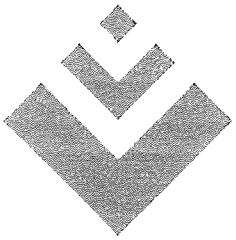
Forschungszentrum Karlsruhe
Technik und Umwelt

Wissenschaftliche Berichte
FZKA 5900



UIS

Baden-Württemberg

Projekt GLOBUS Konsolidierung der neuen Systemarchitektur und Entwicklung erster Produktionssysteme für globale Umweltsachdaten im Umweltinformationssystem Baden-Württemberg Phase III 1996

R. Mayer-Föll, A. Jaeschke (Hrsg.)

Ministerium für Umwelt und Verkehr Baden-Württemberg

Forschungszentrum Karlsruhe
Institut für Angewandte Informatik

Dezember 1996

Forschungszentrum Karlsruhe
Technik und Umwelt

Wissenschaftliche Berichte
FZKA 5900

Projekt GLOBUS

Konsolidierung der neuen Systemarchitektur und Entwicklung erster Produktionssysteme für globale Umweltsachdaten im Umweltinformationssystem Baden-Württemberg Phase III 1996

R. Mayer-Föll, A. Jaeschke (Hrsg.)

Ministerium für Umwelt und Verkehr Baden-Württemberg

Forschungszentrum Karlsruhe
Institut für Angewandte Informatik

Forschungszentrum Karlsruhe GmbH, Karlsruhe
1996

Druck und Vertrieb

Forschungszentrum Karlsruhe GmbH
Postfach 3640, 76021 Karlsruhe

ISSN 0947-8620

Projekt GLOBUS

Konsolidierung der neuen Systemarchitektur und Entwicklung erster Produktionssysteme für globale Umweltsachdaten im Umweltinformationssystem Baden-Württemberg

Phase III - 1996

Projektträger:

Ministerium für Umwelt und Verkehr Baden-Württemberg;
I. Henning, R. Mayer-Föll

Landesanstalt für Umweltschutz Baden-Württemberg (LfU);
M. Müller, E. Schmid, H. Spandl

Projektpartner:

Forschungszentrum Informatik
an der Universität Karlsruhe (FZI);
A. Koschel, R. Kramer, R. Nikolai, C. Rolker

Forschungsinstitut für anwendungsorientierte Wissensverarbeitung
an der Universität Ulm (FAW);
W. F. Riekert, G. Wiest

Forschungszentrum Karlsruhe - Technik und Umwelt (FZK);
W. Geiger, A. Jaeschke, R. Weidemann

Institut für Kernenergetik und Energiesysteme
der Universität Stuttgart (IKE);
F. Schmidt

Institut für Photogrammetrie und Fernerkundung
der Universität Karlsruhe (IPF);
J. Wiesel

Vorwort

Das Ministerium für Umwelt und Verkehr Baden-Württemberg baut gemeinsam mit anderen Ministerien und der Stabsstelle für Verwaltungsreform im Innenministerium das ressortübergreifende Umweltinformationssystem (UIS) als Teil des Landessystemkonzepts Baden-Württemberg bedarfsorientert weiter aus. Das UIS ist der Rahmen für die Bereitstellung von Umweltdaten und die Bearbeitung von fachbezogenen und fachübergreifenden Aufgaben im Umweltbereich. Der aktuelle Stand der Softwaretechnologie bietet modulare Werkzeuge sowie neue Standards und stellt die Verbindung zur Microsoft-Welt der Personalcomputer als Arbeitsumgebung her.

Mit dem Forschungsprojekt GLOBUS - Konsolidierung der neuen Systemarchitektur und Entwicklung erster Produktionssysteme für globale Umweltsachdaten im Umweltinformationssystem Baden-Württemberg, Phase III 1996 - beauftragte das Ministerium für Umwelt und Verkehr das bewährte Konsortium der nachfolgenden Institute:

- Forschungsinstitut für anwendungsorientierte Wissensverarbeitung (FAW) an der Universität Ulm,
- Forschungszentrum Informatik (FZI) an der Universität Karlsruhe,
- Forschungszentrum Karlsruhe - Technik und Umwelt - Institut für angewandte Informatik (FZK/IAI)
- Institut für Kernenergetik und Energiesysteme (IKE) der Universität Stuttgart,
- Institut für Photogrammetrie und Fernerkundung (IPF) der Universität Karlsruhe.

Das Projektmanagement und die Erstellung des Abschlußberichts übernahm das FZK/IAI.

Neue IuK-Standards, neue Softwaretechnologie und das Dienstekonzept bedürfen der Erprobung in der Praxis. Insbesondere in Zeiten knapper Ressourcen ist der zeitnahe Technologietransfer von den Forschungsergebnissen in den täglichen Betrieb sehr wichtig. Im Rahmen von GLOBUS III wurden auf der Basis der neuen, im Verlauf der vorhergehenden Projektphasen erarbeiteten Systemarchitektur erste Prototypen entwickelt und in der Landesanstalt für Umweltschutz erfolgreich getestet. Ein besonderer Augenmerk wurde dabei auf die Integration der Entwicklungen in die PC-Welt als Plattform für die Bürokommunikation gelegt. Gleichzeitig galt es, die vom Markt bereitgehaltenen Softwarelösungen und Schnittstellen zu nutzen.

Die Arbeiten von GLOBUS III fließen in die Rahmenkonzeption des Informationssystems Wasser, Abfall, Altlasten , Boden (WAABIS) und in das im Auftrag des Umweltbundesamtes zu realiserende F+E-Projekt Hypermediatechnik für Umweltdaten (HUDA) unmittelbar ein. WAABIS ist für den IuK-Verbund Land/Kommunen und HUDA für die IuK-Unterstützung der Umweltberichterstattung von großer Bedeutung.

Wir möchten an dieser Stelle den am Projekt beteiligten Instituten für Ihre wertvolle Arbeit herzlich danken und hoffen, daß die Ergebnisse von GLOBUS III zu einer Effizienzsteigerung der Verwaltung und zu einer verbesserten Information der Öffentlichkeit beitragen.

Inge Henning, Roland Mayer-Föll; Ministerium für Umwelt und Verkehr Baden-Württemberg
Andree Keitel, Manfred Müller, Ernst Schmid, Horst Spandl; Landesanstalt für Umweltschutz

Inhalt

Das Projekt GLOBUS

1. Einführung

Die Aufgabe des Umweltinformationssystems des Landes Baden-Württemberg (UIS B-W; UIS) ist die Bereitstellung von Umweltdaten und die informationstechnische Unterstützung bei der Durchführung von fachbezogenen und fachübergreifenden Aufgaben in den Ministeriumsressorts und Behörden mit Umweltbezug. Das System soll zu einer Leistungsverbesserung und Rationalisierung in der Verwaltung beitragen.

Mit der Konzeption des UIS B-W wurde 1984 unter der Federführung des Ministeriums für Ernährung, Landwirtschaft, Umwelt und Forsten begonnen. Die Konzeption wurde 1987 bis 1990 unter Führung des Umweltministeriums verfeinert und erweitert. Parallel zur Konzeption begann die Realisierung von Basis- und Teilsystemen des UIS B-W /1/, /2/, /3/.

Der technische Fortschritt im IuK-Bereich bietet ständig verbesserte Verfahren sowie Hard- und Software-Werkzeuge, die effizientere Problemlösungen ermöglichen. Parallel hierzu ergeben sich z. B. durch Umorganisation in der Verwaltung, neue gesetzliche Auflagen, usw. ständig neue Anforderungen an das System.

Dies erfordert eine kontinuierliche Fortentwicklung des systemtechnischen Rahmenkonzepts für das UIS B-W, an der sich die Neu- und Weiterentwicklungen im Rahmen des UIS orientieren müssen.

Mit dem Ziel einer kontinuierlichen funktionalen und IuK-technischen Aktualisierung des UIS werden vom zuständigen Ministerium ständig Forschungs- und Entwicklungsprojekte zu unterschiedlichen Aspekten der Fortschreibung des UIS-Konzepts durchgeführt /4/, /5/, /6/.

In diesem Rahmen läuft seit 1994 das Projekt GLOBUS.

Das Gesamtziel dieses Projekts ist es, primär für Referenten und Sachbearbeiter im Ministerium, in der Landesanstalt für Umweltschutz (LfU) sowie in anderen Behörden eine aktive, benutzerfreundliche Auskunftskomponente im UIS B-W zu entwickeln, die unterschiedlichste Informationsbestände auch über den Rahmen des UIS hinaus erschließt. Auf diese Weise soll eine Effizienzsteigerung in Sachbearbeitungs- und Entscheidungsvorgängen erreicht werden.

Parallel hierzu sollen Methoden zur Verbreitung von Umweltinformationen an Ingenieurbüros und die Öffentlichkeit erschlossen werden.

An dem F+E-Entwicklungsprogramm Projekt GLOBUS sind die folgenden wissenschaftlichen Institutionen beteiligt:

- das Forschungsinstitut für anwendungsorientierte Wissensforschung an der Universität Ulm (FAW),
- das Forschungszentrum Informatik an der Universität Karlsruhe (FZI),
- das Institut für Angewandte Informatik des Forschungszentrums Karlsruhe (FZK/IAI; FZK) (seit 1995),
- das Institut für Kernenergetik und Energiesysteme der Universität Stuttgart (IKE) und
- das Institut für Photogrammetrie und Fernerkundung der Universität Karlsruhe (IPF).

Das Projekt GLOBUS umfaßt (bisher) drei einjährige Entwicklungsphasen:

In der ersten Phase des Projekts (GLOBUS I) wurde 1994 eine Auskunftskomponente auf Basis des World-Wide Web (WWW) zur Erschließung der Umweltdatenberichte konzipiert und prototypisch realisiert /7/.

Ziel der zweiten Phase des Projekts (GLOBUS II) war es, in 1995 eine übergreifende Rahmenarchitektur für die Auskunftskomponente zu konzipieren und diese an beispielhaften Anwendungen zum Einsatz zu bringen /8/.

Die in diesem Bericht skizzierten Ergebnisse der GLOBUS-Projektphase II und die der vorangegangenen Projekte bilden die Grundlage für die Projektphase III des GLOBUS-Projekts: *'Konsolidierung der neuen Systemarchitektur und Entwicklung erster Produktionssysteme für globale Umweltsachdaten im Umweltinformationssystem Baden-Württemberg'* .

2. F+E-Arbeiten und Ergebnisse von GLOBUS II

Aufbauend auf die Ergebnisse der Phase I des GLOBUS-Projekts wurde in der Phase II das Konzept einer diensteorientierten, WWW- und CORBA (Common Object Request Broker Architecture)-basierten Client-Server Architektur für das UIS B-W entwickelt, die Grundlagen für eine Implementierung der Architektur geschaffen und prototypische Realisierungen auf dieser Basis durchgeführt.

In einer Arbeitsgruppe der Projektpartner wurde als Voraussetzung für einen Architekturentwurf eine im Rahmen einer UIS-Konzeption geeigneten Definition des Dienstebegriffs vorgenommen. Es wurde eine Klassifizierung von Dienstetypen festgelegt, Vorschläge für die Beschreibung der Funktionalität und der Schnittstellen entwickelt und für eine UIS-Konzeption geeignete Dienstetypen funktional definiert. Speziell behandelt wurde die Integration externer Dienste als Verfahren zur Einbindung vorhandener Anwendungssoftware in eine neue Architektur.

Im Rahmen von GLOBUS II wurde die Architektur von WWW-UIS entwickelt. Diese Architektur basiert auf dem World-Wide Web. Die Komponenten dieser Architektur lassen sich horizontal und vertikal gliedern.

Vertikal ergibt sich eine Unterscheidung zwischen Auskunfts- und Datendiensten. Die Auskunftsdienste basieren auf dem Datenmodell des in der Mehrzahl der deutschen Bundesländer sowie in Österreich eingesetzten Umweltdatenkatalogs (UDK). Die Datendienste realisieren den Zugriff auf unterschiedliche Datenquellen der übergreifenden Komponenten des UIS Ba-

den-Württemberg.

Horizontal wird zwischen Anwender- und Systemdiensten unterschieden. Die Systemdienste realisieren den Zugriff auf die unterschiedlichen Datenquellen sowie die Aufbereitung der Ergebnisse zu HTML-Seiten. Anwenderdienste kombinieren in der Regel einen oder mehrere Dienste für Datenzugriffe mit einem Aufbereitungsdienst. Durch diese Struktur sowie den modularen Aufbau der Dienste ergibt sich die Möglichkeit zur Nutzung der Techniken von CORBA in heterogenen Umgebungen.

Als Grundlage für die Nutzung von CORBA wurden im Rahmen von GLOBUS II die auf dem Markt verfügbaren CORBA-Implementierungen anhand genereller und UIS-spezifischer Kriterien (Funktionsumfang, Erfüllung der Standards, Performance, unterstützte Plattformen usw.) evaluiert. Aus 15 CORBA-Implementierungen wurden aufgrund der verfügbaren Produktbeschreibungen und -informationen drei ausgewählt und in geeigneter Testumgebung installiert. In diesen Installationen wurden anhand UIS-anwendungsnaher Dienste die Systeme ObjectBroker von DEC, Orbix von IONA (spez. bzgl. Kernsystem) und DAIS von ICL (spez. bzgl. Trader) ausgetestet. Die Testergebnisse zeigen den Stand der CORBA-Implementierungen, beweisen die Eignung von CORBA als Bausteine einer diensteorientierten UIS-Architektur und zeigen die Nutzungsmöglichkeiten von CORBA an prototypischen praxisnahen Beispielen.

Als weiteres Teilthema umfaßte das Projekt GLOBUS II die Konzeption eines Managementsystems für verteilte Informationsquellen im UIS B-W. Das Konzept basiert und ist abgestimmt auf die Arbeiten und Ergebnisse des Projekts INTEGRAL /4/. Unter Nutzung von Synergien mit diesem Forschungsvorhaben wurden in GLOBUS II der Informationsquellenkatalog prototypisch realisiert. Er bildet das Kernstück des Konzepts, das Filter- und Brokerfunktionen umfaßt, die das Auffinden von Umweltinformationen aus verteilten, heterogenen Informationsquellen im WWW unterstützt, wobei Metainformationen aus dem UDK, aus einem polyhierarchischen Thesaurus, aus Ortsverzeichnissen (Gazetteers), einem Netzwerkdiensteverzeichnis oder einem Benutzerverzeichnis zur Navigationsunterstützung genutzt werden können.

Im Rahmen dieser Arbeiten wurden zwei generische Tools realisiert, die als wesentliche Komponenten beim Aufbau von Informationsservern in Weitverkehrsnetzen einsetzbar sind:

Das Tool WebQuery ermöglicht es, Anfragen an ORACLE-Datenbanken als Dienste unter WWW zur Verfügung zu stellen.

Da das WWW kein Werkzeug anbietet, um Sessions zu verwalten, wurde ein entsprechendes Tool, das die wichtigsten Funktionen eines Sessionmanagements übernimmt, konzipiert und prototypisch realisiert.

Die vom WWW angebotene Funktionalität reicht für viele Anwendungen im Rahmen des UIS nicht aus. Deswegen wurden in GLOBUS II Möglichkeiten untersucht, clientseitig Anwenderapplikationen und serverseitig externe Dienste in das WWW einzubinden. Die untersuchten Lösungsansätze basieren serverseitig auf dem Common Gateway Interface CGI und clientseitig auf dem für MOSAIC verfügbaren Common Client Interface CCI. Als Anwendungsbeispiel wurde das Programm CRAYSIM zur Simulation der Ausbreitung von Spurenstoffen in der Athmosphäre als externer Dienst unter WWW verfügbar gemacht.

Ein weiterer Teil der Aufgabenstellung in GLOBUS II umfaßte Arbeiten zum Einsatz von Werkzeugen zur Präsentation von Umweltdaten in WWW-basierter Client-Server-Umgebung. Hierzu wurden marktverfügbare Werkzeuge zur Visualisierung von Geodaten, zur Tabellenkalkulation, Textverarbeitung usw. unter den Aspekten Funktionalität, Leistung und Integrierbarkeit im WWW evaluiert und Architekturen für den client- und serverseitigen Einsatz analysiert. Als prototypische Anwendung wurde eine Schnittstelle zum Räumlichen Informations- und Planungssystem (RIPS) der LfU definiert und eine einfache Auskunftskomponente für den clientseitigen Zugriff auf RIPS-Informationen geschaffen.

Als praktische Anwendung, die die Ergebnisse von GLOBUS I und vorangegangener Entwicklungsprojekte im Rahmen des UIS B-W umsetzt, wurde in GLOBUS II die erste Version des Altlasten-Fachinformationssystems AlfaWeb entwickelt. Ziel von AlfaWeb ist es, den Altlasten-Sachbearbeitern in den Behörden und Ingenieurbüros Arbeitshilfen in Form von Berichten, Handbüchern, Daten aus Datenbanken usw. über WWW zur Verfügung zu stellen und durch rechnergestützte Zugangshilfen ein schnelles Auffinden der benötigten Fachinformationen zu ermöglichen. In GLOBUS II wurden mehrere Altlasten-Handbücher und -Berichte für das WWW aufbereitet, auf der Basis des UBA-Thesaurus verschlagwortet und in das HTML-Format des WWW konvertiert. Daneben wurde eine Zugangsstruktur über Schlagworte sowie über Volltextsuche und die Berichtsstruktur realisiert. Für eine einfache Konvertierung zukünftiger Berichte ins WWW wurde eine Richtlinie zur Erstellung von Dokumenten erarbeitet.

3. GLOBUS Systemkern

Eine wesentliche Aufgabe im Rahmen von GLOBUS ist die kontinuierliche Übernahme von Forschungs- und Entwicklungsergebnissen in den produktiven Betrieb. Hierzu wurde der Begriff des *Systemkerns* eingeführt, unter dem die zu übernehmenden Teilergebnisse zusammengefaßt wurden.

Der GLOBUS Systemkern umfaßt zum einen alle Komponenten (Module und Schnittstellen) aus dem GLOBUS Umfeld, deren technische Ausprägung soweit konsolidiert und in die bestehende Systemumgebung integrierbar ist, daß sie vom Informationstechnischen Zentrum (ITZ) der Landesanstalt für Umweltschutz Baden-Württemberg (LfU) technisch und personell betreut werden können. Mit anderen Worten: Alles, was praktisch schon oder mit Hilfe einer relativ kurzen Konsolidierungsphase *Produktionscharakter* bekommen kann. Darüber hinaus werden die im folgenden beschriebenen Konzepte als Basis für die Weiterentwicklung des Systemkerns festgeschrieben. Ihre spätere Übernahme in den Systemkern wird vom jeweils erreichten Entwicklungsstand abhängig gemacht. Aktuell laufende, noch nicht abgeschlossene Arbeiten sind vorerst kein Bestandteil des Systemkerns. Die folgende Übersicht gibt den Status des Systemkerns im November '96 wieder. Die Zusammenstellung wird kontinuierlich überprüft und bei Bedarf anhand des Arbeitsfortschritts vom ITZ aktualisiert. Der GLOBUS Systemkern hat somit wenig mit dem Begriff des Systemkerns aus der Betriebssystemwelt gemeinsam, da die GLOBUS Komponenten einer wesentlich loseren Kopplung unterliegen.

3.1 Basiskomponenten des GLOBUS Systemkerns

Die Basiskomponenten bestehen aus den Softwarepaketen, mit deren Hilfe die verschiedenen GLOBUS Teilkomponenten implementiert werden. Dabei werden sowohl frei verfügbare als auch kommerzielle Softwarewerkzeuge eingesetzt. Sie bilden den informationstechnischen Hintergrund von GLOBUS.

- Freeware (frei erhältliche Software):
  - WWW-Server von NCSA
  - die Scripting-Sprachen perl und Tcl
  - Geo-Informationssystem grass
  - diverse Graphikkonverter, GIF-Tools (pbmplus, gnuplot, gifdraw, fly)
  - HTML-Textkonverter (rtftoweb) und HTML-Freitextsuche (swish)
- Kommerzielle Software (die Zielumgebung im ITZ):
  - Datenbanksystem Oracle v7.1 (oder höher)
  - Betriebssystem Digital UNIX v3.2 (oder höher)
  - WWW-Browser, der mindestens HTML 2 mit den Erweiterungen um Tabellen, Java Applets und die Netscape-Frames darstellen kann, insbesondere der Netscape Navigator v2.x (oder höher),
  - MS-Office v4.x (oder höher)
  - ArcView v2.x (oder höher)
- Weitere Software:
  - Umweltdatenkatalog UDK v3.1 (oder höher)

3.2 Basiskonzepte des GLOBUS Systemkerns

Aus den Ergebnissen der vorangegangenen GLOBUS Projektphasen wurden verschiedene Basiskonzepte herausgearbeitet, die eine konsolidierte Grundlage für die Weiterentwicklung innerhalb von GLOBUS III bilden. Die Basiskonzepte beschreiben somit die übergreifenden Rahmenbedingungen für die einzelnen Teilprojekte, die in den weiteren Abschnitten des Abschlußberichts zu GLOBUS III vorgestellt werden.

- Navigationskonzept: Dieser Punkt umfaßt vor allem allgemeine Gestaltungsrichtlinien, wie HTML-Seitenrahmen und durchgängige Semantik für Icons/Kommandos. Der Einstieg in die Umweltinformationen im WWW soll auf verschiedenste Arten möglich sein. Die jeweiligen Aufsetzpunkte für einzelne Navigationsgruppen, wie z.B. ein Selektor, der aus mehreren HTML-Seiten besteht, sollen deshalb direkt als Uniform Ressource Locator (URL) angesprochen werden können. Bei solchen Navigationsketten soll zunächst ein einfacher Einstieg vor möglicherweise speziellen Expertenfunktionalitäten liegen. Defaultbelegungen für Anfragen sind, soweit sinnvoll, beim einfachen Einstieg zu verwenden. Benötigt eine Teilkomponente Metadaten für ihre Navigation bzw. zur Erfüllung ihrer Aufgabe, so sind die Grundsätze aus dem Abschnitt zu Metadaten zu beachten.

- Metadatenkonzept: Der Umweltdatenkatalog UDK ist die Grundlage des Meta-Auskunftssystems für GLOBUS. Das Modul WWW-UDK bildet somit die primäre

Suchmaschine für GLOBUS, in dem Sinne, daß jedwede abfragbare Metainformation, die im UDK darstellbar ist, zumindest über die Funktionalitäten von WWW-UDK verfügbar sein muß. Bei der Speicherung und Nutzung von Metainformation soll, wenn die Möglichkeit besteht, die Datenstruktur des UDK verwendet werden.

Verantwortlich für die Struktur der UDK-Datenbank ist die Bund-Länder-Kooperation UDK. Die UDK-Inhalte und Erfassungskonventionen für Baden-Württemberg werden von ITZ festgelegt. Für den Zugriff auf die UDK-Datenbestände im WWW wird die Lösung WWW-UDK eingesetzt /8/. Die Erstellung der erforderlichen Zusatztabellen für die Verknüpfungsfunktionalität des WWW-UDK, sowie eventuell erforderliche Erweiterungen des UDK-Datenmodells (Zusatztabellen) erfolgen nach Abstimmung mit dem ITZ konzentriert vom zuständigen Projektpartner.

Der Ausbau des Gazetteers /8/ zu einem Geo-Referenzsystem wird in Abstimmung mit den GLOBUS-AGs *GIS/Kartographie* und *Metadaten* in Form eines eigenen Moduls weiterentwickelt.

- Selektorenkonzept: Das GLOBUS Architekturkonzept sieht bei den Selektoren für Datenzugriffe eine Aufteilung der einzelnen Komponenten in Anwendungs- und Systemdienste vor. Grob formuliert sind die Anwendungsdienste für die Selektor-Schnittstelle (HTML) und die Systemdienste für die Beschaffung und Aufbereitung der erforderlichen Daten zuständig (zu den grundsätzlichen Anforderungen an die Navigation siehe oben). Die Implementierung von Selektoren (Anwendungs- als auch Systemdienste) erfolgt nach interner Abstimmung unter Beachtung der LfU-Standards des ITZ für Datenbankentwicklungen.

  Grundlage für allgemeine Anfragen bildet das Tool WebQuery /8/. Die neue Version, die einige zusätzliche Anforderungen erfüllt, sollte nach Möglichkeit für die Konstruktion von allen Selektoren und Datenzugriffen genutzt werden. Jeder fertige Selektor bzw. Anwendungs- und Systemdienst mitsamt den zugehörigen Schnittstellenbeschreibungen gehört zum GLOBUS Systemkern. Soweit möglich, sollte bei den laufenden Arbeiten auf fertige Teilkomponenten zurückgegriffen werden.

  Anmerkung: Die CORBA-Entwicklungen sind als Teil von GLOBUS III noch nicht Bestandteil des Systemkerns. Trotzdem sollte jetzt schon bei den jeweiligen Arbeiten darauf geachtet werden, daß einem späteren Übergang zu einer CORBA-basierten Architektur keine grundsätzlichen Hindernisse entgegenstehen.

- Konzept der Graphik- und Kartendienste: Die Webserver-basierten Graphik- und Kartendienste aus GLOBUS II (Diagramme, Karten und entsprechende HTML-Aufbereitung) stehen als Rahmenprogramm zur Verfügung, die in Abstimmung mit der AG *GIS/Kartographie* für andere Nutzungen entsprechend angepaßt werden müssen. Aufgrund neuer Entwicklungen mit Java werden die einzelnen Module vorerst auf dem derzeitigen Stand eingefroren und konsolidiert, aber nicht mehr funktional weiterentwickelt.

Bei der Anbindung der verschiedenen Kartendienste (Server-basiert mit GRASS, Client-basiert mit ArcView, Server- und/oder Client-basiert mit Java) sollen die von den AG's *Dienste* bzw. *GIS/Kartographie* festzulegenden einheitlichen Datenschnittstellen (siehe auch unter Punkt GLOBUS-Datentypen) verwendet werden.

- Berichte-Konzept: Die Werkzeuge zur Generierung von HTML-Dokumenten, zur Verschlagwortung und zur späteren Freitextsuche sind Bestandteil des Systemkerns. Hierzu zählen auch die Dokumentenrichtlinien für Word-Dateien /8/.

- Sessionkonzept: In GLOBUS II wurden unterschiedliche Varianten für ein Sessionmanagment realisiert. So wurde einerseits ein umfassendes Managementkonzept erstellt, das jeweils individuell implementiert werden muß, andererseits gibt es eine anwendungsneutrale Software, die um bestehende Module herumgelegt werden kann, ohne daß diese geändert werden müßten. Die beiden Ansätze ergänzen sich. Eine Zusammenführung wurde geprüft und als nicht praktikabel verworfen. Die genauen Anforderungen an ein Sessionkonzept müssen überprüft und gegebenenfalls neu formuliert werden.

- Userverwaltung: Die implementierte Userverwaltung ist ein Zusatzmodul für die Datenbankrecherche, falls der Datenzugriff eingeschränkt werden soll (pro Selektor), oder wenn Daten längere Zeit gespeichert werden sollen. Die genauen Anforderungen müssen ebenfalls überprüft und gegebenenfalls neu formuliert werden

- Konzept der GLOBUS-Datentypen: Die Form der zwischen den diversen Diensten des Systemkerns ausgetauschten Daten muß abgestimmt werden. Mögliche Varianten für formale Schnittstellenbeschreibungen (SGML, EDIFACT etc.) müssen untersucht werden, um ein geeignetes Verfahren auswählen und implementieren zu können.

3.3 Zusammenspiel mit den Teilprojekten

Die Beiträge zum GLOBUS Systemkern befanden sich in kontinuierlicher Rückkopplung zu den einzelnen Arbeitspaketen in GLOBUS III. So war es einerseits möglich, konsolidierte Zwischenergebnisse rasch einer breiten Nutzung innerhalb des Projekts zuzuführen; andererseits haben die Standardisierungsbemühungen des GLOBUS Systemkerns eine gute Ausgangsbasis geschaffen, um den zu erwartenden Aufwand bei der Übernahme der neuen Ergebnisse aus GLOBUS III in den produktiven Betrieb zu reduzieren, ohne die Forschungsarbeiten allzusehr einzuschränken.

4. F+E-Vorhaben in Phase III des Projekts GLOBUS

In der folgenden Zusammenstellung der F+E-Aufgaben für GLOBUS III sind Anpassungen in Aufgabenstellung und Prioritätssetzung, die im Projektverlauf vereinbart wurden, bereits berücksichtigt.

4.1 Übergreifende F+E-Arbeiten in GLOBUS III

Im Rahmen von GLOBUS III waren von den Projektpartnern in Arbeitsgruppen (AGs) gemeinsam folgende Aufgaben zu erledigen:

- Definition von Diensten -

Die in GLOBUS II entwickelten Dienstekonzepte waren in GLOBUS III zu vertiefen, zu konsolidieren und die Auswirkungen des Konzepts auf die Rahmenkonzeption des UIS festzulegen. Die Voraussetzungen für die praktische Umsetzung des Dienstekonzepts und für seine Implementierung waren zu schaffen. Zur Bearbeitung dieser Aufgaben wurde die AG *Dienste* etabliert.

- Evaluierung von Java -

Die oo Programmiersprache Java ermöglicht über die Integration von Java-Applets in HTML die clienseitige Realisierung komfortabler, interaktiver Benutzeroberfläche im WWW. Im Hinblick auf den Einsatz von Java im UIS B-W sollte daher im Rahmen von GLOBUS III Java und ähnliche neue Entwicklungen speziell auch unter dem Aspekt der Sicherheit evaluiert werden. Diese Aufgaben wurden von der AG *Java* bearbeitet.

4.2 F+E-Aufgaben für das FZI

- Weiterentwicklung von WWW-UIS II -

Die Aufgabenstellung für das FZI in GLOBUS III umfaßt die beiden Themenschwerpunkte Metadaten/UDK und Interoperabilität in verteilten heterogenen Umgebungen.

Im Rahmen von GLOBUS III war es das Ziel, die im Projekt WWW-UIS realisierten UDK-basierten Auskunftsdienste WWW-UDK zu erweitern (erweiterte Suchfunktionalität, mehrsprachiges Hilfesystem, verbesserte GIS-Anbindung (in Zusammenarbeit mit dem IPF)) und sie den aktuellen Weiterentwicklungen des UDK-Datenmodells anzupassen.

Wichtig für die Nutzung des UDK ist die Aktualität und Vollständigkeit der Metadatenbestände im UDK. In GLOBUS III sollten daher Konzepte und Werkzeuge für die automatische Fortschreibung der Metadaten im UDK entwickelt werden.

Zur Interaktion der Dienste in einer in heterogener Umgebung realisierten Dienstearchitektur

ist - der Evaluierung in GLOBUS II folgend - der Einsatz von CORBA als Middleware vorgesehen. Ziel in GLOBUS III war es, anhand von Beispielen im Rahmen des UIS die Interoperabilität zwischen Anwendungen auf heterogenen, verteilten Plattformen praktisch zu erproben. Hierzu waren die notwendigen CORBA-Installationen vorzunehmen und beispielhaft Datenbankdienste im Zusammenspiel mit einer Ausbreitungsrechnung (mit IKE; siehe Aufgabenstellung IKE) in einer CORBA-basierten Umgebung zur Verfügung zu stellen.

Das im Projekt WWW-UIS entwickelte Prototypsystem war im Rahmen von GLOBUS III funktional abzurunden und in wesentlichen Komponenten in ein Produktionssystem zu überführen.

4.3 F+E-Aufgaben für das FAW

- Weiterentwicklung von UFIS -

Die F+E-Aufgaben des FAW sind ausgerichtet auf die Weiterentwicklung des Umwelt-Führungs-Informationssystem UFIS bis hin zu einer konsolidierten Betriebsversion. Der Entwicklungszeitraum hierfür ist auf zwei Jahre angesetzt. Der erste Einjahresabschnitt ist eingebunden in das Projekt GLOBUS III.

Das Aufgabenspektrum umfaßt die Überarbeitung und Fortschreibung der Konzeption des Systems, orientiert an der neuen Systemarchitektur des UIS. Es beinhaltet die Erschließung der Hauptfunktionalitäten von UFIS in einer Dienstestruktur und die Realisierung der UFIS-Anwenderdienste unter WWW ggfs. mit Einsatz von CORBA sowie auf dieser Basis den Aufbau eines modellhaften PC-UIS-Arbeitsplatzes als UFIS-Client.

Ausgerichtet auf dieses Ziel umfaßte das Arbeitsspektrum für das FAW in GLOBUS III - 1996 die Erstellung einer Version von UFIS mit allen Funktionen des individuellen Arbeitsplatzes und Integration von ArcView, Excel und Word sowie eine komfortable Benutzerführung mit Hilfe von Entscheidungsregeln auf der Basis von Metainformation.

4.4 F+E-Aufgaben für das FZK/IAI

- Das Altlasten-Fachinformationssystem AlfaWeb -

Die Aufgabenstellung für das FZK im Rahmen von GLOBUS III umfaßte den weiteren Auf- und Ausbau des Altlasten-Fachinformationssystem AlfaWeb. Dies beinhaltete die Punkte:

- Inhaltlicher Ausbau des AlfaWeb

Bis Ende '96 sollten alle relevanten Berichte der LfU zum Thema Altlastenbewertung von der LfU gemäß der aktuellen Richtlinien aufbereitet werden und von FZK/IAI nach HTML konvertiert und in das Altlasten-Informationssystem eingebunden werden. Anhand eines geeigneten Systems sollte die Anbindung von Anwendungsprogrammen in AlfaWeb beispielhaft durchgeführt werden.

- Weiterentwicklung der Werkzeuge in AlfaWeb

Die in AlfaWeb angewendeten Verfahren und Tools zur Aufbereitung der Berichte als Hypertexte in HTML waren unter den Gesichtspunkten Funktionalitäten und Handhabbarkeit zu optimieren. Die verschiedenen Tools waren in einem durchgehenden Werkzeug zu integrieren.

- Fortentwicklung des Zugangssystems

Der Zugriff auf AlfaWeb über den Umweltdatenkatalog UDK sollte ermöglicht werden. Dazu waren geeignete Metainformationen zu spezifizieren, die (mit Unterstützung des FZI) in den UDK einzubringen waren. Das System zur Volltextsuche war zu erweitern.

- Installation

Neben der Entwicklungsversion im FZK/IAI war die jeweils aktuelle Produktionsversion von AlfaWeb auf einem Server der LfU zu installieren. Eine CD-Rom-Version des Systems zur lokalen Verwendung auf PCs sollte erstellt werden.

4.5 F+E-Aufgaben für das IKE

- Externe Dienste im UIS -

Das Aufgabenspektrum des IKE im GLOBUS III ist ausgerichtet auf die Integration externer Dienste in die WWW- und CORBA-basierte Architektur des UIS B-W.

Es war hierfür auf Serverseite ein allgemeines Konzept zu entwerfen und ein Client für die Einbindung von Viewern zu erstellen. Der Client soll auch im UFIS-PC-Arbeitsplatz (siehe Aufgabenstellung FAW) einsetzbar sein.

Als Beispiel einer externen Anwendung diente das Simulationssystem CRAYSIM des IKE zur Berechnung der luftgetragenen Ausbreitung von Spurenstoffen. Es sollte im Rahmen von GLOBUS so weiterentwickelt werden, daß die Möglichkeiten, die sich durch das Dienstekonzept ergeben, prototypisch umgesetzt werden können.

4.6 F+E-Aufgaben für das IPF

- Clienseitige Kartendienste -

Die Aufgabenstellung in Phase III des Projekts GLOBUS zielt auf eine Umstellung der Kartendienste von einer batch-orientierten serverseitigen Lösung auf eine client-orientierte Lösung. Alle bisher unter WWW für das UIS B-W realisierten Dienste sind so konzipiert, daß sie auf dem Server des Diensteanbieters ablaufen. Daraus resultierende Nachteile (hohe Netzbelastung, mangelhafte Interaktionsmöglichkeiten usw.) können durch Verlagerung interaktivorientierter Verarbeitungsschritte vom Server in den Client beseitigt werden.

Die Umstellung der Kartendienste sollte auf Basis der neuesten interaktionsunterstützenden Erweiterungen von HTML (JAVA o ä. und enstprechender WWW-Browser) vorgenommen werden.

Die Aufgabe des IPF bestand in der Auswahl und Installation einer geeigneten Systemumgebung. Die Ergebnisse der AG Java waren in den Konzepten und Realisierungen zu berücksichtigen.

Für die Umstellung der serverorientierten Kartendienste waren die clientfähigen Funktionen zu extrahieren und clientseitig zu implementieren. Der Prototyp des Kartendienst-Clients war bei der LfU zu installieren.

5. Gliederung des GLOBUS III - Berichts

Der vorliegende Ergebnisbericht des Projekts GLOBUS - Phase III ist unterteilt in:
Teil 1: **Konzepte und Werkzeuge zur Systemarchitektur** und
Teil 2: **Konzeption und Implementierung von Anwendungskomponenten**.

Teil 1

Das erste Kapitel des Teil 1 **Die Ergebnisse des Projekts GLOBUS als Beitrag für die Fortschreibung der Rahmenkonzeption des Umweltinformationssystems Baden-Württemberg** beschreibt die Randbedingungen und die Notwendigkeit einer kontinuierlichen Anpassung von Konzeption und Technik des UIS B-W. Es beschreibt die Konsequenzen der GLOBUS-Ergebnisse auf die Weiterentwicklung des Systems.

Das Kapitel **Dienste im UIS Baden Württemberg** enthält den Bericht über die Ergebnisse der Arbeitsgruppe *Dienste*. Es stellt die Ergebnisse der AG zu den Themenbereichen Beschreibung von Diensten und ihrer Schnittstellen, Einsatzmöglichkeiten von CORBA im UIS B-W und Auswirkungen des Dienstekonzepts auf die Rahmenkonzeption des UIS dar.

Die nächsten vier Kapitel widmen sich der Thematik **CORBA im UIS B-W**.

Das erste Kapitel gibt eine kurze Übersicht über die Arbeiten zum CORBA-Einsatz im UIS, die im Rahmen von GLOBUS III durchgeführt wurden.

Das Kapitel **CORBA als Middleware-Komponente für das UIS** aktualisiert die Aussagen der in GLOBUS II durchgeführten CORBA-Evaluierung und geht hierbei speziell auf die CORBAservices und ihre Anwendungsmöglichkeiten ein.

Das Kapitel **Integration und Kombination von Simulationsdiensten und Datenbankzugriffsdiensten mit CORBA** beschäftigt sich mit dem Einsatz von CORBA zur transparenten Integration heterogener, verteilter Dienste im UIS. In einer beispielhaften Anwendung wird für die Integration unterschiedlicher Dienste eine Architektur entwickelt und in einem Demonstrationssystem implementiert.

Das letzte der CORBA-Kapitel **GIS-Operatoren unter CORBA** behandelt die Einbindung von GIS-Funktionen in ein diensteorientiertes UIS auf der Basis CORBA und beschreibt eine hierfür geeignete Architektur, die in einem einfachen Prototyp demonstriert wird.

Im Kapitel **Einsatz von Java im UIS B-W** werden die Arbeiten und Resultate der Arbeitsgruppe *Java* dargestellt. Es wurde eine Evaluierung von Java und JavaScript in Hinblick auf

den Einsatz im UIS B-W durchgeführt, Konzepte für den Einsatz von Java entwickelt und prototypisch realisiert.

Teil 2

Das erste Kapitel des Teil 2 **WWW-UDK / Metadaten** beschreibt die Arbeiten und Ergebnisse des FZI zum Thema on-line-Zugriff über WWW auf den Umweltdatenkatalog. Dargestellt wird die Umstellung von WWW-UDK auf das internationalisierte Datenmodell UDK 3.0 und der aktuelle Stand der Betriebsversion des WWW-UDK. Weiter beschrieben wird ein Konzept zur Automatisierung der Metadatenfortschreibung und dessen Realisierung an einem praktischen Beispiel.

Im Kapitel **Grenz-/Richtwertdatenbank** wird die Architektur und Realisierung einer Komponente zur unscharfen Datenbankabfrage im UIS über WWW beschrieben.

Im Kapitel **Fachübergreifende Umweltdaten für Generalisten und Führungskräfte** werden die Arbeiten des FAW zur Weiterentwicklung des Umwelt-Führungsinformationssystems (UFIS II) - speziell die Überarbeitung der Architektur und der in UFIS I entwickelten Selektoren - beschrieben. Es wird weiter ein Ansatz zur einheitlichen Meta-Beschreibung von Diensten im UIS und die Entwicklung eines Gazetteers zum raumbezogenen Zugriff auf Umweltinformationen dargestellt.

Das Kapitel **Altlasten-Fachinformationssystem AlfaWeb** stellt die Arbeiten des FZK im Rahmen von GLOBUS III dar. Es beschreibt den inhaltlichen Ausbau von AlfaWeb, das jetzt den Großteil der Altlastenberichte umfaßt. Weiterhin wird auf den Funktionsumfang des Systems und der zugehörigen Werkzeuge eingegangen. Angaben zur bisherigen Nutzung und zu den Zugriffsmöglichkeiten beschließen den Bericht.

Die folgenden beiden Kapitel behandeln den Themenbereich **GIS-Werkzeuge im UIS B-W** und umfassen damit die zentrale Thematik des IPF.

Der erste Beitrag **Konsolidierung der serverseitigen Kartendienste und Überführung in die GLOBUS-Betriebsversion** dokumentiert den Ausbau der im Rahmen von GLOBUS II entwickelten Kartendienste im WWW-UIS in Hinblick auf einen praxisgerechten Funktionsumfang und Bedienungskomfort. U. a. umfaßt dies die Aspekte Seitenmenueführung, Defaultbesetzungen, Sessionmanagement, ORACLE-Datenbankzugriffe usw..

Der zweite Beitrag des IPF **Architektur eines GIS-Terminals zur Visualisierung von Geodaten** beschreibt das Konzept/Schichtenmodell eines GIS-Arbeitsplatzes für die Aufbereitung und Darstellung von Geodaten aus heterogenen Informationsquellen. Das Konzept basiert auf Java und CORBA und wird anhand eines Prototyps mit eingeschränkter Funktionalität demonstriert.

Das letzte Fachkapitel des Berichts **Externe Dienste im UIS** behandelt als Beitrag des IKE die Einbindung externer Dienste in ein WWW-basiertes UIS. Anhand eines Beispiels (Luftgetragene Ausbreitung von Spurenstoffen) werden die Anforderungen und Randbedingungen für die Integration untersucht. Mit dem Entwicklung eines Personal Data Node (PDN) wird ein Framework für die Handhabung externer Dienste in einem persönlichen Arbeitsplatze beschrieben.

Im anschließenden *Ausblick* sind Perspektiven für eine Fortführung der im Projekt GLOBUS durchgeführten Arbeiten zusammengestellt.

6. Literatur

/1/ R. Mayer-Föll, P. Schilling, D. Weigert et al;
Konzeption für das Umweltinformationssystem Baden-Württemberg.
Ministerium für Ernährung, Landwirtschaft, Umwelt und Forsten
Baden-Württemberg,
Mai 1986

/2/ Umweltministerium Baden-Württemberg / McKinsey and Company, Inc.;
Konzeption des ressortübergreifenden Umweltinformationssystems
im Rahmen des Landessystemkonzepts Baden-Württemberg
(Phase I: Bestandsaufnahme und inhaltliche Konzeption;
Phasen II/III: Systemkonzeption und Umsetzungsplanung;
Phase IV: Weiterentwicklung der Rahmenkonzeption;
Phase V: Umsetzung der Rahmenkonzeption).
1987-1990

/3/ Arbeitsgemeinschaft Diebold - Dornier - Ikoss;
Erstellung eines Landessystemkonzepts für einen rationellen und
wirtschaftlichen Einsatz der Informations- und Kommunikationstechniken
in der öffentlichen Verwaltung des Landes Baden-Württemberg,
Dezember 1984

/4/ K. Blank, W.-F. Riekert, F. Schmidt, M. Tischendorf;
Projekt Integral: Integration heterogener Komponenten
des Umweltinformationssystems (UIS) Baden-Württemberg.
Abschlußbericht der Phase I.
FAW Ulm und IKE Universität Stuttgart, 1994

K. Blank, W.-F. Riekert, Th. Kirst, R. Münster, F. Schmidt, M. Tischendorf, K. Kübler;
Projekt Integral: Integration heterogener Komponenten
des Umweltinformationssystems (UIS) Baden-Württemberg.
Abschlußbericht der Phase II.
FAW Ulm und IKE Universität Stuttgart, 1994

FAW Ulm, IKE Universität Stuttgart
Projekt Integral: Integration heterogener Komponenten
des Umweltinformationssystems (UIS) Baden-Württemberg.
Abschlußbericht der Phase III.
in Vorbereitung

/5/ R. Kramer, R. Nikolai;
WWW-UIS Anwenderhandbuch.
Forschungszentrum Informatik (FZI), Karlsruhe, 1995

R. Kramer, R. Nikolai;
WWW-UIS Administrationshandbuch.
Forschungszentrum Informatik (FZI), Karlsruhe, 1995

R. Kramer, R. Nikolai;
WWW-UIS: Auskunfts-, Administations- und Verwaltungsdienste.
Forschungszentrum Informatik (FZI), Karlsruhe, 1995

/6/ Umweltministerium Baden-Württemberg, FAW Ulm;
Umweltinformationssystem Baden-Württemberg -
Fortschreibung der UIS Rahmenkonzeption. Abschlußbericht der Phase I.

/7/ J. Wiesel, F. Schmidt, W.-F. Riekert, R. Kramer;
GLOBUS - Konzeption und prototypische Realisierung
einer aktiven Auskunftskomponente für globale Umwelt-Sachdaten.
Abschlußbericht Phase I.
Umweltministerium Baden-Württemberg, 1994

/8/ R. Mayer-Föll, A. Jaeschke
Projekt GLOBUS - Konzeption und prototypische Realisierung einer aktiven Auskunftskomponente für globale Umwelt-Sachdaten im Umweltinformationssystem Baden-Württemberg
Phase II 1995
Wissenschaftliche Berichte FZKA 5700

Teil 1:

Konzepte und Werkzeuge zur Systemarchitektur

Die Ergebnisse des Projekts GLOBUS Phase III als Beitrag für die Fortschreibung der Rahmenkonzeption des Umweltinformationssystems Baden-Württemberg

R. Mayer-Föll,
Ministerium für Umwelt und Verkehr, Baden-Württemberg,
Kernerplatz 9,
70182 Stuttgart

J. Strohm, A. Schultze
Forschungsinstitut für anwendungsorientierte Wissensverarbeitung (FAW),
Helmholtzstr. 16,
89081 Ulm/Donau

1. Einführung

Das Umweltinformationssystem Baden-Württemberg (UIS-BW) hat mittlerweile eine beachtliche Leistungsfähigkeit erreicht. In den verschiedenen Ministerien des Landes Baden-Württemberg mit umweltrelevanten Aufgabenbereichen gibt es mittlerweile über 80 größere IuK-Vorhaben mit 200 Projekten, in denen etwa 1200 Einzelprogramme zur Anwendung kommen. Von diesen Systemen befinden sich etwa die Hälfte im Geschäftsbereich des Ministeriums für Umwelt und Verkehr. Weitere wesentliche UIS-relevante Vorhaben werden in den Geschäftsbereichen des Minsteriums Ländlicher Raum, des Wirtschaftsministeriums, Innenministeriums, Sozialministeriums, Finanzministeriums und des Wissenschaftsministeriums eingesetzt. Hinzu kommen noch Systeme auf kommunaler Ebene und länderübergreifende Anwendungen, wie der Umweltdatenkatalog (UDK). Aus der Vielzahl dieser Systeme wird der starke fach- und ressortübergreifende Charakter des UIS deutlich. Das Zusammenwirken und die Arbeitsteilung der verschiedenen Systeme soll die Darstellung der UIS-Pyramide (s. Abbildung 1) verdeutlichen. Der mit dem übergreifenden Ansatz verbundene Nutzwert für die Qualität und die Quantität der Daten bringt allerdings auch einen Abstimmungsaufwand mit sich und macht eine enge Zusammenarbeit der beteiligten Stellen erforderlich.

Der übergreifende Systemansatz und die Einbeziehung eines wesentlichen Teils der Verwaltung stellt eine große Herausforderungen für Konzeption, Entwicklung, Ausbau und Betrieb des UIS-BW dar. So führen die vorhandenen Systemplattformen und Bürokommunikationswerkzeuge zu einem noch zu heterogenen Gesamtsystem, was sich nachteilig bei der übergreifenden Nutzung von Systemen und Daten bemerkbar macht. Diese Problematik verschärft sich noch zusätzlich durch die zunehmende Verflechtung mit Bund, Ländern und Kommunen.

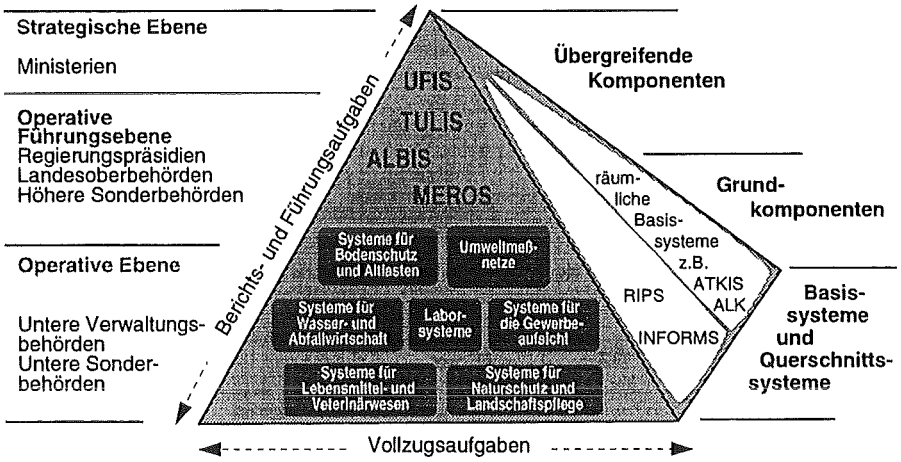


Abbildung 1: UIS-Pyramide Baden-Württemberg - Sicht auf die Systeme. Die Auswahl der Systeme ist beispielhaft.

In diesem Beitrag erfolgt eine Einschätzung der GLOBUS-Ergebnisse aus Sicht der Fortschreibung der UIS-Rahmenkonzeption sowie eine Einbettung dieser in GLOBUS entwickelten Konzepte in die UIS-Rahmenkonzeption.

2. Die Notwendigkeit der Fortschreibung der UIS-Rahmenkonzeption

Die Handlungsmotivation für die Fortschreibung der Rahmenkonzeption des UIS-BW ergibt sich aus der Veränderung der Rahmenbedingungen besonders in drei Bereichen:

Zum einen haben sich die noch vor Jahren gültigen Rahmenbedingungen aufgrund der rasanten Entwicklung der Computertechnologie deutlich verschoben. Bislang wurden einige der wesentlichen Komponenten des UIS noch auf zentralen Dienststellenrechnern unter DEC-VMS und IBM MVS oder auf PC-Einzelplatzsystemen unter DOS betrieben. Heute sind dagegen Betriebssysteme wie UNIX oder WINDOWS NT auf neuen, leistungsfähigen Prozessorgenerationen im Workstation- und PC-Bereich üblich. Einschneidend war zudem die Entwicklung von zentralen Mainframearchitekturen hin zu verteilten PC-Netzen und Client-Server-Architekturen und die damit verbundene Bedeutung übergreifender Netzwerke. Die zunehmende Verbreitung des Internet mit Mechanismen wie dem World-Wide Web (WWW) und das Entstehen neuer Standards im Middleware-Bereich wie der Common Object Request Broker Architecture (CORBA) ermöglichen heute im Rahmen einer Client-Server-Architektur die weltweite Nutzung von Ressourcen.

Zum zweiten erfolgte eine weitgehende Ergänzung der seitherigen Aufgabenstellung: Die Unterstützung der Umweltfachbehörden bei der Erledigung ihrer Aufgaben steht nach wie vor im Vordergrund. Die übergreifende Zusammenarbeit mit anderen Ländern, dem Bund und dem kommunalen Bereich sowie die Umweltberichterstattung und Harmonisierung von Daten und Systemen bis auf die europäische Ebene haben heute jedoch einen anderen Stellenwert als noch vor einigen Jahren. In allen Bereichen sind inzwischen Umweltinformationssysteme entwickelt worden, wodurch sich bei einer weitgehenden Harmonisierung der Daten und Systeme eine deutliche Verbesserung der Qualität und der Verschneidbarkeit des Datenmaterials ergeben kann. Eine solche Harmonisierung ist auch aus wirtschaftlichen Gründen sinnvoll, da die übergreifende Zusammenarbeit sowie die Bündelung von Ressourcen letztlich die, in Zeiten knapper Finanzmittel, wirtschaftlichste Vorgehensweise darstellen. Eine weitere Ergänzung ist in dem mittlerweile gesetzlich festgeschriebenen Recht des Bürgers auf freien Zugang zu Umweltinformationen zu sehen. Diesem Anspruch ohne große Mehrkosten nachzukommen, stellt eine beachtliche Herausforderung an das UIS-BW als Gesamtsystem dar.

Als drittes ist die Veränderung der Landesverwaltung durch die Verwaltungsreform zu nennen, die zu einer neuen Organisationsstruktur der Umweltverwaltung geführt hat. Hinzu kommt, daß im UIS-BW heute, wie in der Landesverwaltung ganz allgemein, betriebswirtschaftliche Strukturen eine weitaus wichtigerer Rolle spielen, als noch vor einigen Jahren. Außerdem sind die verschiedenen Stellen der Kommunen, der Länder, des Bundes sowie der Europäischen Union heute enger verflochten, als dies früher der Fall war.

3. Die GLOBUS-Ergebnisse

Eine Leitlinie für die Entwicklung des UIS stellen marktorientierte Mechanismen dar. Der grundsätzliche Gedanke hierbei ist es, Anbietern und Nutzern von Diensten oder Daten eine gemeinsame Plattform zur Verfügung zu stellen, über die eine Informationsweitergabe erfolgen kann. In einem solchen, verteilten System stellt die zentrale UIS-Verwaltung dann die Mechanismen zur Verfügung, mit deren Hilfe die Interoperabilität der Einzeldienste, die Konsistenz der Daten und die Aktualität der Metainformationen gewährleistet werden kann.

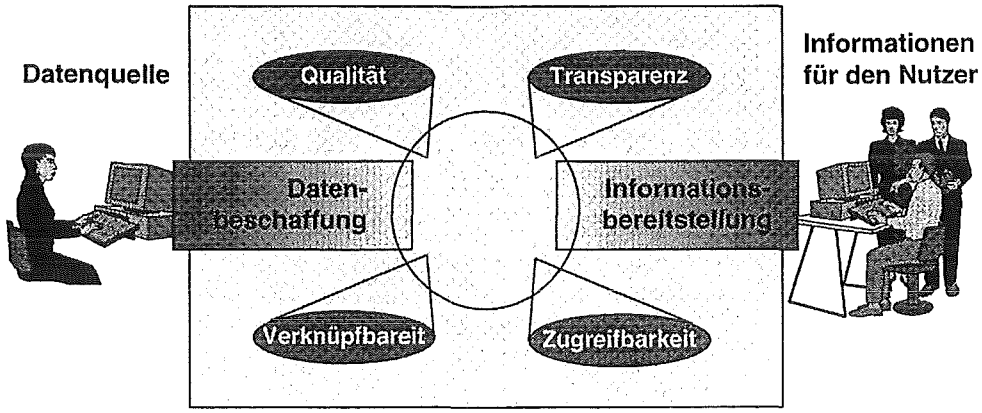


Abbildung 2: Informationsmanagement im UIS Baden-Württemberg auf der Basis des Umweltdatenkatalogs (UDK).

Vorteile sind, daß die Bereitstellung der Metainformationen in einem solchen Informationssystem dezentral von den Anbietern übernommen werden kann (s. Abbildung 2). Davon ist eine Reduktion des zentralen Verwaltungsaufwands und damit auch eine Kostensenkung zu erwarten. Modulare Anwendungssysteme sind außerdem sehr flexibel, so daß Daten und Dienste verschiedener Anbieter kombiniert und zu komplexen Systemen zusammengefügt werden können.

Voraussetzung hierfür ist die Schaffung eines effizienten Metainformationssystems. Als Kern dieses Systems wird im UIS der Umweltdatenkatalog (UDK) der Bund/Länder-Kooperation eingesetzt. Er stellt ein einheitliches Metadatenmodell zur Verfügung, das als Grundlage für die Bereitstellung von Filter- und Brokermechanismen sowie von Suchfunktionalitäten wie Repositories und Thesauri dient. Weitere Voraussetzungen sind wirkungsvolle Benutzer- und Sicherheitsmanagementsysteme, Verwaltungskomponenten und Verfahrensregeln.

4. Die GLOBUS-Ergebnisse aus technischer Sicht

Für eine detailliertere Beschreibung der im Forschungsprojekt GLOBUS erreichten Ergebnisse sei hier auf die entsprechenden Beiträge in diesem Abschlußbericht verwiesen. An dieser Stelle sollen diese Konzepte kurz zusammengefaßt werden:

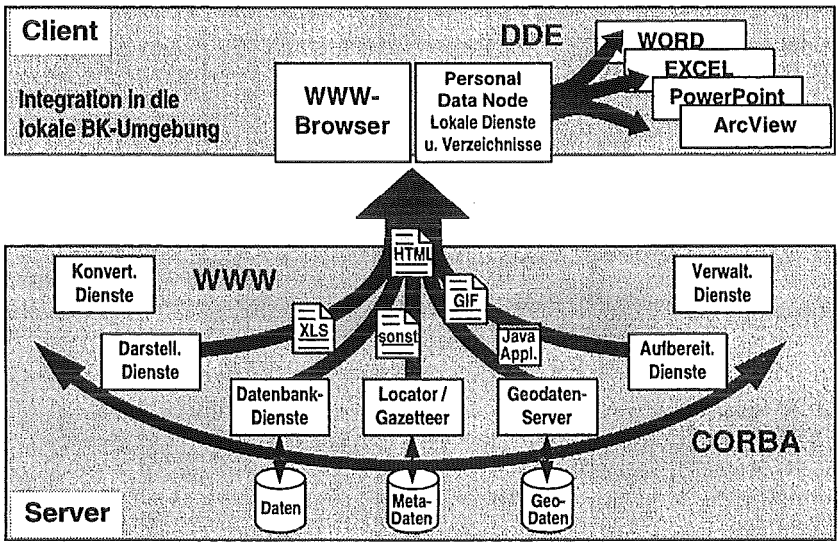


Abbildung 3: Die UIS-Dienstekonzeption - heutiger Stand der Umsetzung.

Das Konzept basiert auf dem Begriff des „Dienstes“ unter dem ganz allgemein eine abgeschlossene Programmeinheit verstanden wird, die eine spezielle Aufgabe erfüllt und die von einem Dienstanbieter bereitgestellt bzw. von einem Dienstenutzer verwendet werden kann. Letztlich stehen dann die Schaffung von Mechanismen zur Nutzung dieser Dienste bzw. zu ihrer Rekombination in größere Anwendungen im Vordergrund. Aus technischer Sicht werden hierfür die Werkzeuge und Techniken des World-Wide Web (WWW) bzw. neue Middlewarekonzepte wie die Common Object Request Broker Architecture (CORBA) in einem Client/Server-Architekturmodell verwendet. Das WWW stellt dabei für den Anwender ein geeignetes Medium dar, um Informationen aus dem UIS-Netzwerk (UIS-Intranet) bei verschiedenen Informationsservern abzufragen und in seine eigene Arbeitsumgebung einzubringen (s. Abbildung 3). Besondere Vorteile des WWW bestehen dabei in seiner Eigenschaft, die Heterogenität der unterschiedlichen Plattformen zu verbergen und auf allen Systemen ein annähernd identisches Look-and-Feel zu ermöglichen sowie in der verhältnismäßig problemlosen Übertragung der erhaltenen Daten in die lokale Arbeitsumgebung.

CORBA dient hingegen vorwiegend der serverseitigen Kommunikation zwischen einzelnen Diensten und ermöglicht damit die Erstellung komplexerer Anwendungen aus im Netz verteilten Einzeldiensten. Dies bleibt für den Endnutzer im wesentlichen transparent. Vorteile sind

hier unter anderem die Möglichkeit einer mehrfachen Verwendung von bestehenden Diensten und der hieraus resultierende geringere Pflegeaufwand, der sich letztlich in niedrigeren Entwicklungskosten niederschlagen sollte.

Ein entscheidender Punkt im Systemkonzept ist die Bereitstellung ausreichender Metainformation über die im UIS-Intranet verfügbaren Daten und Dienste sowie die Bereitstellung von Werkzeugen, die dem Nutzer die Dienst- und Datenauswahl erleichtern. Als Grundlage für die Bereitstellung von Metainformation dient der Umweltdatenkatalog.

5. Diskussion der GLOBUS-Ergebnisse aus Sicht der UIS-Rahmenkonzeption

Die in GLOBUS erreichten Ergebnisse sind wichtige Beiträge für die Weiterentwicklung des UIS im Hinblick auf die oben erwähnten geänderten Randbedingungen. So ist festzustellen, daß das Konzept den Erfordernissen des stark heterogen geprägten UIS-BW nach plattformtransparenten Systemkonzeptionen durch die Verwendung von WWW und CORBA nachkommt. Zudem befinden sich die eingesetzten Systemkomponenten im Einklang mit dem „Architekturmodell der Landesverwaltung für offene Systeme“ und den Standards und Regeln des Landessystemkonzepts. Die Offenheit des Ansatzes sollte darüberhinaus sicherstellen, daß zukünftig auf informationstechnische Weiterentwicklungen flexibel reagiert werden kann und Änderungen der Systeme mit verhältnismäßig geringem Aufwand erzielt werden können. Die Systementwicklungen, die im Rahmen des GLOBUS Projektes in den Phasen I-III erfolgten, zeigen darüberhinaus, daß das Konzept umsetzbar ist und praktisch funktionsfähige Anwendungen damit erstellt werden können (bezüglich näherer Einzelheiten sei auf den Bericht über die Arbeiten der AG Dienste in dem vorliegenden Abschlußbericht verwiesen).

Erste Erfahrungen aus dem Einsatz dieser Systeme zeigen zudem, daß die Akzeptanz für WWW-basierte Fach- und Auskunftssysteme in den potentiellen Nutzerkreisen außerordentlich hoch ist. Dies ist zum großen Teil auf die unmittelbare Sichtbarkeit der Vorteile zurückzuführen, die sich für den Nutzer aus der Verwendung des WWW ergeben. Für die Entwicklung übergreifender Informationssysteme ist dies eine wertvolle Erkenntnis, die erwarten läßt, daß Investitionen in Anwendungssysteme, auf Basis dieser Technologie, verhältnismäßig zukunftssichere Investitionen mit einem hohen Nutzen/Kosten-Verhältnis darstellen. Aufgrund des hohen Akzeptanzgrades und der Einfachheit der WWW-Technologie ist auch zu vermuten, daß deren Integration in bestehende Anwendungen verhältnismäßig schnell vonstatten gehen wird. Allerdings ist hierbei zu beachten, daß das WWW im Sicherheitsbereich noch Mängel aufweist, die vor einem breiten Einsatz dieser Technologie, insbesondere in Fachsystemen, noch Gegenstand weiterer Arbeiten sein müssen.

Die Umsetzung der CORBA-Konzepte wird sich hingegen etwas langsamer vollziehen. Gründe sind zum einen, daß der ganze Umfang des möglichen Nutzenpotentials, erst dann sichtbar und nutzbar wird, wenn bereits eine größere Anzahl vergleichbarer Anwendungen auf der Basis von CORBA implementiert worden sind, weil erst in diesem Fall Synergieeffekte in der Systementwicklung und -pflege erkennbar werden, und zum anderen, daß die Umstellung bestehender Anwendungen verhältnismäßig aufwendig ist. Um das Potential für zukünftige Synergieeffekte Schritt für Schritt aufzubauen, ist es von Seiten der Rahmenkonzeption sinn-

voll, die Nutzung von CORBA-Middleware bei der Neukonzeption oder der grundlegenden Überarbeitung von Anwendungen, die ohnehin notwendig werden, anzuregen. Dies ist insbesondere schon deswegen vernünftig, weil die Verwendung von CORBA als Middleware auch für eine separate Anwendung Vorteile bietet.

CORBA weist, ähnlich wie das WWW, noch Schwächen auf, die zum Teil auch den Bereich Sicherheit betreffen. Insofern gibt der Zeitverzug zwischen konzeptioneller Systembeschreibung und praktischer Umsetzung Gelegenheit, diese noch vorhandenen Schwachstellen zu beheben.

6. Sicherheitskonzepte - ein wichtiges Aufgabengebiet

Für die Umsetzung der GLOBUS-Ergebnisse sind Sicherheitskonzepte unumgänglich. Der Umgang mit dem Datenmaterial des UIS muß dabei in einer Weise erfolgen, in der die Vorschriften zu Datenschutz und Geheimhaltung erfüllt werden können. Im bisherigen System war dies dadurch gewährleistet, daß die Fachsysteme, die den größten Anteil an vertraulichen Daten enthalten, durch Abschottungsmaßnahmen nur einem bestimmten, klar definierten Nutzerkreis zugänglich waren. Der Export des Datenmaterials in die übergreifenden Systeme mußte dabei gezielt und unter Berücksichtigung der relevanten Vorschriften erfolgen. Aufgrund dieses Umstands sind Aspekte des Datenschutzes und der Datensicherheit bislang auch nicht Gegenstand der Rahmenkonzeption des UIS-BW gewesen.

In verteilten und offenen Systemen, wie es das GLOBUS-Dienstekonzept für das UIS vorschlägt, ist die Situation jedoch eine andere. Dienste sind hier innerhalb des Netzwerks aus technischer Sicht prinzipiell für alle Nutzer frei verfügbar. Für den Zugriff auf vertrauliches Datenmaterial muß dieser freie Zugang daher eingeschränkt werden. Der Sicherheitsstandard hierfür, ist dabei so hoch anzusetzen, daß keine datenschutzrechtlichen Bedenken von Seiten der Betreiber und Nutzer des Systems mehr bestehen. Dieser Aspekt ist ein wichtiger Faktor des Dienstekonzepts, weil die Einführung dieses Konzeptes im Bereich der Fachsysteme nur gelingen kann, wenn dem Bedürfnis nach Sicherheit der Systembetreiber, d.h. der einzelnen Fachbehörden, ausreichend Rechnung getragen wird.

Der Gesamtkomplex „Sicherheit" umfaßt drei Themenschwerpunkte: Die technische Seite betrifft die **Übertragungssicherheit**. Diese wird durch das Landesverwaltungsnetz Baden-Württemberg gewährleistet, für das ein fortgeschriebenes Datenschutz- und Sicherheitskonzept des Innenministeriums vorliegt. Auf organisatorischem Gebiet ist zu diskutieren, wie der **Zugang zu den Daten** reglementiert und die Beachtung datenschutzrechtlicher Vorgaben gewährleistet werden kann. Hierfür sind im UIS-BW weitere Vorgaben zu erarbeiten. Als drittes ist zu diskutieren, wie ein hinreichender **Qualitätsstandard** bei dem zur Verfügung gestellten Datenmaterial gewährleistet werden kann und wie Mißinterpretationen der bereitgestellten Daten verhindert werden können. Damit verbunden können auch Gewährleistungs- und Haftungsfragen sein.

Üblicherweise erfolgt die Zugangsreglementierung über die Definition s.g. Nutzerrollen. Dabei werden Nutzertypen Tätigkeitsbeschreibungen zugewiesen, die zum Zugriff auf bestimmte

Anwendungen oder zum Abruf bestimmter Daten berechtigen. Vorteilhaft ist dies deshalb, weil sich dieses System bei personellen Veränderungen leicht anpassen läßt, da Berechtigungen nicht an einzelne Personen, sondern an definierte Rollen geknüpft sind. Für das UIS sollte eine Kategorisierung in Fachsystemnutzer, Nutzer übergreifender Systeme und öffentliche Nutzer ausreichen. Die Anwendungen müssen dann so konzipiert werden, daß eine automatische Anpassung der Nutzersicht auf ein System oder einen Datenbestand entsprechend der Rollendefinition des anfordernden Nutzers erfolgt. Die Definition von Nutzerrollen muß grundsätzlich von einer vertrauenswürdigen Institution aus zentral erfolgen (Trust Center).

Um die Qualität und sinnvolle Verwendung der angebotenen Daten und Dienste sicherzustellen, müssen Metainformationen bereitgestellt werden. Diese müssen insbesondere auch Information über die Randbedingungen enthalten, unter denen die Daten zustande gekommen sind, bzw. Rückschlüsse auf die Funktionsweise eines Dienstes zulassen.

7. Wirtschaftlichkeit - die wichtigste Forderung heute

Der wirtschaftliche Nutzen der GLOBUS-Ergebnisse ergibt sich auf mehreren Ebenen: Die Unterstützung der Fachaufgaben wird durch übergreifende Ansätze und die bessere Integration der UIS-Systeme in die Bürokommunikationsumgebungen verbessert. Mehrfachentwicklungen im Softwarebereich werden ebenso vermieden, wie redundante und inkonsistente Datenhaltung bzw. -fortführung. Umständliche und langwierige Rechercheverfahren werden durch Informationstechnik schneller und qualitativ hochwertiger. Damit rückt die kooperative Nutzung verschiedenster Informationsquellen im Umweltbereich durch die GLOBUS-Ergebnisse in greifbare Nähe. Zwar sind wichtige Probleme wie die Erarbeitung eines Dienstemanagements oder die Einrichtung leitungsfähiger Sicherheitsmechanismen erst in Ansätzen gelöst, entscheidende Schritte wurden jedoch mit der Entwicklung eines umfassenden Informationsmanagementsystems und der sukzessiven Anbindung verschiedener Fachdatenbanken im Rahmen des GLOBUS-Projektes bereits unternommen.

Die GLOBUS-Ergebnisse ermöglichen auch, dem Anspruch des Bürgers auf freien Zugang zu Informationen über die Umwelt besser als bisher nachzukommen. Durch die Entwicklung verteilter UIS-Komponenten auf Basis des WWW und aufgrund der starken Verbreitung, die dieses Medium in den weltweiten Netzwerken findet, wird sich der Informationsbedarf der Öffentlichkeit in Zukunft auf wirtschaftlichere Weise decken lassen.

Das Konzept, das UIS in Richtung eines verteilten Systems auf Basis des WWW weiterzuentwickeln, dient auch der Kooperation über die Landesgrenzen hinaus. Mit dem Bundesministerium für Umwelt, Naturschutz und Reaktorsicherheit (BMU) und dem Umweltbundesamt (UBA) wird aufbauend auf die GLOBUS-Ergebnisse das F+E-Vorhaben Hypermediatechnik für Umweltdaten (HUDA) durchgeführt. Daneben steht die Verbesserung der Zusammenarbeit mit dem kommunalen Bereich im Vordergrund. Aufgrund der vielfältigen Kompetenz- und Zuständigkeitsverflechtungen im Umweltbereich zwischen dem Land und den Kommunen ist dies eine wichtige und komplexe Aufgabe. Die GLOBUS-Ergebnisse werden im neuen IuK-Verbund-Vorhaben Informationssystem Wasser, Abfall, Altlasten, Boden (WAABIS) verwendet.

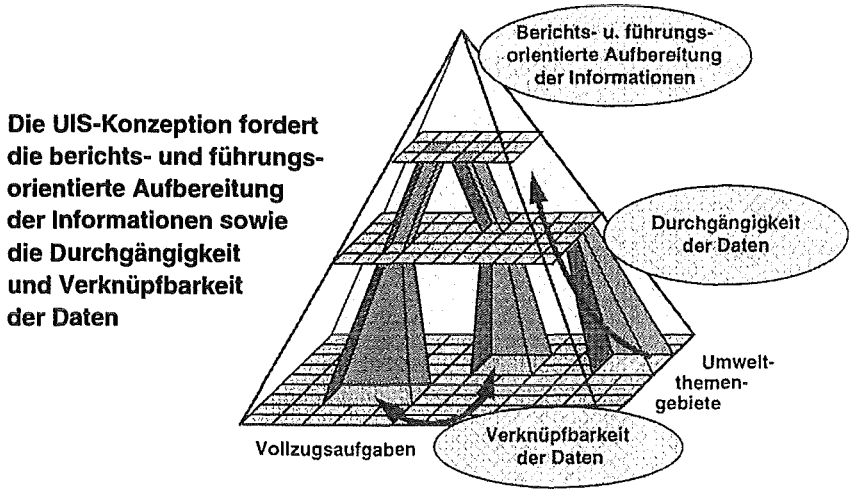


Abbildung 4: Paradigmen der Rahmenkonzeption des UIS Baden-Württemberg.

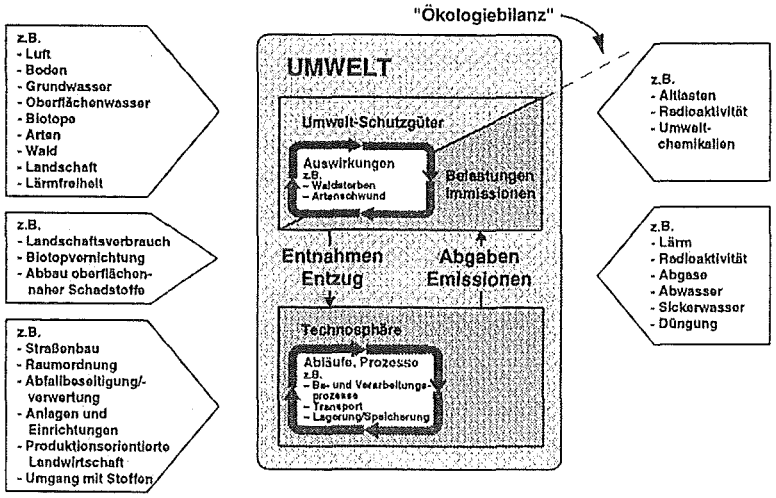


Abbildung 5: Das Ökologiemodell

Mit der Erweiterung des Funktionsumfangs und der Einbindung weiterer, bislang nur lose angebundener Anwendungen in die Gesamtstruktur wird sich das Potential des UIS Baden-Württemberg weiter vergrößern und sein Nutzen damit weiter zunehmen. Die Paradigmen der Rahmenkonzeption des UIS-BW (s. Abbildung 4) haben weiterhin Bestand. Teile des Ökologiemodells (s. Abbildung 5) können mit den GLOBUS-Ergebnissen erstmals bedarfsorientiert realisiert werden.

8. Literatur:

/1/ Birn, H./ Kaufhold, G./ Keitel, A./ Mayer-Föll, R. (1994): Umweltinformationssystem Baden-Württemberg - Die Konzeption, in: ONLINE Nr 10-1994, DATACOM-Verlag, Stuttgart.

/2/ Birn, H./ Radermacher, F.J./ Schmidt, F.(1993): Das Umweltinformationssystem Baden-Württemberg (UIS) als kooperatives und integrierendes System: Stand und Ausblick, in: Jaeschke A. et al. (Hrsg.), Informatik für den Umweltschutz, Springer-Verlag.

/3/ Bußmann, M./ Heißler, W./ Henning, I./ Müller, M. (1994): Umweltinformationssystem Baden-Württemberg - Die Konzeption, in: ONLINE Nr 11-1994, DATACOM-Verlag, Stuttgart.

/4/ Ebbinghaus, J./ Grünreich, D. et al. (1996): Fachdatenintegration in ATKIS für das Umweltinformationssystem Baden-Württemberg - Abschlußbericht, Ministerium für Umwelt und Verkehr Baden-Württemberg

/5/ Günther, O. (1995): Gutachten zur Entwicklung des Umwelt-Datenkatalogs (UDK), Umweltministerium Niedersachsen, Hannover.

/6/ Hess, G./ Schultze A. (1995): Dokumentation des UIS-Workshops am 15.5.1995 im Umweltministerium Baden-Württemberg, Umweltministerium Baden-Württemberg, FAW Ulm.

/7/ Keitel, A./ Müller, M. (1995): Die Integration von Sachdaten, Geodaten und Metadaten im Umweltinformationssystem Baden-Württemberg, Landesanstalt für Umweltschutz, Karlsruhe.

/8/ Mack, J./ Page, B. (1996): Zum Stand der Umweltinformationssystem-Entwicklung auf Landes- und Bundesebene, in: Lessing, H./ Lipeck U.W. (Hrsg.), Informatik für den Umweltschutz, 10. Symposium Hannover 1996, Metropolis Verlag, Marburg

/9/ Mayer-Föll, R./ Strohm, J./ Schultze, A. (1996): Das Umweltinformationssystem Baden-Württemberg - Überblick Rahmenkonzeption, in: Lessing, H./ Lipeck U.W. (Hrsg.), Informatik für den Umweltschutz, 10. Symposium Hannover 1996, Metropolis Verlag, Marburg

/10/ Mayer-Föll, R./ Jaeschke, A. (Hrsg.) (1995): Projekt GLOBUS: Konzeption und prototypische Realisierung einer aktiven Auskunftskomponente für globale Umwelt-Sachdaten im Umweltinformationssystem Baden-Württemberg Phase II - 1995, Forschungszentrum Karlsruhe, Wissenschaftliche Berichte, FZKA 5700.

/10/ Mayer-Föll, R./ Schilling, P./ Weigert, D. et al. (1986): Konzeption für das Umweltinformationssystem des Landes Baden-Württemberg, Ministerium für Ernährung, Landwirtschaft, Umwelt und Forsten Baden-Württemberg.

/12/ Riekert, W.-F. (1995): Cooperative Management of Data and Services for Environmental Applications, GISI 95, Informatik aktuell, Springer.

/13/ Seggelke, J./ Lessing, H. (1996): Globales Umweltinformationsnetz: Eckpunkte, Chancen und Gefahren, in: Lessing, H./ Lipeck U.W. (Hrsg.), Informatik für den Umweltschutz, 10. Symposium Hannover 1996, Metropolis Verlag, Marburg

/14/ Umwelt- und Verkehrsinformationen Baden-Württemberg, WWW-Server: http://www.uis-extern.um.bwl.de/

/15/ Umweltministerium Baden-Württemberg (1995): Umweltinformationssystem Baden-Württemberg, Band 6 der Schriftenreihe Verwaltung 2000 des Innenministeriums, Stuttgart.

/16/ Umweltministerium Baden-Württemberg/ McKinsey & Company (1988-1990): Konzeption des ressortübergreifenden Umweltinformationssystems (UIS) im Rahmen des Landessystemkonzeptes Baden-Württemberg, 12 Bände, Stuttgart.

Dienste im Umweltinformationssystem Baden-Württemberg

Bericht der AG-Dienste

A. Koschel,
Forschungszentrum Informatik (FZI),
Haid-und-Neu-Straße 10-14,
76131 Karlsruhe

F. Schmidt, M. Schöckle,
Universität Stuttgart,
Institut für Kernenergetik und Energiesysteme (IKE),
Abteilung Wissensverarbeitung und Numerik, Pfaffenwaldring 31,
70550 Stuttgart

J. Strohm,
Forschungsinstitut für anwendungsorientierte Wissensverarbeitung (FAW),
Helmholtzstr. 16,
89081 Ulm/Donau

M. Tischendorf,
Arkusa GbR,
Brühlstr. 2,
74379 Ingersheim

1. Auftrag der AG Dienste

Ein zentrales Konstrukt in GLOBUS ist das Konstrukt „Dienst". Nachdem im Projekt GLOBUS II hierauf alternative Sichten entwickelt wurden, war es Aufgabe der AG Dienste, den Begriff des Dienstes im UIS weiter zu präzisieren und so darzustellen, daß die zugrundeliegende Idee und die daraus ableitbaren Folgerungen für das UIS leichter vermittelbar werden. Um dies zu erreichen, hat sich die AG Dienste mit folgenden Aufgabestellungen befaßt:

- Beschreibung von Diensten
- Klassifizierung von Diensten
- Erstellung einer Sammlung verfügbarer Dienste
- Integration der Anwendungsentwickler
- Kriterien für die Auswahl von Middleware
- Konsequenzen für die Architektur des UIS.

Die AG Dienste hat diese Fragen in insgesamt 7 Sitzungen diskutiert. Der vorliegende Bericht faßt die Ergebnisse dieser Diskussionen zusammen. Zunächst werden die Grundelemente eines verteilten UIS dargestellt und der Begriff „Dienst" eingeführt. Darauf aufbauend werden verschiedene Sichtweisen auf Dienste beschrieben und ein Architekturmodell für ein diensteorientiertes UIS vorgeschlagen. Als Middleware zur Realisierung des verteilten UIS wurde CORBA untersucht. Sowohl die theoretischen Analysen als auch die praktischen Erfahrungen aus verschiedenen GLOBUS-Projekten unterstützen diesen Ansatz. Anschließend werden die z.T. beträchtlichen Auswirkungen des Dienstekonzepts auf die Rahmenkonzeption des UIS diskutiert. Daraus leiten sich eine Vielzahl von Schlußfolgerungen und Empfehlungen für zukünftige Arbeiten im UIS ab.

In einem ergänzenden Bericht /1/ werden folgende Schwerpunkte detaillierter behandelt:

- Beschreibung von Diensten und ihrer Schnittstellen
  (Koordination: F. Schmidt),
- CORBA im UIS Baden-Württemberg: Einsatzmöglichkeiten und -beispiele
  (Koordination: A. Koschel),
- Auswirkungen des Dienstekonzeptes auf die Rahmenkonzeption des UIS
  (Koordination: J. Strohm).

Detailliertere Darstellungen finden sich ebenfalls in den Abschlußberichten der Projekte GLOBUS II /GLOBUS-II/ und GLOBUS III /GLOBUS-III/.

2. Grundelemente eines verteilten UIS

Das UIS Baden-Württemberg wurde als fach- und ressortübergreifendes Informationssystem konzipiert. Grundlegende Architekturmerkmale des UIS Baden-Württemberg bezüglich der Daten sind **Durchgängigkeit** (der „vertikale Zugriff" auf Daten innerhalb der Verwaltungs- und Systemhierarchie) und **Verknüpfbarkeit** (die Möglichkeit „horizontaler Verschneidungen" von Daten gleicher Aggregationsstufen). Die Verknüpfbarkeit spiegelt vor allem den fach- und ressortübergreifenden Charakter von Umweltaufgaben wider. Die Durchgängigkeit

der Daten in der Systemarchitektur soll mittelfristig ermöglichen, daß Führungsinformationen für die Ministerien bzw. Regierungspräsidien weitgehend ohne manuelle Eingriffe direkt aus den Primärdaten, wie sie bei den Fachdienststellen vorliegen, erzeugt werden können.

Um Durchgängigkeit und Verknüpfbarkeit zu erreichen, werden Daten, Methoden und Metadaten im UIS in getrennten Systemen zur Verfügung gestellt. Man unterscheidet dabei drei Systemkategorien:

- Basissysteme,
- Grundkomponenten und
- übergreifende UIS-Komponenten.

Um die Integration dieser Komponenten informationstechnisch zu unterstützen, wurde im Projekt INTEGRAL das Konzept der externen Dienste entwickelt. „Externe Dienste" im Sinne des Projektes INTEGRAL waren abgeschlossene Einheiten, die eine spezielle Aufgabe erfüllen. Sie werden von Diensteanbietern erbracht und von Dienstenutzern verwendet. Der Diensteanbieter ist in der Regel ein Spezialist, der in der Lage ist, den Dienst sachgerecht einzusetzen, zu warten und entsprechend dem Stand der Wissenschaft und der Gesetzgebung weiter zu entwickeln. Aus dieser Sicht auf Dienste war es sinnvoll, die Ergebnisse einer Diensteinanspruchnahme als Dokumente zu verschicken. Im Projekt INTEGRAL bot es sich daher an, WWW-Technologie als integrierende Komponente zu verwenden, das zum einen HTML-Standards für die Diensteinanspruchnahme und die Übermittlung der Ergebnisse zur Verfügung stellte und zum anderen eine Lösung für die Kommunikation zwischen Klienten und Servern anbot. In diesem Stadium stellen Dienste also primär ein Konzept zur Integration von Anwendungen dar /2/.

In den vergangenen Jahren haben sich sowohl die Leistungsfähigkeit der Hardware (z.B. Workstation PC auf Basis von Pentium Prozessoren) und der Nutzung von Netzen (z.B. INTERNET und WWW), als auch die Mächtigkeit der Softwarekomponenten zur Implementierung verteilter Systeme (DCE, CORBA, JAVA) gewaltig gesteigert. Dies erlaubt es, Dienste immer feinerer Granularität zuzulassen, um schließlich Baukastensätze zu entwickeln, aus denen sich Dienste und Anwendungen zusammensetzen lassen. Diese Idee wurde im Rahmen des UIS erstmals im Projekt WWW-UIS in einer WWW und CORBA-basierten Architektur umgesetzt /4/. Dadurch traten neben den Integrationsaspekt in verstärktem Maße auch der Baukastenaspekt. Beide Aspekte lassen sich aus Grundprinzipien des Softwareengineerings wie Abstraktion und Kapselung ableiten. Da sie aber verschiedene Sichten repräsentieren, sind bei ihrer Umsetzung verschiedene Schwerpunkte zu setzen. Trotzdem ist es hilfreich, beide Aspekte gleichzeitig zu betrachten, da die Übergänge in der Regel fließend sind.

2.1 Das UIS als verteiltes System

Die frühzeitige Beschäftigung mit der Integration verteilter Komponenten zusammen mit der rasanten technologischen Entwicklung haben dazu geführt, den Versuch zu unternehmen, das UIS-BW nicht nur als ein fach- und ressortübergreifendes Informationssystem zu konzipieren, sondern mit ihm auch die in der Landesverwaltung verfügbaren Kompetenzen zur Lösung von Aufgaben mit Umweltbezug zur Verfügung zu stellen. Das kann nur gelingen, wenn diese Kompetenzen lokal weiterentwickelt und gleichzeitig in Form von Dienstangeboten landesweit zur Verfügung stehen. Das UIS-BW muß daher als ein verteiltes System konzipiert und

entwickelt werden. Das Dienstekonzept versucht dafür einen Rahmen vorzugeben.

2.2 Dienste im UIS

Die Umsetzung des Konzeptes erfordert zunächst die Identifikation abgeschlossener Aufgaben, die von einem Diensteanbieter entwickelt, gewartet und angeboten werden und die so interessant, effizient und leistungsfähig sind, daß Dienstenutzer sie zur Erfüllung ihrer Aufgaben heranziehen. Das Spektrum solcher Dienste ist sehr groß. Es reicht von **fachspezifischen Diensten** über **systemnahe Dienste** bis hin zum lokalen Einsatz von Standardwerkzeugen etwa zur Bürokommunikation oder der Darstellung von Ergebnissen im Kontext geographischer Informationen. Um diese Vielfalt in den Griff zu bekommen, wurden in der AG Dienste folgende Vereinbarungen getroffen:

- Schwerpunkt der Untersuchungen ist der Bereich der systemnahen Dienste. Auf sie treffen die Aussagen der folgenden Kapitel besonders zu.
- Fachspezifische Dienste wie etwa die Bereitstellung von Informationskatalogen, das Sammeln und Bewerten von Daten oder das Verfügbarmachen von Auswerte- und Simulationsmethoden werden über die von dem jeweiligen Dienst angebotenen Schnittstellen eingebunden. Es werden dazu zum einen Empfehlungen über die Gestaltung solcher Schnittstellen gemacht und zum anderen an UIS-typischen Beispielen gezeigt, wie diese Einbindung über Middlewareprodukte geschehen kann (siehe dazu Abschnitt über CORBA in diesem Bericht).
- Standardwerkzeuge, die von den Nutzern aufgabenorientiert eingesetzt werden, sollten lokal zur Verfügung stehen und entsprechend den Anforderungen der Nutzer mit fachspezifischen Methoden erweitert werden.

2.3 Systemnahe Dienste

Systemnahe Dienste sind funktionale Einheiten, die von den Systemmanagern gepflegt und Anwendungsentwicklern genutzt werden. Sie garantieren die Erfüllung der Grundaufgaben des UIS, können in Anwendungen genutzt werden und ermöglichen den Zugang zu fachspezifischen Diensten. Beispiele sind Dienste für Navigation, Datenzugriffe, Kommunikation oder Darstellung in Form von Karten. Systemnahe Dienste lassen sich in der Regel als Objekte oder Module realisieren.

2.4 Middleware

Middleware ist eine Softwareschicht, die die Verbindung zwischen Klienten und Diensten herstellt, den Entwickler davor bewahren soll, Unterschiede in den Kommunikationsprotokollen, den Betriebssystemen und den Hardwareplattformen beachten zu müssen und die darüber hinaus möglichst viele der Aufgaben erledigt, die erfüllt werden müssen, wenn mehrere Klienten auf denselben Dienst zugreifen bzw. zwischen alternativen Diensten wählen können.

Im UIS werden zwei Arten von Middleware eingesetzt:

- Dokumentenorientierte Middleware in Form des Kommunikationsinterpreters KIP und des WWW.

- Objektorientierte Middleware in Form von Object Request Brokern.

Die Middleware des UIS sollte folgenden Anforderungen genügen (Kurzliste, Begründung in /1/):

- Offenheit entsprechend den Richtlinien des Landessystemkonzeptes,
- Mächtigkeit zur Nachbildung der komplexen Daten- und Metadatenstrukturen des UIS,
- Sicherheit der Datenübertragung sowohl im technischen als auch im datenschutzrechtlichen Sinn,
- Plattformunabhängigkeit zur Einbindung existierender Systeme in das Gesamtsystem,
- Mechanismen zur Einbindung existierender Anwendungen wie Kapselung, asynchrone Anbindung und Transfer von Massendaten,
- Komponenten zum Management des verteilten Systems wie Session, Transaktions- oder Nutzermanagement und Protokollfunktionen.
- Broker- und Traderfunktionalitäten.
- Die Unterstützung von Maßnamen zur Sicherheit wie Authentifizierung, Autorisierung oder Verschlüsselung.

2.5 Klienten im UIS

Aufgabe des Klienten ist es, dem Nutzer eine spezifische Sicht auf das UIS zu erlauben. Er unterstützt den Nutzer beim Auffinden der für seine Aufgabe relevanten Informationen, hilft diese aus lokalen oder entfernten Quellen zu beschaffen und ermöglicht es, die beschafften Daten und Informationen lokal zu verwalten, mit problemspezifischen Methoden zu bearbeiten und zu neuen Informationen zusammenzufügen. Der Klient enthält dazu neben einem Browser zum Auffinden geeigneter Informationen über Kataloge (etwa den UDK) oder einen Gazetteer[1] drei Klassen von Komponenten: Repositories mit Daten und arbeitsplatzspezifischen Methoden, Werkzeuge zur Handhabung der Informationsobjekte sowie ein Management zur Verwaltung von Werkzeugen und Methoden und ihrer Zuordnung zu den vom Klienten bearbeitbaren Datentypen. Wichtig ist dabei die klare Trennung von Werkzeugen, Daten und ihrer Verknüpfung über arbeitsplatzspezifische Methoden. Erst sie erlaubt es, den Klienten nutzerspezifisch zu konfigurieren und ermöglicht dadurch die Dienste des UIS projektspezifisch optimal einzusetzen.

3. Logische und physikalische Sicht auf Dienste

3.1 Beschreibung von Diensten

Dienste sind definiert als abgeschlossene Einheiten (funktionale Komponenten), die eine für sie charakteristische Aufgabe erfüllen. Sie werden durch ihre Schnittstelle, ihre Funktionalität und evt. durch die Parameter, die die Bedingungen ihrer Nutzung beschreiben, definiert. Man

[1] Unter einem Gazetteer versteht man ein System, das die Auswahl von Datenobjekten über die Angabe von Ortsbezeichnungen oder Ortskoordinaten ermöglicht. Dabei werden insbesondere Zusammenhänge zwischen Ortsbezeichnungen in einer thesaurusartigen Datenstruktur modelliert und verwaltet.

kann Dienste auf die verschiedensten Arten klassifizieren. Beispiele sind ihre Herkunft, ihre Granularität, ihre Funktionalität, ihre Verfügbarkeit oder die Art und Menge der Ergebnisse ihrer Inanspruchnahme. Systemnahe Dienste sind Dienste, die der Zuständigkeit des Systemverwalters obliegen, die meist selber aus Basisdiensten zusammengesetzt sind, also eine mittlere Granularität haben, häufig auftretende Aufgaben wie etwa Selektion von Daten erfüllen und in der Regel ein einziges, definiertes Ergebnis liefern.

Für die Auswahl von Diensten ist die Beschreibung der Dienste und ihrer Schnittstellen erforderlich. Durch die Beschreibung wird es möglich, aus einem Diensteangebot einen gewünschten Dienst auszuwählen und ihn mit den richtigen Eingabedaten aufzurufen und die Ausgabedaten richtig zu interpretieren. Die Beschreibung erfolgt über Metainformationen. Es werden vier Typen von Metainformationen unterschieden:

- syntaktische Metainformation
  (Beschreibung der Typen von Daten oder die Aufrufsyntax von Diensten)
- semantische Metainformation
  (Beschreibung der Ein- und Ausgabedaten und der Funktionalität eines Dienstes)
- navigatorische Metainformation und
  (Beschreibung eines Zugangs zu Daten oder zu einem Dienst)
- regelbasierte Metainformation
  (Beschreibung von Beziehungen zwischen Daten und Diensten).

Mit diesen Metainformationen werden folgende Aufgaben erfüllt:

- Beschreibung der Parameter eines Diensteaufrufes nach Datentyp (Syntax) und nach Attributen (Semantik).
- Beschreibung der Funktionalität eines Dienstes durch Angabe des Funktionsaufrufes und der attributierten Funktionsbeschreibung.
- Beschreibung eines Zugangs zu einem Dienst und die Angabe diensteübergreifender Attribute etwa zur Bestimmung des Quality of Service.

Syntaktische Beschreibungen können durch oder in Anlehnung an die Interface Definition Language von CORBA erfolgen. Zur semantischen Beschreibung wurde eine Meta Data-Definition Language auf Basis der Standard Generalized Markup Language (SGML) vorgeschlagen. Somit steht eine Beschreibung für den Kontext von Daten und Diensten zur Verfügung, die weitere Entwicklungen nicht ausschließt sondern erweitert werden kann. Diese Möglichkeit der Erweiterung ist wichtig, da weder eine vollständige Beschreibung des Kontextes möglich, noch eine Erstellung eines Kontextes, der alle zukünftigen Anforderungen erfüllt, zu leisten ist.

3.2 Vermittlung von Diensten

Dienste können auf verschiedene Arten bekannt gemacht werden. Dazu gehören die Aufnahme in System- und Klassenbibliotheken, die Eintragung in Kataloge oder die Anmeldung bei Tradern oder Strategiekomponenten. Die Aufnahme in Bibliotheken ist das probate Mittel für Basis- und systemnahe Dienste. Anwenderdienste sollten in Katalogen bekannt gemacht und durch Trader vermittelt werden (Vermittlung alternativer Dienste ohne Berücksichtigung der Semantik). Zur sicheren Benutzung eines verteilten Systems ist die zusätzliche Berücksichti-

gung der Semantik aber unerläßlich. Für eine Zuordnung von Diensten zu einer Problemstellung muß die gesamte Dienstebeschreibung ausgewertet werden. Dies kann über eine Strategiekomponente erfolgen. Sie muß nicht nur Zugang zu aktuellen Metainformationen über die im System verfügbaren Dienste haben, sondern auch in der Lage sein, aufgrund einer Nutzeranfrage den zur Zeit geeignetsten Weg zur Lösung eines Problems zu erkennen, die dazu nötigen Dienste zu vermitteln und dabei sicherstellen, daß die für das System gültigen Sicherheitsanforderungen nicht verletzt werden. Dazu ist ein enges Zusammenspiel mit der Middleware notwendig.

3.3 Architekturmodell für das diensteorientierte UIS

Die Architektur moderner, verteilter Systeme wird durch die drei Schichten Klient - Middleware - Dienst beschrieben. Der Ansatz des diensteorientierten Systems versucht in seiner Sichtweise, die bestehende Kommunikationsinfrastruktur, die Anwendungslandschaft wie auch die Sicht auf die Bürokommunikationsumgebung der Nutzer des UIS zu integrieren. Damit geht das Dienstekonzept über eine Middleware- oder Protokollkonzeption hinaus und bildet einen umfassenden Rahmen für die zukünftige technische Entwicklung des UIS. Abb. 1 illustriert aus Nutzersicht einen 3-Ebenen-Aufbau des Dienstekonzeptes.

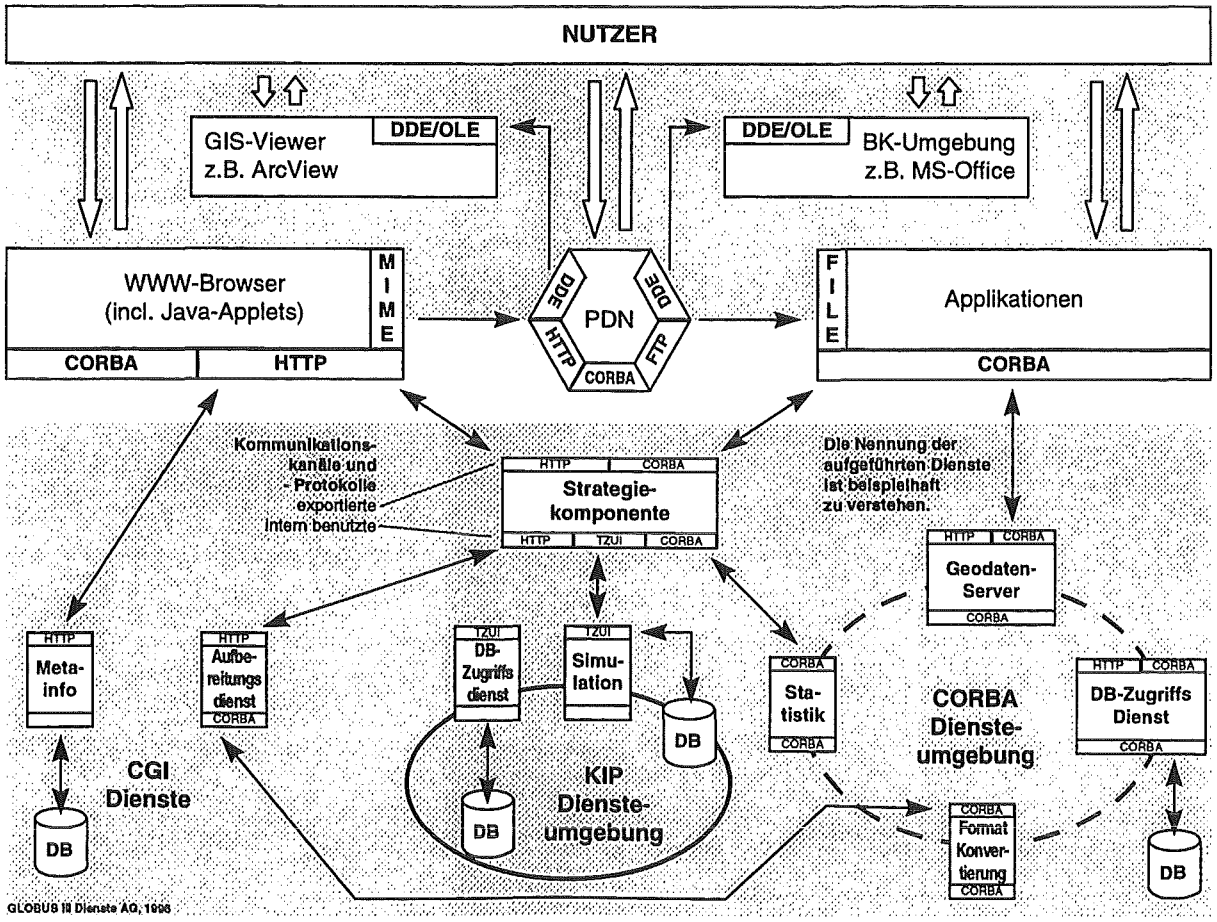


Abb. 1: Technische Sicht des Dienstekonzepts

- Auf der unteren Ebene ist die Kommunikation zwischen entfernten Diensten für den Nutzer in der Regel transparent. Dabei greifen Dienste automatisch auf Daten oder Funktionalitäten innerhalb des Netzes (Intranet) zu, ohne daß der Nutzer dies wissen oder steuern müßte. Für diese Kommunikation benötigt man ein ausgereiftes und mög-

lichst objektorientiertes Middlewarekonzept. CORBA (Common Object Request Broker Architecture) stellt ein solches Middlewarekonzept zur Verfügung. Bestehende Middlewarekomponenten (KIP/TZUI) können dies, wo nötig ergänzen.

