächen, Erfassen der Karte ohne Scrollen) betrachtet werden können.

**Tab. 1: Minimaldimensionierung in der Kartengraphik bei Bildschirmvisualisierung**

Minimaldimension	Auflösung (17" Lochmasken-Bildschirm)		
	640 x 480	800 x 600	1.280 x 1.024
<b>Strichstärke</b>	0,5 mm	0,3	0,3
<b>Linienabstand</b>	0,6 mm	0,6	0,5
<b>Formerkennbarkeit: Dreieck</b>	2,3 mm	1,2	1,2
<b>Linienabstand</b>	1,4 mm	1,2	1,2
<b>Formerkennbarkeit: Kreis</b>	3,3 mm	2,7	1,7

Quelle: MALIC 1998, S. 108f., zitiert nach DICKMANN 2001, S. 161

In Abhängigkeit von Betriebssystem, Leistungsfähigkeit der Grafikkarte, gewählter Farbtiefe, Bildschirmgröße und individuellen Bildschirmereinstellungen (z.B. Helligkeit, Kontrast) wird die Wiedergabe einer Karte zudem beträchtlich beeinflusst. Auch aus diesem Umstand ist für die Web-Kartographie eher die Verwendung einfacher, gut unterscheidbarer Signaturen abzuleiten.

## 2.2.2 Anforderungen an die Gestaltung der Karte des Informationssystems

Die Folgen der genannten Probleme für die Gestaltung der Karte des zu entwickelnden Web-Informationssystems werden nachfolgend anhand der Probestellensymbole diskutiert. Eine derartige Beschränkung ist damit zu begründen, dass alle weiteren Signaturen lediglich einer besseren Orientierung dienen (z.B. Uferlinien, Wege). Diese flächen- und linienhaften Signaturen können vermutlich – unter Beachtung von Mindestgrößen und einer farblichen Differenzierbarkeit – unverändert in das System übernommen werden. Eine adäquate Gestaltung der Probestellensymbole ist aber umso wichtiger, damit eine möglichst einfache und schnelle Suche nach Dokumenten möglich ist. Dies sollte dann gegeben sein, wenn erstens das Symbol einer Probestelle eindeutig einem Probestellen-Typ zugeordnet werden kann und zweitens möglichst

zielführende Aussagen über die mit den Probestellen verknüpften Dokumente visualisierbar sind.

Der ersten Forderung sollte im Kontext des WWW am ehesten mit der Repräsentation der Probestellen durch einfache geometrische Formen entsprochen werden können. Dabei sollte eine Unterscheidbarkeit in Form und/oder Farbe gegeben sein. Eine Differenzierung mittels sprechender Signaturen würde zwangsläufig zu einem sehr hohen Ikonizitätsgrad führen, wenn die nur graduellen Unterschiede der Probestellentypen zum Ausdruck gebracht werden sollten. Dies ist aufgrund der Darstellungsrestriktionen von Internetkarten kaum anzustreben. Stattdessen sollte versucht werden, die Identifizierbarkeit der Symbole mittels geeigneter Farbassoziationen zu verbessern.

Um der zweiten Forderung zu genügen, sollte der Suchprozess nach den Dokumenten über eine möglichst frühzeitige qualitative oder quantitative Eingrenzung der Suche erleichtert werden können. Dafür ist die Integration weiterer Informationen in die Probestellensymbole notwendig, so dass diese eine Aussage über die mit ihnen verknüpften Dokumente gestatten. Zu beachten ist jedoch, dass die Aussagedichte der Probestellensymbole nur sehr begrenzt erhöht werden kann. Dies ist insbesondere auf die genannten darstellungsbezogenen Restriktionen zurückzuführen. Dazu kommt, dass mit steigender Aussagedichte auch eine Zunahme der Komplexität der Karte einhergeht. Dies könnte einen Benutzer des Informationssystems aber leicht überfordern oder zumindest eine längere Auseinandersetzung mit der Karte erzwingen. Die Bereitschaft dazu ist bei Internetkarten jedoch im Vergleich zu konventionellen Karten in der Regel geringer (vgl. DICKMANN 2001). Aus diesen Gründen führt eine höhere Informationsdichte nicht grundsätzlich zu einer einfacheren Erschließung der Dokumente.

Ein vielversprechenderer Ansatz dürfte darin liegen, die Informationen über die Dokumente einer Probestelle nicht gleichzeitig anzuzeigen. Es wäre vielmehr vorteilhafter, derartige Informationen alternativ zueinander zu visualisieren und dem Benutzer eine Funktion zur interaktiven Auswahl der Informationen bereitzustellen. Auf diese Weise sollten dem Benutzer deutlich mehr

Informationen zugänglich gemacht werden können und gleichzeitig die Informationsdichte und die Komplexität des Kartenbildes relativ gering zu halten sein.

Eine weitere wichtige Frage liegt darin zu klären, wie die zusätzlichen Informationen geeignet in die Grundsymbole integriert werden können. Dabei muss noch mehr als bei der Gestaltung der Grundsymbole eine sehr einfache Lösung bevorzugt werden. Dies kann mit der Kombination der Grundsymbole der Probestellen mit nur wenigen sekundären Symbolen verwirklicht werden. In diesem Zusammenhang sollten zur Gewährleistung der Kombinationsfähigkeit primärer und sekundärer Symbole auch letztere geometrisch möglichst einfach gestaltet und farblich gut unterscheidbar sein. Ist auch in der kombinierten Form die Bedeutung der Symbole eindeutig abzulesen, so können mit diesem Ansatz qualitative Informationen (z.B. ein thematischer Bezug zu den mit einer Probestelle in Relation stehenden Dokumenten) dem Systembenutzer zugänglich gemacht werden.

Etwas problematischer dürfte die Visualisierung quantitativer Informationen sein. Für das zu entwickelnde Informationssystem ist dafür insbesondere die Variierung der Probestellensymbole abhängig von der Anzahl der damit verknüpften Dokumente von Interesse. Das Hauptproblem liegt in diesem Fall darin, dass die Anzahl der zu einer Probestelle in Bezug stehenden Dokumente sehr variabel sein kann. Dadurch kann auch die Spannweite der Anzahl der Dokumentenbezüge (maximale Anzahl der Dokumente einer Probestelle abzüglich der minimalen Anzahl der Dokumente einer Probestelle) erheblich variieren. Diese Dynamik ist für die Form der Visualisierung der quantitativen Informationen maßgeblich. So ist die Variation der Größe der Probestellensymbole in Abhängigkeit von der Anzahl der Dokumente als ungeeignet einzustufen. Denn dies könnte – insbesondere bei sehr großen Spannweiten – zu sehr großen Symbolen führen, die sich gegenseitig überdecken und die Betrachtung bzw. das Anklicken mancher Probestellen nicht mehr zulassen. Aus demselben Grund ist auch eine Darstellung nach dem bildstatistischen Prinzip nicht möglich. Dabei würde eine Probestelle je nach Anzahl der Dokumente durch eine Vielzahl von Symbolen repräsentiert, wobei ein Symbol für  $n$  Dokumente stehen würde. Eine weitere Möglichkeit wäre eine abgestufte

Rasterung der Symbole, wobei eine zunehmende Anzahl an Dokumenten eine dichtere Darstellung eines Grundmusters zur Folge hätte. Dies lässt sich allerdings nicht mit der geforderten Einfachheit der Signaturen in Einklang bringen. Je nach eingestellter Auflösung wäre ein Unterschied der Signaturen kaum noch erkennbar. Dies gilt in gleichem Maße für die Darstellung mittels Farbabstufungen. Daher wird an dieser Stelle die Angabe von Absolutzahlen innerhalb eines Symbols als die bestgeeignete Methode angesehen. Dies ermöglicht eine Darstellung der Information unabhängig von dem Flächenbedarf und der Musterung der Signaturen. Zudem entfällt die Problematik der Klassifizierung, welche sich bei Veränderungen im Dokumentenbestand automatisch an Anzahl und Spannweite anpassen müsste.

Für die Vereinfachung der Dokumentensuche sollte über die Integration quantitativer und qualitativer Informationen hinaus auch die Möglichkeit zur Ausblendung von Probestellensymbolen bestehen, die keinen Dokumentenbezug aufweisen. Dies ist für die Vermeidung leerer Suchanfragen von Bedeutung.

Mit Hilfe der genannten Funktionen sollte insgesamt eine nicht nur räumliche, sondern auch inhaltsbezogene Eingrenzung der Suche nach Dokumenten realisierbar sein, ohne die Komplexität der Karte zu groß werden zu lassen.

### **2.2.3 Möglichkeiten der Minimierung internetspezifischer Restriktionen**

Neben den bereits genannten interaktiven Funktionen liegt der darüber hinaus notwendige Grad an Interaktivität in den beschriebenen Unzulänglichkeiten und Restriktionen der Internetkartographie begründet. Ganz entscheidend ist, dass die Bedeutung der Restriktionen in Bezug auf die Auflösung durch die Implementierung von Zoom-Funktionen minimiert werden können. Nur dadurch ist es für die Benutzer möglich, die Karte in der Detailgenauigkeit zu betrachten, die sie für angemessen halten. In der Folge können kartographische Mindestgrößen etwas vernachlässigt werden. Die Bereitstellung einer Zoom-Funktion erfordert in der Konsequenz eine Funktion zur Verschiebung des Kartenausschnitts (Pan), da die Karte nicht mehr in vollem Umfang angezeigt

werden kann. Zusätzlich erwächst daraus die Forderung nach einer Übersichtskarte. Diese soll dazu dienen, den augenblicklich sichtbaren Kartenausschnitt in die Gesamtkarte einordnen zu können und damit eine ausreichende Orientierung zu gewährleisten.

Aus Darstellungsgründen war die Forderung erhoben worden, die Informationsdichte von Internetkarten möglichst gering zu halten. Mit weiteren interaktiven Funktionen kann auch dieser einschränkende Aspekt etwas abgeschwächt werden. Dies ist durch eine „*Zerlegung mehrschichtig-komplexer Kartengraphiken in eine Anzahl analytischer Kartenpräsentationen*“ möglich (ASCHE 2001, S. 13). Bei der Implementierung sollte deshalb eine Möglichkeit geschaffen werden, einzelne Layer ein- bzw. auszublenden. Dann kann das Kartenbild individuell so angepasst werden, dass für einen Benutzer irrelevante Informationen nicht angezeigt werden. Zudem kann der Zugriff auch auf sich gegenseitig überdeckende Symbole verschiedener Layer über das Ausblenden des im Vordergrund stehenden Layers gewährleistet werden. Dies ist für die uneingeschränkte Möglichkeit der Datenabfrage über die kartographischen Symbole von grundlegender Bedeutung.

Wünschenswert ist außerdem, an der Karte vorgenommene Änderungen zu speichern und in dieser Form wieder abrufen zu können. Eine Implementierung dieser Funktion würde die geforderte Einfachheit im Umgang mit der Karte erhöhen.

Kann dieses Maß an Interaktivität umgesetzt werden, sollte es auch zu gewährleisten sein, den Möglichkeiten des Mediums Internet gerecht zu werden und damit die genannten Probleme der Internetkartographie zu minimieren.

Wie aus dieser Diskussion hervorgegangen sein sollte, ist die Gestaltung einer Internetkarte keineswegs trivial. Es bestehen mitunter Konflikte zwischen kartographischer Qualität, Interaktivität und Benutzerfreundlichkeit bzw. Einfachheit einer Karte. Im nachfolgenden Kapitel ist mit der Wahl einer geeigneten Web-Mapping-Technologie ein möglichst zielführender Kompromiss zwischen diesen verschiedenen Anforderungen zu finden.

## 2.3 Web-Mapping-Technologien

Für die Zielsetzung dieser Arbeit sind weniger Verarbeitungsfunktionen als vielmehr Darstellungsfunktionen von Geoinformation von Interesse. Aus diesem Grund wird die weitere Diskussion über Web-Mapping auf Darstellungs- und Interaktionsmöglichkeiten begrenzt. Aus diesem Betrachtungswinkel werden im Folgenden Web-Mapping-Technologien vorgestellt und in Bezug auf ihre Eignung für das zu entwickelnde Informationssystem bewertet.

Für die Systematisierung unterschiedlicher Web-Mapping-Technologien können drei wesentliche Kategorien herausgestellt werden:

- offene, herstellerunabhängige Datenformate bzw. Architekturen im Gegensatz zu proprietären Datenformaten bzw. Architekturen
- die Unterscheidung zwischen unveränderlichen und interaktiven Karten
- die Unterscheidung zwischen Raster- und Vektordatenformaten

Nach KRAAK (2001) kann auch eine Unterscheidung zwischen statischen und dynamischen Karten erfolgen. Letztere sind als animierte Karten zu verstehen. Da diese erst einen Mehrwert bei der Visualisierung von Prozessen finden, wird dieser Bereich des Web-Mapping im Folgenden ausgeklammert. Es werden demzufolge nur statische Karten diskutiert.

Während die Aspekte unterschiedlicher Datenmodellierung in Form von Raster- oder Vektorformaten in der Literatur nahezu erschöpfend diskutiert wurden, sind offene, herstellerunabhängige Ansätze und interaktive Karten wesentlich jüngere Entwicklungen. Daher kann in diesen Bereichen noch nicht von einem abgeschlossenen wissenschaftlichen Diskurs gesprochen werden. Aus diesem Grund sollen diese Aspekte in der folgenden Diskussion ein besonderes Gewicht erhalten.

### 2.3.1 Einfache rasterbasierte Lösungen

Die Web-Kartographie ist bis heute geprägt von statischen, nicht oder nur wenig interaktiven Karten im Rasterformat. Dabei handelt es sich um Bildformate (JPEG, GIF oder PNG), die über das <image> Tag in ein HTML-Dokument

eingebettet werden. Eine Erstellung solcher Karten erfolgt häufig über das Einscannen von Papierkarten oder die Produktion von Karten mit Hilfe von Desktop-Mapping-Programmen. Auch GIS können für die Generierung von Rasterkarten verwendet werden, indem der im GIS gewählte Kartenausschnitt in ein Bildformat exportiert wird. Diese Form des Web-Mappings ist für den Entwickler daher zunächst mit einem verhältnismäßig geringen Aufwand verbunden.

Der größte Vorteil der Verwendung von Rasterformaten ist, dass sie ohne Zugangshindernisse wie die Installation von Plugins von allen Browsern dargestellt werden können. Ein weiterer Vorteil besteht darin, dass sie den Ausgabe- und Darstellungsmedien der heutigen PCs entsprechen. So erfolgt die Ausgabe auf Bildschirmen und Druckern vorwiegend in Raster- bzw. Pixelform. Damit sind keine weiteren Rechenoperationen für die Darstellung notwendig, was zu einer Entlastung der Systemressourcen führt. Außerdem gibt es recht gute Kompressionsalgorithmen, welche helfen, die Ausmaße einer Rasterdatei zu begrenzen (vgl. ZEH 2002).

Dem steht eine nicht unerhebliche Anzahl an Nachteilen gegenüber: So muss jegliche Information, die einer Karte zugrunde liegt, unmittelbar in die Karte eingetragen werden. Weitere Daten können nicht – wie bei Vektorkarten – in Form von Attributtabelle(n) bereitgehalten und bei Bedarf eingeblendet werden. Auch ist es nicht möglich, Teilinformationen einer Karte ein- bzw. auszublenden. Da im Internet mit einer sehr heterogenen Benutzergruppe mit unterschiedlichem Informationsbedarf zu rechnen ist, muss zumeist eine Vielzahl von Informationen präsentiert werden. Dies führt zu einer hohen Informationsdichte, wobei die einzelnen Informationen dann nur bei entsprechend hohen Auflösungen erkennbar sind. Hohe Auflösungen führen bei Rasterformaten allerdings sehr schnell zu großen Dateien mit langen Ladezeiten.

Des Weiteren muss der Maßstab einer Karte dauerhaft festgelegt werden. Eine spätere Vergrößerung (Zoom) der Karte ist zwar theoretisch möglich, ist aber in der Praxis nicht anzutreffen, da in diesem Fall einzelne Pixel erkennbar werden und die Qualität der Darstellung entsprechend nachlässt.

Im Allgemeinen müssen daher die Faktoren Detailgenauigkeit, Auflösung, Maßstab und Dateigröße gegeneinander abgewogen werden.

Die Bereitstellung interaktiver (über die Skalierung hinausführender) Funktionen gestaltet sich ebenfalls etwas problematisch. Eine interaktive Veränderung der Karte ist in einem Rasterbild nicht möglich. Da eine Rastergrafik nur Pixel und keine Objekte kennt, ist auch eine Abfrage über räumliche Objekte verhältnismäßig schwierig. Es ist in HTML zwar möglich, mittels so genannter Hotspots einzelne Bereiche einer Grafik unterschiedlich zu verlinken. Damit kann ein Objekt aber nur annäherungsweise richtig erfasst und verlinkt werden. Eine klare, überlappungsfreie Abgrenzung graphischer Objekte ist mit Hotspots nur mit einer sehr genauen Arbeitsweise durchführbar. Alternativ kann ein kartographisches Objekt in einer Rastergraphik auch über die Pixel-Position des Cursors identifiziert werden. Dies erfordert aber weitere Berechnungen für die Objektidentifikation, zudem entsteht ein nicht zu unterschätzender Implementierungsaufwand. Aufgrund unterschiedlicher Bildschirmauflösungen kann es aber leicht dazu kommen, dass über einen angeklickten Pixel nicht das gewünschte Objekt getroffen und damit die falsche Information abgerufen wird (vgl. LAKE 2001).

Die Nachführung von Rasterkarten ist – zumindest im Falle der Haltung der Daten in GIS – relativ umständlich. So muss eine Änderung zuerst im jeweiligen GIS durchgeführt und dann erneut ein Rasterbild exportiert werden, um abschließend wiederum alle Hotspots neu zu definieren bzw. die Pixelberechnungen zu aktualisieren.

### **Bewertung für die Entwicklung des Informationssystems**

Die Vorteile dieses Ansatzes für das zu entwickelnde System liegen darin, dass eine Rasterkarte einfach aus dem bestehenden ArcView-Projekt generiert und im Browser ohne weitere Zusatzprogramme für alle Internetnutzer gleichermaßen zugänglich gemacht werden kann. Dennoch ist dieser Ansatz für das zu implementierende Web-Informationssystem insgesamt als ungeeignet einzustufen. Dies ist neben Problemen der Aktualisierung und der Performance insbesondere auf den ungenügenden Grad der umsetzbaren Interaktivität

zurückzuführen. Es ist zwar durchaus möglich, Datenbankabfragen über Rasterkarten durch die Definition von Hotspots in Verbindung mit Skriptprogrammierung zu gewährleisten, das Fehlen von Zoomfunktionen sowie die Unveränderlichkeit von Karteninhalt und Kartenausschnitt ist für das angestrebte System aber nicht akzeptabel und wird den gestellten Anforderungen nicht gerecht.

### **2.3.2 Dynamisch generierte Rasterkarten**

Einen möglichen Ausweg aus den Beschränkungen einfacher Rasterkarten bieten so genannte Mapserver. Dies soll am Beispiel des ESRI ArcIMS in Verbindung mit dem HTML Viewer als clientseitiger Komponente diskutiert werden. Der HTML Viewer besteht aus mehreren HTML- und JavaScript-Dateien. Damit kann er von jedem gängigen Browser dargestellt werden. Der HTML Viewer hat in der Standardversion einen verhältnismäßig geringen Funktionsumfang. Er ist nur für die Betrachtung von Karten und die Abfrage der dazugehörigen Attributinformationen ausgelegt. Eine Anbindung an Datenbanken ist möglich.

Auch bei Verwendung dieser Architektur werden Karten als Rasterformate in HTML-Seiten eingebunden. ArcIMS gestattet jedoch eine interaktive Auseinandersetzung mit den Daten. Zoom-Operationen, die Wahl anderer Kartenausschnitte, interaktive Änderungen der Darstellung sowie das Ein- und Ausblenden von Layern sind nun möglich. Dies erfordert jedoch die serverseitige Generierung neuer Rasterdateien. So muss ArcIMS z.B. bei jeder Zoom-Operation für den ausgewählten Bereich der Karte eine neue Rasterdatei generieren und diese an den Client übermitteln (vgl. ESRI 2000). Dies kann sich sehr ungünstig auf die Performance des Systems auswirken.

### **Bewertung für die Entwicklung des Informationssystems**

Die wichtige Bedeutung von Pan- und Zoom-Funktionen (vgl. Kapitel 1.4) lässt in dieser Form einen nicht unerheblichen Kommunikationsbedarf zwischen Client und Server erwarten, was aufgrund langer Wartezeiten zu einer

Reduzierung der Benutzerfreundlichkeit und damit auch der Akzeptanz des Systems führen kann. Dies ist ein zentrales Argument gegen den Einsatz des HTML Viewers bei der Entwicklung des geforderten Systems. Außerdem können weiterführende Formen der Interaktivität wie das serverseitige Speichern von in der Karte vorgenommenen Änderungen nicht ohne weiteres umgesetzt werden. Sehr problematisch bei dieser Architektur ist auch, dass eine Abhängigkeit von ESRI Produkten besteht. Der Erwerb des ArcIMS würde erhebliche finanzielle Kosten verursachen, die im Rahmen dieses Projekts nicht getragen werden könnten. Dieses Problem wäre durch den Einsatz von OpenSource Mapservern wie dem Mapserver der University of Minnesota (URL 1) prinzipiell lösbar, aber in der Regel mit der Notwendigkeit von erheblichen Anpassungen verbunden.

Die weiteren problematischen Gesichtspunkte einer auf Rasterformaten basierenden Architektur bleiben allerdings auch bei diesem Ansatz bestehen. Daher ist für das zu implementierende System der Einsatz von Rasterformaten als insgesamt wenig geeignet einzustufen. Im Folgenden wird die Diskussion daher auf Lösungen konzentriert, die auf Basis von Vektorformaten implementiert werden können.

### **2.3.3 Java Applets**

Ein vielversprechender alternativer Ansatz besteht in der Visualisierung von Geoinformation und der Bereitstellung von Interaktionskomponenten in Form von Java Applets. In diesem Kapitel soll mit Bezugnahme auf das Java-basierte Informationssystem GIStern diskutiert werden, inwiefern Java Applets dazu geeignet sind, die geforderten Anforderungen zu erfüllen.

Java ist eine vollwertige, objektorientierte Programmiersprache. In Webseiten können Javaprogramme in Form so genannter Applets integriert werden. Applets sind Programme, die mit Hilfe eines Browsers ausgeführt werden können. Der Programmcode wird in einer HTML-Datei referenziert und vom Java-Interpreter des Browsers ausgeführt. Dank dieses Ansatzes ist Java unabhängig von Betriebssystem und Hardware einsetzbar. Aufgrund des sehr

umfangreichen Sprachumfangs ist zudem die Umsetzung interaktiver Funktionen verhältnismäßig unproblematisch.

Mit dem Abstract Windowing Toolkit (AWT) und mit Java 2D stellt Java Klassenbibliotheken zur Verfügung, welche die Ausgabe von anspruchsvollen Vektorgraphiken unterstützen. Beispielsweise ist mit der Funktion Antialiasing eine Komponente in Java 2D für die nicht-abgestufte Darstellung von Vektorgraphiken enthalten (vgl. Kapitel 2.2.1). Von den Funktionen des AWT werden für die Gestaltung von Karten graphische Primitivoperationen zum Zeichnen von Linien, zum Füllen von durch Linien eingeschlossenen Flächen sowie zur Ausgabe von Text benötigt.

GISterm wurde von der Firma disy als eine vollständig auf Java basierende Lösung entwickelt. Die Architektur basiert auf Standards wie OpenGIS, JDBC (Java Database Connectivity), RMI (Remote Method Invocation) und XML (Extensible Markup Language). Daher kann GISterm plattformunabhängig eingesetzt werden. Außerdem ist eine einfache Integration in bestehende Systeme und der Einbezug weiterer Komponenten möglich. Aufgrund der Architektur als Java Klassenbibliothek ist GISterm durch eigene Programmierung relativ einfach an gewünschte Anforderungen anpassbar (vgl. DISY 2003).

In der Standardversion ist GISterm aufgrund seines Funktionsumfangs durchaus als GIS zu bezeichnen. Es verfügt über zahlreiche Interaktionsmöglichkeiten, wie Pan, Zoom, das Ein- und Ausblenden von Layern, die Editierbarkeit der Legende, das Abspeichern von Karten und Legenden sowie über diverse Analysefunktionen. Darüber hinaus kann das Applet zahlreiche Datenformate einlesen, darunter ESRI Shapefiles, georeferenzierte Rasterdateien und Smallworld Dateien. Über JDBC ist eine Datenbankanbindung an Oracle, Informix, MS SQLServer und PostgreSQL möglich.