- Auf der oberen Ebene befindet sich die Bürokommunikationsumgebung des UIS-Nutzers. Hier werden letztlich die Daten und Informationen des UIS benötigt und weiterverarbeitet.
- Die mittlere Ebene schafft die Verbindung von den Serverdiensten der unteren Ebene zur BK-Umgebung des UIS-Nutzers. Diese Ebene muß so gestaltet sein, daß Dienste und Daten des UIS möglichst einfach und unkompliziert genutzt und die verwendeten Plattformen neutralisiert werden können. Das World-Wide Web (WWW) bietet sich mit seinen Möglichkeiten, als Grundlage für diese mittlere Ebene, an.

Zu beachten ist in Abb. 1 die Funktion der Strategiekomponente, der serverseitig eine zentrale Rolle in der Vermittlung von geeigneten Diensten an den Nutzer zukommt. Neben HTTP (als WWW-Protokoll) und CORBA spielt DDE/OLE als „Windows-Middleware" im Bereich der Bürokommunikationsumgebung eine große Rolle.

Die Architektur des Klienten PDN, der ein Arbeiten mit dieser Umgebung unterstützt, wird im Kapitel über externe Dienste (3.5) vorgestellt. Dienste, die nicht blockierend genutzt werden können, müssen außer den schon beschriebenen Eigenschaften Schnittstelle und Funktionalität auch Methoden anbieten, mit deren Hilfe Statusabfragen gemacht, Zwischenergebnisse erfragt oder Teilergebnisse zum Klienten oder andere Dienste übertragen werden können. Basis solcher Methoden kann ein dem Dienst zugeordnetes Repository-Objekt sein, das bei Bedarf in der Lage sein muß, das Dienstobjekt zu überleben (Persistenz).

3.4 Implementierung von Diensten

Dienste können auf verschiedene Arten implementiert werden. Beispiele sind Objekte, Module, Systeme. Dabei ist es möglich, daß Dienste andere Dienste nutzen (hierarchischer Aufbau). Für systemnahe Dienste empfiehlt die AG Dienste, eine Implementierung in Form von CORBA-Objekten.

4. CORBA als Middleware-Komponente für das UIS

Bezugnehmend auf die im vorangehenden Abschnitt dargestellten Anforderungen an Middleware-Komponenten für das UIS, wird hier CORBA als Middleware-Layer im Rahmen der Dienstearchitektur des UIS in Form eines kurzen Überblicks betrachtet und begründet, warum die Dienste-AG die Verwendung von CORBA als Middleware empfiehlt.

Die Common Object Request Broker Architecture *CORBA* /6/ der Object Management Group (OMG), ist ein breit unterstützter *offener* Industriestandard einer Organisation von inzwischen über 700 DV-Herstellern und -Anwendern. CORBA erfüllt wesentliche Anforderungen, um als Middleware Layer für verteilte, heterogene Informationssysteme, wie das UIS, zu dienen. CORBA dient zur Interoperabilität verteilter Komponenten, dargestellt in Form von Objekten. Im Rahmen der sogenannten Object Management Architecture (OMA) der OMG beinhaltet der Standard folgende Teile: Dies ist zunächst der CORBA-Kern, der sogenannte Object Re-

quest Broker (ORB), der u.a. den lokalisierungsstransparenten Zugriff auf Objekte in verteilten Umgebungen ermöglicht. Er ist unabhängig von Systemplattformen und Programmiersprachen definiert und somit gut geeignet die technische Heterogenität im UIS zu verdecken. Ferner werden sogenannte CORBAservices standardisiert, die allgemein verwendbare Dienste wie z.B. Persistenz und Transaktionen darstellen und darauf aufbauend sogenannte CORBAfacilities die höhere Dienste wie Benutzerschnittstellen und Informationsverwaltung oder auch Dienste für bestimmte Anwendungsfelder (CIM, Finanzdienste, Medizinbereich, etc.) definieren. Da CORBAfacilities noch kaum verfügbar sind, werden sie hier nicht weiter betrachtet.

4.1 UIS-relevante Eigenschaften des CORBA Kerns

Bezugnehmend auf die Anforderungen des UIS an eine Middleware-Komponente im vorangehenden Kapitel sind folgende Eigenschaften des CORBA-Kerns besonders bedeutsam:

- Modulare Nutzbarkeit von Systemkomponenten: Zur Verdeckung von Verteilung und Heterogenität von Informationsquellen, wird in CORBA die Schnittstelle zu Objekten mit allen Operationen und Attributen über die CORBA-eigene objektorientierte Schnittstellenbeschreibungssprache IDL definiert. Der Zugriff auf Objekte wird somit indirekt über den ORB ausgeführt, und zwar über eine wohldefinierte Schnittstelle (Kapselungsprinzip). Durch diese Eigenschaften können Dienste im UIS als CORBA-Objekte in standardisierter Form beschrieben werden und problemlos Nutzern und Entwicklern zur Verfügung gestellt werden.
- Weiterentwicklung: Zur Weiterentwicklung von Systemkomponenten bietet CORBA die Mittel der Objektorientierung, also u.a. Kapselung und Vererbung für spezialisierte (weiterentwickelte) Objekte oder Austauschbarkeit von Objektimplementierungen bei gleichbleibender Schnittstelle.
- Offenheit: CORBA, CORBAservices und CORBAfacilities sind nicht-proprietäre, offene Industriestandards, die durch die Object Management Group definiert werden.
- Plattformunabhängigkeit: CORBA ist plattformunabhängig definiert, d.h. abstrahiert von Betriebssystemen, (Netzprotokollen) und Programmiersprachen. Für CORBA-IDL sind eine Reihe von Bindungen an konkrete Programmiersprachen definiert (derzeit ADA, C, C++ und Smalltalk, COBOL und Java). CORBA Implementierungen existieren von einer Reihe von Herstellern für alle gängigen UIS-Plattformen.
- Transparenz: CORBA-Objekte können in jeder Programmiersprache für die es eine IDL Abbildung gibt implementiert werden (bzw. mittels Wrapper gekapselt, s.u.). Ein Aufrufer braucht nicht zu wissen, auf welchem Rechner sich eine Objektimplementierung befindet, sondern kann über den ORB ortstransparent auf Objekte zugreifen.
- Kapselung: Ein wesentliches Ziel beim Entwurf von CORBA war die Möglichkeit des Einbindens bestehender Software-Systeme. Hierfür zentral ist das Konzept der *Wrapper* (Kapseln) für nicht CORBA Komponenten, die eine Schale mit definierten Schnittstellen um diese Komponenten legen und damit zumindest einen syntaktisch einheitlichen Zugriff (via IDL) auf diese ermöglichen. Das Wrapper-Konzept selbst ist unabhängig von CORBA. Nicht Bestandteil von CORBA - auch künftig im Rahmen sogenannter CORBAfacilities nur für Spezialfälle - ist hingegen der Aufbau solcher Wrapper und die Semantik hinter ihren Schnittstellen. Das Wrapperkonzept stellt eine gute Möglichkeit dar, die Nutzbarkeit bestehender UIS-Komponenten auch in der neuen Architektur zu gewährleisten. Aus diesem Grund stellen Wrapper einen Schwerpunkt bis-

heriger Arbeiten zu CORBA im UIS dar.

- Verteilte Systeme: Die Verteilungstransparenz für Objekte ist wesentlicher Kernbestandteil von CORBA und erfüllt damit eine wichtige Anforderung des UIS.
- Datenmengen, Performance: Im Hinblick auf die Performance in einem CORBA-basierten verteilten System, wurde bereits in der Evaluierung für das UIS in GLOBUS II festgestellt, daß CORBA selbst keinen grundsätzlichen Leistungsengpaß in einer WWW-orientierten Umgebung, wie sie das UIS darstellt, bedeutet. Dies gilt zumindest solange nicht ständig feingranulare Objektzugriffe erfolgen. Im UIS-Umfeld ist dies bisher typischerweise nicht der Fall, sollte aber bei Entwicklungen (z.B. von Wrappern) stets berücksichtigt werden.
- Interoperabilität, Anbindung an BK-Systeme (unter Windows): Mit der aktuellen Version 2.0 von CORBA wurde das Internet Inter Orb Protocol (IIOP) standardisiert, das zur Interoperabilität von ORB's verschiedener Hersteller dient, insbesondere auch für Java-basierte ORBs (in Applets). D.h. hierdurch ist ein Verbindungsprotokoll zu WWW-Browsern gegeben, das im Gegensatz zu HTTP nicht mehr zustandslos ist. WWW-Browser wie Netscape, wollen IIOP künftig sogar direkt (allerdings nur als Klient) unterstützen. Ebenfalls durch eine Reihe CORBA-Produktherstellern wird eine - allerdings derzeit i.w. herstellerspezifische - CORBA-Integration mit den Microsoft OLE bzw. COM Protokollen angeboten, d.h. bspw. ein Zugriff aus Visual Basic heraus auf CORBA-Objekte ermöglicht. Von der OMG wurde kürzlich ebenfalls eine Schnittstelle zu COM definiert.
- Sessionmangement: CORBA unterstützt zwar direkte, also zustandsbehaftete Verbindungen von Klienten zu Server-Objekten, jedoch von sich aus noch kein Sitzungsmodell. Eine relativ primitive Form der Session-Verwaltung läßt sich derzeit durch die Verwendung sog. „unshared" Server-Objekte zur Diensteimplementierung erreichen. Bei diesen wird für jeden Klienten bei Benutzung eines Dienstes ein eigenes Server-Objekt erzeugt, das erst wieder gelöscht wird, wenn der Klient meldet, daß er diesen Dienst nicht weiter in Anspruch nehmen will. Ferner ist es natürlich möglich für Komponenten eine eigene Session-Verwaltung zu entwickeln. Dies bedeutet jedoch ggf. einen erheblichen Entwicklungsaufwand. Im Rahmen der CORBAfacilities könnte ein Sessionmanagement definiert werden, derzeit ist hier jedoch eine UIS-spezifische Lösung notwendig.

4.2 UIS relevante Eigenschaften der CORBAservices

CORBAservices stellen Hilfsmittel für die Implementierung von Objekten einer verteilten Anwendung dar. Zu ihrer Nutzung erben Objekte die Schnittstelle von CORBAservice Objekten. Es existieren inzwischen eine Reihe von implementierten Services, weitere sind in der Spezifikationsphase. Im folgenden werden exemplarisch die CORBAservices Transactions/Concurrency, Security und Trading vorgestellt, die in verschiedenen ORBs schon (teilweise rudimentär) implementiert wurden. Eine Beschreibung aller CORBAservices findet sich in der CORBA Übersicht des GLOBUS III Dokuments /GLOBUS-III/. Dort wird auch auf den aktuellen Status der Implementierung der CORBAservices eingegangen.

Der Object Transaction Service (OTS) erweitert die Transaktionssemantik auf verteilte, objektorientierte Systeme. Hierzu stellt der OTS eine Schnittstelle bereit, um in einer verteilten Umgebung flache und geschachtelte Transaktionen nach dem ACID Paradigma zu *verwalten*. Natürlich müssen die an einer Transaktion beteiligten Objekttypen diese Schnittstelle für ihren

Typ selbst implementieren. Der Concurrency Service verwaltet konkurrierende Zugriffe auf Objekte durch Vergabe von Sperren, so daß der Zustand der Objekte konsistent bleibt. Er kooperiert hier mit dem OTS. Auch für diesen allgemein gehaltenen Service gilt, daß die Schnittstellenfunktionalität vom Objekttypen, der ihn verwendet implementiert werden muß. Bisher erfolgen im UIS im wesentlichen lesende Zugriffe auf Informationsquellen, so daß Transaktionen und Concurrency zum jetzigen Zeitpunkt noch keine große Rolle spielen. Dies könnte sich bei komplexen Folgen von Dienstaufrufen ändern, so daß dieser Service künftig für das UIS bedeutsam werden kann.

Der Trader Service ermöglicht das Auffinden und Registrieren von Objekten über Angabe von Attributen zur Beschreibung von Objekt-Eigenschaften oder -Fähigkeiten. Er stellt somit eine Art „gelbe Seiten" für Objekte im verteilten System dar. Der Trader Service ist gut geeignet, die steigende Zahl an Diensten im UIS zu verwalten. Durch Einsatz eines Traders zur Dienstevermittlung zwischen Klient und Server können bereits bei der Auswahl des diensterbringenden Servers bestimmte Kriterien berücksichtigt werden. Damit kann der Trader Service ein wesentlicher Bestandteil einer Strategiekomponente im UIS sein.

Sicherheit ist ein wichtiger Aspekt in verteilten Systemen. Unter Berücksichtigung der vielen bereits existierenden Sicherheitskonzepte, ist der Security Service als abstraktes Referenzmodell konzipiert, das einen Rahmen für die Einbindung verschiedener Sicherheitskonzepte realisiert. Der Security Service soll sowohl Sicherheitsmechanismen für Objekte, die sich dieser gar nicht „bewußt" sind bieten, als auch für Objekte, die Sicherheitsmechanismen durch eigene, anwendungsspezifische Funktionen aktiv kontrollieren. Hierbei ist zu beachten, daß durch verschachtelte Aufrufketten, wie sie in CORBA-basierten verteilten Umgebungen durchaus typisch sind, für den Aufrufer transparent Objekte auf „unsicheren" Systemen an der Funktionserbringung beteiligt sein können.

4.3 Bisherige Erfahrungen mit CORBA im UIS

Die Eignung von CORBA als Middlewarekomponente im UIS wurde bereits in der CORBA-Evaluierung und im Projekt WWW-UIS in GLOBUS II /GLOBUS-II/ untersucht. Dabei wurde die grundsätzliche Eignung von CORBA für das UIS durch eine praktische Erprobung mehrerer CORBA-Implementierungen und groben Leistungsmessungen in UIS-Szenarien herausgearbeitet. Insbesondere wurde festgestellt, daß viele der in den vorangehenden Abschnitten dargestellten UIS-relevanten Eigenschaften von CORBA in Form von Produktimplementierungen nutzbar sind. In den für GLOBUS III durchgeführten weiteren Teilprojekte zum Thema CORBA wurden darüber hinausgehend folgende Erfahrungen gesammelt:

- Das Kapseln und die Benutzung von bestehenden UIS-Diensten in einer CORBA-basierten Diensteumgebung durch Wrapper wurde anhand von Diensten unterschiedlicher Komplexität prototypisch und erfolgreich durchgeführt. Beispiele sind Datenbank-Zugriffsdienste und Simulationsdienste.
- Bei den Prototyp-Entwicklungen innerhalb der Teilprojekte erwies sich CORBA als ein sehr mächtiges, dadurch aber auch anspruchsvolles Werkzeug zur Implementierung von Diensten. In den verwendeten CORBA-Produkten wurden ferner für eine derartig neue Technologie erstaunlich wenige Software-Fehler festgestellt. Zu beachten ist ebenfalls, daß die Entwicklung der CORBA-Produkte derzeit noch sehr dynamisch ist und bisher noch kein Produkt sämtliche der spezifizierten CORBAservices enthält.

- Die gute Eignung von CORBA als Plattform zur modularen und verteilten Entwicklung von Diensten wurde unter anderem durch die getrennte Entwickung und nachträgliche Kombination zweier unterschiedlicher Dienste am FZI und IKE unterstrichen. Hierzu trägt besonders die Beschreibung von Schnittstellen mit CORBA-IDL bei.
- Die relativ problemlose Integration von Komponenten auf heterogenen UIS-Plattformen sowie das standardgemäße Zusammenspiel verschiedener CORBA-Implementierungen wurde gezeigt.

5. Auswirkungen des Dienstekonzeptes auf die Rahmenkonzeption des UIS

Das Dienstekonzept führt im UIS zu weitreichenden konzeptionellen Konsequenzen. Bislang besteht das UIS aus einer Reihe von Systemkomponenten, die sich zwar in ein gemeinsames Konzept einfügen, technisch betrachtet, jedoch weitgehend unabhängig voneinander sind. Konzepte wie die „Durchgängigkeit" und die „Verknüpfbarkeit" von Daten erfolgen entlang hierarchischer Strukturen und dienen in erster Linie der führungsorientierten Aufbereitung des Datenmaterials.

Dies kann sich durch das Dienstekonzept grundlegend ändern. Im Zuge seiner Umsetzung ist mit einer starken Verknüpfung einzelner Systemkomponenten in den unterschiedlichen UIS-Systemkategorien zu rechnen. Dies sowohl in technischer Hinsicht, als auch im Hinblick auf die übergreifende Nutzung der jeweiligen Informationsinhalte.

Durch diese enge Verzahnung der einzelnen Systemkomponenten entstehen Probleme vor allem organisatorischer Art, die in der Fortschreibung der Rahmenkonzeption des UIS angegangen werden müssen.

5.1 Managementaspekte des verteilten UIS

In der bisherigen Konzeption des UIS wurden Fragestellungen der Datensicherheit und des Datenschutzes größtenteils nur am Rande behandelt. Aufgrund des Umstands, daß vertrauliche Daten im wesentlichen in abgeschlossenen Fachsystemen vorgehalten wurden, war eine weitergehende Diskussion auch nicht nötig. In einem offenen und verteilten System, wie es das Dienstekonzept für das UIS vorsieht, stellt sich die Situation allerdings grundlegend anders dar, so daß eine ausführliche Behandlung dieser Thematik notwendig wird. Konkret stellen sich die Fragen, welches Datenmaterial in welcher Weise und in welchem Umfang bereitgestellt werden soll, wie die Kommunikation sicher gestaltet und der Zugang reglementiert werden kann und wie die Qualität der vorgehaltenen Daten und Dienste gesichert werden soll.

5.1.1 Datenbestände

Bislang werden Daten, die übergreifend genutzt werden sollen, in der Datenbank der übergreifenden Komponenten (DB-ÜKO) zentral abgelegt. Dieses Verfahren ist jedoch für die Bereitstellung von Daten aus Fachsystemen nur dann verwendbar, wenn ein relativ kleiner Anteil des Gesamtdatenbestands des Fachsystems übergreifend nutzbar gemacht werden soll. Müs-

sen hingegen größere Datenmengen bereitgestellt werden, sollte die Datenbank direkt verfügbar gemacht werden, um unnötige Redundanz in der Datenhaltung zu vermeiden. Entsprechendes gilt, falls Daten häufig aktualisiert werden müssen.

Neben der Spiegelung von Datenbeständen in größere, übergreifende Datenbanken und der Online-Anbindung von Produktionsdatenbanken, können Daten auch in bereits interpretierter Form als Berichte bereitgestellt werden. Dies empfiehlt sich insbesondere dann, wenn zur Interpretation der Daten große Mengen zusätzlicher Informationen notwendig sind und eine korrekte Interpretation des Datenmaterials nur durch Spezialisten erfolgen kann.

Datenbestände, die einer übergreifenden Nutzung zugeführt werden sollen, müssen stärker, als dies bisher der Fall ist, standardisiert sein. Neben der Standardisierung der mit den Daten zur Verfügung gestellten Metabeschreibung bezieht sich dies vor allem auf die standardisierte Angabe von geographischen Bezügen. Für ersteres sollte der erweiterte UDK (WWW-UDK), für letzteres der RIPS-Pool als Normierungsgrundlage herangezogen werden können. Neben den Rasterkarten, die aus RIPS heraus für die unterschiedliche thematische Anwendungen erzeugt werden können, müssen in Zukunft allerdings auch die ATKIS-Datenobjekte der Vermessungsverwaltung selbst, auf denen das RIPS-System basiert, direkt bereitgestellt werden können.

5.1.2 Nutzerverwaltung

Die UIS-Landschaft, im Rahmen des Dienstekonzepts, wird eine ganze Reihe von Datenbanken mit zugehörigen Zugriffsdiensten aufweisen. Soweit es sich hierbei um vertrauliche Daten handelt, muß der Zugang zu diesen Daten und Diensten auf einzelne Nutzer oder Nutzergruppen eingeschränkt werden können. Dies macht ein übergreifendes Konzept für die Nutzerverwaltung erforderlich. Insbesondere müssen Rollenmodelle für „typische" UIS-Nutzer entwickelt werden, die jeweils den Zugriff auf eine definierte Menge von Daten und Diensten gestatten. Der administrative Aufwand für die Pflege der Nutzerdaten ist in diesem Fall verhältnismäßig gering, da bei personellen Veränderungen lediglich die, den Nutzern zugeordnete, Rollenbeschreibung zu ändern ist und aufwendige Modifikationen einzelner Zugriffsrechte entfallen.

5.1.3 Sicherheitsmanagement

Das Nutzermanagement ist nur sinnvoll, wenn es eingebunden ist in ein übergreifendes Konzept des Sicherheitsmanagements. Durch Authentifizierungs- und Verschlüsselungskonzepte muß gewährleistet werden, daß nur berechtigte Nutzer Zugang zu den jeweiligen Daten erhalten. Die Authentifizierung sollte systemweit einheitlich unter Verwendung des Nutzermanagements erfolgen, so daß für die Nutzung unterschiedlicher Dienste nicht jeweils spezifische Authentifizierungsprozesse erforderlich sind.

Für jeden einzelnen Dienst, der auf u.U. vertrauliches Datenmaterial zugreift, muß gewährleistet sein, daß die Anforderungen des Datenschutzes hinreichend berücksichtigt werden.

Eine wichtige Einrichtung innerhalb eines Sicherheitsmanagementsystems sind s.g. Trust Center, die als vertrauenswürdige Instanzen für die Authentifizierung des Nutzers dienen. Ei-

genentwicklungen und -ansätze des UIS müssen hier jedoch mit der Stabsstelle Information und Kommunikation (SIK) im Innenministerium abgestimmt und mit dem Landessystemkonzept verträglich sein.

Grundsätzlich sollte beim Einsatz von Middlewarekomponenten immer darauf geachtet werden, daß ausreichende Konzepte für ein wirkungsvolles Sicherheitsmanagement vorliegen.

5.1.4 Metainformationen im UIS

Beschreibende und charakterisierende Information über Dokumente, Daten und Dienste sind entscheidend für die Nutzbarkeit des UIS-Informationspools.

Als zentrale Zugangskomponente findet der erweiterte Umweltdaten-Katalog (UDK) der Bund-Länder-Kooperation Verwendung. Der UDK ist aber in erster Linie als Verweissystem für den Bezug von Umweltdaten und Diensten gedacht und ist deshalb zur Aufnahme komplexer, formalisierter Metainformation, wie sie insbesondere für die Schaffung einer automatisierten Aufrufmöglichkeit für Dienste notwendig ist, wenig geeignet. Daher sind weitere Repositories notwendig, die solche komplexeren Metadaten bereitstellen können.

Darüberhinaus kann es auch sinnvoll sein, den UDK in speziellen thematischen Bereichen (z.B. Geodaten) durch weitere Repositories zu ergänzen. Die Einführung eines weiteren Repositories sollte jedoch nur dann erfolgen, wenn es aus fachlichen oder technischen Gründen notwendig und sinnvoll erscheint. Die integrierende Rolle des UDK sollte dadurch nicht gefährdet werden. Beim Aufbau dieser Repositories ist darüberhinaus darauf zu achten, daß redundante Datenhaltung nur im absolut notwendigen Maß erfolgt, um die Aufwendungen für die Konsistenzerhaltung und Pflege der Metadaten möglichst gering zu halten.

Außerdem ist davon auszugehen, daß in einer verteilten Systemarchitektur ein Repository auch mehrfach gespiegelt eingesetzt werden könnte, um beispielsweise Performance-Engpässe zu vermeiden. In der Konzeption von Repositories ist deshalb auch eine übergreifende Nutzung mehrerer solcher Metainformationssysteme vorzusehen.

Der Nutzwert von Metainformationssystemen für den UIS-Nutzer hängt stark von der Aktualität der vorgehaltenen Metadaten ab. Bei der erstmaligen Eintragung eines Datenbestandes oder eines Dienstes muß die zugehörige Metainformation vom Anbieter bereitgestellt werden. Für die Fortführung der Metainformation ist ebenfalls der Anbieter verantwortlich. Systemseitig kann die Aktualität der Metadaten nur eingeschränkt überprüft oder gewährleistet werden. So können Metainformationen im Fall regelmäßig aktualisierter Meßdaten bis zu einem gewissen Grad automatisch generiert werden, oder kann von Diensten die Bereitstellung ihrer zugehörigen Metainformation über spezielle Schnittstellen eingefordert werden. In der Regel erfordert die Erhaltung der Aktualität der Metadaten aber die Verantwortung und aktive Mitarbeit der jeweiligen Systemverantwortlichen. Um den mit dieser Pflegeaufgabe verbundenen Arbeitsaufwand möglichst gering und gleichzeitig den Aktualitätsgrad möglichst hoch zu halten, müssen den Anbietern von Daten und Diensten leistungsfähige und benutzerfreundliche Werkzeuge für die Pflege ihrer Metadaten zur Verfügung gestellt werden.

5.1.5 Dienste in der Anwendungsentwicklung

Aus der Sicht des Anwendungsentwicklers stellt sich die Frage, wie die Systemkonzeption einer Anwendung auf Basis des Dienstekonzepts grundsätzlich aussehen sollte. Für die einzelnen Komponenten einer Anwendung ist zu entscheiden, ob sie jeweils lokal implementiert oder per remote-Zugriff aus dem Intranet beschafft werden sollen. Aspekte, die diese Entscheidung beeinflussen können sind unter anderem:

- Wie spezifisch / allgemein ist die Funktion der fraglichen Komponente?
- Welche Performance wird benötigt?
- Soll die Anwendung im Standalone-Betrieb gefahren werden können?
- Wie aufwendig ist der Pflegeaufwand für (ggf. oft replizierte) lokale Implementierungen und deren Daten? Ist beispielsweise ein Java-Applet geeignet, daß zwar lokal abläuft, aber serverseitig administrierbar ist?
- Wie aufwendig ist die (dann ggf. wieder plattformspezifische) lokale Implementierung einer Komponente bzw. deren lokale Datenhaltung? Wie ist dies in ein Sicherheitskonzept integrierbar?
- Wie groß sind bei welcher Implementierungsart anfallende Lizenzkosten?
- Ist die Verfügbarkeit des Intranet-Dienstes hinreichend gesichert?

Der letztgenannte Aspekt ist aus zweierlei Gründen bedeutsam. Zum einen kann die Verfügbarkeit eines Dienstes durch eine unzureichende Intranetanbindung, eine leistungsschwache Servermaschine oder eine instabile Implementation gefährdet sein, zum anderen kann die langfristige Verfügbarkeit des Dienstes durch die Veränderung von Schnittstellen oder Funktionalitäten eingeschränkt werden. Um Anwendungsentwicklern in dieser Hinsicht Planungssicherheit zu geben, ist sicherzustellen, daß ein einmal im Rahmen des UIS angebotener Dienst in einer bestimmten Version solange verfügbar bleibt, wie Nachfrage nach dieser speziellen Dienstversion besteht.

Dienste, die von Stellen außerhalb des UIS angeboten werden, weisen hierbei eine besondere Problematik auf: Da die Entwicklung dieser Dienste außerhalb des UIS erfolgt, sind die Einflußmöglichkeiten des UIS auf die Weiterentwicklung und die Verfügbarkeit alter Versionen externer Dienste relativ gering. Bei der Verwendung externer Dienste innerhalb von UIS-Anwendungen muß daher grundsätzlich davon ausgegangen werden, daß die Verfügbarkeit dieses Dienstes nicht gesichert ist. Abhilfe könnte durch die Verwendung alternativer Dienste in Verbindung mit einer Strategiekomponente geschaffen werden.

Bei der Anwendungsentwicklung sollte grundsätzlich der Nutzung eines bestehenden Dienstes der Vorzug vor einer eigenen Neuentwicklung gegeben werden. Neuentwickelte Komponenten, die als allgemeine Dienstleistung im UIS von Interesse sein könnten, sollten von Anfang an so konzipiert und realisiert werden, daß eine Bereitstellung im Rahmen des Dienstekonzeptes mit geringem Aufwand möglich ist. Anwendungsspezifische Komponenten, wie die Navigation einer Anwendung, müssen allerdings von Fall zu Fall neu erstellt werden. Allerdings können u.U. „Standard-Softwareelemente“ zur Verfügung gestellt werden, die den Aufbau einer Anwendungsnavigation oder die Anbindung einer Datenbank effizient und standardisiert ermöglichen.

Besondere Anforderungen für die Integration von Diensten in Anwendungen ergeben sich bei

Fachsystemen. Besonders beleuchtet werden müssen hier Fragen der Sicherheit und der Performance. Letzteres ist im Einzelfall zu prüfen, für ersteres ist die Konzeption und Umsetzung von Sicherheitssystemen erforderlich.

5.1.6 Die Strategiekomponente

Für die Verknüpfung von Diensten gibt es mehrere Möglichkeiten. Im einfachsten Fall wird ein bestimmter Dienst fest in eine Anwendung eingebunden. Unter Umständen ist es jedoch günstiger, erst zur Laufzeit einen Dienst mit der gesuchten Funktionalität aus einer Menge gleichartiger Dienste nach bestimmten Kriterien auszuwählen. Die Verknüpfung zwischen den Anwendungskomponenten stellt dann eine Strategiekomponente her, die zu den einzelnen Diensten verfügbare Metainformation nach den vorgegebenen Regeln (regelbasierte Metainformation, aktive Mechanismen) auswertet.

Die Konstruktion einer solchen Komponente kann z.T. auf bestehende Entwicklungen im CORBA-Bereich zurückgreifen. Allerdings geht die für das UIS benötigte Vermittlungsfunktionalität über die üblichen Trader-Konzepte hinaus und stellt technisch und konzeptionell gesehen eine große Herausforderung dar (siehe hierzu /1/).

Sinnvollerweise wird sich die Entwicklung praxisorientiert in einzelnen Schritten vollziehen, da sich das Nutzungspotential einer Strategiekomponente erst in dem Maße vergrößert, in dem sich die Anzahl der verfügbaren Dienste im UIS erhöht. Auf der Ebene der Anwendungsentwicklung wird die Verknüpfung von Diensten daher zunächst direkt und fest erfolgen. Mit zunehmender Anzahl der verfügbaren Dienste wird der Bedarf für eine Strategiekomponente aber zunehmen. In der Konzeption der Strategiekomponente müssen allerdings frühzeitig Ansätze für die benötigten „intelligenten" Verfahren enthalten sein, um zu gewährleisten, daß das System auch später noch konzeptionell trägt.

Bei dem gesamten Entwicklungsprozeß müssen die Entwicklungen kommerzieller Anbieter kontinuierlich verfolgt und gegebenenfalls integriert werden. Dies allerdings unter Berücksichtigung des Landessystemkonzepts und des Architekturmodells für offene Systeme der Stabsstelle Information und Kommunikation im Innenministerium.

5.1.7 Der UIS-Arbeitsplatz

Der Arbeitsplatz des UIS-Nutzers unterscheidet sich wesentlich von dem bisher üblichen. Die einzelne UIS-Anwendung steht nicht mehr isoliert neben der Bürokommunikationsumgebung, sondern ist darin integriert. Die effiziente Weiterleitung von Informationen aus dem UIS in die Anwendungen der BK ist entscheidend für den Nutzwert dieser Information und ist daher eine wichtige konzeptionelle Vorgabe für die weitere Entwicklung von UIS-Komponenten.

Im Dienstekonzept wird eine effiziente Weiterleitung der Information in die BK des UIS-Nutzers durch die Ergänzung des Datenmaterials mit Metainformation erzielt. Durch sie wird auch die Konvertierbarkeit der erhaltenen Informationen in spezielle Formate der BK ermöglicht. Die Weiterverarbeitung der erhaltenen Daten in das gewünschte Format kann dabei wiederum von Formatierungsdiensten und Konvertierungsdiensten übernommen werden.

Eine wichtige Rolle für den Transport der Daten in die BK-Anwendungen des Nutzers spielt

der Personal Data Node (PDN). Diese UIS-eigene Windows-Anwendung realisiert eine Reihe von Werkzeugen, die die Weiterverarbeitung der Daten des UIS auf dem Klienten unterstützen. Neben der Ermöglichung komfortabler Dienstaufrufe durch eine nutzerspezifische Sichtenbildung und der automatisierten Konvertierung und Aufbereitung von Daten anhand vom Nutzer definierter Präferenzen, bietet der PDN auch lokale Zwischenspeicher und Repositories zur Ablage von arbeitsplatzspezifischen Methoden, Daten und Konfigurationsinformation.

5.1.8 Nutzbarkeit von Daten und Diensten

Entscheidend für die Nutzbarkeit der UIS-Information innerhalb einer komplexen Dienstelandschaft sind die folgenden Punkte:

- Ein **homogenisiertes Modell für Daten** und Metadaten innerhalb des UIS sowie die ausreichende Bereitstellung von Metainformation stellt die systemübergreifende Durchgängigkeit, Verknüpfbarkeit und korrekte Interpretation der Informationen sicher. Besonders wichtig ist hier ein einheitlicher Raumbezug der Umweltdaten.
- Eine **einheitliche Schnittstellenbeschreibung** der Dienste
- Die Gewährleistung der **Aktualität der vorgehaltenen Metadaten** stellt ein Problem dar, das in erster Linie organisatorisch gelöst werden muß, da die Aktualisierung der Metainformation technisch zwar unterstützt, aber nicht erzwungen werden kann.
- Eine **hohe Verfügbarkeit** der angebotenen Dienste ist wichtig, um die Lauffähigkeit einer Anwendung garantieren zu können, bei der wesentliche Funktionalitäten über die Nutzung von UIS-Diensten realisiert werden. Besonders wichtig ist dies bei Produktionsanwendungen, d.h. Fachsystemen, die sehr betriebssicher arbeiten müssen.

Diese vier Punkte sind bei externen Diensten, d.h. bei Diensten, die außerhalb des UIS stehen aber auch im UIS Verwendung finden können, besonders schwierig zu gewährleisten, da Regulatorien des UIS außerhalb des UIS nicht greifen können, bzw. eine Einflußnahme des UIS auf die Weiterentwicklung dieser Komponenten in der Regel nicht besteht. Externe Dienste sollten daher nur indirekt, d.h. über einen Adapterdienst (z.B. ein passender CORBA-Wrapper), in das UIS eingebunden werden, wobei die Aktualität der über den externen Dienst vorliegenden Metainformation regelmäßig überprüft werden sollte.

6. Schlußfolgerungen und Empfehlungen

6.1 Schlußfolgerungen

Die Umsetzung des Diensteansatzes führt zu einer neuen Qualität des UIS-Baden-Württemberg. Aufgaben mit Umweltbezug werden in allen Ministerien des Landes wahrgenommen. Dies hat zur Folge, daß sie mit heterogenen Methoden und verteilt, also an verschiedenen Orten, bearbeitet werden. Mit dem vorgestellten Dienstekonzept wird versucht, die in der Umweltverwaltung bewährten Strukturen zur Lösung der Aufgaben mit Umweltbezug zu nutzen, statt sie in einem übergeordneten Informationssystem aufzulösen. Damit wird es erstmals möglich, die Verwaltung auch bei der Lösung von Querschnittsaufgaben effektiv zu unterstützen. Der Ansatz ermöglicht aber auch eine Vielzahl von pragmatischen Vorgehensweisen,

durch die die Komplexität der Aufgabe, ein umfassendes Umweltinformationssystem zu bauen, zu warten und nutzbar zu machen, entscheidend erleichtert wird. Dazu gehören:

- Die Reduktion der Komplexität durch Kapselung einzelner Aufgaben in funktionale Komponenten.
- Die Erhöhung des Nutzungsgrades einzelner Komponenten durch eine Vereinheitlichung ihrer semantischen und syntaktischen Beschreibungen.
- Die Reduktion von Kosten durch Wiederverwendung funktionaler Komponenten beim Aufbau von Anwendungssystemen, durch Nutzung frei verfügbarer Internetsoftware und durch den Einsatz weit verbreiteter Software, die keine extra Schulung benötigt.
- Die Verbesserung der Zukunftssicherheit durch die Möglichkeit alternativer Bausteine (alternative Verfahren, Weiterentwicklungen) ohne größere Modifikation in bestehende Anwendungssysteme einzubinden.
- Die Erhöhung der Investitionssicherheit durch Nutzung der Internet-Technologie und weit verbreiteter kommerzieller Software.

Durch die bessere Integration des UIS in die gewohnte Bürokommunikationsumgebung der UIS-Nutzer läßt sich der notwendige Aufwand für Datenbeschaffung und -verarbeitung in der täglichen, praktischen Anwendung deutlich reduzieren. Damit ergibt sich, im Vergleich zum heutigen System, ein qualitativer und quantitativer Nutzeffekt, der dem Ziel einer schlanken und effizienten Verwaltung zugute kommen wird.

Die in Abb. 1 gezeigte Architektur dienstebasierter Systeme erlaubt es auch, existierende Programme und Verfahren einzusetzen bzw. einer allgemeineren Nutzung zuzuführen. Durch die Integration über Metainformationen und die Verwendung einer Strategiekomponente wird es außerdem möglich, auf feste Verdrahtung einzelner Anwendungen zu verzichten. Dies führt dazu, daß die Anwendungen flexibler aufgebaut, sicherer betrieben, leichter gewartet und effektiver erstellt werden können.

Das von der Dienste-AG ausgearbeitete Architekturkonzept eröffnet durch die Nutzung der Techniken und Werkzeuge des World-Wide Web, eine kostengünstige Zugangsmöglichkeit zu Umweltinformation für den Bürger. Dabei ist insbesondere zu betonen, daß die Landesverwaltung in Erfüllung dieser, gesetzlich festgelegten, Informationspflicht mit verhältnismäßig geringen Investitionen deutliche Fortschritte erzielen kann und daß, aufgrund des hohen Ausstattungsgrades der Bevölkerung mit Internet-Anschlüssen, auch ein weitaus größeres Spektrum der Bevölkerung direkt und zeitnah erreicht werden kann, als dies bisher der Fall war.

6.2 Empfehlungen

Aus den in diesem Dokument vorgestellten Überlegungen ergeben sich eine Reihe von Empfehlungen für die Weiterentwicklung des UIS.

6.2.1 Allgemeine Empfehlungen

Von allen Systemen, die Aufgaben der Middleware erfüllen, haben CORBA-basierte Objekt Request Broker zur Zeit das größte Entwicklungspotential. Von daher wird empfohlen, sich bei weiteren Entwicklungen an CORBA zu orientieren.

- Neue funktionale Komponenten sollten als CORBA-Objekte entwickelt werden.
- Die syntaktische Beschreibung der Schnittstellen von Diensten, die nicht als CORBA-Objekte zur Verfügung stehen, sollte ebenfalls über die CORBA-Interface Definition Language erfolgen.
- Zur semantischen Beschreibung kann der entwickelte Subset der SGML verwendet werden (siehe etwa Kap. 3.5 und /3/).
- Wo sinnvoll möglich, sollten CORBA-Services genutzt oder doch wenigstens ihre Spezifikation beachtet werden.
- Existierende funktionale Komponenten sollten über Wrapper angebunden werden.
- Neben der Verwendung von CORBA sollte es möglich bleiben, funktionale Komponenten alternativ einzubinden. Dafür kommt vor allem dokumenten-orientierte Middleware, wie sie schon bisher verwendet wird, in Frage.
- Die Möglichkeit, bei Bedarf funktionale Komponenten auf dem Klienten als Werkzeuge zu nutzen, sollte ausgebaut werden (Beispiele sind Bürokommunikation, Visualisierung, Spezialauswertungen). Insbesondere sollten für die Aufbereitung der Daten auf Klientenseite Systeme für die Verarbeitung von Entscheidungsregeln verfügbar gemacht werden.
- Die Auswahl der Dienste sollte durch eine Strategiekomponente unterstützt werden. Die dazu notwendigen Anforderungen sind zu spezifizieren.

6.2.2 Empfehlungen für die Systementwicklung

Das Potential des hier vorgestellten Diensteansatzes basiert auf drei Grundsäulen

- Bereitstellung neuer, bzw. Einbindung bestehender funktionaler Komponenten.
- Erleichterung des Einsatzes funktionaler Komponenten durch Standardisierung der Schnittstellen und der Beschreibung ihrer Funktion.
- Ermöglichung von Erweiterungen funktionaler Komponenten durch Techniken wie Vererbung oder Überladung.

Daraus ergeben sich folgende Empfehlungen für künftige Entwicklungen im UIS:

- Für die Datenobjekte des UIS sollten Basisdienste entwickelt und in Form von Bibliotheken bereitgestellt werden.
- Systemnahe Dienste sollten als Objekte zur Verfügung gestellt werden. Dies umfaßt insbesondere die Forderungen nach Kapselung und der Möglichkeit der projektbezogenen Erweiterbarkeit.
- Die Entwicklung von Konventionen zur Erstellung, Dokumentation und Validierung von Diensten sollte vorangetrieben und ihre Einhaltung verstärkt eingefordert werden.

6.2.3 Empfehlungen für Schwerpunkte künftiger Entwicklungen

Die in diesem Abschnitt aufgeschriebenen Empfehlungen sind Resultate aus den Überlegungen zum Diensteansatz. Sie sind duch die folgenden Feststellungen zu ergänzen.

- Der mit CORBA eingeschlagene Weg sollte weiterverfolgt werden. Entwicklungen bei den verfügbaren ORB's sollten verfolgt und bei Bedarf genutzt werden.
- Entwicklungen auf dem Markt der CORBA-Services sollten verfolgt und auf ihren

Nutzen für das UIS abgefragt werden.

Der Ansatz Strategiedienste projektbezogen entweder auf Klient- oder Serverseite einzusetzen, sollte weiterverfolgt werden. Er ermöglicht es, die zu transferierende Datenmenge zu steuern. Eine Technologie, auf deren Basis dies geschehen kann, ist die Verbindung von CORBA-Objekten und JAVA Applets.

- Wichtige systemnahe Dienste, die weiterentwickelt bzw. prototypisch realisiert werden sollten, sind
  - Verwaltungsdienste (Sessionmanagement, Nutzermanagement)
  - Repositorydienst
  - Strategiekomponente.
- In Abstimmung mit dem IKSUM und der LfU ist ein Sicherheitskonzept zu entwickeln und Vorschläge für seine Realisierung auszuarbeiten.
- Es sollte ein Dienst zur regelbasierten flexiblen Spezifikation allgemein verwendbarer aktiver Mechanismen bereitgestellt werden. Anwendungsbeispiele sind Metadatenfortschreibung, Benutzerinformationen über Änderungen von Basisdaten oder die Auswahl von Diensten durch die Strategiekomponenten.
- Anwendungen sollten verstärkt unter Nutzung von Basisdiensten und systemnahen Diensten entworfen und implementiert werden.
- Bei der Weiterentwicklung bestehender Anwendungen sollte die Kapselung vorhandener Systemkomponenten und die Einbeziehung externer Funktionalitäten in Form von Diensten, auf der Basis von CORBA, in Betracht gezogen werden.

7. Literatur

/1/ Schmidt, F., et al.: Dienste im UIS. Bericht über die Arbeiten der AG Dienste im Projekt GLOBUS III. Interner Bericht, Stand Nov. 1996.

/2/ Integration heterogener Komponenten des Umweltinformationssystems Baden-Württemberg, Phase III, Abschlußbericht. Forschungsinstitut für anwendungsorientierte Wissensverarbeitung an der Universität Ulm und Institut für Kernenergetik und Energiesysteme der Universität Stuttgart.

/3/ Kopetzky, R., Schmidt, F.: Entwurf und Implementierung eines Klienten zur Verarbeitung von Informationen aus verteilten Systemen. IKE-Bericht in Vorbereitung.

/4/ Koschel, A.; Kramer, R.;Nikolai, R. (1995): Architektur des WWW- und CORBA-basierten UIS, in: Umweltministerium Baden-Württemberg (Hrsg.), Projekt GLOBUS, Konzeption und prototypische Realisierung einer aktiven Auskunftskomponente für globale Umwelt-Sachdaten im Umweltinformationssystems Baden-Württemberg Phase II, Karlsruhe, S.47-92.

/5/ Koschel, A.; Kramer, R.; Theobald, D. (1995): CORBA-Evaluierung für das UIS Baden Württemberg, in: Umweltministerium Baden-Württemberg (Hrsg.), Projekt GLOBUS, Konzeption und prototypische Realisierung einer aktivern Auskunftskomponente für globale Umwelt-Sachdaten im Umweltinformationssystems Baden-Württemberg Phase II, Karlsruhe, S.93-192.

/6/ Object Management Group (1995): The Common Object Request Broker, Architecture and Specification Version 2.0.

CORBA als Middleware-Komponente im UIS Baden-Württemberg - Übersicht und Einsatzbeispiele -

A. Koschel,
Forschungszentrum Informatik (FZI),
Haid-und-Neu-Straße 10-14,
76131 Karlsruhe,

Unter Mitwirkung von:
D. Hoffmann (FZI)

1. Einleitung

1.1 Arbeiten zu CORBA in GLOBUS III

Die übergreifenden Komponenten des Umweltinformationssystems (UIS) Baden-Württemberg enthalten Informationen aus vielen unterschiedlichen Fachbereichen, wie Wasser, Boden, Luft, usw. Fachlich und organisatorisch bedingt, liegen die benötigten Informationen häufig verteilt auf unterschiedlichen DV-Systemen vor, die sich jeweils aus technisch heterogenen Systemplattformen, unterschiedlichen Datenbanksystemen, Expertensystemen, Geo-Informationssystemen usw. zusammensetzen können. Die bereits seit 1990 konzipierten und im Einsatz befindlichen übergreifenden Komponenten des UIS erfüllen zwar effizient die Anforderungen an fachliche Berichtspflichten jeweils spezifischer Nutzergruppen, z.B. in den Bereichen Naturschutz oder Luftreinhaltung, unterliegen jedoch Einschränkungen bei einer inzwischen veränderten Bedarfslage. Ferner stehen inzwischen eine Reihe neuerer Technologien zur Verfügung, deren Einsatz den Aufbau einer stärker diensteorientierten Architektur der übergreifenden Komponenten des UIS ermöglicht. Insbesondere ist hierdurch eine noch bessere Modularisierung und damit Wiederverwendbarkeit von Systemteilen (Diensten) möglich. In mehreren Projekten /7/, /10/ wird deshalb die Funktionalität einer Reihe von UIS-Applikationen in einer diensteorientierten Architektur zur Verfügung gestellt. Eingesetzt werden hierfür u.a. Techniken und Werkzeuge des World-Wide Web inklusive Java. Das World-Wide Web löst jedoch nicht alle Probleme. Z.B. ist eine inhaltliche Verknüpfung von Diensten, die auf unterschiedlichen Knoten residieren, nicht vorgesehen; diese Dienste können im WWW nur unabhängig voneinander genutzt werden. Ferner ist keine standardisierte Definition der Schnittstellen zu diesen Diensten möglich.

Eine mögliche Antwort auf eine Reihe dieser Probleme bietet die Common Object Request Broker Architecture (CORBA) der Object Management Group (OMG) /1/, /2/, /3/, /4/, /5/, /6/, der Industriestandard für offene verteilte Systeme. Bei CORBA handelt es sich um eine sogenannte Middleware (also eine mittlere Architekturschicht), die systemunabhängig definiert ist. In CORBA können mittels einer standardisierten Beschreibungssprache (Interface Definition Language, IDL) syntaktisch einheitlich die Schnittstellen zu beliebigen Software-Komponenten programmiersprachenunabhängig definiert und diese damit netzwerkübergreifend eingebunden werden. Als Benutzeroberflächen können darauf basierend beispielsweise auch World-Wide Web Werkzeuge (HTML/CGI, Java-Applets) eingesetzt werden. Somit ist CORBA gut geeignet, die serverseitige technische Heterogenität im UIS Baden-Württemberg zu verdecken.

Aufbauend auf den Arbeiten des FZI zur Architektur eines CORBA- und WWW-basierten UIS /8/ und der CORBA-Evaluierung für das UIS /9/, /10/ im GLOBUS-II Projekt /7/ werden deshalb in einer Reihe von Teilarbeiten zum GLOBUS-III Projekt /11/ Einsatzbeispiele für CORBA im UIS untersucht, dies besonders auch in der Kombination mit WWW-Technologie. In GLOBUS-III stehen folgende Arbeiten in Bezug zu CORBA:

- Einen Rahmen für die Eingliederung von CORBA in die Dienstearchitekur des UIS geben die Arbeiten der AG-Dienste /12/, an denen Arkusa, FAW, FZI und IKE beteiligt waren. Diese Arbeiten bauen besonders auf Ergebnissen der Projekte INTEGRAL und

WWW-UIS auf.

- Das vorliegende Dokument des FZI gibt zunächst eine Gesamtübersicht über die GLOBUS-Arbeiten speziell zu CORBA. Sodann wird ein Überblick über die derzeit für das UIS wesentlichen Teile von CORBA gegeben. Dies sind vor allem der CORBA Kern und die CORBAservices.
- Einen Schwerpunkt der praktischen Arbeiten zu CORBA im UIS bildet die Erschließung bestehender UIS-Dienste für eine CORBA-basierte Diensteumgebung. Hierfür eingesetzt wird das Konzept der (CORBA) Wrapper, die es erlauben, bestehende Informationsquellen zu kapseln, d.h. mit einer Zugriffsschnittstelle zu versehen. Speziell in CORBA werden diese Schnittstellen mit Hilfe der CORBA Interface Definition Language syntaktisch einheitlich und standardisiert spezifiziert.
  - Eine allgemeine Übersicht zur Wrapper Entwicklung besonders eingesetzt zur Kapselung von Datenbank-Zugriffen findet sich in einem separaten Bericht der AG-Dienste /12/.
  - Die Arbeiten von FZI und IKE zur Integration und Kombination von Datenbank-Zugriffsdiensten und von Simulationsdiensten ist im Teildokument „Integration und Kombination von Simulationsdiensten und Datenbankzugriffsdiensten mit CORBA“ in /11/ beschrieben. Gerade in dieser Arbeit werden auch Verbindungen von CORBA und WWW-Technologie (HTML/CGI, Java-Applets) eingesetzt. In einem umfangreichen Prototyp werden die Ergebnisse demonstriert.
  - Im Teildokument „GIS-Operatoren unter CORBA“ in /11/ wird die (grobe) Konzeption der Integration von Geo-Informationssystemen, als eine bedeutsame Komponente im UIS, in eine CORBA-Umgebung vorgestellt. Die Arbeiten von FZI und IPF folgen hierbei der Idee universeller GIS-Operatoren als Basis der Interoperabilität von Geo-Informationssystemen.

Gemeinsames Fazit aus allen Teilarbeiten ist die voraussichtlich gute Eignung von CORBA als Middleware Schicht im UIS Baden-Württemberg.

1.2 UIS-relevante Teile von CORBA

Der Hauptteil des vorliegenden Dokuments geht auf wesentlichen Anforderungen aus /12/ ein, die für das UIS-Baden-Württemberg /7/, /11/ an eine Middleware-Komponente gestellt werden und skizziert, inwieweit diese durch die Object Management Architecture (OMA) /4/ der Object Management Group (OMG) erfüllt werden. Die OMA gliedert sich in mehrere Bestandteile, deren Kern die Common Object Request Broker Architecture (CORBA, /3/, /4/) ist. Neben grundlegenden Eigenschaften des ORBs spielen hier die standardisierten CORBAservices /6/ eine wichtige Rolle. Noch eher unwichtig, da noch sehr im Fluß, sind CORBAfacilities /5/.

Da auf die ORB-Eigenschaften bereits in der CORBA-Evaluierung /9/, /10/ für das UIS in /7/ eingegangen wurde, werden sie in diesem Dokument nur kurz betrachtet. Im Vergleich zum Zeitpunkt dieser, sind allerdings inzwischen eine Reihe weiterer CORBAservices verabschiedet worden und ferner bereits Implementierungen einer Reihe von Services verfügbar. Die CORBAservices dienen dazu, den Entwurf und die Implementierung von Objekten einer verteilten Anwendung so zu unterstützen, daß häufig benötigte Funktionen nicht extra neu entworfen werden müssen. Sie stellen damit die grundlegenden,

standardisierten Schnittstellen zur Verfügung, welche die systemnahen Funktionen bereitstellen, die praktisch alle Objektimplementierungen benötigen. Auf sie wird im zweiten Teil des Dokuments näher eingegangen und die Bezüge zu den UIS-Anforderungen hergestellt. Ein kurzer, tabellarischer Marktüberblick über den Stand von Service-implementierungen in den in der CORBA-Evaluierung /9/, /10/ ausgewählten Produkten des 2. Auswahlschrittes, bildet den Abschluß des Dokuments.

2. Einige grundlegende Eigenschaften des ORB

2.1 Modulare Nutzbarkeit von Systemkomponenten

Für das UIS BW mit seinem technisch stark heterogenem Charakter ist es wichtig, daß viele verschiedenartige Systeme über die Middleware modular nutzbar sind. Hierfür wurde von der OMG die Schnittstellenbeschreibungssprache IDL (Interface Definition Language) eingeführt. Jede Schnittstelle zu einem Objekt wird über die objektorientierte IDL Beschreibung definiert. Diese Schnittstelle legt genau fest, welche Operationen mit welchen Parametern in welcher Zugriffsrichtung auf ein Objekt aufgerufen werden können. Der Zugriff auf das Objekt wird nicht direkt durchgeführt, sondern der Klient aktiviert über einen Client-Stub Operationen im Server-Skeleton, welche die gewünschten Aufrufe im eigentlichen Serverobjekt veranlassen. Als vermittelnde und überwachende Instanz zwischen diesen beiden Komponenten fungiert der ORB. Er realisiert u.a. den netzwerkspezifischen Teil der Kommunikation. Damit ist zum einen sichergestellt, daß nicht direkt auf ein Objekt zugegriffen werden kann (aufrufendes und empfangendes Objekt sind vollständig voneinander entkoppelt) und zum anderen ist klar festgelegt, welche Funktionalität ein Objekt erfüllen kann. Durch diese Vorgehensweise kann ein CORBA-Objekt plattformunabhängig und in jeder Programmiersprache für die es eine IDL Abbildung gibt implementiert werden. (Zur Zeit: Ada, C, C++, Smalltalk; an COBOL und Java wird gearbeitet). Auf das ebenfalls mögliche Kapseln bestehender Funktionalität (z.B. eines DBMS) mittels entsprechender IDL-Wrapper wird an anderer Stelle in GLOBUS-III /11/, in /1/, /2/ und weiter unten eingegangen (2.4 Kapselung bestehender Anwendungen).

2.2 Einfache Weiterentwicklung von Systemkomponenten

Durch die Objektorientierung von IDL und die Entkopplung von Objektimplementierung und IDL-Schnittstelle, ist es ggf. nicht notwendig, bei Weiter- oder Neuentwicklung einer Systemkomponente eine komplett neue Schnittstelle zu entwerfen. Es besteht vielmehr die Möglichkeit, die alte Schnittstelle zu vererben, und so die Schnittstelle um die neue Funktionalität zu erweitern. Auf diese Art und Weise bietet man dem Klienten sowohl die alte Basisschnittstelle an, über die er auch weiterhin unverändert seine Aufrufe tätigen kann, als auch die neue Schnittstelle mit dem vollen Funktionsumfang des Objekts. Das zugrunde liegende Objekt kann dabei nach belieben durch andere Objektimplementierungen ersetzt werden, solange diese die volle Schnittstellenfunktionalität erbringen.

Es ist auch möglich, mehrere Objektimplementierungen gleichzeitig über eine Schnittstelle anzubieten. Bei Aufruf einer Operation dieser Schnittstelle wird dann anhand des Naming Services (3.3 Naming Service), des Trader Services (3.9 Trader Service) oder anderer

Mechanismen entschieden, welche Implementierung aufgerufen wird. Ebenso ist es möglich, daß ein Objekt mehrere Schnittstellen implementiert. Die Implementierung stellt sich so nach außen dar, als gäbe es disjunkte Objekte. Diese Eigenschaften könnten bspw. gut in der Strategiekomponente für das UIS ausgenutzt werden.

2.3 Transparenz

Wie oben beschrieben, sind der Operationsaufruf und der Objektzugriff durch den ORB entkoppelt. Soll durch einen Klient auf Objekte zugegriffen werden, so muß dies über den ORB geschehen. Das hat den Vorteil, daß der Klient nicht wissen muß, wo sich ein Objekt befindet, sondern über den ORB ortstransparent auf ein Objekt zugreifen kann. CORBA selbst ist plattformunabhängig definiert, d.h. prinzipiell kann z.B. unter SunOS auf Objekte aus einer OpenVMS oder gar MS-Windows Umgebung zugegriffen werden (und umgekehrt), eine Eigenschaft, die von besonderer Wichtigkeit wegen der Heterogenität des UIS ist. Auf verschiedenen Plattformen würden jeweils systemspezifische ORBs „laufen". Gleiches gilt für die Netzwerkprotokolle zwischen den Systemen und für die Programmiersprachen der Objektimplementierungen. CORBA bietet also prinzipiell eine gute Möglichkeit, die technische Heterogenität der Systeme im UIS BW zu verdecken. Zur Flexibilität von CORBA trägt ferner bei, daß ein Zugriff auf Objekte sowohl statisch über die entsprechende IDL-Schnittstelle (IDL-Stubs) als auch dynamisch möglich ist. Für den dynamischen Zugriff existiert unterstützend ein Interface Repository, das alle verfügbaren Schnittstellenbeschreibungen von Objektimplementierungen enthält. Ferner ist eine Fehler- und Ausnahmebehandlung für Objektzugriffe definiert.

2.4 Kapselung bestehender Anwendungen (Wrapper)

Zur Integration bestehender Datenquellen und Softwaresysteme, für deren Programmiersprache bspw. keine IDL-Anbindung definiert ist oder für die kein Quellcodezugriff möglich ist, bedient man sich sogenannter IDL-Wrapper. Allgemein definiert, konvertiert ein Wrapper die Schnittstelle eines Objekts in eine Schnittstelle, die ein Klient zum Arbeiten benötigt. Ein Wrapper ermöglicht damit einem Klienten indirekt auf Objekten zu operieren. Es gibt verschiedene Ausprägungen von Wrappern für unterschiedliche Anforderungen. Dies stellt einen Schwerpunkt der bisherigen Arbeiten mit CORBA für das UIS dar /11/, /13/.

2.5 Kommunikationsaufkommen

Von einem theoretischen Standpunkt aus spielt die Anzahl und Größe der einzelnen Komponenten, in die ein Software-System beim Entwurf zerlegt wird, höchstens eine untergeordnete Rolle. Ähnliches gilt für die Größe der Datenpakete, die zwischen diesen Komponenten ausgetauscht werden. Für ein Software-System, das auf einer einzelnen Maschine oder gar innerhalb eines einzigen Adressraums ausgeführt wird, ist diese Auffassung auch in der Praxis gültig, da sämtliche Kommunikation über lokale (Hardware-) Mechanismen abläuft und somit nur minimalen Einfluß auf die Performance hat. In einem verteilten System ist die „Granularität" des Entwurfs jedoch ein entscheidendes Merkmal. Da bei solchen Systemen die Kommunikation zwischen den Komponenten jedesmal das

Verschicken von Daten über relativ langsame Netzverbindungen zur Folge haben kann, hat das Kommunikationsaufkommen zwischen den Software-Komponenten einen wichtigen Einfluß und sollte daher möglichst niedrig gehalten werden. Deshalb muß die Komponentengröße so gewählt werden, daß möglichst wenige Daten zwischen den Komponenten ausgetauscht werden müssen. Außerdem sollten die versendeten Datenpakete möglichst groß sein, um ein möglichst gutes Verhältnis Nutzdaten/Protokolldaten zu erreichen. Andererseits sollen die Komponenten noch so klein und ihre Schnittstellen differenziert sein, daß eine Wiederverwendung oder Erweiterung ohne übermäßigen Aufwand möglich ist. Es gilt hier einen Kompromiß zwischen den Anforderungen eines theoretisch optimalen Designs und den durch die Verteilung auferlegten Beschränkungen zu finden.

2.6 Performance

Bereits in der CORBA-Evaluierung für das UIS /9/ in /7/ wurde festgestellt, daß CORBA selbst keinen grundsätzlichen Leistungsengpaß in einer WWW-orientierten Umgebung, wie sie das UIS repräsentiert, darstellt. Dies gilt zumindest solange, wie nicht ständig sehr kleine Objektzugriffe erfolgen. Im UIS-Umfeld ist dies bisher typischerweise nicht der Fall, sollte aber bei Entwicklungen stets berücksichtigt werden. Leistungsprobleme könnten also höchstens durch die generellen Merkmale der verteilten Umgebung des UIS BW, also insbsondere durch die Netzkapazität auftreten, jedoch eher nicht durch den Einsatz von CORBA.

2.7 Interoperabilität, Anbindung an MS Windows-Systeme

Mit der aktuellen Version 2.0 von CORBA wurde das Internet Inter Orb Protocol (IIOP) standardisiert, das zur Interoperabilität von ORB's verschiedener Hersteller dient, insbesondere auch für Java-basierte ORBs (in Applets). D.h. hierdurch ist ein Verbindungsprotokoll zu WWW-Browsern gegeben, das im Gegensatz zu HTTP nicht mehr zustandslos ist. WWW-Browser wie Netscape, wollen IIOP künftig sogar direkt (allerdings nur als Klient) unterstützen. Ebenfalls durch eine Reihe CORBA-Produktherstellern wird eine - allerdings derzeit im wesentlichen herstellerspezifische - CORBA-Integration mit den Microsoft OLE bzw. COM Protokollen angeboten, d.h. beispielsweise ein Zugriff aus Visual Basic heraus auf CORBA-Objekte ermöglicht. Von der OMG wurde kürzlich ebenfalls eine Schnittstelle zu COM definiert.

2.8 Sessions

CORBA unterstützt nicht von sich aus ein Sitzungsmodell für verteilte (Sub-)systeme. Je nach Art der zu integrierenden Komponente bieten sich hier mehrere Vorgehensweisen an.

- Zunächst besteht die Möglichkeit, bei jeder Operation, die von einem entfernten Klienten angefordert wird, eine neue Verbindung aufzubauen, die nach Beendigung der Operation wieder abgebaut wird. In diesem Fall müssen die Operationen, welche von der Schnittstelle der Komponente angeboten werden, so umfangreich sein, daß der Klient möglichst wenige kontextbezogene Daten verwalten muß. Andernfalls wird die Austauschbarkeit der gewrappten Komponente aufwendig oder unmöglich, da solche Daten im allgemeinen komponentenspezifisch sind.

Wenn es nicht möglich oder erwünscht ist, eine solche Schnittstelle zu schaffen, bieten sich zwei weitere Möglichkeiten an, ein Session-Konzept für entfernte Systemkomponenten zu realisieren:

- Für die entsprechenden Komponenten wird eine eigene Session-Verwaltung entwickelt und entsprechende Operationen werden von ihrer Schnittstelle angeboten. Dies bedeutet unter Umständen einen erheblichen Entwicklungsaufwand. Zudem muß eine solche Session-Verwaltung bei einem Austausch der Komponente ggf. vollständig neu entwickelt werden. Vorteilhaft ist jedoch, daß auf diese Weise ein Session-Konzept realisiert werden kann, das die Eigenschaften der zu integrierenden Komponenten und ihrer lokalen Umgebung optimal berücksichtigt und ausnutzt.
- Eine relativ primitive Form der Session-Verwaltung läßt sich durch die Verwendung von „unshared" Servern erreichen: für jeden Klienten, der einen bestimmten Dienst verwenden will, wird ein eigener Server gestartet, der erst dann wieder beendet wird, wenn der Klient meldet, daß er diesen Dienst nicht weiter in Anspruch nehmen will.

Die Verwendung von „unshared" Servern bedeutet gegenüber der Entwicklung einer „echten" Session-Verwaltung eine erhebliche Einsparung des Entwicklungsaufwands. Sie kann aber bei gleichzeitiger Verwendung eines Dienstes durch viele Klienten ggf. zu einer Überlastung des entsprechenden Subsystems führen, da jeder Server in einem eigenen Prozeß abläuft und viele Daten z.T. redundant repliziert werden müssen.

3. Die CORBAservices

Dieses Kapitel gibt eine kurze Beschreibung der CORBAservices /6/, sowie mögliche Anwendungsmöglichkeiten im Rahmen des UIS.

3.1 Lifecycle Service

Der Lifecycle Service definiert Dienste und Vereinbarungen für das Erzeugen, Zerstören, Kopieren und Bewegen von Objekten. Durch die Forderung nach Ortstransparenz aller Objekte in CORBA stellen sich dabei einige grundlegende Probleme. Daher bedient man sich in CORBA sogenannter Fabrik Objekte, um Objekte zu erzeugen, auf die Funktionen des Lifecycle Services aufgerufen werden können. Diese Fabrik Objekte sind wiederum ganz normale CORBA Objekte. Jede spezielle Fabrik erzeugt dabei einen speziellen Objekttyp, daher gibt es keine Standardschnittstelle für Fabriken. Ein Objekt, daß einen ortstransparenten Lebenszyklus hat, muß die entsprechende Schnittstelle des Lifecycle Services erben und gegebenenfalls implementieren.

Dieser Service stellt die Grundlage für die transparente Verteilung von Objekten dar. Durch seine Grundfunktionen move, copy und delete kann jedes Objekt, welches die Lifecycle Schnittstelle implementiert, auf einheitliche Art und Weise migriert, erzeugt, dupliziert und zerstört werden. Der CORBA-ORB selbst ermöglicht im Gegensatz dazu, i.w. nur das Erzeugen und Aufrufen von Objekten, also nur Teile des Lifecycle Service. In dieser Form wird er bisher im UIS genutzt.

3.2 Relationship Service

Objekte werden häufig benutzt, um Gegenstände der realen Welt zu modellieren. Diese Gegenstände können in einer bestimmten Beziehung zueinander stehen, und die Gegenstände nehmen in diesen Beziehungen jeweils bestimmte Rollen an. In einem Compound Document Objekt beispielsweise hat jedes beinhaltete Objekt, wie auch das Compound Document selbst eine bestimmte Rolle (z.B. als Graphik, Text oder als Unterkapitel). Diese Objekte stehen untereinander wiederum in Beziehungen (z.B. ein Unterkapitel gehört zu einem Kapitel). Das Compound Document ist somit aus Sicht des Relationship Services ein Container, in dem alle Referenzen auf Objekte die es repräsentieren gesammelt werden (und natürlich die Rollen, die die Objekte in den jeweiligen Beziehungen spielen, sowie die Beziehungen selbst). Dabei ist es irrelevant, woher diese Objekte stammen, wer sie wie erzeugt hat und wo sie sich gerade befinden. Der Relationship Service bietet des weiteren eine Schnittstelle zum Durchwandern der Graphen, die beim Aufbau von Beziehungen zwischen Objekten entstehen.

Im Rahmen des UIS können somit verschiedene Objekte, die unterschiedliche Dienste innerhalb des UIS anbieten in verschiedenen Rollen unterschiedliches Verhalten zeigen (z.B. Berechnungen durchführen oder nur Darstellungen erzeugen). Man kann Beziehungen zwischen verschiedenen Diensten herstellen und gegebenenfalls einen „Beziehungsgraphen" von einem Dienst zum nächsten abwandern. Ein weiterer Anwendungsaspekt liegt darin unterschiedliche Objekte zu gruppieren und für diese Gruppierungen eine einheitliche Dienstschnittstelle anzubieten.

3.3 Naming Service

Der Naming Service stellt die Funktionalität bereit, um Objekte benennen zu können. In einer verteilten Umgebung kann ein Objekt nur innerhalb eines lokalen Bereichs eindeutig benannt werden. Für diesen Zweck werden sogenannte Namenskontextobjekte bereitgestellt, welche innerhalb ihres Bereichs Objekten eindeutige Namen zuordnen. Außerdem legt der Naming Service die Konventionen über die strukturelle Form der Namen fest. Um nicht neben den in diversen Filesystemen bereits existierenden Konventionen noch eine neue Konvention für die Namensgebung von Objekten zu entwickeln, besteht ein Name in CORBA aus einer geordneten Sequenz von Namenskomponenten. Dabei ist nur die letzte Namenskomponente mit einer Objektreferenz verknüpft. Alle anderen sind an sogenannte Namenskontexte gebunden. Ein solcher Namenskontext ist selbst wiederum nur ein gewöhnliches Objekt, daß innerhalb seines Bereichs einem Objekt einen eindeutigen Namen zuordnet. Hat man beispielsweise einen Namen für ein Objekt, der aus fünf Namenskomponenten zusammengesetzt ist, so müssen, falls keine Vorkenntnisse über den Namen vorhanden sind, zunächst die ersten vier Namenskomponenten aufgelöst werden, bevor die fünfte Komponente dem eigentlichen Objekt zugeordnet werden kann. Auf diese Art und Weise läßt sich jedes Filesystem nachbilden.

Dieser Service ist eine Grundlage für den Zugriff auf ortstransparente Objekte und Gruppierungen von Objekten. Dieser Service wird im FZI in einer vorliegenden Implementierung (Orbix, Beta-Version) betrachtet. Im UIS könnte er z.B. im Rahmen der Strategiekomponente eingesetzt werden.

3.4 Persistent Object Service

Der Persistent Object Service (POS) stellt eine Schnittstelle bereit, um die Zustände von Objekten dauerhaft zu speichern und zu verwalten. Der Zustand eines Objekts wird durch die Belegung seiner Datenstrukturen mit Werten repräsentiert. Der Persistent Object Service stellt dabei im wesentlichen Dienste zur Verfügung, die dazu benutzt werden, um den Zustand einfacher Objekte zwischen zwei Aufrufen zu speichern. (Ein Klient, der ein Objekt erzeugt, muß sich darauf verlassen können, daß dieses Objekt zu einem späteren Zeitpunkt noch unverändert vorhanden und aufrufbar ist.) Der Persistent Object Service stellt einen einfachen Dienst zur Verfügung um eigene Objekte persistent zu speichern. Er ist kaum geeignet, um den Zustand von komplexen bes. gekapselten Systemen zu sichern (z.B. gekapselte DBMS).

Da im UIS bereits verschiedene DBMS u.ä. zur Haltung umfangreicher Datenbestände eingesetzt werden, ist eine einfache Form, wie durch den Persistence Service geboten, eher nicht bedeutsam. Eine Kapselung und damit Weiternutzung der bestehenden DBMS ist sinnvoller. Hierfür sind für das UIS bereits entsprechende Wrapper entwickelt worden und inzwischen auch erste kommerzielle Werkzeuge verfügbar.

3.5 Externalization Service

Der Externalization Service ist dem Persistent Object Service im Grunde sehr ähnlich. Beide Services stellen über eine Schnittstelle Dienste bereit, um die Zustände von Objekten dauerhaft zu sichern und zu verwalten. Im Gegensatz zum POS bietet der Externalization Service jedoch Dienste, um die Daten der Objekte in ein bestimmtes Format umzuwandeln und abzulegen (externalize). Ebenso können die abgelegten Daten wieder in dasselbe oder ein anderes Objekt eingelesen werden (internalize). Beide Vorgänge werden über das aus C++ bekannte Stream-Konzept realisiert. Beim Auslagern werden zunächst wichtige Implementationsabhängige Informationen des Objekts in den Stream geschrieben. Diese Informationen benötigt das entsprechende Fabrik Objekt, um das Objekt beim Einlagern wieder ordnungsgemäß rekonstruieren zu können. Erst dann werden die eigentlichen Zustandsdaten des Objekts in den Stream geschrieben.

Für das UIS könnte der Externalization Service eine größere Bedeutung, als der Persistent Object Service (POS) haben. Der Externalization Service wird im Gegensatz zum POS häufig dazu benutzt, Objektdaten von IDL-Objekten in ein standardisiertes Format zu wandeln (ein einfaches, ggf. erweiterbares Format ist in CORBA vordefiniert). Im Rahmen des UIS gibt es für die verschiedenen Anwendungen eine Vielzahl von Daten in standardisierten Datenformaten, die innerhalb der jeweiligen Dienste gespeichert oder unter den vielen UIS-Diensten ausgetauscht werden. Der Externalization Service stellt dabei sicher, daß diese Daten über einen langen Zeitraum in einem einheitlichen, von vielen Anwendungen lesbarem Format gesichert werden können, könnte also vielleicht als ein einfaches Austauschformat einsetzbar sein.

3.6 Event Service

Der Event Service entkoppelt die Kommunikation zwischen Objekten, indem er einen sogenannten Ereigniskanal zwischenschaltet. Die Objekte, welche nun an der Kommunikation teilnehmen wollen, senden Ereignisdaten über den Kanal an ihre Partner. Dabei nehmen die Objekte die Rolle eines Lieferanten (supplier) oder eines Verbrauchers (consumer) ein. Ein Lieferant produziert Ereignisdaten, indem er Nachrichten in den Ereigniskanal schreibt, ein Verbraucher verarbeitet Ereignisdaten, indem er Nachrichten aus dem Ereigniskanal liest. Der Ereigniskanal tritt dabei sowohl als Lieferant, als auch als Verbraucher auf. Er ist somit eine Art Zwischenspeicher. Das Transportieren der Ereignisdaten geschieht über standardisierte CORBA-Aufrufe, für welche die OMG im Ereignisdienst entsprechende Schnittstellen definiert hat. Durch das push/pull Modell des Event Services werden neben synchronen und rein asynchronen Mechanismen, auch sogenannte deferred-synchronous Mechanismen ermöglicht. Dabei kann der Aufrufer nach Absetzen des Aufrufs mit seiner Arbeit fortfahren und zu einem späteren Zeitpunkt abfragen oder davon unterrichtet werden, ob das Aufrufergebnis schon eingegangen ist. Es ist ebenfalls möglich, ein Ereignis an mehrere mit dem Ereigniskanal verbundene Klienten zu propagieren.

Der Event Service bietet somit eine Kommunikationsbasis an, z.B. zur Unterstützung aktiver Mechanismen für verteilte UIS-Objekte. Mittels Regeln flexibel spezifizierbare aktive Mechanismen für CORBA /14/ könnten im UIS wiederum gut zur Benachrichtigung von Anwendern, z.B. über Grenzwertüberschreitungen oder beim Eintreffen neuer UDK-Daten eingesetzt werden. Über den Event Service können ferner call-back Aufrufe des Servers zum Klienten realisiert werden.

3.7 Object Transaction Service

Der Object Transaction Service (OTS) erweitert die Transaktionssemantik auf verteilte, objektorientierte Systeme. Hierzu stellt der OTS eine Schnittstelle bereit, um in einer verteilten Umgebung flache und geschachtelte Transaktionen, nach dem aus DBMS bekannten ACID Paradigma zu unterstützen. Um eine Transaktion zu kontrollieren, wird ein Kontrollobjekt bereitgestellt, welches eine Transaktion nach außen repräsentiert. Ein Klient ruft zur Erzeugung einer Transaktion eine entsprechende Operation einer Transaktions-Fabrik auf. Die Fabrik stellt das entsprechende Kontrollobjekt zur Verfügung. Mit dem Kontrollobjekt kann explizit der Kontext einer Transaktion manipuliert und verwaltet werden oder in Aufrufen weitergegeben werden. Ein Objekt kann jedoch nur über den OTS in einer Transaktion involviert sein, wenn die entsprechenden Schnittstellenfunktionen für dieses Objekt implementiert wurden. Der OTS kann keine vollständige Transaktionsfunktionalität für Objekte bereitstellen, welche nicht durch Implementierung der entsprechenden Schnittstelle dafür vorgesehen sind. Durch die Möglichkeit existierende Transaction Processing (TP) Systeme mittels OTS zu wrappen, müssen vorhandene Transaktionssysteme nicht vollständig neu implementiert werden. Dabei werden sowohl flache, als auch hierarchische und verschachtelte Transaktionsmechanismen unterstützt.

Allerdings stellt der OTS, wie gesagt, lediglich eine Verwaltung von Transaktionsobjekten dar, d.h. Objekte müssen entsprechende Schnittstellen selbst unterstützen. Sie müssen also

bspw. eine Rollback-Funktionalität implementieren, die dann über den OTS aufrufbar ist. Z.B. eine Reihe von DBMS (im UIS u.a. Oracle) bieten hierfür Schnittstellen an. Bei anderen mittels Wrappern gekapselten Systemen, die nicht über entsprechende Funktionen und Schnittstellen verfügen, kann es allerdings einen größeren Implementierungsaufwand bedeuten oder prinzipiell nur schwer möglich sein. Dies ist allerdings ein generelles Problem bei der Unterstützung von Transaktionen in heterogenen Systemumgebungen. Der Transaction Service umgeht dieses Problem, indem er u.a. zwischen (mittels eines Rollback) rücksetzbaren und nicht-rücksetzbaren Quellen unterscheidet. Bei einer nicht-rücksetzbaren Quelle, bewirkt der Abbruch einer Transaktion einfach nichts, was natürlich nicht immer befriedigend ist.

Bisher erfolgen im UIS im wesentlichen lesende Zugriffe auf Informationsquellen, so daß Transaktionen noch keine große Rolle spielen. Dies kann sich allerdings z.B. bei komplexen Kombinationen von (System-)Diensten ändern, so daß zumindest für eine spezifische Menge von Diensten, eine Transaktionsfähigkeit sinnvoll ist. Für solche Fälle kann der OTS wiederum eine gute Basis darstellen.

3.8 Concurrency Service

Der Concurrency Service verwaltet konkurrierende Zugriffe auf Objekte durch Vergabe von Sperren, so daß der Zustand der Objekte konsistent bleibt. Dieser Service ist dabei so allgemein gehalten, daß er die Objekte auf welche zugegriffen wird, abstrakt als Ressourcen betrachtet. Dabei kann es sich um ein normales Objekt handeln, oder ein Objekt das die Schnittstelle des OTS unterstützt. Für jede Sperre existieren fünf Abstufungen, so daß eine feine Granularität beim Sperren und Entsperren einer Ressource ermöglicht wird. Weiterhin besteht die Möglichkeit Sperrmengen zu definieren. Dabei kann ein Klient eine Menge von Sperren auf einer Ressource gleichzeitig besitzen.

Wie beim Transaction Service gilt jedoch auch für den Concurrency Service, daß zu einem bestehenden System/Objekt keine Funktionalität „von außen" hinzugefügt werden kann. Zur sinnvollen Nutzung des Concurrency Services muß ein Objekt also wiederum die passende Schnittstelle erben und sie sinnvoll implementieren. Der Concurrency Service stellt jedoch einen sorgsam konzipierten Mechanismus dar, der ohne großen Overhead feingranulares Sperren von Objekten in verteilten Systemen ermöglicht. Innerhalb des UIS lassen sich mit Hilfe des Concurrence Services die Anzahl der Zugriffe auf ein Objekt einschränken. So kann exklusiver Zugriff auf ein Objekt oder eine Begrenzung der Anzahl von gleichzeitigen Zugriffen auf ein Objekt realisiert werden.

3.9 Trader Service

Der Trader Service ermöglicht das Auffinden und Registrieren von Objekten über Angabe von Eigenschaften oder Fähigkeiten. Er stellte somit eine Art „gelbe Seiten" für Objekte in verteilten Systemen dar. Ein Trader ist ein Objekt, das die Trader Service Schnittstelle implementiert. Mit Hilfe dieses Trader Objekts können andere Objekte ihre Eigenschaften und Fähigkeiten anbieten (export) oder Objekte finden, die ihre Anforderungen erfüllen (import). Ein Objekt, welches seine Eigenschaften und Fähigkeiten anderen Objekten anbieten will, übergibt dem Trader eine Beschreibung des Dienstes, den es anbieten will und eine Referenz

auf eine Schnittstelle, welche diesen Dienst implementiert. Um ein Objekt zu finden, welches bestimmten Anforderungen entspricht, übergibt das suchende Objekt dem Trader eine Liste mit den Anforderungen an dieses Objekt. Der Trader gleicht die Anforderungen mit den Eigenschaften, der registrierten Objektbeschreibungen ab und gibt bei Erfolg die Referenz auf die gewünschte Schnittstelle zurück. Um bei der Vielzahl an Objektdiensten, die bei einem Trader registriert werden können effizient arbeiten zu können und die Skalierbarkeit des Gesamtsystems zu wahren, wird der Traderdienst durch mehrere verteilte Trader Objekte realisiert. Jeder Trader ist in erster Linie für einen begrenzten Teil des Gesamtsystems (z.B. eine Internet Domäne) zuständig. Kann er eine Anfrage nicht befriedigen, so propagiert er die Anfrage an benachbarte Trader.

Der Trader Service ist gut geeignet, die steigende Zahl an Diensten im UIS zu verwalten. Durch den Einsatz eines Traders zur Dienstevermittlung zwischen Klient und Server können bereits bei der Auswahl des diensterbringenden Servers Kriterien, wie beispielsweise Kosten oder Dauer der Diensterbringung, berücksichtigt werden. Im Rahmen des UIS sind viele Einsatzmöglichkeiten denkbar, zum Beispiel automatische Auswahl einer bestimmten Darstellungsart eines Ergebnisses (HTML oder ASCII), einer oberen Grenze für die Auflösung einer Kartendarstellung oder auch des Aktualitätsgrades der gelieferten Daten. Durch Duplizieren von Diensten auf unterschiedlichen Rechnerknoten kann mit Hilfe des Trader Services Redundanz erzielt werden. Erfragt das Objekt beim Trader einen Dienst, der auf einem ausgefallenem Rechnerknoten implementiert ist, so kann der Trader automatisch eine Implementierung auf einem funktionierenden Rechnerknoten auswählen (dynamisches Trading). In jedem Fall ist der Trader Service ein wesentlicher Bestandteil einer Strategiekomponente im UIS.

3.10 Security Service

Zur Unterstützung von Zugriffssicherheit wurde der CORBA Security Service spezifiziert. Sicherheit ist ein wichtiger Aspekt in verteilten Systemen. Zum einen werden Informationen über meist öffentliche Netzwerke verschickt und sind deshalb anfälliger gegenüber Angriffen. Zum anderen sind verteilte Systeme sehr viel komplexer als traditionelle Client/Server Systeme und bieten eine viel größere Angriffsfläche für potentielle Angreifer. Da die OMA eine Vielzahl von existierenden Sicherheitskonzepten in ihre Architektur integrieren soll, ist die Verwendung eines einzelnen Sicherheitsmodells nicht angemessen. Statt dessen wird ein Referenzmodell definiert, welches einen Rahmen für die Einbindung vieler verschiedener Sicherheitskonzepte realisiert. Zu den Sicherheitsrisiken gegen die der Security Service Schutz bieten soll gehören:

- Autorisierte Benutzer eines Systems, die versuchen auf Informationen zuzugreifen, auf die sie kein Zugriffsrecht haben.
- Benutzer, die sich für einen anderen Benutzer ausgeben, um Aktionen durchzuführen, für die dieser Benutzer autorisiert ist.
- Umgehung von Sicherheitsvorkehrungen.
- Abhorchen eines Kommunikationskanals.
- Fehlende Abrechnungsmöglichkeiten für geleistete Dienste durch beispielsweise mangelhafte Identifizierung eines Benutzers.

Es sei angemerkt, daß die Spezifikation des Security Service nicht gegen alle existenten Sicherheitsrisiken, die in einem verteilten System auftreten könne Schutz bietet. Als Beispiel für ein solches Sicherheitsrisiko sei hier das Sammeln von Informationen durch die Analyse des Datenstroms genannt.

Die Verwirklichung eines Sicherheitskonzepts durchdringt die gesamte OMG Architektur. Es hat direkte Auswirkungen auf alle Komponenten des verteilten Systems, sowohl auf den ORB selbst, die meisten CORBAservices, die CORBAfacilities, als auch auf die einzelnen Objektimplementierungen. Die Spezifikation des Security Services deckt dabei zum einen die untere Schicht der Sicherheitsfunktionen ab. Dies sind Funktionen für Anwendungen, die sich der Sicherheitsmechanismen und wie diese durchgeführt werden nicht bewußt sind. Zum anderen bietet der Security Service die Möglichkeit durch die Verwendung anwendungseigener Sicherheitsfunktionen die Sicherheitsmechanismen aktiv zu kontrollieren. Wichtig hierbei ist, daß es dem Anwender möglich sein soll, seine Sicherheitspolitik unabhängig von den zugrunde liegenden Funktionen festzulegen. Dies kann dadurch geschehen, daß der ORB sogenannte generische Interceptor Schnittstellen verwendet. Durch diese Schnittstellen können sicherheitsrelevante Objektaufrufe abgefangen werden, um zunächst in der Interceptorimplementierung überprüft zu werden. Dies hat den Vorteil, daß der ORB keinelei sicherheitsrelevanten Informationen enthält. Diese sind in der Interceptorimplementierung enthalten, die sich auf einem „sicheren" System befinden kann. Auch ohne die Verwendung von Interceptoren muß der ORB die Aufrufe von Security Service Funktionen über abstrakte Schnittstellen tätigen. So ist es möglich, die Security Service Implementation unabhängig vom verwendeten ORB auszuwechseln. Die Implementierung des Security Services benötigt durch die Verwendung der abstrakten Schnittstellen keinerlei Informationen über die Funktionsweise des ORBs, z.B. wie dieser die Security Objekte lokalisiert.

Um den am Anfang genannten Sicherheitsrisiken zu begegnen, definiert der Security Service folgende Funktionalität:

- Identifikation und Authentifikation: Benutzer und Objekte, die in eigener Verantwortung Aktionen ausführen, müssen sich eindeutig den benutzten Objekten gegenüber identifizieren können. Innerhalb dieser Identifikation müssen sie eindeutig „beweisen" (Authentifikation), daß sie auch derjenige sind, für den sie sich ausgeben. Gleiches gilt für Objekte, die sich untereinander aufrufen.
- Autorisierung und Zugriffskontrolle: In einem verteilten System werden Rechte an Objekte delegiert (Autorisierung), damit diese Aufrufe bei anderen Objekten mit den Rechten des Delegierenden tätigten können. Das aufgerufene Objekt entscheidet mit Hilfe der Autorisierung des Aufrufers, auf welche Informationen der Aufrufer zugreifen darf.
- Sicherheitsaufzeichnungen: Um Benutzer für ihre sicherheitsrelevanten Aktionen verantwortlich machen zu können, muß der Benutzer bei jeder Aktion eindeutig identifiziert werden können, auch bei einer Kette von Aufrufen über mehrere Objekte.
- Kommunikationssicherheit: Dies bedeutet, daß Aufrufer und Aufgerufener ein Vertrauensverhältnis herstellen, aufgrund dessen die Identität des Kommunikationspartners und die Integrität der ausgetauschten Daten sichergestellt wird.
- Nichtzurückweisbarkeit (non-repudiation) einer getätigten Aktion: D.h., es soll sicher

feststellbar sein, ob ein Benutzer eine Aktion durchgeführt hat oder nicht. Es soll dem Benutzer nicht möglich sein eine von ihm tatsächlich durchgeführte Aktion, leugnen zu können.

- Administrierung von sicherheitsrelevanten Informationen: Für manche Sicherheitspolitiken ist es notwendig, sicherheitsrelevante Informationen zu sammeln und zu verwalten.

Im Rahmen der Sicherung der Datenintegrität während der Übertragung können in manchen Umgebungen die Parameter, die dem ORB übergebenen werden, von diesem verschlüsselt werden. Dies ermöglicht die Wahrung der Datenintegrität für Objekte, die bei der Übertragung selbst keine Sicherheitsmechanismen anwenden.

Einfache Security Funktionalität läßt sich zur Zeit schon durch herstellereigene Mechanismen (z.B. in Orbix: Filter kombiniert mit ebenfalls unterstützter Authentifizierung auf UNIX Basis) erreichen.

3.11 Licensing Service

Wie bei anderen CORBAservices auch (z.B. Naming Service) existieren im Bereich der Lizensierungssoftware schon viele verschiedene proprietäre Lösungen. Aus diesem Grund hat die OMG auch hier kein neues Lizensierungsmodell entwickelt. Statt dessen bietet der Licensing Service mehrere generische Schnittstellen an, die unabhängig von dem zugrunde liegenden Lizensierungsmodell von jedem Softwareprodukt verwendet werden können.

3.12 Object Properties Service

Der Object Properties Service ermöglicht es, zusätzliche Eigenschaften (Properties) an ein Objekt „anzuhängen". Eine solche Eigenschaft ist im Prinzip wie ein Objektattribut zu verwenden, außer, daß es dynamisch dem Objekt hinzugefügt wird und nicht statisch durch die IDL Definition des Objekts. Jede Eigenschaft hat dabei eine Namen und einen Wert (any). Durch diese Erweiterung können u.a. nachträglich und von außen Einstufungen vorgenommen werden, die es erleichtern Objekte zu verwalten. Beispielsweise kann man Objekten eine Eigenschaft „Wichtigkeit" zuordnen, um sie besser verwalten zu können. Diese zusätzlichen Eigenschaften haben keinen Einfluß auf die Funktionsweise und Wirkung der Objektimplementierung an die sie „angehängt" werden.

3.13 Object Query Service

Mit dem Object Query Service steht eine mächtige Schnittstellendefinition zur Verfügung, um allgemeine Anfragen an Objekte zu gestalten. Dabei spielt es keine Rolle, welches Datenmodell das Objekt zur Speicherung seiner Datenbasis verwendet (RDBMS, OODBMS, etc.). So lange das Objekt die Schnittstelle implementiert, steht für den Anwender eine einheitlich Anfrageschnittstelle zur Verfügung. Eine Implementierung dieser Schnittstelle muß entweder Anfragen gemäß dem SQL-92 Standard oder aus dem ODMG-OQL Industriestandard unterstützen. Alle Objekte, welche den OQS implementiert haben, können somit homogen Anfragen unterstützen.

Für das UIS könnte dieser Service eine Bedeutung für DBMS-Zugriffe haben. Ob er allerdings einen deutlichen Mehrwert gegenüber UIS-eigenen Schnittstellen (Wrappern) darstellt, ist sicher noch fraglich.

4. CORBAservices[1] in Produktimplementierungen

Produkt / Firma	1	2	3	4	5	6	7	8	9	10	12	13
Orbix / Iona	P		X	P		X	P					
Object Broker / Digital	P					P				P		
NEO / Sun	X	X	X	P	P	X	P	P			X	P
ORB+ / HP	X	P	X	P	P	X	P				P	P
PowerBroker / Expersoft	P	P	X	P	P	+	P	P				
DAIS / ICL	P		P			P	P	P	X	P		

P: proprietäre Implementierung,

X: Implementierung gemäß OMG Spezifikation,

+: in Kürze implementiert gemäß OMG Spezifikation

1	Lifecycle Service	8	Concurrency Service
2	Relationship Service	9	Trader Service
3	Naming Service	10	Security Service
4	Persistent Object Service	11	Licensing Service
5	Externalization Service	12	Object Properties Service
6	Event Service	13	Object Query Service
7	Object Transaction Service		

Für die ORBs von Sun, Iona und Digital (sowie von VisiGenic) existieren schon Anbindungen an Java. Die ORBs von Expersoft, Iona, Digital und HP enthalten Schnittstellen zu OLE bzw. COM.

[1] Stand Implementierungen der CORBAservices September 1996 nach Herstellerangaben.

5. Zusammenfassung

In diesem Dokument wurden die wichtigsten Elemente von CORBA in Bezug auf ihre Einsetzbarkeit im UIS BW dargestellt. Die grundsätzliche Eignung von CORBA als Middlewarekomponente im UIS wurde bereits in der CORBA-Evaluierung und im Projekt WWW-UIS in GLOBUS II /9/, /10/ festgestellt. Dort wurde die grundsätzliche Eignung von CORBA für das UIS BW durch eine praktische Erprobung mehrerer CORBA-Implementierungen und durch grobe Leistungsmessungen in UIS-Szenarien herausgearbeitet. Inzwischen haben sich CORBA und insbesondere seine Implementierungen weiterentwickelt, so daß viele der in den vorangehenden Abschnitten dargestellten UIS-relevanten Eigenschaften von CORBA, wie Java-Anbindungen und mehrere CORBAservices jetzt in Form von Produktimplementierungen nutzbar sind. Die praktischen Erfahrungen aus einer Reihe weiterer Teilprojekte mit CORBA-Bezug im Rahmen von GLOBUS-III /11/ belegen die gute Einsetzbarkeit dieser Technologie als Middleware für das UIS BW.

6. Literatur

/1/ Siegel, Jon (1996): CORBA, Fundamentals and Programming, New York.

/2/ Mowbray, Thomas J.; Zahavi, Ron (1995): The Essential CORBA, Systems Integration Using Distributed Objects, New York.

/3/ Object Management Group (1995): The Common Object Request Broker, Architecture and Specification Version 2.0.

/4/ Soley, R. M. (1990): Object Management Architecture Guide.

/5/ Object Management Group (1995): Common Facilities Architecture.

/6/ Object Management Group (1995): CORBAservices, Common Object Services Specification.

/7/ Umweltministerium Baden-Württemberg (1995): Projekt Globus, Konzeption und prototypische Realisierung einer aktiven Auskunftskomponente für globale Umwelt-Sachdaten im Umweltinformationssystem Baden-Württemberg Phase II, Karlsruhe.

/8/ Koschel, A.; Kramer, R.; Nikolai, R. (1995): Architektur des WWW- und CORBA-basierten UIS, in /7/, Seite 47-92.

/9/ Koschel, A.; Kramer, R.; Theobald, D. (1995): CORBA-Evaluierung für das UIS Baden-Württemberg, in /7/, Seite 93-192.

/10/ Koschel, A.; Kramer, R.; Theobald, D.; von Bültzingsloewen, Günter; Hagg, Willhelm; Wiesel, Joachim; Müller, Manfred (1996): Evaluierung und Einsatzbeispiele von CORBA-Implementierungen für Umweltinformationssysteme, 10. Symposium Umweltinformatik, Hannover.

/11/ Umweltministerium Baden-Württemberg (1996): Projekt Globus, Konzeption und prototypische Realisierung einer aktiven Auskunftskomponente für globale Umwelt-Sachdaten im Umweltinformationssystem Baden-Württemberg Phase III, Karlsruhe.

/12/ Koschel, A.; Schmidt, F.; Schöckle, M; Strohm, J.; Tischendorf, M.(1996): *Bericht der AG-Dienste*, in: /11/.

/13/ Schmidt, F., et al.: Dienste im UIS. Bericht über die Arbeiten der AG-Dienste im Projekt GLOBUS III. Interner Bericht, Stand Nov. 1996.

/14/ v. Bültzingslöwen, G.; Koschel, A.; Kramer, R.: *Active Information Delivery in a CORBA-based Distributed Information System*, in: Aberer, K.; Helal, A. (Hrsg.), Proc. First IFCIS International Conference on Cooperative Information Systems (CoopIS'96). IEEE Computer Society Press, Los Alamitos, California. Brüssel, Belgien. Juni 1996. S.218-227.

Integration und Kombination von Simulationsdiensten und Datenbankzugriffsdiensten mit CORBA

A. Koschel,
Forschungszentrum Informatik (FZI),
Haid-und-Neu-Straße 10-14,
76131 Karlsruhe

M. Schöckle,
Universität Stuttgart,
Institut für Kernenergetik und Energiesysteme (IKE),
Abteilung Wissensverarbeitung und Numerik, Pfaffenwaldring 31,
70550 Stuttgart

unter Mitwirkung von:
F. Schweizer (IKE), C. Weinand (FZI)

1. Einleitung

Die übergreifenden Komponenten des Umweltinformationssystems Baden-Württemberg (UIS BW) (siehe auch /4/, /16/) enthalten Informationen aus vielen unterschiedlichen Fachbereichen, wie Wasser, Boden, Luft, Naturschutz usw. Diese Informationen liegen – fachlich und organisatorisch bedingt – häufig verteilt auf unterschiedlichen DV-Systemen, die sich wiederum aus technisch heterogenen Systemplattformen, unterschiedlichen Datenbanksystemen, Expertensystemen, Geo-Informationssystemen, Berechnungskomponenten usw. zusammensetzen können.

In einer Reihe von Arbeiten für das UIS BW /16/, /17/ werden derzeit neue Technologien untersucht, die eine Bereitstellung der übergreifenden Komponenten des UIS BW in einer diensteorientierten Architektur ermöglichen, wodurch eine bessere Modularisierung und damit Wiederverwendbarkeit bzw. Kombinierbarkeit von Systemteilen in Form von Diensten angestrebt wird. Hierdurch soll dazu beigetragen werden, die Komplexität und Heterogenität des UIS BW beherrschbarer zu gestalten.

Als besonders vielversprechende Technologiekombination wird hierbei die Verbindung von Techniken und Werkzeugen des World-Wide Webs (WWW), wie Web-Browser, HTML/HTTP, Java und der Common Object Request Broker Architecture (CORBA) der Object Management Group (OMG) /15/ angesehen. Während WWW-Technologie besonders für den clientseitigen plattformunabhängigen Zugriff auf Server-Ressourcen geeignet ist, bietet CORBA die Verdeckung serverseitiger Heterogenität, kann also konkret sehr gut zur serverseitigen Integration von UIS-Diensten dienen. Die diesbezüglichen Eigenschaften von CORBA wie z.B. IDL (Interface Definition Language) werden ausführlicher in /6/, /7/, /8/, /13/ und /18/ diskutiert.

Im Rahmen dieser Arbeit wird die Integration und Kombination zweier bedeutsamer Arten von UIS-Diensten in einer derartigen diensteorientierte UIS-Architektur auf der Basis von CORBA- und WWW-Technologie untersucht. Bei diesen Dienstearten handelt es sich zum einen um sogenannte technische Dienste, konkret um Module eines Simulationssystems zur Ausbreitung luftgetragener Spurenstoffe. Zum anderen handelt es sich um Datenbankzugriffsdienste, hier Zugriffe auf das kommerzielle Datenbanksystem Oracle, das eine der zentralen Datenhaltungskomponenten im UIS darstellt. Die Datenbankzugriffsdienste dienen hier der Bereitstellung der notwendigen Eingabeparameter (Luftmeßwerte) für die Simulationsrechnung. Die Dienste werden als entsprechende CORBA-Objekte (Server) bereitgestellt. Die Kombination der Dienste wird anhand eines einfachen Demonstrationssystems vorgeführt.

Die besonderen Herausforderungen in dieser Aufgabenstellung ergeben sich hierbei aus den technischen Randbedingungen, woraus sich wiederum der wesentliche Nutzen dieser Arbeiten für das UIS BW ergibt. Die in dieser Arbeit betrachteten Komponenten sind auf einer Reihe heterogener Betriebssystemplattformen implementiert, und zwar Sun Solaris (Sparc-Rechner), Digital Unix (Alpha-Rechner), Digital VMS (VAX-Rechner) und Windows NT (PC), die i.w. die typischen Plattformen der UIS-Komponenten darstellen /16/. Zur Demonstration laufen die Komponenten verteilt im Internet zwischen Stuttgart (IKE und UM) und Karlsruhe (FZI)

ab. Hierdurch wird die Eignung von CORBA zur transparenten Integration heterogener, sogar in Weitverkehrsnetzen verteilter Dienste für das UIS demonstriert. Ferner werden bei der Ergebnismeldung an den WWW-Client sowohl „klassische" WWW-Technologie (HTML/HTTP/CGI) eingesetzt, als auch ein sogenannter Java-ORB. Dies ist ein CORBA Object Request Broker, der als Java-Applet in einen WWW-Browser geladen werden kann und sich dann über ein CORBA-eigenes Protokoll (IIOP: Internet Inter ORB Protokoll) mit einem ORB auf Serverseite verbinden kann. Da für den Java-ORB und den ORB auf Serverseite ORBs verschiedener Hersteller eingesetzt werden – Visibroker for Java /21/ bzw. Orbix 2.0.1 /20/ –, wird hierdurch für das UIS einerseits die Interoperabilität zwischen CORBA-Implementierungen verschiedener Hersteller und andererseits die sehr gute Kombinierbarkeit von CORBA und Java-Technologie praktisch demonstriert.

Neben dieser umfassenden Demonstration der sinnvollen praktischen Einsetzbarkeit der genannten Technologien für das UIS BW werden ferner wesentliche Elemente der Konzepte der AG Dienste /18/, z.B. systemnahe Metainformationen zur Beschreibung von Diensteschnittstellen und -ergebnissen, eingesetzt. Nicht zuletzt stehen dem UIS nun mehrere komplexe Dienste zu Verfügung, die Windfeldberechnung (NOABL), die Ausbreitungrechnung (PAS) und ein Zugriff auf die Windmeßwert-Datenbank. Die hier eingesetzten Techniken und Konzepte zur Integration der Dienste mittels sogenannter Wrapper in CORBA und ihre Anbindung an WWW-Technologie sind darüber hinaus sehr gut in zukünftige Arbeiten für das UIS übertragbar.

Das Dokument ist wie folgt gegliedert: In **Kapitel 2** wird die für diese Arbeit Architektur ausgewählt und vorgestellt und entsprechende Alternativen diskutiert. In **Kapitel 3** werden die wesentlichen Entwurfs- bzw. Implementierungsentscheidungen bei den einzelnen Systemkomponenten bzw. Ihre Umsetzung vorgestellt. **Kapitel 4** zeigt die Realisierung des oben skizzierten Demonstrationssystems. Den Abschluß bildet **Kapitel 5** mit einer Zusammenfassung und einem Ausblick auf mögliche Folgearbeiten.

2. Architektur und Idee

Das in diesem Projekt implementierte Demonstrationssystem soll über CORBA als Middleware prototypisch Dienste zur Verfügung stellen, die auf der bereits vorhandenen Simulationssoftware MESYST und dem Datenbanksystem Oracle basieren. Dazu ist zunächst festzustellen, welche Einzelkomponenten sich bei diesen Anforderungen manifestieren lassen. In einem zweiten Schritt werden die daraus resultierenden Alternativen diskutiert. Als letzter Schritt kann dann eine konkrete Entscheidung für ein bestimmte Architektur getroffen werden.

2.1 Übersicht

In die Konzeption des Systems fließen folgende Aspekte mit ein:

- *Verteilung:* Die Vision des UIS sieht vor, daß Dienste auf vernetzten Rechnern angeboten werden.
- *Heterogenität:* Im UIS werden unterschiedlichste Rechnerplattformen eingesetzt.

- *Restrukturierung:* Die anzubietenden Dienste basieren zum einen auf einer bereits existierenden Simulationssoftware und zum anderen auf einem ebenfalls existierenden kommerziellen Datenbanksystem. Das impliziert jeweils eine Vermittlerkomponente, um den von den Anwendungen vorgegebenen Schnittstellen gerecht zu werden.

Die ersten beiden Punkte werden gemeinsam betrachtet, da die Heterogenität z.T. aus der Verteilung resultiert. Der dritte Aspekt wird in zwei Abschnitten genauer beleuchtet – einer, der die Simulationssoftware untersucht und einer, der auf die Probleme der Datenbanksoftware eingeht.

2.1.1 Generelle Anforderungen in verteilten heterogenen Informationssystemen

In verteilten heterogenen Informationsystemen sind im wesentlichen zwei Probleme zu bewältigen: (1) die Systemkomponenten müssen untereinander kommunizieren und kooperieren, was mitunter aufwendiger ist als in einem herkömmlichen monolithischen System und (2) unterschiedlichste Systemkomponenten müssen transparent und über ein vereinbartes Protokoll miteinander Daten austauschen können. Als Lösung zur transparenten Kommunikation von Rechnerplattformen unterschiedlicher Hersteller wird hier CORBA eingesetzt. Damit schließt sich größtenteils die Lücke zwischen lokalen und verteilten Anwendungen. Damit lehnt sich dieses Projekt sehr eng an die in der AG Dienste erarbeiteten Vorschläge und Empfehlungen an /18/.

2.1.2 MESYST – Beispiel eines komplexen technischen Dienstes

Das modulare Simulationssystem MESYST zur Simulation der Ausbreitung luftgetragener Spurenstoffe besteht aus insgesamt 19 Programm-Modulen, die eine Sammlung verschiedener, komplexer Simulationsmodelle enthalten und teilweise ähnliche bzw. identische Aufgabenstellungen alternativ bearbeiten. Dies ermöglicht es, aus den Modulen unterschiedliche Konfigurationen zu bilden, um spezielle Simulationsmodelle für spezielle Anwendungsfälle zu erstellen. Wegen der teils erheblichen Größe der Simulationsmodelle wird der Einsatz spezieller Hardware wie Vektor- oder Parallelrechner unterstützt.

Die Simulation luftgetragener Spurenstoffe mit MESYST gliedert sich in folgende Teile:

- Grunddatendefinition
- Datenbeschaffung
- Windfeldberechnung
- Ausbreitungsrechnung
- sonstige Berechnungen (optional), z.B. Dosisberechnung

Die Definition von Grunddaten erfolgt durch den Benutzer der Simulation. Der Benutzer legt dabei z.B. den Ort einer Emmisionsquelle fest und wählt ein Gebiet, für das die Simulation durchgeführt werden soll.

Als Datenbeschaffung wird der Vorgang bezeichnet, durch den alle für eine Simulation benötigten Eingabedaten besorgt werden. Zu den Eingabedaten gehören z.B. die Topographiedaten des Simulationsgebietes und meteorologische Daten wie Windmeßwerte

oder Niederschlagsmeßwerte im Simulationsgebiet. Die Datenbeschaffung ist die für den Anwender die aufwendigste Tätigkeit zur Durchführung einer Simulation mit MESYST, da verteilte und große Datenmengen beschafft werden müssen.

Für Windfeldberechnung, Ausbreitungsrechnung oder sonstige Berechnungen stehen in MESYST jeweils spezialisierte Programmodule zur Verfügung /11/.

Der vorliegende Integration von MESYST in eine diensteorientierte Umgebung sollte im Rahmen von Globus ein durchaus hoher Stellenwert eingeräumt werden, da anhand dieses komplexen Beispiels zum einen einige vom Dienstekonzept zu erwartetende Verbesserungen praktisch nachgewiesen und zum anderen praktische Erfahrungen für die weitere Evolution des UIS BW gesammelt werden können. Die speziell für die Benutzbarkeit von MESYST zu erwartenden Vorteile sind zum einen die wesentlich vereinfachte Datenbeschaffung in einer hierfür ausgelegten Diensteumgebung, die die Anwendbarkeit von MESYST wesentlich vereinfacht. Zum anderen ergeben sich durch die Verfügbarmachung von MESYST als Dienst und die dadurch gesteigerte Verfügbarkeit eventuell weitere Anwendungmöglichkeiten. Außerdem kann der für eine Simulation erforderliche Resourcenbedarf (Rechenleistung, Hauptspeicher) in einer verteilten, diensteorientierten Umgebung wegen deren prinzipieller Skalierbarkeit leichter zur Verfügung gestellt werden, als auf einem einzigen zentralen Rechner.

2.1.3 Datenbanken in Informationssystemen

Datenbanksysteme ermöglichen es, große Datenmengen effizient zu speichern und zu verwalten. Sie erlauben darüber hinaus, die Struktur der Daten in Form von Metadaten ebenfalls zu speichern /5/. Damit sind sie in einem Umweltinformationssystem, in dem eine große Anzahl von Meßwerten in verschiedensten Datenformaten auftreten, in idealer Weise einsetzbar.

Allerdings werden die Daten im UIS in unterschiedlichen Datenbanksystemen an unterschiedlichen Orten gehalten und verwaltet. An dieser Stelle ist ein netzweiter Zugriff auf die Daten sinnvoll. Im Kontext dieses Projektes soll die Simulation die als Eingabe nötigen Meßwerte direkt aus einer Datenbank erhalten, wo sie nach der Messung abgelegt werden. Bislang muß der Benutzer die Eingabedaten für die Simulation noch selbst besorgen.

2.2 Diskussion möglicher Alternativen

Bei der Frage möglicher Architekturvarianten sind zwei Punkte detailierter zu diskutieren. Die erste ist mögliche Aufteilung von MESYST in Dienste und die zweite ist die Frage nach den Alternativen der Datenbankanbindung. Wir nehmen bei der folgenden Diskussion ein Client an, der eine Simulation durchführen möchte und dazu Luftmeßwerte als Eingabedaten aus einer Datenbank benötigt.

2.2.1 Integration von MESYST in eine diensteorientierte Umgebung

Für die Integration von MESYST in eine diensteorientierte Umgebung bieten sich mehrere Möglichkeiten:

1. Einbinden einer oder mehrerer Komplettkonfigurationen von MESYST als Dienst
2. Einbinden der Einzelmodule von MESYST als separate Dienste

Lösung 1 hat den Vorteil einer einfacheren und schnelleren Realisierbarkeit. MESYST bleibt als Ganzes erhalten und muß nur geeignet gekapselt werden. Damit erhält die Simulation aber nur „neue Kleider" und weicht vom Konzept der Diensteorientierung, einzeln sinnvolle Funktionalität als eigenen Dienst anzubieten, ab. Diese Lösung ist somit unflexibler und die Nutzung von MESYST-Komponenten für andere Aufgaben ist nur mit einem gewissen Overhead möglich. Außerdem ist das Problem der Datenbeschaffung, bisher muß der Anwender die Eingabedaten selbst beschaffen, in dieser Lösung nicht entscheidend verbessert worden.

Lösung 2 ist hingegen die flexiblere Variante, allerdings auch aufwendiger zu realisieren. Da der Vorteil durch die höhere Flexibilität den Nachteil des höheren Aufwands überragt, wurde diese Lösung favorisiert. Um den Umfang des Projektes nicht zu sprengen werden jedoch zunächst nur zwei Teilkomponenten von MESYST als eigenständige Dienste realisiert.

2.2.2 Integration des Datenbankzugriffs

Zur Integration des Datenbankzugriffs und Kombination mit der Simulation stehen zwei mögliche Alternativen zur Auswahl:

1. Zugriff vom Client aus
2. Zugriff von der Simulation aus

Bei der ersten Zugriffsvariante, die in Abbildung 1 (a) skizziert ist, besorgt der Client zunächst die Daten von der Datenbank und übergibt sie beim Aufruf der Simulation an diese. Die Vorteil sind eine einfachere und schnellere Realisierung und eine einfachere Kapselung der Simulation. Die Nachteile sind eine geringere Performanz und Ökonomie, da die Daten zweimal übertragen werden müssen. Außerdem muß der Client zusätzliches Wissen über die Daten haben, obwohl für ihn nur die Ergebnisse der Simulation von Interesse sind.

In Abbildung 1 (b) ist die zweite genannte Zugriffsalternative dargestellt. Hier wendet der Client sich nur an die Simulation, die sich dann die benötigten Daten direkt aus der Datenbank besorgt. Die Vorteile sind ein deutlich einfacherer Client, ein ökonomischere, weil einmalige, Übertragung der Daten und eine bessere Konformität mit dem Dienstekonzept. Nachteilig wirkt sich allerdings die aufwendigere Schnittstelle für verschiedene Parameter auf Seiten der Simulation aus, da hier der Simulationsdienst den Datenbankzugriffsdienst nutzt.

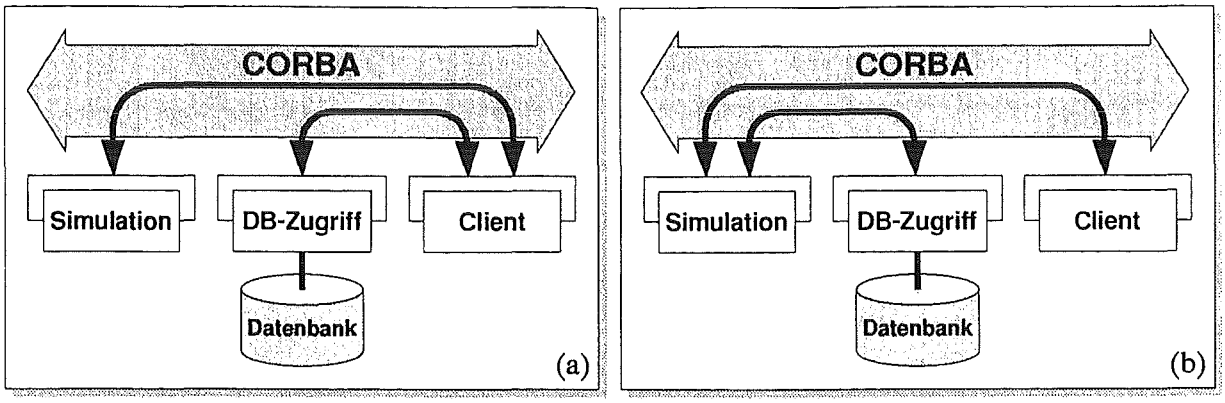


Abbildung 1: Alternativen des Datenbankzugriffs; (a) vom Client aus, (b) von der Simulation aus.

Die Entscheidung fiel auf die Variante des von der Simulation ausgehenden Zugriffs. Die Vorteile überwiegen und die gewählte Alternative paßt sich nahtloser in das Prinzip der Diensteorientierung ein.

3. Entwurf und Implementierung

Zur Integration bestehender Datenquellen und Softwaresysteme in CORBA-basierte Architekturen, für deren Programmiersprache keine IDL-Anbindung definiert ist oder für die kein Quellcodezugriff möglich ist, bedient man sich sogenannter IDL-Wrapper. Ein Wrapper konvertiert die Schnittstelle einer Anwendung oder eines Moduls in eine Schnittstelle, die ein Client zum Arbeiten benötigt /14/. Ein Wrapper ermöglicht damit einem Klienten indirekt auf Objekten zu operieren, deren Schnittstelle er ohne den entsprechenden Wrapper aus Inkompatibilitätsgründen nicht benutzen könnte. Es empfiehlt sich hierbei jedoch, beim Kapseln einer Komponente stärker von deren Schnittstelle zu abstrahieren, um die neue Wrapper-Schnittstelle von der darunterliegenden, gekapselten Implementierung unabhängig zu halten. Hierdurch können Komponenten mit ähnlicher oder gleicher Funktionalität unter einem Wrapper, welcher hierzu eine allgemeine Schnittstelle anbietet, zusammengefaßt werden. Darüberhinaus können Komponenten gegeneinander ausgetauscht werden, ohne die aufrufenden Programme, welche jene verwenden, ändern zu müssen. Desweiteren bieten Wrapper die Möglichkeit, die Fähigkeiten des gekapselten Objekts zu erweitern, indem sie dynamisch Funktionalität zum Objekt hinzufügen und diese an ihrer Schnittstelle nach außen anbieten.

3.1 Wrapper für Simulationsmodule

Das im Rahmen dieses Teilprojekts vorgenommene Aufbrechen von MESYST in einzeln nutzbare Teilmodule und die Bereitstellung dieser Module als Dienste in einer CORBA-basierten Umgebung ist keine einfache Aufgabe. Schwierigkeiten ergeben sich durch (siehe auch /17/):

- die Bereitstellung der für die Simulation benötigten heterogenen Datensätze
- die Verwaltung der durch die Simulation erzeugten großen Datenmengen

- die i.d.R. langen Ausführungszeiten für Simulationsdienste, die es erfordern, daß ein Dienst sowohl von mehreren Clienten gleichzeitig als auch asynchron benutzt werden kann, um die Clienten nicht zu blockieren
- die hierarchische Unterteilung eines Dienstes in Unterdienste, die sich gegenseitig nach einem bestimmten Muster benutzen müssen.
- die Wartung und Weiterentwicklung dieser Dienste, die ein Versionsmanagement erfordert, um auch auf ältere Versionen eines Dienstes zugreifen zu können. Dies ist z.B. für die Reproduzierbarkeit und Vergleichbarkeit von Simulationsergebnissen wichtig.

Um das Kapseln von Simulationsmodulen zu unterstützen, wurde am IKE das auf CORBA aufbauende OODS-Framework (Object Oriented Distributed Simulation) entwickelt. Das OODS Framework unterstützt durch seine Basisklassen die nichtblockierende und nebenläufige Simulationsausführung sowie die Erzeugung von hierarchisch geschachtelten komplexen Simulationsdiensten. Ferner stellt OODS Basismechanismen für die Verwaltung von Simulationsdiensten und -ergebnissen zur Verfügung. Durch die Nutzung der in OODS enthaltenen Konzepte konnte das Kapseln der MESYST-Simulationsmodule wesentlich vereinfacht werden.

Für das Kapseln der Simulationsmodule stehen die Basisklassen für Simulationsmodelle in OODS zur Verfügung. Die Basisklassen definieren Schnittstellen und grundlegende Methoden, die für Simulationsdienste benötigt werden. Zur Erstellung der Wrapper für die MESYST-Module werden von den Basisklassen des OODS-Frameworks für MESYST-Module spezialisierte Klassen durch Vererbung abgeleitet. Durch die Vererbung können eine Reihe wichtiger Funktionalitäten aus dem Framework einfach wiederverwendet werden. Diese sind neben den für die Integration in ein CORBA-basiertes System benötigten Kommunikationsmechanismen hauptsächlich die zur Kombination mehrerer Simulationsmodelle benötigten Schnittstellen und Protokolle.

Zu den Verwaltungsmechanismen gehören eine Komponente zur Erzeugung von Simulationsdiensten aus Klasseninformationen (sog. *Factory*) sowie ein Repository für Simulationsdienste und -ergebnisse.

Die explizite Verwaltung von Dienste-Klassen für Simulationsdienste ist ein wichtiges Leistungsmerkmal von OODS. Durch die Definition von Diensteklassen kann ein Dienstanbieter verschiedene Typen oder Konfigurationen von Diensten, die aus Grundbausteinen zusammengesetzt sind, definieren. Dies ist für die Anwendbarkeit von Simulationssystemen wie MESYST, die von der Anlage her eher einem Baukasten als einer kompletten Anwendung ähnlich sind, von großer Bedeutung. Der Dienstnutzer entscheidet sich zunächst für den zu benutzenden Diensttyp. Für die eigentliche Benutzung des Dienstes wird dieser zunächst aus der Klasseninformation dynamisch erzeugt, bevor die Funktionalität des Dienstes ausgeführt wird. Das Klassenkonzept kann darüber hinaus auch zur Versionsverwaltung von Simulationsdiensten verwendet werden, falls für jede bestimmte Version eines Dienstes genau eine Klasse definiert wird.

Das OODS-Framework ist in /9/, /10/ allgemein beschrieben. Eine genaue Beschreibung der Wrapper für die MESYST-Module findet sich bei /11/.

3.2 Integration anderer Kommunikationsmechanismen am Beispiel KIP

Das UIS BW ist ein äußerst heterogenes System, in dem sehr viele verschiedene Hard- und Softwareplattformen sowie Techniken integriert werden müssen. Dies erfordert, zusammen mit der Tatsache, daß CORBA eine noch relativ junge Technologie ist, die sich derzeit erst beginnt durchzusetzen, daß neben CORBA auch noch andere Konzepte zur Diensteintegration Verwendung finden müssen. Ein Ziel ist dabei die möglichst nahtlose Kombination dieser Konzepte.

Ein zur Integration von Diensten entwickeltes Konzept ist der Kommunikationsinterpreter (KIP) /12/. Da derzeit für VAX/VMS-Systeme ein dem aktuellen CORBA-Standard entsprechendes Produkt erst angekündigt, aber noch nicht verfügbar ist, wird ein Dienst mit Hilfe des KIP integriert und dadurch auch die VMS-Plattform in die Demonstration mit eingeschlossen.

Wie für die anderen Simulationsdienste wurde auch für diesen Dienst eine spezialisierte Klasse aus OODS abgeleitet. Anstatt jedoch den Dienst direkt als Funktionalität dieser Klasse zu implementieren, realisiert diese Klasse nur eine Stellvertreterfunktion (sog. *Proxy*). Dies ermöglicht es, den Dienst framework-konform zu implementieren, d.h. die Vorteile und Möglichkeiten von OODS und CORBA können genutzt werden. Da die Implementierung des Dienstes komplett hinter der Schnittstelle der Klasse verborgen ist, kann gleichzeitig intern KIP verwendet werden, um die Funktionalität des Dienstes auf einem entfernten System, z.B. einem VAX/VMS-System, zu nutzen.

3.3 Wrapper für Datenbank-Zugriffe

Der durch das FZI realisierte Datenbankzugriff stellt einen eigenständigen Dienst im UIS in Form eines CORBA-Objektes dar. Da es im Rahmen eines Demonstrationssystems als nicht praktikabel erschien, einen Zugriff auf eine real im UIS existierende Datenbank für Windmeßwerte zu realisieren, wurde statt dessen eine kleine Datenbank speziell für das Demonstrationssystem implementiert. Dadurch ist der implementierte Datenbankzugriffs-dienst im UIS so nicht direkt einsetzbar, da er als Ganzes nur eine beispielhafte Realisierung darstellt. Der Wrapper für den Datenbankzugriffsdienst kann jedoch auf jeden Fall für reale Datenbankzugriffe verwendet werden. Dafür muß sie nur auf einen Datenbank-Server mit einer realen Datenbank portiert werden. Außerdem kann dieser Dienst auch gut als Schablone für weitere Datenbank-Zugriffe dienen.

3.3.1 Datenbanksysteme in einer CORBA-Umgebung

Der Datenbank-Wrapper abstrahiert von dem eingesetzten Datenbanksystem und entkoppelt den Rest des Systems von der Auswahl und den Eigenheiten eines speziellen Datenbankproduktes.

Im Rahmen dieser Arbeit wurde das relationale Datenbanksystem Oracle eingesetzt. Datenanfragen können über die Datenbankanfragesprache SQL gestellt werden.

Um die Datenbank in eine CORBA-Umgebung zu überführen, muß in der Wrapper-Komponente eine Abbildung zwischen in IDL definierten Methoden auf CORBA-Seite und SQL-Anfragen auf auf Datenbankseite durchgeführt werden. Dazu stehen in unserem Fall folgende zwei Möglichkeiten zur Auswahl:

1. Eine Methode erhält die SQL-Anfrage als String-Parameter.
2. Ausgewählte Datenbankanfragen werden als eigene Methoden statisch realisiert.

Die erste, *dynamische* Variante (Abbildung 2 (a)) bietet eine maximale Flexibilität, da uneingeschränkte SQL-Anfragen gestellt werden können. Die Bindung an SQL ist aber auch ein Nachteil bei der Abstraktion, da andere Datenbanksysteme, z.B. objektorientierte u.U. kein SQL anbieten. Außerdem kann im Wrapper fast keine Fehlerüberprüfung stattfinden und die von IDL gebotenene Typsicherheit wird mit dieser Methode schlicht umgangen.

Die zweite, *statische* Variante (Abbildung 2 (b)) bietet eine IDL-Typisierung. So können einige Fehler schon zum Übersetzungszeitpunkt gefunden werden. Darüberhinaus ist diese Lösung effizienter, da die SQL-Anfragen mit dem Serverprogramm übersetzt und optimiert werden können. Daraus ergibt sich der einzige nennenswerte Nachteil: durch die feste Verdrahtung der SQL-Anfrage muß bei Änderungen auf dieser Ebene der komplette Server neu übersetzt werden.

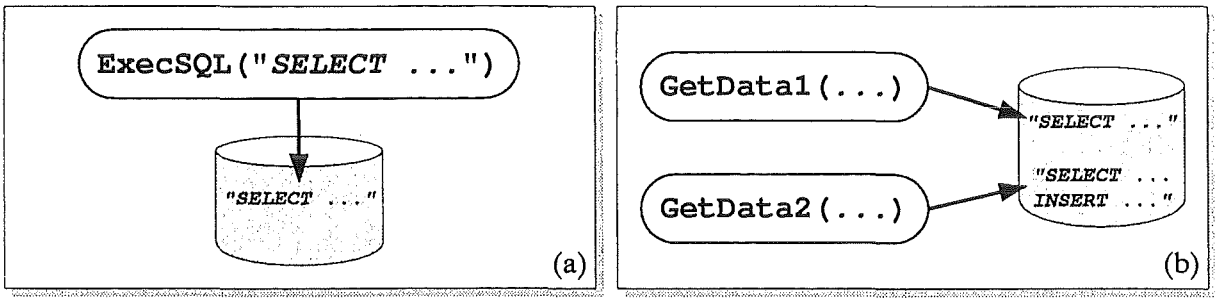


Abbildung 2: Abbildungmöglichkeiten von CORBA-Methoden auf SQL-Anfragen; (a) dynamische Variante, (b) statische Variante.

Da für unser System keine bedeutende Flexibilität gefordert ist, wurde Variante 2 der Vorzug gegeben. Damit entfällt auf Seiten des Clients zusätzlich jede Notwendigkeit, sich mit SQL oder den Details der Datenbasis (z.B. Metadaten) auseinanderzusetzen. Eine ausführliche Diskussion dieser und anderer Varianten des Datenbankzugriffs kann in /3/, /18/ vertieft werden.

3.3.2 IDL-Schnittstelle für den Datenbankzugriff

Der Datenbank-Wrapper ist ein CORBA-Objekt, dessen IDL interface `Access` heißt. Sie soll für ein gegebenes Gebiet und für einen gegebenen Zeitraum alle Meßwerte liefern. Um den Zugriff feingranularer und zugleich ökonomischer zu machen, wurde ein zweistufiges Protokoll zwischen Client und Server entworfen. Die zwei Phasen des Datenzugriffs, die sich auch in zwei Methoden wiederfinden, sind:

1. `getSites()`: Liefert zuerst alle *Meßstellen* in einem *Gebiet*
2. `getValues()`: Liefert zu einer *Meßstelle* und einem *Zeitraum* alle *Meßwerte*

3.3.3 Einige Details der Datenbankanbindung

Um die eigentliche Anbindung des konkreten Datenbanksystems (hier Oracle) weiter vom CORBA-Bereich zu entkoppeln, wurde der Wrapper zweigeteilt. Der unterste Teil bindet die Datenbank und deren Rückgabewerte in die Programmiersprache (hier C++) ein. Der obere benutzt den unteren und wandelt – wenn notwendig – die C++-Typen in CORBA-Typen um. So wird der ganze Wrapper in die CORBA-Umgebung eingebunden (siehe Abbildung 3).

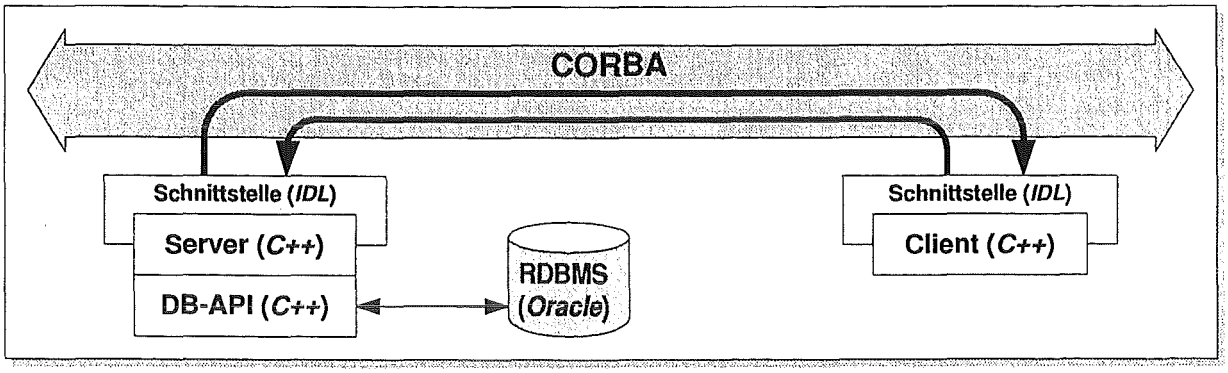


Abbildung 3: Schema der Datenbankanbindung.

Die Datenbankschnittstelle wurde mit C++ und einem Embedded-SQL-Precompiler von Oracle erstellt. Dabei können SQL-Anfragen direkt mit C++-Code gemischt werden. Der Oracle-Compiler übersetzt zunächst die Anfragen in C++-Code, so daß der C++-Compiler das übrige leisten kann.

3.4 Eine Komponente zur Ereignisvisualisierung

Diese Komponente ermöglicht es, den Status der mitunter lange laufenden Anwendung zu visualisieren. Sie stellt die einfachste Art eines Monitors (siehe auch Abschnitt 5.1) dar und wird über einfache Methodenaufrufe, die den Namen der Komponente und deren Status als String erhalten, angesteuert.

Realisiert wurde der Ereignisvisualisierer als Java-Applet, das durch eine Java-Anbindung an CORBA (VisiBroker) mit den anderen Komponenten indirekt über das IIOP kommuniziert. Dazu wird in der Orbix-Domäne zusätzlich ein Objekt benötigt, welches das Visualisierungsobjekt bzw. eine interoperable Objektreferenz darauf kennt und Ereignisse an dieses weiterleitet. Dabei wird die im CORBA2-Standard /6/ vorgeschlagene Vorgehensweise aufgegriffen; das Vermittlerobjekt funktioniert dabei als sog. *Half-Bridge*. Auf diesem Wege kann das Überschreiten der Grenzen zwischen unterschiedlichen ORBs weitestgehend transparent gehalten werden.

Interessant ist hier auch der Einsatz des IIOP, das letztenendes die CORBA-Verbindung von Objekten über ORBs verschiedener Hersteller hinweg schafft. Außerdem hat man mit der Java-Anbindung auch einen Brückenschlag zwischen der WWW-Welt und der CORBA-Welt erreicht, der im Gegensatz zur einer HTML/CGI basierten Lösung nicht mehr nur auf einem zustandslosen Protokoll basiert.

4. Realisierung eines Demonstrationssystem

Die in den vorangegangenen Abschnitten erläuterten Dienste wurden vom FZI und IKE entsprechend der jeweiligen Kompetenzbereiche unabhängig entwickelt. Um die praktische Verwendbarkeit von CORBA zur Implementierung einer Diensteumgebung zu demonstrieren, wurde gemeinsam ein realitätsnahes Anwendungsszenario entworfen und ein Demonstrationssystem implementiert.

Das Anwendungsszenario umfaßt die Benutzung eines entfernten Dienstes zur Schadstoffausbreitungssimulation, im folgenden *MESYST-Demo* genannt, der aus einem Subset der MESYST-Module (NOABL und PAS) konstruiert ist. Dabei wird davon ausgegangen, daß die Lokation des Dienstes bekannt ist. Das Szenario umfaßt weiterhin die Benutzung von gemessenen und in Datenbanken gespeicherten Windmeßwerten als Eingabedaten für den Simulationsdienst. Da der Zugang zu einer real existierenden Datenbank für Demonstrationszwecke als zu aufwendig eingeschätzt wurde, ist am FZI eine kleine Demo-Datenbank auf Basis von Oracle implementiert worden. Alle weiteren, während der Datenbeschaffung für MESYST zu beschaffenden Daten werden als bekannt vorausgesetzt. Prinzipiell können aber auch die außer den Windmeßwerten benötigten Daten, z.B. Topographiedaten, in analoger Weise mit Diensten beschafft werden, sofern solche Dienste verfügbar sind.

Das Demonstrationssystem bzw. seine Komponenten wurden auf verschiedenen Rechnern am FZI und IKE implementiert. Die dabei eingesetzten Plattformen sind sehr verschieden. Sie reichen von Window NT für den Arbeitsplatz bzw. die Benutzeroberfläche der Demonstration, über UNIX in Form von Sun Solaris 2.4 und 2.5 auf unterschiedlichen Sun SparcStations sowie Digital UNIX 3.2 auf Digital Alpha bis hin zu Digital VAX/VMS. Dies unterstreicht die Handhabbarkeit einer heterogenen Umgebung mit CORBA.

4.1 Komponenten des Demonstrationssystems

Das vorgestellte Demonstrationssystem umfaßt viele Komponenten, auf die im folgenden kurz eingegangen werden soll. Die Abbildung zeigt die Komponenten des Demonstrationssystems im Überblick. Es werden drei Arten von Komponenten unterschieden. Erstens lokal am Arbeitsplatz vorhandene Komponenten ohne direkten Zugang zu den Diensteumgebungen. Beispiele für diese Komponenten sind die Applikationen der Bürokommunikation, also MS Office oder ArcView. Zweitens lokal am Arbeitsplatz vorhandene Komponenten mit direkten Zugang zu den Diensteumgebungen. Beispiele hierfür sind WWW-Browser im Zusammenspiel mit Java Applets oder speziell dafür eingerichtete WWW-Server. Drittens die eigentlichen Dienste, die entfernt vom Arbeitsplatz von speziellen Server-Rechnern angeboten werden.

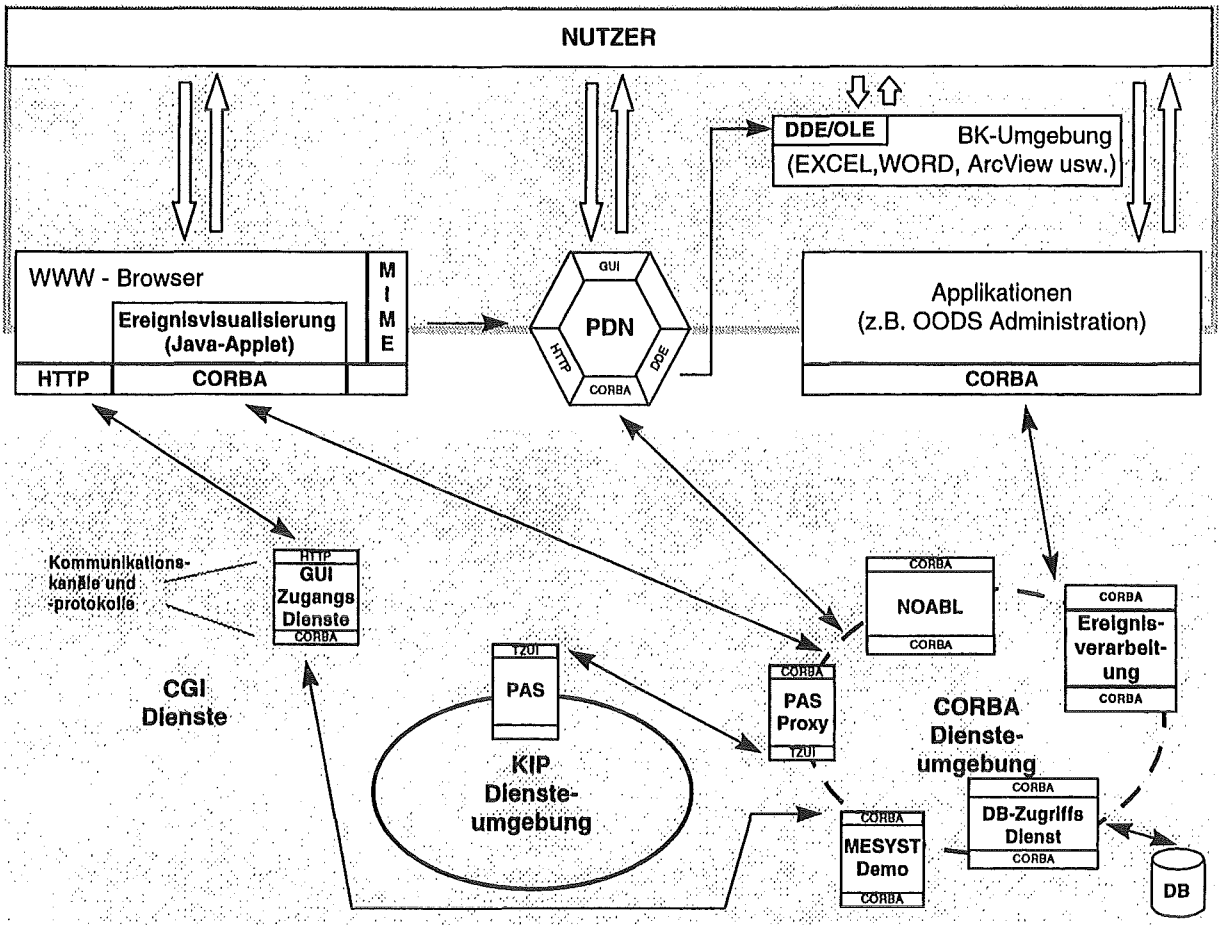


Abbildung 4: Komponenten des Demonstrationssystems

4.1.1 Der Arbeitsplatz

Der Arbeitsplatz, von dem aus das Demonstrationssystem benutzt werden soll, ist eine zentrale Komponente des Demonstrationssystems, da er das System für den Benutzer repräsentiert. Die AG Dienste weist in ihrem Bericht /17/ auf die möglichen Veränderungen des typischen UIS Arbeitsplatzes in einem diensteorientierten UIS hin. Im Demonstrationssystem wird ein Arbeitsplatz auf der Basis von Windows NT vorgestellt, wie er in einem diensteorientierten UIS denkbar ist. Die zentrale Komponente dieses Arbeitsplatzes ist der am IKE entwickelte Personal Data Node (PDN). Der PDN ermöglicht es, die von den benutzten Diensten erzeugten Daten mit unterschiedlichen, lokal am Arbeitsplatz verfügbaren Werkzeugen zu verknüpfen. Eines dieser Werkzeuge ist das Geo-Informationswerkzeug ArcView. Mit ArcView soll eine Visualisierung der vom Dienst der Ausbreitungssimulation berechneten Daten erfolgen. Der Zugang zum Demonstrationssystem (Dienstanforderung) wird über ein System von WWW-Seiten gewährleistet, so wie dies bereits heute auch von anderen Anwendungen im UIS her bekannt ist. Deswegen gehört auch ein WWW-Browser zur Darstellung der WWW-Seiten und Ausführung von Java-Applets zur Ausstattung des Arbeitsplatzes.

4.1.2 WWW-Server

Um die WWW-Seiten für die Dienstanforderungen zur Verfügung zu stellen wird ein WWW-Server benötigt. Die für das Demonstrationssystem verwendeten WWW-Seiten sind durchweg von CGI-Programmen erzeugte, dynamische Seiten. Über die selben CGI-Programme wird auch der Zugang zu den anzusprechenden Diensten über CORBA realisiert. Der WWW-Server für das Demonstrationssystem wird am IKE auf einer Sun SparcStation unter Solaris 2.5 betrieben.

4.1.3 Simulationsdienste

Die Simulationsdienste werden von einem am IKE mit Hilfe des in Abschnitt 3.1 beschriebenen OODS-Framework entwickelten Simulationsserver angeboten. Der Zugang zu diesem Server erfolgt mit CORBA über den vorgestellten WWW-Server. Der Simulationsserver wird auf einem DEC Alpha AXP Server unter Digital UNIX ebenfalls am IKE betrieben.

Die Konstruktion korrekter Simulationsmodelle für komplexe Anwendungen wie die Ausbreitungssimulation ist prinzipiell Expertenarbeit. Es dürfen in MESYST zum Beispiel nur bestimmte Modelle miteinander gekoppelt werden. Werden nicht für einander vorgesehene Modelle gekoppelt, kann eine Simulation nicht oder nur fehlerhaft durchgeführt werden. Dem Dienstenutzer muß daher ein korrekt konfigurierter Simulationsdienst angeboten werden.

Das OODS-Framework unterstützt zu diesem Zweck die hierarchische Konfiguration von Simulationsdiensten aus Komponenten durch Diensteklassen. Für die Demonstration wurde daher die Diensteklasse MESYST-Demo geschaffen. Diese Diensteklasse definiert einen Simulationsdienst bestehend aus den Unterdiensten NOABL und PAS. Außerdem werden die Verbindungen zwischen den Unterdiensten beschrieben. Im vorliegenden Beispiel wird zum Beispiel angegeben, daß PAS von NOABL Windfelder anfordern kann.

Die Funktionalität des Simulationsdienstes NOABL ist ebenfalls auf dem erwähnten Digital Alpha AXP Server implementiert. Deshalb wird der Dienst NOABL direkt von diesem Server erbracht. Der Simulationsdienst PAS ist auf einem VAX/VMS Server, ebenfalls am IKE, implementiert. Daher verwaltet der Simulationsserver, wie in Abschnitt 3.2 (3.2 Integration anderer Kommunikationsmechanismen am Beispiel KIP) dargestellt, nur ein Proxy für den Simulationsdienst PAS. Das Proxy benutzt intern, für den Dienstanforderer unsichtbar, KIP, um den Dienst auf dem VAX/VMS-Server anzusprechen.

4.1.4 Datenbankdienste

Der Dienst NOABL benötigt als eine seiner Eingaben Windmeßwerte. Diese werden im Demonstrationssystem von einem Datenbankdienst automatisch beschafft. Der Datenbankdienst wurde am FZI entwickelt und wird dort auf einem Sun Sparc-Rechner unter Solaris 2.5 angeboten.

Als Datenbasis dienen z.Zt. noch statisch mit realen Meßwerten erzeugte Datenbanktabellen.

Im Rahmen einer Demonstration ist dieses Vorgehen angemessen weil ausreichend. Für einen späteren Einsatz im UIS sollte der Datenbankdienst natürlich aktuelle Meßwerte besorgen. All das, also auch eine aufwendigere Datenbeschaffung unter Alltagsbedingungen, bleibt dem Dienstnehmer verborgen. Das Wrapperkonzept ermöglicht einen nahtlosen Übergang zu beliebigen Vorgehensweisen bei der Datenbeschaffung.

4.2 Ablauf der Demonstration

In diesem Abschnitt sollen die einzelnen Schritte, die zur Benutzung des prototypischen Simulationsdienstes durchzuführen sind, kurz erläutert werden.

4.2.1 Erzeugen eines neuen Simulationsdienstes

Der erste Schritt zur Benutzung eines Dienstes beinhaltet ganz allgemein das Auffinden desselben. Das Auffinden von Diensten ist prinzipiell die Aufgabe einer Strategiekomponente oder eines Traders /18/ und daher nicht Teil der Demonstration. Es wird vielmehr davon ausgegangen, daß der Ort, d.h. der Rechner, des zu benutzenden Dienstes bekannt ist. Deshalb kann im ersten Schritt der Demonstration sofort aus einer Liste von verfügbaren Simulationsdiensten der gewünschte Dienst MESYST-Demo ausgewählt werden.

An dieser Stelle soll nochmals darauf hingewiesen werden, daß hier zunächst nur eine Dienstklasse ausgewählt werden kann. Nachdem der Anwender die Auswahl bestätigt, wird auf dem Server ein entsprechender Dienst dynamisch erzeugt.

4.2.2 Eingabedatenbeschaffung und Simulation

Nachdem im ersten Schritt ein Dienst für die Benutzung ausgewählt und erzeugt wurde, kann im zweiten Schritt nun eine von der Simulation zu bestimmende Größe ausgewählt werden. Nachdem die Auswahl bestätigt wird, wird der Dienst automatisch gestartet.

Für jeden Simulationsdienst, der mit OODS erzeugt wird, werden automatisch sinnvolle Standardwerte zur Parametrisierung der Simulationsmodelle erzeugt. Für das Demonstrationssystem wurde darauf verzichtet, eine Schnittstelle zur allgemeinen Parametrisierung des Dienstes anzubieten. Eine solche Schnittstelle wird in anderen Arbeiten vom IKE entwickelt /19/.

Der im Rahmen der Demonstration wichtigste eingesetzte Standardwert ist die Benutzung des am FZI implementierten Datenbankdienstes zur Beschaffung der für die Simulation benötigten Windmeßwerte. Im Verlauf der Simulation wird dieser Dienst mehrmals benutzt, um die benötigten Eingaben zu beschaffen. Die Benutzung des Datenbankdienstes findet dabei über ein Weitverkehrsnetz (in diesem Fall das Internet zwischen Karlsruhe und Stuttgart) statt.

Die gegenseitige Benutzung der Dienste orientiert sich dabei automatisch an ihren Abhängigkeiten. Der Dienst PAS benötigt z.B. für seine Funktion ein Windfeld als Eingabe, das im gewählten Demonstrationssystem vom Dienst NOABL berechnet wird. Diese Abhängigkeiten, die die Charakteristik eines bestimmten Simulationsmodells bestimmen, sind

in der Klasseninformation des Diensttyps gespeichert.

4.2.3 Kontrolle der Dienstausführung durch Ereignisvisualisierung

Der Benutzer eines Dienstes muß mitunter lange Antwortzeiten und viele potentielle Fehlerquellen (Netzwerkprobleme, etc.) bei der Inanspruchnahme von Diensten in Kauf nehmen. Um ihn aber über den aktuellen Status der Anwendung nicht im Unklaren zu lassen, soll die vorgeschlagene Komponente zur Ereignisvisualisierung ihm darüber die nötige Information bereitstellen.

Die zur eigentlichen Anwendung zählenden Komponenten, also die Simulationsmodule und der Datenbank-Wrapper, liefern zu sinnvollen Zeitpunkten Statusberichte in Form von Ereignissen über die Orbix-Half-Bridge an die Visualisierungskomponente. Diese nutzt die Fähigkeiten von Java im WWW aus, um die Statusmeldungen dem Benutzer über seinen WWW-Browser zu präsentieren.

4.2.4 Verarbeitung der Simulationsergebnisse

Als Simulationsergebnis gibt der benutzte Simulationsdienst nicht direkt die numerischen Ergebnisse zurück, sondern eine Menge von Metadaten, die die Simulationsergebnisse beschreiben /18/. Der für den Dienstaufruf verwendete WWW-Browser gibt diese Metadaten an den PDN (siehe Abschnitt 4.1.1) durch sog. MIME-Types gesteuert weiter. Mit dem PDN können dann ausgewählte Simulationsergebnisse mit lokal am Arbeitsplatz verfügbaren Werkzeugen weiterbearbeitet werden. In der Demonstration wird die Visualisierung einer Schadstoffwolke mit ArcView durchgeführt.

5. Zusammenfassung und Ausblick

Die Eignung von CORBA als Middlewarekomponente im UIS wurde bereits in der CORBA-Evaluierung und im Projekt WWW-UIS in Globus II /16/ untersucht. Dabei wurde die grundsätzliche Eignung von CORBA für das UIS durch eine praktische Erprobung mehrerer CORBA-Implementierungen und groben Leistungsmessungen in UIS-Szenarien herausgearbeitet. Insbesondere wurde festgestellt, daß viele der in den vorangehenden Abschnitten dargestellten UIS-relevanten Eigenschaften von CORBA in Form von Produktimplementierungen nutzbar sind. Bei der in diesem Bericht beschriebenen Integration heterogener Dienste wurden darüber hinausgehend folgende Erfahrungen gesammelt:

- Das Kapseln und die Benutzung von bestehenden UIS-Diensten in einer CORBA-basierten Diensteumgebung durch Wrapper wurde anhand von Diensten unterschiedlicher Komplexität prototypisch und erfolgreich durchgeführt. Beispiele sind Datenbankzugriffsdienste und Simulationsdienste.
- Bei den Prototyp-Entwicklungen erwies sich CORBA als ein sehr mächtiges, dadurch aber auch anspruchsvolles Werkzeug zur Implementierung von Diensten. In den verwendeten CORBA-Produkten wurden ferner für eine derartig neue Technologie erstaunlich wenige Software-Fehler festgestellt. Zu beachten ist ebenfalls, daß die

Entwicklung der CORBA-Produkte derzeit noch sehr dynamisch ist und bisher noch kein Produkt sämtliche der spezifizierten CORBAservices enthält.

- Die gute Eignung von CORBA als Plattform zur modularen und verteilten Entwicklung von Diensten wurde unter anderem durch die getrennte Entwickung und nachträgliche Kombination zweier unterschiedlicher Dienste am FZI und IKE unterstrichen. Hierzu trägt besonders die Beschreibung von Schnittstellen mit CORBA-IDL bei.
- Die relativ problemlose Integration von Komponenten auf heterogenen UIS-Plattformen sowie das standardgemäße Zusammenspiel verschiedener CORBA-Implementierungen wurde gezeigt.

Im folgenden wird noch auf weitere wichtige durchzuführende Arbeiten hingewiesen.

5.1 Ereignisverarbeitung

Wie schon zuvor erklärt können Ereignisse über sogenante *Monitor*-Komponenten überwacht und visualisert werden /1/, /2/ (siehe auch Kapitel 3.4); damit kann z.B. der aktuelle Status der Simulation dokumentiert werden. Darüberhinaus können Ereignisse über eine geeignete Regelverarbeitung weiterverarbeitet werden und andere Aktionen im System anstoßen. Am weitesten verbreitet sind sog. *ECA-Regeln* (Event, Condition, Action), besonders bekannt aus dem Bereich aktiver Datenbanksysteme. Mit ECA-Regeln ist z.B. die Überwachung von Grenzwerten einfach und ökonomisch realisierbar.

Ein Anwendungsbereich ist z.B. die Erkennung und Überwachung komplexer Situationen. Wenn alle zu überwachenden Systemkomponenten bei eintreten bestimmter Situationen entsprechende Ereignisse auslösen, kann durch die Kombination der Ereignisse auf einen entsprechende kritische Situation im Gesamtsystem automatisch geschlossen werden. So könnten z.B. überschrittene Grenzwerte direkt vom Datenbankdienst und kritische Schadstoffausbreitungen direkt vom Simulationsdienst signalisiert werden.

Darüberhinaus bietet eine Ereignisverarbeitung die Möglichkeit, Komponenten noch weiter voneinander zu entkoppeln. So können z.B. komplexe workflow-artige Anwendungen, die aus vielen Einzelanwendungen bestehen, mittels Ereignissen dynamisch verbunden werden.

5.1.1 Wrapper als Ereignisquellen

Um die Ereignissteuerung konsequent durchzuführen, benötigen alle Komponenten die Möglichkeit, Ereignisse auszulösen. Bei neuen Systemteilen können Ereignisse an gewünschten Stellen ausgelöst werden. Bei bestehenden Systemen, wie z.B. Datenbanken, ist das etwas schwieriger. In solchen Fällen können die Ereignisse allerdings an entsprechenden Stellen eines Wrappers oder z.B. bei aktiven Datenbanken durch sog. Trigger ausgelöst werden. So kann in Wrappern beim Beginn und am Ende jeder Methode ein entsprechendes Ereignis ausgelöst werden, z.B. bei MESYST „*Simulation gestartet*“ bzw. „*beendet*“ oder „*Zeitschritt i gestartet*“ bzw. „*gestoppt*“. Ausführliche Informationen dazu finden sich in /2/.

5.2 Erweiterungen der Datenbank-Zugriffe

Die Datenbankzugriffe, die in diesem Projekt realisert wurden, sind natürlich nur einfache erste Ansätze. Sie sollen zunächst einmal die Machbarkeit demonstrieren. Datenbanksysteme sind darauf konzipiert, große Datenmengen zu speichern und zu verwalten. Herkömmliche Filesysteme sind besonderes in verteilten Umgebungen diesen unterlegen. Ein breiterer weitergehender Einsatz von Datenbanksystemen ist also empfehlenswert.

Im Fall des Simulationsdienstes ist es sinnvoll, alle von der Simulation benötigten Daten, z.B. Topographiedaten, an entsprechenden IDL-Schnittstellen anzubieten und in einer Datenbank abzulegen. Außerdem ist es denkbar, Meßwerte, die von den Meßstellen direkt in Datenbanken ablegt werden, auch anzubieten, so daß die Simulation immer die aktuellsten Werte zu Verfügung hat. In dieser Hinsicht wäre die Schnittstelle um schreibende Zugriffe zu erweitern. Hiermit wäre dann auch die Speicherung der Simulations-(Zwischen)-Ergebnisse in der Datenbank als Alternative zu betrachten.

Eine andere denkbare Ergänzung des Demonstrationssystems wäre es, die Möglichkeiten von CORBA noch weitreichender auszuschöpfen. Der Trader-Service (Details in /18/) ist z.B. bei der Auswahl und Lokalisation von geeigneten Datenquellen von zentraler Bedeutung. Durch einen Trader-Service wäre der Vorgang der Datenauswahl für den Client des Demonstrationssystems deutlich komfortabler, da dieser von den eigentlichen Datenquellen entkoppelt wäre und über ein Trader-Objekt mit diesen in Verbindung treten könnte (siehe Abbildung 5).

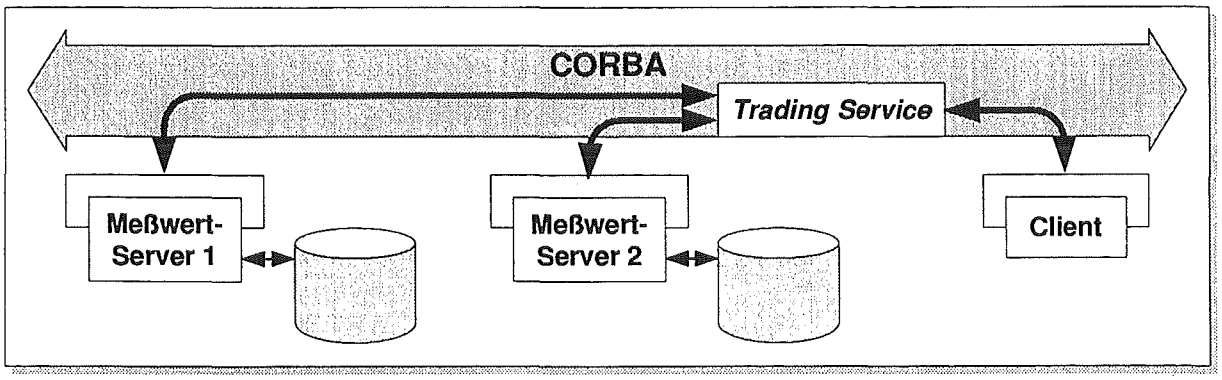


Abbildung 5: Zugriffe auf verschiedene Meßwertdatenbanken unter Kontrolle eines Traders.

5.3 Vollständige MESYST-Integration

Für das beschriebene Demonstrationssystem wurden zwei wesentliche Komponenten aus MESYST prototypisch gekapselt. Um den vollständigen Funktionsumfang von MESYST in einer CORBA-basierten Diensteumgebung im UIS zur Verfügung zu stellen und um weitere Erfahrung mit Simulationsdiensten zu sammeln, müssen auch die restlichen 17 MESYST-Module geeignet gekapselt werden. Mit den bei der Implementierung des Demonstrationssystems gemachten Erfahrungen sollte dies jedoch keine allzu schwierige Aufgabe sein.

Durch die Integration sämtlicher MESYST-Module in die vorgestellte Diensteumgebung könnte sowohl der potentielle Nutzerkreis von MESYST vergrößert, als auch die Benutzung von MESYST selber stark vereinfacht und komfortabler werden, da die zeitaufwendige Datenbeschaffung in einer Diensteumgebung deutlich vereinfacht werden kann. Außerdem steht durch das zur Kapselung der MESYST-Module verwendete Framework ein flexibles System zur Verwaltung und Verteilung der MESYST-Funktionalitäten auf verschiedene Rechner zur Verfügung. Dies ermöglicht zum einen eine bessere Ausnutzung vorhandener Rechnerressourcen, und damit eine Leistungssteigerung sowie eine höhere Verfügbarkeit für MESYST, und zum anderen eine leichtere Anpaßbarkeit von MESYST an eine sich stetig wandelnde Hardwarelandschaft.

6. Literaturverzeichnis

/1/ v. Bültzingslöwen, G.; Koschel, A.; Kramer, R.: *Active Information Delivery in a CORBA-based Distributed Information System*, in: Aberer, K.; Helal, A. (Hrsg.), Proc. First IFCIS International Conference on Cooperative Information Systems (CoopIS'96). IEEE Computer Society Press, Los Alamitos, California. Brüssel, Belgien. Juni 1996. S.218-227.

/2/ v. Bültzingslöwen,G.; Koschel, A.; Kramer, R. (1996): *Accept Heterogeneity: An Event Monitoring Service for CORBA-based Heterogeneous Information Systems*, submitted for publication.

/3/ Coldewey, Jens; Keller, Wolfgang (1996): *Objektorientierte Datenintegration - Ein Migrationsweg zur Objekttechnologie*, in: ObjektSpektrum, Heft 4, Jg. 1996, SIGS Publications, Bergisch Gladbach, Juli/August 1996.

/4/ Koschel, A.; Kramer, R.; Theobald, D.; von Bültzingslöwen, Günter; Hagg, Willhelm; Wiesel, Joachim; Müller, Manfred (1996): *Evaluierung und Einsatzbeispiele von CORBA-Implementierungen für Umweltinformationssysteme*, Hannover.

/5/ Lang, S. M.; Lockemann, P. C.(1995): *Datenbankeinsatz*, Springer Verlag.

/6/ Object Management Group (1995): *The Common Object Request Broker, Architecture and Specification Version 2.0.*

/7/ Orfali, R.; Harkey, D (1995): *Client/Server with Distributed Objects*, in: BYTE, April 1995, S.114-122.

/8/ Orfali, R.; Harkey, D.; Edwards, J. (1996): *The Essential Distributed Objects Survival Guide*, New York.

/9/ Schöckle, Martin (1995): *A Framework for the Development of Simulation Software for Complex Continuous Systems*, Interner Bericht, IKE Universität Stuttgart, IKE 4-TF-10, Dezember 1995.

/10/ Schöckle, Martin (1996): *Object Oriented and Distributed Simulation of Complex Continuous Systems*, European Simulation Multiconference, Budapest, Juni 1996.

/11/ Schweizer, Frank (1996): *Management von technischen Diensten mit CORBA*, Diplomarbeit, IKE Universität Stuttgart, IKE 4-D-206, Oktober 1996.

/12/ Schuch, Armin; Tischendorf, Manfred; Bußmann, Martina; Schmidt, Fritz (1996): *Benutzerhandbuch Kommunikationsinterpreter, Version 1.0*, Interner Bericht LfU Karlsruhe, Mai 1996.

/13/ Siegel, J. (1996): *CORBA, Fundamentals and Programming*, New York.

/14/ Sneed, Harry M. (1996): *Die Einbindung alter Host-Software in Client/Server-Architekturen*, in: ObjektSpektrum, Heft 4, Jg. 1996, SIGS Publications, Bergisch Gladbach, Juli/August 1996.

/15/ Soley, R. M. (1990): *Object Management Architecture Guide.*

/16/ Umweltministerium Baden-Württemberg (1995): *Projekt Globus, Konzeption und prototypische Realisierung einer aktiven Auskunftskomponente für globale Umwelt-Sachdaten im Umweltinformationssystem Baden-Württemberg, Phase II*, Karlsruhe.

/17/ Umweltministerium Baden-Württemberg (1996): *Projekt Globus, Konzeption und prototypische Realisierung einer aktiven Auskunftskomponente für globale Umwelt-Sachdaten im Umweltinformationssystem Baden-Württemberg, Phase III*, Karlsruhe.

/18/ Koschel, A.; Schmidt, F.; Schöckle, M; Strohm, J.; Tischendorf, M.(1996): *Bericht der AG-Dienste*, in: /17/.

/19/ Weigele, Michaele; Lurk, Alexander; Kopetzky, Roland; Schmidt, Fritz (1996): *Beschreibung des Dienstes Ausbreitungsrechnung*, 1. Meilensteinbericht, IKE Universität Stuttgart, Oktober 1996.

/20/ IONA Technologies Ltd. (1995): *Orbix 2 Reference Guide*.

/21/ Visigenic Software, Inc. (1996): Visibroker for Java, in: http://www.visigenic.com

GIS-Operatoren unter CORBA

A. Koschel,
Forschungszentrum Informatik (FZI),
Haid-und-Neu-Straße 10-14,
76131 Karlsruhe

J. Wiesel,
Institut für Photogrammetrie und Fernerkundung (IPF),
Englerstraße 7,
76128 Karlsruhe

unter Mitwirkung von:
D. Hoffmann (FZI)

1. Einleitung

1.1 Motivation

Geo-Informationssysteme (GIS) sind wichtige Bestandteile der übergreifenden Komponenten des Umweltinformationssystems (UIS) Baden-Württemberg (UIS BW) /7/. Sie dienen der Haltung von geobezogenen Daten, zur Verarbeitung und zur Visualisierung unterschiedlichster Informationen aus vielen Fachbereichen, wie z.B. Wasser, Boden, Luft, Naturschutz.

Aus der Entwicklungshistorie bedingt, existieren aktuell eine Vielzahl verschiedener Geo-Informationssysteme /12/, die sich oft grundlegend unterscheiden. Wesentliche Unterschiede bestehend u.a. im Datenmodell, in Funktionsaufrufsyntax und -umfang und auch in der Darstellung der enthaltenen Geodaten. Für die Integration in ein heterogenes verteiltes System wie das UIS BW, in dem durchaus mehrere verschiedene GIS vorhanden sind, ist dies ein Umstand, der die Eingliederung in vielerlei Hinsicht problematisch gestaltet. Die Einbindung von GIS-Funktionen in ein diensteorientiertes UIS ist auch deshalb nicht einfach, weil sich generische GIS-Operationen (z.B. eine Nachbarschaftsanalyse) auf existierende GIS-Software oft nicht abbilden lassen, da es meist keine Auftragsschnittstelle (z.B. ein Call-Interface) gibt.

1.2 Umfeld: WWW-UIS II im Rahmen des GLOBUS III Projekts

Fachlich und organisatorisch bedingt, liegen die Informationen des UIS BW häufig verteilt auf unterschiedlichen DV-Systemen, die sich wiederum aus technisch heterogenen Systemplattformen, unterschiedlichen Datenbanksystemen, Expertensystemen, Berechnungskomponenten usw. zusammensetzen können. In einer Reihe von Arbeiten (/7/, /8/, /9/, /10/, /11/) für das UIS BW werden derzeit neue Technologien eingesetzt, die eine Bereitstellung der übergreifenden Komponenten des UIS BW in einer diensteorientierten Architektur ermöglichen, wodurch eine bessere Modularisierung und damit Wiederverwendbarkeit bzw. Kombinierbarkeit von Systemteilen erreicht werden soll. Bei den eingesetzten Technologien handelt es sich zum einen um Technologien des World Wide Web (HTTP/CGI, Java) für den clientseitigen (plattformunabhängigen) Zugang zu UIS-Diensten. Zum anderen wird, zur Verdeckung der serverseitigen Heterogenität im UIS, die Common Object Request Broker Architecture (CORBA) /3/, /5/, /6/, der Object Management Group (OMG) /4/ betrachtet.

Bei CORBA handelt es sich um eine sogenannte Middleware (also eine mittlere Architekturschicht), die systemunabhängig definiert ist. In CORBA können mittels einer syntaktisch einheitlichen, standardisierten Beschreibungssprache (Interface Definition Language, IDL) die Schnittstellen zu beliebigen Software-Komponenten programmiersprachenunabhängig definiert und diese damit netzwerkübergreifend eingebunden werden. Insbesondere können hierdurch bestehende Softwaresysteme, hier also Geo-Informationssysteme, mittels sogenannter IDL-Wrapper in CORBA-Umgebungen

integriert werden. Ausführlichere Darstellungen von CORBA finden sich an anderer Stelle des GLOBUS-III Berichtes bzw. in /1/, /2/, /3/, /4/, /5/, /6/.

GRASS (Geographical Ressources Analysis Support System, /14/) ist eine GIS-Software, die vom U.S. Army Construction Engineering Laboratory (CERL) und einer Nutzergemeinde entwickelt wurde. Es enthält ca. 200 Programme zur Erfassung, Speicherung, Verarbeitung, Analyse und Präsentation geographischer Daten in Raster- und Vektorform. GRASS ist Public Domain Software und im Quellcode verfügbar. Eine kommerzialisierte Version für Windows-NT (GRASSLAND) ist mittlerweile ebenfalls verfügbar.

GRASS wird im WWW-UIS /8/ bereits als Toolbox zur serverseitigen Erzeugung von Kartenbildern und zur temporären Haltung von Geodaten aus der RIPS-Datenbank eingesetzt.

1.3 Ziel, Lösungsidee

Ziel speziell dieser Arbeit ist es nun, erste (Architektur-)Konzepte zur Integration von Geo-Informationssystemen in eine CORBA-Umgebung für das UIS BW zu skizzieren. Die zentrale Idee hierbei ist, diese Integration auf Basis universeller Geo-Operationen und universeller Geo-Datenstrukturen durchzuführen, also eine entsprechende CORBA-Schnittstelle für solche Operationen zu entwerfen. Eingesetzt werden hierzu die aktuellen Arbeiten der OGC (Open GIS Consortium http://ogis.org) /15/ zur abstrakten Beschreibung universeller Geo-Datenstrukturen (OpenGIS Project Document Number 96-015R1 http://ogis.org/public/abstract.html) und Operationen. In /13/ untersucht Albrecht existierende GIS und reduziert deren ca. 144 vorhandenen Operationen durch geeignete Kombination von Basisoperationen auf insgesamt 20 universelle Operationen. Aus diesen lassen sich alle anderen wieder ableiten. Er definiert 6 Obergruppen (Search, Location Analysis, Terrain Analysis, Distribution/Neighbourhood, Spatial Analysis, Measurements).

Neben neuen Arbeiten in internationalen Normungsgremien (CEN, ISO), die aber bisher noch nicht sehr weit fortgeschritten sind, beschäftigt sich das Open GIS Consortium Inc. (OGIS) in den USA im Auftrag der US-Regierung und in enger Kooperation mit der GIS-Industrie mit der Problematik des Austauschs von Geodaten und Interoperabilität von Software, die diese Daten verarbeitet.

Wesentliche Ziele von OGC sind in /15/ zusammengefaßt und bestehen darin, technische Lösungen zu finden um

- Geodaten einfach nutzen zu können
- heutige monolithische GIS-Software in kooperierende Komponenten zu zerlegen, die in einer verteilten Umgebung eingesetzt werden kann.

Dabei nennen die Autoren ausdrücklich CORBA und COM als mögliche zu verwendende Middleware.

Bestandteile der Arbeit sind somit eine Übersicht zum Konzept der universellen GIS-Operatoren zur GIS-Interoperabilität, die Konzeption einer GIS-Interoperabilitätsarchitektur mittels CORBA und die (sehr einfache) prototypische Umsetzung einiger Komponenten der Architektur. Hierzu werden das Geo-Informationssystems GRASS und die CORBA-Implementierung Orbix eingesetzt.

2. Universelle GIS-Operatoren als Basis für GIS-Interoperabilität

Wie bereits in der Einleitung erwähnt, gruppiert Albrecht in /13/ analytische GIS-Operatoren in 6 Bereiche:

- Search
  - Interpolation unbekannter Punkte aus bekannten
  - Objektauswahl durch Suchfenster oder Maske
  - Objektauswahl durch Attribute
  - Reklassifizierung
- Locational Analysis
  - Buffer
  - Corridor
  - Verknüpfung von Datenebenen (Clip, split, intersect union, erase)
  - Nachbarschaftsoperationen
  - Regionalisierung
  - Abflüsse zwischen Regionen
- Terrain Analysis (3D-Analyse)
  - Geländeneigung und Projektionen der Ansicht
  - Sammelbecken, Punktabflüsse
  - Drainagen/Netzwerke
  - Sichtbarkeitsanalysen
- Distribution/Neighbourhood (Beziehungen zwischen Objekten, Verteilungen)
  - Cost/Diffusion/Spread
  - Proximity (Nähe)
  - Nearest Neighbour
- Spatial Analysis
  - Multivariate Analysis
  - Pattern/Dispersion
  - Centrality/Connectedness
  - Shape
- Measurement
  - Größe, Ausrichtung, Kompaktheit, Punktförmigkeit, Streuung

Aus dieser Vielzahl von z.T. sehr speziellen Funktionen greifen wir für die prototypische Realisierung die Operationen „Spatial Search" und „Thematic Search" heraus. Sie erfüllen für die jetzige Implementierung der graphischen Dienste im WWW-UIS-II alle Anforderungen.

Ziel der räumlichen Objektauswahl ist, alle Geoobjekte innerhalb einer vorgegebenen polygonal begrenzten Fläche auszuwählen. Ziel der thematischen Auswahl ist, alle Geoobjekte, die einer Liste von gegebenen Objektattributen genügen, auszuwählen.

3. Konzeption: Interoperabilitätsarchitektur mit CORBA

Ein wesentlicher Bestandteil dieser Arbeit ist die Konzeption einer Interoperabilitätsarchitektur für GIS mittels CORBA. Primäre Aufgabe der Komponenten dieser Architektur ist die Abstraktion von einem zugrundeliegenden GIS durch Einsatz von IDL Wrappern und den universellen GIS-Operatoren bzw. GIS-Datenstrukturen, wie oben dargestellt.

Abbildung 1 zeigt eine Übersicht der geplanten Interoperabilitätsarchitektur, die als Ergebnis der Arbeiten von FZI und IPF entstand. (Im Rahmen dieser Arbeiten wurden auch eine Reihe von Architekturalternativen diskutiert, die hier aus Platzgründen nicht dargestellt werden.) Auf einem lokalen (Client) System läuft die GIS Anwendung (hier allgemein als GIS Terminal bezeichnet), welche die Informationen, die das GIS liefert, darstellt oder weiterverarbeitet. Um in einer verteilten CORBA-Umgebung auf die GIS-Daten zugreifen zu können, bzw. GIS-Operationen ausführen zu können, fungiert das GIS-Terminal als CORBA Client. Es nutzt also entsprechende CORBA-Methodenaufrufe um auf das mittels eines IDL-Wrappers gekapselte GIS (ein CORBA GIS-Server) zuzugreifen.

Für einen GIS Zugriff verbindet sich das GIS-Terminal (CORBA Client) mit einem sogenannten Moderator, der als Vermittler bei mehreren (gleichzeitigen) Zugriffen auf das GIS fungiert. In diesem Rahmen kann der Moderator eine Vielzahl von Funktionen übernehmen. Er kann beispielsweise, ähnlich einem Trader, zu eventuellen Anforderungen der Anwendung ein passendes GIS auswählen. Ferner kann der Moderator ein Multiuser-Sessionkonzept realisieren, also den Zugriff mehrerer Clients auf ein GIS erlauben. Ebenfalls kann er die Funktionalität von verschiedenartigen GIS kombinieren oder übergreifende Anfragen an mehrere GIS stellen und das Ergebnis kombiniert zurückliefern. An dieser Stelle wäre auch zur Leistungssteigerung ein Cache für Geo-Objekte denkbar. Bedingt durch die Verteilungstransparenz von CORBA, kann der Moderator sowohl auf dem lokalen System, wie auf dem System auf dem das GIS läuft, als auch auf einem beliebigen dritten System des CORBA-basierten verteilten Systems installiert sein.

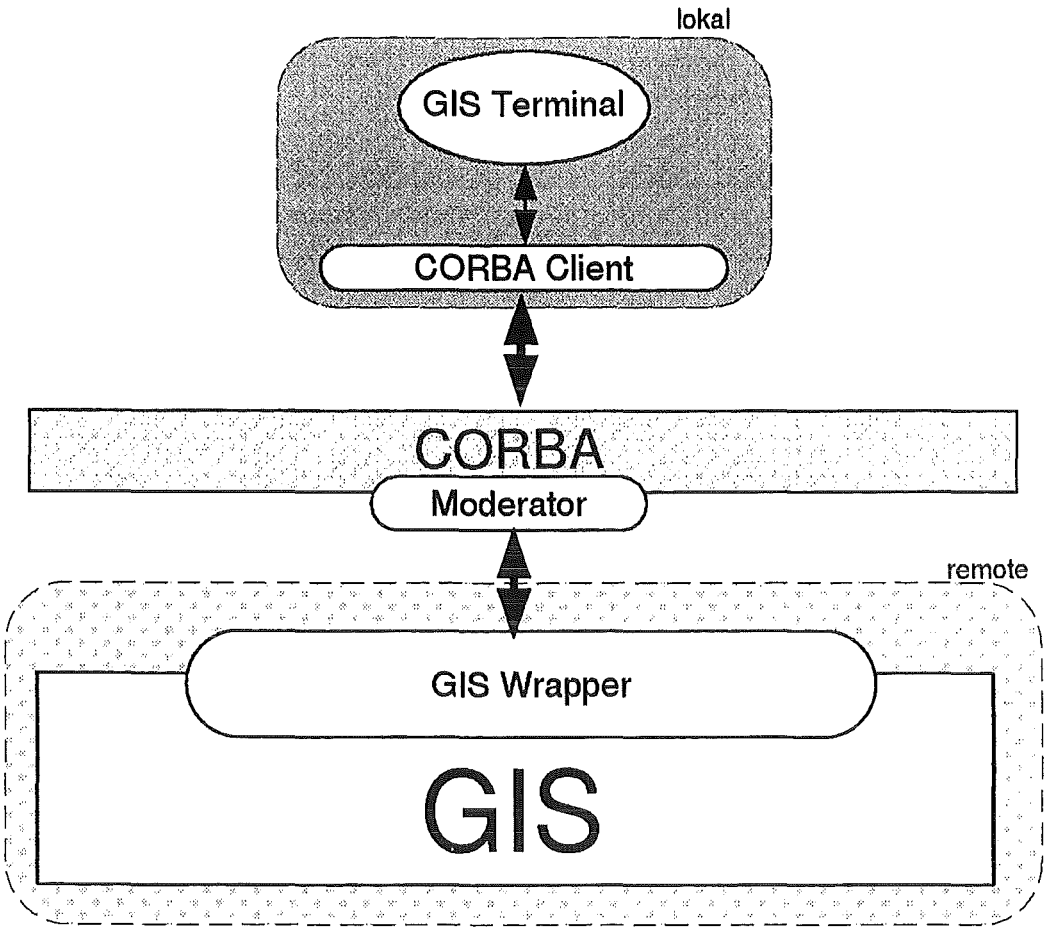


Abbildung 1: Interoperabilitätsarchitektur - Übersicht

Im nächsten Schritt einer GIS-Terminal Anfragebearbeitung verbindet sich der Moderator wiederum mit dem Wrapper (siehe Kap. 4.4) des jeweiligen GIS.

Abbildung 2 verdeutlicht die vollständige Interoperabilitätsarchitektur des Projekts, hier schon mit den konkreten Projektwerkzeugen, also GRASS als GIS und Orbix als CORBA-Implementierung.

Konzeptionell wird der (bzw. jeweils einer je GIS) GIS-Wrapper in 4 logische Einheiten aufgeteilt (siehe Abbildung 2). Jede dieser Einheiten kapselt („wrappt") einen charakteristischen Aufgabenbereich von GIS. Im einzelnen sind dies die Aufgabenbereiche GIS-Metadatenzugriff, zum Erfragen von Metainformationen über das zugrunde liegende GIS und dessen verwaltete Geo-Objekte, GIS-Anfragedienste, welche Informationen der GIS-Datenbasis, die u.U. zuvor aufbereitet werden liefern, analytische Operationen, die eine interaktive Analyse auf Teile der GIS-Datenbasis durchführen und Präsentationsdienste, die Ergebnisse der analytischen Operationen und der Anfragedienste in Form von beschrifteten Tabellen oder Grafiken aufbereitet. Durch dieses Vorgehen wird die Architektur offener

gegenüber künftigen Entwicklungen und Veränderungen, sie bleibt allgemeingültiger und ist nicht zu sehr auf ein bestimmtes Problem oder ein bestimmtes Anwenderprogramm ausgerichtet. Es besteht eine klare Schnittstelle zur Querverbindung zwischen einzelnen Einheiten.

Zu beachten ist in Abbildung 2, daß speziell in GRASS GIS-Operationen sowohl über Datenzugriffsfunktionen, als auch direkt auf die Geo-Daten zugreifen können. Als Ausblick ist in dieser Abbildung ferner eine Ereigniserkennung skizziert, mit deren Hilfe bspw. Änderungen in GIS-Datenbeständen oder Aufrufe bestimmter GIS-Operatoren erkannt werden können.

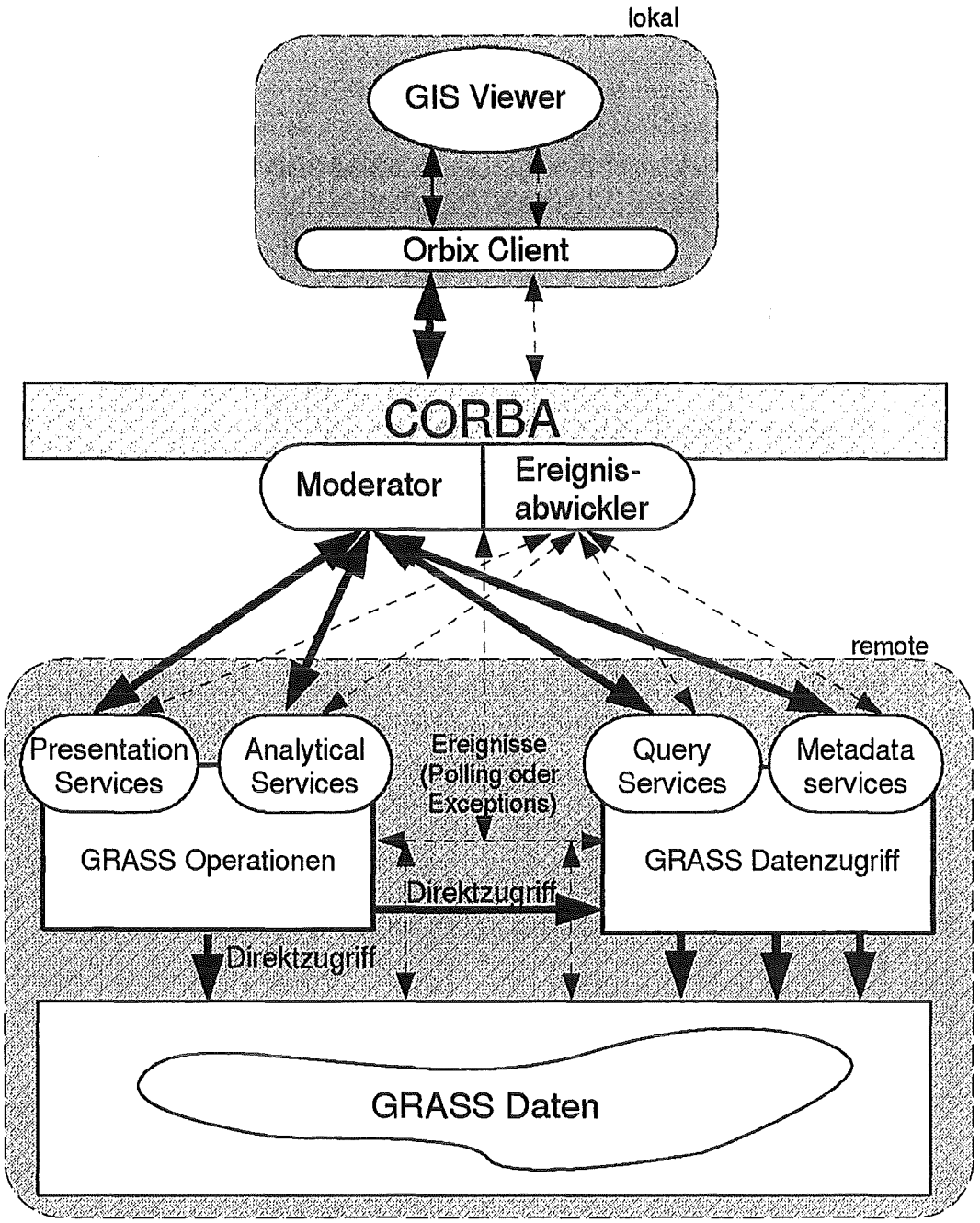


Abbildung 2: Interoperabilitätsarchitektur

4. GIS unter CORBA: Ein einfacher Prototyp

4.1 Verwendete Technologie

Zur Implementierung des CORBA-Teils in diesem Projekt wurde die CORBA-Implementierung Orbix, Version 2.0.1, von IONA (http://www-usa.iona.com/Orbix/) benutzt. Orbix zeichnet sich durch seine einfache Handhabung, seine Verbreitung und seiner

Verfügbarkeit auf sehr vielen Plattformen aus. Die im Rahmen dieser Arbeit eingesetzte Orbix-Version 2.0.1 ist zum CORBA 2.0 Standard /3/ konform.

GRASS (Geographic Resources Analysis Support System http://www.cecer.army.mil/grass/GRASS.main.html) ist ein weit verbreitetes, frei erhältliches raster- und vektorbasiertes GIS, das vom US Army Corps of Engineers konzipiert wurde. Es besteht aus vielen kleinen Programmen zum Bearbeiten und Analysieren von Geo-Daten, welche zumeist in C geschrieben sind. Ein wichtiges Merkmal dieses GIS ist, neben umfassender Funktionalität, die freie Verfügbarkeit seiner Quellen (ohne Lizenzkosten). Dies ermöglicht hier insbesondere die recht gute Integrierbarkeit von GRASS in eine CORBA-Umgebung bzw. generell eine gute Erweiterbarkeit seiner Funktionalität.

4.2 GIS-Operatoren für den Prototyp

Auf GRASS Seite wurden, neben einigen benötigten Metadatenoperationen, zwei Operationen für die Anfragedienste implementiert. Zum einen die Operation GetObjByRegion, die für eine bestimmte Karte einen polygonalen Ausschnitt zurückliefert, zum anderen die Operation GetObjByAttr, die für eine Karte ausgewählte Geo-Objekte des Kartenbereichs zurückliefert.

Auf CORBA Seite wurden die Metadatenoperationen connect zum Binden an ein GIS und die Operation get_geoobject_descriptions, die eine Beschreibung der durch das GIS verwalteten Geo-Objekttypen liefert, realisiert. Die Anfragedienstoperationen, welche auf GRASS Seite verwirklicht wurden, werden durch CORBA IDL-Schnittstellen für query_by_region und query_by_attribute Methoden gekapselt.

4.3 Basis für GIS-Operatoren: Grass-Funktionalität

Da GRASS eine interne Layerstruktur für die Geodatenverwaltung benutzt und Raster- und Vektordaten in verschiedenartigen Verwaltungseinheiten intern abgelegt sind, müssen die geoobjekt-orientierten Operationen auf entsprechende GRASS-Primitive abgebildet werden. In der Abb 3 ist die prozedurale Struktur der auf der GRASS-Programmierbibliothek aufgesetzten Funktionen zur Implementierung der beiden Operatoren schematisch dargestellt. Teile dieser Funktionsliste wurden implementiert:

- GetMapsetName (liefert die Liste der verfügbaren Kartensätze)
- GetMapTypes (liefert eine Tabelle mit der Anzahl der Karten der Typen Raster, Vektor, Punkt zurück)
- GetMapNames (liefert die Namen der Karten eines bestimmten Types)
- GetAllMapNames (liefert alle Kartennamen aus einem Kartensatz zurück)

Daneben wurden

- GetRasterObjectByRegion
- GetMapType (liefert Datenart zurück)
- GRASS2GIF (konvertiert GRASS Raster nach GIF)

implementiert.

Abbildung 3: Prozedurale Struktur der GRASS-Implementierung

4.4 CORBA-Wrapper für Grass

Die freie Verfügbarkeit der GRASS Quelltexte ermöglicht es die C APIs der GRASS Programme direkt im CORBA-Wrapper zu kapseln. Dabei werden die Prototypen der verwendeten GRASS Funktionen als externe C Funktionen deklariert, und die Objektdateien der Funktionen zum Wrappercode hinzugelinkt. Dies ist die einfachste Möglichkeit des Kapselns, da keine zusätzlichen Mechanismen zum Austausch von Informationen zwischen Wrapper und gekapseltem System notwendig sind. Diese Art des Kapselns ist jedoch nur möglich, wenn ein klar definiertes API für das gekapselte System vorhanden ist oder seine Quelltexte (hier Header- und Objektfiles) zugänglich sind.

Um an der vom GRASS Wrapper angebotenen Schnittstelle eine allgemeine, abstrakte Sicht der Geo-Objekte und Geo-Operatoren zu erhalten, wurde ein (im Prototyp einfacher) allgemeiner Geo-Objekttyp als CORBA-IDL Objekttyp definiert. Dies ermöglicht, den Geo-Objekttyp selbst als verteiltes Objekt zu behandeln und die Verwaltung der Geo-Objekttypen, auf denen der Wrapper operiert, von den speziellen Geo-Objekten des GIS zu trennen. Wie oben skizziert, wurden die speziellen GRASS Operationen aus Kapitel 4.2 auf allgemeinere, abstrakte GIS-Operatoren abgebildet.

4.5 Demo-Beispiel

Nachfolgend beispielhaft ein kurzer Ausschnitt der Moderator-Schnittstelle für die Anfragedienste in CORBA IDL:

```
interface query_services {
    exception query_services_error { string reason;};

    void query_by_region(in GeoObjTypeSequence ingeoobjtypeSeq,
                         in PointSequence pointSeq,
                         out GeoObjTypeSequence outgeoobjtypeSeq);

    void query_by_attribute(in GeoObjTypeSequence ingeoobjtypeSeq,
                            out GeoObjTypeSequence outgeoobjtypeSeq);
};
```

Die Funktionen dieser Schnittstelle werden von der Anwendung über das Grass-Terminal aufgerufen. In den Implementierungen dieser Moderator-Schnittstellenfunktionen werden über eine vergleichbare Grass-Schnittstelle (Wrapper)Funktionen aufgerufen, die die eigentlichen Grass C-Funktionsaufrufe (GetObjByRegion, GetObjByAttr) durchführen.

5. Zusammenfassung und Ausblick

Anhand von Beispielen aus der GIS-Industrie und Normungsbestrebungen (OGIS, CEN, ISO) wurde gezeigt, daß verteilte, auf objektorientierter Middleware aufgebaute GIS-Software sich in der Entwicklung befindet. Die Industrie beginnt damit, ihre monolithischen GIS-Pakete in Komponenten zu zerlegen und zukünftig Server für Geodaten und Geooperationen bereitzustellen. Auf der Basis „Universeller GIS-Operatoren" kann Zug um Zug ein „Virtuelles GIS" aufgebaut werden, dessen generische Operationen (z.B. Objektauswahl) auf existierende GIS-Software abgebildet werden.

An einem einfachen Beispiel wurde eine Implementierung dieses Basisoperators mittels ORBIX und GRASS vorgenommen, um die generelle Tragfähigkeit des Konzepts zu verifizieren.

Die Vorteile der vorgeschlagenen Lösung sind:

- Räumliche Unabhängigkeit von Klient und Server
- Kapselung existierender proprietärer Software gegenüber der Anwendungsschale
- Einsatzmöglichkeit preiswerter Universalarbeitsplätze auf Java-Basis ohne teure Softwarelizenzen
- Erschließen verschiedener Datenquellen unter einer Anwendungsschale

In den nächsten Schritten sollte die hier vorgestellte Konzeption sukzessive verfeinert und implementiert werden, um eine Einbettung der GIS-Software in eine CORBA-basierte Diensteumgebung für das UIS Baden Württemberg zu erreichen.

6. Literatur

/1/ Siegel, Jon (1996): CORBA, Fundamentals and Programming, New York.

/2/ Mowbray, Thomas J.; Zahavi, Ron (1995): The Essential CORBA, Systems Integration Using Distributed Objects, New York.

/3/ Object Management Group (1995): The Common Object Request Broker, Architecture and Specification Version 2.0.

/4/ Soley, R. M. (1990): Object Management Architecture Guide.

/5/ Object Management Group (1995): Common Facilities Architecture.

/6/ Object Management Group (1995): CORBAservices, Common Object Services Specification.

/7/ Umweltministerium Baden-Württemberg (1995): Projekt Globus, Konzeption und prototypische Realisierung einer aktiven Auskunftskomponente für globale Umwelt-Sachdaten im Umweltinformationssystems Baden-Württemberg Phase II, Karlsruhe.

/8/ Koschel, A.; Kramer, R.;Nikolai, R. (1995): Architektur des WWW- und CORBA-basierten UIS, in: /7/, S.47-92.

/9/ Koschel, A.; Kramer, R.; Theobald, D. (1995): CORBA-Evaluierung für das UIS Baden-Württemberg, in: /7/, S.93-192.

/10/ Umweltministerium Baden-Württemberg (1996): Projekt Globus, Konzeption und prototypische Realisierung einer aktiven Auskunftskomponente für globale Umwelt-Sachdaten im Umweltinformationssystems Baden-Württemberg Phase III, Karlsruhe.

/11/ Koschel, A.; Kramer, R. ; Theobald, D.; von Bültzingsloewen, Günter; Hagg, Willhelm; Wiesel, Joachim; Müller, Manfred (1996): Evaluierung und Einsatzbeispiele von CORBA-Implementierungen für Umweltinformationssysteme, Hannover.

/12/ Bartelme, N. (1995): Geoinformatik, Modelle Strukturen Funktionen, Berlin.

/13/ Albrecht, J. (1996): Universal Analytical GIS Operations. Dissertation am ISPA der Universität Osnabrück.

/14/ GTZ (1993): GRASS Handbook Version 1.0., Eschborn.

/15/ OGC Inc. (1996): The OpenGIS Guide, OGIS TC Doc. 96-001, Wayland, MA.

Einsatz von Java im UIS Baden-Württemberg

Ergebnisse der AG Java

R. Nikolai, C. Rolker,
Forschungszentrum Informatik (FZI),
Haid-und-Neu-Straße 10-14,
76131 Karlsruhe

C. Hofmann,
Institut für Photogrammetrie und Fernerkundung (IPF),
Englerstraße 7,
76128 Karlsruhe

R. Kopetzky,
Universität Stuttgart,
Institut für Kernenergetik und Energiesysteme (IKE),
Abteilung Wissensverarbeitung und Numerik, Pfaffenwaldring 31,
70550 Stuttgart

M. Gaul,
Forschungsinstitut für anwendungsorientierte Wissensverarbeitung (FAW),
Helmholtzstr. 16,
89081 Ulm/Donau

1. Einleitung

Im vorliegenden Bericht sind die Arbeiten und Ergebnisse der AG Java zusammengestellt. Aufgabe war die Evaluierung des Einsatzes von Java und JavaScript für das Umweltinformationsystem (UIS) Baden-Württemberg. Ausgangspunkt dieser Evaluierung bildeten die in der zweiten Phase des Projektes GLOBUS erzielten Ergebnisse bei der Nutzung der Techniken und Werkzeuge für das World-Wide Web /4/.

An der Evaluierung aktiv beteiligt waren neben dem Forschungszentrum Informatik (FZI), bei dem die Federführung lag, das Forschungsinstitut für anwendungsorientierte Wissensverarbeitung an der Universität Ulm (FAW), das Institut für Kernenergetik und Energiesysteme (IKE), Universität Stuttgart, sowie das Institut für Photogrammetrie und Fernerkundung (IPF), Universität Karlsruhe.

Die Ergebnisse der ersten Phase der AG Java sind in /5/ zusammengefaßt. Dieser Bericht enthält ausführliche Übersichten und Bewertungen über Java und JavaScript und war Grundlage, um über den Einsatz von Java, JavaScript sowie der damit zusammenhängenden Werkzeuge im UIS Baden-Württemberg fundiert entscheiden zu können.

Der Lenkungsausschuß GLOBUS III stimmte auf seiner dritten Sitzung den Empfehlungen der AG Java zu.

Da Java einer starken Weiterentwicklung unterliegt, sollten nur wenige Komponenten für den Einsatz von Java ausgewählt werden. Diese Komponenten sollten besonders geeignet sein für eine Erprobung und Demonstration der Fähigkeiten von Java. Die AG Java legte daraufhin geeignete Komponenten für die Umstellung fest und entschied, ein größeres Beispiel mit mehreren beteiligten Instituten noch während der Projektlaufzeit zu realisieren. Dieses Beispiel sollte zum einen die Vorteile der neuen Technologie gegenüber dem klassischen HTML und CGI-Ansatz verdeutlichen, zum anderen aber auch Grundlage für Schätzungen des erforderlichen Aufwands sein.

Im folgenden werden die Grundlagen und Bewertungen, die in /5/ bereits ausführlich dargelegt wurden, kurz zusammengefaßt. Desweiteren werden Konzepte für das Zusammenspiel mehrerer Java-Komponenten, die zusammen eine komplexere Anwendung realisieren, aufgezeigt. Der Beschreibung der verschiedenen Möglichkeiten der Inter-Applet-Kommunikation, schließt sich eine Erläuterung zweier für das UIS entwickelten Java-Anwendungen, des GIS-Viewer sowie des Poll-Applets an. Schließlich wird das gewählte Integrationsbeispiel, der Zugriff und die Visualisierung ausgewählter Luftmeßdaten, vorgestellt.

2. Java und JavaScript

Im folgenden werden die Sprachen Java und JavaScript vorgestellt, erst einzeln bewertet und dann anhand einer Liste von Kriterien gegenübergestellt. Diese Evaluierung wurde in der Phase 1 der Java AG durchgeführt. Die Ergebnisse werden ausführlich in /4/ und /5/ vorgestellt.

2.1 Java (FZI)

Java ist eine von Sun Microsystems entwickelte objektorientierte Programmiersprache, die speziell dafür ausgelegt ist, sichere und architekturunabhängige Programme, also Programme, die ohne Neuübersetzung und Portierung auf unterschiedlichen Hardware- und Betriebssystemplattformen ablauffähig sind, für heterogene Netzwerke zu schreiben. Java hat viel aus den Fehlern anderer Programmiersprachen gelernt und vereinfacht das Programmieren erheblich. Hierzu trägt auch die umfangreiche mitgelieferte Klassenbibliothek bei. Java bietet mit der C++-ähnlichen Syntax sowie bekannten Sprachkonzepte einen einfachen Umstieg für viele Programmierer an.

Neben dem Einsatz als einfachen Ersatz für andere Programmiersprachen bietet Java die Möglichkeit, Programme in HTML-Seiten einzubetten und auf Client-Seite sicher ablaufen zulassen. Diese Programme werden Java-Applets genannt und werden durch einen Java-kompatiblen WWW-Browser interpretiert. Die Interaktivität von WWW-Anwendungen kann durch den Einsatz von Java-Programmen beträchtlich gesteigert werden. Ohne Java-Browser ist von HTML-Seiten mit Java der HTML-Rahmen auf jeden Fall sichtbar, das Java-Applet jedoch nicht.

Die Java-Unterstützung kann mittlerweile (Oktober '96) als sehr gut bewertet werden. WWW-Browser, die Java unterstützen sind z.B. Netscape Navigator seit der Version 2.0, Microsoft Internet Explorer seit der Version 3.0 und HotJava. Selbst auf älteren Windows Plattformen (Windows 3.1 und Windows 3.11) bietet Microsoft mit dem Internet Explorer bereits einen Java-fähigen Browser an. Dies zeigt die Bedeutung und breite Akzeptanz, die dieser neuen Programmiersprache entgegengebracht wird.

Das Laden von Programmcode über ein offenes Netzwerk und dessen Ausführung auf dem eigenen Computer wirft sofort Sicherheitsfragen auf. Absolute Sicherheit kann nicht gewährt werden, doch ist Java mit dem 4-stufigen Sicherheitskonzept sicherer als alle Alternativen. Es bestehen derzeit keine generellen Sicherheitsbedenken.

2.2 JavaScript (IPF)

JavaScript ist die Skriptsprache der Firma Netscape, um einfache Interaktionen Client-seitig ausführen zu können. Es bietet keinerlei graphische Funktionen, kann jedoch für textorientierte Dialoge (z.B. Plausibilitätsprüfungen, Berechnungen) gut verwendet werden. JavaScript ist in der Syntax ähnlich zu Java. Es ist ein Produkt der Firma Netscape, das in anderen Browsern nicht verfügbar ist und auch wahrscheinlich nicht sein wird. Untersuchungen haben gezeigt, daß es für einen Einsatz im Internet (im Gegensatz zu einem Intranet) aus Sicherheitsgründen nicht geeignet ist. Als Plattformen sind alle Systeme mit Netscape Navigator 2.0 und höher geeignet.

Es wird nicht empfohlen, JavaScript für externe UIS-Anwendungen einzusetzen.

2.3 Beurteilung technischer Alternativen im WWW (IPF)

Bei der Weiterentwicklung des WWW-UIS, aber auch bei sonstigen Vorhaben im Bereich des WWW existieren eine Reihe von Aspekten, die zu beachten sind, um die effektive und erfolgreiche Umsetzung eines Projekts zu garantieren.

Die nachfolgende Übersicht bewertet diverse Aspekte für unterschiedliche Techniken im WWW. Unterschieden wird dabei zwischen HTML, JavaScript und Java. Desweiteren erfolgt eine Bewertung für den Netscape Navigator in den Punkten, in denen er sich durch seine erweiterten HTML-Merkmale von Standard-HTML abhebt. JavaScript ist auch eine Erweiterung des Netscape Navigator, wird aber auf Grund gänzlich anderer Möglichkeiten und Absichten gegenüber HTML gesondert betrachtet. JavaScript kann ohne HTML nicht eingesetzt werden und erbt somit die Funktionalität von HTML.

	HTML	Netscape HTML	JavaScript	Java	Anmerkung
Browserverfügbarkeit	**4**	**2**	**2**	**2**	JavaScript beruht auf Netscape, Java ist offengelegt und die Unterstützung von allen namhaften Firmen angekündigt
standardisiert	**4**	**0**	**0**	**2**	Java ist spezifiziert und offen, ein offizielles Gremium fehlt noch
Clientanforderungen	**4**	**2**	**2**	**3**	JavaScript läuft in jeder Netscape Version
Serverbelastung	**0**		**2**	**4**	unter dem Gesichtspunkt umfangreicher Benutzerinteraktionen
Datensicherheit	**4**		**1**	**4**	für Daten beim Client und beim Server
Betriebssicherheit	**4**		**1**	**3**	kein Programm ist fehlerfrei und damit perfekt
DB-Zugriff: lesend	**2**		**2**	**4**	schreibende DB-Zugriffe sind in HTML denkbar, aber völlig unpraktisch; Java bietet mehr als C++
DB-Zugriff: schreibend	**0**		**0**	**4**	
Sessionverwaltung	**1**		**1**	**4**	
Programmieraufwand:					HTML ist für textuelle Darstellung gedacht, Java für Interaktivität
passive Darstellung	**3**		**1**	**0**	
Interaktivität	**0**		**1**	**4**	

Geschwindigkeit	0		1	3	bei Interaktivität, HTML ist auf den Server angewiesen, Java ist Client-basiert bedarf aber eines größeren Ladevorgangs
Graphikdarstellung	1		1	4	Bilder können von HTML und JAVA dargestellt werden, Java erlaubt aber auch deren Manipulation
Interaktivitätspotential	1		2	4	bedingt durch Zielsetzung
Ergonomie:					bedingt durch Zielsetzung
Programmstruktur	0	1	1	4	
Hypertext	3	4	3	0	
Lernaufwand für Grundfunktionalität	4	3	1	0	die Grundfunktionalität ist bei Java wesentlich höher als bei HTML

Die Bewertung umfaßt fünf Stufen von eher negativ (Wert: 0) bis eher positiv (Wert: 4). Dabei ist die Bewertung relativ zu sehen, d.h. im Vergleich HTML, JavaScript und Java. So muß z.B. ein Wert von 0 nicht heißen, daß dieser Aspekt nicht realisierbar ist, sondern daß der Aufwand gegenüber einem Wert von 4 extrem hoch ist.