### **Bewertung für die Entwicklung des Informationssystems**

Wie das Beispiel GISterm zeigt, ist mit Java Applets der für das zu erarbeitende System geforderte Grad an Interaktivität bei guter Darstellungsqualität prinzipiell umsetzbar. Dieser Ansatz erübrigt darüber hinaus die Durchführung von

Konvertierungsschritten, wenn – wie im Falle von GIStern – Shapefiles direkt in das System importiert werden können.

Als clientseitige Systemvoraussetzung ist jedoch die Installation der Java Runtime Environment (JRE) notwendig. Neben dieser Installationshürde ist die Performance als ein weiterer kritischer Aspekt zu nennen. Bei dieser Architektur muss das gesamte Applet auf den Client-Rechner geladen werden. Dieser Vorgang kann bei GIStern mehrere Minuten in Anspruch nehmen. Wünschenswert wäre daher eine flexiblere, modulare Integration der benötigten Funktionen in das System. Denn in der bisherigen Form muss immer das komplette Applet – auch mit den nicht relevanten Funktionen – geladen werden. Gegen einen Einsatz von GIStern im Rahmen dieser Arbeit spricht zudem, dass Kosten für die Lizenzierung der Klassenbibliothek entstehen würden, wengleich diese deutlich unter den Kosten des ArcIMS liegen dürften.

Zur Erfüllung aller wesentlichen Anforderungen an das System müsste der Java-Code eines Systems wie GIStern in einzelne kleinere Komponenten zergliedert werden können und frei verfügbar sein. Da derartige Systeme nicht ausfindig gemacht werden konnten, wäre die einzige Alternative die Programmierung eines neuen Applets, um ein vollständig kontrollierbares bzw. adaptierbares und performantes Java-basiertes Informationssystem verwirklichen zu können. Dies ist im Rahmen dieser Master Thesis aufgrund des hohen Zeitaufwandes nicht vertretbar.

Mit der Entwicklung, welche Java in den letzten Jahren erfahren hat und der damit verbundenen Verbesserungen in puncto Robustheit und Performance (vgl. KRÜGER 2002) können Java Applets für den Bereich des Web-Mapping aber durchaus eine interessante Alternative darstellen.

### **2.3.4 Flash**

Bei Macromedia Flash handelt es sich um ein herstellerabhängiges binäres Vektorformat. Flash-Dateien können neben Vektorgraphiken auch Text, Videoformate, Rasterbilder und Audiodateien integrieren. Bekannt wurde das Format insbesondere durch die Möglichkeit, Graphiken zu animieren und diese über das Internet zugänglich zu machen.

Ein grundlegendes Kennzeichen von Flash-Dateien ist, dass sie binär und hoch komprimiert sind. Deshalb ist eine sehr gute graphische Qualität bei verhältnismäßig geringer Dateigröße erzielbar. Allerdings sind diese Dateien von Menschen weder lesbar noch editierbar. Dies ist aus Sicht des Copyrightschutzes zu begrüßen. Andererseits können die Dateien nicht ohne weiteres – wie textbasierte Dateien – interoperabel zwischen beliebigen Systemen ausgetauscht, gelesen und verändert werden. Die Generierung von Flash-Dateien ist damit auch nicht in einfachen Texteditoren möglich. Stattdessen muss auf eigens für Flash entwickelte Autorenprogramme zurückgegriffen werden.

Macromedia bezeichnet das Format Flash als offen (vgl. MACROMEDIA 2002), dies gilt jedoch nur mit Einschränkungen. Die Spezifikation der aktuellen Flash-Version ist zwar im Internet offen zugänglich (vgl. OPENSWF 2003), Macromedia besitzt aber weiterhin exklusive Rechte für die Weiterentwicklung. ZEH nennt Flash daher ein „*offenes proprietäres*“ Format (ZEH 2002, S. 5). In Folge der Offenlegung entstanden zahlreiche Programme für die Erstellung von Flash-Dateien (z.B. KoolMoves: URL 2, 3D Flash Animator: URL 3). Diese sind deutlich günstiger als der Editor von Macromedia, dafür werden Änderungen in den Nachfolgeversionen des Dateiformats immer zuerst in Macromedia-Programmen umgesetzt werden können. Aufgrund der rechtlichen Situation bezüglich der Weiterentwicklung von Flash besteht weiterhin eine Abhängigkeit von den durch Macromedia verfolgten Entwicklungslinien des Flash-Formats.

Für die Darstellung der Flash-Dateien im Browser wird ein Plugin benötigt. Dies ist aber insofern unproblematisch, da es mit ca. 200 KB sehr klein ist und bereits eine sehr hohe Marktpenetration aufweist. Laut einer NPD Online Studie vom September 2002 verfügen bereits 97,8% aller Internetnutzer über dieses Plugin (vgl. MACROMEDIA 2002), in neueren Browsern von Microsoft und Netscape ist es bereits standardmäßig enthalten. Damit kann mit Flash ein einfacher, kostenneutraler Zugang gewährleistet werden.

## **Generierung und Nachführung von Flash-Karten**

Die Erstellung von Flash-Karten aus GIS ist etwas umständlich, da sie nicht über Exportroutinen direkt erfolgen kann (vgl. NEUMANN 2002). Auf indirektem Wege können Vektordaten aber über das AutoCAD Format (dxf) in das Flash-Format konvertiert werden. Sehr ungünstig ist jedoch, dass bei der Konvertierung die benötigten Attributdaten (IDs der Probestellen) und die Legendeninformationen verloren gehen. Beides dürfte zu einem erheblichen Nachbearbeitungsbedarf führen. Dementsprechend ist auch die Nachführung der Flash-Karte als sehr zeitaufwendig einzustufen. Eine Alternative besteht in der Verwendung von Grafikprogrammen wie dem Adobe Illustrator. Mit dem GIS-Plugin MAPublisher können Shape-Dateien importiert und anschließend in das Flash-Format konvertiert werden (vgl. PITTI, JESSEE und RAMSAY 2002). Dies erscheint wesentlich einfacher, ist aber mit nicht unerheblichen finanziellen Kosten verbunden.

Die Verortung graphischer Objekte wird in Flash nicht mittels echter Koordinatenwerte vorgenommen, sondern ist lediglich in einer Pixeleinheit möglich, welche den Abstand zur linken oberen Ecke eines Flash-Dokuments angibt (vgl. NEUMANN 2002). Dies erschwert die Aktualisierung von Internetkarten erheblich, denn graphische Symbole können so nicht einfach in einem Editor kopiert und mit den realen Koordinatenwerten korrekt ausgerichtet werden. Stattdessen ist entweder die Umrechnung in die Pixeleinheit oder eine erneute vollständige Generierung der Karte notwendig.

## **Umsetzbarkeit interaktiver Funktionen**

Zur Verarbeitung von Benutzereingaben wurde von Macromedia die Skriptsprache ActionScript entwickelt. Dabei handelt es sich allerdings nicht um eine vollwertige Programmiersprache. Der Sprachumfang von ActionScript ist deutlich begrenzter verglichen mit Programmiersprachen wie Java oder C++. ActionScript lehnt sich sehr stark an JavaScript an, beide Sprachen sind jedoch nicht identisch. Dies ist als Nachteil zu werten, da die Eigenheiten dieser Sprache u.U. erst durchdrungen werden müssen.

Die für das zu entwickelnde Informationssystem notwendigen interaktiven Funktionen dürften mit Flash aber insgesamt relativ einfach umzusetzen sein. Es gibt im Internet bereits zahlreiche Beispiele für interaktive Flash-Karten, welche die Funktionen Zoom und Pan beherrschen (vgl. URL 4 und URL 5). Das Ein- und Ausblenden von Datenebenen dürfte ohne Probleme zu realisieren sein. Dies ist darauf zurückzuführen, dass graphische Objekte im Flash-Autorenprogramm von Macromedia in Ebenen organisiert werden können. Ein Beispiel für die Umsetzung dieser Funktion ist das touristische Informationssystem von Wiltshire (URL 6). Auch die Anbindung an eine Datenbank ist möglich. Die interaktive Veränderung der Symbole ist über ein Vor- bzw. Zurückbewegen auf der Zeitleiste realisierbar. In der Entwicklungs-umgebung von Macromedia ist die Zeitleiste eine zentrale Komponente. Dabei kann die Veränderung von graphischen Objekten im Zeitverlauf definiert werden. Mittels des Einsatzes von Sprungbefehlen kann direkt an die gewünschte Stelle – und damit zur gewünschten Visualisierungsform eines Symbols – gewechselt werden.

Problematischer dürfte dagegen die Speicherung von in einer Karte durchgeführten Änderungen sein. Eine unmittelbare Speicherung in der Flash-Datei ist nicht möglich. Es gibt zwar Möglichkeiten, Flash-Dateien automatisch zu generieren (z.B. mit dem Swift Generator, URL 7) und dabei benutzer-spezifische Anforderungen zu berücksichtigen. Die Grenzen des Flash-Einsatzes dürften aber darin liegen, dass vom Benutzer durchgeführte Operationen nicht – oder nur sehr aufwendig – so verfolgt werden können, dass die Speicherung dieser Informationen für die anschließende Generierung einer benutzerspezifischen Karte möglich ist.

## **Fazit**

Zusammenfassend kann festgehalten werden, dass mit Flash einem weiten Benutzerkreis qualitativ hochwertige interaktive Karten mit relativ geringen Dateigrößen bereitgestellt werden können. Dabei ist ein hoher Grad an Interaktivität umsetzbar, dessen Grenzen (das Speichern interaktiv veränderter Karten) auf die binäre Form des Datenformats zurückzuführen ist. Ein

entscheidender Nachteil für das zu entwickelnde System ist aber die umständliche, zeit- bzw. kostenintensive Konvertierung der Shapefiles in Flash-Karten. Dies hängt nicht zuletzt mit dem (semi-)proprietären Status von Flash zusammen. Zur Vermeidung der damit verbundenen Probleme erfolgt im Weiteren die Auseinandersetzung mit uneingeschränkt offenen Standards.

## **2.3.5 Offene, XML-basierte Standards**

### **2.3.5.1 Einführung in das Prinzip von XML**

In den vorangegangenen Kapiteln haben sich wiederholt Nachteile proprietärer Formate gezeigt. Diese äußern sich zunächst in relativ hohen finanziellen Kosten für die Anschaffung von Software wie z.B. Editoren oder Mapservern. Dazu kommt die mangelnde Kontrolle über diese Formate. Dies hat zur Folge, dass Anpassungen an individuelle Anforderungen oft nicht möglich sind und radikale Entwicklungsschritte oder gar die Abschaffung eines Formats zu erheblichen Problemen für die Weiterführung eines Systems führen können. Außerdem gestalten sich Erweiterungen derartiger Systeme sehr schwierig, wenn verschiedene proprietäre Formate in ein System integriert werden müssen, die dann – häufig in Verbindung mit Qualitätsverlusten – erst noch in ein einheitliches Format konvertiert werden müssen. Die Mängel proprietärer Formate bzw. Systeme sind gerade in einem integrativen und verteilten Medium wie dem Internet oft ein entscheidendes Entwicklungshemmnis. Um diesen Problemen zu begegnen, wurde vom World Wide Web Consortium (W3C) mit XML ein sehr vielversprechender Standard entwickelt. Dieser Standard und mögliche Anwendungen von XML bei der Entwicklung des Web-Informationssystems sollen im Folgenden ausführlich diskutiert werden.

XML hat in der Version 1.0 seit dem 10. Februar 1998 den Status einer „W3C Recommendation“ (vgl. BRAY 1998). Diese Empfehlung durch das W3C macht XML zu einem offiziellen Internet-Standard.

XML ist eine Metasprache, also eine Sprache zur Definition von Auszeichnungssprachen, die dem grammatischen System von XML entsprechen (vgl. SALATHÉ 2001). Man nennt nach den Regeln von XML definierte

Sprachen auch XML-Applikationen oder Instanzen von XML. Mit GML (Geography Markup Language) und SVG (Scalable Vector Graphics) sollen zwei der zahlreichen Auszeichnungssprachen, die auf XML basieren, in den folgenden Kapiteln diskutiert werden.

In einer Auszeichnungssprache sind verschiedene Befehle definiert, die der Strukturierung von Informationen dienen. Einige solcher Befehle sind aus HTML bekannt:

```
<k> Ich bin kursiv. </k>  
<h2> Ich bin eine Überschrift zweiter Ordnung. </h2>  
<blockquote> Ich bin ein Zitat. </blockquote>
```

Ein Auszeichnungsbefehl, auch Element genannt, besteht in der Regel aus zwei Teilen. Einem eröffnenden Tag `<Element>` und einem schließenden Tag `</Element>`. Dazwischen steht die Information, der Inhalt, der durch dieses Element näher charakterisiert wird. Ist ein Element nicht Träger eines Inhalts, wird es als leeres Element bezeichnet und mit `<Element/>` notiert. Elemente können zudem über die Zuweisung von Werten an Attribute näher charakterisiert werden:

```
<Person Alter="25" Größe="190"/>
```

Auf Basis einer Auszeichnungssprache können Dokumente definiert und verarbeitet werden (vgl. WIELAGE und POTT 2001). Damit ist bereits ein zentrales Konzept von XML angesprochen: der Dokumenttyp. Dieser bezeichnet eine Klasse von Dokumenten, die einer XML-Applikation zugeordnet werden können. Alle Dokumente eines Dokumenttyps gleichen sich in ihrem strukturellen Aufbau (vgl. MINTERT 1999). Entscheidend dafür sind die zur Verfügung stehenden Auszeichnungsbefehle, die Regeln für die hierarchische Anordnung derselben sowie die Definition von Attributen innerhalb von Elementen und deren zulässige Wertebereiche (Domains). Dies wird in so genannten DTDs (Document Type Definition) festgelegt. DTDs offizieller Standards werden durch das W3C vorgegeben. Daneben besteht auch die Möglichkeit, eigene DTDs zu definieren.

Anhand der angegebenen DTD wird überprüft, ob ein Dokument syntaktisch korrekt ist. Bei Einhaltung der Regeln von XML heißt ein Dokument wohlgeformt

(well-formed). Ein wohlgeformtes Dokument, das entsprechend der Regeln einer angegebenen DTD erstellt wurde ist gültig (valid). Die Überprüfung von XML-Regeln und DTD-Konformität gewährleistet eine syntaktisch korrekte Struktur der Dokumente.

Einige grundlegende Eigenschaften und Prinzipien von XML sollen nachfolgend ausgeführt werden:

- **XML ist textbasiert**

Im Gegensatz zu binären Datenformaten sind die in XML enthaltenen Informationen von Mensch und Maschine lesbar. Letzteres ist z.B. für Suchmaschinen und für die automatische Weiterverarbeitung von Daten von Bedeutung. Für Programmierer ist besonders die kostengünstige und unkomplizierte Erstellung von XML-Code in einfachen Texteditoren vorteilhaft.

- **XML ist ein offener, herstellerunabhängiger Standard**

Einige Nachteile proprietärer Datenformate wurden bereits angesprochen. Demgegenüber lassen sich fünf entscheidende Vorteile offener Standards herausstellen: geringe Kosten, weitreichende Unterstützung durch eine Vielzahl von Entwicklern, die Minimierung von Risiken durch die Umgehung der Abhängigkeit von speziellen Formaten, die Implementierung eines Standards ohne die Entstehung von Lizenzgebühren sowie maximale Interoperabilität. Gerade Interoperabilität ist im Internet aufgrund der verteilten Datenorganisation und der Integration unterschiedlicher Medien von hoher Bedeutung. Wird ein Standard von vielen Systemen unterstützt und ist er einfach und kostengünstig zu implementieren, dann kann ein einfacher Austausch und Transport zwischen diesen Systemen erfolgen. Damit sollte ein System insgesamt unproblematisch erstellt bzw. weitergeführt oder skaliert werden können. Dazu tragen auch die zahlreichen, frei verfügbaren Verarbeitungsprogramme bei, die für alle XML-Applikationen eingesetzt werden können.

Da alle XML-basierten Auszeichnungssprachen den Regeln von XML unterliegen, können Programmierer ihr Wissen von einer Auszeichnungssprache einfach auf andere Auszeichnungssprachen übertragen. Bei

proprietären Formaten ist dagegen häufig das Erlernen von speziellen Programmier- oder Skriptsprachen (z.B. ActionScript oder Avenue) oder die Verwendung spezieller Software notwendig.

- **XML ist erweiterbar**

Da XML die Definition von eigenen Auszeichnungssprachen gestattet, können in praktisch allen Anwendungsbereichen Sprachen implementiert werden, die den individuellen Anforderungen gerecht werden. Zudem ermöglicht XML die Erweiterung bestehender Auszeichnungssprachen um eigene Auszeichnungsbefehle. Darüber hinaus ist es möglich, verschiedene Auszeichnungsbefehle unterschiedlicher Dokumenttypen in einem Dokument zu integrieren. Durch das Voranstellen der Dokumenttypenbezeichnung vor einen Befehl (so genannte Namespaces) können alle Auszeichnungsbefehle eindeutig einem Dokumenttyp zugeordnet werden (Bsp.: `<svg:polyline>`, `<xsl:transform>`). Die Erweiterbarkeit trägt der Tatsache Rechnung, dass benötigte Dokumentstrukturen häufig nur geringfügig von bereits bestehenden Dokumenttypen abweichen und ermöglicht damit eine sehr effiziente Form der Dokumentengestaltung.

- **XML unterscheidet logische und physische Auszeichnung**

In Auszeichnungssprachen unterscheidet man zwischen zwei verschiedenen Auszeichnungstypen mit unterschiedlicher Zielrichtung. Eine Möglichkeit ist die logische oder semantische Auszeichnung, daneben können Informationen auch nach Darstellungskriterien (physisch) ausgezeichnet werden.

Logische Auszeichnungsbefehle beziehen sich auf die Bedeutung des Inhalts. So kann beispielsweise der Inhalt „15“ durch den Einschluss zwischen den Tags `<Hausnummer>` und `</Hausnummer>` die Bedeutung einer Hausnummer zugeteilt werden. Wie diese Information dargestellt werden soll ist dabei nicht definiert. Physische Auszeichnungsbefehle legen dagegen fest, wie die Zahl „15“ dargestellt werden soll: `<k>15</k>`. In diesem Fall wird die Zahl kursiv ausgegeben, die Bedeutung der Zahl ist aber allenfalls für Menschen aus dem Kontext zu erschließen, Maschinen können dies im Allgemeinen nicht leisten.

XML verlangt eine strikte Trennung von Semantik und Darstellung. Elemente in XML dienen typischerweise der logischen Auszeichnung. Für die Darstellung logisch ausgezeichneter Inhalte wurde XSL (Extensible Stylesheet Language) entwickelt. Damit können für alle Elemente Regeln definiert werden, wie ihre Darstellung erfolgen soll. Durch die Anwendung unterschiedlicher XSL-Dateien können dieselben Inhalte sehr unterschiedlich dargestellt werden. Dies ist besonders für die Ausgabe auf unterschiedlichen Medien interessant. So können Inhalte für einen Browser, für ein Mobiltelefon oder für ein Navigationssystem einheitlich gespeichert und unterschiedlich aufbereitet werden. Damit wird das Prinzip der Redundanzminimierung unterstützt.

- **XML ermöglicht komplexe Verlinkungen**

XML unterstützt mit der Technik XLink, anders als HTML, nicht nur einfache Verlinkungen der Relation 1:1, sondern auch der Relationen 1:n und n:m. Damit können auch komplexe Beziehungen, wie sie z.B. in Datenbanken auftreten, direkt in einer XML-Instanz gespeichert werden.

- **XML erlaubt einen objektorientierten Zugriff auf die Elemente eines Dokuments**

Ein Zugriff auf die Elemente von XML-Dokumenten erfolgt über das so genannte Document Object Model (DOM). Damit können u.a. SVG-, GML- oder HTML-Dokumente Variablen zugewiesen und als Objekte angesprochen werden. Über diese Objekte kann aus diversen Skriptsprachen auf die Elemente des jeweiligen Dokuments zugegriffen, deren Attribute abgefragt oder verändert werden.

## **Fazit**

Von den an das System gestellten Anforderungen können durch XML insbesondere ein einfacher, kostengünstiger Zugang und eine unkomplizierte Wartung und Aktualisierung unterstützt werden. Techniken wie XSL und DOM lassen einen hohen Grad an Interaktivität erwarten.

In den nächsten Kapiteln soll untersucht werden, inwiefern die XML-Instanzen GML und SVG den weiteren Anforderungen an das System gerecht werden können.

### **2.3.5.2 SVG (Scalable Vector Graphics)**

SVG hat in der Version 1.0 erst seit dem 4. September 2001 den Status einer „W3C Recommendation“. Damit ist SVG erst seit kurzer Zeit ein offizieller Standard. Die aktuelle Version (SVG 1.1) erreichte den Status der „W3C Recommendation“ am 14. Januar 2003 (vgl. FERRAILOLO 2001 und 2003). Da es noch eine geraume Zeit dauern kann, bis die Version 1.1 von entsprechenden SVG Viewern unterstützt wird, erfolgt die weitere Diskussion noch anhand der Version 1.0.

SVG ist eine Auszeichnungssprache für die Spezifikation und Darstellung (und darüber hinaus auch der Animation) von Vektorgraphiken. Die weiter unten vorgestellten graphischen Objekte haben in SVG also keine logische Bedeutung. Im Folgenden sollen einige SVG-Elemente im Hinblick auf ihre Einsatzmöglichkeiten für das zu entwickelnde Informationssystem näher betrachtet werden.

Das Element `<svg>` ist das Wurzelement, welches alle weiteren Elemente des Dokuments einschließt. Seine Attribute „width“ und „height“ geben die Höhe bzw. Breite der Zeichenfläche, des so genannten Viewports oder Canvas an, auf welchem die Vektorgraphiken gezeichnet werden können. Das Attribut „viewBox“ definiert ein Koordinatensystem, dem alle Subelemente von `<svg>` unterworfen sind. Die Domain für die viewBox besteht aus vier Werten: den x- und y-Koordinaten für die linke obere Ecke des Viewports sowie Werte für dessen Länge und Breite. Im Gegensatz zu Flash können die Distanzeinheiten frei gewählt werden, wobei keine explizite Angabe der Einheit möglich bzw. nötig ist. Deshalb können die Koordinaten realer Koordinatensysteme (z.B. Gauß-Krüger-Koordinaten) relativ einfach in SVG übernommen werden.

SVG unterstützt zudem die Transformation von geometrischen Figuren über das Attribut „transform“. Dies erlaubt die Verschiebung, Skalierung, Drehung oder Verzerrung von einzelnen oder gruppierten geometrischen Figuren

respektive des gesamten Karteninhalts. Auch über die Veränderung der Höhe und Breite der `viewBox` ist eine Skalierung des Karteninhalts möglich. Diese Sprachmerkmale stellen eine wichtige Grundlage für die Realisierung von Pan- und Zoom-Funktionen dar.

Die geometrischen Formen selbst können in SVG unter Verwendung der entsprechenden Auszeichnungsbefehle einfach umgesetzt werden. Dafür sind in der SVG-DTD Elemente für geometrische Standardformen definiert. Dazu gehören u.a. Pfade, Kurven, Rechtecke, Kreise und Polygone. Pfade sind Polylinien, die über die Angabe mehrerer Koordinatenpaare definiert werden. Diese Information wird einem Attribut des `<path>` Elements übergeben. Die Angabe der Koordinaten für einen Vertex kann dabei in absoluten oder in relativen Koordinaten zum Startpunkt der Polylinie erfolgen. Polylinien mit unregelmäßigen Verläufen können aufgrund der Vielzahl der benötigten Koordinatenpaare zu sehr großen Dateien führen. Abb. 2 zeigt mögliche Ausprägungen der Formen Kreis, Rechteck und Pfad in Verbindung mit ihrer SVG-Notation.



```
<circle id="kreis1" cx="40" cy="40" r="20" style="fill:green; stroke:blue; stroke-width:2"/>
```



```
<rect id="rechteck1" x="20" y="100" width="40" height="40" style="fill: yellow; stroke: black; stroke-width:3"/>
```



```
<path id="pfad1" d="M 20 180 L 30 220 60 240 40 240 90 190" style="fill: none; stroke: black; stroke-dasharray:5,2"/>
```

**Abb. 2: Geometrische Formen und deren Notation in SVG**

Quelle: eigener Entwurf

Eine nähere Spezifizierung von Position, Größe und Farbe geometrischer Formen ist über die Attribute der entsprechenden Elemente möglich. Allen derartigen Elementen sind die Attribute „id“ und „visibility“ zu eigen. Mittels der

ID kann jede Ausprägung eines Elements identifiziert, abgefragt oder verändert werden. Die Belegung des „visibility“-Attributs entscheidet darüber, ob ein graphisches Objekt überhaupt angezeigt werden soll.