HTML, JavaScript und Java verfolgen unterschiedliche Ziele im WWW, so daß eine Bewertung auch hinsichtlich der Aufgabenstellung zu sehen ist. So ist z.B. für eine rein textuelle Präsentation von Informationen Java nicht erforderlich und zu aufwendig, dagegen erlaubt HTML keinerlei Interaktivität in Form von lokalen PC-Anwendungen. Den Bewertungen liegen die zu erwartenden Funktionalitäten von GLOBUS III zu Grunde.

Bzgl. des Zusammenspiels der Komponenten läßt sich sagen, daß JavaScript auf HTML angewiesen ist. Eine Schnittstelle zwischen JavaScript und Java ist angekündigt aber nicht absehbar. Java und HTML lassen sich zwar kombiniert darstellen, erlauben aber keinerlei Datenaustausch.

3. Einsatz von Java im UIS

In der Phase 1 der Java AG wurden Einsatzgebiete für Java im UIS untersucht. Es wurde jeweils erarbeitet, ob und wie gut sich Java für das jeweilige Einsatzgebiet eignet. Das Ergebnis der Evaluierung ist eine Empfehlung zur schrittweisen Umstellung einzelner Komponenten des GLOBUS-Systemkerns. Die Möglichkeiten zum Zusammenführen von separat evaluierten Teillösungen wurden im Rahmen der Evaluierung der Einsatzgebiete von Java mit Teilbeiträgen von IPF, FAW und FZI bereits erfolgreich erprobt. Im folgenden werden die untersuchten Einsatzgebiete genauer vorgestellt.

3.1 Datenbankankopplung an Java (FZI)

Zur Ankopplung von Datenbanken an die Programmiersprache Java werden aufgrund der Bedeutung dieser Problematik derzeit unterschiedliche Ansätze verfolgt. Die interessanteste der aktuellen Entwicklungen ist JDBC, ein Application Programming Interface (API) für Datenbankzugriffe von Sun, das in Java den Zugriff auf relationale Datenbanken spezifiziert. JDBC wird von zahlreichen Unternehmen, darunter die Datenbankhersteller Oracle, Informix Software, IBM (DB2), Object Design und Sybase unterstützt. Ein Treiber für Oracle wurde vom FZI entwickelt und erprobt. Vierschiedene Hersteller (z.B. JavaSoft, Intersolv) haben eine JDBC/ODBC-Bridge angekündigt, die es ermöglicht, die Vielzahl bereits vorhandener ODBC-Treiber für die unterschiedlichsten Plattformen (Betriebssysteme/Datenbanken) zu nutzen.

Als Zusammenfassung gilt, daß für einfache Datenbankanfragen wie bisher HTML und CGI-Skripte verwendet werden können, für komplexere Anwendungen, die eine höhere Interaktion und z.B. ACID-Transaktionen benötigen, Java und JDBC eingesetzt werden sollten.

3.2 Java für die graphischen Dienste (IPF)

Die Möglichkeit, Java-Bytecode auf allen gängigen Rechnerplattformen direkt ausführen zu können, ist einer der großen Vorteile den die Java-Technologie mit sich bringt. Ein weiterer, nicht zu unterschätzender Vorteil von Java ist die Definition eines graphischen Fenstersystems innerhalb der standardisierten Java-Klassenbibliothek. Erst dieses Fenstersystem, das Abstract Window Toolkit (AWT), ermöglicht es Anwendungen mit graphischen Benutzeroberflächen und graphischer Ausgabe unabhängig von einem tatsächlich vorhandenen Fenstersystem (X11-Motif, Windows, etc.) zu implementieren und auszuführen. Das Toolkit bildet hierzu alle graphischen Ausgabekomponenten über sogenannte Peer-Objekte auf Komponenten des realen Fenstersystems ab. Ebenso werden alle Eingabeoperationen (Benutzerinteraktionen) über diese Schnittstelle dem Java-Programmierer in einer vom realen System abstrahierten Form zu Verfügung gestellt. Dieser Vorgang ist für den Programmierer transparent.

Aus der Sicht der graphischen Dienste zeigt sich die ganze Stärke von Java als mächtige, portable und mit zahlreichen nützlichen Klassenbibliotheken versehene objektorientierte Sprache. Durch die Fähigkeit, das Fenstersystem und somit auch die Graphikfähigkeiten des Client-Rechners innerhalb einer WWW-Anwendung zu nutzen, wird der WWW-Server entscheidend entlastet sowie hohe Interaktivität gewonnen. Um die genannten Möglichkeiten zu demonstrieren wurde am IPF ein Prototyp einer javabasierenden Darstellungskomponente für die Karten- und Diagrammdarstellung entwickelt. Diese Komponente besaß den Arbeitsnamen GIS-Viewer und wurde mittlerweile in ein Gesamtkonzept, das GIS-Terminal (GISterm), integriert. Für weitergehende Informationen wird an dieser Stelle auf den Bericht „Architektur eines GIS-Terminal zur Visualisierung von Geodaten“ verwiesen.

Wegen der wesentlich verbesserten Darstellungsmöglichkeiten und der rein clientseitigen Interaktionsfähigkeiten wird eine Umstellung der serverseitigen Graphikdienste des UIS auf eine Java-Lösung dringend empfohlen.

3.3 Client-seitige Anbindung von PC-Tools (FAW)

Untersucht werden sollte, ob es möglich ist, PC-Tools über Java-Applets anzukoppeln. Es stellte sich heraus, daß aufgrund der Sicherheitsvorkehrungen für über das Netz geladene Applets derzeit keine direkte Anbindung von PC-Tools möglich ist.

3.4 Kopplung Java-CORBA (FZI)

Die Kombination von CORBA, dessen Stärken bei der server-seitigen Integration liegen, und Java, dessen Stärke auf Client-Seite zu finden sind, bietet eine vielversprechende, zukunftsträchtige Basis zur Entwicklung von Client-/Server-Architekturen. Besonderes Feature gegenüber klassischen Client-/Server-Applikationen sind die client- und server-seitige Hardware- und Betriebsystemunabhängigkeit, die zum einen durch die Portabilität der Sprache Java, zum anderen durch Kapselung bereits vorhandener Systeme mittels CORBA erzielt wurde.

Es existieren derzeit (Dezember '96) mehrere Produkte namhafter Hersteller, wie Visigenic, Iona und Sun. Diese Produkte stellen in Java geschriebene "schlanke" CORBA-ORBs (Orblets) dar, die mit jedem beliebigen Java-fähigen WWW-Browser als Client genutzt werden können, um mit einem ORB auf Server-seite zu kommunizieren. Mit zwei Produkten, VisiBroker for Java (Visigenic) und OrbixWeb (Iona), konnten am FZI bereits praktische Erfahrungen gewonnen werden. Zur Implementierung des Ereignis-Monitors in der Diensteintegrationsdemonstration von FZI und IKE wurde VisiBroker for Java erfolgreich eingesetzt (s.a. den entsprechenden Beitrag im GLOBUS-III Abschlußbericht). Aus Kostengesichtspunkten bedeutsam ist, daß bspw. für Orbix-Web keine Client-Laufzeitlizenzgebühren anfallen.

Ein CORBA-orientiertes Design server-seitiger Java-Klassen sowie eine weitergehende Erprobung anhand von UIS-Szenarien wird empfohlen.

3.5 Asynchrone Ergebnisrückmeldung mittels Java (IKE)

Das am NCSA entwickelte Common Client Interface (CCI), eine Schnittstelle des Browsers Mosaic, über die asynchrone Ergebnisrückmeldungen realisiert wurden, konnte sich nicht als Standard etablieren. Deswegen wurde die Frage untersucht, ob Java Möglichkeiten bietet, die bisherigen CCI-Funktionalitäten zu ersetzen.

Die Sicherheitsmechanismen des Java-fähigen Browsers Netscape verhindern den direkten Verbindungsaufbau eines beliebigen Servers zu einem Java-Applet. Der Verbindungsaufbau des Applets zu dem Server, von dem es geladen wurde, ist hingegen zulässig. Daraus ergeben sich zwei Lösungsansätze:

- Einsatz eines Poll-Applets, das in periodischen Intervallen eine Anfrage nach neuen Ergebnissen an seinen Ursprungs-Server richtet.
- Einsatz zusätzlicher Technologien wie z.B. CORBA (siehe "Integration und Kombination von Simulationsdiensten und Datenbankzugriffsdiensten mit CORBA", Abschnitt 3.4).

Am IKE wurde ein Poll-Applet entwickelt, das die asynchrone Ergebnisrückmeldung ermöglicht (siehe Abschnitt 5).

3.6 Statistische Aufbereitung (FAW)

Untersucht werden sollte die Möglichkeit, bisher server-seitige statistische Auswertungen in Java-Applets zu verlagern. Es zeigte sich, daß es problemlos möglich ist, die Berechnung von statistischen Werten auf dem Client vorzunehmen. Ob diese Verlagerung sinnvoll ist, ist im Einzelfall (abhängig von zu übertragender Datenmenge bzw. gewünschter Oberflächenfunktionalität) zu überprüfen.

4. Erarbeitete Konzepte in der Phase 2

4.1 Integration von Java-Teillösungen

Für die Entwicklung von institutsübergreifenden WWW-Anwendungen mit Java besteht die Frage nach der Integration von Teillösungen, welche an verschiedenen Instituten entwickelt werden. Es gibt drei verschiedene Realisierungsmöglichkeiten:

1. ein großes Applet ("BigApplet-Lösung"): Bei dieser Variante werden als Teillösungen verschiedene Klassen realisiert, die dann zu einem großen Applet vereinigt werden.
2. verschiedene Applets mit on-line Inter-Applet-Kommunikation: Bei dieser Variante werden Applets als Teillösungen realisiert. Die Kommunikation der Applets läuft beispielsweise über Sockets oder Kommunikationsmechanismen für verteilte Java-Objekte wie RMI (Remote Method Invocation) oder über CORBA (siehe Abschnitt 3.4).
3. verschiedene Applets mit off-line Inter-Applet-Kommunikation: Bei dieser Variante werden Applets als Teillösungen realisiert. Die Kommunikation der Applets läuft beispielsweise über Dateien.

Unabhängig von der Wahl der Realisierungsmöglichkeit müssen Schnittstellen definiert werden. Desweiteren sind auch Seiteneffekte möglich.

Für die Integration von Java-Applets gibt es bisher wenig Erfahrungen, da Java eine neue Technologie ist. Die Bewertung der oben vorgestellten Ansätze ist deshalb sehr schwierig. Aus den praktischen Erfahrungen mit anderen Sprachen hat sich für die Integration von Java-Teillösungen folgende Bewertung der drei Möglichkeiten ergeben:

Für kleinere Anwendungen ist die gegenseitige Nutzung von Klassenbibliotheken (1. Möglichkeit) sehr effektiv. Für große Anwendungen kann sich eine stärkere Trennung der Codestücke (2./3. Möglichkeit) als sinnvoll herausstellen.

Die Bewertung erfolgt aus folgenden Überlegungen:

- Realisierungsaufwand der einzelnen Teillösungen: Der Realisierungsaufwand für die 1. Möglichkeit ist geringer als für die 2. oder 3. Variante, da bei diesen die on-line- oder off-line-Kommunikation über den Server noch realisiert werden muß.
- Integrationsaufwand: Der Aufwand für das Zusammenfügen der fertiggestellten Teillösungen ist für die 1. Möglichkeit höher als für die 2. oder 3. Möglichkeit.

- Übertragung über das Netz: Bei der BigApplet-Lösung wird das gesamte Applet einmal übertragen. Danach findet für die Kommunikation zwischen den Komponenten keine Client-Server-Kommunikation mehr statt. Dies steht im Gegensatz zu den Möglichkeiten 2. und 3., bei denen jede Kommunikation zwischen den Applets über den Server laufen muß.
- Wartbarkeit: Die BigApplet-Lösung ist schlechter wartbar, im Gegensatz dazu sind die Möglichkeiten 2. und 3. aufgrund ihres modularen Aufbaus besser wartbar.

Im durchgängigen Beispiel (siehe Abschnit 5.2) wurde die 1. Möglichkeit realisiert. Für zukünftige (große) Java-Entwicklungen ist aber die 2. oder 3. Möglichkeit vorzuziehen.

Aufgrund der zusätzlichen Möglichkeit für asynchrone Ergebnisrückmeldungen der 3. Möglichkeit im Gegensatz zur 2. Möglichkeit fiel vorerst die Entscheidung zugunsten der 3. Alternative, die im folgenden als Datentopf-Konzept bezeichnet wird. Das Konzept und der Datenfluß sind in Abbildung 1 anhand einer Datenbank- und Graphikanwendung dargestellt: Eine Datenbankanfrage wird von einem Datenbankapplet gestellt, das Daten von der Datenbank erhält und diese an den Datentopfverwalter weitergibt. Dieser legt sie im Datentopf ab. Bei Gelegenheit gibt der Datentopfverwalter die Daten an das Darstellungsapplet weiter oder das Darstellungsapplet fragt den Datentopfverwalter nach den Daten und stellt sie in jedem Falle dar.

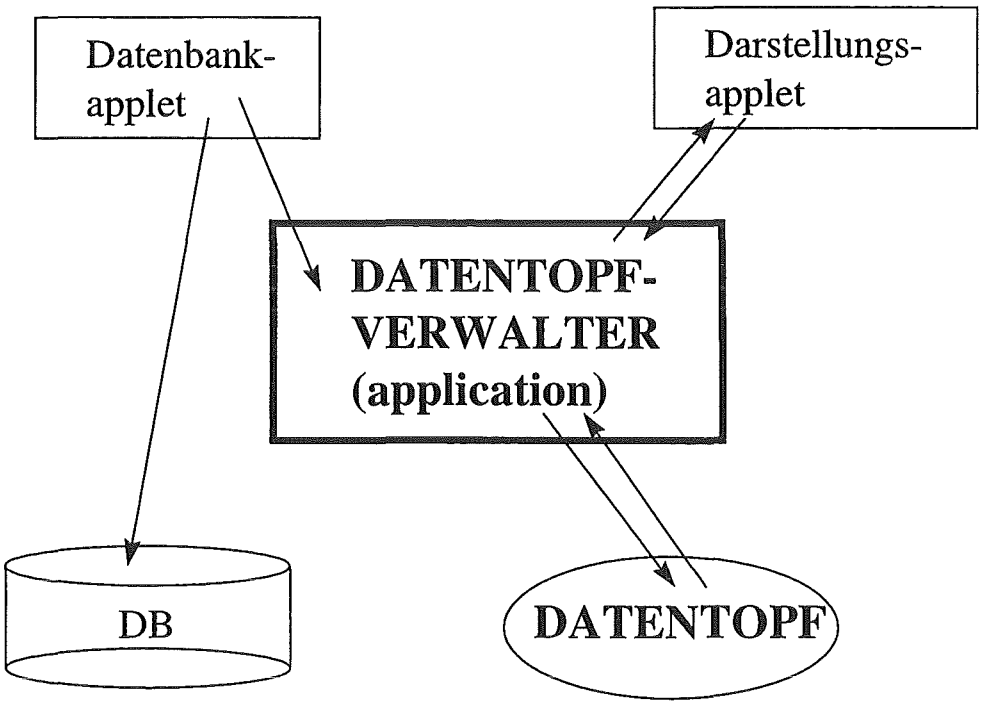


Abb. 1: Konzept des Datentopfs

Ein Datentopfverwalter muß die folgenden Eigenschaften besitzen:

- persistente oder temporäre Datenhaltung,
- Benutzerverwaltung,
- Sessionverwaltung,
- Unterstützung verschiedener Datenformate,
- aktive und passive Benachrichtigung.

Folgende Schnittstellen muß ein Datentopfverwalter minimal anbieten:

- Daten speichern,
- temporäre Daten persistent machen,
- Daten löschen,
- Daten holen.

4.2 Graphische Konzepte

Wie bereits im Abschnitt „Java für die graphischen Dienste" erläutert, definiert Java ein graphisches Fenstersystem, das Abstract Windowing Toolkit (AWT), innerhalb der standardisierten Java-Klassenbibliothek. Es abstrahiert hierzu von dem tatsächlich im Client-Rechner vorhandenen Graphiksystem (X11/Motif bzw. Windows). Aufgrund der Komplexität und der unterschiedlichen Architektur der abstrahierten Fenstersysteme bietet AWK nur eine Untermenge der Möglichkeiten dieser darunterliegenden realen Systeme an. Für die Ausgabe von Graphik definiert die abstrakte Klasse Graphics die Graphikprimitive Linie, Rechteck, Kreis, Oval, Polygon, Bilder und Text zusammen mit einer einfachen Attributierung. Eine Implementierung dieser Graphikklasse wird für die GUI-Komponente Canvas (eine gedächtnis- und aktivitätslose Zeichenfläche) angeboten. Für das Neuzeichen von Bildteilen, sowie die Skalierung der Graphik ist das Anwenderprogramm selbst verantwortlich. Eine Möglichkeit einer vom Ausgabegerät unabhängigen Graphikausgabe, etwa als PostScript- oder HPGL-Ausgabedatei, existiert standardmäßig bislang nicht.

Für die graphische Darstellungskomponente ist es unerläßlich, einen graphischen Kern aufbauend auf die von Java gebotene Grundfunktionalität zu implementieren. Erst durch einen solchen allgemeinen Graphikkern ist es möglich, komplexere Graphiken darzustellen, zu verwalten und zu modifizieren. Dieser Kern soll, angelehnt an graphische Standards wie GKS (Graphics Kernel System, ISO1985) Funktionalitäten wie

1. abstrakte Ausgabegeräte (PostScript, CGM, HPGL, ...),
2. abstrakte Eingabegeräte (Pick, Locator, ...),
3. strukturierte, hierarchische Graphikprimitive,
4. Weltkoordinatensysteme

bieten. Des weiteren soll er die Möglichkeiten, die sich durch die objektorientierte Sprache Java ergeben, nutzen. Am IPF wird derzeit ein solcher Graphikkern für Java entworfen und implementiert. Der zur Demonstration der Möglichkeiten von Java für das UIS entwickelte Prototyp des GIS-Terminal basiert auf ersten Ergebnissen dieses graphischen Kerns. Da die Firmen Sun und Adobe im Rahmen des Bravo-Projektes derzeit an einer starken Erweiterung der Graphikfähigkeiten von Java arbeiten, setzt der Entwurf seinen Schwerpunkt vor allem in die Realisierung von höheren, abstrakten Funktionalitäten und umgeht derzeit bewußt eine

Erweiterung der bislang schwach ausgeprägten Attributierung. Sobald die Graphikerweiterungen vorliegen, können diese in die unteren Schichten des Kerns integriert werden. Aufbauend auf diesen Graphikkern können weitere Schichten die Modellierung und Verwaltung von geographischen Objekten und Diagrammvisualisierungen realisieren, wie es bereits der Prototyp zeigt.

5. Demonstrationsbeispiele zur Veranschaulichung der Möglichkeiten

5.1 Entwicklung eines PollApplets zur Überwachung nichtblockierender Dienste im WWW

Externe Dienste (siehe Abschnitt 3.5) können nicht immer blockierend in Anspruch genommen werden. Um sie trotzdem im WWW nutzen zu können, muß folgendes Szenario realisierbar sein:

Über eine Reihe von HTML-Dokumenten, die von einem WWW-Server geladen werden, wird ein Eingabedatensatz erzeugt und zum Server zurückgeschickt. Dieser wird dann von dort an einen leistungsfähigen Rechner zur Auswertung weitergereicht. Nach dem Komplettieren des Eingabedatensatzes wird vom Server ein HTML-Dokument geladen, in das ein Applet mit einer "Poll-Funktion" integriert ist. Dieses Applet soll den Namen "PollApplet" tragen.

Über eine graphische Benutzeroberfläche kann der Nutzer dann das Pollintervall einstellen. Die Abfrageroutine läuft in einem separaten Thread, d.h. im Hintergrund, womit der Client-Rechner für andere Arbeiten nutzbar bleibt. In den eingestellten Zeitabständen öffnet der Thread des PollApplets nun eine Socketverbindung über einen definierten Port zum Server und wartet auf Meldungen von diesem. Für das PollApplet wurde der nicht belegte Port 5000 gewählt. Über die geöffnete Verbindung können beliebige Datenströme ausgetauscht werden.

Auf der Serverseite muß während der ganzen Zeit eine (Java-) Applikation laufen, die auf mögliche Verbindungsanfragen an einem bestimmten Port lauscht. Öffnet nun das Poll-Applet eine Socketverbindung zum Server, so startet dieser einen neuen Thread, der die Verbindung alleine abhandelt und nach Verbindungsabbruch durch den Client ebenfalls stirbt. Das Prinzip der Socketverbindung ist in folgender Graphik dargestellt.

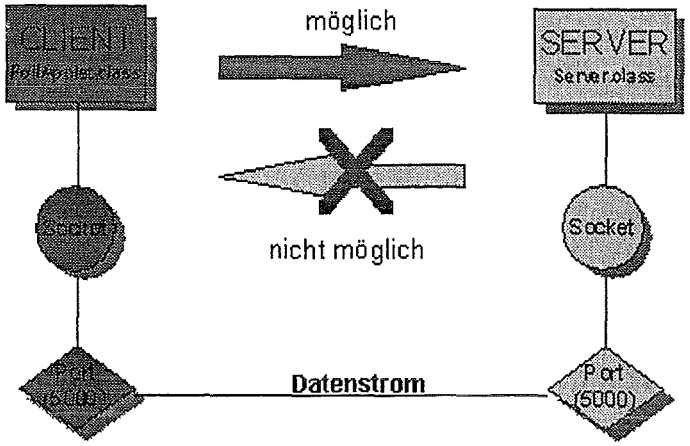


Abb. 2: Stream-Socket-Verbindung zwischen Client und Server

Liegt auf dem Server bei einer Anfrage weder ein Zwischen- noch ein Endergebnis der gestarteten Berechnung vor, so wird dies dem Client signalisiert, und er schließt die Socketverbindung. Anschließend wartet das PollApplet wieder für den vorgegebenen Zeitraum, um dann erneut eine Anfrage zu starten. Sobald aber ein Zwischen- oder Endergebnis auf dem Server vorliegt, wird dieses bei der nächsten Anfrage dem PollApplet mitgeteilt. Der Benutzer kann dann entscheiden, ob er das Ergebnis unmittelbar nach dem Eintreffen auf dem Client-Rechner, oder erst zu einem späteren Zeitpunkt weiterverwenden möchte.

Das folgende Diagramm zum Kommunikationsablauf erklärt schematisch die Arbeitsweise des PollApplets.

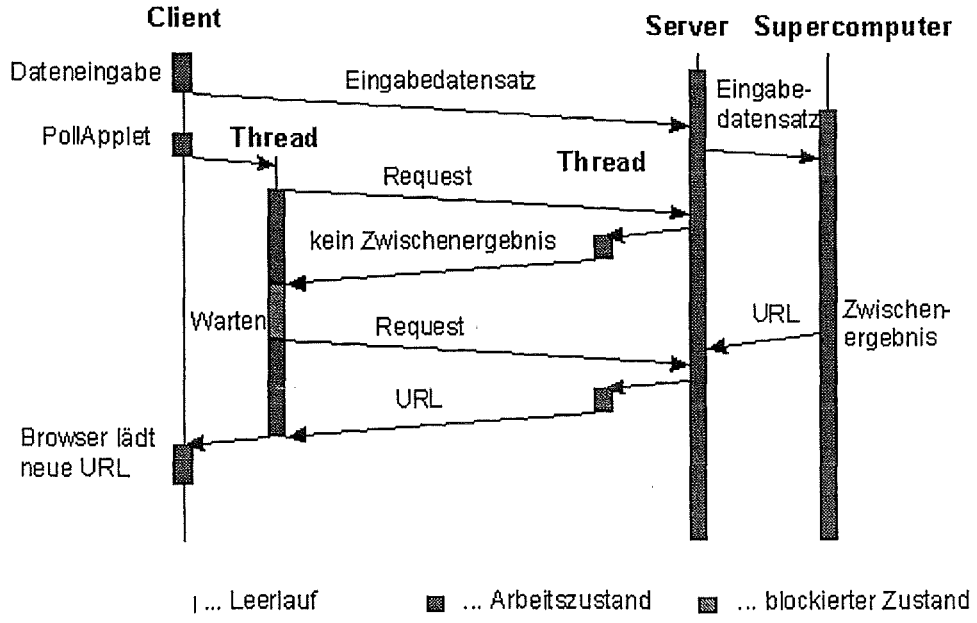


Abb. 3: Kommunikationsablauf des PollApplets

5.1.1 Die graphische Benutzeroberfläche des PollApplets

Im folgenden Screenshot ist die graphische Benutzeroberfläche des PollApplets abgebildet.

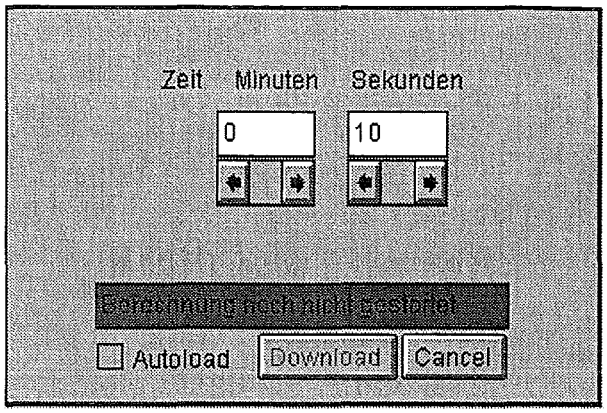


Abb. 4: Screenshot vom PollApplet

Des GUI ermöglicht dem Benutzer ein Eingreifen in den Pollvorgang. Das Zeitintervall zwischen zwei Requests am WWW-Server kann über zwei Scrollbars im Bereich Zeit eingestellt werden. Der Defaultwert der Gesamtzeit beträgt 10 Sekunden. Das größte einstellbare Zeitintervall hat eine Dauer von 59 Minuten und 59 Sekunden.

Unterhalb des Zeitbereichs liegt das Ergebnisfenster. Dieses Textfeld zeigt das Ergebnis der letzten Anfrage beim Server über den "Berechnungsstand" an, sowohl in Form einer Meldung in Textform, als auch durch Ampelsignalfarben. Zu Beginn des Pollvorganges ist dieses Fenster rot und enthält den Text: "Berechnung noch nicht gestartet.". Sobald die erste Anfrage erfolgt ist und dort noch kein Ergebnis vorliegt wird dies im Textfeld durch eine gelbe Einfärbung und der Meldung "Berechnung noch nicht fertig!" ausgegeben. Liegt das Ergebnis auf dem Server vor, so schaltet das Textfeld nach der nächsten Anfrage auf die Hintergrundfarbe grün um und signalisiert den augenblicklichen Status mit der Meldung: "Ergebnis liegt auf Server vor!".

Die unterste Zeile der graphischen Oberfläche des PollApplets ermöglicht dem Benutzer die Wahl zwischen einer Autoload-Checkbox und einem Download-Button. Zusätzlich ist noch ein Cancel-Button vorhanden, der jederzeit einen Abbruch des Pollvorganges ermöglicht indem er den "pollenden" Thread zerstört. Hat der User die default-mäßig deaktivierte Autoload-Checkbox aktiviert (Kreuz vorhanden), so wird automatisch bei der ersten Anfrage nach dem Eintreffen eines (Zwischen-) Ergebnisses auf dem Server, dieses zum Client übertragen. In diesem Fall bleibt der Download-Button deaktiviert. Ist die Autoload-Checkbox in ihrem Grundzustand, d.h. deaktiviert, so wird bei der ersten "positiven" Anfrage, also sobald ein Ergebnis auf dem Server vorliegt, der Download-Button aktiviert. Nun bleibt es dem Benutzer überlassen, wann er durch klicken auf den Download-Knopf die Ergebnisse vom Server lädt.

5.1.2 Vergleich des PollApplets mit CCI

Das Common Client Interface (CCI) ermöglicht externen Anwendungen auf Informationen und Funktionalitäten aus dem WWW zuzugreifen. Mit dieser Schnittstelle ist es möglich, innerhalb des NCSA Mosaic-Browsers, dem Netscape-Vorgänger, über vordefinierte Funktionalitäten (get, display, post, send, disconnect, quit) bestimmte Operationen lokal oder verteilt

durchzuführen. Im Rahmen von GLOBUS II wurde diese Technik verwendet, um zwischen verschiedenen Rechnern und darauf laufenden Prozessen Daten auszutauschen.

Vergleicht man nun die Einsatzmöglichkeiten von Java-Applets mit den Features der CCI-Schnittstelle am Beispiel der asynchronen Ergebnisrückmeldung über das Poll-Applet, so fallen folgende Vorteile zugunsten des Java-Applets auf:

CCI:

- Nach dem Starten der Simulation ist der Browser solange blockiert bis die Berechnung abgeschlossen ist und das Endergebnis zu ihm transferiert wurde.
- Mit dem Öffnen eines weiteren (Slave-)Browsers können jedoch Zwischenergebnisse vom Server empfangen und dargestellt werden.
- Ein interaktives Eingreifen in die Simulation durch den Benutzer vom Client-Rechner aus ist nicht vorgesehen.

Java-Applet:

- Wie das Ablaufdiagramm in Abb. 2 zeigt wird die Anfrage nach Zwischenergebnissen von einem separaten Thread aus durchgeführt. Der Browser bleibt deshalb während des Pollvorganges im Leerlauf und kann für andere Operationen genutzt werden.
- Zwischenergebnisse werden aufgrund des oben beschriebenen Thread-Handlings mit demselben Browser dargestellt, von dem auch die Simulation gestartet wurde. Ein Slave-Browser wird nicht benötigt.
- Das PollApplet sieht mit den vorhandenen Funktionen keine Interaktion mit der laufenden Berechnung vor. Lediglich eine Abfrageprozedur nach Zwischen- und Endergebnissen ist realisiert. Das vorhandene Potential von Java würde jedoch auch eine Interaktion übers Netz mit einer laufenden Berechnung ermöglichen, sofern die vorhandenen Sicherheitsrestriktionen beachtet werden.

5.2 Integrationsapplet: Graphische Darstellung von MEROS-Daten

5.2.1 Motivation

Innerhalb des Projektes GLOBUS wird von verschiedenen Projektpartnern Java eingesetzt, um die Benutzungsfreundlichkeit und Interaktivität von WWW-Applikationen zu erhöhen. Im Rahmen der AG Java entstand die Idee, noch im Laufe des Jahres 1996 eine komplexere Anwendung zu realisieren, die das Zusammenspiel der von den Partnern in Java entwickelten Komponenten demonstriert. Dieses Integrationsbeispiel soll

- die Vorteile einer Java-Lösung gegenüber einer klassischen Lösung mittels HTML- und CGI-Skripten zeigen,
- die Tragfähigkeit der neuen Technologie für größere Anwendungen prüfen,
- in einem Entwicklungsprojekt mit mehreren beteiligten Partnern die Integrationsmöglichkeiten aufzeigen (s. Abschnitt Inter-Applet-Kommunikation) sowie
- erste Anhaltspunkte für eine Einschätzung des Integrationsaufwandes liefern.

Eine ursprünglich geplante Beteiligung aller Institute war aufgrund der sehr begrenzten Zeit-

ressourcen nicht möglich. So erwies sich etwa der Einsatz des Poll-Applets für die asynchrone Ergebnisrückmeldung (IKE) im Rahmen eines ersten Integrationsbeispiels als nicht sinnvoll. Auch die statistische Aufbereitung der Ergebnisse wurde vom FAW nicht weiterverfolgt.

5.2.2 Szenario

Das Meßreihenoperationssystem (MEROS) /1/ verwaltet Meßwerte (Orts- und Zeitreihen), die von den verschiedenen Meßnetzen des Landes Baden-Württemberg geliefert werden. Ein Teil dieser Informationen wurde bereits im Laufe des Jahres 1995 über das WWW verfügbar gemacht (vgl. z.B. /3/). Dabei erwiesen sich allerdings die Möglichkeiten von HTML als nicht ausreichend, um interaktive Benutzungsoberflächen zu gestalten, die den von PC-Programmen gewohnten Komfort bieten, der für die Bedienung komplexer Systeme notwendig ist (vgl. /2/).

Das Integrationsbeispiel soll über das WWW und Java Zugang zu (ausgewählten) von MEROS verwalteten Luftmeßwerten bieten und die Meßreihen als Business-Graphiken im GIS-Viewer darstellen. Dazu ist eine komfortable Benutzungsoberfläche zu realisieren, die auch den mit den Daten nicht vertrauten Benutzern bei der Auswahl der ihn interessierenden Daten hinreichend unterstützt. Dabei sollen Eingabeüberprüfungen, soweit möglich, bereits auf Client-Seite im Moment der Eingabe stattfinden, um dem Benutzer bei Eingaben, die keine Ergebnisse liefern würden, rechtzeitig darauf hinzuweisen. Die Datenbankzugriffe auf die Oracle-Datenbank aus Java heraus sind mit der standardisierten Programmierschnittstelle JDBC zu realisieren. Die Ergebnisse sollen dem Benutzer auf übersichtliche Art und Weise präsentiert werden; einmal vom Server zum Client übertragene Ergebnisse sollen auf unterschiedliche Arten dargestellt werden können. Wichtige Darstellungsart sind insbesondere Graphen der Meßreihen, deren Raumbezug von der Darstellungskomponente und dem darauf basierenden GIS-Viewer veranschaulicht wird.

5.2.3 Komponenten

Das Integrationsbeispiel besteht aus mehreren Komponenten, die im folgenden kurz erläutert werden sollen.

5.2.3.1 JDBC-Treiber

Für den Zugriff auf ORACLE-Datenbanken wurde am FZI im Auftrag eines weiteren Forschungszentrums, ein JDBC-Treiber entwickelt. Dieser wird eingesetzt, um die Schnittstelle von Java zu den Meßwerten in der Datenbank zu realisieren.

Der Treiber ist kompatibel zur JDBC-Spezifikation Version 1.10. Er besteht aus mehreren Teilkomponenten: Die JDBC-Funktionen werden Client-seitig ausgeführt und kommunizieren mit einem entsprechenden Gegenpart auf Server-seite. Weil WWW-Browser wie Netscape nur eine Verbindung aus einem Applet zu dem WWW-Server erlauben, von dem das Applet geladen wurde, ist es erforderlich, eine Teilkomponente des JDBC-Treibers auf demselben Rechner laufen zu lassen, wie den WWW-Server. Da sich auf diesem Rechner weder die Datenbank, noch entsprechende Datenbankzugriffsprogramme, wie etwa SQL*net, befinden müssen, wurde ein eigenes Protokoll implementiert. Dieses erlaubt den entfernten Zugriff auf

JDBC-Treiber-Komponenten, die sich auf dem Datenbankrechner befinden. Ist die Firewall entsprechend freigeschaltet, kann so auch ein Firewall-Durchgriff geschehen. Ein solcher ist notwendig, wenn beispielsweise Daten öffentlich zugänglich gemacht werden sollen und die entsprechende Datenbank aufgrund des sehr großen Datenvolumens nicht vor die Firewall repliziert werden kann. Dies wäre voraussichtlich der Fall, wenn MEROS-Daten der Öffentlichkeit über on-line-Datenbankzugriffe verfügbar gemacht werden sollen.

5.2.3.2 Datenbank-Frontend

JDBC ist eine low-level Programmierschnittstelle, die eine gemeinsame Basis bietet, um "höhere", d.h. komplexere Werkzeuge mit einer abstrakteren Schnittstellen darauf aufzusetzen. Für die Benutzungsschnittstelle zu den MEROS-Daten wurden einige solcher Elemente konzeptioniert und realisiert. Diese Elemente sind allgemein einsetzbar und unabhängig vom konkreten Datenbankschema.

Wichtiges und häufig benötigtes Schnittstellenelement sind dynamische Selektionslisten. Diese Selektionslisten zeigen bestimmte Datenbankinhalte an, aus denen der Benutzer die ihn interessierenden Elemente auswählen kann. Ein Beispiel sind die Namen aller Meßstationen, aus denen der Benutzer diejenigen auswählen kann, von denen er Meßwerte erhalten möchte.

Die dynamischen Selektionslisten wurden in zwei Varianten mit jeweils unterschiedlichem Aussehen, aber prinzipiell identischer Funktionalität implementiert: In der ersten Form werden die Daten aus der Datenbank gelesen und als Liste mit Radio-Buttons vor jedem Listenelement angezeigt.

Durch Anklicken der Radio-Buttons können einzelne Listenelemente selektiert werden. Weitere Knöpfe bieten die Möglichkeit, die aktuelle Auswahl zu invertieren, alle Listenelemente auszuwählen oder die Auswahl rückzusetzen.

Liefert die Datenbank sehr viele Ergebnisse, eignet sich die beschriebene Form der Darstellung nur bedingt. Aus diesem Grund werden in einer weiteren Darstellungsform die Ergebnisse in einer scrollbaren Liste angezeigt. Ein Eingabefeld unterhalb der Liste ermöglicht es, durch Eingabe eines Buchstabens oder Wortanfangs innerhalb dieser Liste zu suchen. Durch-Doppelklick auf eines der Listenelemente kann dieses selektiert und in eine Selektionsliste übernommen werden.

Ein weitere häufig benötigte Funktionalität ist die Eingabe eines Zeitraums. Hier wurde ein Element zur Datumseingabe realisiert, das nur zulässige Daten als Eingaben erlaubt. Es kann weder der 30. Februar, noch ein Tag, der in der Zukunft liegt, eingegeben werden. Mittels eines Scrollbars kann eine Zeitspanne eingegeben werden, wie sie etwa erforderlich ist, wenn der Benutzer sich für die Meßwerte mehrerer Tage interessiert.

Die Benutzungsoberfläche des Datenbank-Frontends ist in Abb. 5 dargestellt.

Abb.5: Java-Datenbankfrontend für MEROS-Luftmeßwerte

Funktionalitäten, wie sie die dynamischen Selektionslisten, aber auch das Element zur Datumseingabe bieten, wären mit HTML-Formularen allein nicht realisierbar. Jede Benutzereingabe erforderte eine Client-Server-Kommunikation, damit das System die Eingaben entsprechend überprüfen und verarbeiten kann.

5.2.3.3 Prototyp GIS-Terminal

Der Prototyp des GIS-Terminal ermöglicht die Darstellung von Meßdaten in Form von Liniendiagrammen sowie die Darstellung von geographischen Daten innerhalb einer Kartendarstellung. Weiterhin wird die gemeinsame Präsentation beider Visualisierungen unterstützt. Für eine ausführliche Beschreibung des Konzepts und der Architektur des GIS-Terminals kann auf den Bericht „Architektur eines GIS-Terminal zur Visualisierung von Geodaten“ verwiesen werden. An dieser Stelle soll nur ein Überblick über die Möglichkeiten des Prototyps sowie

das Zusammenwirken mit dem MEROS-Selektor gegeben werden.

Das GIS-Terminal besitzt eine Benutzeroberfläche, die an die gängigen Standards (Windows, X11-Motif) angelehnt ist. Das Hauptfenster des GIS-Terminal ist in Abbildung 6 zu sehen. Man erkennt, daß es nicht im WWW-Browser dargestellt wird, sondern sich wie eine eigene Applikation transparent in die Benutzeroberfläche des Client-Rechners integriert. Durch diese Integration, sowie durch die Berücksichtigung etablierter Benutzeroberflächenstandards soll der Anwender die ihm bekannten Konzepte (etwa aus diversen Office-Tools) auf die Bedienung des GIS-Terminal übertragen können und dadurch eine kurze Einlernzeit erreichen.

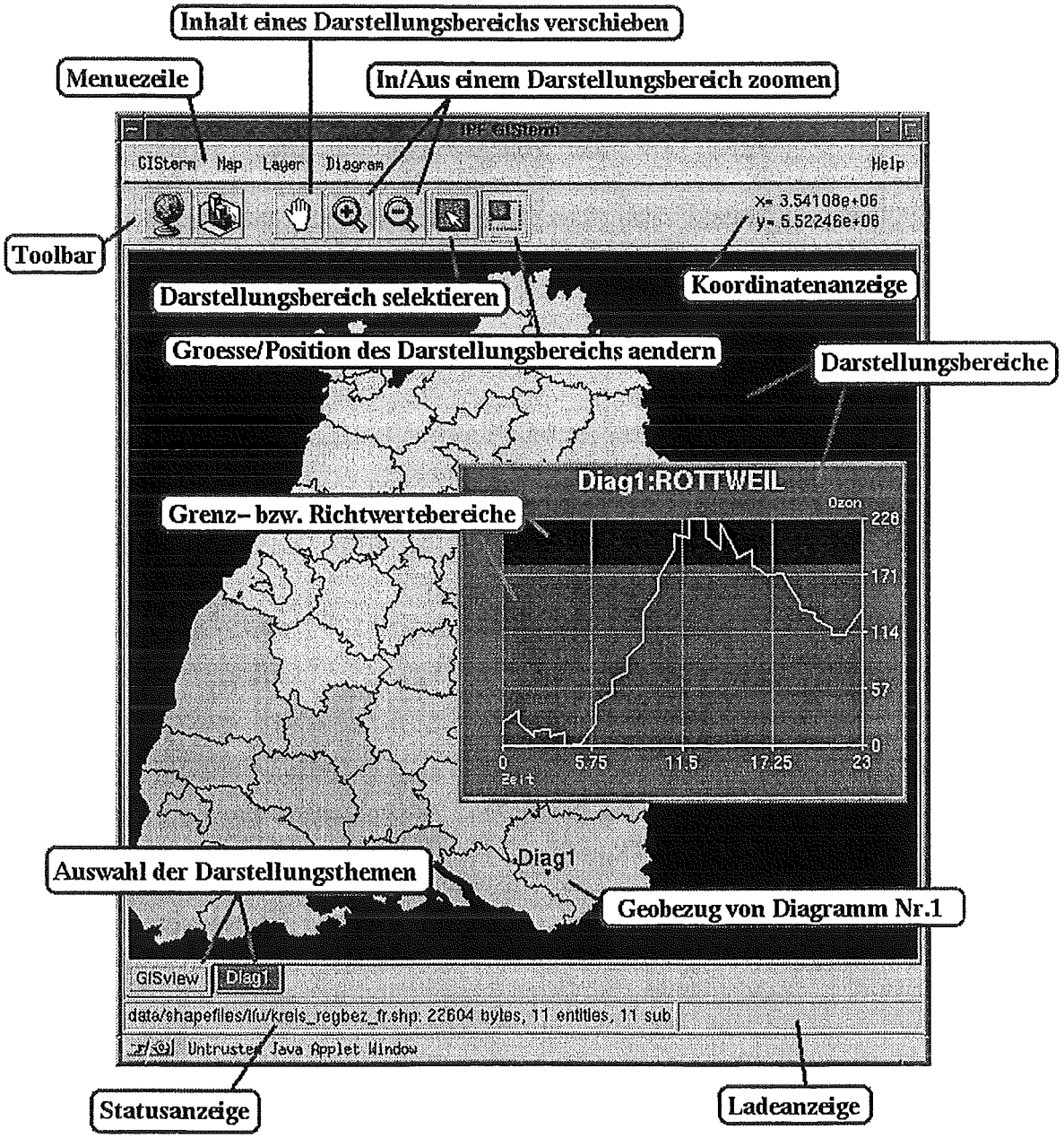


Abb. 6: Hauptfenster des GIS-Terminal

Das Fenster besitzt demnach eine Standardaufteilung, die sich in folgende Komponenten gliedert:

- Menüzeile am oberen Fensterrand

- Toolbar unterhalb der Menüzeile, in der häufig benötigte Operationen zugänglich sind.
- Der Arbeitsbereich, in dem die Graphik angezeigt wird.
- Ein Karteikartensystem unterhalb des Arbeitsbereichs, mit dem zwischen verschiedenen Arbeits- bzw. Darstellungsthemen umgeschaltet werden kann.
- Einer Statuszeile sowie einer Ladeanzeige am unteren Fensterrand.

Hat ein Benutzer, mit Hilfe des oben beschreibenen MEROS-Selektors, eine Meßreihe ausgewählt und die Abfrage an die Datenbank abgeschickt, werden die Ergebnisdaten nach erfolgreicher Bearbeitung vom MEROS-Selektor in einer Textliste dargestellt. Er kann nun diese Daten innerhalb des GIS-Terminal visualisieren. Dazu muß er lediglich den Button (Bedienknopf) „Visualisierung" unterhalb der Textliste betätigen. Der MEROS-Selektor übergibt jetzt die Meßwerte an das GIS-Terminal. Dort wird daraus ein Liniendiagramm generiert und dargestellt. Zu bemerken ist hierbei, daß weder zur Datenübergabe noch zur anschließenden Visualisierung auf den Server zugegriffen werden muß. Alle Aktionen laufen auf dem Client-Rechner ab, und belasten somit weder Netz noch Server. Falls der Benutzer mehrere Meßreihen erfragen und visualisieren möchte, kann er diesen Vorgang beliebig wiederholen. Jedes erzeugte Diagramm wird als eigenständiges Darstellungsthema in einer eigenen Anzeige visualisiert. Der Benutzer kann durch das Karteikartensystem im unteren Fensterteil beliebig zwischen diesen Darstellungsthemen wechseln. Falls für das jeweilige Diagramm Grenz- bzw. Richtwerte vorhanden sind, werden diese im Diagramm durch entsprechend rote bzw. grüne Bereiche angezeigt. Das Darstellungsthema „GISview" ist immer vorhanden. Hier werden geographische Daten visualisiert. Wählt man dieses Darstellungsthema, werden zunächst die geographischen Bezüge der anderen Darstellungsthemen (Diagramme) in einer noch leeren Karte angezeigt. Man kann jetzt unter dem Menüpunkt „Layer" verschiedene Hintergrundkarten in das GIS-Terminal laden und darstellen. Derzeit werden dazu die folgenden Kartenlayer zur Auswahl angeboten.

- Regierungsbezirke in Baden-Württemberg
- Kreise in Baden-Württemberg aufgeteilt nach Regierungsbezirken
- Gemeinden in Baden-Württemberg aufgeteilt nach Regierungsbezirken

Weiterhin können alle weiteren Darstellungsthemen in den „GISview" interaktiv übernommen werden. Dazu muß man in das entsprechende Darstellungsthema wechseln und den Menüpunkt „Diagam/Add diagram to map" anwählen. Das in Abbildung 6 gezeigte Diagramm wurde auf diese Weise in die Kartendarstellung übernommen. In allen Darstellungen können interaktive Operationen ausgeführt werden. So zeigt die Koordinatenanzeige immer die Koordinaten der Mausposition in der Einheit des jeweiligen Darstellungsthemas an. Bei Diagrammen werden z.B. Zeitpunkt und Ozonkonzentration des unter dem Mauscursor liegenden Punktes angezeigt. Innerhalb der Kartenanzeige werden die geographischen Koordinaten im Gauß-Krüger-System angezeigt. Die in der Toolbar angeordneten Operationen ermöglichen dem Benutzer eine Datenpräsentation nach seinen Wünschen interaktiv zu erstellen. Hierzu kann er in Darstellungen hinein- bzw. herauszoomen. Er kann die dargestellten Inhalte mit der Maus verschieben. Weiterhin kann er die in der Karte angezeigten Darstellungsthemen beliebig auf der Karte positionieren sowie in der Größe interaktiv anpassen.

Der Prototyp zeigt in anschaulicher Weise die Interaktionmöglichkeiten die mit Java innerhalb einer WWW-Anwendung erreicht werden können. Im Vergleich zu einer rein serverseitig bzw. HTML basierenden Lösung ist Java als deutliche Innovation zu werten.

5.2.4 Integration

Um die einzelnen Komponenten zu einem Gesamtsystem zu integrieren, wurde als optimale Lösung das Datentopf-Konzept angesehen. Allerdings hätte eine vollständige Realsierung dieses Konzeptes den zeitlich engen Rahmen bei weitem gesprengt. Eine deutlich vereinfachte Realisierung allein für dieses Beispiel, die anschließend hätte verworfen werden müssen, wurde für nicht sinnvoll erachtet. Eine Integration auf Klassen- und Methodenebene („Big Applet"-Lösung) wurde für das Beispiel bevorzugt. Diese Integrationslösung bot zudem den Vorteil des effektiven Datenaustauschs der Komponenten über die Methodenschnittstellen. Zudem kann das ganze Applet komprimiert abgelegt werden und vergleichsweise schnell über das Netz übertragen werden. Der erforderliche erhöhte Aufwand für die Abstimmung fiel aufgrund der räumlichen Nähe der beteiligten Projektpartner nicht ins Gewicht.

5.2.5 Zusammenfassung der Ergebnisse

Java-Applets ermöglichen architektur-neutrale und benutzungsfreundliche Schnittstellen zu verschiedenen Arten von Anwendungen, wie etwa Datenbank-Anwendungen und Geoinformationssystemen. Das realisierte Beispiel zeigt, wie derartige komfortable Anwendungen der gewohnten Bedienungsfreundlichkeit von PC-Anwendungen nahe kommen. Die von diesen Anwendungen gebotene Performanz kann von in Java realisierten WWW-Anwendungen allerdings nicht erreicht werden, da zum einen Daten und Programme nicht von der lokalen Festplatte geladen werden, sondern über das Netz von entfernten Servern. Allerdings werden einmal geladene Applets wie statische HTML-Seiten im Cache des WWW-Browsers gehalten und können anschließend innerhalb der aktuellen Sitzung von der lokalen Festplatte geladen werden. Zum anderen ist die Geschwindigkeit der Interpretation des Java-Byte-Code auf verschiedenen Plattformen sehr unterschiedlich. Verglichen mit WWW-Anwendungen, die mittels klassischem HTML- und CGI-Ansatz realisiert sind, kann aufgrund leistungsfähiger Übertragungsprotokolle und einer möglichen Lastbalancierung zwischen Client- und Server allerdings eine deutliche Leistungssteigerung erzielt werden.

Der Aufwand für eine Integration verschiedener Komponenten variiert mit der gewählten Integrationsmethode. Er muß prinzipiell nicht größer als der Aufwand für die Integration mehrerer HTML- und CGI-Komponenten sein. Durch den Gewinn an Möglichkeiten kann eine den jeweiligen Anforderungen optimal entsprechende Lösung gewählt werden.

Der gewählte Ansatz der Integration auf Klassen- und Methodenebene unterscheidet sich grundsätzlich nicht von der Integration von Komponenten anderer objektorientierter Programmiersprachen. Gegenüber C++ allerdings ergeben sich wesentliche Vorteile aufgrund einiger verbesserter Spracheigenschaften, wie etwa der Bereichsüberprüfung von Feldvariablen. Dies führt zu einer Verminderung unbeabsichtigter Seiteneffekte und verringert den Aufwand für Debugging, Wartung und Pflege.

Deutlich mehr Möglichkeiten gegenüber HTML pur bietet Java in Bezug auf die Gestaltung der Benutzungsoberfläche. Den enormen Vorteilen steht als Nachteil auch die Gefahr inkonsistenter und unterschiedlicher Benutzungsoberflächen verschiedener Komponenten eines Systems gegenüber. Hier gilt es rechtzeitig durch die Festlegung entsprechender Richtlinien gegenzusteuern.

6. Zusammenfassung

Das Jahr 1996 erlebte einen Boom einer neuen, objektorientierten Programmiersprache: Java. Selbst Medien, die ansonsten selten über informationstechnische Entwicklungen berichten, brachten Beiträge über diese Programmiersprache und entfachten ein Öffentlichkeitsinteresse, das etwa bei der Einführung von C++ undenkbar gewesen wäre.

Ob es tatsächlich die prophezeite Java-Revolution gibt, hängt wesentlich davon ab, ob die Konzepte in der Praxis ihre Einsatzfähigkeit beweisen und ob mit Java entwickelte Anwendungen halten, was Sun verspricht.

Schwierige Aufgabe der AG Java war die Untersuchung einer neuen Technologie, die sich rasant wie kaum eine andere weiterentwickelt. Diese Untersuchung sollte Grundlage der Bewertung des möglichen Einsatzes von Java im Rahmen des Umweltinformationsystem (UIS) Baden-Württemberg sein.

Trotz einiger Kinderkrankheiten, die weniger der Sprache selber sondern eher den Werkzeugen wie Entwicklungsumgebungen und Java-fähigen WWW-Browsern anhaften, ist das Bild der AG Java insgesamt positiv ausgefallen. Innerhalb kürzester Zeit konnte das notwendige Know-How erworben werden, um auch größere Anwendungen mit Java zu realisieren, die viele Schwächen von klassischen WWW-Anwendungen beseitigen konnten. Die mittlerweile breite Java-Unterstützung verschiedenster Hersteller - selbst Suns Erzrivale Microsoft konnte sich dem Java-Fieber nicht entziehen; IBM beschäftigt nach eigenen Angaben mittlerweile (Dezember 96) mehr Java-Programmierer als Sun - scheint die AG Java zu bestätigen: Mit Java beginnt eine neue Epoche von WWW-Anwendungen.

Zu überschwenglicher Euphorie besteht allerdings kein Anlaß. Welche bereits existierenden Anwendungen auf Java umgestellt werden, welche Neuentwicklungen in Java entwickelt werden oder ob einem Warten auf die Beseitigung einiger Unzulänglichkeiten der Vorzug zu geben ist, sollte jeweils im Einzelfall entschieden werden. Java allein ist auch kein Allheilmittel: Zum Teil sind neue Konzepte erforderlich; zum Teil ist es durchaus sinnvoll, bislang bewährte Technologien auch weiterhin einzusetzen; zum Teil kann auch ein Zusammenspiel von Java mit anderen Technologien die optimale Lösung darstellen.

7. Literatur

/1/ Ackermann, Klaus (1995): MEROS-Dokumentation. Technischer Bericht, Landesanstalt für Umweltschutz (LfU), Karlsruhe

/2/ Bußmann, Martina; Heißler, Werner (1996): Testbericht GLOBUS II, unveröffentlicht, Landesanstalt für Umweltschutz, Karlsruhe

/3/ Scheffzcyk, Oliver (1995): Entwurf und Realisierung von Umweltinformationssystem-Diensten zur Nutzung in Weitverkehrsnetzen. Diplomarbeit, Universität Karlsruhe, Fakultät für Informatik, Karlsruhe

/4/ Kramer, Ralf; Nikolai, Ralf; Rolker, Claudia (1996): Java-Evaluierung für das UIS Baden-Württemberg - Phase 1. FZI Bericht 5/96.

/5/ Behrens, Sven; Nikolai, Ralf (1996): Interaktive und dynamische WWW-Anwendungen mit Java, FZI-Bericht 2/96

Teil 2:

Konzeption und Implementierung von Anwendungskomponenten

WWW-UDK/Metadaten

R. Kramer, R. Nikolai,
Forschungszentrum Informatik (FZI),
Haid-und-Neu-Str. 10-14,
76131 Karlsruhe

1. WWW-UDK - Der Standard der deutschsprachigen Länder für UDK-Zugriffe im WWW

1.1 Ausgangsbasis

Um (komplexe) Fragestellungen mit Umweltbezug beantworten zu können, ist es erforderlich, Informationen aus verschiedenen Anwendungsbereichen, die verteilt an unterschiedlichen Stellen vorliegen können, zu Verfügung zu haben. Nur bei ausreichender Verfügbarkeit der relevanten Informationen ist das Erkennen komplexer Zusammenhänge und Kausalketten, eine effektive Überwachung beschlossener Umweltgesetze, aber auch die wechselseitige Nutzung vorhandener Informationen möglich. So können Doppelerhebungen, aber auch sogenannte „Datenfriedhöfe", vermieden werden.

Die Beantwortung einer bestimmten Fragestellung erfordert also zuerst, in Erfahrung zu bringen, welche Informationen verfügbar sind, wo sie verwaltet werden und wie diese Informationen bezogen werden können. Die dazu notwendigen Informationen über Informationen werden Metainformationen genannt /10/. Sie können verglichen werden mit den Informationen klassischer Karteikarten in Bibliothekskatalogen, die Bücher beschreiben, aber nicht die Bücher selbst sind /4/. Bei der derzeitigen regelrechten Datenexplosion, zu der die Sammlung verschiedenartigster Daten führt, ist der Datensuchende mehr denn je auf Informationen über vorhandene Informationen angwiesen, um die für ihn relevanten zu finden. Den Bibliothekskatalogen, die Karteikarten, also Metainformationen enthalten, entsprechen in der allgemeineren, elektronischen Form Metainformationssysteme, auch Auskunfts- und Nachweissysteme oder einfach Datenkataloge genannt.

1.1.1 Umwelt-Datenkatalog (UDK)

Der Umwelt-Datenkatalog (UDK) ist ein solches Metainformationssystem /18/, /19/, /21/, /28/, speziell für Umweltinformationen. Ursprünglich wurde er auf eine Initiative des Niedersächsischen Umweltministeriums hin entwickelt. Im Laufe der Zeit hat sich eine internationale Länderkooperation gebildet, die für seine Einführung sorgte und dessen Weiterentwicklung fördert. Inzwischen wird er in fast allen Bundesländern sowie in Österreich eingesetzt und hat sich als Standard für Metainformationen über Umweltinformationen etabliert.

Im UDK werden beschreibende Informationen und Zugangswege[1] zu Datenbeständen des Umweltinformationssystems (UIS) bereitgestellt. Der UDK im UIS Baden-Württemberg wird den Übergreifenden UIS-Komponenten zugeordnet. Das sind Systeme, die der Zusammenführung und fachübergreifenden Nutzung von Informationen aus verschiedenen Umwelt- oder Zuständigkeitsbereichen dienen /23/. Damit spielt der UDK im UIS eine zentrale Rolle.

[1] Häufig auch als navigatorische Metainformationen bezeichnet.

1.1.2 World-Wide Web (WWW)

Das World-Wide Web (WWW) bietet die Möglichkeit, Zugriffe auf heterogene Datenbestände plattformunabhängig und unter einer einheitlichen Oberfläche zugänglich zu machen. Schlüsselfaktoren seines enormen Erfolges sind die Einfachheit, mit der der Benutzer weltweit verteilte Informationen benutzen, referenzieren und auch bereitstellen kann; die komfortable graphische Benutzungsoberfläche zum plattformunabhängigen Browsen von WWW-Dokumenten in Form von WWW-Browsern wie Netscape Navigator oder Microsoft Internet Explorer; seine Kompatibilität zu bereits existierenden Protokollen wie etwa dem File Transfer Protocol (FTP) oder dem Network News Transfer Protocol (NNTP); eine Menge öffentlicher Spezifikationen (z.B. HTML, HTTP) zusammen mit frei verfügbaren Quell-Code-Biblioteken. Aufgrund dieser Faktoren bietet sich das WWW ganz besonders für die Veröffentlichung von für Recherchen notwendigen Metainformationen an, da ein sehr großer Benutzerkreis Interesse an derartigen Informationen hat und dieser über das WWW erreicht werden kann. Mittels WWW-Technologie, die nicht nur im Internet, sondern auch in firmen- oder behördeneigenen Intranets eingesetzt werden kann, ist es möglich, sowohl behördeninternen Informationsbedarf als auch den Informationsbedarf der Öffenlichkeit kostengünstig zu befriedigen.

1.1.3 Umweltinformationsgesetz (UIG)

Der Zugang zu Umweltinformationen für die interessierte Öffentlichkeit ist durch das am 16. Juli 1994 in Kraft getretene „Gesetz zur Umsetzung der Richtlinie 90/313/EWG des Rates vom 7. Juni 1990 über den freien Zugang zu Informationen über die Umwelt“ rechtlich gesichert. Zugänglich sind nach § 2 UIG Umweltinformationen, die bei Umweltbehörden und bei Personen des Privatrechts, derer sich Umweltbehörden zur Erfüllung ihrer umweltbezogenen öffentlich-rechtlichen Aufgaben bedienen, aufgrund der Ausführung dieser Aufgaben vorhanden sind.

1.1.4 WWW-UDK - Historie

Der WWW-UDK soll dazu beitragen, das Umweltinformationsgesetz umzusetzen. Er ist das Ergebnis eines aus den oben genannten Gründen 1994 am Forschungszentrum Informatik (FZI, Forschungsgruppe Datenbanksysteme) gestarteten Forschungsauftrages des Umweltministeriums des Landes Baden-Württemberg /13/, /14/, /15/, /17/. Ende des Jahres 1994 wurde der erste prototypische on-line-Datenbankzugriff mit den Techniken und Werkzeugen des WWW auf die Datenbestände des UDK entwickelt und realisiert. 1995 wurde die Lösung im Auftrag des Umweltministeriums Baden-Württemberg erweitert sowie im Auftrag des Bundesministeriums für Umwelt, Österreich, an spezifische Anforderungen, wie etwa mehrsprachige Masken, angepaßt.

Im Laufe des Jahres 1996 fand die Umstellung der WWW-Anwendung auf das internationalisierte Datenmodell UDK 3.0 statt. Um den Anforderungen des globalen Internets gerecht zu werden, wurden mehrsprachige Masken sowie mehrsprachige Hilfetexte realisiert. Den Anforderungen des gelegentlichen Benutzers nach einem komfortbalen und einfach zu bedienendem Recherchetool wurde durch eine Verbesserung der Software-Ergonomie unter Berücksichtigung der raschen technologische Entwicklung Rechnung getragen. Die

Konfigurierbarkeit gewährt eine den verschiedenen Ministerien angepaßte Software-Lösung. In den folgenden Abschnitten wird der aktuelle Stand der Betriebsversion des WWW-UDK näher erläutert.

1.2 Anforderungen

Die Anforderungen können wie folgt zusammengefaßt werden:

- Die Modellierung der Metadaten des UDK soll unverändert bleiben. Nur so ist gewährleistet, daß die Daten weiterhin auch von der PC-Applikation des UDK gelesen und gepflegt werden können. Vollständig kompatible Erweiterungen des Datenmodells, die die Funktionsfähigkeit der PC-Applikation nicht beeinträchtigen, sind möglich. Dies kann etwa in Form von Zusatztabellen geschehen, die „neben" die Kern-Tabellen getsellt werden.
- Der Zugriff über das WWW auf den UDK ist sowohl im verwaltungsinternen Netz (Intranet) als auch im Internet hauptsächlich lesend, da im WWW derzeit nur die Recherche, nicht aber das Erfassen der Metadaten unterstützt werden soll. Die Erfassung der Metadaten soll wie zuvor mit der PC-Applikation geschehen.
- Die Recherchefunktionalität der WWW-Applikation soll in etwa der Recherchefunktionalität der PC-Applikation entsprechen. Da über das WWW allerdings ein weitaus größerer Benutzerkreis erreicht werden kann und insbesondere die Zahl der gelegentlichen Nutzer weitaus größer ist, ist die Benutzungsfreundlichkeit gegenüber der PC-Appliaktion deutlich zu verbessern. Die Komplexität des Datemmodells sowie der Anfragen sind vor dem Benutzer zu verbergen.
- Die Systemarchitektur soll auf den Techniken und Werkzeugen des WWW basieren. So ist etwa das Hypertext Tranfer Protocol (HTTP) als Kommunikationsprotokoll und das Common Gateway Interface /24/ als Schnittstelle zwischen WWW-Server und der UDK-Datenbank beziehungsweise den darauf operierenden Diensten zu benutzen.
- WWW-Anwendungen sind, wenn sie im Internet verfügbar gemacht werden, weltweit erreichbar. Für nicht deutschsprachige Informationssuchende soll eine englischsprachige Benutzungsschnittstelle zur Verfügung stehen. Neben Deutsch und Englisch sollen weitere Sprachen leicht ergänzt werden können.
- WWW-UDK soll nicht nur auf einer bestimmten Unix-Plattform mit einem bestimmten Datenbankmanagementsystemen (DBMS) laufen, sondern ohne großen Aufwand auf verschiedene Unix-Derivate und relationale DBMSe portiert werden können. Dies ermöglicht einen Einsatz auch außerhalb des Umweltinformationssystems Baden-Württemberg.

1.3 System-Architektur

Um von Datenbankmanagementsystemen (DBMS) verwaltete Daten im WWW verfügbar zu machen, existieren verschiedene Möglichkeiten. Wenn offene und portable Systeme verlangt sind, die in heterogenen Systemumgebungen ausgeführt werden sollen (z.B. verschiedene WWW-Server, DBMSe, Betriebssysteme), sind proprietäre Lösungen nicht akzeptabel.

Auch die periodische Extraktion von Daten und deren Aufbereitung zu statischen HTML-Seiten, die untereinander vernetzt werden, bietet sich nicht für flexible Recherchen in einem Metainformationssystem an. Der Aufbau eines Netzes statischer HTML-Seiten bedingt, daß

die Zugriffsfragen der zukünfigen Benutzer im voraus bekannt sind, da der Zugriff primär navigierend, d.h. entlang der aufgebauten Verzeigerung der Seiten, erfolgt. Deklarative Suchen sind demgegenüber nur sehr eingeschränkt möglich, da keine vollständige Anfragesprache, wie SQL bei relationalen Datenbanksystemen, zur Verfügung steht.

Um die erforderliche mächtige und flexible Anfragefunktionalität zu realisieren, wurden online Datenbankzugriffe implementiert. Breit unterstützte und weit verbreitete Standards und Techniken wie HTTP, HTML, CGI-Skripte, C-Programme mit eingebetteten SQL-Anweisungen garantieren sowohl Client- als auch Server-seitige Portierbarkeit.

Der Ansatz ist diensteorientiert, d.h. es wurden Dienste zum Informationszugriff (Datenzugriffsdienste), zur Informationsverarbeitung (Datenverarbeitungsdienste bzw. Datenaufbereitungsdienste) sowie zur Informationsaufbereitung für den Benutzer (Darstellungsdienste) realisiert.

Horizontal kann zwischen Systemdiensten und Anwenderdiensten unterschieden werden. Sytemdienste stellen systemnahe Basisfunktionalitäten zur Verfügung, wie etwa die eigentlichen Datenbankzugriffe. Anwenderdienste sind grundsätzlich eine Abfolge von Systemdiensten, etwa eines Datenbankzugriffsdienstes, um die Daten aus der Datenbank anzufordern, eines Datenverarbeitungsdienstes um auf den erhaltenen Daten Operationen auszuführen sowie eines Darstellungssdienstes, um anhand der prozessierten Daten HTML-Seiten zu erzeugen. Anwenderdienste stellen somit die Schnittstelle zum Benutzer dar.

Der gewählte Ansatz erlaubt mit vergleichbar geringem Aufwand die Integration der Dienste in eine Middleware-basierte (oder allgemeiner: n-tier) Architektur. Exemplarisch wurde dies bereits für ausgewählte UDK-Dienste durchgeführt. In einer solchen Architektur ist es prinzipiell auch möglich, die Systemdienste aus Java-Applets heraus auszurufen.

1.4 Datenmodell UDK 3.0.1

1.4.1 Allgemeines

Die Datenbestände des UDK werden von einem relationalen Datenbankmanagementsystem verwaltet. Die Daten können drei verschiedenen Kategorien zugeordnet werden: Umweltmetadaten, Adressen und Thesaurusdaten. Die Umweltmetadaten beschreiben Umweltdatenbestände einschließlich ihres Fach-, Raum- und Zeitbezugs. Sie bestehen aus sogenannten UDK-Objekten. Jedes UDK-Objekt beschreibt genau ein Umweltdatenobjekt. Eine Indexierung der UDK-Objekte findet mit dem Umweltthesaurus (vgl. 1.7.5) des Umweltbundesamtes /1/ statt, der ein kontrolliertes Zugangsvokabular liefert. Die Menge der UDK-Objekte wird durch einen hierarchischen Strukturbaum, den Primärkatalog, geordnet. Um eine andere Sicht auf UDK-Objekte zu erzeugen, können zusätzlich Sekundärkataloge angelegt werden. Jedem UDK-Objekt ist die Adresse einer datenhaltenden Stelle zugeordnet, zusätzlich kann eine weitere Adresse angegeben werden, die für Datenauskünfte zur Verfügung steht. Diese Institution ist Ansprechpartner für Fragen bezüglich der Metadaten und auch der von den Metadaten beschriebenen Basisdaten.

1.4.2 Änderungen gegenüber UDK 2.0

Bei der 3. Version des UDK wurden einige Verbesserungen im Datenmodell vorgenommen. So wurden neue Tabellen eingeführt (z.B.: ThesVar) bzw. aus bereits bestehenden gebildet (z.B.: die Tabellen obj und obj_bezug wurden zur Tabelle obj zusammengefaßt). Außerdem wurden in einigen Tabellen neue Attribute eingefügt (z.B.: obj_class in der Tabelle obj und adr_type in der Tabelle institution), bereits vorhandene durch neue ersetzt (z.B.: kennzeichen und katalogart in der Tabelle catalogue wurden durch cat_type ersetzt), wodurch sich manchmal auch der Wertebereich der Attribute änderte, und einige Attribute wegfielen (z.B.: pstaat, pland in der Tabelle dek). Darüberhinaus wurden die Bezeichner aller Tabellen und Attribute auf englisch umgestellt, um für den internationalen Einsatz gerüstet zu sein. Da es sich nicht immer um die exakte Übersetzung handelt, wurde die Zuordnung von alten Attributen zu neuen erschwert; dafür wurden aber die Attributnamen besser strukturiert. So beginnen jetzt zum Beispiel alle Attribute der Tabelle obj, die zum Fachbezug gehören, mit subj_ und alle, die zum Zeitbezug gehören, mit time_.

Eine wesentliche konzeptionelle Neuerung in der 3. Version ist die Einführung eines Klassenkonzepts, so daß momentan zwischen UDK-Objekten aus neun Klassen unterschieden werden kann. Das objektorientierte Klassenkonzept wird auf die relationale Datenbank des UDK mit Hilfe einer „großen" Relation abgebildet, die alle Attribute aller Klassen enthält. In Abhängigkeit davon, zu welcher Klasse (Attribut obj_class in der Tabelle obj) ein Objekt gehört, wird es mit Hilfe bestimmter Attribute beschrieben. Dies wird jedoch nicht vom Datenmodell erzwungen, sondern muß mit Hilfe eines Erfassungsprogramms sichergestellt werden. So gibt es Attribute, die für alle Objekte einen Wert annehmen können, wie z.B.: Name, UDK-Klasse (obligatorisch, auch wenn es nicht vom Datenmodell erzwungen wird), gesetzliche Grundlage, Zeitbezug und solche, die nur bei Objekten einer bestimmten Klasse einen Wert annehmen, wie z.B.: Blattnummer und Maßstab für Karten, Nachweisgrenze für empirische Daten, Anwendungszweck bzw. -gebiet für Produkte bzw. Beteiligte bei Vorhaben.

Eine detaillierte Beschreibung des UDK-Datenmodells findet sich in der Original-UDK-Dokumentation /7/.

1.4.3 Auswirkungen der Änderungen auf die WWW-Applikation

Aufgrund der beschriebenen Änderungen des Datenmodells mußten alle bestehenden Programmmodule, insbesondere die dynamisch zusammengesetzten SQL-Anfragen, geändert und um die Auswahl einer Objektklasse ergänzt werden. Neben der Umbenennung aller Tabellen und Attribute mußten die Anfragen auch an Änderungen der Wertebereiche mancher Attribute (z.B.: type der Tabelle obj_search konnte in der 2. Version die Werte "F" bzw. "T" für freie bzw. Thesaurus-Suchbegriffe annehmen und in der 3. Version die Werte "1" bzw. "2") sowie an das Klassenkonzept angepaßt werden, was die Abfrage weitere Attribute in Abhängigkeit der Objektklasse erfordert.

Auch die Detail-Ausgabe der Objekte mußte an das neue Klassenkonzept angepaßt werden, d.h. es werden nur die für die entsprechende Klasse in Frage kommenden Attribute in einer Tabelle ausgegeben. Dabei werden die Zeilen der Tabellen, ebenfalls in Abhängigkeit der

Klasse, mit einem entsprechenden Text versehen (z.B.: wird das Attribut sub_name der Tabelle obj bei Objekten der Klasse „Gutachten" unter der Bezeichnung Titel/Thema, bei „Empirischen Daten" unter Erhebungsgröße, bei „Daten zu Anlagen" unter Anlagentyp und bei „Vorhaben" unter Gegenstand ausgegeben).

Die unterschiedliche Art der Datenspeicherung in Österreich und in den deutschen Bundesländern (trotz identischem Datenmodell) wird vom System für den Benutzer verdeckt.

1.4.4 Ergänzungen des Datenmodells für die WWW-Applikation

Das für die PC-Version des UDK entwickelte Datenmodell bietet keinerlei Möglichkeiten, um auf die von den Metadaten im UDK beschriebenen (Basis-)Daten, also die Umweltdatenobjekte wie etwa Umweltberichte und Umweltdatenbanken on-line zuzugreifen. Um als Informationssuchender zu den eigentlichen Informationen zu gelangen, ist der angegebene Ansprechpartner zu kontaktieren. Für das UIS Baden-Württemberg wurden aber bereits verschiedene (Basis-)Datenbestände speziell für das WWW aufbereitet (vgl. etwa /9/, /22/, /25/ und uisextern: http://www.uis-extern.um.bwl.de/). Dieses Angebot von umweltrelevanten Informationen im WWW wird in Zukunft weiter wachsen. Daher ist es ist für den WWW-basierten UDK wünschenswert, on-line auf diese im WWW verfügbaren Datenbestände zugreifen zu können. WWW-UDK erlaubt dazu die Verknüpfung von bis zu 99 URLs je UDK-Objekt sowie zusätzlich Typinformationen, die Hinweise auf Format und Umfang der Informationen, auf die verwiesen wird, geben. Die Funktionalität des WWW-UDK als Auskunfts- und Nachweissystem wird somit um die Funktionalität eines Verweissystems ergänzt.

Eine wichtige Anforderung war, daß die Modellierung von Metadatenbeständen im UDK nicht modifiziert wird, damit die Kompatibilität zum weiterhin für die Erfassung eingesetzten PC-Programm gewährleistet ist. Ohne Kompatibilitätseinschränkungen möglich ist allerdings die Ergänzung des Datenmodells um neue Relation, die zusätzliche Informationen - wie etwa URLs als Verweise von den Metadaten auf die Basisdaten - zu den UDK-Objekten enthalten. Diese ergänzenden Relationen sind zwar nicht für das PC-Programm sichtbar, beeinflussen es aber auch nicht. Für den Durchgriff auf (Basis-)Daten wurden die beiden Zusatztabellen mime_type und obj_links angelegt.

1.4.4.1 Objektidentifikator (OID)

Um in Zusatztabellen abgelegten Daten mit UDK-Objekten (die in den Standardtabellen abgelegt sind) zu verknüpfen, war es bisher notwendig, in die Zusatztabellen die dekadische Motation (obj.uknot) und die Katalog-ID (obj.ucat_id) als Fremdschlüssel aufzunehmen. Bei Umstrukturierungen des Primärkataloges, wie sie etwa bei organisatorischen Umstrukturierungen von Behörden notwendig wird, ändert sich allerdings die dekadische Notation eines Objektes. Während das PC-UDK-Programm diese Änderungen in den Standardtabellen automatisch vollzieht, werden die Einträge in den - für das PC-UDK-Programm nicht sichtbaren - Zusatztabellen nicht angepaßt und müssen dort per Hand nachgeführt werden.

Die Einführung eines eindeutigen Objektidentifikators (OID), der unabhängig von der

dekadischen Notation ist, und der statt dekadischer Notation und Katalog-ID in den Zusatztabellen geführt wird, macht Änderungen in den Zusatztabellen bei Umstrukturierung der UDK-Objekte überflüssig, so daß kein zusätzlicher Administrationsaufwand anfällt.

Da der OID nur für UDK-Objekte notwendig ist, bei denen Einträge in Zusatztabellen vorliegen, ist es ausreichend, OIDs für diese Objekte zu erzeugen. Die OID wird für ein Objekt erzeugt, wenn ein erster Eintrag in eine Zusatztabelle erfolgt. Bestandteile der OID sind die Host-ID (8 Zeichen; eindeutige Kennung des Rechners verhindert daß bei dezentraler Erfassung identische OIDs für verschiedene UDK-Objekte vergeben werden), das Datum (14 Zeichen) und eine Zufallszahl (3 Zeichen). Das Attribut obj_internal der Relation obj wurde bisher nicht genutzt und bietet mit einer Feldlänge von 25 Zeichen ausreichend Platz für die Ablage des OID.

Der OID muß in den unterschiedlichen UDK-Datenbanken geführt werden, das heißt sowohl in der Datenbank für den öffentlichen bzw. verwaltungsinternen WWW-Server (Baden-Württemberg, Niedersachsen, Österreich: ORACLE- Datenbank, Sachsen-Anhalt: Informix-Datenbank), als auch in der für die Erfassung genutzten Datenbank (WATCOM-Datenbank auf PC). Da der OID erst erzeugt wird, wenn ein Eintrag in den Zusatztabellen stattfindet, wird der OID immer in der gleichen (ORACLE- bzw. Informix-) Datenbank erzeugt. Da die WATCOM-Datenbank nicht ständig on-line erreichbar ist, wird beim Erzeugen des OID eine Mail generiert und an den für die Pflege der WATCOM-Datenbank zuständigen Mitarbeiter geschickt. Diese Mail enthält die für ein Nachtragen des OID in der WATCOM-Datenbank notwendigen Informationen. Der OID wird von dem entsprechendem Mitarbeiter dann per Hand eingetragen.

1.4.4.2 Tabelle mime_type

Die Typisierung der Verweise von UDK-Objekten auf Basisdaten orientiert sich an den im MIME-Standard definierten MIME-Inhaltstypen (engl.: content types), die neuerdings auch Medientypen (engl.: media types) genannt werden. Die Abkürzung MIME steht für Multipurpose Internet Mail Extension und bezeichnet einen Standard, der ursprünglich für Internet-Mails entwickelt wurde. Dieser Standard wurde 1992 endgültig festgelegt und ist in /3/ veröffentlicht. Bei der Definition des Standards wurde viel Wert auf Offenheit, Kompatibilität und Erweiterbarkeit gelegt, auch eigene Erweiterungen werden erlaubt.

Ein Medientyp wird durch 2 Attribute angegeben: Typ und Untertyp. Derzeit sind 7 verschiedene Typen (mit jeweils einer ganzen Reihe von Untertpen) standardisiert und bei IANA[2] registriert. Beispiele sind:

- **text:** kenzeichnet unterschiedliche Darstellungen von Texten, z.B.\ text/plain für ASCII-Texte und text/html für HTML-Dokumente.
- **image:** wird für die Kennzeichnung von Grafik-Dokumenten verwendet, z.B. image/gif oder image/jpeg.
- **application:** wird verwendet, um Daten für Anwendungen zu kennzeichnen, z.B. application/msword für MS Word-Dokumente.

[2] IANA ist die Abkürzung für Internet Assigned Names Authority. Diese Organisation nimmt die Registrierung von Datentypen vor.

Das Attribut X- steht als „Vorsilbe“ für private Erweiterungen und zu Testzwecken zur Verfügung. Von dieser Möglichkeit kann z.B. mit application/x-uis-service für die Kennzeichnung von UIS-Diensten Gebrauch gemacht werden.

Da Medientypen prinzipiell die Beschreibung beliebiger Daten erlauben, werden sie mittlerweile nicht mehr ausschließlich für die Kennzeichnung der Bestandteile von Internet-Mails benutzt. So werden beispielsweise dynamisch erzeugte HTML-Seiten mittels Medientypen als text/html gekennzeichnet, damit der Web-Browser die Daten korrekt interpretiert. Die Flexibilität und Erweiterbarkeit, sowie die breite Unterstützung, die der MIME-Standard in der Internet-Gemeinde findet, waren wesentliche Gründe, um Umweltdaten, die von UDK-Objekten beschrieben werden, anhand standardkonformer Medientypen zu klassifizieren.

Da es unmöglich ist, im voraus alle im UIS vorkommenden Medientypen zu nennen, wird eine Dynamik der Menge der verwendeten Medientypen unterstützt. Der WWW-UDK-Administrator hat dazu die Möglichkeit, verwendetete Medientypen in die Tabelle mime_type aufzunehmen. Diese enthält im wesentlichen den Typ und den Untertyp im Klartext, eine Erläuterung sowie eine ID. Alle in dieser Tabelle aufgeführten Medientypen können zur Typisierung von Umweltdaten verwendet werden.

1.4.4.3 Tabelle obj_links

Bereits in älteren WWW-UDK-Versionen war es möglich, zu einem UDK-Objekt eine URL abzulegen, die auf (Basis-)Daten verwiesen hat. Mit zunehmend im WWW verfügbarer Datenmenge erwies sich diese Lösung aber als unzureichend, da durchaus mehrere WWW-Dokumente zu einem UDK-Objekt gehören können. Beispielsweise kann zu einer Meßstelle sowohl eine textuelle Beschreibung, ein Foto, eine Karte als auch ein Dienst, der den Zugriff auf die Meßwerte ermöglicht, vorhanden sein. Aus diesem Grunde wurde die Tabelle obj_links eingeführt, die damit die in älteren WWW-UDK-Versionen verwendetete Tabelle wu_ras ersetzt.

In der Tabelle obj_links können pro UDK-Objekt bis zu 99 URLs abgelegt werden. Die wichtigsten Attribute dieser Tabelle sind ein Objektidentifikator, der die Verbindung zum UDK-Objekt herstellt (s. Abschnitt 1.4.4), eine URL, ein Typidentifikator, der zur Typisierung des Dokuments, auf das verwiesen wird, dient (s. Abschnitt 1.4.4), ein Text, der bei der Darstellung des Links ausgegeben wird und optional das Datenvolumen des Dokuments, eine URL, die auf ein darzustellendes Icon verweist (bei großen Bildern z.B. sogenannte thumbnails, die verkleinert und auflösungsreduziert einen ersten Eindruck des zu erwartenden Dokuments vermitteln) sowie ein alternativer Text, der dargestellt wird, wenn der Web-Browser derart konfiguriert ist, daß keine Graphiken angezeigt werden sollen.

Die in der Tabelle obj_links zu einem UDK-Objekt gespeicherten Verweisinformationen werden in der tabellarischen Detaildarstellung (s. 1.7.2) des Objektes angezeigt, so daß der Benutzer per Mausklick von den Metadaten zu den Basisdaten gelangen kann. Die Typisierung erlaubt es dem Benutzer zu erkennen, auf welche Art von Informationen/Medien der Link verweist, ohne daß erst dem Link gefolgt werden muß.

1.5 Datenorganisation

Die UDK-Objekte sind in sogenannten Katalogen, einer baumförmigen Hierarchie, organisiert. Jedes UDK-Objekt ist im Primärkatalog enthalten, der im wesentlichen Zuständigkeiten und Verantwortlichkeiten für die UDK-Objekte festlegt. In Sekundärkatalogen können weitere Beziehungen - etwa Aggregationsbeziehungen - zwischen UDK-Objekten modelliert werden.

1.6 Retrieval von Umweltdaten und Adressen

Die Benutzungsschnittstelle wurde konform zum HTML 2.0 Standard realisiert, zusätzlich werden die von den marktgängigen WWW-Browsern unterstützten Erweiterungen Tabellen (entsprechend HTML 3.2) und Frames eingesetzt. Beide Erweiterungen helfen - bei gut konzeptioniertem Einsatz! - die Benutzungsschnittstelle wesentlich übersichtlicher zu gestalten. Um auch den Zugang mittels nicht frame-fähigen WWW-Browser zu ermöglichen, existiert eine Umschaltmöglichkeit auf eine frame-freie Version.

Es gibt zwei Arten von Suchen: Suche nach UDK-Objekten (also nach Umweltdaten) und Suche nach Adressen, wobei sich erstere noch weiter in allgemeine Suche, Detailsuche, Schlagwortsuche und kartographische Suche unterscheiden läßt. Bei der Adreßsuche und bei der allgemeinen Objektsuche wird ein einzelner Suchbegriff bzw. ein Teilwort auf eine ganze Reihe von Suchattributen abgebildet. Im Gegensatz dazu wird in der Detailobjektsuche und Schlagwortobjektsuche nach bestimmten Attributen bzw. Attributgruppen gezielt gesucht.

Allen Suchen gemeinsam ist der prinzipielle Ablauf: ein Formular dient zur Eingabe der Suchbegriffe, die gefundenen UDK-Objekte bzw. Adressen werden in einer Übersichtsliste präsentiert, aus dieser können die Objekte bzw. Adressen ausgewählt werden, über die weitere Informationen gewünscht sind, die dann in einer detaillierten Darstellung ausgegeben werden. Diese Vorgehensweise dient der Minimierung des zu transferierenden Datenvolumens, der insbesondere bei Weitverkehrsnetzen mit einer geringen Bandbreite eine große Bedeutung zukommt. Zusätzlich erhöht sie die Übersichtlichkeit.

Für jede Suchanfrage ist einstellbar, ob nach vollständigen Begriffen oder Teilworten gesucht werden soll, und ob dabei die Groß-/Kleinschreibung beachtet oder ignoriert werden soll. Für erfahrene Benutzer besteht darüberhinaus die Möglichkeit, auch innerhalb der Suchstrings die gewohnten Platzhalter (wildcards) zu verwenden. Das Fragezeichen (?) ist dabei Platzhalter für ein beliebiges einzelnes Zeichen. Als Platzhalter für beliebig viele Zeichen (keines oder mehrere) wird der Stern (*) verwendet. Wenig erfahrene und gelegentliche Benuzter werden durch ein Hypertext-Hilfesystem in die Bedienung des WWW-UDK eingeführt. Hier werden kurz Entwicklung sowie Ziele des UDK dargestellt und die verschiedenen Suchen ausführlich erläutert.

Um bei allen Suchen nach Umweltdaten auch den fachfremden Benutzer durch ein geeignetes Zugangsvokabular zu unterstützen, kann bei diesen Suchen der Umweltthesaurus des Umweltbundesamtes benutzt werden, s. Abschnitt 1.7.5.

Die Benutzungsoberfläche des WWW-UDK ist zweisprachig (deutsch/englisch) verfügbar. Dabei ist allerdings zu beachten, daß die im UDK gespeicherten Daten nur in deutsch

vorliegen und nicht automatisch übersetzt werden. Auch in den auf englisch beschrifteten Formularen müssen Suchbegriffe auf deutsch eingegeben werden.

Die verschiedenen Suchmodi werden in den folgenden Abschnitten kurz erläutert.

1.7 Umweltdaten

1.7.1 Suchmöglichkeiten

Bei den Suchen in der Menge der UDK-Objekte nach relevanten Umweltdaten gibt es die Möglichkeit, die Suche auf einzelne Objektklassen zu beschränken, oder aber, bei Beibehaltung der Voreinstellung, in allen Objektklassen gleichzeitig suchen zu lassen. Außerdem kann zwischen verschiedenen Katalogen gewählt werden, in denen nach Objekten gesucht werden soll. Voreingestellt ist hierbei die Suche in allen Primärkatalogen.

Während bei der allgemeinen Suche ein einzelner Suchbegriff bzw. ein Teilwort auf eine ganze Reihe von Suchattributen abgebildet wird, wird in der Detailobjektsuche und der Schlagwortobjektsuche nach bestimmten Attributen bzw. Attributgruppen gezielt gesucht und es können mehrere Suchbegriffe eingegeben werden, die mit „und“ oder „oder“ verknüpft werden können. In der Detailobjektsuche besteht darüberhinaus die Möglichkeit, den Zeitbezug der gesuchten UDK-Objekte durch Angabe eines Zeitraums einzuschränken. Während eine Eingabeunterstützung bei der Eingabe eines Zeitbezugs bisher nicht statt fand, da in früheren Versionen des UDK die Zeit in beliebigem Format eingetragen werden konnte, ist seit UDK 3.0 eine Eingabeunterstützung möglich und sehr sinnvoll. Es finden Plausibilitätsüberprüfungen statt.

Die kartographische Suche erlaubt es, in einer Karte eine Bounding-Box per Mausklick zu spezifizieren (vgl. Konsolidierung der servereigenen Kartendienste und Überführung in die GLOBUS-Betriebsversion im Kapitel GIS-Tools in diesem Bericht). Anschließend werden im UDK alle UDK-Objekte gesucht, deren Raumbezug in dieser Bounding-Box enthalten ist. Dazu werden sowohl Raumbezugsdaten in Form von Verwaltungseinheiten wie etwa angegebenen Gemeinden und Kreisen als auch eventuell numerisch gegebene Gauß-Krüger-Koordinaten berücksichtigt.

1.7.2 Detaillierte Darstellung

Bei der detaillierten Darstellung von UDK-Objekten werden sämtliche Attribute der gefundenen UDK-Objekte tabellarisch ausgegeben. Um die Übersichtlichkeit der Objektdetaildarstellung zu erhöhen, d.h. nicht zu viele „Leerzeilen“ in der Tabelle auszugeben, und um die Vergleichbarkeit der Darstellung zweier Objekte weiterhin zu gewährleisten, wurde die Tabelle in die folgenden fünf Teile zerlegt: Allgemeine Information, Zusatzinformation, Fachbezug, Raumbezug, Zeitbezug.

Die einzelnen Teile werden nur angezeigt, wenn mindestens eines der darin auftretenden Attribute einen Wert enthält. Der Teil "Allgemeine Information" wird immer ausgegeben, da jedes ausgewählte Objekt mindestens einen Objektnamen und eine dekadische Notation besitzt. Enthält ein Attribut als Wert nur eine Abkürzung (z.B. obj_class hat die Werte

"AOB", "BO", "ED", "KAR", ...), so wird in der Tabelle trotzdem der vollständige Text, der in einer Textdatei abgelegt ist (und deshalb auch mehrsprachig zur Verfügung steht), ausgegeben (also für Objektklasse: "Allgemeine Objekte", "Basisobjekte", "Empirische Daten", "Karten", ...). Welche Attribute in den einzelnen Teilen ausgegeben werden, ist abhängig von der Objektklasse, zu der das Objekt gehört.

Es besteht außerdem die Möglichkeit, die Adressen der datenhaltenden sowie der auskunftgebenden Stelle zu erhalten (und somit zur vollständigen Ausgabe der Adresse zu gelangen, vgl. Abschnitt 1.8.2), indem die entsprechenden Links bei Adreßkennzeichen bzw. Datenauskunft angeklickt werden.

Zudem wird für jedes Objekt, soweit vorhanden, eine Liste der Vorgänger und der Nachfolger im Primärkatalog ausgegeben, aus der eine Auswahl möglich ist. Wird die Auswahl zu einem Objekt bestätigt, so erfolgt eine erneute vollständige Ausgabe der ausgewählten Vorgänger- und Nachfolgerobjekte. Dies ermöglicht eine Navigation entlang der Struktur des Primärkatalogs.

1.7.3 Verbindung von Metainformationen und Informationen

Der UDK ist ein Metainformationssystem, das nur Metadaten, nicht aber die eigentlichen Umweltdaten verwaltet. Wichtige Funktion der Metadaten ist die Unterstützung des Recherche des Benutzers nach Umweldaten. Das Benutzerinteresse ist in der Regel nicht nach Einsicht der Metadaten erschöpft, sondern er möchte auch auf die von den Metadaten beschriebenen Datenbestände zugreifen. Um zumindest die bereits im WWW verfügbaren Umweltdaten von den Metadaten auf Mausklick hin erreichbar zu machen, wurde die in Abschnitt 1.4.4 beschriebene Tabelle obj_links eingeführt.

Falls zu den angezeigten UDK-Objekten Verweise auf (Basis-)Daten wie etwa Umweltberichte oder WWW-basierte Datenbankfrontends existieren, werden diese einschließlich eines (optionalen) Icons sowie eines Textes und Typ- sowie Datenvolumeninformationen in der detaillierten Objektausgabe angezeigt. Die Typisierung sowie das Anzeigen des Datenvolumens erleichtern es dem Benutzer, bereits vor Verfolgung des Links zu erkennen, ob es für ihn interessant ist, diese Daten in den Browser zu laden. Das Konzept ist derart allgemein, daß prinzipiell auf beliebige, weltweit verteilte Informationen in den unterschiedlichsten Datenformaten verwiesen werden kann.

Abbildung 1: Detaillierte Ausgabe des UDK-Objektes „Luftmeßstelle Baden-Baden“

Ein Beispiel der detaillierten Darstellung des UDK-Objektes „Luftmeßstelle Baden-Baden“ zeigt Abbildung 1. Es sind zwei Verweise auf Daten eingetragen: zu einer Tabelle mit den aktuellen Meßwerten dieser Station sowie zu einer Übersichtskarte mit allen VIKOLUM-Luftmeßstationen.

1.7.4 Visualisierung des Raumbezugs von UDK-Objekten

Es ist nicht nur möglich, in einer Karte eine Bounding-Box zu spezifzieren, um innerhalb dieser nach UDK-Objekten (und somit letzendlich Umweltdaten) zu suchen, sondern auch, den Raumbezug von UDK-Objekten in einer dynamisch generierten Karte zu visualisieren (vgl. Konsolidierung der servereigenen Kartendienste und Überführung in die GLOBUS-Betriebsversion im Kapitel GIS-Tools in diesem Bericht). Diese Visualisierung erleichtert die geographische Einordnung der Umweltdaten enorm.

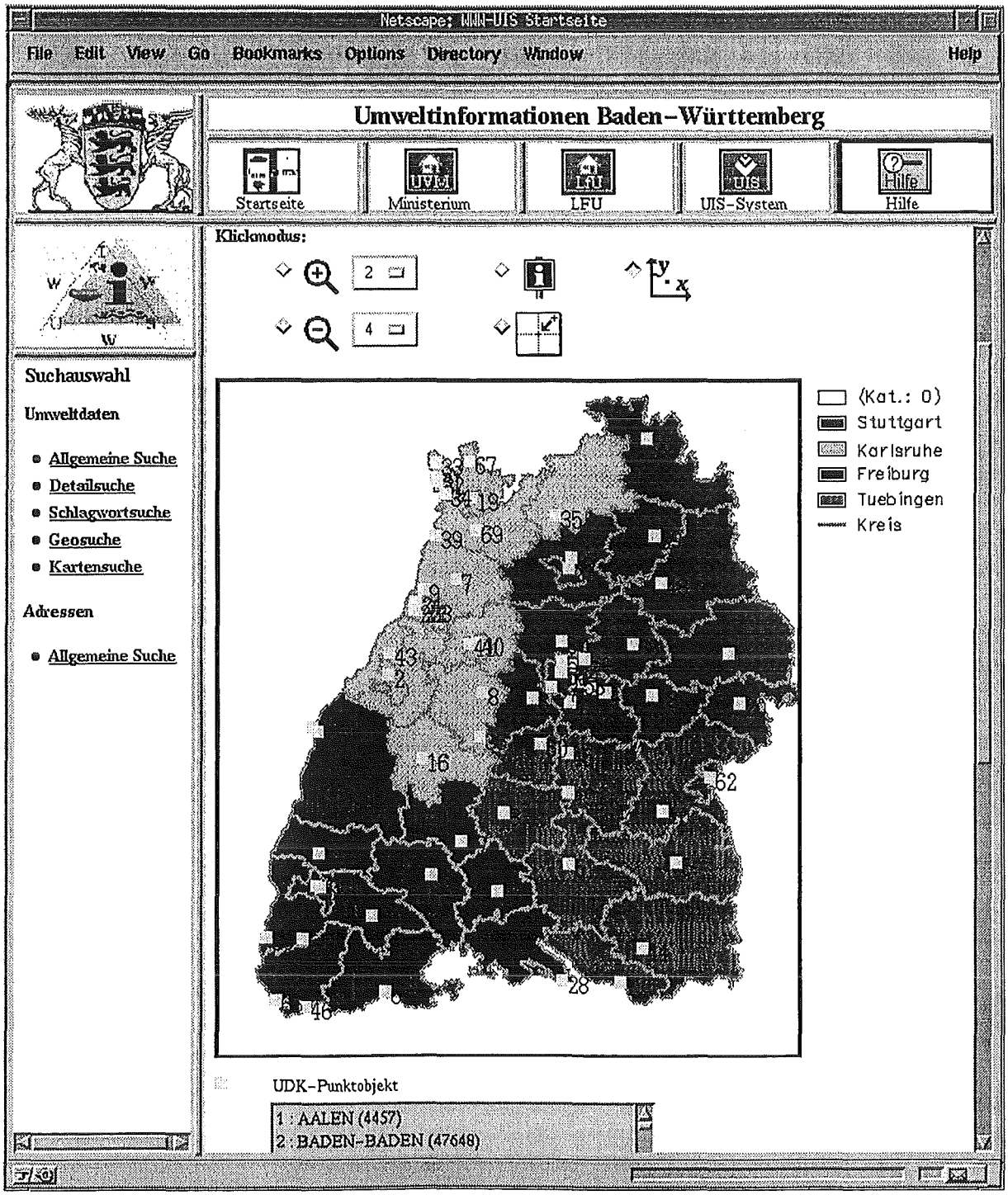


Abbildung 2: Mehrere UDK-Objekte gleichzeitig vom GIS dargestellt.

Von der detaillierten Objektdarstellung kann in die Kartendarstellung verzweigt werden, wenn zu einem UDK-Objekt Gauß-Krüger-Koordinaten in Form eines Punktes oder einer Bounding-Box gegeben sind. Existieren auf der gleichen Ebene des Primärkatalogs „verwandte" UDK-Objekte, können diese UDK-Objekte auch gleichzeitig auf einer Karte dargestellt werden. Ein Beispiel sind etwa die verschiedenen Meßstationen eines Meßnetzes die gemeinsam in einer Karte dargestellt werden, der Benutzer kann also auf einen Blick Lage und Verteilung der Meßstationen erkennen, s. Abbildung 2.

1.7.5 Thesuraus

Ein Thesaurus dient zum einen der fachlich-inhaltlichen Erschließung von Daten und zum anderen, bei On-line-Datenbanken wie dem UDK, dem Nutzer zur Vorbereitung und Durchführung der Recherche.

Zur Verschlagwortung von und zur Suche nach UDK-Objekten kann der Umweltthesaurus des Umweltbundesamtes eingesetzt werden /1/. Hierbei handelt es sich um einen sogenannten poly-hierachischen Thesaurus. Das bedeutet, daß ein einzelner Begriff im Thesaurus mehrere Vorgänger und mehrere Nachfolger haben kann. Ferner werden dort die Begriffe erläutert sowie Synonyme und verwandte Begriffe bereitgestellt.

Für eine Suche im Thesaurus, die ausgehend von den Suchen nach UDK-Objekten erreicht werden kann, gibt man zunächst einen Suchbegriff ein. Die verschiedenen Suchmodi und Konfigurationen stehen, wie bei allen anderen Suchen, zur Verfügung. Das Ergebnis der Suche im Thesaurus wird in Listenform ausgegeben. Innerhalb dieser Liste besteht die Möglichkeit, einen Term auszuwählen und sich diesen vollständig ausgeben zu lassen.

Bei der vollständigen Ausgabe werden der gewählte Deskriptor sowie die übergeordneten bzw. untergeordneten Begriffe ausgegeben. Ferner werden Synonyme zum Deskriptor sowie eine Erläuterung ausgegeben, sofern diese vorhanden sind. Ausgehend von dieser Ergebnismaske kann einer der vorhandenen übergeordneten oder untergeordneten Deskriptoren für eine erneute Detaildarstellung ausgewählt werden. Ferner besteht die Möglichkeit aus der Begriffshierarchie, den verwandten Begriffen oder den Synonymen - wie bereits von der Liste der Deskriptoren - einen Term auszuwählen und in die vorausgehende Objektsuchmaske zu übernehmen.

1.8 Adressen

Außer den eigentlichen Metadaten über Umweltdaten enthält der UDK auch Adressen von Personen und Institutionen, die für die Verwaltung und Auskunft über die Umweltdaten zuständig sind. Um nach bestimmten Personen oder Institutionen unabhängig von UDK-Objekten suchen zu können, wurde eine eigene Suche nach Adressen realisiert.

1.8.1 Suchmöglichkeiten

Bei der Adreßsuche kann ein einzelner Suchbegiff eingegeben werden, der entsprechend dem gewählten Suchmodus auf die folgenden Attribute der Adreßbeschreibung im UDK abgebildet wird: Name der Institution, Abteilung, Unterabteilung, Referat, Name der Abteilung,

Adreßkennzeichen der Stelle, Nachname des Mitarbeiters, Suchbegriffe zur UDK-Adresse.

Die gefundenen Adressen werden in einer Liste präsentiert. Wird kein Suchbegriff eingegeben, so werden alle Adressen in der Liste aufgeführt. Die interessierenden Adressen können nun durch Anklicken ausgewählt werden und vollständig angezeigt werden.

1.8.2 Detaillierte Darstellung

In der detaillierten Adressdarstellung werden alle Informationen, die zu den ausgewählten Adressen vorliegen, in einer kompakten und übersichtlichen Tabelle dargestelt.

1.9 Zusammenfassung und Ausblick

WWW-UDK ist als Umwelt-Metainformationsystem das primäre Rechercheinstrument im WWW-basierten UIS Baden-Württemberg. Jedwede Metainformation des UIS, die adäquat im Datenmodell des UDK abgebildet werden kann, soll zukünftig in den UDK aufgenommen werden und über die Funktionalitäten von WWW-UDK verfügbar sein.

Mit der Version WWW-UDK 3.0.2 existiert eine ausgereifte Betriebsversion, die mit einer komfortablen Benutzungsoberfläche und unterschiedlich komplexen Suchmöglichkeiten ein Recherche- und Retrievalwerkzeug für unterschiedlichste an Umweltinformationen interessierte Benutzergruppen - vom Umweltgeneralisten über den Sachbearbeiter bis hin zur interessierten Öffentlichkeit - darstellt. Die Metainformationen werden in übersichtlichen Tabellen präsentiert und erlauben einen Zugriff auf die eigentlichen Umweltinformationen mittels einfachem Mausklick.

In Verbindung mit dem vom IPF entwickelten Geoinformationssystem kann zum einen der Raumbezug von UDK-Objekten verannschaulicht werden, zum anderen bietet sich ein komfortables Interface für die raumbezogene Suche.

Das entwickelte System WWW-UDK wird zum einen im interen Verwaltungsnetz (LVN), dem sog. Verwaltungs-Intranet, zum anderen auch auf dem öffentlich zugänglichen Internet eingesetzt. Weitere Installationen außerhalb Baden-Württembergs sind derzeit im Umweltministerium Niedersachsen, im Ministeriums für Umwelt, Naturschutz und Raumordnung des Landes Sachsen-Anhalt (derzeit noch nicht öffentlich zugänglich) sowie im österreichischen Bundesministeriums für Umwelt, Jugend und Familie, Wien, in Betrieb. Alle öffentlich zugänglichen Installationen können über diese WWW Seite (http://www.fzi.de/divisions/dbs/applAreas/wwwudk.html) erreicht werden.

Die derzeitige WWW-UDK-Version bietet zwar sowohl einen deutsche als auch eine englische Benutzungsoberfläche, allerdings sind sie Suchbegriffe immer auf deutsch einzugeben. Der Einsatz eines multilinguale Thesaurus würde eine mehrsparchige Suche unterstützen /16/, erfordert allerdings kleine Änderungen bei der Verbindung von UDK-Objekten mit Thesaurus-Begriffen. Desweiteren müßte das Thesaurus-Datenmodell, das derzeit weder einen mehrsprachigen Thesaurus, noch die Verwendung mehrerer Thesauri zuläßt, entsprechend geändert werden. Eventuell kann statt des Datenmodells des UDK-Thesaurus das den Anforderungen genügende Datenmodell des GEMET-Thesaurus /27/

verwendet werden.

Um die Interaktivität der Benutzungsoberfläche zu erhöhen, bietet sich in ausgewählten Modulen der Einsatz von Java an. So könnte etwa eine Navigation entlang der hierarchischen Baumstruktur der UDK-Objekte mittels Java realisiert werden, die eine ähnliche Oberfläche wie die von PC-Oberflächen bekannten Dateisystembrowser bietet.

Eine Fragebogenaktion, die über alle WWW-UDK-Installationen erreichbar ist, soll den WWW-UDK-Benutzern die Möglichkeit der Einflußnahme auf zukünftige Entwicklungen der Software geben. Anhand des Rücklaufes sollen Schwerpunkte weiterer Benutzeranforderungen analysiert und identifiziert werden.

2. Automatisierung der Metadatenfortschreibung

2.1 Motivation und Zielsetzung

Nachdem sich in den letzten Jahren Metainformationssysteme sowie entsprechende Standards für Datenmodelle und Anwendungen etabliert haben, wird nun verstärkt die möglichst vollständige Erfassung der Metainformationen voran getrieben. Mit zunehmenden Metadatenvolumen wächst aber auch die Problematik der Konsistenshaltung von Informationen und den in einem separaten Metainformationssystem verwalteten Metainformationen. Eine wesentliche Anforderung an ein Metainformationssystem ist die Aktualität der von diesem System verwalteten Daten.

Da Datenbestände durch hinzufügen, löschen und ändern von Datensätzen ständig aktualisiert werden, müssen diese Änderungen auch in die entsprechenden Metadatenbestände übernommen werden. Eine manuelle Fortschreibung der Metadaten ist jedoch oft langwierig und fehleranfällig. Deshalb ist eine automatische Aktualisierung des Metadatenbestandes anzustreben, da so eine schnelle und fehlerfreie Fortschreibung erfolgen kann. Denn nur eine Dokumentation der Datenbestände, die auf dem aktuellen Stand ist, ermöglicht dem Nutzer eine umfassende Recherchierbarkeit und kann verhindern, daß unnötige Kosten für nochmalige Datenerhebungen und -verarbeitungen ausgegeben werden.

Während die Metadatenersterfassung in einem Metainformationsystem wie dem Umwelt-Datenkatalog (UDK) vorwiegend ein intellektueller Prozess ist, der sich nicht automatisieren läßt, liegt es insbesondere bei einem umfassenden Umweltinformationssystem, wie es das UIS Baden-Württemberg darstellt, nahe, zu untersuchen, inwieweit die Fortschreibung von in einem separaten Metainformationssystem gehaltenen Metadaten automatisiert werden kann. Zielsetzung ist die Konzeption sowie prototypische (d. h. Realisierung an ausgewählten Beispielen) Realisierung einer automatisierten Metadatenfortschreibung der im UDK repräsentierten Datenbestände des UIS Baden-Württembergs.

2.2 Grundlagen

2.2.1 Entstehung von Metadaten

Aus der Umwelt werden Daten durch Beobachtung, Messung u. Bewertung abgeleitet. Genauer gesagt, bedeutet dies, daß ein Datenbestand zu einem bestimmten Zweck erstellt wird und nur die Diskurs- oder Miniwelt zu einem bestimmten Zeitpunkt oder Zeitraum repräsentiert. Aus diesem Datenbestand wiederum werden Metadaten gewonnen, d.h. sie werden auf analoge Weise aus den Daten abgeleitet, um diese Datensammlung recherchierbar zu machen. Das Ziel der Metadaten ist es, dem Datensuchenden die zur Datensuche nötigen Daten zu liefern und ihn bei der Entscheidung, ob die gefundenen Daten für ihn nützlich sind, zu unterstützen. Metadaten dienen somit als Hilfsmittel zur Suche, Validierung und Analyse von Daten.

Ein Problem ist, daß Metadaten den aktuellsten Stand des Datenbestandes widerspiegeln müssen, um dem Datensuchenden alle für ihn eventuell nützlichen Datenbestände aufzuzeigen. Dies setzt voraus, daß die Metadaten entweder regelmäßig per Hand aktualisiert werden oder besser, daß sie automatisch fortgeschrieben werden. Der nächste Abschnitte befaßt sich deshalb genauer mit der Erfassung und Aktualisierung der Metadaten, sowie mit dem Metadaten-Erfasser.

2.2.2 Erfassung von Metadaten

Bei der Gewinnung von Metadaten aus Daten unterscheidet man zwischen initialer Metadatenerfassung und Aktualisierung.

2.2.2.1 Initiale Metadatenerfassung

Unter initialer Metadatenerfassung versteht man die erstmalige Beschreibung eines Datenbestands durch Metadaten, d.h. die Aufnahme des Datenbestandes in einen Metadatenkatalog. Dies ist ein weitgehend intellektueller Prozess der sich nicht vollständig automatisieren läßt. Es ist jedoch durchaus möglich auch hier gewisse Metadaten automatisch zu erfassen; ein Beispiel ist die automatische Verschlagwortung von Berichten.

Die initiale Metadatenerfassung kann auf folgende Arten erfolgen:

- **Manuell**: menschlicher Bediener gibt die Metadaten ein.
- **Automatisiert**: Metadaten werden maschinell aus den Daten abgeleitet.
- **Teilautomatisch**: Aufteilung in einen manuellen und in einen automatisierten Teil, d.h. die Metadaten werden entweder zuerst maschinell erfassen und dann manuell vervollständigen oder aber zuerst manuelle Erfassung bestimmter Elemente, aufgrund derer die maschinelle Vervollständigung erfolgt.

Metadaten können jedoch nicht allein aus den Daten abgeleitet werden, da auch das Umfeld mit einbezogen werden muß, d.h. Datensammlungen dienen als Grundlage der Metadaten, aber nicht als alleinige Quelle.

2.2.2.2 Aktualisierung / Fortschreibung

Ein Metadatenbestand beschreibt einen Datenbestand zu einem bestimmten Zeitpunkt. Ändert sich der Datenbestand, so ist auch eine Anpassung des Metadatenbestandes notwendig. Dabei hat man bezüglich des Aktualisierungszeitpunktes und der Vorgehensweise dieselben Auswahlmöglichkeiten wie bei der initialen Erfassung. Welche Metadaten wie erfaßt werden, bedarf jeweils einer genauen Überlegung. Eine automatische Fortschreibung ist jedoch immer vorzuziehen, da sie weniger fehleranfällig und schneller ist als die manuelle Methode.

Ein weiteres Problem tritt auf, wenn Daten und Metadaten in einem verteilten System abgelegt sind. Werden die Daten an einem anderen Ort gehalten als die Metadaten, so hat dies zur Folge, daß der aktuelle Stand der Metadaten nicht zu jedem Zeitpunkt mit dem aktuellen Stand der Daten übereinstimmen kann, da erkannte Datenänderungen noch propagiert werden müssen, was einen gewissen Zeitaufwand erfordert.

2.2.2.3 Metadaten-Erfasser

Die Erfassung von Metadaten kann manuell, teilautomatisch oder automatisch erfolgen.

Die manuelle Erfassung von Metadaten kann durch den (Original-)Datenerfasser oder einen Bibliothekar, d.h. einen Fachmann, der zwar nicht mit der Datenerfassung selbst zu tun hatte, der aber die Daten und deren Umfeld kennt und der insbesondere für die Erfassung von Metadaten zuständig ist, erfolgen. Am vorteilhaftesten ist es natürlich, wenn der Datenerfasser selbst auch die Metadaten erzeugt, da er das größte Wissen (auch über Besonderheiten) der Daten hat. Er kennt sich jedoch im Gegensatz zum Bibliothekar nicht so sehr mit dem Datenkatalog und den möglichen Metadaten aus.

Die automatische Erfassung von Metadaten kann sich nur auf Metadaten beziehen, die in den (Original-)Daten enthalten oder aus ihnen ableitbar oder berechenbar sind. Dies hat zur Folge, daß Kommentare, Besonderheiten und das Umfeld der Daten in einer solchen Beschreibung fehlen. Um Metadaten überhaupt automatisch Erzeugen zu können ist es jedoch zuerst notwendig Regeln aufzustellen, nach denen die Metadaten dann aus den (Original-)Daten abgeleitet werden.

Bei der teilautomatisierte Erfassung von Metadaten werden nicht alle Metadaten auf einmal erfaßt. Sie teilt sich in einen manuellen und einen automatischen Teil der Metadaten-Erfassung die nacheinander ausgeführt werden, d.h. die Metadaten werden entweder zuerst maschinell erfaßt und dann manuell vervollständigt oder es werden zuerst bestimmter Elemente manuelle erfaßt, aufgrund derer die maschinelle Vervollständigung erfolgt.

Ein Beipiel für die vollautomatische Erfassung und Aktualisierung von Metadaten sind WWW-Suchmaschinen. WWW-Suchmaschinen wie zum Beispiel Lycos, Infoseek, Magellan und Yahoo bieten dem Nutzer die Möglichkeit, die für ihn relevante Daten im Internet zu finden. Zu diesem Zweck greifen sie auf einen Metadatenbestand zurück, der ständig durch Navigieren durch das Internet automatisch aktualisiert wird. Wesentlicher Nachteil von derartigen Suchmaschinen ist allerdings, daß nur die Inhalte statischer HTML-Dokumente erfaßt werden. Daten in Datenbanken oder weiteren Anwendungen, die über das WWW-erreicht werden können, bleiben unbeachtet.

2.3 Anforderungen und Randbedingungen

Eine (teil-)automatische Konsistenzhaltung der Datenbestände und des Metadatenbestandes erfordert es, Änderungsereignisse (Einfügen, Modifizieren, Löschen) in den Datenbeständen zu erkennen und entsprechend zu reagieren, d.h. die Metadaten erforderlichenfalls den veränderten Datenbeständen anzupassen und den für den Metadatenbestand verantwortlichen Mitarbeiter zu informieren.

In einer ersten Phase werden vorerst die von ORACLE-Datenbankverwaltungssytemen verwaltete Datenbestände betrachtet. Das zu entwickelnde System ist so zu konzeptionieren, daß in Folgeversionen sowohl weitere Datenbestände (wie etwa Geoinformationssysteme) berücksichtigt werden können, als auch eine Integration in eine CORBA-Umgebung möglich ist. Dies ermöglicht das zukünftige Einbeziehen heterogener, verteilter Datenquellen.

In den relevanten Datenquellen müssen alle relevanten Ereignisse erkannt werden (Vollständigkeit); einmal erkannte Ereignisse dürfen nicht verloren gehen (Persistenz). Die Ereignisse sollen chronologisch abgearbeitet werden (Chronologie). Außerdem soll es möglich sein zusammengesetzte Ereignisse, also Ereignisse, die aus mehreren Elementarereignissen bestehen, zu verarbeiten (komplexe Ereignisse).

2.4 Konzeption

Die Metadatenfortschreibung umfaßt die drei Komponenten Ereigniserkennung, Ereignisverwaltung und Ereignisverarbeitung, deren zusammenwirken in Abbildung 3 dargestellt wird.

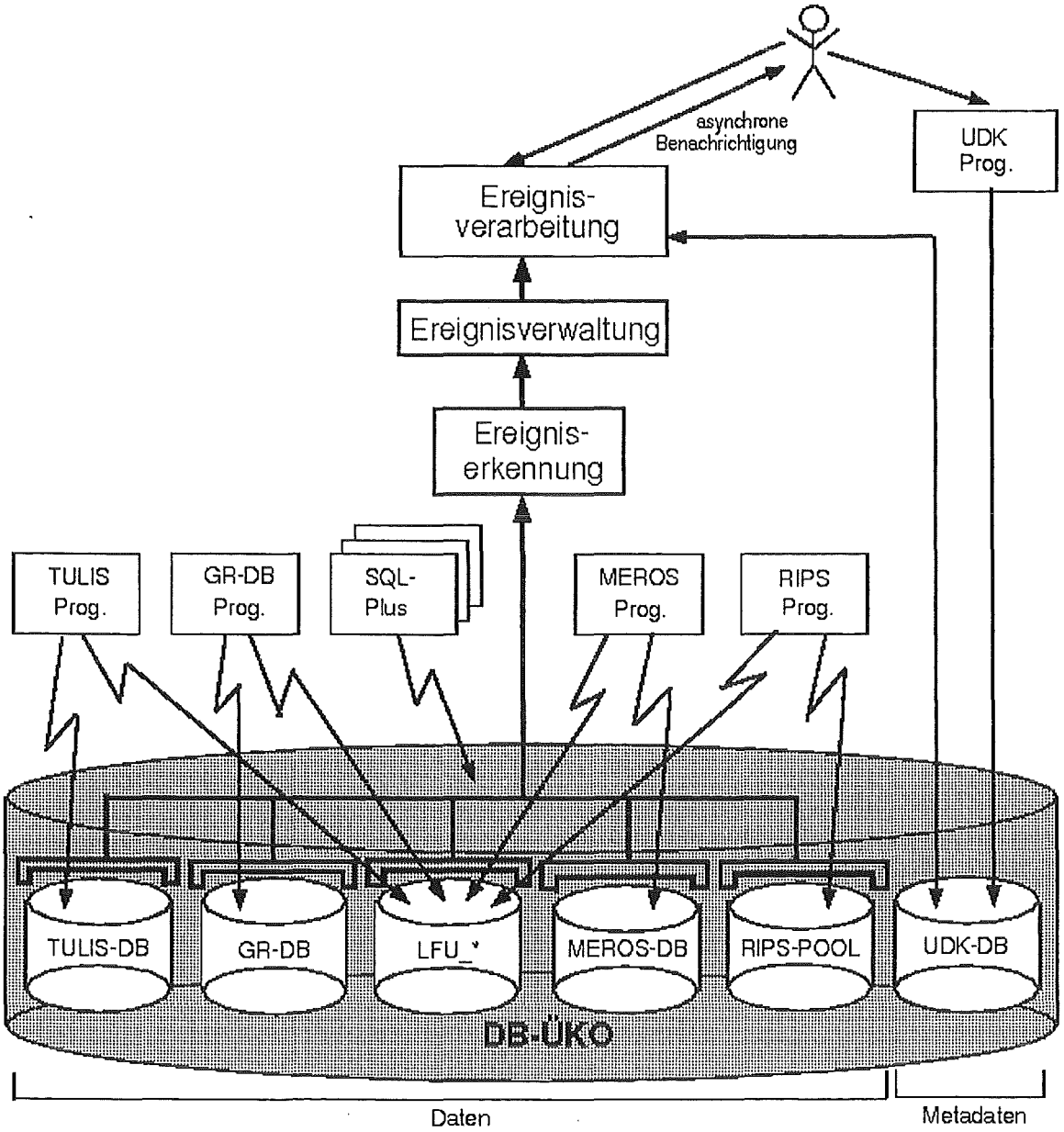


Abbildung 3: Ereigniserkennung, -verwaltung und -fortschreibung

Für jede Komponente stehen verschiedene Realisierungsalternativen zur Verfügung, von denen in den folgenden Abschnitten die jeweils gewählte Alternative vorgestellt wird.

2.4.1 Ereigniserkennung

Unter der gegebenen Voraussetzung, daß es sich bei den Datenquellen (zunächst) nur um ORACLE-Datenbanken handelt - insbesondere werden etwa die Datenbestände der Datenbank der übergreifenden Komponente (DBÜKO) und des Technosphäre- und Luft-Informationssystems (TULIS) von ORACLE verwaltet - werden zur Ereigniserkennung ORACLE-Datenbank-Trigger verwendet.

Datenbanktrigger werden zu den allgemeinsten Konsistenzsicherungsmechanismen gezählt.

Mit ihnen kann festgelegt werden, daß bei Eintreten bestimmter Ereignisse Folgeaktionen durchgeführt werden sollen, die zur Konsistenzsicherung erforderlich sind und bieten damit die Möglichkeit zur Formulierung von Nachbedingungen. Datenbanktrigger bestehen aus den folgenden drei Teilen:

- auslösendem Ereignis (Einfügen, Löschen oder Ändern eines Tupels);
- einschränkende Bedingung;
- Aktion (DB-Prozedur).

Datenbanktrigger, im folgenden kurz als Trigger bezeichnet, treten bei Veränderungen der Datenbank in Aktion und erhalten semantische Beziehungen aufrecht, die dem Benutzer eventuell verborgen bleiben. Sie sind vor allem notwendig, wenn Veränderungsoperationen andere Datenstrukturen beeinflussen oder wenn für unterschiedliche Datenstrukturausprägungen alternative Reaktionen erfolgen sollen.

Bei ORACLE-Triggern handelt es sich um Prozeduren, die implizit ausgeführt werden, wenn auf die Tabelle eine INSERT, UPDATE oder DELETE Anweisung ausgeführt wird /2/. Sie können SQL- oder PL/SQL-Anweisungen enthalten oder gespeicherte Prozeduren aufrufen und werden getrennt von den entsprechenden Tabellen gespeichert.

Veränderungen in ORACLE-Datenbeständen, deren Beschreibung im Metadatenbestand des UDK enthalten ist, können durch entsprechende Trigger erkannt und an die Ereignisverwaltung weitergereicht werden.

2.4.2 Ereignisverwaltung

Die Ereignisverwaltung nimmt alle Ereignisse entgegen, die sich im Gesamtsystem ereignen und gibt sie zur Verarbeitung weiter. Sie wurde entwickelt, um folgende fünf Ziele zu verwirklichen:

- Durch die Ereignisverwaltung soll eine Entkopplung zwischen Ereignisentdecker und regelverarbeitendem System stattfinden, so daß der Ereignisentdecker nach Übergabe des Ereignisses sich wieder seine Aufgabe widmen kann und nicht warten muß, bis die regelverarbeitende Einheit bereit ist, das Ereignis in Empfang zu nehmen.
- Die Ereignisverwaltung soll grundsätzlich einen FIFO (first in, first out) hinsichtlich der Weitergabe der Ereignisse realisieren. Wenn aber die Ereignisse nicht in der zeitlichen Reihenfolge des Eintritts zu der Ereignisverwaltung gelangen, dann soll diese die Ereignisse, wenn möglich, umsortieren und in der sortierten Weise weitergeben. Somit soll die Ereignisverwaltung Kommunikations- oder Komponentenprobleme ausgleichen.
- Alle Ereignisse sollen persistent in der Ereignisverwaltung abgelegt werden, damit bei Ausfall der regelverarbeitenden Einheit die Ereignisse nicht verloren gehen und ein Wiederanlauf möglich ist, ohne daß entstandene Ereignisse ignoriert werden.
- Die Persistenz der Ereignisse in der Ereignisverwaltung ist desweitern wichtig, um in Bedingungen auf alte Ereignisse zurückgreifen zu können.

Werden innerhalb der regelverarbeitenden Komponente andere Ereignisverbrauchstrategien als die chronologische realisiert, dann benötigt man eine Ereignishistorie. Das Ereignisverwaltungssystem mit den persistenten Ereignissen kann dann diese Aufgabe

übernehmen.

Die Ereignisbeschreibung enthält folgende Angaben: Zeit und Datum des Eintritts (Erkennen) des Ereignisses, Typ der Operation (z.B.: löschen, ändern oder einfügen eines Datensatzes), Name der Datenbank, Name der Tabelle, ID des Datensatzes, Attributwerte, -namen und -typen.

ORACLE-Tabellen eignen sich für die Ereignisverwaltung, da sie eine "unbegrenzte" Aufnahmefähigkeit bezitzen, so daß Ereignisse nicht verloren gehen können. Zur Beschreibung eines Ereignisse benötigt man zwei Tabellen. In der einen Tabelle wird das Ereignis an sich näher beschrieben, während mit Hilfe der anderen Tabelle alle Auswirkungen eines UPDATE-Ereignisses genau festgehalten werden, d.h. es wird vermerkt, welches Attribut welchen neuen Wert annimmt. Um den Zusammenhang zwischen den beiden Tabellen herzustellen, ist ein eindeutiger Identifikator des Ereignisses notwendig. Dabei kann es sich zum Beispiel um einen globalen Zeitstempel handeln. Voraussetzung in einer verteilten Umgebung ist dann, daß die Systemuhren gut synchronisiert sind.

2.4.3 Ereigniverarbeitung

Um die komplexe Verarbeitung erkannter Ereignisse und schließlich deren Auswirkungen auf den Metadatenbestand des UDK zu realisieren, wird ein regelbasiertes System eingesetzt.

Fakten und Regeln bilden die Grundlage regelbasierter Programmierung. Die Fakten beschreiben den aktuellen Zustand eines regelbasierten Systems (statischer Teil) und die Regeln legen fest, was in einer bestimmten Situation getan werden soll (dynamischer Teil).

Bei den Regeln handelt es sich um CA-Regeln, die aus einer Bedingung (Condition) und einer Aktion (Action) bestehen. Die Bedingung bezieht sich auf den aktuellen Systemzustand und damit auf die Fakten, während die Aktion den Systemzustand ändern kann oder andere Aktivitäten, wie z.B. die Ausgabe einer Zeile auf dem Bildschirm oder die Benachrichtigung eines Benutzers, ausführen.

Startet man ein regelbasiertes System, so werden zunächst alle ausführbaren Regeln bestimmt, d.h. alle Regeln deren Bedingung erfüllt ist. Aus diesen Regeln wird dann (meist zufällig und nicht beeinflußbar) eine Regel ausgewählt und diese wird ausgeführt, d.h. ihr Aktionsteil wird gestartet. Dies wird solange wiederholt, bis keine Regel mehr anwendbar ist.

Die frei verfügbare Expertensystemshell CLIPS /5/, /6/, /26/ unterstützt das regelbasierte, das prozedurale und das objektorientierte Programmierparadigma. Mit Hilfe der regelbasierten Programmierung kann Wissen - in unserem Fall das Wissen über die Zusammenhänge von Daten und Metadaten - in Form von Heuristiken dargestellt werden. Das objektorientierte Programmieren bietet den Vorteil, daß komplexe Systeme in Module zerlegt werden können. Die prozeduralen Programmierfähigkeiten von CLIPS entsprechen denen in den Sprachen C, Pascal, ADA und LISP.

CLIPS ist portabel, da es in C geschrieben ist und somit auf jedem System läuft, auf dem ein ANSI C-Compiler installiert ist. Da auch der Quellcode mitgeliefert wird, kann CLIPS vom Benutzer auf dessen spezielle Wünsche angepaßt werden. Die Standardversion von CLIPS ist

eine interaktive, textorientierte Entwicklungsumgebung, die gerade für Unerfahrene in Expertensystemen den Anfang leicht macht.

CLIPS bietet eine gute Integration von externen Systemen. Es kann in prozeduralen Code eingebettet werden, als Unterroutine aufgerufen oder in Sprachen wie C, FORTAN und ADA integriert werden. Darüber hinaus können auch externe Programme (UserDefinedFunctions) vom Bedingungs- oder Aktionsteil der Regeln aufgerufen werden. Dies bietet etwa die Möglichkeit, innerhalb der Bedingung auf weitere Datenbankbestände zuzugreifen und in der Aktion den Metadatenbestand des UDK entsprechend zu aktualisieren sowie eine elektronische Post an den UDK-Administrator zu versenden.

Wenn anhand der vorliegenden Ereignisse, der Daten- und Metadatenbestände sowie des Wissens über deren Zusammenhänge in Form von Regeln nicht vollautomatisch eine Aktualisierung des Metadatenbestands vorgenommen werden kann, sind die dem System vorliegenden Informationen entsprechend aufzubereiten und dem Metadaten-Verwalter mitzuteilen. Dies muß asynchron geschehen, da zum Zeitpunkt der Ereignisverarbeitung nicht davon ausgegangen werden kann, daß der Metadaten-Verwalter erreichbar ist. Hier bietet sich die Aufbereitung einer HTML-Seite an, deren URL dem Verwalter per elektronische Post mitgeteilt wird und die soweit möglich vom System bereits ausgefüllt ist. Der Verwalter kann fehlende Informationen ergänzen und durch Abschicken des HTML-Formulars die Aktualisierung der Metadaten anstoßen.

2.5 Realisierung an einem Beispiel

Als Beipiel für die automatische Fortschreibung von Metadaten wurden die Meßstationen des Vielkomponenten-Luftmeßnetzes VIKOLUM gewählt. Dieses Meßnetz besteht aus 69 Stationen, die zum Teil regelmäßig ihren Standort wechseln. Alle Meßstationen sowie das gesamte Meßnetz sind jeweils als ein UDK-Objekt im Metadatenbestand repräsentiert.

2.5.1 Änderungen am Datenbestand und deren Erkennung

Es kann zwischen folgenden Änderungen der VIKOLUM-Meßstation unterschieden werden:

- **Modifikation einer Meßstation**: Mögliche Modifikationen einer Meßstation sind etwa die Änderung des Ortes, der gemessenen Parameter, der Meßgenauigkeit, des Meßverfahrens sowie des Meßintervalls.
- **Einfügen einer neuen Meßstation**: Findet statt, wenn das Meßnetz um weitere Meßstationen erweitert wird.
- **Löschen einer Meßstation**: Dieser Fall ist wohl eher hypothetisch. Auch wenn eine Meßstation stillgelegt wird, werden deren Daten weiterhin in der Datenbank bleiben.

Im weiteren soll der Fall betrachtet werden, daß der Ort einer Meßstation verändert wird. Um dies zu erkennen, ist ein UPDATE-Trigger auf die Attribute „rechtswert", „hochwert" der Tabelle „lfu_messstelle" notwendig. Die Beobachtung der Koordinaten genügt, da sie sich bei einem Ortswechsel auf jeden Fall ändern, was für die Gemeinde, den Kreis oder gar den Regierungsbezirk nicht gelten muß. Tritt ein derartiges Änderungs-Ereignis ein, wird es der Ereignisverwaltung gemeldet, die es der Ereignisverarbeitung weiterrreicht.

2.5.2 Ereignisverarbeitung

Der Bedingungsteil der Regeln der Ereignisverarbeitung überprüft im Falle eines Änderungsereignisses von Meßstation-Koordinaten,

- ob es sich um eine VIKOLUM-Meßstelle handelt, die im UDK als eigenes UDK-Objekt repräsentiert ist;
- ob der Zeitstempel des Ereignisses neuer ist als der Zeitpunkt der letzten Modifikation des UDK-Objektes.

Sind alle Bedingungen erfüllt, wird in der entsprechenden Aktion das bisher die Meßstation beschreibende UDK-Objekt modifiziert, indem die Messungen dieser Meßstelle als abgeschlossen markiert werden. Ein neues UDK-Objekt, das dem bisherigen bis auf den Raumbezug entspricht, wird angelegt und der Raumbezug entsprechend den neuen Bedingungen angepaßt. Der Metadatenbestand ist nun wieder konsistent mit dem Datenbestand, ohne daß der Metadatenverwalter eingreifen mußte. Allerdings wird er über jede automatisch durchgeführte Änderung vom System über elektronische Post informiert.

2.6 Zusammenfassung und Ausblick

Gerade das rasche Wachstum der Metadatenbestände, wie es aufgrund der derzeitigen Nacherhebung von Metadaten sowie der Anforderung, Datenbestände möglichst vollständig durch Metadaten zu beschreiben, zu beobachten ist, erfordert verstärkt Automatismen, um einmal erhobene Metadaten aktuell zu halten.

Das vorgestellte Konzept bietet einen solchen Automatismus und ermöglicht damit eine weitestgehend automatisierten Aktualisierung der Metadaten in dem Sinne, daß der Metadatenbestand konsistent mit dem eigentlichen Datenbestand gehalten wird. Da eine vollständige Automatisierung nicht in jedem Fall erreicht werden kann, wurde eine Interaktionsmöglichkeit mit dem Metadaten-Verwalter berücksichtigt, die letztendlich auch mit einer Qualitätsprüfung der automatischen Metadaten-Fortschreibung einhergehen kann. Das System ist flexibel an unterschiedliche Anforderungen anpaßbar und kann allgemein eingesetzt werden.

Die Realisierung einiger konzeptionierter Systemkomponenten wie der asynchronen Interaktion des Systems mit dem oder den Metadaten-Verwalter(n) über generierte HTML-Seiten ist in einer weiteren Phase entsprechend vorzusehen.

Für eine verteilte UIS-Umgebung sind Mechanismen wie das Überprüfen einer Bedingung und - wenn die Bedingung erfüllt ist - die Ausführung einer Aktion quellenübergreifend erforderlich, um bspw. in einer Bedingung einen neuen Meßwert mit einem Grenzwert aus einer anderen Datenbank vergleichen zu können oder gar eine Berechnungsmethode (z.B. zur Ableitung aggregierter Metadaten) aufrufen zu können. Für derartige quellenübergreifende Zugriffe ist folglich eine entsprechende Funktionalität zu entwickeln, die als Regeldienst in eine CORBA-Umgebung integriert werden kann. Eine (Teil-) Entwicklung einer derartigen Funktionalität wird in /29/ vorgestellt. Neben einer Benutzerinformation, z.B. über neue (Meta-) Daten sowie der teilautomatischen Fortschreibung von Metadaten können entsprechende Regelmechanismen auch gut bei der Implementierung der Strategiekomponente

oder für Ablaufsteuerungen eingesetzt werden. Dies ist im Ausblick zum Bericht zur Integration von Ausbreitungsrechnung und Datenbankzugriffen mittels CORBA innerhalb dieses Globus III Abschlußberichtes angedeutet.

Anhand eines einfachen Beipiels wurde die Funktionsweise des entwickelten Konzeptes gezeigt. Für VIKOLUM-Meßstationen werden bereits weitere Ereignisse erkannt und entsprechende Regeln sorgen für eine Aktualisierung des Metadatenbestandes des UDK. Für weitere Datenbestände steht eine Realisierung noch aus. Idealziel ist es, den Metadatenerfasser bereits während der Ersterfassung die notwendigen Ereignisse und Regeln spezifizieren zu lassen. Dazu muß dies auf eine einfache und leicht zu erlernende Art und Weise möglich sein, um den zusätzlichen Aufwand so gering wie möglich zu halten. Dem zusätzlichen Aufwand bei der Ersterfassung stehen zwei wesentliche Vorteile gegenüber: 1. Der Pflegeaufwand des Metadatenverwalters kann - bei Verringerung der Fehleranfälligkeit - deutlich reduziert werden. 2. Dem Informationssuchenden stehen stets aktuelle Metadaten zur Verfügung, die eine wesentliche Voraussetzung für die Relevanz und Vollständigkeit der Suchergebnisse sind.

Insgesamt kann damit der hier vorgestellte Ansatz zur Automatisierung der Metadatenfortschreibung

- zu einer Entlastung des verantwortlichen Sachbearbeiters und
- zu einer Verbesserung der Datenqualität führen.

3. Literatur

/1/ Batschi, W.-D.(1994): Environmental Thesaurus and Classification of the Umweltbundesamt, in: Stancikova, P.; Dahlberg, I., Environmental Knowledge Organization and Information Management, Berlin, S. 57-62

/2/ Bobrowski, Steve (1993): Oracle server, Begriffe, Berlin

/3/ Borenstein, N.; Freed, N. (1993): RFC 1521 MIME (Multipurpose Internet Mail Extensions), Mechanisms for Specifying and Describing the Format of Internet Message Bodies, New York

/4/ Crossley, David (1996): Wais through the web (http://www.ncsa.uiuc.edu/SDG/IT94/Proceedings/Searching/crossley/paper) discovering environmental information

/5/ Culbert, C.; Rigley, G.; Donnell, B. (1993): CLIPS Reference Manual Volume1, Basic Programming Guide, NASA

/6/ Culbert, C.; Rigley, G.; Donnell, B. (1993): CLIPS Reference Manual Volume2, Advanced Programming Guide, NASA

/7/ Dr. Lippke; Dr. Wagner (1995): Das Thesaurusmodul im UDK, Bedienungsanleitung, Niedersächsisches Umweltministerium

/8/ Dr. Lippke; Dr. Wagner (1995): UDK Benutzerhandbuch, Niedersächsisches Umweltministerium

/9/ Geiger, W.; Reissfelder, M.; Weidemann, R. (1996): Das WWW-basierte Altlasten-Fachinformationssystem AlfaWeb, in: /20/, S.211-220

/10/ Greve, K.; Häuslein, A. (1994): Metainformationen in Umweltinformationssystemen, in: /11/, S.169-178.

/11/ Hilty, L.M.; Jäschke, A.; Page, B.; Schwabl, A. (1994): Informatik für den Umweltschutz, 8 Symposium, Marburg

/12/ Huber-Wäschle, F.; Schauer, H.; Widmayer, P. (1995): Herausforderungen eines globalen Informationsverbundes für die Informatik, 25 GI-Jahrestagung und 13 Schweizer Informatikertag, Zürich

/13/ Koschel, A.; Kramer, R.; Nikolai, R.; Hagg, W.; Wiesel, J. (1996): A Federation Architecture for an Environmental Information System incorporating GIS, the World-Wide Web, and CORBA, in: Third International Conference/Workshop Integrating GIS and Environmental Modeling, Santa Fe, New Mexico

/14/ Kramer, R.; Spandl, H. (1995): Metadatenzugriff in Weitverkehrsnetzen, Eine Realisierung am Beispiel des Umweltdatenkatalogs UDK, in: /12/, S.610-617

/15/ Kramer, R.; Quellenberg, T. (1996): Global Access to Environmental Information, in: Denzer, R.; Russel, D.; Schimak, G. (1996): Environmental Software Systems, Proceedings of the International Symposium on Environmental Software Systems, S.209-218, London

/16/ Kramer, R.; Nikolai, R. (1996): Accessing multilingual, heterogeneous data sources in wide area networks, requirements and approach, in: Christiansen, H.; Larsen, H.L.; Andreasen, T., Flexible Query-Answering Systems, Proceedings of the 1996 workshop (FQAS'96), Denmark, S.255-264

/17/ Kramer, R.; Nikolai, R.; Keitel, A.; Legat, R.; Zirm, K. (1996): Enhancing the Environmental Data Catalogue UDK for the World Wide Web, in: /20/ S.59-68

/18/ Lessing, H.;Schütz, T. (1995): Der Umwelt-Datenkatalog als Instrument zur Steuerung von Informationsflüssen, in: /11/, S.159-167

/19/ Lessing, H.; Günther, O.; Swoboda, W. (1995): Ein objektorientiertes Klassenkonzept für den Umweltdatenkatalog (UDK), in: Kremers, H.; Pillmann, W., Raum und Zeit in Umweltinformationssystemen, 9th International Symposium Computer Science for Environmental Protection CSEP'95, in: Umwelt-Informatik Aktuell, Marburg, S.391-399

/20/ Lessing, H.; Lipeck, U.W. (1996):. Informatik für den Umweltschutz, 10 Symposium, Hannover

/21/ Günther, O., Lessing, H.; Swoboda, W. (1996): UDK: A European Environmental Data Catalogue, in Third International Conference/Workshop Integrating GIS and Environmental Modeling, Santa Fe

/22/ Mayer-Föll, R.; Jäschke, A. (1995): Projekt GLOBUS, Konzeption und prototypische Realisierung einer aktiven Auskunftskomponente für globale Umwelt-Sachdaten im Umweltinformationssystems Baden-Württemberg, Phase II 1995. Nummer FZKA 5700, in Wissenschaftliche Berichte, Forschungszentrum Karlsruhe Technik und Umwelt, Karlsruhe

/23/ Mayer-Föll, R.; Strohm, J.; Schultze, A. (1996): Das Umweltinformationssystem Baden-Württemberg, Überblick Rahmenkonzeption, in /20/

/24/ McCool, R. (1994): Common gateway interface overview, Technical report, National Center for Supercomputiong Applications (NCSA) http://hoohoo.ncsa.uiuc.edu/cgi/overview.html

/25/ Riekert, W.F.; Gaul, M.; Klügl, G.; Wiest, G.; Henning, I.; Schmidt, F. (1996) Management verteilter und heterogener Informationsquellen für das UIS Baden-Württemberg, in /20/

/26/ Riley, G. (1995): What are Expert Systems? What is Clips?, CLIPS World Wide Web Page http://www.jsc.nasa.gov/clips/CLIPS.html

/27/ CNR Rome; Umweltbundesamt Berlin (1996): GEMET, General European Multilingual Environment Thesaurus, Technical report, European Environment Agency

/28/ Swoboda, W.; Lessing, H.; Grolimund, P.; Günther, O.; Haas, U.; Legat, R.; Vogler, M.; Zirm, K. (1995): Metadatenklassen im Umweltdatankatalog (UDK), in /12/, S601-609

/29/ v. Bültzingslöwen, G.; Koschel, A.; Kramer, R.: *Active Information Delivery in a CORBA-based Distributed Information System*, in: Aberer, K.; Helal, A. (Hrsg.), Proc. First IFCIS International Conference on Cooperative Information Systems (CoopIS'96). IEEE Computer Society Press, Los Alamitos, California. Brüssel, Belgien. Juni 1996. S.218-227

Grenz-/ Richtwertdatenbank

R. Kramer; C. Rolker,
Forschungszentrum Informatik (FZI),
Haid-und-Neu-Str. 10-14,
76131 Karlsruhe

1. Einleitung

Grenzwerte werden gebraucht, um gemessene Umweltdaten bearbeiten, bewerten und interpretieren zu können. Sie sind in einer Vielzahl von Richtlinien, Gesetzen und Verordnungen für einzelne Bundesländer, Staaten, aber auch für Ländergemeinschaften wie die EU und die UN festgelegt. Diese Richtlinien unterscheiden sich u.a. in ihrer Terminologie und in den Details ihrer Festlegungen (z.B. Grenzwert für einzelne Schwermetalle oder aber für schwermetallhaltige Verbindungen insgesamt).

Werden Grenzwerte aus unterschiedlichen Quellen in Datenbanksystemen erfaßt, sollten daher insbesondere auch mit der Materie nur wenig vertraute Benutzer ohne langwieriges Reformulieren von Datenbankanfragen die gesuchten Ergebnisse erhalten. Für die zentrale, relationale Grenz-/Richtwertdatenbank (GRDB) des UIS, die auf Oracle basiert, wurde deshalb ein Modul zur Realisierung benutzerkonfigurierbarer, unscharfer Datenbankanfragen realisiert.

Anfragen innerhalb des Datenbankservers der GRDB werden in einer Client/Server-Architektur bearbeitet. Es können Komponenten des UIS, für die die Grenzwerte von Interesse sind wie z.B. TULIS (Technosphäre- und Luft-Informationssystem) die Rolle des Clients übernehmen. Darüberhinaus können aber auch Module, die die Mensch-Maschine-Schnittstelle realisieren, als Clients das unscharfe Datenbankabfragemodul verwenden, um Benutzern eine komfortable Suche nach Grenzwerten zu ermöglichen. Um auf die GRDB im Internet bzw. Intranet zugreifen zu können, wurde die Mensch-Maschine-Schnittstelle mit HTML verwicklicht.

Mittlerweile sind in der GRDB über 1600 Grenzwerte eingetragen, und der Bestand wächst stetig. Das große Interesse an der GRDB ist auch daran abzulesen, daß das ursprüngliche Datenmodell der GRDB in den letzten Jahren von den GRDB-Verantwortlichen der LfU erweitert wurde, um noch komfortablere Suchanfragen zu ermöglichen. Die Module zum Zugriff auf die GRDB haben inzwischen durch entsprechende Erweiterungen und Tests das Prototypstadium verlassen.

2. Lösungsansätze: komfortable Suchdienste

Die heute kommerziell verfügbaren relationalen (und auch objektorientierten) Datenbanksysteme unterstützen lediglich Anfragen, die der klassischen booleschen Logik genügen. Insbesondere dann, wenn komplexe Sachverhalte modelliert und in Datenbanksystemen erfaßt werden sollen, ist diese Situation wenig zufriedenstellend, da die Formulierung der Suchprädikate in solchen präzisen Anfragen sehr schwierig werden kann. Wird eine Anfrage zu sehr einschränkend formuliert, besteht die Gefahr, daß keine Ergebnisse gefunden werden und daraus fälschlicherweise der Schluß gezogen wird, die gesuchten Informationen lägen nicht vor. Wird hingegen nicht hinreichend genug eingegrenzt, dann wird der Anwender mit einer unstrukturierten Ergebnismenge konfrontiert. In solchen Datenbanksystemen müssen Datenbanknutzer ihre Anfragen daher mehrfach umformulieren und neu erstellen, bis sie das tatsächlich gesuchte Ergebnis erhalten.

Ein Ansatz, die beschriebenen Probleme zu lösen, besteht darin, nicht nur den exakt angefragten Wert sondern auch Werte in seiner Umgebung als Anfrageergebnis mit auszugeben (/1/). Die Umgebung kann sich sowohl auf die Groß-/Kleinschreibung als auch auf die Semantik beziehen.

Ein solcher Ansatz wurde für die GRDB realisiert, da die Richtlinien, aus denen die Grenzwerte stammen, Unterschiede in Detail und Terminologie aufweisen und es deshalb für wenig vertraute Benutzer ansonsten zu langwierigem Reformulieren der Datenbankanfragen führen würde. Im folgenden werden die komfortablen Suchdienste vorgestellt, durch die dieser Ansatz verwirklicht wird:

1. *(Nicht-)Berücksichtigung der Groß- und Kleinschreibung*: Es kann mit Berücksichtigung oder ohne Berücksichtigung der Groß- und Kleinschreibung gesucht werden. Wird unabhängig von der Groß- und Kleinschreibung der Eingabe gesucht, dann ist es ohne Bedeutung, ob der Benutzer „Schwefel" oder „schwefeL" oder „SCHWEFEL" eingibt. Das Ergebnis ist immer gleich. In manchen Fällen ist jedoch eine Berücksichtigung der Groß- und Kleinschreibung sinnvoll, z.B. wenn mit „%schwefel%" nach Datensätzen gesucht werden soll, in denen die Buchstabenfolge „schwefel" tatsächlich klein geschrieben ist und deshalb nicht am Wortanfang vorkommen kann.
2. *Synonymsuche*: Bei einer Synonymsuche wird nicht nur nach den angegebenen Attributen gesucht, sondern es werden auch Grenzwerte gefunden, die unter dem Synonym abgespeichert wurden.
3. *Ähnlichkeitssuche*: Wird eine Ähnlichkeitssuche durchgeführt, dann wird nicht nur nach den angegebenen Suchparametern exakt gesucht, sondern es werden auch Grenzwerte mit ähnlichen Suchparametern ausgegeben.
4. *(Nicht-)Berücksichtigung änderner Herkünfte*: Bei der Suche nach Grenzwerten unter Angabe der Herkunft (z.B.76/160/EWG: Richtlinie des Rates vom 8.12.75 über die Qualität von Badegewässer) kann nach Grenzwerten in der Datenbank gesucht werden, die die angegebene Herkunft haben, ohne zu beachten, daß diese durch eine andere Herkunft inzwischen geändert wurden. Mit der Suche nach ändernden Herkunften werden zusätzlich die Grenzwerte aus den ändernden Herkunften mit ausgegeben.
5. *Dimensionseingabe als Suchkriterium oder als Ausgabedimension:* Mit der Angabe der Dimension können verschiedene Absichten verbunden sein. Entweder wird die Dimension angegeben, um die Suche einzuschränken und nur Grenzwerte als Ergebnis der Suche zu erhalten, die die angegebene Dimension besitzen. Oder es soll die Suche der Grenzwerte unabhängig von der eingesetzten Dimension durchgeführt werden und erst bei der Ausgabe sollen die Grenzwerte, die in die angegebene Dimension umrechenbar sind, in dieser angegeben werden. Die übrigen Ergebnisgrenzwerte sollen mit ihrer nicht-umrechenbaren Dimension ausgegeben werden.

Um Ähnlichkeitssuchen durchführen zu können, müssen suchattributspezifische Festlegungen der Ähnlichkeiten definiert werden. Jeder Ähnlichkeitswert muß zwischen 0 und 1 liegen. 0 entspricht überhaupt nicht ähnlich, 1 entspricht gleich. In der GRDB müssen in zusätzlichen Tabellen die Ähnlichkeiten abgelegt werden, die dann im Falle einer Ähnlichkeitssuche durchsucht werden. Beispiele für Ähnlichkeiten bezüglich des Werttyps Grenzwert sind in der folgenden Tabelle zu finden.

zu ersetzender Werttyp	ersetzt durch	Ähnlichkeit
Grenzwert	Richtwert	0.9
Grenzwert	Leitwert	0.7
Grenzwert	Schätzwert	0.5
Grenzwert	Orientierungswert	0.3

Der Werttyp beschreibt die Art einer Richtlinie.

Die verschiedenen Suchdienste schließen sich nicht gegenseitig aus, sondern können beliebig kombiniert werden.

3. Architektur

Der Zugriff auf die GRDB basiert auf dem Client/Server-Konzept und ist sehr modular realisiert worden. Nur deshalb ist es möglich, sowohl eine interaktive Benutzerschnittstelle als auch eine saubere Systemschnittstelle anbieten zu können. Eine Übersicht über die Architektur gibt die Abbildung 1.

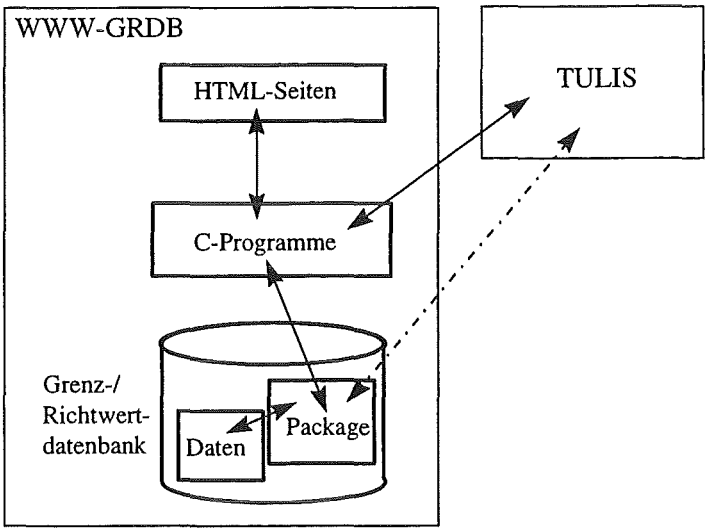


Abbildung 1: Systemarchitektur mit Aufrufhierarchie und Nutzungsmöglichkeiten

In der Oracle-Datenbank sind neben den eigentlichen Daten, die die Grenzwerte repräsentieren, Oracle-Procedures in einem Package abgelegt, die für eine schnelle Bearbeitung der Anfragen inklusive der Suchdienste sorgen. Darüber liegen C-Programme,

die die Procedures aufrufen und die von den anderen Komponenten des UIS (z.B. TULIS) zum Zugriff auf die Grenzwerte aufgerufen werden können. Um auf die GRDB im Internet bzw. Intranet zugreifen zu können, wurde die Mensch-Maschine-Schnittstelle mit HTML verwicklicht. Innerhalb der HTML-Seiten werden die C-Programme als CGI-Skripte ausgeführt.

Die WWW-GRDB umfaßt die HTML-Seiten, die Oracle-Datenbank inklusive der Procedures und die C-Programme.

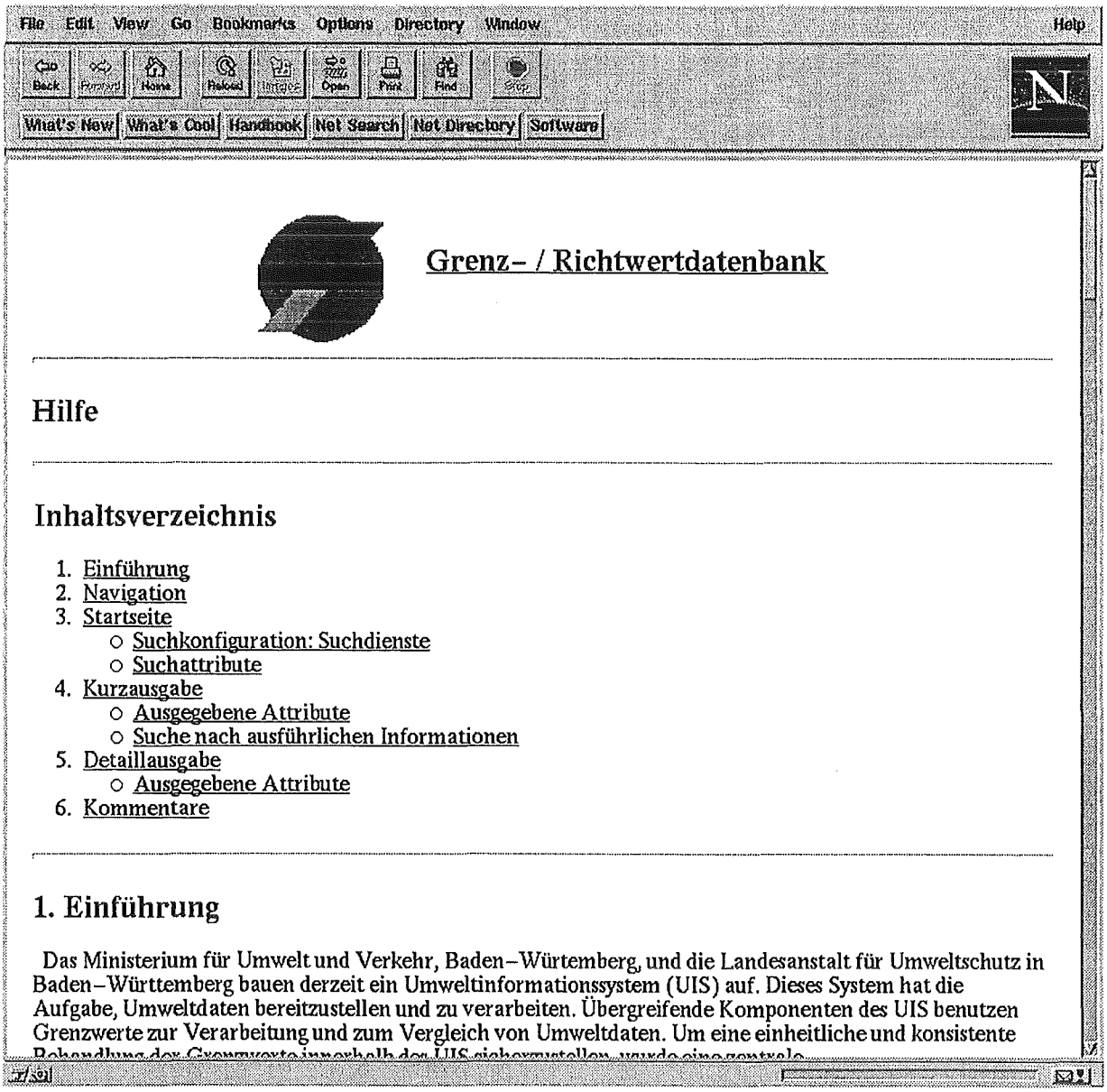


Abbildung 2: Hilfeseite der WWW-GRDB

4. Beispielsuche in der WWW-GRDB

Die WWW-GRDB besitzt 4 verschiedene HTML-Seiten. Auf der Start-HTML-Seite wird die Suche spezifiziert. Danach öffnet sich eine HTML-Seite, die die Ergebnisgrenzwerte der

Suche ausgibt. Allerdings werden nur die 10 wichtigsten Informationen zu jedem Grenzwert geliefert. Der Benutzer hat nun die Möglichkeit, die ihn interessierenden Grenzwerte auszuwählen. Alle selektierten Grenzwerte werden dann in einer neuen HTML-Seite detaillierter dargestellt. Von diesen drei HTML-Seiten bestehen mehrere Links auf eine Hilfeseite (siehe Abbildung 2), die die Suche genau beschreibt und dem Benutzer die Bedienung der WWW-GRDB erleichtert. Da sich bei Aufruf der Hilfeseite ein neues Fenster öffnet und das alte bestehen bleibt, kann der Benutzer parallel die Hilfeseite lesen und seine Einstellungen machen.

Abbildung 3: Startseite der WWW-GRDB

Die Suche nach Grenzwerten in der GRDB über die HTML-basierte Schnittstelle geschieht über ein 4 Phasen-Konzept. Dieses wird im folgenden anhand einer Beispielsuche nach Schwefeldioxid-Grenzwerten, die in Deutschland gültig sind, vorgestellt.

Auf der Startseite (siehe Abbildung 3) der WWW-GRDB durchläuft der Benutzer zwei Phasen. In der ersten Phase legt der Benutzer durch Drücken einiger Selekt-Buttons die *Suchkonfiguration* fest: Bei der Beispielsuche soll die Groß-/Kleinschreibung nicht berücksichtigt werden. Es soll eine Ähnlichkeitssuche und eine Synonymsuche durchgeführt werden.

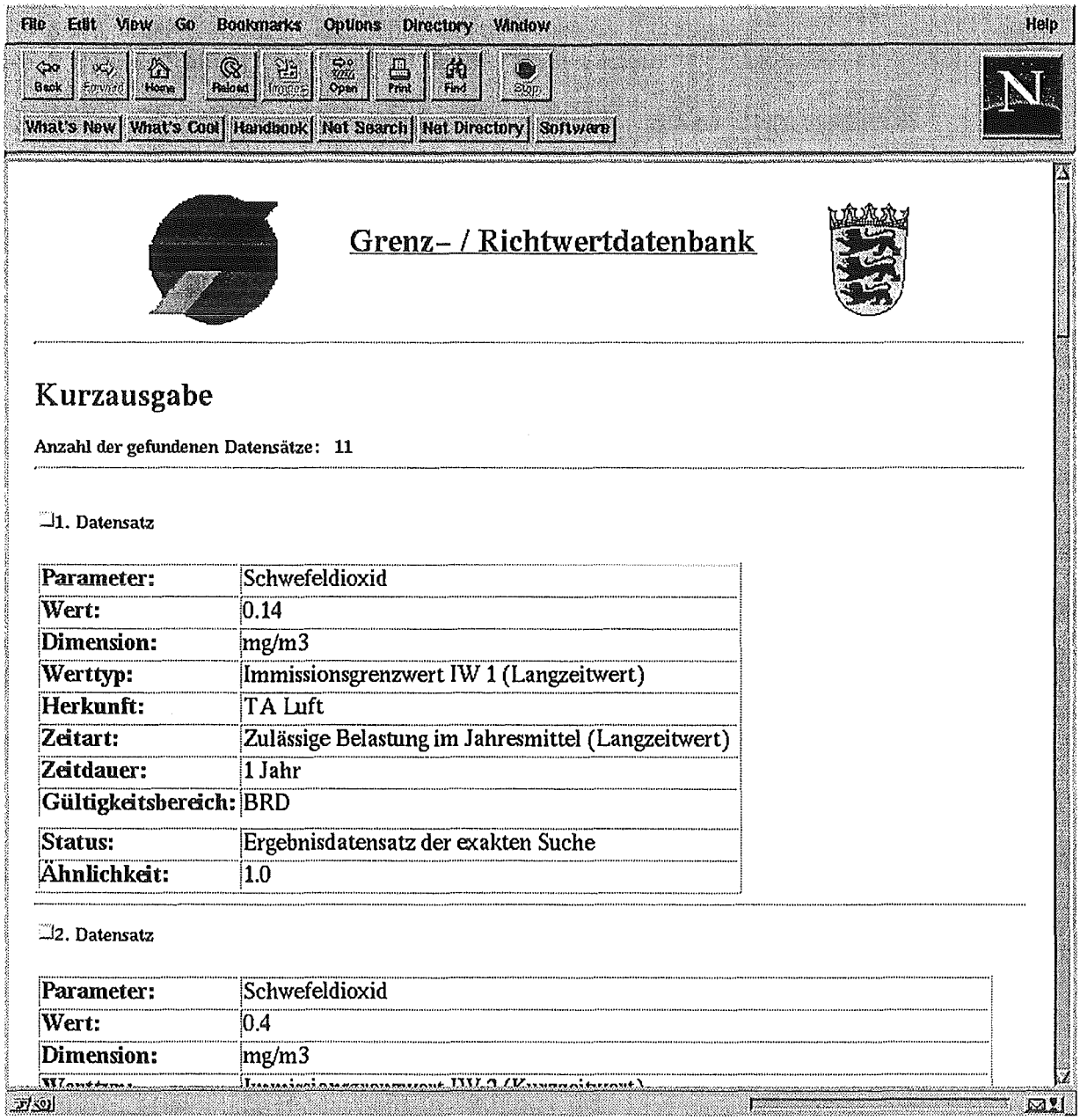


Abbildung 4: Kurze, übersichtliche Ergebnisausgabe der WWW-GRDB

Zuletzt kann der Benutzer sich noch einen der *Regelungsbereiche* (z.B. Emissionen, Immissionen, etc.) aussuchen, in dem gesucht werden soll. Der Benutzer braucht die in der GRDB vorhandenen Regelungsbereiche nicht zu kennen, sondern kann sich in einem Pull-Down-Menü einen Regelungsbereich aussuchen. Der Inhalt des Pull-Down-Menüs wurde beim Aufruf der Start-Seite durch eine Datenbankanfrage gefüllt und enthält so immer die aktuell in der GRDB vorhandenen Regelungsbereiche. Ist sich der Benutzer unsicher bei der Wahl des Regelungsbereichs, dann kann er aber auch in allen Regelungsbereichen suchen

lassen. In der Beispielsuche soll im Regelungsbereich Luft-Immissionen gesucht werden.

In der zweiten Phase der Suche gibt der Benutzer die gewünschten Suchattribute ein. Er kann für einen oder mehrere von 12 vorgegebenen Attributen die Werte spezifizieren. Im Beispiel wird Schwefeldioxid als Parameter und Deutschland als Gültigkeitsbereich eingegeben.

Nachdem der Benutzer die erste und zweite Phase durchlaufen hat, kann er nun die Suche starten, indem er unten auf der HTML-Seite den entsprechenden Button drückt. Die Suche wird nun durchgeführt und eine HTML-Seite mit den Ergebnissen der Suche wird statt der Startseite geöffnet.

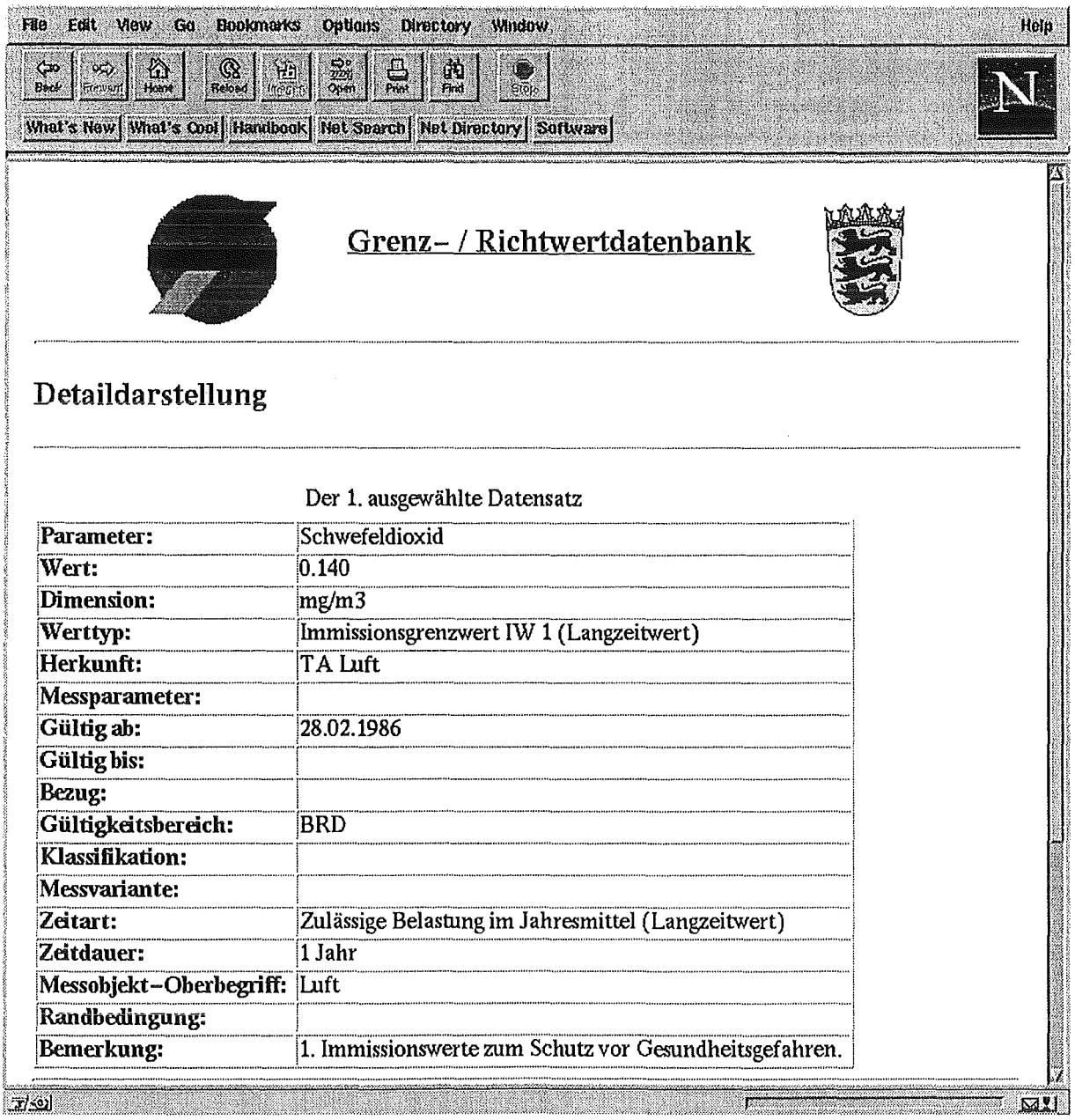


Abbildung 5: Detaillierte Ergebnisausgabe der WWW-GRDB

Nun beginnt die dritte Phase der Suche. Der Benutzer hat nun eine HTML-Seite vor sich, die die gefundenen Grenzwerte jeweils mit 10 beschreibenden Informationen ausgibt (siehe Abbildung 4). Durch die Ähnlichkeitssuche werden nicht nur deutschlandweite Grenzwerte

bzgl. Schwefeldioxid, sondern auch z.B. europaweite Schwefeldioxidgrenzwerte ausgegeben.

Der Benutzer hat nun die Möglichkeit, durch Anklicken der entsprechenden Check-Boxen einige Grenzwerte auszusuchen, die ihn näher interessieren. Danach drückt er den Button zur detaillierten Ausgabe. In der Beispielsuche wird eine ausführliche Information zum 1. Grenzwert gewünscht.

Nun befindet sich der Benutzer in der 4. Phase. Eine neue HTML-Seite (siehe Abbildung 5) wird erzeugt, die die Grenzwerte mit 17 Attributen beschreiben. Der Benutzer hat nun das ausführliche Ergebnis seiner Grenzwertsuche vor sich.

5. Resümee

Mit der Grenz-/Richtwertdatenbank des Ministeriums für Umwelt und Verkehr, Baden-Württemberg, wurde gezeigt, wie eine komfortable Suchfunktionalität auf der Basis eines relationalen Datenbanksystems realisiert werden konnte. Durch die Wahl eines dienstebasierten Ansatzes konnten die wesentlichen Komponenten der Implementierung von der ursprünglichen Oracle-Forms-Umgebung auch für eine WWW-Umgebung, die sich gleichermaßen für Internet und Intranet eignet, übernommen werden. Die hohe praktische Bedeutung zeigt sich in der stark wachsenden Anzahl erfaßter Grenz-/Richtwerte durch die Fachabteilungen sowie in der aus der praktischen Arbeit resultierenden Weiterentwicklung des Datenmodells.

6. Literatur

/1/ Kramer, Ralf (1995): Unscharfe Anfragen an eine Grenzwert-Datenbank - Softwarearchitektur, Anfragebearbeitung und Leistungsoptimierung, in: Lausen, G. Datenbanksysteme in Büro, Technik und Wissenschaft, GI-Fachtagung, Dresden, 22.-24. März 1995, Informatik aktuell, Springer, März 1995, S. 393-402

Fachübergreifende Umweltdaten für Generalisten und Führungskräfte

M. Gaul, I. Henning, G. Klügl, B. Münst, W.-F. Riekert, G. Wiest,
Forschungsinstitut für anwendungsorientierte Wissensverarbeitung (FAW)
an der Universität Ulm,
Helmholtzstr. 16, 89081 Ulm,
Postfach 2060, 89010 Ulm

1. Weiterentwicklung des Umwelt-Führungsinformationssystems (UFIS II)

1.1 Einführung

Im Umweltinformationssystem (UIS) Baden-Württemberg wurden seit den Anfängen im Jahr 1984 inhaltlich und funktional weit fortgeschrittene Informationssysteme entwickelt, die inzwischen die Tauglichkeit für den praktischen Einsatz bewiesen haben. In den einzelnen Dienststellen wurde ein umfangreicher Bestand an Informationen und Wissen aufgebaut, der in Form von Daten, Methoden und multimedialen Dokumenten auf verschiedenen Computersystemen bereitgehalten wird.

Zunehmende Bedeutung gewinnt die Aufgabe, diesen Schatz an Informationen und Wissen an den (PC-)Arbeitsplatz eines jeden Mitarbeiters in der Umweltverwaltung zu bringen, soweit dies bei der Erledigung der Dienstaufgaben zusätzlichen Nutzen bringt. Besonders wichtig ist es in diesem Zusammenhang, einen sogenannten Umweltgeneralisten zu unterstützen, der sich einen fachübergreifenden Überblick über bestimmte Umweltfragen verschaffen möchte, um beispielsweise ganzheitliche Bewertungen der Umweltsituation vorzunehmen oder für Planungen Entscheidungsgrundlagen zusammenzustellen. Zu diesem Zweck entwickelt das FAW seit Anfang 1996 im Auftrag des Ministeriums für Umwelt und Verkehr Baden-Württemberg auf der Basis von World Wide Web (WWW) und der neuen Systemarchitektur des UIS als Teil des Landessystemkonzeptes (LSK) eine grundlegend überarbeitete Betriebsversion des *Umwelt-Führungs-Informationssystems* (Projekt UFIS II). Dabei wird insbesondere auf Entwicklungen aus den Forschungs- und Entwicklungsprojekten INTEGRAL III und GLOBUS, insbesondere den GLOBUS-Systemkern, zurückgegriffen. Koordiniert wird das Projekt mit anderen verwandten Projektaktivitäten im Rahmen des Verbundvorhabens GLOBUS III.

Ein wesentliches Ergebnis aus INTEGRAL und GLOBUS war die Erkenntnis, daß es mit Hilfe moderner Telematiktechniken, also mit Hilfe einer Kombination aus Kommunikationstechnologie und Informatik möglich ist, die im UIS vorhandenen Einzellösungen miteinander zu integrieren und zu einem verteilten Informationsverbund zusammenzuschließen. Die hierbei angewandten Vernetzungskonzepte, über die nachfolgend ein Überblick gegeben wird, erlauben eine wirtschaftliche Nutzung der bei den einzelnen Dienststellen vorhandenen Hard- und Software-Ressourcen. Der Vorteil für die Nutzer liegt in dem großen Angebot an Informationsquellen, die über das Netz bereitgestellt werden. Diese müssen nicht lokal vorgehalten und auch nicht lokal gepflegt werden. Es können daher preisgünstige und leicht wartbare PC-Systeme als Rechnerarbeitsplätze im UIS genutzt werden.

1.1.1 Informationsquellen im UIS

Die existierenden UIS-Komponenten halten im wesentlichen drei Arten von Informationsquellen bereit, nämlich Daten, Methoden und multimediale Dokumente.

Die primären Informationsquellen im UIS sind *Daten* über die Umwelt, z.B. aus Meßnetzen, statistischen Erhebungen oder Satellitensensoren, auch in aufbereiteter Form wie korrigierte

Daten, aggregierte Daten oder komplexe Daten (z.B. Geodaten). Daten sind allerdings nur mittelbare Informationsquellen. Um Informationen zu erhalten, muß noch eine Interpretation erfolgen. Dies ist zugleich ein Nachteil und ein Vorteil: Zum einen erfordert die Interpretation von Daten Wissen, über das nicht jeder Informationssuchende verfügt. Andererseits sind Daten vielseitig interpretierbar und nicht auf eine einseitige Art der Nutzung festgelegt.

Viele Dienststellen in der Umweltverwaltung verfügen über geeignete Verfahren, um aus Daten Informationen abzuleiten. Häufig sind diese Verfahren in Form von *Methoden* oder Dienstprogrammen auf einem Computer automatisch ablauffähig. Diese Methoden dienen als Informationsquellen, indem sie Informationen aus vorhandenen Daten generieren. Beispiele für Methoden sind Simulationen, Datenbankzugriffe, statistische Analysefunktionen oder Präsentationsfunktionen (z.B. zur Generierung von Reports oder Landkarten).

In vielen Fällen ist es sinnvoll, nicht nur auf Originaldaten zurückzugreifen, sondern darüber hinaus auch auf Arbeitsergebnisse aus früheren Interpretationen solcher Daten. Diese Arbeitsergebnisse liegen in der Regel in Form von Berichten und Karten, kurz, in Form *multimedialer Dokumente* vor. Berichte und Karten sind sehr wichtige Informationsquellen, da sie – in impliziter Form – auch die Meta-Daten enthalten, die zur Interpretation der in das Dokument eingegangenen Originaldaten erforderlich sind. Insbesondere für die Öffentlichkeit und für Entscheidungsträger ist diese Art von Informationsquellen unverzichtbar.

Um all diese Informationsquellen – Daten, Methoden und multimediale Dokumente – auf rationelle Weise ortsunabhängig nutzen zu können, wurde im Rahmen des Vorhabens GLOBUS ein Dienstekonzept entworfen und in einer ersten Ausbaustufe realisiert. Die vorhandenen UIS-Komponenten werden dabei in Einzelfunktionalitäten zerlegt, die als separat ansprechbare Netzwerkdienste auf einem Weitverkehrsnetz bereitgestellt werden.

1.1.2 Management von Informationsquellen im World-Wide-Web

Für UFIS II wurden die Mechanismen des WWW gewählt, um die verschiedenen, verteilten Informationsquellen im UIS, also Daten, Methoden und multimediale Dokumente, in Form von Netzwerkdiensten bereitzustellen und in einen virtuellen Informationspool zu integrieren. Diese Informationsquellen sind dabei netzweit eindeutig durch eine WWW-Adresse, eine sogenannte URL (Uniform resource Locator) identifiziert. Die Aktivierung einer Informationsquelle erfolgt durch Angabe einer URL oder durch Anklicken eines Hyperlinks, der mit einer solchen URL hinterlegt ist.

So lassen sich verschiedene Arten von Daten, die in Dateien abgelegt sind, im WWW über eine URL ansprechen. Dies bewirkt, daß die Datei auf den Client-Rechner übertragen wird und dort entweder im Filesystem abgelegt oder von einem zugehörigen clientseitiges Verarbeitungsprogramm weiterverarbeitet werden kann. Ein solches Programm – man unterscheidet eigenständig ablauffähige *Viewer* und dynamisch in den WWW-Browser eingebundene *Plug-Ins* – kann den Nutzer dann bei der Interpretation der Daten unterstützen, d.h. bei der Erzeugung von Information aus den Daten. Typische, im Rahmen des Projekts INTEGRAL genutzte Viewer sind Tabellenkalkulationssysteme, Grafiksysteme, Textverarbeitungssysteme (insbesondere aus der MS-Office-Suite) und Systeme für das Desktop Mapping (ArcView).

Methoden treten im WWW als parametrisierbare Dienstprogramme in Erscheinung. Der Aufruf eines solchen Dienstprogramms über das WWW erfolgt durch Aktivieren einer URL mit Hilfe eines WWW-Formulars. Das Dienstprogramm erzeugt als sichtbares Ergebnis einen Datenstrom, der die Definition einer Hypertextseite enthält, die schließlich dem Dienstenutzer angezeigt wird. Komplexere Dialogformen im WWW sind darüber hinaus mit Hilfe der Programmiersprache JAVA realisierbar.

Die Repräsentation *multimedialer Dokumente* im WWW bereitet keinerlei Probleme, da WWW ursprünglich speziell hierfür konzipiert wurde. Neuere Systeme zur Textverarbeitung und zum Desktop Publishing stellen bereits Generatoren für die im WWW verwendete Dokumentendefinitionssprache HTML (Hypertext Markup Language) bereit.

1.1.3 Bereitstellung von Informationen aus Umweltdatenbanken

Eine wichtige Zielsetzung in UFIS II ist es, die Bereitstellung und Nutzung von Informationen aus Umweltdatenbanken zu unterstützen. Der gewählte Ansatz besteht dabei darin, den Online-Zugriff auf Informationen in solchen Datenbanken über das WWW zu ermöglichen. Hierfür hat das FAW eine Softwarelösung, ein *generisches Umweltdatenbank-Gateway*, entwickelt. Dieses Gateway erlaubt es, beliebige relationale Datenbanken an das WWW anzuschließen. Es wird im Rahmen des Projekts UFIS II für die Anbindung der Datenbank der übergreifenden UIS-Komponenten (DB-ÜKO) genutzt. Das Gateway ermöglicht die Einrichtung von Netzwerkdiensten, sogenannten *Selektoren*, die den Zugriff auf ausgewählte Informationsbestände aus der Datenbank erlauben. Diese Selektoren treten dem Benutzer gegenüber in Form von Bildschirmformularen (Masken) in Erscheinung. Die vom Benutzer eingetragenen Formularinhalte dienen zur Parametrisierung von Datenbankanfragen. Das Ergebnis einer Anfrage ist eine Tabelle, die in Form eines dynamisch generierten Hypertextdokuments im WWW-Browser angezeigt wird und auf Wunsch auch (in Datenform) auf das Clientsystem („Individueller Rechnerarbeitsplatz") des Benutzers übertragen werden kann. Dort kann die Tabelle über den Viewer-Mechanismus in einem Tabellenkalkulationssystem (Excel), einem Desktop-Mapping-System (ArcView) oder einem Textverarbeitungssystem (MS-Word) zur Erstellung von Studien und Berichten genutzt werden.

Dabei benötigt die datenanbietende Stelle keine besonderen Programmierkenntnisse, um Selektoren für dieses Gateway definieren zu können. Zum Anschluß einer Umweltdatenbank an das WWW genügt die Erstellung eines Satzes von Beschreibungsdateien, wofür lediglich Kenntnisse der Hypertext-Definitionssprache HTML und der Datenbanksprache SQL erforderlich sind. *WebQuery*, ein am FAW für das generische Umweltdatenbank-Gateway entwickelter Interpreter, nutzt diese Beschreibungsdateien zur Verarbeitung der Nutzeranfragen. Das Umweltdatenbank-Gateway stellt dabei unabhängig von der angeschlossenen Datenbank allgemeine Funktionen hinsichtlich Sitzungsverwaltung, Mehrbenutzerbetrieb, Benutzerverwaltung, Statistik, Verwaltung von Zwischenergebnissen, Download usw. bereit, ohne daß hierfür eine Programmierung erforderlich wäre.

Die Selektoren der Umweltdatenbank sind einzelne, unabhängig voneinander ansprechbare und nutzbare Dienste. Für diese Dienste können an unterschiedliche Benutzergruppen unterschiedliche Berechtigungen vergeben werden, so daß sich die Datenbank speziellen Nutzergruppen (Einzelpersonen, Dienststellen, Verwaltung allgemein, Öffentlichkeit usw.)

gegenüber mit unterschiedlichen Inhalten und Abfrageoptionen darstellen kann.

1.1.4 Unterstützung des Retrievals von Umweltinformationen

Das wachsende Angebot von umweltrelevanten Diensten auf dem Internet bringt mit sich, daß einzelne Informationsangebote oft nur sehr schwer auffindbar sind. Erforderlich ist deshalb eine *Filter- und Brokerlösung* auf der Basis von Meta-Information, mit welcher es den UFIS-Nutzern möglich ist, gesuchte Informationen zu finden (Brokerfunktionalität), ohne durch ungewünschte Informationen gestört zu werden (Filterfunktionalität). Das FAW hat für das Retrieval von Umweltinformationen im WWW einen Prototyp eines Filters und Brokers entwickelt, der bei der Weiterentwicklung des Umwelt-Führungs-Informationssystems UFIS zur Gestaltung einer Nichtexpertenoberfläche genutzt wird und der in einer multilingualen Form für das deutsche Umweltinformationssnetz GEIN (German Environmental Information Network) als sogenannter *Locator* für Umweltinformationsquellen prototypisch eingesetzt wird.

Der Locator enthält ein durchsuchbares Verzeichnis aller verfügbaren Informationsquellen (Daten, Dienstprogramme, Dokumente etc.). Diese Informationsquellen können hinsichtlich Ortsbezug, Zeitbezug, Fachbezug und datenhaltender Stelle recherchiert werden. Das Verzeichnis der Informationsquellen wurde auf der Basis der Datenbank des Umweltdaten-katalogs (UDK) realisiert. Der Locator hält für unterschiedliche Benutzergruppen spezifische Sichten auf den Informationsbestand bereit (Filterfunktion). Dadurch sind für die jeweilige Benutzergruppe nur diejenigen Informationsangebote sichtbar, die jeweils von Interesse sind.

Das Ergebnis einer Suche ist eine Liste von Informationsquellen. Durch einfachen Mausklick kann man zu jeder aufgelisteten Informationsquelle eine Detailbeschreibung abrufen sowie auf die Originalinformation durchgreifen, sofern diese auf einem Server bereitgehalten wird.

Der Locator bietet einen vereinfachten Zugriff auf die UDK-Datenbestände. Für die Eingabe von Fachbezügen kann auf den polyhierarchischen Umweltthesaurus des Umweltbundesamts zurückgegriffen werden. Semantische Beziehungen zwischen Begriffen werden vom System bei der Suche nach Informationsquellen ausgewertet und genutzt. Für die Informationssuche im internationalen Rahmen wurde der GEIN-Locator mit einer mehrsprachigen Thesauruserweiterung ausgestattet.

Die Eingabe von Raumbezügen wird durch einen sogenannten *Gazetteer*, d.h. ein strukturiertes Ortsverzeichnis unterstützt. Mit Hilfe des Gazetteers kann eine ortsbezogene Suche nach Umweltinformationen sowohl über die Angabe eines Suchrechtecks als auch durch Angabe geographischer Namen erfolgen. Da geometrisch-topologische Beziehungen zwischen Raumbezügen im Gazetteer repräsentiert sind, findet der Locator bei einer ortsbezogenen Suche auch Umweltinformationen mit übergeordnetem, untergeordnetem oder überlappenden Ortsbezug.

1.2 Das System UFIS II

1.2.1 Überblick

Das System UFIS II (Umwelt-Führungs-Informations-System) verfolgt verschiedene Ziele, denen der bisherige Titel des Systems nicht mehr ganz gerecht wird. Das neue System UFIS II soll nicht mehr ausschließlich Informationssystem der Führungseben darstellen, sondern vielmehr Informationen für den Entscheider und Umweltgeneralisten zur Verfügung stellen. Hierbei sind Systemoberflächen gefordert, die eine einfache Handhabung garantieren und dem Anfragenden Informationen ohne große Interaktionen bereitstellen. Das System muß daher zusätzlich zu der bisherigen Navigation eine Recherche nach vorhandenen und verfügbaren Informationen beinhalten, die themenspezifisch nach Informationen recherchiert und das Durchgreifen ermöglicht. Für die UFIS-Daten würde ein Durchgreifen in einer Selektion der UFIS-Rohdaten aus DB-ÜKO resultieren. Diese selektierten Rohdaten sind in ihrer Grundform nicht sehr aussagekräftig und bedürfen daher meist einer visuellen Aufbereitung in Form von z.B. Karten oder Diagrammen. Die Aufbereitung und Visualisierung sollte nach Möglichkeit innerhalb der gewohnten PC-Arbeitsplatzumgebung unter Einsatz bereits existierender MS-Office-Applikationen und dem Desktop-Mapping-Tool ArcView stattfinden. Diese möglichst automatische Integration in die entsprechend auf dem PC installierten Applikationen erfordert die Entwicklung eines intelligenten UFIS-Client, der dem Anfragenden die Entscheidung abnimmt welche Applikationen zur Aufbereitung herangezogen werden sollen. Darüber hinaus müssen jedoch auch serverseitig Aufbereitungdienste zur Verfügung stehen um auch von einer minimal konfigurierten Client-Station einen Zugang zu UFIS-Daten und eine aussagekräftige Ergebnispräsentation zu ermöglichen. Das System UFIS2 läßt sich daher in 3 Schichten strukturieren :

- Recherche (Auffinden themenspezifischer Informationsquellen)
- Selektion (Selektion der Rohdaten)
- Auswertung, Integration und Aufbereitung (Visualisieren der abgefragten Daten)

Hierbei muß die Recherche ein Auffinden sämtlicher Informationen eines verteilten Informationsverbundes anhand ihrer beschreibenden Attribute gewährleisten und anschließend den direkten Zugang zu den Informationsquellen ermöglichen. Für den Durchgriff auf die Daten ist die anbietende Stelle selbst verantwortlich und muß hierfür entsprechende Formulare zur Verfügung stellen. Im Falle von UFIS endet die Recherche mit der Auflistung der themenspezifischen Selektoren, die einen Zugang zu den Abfrageselektoren für Rohdaten beim UFIS-Datenserver mittels „Mausklick“ ermöglichen. Beim Abruf der Selektorformulare vom Datenserver sollten die bereits in der Recherche spezifizierten Attribute soweit möglich übernommen werden. Hierzu müssen spezielle Schnittstellen generiert werden. Nach erfolgter Selektion der gewünschten Daten erfolgt die Integration und Aufbereitung auf dem UFIS-Client abhängig von der Semantik der Rohdaten. Zusätzlich zu den Selektionsdaten sind daher Beschreibungsinformationen notwendig, die auf den PC-Client transferiert werden. Dort wertet eine intelligente Software (Decision Manager) die Beschreibungsinformation aus und listet die für die Daten möglichen Darstellungen auf. Nach erfolgter Auswahl einer Visualisierungsart werden die Rohdaten. in die, mittels speziellen Macros und Skripten erweiterten Applikationen integriert und aufbereitet.

1.2.2 Die Systemarchitektur von UFIS II

Die Umsetzung des Drei-Schichten-Modells erfordert eine Neukonzeption der Systemarchitektur, aufbauend auf den Ergebnissen der Projekte INTEGRAL und GLOBUS und dem Landessystemkonzept. Das neue System basiert auf einer verteilten Client- / Serverarchitektur ,die moderne Telekommunikationstechniken wie WWW und Internet bzw. Intranet nutzt. Eine mittels WWW konzipierte Systemarchitektur bietet den Vorteil, das sie auf allen TCP/IP-fähigen Netzen lauffähig ist und einen ortsunabhängigen Zugang zu Daten und Diensten, die mit einer entsprechenden Schnittstelle ausgestattet sind, ermöglicht. Server- und Clientsoftware für einen aktiven und passiven Zugang zum Internet stehen derzeit für fast alle Plattformen zur Verfügung und garantieren somit einen plattformunabhängigen Durchgriff auf alle im Netz zugänglichen Informationen. Abbildung 1 veranschaulicht die Systemarchitektur von UFIS II und die Umsetzung der drei Schichten.

Wichtigste Komponente der neuen Systemarchitektur stellt der UFIS-Client dar, der mittels eines WWW-Browsers, z.B. Netscape, MS-Explorer Zugang zu einem Metainformationssystem und den einzelnen Informations- bzw. Datenservern sowie Diensteservern ermöglicht. Für die Aufbereitung der Daten stehen MS-Office Applikationen und ArcView zur Verfügung. Zusätzlich wird der PC-Client mit einer Software ausgestattet, welche die PC-Konfiguration verwaltet und abhängig von den selektierten Daten die vorhandene Applikation aufruft und somit die Aufbereitung der Daten steuert. Zentrale Komponente des Systems bildet ein Metainformationssystem (MIS), über das der Anfragende von seinem PC-Client mittels des WWW-Browsers zugreifen kann. Dieses MIS dokumentiert sämtliche in diesem verteilten Informationsverbund vorliegenden Daten mittels Attributen. Diese Attribute unterteilen einen in für die Recherche notwendige Beschreibungselemente wie Raumbezug, Zeitbezug, datenhaltende Stelle, Art der Information und Sachbezug. Für die Zuordnung der Informationen zu einem Sachbezug und die anschließende Auswahl für den Anfragenden wird der UDK-Thesaurus als Basis verwendet. Andererseits beinhalten die Attribute des MIS Zusatzinformationen, die dem Anfragenden eine detaillierte Beschreibung der recherchierten Daten liefert. Dadurch wird eine Entscheidungshilfe geliefert, inwieweit die gefundenen Informationsquellen mit ihren Daten den Anforderungen des Anfragenden gerecht werden, bevor eine Durchgriff zur datenhaltenden Stelle stattfindet. Auf Grund derzeitiger Netzgeschwindigkeiten kann dieser unter Umständen längere Zeit in Anspruch nehmen, was bei einem nicht zufriedenstellenden Ergebnis die Bearbeitungszeit unnötig verlängern und die Akzeptanz des Systems erheblich mindern würde. Die Metadaten der Informationsquellen sind die einzigen Informationen, die in der neuen Architektur zentral gehalten werden, wobei dieses MIS nur einen minimalen Satz der beschreibenden Attribute der einzelnen Systeme beinhaltet. Dies verringert Redundanzen in der Datenhaltung und minimiert den Erfassungs- und Pflegeaufwand.

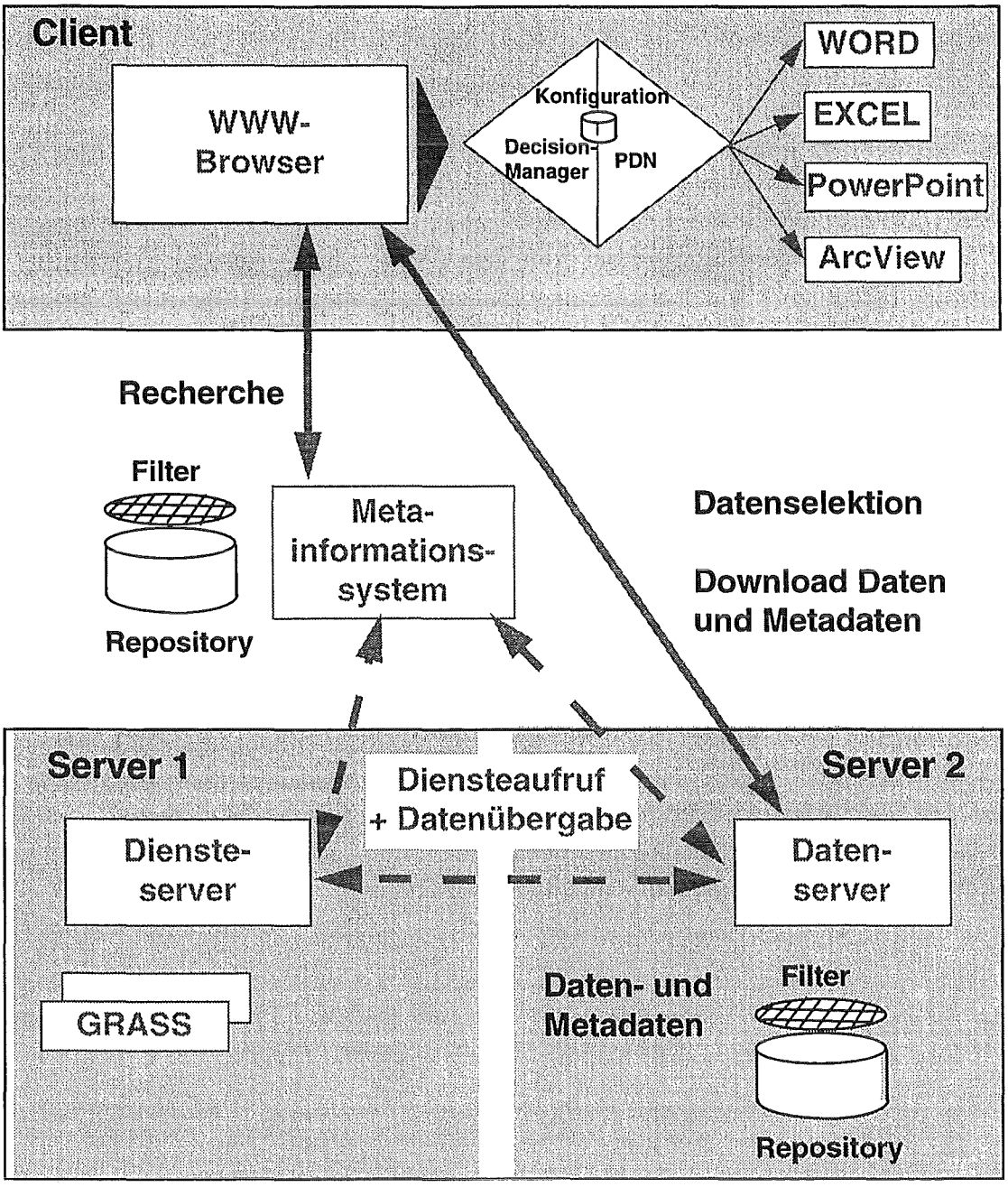

Client
WWW-
Browser
Konfiguration
Decision-
Manager
PDN
WORD
EXCEL
PowerPoint
ArcView
Recherche
Filter
Repository
Meta-
informations-
system
Datenselektion
Download Daten
und Metadaten
Server 1
Diensteaufruf
+ Datenübergabe
Server 2
Dienste-
server
Daten-
server
GRASS
Daten- und
Metadaten
Filter
Repository

Abbildung 1

Greift ein Benutzer nach erfolgter Recherche für eine Datenabfrage auf die datenhaltende Stelle durch, erfolgt ein für den Benutzer nur durch sich eventuell änderndes Seitenlayout ersichtlicher Wechsel vom zentralen Metainformationsserver zu den Datenservern. Die Daten- und Diensteserver sind dezentral in Netz verteilt und nur mittels der Beschreibungsdaten im MIS zu einem recherchierbaren Verbund gekoppelt. Die Verantwortung für die Daten und die entsprechenden WWW-Schnittstellen auf die der Anfragende zur Datenabfrage direkt zugreift liegen daher in der Eigenverantwortung des Datenanbieters,. Nach erfolgter Selektion kann der Benutzer die bis dahin nur in tabellarischer Struktur vorliegenden Daten in verschiedener Form aufbereiten, z.B. Karten, Säulendiagramme, Balkendiagramme oder Tortendiagramme. Für Client-Stationen, die eine solche Aufbereitung aus technischen Gründen nicht ermöglicht, stehen Aufbereitungsdienste zur Verfügung, die auf ebenfalls verteilten Diensteservern vorliegen. Die Eingabe- und Ausgaberäume dieser Aufbereitungsdienste müssen in einem zentralen Dienstерepository beschrieben sein, damit ein intelligenter Auswertemechanismus dem Benutzer gemäß der Semantik der Rohdaten und der Eingaberäume der Dienste eine Auswahl der geeigneten Dienste anbieten zu können. Anschließend erfolgt der Diensteaufruf mit entsprechender Datenübergabe. Für die UFIS-Anwender wird allerdings von einem PC-Client mit sämtlichen für die UFIS-Aufbereitung notwendigen Konfigurationen ausgegangen. Erfolgt eine Abfrage von einem so konfigurierten Arbeitsplatz, werden die Daten auf den Client „heruntergeladen“ und ausgewertet, und in einer aussagekräftigeren Form aufbereitet. Hierfür werden in erster Linie die MS-Office Applikationen und ArcView verwendet, die auf den PC-Arbeitsplätzen der Sachbearbeiter der Ministerien vorliegen. Die Rohdaten können abhängig von deren Inhalt in unterschiedlichen Formen visualisiert werden.

1.2.3 Datenselektion und Ergebnisaufbereitung

Das System UFIS II sollte für einen Umweltgeneralisten nach Möglichkeit einer Information auf Knopfdruck nahekommen. Deshalb wurde bei der Systemkonzeption die Menge der Benutzerinteraktionen durch beschreibende Zusatzinformationen bei den Daten und intelligenten Softwarekomponenten auf dem Client weitestgehend eingeschränkt. Folgendes Schaubild zeigt die Benutzerinteraktionen und die im Hintergrund stattfindenden Systemabläufe beim Durchgriff auf den UFIS-Datenserver.