Auf die Problematik, dass geometrische Formen – insbesondere bei automatischer Generierung oder Anpassung – sich gegenseitig überdecken können, war bereits in Kapitel 2.2 hingewiesen worden. In SVG gilt dafür folgende Regel: Je weiter unten ein Objekt innerhalb eines Dokuments definiert ist, desto weiter vorne wird es am Bildschirm ausgegeben. Flächenhafte Objekte sollten daher grundsätzlich vor punkt- oder linienförmigen Objekten definiert werden, um diese nicht zu überdecken.

Neben geometrischen Formen können mit SVG auch Textlabels über das `<text>` Element in Karten integriert werden. Auch hier besteht die Möglichkeit, über Attribute die Darstellung des Textes (z.B. Schriftart, Schriftgröße) zu beeinflussen. Darüber hinaus können mit Hilfe des `<image>` Elements SVG-Karten mit Rasterbildern ergänzt werden.

Ein sehr hilfreiches Feature von SVG ist das Gruppierungselement `<g>`. Die zwischen dem eröffnenden und schließenden `<g>` Tag eingeschlossenen Subelemente werden zu einer logischen Einheit zusammengefasst und sind als Ganzes adressierbar. Im Zusammenhang mit Karten ist dies besonders für die Definition eines Layers über die Zusammenfassung gleichartiger Symbole von Interesse. Beispielsweise ist es möglich, die Sichtbarkeit aller zu einer Gruppe gehörenden Symbole über das Attribut „visibility“ des `<g>` Elements zu verändern.

Ein ähnlicher Ansatz liegt dem Element `<symbol>` zugrunde. Auch damit können mehrere graphische Objekte zu einer logischen Einheit zusammengefasst werden. Der Unterschied besteht darin, dass die dort angegebenen Graphiken nicht gezeichnet werden. Dafür können sie über ihre ID referenziert und damit Instanzen des Symbols mehrfach verwendet werden. Dies ist mit Hilfe des Elements `<use>` möglich. Über die Belegung des Attributs „xlink:href“ wird eine Instanz des Symbols erzeugt und am Bildschirm ausgegeben. Sehr vorteilhaft ist die Verwendung des `<use>` Elements beim Entwurf einer Karte. Gleichartige räumliche Objekte werden durch dasselbe Symbol repräsentiert.

Dieses muss nur einmal definiert werden und wird dann den entsprechenden Objekten zugewiesen. Über die nachträgliche Zuweisung anderer Symbole ist eine hohe Flexibilität für die Darstellung der räumlich verorteten Objekte gegeben. Dies ist im Rahmen dieser Arbeit in besonderem Maße für die flexible Visualisierung von Probestellen von Interesse.

### **Veröffentlichung und Performance von SVG-Karten**

Ein Vorteil von SVG für das zu entwickelnde System besteht zweifelsohne in der vektoriellen Modellierung der Daten. So ist die Betrachtung der SVG-Karten auch in großen Maßstäben bei optisch guter Qualität, verhältnismäßig geringen Datenmengen und der daraus resultierenden relativ guten Performance möglich. In diesem Zusammenhang ist auch zu erwähnen, dass SVG Antialiasing unterstützt und solcherart den auflösungsbedingten gezackten Verlauf von Linien unterbinden kann (vgl. Kapitel 2.2).

Mit den diskutierten SVG-Elementen können Vektorkarten, nach Layern gegliedert, im Internet veröffentlicht werden. Dies ist gegenwärtig jedoch in keinem Browser ohne zusätzliche Installationen möglich. Für die Visualisierung eines SVG-Dokuments ist ein Plugin notwendig. Dieses Plugin, der SVG Viewer 3.0 der Firma Adobe, ist auf der Adobe Homepage (URL 8) kostenlos downloadbar. Die Dateigröße beträgt ungefähr 2,25 MB, eine Installation ist unter Windows 98, Windows XP, RedHat Linux 7.1 und diversen Macintosh-Betriebssystemen möglich (vgl. ADOBE 2003). Die Marktpenetration liegt allerdings erst bei ca. 10 %, wenn auch mit steigender Tendenz (vgl. NEUMANN 2002). Deshalb ist kritisch anzumerken, dass Download und Installation eine Hürde für den Zugriff auf SVG-basierte Karten darstellen können. Microsoft stellt auch noch keine baldige Integration von SVG in den Internet Explorer in Aussicht. Dagegen wird im Mozilla SVG Project gegenwärtig an einer Lösung für den auf Netscape basierenden Open-Source-Browser Mozilla gearbeitet (vgl. MOZILLA.ORG 2002). Folglich ist der Zugang zu SVG-basierten Internetkarten gegenwärtig zwar mit keinen finanziellen Kosten verbunden und plattformunabhängig realisierbar, eine Einstiegshürde bleibt aber vorerst noch bestehen. Die Strategie von Adobe, das Plugin bei Installation des weit

verbreiteten Acrobat Reader ab der Version 5.0 automatisch mitzuinstallieren, lässt aber auf eine schneller steigende Verbreitung hoffen.

Sehr vorteilhaft ist, dass das Plugin Zoom- und Pan-Funktionen bereitstellt. Damit sind im Gegensatz zu Lösungen mit Rasterformaten Maßstabsänderungen und eine Verschiebung des Kartenausschnitts möglich, ohne serverseitig neue Daten generieren und an den Client übertragen zu müssen. Vielmehr können diese Funktionen alle clientseitig ausgeführt werden. Damit ist allerdings auch ein Nachteil verbunden: SVG-Dateien werden immer als Ganzes geladen. Außerhalb des Sichtfensters liegende Kartenausschnitte und als unsichtbar gekennzeichnete Informationen müssen daher auch sofort geladen werden. In diesem Zusammenhang ist auch ein Nachteil gegenüber dem vektorialen Flash-Format zu nennen. Die binäre Datenstruktur von Flash erlaubt eine höhere Komprimierung als die textbasierten SVG-Dateien. Daraus ist zu schließen, dass die Performance von SVG zumindest gegenüber Flash etwas zurückfällt.

Aber auch für SVG bestehen Komprimierungsmöglichkeiten. Mit dem Programm win-gz (URL 9) kann die Größe von SVG-Dateien um bis zu 70% verringert werden (vgl. GROLIG, SCHENK, WALDIK 2001). Eine Datei der Größe von 1 MB lässt sich also auf ungefähr 300 KB reduzieren. Die Dateierweiterung wird dabei von .svg in .svgz umgeändert. Die Dekomprimierung erfolgt automatisch im Internet-Browser. Damit kann die Performance des Systems entscheidend verbessert werden. Da es sich um eine verlustfreie Komprimierung handelt, entstehen keine optischen Qualitätsverluste.

Der bislang beschriebene Grad an Interaktivität ist zwar ein erheblicher Fortschritt gegenüber auf Rasterformaten basierenden Systemen, den genannten Anforderungen, insbesondere der Datenbankanbindung und den Anpassungsmöglichkeiten der Internetkarte ist aber mit den bislang diskutierten Sprachmerkmalen von SVG allein nicht gerecht zu werden. Im Folgenden soll dargestellt werden, wie diese Möglichkeiten durch Skriptprogrammierung erweitert werden können. Dies wird getrennt nach clientseitigen und serverseitigen Skripten vorgenommen.

## **Clientseitig implementierbare Interaktivität**

Für das zu entwickelnde Informationssystem wird es als sinnvoll erachtet, wenn möglichst viele Änderungen an einem SVG-Dokument ausschließlich clientseitig durchgeführt werden können. Dies bedeutet, dass keine Daten zwischen Client und Server hin- und hergeschickt werden müssen, was zu einem deutlich besseren Antwortzeitverhalten führt.

Für die clientseitige Kommunikation wird derzeit überwiegend JavaScript eingesetzt. JavaScript ist eine der am weitesten verbreiteten clientseitigen Skriptsprachen und wird von den meisten Internet-Browsern unterstützt.

Mit dem Einsatz von JavaScript ist es möglich, über das SVG-DOM Attribute von SVG-Elementen abzufragen und zu ändern. Anschauliche Beispiele für den in Internetkarten damit realisierbaren Grad an Interaktivität erbrachten bereits ANDREAS NEUMANN und ANDRÉAS M. WINTER am Institut für Kartographie der ETH Zürich. Schon Anfang 2000, kurz nach dem Erscheinen des ersten SVG-Plugins, wurde von NEUMANN die vermutlich erste interaktive Internetkarte mit den clientseitigen Techniken SVG, HTML und JavaScript erstellt (vgl. NEUMANN 2000). Dabei handelt es sich um ein Web-Informationssystem für die Visualisierung und Abfrage der sozioökonomischen Strukturen der Stadt Wien. Dort wurde bereits das Ein- und Ausblenden von Layern umgesetzt. Eine Übersichtskarte mit Pan-Funktion realisierte eine weitere Funktion, die auch für das im Rahmen dieser Arbeit zu entwickelnde System gefordert wurde. Überdies können administrative Flächen abhängig von einer durch den Benutzer ausgewählten Variablen (z.B. Alter oder Bildung der Bevölkerung) und deren interaktiv bestimmten Klassifizierung dargestellt werden. Zwar sollen in dem zu erstellenden Informationssystem Punktsignaturen und nicht Flächensignaturen Gegenstand interaktiver Veränderungen sein, das von NEUMANN implementierte Prinzip kann aber aller Voraussicht nach in ähnlicher Weise realisiert werden.

In einer Arbeit von WINTER (2000) wurde eine weitere interessante Funktion umgesetzt. WINTER erstellte einen interaktiven Europa-Atlas. Dieser bietet neben dem Ein- und Ausblenden von Ebenen und der Auswahl verschiedener Themen weitere Interaktionsmöglichkeiten. So können Diagramme dynamisch

auf Anforderung aus statistischen Daten generiert werden. Gespeichert werden diese Daten in JavaScript-Listen. Der Benutzer kann Diagrammparameter einstellen, auf Farben, Diagrammgröße und Diagrammtyp Einfluss nehmen. Seine Arbeit zeigt also u.a. auf, wie die attributabhängige Darstellung von kartographischen Punktsymbolen mit SVG in interaktiven Karten umgesetzt werden kann. Auch bei der Entwicklung des geplanten Web-Informationssystems ist die Visualisierung von Punktsignaturen (den Probestellensymbolen) in Abhängigkeit von Attributdaten von Interesse.

Die bislang recht allgemein thematisierten Möglichkeiten clientseitiger Interaktivität werden im Folgenden anhand von JavaScript Code-Fragmenten in Bezug auf die benötigten Funktionen genauer angesprochen.

Der Zugriff auf ein in HTML eingebettetes SVG-Dokument erfolgt gemäß dem SVG-DOM mit folgender Syntax:

```
var doc = document.['name'].getSVGDocument();
```

Eine Einbettung in HTML ist über das `<embed>` Element möglich. Darin wird das SVG-Dokument über das Attribut „name“ eindeutig identifiziert.

Für den Zugriff auf die Elemente stehen zwei Methoden zur Verfügung:

```
getElementById();  
getElementsByTagName();
```

Diese sind an das Objekt zu richten, welches das SVG-Dokument repräsentiert. Der Befehl `doc.getElementById('myRect')` gibt beispielsweise ein Rechteck zurück, das mit der ID „myRect“ innerhalb des Dokuments „doc“ spezifiziert wurde. Die Methode `doc.getElementsByTagName('rect')` gibt dagegen alle Elemente des Typs „rect“, also alle Rechtecke des Dokuments zurück. Die Abfrage eines Attributes des Rechtecks könnte wie folgt aussehen:

```
var breite = rect.getAttribute('width');
```

Diese Funktion ermittelt die Breite des Rechtecks und weist sie der Variablen `breite` zu. Mit dem Befehl `setAttribute('attribute', 'value')` können außerdem alle Attribute der Elemente eines SVG-Dokuments verändert werden, was beträchtliche Interaktionsmöglichkeiten für den Benutzer des Systems bietet. Dies gilt insbesondere bezüglich des bereits angesprochenen Elements `<use>`.

Denn damit können jedem als `<use>` Element definierten Punktobjekt der Karte vom Benutzer interaktiv verschiedene Symbole zugewiesen werden.

Damit der Benutzer solche Veränderungen der Attributwerte durchführen kann, müssen JavaScript-Funktionen direkt mit Benutzeraktionen verknüpft werden. Es kann für jedes graphische Objekt spezifiziert werden, bei welcher Benutzeraktion welche Funktion aufgerufen werden soll. Da das Programm auf die Benutzeraktion wartet, im Hintergrund also permanent überprüft, ob eine solche Aktion ausgeführt wird, spricht man auch von einem Event-Listener oder Ereignisempfänger. Ein Beispiel dafür lautet in SVG-Notation:

```
<circle onmousedown="doSomething(evt)" />
```

Diese Angabe registriert einen Event-Listener des Mausclick-Typs „mousedown“ für die angegebene Kreisform. Klickt der Benutzer mit der linken Maustaste auf dieses graphische Objekt, wird die oben genannte Funktion (*doSomething*) aufgerufen. Der Funktion kann optional ein Event-Objekt (*evt*) als Parameter übergeben werden. Dieses Objekt wird bei einer Benutzeraktion automatisch generiert und erlaubt nützliche Abfragen, wie z.B. nach den Koordinaten des Mausclicks oder dem graphischen Objekt, von welchem aus die Funktion aufgerufen wurde.

Auch die dynamische Registrierung und Entfernung von Event-Listnern ist mit Hilfe von JavaScript möglich. Diese erlaubt es z.B., dieselbe Benutzeraktion in mit verschiedenen Funktionen zu verknüpfen bzw. die Anzahl der aufgrund einer Benutzeraktion auszuführenden Funktionen zu variieren.

Damit besteht insgesamt ein erhebliches Potenzial für clientseitig implementierte Interaktivität auf Basis von SVG-Karten.

### **Serverseitig implementierbare Interaktivität**

Neben den besprochenen interaktiven Funktionen gibt es weitere wünschenswerte Funktionen, die nur serverseitig ausgeführt werden können. Dies betrifft für das geplante System alle Funktionen, für welche die Generierung von HTML- oder SVG-Dokumenten zur Abfragezeit notwendig ist sowie diejenigen Funktionen, die mit dauerhaft zu speichernden Daten zusammenhängen.

Insbesondere ist der Zugriff auf die Metadatenbank nur mit serverseitigen Komponenten umsetzbar. Der Auslöser eines solchen Zugriffs kann über an SVG-Elemente gekoppelte Event-Listener erfolgen, die dann ein serverseitiges Skript – ein so genanntes CGI-Skript (Common Gateway Interface) – aufrufen.

Das Abspeichern von Benutzereinstellungen, also von Veränderungen des Karteninhalts, muss serverseitig erfolgen, um beim erneuten Aufruf des Systems das betreffende Kartenbild wieder bereitstellen zu können. Das Speichern eines SVG-Dokuments ist über das Kontextmenü des SVG Viewers möglich. Dabei werden interaktiv durchgeführte Änderungen jedoch nicht berücksichtigt. Demzufolge muss entweder mit Hilfe von serverseitigen Skripten eine neue SVG-Datei generiert werden oder es müssen die durchgeführten Änderungen separat von der Karte gespeichert und nach dem Laden der Karte auf diese wieder automatisch angewendet werden. Beide Möglichkeiten lassen bezüglich Performance und Implementierungsaufwand eine gleichwertige Umsetzung dieser Funktion erwarten.

Aus den genannten clientseitigen und serverseitigen Interaktionsmöglichkeiten kann gefolgert werden, dass alle geforderten interaktiven Funktionen durch die Unterstützung von SVG mit Skriptsprachen prinzipiell umsetzbar erscheinen.

### **Generierung und Nachführung von SVG-Karten**

Abschließend soll auf die Eignung von SVG bei der Erstellung und Änderung von Karten eingegangen werden. Wie bereits bei der Diskussion über die Erstellung von Flash-Dateien zu sehen war (vgl. Kapitel 2.3.4), kann dies ein sehr kritischer und zeitraubender Vorgang sein. Aufgrund des textbasierten und offenen Ansatzes von SVG ist dabei grundsätzlich mit weniger Problemen zu rechnen. So existieren für die Transformation von Shapefiles nach SVG bereits diverse Konvertierungsprogramme. Für das zu entwickelnde System kommt für die Konvertierung der bestehenden Vektordaten die Testversion des SVG Mapper 1.3.9 (Download unter URL 10) in Frage. Dabei handelt es sich um eine ArcView Extension, die eine solche Konvertierung aus ArcView heraus unterstützt. Die Verwendung und Editierung des generierten SVG-Codes wird auch den Benutzern der Testversion gestattet, sofern ein Verweis auf das

Konvertierungsprogramm enthalten ist. Der Einsatz alternativer Programme ist aufgrund finanzieller oder lizenzrechtlicher Umstände im Rahmen dieser Arbeit nicht in Betracht zu ziehen.

Die Editierung von SVG-Dateien ist mit jedem einfachen Texteditor möglich und damit kostengünstig durchzuführen. Änderungen der Anzahl der Probestellensymbole sollten daher ohne großen Aufwand möglich sein. Für eine Aktualisierung im SVG-Code können die Punktsymbole eines Layers einfach kopiert und mit den entsprechenden Koordinatenwerten versehen werden. Es ist jedoch anzumerken, dass dieses Vorgehen nicht ohne weiteres auf die Nachführung von Flächen- oder Liniensignaturen übertragen werden kann.

### **Fazit**

SVG kann insgesamt als sehr gut geeignet für die Entwicklung dieses Informationssystems bewertet werden. Neben der hohen graphischen Qualität der Vektorgraphiken, den umsetzbaren Funktionalitäten und den generellen Vorteilen offener xml-basierter Formate sollte auch die Möglichkeit einer einfachen Wartung sowie – zumindest im Vergleich zu Rasteransätzen – die Realisierbarkeit einer akzeptablen Performance für eine Umsetzung des Systems mit SVG sprechen. Abstriche sind dagegen insbesondere beim Zugang auf ein mit SVG implementiertes System zu machen. Die Notwendigkeit der Installation eines Plugins ist zweifelsohne als negativer Faktor zu werten. Da zukünftig mit einer Verbesserung dieser Situation zu rechnen ist, sollte es sich dabei jedoch nicht um einen entscheidenden Gesichtspunkt handeln.

### **2.3.5.3 GML (Geography Markup Language)**

GML ist eine weitere auf XML basierende Auszeichnungssprache. Die aktuelle Version ist GML 2.1. Sie wurde vom OpenGIS Consortium (OGC) entwickelt und ist anders als SVG ein Standard, der die logische Auszeichnung geographischer Inhalte zum Ziel hat. Dabei geht es ausschließlich um die Speicherung und den Transport geographischer Daten (vgl. Cox 2002). Von ACKLAND und COX (2001) wird GML wie folgt definiert:

*„Geography Markup Language - GML - is a standard for representing geospatial objects or features, which uses real-world co-ordinates and associates real-valued properties with them.“*

Die Visualisierung spielt also in keinem der in GML vorhandenen Elemente und den dazugehörigen Attributen eine Rolle, dafür haben die Elemente in GML eine Bedeutung, die durch die Spezifikation entsprechender Attribute konkretisiert werden kann. GML entspricht dem XML-Prinzip der Trennung von Inhalt und Visualisierung damit konsequenter als SVG. Dafür kann GML von keinem WWW-Browser dargestellt werden. Für diesen Zweck ist die Transformation von GML-Dokumenten in andere Auszeichnungssprachen wie SVG oder VML (Vector Markup Language) notwendig. Da VML den Status einer Recommendation beim W3C bislang nicht erreichen konnte, ist SVG für diesen Zweck eindeutig vorzuziehen.

Aufgrund der Trennung von Darstellung und Inhalt ist der hier zu diskutierende Ansatz nicht als Ersatz zu SVG zu betrachten, sondern als andere Herangehensweise, wobei die Visualisierung wiederum in SVG erfolgt. Damit sind die im Kontext von XML und SVG bereits angesprochenen Vorteile und die Interaktionsmöglichkeiten mittels Skriptsprachen auch für diesen Ansatz gültig. Im Folgenden soll genauer auf die Spezifikation von GML eingegangen werden. Auf dieser Grundlage sollen die Vor- und Nachteile einer Speicherung der Informationen in GML mit Bezugnahme auf ein Implementierungsbeispiel diskutiert werden.

Das Grundelement für die Repräsentation geographischer Inhalte in GML ist das <Feature> Element. GML basiert auf der OGC Abstract Specification (vgl. OGC 2000). Darin wird ein Feature definiert als *„abstraction of a real world phenomenon; it is a geographic feature if it is associated with a location relative to the Earth“* (OGC 2000, Seite xiv). Features werden durch die Angabe von Sach- und Geometriedaten näher beschrieben. Es können praktisch alle Objekte, die auf der Erde verortet werden können, durch ein Feature repräsentiert werden. Damit ist es möglich, so unterschiedliche Objekte wie Pipelines, Fahrtrouten, Häuser oder Bäume mit GML zu modellieren und so die gesamte Erdoberfläche als eine Menge von Features abzubilden.

Der Zugang zur Modellierung dieser bereits sehr komplexen Objekte besteht in der Verwendung einfacher Bausteine. Diese werden vom OGC Simple Features genannt und bezeichnen „*features whose geometric properties are restricted to ‘simple’ geometries for which coordinates are defined in two dimensions and the delineation of a curve is subject to linear interpolation*“ (Cox 2002, Kapitel 1.1). Damit sind Punkte, Linien und Polygone die Grundbausteine für die geometrische Spezifikation der Elemente in GML. Geometrische Formen, die Kurvenverläufe beschreiben, werden aus den angegebenen Koordinaten interpoliert. Simple Features können zu beliebig komplexen Objekten – den Features – zusammengesetzt werden. Features können wiederum zu Feature Collections kombiniert werden. Dabei bleiben alle Eigenschaften der beteiligten Features erhalten, zusätzlich können weitere Eigenschaften angegeben werden, die nur für das zusammengesetzte Objekt gültig sind.

So vielfältig die Objekte der Erdoberfläche sind, so vielfältig sind auch die Interessen und Ziele einer Modellierung der Erdoberfläche. Demzufolge bestehen in verschiedenen Anwendungsbereichen sehr verschiedene Sichten auf die Welt, welche die Modellierung in einer Vielzahl von Features erfordern. In diesem Zusammenhang ist die Erweiterbarkeit XML-basierter Sprachen von hohem Interesse. Während für SVG Erweiterungen aufgrund der Abhängigkeit von den Darstellungsmöglichkeiten der Plugins bislang keine wesentliche Rolle spielen, sind sie für GML grundlegend. GML 2.1 berücksichtigt die XML Schema Recommendation des W3C (XMLSchema1 und XMLSchema2) und die XML Namespaces Recommendation (XMLName). Mit XML Schema können eigene Features definiert werden, mit XMLName können sie – insbesondere in Kombination mit in anderen Projekten definierten Features – über so genannte Namespaces eindeutig identifiziert werden. In einem XML Schema werden, analog zu Definitionen in DTDs, die Eigenschaften eines Features festgelegt. Dabei sind in GML Eigenschaften nach räumlichen und nicht-räumlichen Eigenschaften zu unterscheiden. Diese können als Attribut oder durch die Assoziation mit anderen Elementen ein Feature näher spezifizieren. Bemerkenswert ist, dass trotz dieser Vielfalt implementierbarer Feature-Typen die Interoperabilität gewährleistet werden kann. Dies ist deshalb möglich, weil die Definition von Features in eigenen Schemas (Application Schema) in GML

an drei Basis-Schemas (Feature Schema, Geometry Schema und XLink Schema) gebunden ist. Auf dem Prinzip der Vererbung können aus den genannten Basis-Schemas Elemente für die Definition eigener Features verwendet und kombiniert werden und erben dabei deren Eigenschaften. Mit dem Geometry Schema werden z.B. auf Basis der Simple Features grundlegende Auszeichnungsbefehle für die Definition der räumlichen Eigenschaften eines Features bereitgestellt, welche als Basis für die Konzeption komplexerer eigener Features zu verwenden sind (vgl. Cox 2002).

Die Auszeichnung von lokalisierbaren Objekten in GML soll mit folgendem Beispiel verdeutlicht werden. Es ist aus der GML Specification entnommen (Cox 2002, Kapitel 4.4). Darin wird die Person „Dean“ in GML-konformer Syntax definiert. Neben Sachattributen (Nachname, Spitznamen, Alter) ist der Ort dieser Person angegeben:

```
<Dean>
<familyName>Smith</familyName>
<age>42</age>
<nickName>Smithy</nickName>
<nickName>Bonehead</nickName>
<gml:location>
<gml:Point>
<gml:coord><gml:X>1.0</gml:X><gml:Y>1.0</gml:Y></gml:coord>
</gml:Point>
</gml:location>
</Dean>
```

Mit dem XLink Schema war bereits ein weiteres grundlegendes Konzept von GML genannt worden. Damit können für Features komplexe Links spezifiziert werden (vgl. Kapitel 2.3.5.1). Dies hat zur Konsequenz, dass Features – völlig unabhängig von dem Ort ihrer physischen Speicherung – über das Internet zueinander in komplexe Beziehungen gesetzt werden können und eine riesige relationale Datenbank bilden können (vgl. JOHANSSON und SIIRILÄ 2001). Dies ermöglicht zugleich die einfache Integration verteilter räumlicher und nicht-räumlicher Daten (vgl. LAKE 2001).

Als weiterem Aspekt soll an dieser Stelle auf die Implikationen der Trennung von Bedeutung und Darstellung eingegangen werden. Die konsequente Reduzierung auf Inhalte bringt für GML die bereits für XML allgemein diskutierten Vorteile der Unabhängigkeit (von Ausgabemedien) und der Flexibilität ein (vgl. Kapitel 2.3.5.1). Denn der Einsatz verschiedener XSL-Dateien erlaubt die Darstellung derselben geographischen Daten auf unterschiedliche Art und Weise. Beispielsweise können die Informationen eines GML-Dokuments neben einer SVG-Karte auch in tabellarischer Form in HTML oder als gesprochener Text (z.B. auf dem Mobiltelefon) ausgegeben werden. Dies erhöht die Flexibilität im Umgang mit den Daten beträchtlich, erweitert die potenziellen Anwendungsfelder und ermöglicht es, die Darstellung der geographischen Informationen noch mehr auf die Wünsche des Benutzers auszurichten.

### **Bewertung von GML für die Entwicklung des Informationssystems**

Auf Basis der Spezifikation von GML und den dabei diskutierten Eigenschaften soll nun geklärt werden, ob der Einsatz von GML für die Speicherung der Geobjekte für das zu entwickelnde System als geeigneter einzustufen ist als die ausschließliche Verwendung von SVG.

Dafür sollen zunächst die aus dem Design von GML resultierenden Vorteile kurz zusammengefasst werden:

- GML ermöglicht als interoperabler Standard die Integration von Sach- und Geometriedaten aus verschiedenen Datenquellen. Damit hat GML eine große Bedeutung als Schnittstelle zwischen verschiedenen proprietären Datenformaten. Erst über diese Schnittstelle ist eine einheitliche Darstellung verteilter Daten in SVG möglich. Erfolgt die Konvertierung aus verschiedenen Systemen direkt in SVG, so ist damit zu rechnen, dass die Darstellungen nicht miteinander kompatibel sind und nur mit erheblichem Zusatzaufwand integriert werden können.
- Die Elemente in GML haben eine konkrete Bedeutung und können deshalb von Maschinen ausgewertet und weiterverarbeitet werden. Somit ist die

Umsetzung von Dienstleistungsketten einfacher und umfangreicher möglich als mit SVG.

- GML kann die zu transformierenden Datenmengen reduzieren. Wird dem Benutzer eines Informationssystems die Möglichkeit gegeben, die für ihn interessanten Datenebenen auszuwählen, so kann der Inhalt der zu generierenden SVG-Datei auf die gewünschten Objekte reduziert werden. Sie enthält dann nur die angeforderten Objekte. In SVG muss dagegen die komplette Information an den Client übertragen werden. Dies gilt auch dann, wenn einzelne Layer nicht angezeigt werden sollen.
- GML entspricht im Gegensatz zu SVG der von XML geforderten Trennung von Inhalt und Darstellung in aller Konsequenz. Dadurch besteht eine höhere Flexibilität in der Darstellung der Inhalte (Design und Form der Ausgabe, Medium der Ausgabe).

Diesen Vorteilen ist gegenüberzustellen, dass für die automatische Generierung von SVG aus GML ein nicht unerheblicher Aufwand zu leisten ist. Das System wird in dieser Form deutlich komplexer. Das Ausmaß der notwendigen Implementierung soll anhand einer Fallstudie von JOHANSSON und SIIRILÄ näher bestimmt werden (vgl. JOHANSSON und SIIRILÄ 2001).

JOHANSSON und SIIRILÄ haben an der University of Gävle einen „Internet Vector Map Server“ entwickelt. Die Zielsetzung war dabei, ausschließlich offene, herstellerunabhängige Standards zu verwenden. Der von ihnen entwickelte Mapserver wurde so konzipiert, dass auf eine Anfrage im Browser die Daten automatisch aus dem vorliegenden MapInfo Interchange Format (MIF) in GML und davon wahlweise in HTML oder SVG konvertiert werden können. Ihre Studie zeigt einerseits, dass ein XML-basiertes Informationssystem diese Transformationsschritte geeignet bewältigen kann. Andererseits wurde auch deutlich, dass dies noch mit einem erheblichen Implementierungsaufwand verbunden ist. Zwar gibt es zahlreiche Programme, welche die Verarbeitung von XML grundsätzlich gewährleisten (z.B. Cocoon, AxKit, Zope). Diese wurden den von JOHANSSON und SIIRILÄ gestellten Anforderungen jedoch nicht gerecht. Daher mussten für die Transformationsschritte jeweils eigene Konverter in Perl

programmiert werden, wobei auf die Unterstützung von verschiedenen Perl-Modulen zurückgegriffen werden konnte (vgl. JOHANSSON und SIIRILÄ 2001).

Dieser Ansatz ist insgesamt mit einem deutlich höheren Implementierungsaufwand und einer nicht zu unterschätzenden höheren Komplexität des Systems verbunden. Für das geplante System wäre vermutlich analog zur Vorgehensweise von JOHANSSON und SIIRILÄ die Erstellung eines Konvertierungsprogramms für die Transformation von Dateien aus dem ESRI Shapefile-Format in GML und ein Konvertierungsprogramm für die Umwandlung von GML in SVG notwendig. Im günstigsten Fall könnten bestehende Konvertierungsprogramme in das System eingebunden und zu automatisierten Verarbeitungsketten kombiniert werden. Das System würde dadurch zu einem Mapserver erweitert werden.

Eine Abwägung des Nutzens einer auf GML basierenden Architektur im Vergleich zu dem damit verbundenen Aufwand wird nachfolgend für das zu entwickelnde System und dessen Besonderheiten vorgenommen.

Wie in Kapitel 1.4 beschrieben, besteht eine Besonderheit des zu entwickelnden Systems darin, dass die Anzahl der kartographisch abzubildenden Objekte, und damit auch der Probestellen, einer relativ geringen Dynamik unterliegen. Die Karte des Informationssystems ist demzufolge verhältnismäßig statisch. Außerdem sind die graphischen Objekte allesamt in einem Ausgangsdatenbestand vorhanden. Die Vorteile der Datenintegration von GML und der „on-the-fly“-Generierung von SVG-Karten sind damit zu vernachlässigen. Zwar kann – wie oben diskutiert – die Menge der zu transferierenden Daten verringert werden. Dies gilt aber nur, wenn nicht ein wiederholtes Ein- und Ausblenden der Layer erfolgt, weil sonst jeweils neue SVG-Dateien generiert, komprimiert und dann an den Client übermittelt werden müssten.

Eine weitere Besonderheit der vorliegenden Geometriedaten ist, dass neben den Probestellen-IDs keine weiteren Sachattribute für die abgebildeten Objekte vorliegen. Zudem ist in SVG mit der Verwendung des <use> Elements und dessen Attribut „xlink:href“ die Darstellung von Probestellen durch unterschiedliche Symbole möglich und interaktiv clientseitig steuerbar. D.h.

dass die Probestellen in SVG letztlich nur durch die Koordinaten des Punktobjekts bzw. der Punktsignatur definiert werden. Davon würde eine Modellierung in GML nicht entscheidend abweichen.

Von gewichtiger Bedeutung ist auch, dass aufgrund des bereits bestehenden Datenbankansatzes zur Verwaltung der Dokumentenbestände eine größtmögliche Aktualität über Datenbankabfragen aus einer SVG-Karte unabhängig von weiterer XML-Technologie möglich ist. Eine Transformierung der Datenbankinhalte in XML-Dokumente als standardisierte Schnittstelle wäre im Falle einer verteilten Organisation der Metadaten in Betracht zu ziehen oder dann, wenn das System auf andere Plattformen übertragen werden müsste, bei denen die serverseitigen Voraussetzungen für eine Datenbankanbindung nicht gegeben wären.

Einige Vorteile von GML, wie die Möglichkeit der Ausgabe der gespeicherten Informationen auf verschiedenen Medien, die Integration und Visualisierung verteilter Daten innerhalb des Informationssystems sowie Erweiterungen, wie die Bereitstellung von einzelnen Diensten und deren Integration in eine Dienstleistungskette sind nicht Gegenstand der gegenwärtigen Anforderungen an das System. Die Vorteile beziehen sich also insgesamt auf Aspekte, die eher langfristiger Natur sind und derzeit nicht intendierten bzw. vorhersehbaren Nutzungsänderungen oder Erweiterungen des Systems entgegenkommen würden. Daher wird der Aufwand der Implementierung einer Architektur mit GML im Vergleich zu dem potenziellen Nutzen als zu groß erachtet.

Aus den durchgeführten Untersuchungen verschiedener Technologien bzw. deren beispielhaft herausgegriffenen Implementierungen und aus den Abwägungen der jeweiligen Vor- und Nachteile lässt sich folgern, dass SVG in Bezug auf die an das zu entwickelnde System gestellten Anforderungen insgesamt am besten geeignet ist. Diese Aussage soll in den folgenden Hypothesen konkretisiert werden.

## 2.4 Hypothesen

### **These 1:**

Eine offene, herstellerunabhängige Architektur genügt den an das Informationssystem gestellten Anforderungen am besten. Dies ist v.a. damit zu begründen, dass auf diese Weise ein plattformunabhängiger Zugriff auf die Dokumentenbestände und eine Minimierung der finanziellen Aufwendungen in der Entwicklung und Wartung des Systems sowie für die potenziellen Benutzer erreicht werden kann.

### **These 2:**

SVG eignet sich, ergänzt um weitere Web-Technologien wie CGI-Skripte und JavaScript, für die Konzipierung interaktiver Karten. Der für dieses Informationssystem notwendige Grad an Interaktivität ist daher umsetzbar.

### **These 3:**

Für die Erreichung der formulierten Ziele sind Vektor-Formate besser geeignet als Raster-Formate. Dies ist darauf zurückzuführen, dass mit Vektor-Formaten einerseits mehr Interaktivität und andererseits eine bessere Performance bei gleichzeitig hoher Auflösung zu realisieren ist.

### **These 4:**

Mit SVG können Karten erstellt werden, die eine einfache, intuitive Erschließung von Dokumenten gewährleisten. Zu begründen ist dies damit, dass SVG allgemeinen wie auch internetspezifischen kartographischen Ansprüchen gerecht wird.

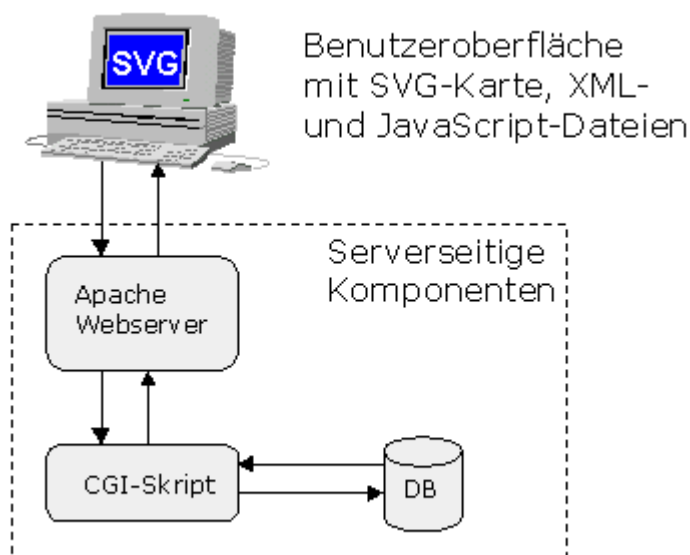
### 3 Entwicklung des Web-Informationssystems

#### 3.1 Systemarchitektur

Architektur wird vom OGC (2003) im Kontext der Informationsverarbeitung definiert als „*An abstract technical description of a system or collection of systems. (...) Conceptually based, architecture does not contain the level of detail needed for construction*“.

In Übereinstimmung mit dieser Definition sollen in diesem Kapitel die grundlegenden Komponenten in ihrer Bedeutung und – soweit noch nicht geschehen – auch in ihrer Eignung ausgeführt werden.

Wie Abb. 3 zeigt, ist die Architektur in eine clientseitige Komponente (die Benutzeroberfläche mit SVG-Dokumenten, HTML-Dokumenten und JavaScript-Dateien) und in drei serverseitige Komponenten (Datenbank, Webserver und CGI-Skripte) gegliedert.



**Abb. 3: Systemarchitektur**

Quelle: eigener Entwurf

### 3.1.1 Webserver

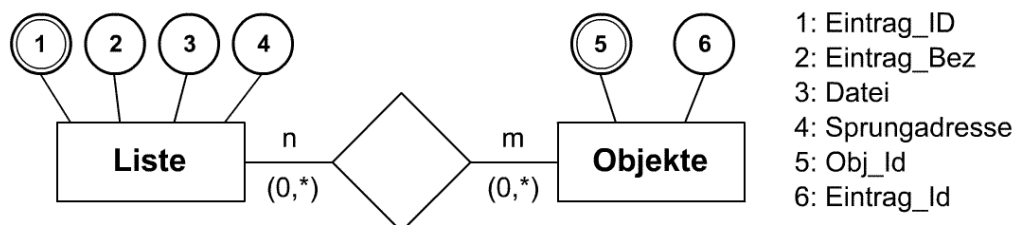
Serverseitig ist ein Webserver ein integraler Bestandteil des Web-Informationssystems. Dabei handelt es sich um ein Programm, welches die vom Benutzer über die Eingabe einer URL (Uniform Resource Locator) im Internet-Browser angeforderte Internetseite an den Client übermittelt. Für die Realisierung der interaktiven Funktionen mit Lese- und Schreibzugriff muss der Webserver zudem serverseitig Programme ausführen können. In diesem Zusammenhang sollte der Webserver außerdem die Möglichkeit bieten, den Zugriff auf Dateien passwortgeschützt zu begrenzen. Dafür ist die Fähigkeit zur Ver- und Entschlüsselung der Passworte notwendig. Für das zu implementierende Web-Informationssystem wird der „HTTP (Web) Server“ von Apache verwendet. Er erfüllt die oben genannten Bedingungen, läuft stabil unter dem serverseitig installierten Betriebssystem Windows NT und ist zudem gratis über die Apache Homepage (URL 11) zu beziehen.

### 3.1.2 Datenbank

Die Datenbank des Informationssystems basiert auf MS Access. Darin werden die Metadaten über die Dokumentenbestände verwaltet. Diese Komponente konnte ohne wesentliche Veränderungen von dem System von RAIBER (2001) übernommen werden. Die Verwendung von offenen Datenbanken wie MySQL würde zwar insgesamt besser zur offenen Gesamtarchitektur passen, ein Mehrwert ist durch eine derartige Migration gegenwärtig aber nicht erkennbar. Entscheidend ist, dass über das Internet ein plattformunabhängiger Zugriff auf die Datenbank geschaffen werden kann. Da der Server auf einem Windows-Rechner läuft, kann als Schnittstelle zur Datenbank ODBC (Open Database Connectivity) eingesetzt werden. ODBC gestattet die Kommunikation mit der Datenbank via dem Datenbankabfrage-Standard SQL (Structured Query Language). Deshalb ist dieser Ansatz sehr gut für plattformunabhängige Zugriffe geeignet.

Die Metadaten liegen in zwei Datenbank-Tabellen vor. Die Tabelle „Liste“ enthält Informationen über die Dokumentenbestände. Für jedes Dokument werden u.a. ID (Eintrag\_ID), Titel (Eintrag\_Bez), URL (Datei) und

Sprungadresse eines Dokuments gespeichert. In der Tabelle „Objekte“ sind die Identifikationsnummern aller Probestellen (Obj\_Id) und die IDs der dazugehörigen Dokumente (Eintrag\_Id) festgehalten. Jede Zeile der Tabelle „Objekte“ kann über die ID des Dokuments mit den Probestellen in der Tabelle „Objekte“ verknüpft werden. Über diese Verbindung können zu einer Probestelle die thematisch in Bezug stehenden Dokumente ermittelt werden. Zwischen Probestellen und Dokumenten besteht eine n:m-Beziehung, in beiden Fällen mit den Kardinalitäten (0,\*). Es können also sowohl zu einer Probestelle mehrere Dokumente als auch zu einem Dokument mehrere Probestellen in Beziehung stehen. Einige Probestellen haben (noch) keinen Bezug zu Dokumenten, manche Dokumente weisen überhaupt keinen Raumbezug auf. Diverse Dokumente stehen zwar logisch in einer Beziehung zu Probestellen, sind in der Datenbank aber noch nicht damit verlinkt. Die Relationen der beiden Tabellen sind im Entity-Relationship-Diagramm (ER-Diagramm) in Abb. 4 dargestellt:



**Abb. 4: ER-Diagramm der Metadatenbank**

Quelle: eigener Entwurf

Darüber hinaus erhält die Datenbank die Funktion der Speicherung von Benutzereinstellungen in einer weiteren Tabelle.

### 3.1.3 Common Gateway Interface

Die Kommunikation zwischen Webserver und Datenbank läuft über das Common Gateway Interface ab. Dadurch können dem Benutzer des Systems aktuelle Daten aus der Datenbank zur Verfügung gestellt werden. Dies

funktioniert so, dass vom WWW-Browser an den Webserver statt der Adresse einer HTML-Seite die Adresse eines CGI-Skripts mit optionalen Parametern gerichtet wird. Der Webserver führt dann das CGI-Skript unter Berücksichtigung der angegebenen Parameter aus und ermöglicht auf diese Weise über ODBC einen Zugriff auf die Datenbank. Damit liefert CGI den Rahmen für die Ausführung von Diensten, die direkt auf die im WWW-Browser definierten Wünsche des Benutzers eingehen (vgl. BOWEN 1997).

CGI-Programme werden zumeist in Visual Basic, C oder Perl erstellt. Aufgrund der freien Verfügbarkeit und problemlosen Installation wird im Rahmen dieses Web-Informationssystems Perl für die Durchführung von Datenbankabfragen eingesetzt. Da Perl nur über eine sehr begrenzte Anzahl an vordefinierten Funktionen verfügt, war die Installation von zwei Modulen notwendig, welche diesen Funktionsumfang für den Zugriff auf Datenbanken erweitern. Diese Module heißen DBI und DBD-ODBC. Sie sind im Internet gratis unter URL 12 zu beziehen. Nach Installation dieser Module kann über die Datenbankschnittstelle ODBC auf die dort registrierten Datenbanken zugegriffen werden.

### **3.1.4 Clientseitige Komponente**

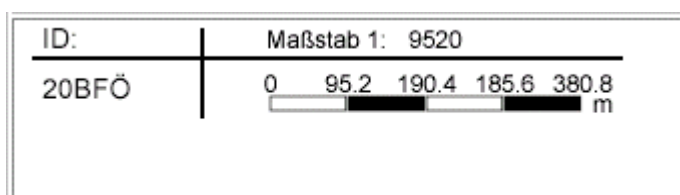
Die clientseitige Komponente des Systems besteht aus verschiedenen Dateien, die beim Aufruf des Informationssystems an den Client übermittelt werden. Die fünf Frames der Benutzeroberfläche stehen – mit Ausnahme der in HTML programmierten Toolbox – jeweils für ein SVG-Dokument (vgl. Abb. 5). Für die Anpassung dieser Dateien müssen clientseitig JavaScript-Funktionen verfügbar sein. Aufgrund des Umstandes, dass die Browser von Microsoft und Netscape insbesondere bei dynamischen Anwendungen unterschiedliche Befehle erfordern, erfolgte die Implementierung des Systems vorläufig nur für den Internet Explorer von Microsoft. Die Architektur ist aber so ausgelegt, dass eine Anpassung an andere Browser prinzipiell möglich ist.



**Abb. 5: Einteilung der Oberfläche des Systems nach Funktionseinheiten**

Quelle: eigener Entwurf

Den größten Teil der Benutzeroberfläche nimmt die eigentliche Karte ein, über welche die Dokumentabfragen ausgeführt werden können. Rechts oben befindet sich ein Frame mit einer Legende zur Erläuterung der in der Karte verwendeten Signaturen. Darunter ist eine weitere Karte angebracht, welche das gesamte Untersuchungsgebiet im Überblick vollständig darstellt (Übersichtskarte). Unten links ist eine Statusleiste platziert (vgl. Abb. 6), in welcher der gewählte Maßstab in Form einer textuellen Angabe und einer Maßstabsleiste angezeigt wird. Zudem wird dort die ID derjenigen Probestelle eingeblendet, über die der Mauszeiger bewegt wird. Im Fenster Toolbox sind Funktionalitäten für die Manipulation der Legendeneinstellungen (vgl. Kapitel 3.4.3) sowie der Pan- bzw. Zoomeinstellungen (vgl. Kapitel 3.4.4) untergebracht.



**Abb. 6: Statusleiste**

Quelle: eigener Entwurf

## 3.2 Datenaufbereitung

### 3.2.1 Generierung der SVG-Karte

Als grundlegender Arbeitsschritt für die Umsetzung der im vorigen Kapitel vorgestellten Architektur war die Konvertierung der ESRI Shapefiles aus dem bestehenden ArcView-Projekt in das SVG-Format notwendig. Dafür wurde die ArcView Extension SVG Mapper 1.3.9 verwendet. Um den Ansprüchen des geplanten Informationssystems Rechnung zu tragen, mussten für die Karte zahlreiche Verarbeitungsschritte durchgeführt werden, die im Folgenden genannt werden sollen.

Die bedeutendsten Signaturen der Karte sind die Punktsymbole, welche die Probestellen repräsentieren. Aufgrund der in Kapitel 2.2 diskutierten Anforderungen war der Entwurf neuer Symbole notwendig, welche den Rahmenbedingungen der Internetkartographie gerecht werden. Eine unveränderte Übernahme der Symbole des Systems von RAIBER (2001) war aufgrund des veränderten Anwendungskontextes also nicht möglich.



**Abb. 7: Vergleich der Probestellensymbole**

Quelle: eigener Entwurf

In Abb. 7 sind die neu gestalteten Symbole (mit WWW übertitelt) den Symbolen des Systems von RAIBER (mit ArcView übertitelt) gegenüber gestellt, an denen sie sich zu einem gewissen Grad orientieren. Die Symbole sind deutlich in zwei

übergeordnete Klassen getrennt: in terrestrische und in aquatische Probestellen. Für letztere wurden weitreichende Änderungen vorgenommen. Eine eindeutige Identifikation wird über die Verwendung einfacher, klar unterscheidbarer geometrischer Formen angestrebt. Eine Differenzierung über die Symbolgröße wurde aufgrund möglicher quantitativer Assoziationen nicht umgesetzt. Die farbliche Gestaltung bedient sich verschiedener Farbtöne im blau-grünen Spektrum. Damit sollen Assoziationen zu Gewässern hervorgerufen werden. Die Farbgebung der Probestellentypen „aquatische Probe-strecke“ und „untersuchtes Stillgewässer“ wurden in Anlehnung an die gleichnamigen Flächensymbole getroffen. Der Farbton wurde dabei übernommen, aus Gründen der Unterscheidbarkeit wurde die Helligkeit etwas erhöht (vgl. Abb. 8). Einer weiteren farblichen Differenzierung nach mnemotechnischen Aspekten stand die ausgesprochene Ähnlichkeit der Probestellentypen entgegen.

Die beiden terrestrischen Probestellentypen werden durch Kreissymbole dargestellt. Farblich heben sie sich deutlich von den aquatischen Probestellensymbolen ab. Da die Unterscheidung von nur zwei Probestellentypen in diesem Fall über die Farbe eindeutig möglich ist, konnte auf die Verwendung einer weiteren geometrischen Form verzichtet werden. Die Differenzierung dieser Symbole mittels eines Punktes in der Kreismitte des Symbols „terrestrische Untersuchungsflächen und LfU Dauerbeobachtungsflächen“ konnte im Hinblick auf die Kombinierbarkeit der Probestellensymbole mit weiteren Symbolen (vgl. Kapitel 3.4.3) nicht von RAIBER übernommen werden.

Die in Abb. 7 dargestellten Symbole sollten den gestellten Forderungen insgesamt entsprechen können. Sie sind in Form oder Farbe klar unterscheidbar, haben eine einfache Grundform, sind mit anderen Symbolen kombinierbar und bedienen sich – soweit als möglich – entsprechender Farbassoziationen.