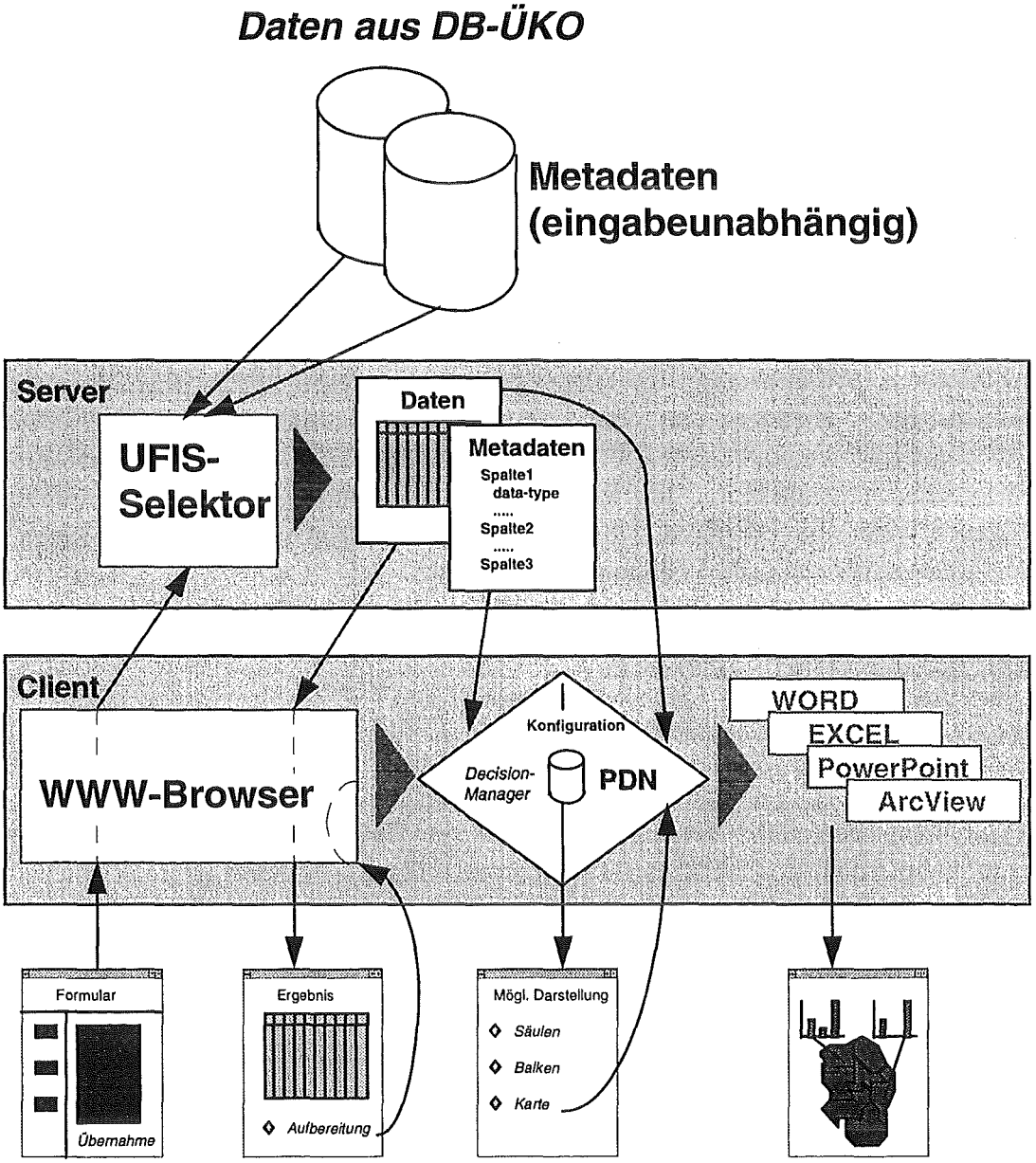

Daten aus DB-ÜKO
Metadaten
(eingabeunabhängig)
Server
UFIS-
Selektor
Daten
Metadaten
Spalte1
data-type
.....
Spalte2
.....
Spalte3
Client
WWW-Browser
Konfiguration
Decision-
Manager
PDN
WORD
EXCEL
PowerPoint
ArcView
Formular
Übernahme
Ergebnis
Aufbereitung
Mögl. Darstellung
Säulen
Balken
Karte
Benutzerinteraktionen

Abbildung 2

Nach erfolgter Recherche und dem Durchgriff auf einen UFIS-Selektor des UFIS-Datenservers erhält der Benutzer ein Abfrageformular, in dem er die für eine Selektion der Daten notwendigen Parameter spezifizieren kann. Aktiviert der Anfragende den Button „Selektor starten", wird auf dem Datenserver eine parametrisierte Datenbankabfrage angestoßen. Abhängig von den spezifizierten Parametern und der Semantik der Rohdaten werden beschreibende Informationen generiert und abgespeichert. Die Ergebnisdaten werden in ihrer originalen Tabellenstruktur dem Benutzer auf einer HTML-Seite visualisiert. Aktiviert der Benutzer die Aufbereitung, werden die Metadaten auf den Client transferiert und einem Personal Data Node (PDN) übergeben. Der PDN beinhaltet einen Decision Manager, der für die Entscheidungsfindung der möglichen Aufbereitungen verantwortlich ist. Der PDN verwaltet die PC-Konfiguration und ist für das Transferieren der Daten sowie für den Applikationsaufruf mit den Erweiterungsskripten verantwortlich. Die beschreibenden Daten werden somit vom Decision Manager auf dem Client analysiert und resultierend die möglichen Visualisierungen angeboten. Diese Ergebnismenge der Darstellungsarten reduziert sich abhängig von der auf dem Client vorhandenen Konfiguration, d.h. nur wenn die erforderlichen Applikationen und Zusatzskripte vorhanden sind, wird die entsprechende Darstellungsart aufgelistet und dem Benutzer zur Auswahl gestellt. Die Auswahl einer Aufbereitungsart veranlaßt den PDN, die Daten auf den Client zu transferieren und die entsprechende Applikation mit den Erweiterungsmacros und -skripten aufzurufen. Im folgenden wird nun detailliert die Interaktion des Benutzers mit dem System UFIS II beschrieben sowie die dafür benötigten Programme.

1.3 Selektoren UFIS II

1.3.1 Überblick

Im Rahmen von UFIS II wurden für die in UFIS I erstellten Selektoren eine neue benutzerfreundlichere Oberfläche konzipiert und umgesetzt. Diese neue Oberfläche basiert auf dem Konzept der sog. „Frames", die ab der Version 2 des Netscape-Browsers verfügbar sind. Mit diesen Frames ist es möglich, das Browserfenster in verschiedene Bereiche aufzuteilen, so daß gleichzeitig mehrere WWW-Seiten dargestellt werden können.

Des weiteren wurde die neue Selektor-Oberfläche um eine Hilfefunktionalität erweitert, die zum einen die Hilfetexte des ursprünglichen UFIS integriert, und zum anderen zusätzlich dynamisch generierte Informationen aus der Datenbank umfaßt.

Für die Realisierung der framebasierten Oberfläche wurden zusätzliche Programme für die „Kommunikation" zwischen den Frames entwickelt. Neben diesen Programmen fand das im Rahmen von UFIS I entwickelte Basistool „WebQuery" (s. Abschlußbericht GLOBUS II) zur Anbindung von Datenbanken an WWW-Anwendungen Verwendung.

Ebenfalls erweitert wurden die Möglichkeiten zur Aufbereitung der aus einer Datenbank-Abfrage erstellten Ergebniswerte. Dazu wird bei jeder Abfrage eine Metabeschreibung (in Form einer SGML-Datei) erstellt, die eine Interpretation der selektierten Daten erlaubt; diese Metabeschreibung wird vom Personal Data Node (PDN) verwendet.

1.3.2 Selektoren aus Benutzersicht

Für jeden Selektor wird die Bildschirmseite in drei Bereiche aufgeteilt: ein Frame mit den Funktionsbuttons, ein Frame mit dem aktuellen Selektorformular und ein Aktionsfenster, in dem die Parameter für das Suchformular eingestellt werden können und in dem später das Abfrageergebnis angezeigt wird. Durch diese Aufteilung wird gewährleistet, daß dem Benutzer zu jeder Zeit sämtliche Funktionen zur Verfügung stehen, ohne daß er in den Seiten zurückblättern muß; gleichzeitig ist er durch die Anzeige des Suchformulars immer über die aktuell eingestellten Parameter informiert.

Die folgende Abbildung zeigt die Aufteilung des Browserfensters in verschiedene Frames am Beispiel des Selektors „Verkehr".

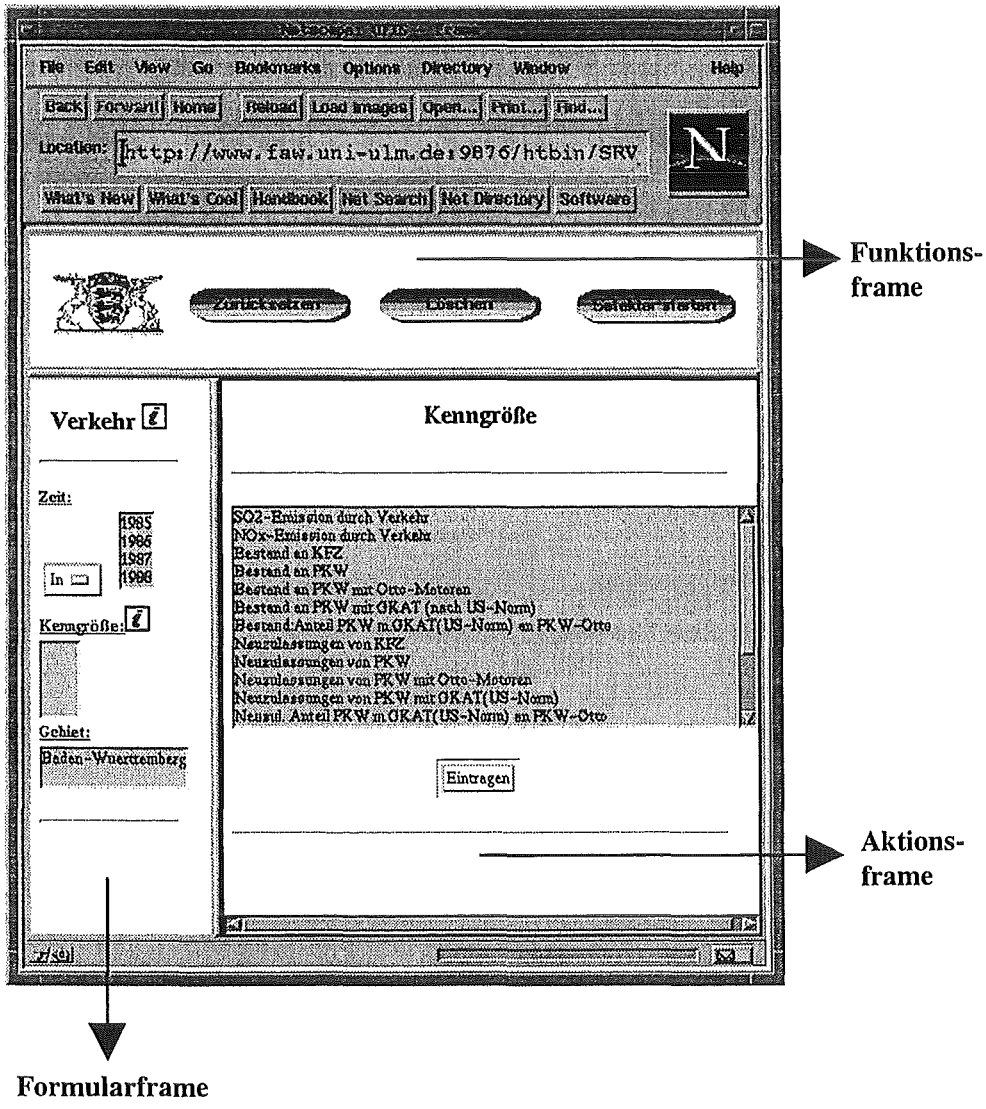


Abbildung 3

Der Ablauf bei der Formulierung einer Abfrage sieht typischerweise so aus, daß nach dem Anzeigen der Einstiegsseite für einen Selektor eine der sensitiven Überschriften im Formularframe angeklickt wird, z. B. die Kenngröße. In der daraufhin im Aktionsframe angezeigten Listbox können die gewünschten Werte selektiert werden, bestätigt werden sie

durch „Eintragen". In derselben Weise werden auch in die anderen Felder des Auswahlformulars Werte eingetragen. Wenn alle benötigten Werte ausgewählt wurden, kann im Funktionsframe „Selektor starten" aufgerufen werden; die UFIS-Abfrage wird dann mit den im Formularframe angezeigten Parametereinstellungen durchgeführt. Werden für einen Parameter keine Werte ausgewählt, entspricht das dem Parameter „Alle". Dies bedeutet für die Abfrage, daß hinsichtlich dieses Parameters keine Einschränkungen vorgenommen werden.

Anschließend wird im Aktionsframe die Anzahl der gefundenen Sätze für diese Abfrage angezeigt, es kann nun entweder das eigentliche Suchergebnis angezeigt werden, oder auch die Abfrage neu spezifiziert werden durch Setzen weiterer Parameterwerte.

Nachdem die Ergebnisdaten tabellarisch angezeigt wurden, können diese selektierten Daten weiter aufbereitet werden. Hierzu werden sie auf den Client transferiert und in Applikationen wie z. B. Excel oder ArcView integriert.

Hilfetexte zu den Selektoren bzw. zu einzelnen Parametern können angezeigt werden, indem das entsprechende Hilfesymbol angeklickt wird; die Anzeige erfolgt immer in einem neuen Browser-Fenster. Im oben abgebildeten Frame kann z. B. durch Anklicken des Symbols neben der Überschrift „Verkehr" eine allgemeine Information zu diesem Selektor abgerufen werden, der aus einem Hilfetext besteht. Hilfe zu den einzelnen Kenngrößen erhält man durch Anklicken des Symbols neben der sensitiven Überschrift. Diese Hilfe umfaßt einen Hilfetext und aus der Datenbank generierte Information (z. B. für welchen Zeitraum zu einer bestimmten Kenngröße Werte in der UFIS-DB vorhanden sind).

1.3.1 Programmbeschreibungen

Für alle Datenbankabfragen wird „WebQuery" eingesetzt (s. Abschlußbericht GLOBUS II) und für die Kommunikationen zwischen den Frames die unten beschriebenen Shell-Skripte bzw. Perl-Programme.

Von welchen WWW-Seiten aus welche Funktionen aufgerufen werden können, ist der nachfolgenden Abbildung zu entnehmen.

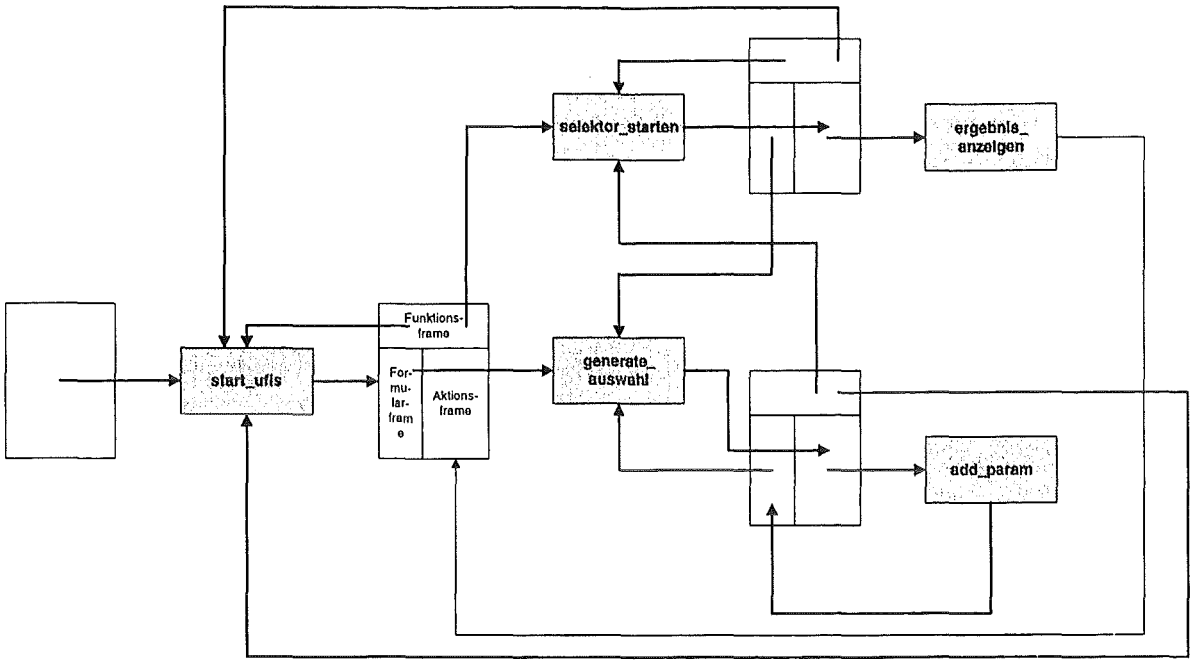


Abbildung 4

Beim Aufruf eines Selektors wird das Skript „start_ufis" aufgerufen, welches die erste Seite mit Frames generiert. Von dieser ersten Seite aus kann vom Formularframe aus das Skript „generate_auswahl" gestartet werden, und vom Funktionsframe aus das Skript „selektor_starten" bzw. „start_ufis". „generate_auswahl" listet im Aktionsframe die zum Setzen des jeweiligen Parameters möglichen Parameterwerte auf, von dort aus kann dann das Skript „add_param" aufgerufen werden. Nach wie vor bleiben die Funktionalitäten der beiden anderen Frames jederzeit aufrufbar. Wird „selektor_starten" aufgerufen, zeigt der Aktionsframe anschließend die Anzahl der gefundenen Ergebnissätze an mit der Möglichkeit, „ergebnis_anzeigen" zu starten, wodurch das Ergebnis der Datenbankabfrage im Aktionsframe tabellarisch aufgelistet wird. Beim Aufruf von „start_ufis" aus dem Funktionsframe wird, wie beim ersten Aufruf auch, die Seite mit Frames neu generiert (d. h. alle aktuell gesetzten Attribute werden zurückgesetzt).

Im folgenden werden die einzelnen Programme detaillierter beschrieben.

- start_ufis

  „start_ufis" generiert alle für einen bestimmten Selektor benötigten Frame-Seiten, d. h. das für diesen Selektor benötigte Auswahlformular (das im Formularframe angezeigt wird) und den Frame mit den Funktionsbuttons, und legt sie in einem temporären Verzeichnis ab. Dazu muß dem Skript ein Kurzname des Selektors als Parameter übergeben werden. Gleichzeitig werden in diesem Skript bereits die für das Starten des Selektors mit „WebQuery" benötigten Parameter „queryfile" und „resultfile" festgelegt (d. h. im temporären Verzeichnis werden die Dateien „queryfile.par" und „resultfile.par" angelegt, die die kompletten Dateinamen enthalten).

- generate_auswahl

„generate_auswahl" generiert für jeden Parameter eines Selektors (z. B. Gebiet oder Kenngröße) die entsprechende HTML-Seite, mit der die Werte für diesen Parameter ausgewählt werden können. Dabei kann es sich um statische Seiten mit bereits vorher feststehenden Parameterwerten handeln oder um dynamisch erzeugte Seiten, die die aktuellen Parameterwerte aus einer Datenbankabfrage erhalten. Das Skript benötigt als Übergabeparameter die Variable „QUERY_STRING", die den Kurznamen des Selektors und den Namen des Parameters enthalten muß. Es prüft zunächst, ob die HTML-Seite zum Setzen dieses Parameters bereits im temporären Verzeichnis existiert. Falls nicht, wird sie entweder dorthin kopiert (falls es sich um eine statische Seite handelt), oder es wird „WebQuery" gestartet, um die Auswahlwerte aus der Datenbank auszulesen. Anschließend wird die HTML-Seite im Aktionsframe angezeigt.

- add_param

  Werden im Aktionsframe Werte für einen Parameter ausgewählt, müssen diese in das Auswahl-Formular und in Parameterdateien (z. B. Gebiet.par, Zeit.par) übernommen werden. Diese Aufgabe übernimmt „add_param". Dazu muß dem Programm die Variable „QUERY_STRING" mit den Parameterwerten und das Verzeichnis übergeben werden, in dem die Dateien mit den Parameterwerten abgelegt werden sollen. Im Auswahl-Formular werden die Parameter im Klartext eingetragen (z. B. der Kenngrößenname), in den Parameterdateien in der Form, in der sie später im SQL-Statement zur Datenbankabfrage benötigt werden (z. B. die Kenngrößennummer). Es können gleichzeitig verschiedene Parameter übergeben werden, auch mit jeweils mehreren Werten (die Reihenfolge im QUERY_STRING spielt dabei keine Rolle).

- selektor_starten

  „selektor_starten" kann zu jedem Zeitpunkt vom Funktionsframe aus aufgerufen werden und startet dann die UFIS-Datenbankabfrage mit den aktuell gesetzten Parameterwerten (so wie sie im Auswahl-Formular angezeigt sind). Dazu werden von „selektor_starten" die Module „makesel" und „sqlquery" des Programmes „WebQuery" aufgerufen. Den für diesen Aufruf benötigten QUERY_STRING setzt ein Perl-Programm („set_query_string.pl") aus den Parameterdateien des temporären Verzeichnisses zusammen. Nach erfolgter Datenbankabfrage generiert „selektor_starten" eine Ergebnisseite, die die Anzahl der gefundenen Sätze im Aktionsframe anzeigt.

- ergebnis_anzeigen

  „ergebnis_anzeigen" wird gestartet, wenn das Suchergebnis einer Datenbankabfrage in einer HTML-Seite angezeigt werden soll. Dazu ruft „ergebnis_anzeigen" das Modul „htmlaufb.pl" von „WebQuery" auf, welches die Datensätze in HTML-Darstellung konvertiert. Anschließend wird die generierte HTML-Seite im Aktionsframe angezeigt.

Die folgende Abbildung gibt einen Überblick über die wichtigsten von den einzelnen

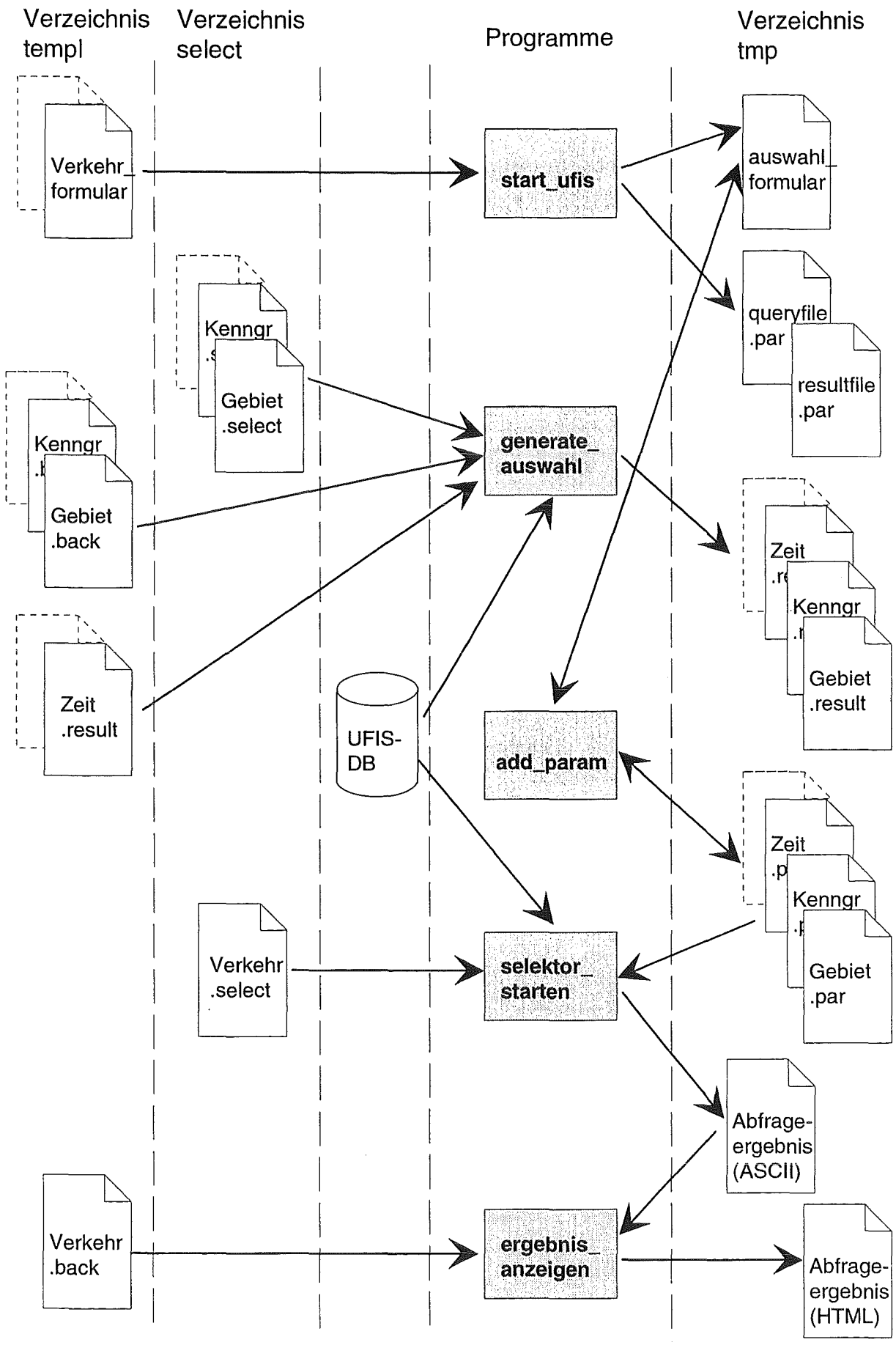

Verzeichnis templ
Verzeichnis select
Programme
Verzeichnis tmp
Verkehr_ formular
start_ufis
auswahl_ formular
Kenngr
Gebiet .select
queryfile .par
resultfile .par
Kenngr
Gebiet .back
generate_ auswahl
Zeit .result
Zeit
Kenngr
Gebiet .result
UFIS-DB
add_param
Zeit
Kenngr
Gebiet .par
Verkehr .select
selektor_ starten
Abfrage-ergebnis (ASCII)
Verkehr .back
ergebnis_ anzeigen
Abfrage-ergebnis (HTML)

Abbildung 5

2. Beschreibung der Metadaten

2.1 Überblick

Das Ziel des hier beschriebenen Ansatzes ist die Definition einer Meta-Struktur zur weitergehenden Nutzung von Diensten im Rahmen des UIS. Dabei soll versucht werden, für alle im UIS vorkommenden Dienste bzw. deren Parameter eine einheitliche logische Meta-Beschreibung zu erarbeiten. In dieses Dienste-Schema (*logisches Dienste-Repository*) sollen alle Dienste einschließlich ihrer Schnittstellenbeschreibungen sowie dem zugrundeliegenden Kontext (Semantik eines Dienstes) eingetragen und beschrieben werden können. Dieses logische Dienste-Repository stellt somit vor allem die fachliche Basis für Kopplungsfunktionalitäten zwischen mehreren Diensten dar.

Der hier beschriebene Ansatz erhebt nicht den Anspruch eines technischen Dienste-Repositories wie es z.B. im Rahmen der CORBA-Architektur Verwendung findet. Vielmehr geht es hier um eine semantische Ergänzung der Dienste-Beschreibungen, die über eine reine Schnittstellenbeschreibung wie es z.B. im Rahmen der CORBA-Architektur mit Hilfe der IDL möglich ist, hinausgehen soll. Somit soll nicht nur die transparante Ausführung von Diensten wie sie in verteilten Umgebungen mit Hilfe von Middleware-Komponenten möglich ist, gewährleistet werden, sondern es soll eine semantisch (und technisch) sinnvolle Zusammenarbeit von Diensten erreicht werden.

Um eine Zusammenarbeit von Diensten im Sinne der Weiterver(be)arbeitung von Ergebniswerten zu erreichen, sollen die Dienste-Beschreibungen für die Generierung von Meta-Daten Verwendung finden. Diese Meta-Beschreibungen müssen von den Diensten generiert werden. Die dafür notwendigen Daten stehen, soweit sie statisch für die jeweiligen Dienste beschreibbar sind, im Repository zur Verfügung. Teile der Meta-Beschreibungen müssen u.U. zur Laufzeit dynamisch vom Dienst generiert werden (z.B. Anzahl der Ergebnisdaten). Dies ist u.a. dann der Fall, wenn fachliche Abhängigkeiten bzgl. der Instanzierung des Eingaberaums zur Instanzierung des Ausgaberaums vorhanden sind, welche wiederum die Semantik (evtl. Verwendbarkeit oder Interpretation) der Ausgabedaten des Dienstes beeinflussen.

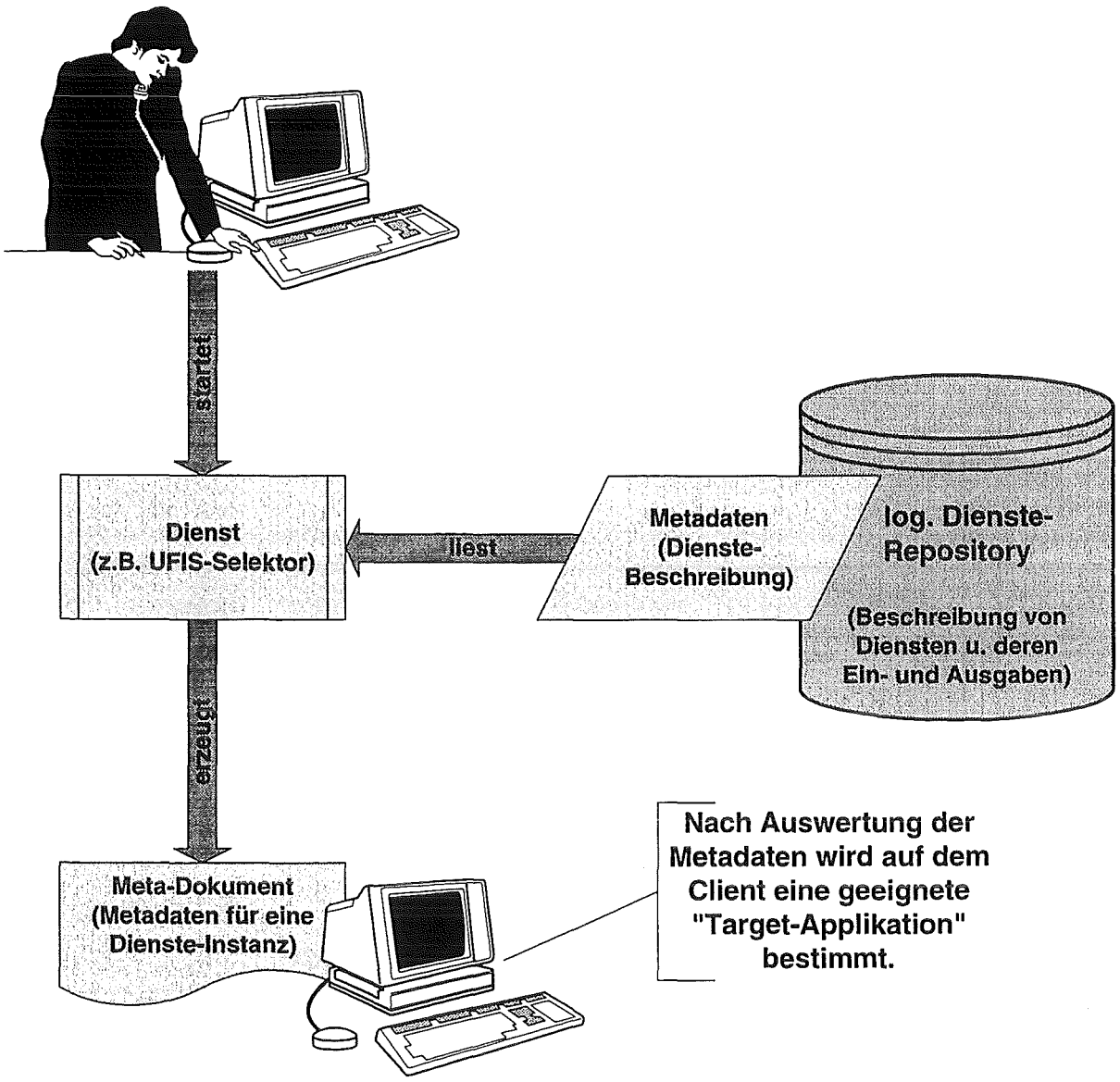


Abbildung 6

2.2 Metadaten-Beschreibung mit SGML

2.2.1 Einführung

Die Beschreibung der Metadaten zu den Ausgaberäumen von Diensten muß in einer einheitlichen Form dargestellt werden können. SGML bietet bereits die dafür notwendigen Sprachmittel, sowie Werkzeuge wie z.B. einen Parser, der einen Überprüfung der Syntax gewährleistet und den Zugriff auf SGML-Daten gestattet.

Mit der *Standard Generalized Markup Language* (SGML) stehen die grundlegenden Strukturen und Sprachmittel zur Verfügung, um eine Beschreibungssprache für Meta-Daten zu definieren. Bei SGML handelt es sich um eine abstrakte Sprache zur Beschreibung von Dokumenten, ihres Aufbaus und Inhalts. Dabei spielt das Layout, also die optische Strukturierung, keine Rolle. SGML kennt keine vorgegebene Dokumentenstruktur, sondern der Anwender muß sich diese selbst definieren. Eine solche selbstdefinierte Dokumenten-

struktur heißt *Document Type Definition* (DTD). Sie bildet zusammen mit einem Text (Instanz), der diesem Typ entspricht, das SGML-Dokument.

Meta-Daten werden als Dokumente, die Daten beschreiben, aufgefaßt. Zur Beschreibung von Meta-Daten wird jedoch z.Zt. nur eine kleine Untermenge der Sprachkonstrukte, die SGML zur Verfügung stellt, benötigt.

Verwendet werden 4 Sprachelemente der SGML: *Dokumenttyp*, *Elemente, Attribute* und *Kommentare*. Der Dokumenttyp gibt an, um welche Klasse von Metadaten es sich handelt. Mit den Elementen wird die Struktur einer Meta-Daten-Klasse gebildet, die Attribute sind Parameter der Element-Instanzen, die an den Generator weitergereicht werden.

2.2.2 Instanz einer SGML-Meta-Beschreibung

Es sollen folgende Ergebnisdaten (Auszug aus einer UFIS-Abfrage) beschrieben werden:

VE_KENN GROESSE N_NR	DIMENSI ONS_NR	ZEITTY P	MESSOBJE KT	ORTSTY P	WERT	KURZNA ME	ZEIT
168	107	JJJJ	--	VE	28435	t	1989
168	107	JJJJ	--	VE	23514	t	1992
168	107	JJJJ	--	VE	22894	t	1993
168	107	JJJJ	--	VE	50043	t	1989
168	107	JJJJ	--	VE	48607	t	1992
168	107	JJJJ	--	VE	28717	t	1993
168	107	JJJJ	--	VE	21865	t	1989
168	107	JJJJ	--	VE	18764	t	1992
168	107	JJJJ	--	VE	19629	t	1993
168	107	JJJJ	--	VE	36873	t	1989
168	107	JJJJ	--	VE	33434	t	1992
168	107	JJJJ	--	VE	28039	t	1993

Die Beschreibung dieser Ergebnisdaten liefert (beispielhaft an zwei Spalten) folgendes Ergebnis:

```
<!DOCTYPE meta-dat "meta-dat.dtd">
```

```
<meta-dat>
      <general>
            <name>Meta-Daten UFIS-Abfallaufkommen absolut
            <service> UFIS-Standardselect
            <content-typ> application/pdn-data
      </general>
      <access transfer-typ="HTTP">
            <data-server>www.faw.uni-ulm.de
            <server-port>9876
            <data-path>integral_bin/
      </access>
      <archiv>
            <archiv-file>abfallab.apz
            <archiv-size datatyp = "UINT">65423
            <packer-typ>UNZIP
      </archiv>
      <data format = „TXT" charset = „A7">
            <filename>Abf9759.txt
            <data-create-time>20.08.96 13:34
            <data-comment>Abfallaufkommen absolut
            <data-size datatyp = "UINT">720876
            <data-structure>
                  <table column-sep= „TAB" text-ident = „NONE">
                        <eol-char>CR
                        <eof-char>EOF
                        <num-datasets>12
                        <col-def-row>1
                        <table-semantics>
                              <representation-level>Kreisebene
                              <whole-dimension>Baden-Württemberg
                        </table-semantics>

1. Spalte:
                        <column  value-typ= „NUMBER"  presentation = „YES">
                              <name>VE_KENNGROESSE
                              <list-number>1
                              <ui-name>ABFALLART
                              <value-set>
                                    <num-of-dif-values>1
                              </value-set>
                              <key-def  mode = „INSTEAD-OFF">
                                    <allocation>
                                          <allocated-
column>VE_KENNGROESSEN_NAME
                                          <allocated-values>
                                                <key-value>168
                                                <allocated-value>Hausmüll-
Aufkommen
                                          </allocated-values>
                                    </allocation>
                              </key-def>
                              <semantics>
                                    <field-ref>
                              </semantics>
                        </column>
...

5. Spalte:
                        <column  value-typ= „CHAR"  presentation = „YES">
                              <name>ORTSTYP
                              <list-number>5
                              <replacement>
                                    <old-content>VE
                                    <new-content>Verwaltungseinheit
                              </replacement>
                              <value-set>
                                    <num-of-dif-values>1
```

```
                    </value-set>
                    <semantics>
                        <info>
                    </semantics>
                </column>
...
```

Für eine erste Version des PDN wird ein notwendiger Auszug der Metadaten-Instanz als Zeichenkette auf den Client übertragen. Dort werden die notwendigen Auswertungen anhand einer Entscheidungstabelle getroffen.

3. Konzeption eines Gazetteers zum raumbezogenen Zugriff auf Umweltinformationen

3.1 Aktueller Stand

Im derzeitigen Realisierungsstadium des Gazetteers werden Raumbezüge als eine Folge von Attribut/Wert-Tupeln abgelegt, wobei folgende Attribute einen Raumbezug beschreiben:

- Raumbezugs-ID
- Gebietsnamen
- xmin-Koordinate
- xmax-Koordinate
- ymin-Koordinate
- ymax-Koordinate
- Teil_von - Referenz

Mit einer sinnvollen Belegung des „Teil_von"-Attributes läßt sich eine Hierarchie von Raumbezügen aufbauen, wobei allerdings eine Einschränkung dahingehend getroffen worden ist, daß ein Knoten (Raumbezug) innerhalb dieser Hierarchie nur einen Vorgänger (übergeordneten Raumbezug) besitzen kann. Diese Anordnung von Gebieten innerhalb einer hierarchischen Ordnung bietet einige Vorteile sowohl bei der Zuordnung von Informationsobjekten zu Raumbezügen, als auch bei der Selektion bzw. der Einschränkung des Suchraumes der Raumbezüge bei der Suche nach Informationsobjekten.

Der Nachteil der derzeitigen Repräsentationsmethode besteht also weniger in der Ordnung der Raumbezüge untereinander, als vielmehr in der groben, auf eine geometrische Form beschränkten Darstellungsmöglichkeit des Gebietes als solches. Mit den vier Koordinaten des Raumbezugs sind momentan nur „Bounding Boxes", also umschließende Rechtecke, darstellbar. Die Granularität der Raumbezüge ist in Folge dessen oft viel zu grob bzw. ungenügend hinsichtlich der wahren Geometrie des abzubildenden Gebietes. Soll z.B. eine Stadt als Raumbezug abgelegt werden, so ist es immer notwendig, ein umschließendes Rechteck, also vier Koordinaten anzugeben, welche die Stadt in ihrer maximalen Ausdehnung beschreiben. Dies führt natürlich dazu, daß Randzonen, die nicht mehr dazugehören, trotzdem im Raumbezug mit eingeschlossen sind. Dieses Problem verschärft sich noch für den Fall, wenn Geoobjekte mit linearer Ausdehnung wie z.B. Straßen oder Flüsse als Raumbezüge abgebildet werden sollen.

3.2 Geplante Konzeption

3.2.1 Rasterbildung

Um das oben genannte Problem in den Griff zu bekommen, ist es also notwendig, eine wesentlich feinere Granularität von Raumbezugseinheiten zu wählen. Eine zielführende Vorgehensweise ergibt sich aus der Aufrasterung von Bezugsgebieten (z.B. Baden-Württemberg oder der BRD) in sogenannte „Rasterzellen" mit einer bestimmten Längen- und Breitenausdehnung. Die Rasterzellen als solche sind wiederum zu verstehen als kleine Quadrate, die mit entsprechenden Koordinaten-Attributen versehen werden können.

3.2.2 Problem der Rasterweite

Bei der Wahl der Längen- und Breitenausdehnung der Rasterzellen ist es notwendig, sich auf ein bestimmtes, sinnvolles und, vor allem im Hinblick auf die Menge der abzuspeichernden Daten, handhabbares Maß festzulegen. Wird z.B. die Größe der Rasterzellen auf je ein Kilometer Längen- und Breitenausdehnung festgelegt, so ergeben sich bei einem aufzurasternden Gebiet mit einer Ausdehnung von 200 mal 200 Kilometern bereits 40.000 Rasterzellen, die abgelegt und referenziert werden müssen. Auch das oben angesprochene Problem der genauen Abbildung der Gebietsgeometrie ist mit der Einführung von Rasterzellen noch nicht gänzlich behoben, jedoch in einem weitaus geringerem Ausmaß vorhanden. Die Genauigkeit ist dabei umgekehrt proportional zur Dimensionierung der Rasterzellen.

Aufgrund der Heterogenität der Raumbezüge („Baden-Württemberg" oder „Biotop xy") erscheint es nicht sinnvoll, alle Raumbezüge auf ein einheitliches Raster abzubilden. Je nach Aufgabe sollte eine andere Auflösung gewählt werden können. Es ist geplant, Raster in der Größe von 1 km, 5 km und 10 km anzubieten, wobei die Information, welcher Raumbezug in welchem Raster vorliegt, in einer eigenen Tabelle abgelegt ist.

3.2.3 Datenmodell der Rasterung

Wird eine solche Vorgehensweise gewählt, so hat dies auch Auswirkungen auf die Repräsentation von Raumbezügen innerhalb des Gazetteers. In diesem Fall wäre es also nicht mehr möglich, alle Koordinaten des Gebietes innerhalb des Raumbezugs selbst anzugeben. Vielmehr müßte eine eigene Tabelle „Rasterzelle" existieren, die alle vorhandenen Rasterzellen repräsentiert. Diese Tabelle könnte folgende Attribute besitzen:

- Rasterzellen_ID
- xmin-Koordinate
- ymin-Koordinate
- xmax-Koordinate
- ymax-Koordinate

Erweist es sich als nicht notwendig, daß die Rasterzellen mit ihren Koordinaten abgelegt werden, so wäre es auch denkbar und vor allem hinsichtlich der Speichermenge optimierbar,

die Rasterzellen nur mit ihren absoluten Indizes im Rahmen des aufgerasterten Gebietes abzulegen. Stellt man sich das aufgerasterte Gebiet vor wie eine Tabelle, in der die Tabellenzellen (Rasterzellen) absolute Adressen hinsichtlich eines Bezugspunktes erhalten, so könnte sich folgende Darstellungsmöglichkeit ergeben:

- Rasterzellen_ID (evtl. nicht notwendig, da x- und y-Wert der Schlüssel wären)
- Zeile (y-Wert)
- Spalte (x-Wert)

Denkbar wäre auch eine Kombination beider Möglichkeiten, wobei die zweite Möglichkeit vor allem bei der (grafischen) Definition des Gebietes im Rahmen der Suche nach Informationsobjekten Verwendung finden könnte.

Evtl. könnte man auch auf die Tabelle der Rasterzellen verzichten, wenn in der Rasterzellen-ID die x- und y-Werte umkehrbar eindeutig kodiert sind.

Anmerkung: Eine endgültige Aussage über die beste Möglichkeit zur Darstellung der Rasterzellen ist zu diesem Zeitpunkt noch nicht möglich und muß noch weitergehend untersucht werden. Auch die Möglichkeiten zur Berechnung der Rasterzellen_ID müssen noch weiter untersucht werden. Hierbei könnten auch sogenannte Z-Values Einsatz finden, um die Zugriffe zu optimieren.

3.2.4 Konsequenz für den eigentliches Raumbezug

Der Raumbezug als solches würde dagegen weiterhin in einer separaten Tabelle existieren, in der sowohl der Gebietsname und die Referenz auf übergeordnete Gebiete, als auch weitere Attribute wie z.B. die Art des Raumbezugs erfaßt werden können. Damit würde innerhalb des Gazetteers eine differenziertere Beschreibung von Raumbezügen möglich, da im Rahmen der Beschreibung des Raumbezuges selbst die semantischen Bestandteile abgebildet und in der Tabelle der Rasterzellen die geometrischen Eigenschaften der Raumbezüge beschrieben werden könnten. Die Verbindung dieser Beschreibungen kann über eine Beziehungstabelle hergestellt werden, die eine komplexe Beziehung (m:n) zwischen Raumbezug und Rasterzelle ausdrückt.

3.2.5 Raumbezugsarten

Für den semantischen Teil der Beschreibung des Raumbezugs ergeben sich nun weitergehende Möglichkeiten hinsichtlich einer logischen Klassifizierung und Hierarchisierung. So können Gebiete z.B. bezüglich ihrer politischen Gebietsgrenzen (Bundesland, Regierungsbezirk, Kreis, Gemeinde), anhand von Kartenblättern oder sonstigen Unterteilungen wie Straßen, Biotope und Flüsse u.a. klassifiziert werden. Für die Realisierung der Klassifizierung sind mehrere Möglichkeiten denkbar:

1. implizit über ein klassifizierendes Attribut (z.B.: Art)
2. explizit über eine Klassenhierarchie, wobei die einzelnen Klassen über „is_a"-Beziehungen an die Klasse „Raumbezug" gebunden werden.

Sowohl die erste, als auch die zweite Möglichkeit bringen Vor- und Nachteile mit sich. So gestaltet sich bei der Lösung 1 vor allem der Zugriff auf die einzelnen Raumbezüge

wesentlich einfacher als mit Möglichkeit 2. Der Nachteil kann jedoch darin bestehen, daß spezielle, für eine bestimmte Art von Raumbezügen notwendigen Attribute, von anderen Arten von Raumbezügen nicht benötigt und deswegen nicht belegt werden. Eine nachträgliche Einbringung weiterer Raumbezugsarten kann bei der Möglichkeit 1 also immer verbunden sein mit Erweiterungen oder Veränderungen der bestehenden Tabellendefinition.

Bei der Möglichkeit 2 dagegen wäre eine nachträgliche Erweiterung um weitere Arten von Raumbezügen einfach herzustellen. In der Tabelle „Raumbezug" würden dabei nur Attribute erscheinen, die allen Raumbezugsarten gemeinsam sind. Die speziellen Attribute hingegen werden erst in der jeweiligen Spezialisierung der Tabelle Raumbezug berücksichtigt. Der Nachteil dieser Lösung steckt jedoch in der wesentlich komplexeren Zugriffslogik, welche mit ziemlicher Sicherheit auch Auswirkungen auf die Performance hätte und mit einem größeren Pflegeaufwand hinsichtlich der Konsistenz der Klassifizierungshierarchie verbunden wäre.

Eine Entscheidung für eine der beiden Möglichkeiten ist also erst zu treffen, nachdem eine Auflistung aller möglichen Raumbezugsarten und deren notwendigen Attribute vorhanden ist. Erst dann kann eine Abwägung der Vor- und Nachteile der beiden Möglichkeiten zu einer Entscheidung über die Repräsentation der Raumbezüge führen.

3.2.6 Beziehungen zwischen Raumbezügen

Des weiteren kann es notwendig werden, die vorhandene „Teil_von"-Beziehung dahingehend zu erweitern, daß ein vorhandener Raumzug nicht mehr nur Teil eines anderen Raumbezugs, sondern mehrerer anderer Raumbezüge sein kann. So kann z.B. eine Gemeinde nicht nur Teil eines Kreises, sondern auch Teil eines Kartenblattes sein. Ob die Einführung solch komplexer Referenzen zwischen Raumbezügen notwendig wird, hängt im wesentlichen von der Entscheidung ab, welche Ansprüche die Benutzerschnittstelle an die Suche nach und die Zuordnung von Raumbezügen zu Informationsobjekten stellt.

Es erscheint auch geboten, andere als die „Teil_von"-Beziehung zwischen Raumbezügen zu unterstützen, beispielsweise „grenzt an" oder „(Straße) geht durch (Gemeinde)". Dies legt nahe, die Beziehungen aus der Raumbezugstabelle auszulagern und in einer eigenen Tabelle zu sammeln. Die möglichen Beziehungstypen sollten wiederum zusammen abgespeichert sein, so daß nur vordefinierte Typen verwendet werden können. Eine vordefinierte Liste der Beziehungstypen erleichtert dem Anwender die Verwendung der Beziehungen.

3.2.7 Raumbezugsarten und -beziehungen

Ob eine bestimmte Beziehung zwischen zwei Raumbezügen überhaupt sinnvoll ist, hängt von der Art der Raumbezüge ab. Deshalb sollte der Gazetteer eine Liste der möglichen Beziehungen zwischen Raumbezugsarten enthalten. Dies kann bei der menügeführten Zusammenstellung einer räumlichen Selektion dienen, wie auch zur Sicherheit bei der Dateneingabe beitragen.

3.2.8 Hierarchie

Beim bisherigen Stand wurde eine hierarchische Suche durch die „Teil_von"-Referenz ermöglicht. Dieser Beziehungstyp wird weiterhin beibehalten und durch weitere Hierarchien wie „geht durch" ergänzt. Beispielsweise kann eine Auswertung der hierarchischen Beziehung „geht durch" des Flusses „Havel" also auch Ressourcen von Gemeinden an der Havel finden. Darüber hinaus existieren jedoch Beziehungsformen, die nicht hierarchiefähig sind, z.B. „grenzt an".

Anmerkung: Eine endgültige Aussage über die beste Möglichkeit zur Darstellung der Raumbezüge ist zu diesem Zeitpunkt noch nicht möglich, da die genauen Anforderungen noch nicht vollständig bekannt sind und noch vertieft untersucht werden müssen.

Architektur des Gazetteers

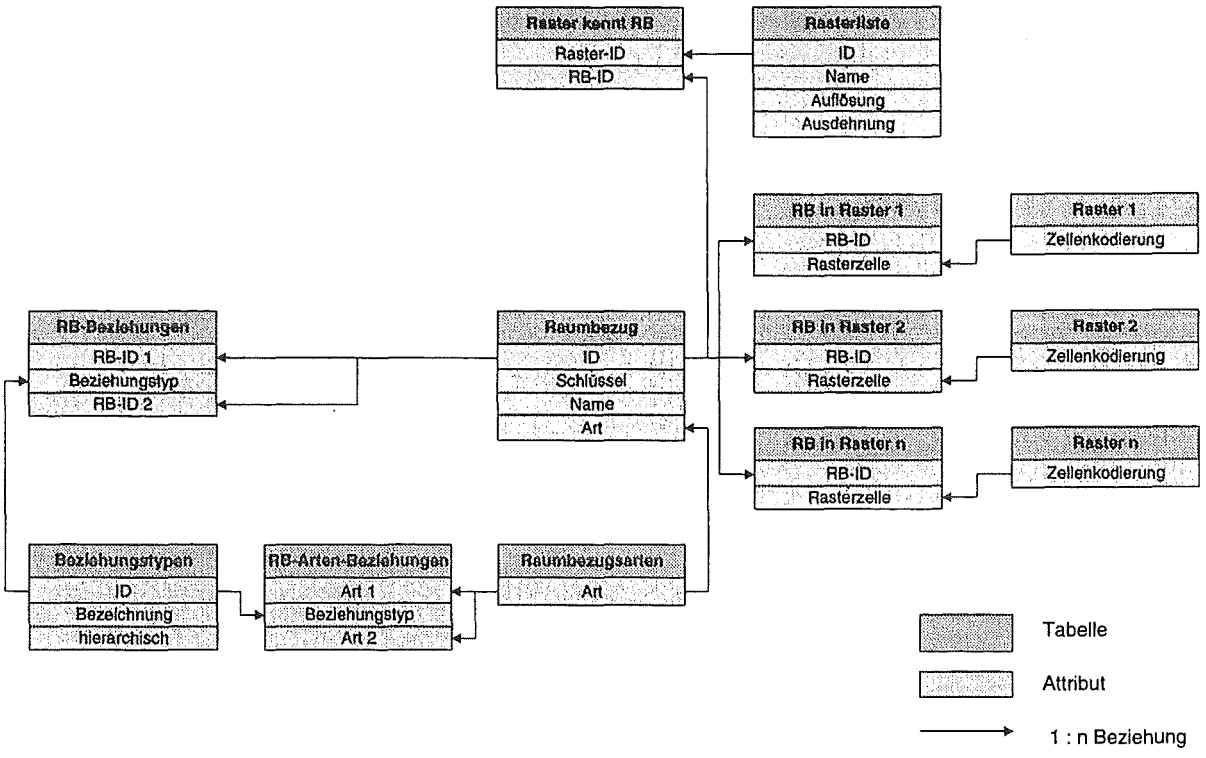


Abbildung 7

3.3 Geplante Aufbereitung von Daten

Die Realisierung der oben beschriebenen Konzeption erfordert zuerst die Definition der benötigten Raumbezüge. Hier wird beschrieben, was wir bereits vorbereiten, ohne die genauen Anforderungen zu kennen.

Mangels realer Daten werden Testdaten aus der GIS-Datenbank *ArcDeutschland'500, Version 1.0.* verwendet. Auf die Mängel dieser Daten wird in den einzelnen Abschnitten eingegangen.

3.3.1 Raumbezüge und deren Beziehungen

3.3.1.1 Verwaltungseinheiten

Davon ausgehend, daß eine Georeferenzierung nach Verwaltungseinheiten, wie Gemeinden oder Bundesländer, unbedingt erforderlich ist, wird geplant, alle Bundesländer, Landkreise und Gemeinden als Raumbezüge bereitzustellen. Eine Verwaltungseinheit wird mit einer internen ID, einem externen Index, dem Namen und einer Klassifikation abgespeichert. Wir wollen nicht den Verwaltungsindex als interne ID verwenden, um flexibler bei späteren Änderungen zu sein.

Die Daten werden zunächst den Karten aus ArcDeutschland´500 entnommen. Diese sind allerdings auf dem Stand von Ende 1992.

Die Verwaltungseinheiten werden mit der Beziehung „ist Teil von" zueinander in Beziehung gesetzt (Gemeinde Sindelfingen ist Teil von Kreis Böblingen, Kreis Böblingen ist Teil des Bundeslandes Baden-Württemberg). Die Raumbezugsart „ist Teil von" gilt aber, wie alle Raumbezugsarten, nur für eine Stufe. Beispielsweise wird die eigentlich richtige Beziehung „Gemeinde Sindelfingen ist Teil des Bundeslandes Baden-Württemberg" nicht gespeichert. Diese Beziehung wird dadurch repräsentiert, daß die Beziehung „ist Teil von" hierarchisch ist.

Eine andere sinnvolle, nicht hierarchische Beziehung zwischen Verwaltungseinheiten kann „grenzt an" sein, um die benachbarten Gemeinden oder Kreise zu finden.

3.3.1.2 Verkehrswege

Umweltdaten könnten auch Straßen, Schienenstrecken, Flughäfen u. ä. zugeordnet werden. Zum Test werden die Autobahnen und Bundesstraßen aus ArcDeutschland´500 eingespielt. Die Straßen werden nicht abschnittsweise, sondern als ganzes gespeichert. Beispielsweise wird man bei der Suche nach der A4 sowohl das Teilstück zwischen Aachen und Olpe wie auch das Teilstück zwischen Bad Hersfeld und Görlitz finden. Auch der Teil bei Köln, der neben der A4 auch zur A3 gehört, wird mit angesprochen.

Als Beziehung zwischen Verkehrswegen ist „mündet/kreuzt" denkbar. Dies gilt für Straße/Straße, Schiene/Schiene wie auch für Straße/Schiene. Aus der Datengrundlage wird allerdings nicht ersichtlich sein, ob man von einer Straße in die andere abbiegen kann oder nicht. Beziehungen von Verkehrswegen sind nicht hierarchisch.

Es läßt sich auch eine Beziehung zwischen Verkehrswegen und den (untersten) Verwaltungseinheiten definieren: „geht durch". So lassen sich leicht alle Gemeinde ansprechen, die an einer Straße liegen, und über deren Hierarchie auch die anderen Verwaltungseinheiten. Ob die Beziehung hierarchiefähig ist, muß noch untersucht werden.

3.3.1.3 Gewässer

Wichtige geographische Objekte in der Umweltverwaltung sind Flüsse und Seen. Hier kann aber nicht das Cover aus ArcDeutschland´500 probehalber herangezogen werden, weil ihm eine wichtige Eigenschaft fehlt: eine eindeutige Identifizierung eines Flusses. Die Flüsse sind

in Teilstücke von einer Flußmündung zur nächsten aufgeteilt. Teilweise liegen zwischen den Teilstücken auch Seen (wie bei der Havel). Der Name eines Flusses ist im Gegensatz zu den Straßen aber nicht eindeutig. Deshalb sind hier noch bessere Daten erforderlich.

Als Beziehung zwischen Gewässer bietet sich „mündet in" an. Dies gilt für Flüsse und Seen gleichermaßen. Die Beziehung ist hierarchiefähig (Spree mündet in Havel, Havel mündet in Elbe).

3.3.2 Georeferenzierung

3.3.2.1 Raster

Es ist geplant, Deutschland in ein 1km-, ein 5km- und ein 10km-Raster einzuteilen. Je kleiner das Raster ist, desto genauer kann ein Raumbezug darauf abgebildet werden. Andererseits nimmt die Datenmenge quadratisch mit der Verkleinerung des Rasters zu und damit die Selektionszeit. Deshalb werden bundesweit drei Raster angeboten. Falls kleinräumige Geo-Objekte verwaltet werden sollen, bei denen eine 1km-Auflösung zu grob ist, müssen weitere Raster definiert werden. Sie sollten aber nur den Ausschnitt enthalten, in dem diese Geo-Objekte liegen. Der Benutzer muß dann vor einer grafischen Selektion das Raster auswählen, auf dem er suchen will.

Bereits bei 1km entsteht eine große Datenmenge. Die Bundesrepublik Deutschland würde dabei in 643 x 868 Zellen eingeteilt. Beim 5km-Raster sind es noch 129 x 174 Zellen und beim 10km-Raster entsprechend 65 x 87 Zellen.

3.3.2.2 Rasterkodierung

Bei den geplanten Rastern soll der Zugriff auf eine Zelle über deren ID erfolgen, in der ihre Lage kodiert ist. Die südwestliche Ecke einer Rasterzelle liegt immer auf einer Kilometermarke des Gauß-Krüger-Koordinatensystems. Die ID besteht dabei aus einer achtstelligen Zahl, deren erste vier Stellen den Rechtswert in Kilometer und deren zweiten vier Stellen den Hochwert in Kilometer angeben. Es wird also die letzte der im Abschnitt 3.2.3 Datenmodell der Rasterung diskutierten Modelle favorisiert.

Diese Vorgehensweise ist allerdings noch nicht endgültig, da sie nicht auf beliebige Rasterungen übertragen werden kann. Erstens reichen bei feinerem Raster 8 Stellen nicht mehr aus, und bei 10 Stellen müßten bereits Gleitkommazahlen verwendet werden. Zweitens kämen bei einer europaweiten Rasterung negative Zahlen vor. Sollte eine Kodierungsart aber bindend vorgegeben werden, sollte besser auf das zweite, oben diskutierte Modell zurückgegriffen werden.

3.3.2.3 Zuordnung von flächigen Raumbezügen

Bei der Einteilung von flächigen Raumbezügen, wie die Verwaltungseinheiten, entsteht das Problem, daß die Rasterzellen den Raumbezug in den Randzonen nicht exakt wiedergeben. Um sicherzustellen, daß zumindest alle gewünschten Objekte zu einem Gebiet gefunden werden, wird ein Raumbezug auch den Rasterzellen zugewiesen, die er teilweise überdeckt.

Dadurch kann es aber auch wie bisher passieren, daß bei einer Auswahl mittels Koordinaten Ressourcen ausgegeben werden, die nur nahe an einem gesuchten Raumbezug liegen. Dies wird aber in einem wesentlich geringeren Umfang vorkommen wie bisher und hängt von der Granularität der Rasterung ab.

Bei der Zuordung der flächigen Raumbezügen zu den Rasterzellen entsteht der größte Speicherbedarf und wohl auch die längsten Zugriffszeiten. Die Zuordnung der Verwaltungseinheiten zum 1km-Raster erfordert ca. 1,35 Millionen Einträge. Beim 5km-Raster sind es ca. 100000 und beim 10km-Raster ca. 50000 Zuordnungen.

3.3.2.4 Zuordnung von linearen Raumbezügen

Auch lineare Raumbezüge, wie Straßen und Flüsse, werden über Rasterzellen georeferenziert. Ein Raumbezug wird dabei durch alle Rasterzellen, durch die er läuft, repräsentiert. Bei einer grafischen Suche nach Ressourcen zu einem Fluß werden somit alle Objekte gefunden, die einen Raumbezug zu einer der Rasterzellen hat, mit denen der Fluß repräsentiert ist.

Bei der Darstellung der linearen Raumbezüge sollte deshalb das Raster nicht zu großmaschig sein. Zur Repräsentation der Autobahnen und Bundesstraßen sind aber bei einer Rasterweite von einem Kilometer auch schon über 66000 Zuordnungen notwendig, bei 5km und 10km sind es ca. 6800.

3.3.2.6 Beispiel

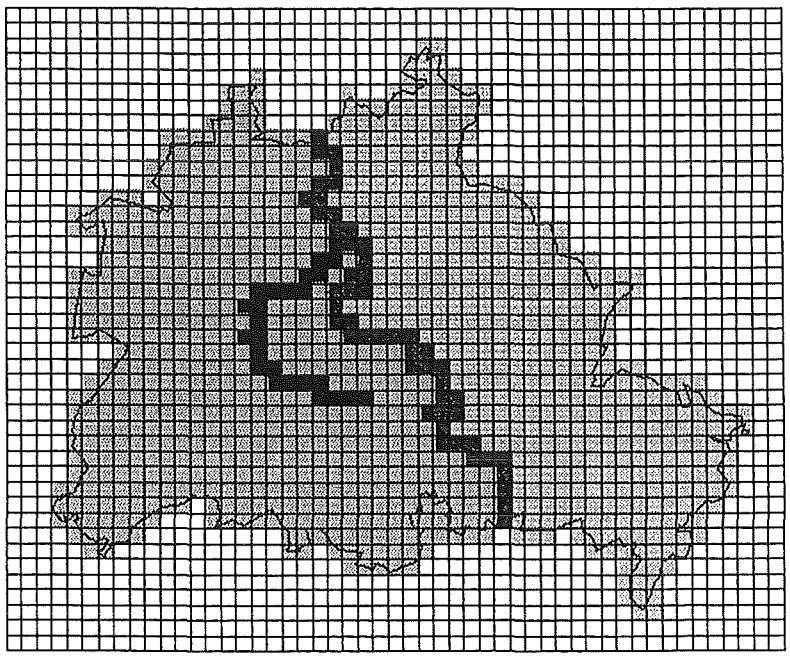


Abbildung 8

Die Abbildung zeigt Berlin im 1km-Raster. Bei einer grafischen Auswahl von Berlin würden die gefärbten Gebiete angesprochen. Bei einer Auswahl von Ostberlin würde das hellgraue Gebiet rechts und der dunkelgraue Streifen entlang der Grenze zwischen Ost- und West-Berlin angesprochen. Im Gebiet von West-Berlin ist noch die A100 eingezeichnet mit den ihr

zugeordneten Rasterzellen.

3.4 Zusammenfassung

Die zweite Version des Gazetteers wird eine wesentlich differenziertere und vielfältigere Georeferenzierung ermöglichen als die alte Version. Die einzelnen räumliche Begriffe werden sowohl durch Sachdaten (Beziehungen) als auch über ihre geographische Lage dargestellt und miteinander verknüpft. Das Konzept sollte dem Anwender eine begreifliche und handhabbare räumliche Zuordnung der Ressourcen erlauben. Der Gazetteer wird aber nicht den Anspruch erheben, mit einem GIS vergleichbar zu sein.

Ob das vorgestellte Konzept so überhaupt realisierbar ist, läßt sich erst beurteilen,

- wenn definiert ist, welche Raumbezüge er enthalten soll, und
- wenn die Datenbank mit realen Daten gefüllt ist.

Erst dann kann festgestellt werden, ob die Architektur eine zufriedenstellende Performance bietet.

Bei der Handhabung ist noch nicht geklärt, ob und wie der Benutzer Einfluß auf die Granularität bei der verknüpften, nicht durch Beziehungen erklärten Suche oder bei der Hierarchiebildung nehmen kann.

4. Literatur

/1/ FAW Ulm / IKE Universität Stuttgart (1996): INTEGRAL III: Integration von heterogenen Komponenten des Umweltinformationssystems (UIS) Baden-Württemberg (Phase III), Abschlußbericht, FAW-Forschungsbericht, Ulm / Stuttgart.

/2/ Mayer-Föll, R. / Jaeschke, A. (Hrsg.) (1995): Projekt GLOBUS – Konzeption und prototypische Realisierung einer aktiven Auskunftskomponente für globale Umwelt-Sachdaten im Umweltinformationssystem Baden-Württemberg – Phase II – 1995, Abschlußbericht, Bericht FZKA 5700. Umweltministerium Baden-Württemberg, Forschungszentrum Karlsruhe.

/3/ Mayer-Föll, R. / Strohm, J. / Schultze A. (1996): Umweltinformationssystem Baden-Württemberg – Überblick Rahmenkonzeption, in: Informatik für dem Umweltschutz, 10. Symposium, Tagungsband, Metropolis-Verlag, Marburg.

/4/ Riekert, W.-F. (1995): Cooperative Management of Data and Services for Environmental Applications, in: Huber-Wäschle, F. / Schauer, H. / Widmayer, P. (Hrsg.): GISI 95 – Herausforderungen eines globalen Informationsverbundes für die Informatik, Informatik aktuell, Springer-Verlag, Berlin – Heidelberg – New York, S. 618–625.

Inhaltlicher Ausbau und Weiterentwicklung des Altlasten-Fachinformationssystems AlfaWeb

R. Weidemann, W. Geiger, M. Reißfelder,
Forschungszentrum Karlsruhe GmbH,
Postfach 3640,
76021 Karlsruhe

E. Schmid, U. Reichert,
Landesanstalt für Umweltschutz Baden-Württemberg,
Griesbachstr. 1,
76185 Karlsruhe

1. Einleitung

Ziel des Vorhabens AlfaWeb (**Al**tlasten-**Fa**chinformationen im World-Wide **Web**) ist es, von der Landesanstalt für Umweltschutz Baden-Württemberg erstellte Arbeitshilfen für eine landeseinheitliche, systematische Altlastenbearbeitung mit den Mitteln moderner Informations- und Kommunikationstechnologien zu erschließen und den Altlasten-Sachbearbeitern über rechnergestützte Navigations- und Zugangshilfen eine effektive Informationsbeschaffung zu ermöglichen. Wesentliche Aufgaben bei der Entwicklung von AlfaWeb sind:

- Erarbeitung von Anleitungen und Hilfsmitteln für die Erstellung von Berichten, die sowohl in Papierversion als auch über das WWW bereitgestellt werden;
- Aufbereitung der vorhandenen Fachberichte und Handbücher für das WWW;
- Entwicklung formular-basierter WWW-Schnittstellen zu ausgewählten PC-Datenbanken und -Fachsystemen;
- Entwicklung eines geeigneten Zugangssystems für eine Internet/Intranet- und eine CD-ROM-basierte Version.

Ein erster Prototyp von AlfaWeb wurde im Rahmen des Projekt GLOBUS II /1/ erstellt und ist im zugehörigen Abschlußbericht ausführlich beschrieben /2/. Die im Rahmen von GLOBUS III durchgeführten Arbeiten lassen sich unter zwei Hauptpunkten zusammenfassen:

- Inhaltlicher Ausbau mit dem Ziel, alle derzeit relevanten Fachberichte und Handbücher in AlfaWeb aufzunehmen und damit eine ausreichende Abdeckung der aktuell in Berichten verfügbaren Fachinformationen zu erhalten und
- Weiterentwicklung der Systemkomponenten mit dem Ziel, für die Altlasten-Sachbearbeiter durch die Nutzung von AlfaWeb eine echte Arbeitserleichterung zu erreichen.

Dieser Bericht beschreibt den aktuellen Stand der Entwicklung. Kapitel 2 enthält einen Überblick der AlfaWeb-Struktur und -Funktionalität. Der inhaltliche Ausbau im Rahmen von GLOBUS III ist Thema des Kapitels 3. Die durchgeführten Arbeiten zur Weiter- bzw. Neuentwicklung einzelner AlfaWeb-Komponenten werden in Kapitel 4 beschrieben. Kapitel 5 enthält Angaben darüber, auf welche Weise AlfaWeb bzw. das im Rahmen des Vorhabens entwickelte Konvertierungswerkzeug genutzt werden können.

2. Überblick AlfaWeb

Informationsbasis von AlfaWeb sind i.w. die von der Landesanstalt für Umweltschutz Baden-Württemberg, Abt. 5, in den Reihen „Materialien zur Altlastenbearbeitung“ und „Texte und Berichte zur Altlastenbearbeitung“ herausgegebenen Fachberichte und Handbücher. Diese sind in AlfaWeb in eine Hypertextstruktur eingebettet, d.h. in handhabbare Einheiten (Unterkapitel) zerlegt und nach strukturellen und inhaltlichen Gesichtspunkten untereinander verzeigert. Ein zielgerichteter Zugang zu den Informationen ist über die AlfaWeb-Startseite (s. Abb.1) möglich.

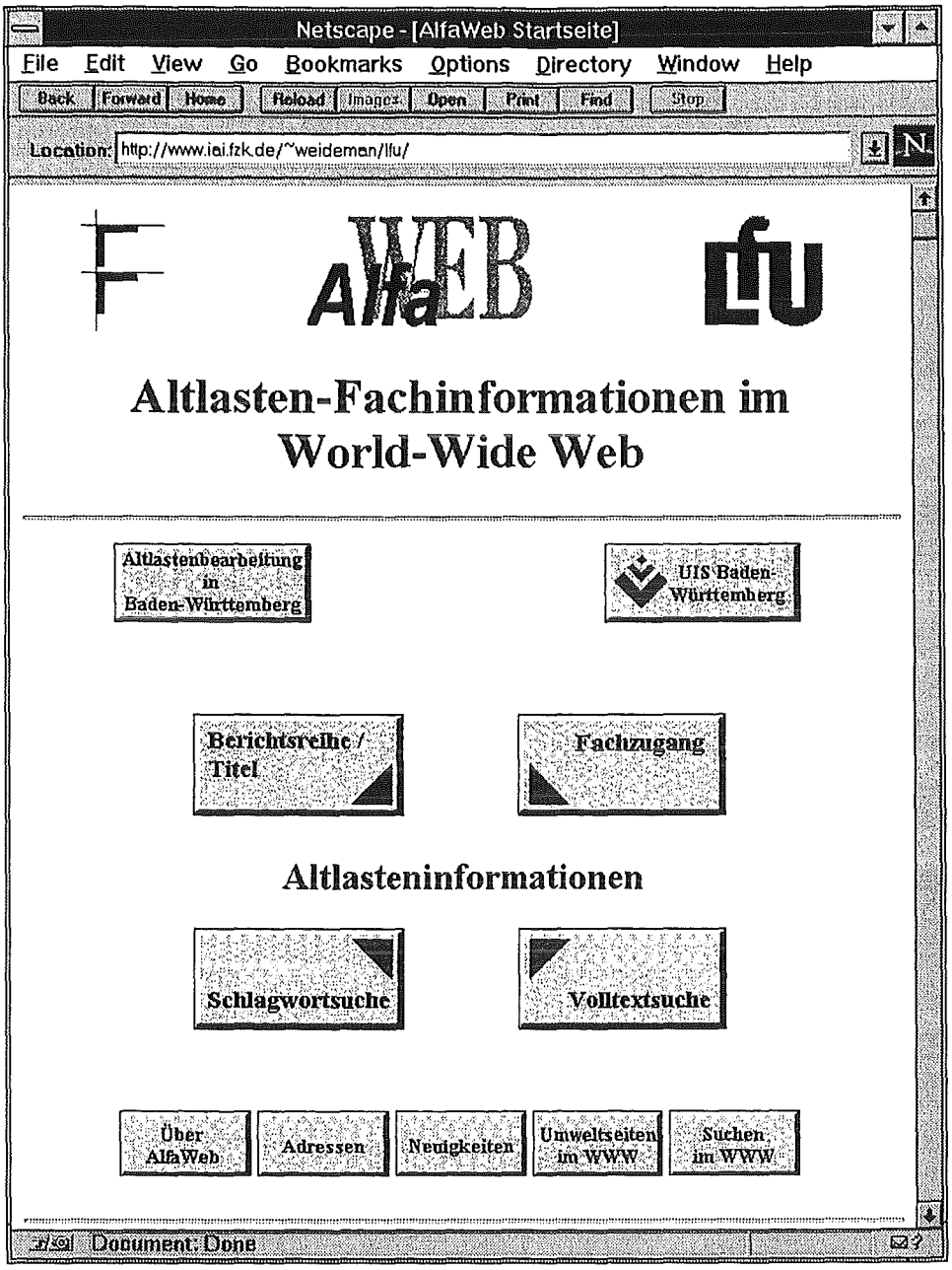


Abbildung 1: AlfaWeb-Startseite

Vier Buttons (Schaltflächen) auf der AlfaWeb-Startseite realisieren unterschiedliche Zugangswege zu den Altlasteninformationen:

Berichtsreihe/Titel:	Hier ist die nach Berichtsreihe und -nummer geordnete aktuelle Literaturliste der LfU/Abt.5 hinterlegt. Zu einem bestimmten Bericht kann eine Kurzbeschreibung, das Inhaltsverzeichnis und/oder der vollständige Bericht verfügbar sein. Kapitel 3 enthält nähere Angaben zum aktuellen Stand.
Fachzugang:	Um einzelne Arbeitsschritte bei der Altlastenbearbeitung direkt unterstützen zu können, werden hier von Praktikern für die einzelnen Phasen der Bearbeitung Verweise auf relevante Dokumente nach fachlichen Kriterien zusammengestellt (s. Kap. 4.2.1).

Schlagwortsuche: Derzeit verwendet AlfaWeb den Thesaurus umweltrelevanter Begriffe des Umweltbundesamts. Berichtsseiten werden automatisch verschlagwortet und sind dann in AlfaWeb unter dem jeweiligen Schlagwort eingetragen (vgl. /2/ und Kap. 4.2.3).

Volltextsuche: Die Volltextsuche erlaubt dem Benutzer die Suche nach beliebigen Begriffen oder Begriffskombinationen in den AlfaWeb-Dokumenten (siehe Kap. 4.2.2).

Weitere Informationsangebote sind über 7 zusätzliche Buttons abrufbar:

Altlastenbearbeitung in Baden-Württemberg: Hier werden einführende Informationen zur Systematik, zum Stand und zu den Rechtsgrundlagen der Altlastenbearbeitung in Baden-Württemberg zusammengestellt.

UIS Baden-Württemberg: AlfaWeb ist Teil des Umweltinformationssystems (UIS) Baden-Württemberg.

Über AlfaWeb: Unter diesem Stichpunkt wird das Vorhaben AlfaWeb und dessen Einbindung in das Projekt GLOBUS vorgestellt.

Adressen: Die Ansprechpartner in der LfU und im Forschungszentrum Karlsruhe (FZK) sind mit Postanschrift, Telefon, FAX und EMail-Adresse aufgeführt.

Neuigkeiten: Zur Information besonders der „Stammkunden" werden hier die letzten Änderungen und Erweiterungen, z.B. neue Berichte angekündigt. Außerdem sind gewisse statistische Informationen einsehbar (Anzahl Seiten, Abbildungen, Zugriffe; s. Kap. 5).

Umweltseiten im WWW: Einige für die Altlastenbearbeitung relevante Informationsquellen im WWW sind auf dieser Seite zusammengestellt.

Suchen im WWW: Um nach beliebigen Informationen im WWW suchen zu können, ist hier aus der schier unüberschaubaren Menge von Verzeichnissen und Suchmaschinen für den gelegentlichen Benutzer eine kleine Auswahl zusammengestellt.

3. Inhaltlicher Ausbau

Der Ende 1995 im Rahmen von GLOBUS II fertiggestellte Prototyp von AlfaWeb enthielt insgesamt 10 Fachberichte und Handbücher der LfU. Zielrichtung in jener Projektphase war die Entwicklung erster Versionen der Hilfsmittel zur Aufbereitung der Berichte und die Erstellung eines Prototys zur Beurteilung der Praxisrelevanz der Lösung. Die ursprüngliche Planung für GLOBUS III sah vor, einige wenige weitere Berichte aufzunehmen. Die positive Resonanz der Anwender führte aber dazu, daß im Laufe des Jahres 1996 die Planung revidiert und als neue Zielvorgabe die Aufnahme eines Großteils der vorhandenen Altlastenberichte der LfU bis zum Jahresende 1996 vereinbart wurde. Damit sollte sichergestellt werden, daß AlfaWeb zu diesem Zeitpunkt aus inhaltlicher Sicht sinnvoll in der Praxis verwendbar wird.

Um Berichte in AlfaWeb einstellen zu können, ist eine Aufbereitung nach bestimmten Vorgaben notwendig (siehe Kap. 4.1 und /5/). Die vorhandenen Berichte lagen teils nur in Papierform oder in Formaten verschiedener Textverarbeitungssysteme vor. Die prinzipielle Vorgehensweise zur Aufbereitung der Berichte wurde bereits in /2/ beschrieben. Die konkrete Durchführung der Arbeiten erfolgte mit Unterstützung des Zentralen Fachdienstes (ZFD) der LfU und von studentischen Hilfskräften. Der ZFD übernahm das Einscannen der nur in Papierform vorliegenden Berichten, die Studenten die Aufbereitung mittels des Textverarbeitungssystems Microsoft Word für Windows™. Tabelle 1 zeigt den inzwischen erreichten Stand.

Berichtsreihen	vorhandene Berichte	Soll AlfaWeb 30.11.96	Ist-Stand AlfaWeb 23.11.95	Ist-Stand AlfaWeb 12.06.96	Ist-Stand AlfaWeb 07.11.96
Materialien zur Altlastenbearbeitung	23	22	4	9	21
Texte und Berichte zur Altlastenbearbeitung	28	19	4	4	18
Sonstige Berichte	3	2	2	2	3
Summe:	54	43	10	15	42

Tabelle 1: Altlastenberichte der LfU in AlfaWeb

Die Differenz zwischen der Anzahl vorhandener Berichte und den Sollvorgaben für AlfaWeb ergibt sich daraus, daß einige Berichte veraltet bzw. aus technischen Gründen noch nicht zur Aufbereitung verfügbar sind.

Abbildung 2 zeigt, wie sich in Folge des inhaltlichen Ausbaus das Netz der WWW-Seiten, d.h. die Informationsbasis von AlfaWeb entwickelt hat. Derzeit (Stand: 07.11.96) enthält das Netz über 4200 HTML-Files (= WWW-Seiten) und ca. 1850 GIF- bzw. JPG-Files (separate Abbildungen oder Tabellen).

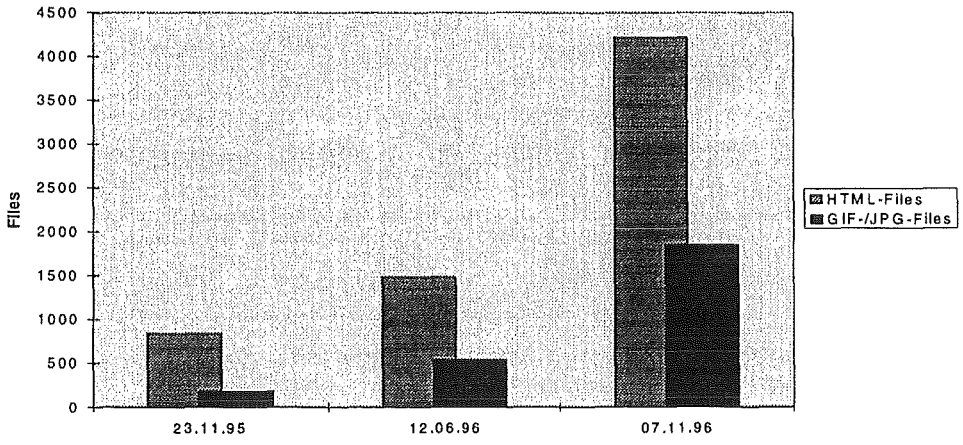


Abbildung 2: Inhaltlicher Ausbau von AlfaWeb

4. Weiterentwicklungen

4.1 Erstellung von Berichten

Im Rahmen von GLOBUS II wurde ein Vorgehensmodell entworfen, das es erlaubt, neue Handbücher und Fachberichte so zu erstellen, daß diese sowohl als Papierdokument ausgedruckt, als auch mit einem weitgehend automatisch ablaufenden Konvertierungswerkzeug in AlfaWeb eingestellt werden können (siehe /2/). Die wichtigsten Eckpunkte sind:

- Berichte werden mit Word für Windows in einer definierten Form erstellt. Unterstützt wird dies durch:
  - eine Dokumentenvorlage, welche die verwendbaren Formatvorlagen enthält und
  - einer Richtlinie, welche die Anwendung der Dokumentenvorlage beschreibt /5/.
- Das Konvertierungsprogramm zerlegt und verzeigert den Word-Bericht[1] gemäß der Kapitelstruktur und konvertiert die Seiten in das HTML-Format.

Das Konvertierungsprogramm, das in der Programmiersprache Perl entwickelt wurde, setzt auf Free-/Shareware-Produkten (rtftohtml und rtftoweb) auf, die auf die Bedürfnisse von AlfaWeb zugeschnitten und erweitert wurden. Weiterentwicklungen in GLOBUS III am Konvertierungsprogramm erfolgten zum einen, indem neue Versionen der Basissoftware rtftohtml / rtftoweb in Betrieb genommen wurden und damit deren neue Funktionalität nutzbar wurde und zum anderen durch Weiterentwicklung der AlfaWeb-eigenen Komponenten. Erweiterungen wurden dabei in folgenden Punkten vorgenommen:

- Hoch- und Tiefstellung von Zeichen kann verwendet werden (z.B. Cr^{3+}, H_2O)
- Tabellen können in echte HTML-Tabellen umgesetzt werden, die auch komplexere Gestaltungsmöglichkeiten in den Zellen erlaubt.
- Auf jeder Berichtsseite sind auswählbare Verweise auf alle übergeordneten Kapitel enthalten.
- Verweise auf den Berichtsseiten entsprechend der Berichtsstruktur werden als Buttons (Schaltflächen) eingefügt.
- Das Literaturverzeichnis eines Berichts kann von jeder Berichtsseite aus über einen speziellen Button erreicht werden. Ebenso kann von jedem Literaturhinweis der entsprechende Eintrag im Literaturverzeichnis direkt angesprungen werden.
- Abbildungen können wahlweise indirekt über einen Link (Button) mit der Berichtsseite verknüpft oder direkt eingefügt werden.
- Verweise im Indexverzeichnis enthalten zur Unterscheidung von Mehrfachreferenzen jeweils die Kapitelüberschrift.

Während das Konvertierungsprogramm in der im Rahmen GLOBUS II entwickelten Version auf die Verwendung in AlfaWeb beschränkt war, wurden jetzt alle AlfaWeb-spezifischen Angaben aus dem Programm entfernt und in einem speziellen Parameterfile zusammengefaßt. Damit ist eine Verwendung des Programms auch für andere Anwendungen möglich (s.Kap. 5).

---

[1] Genauer: .. den in Word geschriebenen und im RTF-Format (**R**ich **T**ext **F**ormat) abgespeicherten Bericht ..

Entprechend der Weiterentwicklung des Konvertierungsprogramms wurden auch die Dokumentenvorlage und die Richtlinie zur Berichtserstellung fortgeschrieben. Dabei wurden neue, möglichst selbsterklärende, deutsche Bezeichnungen für die Formatvorlagen eingeführt. Die Richtlinie wurde um eine Kurzanleitung ergänzt, welche rezeptartig die Erstellung neuer Berichte unterstützt.

4.2 Zugangssystem

4.2.1 Fachzugang

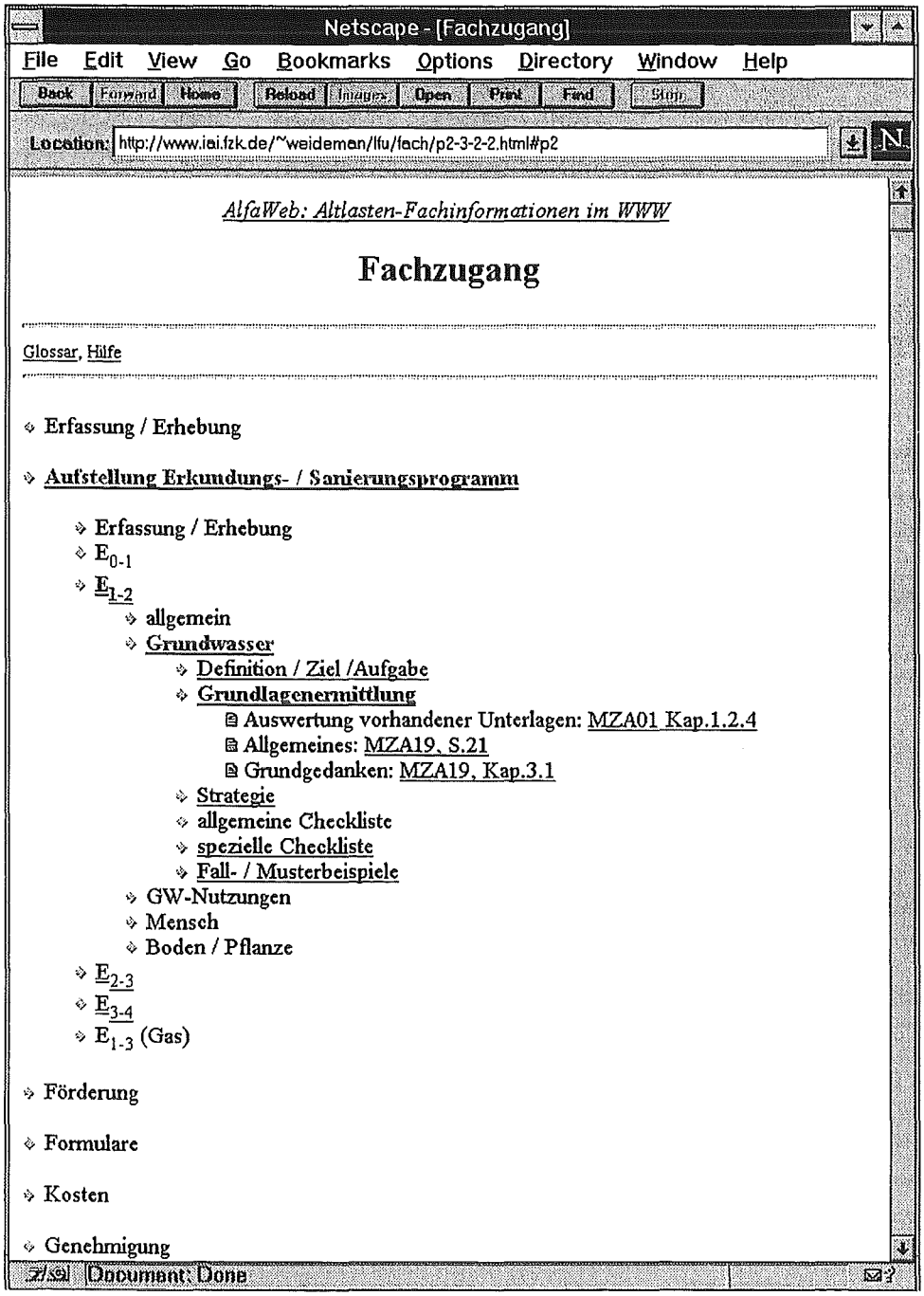


Abbildung 3: Fachzugang

Der Fachzugang zu AlfaWeb wurde in GLOBUS III neu implementiert. Im Gegensatz zu den anderen Zugängen geht man hier von den Vorgängen bei der Altlastenbearbeitung aus. Von Fachexperten werden für die einzelnen Tätigkeiten die wichtigsten Informationen und Hilfsmittel zusammengestellt. Dabei können nicht nur Altlasteninformationen aus AlfaWeb, sondern auch Dokumente anderer WWW-Servern berücksichtigt werden.

Derzeit enthält AlfaWeb eine Aufgliederung der typischen Tätigkeiten bei der Altlastenbearbeitung (s. Abb.3) und in einigen Punkten beispielhaft eine tiefere Gliederung bis zu konkreten Verweisen auf AlfaWeb-Berichte. Die Zusammenstellung weiterer Verweise durch die Fachexperten der LfU ist in Arbeit.

4.2.2 Volltextsuche

Die Volltextsuche erlaubt die Eingabe von einem oder mehreren durch „und" bzw. „oder" verknüpfter Suchbegriffe, wobei auch Teilbegriffe (mit Platzhaltern für beliebige Zeichenfolgen) verwendet werden können. Als Ergebnis werden Verweise auf AlfaWeb-Seiten geliefert, in denen diese Suchbegriffe vorkommen. Dieser bereits in GLOBUS II implementierte Zugangsweg wurde in GLOBUS III weiterentwickelt.

Es stehen jetzt zwei Suchformulare zur Verfügung:

Standardformular	Das einfachere Standardformular erlaubt die Eingabe zweier Suchbegriffe in getrennten Eingabefeldern und die Auswahl der Verknüpfung durch Anklicken mit der Maus.
Expertenformular	Hier können in einem einzigen Eingabefeld komplexere Suchanfragen definiert werden. Dieses Formular ist für den geübten Benutzer gedacht.

In beiden Fällen wird eine Ergebnisseite produziert, wie sie beispielhaft in Abbildung 4 wiedergegeben ist.

Im oberen Teil wird die Suchanfrage wiedergegeben, im Beispiel war dies „Deponie und Oberflächenabdichtung". Verweise auf die gefundenen Dokumente sind im unteren Teil der Seite angegeben. Wird eine größere Anzahl Verweise gefunden, werden nur die ersten 50 mit der höchsten Bewertungsziffer (siehe /2/) und ein Hinweis auf die Gesamtzahl ausgegeben. In diesem Fall, aber auch wenn zu wenige oder keine Verweise gefunden wurden, kann direkt über das Ergebnisformular die Suchanfrage modfiziert und neu abgeschickt werden. Dazu wird ein weiterer Suchbegriff eingegeben und die Anfrage entweder verallgemeinert („oder"-Verknüpfung) oder eingeschränkt („und"-Verknüpfung).

Neben der Überarbeitung der Benutzeroberfläche wurde eine Optimierung der Parametereinstellungen des Suchprogramms durchgeführt, um das aus den USA stammende Tool (swish) besser an die Eigenschaften der deutschen Sprache und der technischen / chemischen Fachbegriffe in AlfaWeb anzupassen. Weitere Anpassungen wurden durch die anwachsende Datenmenge in AlfaWeb notwendig.

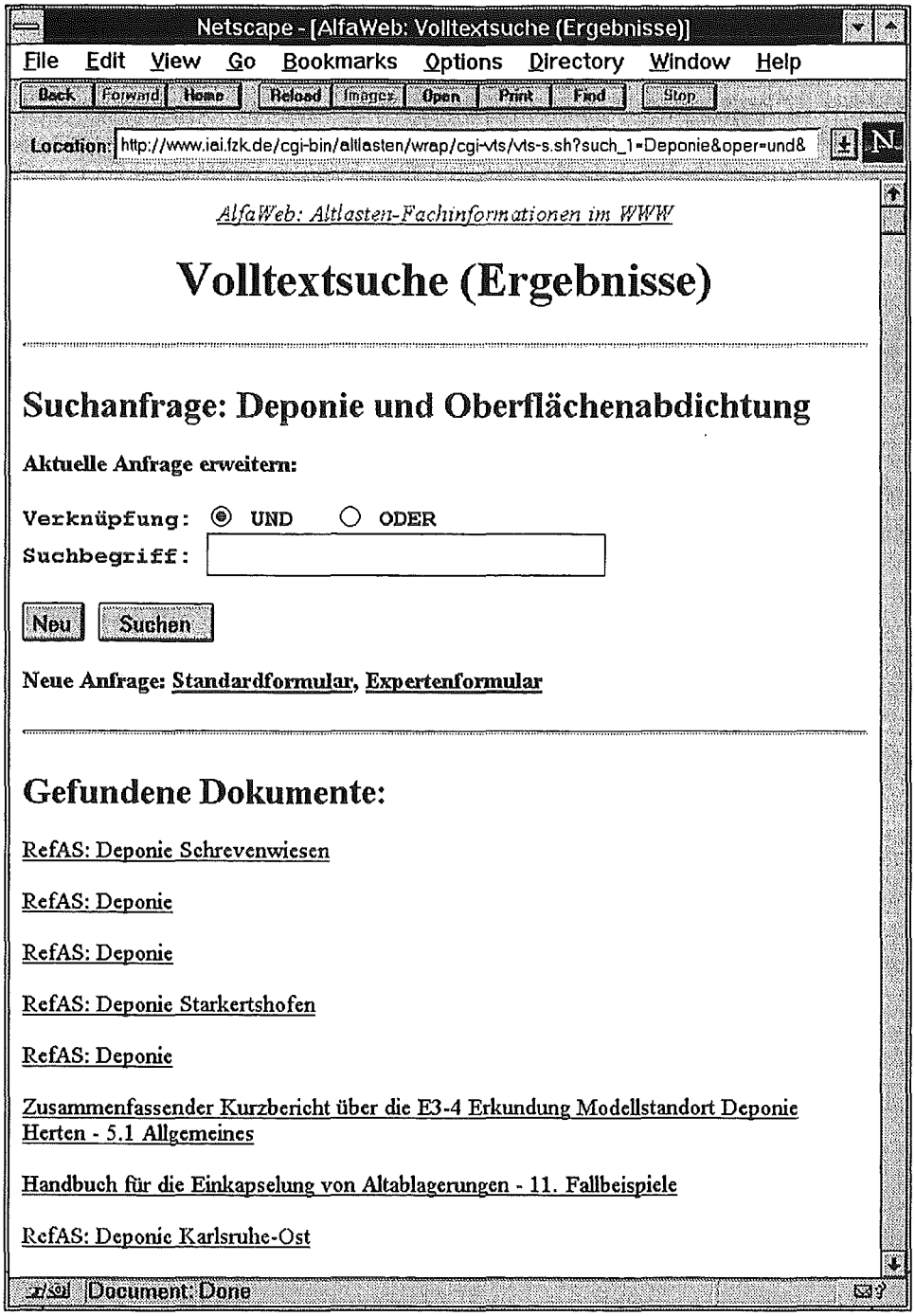


Abbildung 4: Beispiel für ein Suchergebnis der Volltextsuche

4.2.3 Schlagwortsuche

Die Werkzeuge zur Verschlagwortung wurden für die neu aufgenommenen Berichte benutzt, um diese über die Schlagwortsuche zugänglich zu machen. Die Schlagwortsuche selbst (siehe /2/) wurde in GLOBUS III nicht weiterentwickelt. Im Anwendungstest hat es sich erwiesen, daß der verwendete Thesaurus umweltrelevanter Begriffe des Umweltbundesamts für den Altlastenbereich zum einen zu breit angelegt, zum anderen im engeren Fachbereich nicht tief genug ist. Außerdem wird von Seiten der Anwender eine automatische Verschlagwortung als nicht zweckmäßig angesehen.

Gemeinsam mit dem Zentralen Fachdienst (ZFD) und der Fachabteilung der LfU wurde eine neue Lösung für die Schlagwortsuche konzipiert. Der ZFD baut derzeit für verwaltungsinterne Zwecke eine Volltextdatenbank auf, die ebenfalls Altlastenberichte umfaßt und u.a. über eine Schlagwortsuche erschlossen werden soll. Eine enge Abstimmung zwischen den AlfaWeb- und den ZFD-Aktivitäten ist daher unabdingbar. Bezüglich der Verschlagwortung wird folgender Weg beschritten:

- Der ZFD prüft verfügbare Thesauri auf ihre Verwendbarkeit im Altlastenbereich bzw. läßt einen geeigneten Thesaurus an die speziellen Bedürfnisse des Altlastenbereichs anpassen.
- Der ZFD beschafft und installiert ein Softwarepaket zur Pflege des Thesaurus, die zentral vom ZFD vorgenommen werden wird. Ein ensprechendes Produkt ist bereits im Test.
- Die jeweils aktuelle Version dieses Thesaurus wird sowohl für die Volltextdatenbank des ZFD, als auch für AlfaWeb verwendet.

Bis zur endgültigen Klärung von Thesaurusstruktur und Art des Pflegeprogramms (z.B. Export-Schnittstellen) wurde die Überarbeitung der Schlagwortsuche in AlfaWeb zurückgestellt. Die Verschlagwortung eines Berichts ist dann wie folgt geplant:

- Die Berichtsersteller liefern zusätzlich Schlagworte (Vorschläge) zu jedem Bericht entsprechend dem Thesaurus.
- Beim Einstellen eines Berichts in AlfaWeb werden durch den Bearbeiter der LfU Schlagworte endgültig zugewiesen. Der Bearbeiter bestimmt Schlagworte explizit per Hand und zwar aus:
  - den Vorschlägen des Berichtserstellers,
  - einer automatisch generierten Liste von Thesaurusbegriffen, die mittels Volltextsuche im Bericht gefunden wurden,
  - dem Thesaurus.

4.3 Integration von Anwendungsprogrammen

Altlasten-Fachinformationen der Landesanstalt für Umweltschutz wurden bis vor Kurzem ausschließlich als Fachberichte und Handbücher veröffentlicht. Zunehmend gewinnen jedoch DV-unterstützte Lösungen (Datenbanken, Anwendungsprogramme) an Bedeutung. Beispiele dafür sind:

- der Branchenkatalog zur historischen Erhebung von Altstandorten,
- die XUMA-PC-Programme zur Analysenplan-Erstellung bzw. Bewertung /6/,
- der Referenzkatalog Altlasten / Schadensfallsanierung.

Eine der Aufgaben im Rahmen von GLOBUS III war die beispielhafte Integration eines der Anwendungsprogramme in AlfaWeb. In Absprache mit der LfU wurde dazu der Referenzkatalog Altlasten / Schadensfallsanierung - kurz RefAS - ausgewählt. RefAS wird im Auftrag der LfU vom Ingenieurbüro Trischler und Partner realisiert. Das System enthält Informationen - speziell Literaturverweise - zu bereits erfolgten Sanierungen. Es wurde als PC-Anwendung auf der Basis von Microsoft Access™ implementiert. Derzeit wird das PC-Programm in der LfU getestet (Abnahmetest).

Wegen der relativ kleinen Datenbasis (ca. 380 Fälle mit meist einem, z.T. auch mehreren Literaturverweisen) und der zu erwartenden geringen Änderungsrate wurde für AlfaWeb eine relativ direkte Einbindung[2] in die Informationsbasis gewählt. Über ein Generierungsprogramm, das in Perl implementiert wurde und auf Export-Files der Access-Datenbank aufsetzt, wird für jeden Fall eine HTML-Seite erzeugt, welche alle fallbeschreibenden Daten enthält. Ausserdem erzeugt das Generierungsprogramm ein HTML-Formular, über das gezielt nach Fällen gesucht werden kann (s.u.).

Die HTML-Seiten für die Fälle werden wie Berichtsseiten in die Informationsbasis von AlfaWeb eingestellt und sind dann über die normalen Suchmechanismen (Volltextsuche, Schlagwortsuche) ohne weiteres auffindbar. In Abbildung 4 (Kapitel 4.2.2), welche beispielhaft ein Suchergebnis einer Volltextsuche enthält, sind neben Verweisen auf normale Berichtsseiten auch Verweise auf RefAS-Fälle enthalten.

RefAS wurde auch dazu benutzt, dem Anwender sinnvolle Einsatzmöglichkeiten für Frames[3] im WWW zu demonstrieren. Abbildung 5 zeigt die Benutzerschnittstelle von RefAS, wie sie sich mit dem Netscape-Browser darstellt. Das Fenster ist vertikal in zwei Teilfenster aufgespalten:

- Auf der linken Seite kann eine Hilfeseite bzw. ein neues Suchformular angefordert werden. Nach einer Suche werden hier auch die gefundenen Fälle aufgelistet. Im Beispiel wurde bereits eine Suche durchgeführt, die 4 Fälle gefunden hat.
- Die rechte Seite enthält nach dem Start und auch nach einer Anforderung aus dem linken Teilfenster ein leeres Suchformular. Wird nach einer Suche ein im linken Teilfenster aufgeführter Fall mit der Maus angeklickt, erscheinen im rechten Teilfenster die zugehörigen Daten.

Die Zweiteilung des Fensters wird beim Verlassen von RefAS wieder aufgehoben.

[2] Im Gegensatz zu einer indirekten Datenbank-Ankopplung.
[3] Unterteilung des WWW-Browser-Fensters in separate Teilfenster.