Obwohl die Symbole aus ArcView nicht übernommen wurden, war dennoch die Konvertierung der Probestellensymbole aus ArcView mit dem SVG Mapper notwendig. Dieser konvertierte die ArcView Symbole in einfache Kreissymbole

unter Verwendung des Elements `<circle>`. Entscheidend war, dass die Koordinaten der Symbole als Attributwerte den „x“- bzw. „y“-Attributen des `<circle>` Elements zugewiesen wurden. In einem weiteren Verarbeitungsschritt musste dann, unter Beibehaltung der Attribute, der Elementtyp von `<circle>` nach `<use>` geändert werden. Damit ist es möglich, über das „xlink:href“-Attribut die neu erstellten Symbole zu referenzieren. Die geometrischen Formen der Symbole mussten dafür in das Element `<symbol>` gefasst werden. Um eine korrekte Bezugnahme zu ermöglichen, erhielt jedes Symbol eine eindeutige ID. Der SVG Mapper hatte für die Probestellen demzufolge nur die Funktion der Übertragung der Koordinatenwerte. Allerdings wurden den y-Werten dabei negative Vorzeichen vorangestellt. Auf eine Anpassung wurde verzichtet, da dies für die Wartung des Systems in diesem Fall kein signifikantes Problem darstellen sollte.

In einem weiteren Schritt mussten die vom SVG Mapper generierten IDs der die Probestellen repräsentierenden `<use>` Elemente manuell so geändert werden, dass sie den IDs der Datenbank-Tabelle „Objekte“ entsprechen. Dies ermöglicht eine einfache Verlinkung mit der Datenbank.

Daneben wurde eine weitere manuelle Änderung durchgeführt. Die unterschiedlichen Probestellentypen lagen in ArcView in Form nur eines Layers vor. Um dem Benutzer einen größeren Einfluss auf die Auswahl der dargestellten Probestellentypen und damit auf eine differenziertere Variation der Darstellungsdichte zu geben, wurde der Layer mit den Probestellen nach Probestellentypen in einzelne Layer aufgesplittet. Die Gruppierung gleichartiger Objekte in einen Layer wurde mit Hilfe des Elements `<g>` realisiert, indem die Symbole eines Probestellentyps je durch ein eröffnendes und schließendes `<g>` Element zusammengefasst wurden. Um die Layer eindeutig adressieren zu können, wurde dem Attribut „ID“ manuell jeweils eine eindeutige Identifikationsnummer zugewiesen.

Die als Hintergrund bzw. Orientierung dienenden linien- und flächenhaften graphischen Objekte konnten ohne Veränderungen der Signaturen in das System übernommen werden. Etwas problematisch war jedoch, dass gerade diese nur sekundäre Information, insbesondere aber die Ufer- und Wegelinien,

im System den größten Speicherbedarf in Anspruch nehmen. Dies liegt daran, dass für jede Richtungsänderung des Linienvverlaufs ein zusätzliches Koordinatenpaar gespeichert werden muss. Diese Information nimmt im SVG-Dokument der Karte 1.781 von 2.089 Zeilen ein. Damit werden über 85% der 431 KB großen Datei mit Hintergrundinformationen belegt.

Für die Gewährleistung einer hinreichend guten Performance des Systems waren daher Generalisierungen durchzuführen. Da dies mit dem SVG Mapper nicht möglich ist, mussten die Dateien in ArcView angepasst werden. Die ArcView Extension P/PL-Tools bietet mit der Funktion „Generalize Features“ einen geeigneten Zugang. Diese Funktion generalisiert die Liniengeometrien unter Verwendung des Douglas-Peucker-Algorithmus. Die Funktionsweise dieses Algorithmus soll an dieser Stelle kurz skizziert werden. Der Algorithmus entfernt Vertices aus Linien, die unterhalb einer zu definierenden Toleranzgrenze liegen. Dazu wird eine gedachte Hilfslinie zwischen dem Start- und dem End-Vertex herangezogen. Alle Vertices, deren Entfernung (rechtwinklig von der Hilfslinie gemessen) die Toleranzgrenze zu der gedachten Linie unterschreiten, werden gelöscht. Diese Schritte werden dann rekursiv jeweils für den Vertex wiederholt, der den größten Abstand zu der Hilfslinie aufweist. Dabei wird von diesem Vertex eine Hilfslinie zum Start-Vertex und eine Hilfslinie zum End-Vertex konstruiert. Der Algorithmus terminiert, wenn kein weiterer Vertex die angegebene Toleranzgrenze mehr unterschreitet (vgl. WORBOYS 1995).

Nach der Durchführung von Testläufen mit verschiedenen Toleranzwerten wurden die Ufer- und Wegelinien jeweils mit einem Wert von zwei Metern generalisiert. Die Zahl der Vertices der Uferlinien konnte auf diese Weise von 10.008 auf 3.998 reduziert werden. Eine vergleichbare Verminderung ergab sich auch bei den Wegelinien. Die Ausführung des Algorithmus mit höheren Toleranzwerten führte dagegen zu wesentlich deutlicheren Abweichungen des Verlaufs der Linienzüge bei nur unwesentlicher Verringerung der Zahl der Vertices.

Im Anschluss an die Generalisierung wurden die Shapefiles der Ufer- und Wegelinien wieder mit dem SVG Mapper in SVG konvertiert und anstelle der













nicht-generalisierten Linienzüge in die ursprüngliche SVG-Datei eingebunden. Diese Datei konnte dadurch von 431 KB auf nur noch 262 KB reduziert werden. Die weiteren SVG-Dateien (Legende, Übersichtskarte, Statusleiste) haben einen deutlich geringeren Umfang und bedürfen daher keiner Generalisierung.

Darüber hinaus können SVG-Dateien zusätzlich komprimiert werden, was diesbezügliche Nachteile gegenüber binären Formaten wie Flash weiter verringert. Mit der Verwendung des Komprimierungsprogramms win-gz konnte die Größe der SVG-Datei von 262 KB auf nur noch 46 KB nochmals sehr deutlich verringert werden und die Performance des Systems bei gleicher optischer Qualität entscheidend verbessert werden.

### 3.2.2 Generierung der Legende

Die Legende zur Karte wird von der nachfolgenden Abbildung gezeigt:

#### Legende

-  terrestrische Untersuchungsflächen & LfU Dauerbeobachtungsflächen
-  terrestrische Untersuchungsflächen
-  aquatischer Probepunkt
-  Kleingewässer
-  aquatische Probestrecke
-  untersuchtes Stillgewässer
- Text Labels
-  Ufer
-  Wege
-  Damm
-  aquatische Probestrecke
-  untersuchtes Stillgewässer
-  Bauwerke

#### Themenbezüge

-  Boden
-  Ökologie
-  Waldökologie

**Abb. 8: Legende des Web-Informationssystems**

Quelle: eigener Entwurf

Die Legende wurde von Grund auf eigenständig erstellt. Dies ist zwar auch mit dem SVG Mapper möglich, der Umfang der dabei durchzuführenden Änderungen erschien aber zu groß. Für die Legende wurde ein Schaltflächen-Symbol zum Ein- und Ausblenden der Layer entworfen und mit den entsprechenden Layern der Karte verknüpft (vgl. Kapitel 3.4.2). Die Symbole der Probestellen konnten aus der Karte kopiert und in die Legende integriert werden. Die flächen- bzw. linienhaften Signaturen wurden mittels einfacher `<rect>` respektive `<polyline>` Elemente umgesetzt. Anschließend erfolgte eine Komprimierung der Datei.

### 3.2.3 Konvertierung der Dokumente

Die Dokumente, auf welche das zu entwickelnde Informationssystem einen Zugriff ermöglichen soll, lagen als MS Word- und MS Excel-Dateien sowie als ESRI Shapefiles vor. Der Microsoft Internet Explorer kann zwar Word- und Excel Dokumente unmittelbar im Browser anzeigen. Dies ist aber nur möglich, wenn auf dem PC des Nutzers MS Word und MS Excel installiert sind. Dies ist einer breiten Nutzung des Systems mit möglichst geringen Zugangsbeschränkungen nicht dienlich und widerspricht der Zielsetzung, offene Datenformate zu verwenden.

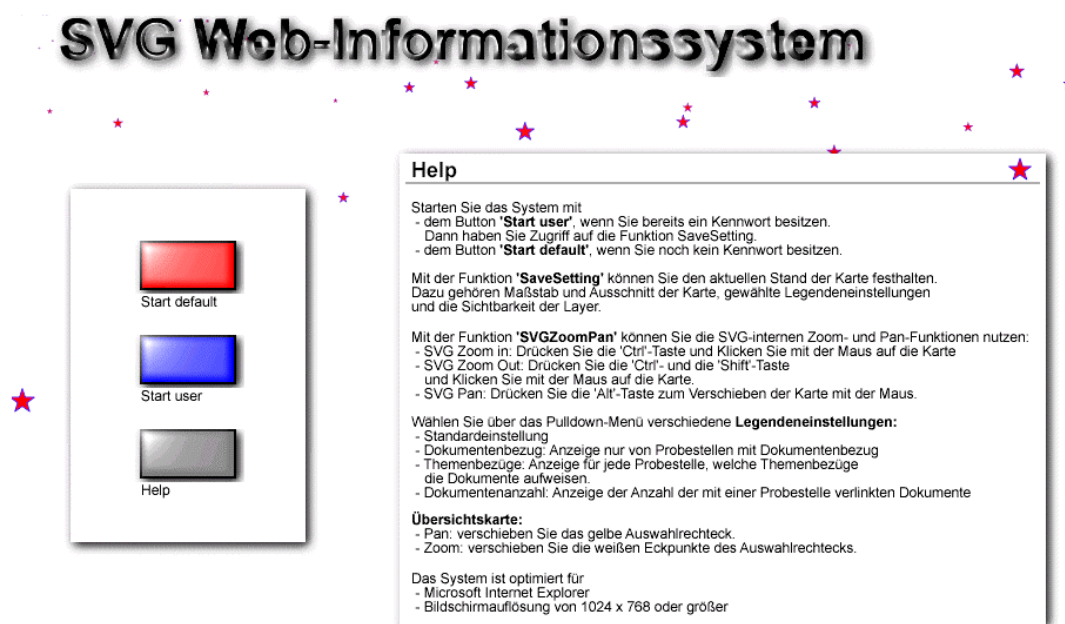
Daher wurden einige ausgewählte Dokumente exemplarisch nach HTML konvertiert. Dies ist aus Word 2000 und Excel 2000 einfach möglich. Sehr vorteilhaft ist überdies, dass auch Excel-Dokumente mit mehreren Blättern durch die Verwendung von Frame-Technologie in geeigneter Form nach HTML konvertiert werden können. Die Konvertierung der ESRI-Shapefiles wurde an einem Beispiel analog zu der Konvertierung der Karte des Informationssystems mit dem SVG Mapper vorgenommen (vgl. Kapitel 3.2.1).

Da Probestellen mitunter nicht mit einem ganzen Dokument, sondern nur mit bestimmten Abschnitten daraus in Beziehung stehen, waren für die generierten HTML-Dokumente noch Anpassungen vorzunehmen, um einen direkten Zugriff auf ein Kapitel innerhalb eines Dokuments zu ermöglichen. In HTML ist diese Funktion mittels der Verwendung so genannter Anker in Verbindung mit Hyperlinks leicht umsetzbar:

```
<a href="document.htm#Anker">Sprungbefehl</a>
<a name="Anker">Ziel der Sprungmarke</a>
```

Das Klicken auf den Text „Sprungbefehl“ führt in diesem Beispiel direkt zum Sprung an die mit dem Namen „Anker“ angegebene Stelle im Dokument. Diese Anker wurden bei der Konvertierung automatisch aus den Word- bzw. Excel-Dokumenten in die neu generierten HTML-Dateien übernommen.

### 3.3 Prozesse beim Start des Systems



**Abb. 9: Startseite des Web-Informationssystems**

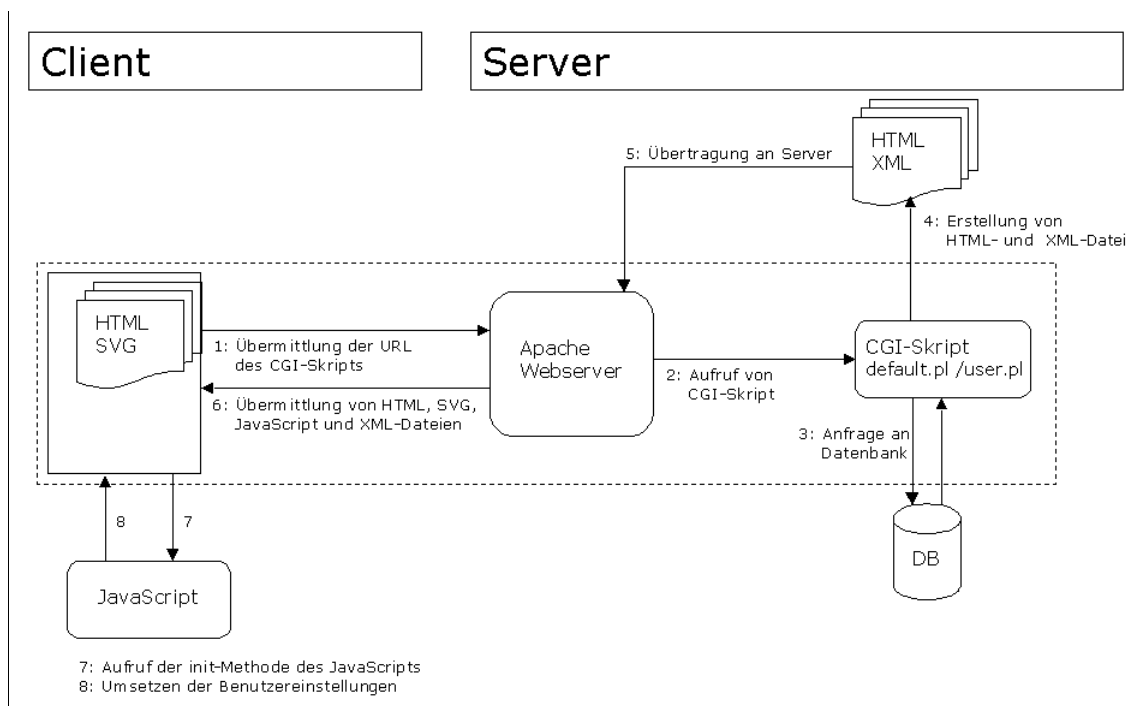
Quelle: eigener Entwurf

Die Startseite des Systems ist ein in HTML eingebettetes SVG-Dokument (vgl. Abb. 9). Dieses wird beim Aufruf des Systems mit einem Perlskript erstellt und an den Client übermittelt.

Von dieser Seite aus gibt es zwei Möglichkeiten, das Web-Informationssystem zu starten. Dafür wurden in zentraler Position zwei <rect> Elemente als Schaltflächen angebracht, die wie gewöhnliche HTML-Buttons anklickbar sind. Dabei kann der Nutzer sich über den Button „Start user“ mittels eines Kenn- und Passwortes einloggen und damit das System mit den für ihn gespeicherten

Benutzereinstellungen starten. Über den Button „Start default“ besteht freier Zugang zum System, jedoch ohne die Funktionalitäten der Speicherung von Benutzereinstellungen. Von der Startseite existiert zudem die Möglichkeit, über den Button „Help“ ein Fenster zu öffnen, das über grundlegende Funktionen des Systems informiert.

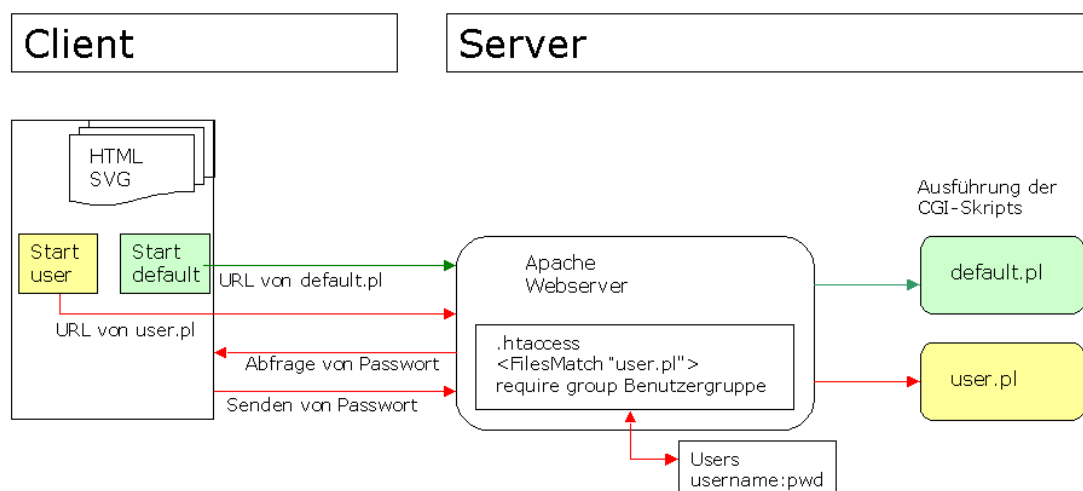
Die Umsetzung der Startseite ist in funktionaler Hinsicht mit HTML genauso gut möglich. Die Entscheidung für eine dynamisch generierte SVG-Seite zielt auf eine variable und optisch ansprechendere Gestaltung.



**Abb. 10: Prozesse beim Start des Systems**

Quelle: eigener Entwurf

Die Prozesse beim Start des Systems werden nachfolgend detaillierter erläutert. Abb. 10 zeigt eine generalisierte Darstellung der Prozesse. Der in der Abbildung in einen gestrichelten Rahmen gefasste Teilprozess wird in Abb. 11 näher ausgeführt.



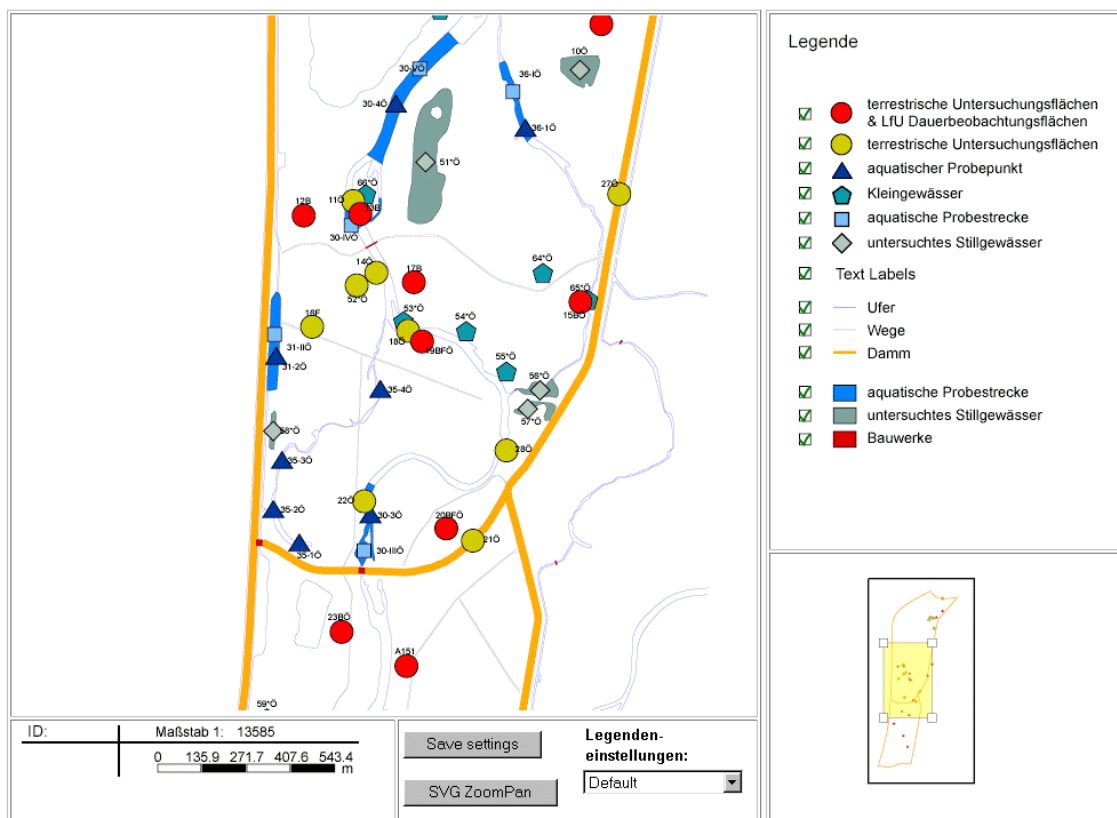
**Abb. 11: Teilprozess aus Abb. 10 mit Einbeziehung der Benutzererkennung**

Quelle: eigener Entwurf

Über das Attribut „onclick“ der Startbuttons wird – abhängig vom gewählten Button – ein Perlskript aufgerufen, welches ein neues HTML-Dokument generiert. In dieses Dokument werden die für die Benutzeroberfläche des Systems notwendigen SVG-Dokumente eingebettet und zusammen mit den benötigten JavaScript- und XML-Dateien an den Client zurückgeschickt.

Der Apache Webserver bietet die Möglichkeit, in der so genannten htaccess-Datei Angaben über Zugriffsrechte für Dateien zu definieren. Der Eintrag `<FilesMatch ...>` legt fest, welche Dateien passwortgeschützt sind. In diesem Fall wird das Perlskript `user.pl` erst dann vom Webserver ausgeführt, wenn ein Benutzer sich korrekt über eine Eingabemaske identifiziert und authentifiziert hat. Die User-Namen stehen zusammen mit den verschlüsselten Passwörtern in einer Text-Datei (Users, vgl. Abb. 11). Die Datei `default.pl` hat hingegen keinen Eintrag in der htaccess-Datei. Deshalb wird sie ohne Durchführung einer Benutzererkennung ausgeführt. Der Unterschied bei der Generierung der HTML-Datei durch die Perlskripte besteht darin, dass sie im Default-Modus keinen Button für die Speicherfunktionalität der Benutzereinstellungen erhält. Außerdem entfällt in diesem Fall das Auslesen und Verarbeiten nutzerspezifischer Daten.

Unabhängig von dem gewählten Modus wird von den Skripten eine XML-Datei generiert. Die darin einzutragenden Informationen werden bei jedem Aufruf des Systems unmittelbar aus der Datenbank gewonnen. Die Übermittlung der nur ca. 2 KB großen Datei kann die Anzahl der Datenbankabfragen während der Benutzung des Systems, namentlich bei der Wahl unterschiedlicher Legendeneinstellungen (vgl. Kapitel 3.4.3), entscheidend verringern. Auf die Details der Generierung, der Inhalte und der Verwendung der XML-Datei wird im Kontext der jeweiligen Funktionen eingegangen (vgl. Kapitel 3.4.3 und Kapitel 3.4.5).



**Abb. 12: Oberfläche des Web-Informationssystems**

Quelle: eigener Entwurf

Direkt beim Aufruf der HTML-Seite werden einige grundlegende und häufig benötigten Variablen in JavaScript initialisiert. Dies sind insbesondere die Variablen, denen gemäß dem DOM SVG-Dokumente in Form eines Objekts zugewiesen werden. Dazu gehören die Karte, die Legende, das Übersichts- und das Statusfenster. Im Falle des Starts über den Button „Start user“ werden

die Daten der Benutzereinstellungen in JavaScript aus der XML-Datei ausgelesen und den entsprechenden Attributen der betroffenen SVG-Elemente zugewiesen (vgl. Kapitel 3.4.5).

Nach Übermittlung der Dateien an den Client zeigt der Internet-Browser die in Abb. 12 dargestellte Systemoberfläche an.

### 3.4 Umsetzung der interaktiven Funktionen

In den folgenden Unterkapiteln wird die Umsetzung der interaktiven Funktionen des Informationssystems vorgestellt und mit Hilfe von Ablaufschemata und Code-Fragmenten dokumentiert.

#### 3.4.1 Aufruf von Dokumenten über den Raumbezug

Einen Überblick über die Prozesse und Zusammenhänge bei der Dokumentensuche geben Abbildung 14 und 15:

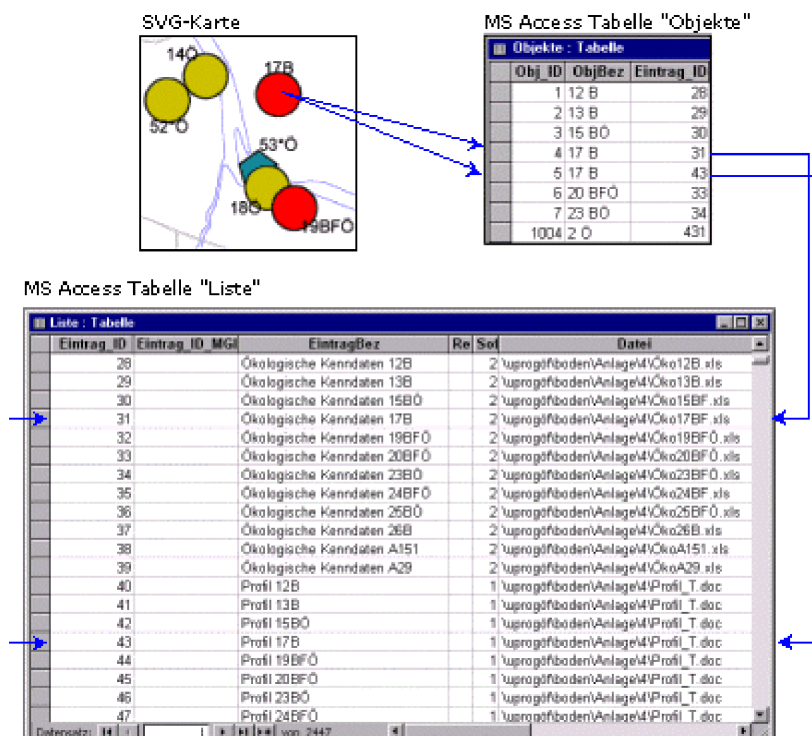
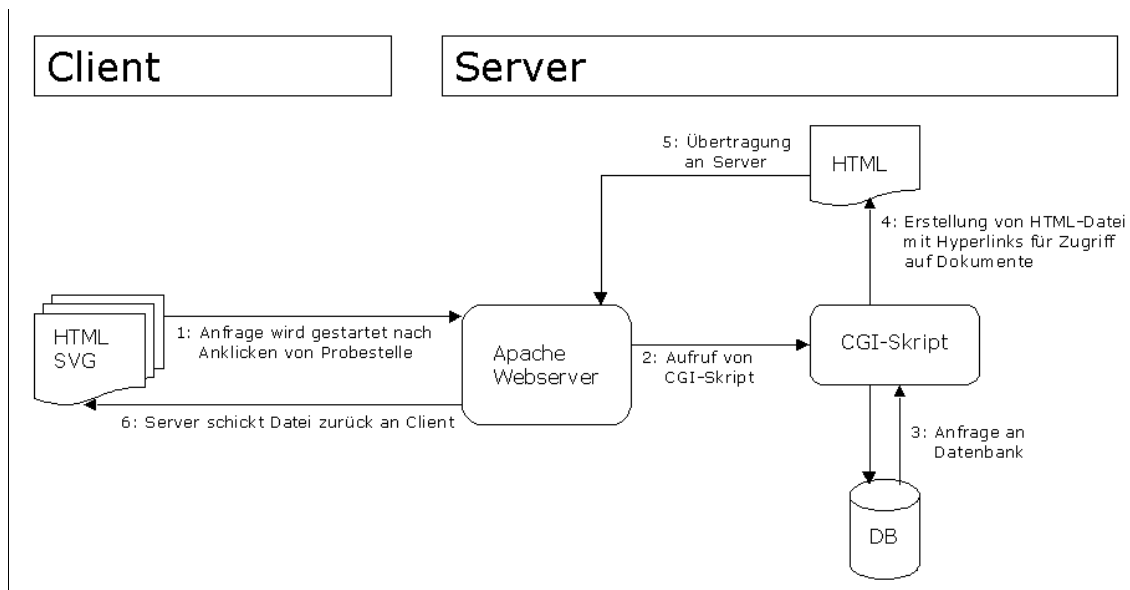


Abb. 13: Verlinkung von Probestellen mit Dokumenten

Quelle: RAIBER 2001, S. 35, verändert



**Abb. 14: Prozesse bei Abfrage nach Dokumenten**

Quelle: eigener Entwurf

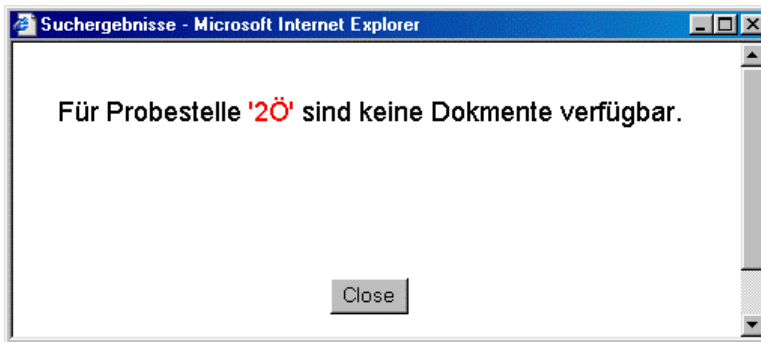
Anfragen nach zu Probestellen vorhandenen Dokumenten startet der Benutzer am Internet-Browser mit einem Mausklick auf das graphische Symbol der betreffenden Probestelle. Über die im Attribut „onclick“ angegebene JavaScript-Funktion wird dem Server die Adresse eines Perlskripts übermittelt. Diese kann wie folgt aussehen:

*<http://localhost/wis/cgi-bin/query.pl?17B>*

Zu beachten ist dabei der mit der Adresse nach dem Fragezeichen übergebene Parameter. Er trägt die ID der angeklickten Probestelle. Dieser Parameter kann nun im Perlskript über die Perl-Umgebungsvariable `$ENV{'QUERY_STRING'}` ermittelt werden und steht zur Weiterverarbeitung zur Verfügung. Daraufhin wird in der Tabelle „Objekte“ der Datenbank mit dem unten aufgeführten SQL-Befehl nach Einträgen zu dieser ID gesucht, wobei die Variable `$bezug` die ID beinhaltet:

*„SELECT Eintrag\_Id FROM Objekte WHERE ObjBez = \$bezug“;*

Wird eine leere Ergebnismenge zurückgegeben, liegen keine Dokumente vor. Dann wird die Datenbankabfrage abgebrochen und an den Internet-Browser eine Meldung über den Misserfolg der Anfrage zurückgeschickt:



**Abb. 15: Rückmeldung nach ergebnisloser Suche**

Quelle: eigener Entwurf

Sind dagegen Einträge vorhanden, werden in der Tabelle „Liste“ über die ermittelten IDs weitere Abfragen nach den Dokumenttiteln sowie den Dokument- und Sprungadressen gestartet:

```
„SELECT Titel FROM Liste WHERE Eintrag_Id = $id“;
„SELECT Url FROM Liste WHERE Eintrag_Id = $id“;
„SELECT Sprungadresse FROM Liste WHERE Eintrag_Id = $id“;
```

Die Ergebnisse werden in Arrays getrennt zwischengespeichert. Über die Elemente dieser Arrays wird anschließend iteriert, wobei die Inhalte in HTML-Code umgesetzt werden:

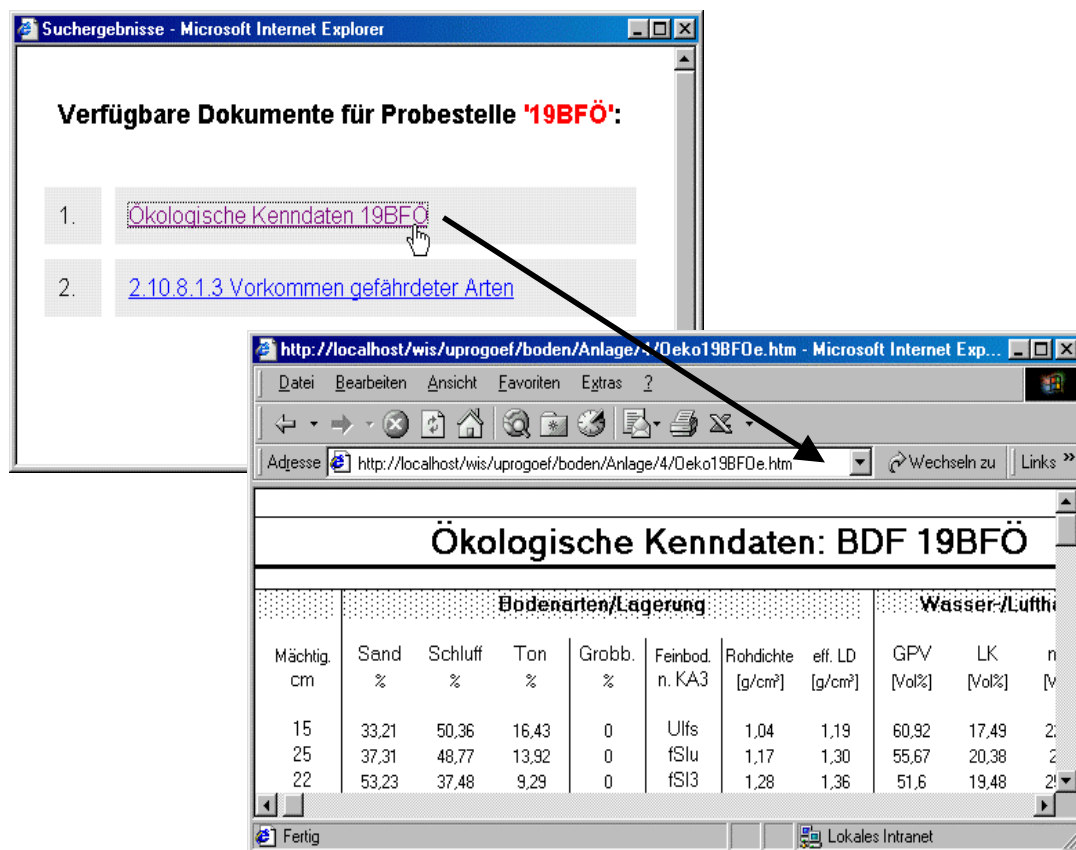
```
print "<a href = \" " . $path . @url[$z] . "\">\" . @titel[$z] . „#“ .
@sprungadresse[z]</a><br>“;
```

Dabei ist *\$path* die über Umgebungsvariablen ermittelte Adresse des Servers. Das Array *@url* enthält an Position *n* die Adresse des Dokuments zu dem in *@titel[n]* gespeicherten Dokumenttitel. Das Array *@sprungadresse[n]* speichert den dazugehörigen Verweis auf die gesuchte Stelle im Dokument. Der dabei generierte HTML-Code kann beispielsweise wie folgt aussehen:

```
<a href=http://localhost/geoportal/wis/uprogoef/boden/
Anlage/4/Oeko12B.htm#t98>&Ouml;kologische Kenndaten</a><br>
```

In diesem Zusammenhang hat es sich als Problem herausgestellt, dass einige der Dokumentadressen Umlaute beinhalten. Der Microsoft Internet Explorer unterstützt ab der Version 5.0 keine Umlaute mehr in Webadressen (vgl. PC WELT 2003). Dies hat zur Folge, dass entsprechende Webseiten nicht mehr angezeigt werden können. Deshalb mussten alle Dateinamen und

Verzeichnisse, die Umlaute enthielten, manuell umgeändert werden. Für die Dokumentadressen in der Datenbank wurde dieser Schritt nicht vollzogen. Deshalb müssen die aus der Datenbank ermittelten Dokumentadressen in Perl noch mit einer Unterfunktion auf Umlaute durchsucht und entsprechend ersetzt werden (z.B. von „ö“ nach „oe“). Dann stimmen Adresse und Verzeichnisstrukturen wieder überein. Dies bereitet einerseits einen geringeren Aufwand als die manuelle Änderung in der Datenbank, zum anderen erleichtert es die Pflege der Datenbank, da auf solche Umstände beim Hinzufügen neuer Dokumente nicht geachtet werden muss. Eine weitere Unterfunktion setzt die Umlaute der Dokumenttitel in internationale HTML-Notation um (z.B. von „Ö“ nach „&Ouml;“).



**Abb. 16: Suchergebnis der Dokumentenabfrage für Probestelle „19BFÖ“**

Quelle: eigener Entwurf

Die vom Perlskript erzeugte HTML-Datei wird dann vom Server an den Internet-Browser zurückgeschickt. Über die Verlinkung der Adresse mit dem Dokumenttitel hat der Benutzer dann Zugriff auf die Dokumente (vgl. Abb. 16).

### 3.4.2 Ein- und Ausblenden der Layer

In der Legende befindet sich neben den Symbolen der Layer jeweils eine Schaltfläche, die dem Ein- und Ausblenden des dazugehörigen Layers in der Karte dient (vgl. Abb. 8). Die Schaltfläche besteht aus einem grünen Häkchen (Auszeichnung als `<path>` Element) inmitten eines Quadrats (Element `<rect>`). Die IDs der Elemente sind an die IDs der damit korrespondierenden Layer der Karte angepasst. Dem Layer mit der ID „0“ entsprechen die Elemente mit der ID „0\_r“ (Rechteck) bzw. „0\_p“ (Path). Beide Symbole besitzen das Attribut „onclick“. Darüber kann eine JavaScript-Funktion aufgerufen werden. Die ID der Symbole (ohne den Zusatz ab dem Unterstrich) wird der Funktion als Parameter übergeben. Mit dieser ID wird dann über das Objekt der SVG-Legende der Wert des Attributs „visibility“ des betreffenden Häkchens ermittelt. Ist es sichtbar, so wird das Attribut auf unsichtbar gesetzt und der Layer in der Karte ausgeblendet. Ergibt die Abfrage des „visibility“-Attributs dagegen, dass das Häkchen bereits unsichtbar war, dann wird es zusammen mit dem dazugehörigen Layer wieder sichtbar gemacht (vgl. Abb. 17).

Der Layer bzw. das Gruppierungselement `<g>` wird in JavaScript, hier am Beispiel des Layers mit der ID „0“, mit folgender Funktion ermittelt, wobei die Variable `svgMapDoc` das SVG-Dokument der Karte repräsentiert:

```
layer = svgMapDoc.getElementById(0);
```

Die Veränderung der Sichtbarkeit wird dann mit der Funktion `layer.setProperty('visibility', 'visible')` bzw. `layer.setProperty('visibility', 'hidden')` durchgeführt. Der Zusammenhang zwischen der Sichtbarkeit der Schaltflächen und der dazugehörigen Layer wird in Abb. 17 verdeutlicht.

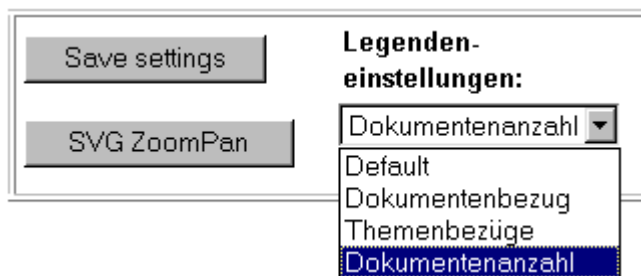
Symbol	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Layer
Element	<code>&lt;rect/&gt;</code>	<code>&lt;path/&gt;</code>	
ID	0_r	0_p	0
Visibility	visible	hidden	hidden
Visibility	visible	visible	visible

**Abb. 17: Sichtbarkeit von Schaltfläche und Layer**

Quelle: eigener Entwurf

### 3.4.3 Manipulation der Legendeneinstellungen

Neben der Standardeinstellung der Legende wurden drei weitere Legendeneinstellungen implementiert. Der Wechsel zwischen diesen Einstellungen ist über das Pulldown-Menü der Toolbox möglich (vgl. Abb. 18). Dabei wird beim Anklicken des gewünschten Listeneintrages die Ausführung einer JavaScript-Funktion initiiert. Für die Unterscheidung der Einträge im Pulldown-Menü wird der Funktion jeweils ein eindeutiger Wert als Parameter übergeben.

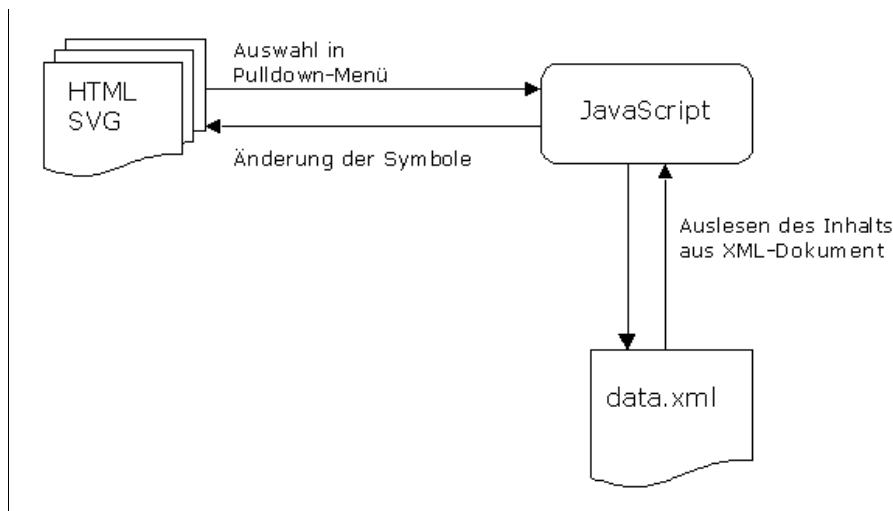


**Abb. 18: Toolbox mit Auswahlmennü für Legendeneinstellungen**

Quelle: eigener Entwurf

Die weiteren Legendeneinstellungen heißen „Dokumentenbezug“, „Themenbezüge“ und „Dokumentenanzahl“. Nach einem Klick auf den Eintrag „Dokumentenbezug“ werden nur noch die Probestellen angezeigt, für die ein Zugriff auf Dokumente möglich ist. Bei der Auswahl des Eintrages „Themenbezüge“ wird für jede Probestelle visualisiert, welchen Inhaltskategorien die Dokumente einer Probestelle zugeordnet werden können. Über „Dokumentenanzahl“ wird die Anzahl der über eine Probestelle zugänglichen Dokumente sichtbar gemacht. Das ursprüngliche Kartenbild kann mit einem Klick auf den Eintrag „Default“ wiederhergestellt werden.

Das Prinzip der bei der Änderung der Legendeneinstellungen ablaufenden Prozesse ist in Abb. 19 dargestellt und soll in den anschließenden Unterkapiteln konkretisiert werden.



**Abb. 19: Prozesse bei der Änderung der Legendeneinstellungen**

Quelle: eigener Entwurf

Die Änderung eines Symbols wird demzufolge immer in Abhängigkeit von den Einträgen der beim Start des Systems generierten XML-Datei vorgenommen. Diese Informationen werden unmittelbar aus der Datenbank gewonnen (vgl. Kapitel 3.3). Bei diesem Prozess werden aus der Tabelle „Objekte“ alle Probestellen mit Dokumentenbezügen ermittelt. Dann wird für jedes Element der Ergebnismenge nach den dazu vorhandenen Dokumenten in der Tabelle „Liste“ gesucht. Dabei wird über die Pfade der Dokumentadressen ermittelt, zu welcher Inhaltskategorie sie gehören. Diese Informationen werden dann für jede Probestelle, die Dokumentenbezüge aufweist, in den Variablen *\$boden*, *\$wald* und *\$oeko* gespeichert. Für die Ermittlung der Anzahl der Dokumente muss pro festgestelltem Dokumentenbezug lediglich der Wert einer Variablen (*\$count*) um Eins erhöht werden.

Für jede Probestelle wird nach Ermittlung der Werte folgender Perl-Befehl ausgeführt und solcherart die ermittelten Informationen in XML umsetzt:

```
print „<docBezug id=\"$bez\" soil=\"$boden\" forest=\"$wald\" eco=\"$oeko\"
count=\"$count\"/>“;
```

Einen Auszug der für die Manipulation der Legendeneinstellungen relevanten Daten dieser XML-Datei zeigt die folgende verkürzt dargestellte Auflistung:

```
<data>
  <docBezug id="12B" soil="true" forest="true" eco="false" count="2"/>
  <docBezug id="13B" soil="true" forest="false" eco="false" count="3"/>
  <docBezug id="15BÖ" soil="true" forest="true" eco="false" count="2"/>
  <docBezug id="17B" soil="true" forest="true" eco="false" count="2"/>
</data>
```

Alle Elemente dieser Auflistung wurden neu definiert und beziehen sich auf keine DTD. Das Wurzelement lautet `<data>`. Das Element `<docBezug>` speichert in den angegebenen Attributen Informationen über die zu den Probestellen vorhandenen Dokumente. Das Attribut `id` erhält dabei dieselbe ID wie die betreffende Probestelle. Die Attribute „soil“, „forest“ und „eco“ können die Werte „true“ oder „false“ annehmen und geben an, ob zu dieser Probestelle Dokumente der entsprechenden Inhaltskategorien existieren (vgl. Kapitel 3.4.3.2). Das Attribut „count“ speichert die Anzahl der Dokumente, auf die von einer Probestelle aus zugegriffen werden kann.

### 3.4.3.1 Dokumentenbezug

Diese Funktion dient der Ausblendung von Probestellen ohne Dokumentenbezüge. Dafür werden die dokumentenbezogenen Daten aus der beim Start des Systems generierten XML-Datei benötigt. Auf sie wird über das XML-DOM zugegriffen und mittels der Funktion `getElementsByTagName('docBezug')` alle Einträge des Elements `<docBezug>` zurückgegeben. Für jeden Eintrag wird mit der Funktion `getAttribute('id')` die ID der Probestelle ermittelt. Nach Ausführung des Prozesses sollen nur noch diejenigen Probestellen angezeigt werden, deren IDs auf diese Weise ermittelt werden konnten.

Die ursprünglich verfolgte Herangehensweise bestand darin, für alle Probestellen, die keinen Eintrag in der XML-Datei haben, das „visibility“-Attribut auf „hidden“ zu setzen. Es hat sich jedoch herausgestellt, dass dieser Weg nur scheinbar zum Ziel führt. Zwar gelingt es tatsächlich, diese Elemente unsichtbar zu schalten, dies hat allerdings auch unerwartete Folgen für den Layer, zu dem

diese Elemente gehören. Wird das „visibility“-Attribut eines Subelements von <g> verändert, so kann die Sichtbarkeit desselben Symbols über das ihm übergeordnete <g> Element nicht mehr verändert werden. Das bedeutet, dass das Ein- und Ausblenden eines Layers dann über die Legende nicht mehr möglich ist.

Daher bedient sich der verfolgte Lösungsansatz des „xlink:href“-Attributs des <use> Elements. Dieses soll dann, anstatt das bestehende Symbol unsichtbar zu machen, ein anderes, immer sichtbares, Symbol referenzieren. Da aber das „xlink:href“-Attribut all jener Elemente verändert werden muss, deren IDs nicht bekannt sind und die daher nicht adressiert werden können, ist eine direkte Lösung nicht möglich. Deshalb werden zunächst allen Probestellen bzw. den sie repräsentierenden <use> Elementen das unsichtbare Symbol („invisible\_symbol“) zugewiesen:

```
probestelle.setAttribute('xlink:href' , 'invisible_symbol' );
```

Anschließend erhalten die Probestellen mit Dokumentenbezügen wieder ihre ursprünglichen Symbole zurück. Diese sind zwar nicht mehr direkt bekannt, aber über die ID des dem Element übergeordneten Layers zu ermitteln: Die Elemente des Layers mit der ID „1“ erhalten jeweils das Symbol mit der ID „MySimb.1“, die Elemente des mit der ID „2“ gekennzeichneten Layers entsprechend das Symbol mit der ID „MySimb.2“ usw.

Damit sind nach Ausführen der Funktion nur noch Probestellen mit Dokumentenbezügen sichtbar. Ergebnislose Suchanfragen können auf diese Weise vermieden und die Informationsdichte auf das Wesentliche reduziert werden.




### **3.4.3.2 Themenbezüge**

Für diese Legendeneinstellung wurde die thematische Unterscheidung nach den Kategorien Boden, Ökologie und Waldökologie getroffen. Der damit erreichte Grad an Differenzierung der Probestellensymbole sollte insgesamt zu einer vertretbaren Komplexität der Karte führen. Die Einteilung in drei Kategorien bot sich zudem aufgrund der Organisation der Dokumente im




Informationssystem von RAIBER (2001) an. Dort sind in die Dokumente in drei übergeordneten Verzeichnissen nach diesen Begriffen geordnet. Da bislang in der Datenbank fast ausschließlich Einträge für die Inhaltskategorie Boden bestehen, wurden für die Probestellen mit den IDs „12B“, „13B“, „17B“, „19BFÖ“ zu Testzwecken zusätzliche Verknüpfungen in der Datenbank vorgenommen.

Die verfolgte Ansatz besteht darin, die Zusatzinformation über die Themenbezüge mittels unterschiedlich gefärbter sekundärer Symbole in das jeweilige Grundsymbol der Probestellen zu integrieren. Durchgeführte Tests mit Quadraten, Dreiecken und Kreisen führten zu der Entscheidung, als sekundäre Symbole ausschließlich Kreisformen zu verwenden. Dies ist darauf zurückzuführen, dass Kreise eine sehr kompakte Form haben, die eine verhältnismäßig gute Erkennbarkeit der Farbe auch bei geringer Größe erlauben und eine platzsparende Integration in das primäre Symbol gewährleisten, sodass dieses noch in Form und Farbe erkennbar ist (vgl. Abb. 21).