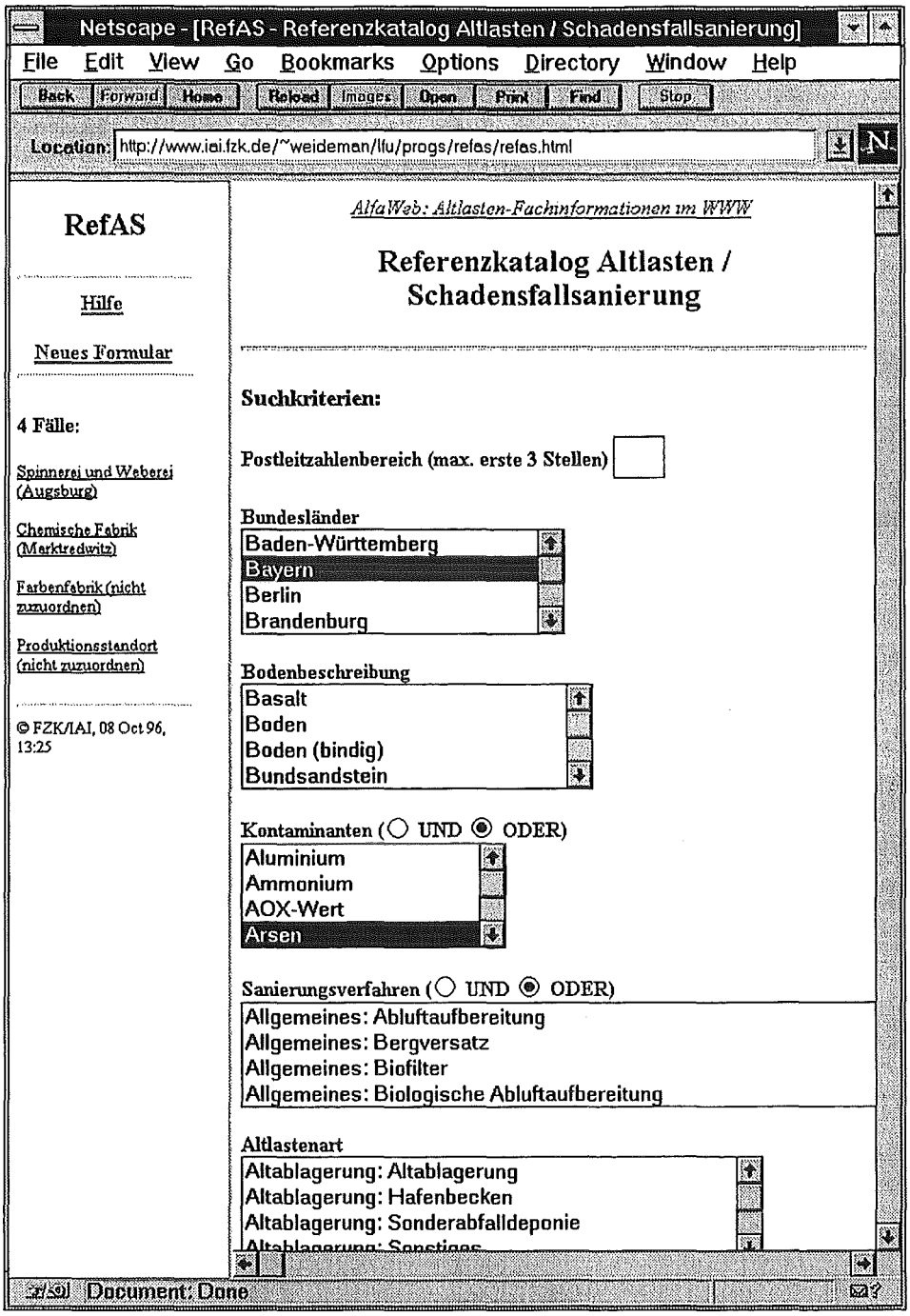


Abbildung 5: RefAS-Formular

Das Suchformular erlaubt Angaben zu folgenden Punkten:

Postleitzahl	Die Suche kann auf alle Orte eingeschränkt werden, deren Postleitzahl mit den angegebenen Ziffern (maximal 3) beginnt.
Bundesländer	In der Liste kann eine beliebige Anzahl von Bundesländern ausgewählt werden. Gesucht wird nach Fällen, die in einem der angegebenen Bundesländer liegen (ODER-Verknüpfung). Alle nachfolgenden Listen sind ebenfalls Mehrfachauswahl-Listen.
Bodenbeschreibung	entsprechend Bundesländer

Kontaminanten	Hier kann zusätzlich angegeben werden, ob bei den gesuchten Fällen einer der angegebenen Kontaminanten (ODER) bzw. alle angegebenen Kontaminanten (UND) erwartet werden.
Sanierungsverfahren	entsprechend Kontaminanten
Altlastenart	entsprechend Bundesländer

Wenn zu mehreren der genannten Punkte Angaben gemacht werden, werden die Fälle gesucht, die alle diese Anforderungen gleichzeitig erfüllen (UND-Verknüpfung).

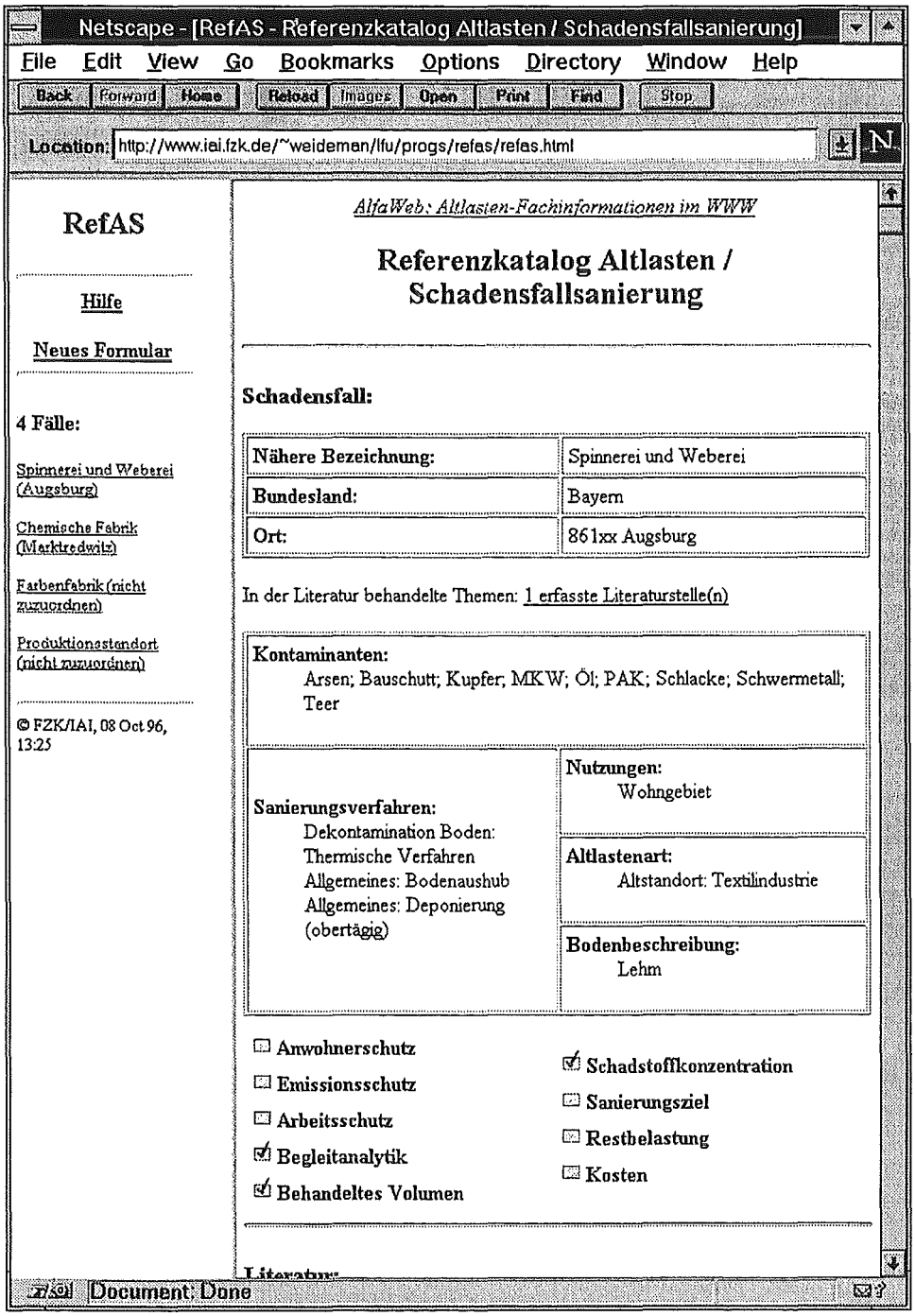


Abbildung 6: Fallbeispiel aus RefAS

Bei Auswahl eines gefundenen Falls im linken Frame, erhält man die zugehörige

Fallbeschreibung, wie sie ausschnittsweise und beispielhaft in Abbildung 6 wiedergegeben ist. Dieselbe Darstellung (rechter Frame) würde man auch erhalten, wenn der Fall über die Volltextsuche gefunden und angewählt worden wäre.

4.4 CD-ROM

Ein Problem bei der Verbreitung der Altlasteninformationen über das WWW ist, daß bisher längst nicht alle Verwaltungsstellen und Ingenieurbüros einen Zugang zum Intranet (z.B. LVN) bzw. Internet haben. Außerdem sind derzeit die Anwortzeiten im Internet sehr unterschiedlich und oft inakzeptabel lang. Aus diesen Gründen ist geplant, zusätzlich eine AlfaWeb-Version auf CD-ROM herauszugeben.

Die CD-ROM ist eine relativ preiswerte Möglichkeit, größere Datenmengen zu verbreiten. Die derzeitige Kapazität einer Scheibe beträgt ca. 600 MB, sodaß sich AlfaWeb (z.Zt. ca. 100 MB) ohne Probleme darauf unterbringen läßt. CD-ROMs können über die an jeden PC anschließbaren CD-ROM-Brenner (Kosten ca. 1500,- DM + ca. 15,- DM pro CD-ROM) erstellt werden. Wegen des Zeitbedarfs (15 - 30 min pro CD-ROM) und der größeren Empfindlichkeit ist dies jedoch nur für Einzelexemplare empfehlenswert. Für größere Auflagen, wie bei den LfU-Berichten üblich, werden die CD-ROMs mit einer speziellen Hardware gepreßt, sodaß die Produktion im Fremdauftrag erfolgen muß (Kosten: ca. 500 - 1000 DM Grundpreis + 2-4 DM pro CD-ROM).

Eine direkte Übertragung eines WWW-basierten Informationssystems wie AlfaWeb auf eine CD-ROM ist allerdings nicht möglich. Um dies verständlich zu machen, ist einerseits zwischen der Client-Seite (Nutzer) und der Server-Seite (Anbieter) und andererseits zwischen statischen[4] und dynamischen WWW-Seiten zu unterscheiden. Aufgabe der client-seitig verwendeten WWW-Browser (Netscape Navigator, Microsoft Explorer) ist es i.w. über eine URL[5] (WWW-Adresse) spezifizierte Dateien anzufordern, falls möglich direkt anzuzeigen oder in vom Benutzer definierter Weise zu behandeln (z.B. Übergabe an einen externen Viewer, sichern auf Festplatte).

Auf Informationsanbieterseite wird in einem WWW-basierten Informationssystem ein WWW-Server betrieben. Aufgabe des WWW-Servers ist es, von den Browsern angeforderte URLs zu liefern. Dies erfolgt für statische Seiten einfach dadurch, daß die URL auf einen Pfadnamen abgebildet, die entsprechende Datei geholt und an den Browser übertragen wird. Zur flexiblen und jeweils aktuellen Beantwortung von Benutzeranfragen können prinzipiell beliebige Programme verwendet werden[6]. Aufgabe des WWW-Servers ist es in diesem Fall, eine Anfrage an das zuständige Programm weiterzugeben und dessen Ergebnis, das die Form einer HTML-Seitenbeschreibung besitzen muß, an den Browser zurückzuliefern.

Beim Ersatz einer WWW-basierten Lösung durch eine CD-ROM-Lösung sind statische Seiten

---

[4] Für **statische** Seiten ist die vollständige Beschreibung in einer Datei abgelegt. Im Gegensatz dazu wird die Beschreibung einer **dynamischen** Seite erst bei Bedarf, d.h. wenn diese von einem Browser angefordert wird, erzeugt.

[5] **U**niform **R**esource **L**ocator

[6] An dieser Stelle wird nur auf die dynamische Erzeugung von Seiten über CGI-Skripts eingegangen. Alternativen wie Java werden nicht betrachtet, da die Problematik grundsätzlich ähnlich ist und damit die Ausführungen hier unnötig weitschweifig würden.

weitgehend (s.u.) problemlos. Da eine URL auch eine lokale Datei z.B. einer CD-ROM bezeichnen kann, sind die Browser in der Lage, für statische Seiten die Aufgaben des WWW-Servers mit zu übernehmen. Dies ist für dynamische Seiten nicht möglich. Daraus ergeben sich einige Probleme:

- Die Plattform-Unabhängigkeit der WWW-Lösung geht verloren. Bei der WWW-Lösung muß die anbieterseitige Funktionalität nur auf dem Rechner, auf dem der WWW-Server läuft, erbracht werden, kann aber von beliebigen Rechnern aus genutzt werden. Im Gegensatz dazu muß die Funktionalität bei der CD-ROM-Lösung auf dem Rechner des Nutzers erbracht werden, d.h. die verwendeten Programme müssen auf dessen spezieller Maschine auch lauffähig sein.
- Die Anbindung der Programme, welche die dynamischen Seiten produzieren, erfolgt normalerweise über den WWW-Server. Im Gegensatz zu den Browsern sind dies aber i.d.R. keine Programme für den Endanwender, d.h. deren Installation und Pflege erfordert schon eingehendere Systemkenntnisse.

Will man die Funktionen von AlfaWeb, welche dynamische Seiten verwenden, mit in die CD-ROM-Lösung übernehmen, ergeben sich daraus die folgenden Konsequenzen:

- Man beschränkt sich auf **eine** Zielplattform, denn eine Unterstützung verschiedener Plattformen wäre zu aufwendig. Die Abt.5 der LfU hat sich auf Windows NT (ab 4.0) und Windows 95 festgelegt. Derzeit laufen AlfaWeb-Server und die eingebundenen Programme unter Unix (SunOS, Sun Solaris, DEC Unix), sodaß eine Portierung nötig wird.
- Es werden Alternativen zur Einbindung dynamischer Seite untersucht. In die engere Wahl kommen:
  - Installation eines vorkonfektionierten WWW-Servers auf dem Nutzer-Rechner
  - Ersatz des WWW-Servers durch eine Eigenentwicklung für die Ankopplung von Programmen (z.B. Plug-in).

Im Rahmen des Vorhabens GLOBUS III wurden dazu folgende Arbeiten geleistet:

- Prototypische Produktion einer CD-ROM, welche nur die statischen Seiten umfaßt;
- Installation und Test einer lokalen Version von AlfaWeb;
- Inbetriebnahme eines CD-ROM-Brenners.

Die Produktion einer CD-ROM mit statischen Seiten wurde durchgeführt, um praktische Erfahrungen mit dem doch noch nicht unproblematischen Brennen von CD-ROMs zu gewinnen und die Verwendbarkeit der produzierten Scheiben unter verschiedenen Zielumgebungen zu testen. Als kritischer Punkt erwiesen sich die verwendeten Dateinamen bei AlfaWeb. Derzeit werden beliebig lange, möglichst vielsagende Dateinamen verwendet, die den Namenskonventionen des Unix-Dateisystems entsprechen. Für die Produktion von CD-ROMs gibt es nun verschiedene Namenskonventionen, die von einer DOS-ähnlichen rigiden Konvention (ISO Level-1) bis zu einer relativ freien Unix-ähnlichen reichen. Die Verwendbarkeit einer CD-ROM in verschiedenen Umgebungen ist umso größer, je stärker die Einschränkung bei den Dateinamen ist. Die AlfaWeb-CD-ROMs wurden mit den Original-Dateinamen produziert. Erfolgreich getestet wurden diese unter Windows 95, Windows NT 4.0, Sun Solaris 2.5 und DEC Unix 4.0. Nicht verwendbar sind sie unter Windows 3.x und

Windows NT 3.51[7]. Eine Umsetzung der Dateinamen auf die ISO-Konvention ist möglich, wobei allerdings auch die in den Dateien verwendeten URLs geändert werden müssen, doch wurde wegen der inzwischen erfolgten Festlegung der Zielumgebung auf Windows NT 4.0 und Windows 95 davon Abstand genommen.

Als weitere Vorarbeit für eine zukünftige AlfaWeb-CD-ROM wurde eine vollständige, lokale Version von AlfaWeb installiert und getestet. Zu diesem Zweck wurde ein vorhandener Rechner der LfU (Sun SPARCstation 10) hardwaremäßig aufgerüstet, das Betriebssystem auf den neusten Stand gebracht (Solaris 2.5) und AlfaWeb auf der Festplatte installiert. Die WWW-Funktionalität wurde über den Netscape Navigator 3.0 und den Apache Server bereitgestellt. Es konnte damit gezeigt werden, daß ein WWW-Informationssystem auch vollständig lokal betreibbar ist. Mit dieser Stand-Alone-Version wurde AlfaWeb im Rahmen der 10. Fortbildungsveranstaltung „Altlasten / Grundwasserschadensfälle" am 1/2. Okt. 1996 in Ilshofen den zuständigen Fachleuten der Verwaltung des Landes vorgestellt /4/.

Um weitere CD-ROMs in kleiner Serie produzieren zu können, insbesondere für eine geplante, breiter angelegte Testaktion, wurde eine entsprechende Umgebung eingerichtet. Dazu wurde an einen vorhandenen leistungsfähigen PC unter Windows NT 3.51 ein CD-Brenner (Yamaha CDE 100II) angeschlossen und die notwendige Brennsoftware (Elektroson Gear und alternativ Adaptec EZ-SCSI CD-Writer) installiert.

4.5 Systemverwaltungskomponente

Wenn AlfaWeb in der Praxis eingesetzt werden soll, ist es erforderlich, daß die Fachabteilung in die Lage versetzt wird, das System selbständig weiter zu pflegen, um z.B. neue Berichte einzubringen. Dazu wird ein benutzerfreundliches Werkzeug benötigt, das über eine menü- und formular-basierte Schnittstelle die Verwaltung des Systems ermöglicht. Im Einzelnen hat die Systemverwaltung folgende Aufgaben:

- Eingabe und Pflege von Metadaten zu den Berichten (z.B. Berichtsreihe, Nummer, Titel, Untertitel, Autor, ...);
- Parametrisierung und Ausführungen der diversen Programme zur Konvertierung, Verschlagwortung etc. der Berichte;
- Generierung der WWW-Seite für den Zugang zu AlfaWeb über Berichtsreihe und Titel (Literaturliste);
- Generierung der Metadaten für den Umweltdatenkatalog (UDK);
- Prüfung der Konsistenz der Verweise innerhalb AlfaWebs.

Um die Durchgängigkeit in AlfaWeb zu wahren, wird die Systemverwaltungskomponente ebenfalls auf WWW-Basis implementiert. Es werden überwiegend dynamische Seiten verwendet, wobei für bestimmte Zwecke auch Java (z.B. Auswahl eines Verzeichnisses für einen neuen Bericht) eingesetzt wird. Die Systemverwaltungskomponente kann im Gegensatz zum eigentlichen Informationssystem nicht frei zugänglich sein, sondern wird wegen der Sensibilität der Daten nur für den AlfaWeb-Systemverwalter freigegeben. Es ist vorgesehen, die Pflege von AlfaWeb auf einer dedizierten Maschine der Abt. 5 vorzunehmen und erst nach

[7] Windows NT 3.51 kennt lange Pfadnamen, sodaß die Nichtverwendbarkeit u.U. am CD-ROM-Treiber der Testmaschine lag.

einer expliziten Freigabe die neue Version auf den internen und externen Server der Verwaltung zu überspielen.

Die Startseite eines ersten Prototyps, der zur Veranschaulichung der Benutzeroberfläche dient und noch nicht über die eigentliche Funktionalität verfügt, zeigt Abbildung 7.

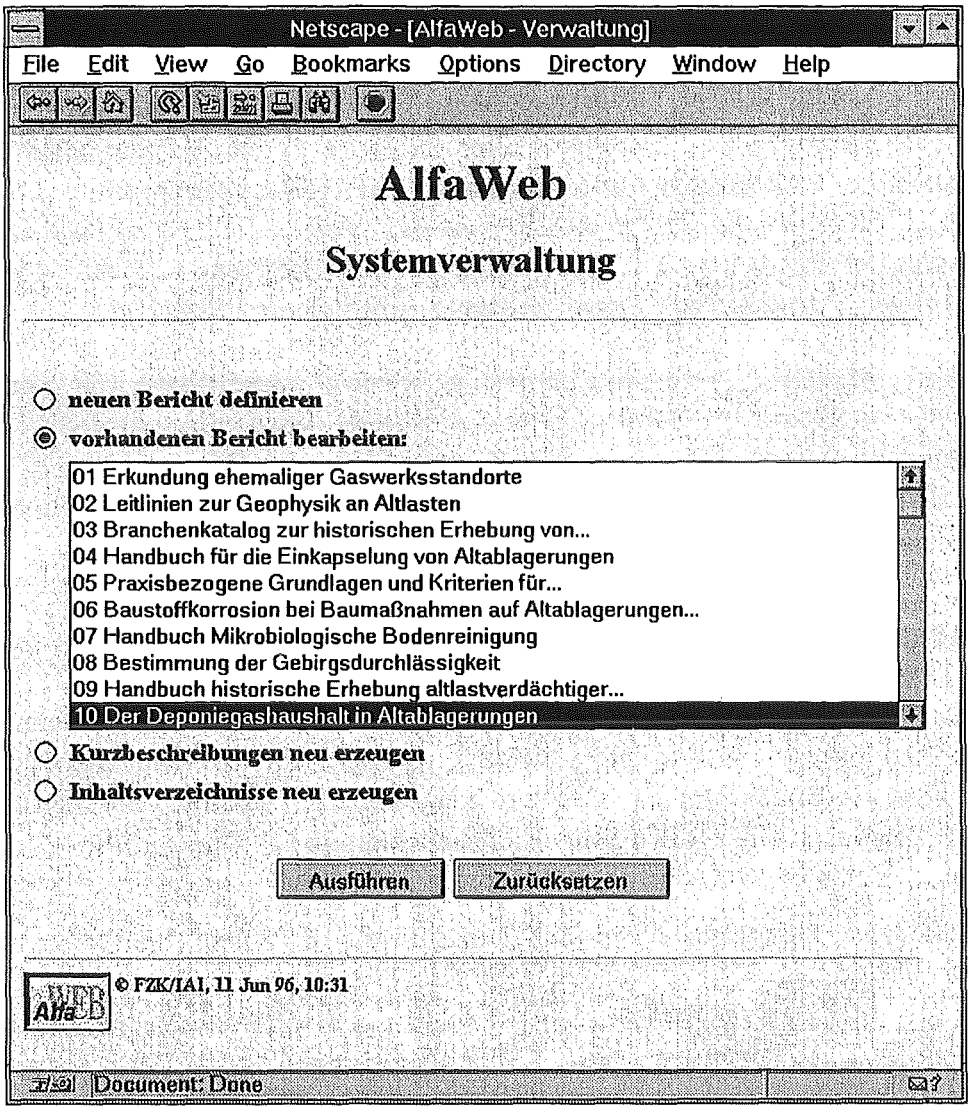


Abbildung 7: Startseite der AlfaWeb-Verwaltung

Abbildung 8 zeigt am Beispiel des Berichts „Der Deponiegashaushalt in Altablagerungen" einen Teil der zu einem Bericht zu verwaltenden Metadaten, die über die Systemverwaltungskomponente erfaßt und geändert werden müssen.

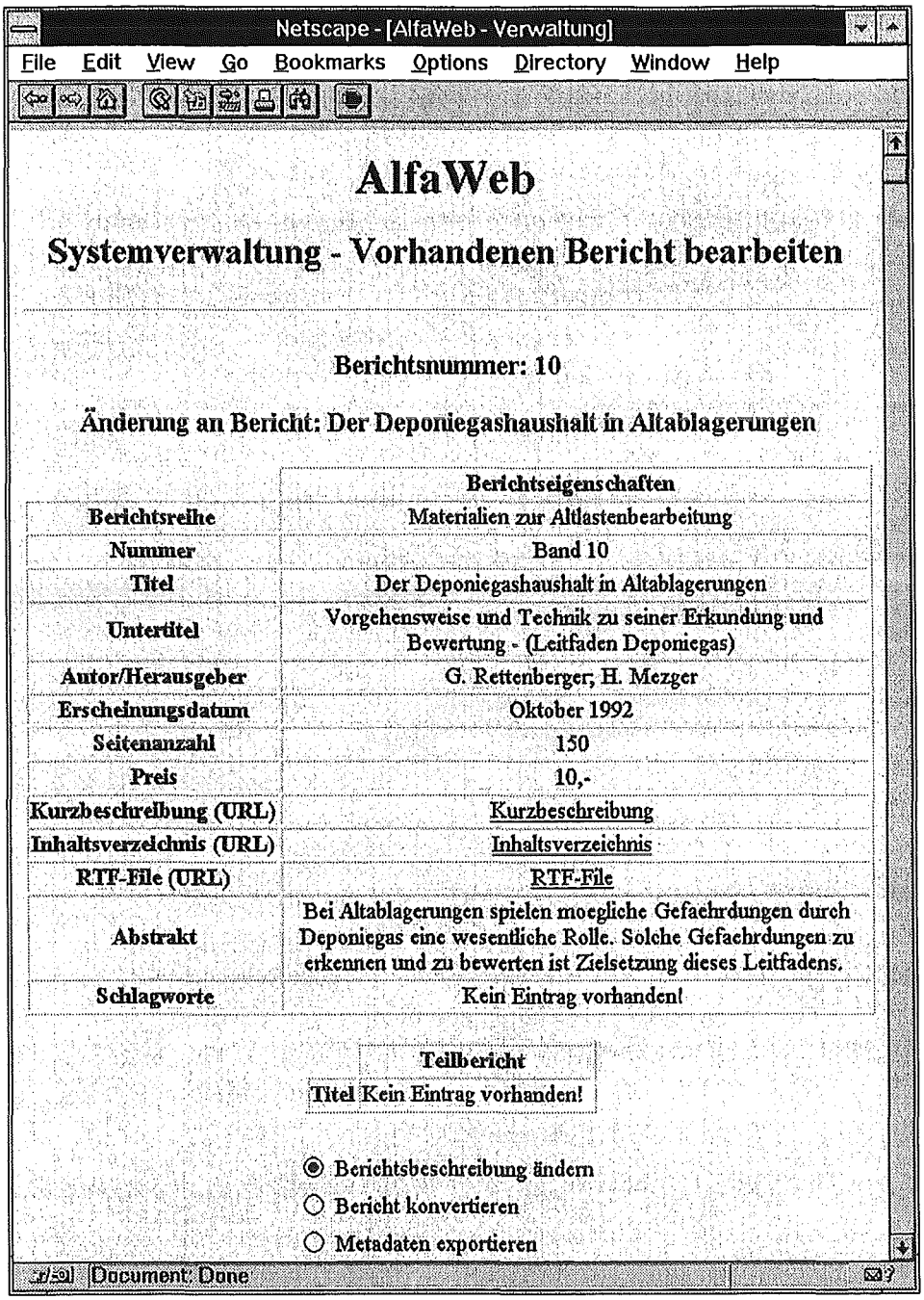


Abbildung 8: Änderung einer Berichtsbeschreibung (Beispiel)

5. Zugriffsmöglichkeiten auf AlfaWeb

AlfaWeb ist bereits seit Beginn der Entwicklung im Internet verfügbar und kann von Interessenten genutzt werden. Neben der in Kapitel 4.4 beschriebenen lokalen Demo-Version existieren 3 verschiedene Versionen für unterschiedliche Verwendungszwecke (s. Tab. 2).

AlfaWeb-Version	URL	Verfügbarkeit	Verwendung
interner LfU-Server (ITZ Karlsruhe)	`http://www.lfuka.um.bwl.de/www_public/abt5/altlasten/`	Intranet	für den schnellen verwaltungs-internen Zugriff
externer UVM-Server (ITZ Stuttgart)	`http://www.uis-extern.um.bwl.de/lfu/abt5/altlasten/`	Internet	offizielle Version
FZK-Entwicklungs-version	`http://www.iai.fzk.de/~weideman/lfu/`	Internet	Test- und Entwicklungs-version

Tabelle 2: Zugriffsmöglichkeiten auf AlfaWeb

Die aktuellste Version, in die auch neue Berichte eingespielt werden, ist die Entwicklungsversion des FZK. In bestimmten Zeitabständen werden die übrigen Versionen aktualisiert, d.h. die Entwicklungsversion wird auf ein DAT-Band kopiert und im ITZ eingespielt. Um den problemlosen Austausch zu ermöglichen, wurden eine Reihe von Maßnahmen ergriffen, um die Umgebungs-Abhängigkeiten zu minimieren:

- In Absprache mit dem ITZ wurde die Grundstruktur des Systems festgelegt und vereinheitlicht;
- Die URLs und Pfadnamen der Einstiegsseiten wurden verbindlich festgelegt;
- Statische Seiten sind relativ zur Einstiegsseite verzeigert;
- Notwenige absolute Pfadangaben in CGI-Skripten (konkret: Perl-Pfad) werden über ein Hilfsprogramm umgesetzt;
- Alle CGI-Skripts laufen über einen Wrapper, der allein die absoluten Adressen kennt (URL der Einstiegsseite, CGI-Verzeichnis, AlfaWeb-Verzeichnis) und diese über Umgebungsvariable den eigentlichen Programmen zur Verfügung stellt.

Neben AlfaWeb selbst, ist auch das bereits erwähnte Konvertierungsprogramm verfügbar. So ist dies zur Verwendung durch die Abt. 5 der LfU auf dem in Kap. 4.4 erwähnten Sun-rechner, der auch die Stand-Alone-Version von AlfaWeb beherbergt, installiert. Andere Interessenten können die Dokumentvorlage und die zugehörige Richtlinie ebenso wie das Konvertierungsprogramm auf Anfrage erhalten.

Um das Interesse an AlfaWeb zu dokumentieren, werden seit Anfang 1996 Zugriffe auf die

Entwicklungsversion protokolliert und ausgewertet. Die folgende Tabelle faßt die wichtigsten Angaben zusammen:

Zeitraum	***von:***	01.01.1996
	bis:	02.11.1996
abgerufene Seiten:		119133
Rechner:		6402
Sitzungen:		15633

Tabelle 3: Zugriffsstatistik

Seit Anfang des Jahres wurden über 100.000 Seiten abgerufen. Angefordert wurden die Informationen von mehr als 6000 verschiedenen Rechnern[8]. Insgesamt fanden über 15000 Sitzungen statt, wobei wir willkürlich alle Zugriffe eines Rechners an einem Tag zu einer Sitzung zusammengefaßt haben. Interne Zugriffe aus dem Institut für Angewandte Informatik des FZK sind in der Statistik nicht mit erfaßt. In den letzen Wochen erfolgten meist zwischen 2000 und 3000 Zugriffe pro Woche (siehe Abb. 9). Während Anfang des Jahres die meisten Nutzer nur einmalig wenige Seiten abgerufen haben, gibt es inzwischen zunehmend „Stammkunden", die öfter und intensiver mit AlfaWeb arbeiten.

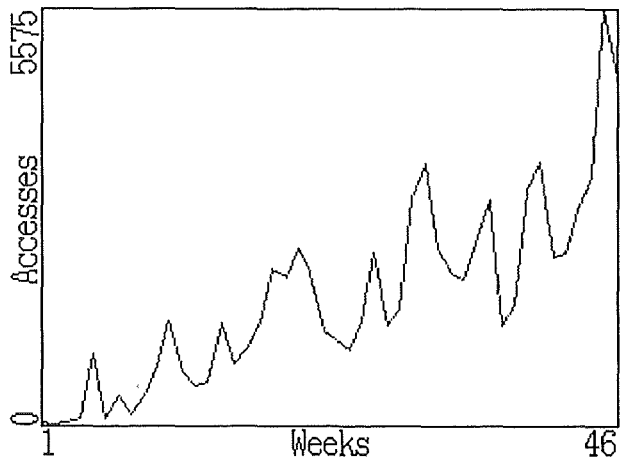


Abbildung 9: Zugriffsstatistik der AlfaWeb-Entwicklungsversion

Bei einer genaueren Auswertung fällt auf, daß ein beträchtlicher Anteil der Zugriffe (20-25%) von Robotern erfolgt. Das sind Programme, welche die Datenbanken der Suchmaschinen des WWW (Altavista, Lycos etc.) mit Inhalt füllen, indem sie automatisch das WWW nach Informationsangeboten absuchen. Dies hat zur Folge, daß Informationen zu AlfaWeb (Startseite, z.T. auch detailliertere Angaben) über praktisch alle großen deutschen und internationalen Suchmaschinen gefunden werden können, ohne daß wir dort einen Eintrag

[8] Genauer: von verschiedenen TCP/IP-Adressen. Während normalerweise eine TCP/IP-Adresse einem Rechner fest zugeordnet ist, vergeben einige Internet-Provider TCP/IP-Adressen aus ihrem Kontingent dynamisch an die gerade angemeldeten Kunden, sodaß für diese keine eindeutige Zuordnung möglich ist.

veranlaßt haben. Nur so erklärt sich auch die weltweite Verteilung der Nutzer. Eine genaue Lokalisierung ist zwar nicht möglich, jedoch läßt sich über den letzten Teil der TCP/IP-Adresse[9] eine einigermaßen verläßliche Länderzuordnung treffen. Die meisten Zugriffe erfolgen wegen den deutschsprachigen Informationen in AlfaWeb naturgemäß aus Deutschland (Länderkennung de). An zweiter Stelle folgen Zugriffe mit der Kennung „com". Dies ist keine Länderkennung, eine exakte Zuordnung ist nicht möglich. Diese Kennung wird von kommerziellen Internetnutzern hauptsächlich in den USA, aber auch von internationalen Konzernen verwendet. So laufen über diese Kennung alle Zugriffe der großen Internet-Provider (Compuserv, AOL) auch aus Deutschland und ebenso wird diese von den meisten Robotern verwendet. Über „de" und „com" erfolgen mehr als 75% aller Zugriffe. Tabelle 4 zeigt eine Übersicht der in den Zugriffsprotokollen von AlfaWeb gefundenen Länderkennungen.

Kennung	Land		Kennung	Land
ar	Argentinien		is	Island
at	Österreich		it	Italien
au	Australien		jp	Japan
be	Belgien		kr	Südkorea
br	Brasilien		lu	Luxemburg
ca	Kanada		mc	Monaco
ch	Schweiz		mil	USA
com	USA oder Internet-Provider		mx	Mexiko
cr	Costa Rica		my	Malaysia
cz	Tschechien		ni	Nicaragua
de	Deutschland		nl	Niederlande
dk	Dänemark		no	Norwegen
edu	meist USA		nz	Neuseeland
ee	Estland		org	meist USA
fi	Finnland		pl	Polen
fr	Frankreich		ru	Russland
gov	USA		se	Schweden
gr	Griechenland		sg	Singapur
hu	Ungarn		si	Slovenien
id	Indonesien		sk	Slovakei
ie	Irland		uk	Großbritannien
			....und weitere	

Tabelle 4: Herkunft der AlfaWeb-Nutzer

[9] Soweit diese nicht als TCP/IP-Adresse (z.B. 141.52.44.8) im engeren Sinne vorliegt, sondern als Domänenname (z.B. uisun4.iai.fzk.de).

6. Literatur

/1/ Mayer-Föll, R.; Jaeschke, A.(Hrsg.) (1995): Projekt GLOBUS - Konzeption und prototypische Realisierung einer aktiven Auskunftskomponente für globale Umweltsachdaten im Umweltinformationssystem Baden-Württemberg - Phase II 1995, Forschungszentrum Karlsruhe, wissenschaftliche Berichte, FZKA 5700.

/2/ Weidemann, R.; Geiger, W.; Jaeschke, A.; Reißfelder, M. (1995): Entwicklung eines WWW-basierten Altlasten-Informationssystems, in /1/, S. 271 - 297.

/3/ Geiger, W.; Reißfelder, M; Weidemann, R. (1996): Das WWW-basierte Altlasten-Fachinformationssystem AlfaWeb, in Lessing, H.; Lipeck, U.W. (Hrsg.) (1996): Informatik für den Umweltschutz, 10. Symposium, Hannover 1996, Metropolis Verlag Marburg, S. 211 - 220.

/4/ Schmid, E.; Reichert, U. (1996): AlfaWeb: Informationstechnik mit Zukunft? 10. Fortbildungsveranstaltung „Altlasten / Grundwasserschadensfälle", Park-Hotel Ilshofen, 1./2. Oktober 1996.

/5/ Weidemann, R.; Geiger, W.; Reißfelder, M. (1996): Richtlinie zur Erstellung WWW-verfügbarer Berichte, (interner Bericht).

/6/ Weidemann, R.; Geiger, W.; Reißfelder, M. (1994): Expertensysteme im Umweltschutz am Beispiel des Altlasten-Expertensystems XUMA, in Schmidt, F.-P. et al. (Hrsg.), Umwelt-Informationssysteme, Kontakt & Studium, Bd. 411, expert-Verlag, Renningen-Malmsheim, S. 71 - 83.

Konsolidierung der serverseitigen Kartendienste und Überführung in die GLOBUS-Betriebsversion

J. Wiesel; W. Weisbrich; M. Böse; J. Feuchter,
Institut für Photogrammetrie und Fernerkundung (IPF),
Universität Karlsruhe,
Englerstr. 7,
76128 Karlsruhe

1. Verbesserungen des Layouts und der Bedienung

1.1 Einleitung

Eine der wesentlichsten Aufgaben des Projektes GLOBUS III ist die Konsolidierung bzw. Stabilisierung des Gesamtsystems WWW-UIS. Als Grundlage diente der Testbericht, der von Fr. Bußmann und H. Heißler erstellt wurde.

1.2 Änderung der Seitenmenüführung

Die Menügestaltung der Graphischen Dienste im Projekt Globus II war auf strikte Führung des Benutzers ausgerichtet. Die zugrundeliegende Strategie sollte dem Benutzer ein wiederholtes Erzeugen von Darstellungen, sowie langes Suchen nach gewünschten Optionen im Menüsystem ersparen.

Eine Folge dieser Strategie ist allerdings, daß unter Umständen eine relativ große Anzahl von Menüseiten abgearbeitet werden mußte, bis erste Visualisierungen sichtbar wurden. Um den Benutzern einen schnelleren Zugriff auf die darzustellenden Daten zu gewähren, wurden Änderungen in der Ablaufstruktur erforderlich. Bei einer solchen Menüführung ist es erforderlich verläßliche Initialisierungsvorschläge berechnen zu können. Solche Defaulteinstellungen erfordern aufwendige Aufträge an die grafikgenerierenden Instanzen. Daher wurde ein schneller Zugriff auf die darstellenden Seiten optional realisiert und es ist weiterhin möglich den strikt geführten Weg zu wählen.

Nach dem Aufruf *Graphische Dienste* wird die Option *Nur Diagramme darstellen* gewählt. Auf der erscheinenden Seite kann ausgewählt werden, welche Parameter an der X-Achse von Diagrammen dargestellt werden sollen. Auf der Folgeseite werden die daraus resultierenden Darstellungsziele angeboten, um den Anwender eine frühzeitige Einschränkungsmöglichkeit anzubieten. Wird diese Menüseite zerbrochen, so muß davon ausgegangen werden, daß das System bei Änderungen der Achsenparameterzuweisungen für die resultierenden Darstellungsziele eine Defaulteinstellung (z. B. *alle Ziele darstellen*) setzt.

Die oben aufgezeigte Abhängigkeit von zwei Menüseiten verhindert ebenfalls ein einfaches Zusammenlegen dieser zwei Seiten. Der Grund liegt darin, weil das Angebot von Darstellungszielen unmittelbar von den Einstellungen der Achsenparameterzuweisung abhängig ist.

Abbildung 1 zeigt schematisch die Menüseitenführung der Graphischen Dienste. Die Knoten entsprechen den dynamisch generierten Seiten.

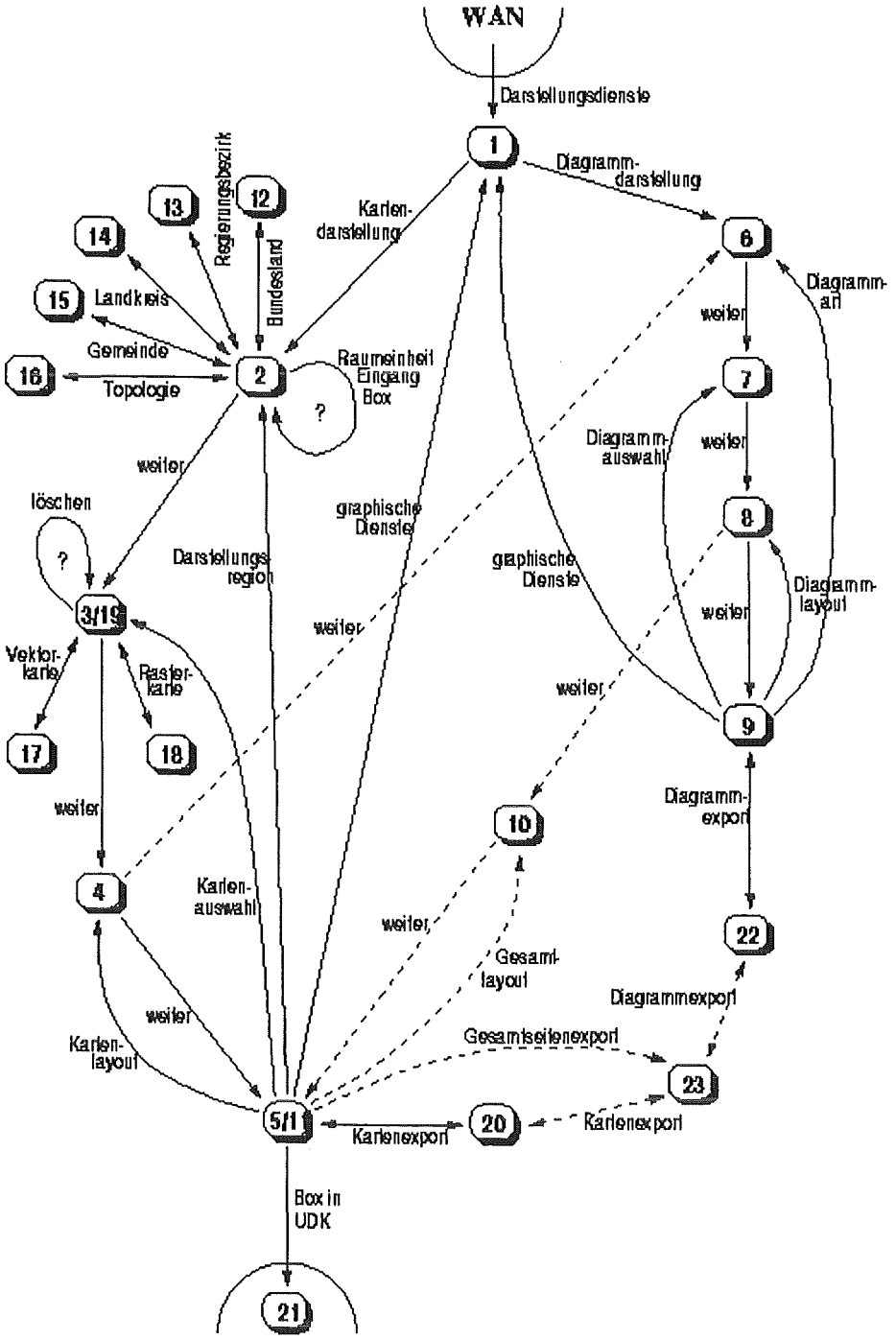


Abbildung 1: alter schematischer Ablauf der Graphischen Dienste

1.3 Die neue Seitenmenüführung im Überblick

Abbildung 2 zeigt schematisch die neue Menüseitenführung der Graphische Dienste. Nach Aufruf der Darstellung besteht die Möglichkeit direkt zu den darstellenden Seiten zu gelangen. Wie in

Abbildung 2 ersichtlich, hat der Benutzer nun die Möglichkeit gleich nach dem Aufruf der Graphischen Dienste auf die Darstellungsseite zu gelangen. Dabei werden Menüseiten übergangen, mit deren Hilfe eine gewünschte Darstellung ausgewählt werden kann. Diese Menüseiten sind keineswegs nur ausschließlich für das Layout der Darstellung zuständig. Auf ihnen wird zunächst festgelegt, welche konkreten Daten visualisiert werden.

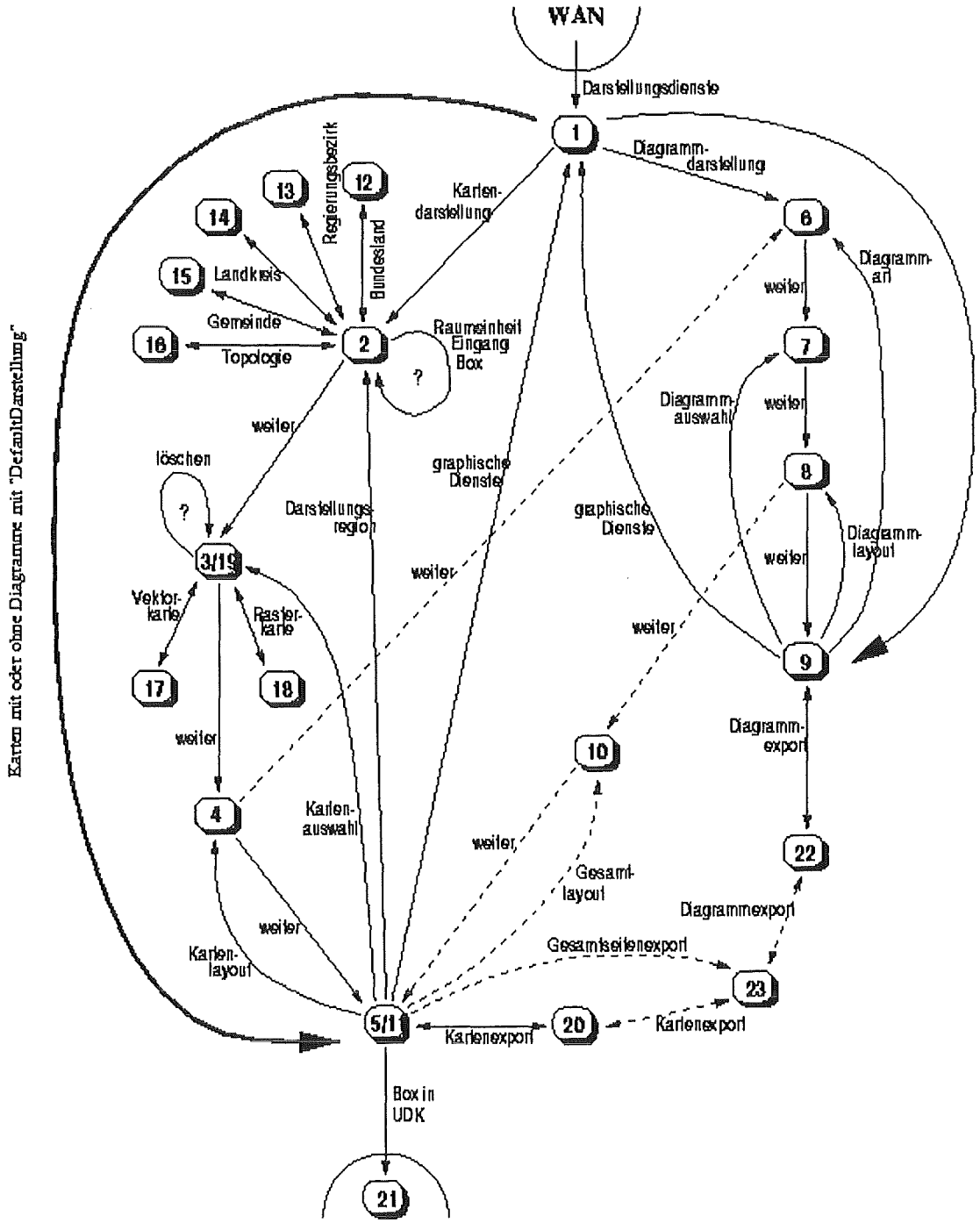


Abbildung 2: neuer schematischer Verlauf der Graphischen Dienste

Die Graphischen Dienste können zwei Arten von Daten darstellen:

- Geodaten und
- skalare Daten

Die Geodaten werden als Karten und die skalaren Daten als Diagramme oder Tabellen visualisiert.Im Zusammenspiel der verschiedenen Dienste des Projektes Globus III werden von den aufrufenden Instanzen Daten zur Visualisierung oder Spezifikationen von anzubietenden Daten erstellt. Bei dem Aufruf der Graphischen Dienste werden z. B. Daten übergeben, mit deren Hilfe eine Boundingbox ermittelt wird. Die Boundingbox gibt einen Bereich an, in denen alle Bezugspunkte der skalaren Daten liegen. Die darzustellende Karte kann nach Benutzerwunsch auf die Boudingboxkoordinaten angepaßt werden. Ebenso wird ein Datenblock skalarer Daten, der für die Darstellung in Diagrammen oder Tabellen geeignet ist, bereitgestellt.

Daher muß eine sinnvolle Einstellung zur Diagrammtypangabe ermittel werden. Hierbei wird festgelegt, welche Parameter an der X-Achse bzw. welcher Meßwerttyp an der Y-Achse dargestellt werden soll.

Über die dynamisch zu ermittelnden Defaults hinaus gibt es zahlreiche Einstellungen, die nicht unmittelbar von den übergebenen Daten abhängen. Hierzu gehören unter anderem die Standardgröße von Diagrammen oder die Farbgestaltung spezifischer Elemente der Kartenvisualisierung.

1.4 Technische Einbindung der Defaultsetzung

Im weiteren wird zwischen statischen und dynamischen Standardeinstellungen unterschieden. Unter einem dynamischen Default soll eine solche Standardeinstellung verstanden werden, die von den übergebenen Daten bzw. Spezifikationen der aufrufenden Instanzen abhängig ist. Dynamische Defaults werden erst nach dem Aufruf der Graphischen Dienste ermittelt.

Da bei der Ermittlung dynamischer Defaults auf die übergebenen Daten der aufrufenden Instanzen eingegangen werden muß, werden die zugehörigen Algorithmen in den jeweiligen Schnittstellenmodulen aktiviert.

Statische Defaults sind von den Daten der aufrufenden Instanzen unabhängig. Diese Standardeinstellungen sind schon vor dem Aufruf der Graphischen Dienste festgelegt. Da nun einige der Menüseiten unter Umständen übersprungen werden, muß diese Aufgabe an eine zentrale Stelle.des Softwaresystems verlagert werden. Um eine möglichst hohe Anpassungfähigkeit des statischen Defaultsystems zu gewährleisten, wurden die Defaulteinstellungen nicht fest einprogrammiert, sondern in Form einer Sessiondatei abgelegt. Diese spezielle Sessiondatei wird beim Erzeugen einer neuen Session automatisch eingelesen. Wird das Menüsystem geändert, kann von dem Administrator relativ einfach eine neue Sessiondatei mit entsprechenden Defaults erzeugt werden.

1.5 Setzen der dynamischen Defaults

Bei Aufruf der Graphischen Dienste werden im wesentlichen zwei Dateien übergeben. Eine Datei enthält die skalare Daten in tabellarischer Form und die andere Datei beinhaltet die Metainformationen zu den skalaren Daten. Zu den Metainformationen gehören unter anderem die Beschreibungen von Wertebereichen oder Spaltentypen.

Aus diesen Daten sind nun möglichst sinnvolle Voreinstellungen für die Visualisierung zu ermitteln. Dabei sollten die Defaulteinstellungen so gewählt sein, damit einerseits keine allzu große Belastungen des Servers entstehen, und andererseits aber möglichst viel Informationen graphisch dargestellt werden.

Zu ermitteln sind:

- mögliche X-Achsentypen für Diagramme
- mögliche Y-Achsentypen für Diagramme
- der Default-X-Achsentyp
- der Default-Y-Achsentyp
- mögliche Darstellungsziele
- die Default-Darstellungsziele

1.5.1 Mögliche X-Achsentypen für Diagramme

An X-Achsen von Diagrammen können Elemente aus Spalten vom Typ

- ORT
- ZEIT
- PARAMETER

eingetragen werden. Alle Spalten dieses Typs werden als mögliche X-Achsentypen anerkannt, falls sie nicht die Ausprägung "eins" besitzen.

Eine Spalte wird dem Typ ORT zugeschrieben, falls die Spaltenart die Kennung "coord", "area-id" oder "point-id" enthält.

Der Spaltentyp ZEIT wird durch die Kennung "time*" erkannt. Dabei drückt der "*" eine beliebige Zeichenfolge aus.

Eine Spalte wird dem Typ PARAMETER zugewiesen, falls die zugehörige Kennung "parameter" ist.

1.5.2 Mögliche Y-Achsentypen für Diagramme

An Y-Achsen von Diagrammen können Elemente aus Spalten vom Typ

- INTEGER
- FLOAT

angetragen werden. Mögliche Y-Achsentypen sind alle Spalten vom Typ INTEGER oder FLOAT,

die in den Metadaten zusätzlich als Darstellungsziel markiert sind. Dabei enthalten die Spalten vom Typ INTEGER ausschließend ganze Zahlen, während Elemente von Spalten des Typs FLOAT auch Nachkommastellen enthalten dürfen. Die oben aufgeführten Spalten werden durch die Kennung "integer" bzw. "float" erkannt.

1.5.3 Der Default-X-Achsentyp

Vom System wird derjenige X-Achsentyp gewählt, der in einer minimalen Menge von möglichen Darstellungszielen resultiert. Dabei wird davon ausgegangen, daß Spalten mit nicht diskreten Werten eine sehr hohe Ausprägung bestizen.

In den Metadaten sind solche Spalten mit den Wertebereichen "KONT" oder "x-y" gekennzeichnet, wobei x und y zwei unterschiedliche Strings darstellen.

Die Grundidee besteht darin, die graphikgenerierenden Instanzen mit nicht zu großen Aufträgen zu belasten. Daher wird der Server möglichst wenig belastet und der Benutzer muß nicht zu lange auf den ersten Systemvorschlag warten.

1.5.4 Der Default-Y-Achsentyp

Der erste mögliche Y-Achsentyp wird als Defaulteinstellung verwendet. Der Y-Achsentyp ist von der Anzahl resultierender Darstellungsziele unabhängig. Daher ist die Wahl dieses Typs zumindest vom zeitlichen Aufwand und damit von der Belastung des Servers unabhängig.

1.5.5 Mögliche Darstellungsziele

Mögliche Darstellungsziele ergeben sich aus

- den möglichen X-Achsentypen
- dem Default-X-Achsentyp
- dem Default-Y-Achsentyp

Darstellungsziele werden jeweils durch einen Tupel von Elementen aller X-Achsentypen beschrieben, die nicht Default-X-Achsentyp sind. In einem Tupel ist jede Komponente aus Spalten mit verschiedenen X-Achsentypen. Darstellungsziele sind alle möglichen Tupel von Elementen, die nach obiger Vorschrift gebildet werden können.

Durch die Wahl des Y-Achsentyps sind damit Urbild- und Bildbereich aller Darstellungsziele eindeutig erklärt. Der Default-Y-Achsentyp gilt für alle Darstellungsziele.

1.5.6 Die Default-Darstellungsziele

Es werden alle möglichen Darstellungsziele als Default gewählt. Aus Sicht der Serverbelastung und dem daraus resultierenden zeitlichen Aufwandes sind diese Defaulteinstellungen tragbar, da der X-Achsentyp so gewählt ist, daß eine minimale Anzahl von Darstellungszielen entsteht.

2. Sessionverwaltung

Im Rahmen der Änderungen der Graphischen Dienste wurde eine weitere mächtige Erweiterung realisiert - die Sessionverwaltung. Mithilfe dieses Moduls kann ein Anwender die erarbeiteten Ergebnisse der Graphischen Dienste permanent machen. Hierbei werden sämtliche Einstellungen und Daten dieser Session abgespeichert und können zu einem späteren Zeitpunkt wieder reaktiviert werden. Soll eine Session zu einem späteren Zeitpunkt wieder hergestellt werden, so gehen auch bei längeren Benutzerzyklen die zuvor erbrachte zeitliche Aufwendungen des Benutzers nicht verloren.

Hierzu wurden zwei zusätzliche Module in die Graphischen Dienste integriert:

- Modul zum Sichern von Session *savesession.tcl*
  Dieses Modul sichert die aktuelle Session unter dem übergebenen Sessionname. Sollte eine Session mit diesem Namen existieren, wird sie ohne Rückfrage überschrieben.

- Modul zum Laden von gespeicherte Sessions *loadsession.tcl*
  Dieses Modul listet die Namen der gespeicherten Sessions des Benutzers in einer Listbox auf. Und durch Auswählen eines Session-Namens kann diese reaktiviert werden.

Dieses Möglichkeit der Sessionverwaltung wurde wie folgt realisiert:

Von der Darstellungsseite aus kann eine laufende Session mit Hilfe der Seite *Session sichern* permanent gemacht werden. Auf dieser Seite werden bereits gesicherte Sessions dieses Benutzers, sowie ein Namensvorschlag für die zu speichernde Session dargestellt. Es ist auch möglich einen anderen Namen als den vorgeschlagenen zu wählen.

Von der Startseite und von der Darstellungsseite aus können von dem Benutzer die zu einem früheren Zeitpunkt gespeicherten Sessions wieder reaktiviert werden. Die zugehörige Menüseite kann durch den Aktionsknopf *Sessionverwaltung* aufgerufen werden. Auf der Seite **Sessionverwaltung** werden die von diesem Anwender bereits gespeicherten Sessions in einer Listbox angezeigt und durch Auswählen kann eine frühere Session wieder aufgenommen werden. Dort ist es dem Benutzer ebenfalls möglich seine eigenen alten gespeicherten Sessions zu löschen.

Da jeweils nur Zugriff auf die Sessions des Benutzers gewährt wird und auch außer der laufenden Session des Benutzers keine weitere Session beteiligt ist, arbeitet diese Funktionalität lokal.

Die Abbildung 3 zeigt die Seite **Aktuelle Session sichern,** in der die bereits vorher gesicherten Sessions des Benutzers aufgelistet werden, und schlägt einen Namen für die Sicherung der aktuellen Session vor, der vom Benutzer akzeptiert oder modifiziert werden kann.

Die Abbildung 4 zeigt die Seite **Sessionverwaltung**, welche die gesicherten Sessions des Benutzers auflistet. Der Benutzer hat die Möglichkeit über den Aktionknopf *Session laden* eine ausgewählte Session zu laden oder über den Aktionsknopf *Session löschen* eine oder mehrere ausgewählte Sessions zu löschen.

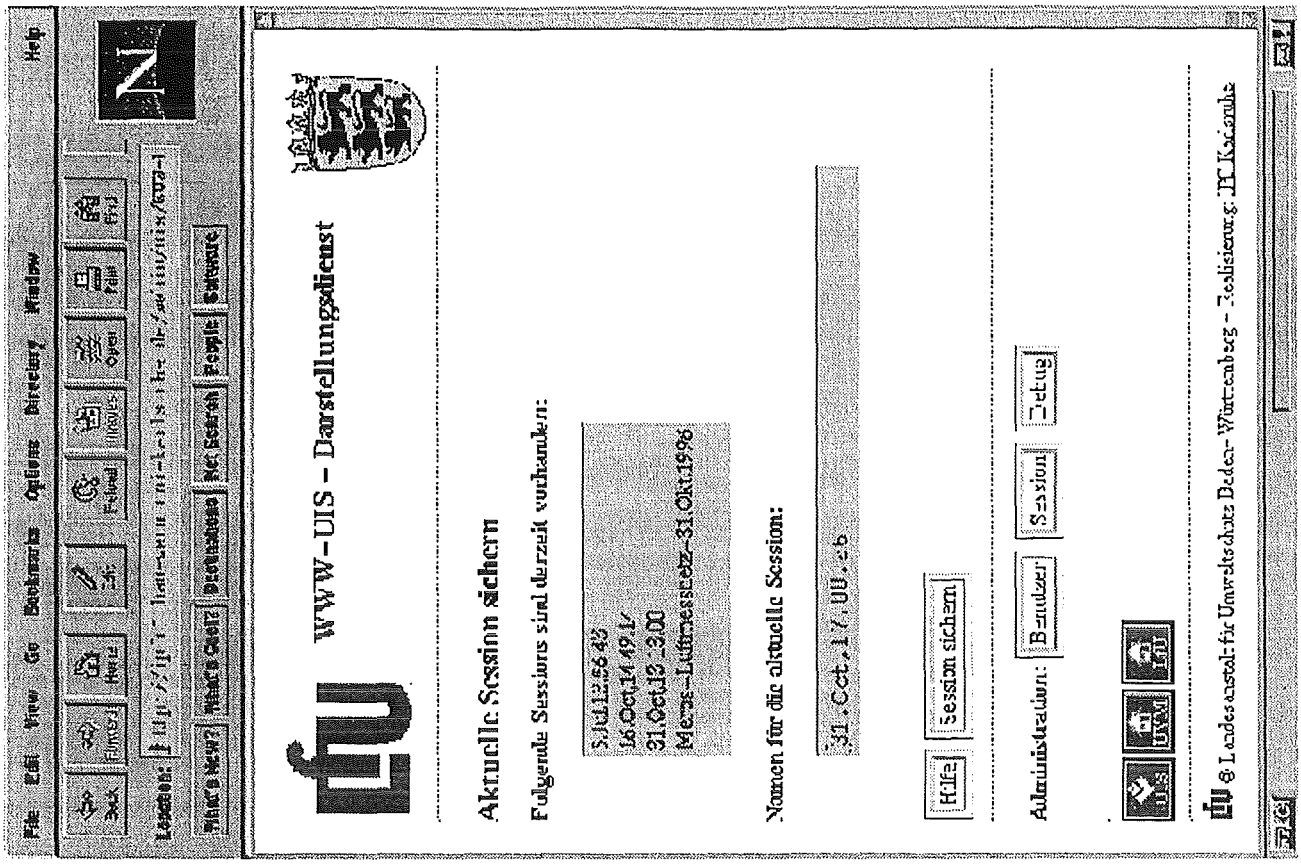


Abbildung 3: Die Seite Aktuelle Session sichern

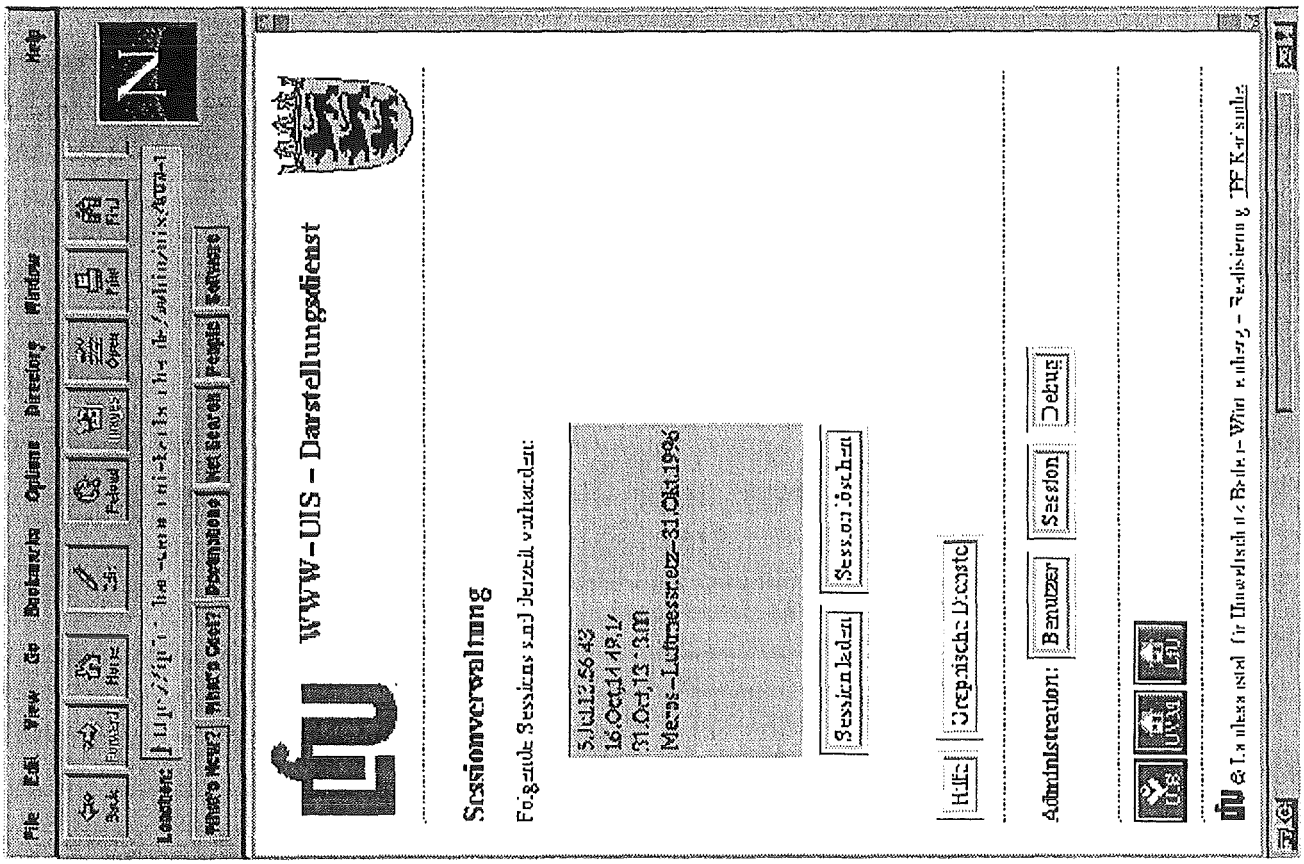


Abbildung 4: Die Seite *Sessionverwaltung*

3. Administration der Graphischen Dienste

3.1 Verschiedene Betriebsmodi der Graphischen Dienste

Bei der Weiterentwicklung der Graphischen Dienste hat es sich bewährt, dem Entwickler bzw. dem Administrator einige besondere Seiten zur Verfügung zu stellen.

Der Anwender darf weder durch interne Informationen des Systems verwirrt werden, noch die Möglichkeit haben, systemweite Einstellungen zu ändern. Daher ist es möglich die Graphischen Dienste in zwei Modi zu betreiben, in dem bereits bekannten Anwender-Modus und im Administrations-Modus.

3.1.1 Der Administrations-Modus

Der Developer-Modus wird erreicht, in dem bei dem Aufruf der Graphischen Dienste eine Hidden-Variable mit Namen *Magic* übergeben wird, die eine vordefinierten Wert besitzt. Dieser Wert kann vom Administrator modifiziert werden. Anschließend erscheint die Startseite der Graphischen Dienste, die wie in Abbildung 5 sichtbar eine zusätzliche Unterteilung mit Namen **Administration** beinhaltet. Darunter werden folgende 3 Aktionsknöpfe angezeigt:

- Benutzer
- Session
- Debug

Zusätzlich zu den aufgeführten 3 Aktionsknöpfen wird dem Administrator sobald die Darstellungsseite sichtbar ein weiterer Aktionsknopf

- Defaultwerte speichern

aufgelistet.

Die Abbildung 6 zeigt einen Ausschnitt der **Darstellungsseite** im Administrationsmodus.

3.1.1.1 Der Aktionsknopf *Benutzer*

Bei Betätigen dieses Aktionsknopfes erscheint die in Abbildung 7 dargestellte Seite **Administration der Benutzerverzeichnisse**. Hier werden die Benutzerverzeichnisse der Anwender aufge-listet. Der Administrator kann die in der Listbox ausgewählten Benutzerverzeichnisse und Benutzerdateien löschen.

3.1.1.2 Der Aktionsknopf *Session*

Hierbei erscheint die Seite **Administration der Benutzersessions** (Abbildung 8). Hier werden die Sessionverzeichnisse der Anwender aufgelistet. Durch Auswählen in der Listbox ist es dem Administrator möglich selektiv Benutzersessions zu löschen.

3.1.1.3 Der Aktionsknopf *Debug*

Bei Betätigen dieses Aktionsknopfes wird eine Debug-Seite erzeugt, die die lokalen und globalen Variablen der zugehörigen Web-Seite sowie allgemeine Verwaltungsvariablen der HTML-Toolkit-Umgebung enthält. Diese als HTML-Seite dient zur Fehlerfindung und Fehlerbehebung für den Entwickler bzw. Administrator.

3.1.1.4 Der Aktionsknopf *Defaultwerte speichern*

Auf der Darstellungsseite wird dem Administrator zusätzlich die Möglichkeit der Defaulteinstellung geboten. Hierzu löst der Administrator auf der Darstellungsseite den Aktionsknopf *Defaultwerte speichern* aus und es erscheint die Menüseite **Einstellung der statischen Defaults** (Abbildung 9). Auf dieser Seite können alle statischen Einstellungen der gerade laufenden Session permanent gemacht werden. Alle weiteren Sessions starten nun mit den Defaultwerten, die den Einstellungen dieser Session entsprechen. Dieser spezielle Dienst wirkt systemweit. Vor dem Benutzen dieses Dienstes sollte der Administrator alle Menüseiten, die bei dem Dienst *Diagramme und Karten darstellen* im Modus *Zuvor Darstellungsattribute einstellen* auf der Startseite der *Graphischen Dienste* erzwungen werden, mindestens einmal aufgerufen werden. Hier werden folgende Einstellungen festgelegt:

- Auswahl der darzustellenden Region
- Auswahl der Karten, Vektorkarten und Punktkarten
- Größe der Karte
- Farbe und Strichbreite der ausgewählten Vektorkarten
- Farbe der ausgewählten Punktkarten
- Defaulteinstellung der X-Achse und Y-Achse der Diagramme
- Darstellungsart der Diagramme (Kurven- oder Balkendiagramm)
- Größe der Diagramme in der Übersichtsdarstellung
- Größe der Diagramme in der Vollbilddarstellung
- automatische Skalierung der Diagramme
- Festlegung des Gesamtlayouts von Diagrammen und Karte
- Größe und Farbe der Symbole von Diagrammen mit Bezug zur dargestellten Karte

Sobald diese Einstellungen festgelegt sind, kann der Administrator mit Hilfe des Aktionsknopfes *Defaultwerte speichern* diese zu Defaulteinstellungen zu definieren.

Jeder Anwender, der mit dem Modus *Defaultdarstellung* auf der Startseite der Graphischen Dienste die Darstellungsseite aufruft, erhält dieses voreingestellte Layout. Es sollte darauf geachtet werden, daß bei der Defaulteinstellung für die Auswahl von Karten keine Karten verwendet werden sollen, die einen hohen Speicherplatzbedarf auf der Serverseite besitzen. Es könnte der Fall sein, daß mehrere Benutzer zeitgleich auf den Server mit dem Modus *Defaultdarstellung* zugreifen und dies würde einer größeren Belastung des Servers bedeuten und eine langsamere Abarbeitung der Session zur Folge haben. Sollte der Benutzer mit Eingangswerten, die den gesetzten Defaulteinstellungen widersprechen, wird die Darstellungseite mit den Eingangswerten des Benutzers initialisiert.

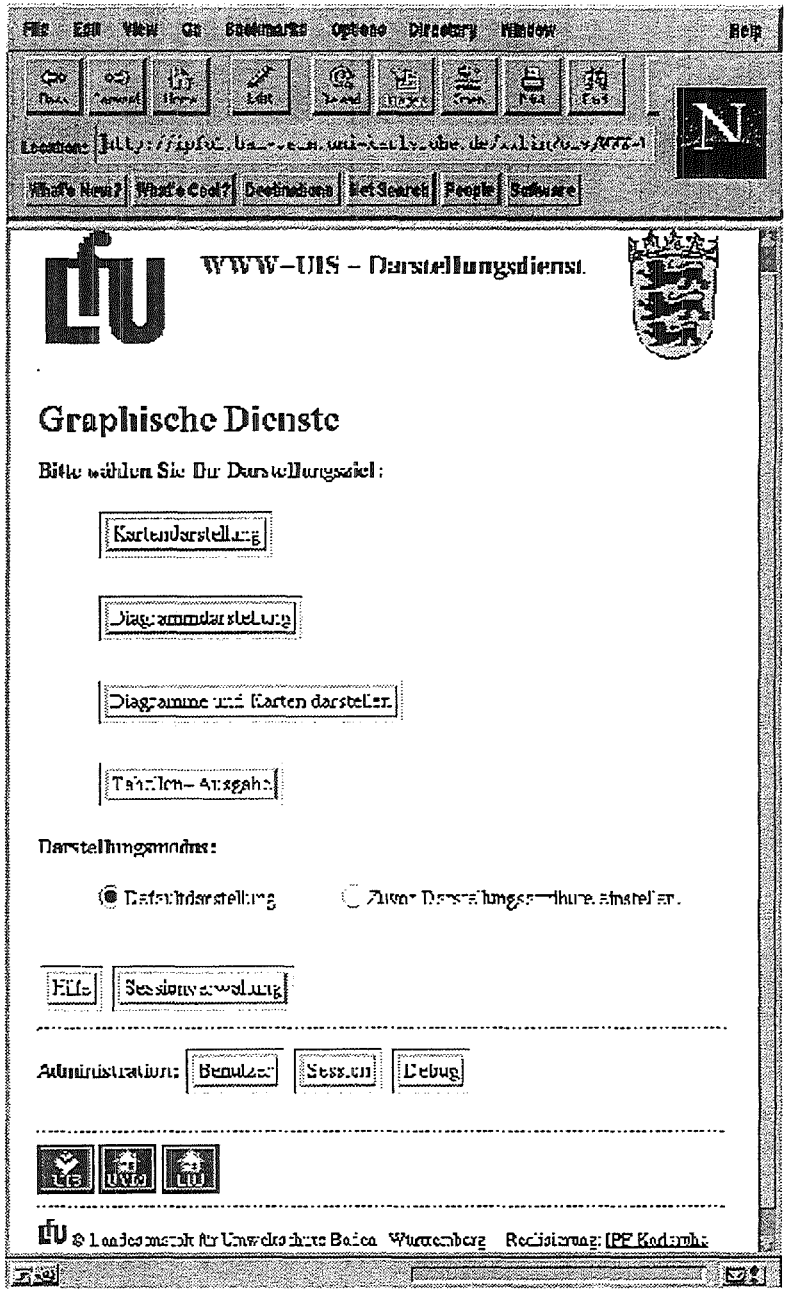


Abbildung 5: Die Startseite der Graphischen Dienste im Administrationsmodus

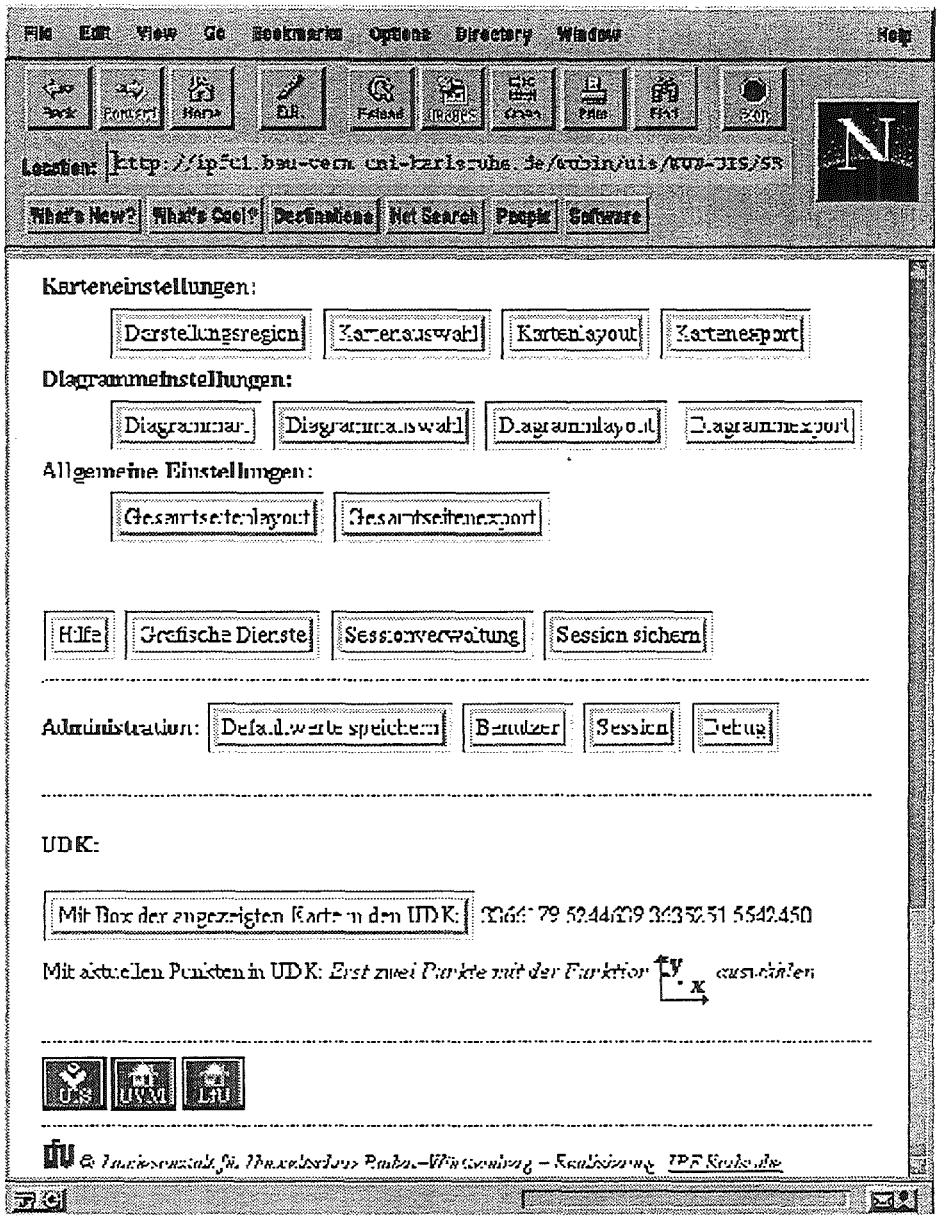


Abbildung 6: Ausschnitt der Darstellungsseite im Administrationsmodus

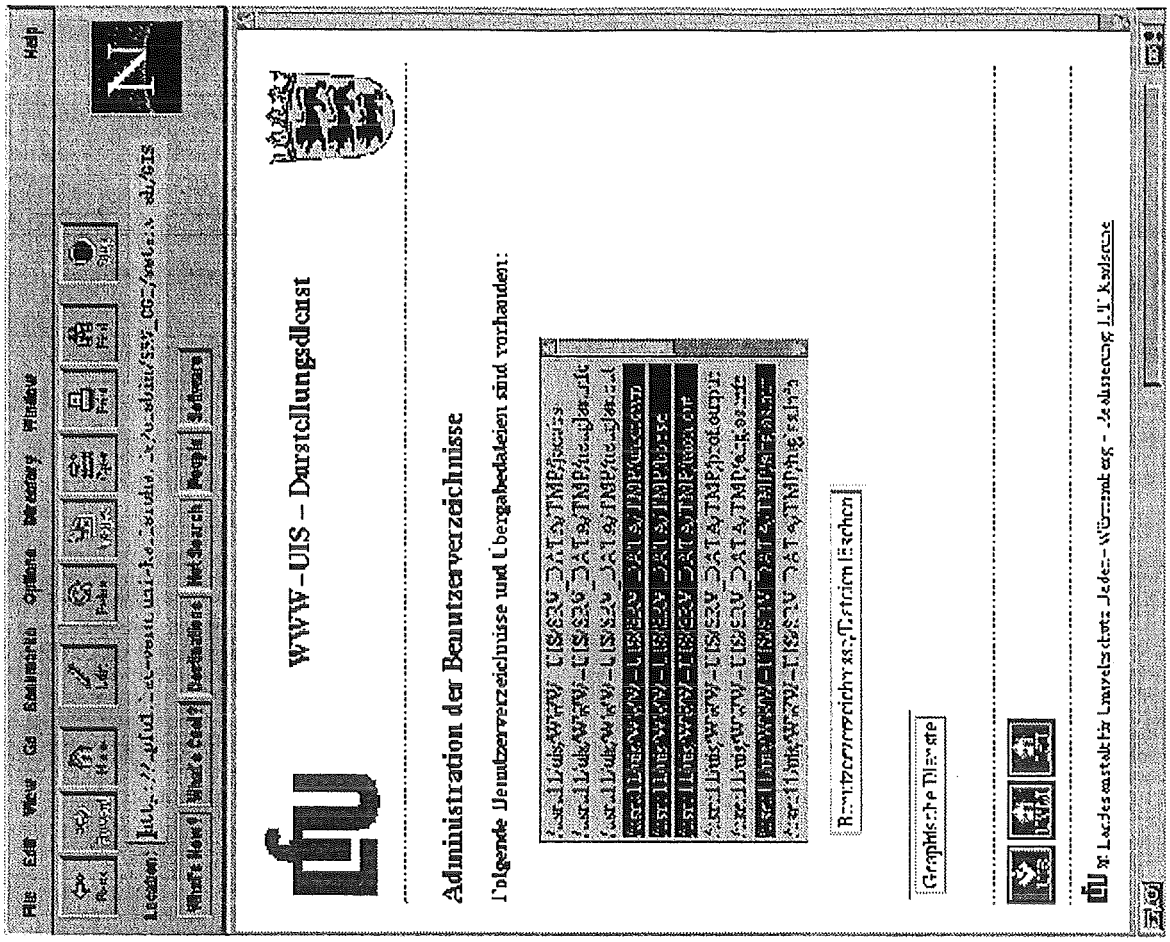


Abbildung 7: Die Seite Administration der Benutzerverzeichnisse

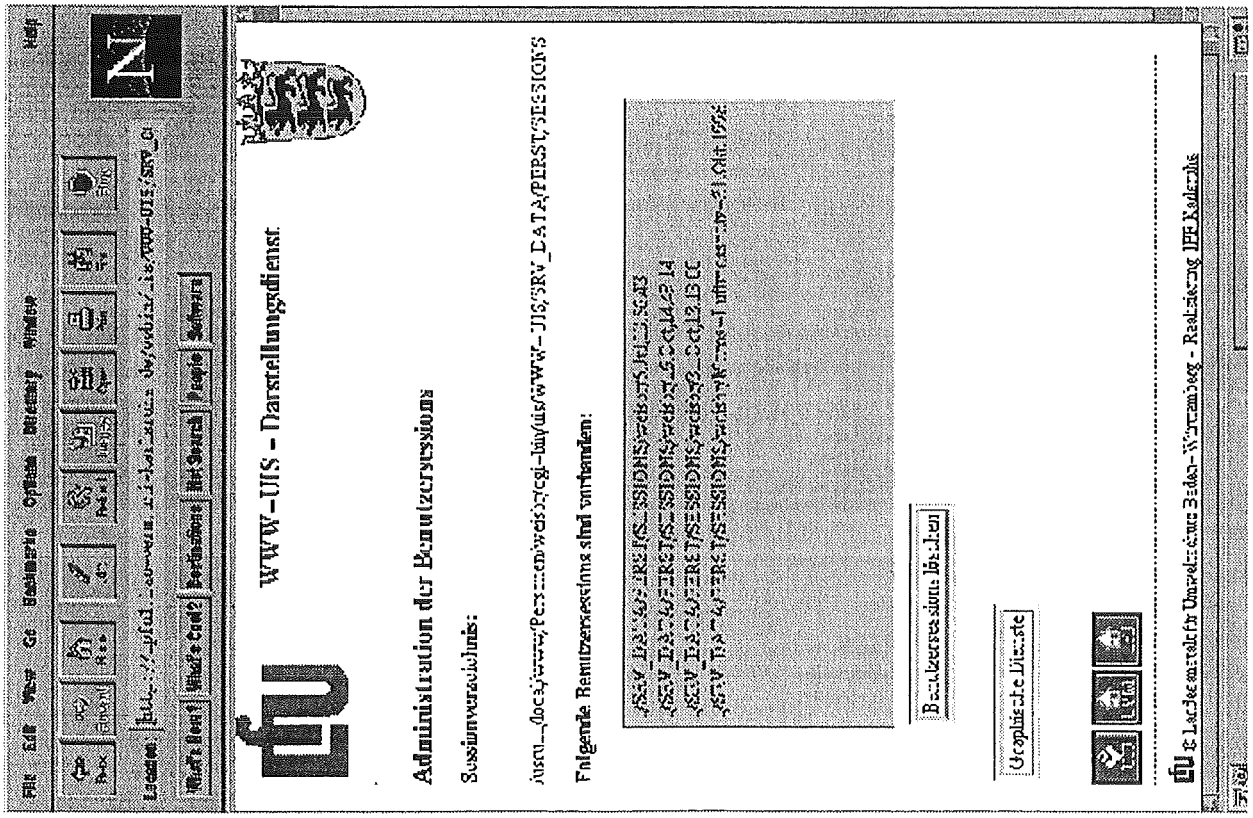


Abbildung 8: Die Seite Admninistration der Benutzersessions

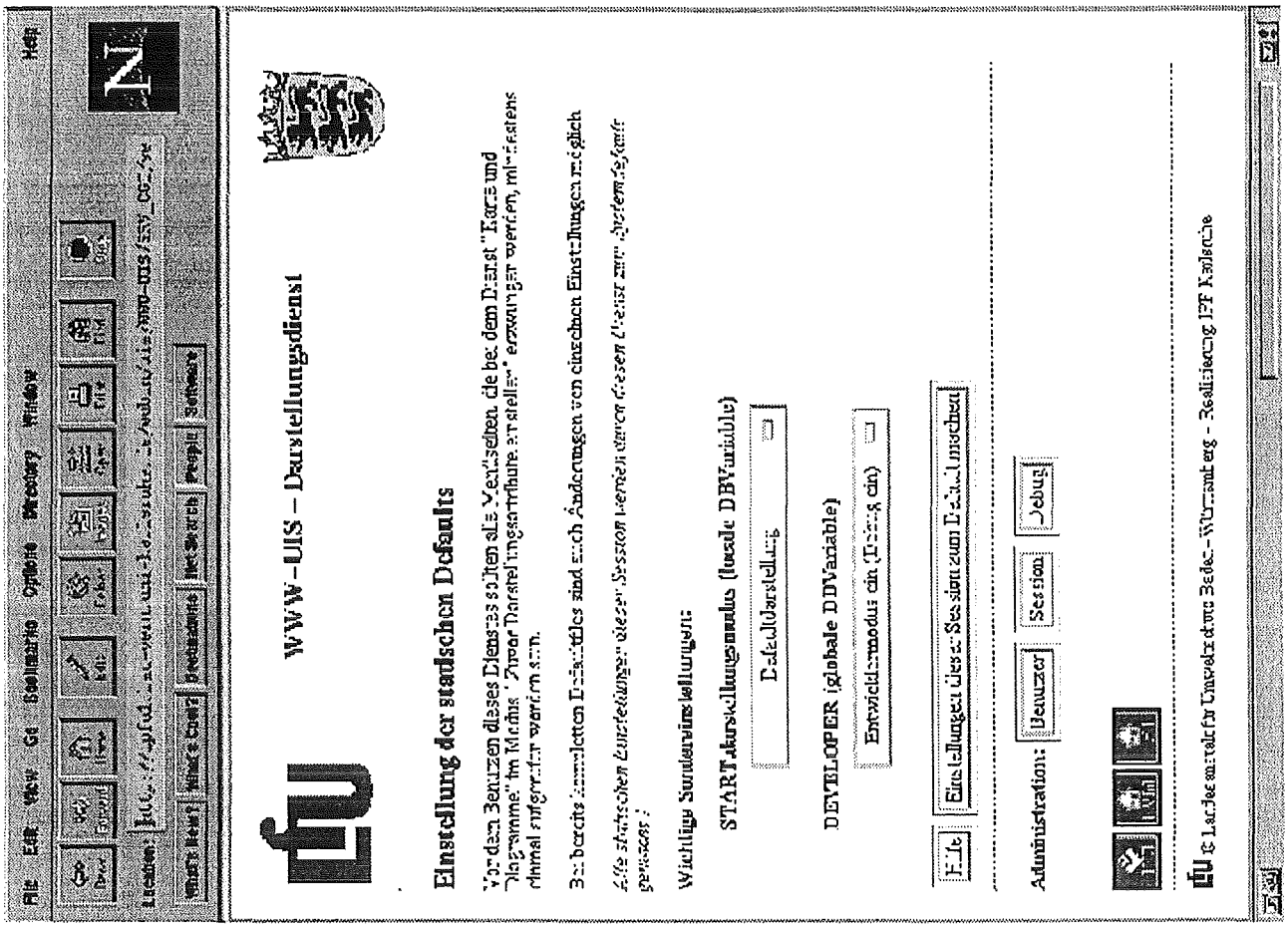


Abbildung 9: Die Seite Einstellung der statischen Defaults

3.2 ORACLE-Datenbank Anbindung

Die im WWW-UIS verwendeten GIS-Daten stammen aus dem Geoinformationssystem Smallworld, in dem unter der Bezeichnung RIPS (Räumliches Informations- und Planungssystem) ein großer Teil der in Baden-Württemberg vorhandenen Geoinformationen liegen. Bedingt durch eine fehlende Online-Schnittstelle zu RIPS wurden bisher innerhalb des Kartendienstes Zugriffe auf Sachdaten mittels einer Dateiorganisation vorgenommen. Diese Lösung ermöglicht zwar einen effizienten Datenzugriff durch den Einsatz mehrerer lokaler Server, ist aber wenig wünschenswert, da durch die redundante Datenhaltung die Konsistenz der Daten nicht gewährleistet ist und ein zusätzlicher Pflegeaufwand erbracht werden muß.

Die Landesanstalt für Umweltschutz hat in diesem Jahr das Projekt DB-ÜKO (Datenbank übergreifende Komponenten) realisiert. Für die verschiedenen (Umwelt-)Informationssysteme UFIS, TULIS, ALBIS/RIPS, UDK und GR-DB existieren verschiedene Datenmodelle, die nur teilweise aufeinander abgestimmt sind, was eine redundante Datenhaltung zur Folge hat. Innerhalb des Projekts wurden Überschneidungen lokalisiert und in einer ORACLE-Datenbank vereinheitlicht.

3.2.1 Tools für den Datenbankzugriff

Im Gegensatz zum Datenbanksystem von RIPS gibt es verschiedene Möglichkeiten, online auf das Datenbankmanagementsystem von ORACLE zuzugreifen. Damit war die Möglichkeit gegeben, die bisherige Lösung der Dateiorganisation durch einen direkten Datenbankzugriff zu ersetzen. Für die Realisierung kommen verschiedene Tools bzw. Programmiersprachen in Frage, die im Rahmen einer Diplom-Arbeit am IPF genauer untersucht wurden: Oratcl, WebQuery, Pro*C und Oracle Call Interface. In die engere Auswahl kamen Oratcl und WebQuery:

Oratcl ist eine Erweiterung der Programmiersprache Tcl für den Zugriff auf ORACLE-Datenbanken. Dabei wird auf den Mechanismus zurückgegriffen, daß der Tcl-Interpreter durch Einbinden von C/C++ Funktionsbibliotheken um neue Schlüsselwörter erweitert werden kann. Im Fall von Oratcl waren das Oracle's OCI („Oracle Call Interface") Libraries. Damit wird der Tcl-Befehlsumfang um einige sehr leistungsfähige Kommandos für den Datenbankzugriff erweitert, wie das folgende kurze Beispiel zeigt:

```
#!/usr/local/bin/oratcl
...
# Datenbankzugang oeffnen
set idpasswd $env(DBUEKO_USER)/$env(DBUEKO_PASSWD)
set handle [oralogon $idpasswd]
...
# Welche Kartentypen sind in der Datenbank?
Get_Kartentypen kart_list
...
# DB-Zugang schliessen
oralogoff $handle

# end

proc Get_Kartentypen { VNkart_list }{
      global oramsg handle
      upvar $Vnkartlist kartlist

      # Handle-String zur Datenbank oeffnen
      set cursor [oraopen $handle]

      # DB-Abfrage absetzen
      orasql $cursor "SELECT DISTINCT typ FROM lfu_kartenblatt"

      # Abfrage-Ergebnis in Variable kart_list ablegen
      set kart_list ""
      orafetch $cursor { lappend kart_list @0 }

      oraclose $cursor
      return
}
```

WebQuery ist eine Entwicklung des FAW, die mit dem primären Ziel entwickelt wurde Anfragen aus HTML-Seiten entgegenzunehmen, an eine Datenbank weiterzuleiten und die Abfrageergebnisse wieder als HTML-Seite aufzubereiten. WebQuery besteht im wesentlichen aus drei Modulen, die voneinander weitgehend unabhängig und deswegen vielfältig verwendbar sind. (vgl. Projekt Globus - Phase II, S.165-S.178)

Letztendlich erhielt Oratcl den Vorzug gegenüber WebQuery, da unter anderem

- die Implementierung der einzelnen Module für die GIS-Anbindung im WWW-UIS in

Tcl erfolgt ist und Tcl als CGI-Skriptsprache somit immer benötigt wird

- mit WebQuery durch das Parsen der Ergebnisdateien eine ungünstigere Performance erzielt wird als mit Oratcl
- mit Oratcl eine flexiblere Programmierung möglich ist
- die Programmpflege mit Oratcl einfacher ist
- es zweckmäßiger ist, ein vorhandenes Sprachinterface zu einer Datenbank zu benutzen, als jedesmal ein externes Programm aufzurufen, wenn ein Zugriff erforderlich ist.

Der Unterschied zwischen einer Verwendung von Oratcl und externen Programmen wie z.B. WebQuery für den Datenbankzugriff ist in Abbildung 10 noch einmal dargestellt. Eine Übersicht und Bewertung der untersuchten Tools gibt die Tabelle 1.

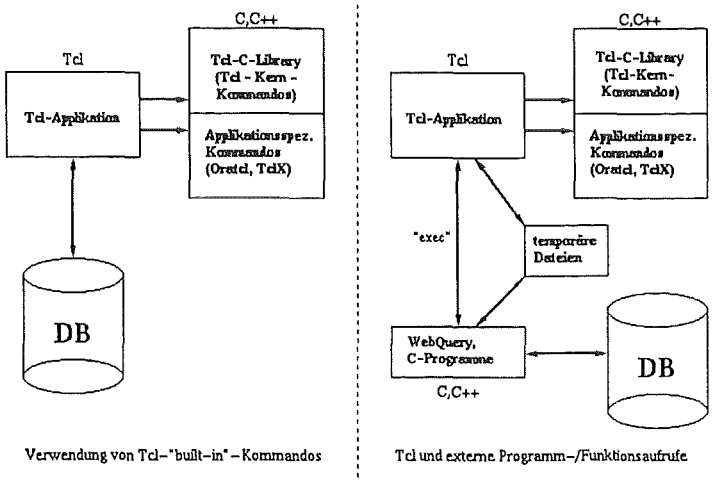


Abbildung 10: Unterschied Tcl-„built-in"-Kommando und Verwendung externer Programme

- negativ ←→ positiv ++	Oratcl	WebQuery	Pro*C	OCI	Bemerkung
Einarbeitung	++	+	o	-	
Entwicklungszeit	++	+	o	-	keine Compiler/Pre-Compilerläufe bei Oratcl
Homogenität	++	-	-	-	bestehende Realisierung ist in Tcl programmiert
Performanz	+	-	o	o	Schreiben, Lesen, Parsen von Dateien, etc.
Portierbarkeit	+	+	+	+	
Programmieraufwand	++	o	-	-	

Tabelle 1: Vergleich der Tools für den Datenbankzugriff hinsichtlich einer Verwendung in den graphischen Diensten

3.2.2 Anbindung an die graphischen Dienste

Bei der Realisierung der Datenbankanbindung an die graphischen Dienste wurde die direkt davon betroffene Regionenauswahl-Seite neu gestaltet. Das Layout und die Funktionalität der ersten Version war, teilweise natürlich auch bedingt durch die starre Struktur der Dateiorganisation, nicht besonders einfach zu verstehen. Die Konzeption der neuen Seite wird nun anhand der Abbildungen 11 - 13 erläutert.

1 - Ausgabe der Eingangsgrößen. Werden die Kartendienste mit Vorgabewerten aufgerufen, z.B. bei einem UDK-Aufruf, wird gegebenenfalls die Datenbank auf die übergebenen Raumeinheiten (Kartennummer, Kartenname, Gemeindenummer, Gemeinde, Landkreis usw.) hin durchsucht und bei Erfolg mit den entsprechenden Größen (ve_kennz, Name, Bounding-Box) hier ausgegeben. Bei falscher Übergabe der Vorgabewerte oder falls kein Eintrag in der Datenbank gefunden wird erfolgt eine Fehlermeldung, die bereits auf der Startseite der graphischen Dienste angezeigt wird. Erfolgreich übernommene Vorgabegrößen werden in die Darstellungsbox eingetragen -> 3.

2 - Auswahlbox 1. Mit den Buttons 6a und 6b kann durch die Hierarchie der Verwaltungseinheiten (Bundesland, Regierungsbezirk, Landkreis, Gemeinde) navigiert werden. Für jede angeklickte Einheit werden, je nach aktionsauslösendem Element, die Ober- bzw. Untereinheiten gesucht. Gibt es keine höheren oder tieferen Verwaltungsebenen mehr, wird eine entsprechende Meldung ausgegeben. Ist nichts ausgewählt, wird für alle Einheiten in der Box gesucht, sonst nur für die Markierten. Mit Button 4a können markierte Verwaltungseinheiten der Darstellungsbox (-> 3) hinzugefügt werden.

3 - Darstellungsbox. Einträge in dieser Box werden für die Darstellung der Karte berücksichtigt. Mit Button 4b lassen sich Einträge entfernen, die nicht berücksichtigt werden sollen.

4a - Aktionsauslösendes Element zur Aufnahme von Einheiten der Auswahlbox 1 (-> 2) in die Darstellungsbox (-> 3). Es werden nur markierte Einheiten eingetragen.

4b - Aktionsauslösendes Element zum Entfernen von Einheiten aus der Darstellungsbox (-> 3).

5 - Aktionsauslösendes Element zum Rücksetzen der Auswahlbox 1 (-> 2) auf eine vorgegebene Verwaltungsebene (Bundesland, Baden-Württemberg).

6a + 6b - Aktionsauslösende Elemente zur Navigation durch die Auswahlbox 1 (-> 2).

7 - Auswahl über Einheitenname. Durch Eingabe eines Namens in das Texteingabefeld kann direkt nach einer Verwaltungseinheit gesucht werden. Wildcards (% oder *) und Platzhalter (_)sind zulässig. Die Suchergebnisse werden in Auswahlbox 2 (-> 11) ausgegeben. Optional kann die Suche durch einen Menüeintrag eingeschränkt werden (z.B. nur Landkreise). Es gibt auch die Möglichkeit einer unscharfen Suche, wenn zum Beispiel die genau Bezeichnung einer Einheit nicht bekannt ist. Dann wird nach einer Einheit gesucht, die eine möglichst große Ähnlichkeit zu dem eingegeben Namen im Texteingabefeld aufweist. Dieses Mittel sollte aber nur gewählt werden, wenn eine Wildcard-Suche zu keinem Ergebnis geführt hat, da es unter Umständen sehr viel Zeit in Anspruch nimmt.