Einige der solcherart zusammengesetzten Symbole sind in Abb. 20 für den Probestellentyp „terrestrische Untersuchungsflächen & LfU Dauerbeobachtungsflächen“ abgebildet. Braune Kreise stehen für „Boden“, grüne Kreise für „Waldökologie“ und blaue Kreise für „Ökologie“. Etwas unbefriedigend ist, dass die Bedeutung der Symbole nicht unmittelbar ersichtlich wird. Es wurde zwar versucht, mit Farbassoziationen die Bedeutung der Symbole zu unterstützen. Gleichwohl ist eine adäquate Unterscheidung der komplexen Begriffe „Ökologie“ und „Waldökologie“ unter den genannten Restriktionen der Internetkartographie nicht ohne weiteres nach farblichen Assoziationen und noch weniger in sprechender Form umsetzbar. Die Kategorie „Boden“ sollte über den braunen Farbton dagegen entsprechend assoziiert werden können.

Symbol	ID	Probestelle mit Themenbezug
	MySimb.2_s	Boden
	MySimb.2_se	Boden und Ökologie
	MySimb.2_sfe	Boden, Ökologie und Waldökologie







Themenbezüge

 Boden     Ökologie     Waldökologie

**Abb. 20: Symbole für Legendeneinstellung „Themenbezüge“**

Quelle: eigener Entwurf

Um alle Möglichkeiten des Vorkommens von Inhaltsbezügen abzudecken, müssten Symbole für alle denkbaren Kombinationen entworfen werden. Darauf soll jedoch an dieser Stelle verzichtet werden. Exemplarisch sollen aber in Abb. 21 für alle Probstellentypen die Kombinationsfähigkeit mit den drei sekundären Kreissymbolen der thematischen Bezüge und die gleichzeitige Unterscheidbarkeit der Symbole herausgestellt werden.

	terrestrische Untersuchungsflächen & LfU Dauerbeobachtungsflächen
	terrestrische Untersuchungsflächen
	aquatischer Probepunkt
	Kleingewässer
	aquatische Probestrecke
	untersuchtes Stillgewässer

**Abb. 21: Differenzierbarkeit kombinierter Probstellensymbole**

Quelle: eigener Entwurf

In der Legende des Informationssystems werden die zusätzlichen Kreissignaturen und die dazugehörenden Erläuterungen getrennt von den Symbolen

der Probestellen angezeigt (vgl. Abb. 8). Dies erschien sinnvoller, als alle zusätzlich entstehenden Symbolkombinationen einzeln anzuzeigen. Der Grund hierfür liegt darin, dass in der Legende nicht genügend Raum zur Darstellung aller theoretisch möglichen Kombinationen besteht. Wenn zudem eine Ausdehnung dieser Funktionalität auf andere Probestellen-Layer durchgeführt werden sollte, würde die Anzahl dieser Einträge noch einmal um ein Vielfaches steigen, was die Übersichtlichkeit erheblich beeinträchtigen würde.

Der interaktive Austausch der Probestellensymbole wird über eine JavaScript-Funktion ermöglicht. Diese Funktion wurde bislang nur exemplarisch für die Probestellensymbole des Layers „terrestrische Untersuchungsflächen & LfU Dauerbeobachtungsflächen“ implementiert. Auch in diesem Fall wird auf die oben genannten Einträge in der XML-Datei über das XML-DOM zurückgegriffen. Für jeden Eintrag werden die Attributbelegungen der Attribute „soil“, „forest“ und „eco“ ausgelesen. „Soil“ steht für die Inhaltskategorie Boden, „forest“ für Waldökologie und „eco“ für Ökologie. Abhängig von den ermittelten Werten wird die betreffende Probestelle durch ein dafür festgelegtes Symbol repräsentiert. Dies soll am Beispiel der Probestelle mit der ID „13B“ verdeutlicht werden. Der Eintrag im XML-Dokument lautet für diese Probestelle:

```
<docBezug id="13B" soil="true" forest="false" eco="false" count="3"/>
```

Ausgelesen wird diese Information mit den Anweisungen:

```
hasSoil = docbezug_element.getAttribute('soil' );  
hasForest = docbezug_element.getAttribute('forest' );  
hasEco = docbezug_element.getAttribute('eco' );
```

Die Zuweisung des Symbols an das die Probestelle repräsentierende <use> Element erfolgt, bedingt durch die Belegung der Variablen, in diesem Fall mit dem Symbol „MySimb.2\_s“:

```
if (hasSoil == ' true' && hasForest == ' false' && hasEco == ' false' ){  
    probestelle.setAttribute('xlink:href' , ' #MySimb.2_s' );  
}
```

Ingesamt sollte der gewählte Ansatz aufgrund der Verbindung räumlicher und thematischer Aspekte bei der Dokumentensuche und der Einfachheit der kombinierten Symbole eine zielführende Visualisierung von zusätzlichen

Informationen gewährleisten und damit die Suche nach Dokumenten vereinfachen können. Auf diese Art und Weise könnte alternativ z.B. auch eine Unterscheidung der Probestellensymbole nach Alter oder Urheber der Dokumente vorgenommen werden.

### 3.4.3.3 Anzahl der Dokumente

Der Benutzer kann sich mit Hilfe dieser Legendeneinstellung für jede Probestelle die Anzahl der damit in Beziehung stehenden Dokumente anzeigen lassen. Die dafür notwendige Information ist von einer JavaScript-Funktion dem Attribut „count“ des der jeweiligen Probestelle entsprechenden Elements <docBezug> aus der XML-Datei über das XML-DOM zu entnehmen.

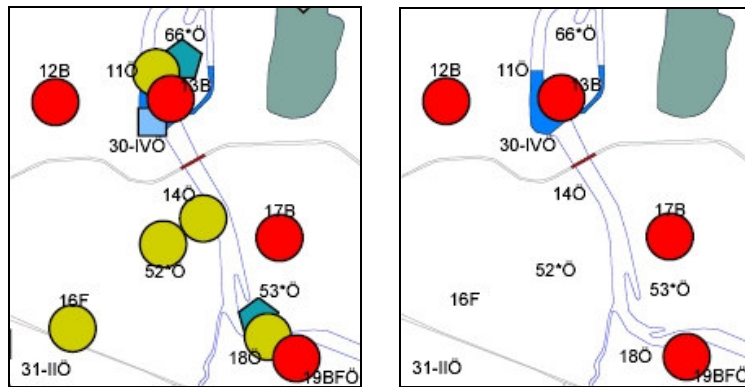
Die Problematik der Darstellung von quantitativen Informationen, die einer häufigen Änderung unterliegen können, war bereits in Kapitel 2.2 herausgestellt worden. Die dort als am geeignetsten bewertete Lösung mit der Angabe von absoluten Werten wurde durch die Kombination des jeweiligen Grundsymbols einer Probestelle mit einem Textfeld umgesetzt (vgl. Abb. 22 d) ). Die solcherart zusammengesetzten Symbole können mittels eindeutiger IDs wiederum – analog zu der Vorgehensweise bei der Legendeneinstellung „Thematische Bezüge“ – den Probestellen zugewiesen werden.

Wird der Wert 1 aus der XML-Datei ausgelesen, so wird dem <use> Element der entsprechenden Probestelle das Symbol „MySimb.2\_1“ zugewiesen:

```
if (count == ' 1' ){  
    probestelle.setAttribute('xlink:href' , ' #MySimb.2_1' );  
}
```

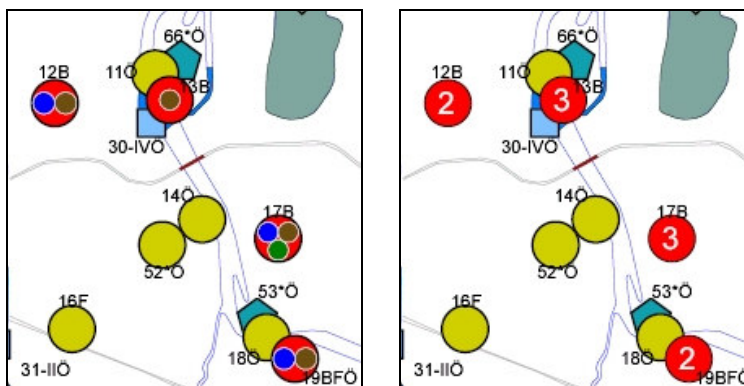
Auch diese Funktion wurde bislang nur exemplarisch für die Probestellensymbole des Layers „terrestrische Untersuchungsflächen & LfU Dauerbeobachtungsflächen“ implementiert.

Zusammenfassend zeigt Abb. 22 einen Ausschnitt der Karte für alle vier Legendeneinstellungen.



a) Standardeinstellung

b) Dokumentenbezug



c) Themenbezüge

d) Dokumentenanzahl

Abb. 22: Kartenausschnitt mit unterschiedlichen Legendeneinstellungen

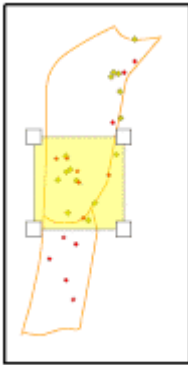
Quelle: eigener Entwurf

### 3.4.4 Pan- und Zoom-Funktionen

Die Übersichtskarte (vgl. Abb. 23) des Untersuchungsgebietes wird im Gegensatz zur eigentlichen Karte immer vollständig angezeigt. Damit soll der jeweils sichtbare Kartenausschnitt innerhalb des gesamten Untersuchungsgebietes eingeordnet werden können. Gleichzeitig soll damit ein weiterer Zugang zu Zoom- und Pan-Funktionen geschaffen werden.

Dafür dürfte es wenig sinnvoll sein, alle Layer der Karte in die Übersichtskarte zu übernehmen. Denn aufgrund ihrer wesentlich kleineren Ausmaße würde die Vielzahl an Informationen zu einer Überfrachtung führen. Deshalb enthält die Übersichtskarte nur drei Layer: den Layer mit der Dammsignatur und die Layer der terrestrischen Probestellen. Gerade der Damm hat aufgrund seiner begrenzenden Funktion der Retentionsfläche eine gute Orientierungswirkung.

Die terrestrischen Probestellen wurden gewählt, da bislang nur diese tatsächlich mit Dokumenten verknüpft sind. Diese Layer sollten daher für die Orientierung ausreichend sein und gleichzeitig die Dateigröße möglichst gering halten.



**Abb. 23: Übersichtskarte mit Auswahlrechteck**

Quelle: eigener Entwurf

Die Übersichtskarte wurde um ein gelbes Auswahlrechteck ergänzt. Es dient dazu, die Zoom- und Pan-Funktionalitäten umzusetzen. Die Idee besteht darin, die x- und y-Koordinaten sowie die Höhe und Breite des Rechtecks mit der `viewBox` des Wurzelements der Karte zu koppeln. Die Veränderungen der Koordinaten des Rechtecks im Übersichtsfenster werden registriert und in die Koordinaten der Karte umgerechnet. Werden diese Werte der `viewBox` zugewiesen, so verändert sich der Kartenausschnitt in Übereinstimmung mit der Verschiebung des Rechtecks.

Um diese Funktionalität umzusetzen, sind gewisse Anforderungen an die Ereignissteuerung zu stellen. Das Verschieben des Rechtecks soll nur mit gedrückter Maustaste möglich sein. Dafür gibt es jedoch keinen speziellen Event-Listener. Deshalb muss eine Verkettung mehrerer Event-Listener aufgebaut werden. Das Rechteck reagiert zunächst auf den Event `mousedown`. Wird also die Maustaste über dem Auswahlrechteck nach unten gedrückt, wird eine JavaScript-Funktion aufgerufen. In dieser Funktion werden die Events `mousemove` und `mouseup` bei dem Rechteck „rect“ registriert:

```
rect.addEventListener(' mousemove' , move, false);  
rect.addEventListener(' mouseup' , end, false);  
rect.addEventListener(' mouseout' , out, false);
```

Dadurch können in den dort angegebenen Funktionen (move, end bzw. out) die Koordinaten der Maus über das Event-Objekt abgefragt und dadurch die Position des Auswahlrechtecks verändert werden. Wird die Maustaste wieder losgelassen (mouseup) oder verlässt der Mauszeiger das graphische Objekt im gedrückten Zustand (mouseout), muss der Verschiebungsprozess des Rechtecks beendet werden. Die zuletzt ermittelten Koordinaten sind dann auf die viewBox zu übertragen. Dazu ist es notwendig, die Events mousemove, mouseup und mouseout beim Rechteck wieder abzumelden, um die weitere Ausführung der damit verknüpften Funktionen zu unterbinden:

```
rect.removeEventListener(' mousemove' , move, false);  
rect.removeEventListener(' mouseup' , end, false);  
rect.removeEventListener(' mouseout' , out, false);
```

Die Zoomfunktionalität ist an die Höhe und Breite des Auswahlrechtecks geknüpft. Werden diese Werte verringert und damit die entsprechenden Werte der viewBox verändert, so führt dies zu einer Vergrößerung des Maßstabes. Bei Erhöhung der Ausmaße des Rechtecks soll der Maßstab der Karte entsprechend verkleinert werden. Um diese Größenänderungen durchführen zu können, ist an allen vier Ecken des Auswahlrechtecks jeweils ein weiteres kleines Rechteck angeordnet. Diese Rechtecke lassen sich genauso wie das Auswahlrechteck selbst verschieben. Zusätzlich definiert ihre Lage auch den betreffenden Eckpunkt des Auswahlrechtecks, der daher immer mit verschoben wird. Auf diese Weise sind die Ausmaße des Rechtecks und damit auch die Ausmaße der viewBox änderbar.

Um auch nach Maßstabsänderungen dem Benutzer den korrekten Maßstab anzuzeigen, werden die im Statusfenster vorhandene textuelle Anzeige des Maßstabs sowie die Maßstabsleiste aktualisiert. Für die Berechnung des Maßstabs ist dabei das Verhältnis von Höhe bzw. Breite des Auswahlrechtecks vor der Skalierung zur Höhe bzw. Breite des Rechtecks nach der Skalierung entscheidend. Der näher bei Eins liegende der beiden Skalierungswerte teilt dann den zuvor bestehenden Maßstab und legt so den neuen Maßstab fest.

Berechnung im Zahlenbeispiel:

- *A) ursprünglicher Maßstab: 1:9.520 m*
- *B) ursprüngliche Breite, Höhe des Auswahlrechtecks: 180, 150*
- *C) neue Breite, Höhe des Auswahlrechtecks: 400, 300*
- *D) Relation Breite alt / Breite neu:  $180 / 400 = 0.45$*
- *E) Relation Höhe alt / Höhe neu:  $150 / 300 = 0.5$*
- *F) da Relation Höhe näher an 1 als Relation Breite:  
dividiere Maßstab aus A) durch 0.5*
- *G) neuer Maßstab: 1:19.040 m*

Nun sind im Adobe SVG Viewer 3.0 Zoom- und Pan-Funktionen auch ohne ein solches Überblicksfenster standardmäßig vorhanden. Vergrößerungen sind mit der linken Maustaste bei gedrückter Ctrl-Taste, Verkleinerungen bei zusätzlich gedrückter Shift-Taste möglich. Die Pan-Funktion wird beim Drücken der Alt-Taste aktiviert. Durch mehrfaches Durchführen von Pan und Zoom kann jedoch die Orientierung auf der Karte sehr schnell verloren gehen, was die Implementierung einer solchen Übersichtskarte durchaus rechtfertigt.

Die Ausführung der SVG Pan- und Zoomfunktionen in der Karte sollten sich jedoch auch auf das Auswahlrechteck in der Übersichtskarte entsprechend auswirken. Dies hat aber bei der Implementierung erhebliche Probleme bereitet. Denn das Pannen und Zoomen über das Plugin führt nicht zu einer Veränderung der Werte der viewBox. Daher ist es nicht möglich, diese Veränderungen in umgekehrter Weise von der viewBox auf das Auswahlrechteck zu übertragen. Die Versuche, diese Veränderungen auf eine andere Art und Weise zu registrieren und zu übertragen führten sehr schnell zu Problemen und Ungereimtheiten, die mit der vorhandenen Literatur nicht zu lösen waren.

Ein potenzieller Zugang besteht über das Attribut „transform“ auf der Ebene des <svg> Elements. Nach der Verwendung der Pan-Funktion kann festgestellt werden, um wie viel Pixel die Verschiebung der Karte auf dem Bildschirm durchgeführt wurde. Auch der Maßstab kann grundsätzlich abgefragt werden. Anders als von EISENBERG (2002) angegeben, war jedoch festzustellen, dass sich zusätzlich zu den Maßstabswerten nach dem Zoom auch die Verschiebungswerte des „transform“- Attributs verändert hatten. Die mehrfache

Wiederholung von Zoom und Pan macht es daher umso schwerer, diese Veränderungen korrekt zu registrieren.

Daher sind die Pan- und Zoom-Funktionen des SVG Viewers nicht mit den im Rahmen dieser Arbeit implementierten Funktionen gleichzeitig zu benutzen und daher wechselseitig auszuschließen. Beim Laden des Systems sind zunächst nur die neu implementierten Pan- und Zoom-Funktionen aktiviert. Es ist möglich, mit dem Button „SVG ZoomPan“ der Toolbox (vgl. Abb. 18) auf die Funktionen des Plugins umzuschalten. Dann wird jedoch zur Vermeidung von Inkonsistenzen das Auswahlrechteck in der Übersichtskarte entfernt. Rückgängig gemacht werden kann dies nur durch erneutes Laden des Systems.

### 3.4.5 Speichern von Benutzereinstellungen

Benutzer, die sich über ein Passwort in das System eingeloggt haben, sollen die in der Karte vorgenommenen Änderungen abspeichern können, um beim nächsten Laden des Systems wieder die gewünschten Einstellungen vorzufinden. Mögliche Umsetzungsvarianten wurden in Kapitel 2.3.5.2 diskutiert. Für die Implementierung wurde dem Ansatz der Vorzug gegeben, welcher eine separate Speicherung der Benutzereinstellungen vornimmt und diese Daten beim Start des Systems zur automatischen Anpassung der Karte verwendet. Dies ist damit zu begründen, dass in diesem Fall die für die interaktiven Funktionen geschriebenen JavaScript-Funktionen vollständig oder zumindest in Teilen wieder verwendet werden können. Folgende Änderungen können auf diese Weise gespeichert werden:

- Sichtbarkeit der Layer
- Kartenausschnitt und Maßstab der Karte
- Wahl der Legendeneinstellung

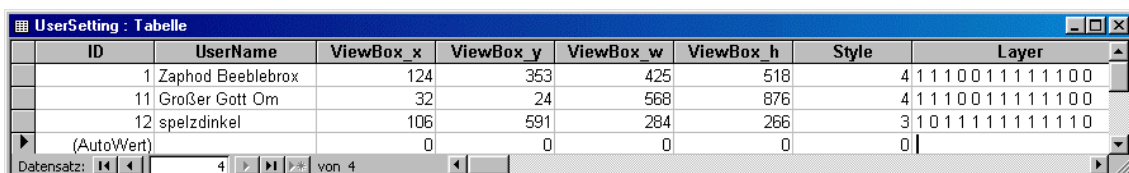
Für die Initiierung des Speichervorgangs steht in der Toolbox der HTML-Button „Save settings“ zur Verfügung (vgl. Abb. 18). Bei Anklicken des Buttons wird eine JavaScript-Funktion aufgerufen, welche die benötigten Informationen ermittelt. Über das <svg> Element wird auf das Attribut „viewBox“ zugegriffen und die Information in die Einzelbestandteile x-Koordinate, y-Koordinate, Höhe

und Breite zerlegt. Mit diesen Werten kann der aktuelle Kartenausschnitt eindeutig festgehalten werden. Dies trifft allerdings – aus den in Kapitel 3.4.4 thematisierten Problemen – nicht auf Skalierungen und Verschiebungen zu, die außerhalb des Auswahlrechtecks durchgeführt wurden. Dabei handelt es sich zugleich um die einzige Benutzeraktion, die nicht gespeichert werden kann.

Die Sichtbarkeit der Layer kann einfach über das „visibility“-Attribut der Schaltflächen in der Legende ermittelt werden (vgl. Abb. 17). Für nicht sichtbare Layer wird dann eine Null, für sichtbare eine Eins gespeichert. Diese Informationen werden durch Leerzeichen getrennt zu einer Zeichenkette verknüpft und in der Variablen *layers* gespeichert. Das Ergebnis kann beispielsweise diese Ausprägung annehmen: 1 1 0 1 1 1 0 0 1 1 1 1 0.

Die Information über die Legendeneinstellungen wird schließlich aus dem internen Wert des Pulldown-Menüs gewonnen und in der Variablen *style* festgehalten. Dann werden alle benötigten Informationen zusammen mit dem Benutzernamen durch Kommata getrennt zu einer Zeichenkette verbunden und an die Adresse eines Perlskripts angehängt:

```
query = x + "," + y + "," + width + "," + height + "," + style + "," + layers + "," + user;
doc = parent.open("./cgi-bin/settings.pl?" + query);
```



ID	UserName	ViewBox_x	ViewBox_y	ViewBox_w	ViewBox_h	Style	Layer
1	Zaphod Beeblebrox	124	353	425	518	4	1 1 1 0 0 1 1 1 1 1 1 0 0
11	Großer Gott Om	32	24	568	876	4	1 1 1 0 0 1 1 1 1 1 1 0 0
12	spelzdinkel	106	591	284	266	3	1 0 1 1 1 1 1 1 1 1 1 1 0
(AutoWert)		0	0	0	0	0	

**Abb. 24: DB-Tabelle „UserSetting“**

Quelle: eigener Entwurf

Die Adresse wird an den Server übermittelt, der dann das Skript ausführt. Das Skript zerlegt die Information wieder in die einzelnen Bestandteile und trägt diese in die Tabelle „UserSetting“ der Datenbank (vgl. Abb. 24) ein. Der dafür verwendete SQL-Befehl ist davon abhängig, ob für einen Benutzer bereits ein Eintrag in der Datenbank existiert. Bei erstmaligem Eintrag ist ein INSERT Befehl, bei späteren Einträgen ein UPDATE Befehl einzusetzen. Deshalb muss das Perlskript die Datenbank zuerst auf bestehende Einträge testen und dann

den entsprechenden SQL-Befehl an die Datenbank richten. Der Benutzer erhält dann eine Rückmeldung über den Erfolg der durchgeführten Speicheraktion:



**Abb. 25: DB-Rückmeldung nach erfolgreichem Eintrag**

Quelle: eigener Entwurf

Beim nächsten Laden des Informationssystems werden diese Informationen aus der Tabelle „UserSetting“ der Metadatenbank gelesen. Dazu wird in einem Perlskript folgender SQL-Befehl ausgeführt:

```
„SELECT ViewBox_x, ViewBox_y, ViewBox_h, ViewBox_w, Style, Layer
FROM UserSetting WHERE UserName = $user“
```

Der Benutzername, der in der Variablen *\$user* gespeichert ist, wird zuvor aus der Umgebungsvariablen ermittelt:

```
$user = $ENV{"REMOTE_USER"};
```

Die Ergebnismenge wird in einem Array gespeichert. Über die Elemente des Arrays wird dann iteriert, die ausgelesenen Werte durch Tags eingefasst (z.B. `<x>$array[0]</x>`) und in eine XML-Datei geschrieben. Die XML-Datei kann folgende Ausprägung annehmen:

```
<settings>
  <x>40</x>
  <y>80</y>
  <height>200</height>
  <width>300</width>
  <style>1</style>
  <layer>0 0 1 1 1 1 0 1 0 0 1 1 0</layer>
  <user>spelzdinkel</user>
</settings>
```

Auch die hierfür verwendeten Auszeichnungsbefehle sind frei gewählt und nehmen auf keine DTD Bezug. Für die Elemente wurden sprechende Namen gewählt, weshalb eine Erklärung ausbleiben kann.

Beim Start des Informationssystems werden die Werte der XML-Datei von einer JavaScript-Funktion ausgelesen und in Variablen gespeichert. Die aus den Elementen `<x>`, `<y>`, `<height>` und `<width>` gewonnenen Informationen werden der `viewBox` des Wurzelements der Karte bzw. den betreffenden Attributen des Auswahlrechtecks der Übersichtskarte zugewiesen. Die Layer der Karte (bzw. die „visibility“-Attribute der entsprechenden `<g>` Elemente) erhalten die für sie im Element `<layer>` festgehaltenen Sichtbarkeitswerte zugeordnet. Der für die Legendeneinstellungen im Element `<style>` gespeicherte Wert löst dieselbe JavaScript-Funktion aus, die auch bei dem entsprechenden Klick auf das Pulldown-Menü aufgerufen wird und führt daher dieselben Änderungen in Karte und Legende durch.

Fehlt nur einer der für die Benutzereinstellungen gespeicherten Werte, ist davon auszugehen, dass der Speichervorgang fehlerhaft war. In diesem Fall wird das System in der Standardeinstellung geladen. Wurden die Einstellungen dagegen korrekt gespeichert, hat der Benutzer wieder den zuletzt gespeicherten Arbeitsstand vor sich und kann sich sofort in der für ihn angepassten Umgebung des Web-Informationssystems weiterbewegen.

## **3.5 Aktualisierung des Systems**

### **3.5.1 Hinzufügen von Probestellen**

Für das Hinzufügen neuer Probestellen sind zunächst ihre Gauß-Krüger-Koordinaten zu erheben. Dann muss im SVG-Dokument der Karte innerhalb des Gruppierungselements `<g>`, das bereits Probestellen desselben Typs zu einem Layer zusammenfasst, ein neues `<use>` Element definiert werden. Die erhobenen Koordinaten werden dem „x“- bzw. „y“-Attribut zugewiesen. Dabei ist zu beachten, dass dem y-Wert ein negatives Vorzeichen voranzustellen ist. Schließlich muss dem Element noch dasselbe Symbol wie den anderen Elementen desselben Layers zugewiesen werden. Wurden bereits Einträge in

der Datenbank angelegt, die mit dieser Probestelle in Zusammenhang stehen, so müssen zusätzlich Event-Listener mit der Probestellen-ID als Parameter implementiert werden, die mit der in der Datenbank angegebenen ID konform gehen. Diese Angaben führen zu Einträgen im SVG-Dokument in folgender Form:

```
<use id="34-1Ö" onmouseover="ShowID('34-1Ö')"  
onclick="startQuery('34-1Ö')" onmouseout="HideID('34-1Ö')"  
x="3409444.25" y="-5370011.50" width="120.89" height="120.89"  
xlink:href="#MySimb.3"/>
```

### 3.5.2 Hinzufügen von Dokumenten

Neue Dokumente sollten zunächst in die Tabelle „Liste“ der Metadatenbank eingetragen werden. Dort ist für jedes neue Dokument zwingend eine eindeutige Bezeichnung (Eintrag\_ID) sowie Titel und Adresse des Dokuments anzugeben. Sofern das betreffende Dokument Bezüge zu Probestellen aufweist, muss in der Tabelle „Objekte“ die ID des Dokuments jeweils im Record der damit in Beziehung stehenden Probestelle eingetragen werden (vgl. Abb. 13).

### 3.5.3 Inkonsistente Zustände

Es wurde ein größtmöglicher Wert darauf gelegt, eine schnelle und automatische Aktualisierung des Systems zu gewährleisten. Deshalb wird bei jeder Dokumentensuche unmittelbar eine Abfrage der Metadatenbank durchgeführt. Veränderungen im Dokumentenbestand sind daher direkt nach einer Aktualisierung der Datenbank für den Benutzer zugänglich.

Die in der Karte sichtbaren Informationen über die Dokumente können dagegen nicht in gleicher Weise aktualisiert werden. Diese Informationen werden immer nur beim Laden des Systems aus der Datenbank abgefragt. Eine automatische Aktualisierung der Karte, die direkt an die Veränderungen in der Datenbank gekoppelt ist, dürfte kaum zu realisieren sein. Dies ist darauf zurückzuführen, dass der Server sonst an alle Clients, die in diesem Moment das

Informationssystem geladen haben, ohne deren Anfrage die aktualisierten Daten verschicken und diese auf die Karte übertragen müsste.

Wird z.B. während der Benutzung des Systems ein Dokument für eine Probestelle hinzugefügt, die bislang noch keinen Dokumentenbezug aufweisen konnte, so tritt der Fall ein, dass diese Probestelle auch weiterhin auf der Karte als Probestelle ohne Dokumentenbezug visualisiert wird, obwohl darüber bereits auf ein Dokument zugegriffen werden kann.

Eine Vermeidung dieser Inkonsistenz wäre dann möglich, wenn die Aktualität der Informationen über die verfügbaren Dokumente auf den Stand beim Start des Systems begrenzt werden würde. Dann wäre aber eine Form der Inkonsistenz möglich, die an dieser Stelle als schwerwiegender bewertet wird: die Bereitstellung von Links auf Dokumente, die nicht mehr existieren.

Somit kann das System nicht völlig frei von Inkonsistenzen sein. Zur Vermeidung der damit verbundenen Probleme kann aber bei Bedarf die Aktualisierung der Karte über den Reload-Button in der Toolbar des Browsers durchgeführt werden.

## 4 Überprüfung der Hypothesen

### These 1:

**Eine offene, herstellerunabhängige Architektur genügt den an das Informationssystem gestellten Anforderungen am besten. Dies ist v.a. damit zu begründen, dass auf diese Weise ein plattformunabhängiger Zugriff auf die Dokumentenbestände und eine Minimierung der finanziellen Aufwendungen in der Entwicklung und Wartung des Systems sowie für die potenziellen Benutzer erreicht werden kann.**

Diese These konnte weitestgehend bestätigt werden. Die Entwicklung des Informationssystems wurde clientseitig ausschließlich mit offener, frei verfügbarer Internettechnologie bewältigt. Die Karte als Zugangsmedium zu den Dokumentenbeständen konnte mit dem offenen Standard SVG erstellt werden. Darüber hinaus war es auch möglich, die Dokumente selbst in die offenen, textbasierten Formate HTML bzw. SVG zu konvertieren.

Somit konnte die Dokumentenzugriffskomponente des ursprünglichen Systems, das die proprietären Formate MS Word, MS Excel und ESRI-Shapefiles verwendete und auf der proprietären GIS-Software ArcView basierte, in ein plattformunabhängiges, frei zugängliches und damit auch kostenneutrales System transferiert werden. Einzig bei der unverändert übernommenen Metadatenbank handelt es sich mit MS Access um eine proprietäre Lösung, welche aber über eine standardisierte Schnittstelle zugänglich ist.

Insgesamt konnte auf diese Weise das geforderte Informationssystem mit einfachen Mitteln (Texteditoren) und ohne finanzielle Aufwendungen entwickelt werden. Dasselbe gilt auch im Hinblick auf spätere Aktualisierungen des Systems.

**These 2:**

**SVG eignet sich, ergänzt um weitere Web-Technologien wie CGI-Skripte und JavaScript, für die Konzipierung interaktiver Karten. Der für dieses Informationssystem notwendige Grad an Interaktivität ist daher umsetzbar.**

Auch diese These konnte bestätigt werden. Das Durchführen von Datenbankabfragen über die Symbole der Karte, die Funktionen zur Veränderung des Maßstabs und der interaktiven Anpassung des Karteninhalts sowie der Speicherung derartiger Änderungen konnten realisiert werden. Damit waren alle geforderten interaktiven Funktionen umsetzbar und es kann gesagt werden, dass sich die Kombination von SVG, JavaScript und CGI dazu eignet, einen zielführenden räumlichen Zugang auf räumlich verortete Dokumentenbestände über das Internet zu gewährleisten.

**These 3:**

**Für die Erreichung der formulierten Ziele sind Vektor-Formate besser geeignet als Raster-Formate. Dies ist darauf zurückzuführen, dass mit Vektor-Formaten einerseits mehr Interaktivität und andererseits eine bessere Performance bei gleichzeitig hoher Auflösung zu realisieren ist.**

Diese These konnte uneingeschränkt bestätigt werden. Die Dateien, die beim Start des Systems zum Client transferiert werden müssen, haben eine Größe von nur 93,7 KB. Dabei kommt der Performance zugute, dass der überwiegende Teil der interaktiven Funktionen clientseitig auf Basis der beim Start des Systems transferierten Informationen ausgeführt werden kann und demzufolge kaum weitere Datentransfers notwendig sind. Dies ist in dieser Form nur mit Vektordaten möglich. Der entscheidende Wert von Vektordaten liegt für derartige Anwendungen also in der relativ einfachen Realisierung interaktiver Funktionen bei gleichzeitiger Minimierung des Kommunikationsaufwandes zwischen Client und Server.

**These 4:**

**Mit SVG können Karten erstellt werden, die eine einfache, intuitive Erschließung von Dokumenten gewährleisten. Zu begründen ist dies damit, dass SVG allgemeinen wie auch internetspezifischen kartographischen Ansprüchen gerecht wird.**

Diese These muss etwas relativiert werden. Bei der Entwicklung des Informationssystems wurde die Erkenntnis gewonnen, dass es äußerst schwierig ist, ein Kartenbild zu produzieren, bei dem Punktsignaturen unterschiedlicher Größe sich nicht gegenseitig überdecken, gleichzeitig möglichst viel Aussage enthalten und dennoch äußerst einfach gehalten sind. Ergänzt um weitere internetspezifische Mapping-Probleme wie unterschiedliche Farbeinstellungen und Auflösungen kann eine einfache und intuitive Erschließung nicht generell garantiert werden. Dies gilt v.a. deshalb, weil die gegenseitige Überdeckung von Punktsignaturen eines Layers nicht generell auszuschließen ist und solcherart den Zugang auf damit verknüpfte Dokumente verhindern kann.

Die für die Probestellen entworfenen einfachen primären wie sekundären Kreissymbole sind nicht sprechend und nicht immer eindeutig assoziativ in ihrer Farbgebung, was einen einfachen und intuitiven Zugang eher behindert, dafür aber eine Kombinierbarkeit der Symbole ermöglicht. Letzteres wird für diese Arbeit aufgrund des Mehrwerts zusätzlich visualisierbarer Informationen und der damit verbundenen schneller einschränkbarer Suche nach Dokumenten als wichtiger erachtet. Überdies sind die angesprochenen Probleme nicht auf die Spezifikation von SVG sondern auf Probleme der Internetkartographie im Allgemeinen zurückzuführen.

## 5 Fazit

Im Rahmen dieser Master Thesis konnte gezeigt werden, dass ein räumlicher und thematischer Zugang zu Dokumentenbeständen über interaktive, SVG-basierte Webkarten möglich ist. Gleichzeitig wurde der außerordentlich hohe Wert offener Standards dargelegt. Diese eröffnen eine langfristige Perspektive sowie einen freien Zugang über Systemgrenzen hinweg.

Der Beitrag dieser Arbeit für das Dokumentenmanagement liegt in einem sehr eng gefassten Bereich. Es betrifft „nur“ den Zugriff auf die Dokumente, nicht die weiteren Prozesse, die im Lebenszyklus eines Dokuments von Interesse sind. In diesem eng umgrenzten Bereich konnte aber für eine breite Öffentlichkeit auf einer kartographischen Grundlage eine räumliche Zugriffskomponente für räumlich verortete Umweltdokumentenbestände geschaffen werden. Die Vereinfachung des Suchprozesses durch die Integration zusätzlicher Informationen über die Dokumente in die Punktsymbole und die getrennte Visualisierung unterschiedlicher Informationen sollte auch für räumlich orientierte Suchprozesse in anderen Dokumentenbeständen zielführend sein.

Die schwerpunktmäßig clientseitige Umsetzung interaktiver Funktionen und die Verwendung eines Vektordatenformats gewährleisteten eine gute Performance des Systems bei einem hohen Grad an Interaktivität. Diese Merkmale sollten unter Verwendung dieser Architektur auch für andere webbasierte Informationssysteme gelten, die einen Datenzugriff über punktförmige graphische Objekte realisieren. Dagegen ist dieser Ansatz für Systeme, die einen solchen Zugriff über flächen- oder linienhafte Signaturen anstreben nicht ohne weiteres übertragbar. In einem solchen Fall ist tendenziell mit einem zu großen Dateiumfang zu rechnen, weshalb die Eignung von SVG dann etwas zu relativieren ist.

Auch die Art der Realisierung der Datenbankzugriffe ist nicht uneingeschränkt übertragbar. Im Falle der Notwendigkeit einer garantierten und dauerhaften Konsistenz ist dieser Ansatz aufgrund des potenziellen Auftretens vorübergehender Inkonsistenzen zwischen Karteninhalt und Dokumenten (vgl. Kapitel 3.5.3) zu modifizieren.

Ein weiteres Anwendungsfeld der implementierten Architektur könnte aber über das klassische Dokumentenmanagement hinaus im räumlichen Zugriff aktueller dynamischer Daten zu finden sein. Ein Beispiel hierfür ist die Bereitstellung interaktiver Wettervorhersagekarten. Dort könnte abhängig von den aktuellen Wetterdaten ein entsprechendes Symbol (z.B. Wolken, Sonne) eingeblendet werden, das den Zugriff auf detaillierte und hochaktuelle Informationen ermöglicht.

Neben dem Aufzeigen von Einsatzmöglichkeiten von SVG konnte im Rahmen dieser Master Thesis eine Abgrenzung dafür erbracht werden, wann GML SVG ergänzen sollte und wann SVG ohne den Einsatz von GML bereits ausreichend ist. Für ein System – wie das im Rahmen dieser Arbeit entwickelte – mit weitgehend statischen geometrischen Informationen, die nur einer Datenquelle entstammen, ist die alleinige Verwendung von SVG als hinreichend zu bewerten. Eine verteilte Erfassung und Organisation geometrischer Daten und Attributdaten, die Umsetzung von Dienstleistungsketten und die Notwendigkeit der Datenausgabe auf unterschiedlichen Medien spricht dagegen für den Einsatz von GML.

In dieser Arbeit wurde zudem versucht, den Defiziten der Webkartographie mit zusätzlichen interaktiven Funktionen zu begegnen. Als zentrales, noch ungelöstes Problem stellte sich aber die mögliche gegenseitige Überdeckung von Punktsignaturen heraus. Eine vollständige Überdeckung ist in diesem Kontext deshalb nicht akzeptabel, weil dann über die überdeckte Signatur keine Dokumente abgefragt werden können. Konnte durch die Aufspaltung des Probestellen-Layers in mehrere Layer und durch die Schaffung einer Möglichkeit für das Ein- und Ausblenden von Layern diese Problematik teilweise gelöst werden, so ist die Gefahr der gegenseitigen Überdeckung der Symbole eines Layers – auch wenn dies in diesem Fall nicht aufgetreten ist – grundsätzlich gegeben. Wenn eine Verschiebung der Punktsignaturen aufgrund der Notwendigkeit einer hohen Lagegenauigkeit nicht in Betracht zu ziehen ist oder aufgrund einer sehr dichten Ballung zahlreicher Punktobjekte kaum möglich ist, müssen andere Ansätze gefunden werden, um den Zugriff auf überdeckte Signaturen zu gewährleisten. Dies wird als zentrale weiterführende Forschungsfrage erachtet. Lösungsansätze könnten darin zu finden sein, eine

Signatur, welche eine andere vollständig überdeckt, als solche zu kennzeichnen und darüber wahlweise (z.B. über ein Kontextmenü) den Zugriff über die im Vordergrund stehende oder die davon überdeckte Signatur zu gewähren.

In einem weiteren Schritt sind aber v.a. automatisierte Lösungen zu suchen und umzusetzen, die eine aufwendige Nachbearbeitung der Karte ersetzen. Ein solcher Ansatz ist besonders dann von Interesse, wenn Punktsignaturen über ihre Symbolgröße quantitative Aussagen vermitteln sollen. Sind diese, wie im vorliegenden Fall, hochgradig variabel, sollte eine Überdeckung vom System idealerweise automatisch erkannt und entsprechend berücksichtigt werden. Eine derartige Funktion wäre sicherlich als Gewinn für die räumliche Erschließung von Informationen über Internetkarten zu werten.

## Literaturverzeichnis

ACKLAND, R. und S. COX (2001): Markup Mapping. [http://www.gisuser.com.au/GU/content/2001/GU47/gu47\\_feature/gu47\\_feature\\_2.html](http://www.gisuser.com.au/GU/content/2001/GU47/gu47_feature/gu47_feature_2.html),  
Zugriff am 08. 02. 2003.

ADOBE (2003): Adobe SVG Viewer Download Area.  
<http://www.adobe.com/svg/viewer/install/main.html>, Zugriff 7.02.2003.

ASCHE, H. (2001): Kartographische Informationsverarbeitung in Datennetzen –  
Prinzipien, Produkte, Perspektiven. In: Herrmann, C., H. Asche (Hrsg.):  
Web.Mapping 1. Raumbezogene Information und Kommunikation im  
Internet. Heidelberg.

BILL, R. (1999): Grundlagen der Geo-Informationssysteme. Band 1: Hardware,  
Software und Daten. 4. Aufl. Heidelberg.

BOWEN R., ET. AL. (1997): CGI Programming with Perl, Visual Basic and C.  
Indianapolis.

BRAY, T. ET. AL. (1998): Extensible Markup Language 1.0,  
<http://www.w3.org/TR/1998/REC-xml-19980210>, Zugriff am 25.02.2003.

COX, S. ET. AL.(2002): OpenGIS Geography Markup Language (GML)  
Implementation Specification, version 2.1.2.  
<http://www.opengis.net/gml/01-029/GML2.html>, Zugriff am 07.02.2003.

DICKMANN, F. (2001): Web-Mapping und Web-GIS. Braunschweig (= Das  
Geographische Seminar).

DISY (2003): disy GISterm.  
[http://www.disy.net/de/Produkte/disy\\_GISterm/index.html](http://www.disy.net/de/Produkte/disy_GISterm/index.html),  
Zugriff am 07. Februar 2003.

DOCUMENT MANAGEMENT AVENUE (2001): Glossary.  
<http://www.documentmanagement.org.uk/pages/glossary.htm>,  
Zugriff am 17.02.2003

- EISENBERG, J. (2002): SVG Essentials. Sebastopol.
- ESRI (2000): Customizing ArcIMS HTML Viewer.  
[http://arconline.esri.com/arconline/documentation/ims\\_/HTMLViewer1.pdf](http://arconline.esri.com/arconline/documentation/ims_/HTMLViewer1.pdf)  
Zugriff am 22. November 2002.
- FERRAILOLO, J. ET.AL. (2001): Scalable Vector Graphics (SVG) 1.0 Specification,  
<http://www.w3.org/TR/SVG/>, Zugriff am 21.05.2002.
- FERRAILOLO, J. ET.AL. (2003): Scalable Vector Graphics (SVG) 1.1 Specification,  
<http://www.w3.org/TR/SVG11/>, Zugriff am 15.02.2003.
- GROLIG, B., A. SCHENK, D. WALDIK (2001): Stand und Tendenzen zur  
Visualisierung von Geoinformationen im WWW. In: Herrmann, C., H.  
Asche (Hrsg.): Web.Mapping 1. Raumbezogene Information und  
Kommunikation im Internet. Heidelberg.
- JOHANSSON und SIIRILÄ (2001): Internet Vector Map Server. An Implementation  
of XML-technologies for Geographic Information, Transformation and  
Vector Graphics. <http://www.gector.org/projects/ivms>,  
Zugriff am 06.02.2003.
- KRAAK, M.-J. (2001): Webmapping - Webdesign. In: Herrmann, C., H.  
Asche (Hrsg.): Web.Mapping 1. Raumbezogene Information und  
Kommunikation im Internet. Heidelberg.
- KRÜGER, G. (2002): Handbuch der Java-Programmierung. 3. Aufl. München.
- LAKE, R. (2001): Top 10 Benefits of Using GML.  
<http://www.spatialnews.com/features/gml/topten.html>,  
Zugriff am 08. 02. 2003.
- MACROMEDIA (2002): Flash Player for Developers and Publishers.  
[http://www.macromedia.com/software/flash/survey/whitepaper\\_sep02.pdf](http://www.macromedia.com/software/flash/survey/whitepaper_sep02.pdf)  
Zugriff am 22. November 2002.

- MINTERT, S. (1999): Einführung in die Extensible Markup Language (XML).  
<http://computerphilologie.uni-muenchen.de/jg99/xml.htm>,  
Zugriff am 21.04.1999.
- MOZILLA.ORG (2002): <http://www.mozilla.org/projects/svg/>,  
Zugriff am 25.02.2003.
- MÜNZ, S. (1998): SELFHTML. <http://www.teamone.de/selfaktuell/>,  
Zugriff am 20. 10. 1999.
- NEUMANN, A. (2000): Vienna – Social patterns and structures.  
[www.karto.ethz.ch/~an/cartography/vienna](http://www.karto.ethz.ch/~an/cartography/vienna), Zugriff am 08.09.2002.
- NEUMANN, A. (2002): Comparing .SWF (Shockwave Flash) and .SVG (Scalable Vector Graphics) file format specifications.  
[http://www.carto.net/papers/svg/comparison\\_flash\\_svg.html](http://www.carto.net/papers/svg/comparison_flash_svg.html),  
Zugriff am 17.02.2003.
- OGC (2000): The OpenGIS Abstract Specification. Topic 1: Feature Geometry (ISO 19107 Spatial Schema), Version 5.  
<http://www.opengis.org/public/abstract/01-101.pdf>,  
Zugriff am 01.03.2003
- OGC (2003): OGC Glossary. <http://ogc.opengis.org/cgi-bin/displayGlossary.pl>,  
Zugriff am 07.02.2003.
- OpenSWF(2003): A Concise Guide to the SWF File Format.  
[http://www.openswf.org/spec/toc1\\_4.html](http://www.openswf.org/spec/toc1_4.html), Zugriff am 17.02.2003.
- PC WELT (2003): Internet Explorer 5. Keine Umlaute in URLs.  
<http://www.pcwelt.de/tipps/online/browser/4021/>, Zugriff am 25.02.2003.
- PITTI, D., C. JESSEE, S. RAMSAY (2002): Multiple Architectures and Multiple Media: The Salem Witch Trials and Boston' s Back Bay Fens Projects.  
<http://www.uni-tuebingen.de/cgi-bin/abs/abs?propid=55>,  
Zugriff am 25.02.2003.

- RAIBER, A. (2001): Archivierung und Analyse von geoökologischen Untersuchungsergebnissen mit Hilfe eines GIS und Erhebung der Daten mittels DBPS. Unveröffentlichte Diplomarbeit am Fachbereich Vermessung und Geoinformatik, Fachhochschule Stuttgart.
- SALATHÉ, M. (2001): SVG Scalable Vector Graphics. München.
- SCHWARTZ, S., K. TOCHTERMANN (2000): Geographische Suchkomponenten für Informationsportale. In: Strobl, J., T. Blaschke, G. Griesebner (Hrsg.): Angewandte Geographische Informationsverarbeitung XII. Beiträge zum AGIT-Symposium Salzburg 2000. Heidelberg.
- STROBL, J. (2001): Online-GIS – das WWW als GIS-Plattform. In: Herrmann, C., H. Asche (Hrsg.): Web.Mapping 1. Raumbezogene Information und Kommunikation im Internet. Heidelberg.
- WIELAGE, G., O. POTT (2001): XML - new technology.  
<http://www.mut.com/media/buecher/xmlInt/data/start.htm>, Zugriff am 07.02.2003.
- WINTER, A. M. (2000): Internetkartographie mit SVG. Prototyp für einen thematischen Atlas. [http://www.carto.net/andre.mw/projects/atlas/internetkarto\\_svg\\_atlas\\_001206.pdf](http://www.carto.net/andre.mw/projects/atlas/internetkarto_svg_atlas_001206.pdf), Zugriff am 21.05.2002.
- WORBOYS, M. (1995): GIS – A Computing Perspective. London.
- ZEH, F.-P. (2002): Multimedialstandards im Internet. Thema 6: Vektorbasierte Bildformate (SVG und Flash). [http://www-ra.informatik.uni-tuebingen.de/lehre/ws02/pro\\_internet\\_ausarbeitung/proseminar\\_zeh\\_ws02.pdf](http://www-ra.informatik.uni-tuebingen.de/lehre/ws02/pro_internet_ausarbeitung/proseminar_zeh_ws02.pdf), Zugriff am 05.02.2003

## URL-Verzeichnis

- URL 1: <http://mapserver.gis.umn.edu> (Mapserver der University of Minnesota)
- URL 2: Flash Autorenprogramm KoolMoves <http://www.koolmoves.com/>
- URL 3: Flash Autorenprogramm 3D Flash Animator <http://www.3dfa.com/>
- URL 4: <http://www.stadtplan.net/> (Flash Stadtpläne)
- URL 5: <http://www.idle.nl/weer/> (Flash Wetterkarte)
- URL 6: [http://www.visitwiltshire.co.uk/pages/wiltshire\\_fs\\_about.htm?inner/wiltshiremap.asp~inner](http://www.visitwiltshire.co.uk/pages/wiltshire_fs_about.htm?inner/wiltshiremap.asp~inner) (Touristische Karte von Wiltshire)
- URL 7: Swift Generator zur dynamischen Erstellung von Flash-Dateien  
<http://www.swift-tools.com/swift-generator.html>
- URL 8: <http://www.adobe.com/svg/viewer/install/main.html> (Adobe SVG Viewer)
- URL 9: <http://home.hiwaay.net/~crispen/src/> (Komprimierungsprogramm win-gz)
- URL 10: <http://www.svgmapper.com> (ArcView Extension SVG Mapper)
- URL 11: <http://httpd.apache.org> (Apache Webserver)
- URL 12: <http://www.activestate.com/PPMPackages/zips/6xx-builds-only/>  
(Perl Module)