8 - Auswahl über Verwaltungseinheiten-Kennziffer (ve_kennz). Durch Eingabe einer ve_kennz kann direkt nach einer Verwaltungseinheit gesucht werden. Wildcards (% oder *) und

Platzhalter (_)sind zulässig. Die Suchergebnisse werden in Auswahlbox 2 (-> 11) ausgegeben.

9 - Auswahl über Karten. Die Darstellungsregion kann auch durch die Auswahl von Karten beeinflußt werden. In der Menüleiste wird aufgelistet, welche Kartentypen in der Datenbank vorliegen. Nach Wahl eines Kartentyps und Anklicken des aktionsaulösenden Elements werden in der Auswahlbox 2 (-> 11) Maßstab, Blattnummer und Blattname des gewünschten Kartentyps ausgegeben.

10 - Erst bei Auslösen des „weiter"-Button am Seitenfuß wird für alle Einheiten in der Darstellungsbox eine umschließende Bounding-Box berechnet, die den später dargestellten Kartenauschnitt enthält. Die Regionenauswahlseite wird verlassen und zur Kartenauswahl gesprungen.

11 - Auswahlbox 2. Sie wird auf einer neuen HTML-Seite ausgegeben. In ihr werden die Suchergebnisse von ->7, -> 8 und -> 9 aufgelistet. Die Aufnahme in die Darstellungsbox (-> 3) wird durch Markieren der gewünschten Einheiten und Auslösen des „weiter"-Buttons erreicht. Auf diese Weise kehrt man auch wieder zur Ausgangsseite der Regionenauswahl zurück.

Falls der Zugang zur Datenbank aus irgendwelchen Gründen nicht möglich ist, wird eine ersatzweise eine Seite mit eingeschränkter Funktionalität ausgegeben. Eingangsgrößen, die mit Koordinaten übergeben werden, wie z.B. Diagrammboxen, Objektboxen oder Punkte, werden in die Darstellungsbox aufgenommen, die übrigen Vorgabewerte können in diesem Fall dann aber selbstverständlich nicht berücksichtigt werden. Ansonsten gibt es noch die Möglichkeit zwischen den vier Regierungsbezirken und dem Land Baden-Württemberg zu wählen.

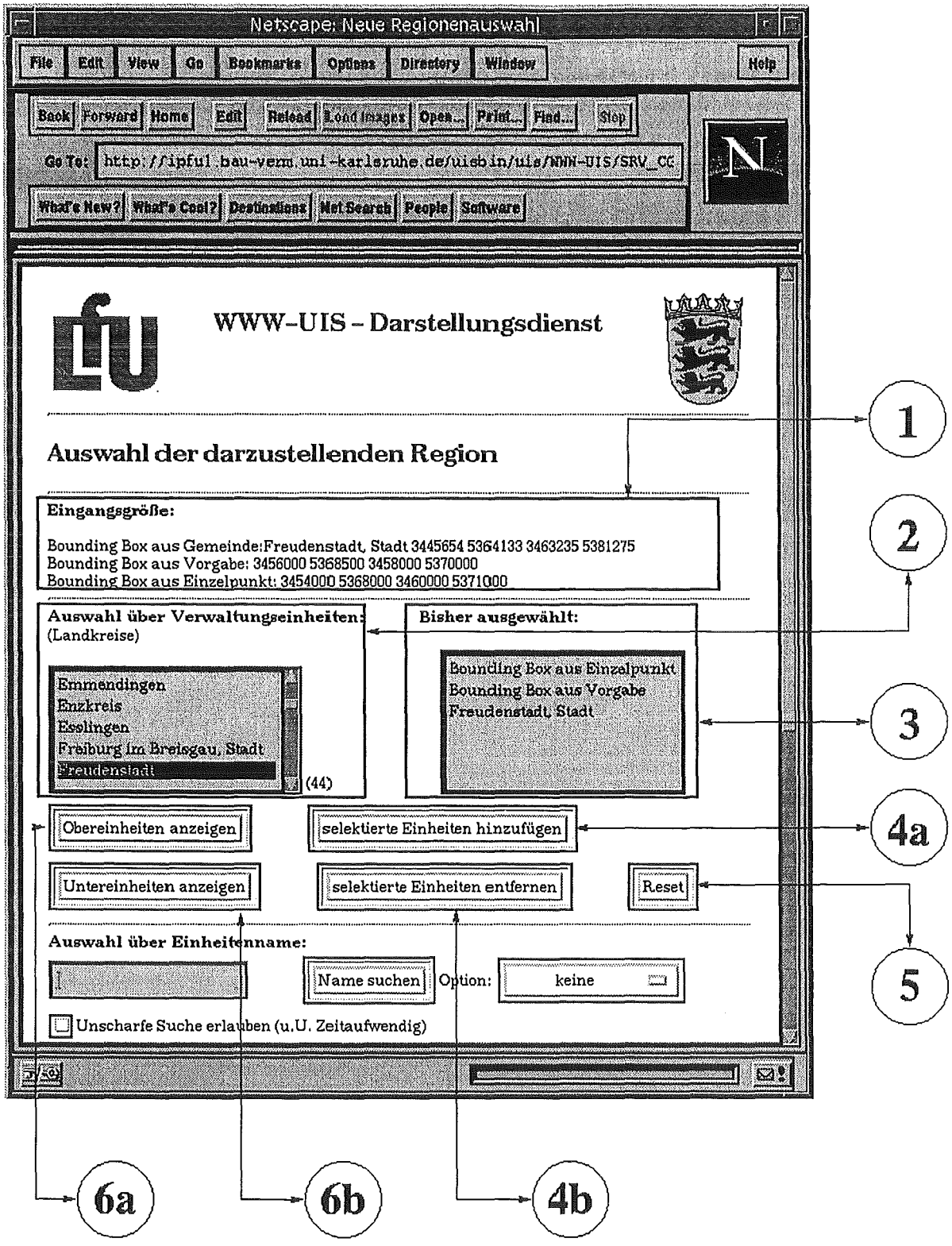


Abbildung 11: Regionenauswahl-Seite, Teil 1a

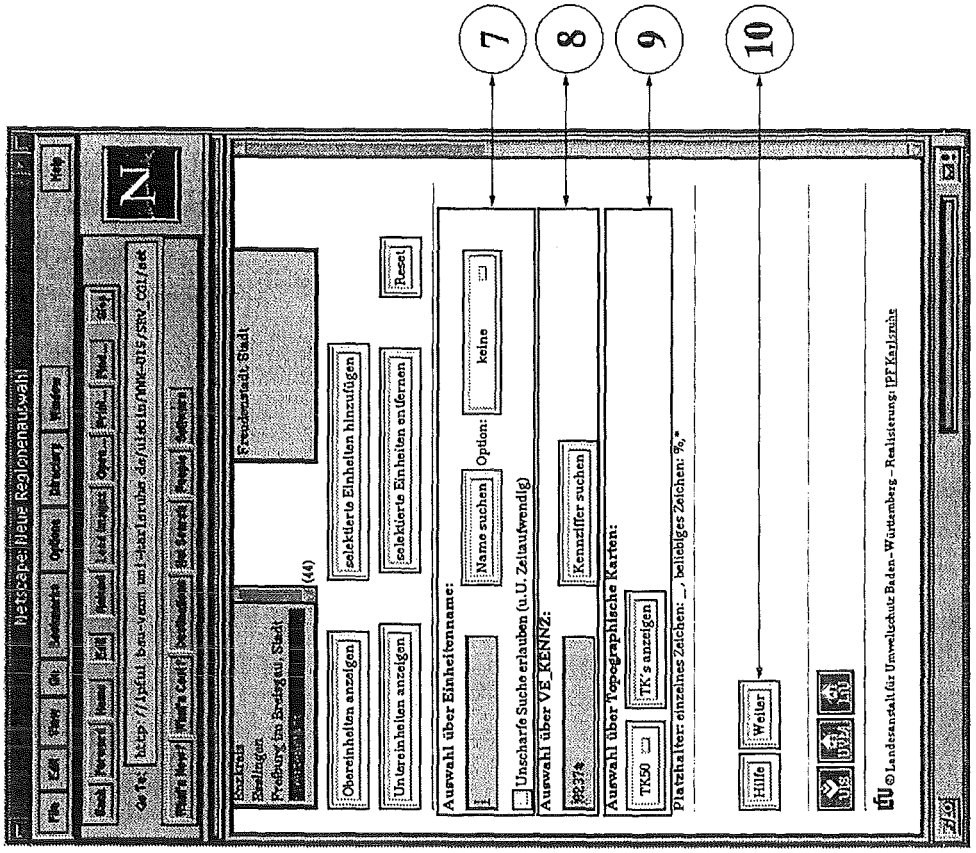


Abbildung 12: Regionenauswahl-Seite, Teil 1b

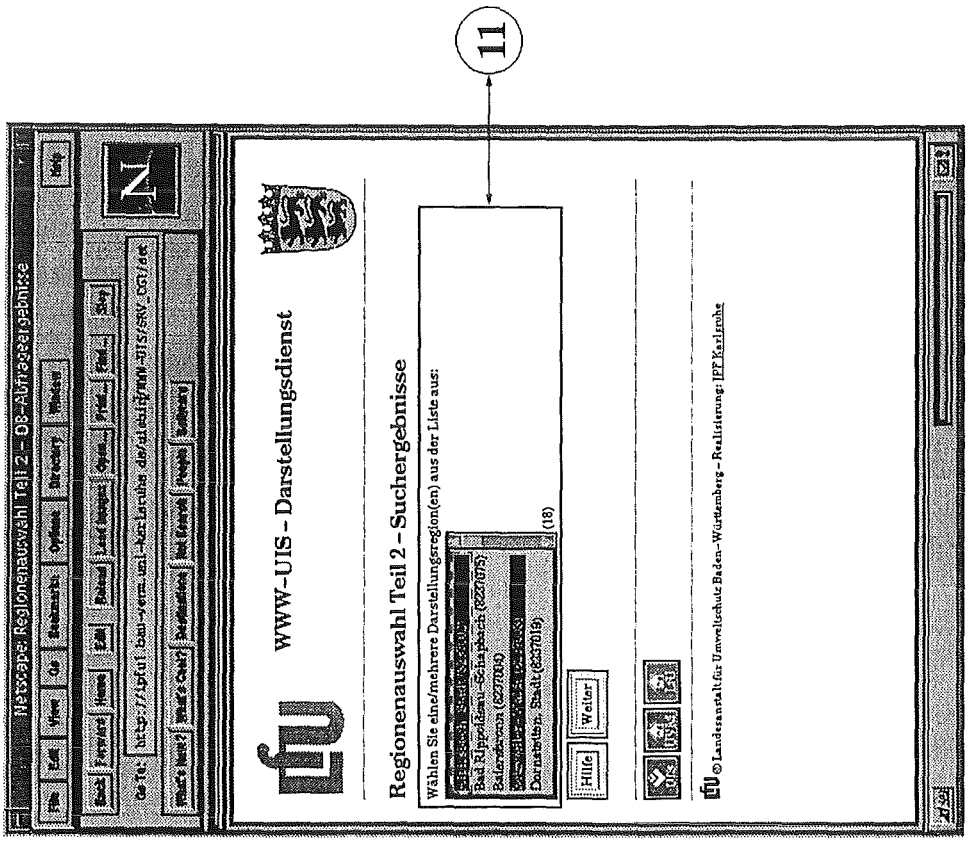


Abbildung 13: Regionenauswahl-Seite, Teil 2

Architektur eines GIS-Terminal zur Visualisierung von Geodaten

J. Wiesel, C. Hofmann,
Institut für Photogrammetrie und Fernerkundung (IPF),
Universität Karlsruhe,
Englerstr. 7,
76128 Karlsruhe

1. Einleitung

Sowohl die Verwaltung als auch die Manipulation und Verteilung von Geodaten erfolgt derzeit in einer Vielzahl von proprietären Datenformaten und Systemarchitekturen. Die Interoperabilität zwischen diesen Systemen ist nicht nur durch die gravierenden Unterschiede innerhalb der Datenformate stark eingeschränkt, sondern wird auch durch die damit einher gehende Differenzen in der Datenmodellierung erschwert. Diese Problematik ist auch verantwortlich für die Probleme, die bei der Beschaffung und Verteilung von Geoinformationen auftreten. Derzeit ist ein Informationsinteressent für die Abfrage, Aufbereitung und Visualisierung von Geodaten auf proprietäre Werkzeuge angewiesen. Der Einsatz solcher Werkzeuge garantiert zwar, daß alle Aspekte des entsprechenden Systems genutzt werden können, verursacht jedoch auch einen unverhältnismäßigen Verwaltungs- und Kostenaufwand. Im Folgenden sollen einige Probleme illustriert werden.

- Es müssen eine große Anzahl von Softwarelizenzen für die Werkzeuge eingekauft werden. Benötigt ein Informationsinteressent Zugriff auf mehrere Systeme, müssen Programme für jedes einzelne System für ihn beschafft werden.
- Der Informationsinteressent ist gezwungen den Umgang mit allen benötigten Werkzeugen zu erlernen. Dies wird durch unterschiedliche Designprinzipien sowie der zu großen Komplexität der Aufbereitungs- und Visualisierungswerkzeuge erschwert.
- Die Systeme stellen im Allgemeinen unterschiedlich hohe Ansprüche an die Hardware- und Betriebssystemkonfiguration. Gerade der Umgang mit Geodaten erfordert oft leistungsfähige und spezialisierte Rechner. Zudem kann beobachtet werden, daß die überwiegende Anzahl von Informationsinteressenten nur einen durchschnittlichen, für den Büroalltag ausgelegten, PC besitzen.
- Die breitflächige Installation von Software und deren lokale Administrierung verursacht einen großen Wartungs- und Kostenaufwand. Änderungen der Rechnerkonfigurationen oder Aktualisierungen von Softwarekomponenten vergrößern den Betriebsaufwand zusätzlich.

Durch innovative Basistechnologien wie die plattformunabhängige und netzwerkfähige Programmiersprache Java, zusammen mit Inter-, bzw. Intranet-Techniken wie CORBA können die genannten Probleme auf technischer Seite elegant angegangen werden. Die Bestrebungen durch Ansätze wie beispielsweise OGIS, eine offene, objektorientierte Geodaten-Spezifikationen zu entwickeln, helfen die Problematik von der Seite der Datenmodellierung zu lösen. Ein ähnlicher Weg wird schon längere Zeit im Produktdatenumfeld mit STEP/EXPRESS erfolgreich beschritten.

Die Architektur des im folgenden Bericht vorgestellten GIS-Terminal nutzt die neuen Basistechnologien um dem Informationsinteressenten eine Vielzahl von Informationsquellen flexibel und wartungsarm zur Verfügung zu stellen. Die neue Architektur beabsichtigt dabei nicht ein neues Informationssystem zu kreieren, sondern hat das Ziel vorhandene Systeme zu nutzen sowie integrativ, flexibel und transparent innerhalb eines Netzwerks zur Verfügung zu stellen.

Zunächst soll hier kurz auf die Programmiersprache Java, der eine Schlüsselrolle innerhalb des Konzepts zukommt, eingegangen werden. Im Anschluß daran wird die Gesamtarchitektur des GIS-Terminal vorgestellt. Abschließend erfolgt ein kurzer Bericht über die vorhandene,

prototypische Realisierung des GIS-Terminal.

2. Basistechnologie Java

Dieses Kapitel gibt nur einen kurzen Überblick über die Programmiersprache Java, soweit er für das konzeptionelle Verständnis der Architektur des GIS-Terminal nötig ist. Für eine ausführliche Analyse der Eigenschaften der Programmiersprache Java und deren Einsatzmöglichkeiten in den grafischen Diensten des WWW-UIS kann an dieser Stelle auf /5/ sowie auf den Bericht der Java-AG verwiesen werden.

Java ist eine objektorientierte Sprache und wurde von der Firma Sun entwickelt. Damit die Sprache eine schnelle Verbreitung findet, stellt Sun ein grundlegendes Entwicklungssystem kostenlos zur Verfügung. Ausgeführt werden können Java-Programme auf jedem Rechner auf dem das Java-Run-Time-System vorhanden ist. Das Run-Time-System besteht aus einem Interpreter, der den vom Compiler erzeugten maschinenneutralen Code in den realen Maschinencode des jeweiligen Rechners übersetzt und ausführt. Das Run-Time-System ist maschinenspezifisch und muß auf jedes Betriebssystem portiert werden. Von Sun werden dabei Microsoft Windows 95/NT, MacOS und das hauseigene Betriebssystem Solaris unterstützt. Eine Portierung auf alle gängigen UNIX-Derivate und OS/2 ist bereits erfolgt.

Während Java als gewöhnliche Programmiersprache wie z.B. C++ eingesetzt werden kann, ist es zusätzlich möglich, Java-Programme in das WWW zu integrieren. Zur Unterscheidung der beiden Einsatzgebiete, und auf Grund der unterschiedlichen Vorgehensweise, werden zwei unterschiedliche Bezeichnungen dafür verwendet. Bei Java-Programmen für das WWW spricht man von sogenannten Java-Applets, während normale Programme als Java-Applikationen bezeichnet werden. Java-Applets sind auf einen Java-kompatiblen WWW-Browser angewiesen. Derzeit ist der gebräuchlichste WWW-Browser, der Java unterstützt, der Netscape Navigator ab Version 2.0.

Ein Java-kompatibler Browser beinhaltet das Java-Run-Time-System, so daß die Installation des Browsers ausreichend für die Verwendung von Java im WWW ist. Ein spezieller HTML-Tag - der `APPLET`-Tag - verweist innerhalb eines HTML-Dokumentes auf ein Java-Applet, welches dann vom Browser geladen wird. Mit Ausnahme einiger Restriktionen kann ein solches Applet beliebige Funktionalitäten erfüllen, die andere Programme auch bieten. Der Unterschied besteht darin, daß die Programme nicht auf jedem Rechner installiert sein müssen, sondern bei Bedarf über das Netz geladen werden.

Durch die Integration von Java in die gängigsten Betriebssysteme entfällt der Einsatz eines Browsers /4/. Des weiteren sind von den Firmen Sun und Oracle sogenannte Netzcomputer angekündigt, die nur noch aus einem minimalen Betriebssystem mit Netzwerkanschluß und dem Java-Run-Time-System bestehen. Dadurch sollen sich die nicht zu unterschätzenden Wartungskosten der PC Systeme innerhalb eines Unternehmens deutlich reduzieren lassen.

Eine weitere innovative Eigenschaft von Java ist die Definition eines grafischen Fenstersystems innerhalb der standardisierten Java-Klassenbibliothek. Erst dieses Fenstersystem, das Abstract Window Toolkit (AWT) ermöglicht es, Anwendungen mit grafischen Benutzeroberflächen und grafischer Ausgabe unabhängig von einem tatsächlich

vorhandenen Fenstersystem (X11-Motif, bzw. Windows) zu implementieren und auszuführen. Das Abstract Window Toolkit bildet hierzu alle grafischen Ausgabekomponenten über sogenannte Peer-Objekte auf Komponenten des realen Fenstersystems ab. Ebenso werden alle Eingabeoperationen (Benutzerinteraktionen) über diese Schnittstelle dem Java-Programmierer in einer vom realen System abstrahierten Form zu Verfügung gestellt. Dieser Vorgang ist für den Programmierer transparent.

3. Gesamtarchitektur GIS-Terminal

In diesem Kapitel wird zunächst die Gesamtarchitektur vorgestellt. Im Anschluß daran werden die einzelnen Schichten des Entwurfs nacheinander in detaillierter Form erläutert.

Das Konzept des GIS-Termial steht in der Tradition der Client-Server Architekturen, auch wenn der Begriff Terminal mit dem zentralistischen „Terminal-Host-Schemata" assoziiert werden kann. Die Architektur verfolgt das Ziel, alle interaktiven und visuellen Teile eines Geografischen Informationssystem (GIS) zu vereinheitlichen und auf einen intelligenten Klienten zu verlagern. Zudem soll serverseitig eine Abstraktion und Integration unterschiedlicher Systeme verwirklicht werden. Dazu muß eine systemabstrahierende Schnittstelle definiert werden, die den vereinheitlichten Zugriff auf vorhandene Informationssysteme ermöglicht. Diese Schnittstelle muß gegenüber den darunter liegenden Systemen als Dienstnehmer auftreten und Differenzen zwischen den Systemarchitekturen, sofern diese für die Visualisierung und Aufbereitung der Daten nötig ist, ausgleichen. Ziel ist es dabei, möglichst viel Ressourcen der vorhandenen Systeme zu nutzen. Soll ein neues System zu der Gesamtarchitektur hinzugefügt werden, muß für das System ein Adapter, in Form eines Treibers implementiert werden. Dieser Treiber hat die Aufgabe, die Funktionalitäten, die in der abstrakten Schnittstelle definiert sind, auf das reale System abzubilden. An dieser Stelle soll nicht verschwiegen werden, daß diese Schnittstelle zunächst allgemeine und gebräuchliche Eigenschaften vereinheitlichen will. Spezielle Fähigkeiten eines bestimmten Systems können dabei im ersten Ansatz nicht abstrahiert und den oberen Schichten als Dienst zur Verfügung gestellt werden. Dies ist jedoch im allgemeinen Fall nicht als Nachteil zu werten, da die Zielgruppe des Systems im Allgemeinen kein Interesse an spezialisierten Funktionen besitzt. Der konzeptionelle Schwerpunkt liegt auf der breiten Verteilung von Geodaten. Der Benutzer soll in einer einheitlichen, transparenten Form das virtuelle GIS benutzen können, ohne Kenntnis über die darunterliegende Systeme und deren Struktur zu besitzen. Weiterhin sollen diese Dienste allen Informationsinteressenten flexibel und dynamisch über das Netzwerk zur Verfügung stehen.

Abbildung 1 verdeutlicht das Konzept und gliedert den Entwurf. Die Architektur ist Modular aufgebaut und besteht aus zwei Softwarekomponenten, die auf dem Client-Server Prinzip beruhen. Der Server (GIStermServer) besitzt gegenüber dem Klient (GISterm) die Serverrolle, gleichzeitig verhält er sich jedoch gegenüber den darunterliegenden Informationssystemen als Klient. Man kann in der Abbildung erkennen, daß das Konzept zwei Rechnergrenzen berücksichtigt. Alle beteiligten Systeme können räumlich verteilt, auf verschiedenen Rechner und Rechnerplattformen beheimatet sein. Der eigentliche Klient (GISterm) kann über das Inter-, bzw. Intranet an alle Rechner im Netzwerk dynamisch verteilt werden. Dazu muß der Klientrechner lediglich einen Netzanschluß sowie einen Java-fähigen WWW-Browser (z.B.

Netscape) besitzen.

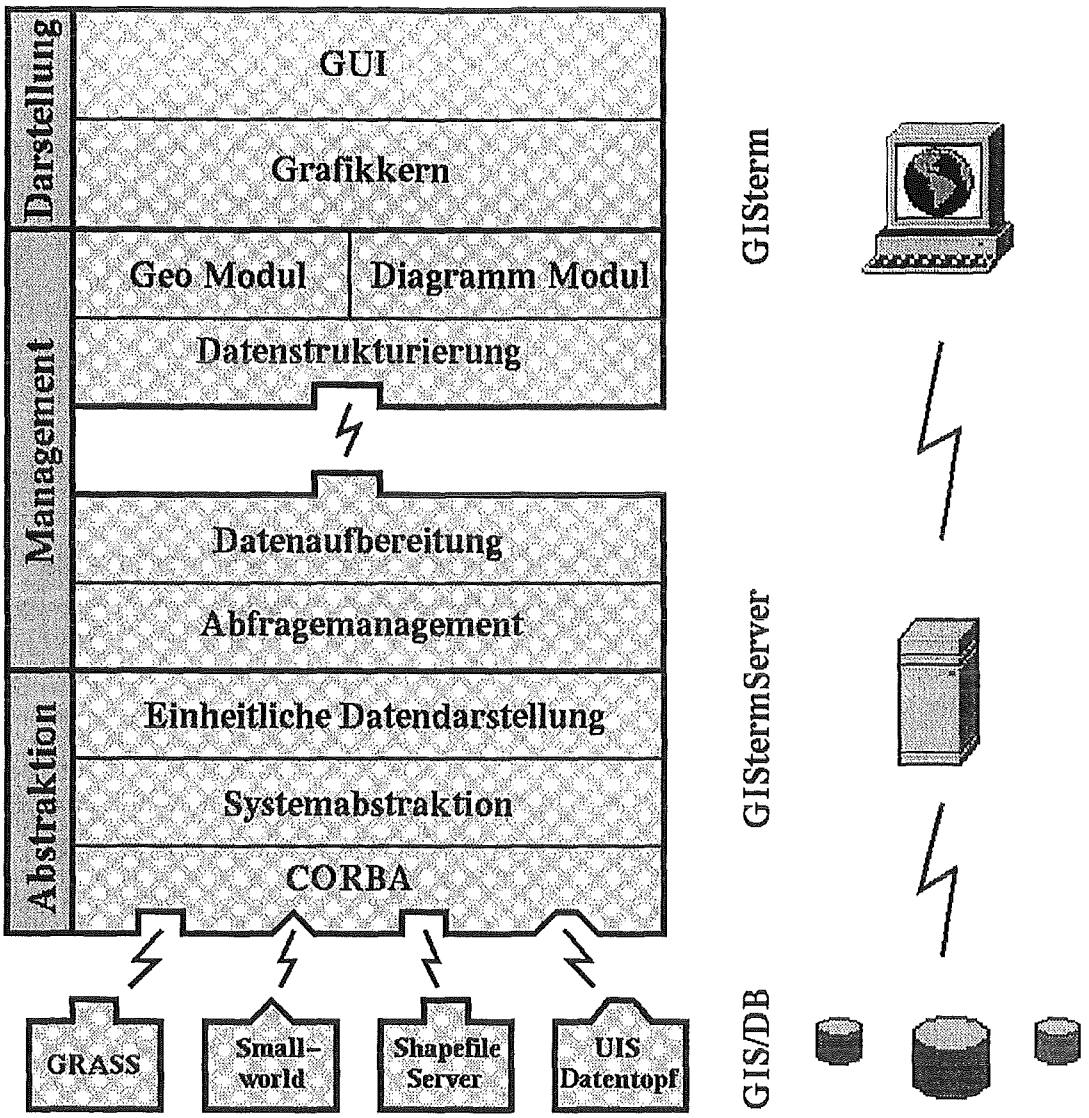


Abbildung 1: Gesamtarchitektur

Der Server (GIStermServer) wird auf dem gleichen Rechnersystem installiert, auf dem der für das Netz zuständige WWW-Server residiert. Dieser Server kann jetzt wiederum auf beliebige im Netz verteilte Informationssysteme zugreifen. Diese Flexibilität wird durch die Basistechnologienen Java und CORBA ermöglicht. Insgesamt wird durch dieses Konzept erreicht, daß die Information, die meist räumlich und organisatorisch verteilt ist, orthogonal und flexibel im Netz zur Verfügung gestellt werden kann.

Das Gesamtsystem des GIS-Terminal läßt sich in drei Hauptschichten zerlegen.

1. Darstellungsschicht
2. Managementschicht
3. Abstraktionsschicht

Dabei besteht die Serverkomponente (GIStermServer) aus der Abstraktionsschicht, sowie aus einem Teil der darauf aufbauenden Managementschicht. Der Klient (GISterm) setzt sich aus

dem zweiten Teil der Managementschicht sowie der Darstellungsschicht zusammen. Die Managementschicht ist aus Realisierungsgründen aufgeteilt. Um unnötige Netzbelastungen zu vermeiden, sowie die Performanz zu steigern, werden Daten in dieser Schicht auf dem Server für die Darstellung aufbereitet und strukturiert. Ziel ist es den Datenstrom so zu modellieren, daß die Netzwerkbelastung minimiert wird. Gleichzeitig soll der Klient die übertragenen Daten eigenverantwortlich halten und manipulieren können. Beispielsweise soll die Aufbereitung der Daten zu einer Präsentation fast ausschließlich auf dem Klienten ablaufen, was zu einer deutlichen Reduzierung der Netzbelastung und Serverauslastung führt.

In den folgenden Abschnitten sollen die einzelnen Schichten nacheinander näher vorgestellt werden. Dabei wird mit der untersten Schicht begonnen.

3.1 Die Abstraktionsschicht

Die unterste Schicht hat die Aufgabe, den darauf aufbauenden Schichten, den Dienst eines einheitliches, virtuellen Informationssystem anzubieten. Als Basistechnologie dieser Schicht ist an dieser Stelle vor allem CORBA zu nennen. Da diese Technologie grundlegend in den wissenschaftlichen Berichten des GLOBUS Projekts zur Phase II /2/ erläutert wird, wird hier auf eine Einführung verzichtet. Zudem geht der Bericht „Geooperatoren unter CORBA“ näher auf diese Thematik ein. An dieser Stelle sollen nur die prinzipiellen Zusammenhänge verdeutlicht werden.

Die Abstraktionsschicht kann in drei Subschichten untergliedert werden, welche folgende Aufgaben besitzen.

1. Einheitliche Datendarstellung
2. Systemabstraktion
3. GIS-Operatoren unter CORBA

Die dritte und unterste Schicht realisiert den Zugriff auf im Netz verteilte Informationssysteme. Dazu werden GIS-Operatoren mittels CORBA implementiert, welche auf die realen Geoinformationssysteme zugreifen und Daten aus den Systemen selektieren und übermitteln. Weiterhin soll an dieser Stelle der Zugriff auf Datenfiles Mithilfe eines Fileserver ermöglicht werden. Dadurch ist es möglich einzelne Datendateien, die in der Praxis vor allem für den Datenaustausch benutzt werden, in das System zu integrieren. Im einzelnen soll der Zugriff auf

- Arc Shapefile-Dateien
- DXF-Dateien

realisiert werden. Weiterhin könnten alle Datenquellen die innerhalb des WWW-UIS realisiert wurden an dieser Stelle in das Gesamtkonzept aufgenommen werden. Im Rahmen der Java-AG wurden erste Planungen zu einem allgemeinen Datentopf bereits diskutiert.

Die auf der CORBA-Schicht aufbauende Systemabstraktionsschicht bietet die Operatoren der untersten Schicht in einer definierten Schnittstelle an. In dieser Schicht wird auch eine Zugriffsverteilung und Systemselektion realisiert. Ein weiteres Modul ist für die Migration der Geometrieformate der einzelnen Systeme in ein einfaches, topologieloses Format zuständig. Dieses generalisierte Geometrieformat soll im Hinblick auf die Aufbereitung und Darstellung

der Geodaten definiert werden. Topologische und semantische Aspekte werden zunächst nicht unterstützt. Sobald sich eine einheitliche Geodatenbeschreibung etabliert hat, soll diese sich an dieser Stelle integrieren lassen. Neben dem Geometrieformat muß diese Schicht ein vereinheitlichtes Format zur Darstellung von Tabellen (Relationen) definieren.

3.2 Die Managementschicht

Innerhalb der Managementschicht, sollen die von der Abstraktionsschicht zur Verfügung gestellten Daten für die grafische Darstellung aufbereitet und strukturiert werden. Wie bereits im Rahmen der Gesamtarchitektur angesprochen, ist diese Schicht auf den Klient (GISterm) als auch auf den Server (GIStermServer) aufgeteilt. Diese verteilte Architektur begründet sich vor allem durch das Bestreben die Netzwerkbelastung zu reduzieren. Die Daten sollen auf dem Server für die Darstellung vorbereitet und in dieser optimierten Form in den Klienten übertragen werden. Der Klient (GISterm) kann dann ohne weitere Netzzugriffe die Daten für die Darstellung verwenden. Der Benutzer hat dadurch die Möglichkeit die Daten auf seinem Rechner interaktiv aufzubereiten und in eine präsentierbare Form zu bringen.

Für die Realisierung der verteilten Architektur innerhalb der Schicht bieten sich drei Möglichkeiten an:

1. **Das HTTP-Protokoll**. Mit diesem Verfahren können Daten über den WWW-Server an den Klient übertragen werden. Die zu übertragenden Daten müssen dazu in einen Datenstrom überführt werden, den der Klient wieder aufschlüsseln und verteilen muß. Dieses Bündeln und anschließende Aufteilen verursacht Mehraufwand und behindert eine klare Programmarchitektur. Weiterhin steigt die Belastung des WWW-Servers deutlich an.
2. **Die Socketverbindung**. Die Netzwerkfähigkeiten der Programmiersprache Java ermöglichen es unabhängig von dem HTTP-Protokoll, Verbindungen zu anderen Rechnern im Netz herzustellen. Die Folge dabei ist, daß auch eigene Protokolle definiert werden müssen. Weiterhin müssen, wie bei der Übertragung mit dem HTTP-Protokoll, die Daten ebenfalls in einen Datenstrom überführt werden.
3. **Verteilte Objekte**. Diese Technik erlaubt die Aufteilung von Objekten auf mehrere Rechner und übernimmt deren Kommunikation. Für den Programmierer bleibt diese Kommunikation verborgen, er muß lediglich eine Zuordnung von Objekt und Rechner erfolgen. Für Java gibt es derzeit mit HORB /6/ und RMI (Remote Method Invocation) /3/ zwei konkrete Ansätze. In diesem Entwurf wird diese Technik favorisiert, da durch verteilte Objekte eine klare und flexible Programmstruktur ermöglicht wird. Ein weiterer Vorteil dieser Lösung ist die dynamischen Verteilung der Objekte auf Server bzw. Klient.

Die Module der Managementschicht sind wie folgt unterteilt:

1. Das **Geo-Modul** sowie das **Diagramm-Modul**. Diese Module haben die Aufgabe, eine Datenvisualisierung interaktiv, oder Mithilfe von Vorgabewerten, zu erstellen.
2. **Datenstrukturierung**. Dieses Modul strukturiert und verwaltet die zu visualisierenden Daten. Hier wird unter Zuhilfenahme der Geo- und Diagramm-Module jeder Gruppe von Darstellungsobjekten ein oder mehrere Visualisierungsobjekte zugeordnet. Auf die Strukturierung der Daten wird nachfolgend detaillierter eingegangen.
3. **Datenaufbereitung**. Dieses Modul hat die Aufgabe neue Daten in die Struktur im

Datenmanagement-Modul einzufügen. Dabei müssen vor allem die geometrischen Daten in Darstellungsobjekte (Objekte, die von dem Grafikkern verwaltet werden können) überführt werden.

4. **Abfragemanagement**. An dieser Stelle wird die Benutzerabfrage, die der Anwender über mehrere Eingabemasken interaktiv spezifiziert hat, in eine formale Abfrage umgewandelt und an das darunterliegende virtuelle Informationssystem übergeben. Wenn das System die gewünschten Daten bereit gestellt hat, werden diese an die übergeordnete Datenaufbereitung weitergeleitet.

Die Struktur der Datenobjekte dieser Schicht wird in dem vereinfachten statischen Modell in Abbildung 2 gezeigt. Das Modell verwendet eine an Rumbaugh /1/ angelehnte Notation.

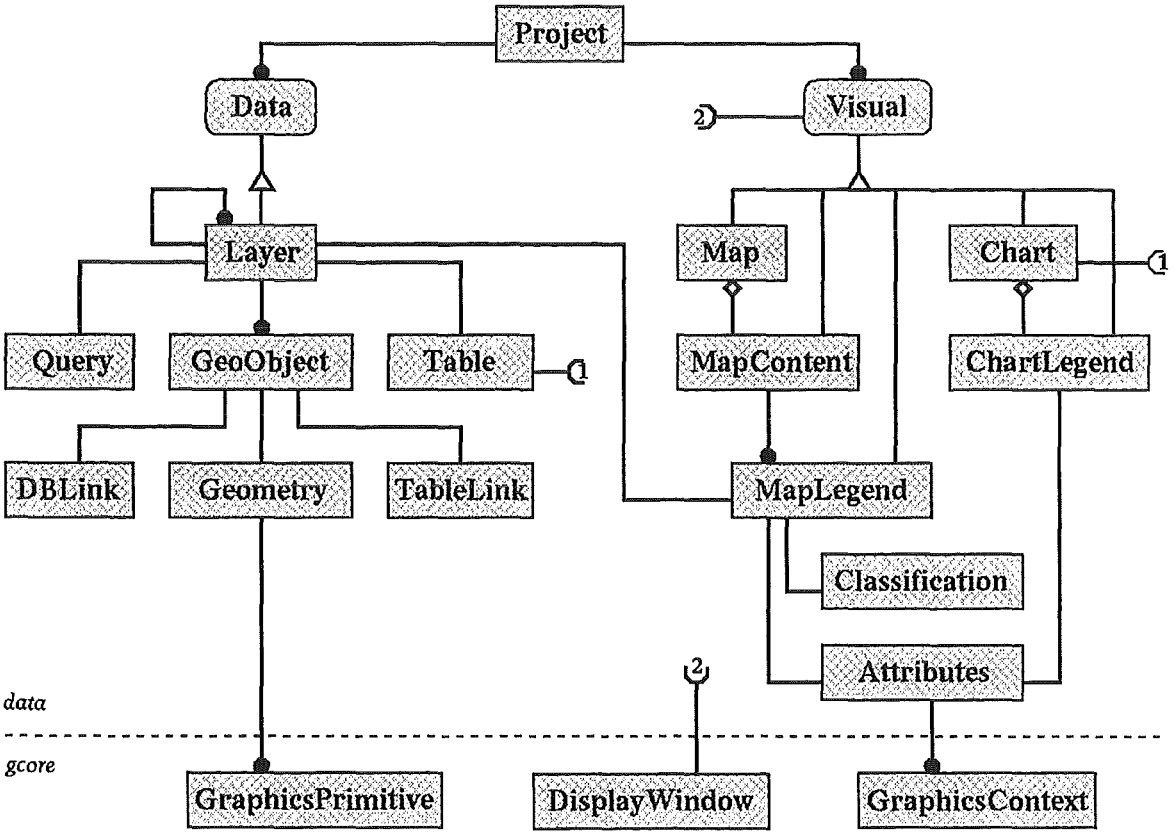


Abbildung 2: Vereinfachtes statisches Model der Managementschicht

Ein Benutzer kann seine Arbeit innerhalb von Projekten (*Project*) verwalten. Ein Projekt gruppiert beliebig viele Daten- und Anzeigeobjekte (*Data*, *Visual*). Datenobjekte sind Geodaten sowie Sachdaten. Die Anzeigeobjekte (*Visual*) visualisieren eine bestimmte Sicht (Interpretation) auf die Datenobjekte. Geodaten werden in *Layer* gruppiert. Dabei stellt ein *Layer* zunächst nur einen Container für semantisch, im Bezug auf eine Datenabfrage *Query*, zusammengehörige Geoobjekte *GeoObject* dar. Es ist möglich, daß ein *Layer* weitere, ihm untergeordnete *Layer* besitzen kann. Wenn sich z.B. eine Datenanfrage auf Flüsse in einem bestimmten Gebiet bezieht, können diese Flüsse durch untergeordnete *Layer* thematisch geordnet werden, etwa in Haupt- und Nebenflüsse. Jeder *Layer* besitzt ein *Table*-Objekt, in dem Sachdaten, die ev. aus einer Abfrage hervorgehen, verwaltet werden. Die im *Layer* gruppierten Geoobjekte *GeoObject* bestehen aus drei weiteren Objekten. Der *DBLink* merkt sich einen Bezug auf das zugehörige Objekt im zugrunde liegenden Informationssystem,

sofern dies möglich ist (setzt im Allgemeinen ein objektorientieres DB-System voraus). Der *TableLink* referenziert einen Datensatz in dem zum *Layer* gehörenden *Table*-Objekt, der sich auf das Geoobjekt bezieht. In dem *Geometry*-Objekt wird die Objektgeometrie in Form von darstellbaren Grafikkern-Primitiven (*GraphicsPrimitive*) gehalten. An dieser Stelle erfolgt ein Übergang zur Darstellungsschicht. Alle Anzeigeobjekte (*Visual*) besitzen ein eigenes *DisplayWindow* in dem sie dargestellt werden. Das *DisplayWindow* wird von der Darstellungsschicht zur Verfügung gestellt und kann einen Ausschnitt aus einer Grafikwelt innerhalb der Benutzeroberfläche darstellen. Anzeigeobjekte lassen sich zunächst in Kartenobjekte (*Map*) sowie Diagrammobjekte (*Chart*) aufteilen. Die Objekte der Klassen *MapContent*, *MapLegend* und *ChartLegend* sind ebenfalls *Visual*, jedoch den Klassen *Map* bzw. *Chart* untergeordnet. Ein *Map*-Objekt besteht neben dem *DisplayWindow* aus einem Inhaltsverzeichnis (*MapContent).* Für jeden *Layer*, der in der Karte sichtbar ist, kennt *MapContent* eine Legende (*MapLegend*). Jede Legende referenziert den *Layer* den sie visualisiert sowie ein *Cassification*-Objekt, in dem die vom Benutzer gewünschte Interpretation des *Layer* festgelegt wird. Diese Interpretation drückt sich für die Visualisierung in einer eindeutigen Attributierung (*Attributes*) aus. Jedes Diagrammobjekt (*Chart)* referenziert ein Tabellenobjekt (Table) in dem die darzustellenden Werte ablegt sind. Weiterhin besitzt jedes Diagramm eine Legende (*ChartLegend*) mit einer zugehörigen Attributierung (*Attributes*).

3.3 Die Darstellungsschicht

Diese Schicht ist für die Grafikausgabe sowie die Benutzerinteraktion zuständig und gliedert sich in zwei Module.

1. GUI
2. Grafikkern

Beide basieren auf dem Abstract Window Toolkit (AWT), ein abstraktes, grafisches Fenstersystem das innerhalb der standardisierten Java-Klassenbibliothek definiert ist. Eine detaillierte Beschreibung der GUI-Komponenten würde den Rahmen dieses Berichtes sprengen. Für das GIS-Terminal werden eine Vielzahl von Dialogkomponenten benötigt, einige davon bietet das Toolkit direkt an, andere müssen durch Spezialisierung und Gruppierung von vorhandenen Komponenten erstellt werden. Das abstrakte Fenstersystem AWT bietet derzeit, aufgrund der Komplexität und der unterschiedlichen Architektur der abstrahierten Fenstersysteme, nur eine Untermenge der Möglichkeiten dieser darunterliegenden realen Systeme an. Der Leistungsumfang ist für die Implementierung einer interaktiven Benutzeroberfläche mehr als ausreichend. Für die Ausgabe von Grafik definiert die abstrakte Klasse Graphics die Grafikprimitive Linie, Rechteck, Kreis, Oval, Polygon, Bilder und Text zusammen mit einer einfachen Attributierung. Eine Implementierung dieser Grafikklasse wird für die GUI-Komponente Canvas (eine gedächtnis- und aktivitätslose Zeichenfläche) angeboten. Für das Neuzeichen von Bildteilen, sowie die Skalierung der Grafik ist das Anwenderprogramm selbst verantwortlich. Eine Möglichkeit der vom Ausgabegerät unabhängigen Grafikausgabe, etwa eine PostScript- oder HPGL-Ausgabedatei existiert standardmäßig bislang nicht.

Für das GIS-Terminal ist es unerläßlich, aufbauend auf die von Java gebotene Grundfunktionalität einen grafischen Kern zu implementieren. Erst durch einen solchen

allgemeinen Grafikkern ist es möglich, komplexere Grafiken darzustellen, zu verwalten und zu modifizieren. Der hierzu zu entwickelnde Kern soll, angelehnt an grafische Standards wie GKS (Graphics Kernel System, ISO1985), Funktionalitäten wie

1. abstrakte Ausgabegeräte (PostScript, CGM, HPGL, ...),
2. abstrakte Eingabegeräte (Pick, Locator, ...),
3. strukturierte, hierarchische Grafikprimitive,
4. Weltkoordinatensysteme

bieten. Da die Firmen Sun und Adobe im Rahmen des Bravo-Projektes derzeit an einer starken Erweiterung der Grafikfähigkeiten von Java arbeiten, setzt der Entwurf seinen Schwerpunkt vor allem in die Realisierung von höheren, abstrakten Funktionalitäten und umgeht derzeit bewußt eine Erweiterung der bislang schwach ausgeprägten Attributierung. Sobald die Grafikerweiterungen vorliegen, können diese in die unteren Schichten des Kerns integriert werden.

Abbildung 3 gibt in einem vereinfachten statischen Modell einen Überblick über die Struktur des Grafikkerns (Notation in Anlehnung an Rumbaugh /1/).

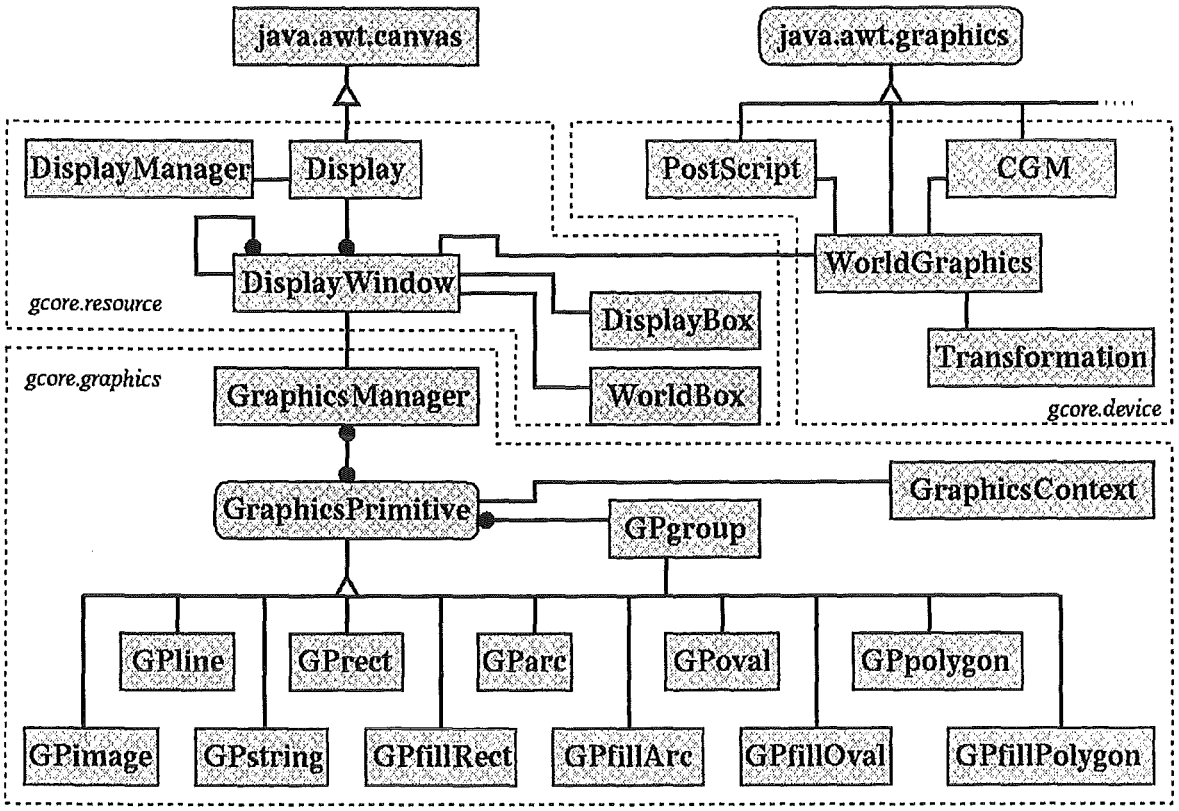


Abbildung 3: Vereinfachtes statisches Model des Grafikkerns

Demnach besteht der Kern ist aus drei Gruppen.

1. **Resource.** Dieser Teil ist für die Ausgabe innerhalb der GUI zuständig. Das *Display* stellt als Spezialisierung der Klasse *Canvas* einen rechteckigen Grafikbereich dar. Da innerhalb des Grafikbereichs Benutzeraktionen ausgelöst werden können, koordiniert die Klasse *DisplayManager* diese Aktionen in Analogie zu einem Window-Manager in

einem Fenstersystem. In einem *Display* können beliebig viele hierarchisch geordnete Ausgabeeinheiten (*DisplayWindow*) angezeigt werden. Ein *Display* besitzt immer ein, den ganzen Bereich ausfüllendes, Root-*DisplayWindow*. Innerhalb des Root-*DisplayWindow* können weitere *DisplayWindow* hierarchisch angeordnet werden. In einem *DisplayWindow* wird ein rechteckiger Ausschnitt aus einer Grafikszene (*GraphicsManager*) angezeigt. Dazu wird der sichtbare Ausschnitt der Grafikszene mit der *WorldBox* in Weltkoordinaten angegeben. Die *DisplayBox* positioniert das *DisplayWindow* relativ zu seinem Vaterfenster. Jedes *DisplayWindow* tätigt seine Ausgaben immer über die *WorldGraphics*-Klasse.

2. **Graphics.** Dieses Modul ermöglicht die Definition einer Grafikszene durch hierarchisch angeordnete Grafikprimitive in Weltkoordinaten. Eine Grafikszene wird durch die Klasse *GraphicsManager* verwaltet. Durch die Fähigkeit des Grafikprimitives *GPgroup* weitere Grafikprimitive zu aggregieren, wobei Gruppen selbst wieder Gruppen enthalten können, entsteht eine Primitivhierarchie. Diese Hierarchie findet bei der Darstellung von komplexen Grafiken breite Anwendung. Jedes Grafikprimitiv referenziert einen *GraphicsContext*, der die Ausgabeattributierung enthält.
3. **Device.** Hier werden Ausgabegeräte angeordnet. Der Grafikkern tätigt alle seine Ausgaben über die Klasse *WorldGraphics*, welche in Zusammenarbeit mit einer *Transformation*, die realen Ausgabegeräte abstrahiert. Der Grafikkern kann alle Ausgabegeräte unterstützen, die eine (um Gleitpunktzahlen erweiterte) Implementierung der *Graphics* Klasse besitzen. Durch diesen Aufbau können frei verfügbare Implementierungen der *Graphics* Klasse einfach genutzt und integriert werden.

4. Prototyp eines GIS-Terminal

Zur Demonstration und Verifizierung der grafischen Leistungsfähigkeit von Java wurde ein Prototyp des GIS-Terminal erarbeitet. Dieser Prototyp implementiert Teile der vorgestellten Architektur (in nicht optimierter Form) und ermöglicht es Arc Shapefiles zu visualisieren. Insbesondere können Standardoperationen wie Zoomen und Verschieben des Grafikbereichs ohne weiteren Netzzugriffe erfolgen. Ein minimales Diagramm-Modul ist ebenfalls enthalten. Die Abbildungen 4 und 5 zeigen Bildschirmfotos des GIS-Terminal.

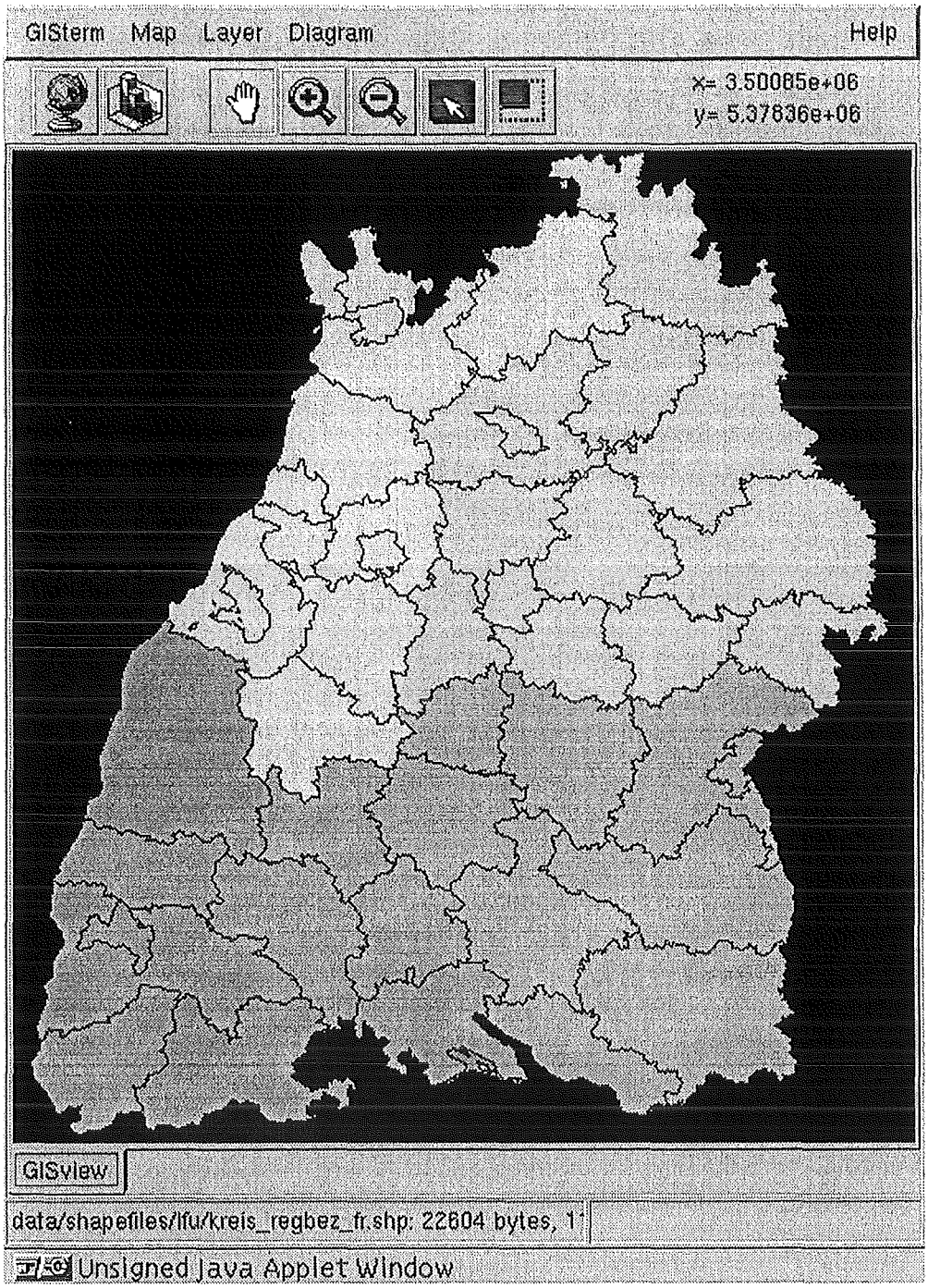


Abbildung 4: Kartenansicht

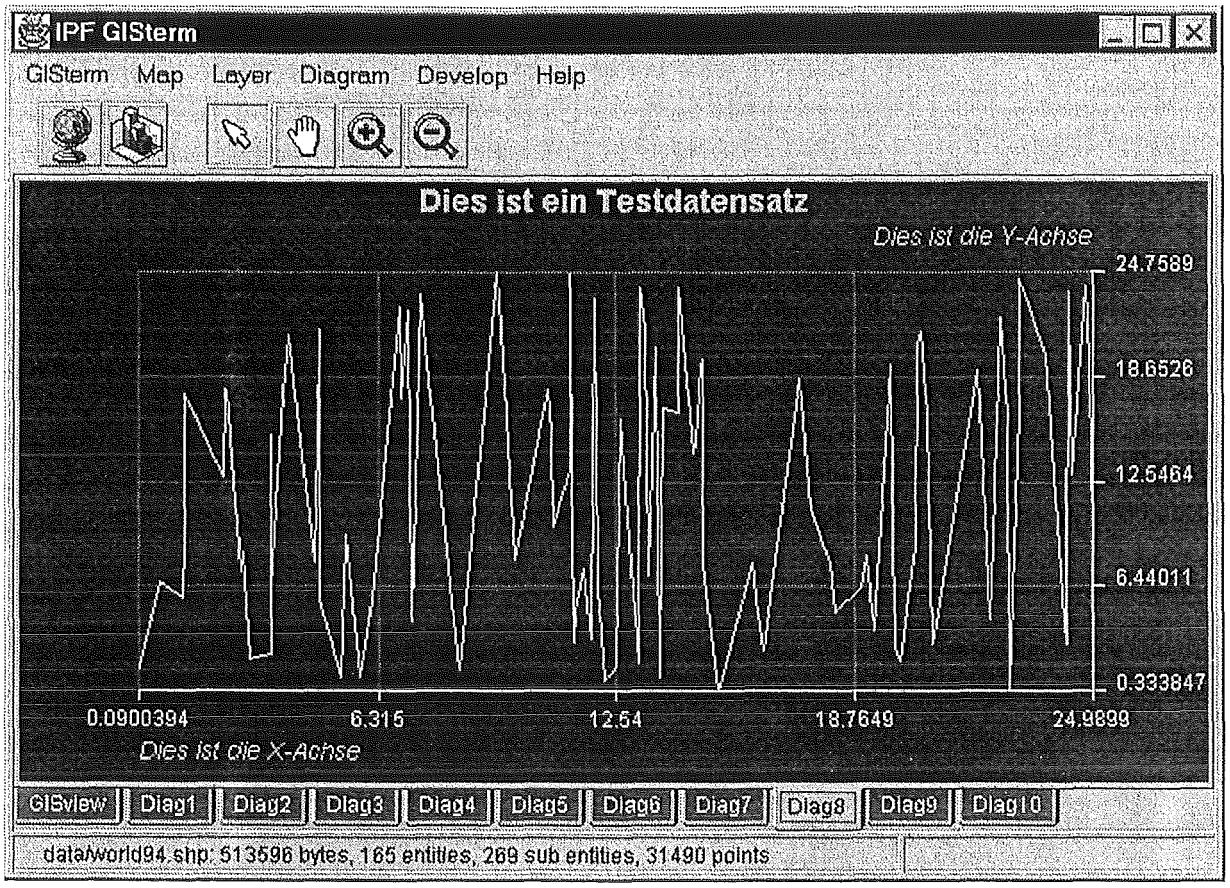


Abbildung 5: Diagrammansicht

5. Literatur

/1/ Balzert, Heide: Methoden der objektorientierten Systemanalyse, Wissenschaftsverlag, ISBN 3-411-17081-6.

/2/ Projekt GLOBUS: Konzeption und prototypische Realisierung einer aktiven Auskunftskomponente für globale Umwelt-Sachdaten im Umweltinformationssystem Baden-Württemberg; Phase II-1995, Wissenschaftliche Berichte des Forschungszentrums Karlsruhe, FZKA 5700, Dezember 1995.

/3/ JavaSoft: Remote method invocation and object serialization, Januar 1996, http://chatsubo.javasoft.com/current/.

/4/ JavaSoft: JavaSoft Announce JavaOS - Companies endorse JavaOS in desktop, consumer and embedded environments, May 1996, http://java.sun.com/java.sun.com/aboutJavaSoft/press/960529-javaos.html.

/5/ Krause, Ralph: Einsatz und Erprobung von Java in den Grafischen Diensten des WWW-UIS, unveröffentlichte Diplomarbeit am IPF und TECO, Universität Karlsruhe.

/6/ Hirano Satoshi: HORB - The Magic Carpet for Network Computing, October 1995, Electrotechnical Laboratory, http://ring.etl.go.jp/openlab/horb/.

Externe Dienste im UIS

F. Schmidt, R. Kopetzky, M. Schöckle, A. Schuch, J.-E. Schulz, M. Weigele,
Universität Stuttgart,
Institut für Kernenergetik und Energiesysteme (IKE),
Abteilung Wissensverarbeitung und Numerik,
Pfaffenwaldring 31,
70550 Stuttgart

1. Motivation

Der Begriff *Dienste im UIS* wurde im Bericht der AG Dienste /6/ eingeführt. Schwerpunkt der Arbeiten des IKE im Projekt GLOBUS III war die Beschäftigung mit externen Diensten. Unter externen Diensten verstehen wir Komponenten, die außerhalb des UIS entwickelt, gewartet und betrieben werden und für das UIS spezielle, wohldefinierte Aufgaben erfüllen. Sie werden von Diensteanbietern erbracht und von Dienstenutzern verwendet. Der Diensteanbieter ist in der Regel ein Spezialist, der in der Lage ist, den Dienst sachgerecht einzusetzen, zu warten und entsprechend dem Stand der Wissenschaft und der Gesetzgebung weiter zu entwickeln. Der typische Nutzer des UIS möchte solche Dienste bei Bedarf in Anspruch nehmen, dabei möglichst aus alternativen Diensteangeboten dasjenige auswählen, das für seine Problematik am geeignetsten ist und die Ergebnisse der Diensteinanspruchnahme im Kontext seiner normalen Tätigkeit weiterverwenden können.

Das IKE hat am Beispiel des externen Dienstes „luftgetragene Ausbreitung von Spurenstoffen" Anforderungen an solche Dienste untersucht. Zusammen mit dem FAW wurde mit dem Personal Data Node ein Klient entwickelt, der den Dienst auch bei blockierender Inanspruchnahme überwachen kann und in der Lage ist, die Ergebnisse der Diensteinanspruchnahme in eine für jeden Nutzer individuell konfigurierbare Arbeitsumgebung zu transferieren. Anforderungen an die Middleware wurden schon im Projekt TZUI spezifiziert. Zusammen mit dem FZI wurde untersucht, wie Wrapper gestaltet werden müssen, um externe Dienste unter CORBA zu nutzen.

2. Der Dienst luftgetragene Ausbreitung von Spurenstoffen

Der Dienst luftgetragener Ausbreitung von Spurenstoffen (MESYST /5, /4/) ist eine Weiterentwicklung des in UFIS/TULIS verfügbaren Dienstes CRAYSIM /3/. Die Weiterentwicklung erfolgte in enger Kooperation mit Fachabteilungen des UVM /8/. Sie führte zu einer substantiellen Verbesserung der Qualität des Dienstes, der von ihm erzeugten Ergebnisse und der Möglichkeit, den Dienst in Anspruch zu nehmen. Abb. 1 zeigt die Komponenten einer numerischen Ausbreitungssimulation, wie sie im Dienst luftgetragener Ausbreitung von Spurenstoffen verwendet werden.

Für die Generierung von Eingabedaten wurde eine graphische Benutzeroberfläche auf Basis der Seitenbeschreibungssprache HTML (Hypertext Markup Language) entwickelt, die laufend ausgebaut wird. Die Eingabe kann dann mit Hilfe eines WWW-Browsers (z.B. Netscape) erfolgen und läßt sich damit aus allen Anwendungskomponenten des Prokektes nutzen.

Es lassen sich zuerst das Gebiet und die Methode wählen mit welcher die Simulation durchgeführt werden soll. Abhängig von dem gewählten Gebiet und der gewählten Methode werden dann für die Eingabeseiten der einzelnen Module Defaultwerte vorgeschlagen, welche gegebenenfalls an den Realfall angepaßt werden können. Eine Übernahme von Eingabedaten aus verschiedenen im UIS verfügbaren Datenbanken ist möglich aber noch nicht realisiert.

Die erzeugten Eingabedaten werden jeweils in einem eigenen, eindeutigen Verzeichnis abge-

legt, welches für die Simulation als Arbeitsverzeichnis dient. Dieses Verzeichnis wird momentan beim Aufruf der Startseite der Ausbreitungssimulation erzeugt. In Zukunft soll jedoch eine Benutzer- und eine Auftragsverwaltung, sowie ein Repository für die Eindeutigkeit und Wiederidentifizierbarkeit einer Simulation und deren Simulationsdaten sorgen.

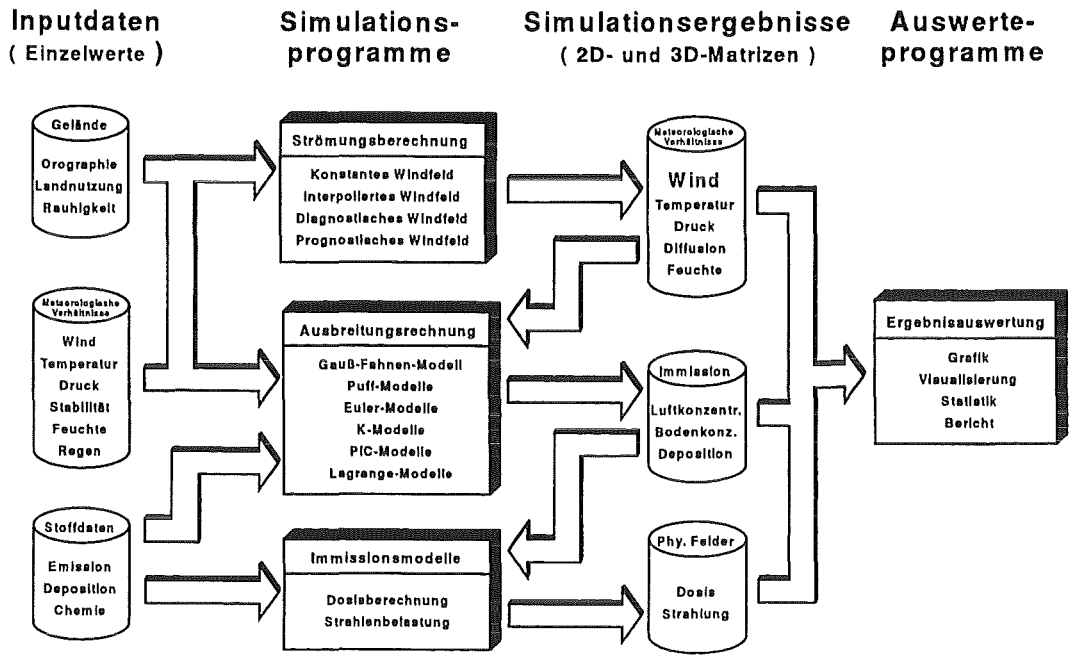


Abb 1 - Komponenten der numerischen Ausbreitungssimulation

2.1 Ausgewählte Module

Für die Durchführung der Simulationsrechnung stehen eine Reihe von Modulen zur Verfügung. Der Simulationsablauf wird durch eine Ablaufsteuerung gesteuert. Jeder einzelne Aufruf eines Moduls erfolgt dabei in einem eigenen Unterverzeichnis. Im folgenden wichtige Module kurz vorgestellt.

2.1.1 Module für die Windfeldberechnung

Für die Windfeldberechnung (Strömungsberechnung in Abb. 1) stehen wahlweise die folgenden zwei diagnostischen Modelle zur Verfügung.

a) WIND04 (WINDO)

WINDO berechnet massenkonsistente Windfelder in einem einem kartesischen Koordinatensystem. Es basiert auf einem diagnostischen Modell. Auf der Basis von, in festen Zeitabständen gemessenen Windmeßdaten lassen sich für diese Zeiten die jeweiligen Windfelder berechnen. Die Ausgabe der Windfelder erfolgt dabei in ein und dieselbe Datei, und ist somit für eine Verteilung des Gesamtproblems Ausbreitungsrechnung nicht optimal geeignet. In Abbil-

dung 2 ist der Datenfluß für dieses Modul dargestellt, von links kommend die Eingabe und rechts die Ausgabe. Die inneren Bezeichnungen geben die Dateinamen im Unterverzeichnis, die äußeren Bezeichnungen die Dateinamen im Arbeitsverzeichnis, auf deren Basis auch der Datenaustausch stattfindet, an. Die Steuerung des Moduls erfolgt durch die Steuerdatei EIN_WINDO.INP. Die in diesem bzw. den nächsten Abbildungen auftretenden Platzhalter xxx, xax und xbx bezeichnen Schleifenzähler für Zeitschritte bzw. Ausführungsnummern.

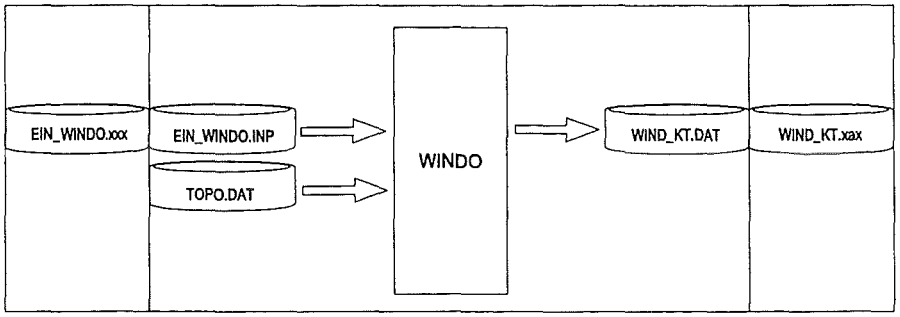


Abb. 2 - Datenfluß für WINDO

b) MCF (NOABL)

NOABL basiert ebenfalls auf einem diagnostischen Modell, berechnet die Windfelder aber in einem geländefolgenden Koordinatensystem. Damit läßt sich vor allem die Topographie besser berücksichtigen. Die Windfelder werden hier zu beliebigen Zeitpunkten durch jeweils einen Start des Moduls mit den entsprechenden Eingabedaten erzeugt.

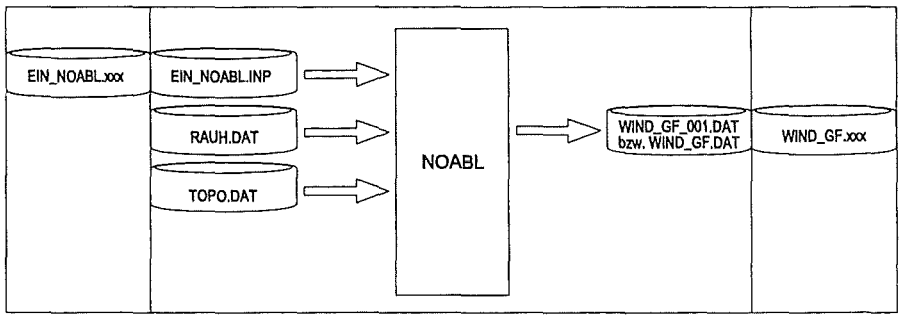


Abb. 3 - Datenfluß von NOABL

Somit ist eine sehr große Flexibilität gegeben. Dieses Modul wird auch im Rahmen des Nesting-Verfahrens eingesetzt. Analog zur Abb. 2 ist in Abb. 3 der Datenfluß für das Modul NOABL dargestellt. Die Steuerung erfolgt durch die Steuerdatei EIN_NOABL.INP. Abb. 4 zeigt den Datenfluß beim Einsatz des Moduls innerhalb des Nesting-Verfahrens, welches jedoch momentan von der graphischen Benutzeroberfläche noch nicht unterstützt wird.

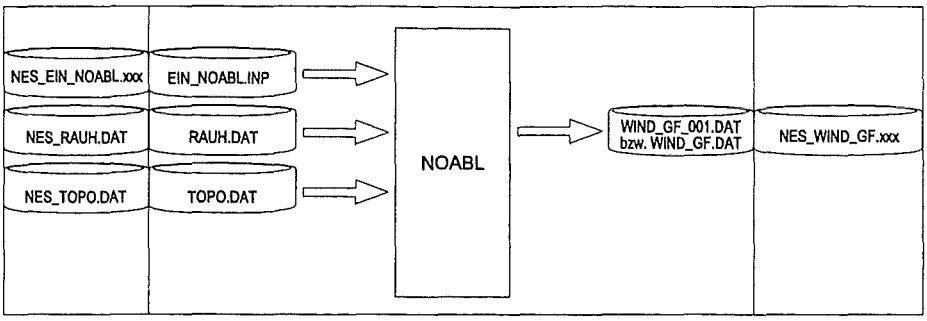


Abb. 4 - Datenfluß von NOABL im Rahmen des Nesting-Verfahrens

2.1.2 Module für die Ausbreitungsrechnung

Die Ausbreitungsrechnung erfolgt mit zwei Lagrange-Modellen.

a) PRWDA

Dieses Ausbreitungsmodul basiert auf kartesischen Windfeldern. Für die Nuklide elementares Jod, sowie Edelgase werden die Konzentrationsverteilungen berechnet. Es ist möglich die Depositionsrate unter Verwendung einer Depositionsgeschwindigkeit und gegebenenfalls eines für das gesamte Gebiet konstanten homogenen Niederschlages und daraus folgend konstanten Washouts zu bestimmen. Für die Berechnung weiterer Nuklidsorten muß das Modul entsprechend häufig gestartet werden. Der Datenfluß ist in Abb. 5 dargestellt. Die Steuerung erfolgt durch die Steuerdatei EIN_PRWDA.INP.

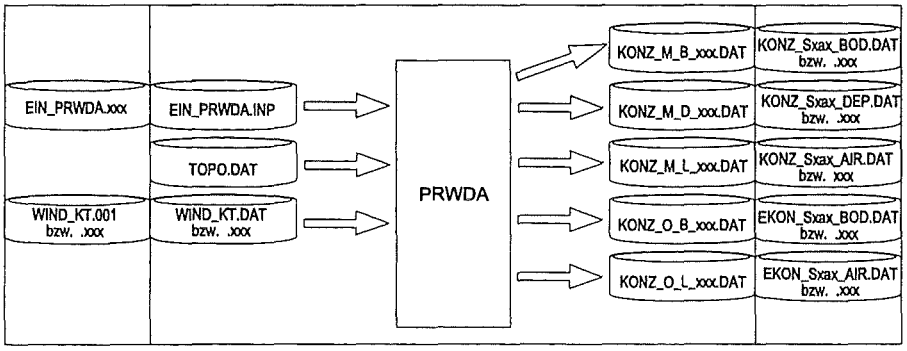


Abb. 5 - Datenfluß von PRWDA

b) PAS

PAS ist ein Lagrange-Ausbreitungsmodul basierend auf geländefolgenden Windfeldern. Die Diffusion wird wahlweise als Dreischichtmodell mit Ausbreitungsparametern oder ortsaufgelöst mit extern berechneten Diffusionskoeffizient modelliert. Es können 4 Nuklide, 3 mit Deposition und eines ohne Deposition, gleichzeitig berechnet werden. Niederschlag kann ortsaufgelöst berücksichtigt werden. Mit den defaultmäßig vorgegebenen Depositionskoeffizienten werden die Nuklidgruppen elementares Jod, organisches Jod, Aerosole und Edelgase berechnet. Andere Deposi- tionskoeffizenten können aber vorgegeben werden. Die Ausgabe erfolgt im kartesischen Koordinatensystem. Der Datenfluß für dieses Modul ist in Abb. 6 dargestellt. Die Steuerung erfolgt durch die Steuerdatei EIN_PAS.INP.

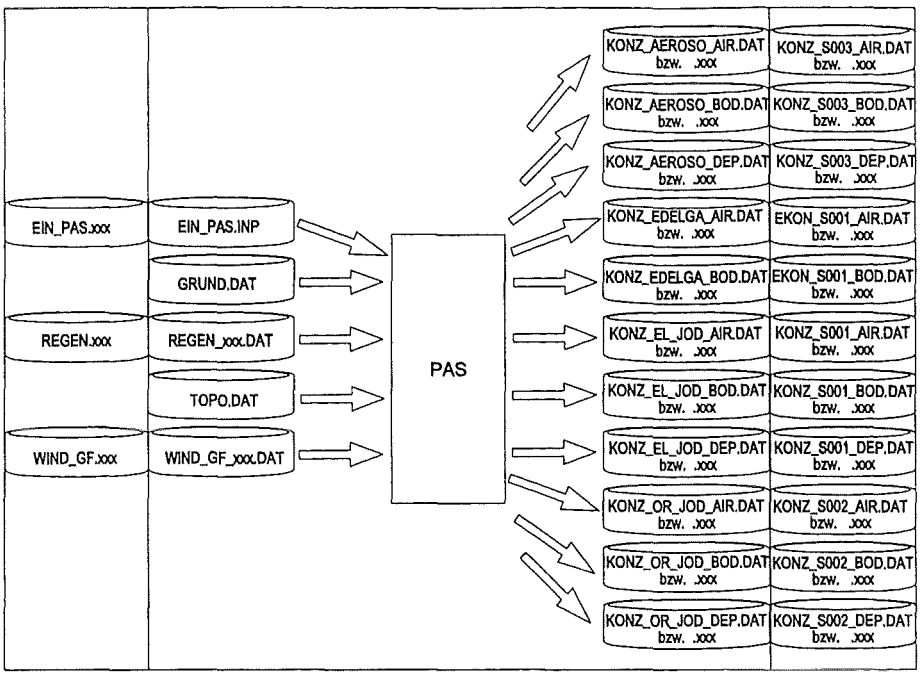


Abb. 6 - Datenfluß von PAS

2.1.3 Module für die Dosisberechnung

Für die Berechnung der Strahlendosis sind zwei Module besonders wichtig:

a) FAKTOR

FAKTOR ist ein Modul zur Umrechnung einer Konzentrationsverteilung in eine Aktivitätsverteilung. Hierbei wird eine Korrektur der trockenen Deposition mittels unterschiedlicher Depositionsgeschwindigkeiten für die Nuklidgruppen elementares Jod, organisches Jod und Aerosole berechnet. Beim Auftreten von Niederschlag ist dieser Weg jedoch nicht mehr gangbar.

b) FAKTOR2.

Bei diesem Modul erfolgt keine nachträgliche Depositionskorrektur, daher ist es auch für Niederschlag einsetzbar. Zusätzlich ist es wesentlich flexibler im Datenhandling und damit für einen Einsatz in einer verteilten Umgebung besser geeignet als das Modul FAKTOR.

Weitere Module zur Berechnung verschiedenartiger Dosen sind verfügbar und in einem eigenen Bericht /8/ beschrieben.

2.1.4 Ergebnisse

Als Ergebnis einer MESYST-Rechnung werden für jeden Zeitschritt Depositionskarten erstellt und in HTML-Seiten integriert. Dies ist in der Regel nur für eine erste Betrachtung ausreichend. Notwendig ist, daß der Benutzer selber angeben kann, wie die Auswertung erfolgen soll. Dazu existiert ein zweistufiges Konzept.

Eine Schnellauswertung der Simulation kann schon während der Simulationsrechnung mit Hilfe des Moduls PLOT_AGL geschehen. Der Nachteil dabei ist, daß schon vorab spezifiziert werden muß was graphisch aufbereitet werden soll. Diese Schnellauswertung ersetzt deshalb keine richtige Auswertung und ist auch nur für den Notfall, wenn nichts anders zur Verfügung steht, zu verwenden.

Im Normalfall erfolgt die Auswertung der Ergebnisdaten auf Seite des Klienten. Dabei wird der Anwender durch den Personal Data Node (PDN) unterstützt /7/.

3. Eigenschaften externer Dienste

Anhand der Beschreibung des Dienstes luftgetragener Ausbreitung von Spurenstoffen können einige charakteristische Eigenschaften externer Dienste aufgezeigt werden. Dazu gehören:

- Externe Dienste sind in der Regel mit langen Transaktionen verbunden. Es muß daher möglich sein, sie nicht blockierend auszuführen und Statusinformationen bzw. Zwischenergebnisse abzufragen. Ferner ist es notwendig, Grundüberlegungen für ein dienstespezifisches Transaktionskonzept anzustellen.
- Externe Dienste benötigen häufig große heterogene Datensätze als Eingabe. Sie sollten in der Regel vom Diensteanbieter bereitgestellt werden, so daß der Dienstenutzer nur noch die für seine Dienstinanspruchnahme charakteristische Parameter angeben muß.
- Externe Dienste produzieren häufig große heterogene Datensätze als Ergebnis. Der Dienstenutzer benötigt davon nur eine Teilmenge, die aber variabel sein kann. Externe Dienste müssen also Zugang zu einem Repository haben, von dem aus Ergebnisse bei Bedarf abgerufen werden können.
- Externe Dienste enthalten häufig eine Vielzahl von Funktionalitäten. Sie sollten daher in einfachere Komponenten aufgebrochen werden, damit sie entsprechend den Wünschen der Diensteanforderer konfigurierbar sind.
- Externe Dienste, die aus Komponenten aufgebaut sind, benötigen eine Ablaufsteuerung, die entsprechend der vom Dienst angebotenen Optionen komplex werden kann.
- Dienstekonfiguration, Datenbereitstellung und Ergebnisinterpretation sind daher häufig Aufgaben, die durch Techniken kooperativen Arbeitens (Multimedia Kooperation) unterstützt werden sollten.
- Externe Dienste können von verschiedenen Nutzern gleichzeitig in Anspruch genommen werden. Es ist daher nötig, daß sie ihre Aufträge, den Status der Auftragsbearbeitung und die Ergebnisse selbständig verwalten.
- Externe Dienste basieren häufig auf Programmen, die über Jahre entwickelt und validiert wurden. Trotzdem werden sie weiterentwickelt und sind daher häufig Änderungen unterzogen. Sie benötigen daher ein Konfigurationsmanagement.
- Externe Dienste sind oft in Sprachen geschrieben, die nicht oder nur wenig objektorientiert sind und verlangen manchmal spezielle Rechner für ihren Betrieb. Um solche Dienste auch aus dem UIS heraus nutzen zu können, sind anpaßbare Zugangsmechanismen nötig.

Aus der Vielzahl dieser Aufgaben haben wir uns im Projekt GLOBUS III auf die Schwerpunkte

- Kommunikation,
- Repositories zur Auftrags- und Ergebnisverwaltung und
- Möglichkeiten der Nutzung von CORBA-Services

konzentriert.

Wichtige Ergebnisse dieser Arbeiten werden in den folgenden Unterabschnitten dargestellt.

4. Kommunikation mit externen Diensten

Im Bereich der Kommunikation haben wir zwei Schwerpunkte weiterverfolgt. Zum einen haben wir uns mit Techniken befaßt, um Dienste, die nicht blockierend in Anspruch genommen werden, zu überwachen. Es zeigte sich, daß dies in Ablösung der noch im Projekt GLOBUS verwendeten Techniken auf CCI-Basis auch über ein in JAVA erstelltes Poll Applet geschehen kann. Dies wird ausführlicher im Bericht der AG-JAVA beschrieben.

Der zweite Arbeitsschwerpunkt lag bei der Weiterentwicklung des Kommunikationsinterpreters (KIP). Der Kommunikationsinterpreter ermöglicht einerseits, daß komplexe, sich öfters ändernde Datenstrukturen für den Austausch von Ein- und Ausgabedaten übertragen werden können und andererseits, daß die Schnittstelle so gestaltet ist, daß Fortschritte im Kommunikationsbereich genutzt werden können, ohne daß Dienst oder Diensteanforderung geändert werden müssen. Dies ist bei Diensten mit langer Einsatzdauer von Vorteil. Außerdem ermöglicht es der Kommunikationsinterpreter Dienste mit langer Laufzeit nichtblockierend in Anspruch zu nehmen. Darüber hinaus wird es möglich, auch proprietäre Kommunikationslösungen und Plattformen als Teile eines Gesamtsystems zu nutzen.

Mit dem verstärkten Einsatz von CORBA als Middleware gewinnt die letzte Eigenschaft besondere Bedeutung. Sie kann genutzt werden, um Interoperabilität verschiedener Middleware-Architekturen zu gewährleisten. Dadurch besteht die Möglichkeit, von den Fähigkeiten unterschiedlicher Ansätze zu profitieren. Dies wird beispielhaft an einer Kooperation des Kommunikationsinterpreters mit der CORBA-Implementierung ORBIX gezeigt. Dabei wird von CORBA die objektorientierte Verteilungsmöglichkeit und der transparente Zugriff auf verteilte Objekte durch ein Repository für die Ablage von Schnittstellenbeschreibungen von Diensten genutzt. Durch die Einbindung des KIP wird es ermöglicht, daß Dienste, die unter VMS zur Verfügung stehen, auch von anderen Betriebssystemen über CORBA in Anspruch genommen werden können. In diesem Falle profitiert CORBA von den Fähigkeiten des Kommunikationsinterpreters, Dienste unter dem VMS-Betriebssystem anzusprechen.

5. Anforderungen an ein serverseitiges Repository

Unter einem Repository verstehen wir einen Behälter (Datentopf), in dem Dienste Daten ablegen können, die von ihnen infolge einer Diensteinanspruchnahme produziert werden. Repositories arbeiten auf der Grundlage von Datenobjekten. Dies sind Daten (z.B. in Dateien) mit einer beliebigen internen Struktur sowie diesen Daten zugeordnete Attribute. Die Attribute enthalten alle benötigten Verwaltungsinformationen.

Alle Operationen eines Repositories müssen garantieren, daß der interne Zustand der Daten eines Datenobjekts nicht verändert wird. Anders verhält sich die Beziehung des Repositories zu dem Attribut-Teil des Datenobjekts. Diese repository-interne Datenstruktur wird nur durch Repositories angelegt, geändert und zusammen mit dem Datenteil des Datenobjekts wieder gelöscht. Zusammenstellungen bestimmter Attribute eines Datenobjekts können durch den Nutzer (Anwender oder Programm) vom Repository erfragt werden.

Wichtige Eigenschaften eines Repositories sind

- **Passivität:** Das Repository führt Operationen nur auf Anforderung durch. Es ist somit nicht in der Lage, selbständig Operationen zu veranlassen.
- **Datensicherung:** Operationen auf Datenobjekten werden nur nach Authentifizierung (durch passende Session-Kennung oder Benutzername) durchgeführt.
- **Identifikation, Kennung:** Alle Datenobjekte in Repositories verfügen über eine bidirektional eindeutige Adresse, ihre Kennung. Der Zugriff auf Daten erfolgt ausschließlich über diese Kennung. Die Abfrage von Attributen kann über diese Kennung oder mittels anderer Methoden (z.B. SQL-Queries) erfolgen.
- **Unveränderbarkeit:** Daten in einem geschlossenen Datenobjekt können nicht mehr verändert (sondern nur noch repliziert oder gelöscht) werden.
- **Alterung:** Jedes Datenobjekt verfügt über eine ihm zugeordnete Lebensdauer. Ist diese überschritten, kann das Datenobjekt (z.B. aus Speicherplatzgründen) von der Zeitverwaltung gelöscht werden.
- **Unabhängigkeit von Implementierungen:** Jede Inkarnation eines Repositories ist ausschließlich für die Verwaltung und Speicherung der ihm unterstellten Datenobjekte zuständig. Es ist nicht zulässig, daß sich zwei Inkarnationen von Repositories einen Speicherbereich teilen.
- **Beliebiges Speicherformat:** Die Art und Weise, wie ein Repository die ihm unterstellten Datenobjekte speichert, ist nicht festgelegt. Somit sind „sichere" (durch Verschlüsselung) und „kompakte" (durch Komprimierung) Repositories realisierbar.

Die Funktionalitäten der Repositories **R** entstehen aus den sinnvollen Kombinationen der Grundoperationen **G**={*erzeugen*[1], *abschließen*, *einfügen*, *herausholen*, *löschen*} mit den Bestandteilen der Datenobjekte **DO**={*Daten*, *Attribute*}. Zum Auffinden von Datenobjekten ist zusätzlich ein Auskunfts- bzw. Abfrage-Mechanismus **Q** (Query) notwendig, welcher Session- **S**, Auftrags- **A** und Nutzer- **U** (User) Attribute zur Bestimmung der Kennung **K** von Datenobjekten verwendet.

Zur Integration von bestehenden Anwendungssystemen kann die bisherige Methodik zur Speicherung von Daten nicht unberücksichtigt bleiben. Aus diesem Grund muß das Repository über Export- und Importfunktionen in Dateisysteme verfügen.

[1] *Erzeugen* und *Abschließen* sind Hilfsoperationen zum iterativen Aufbau von Datenobjekten.

Funktion	*Beschreibung*
R.erzeugen → K	Neues, leeres Datenobjekt im Repository **R** anlegen, liefert Kennung **K** für das Datenobjekt.
R.einfügen (R'.K) → K'	Replizieren eines Datenobjekts mit Kennung **R'.K** aus dem Repository **R'** in das Repository **R** (incl. aller Attribute), liefert Kennung **K'** der Replikation zurück.
R.herausholen (K, Pfad) → DS	Kopieren des Datenobjekts **K** in das Verzeichnis **Pfad** im Dateisystem **DS**
R.löschen (K)	Entfernt ein Datenobjekt **K** aus dem Repository **R**
R.erzeugen (K)	Diese Operation ist nicht verfügbar (entspricht: erzeugen eines Datums ohne Inhalt in einem Datenobjekt)
R.einfügen (K, Pfad) → K'	Kopiert eine Datei mit **Pfad** aus dem Dateisystem in das Datenobjekt **K**. Die Orginaldatei bleibt erhalten.
R.einfügen (K, K')	Fügt ein Datenobjekt **K'** in das Datenobjekt **K** ein, das Repository muß hierbei **K'** nicht unbedingt replizieren.
R.abschließen (K)	Ändern des Zustands eines geöffneten Datenobjekts mit Kennung **K** zu einem geschlossenen Datenobjekt, damit sind auf **K** keine weiteren ***Einfüge*** Operationen zulässig.
R.attribut (K, einfügen)	Einfügen, Ändern und Löschen von beschreibenden Attributen zu einem Datenobjekt. Diese Operationen sind auch auf abgeschlossene Datenobjekte anwendbar.
R.attribut (K, ändern)	
R.attribut (K, löschen)	
Q.K (R, {S, A, U}*) → { K }*	Kennungsanfrage Abfrage-Mechanismus **Q** liefert anhand einer beliebigen Folge von Session- **S** Auftrags- **A** oder Nutzer- **U** Attributen eine Menge von Kennungen **{ K }*** für Datenobjekte in dem Repository **R**
Q.attribut (R, { S | A | U }*) → { S | A | U }*	Attributanfrage, Abfrage-Mechanismus Q liefert anhand von Attributfolge eine Menge von Attributen als Ergebnis.

Tab. 1 - Funktionsumfang des Repositories

6. Umgang mit externen Diensten - Der Personal Data Node als Framework eines Klienten

6.1 Der Personal Data Node

Der *Personal Data Node* (PDN) ist die zentrale Komponente der Klient-Server-Architektur auf Anwender-Seite. Er verknüpft die einzelnen Bereiche des persönlichen Arbeitsplatzes und führt, falls notwendig, die Schritte *Datentransfer, Datenentschlüsselung* und *Datenkonvertierung* durch. Er verfügt über eine Komponente zur persistenten Speicherung von Datenobjekten und Methoden, die im folgenden Text als *Repository* bezeichnet wird. Dabei werden mit *Datenobjekten* die auf Datenstrukturen abgebildeten Informationen bezeichnet, *Methoden* sind Skript-Dateien, die das Wissen um die Verarbeitung bestimmter Informationsstrukturen repräsentieren. Die Zuordnung von Datenobjekten und Methoden zu Werkzeugen und Hilfsmodulen wird durch Konfigurationsdateien auf der Basis von für den Arbeitsplatz gültigen Entscheidungstabellen unterstützt.

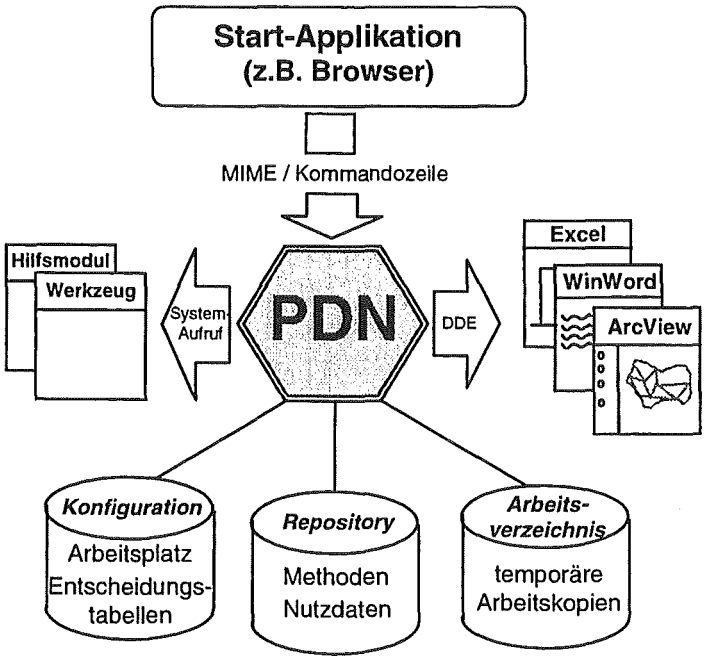


Abb. 7 - Architektur des „Personal Data Node" (PDN)

Der in Abb. 7 dargestellte modulare Aufbau des PDN berücksichtigt die wesentlichen Forderungen nach Erweiterbarkeit und Wartbarkeit. Hauptkomponenten des Kernsystems sind der *PDN-Kernel* und der *PDN-Manager.*

Der *PDN-Kernel* bildet den Kern des Systems. Er integriert die Hilfsmodule zum Datentransfer und der Datenkonvertierung. Die Konfigurierung erfolgt ausschließlich über den *PDN-Manager.* Neben der Verwaltung des persönlichen Arbeitsbereichs sowie der interaktiven Kommunikation mit dem Benutzer über eine graphische Benutzer-Schnittstelle (GUI), ist zusätzlich eine Komponente zur Verwaltung des Repositories integriert. Die Anknüpfung vor-

handener Werkzeuge erfolgt mittels „Dynamik Data Exchange“ (DDE) bzw., bei nicht-DDE-fähigen Programmen, über System-Aufrufe und Kommandozeilen-Parameter.

Unter *Werkzeugen* versteht man in diesem Zusammenhang zum einen Standardapplikationen aus den Bereichen Textverarbeitung, Datenbanken und Tabellenkalkulation, zum anderen zählen hierzu auch Fachanwendungen wie Software zur Datenanalyse und Visualisierung raumbezogener Informationen (z.B. ArcView). Start-Applikationen sind Informations-Browser, die den Anwender bei der Auswahl von Informationen unterstützen.

Interne und externe *Hilfsmodule* sind Programme zur Komprimierung, Dekomprimierung, Verschlüsselung, Entschlüsselung oder Konvertierung. Die Komprimierung dient der Reduzierung der Datenmenge sowohl bei der Übertragung der Nutzdaten als auch bei deren persistenten Speicherung. Ver- und Entschlüsselung mit unterschiedlichen Verfahren bildet die Grundlage für die, im Hinblick auf den Inhalt der Informationen sichere Übertragung der Datenobjekte. Desweiteren bildet die Möglichkeit zur Datenkonvertierung einen wichtigen Zwischenschritt bei der Verknüpfung von Start-Applikationen und Werkzeugen.

6.2 Ablauf einer Sitzung mit dem PDN

In diesem Abschnitt soll an einem kurzen Beispiel der prinzipielle Ablauf einer Sitzung erläutert werden. In deren Verlauf ruft ein „Info-User“ über einen Web-Browser Informationen über das durchschnittliche NO_x-Aufkommen des vergangenen Monats in Baden-Würtemberg ab. Für einen Bericht sollen diese Informationen kartographisch aufbereitet werden. Abb. 8 zeigt schematisch den Ablauf einer solchen Sitzung.

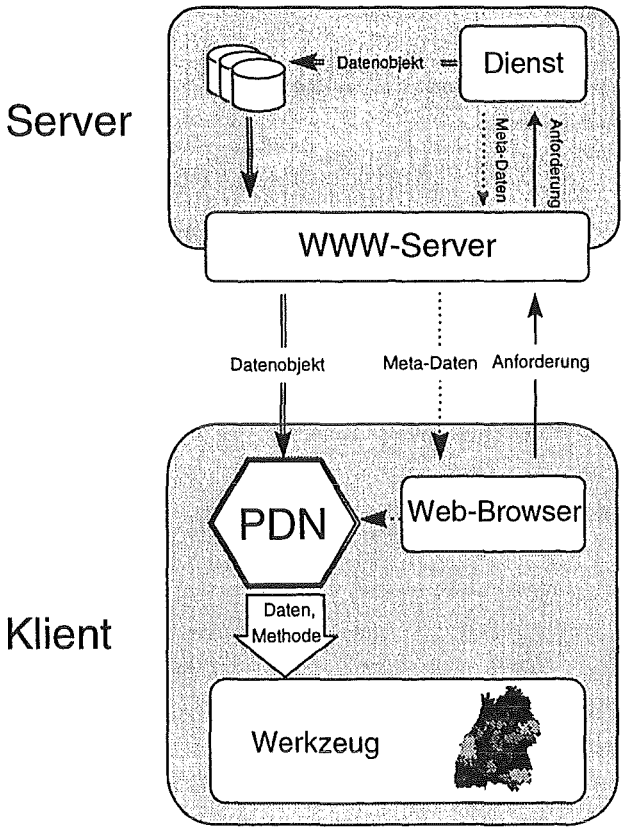


Abb. 8 - Schematischer Ablauf einer Sitzung

Dabei laufen im Einzelnen folgende Schritte ab:

- Über einen Web-Browser (z.B. Netscape) wählt der Anwender die benötigten Informationen aus. Ein geeigneter Dienst stellt die Ergebnisdaten als Datenobjekt auf dem Server bereit und liefert als unmittelbares Ergebnis Informationen über das Datenobjekt (Metadaten).
- Anhand dieser Informationen bietet der PDN dem Anwender eine Auswahl möglicher Darstellungsarten (Methoden), die für jeden Arbeitsplatz individuell angepasst werden können. In diesem Fall wählt der Anwender eine Darstellung als Kreiskarte, wobei die Kreise entsprechend den Mittelwerten des NO_x-Aufkommens eingefärbt werden sollen.
- Sind nun die Randbedingungen festgelegt, wird das Datenobjekt auf den Klienten transferiert, entpackt und für die Erfordernisse des jeweiligen Werkzeugs über Hilfsmodule aufbereitet. Für die Darstellung in ArcView werden mit einem Statistik-Modul die Mittelwerte über die jeweiligen Kreise gebildet und in einer ASCII-Tabelle abgelegt.
- Als letzter Schritt folgt die automatische Generierung der vom Anwender gewählten Darstellung mit Hilfe eines geeigneten Werkzeugs und lokal verfügbarer Methoden. Hier generiert ein Avenue-Skript (Avenue = Skriptsprache von ArcView) aus der bereitgestellten Tabelle die Darstellung auf Kreisebene.

6.3 Werkzeuge des PDN

Unter *Werkzeugen* verstehen wir zum einen Standardapplikationen aus den Bereichen Textverarbeitung, Datenbanken und Tabellenkalkulation, zum anderen zählen hierzu auch Fachanwendungen wie Software zur Datenanalyse und Visualisierung raumbezogener Informationen (z.B. ArcView). Als Kernsystem eines integrierten Arbeitsplatzes ist der PDN in der Lage, die in der jeweiligen Arbeitsumgebung vorhandenen Werkzeuge zu koppeln.

Man unterscheidet, abhängig von den möglichen Kommunikationsmechanismen, zwei Grundtypen von Werkzeugen:

- DDE-fähig Werkzeuge (z.B. ArcView, Excel, Winword...)
- nicht-DDE-fähige Werkzeuge (z.B. Notepad, Gnuplot, ...)

6.3.1 Die Einbindung von Excel

Das Programm Microsoft Excel ist die derzeit meistgenutzte Tabellenkalkulation im Büro- und Heimbereich. Für die Aufbereitung und statistische Analyse von kleinen und mittleren Datenmengen bietet Excel eine breite Palette von Funktionen und Hilfestellungen. Neben der tabellarischen Darstellung können Daten auch in Form von Business-Grafiken visualisiert werden.

Excel besitzt eine leistungsfähige Skriptsprache, um kleinere und größere Anwendungen selbst zu erstellen. *Visual Basic für Applikationen* (VBA) stellt eine vollständige Programmierumgebung bereit, die annähernd identische Funktionalitäten wie die eigenständige Entwicklungsumgebung *Visual Basic v4.0* aufweist.

Für den Datenaustausch zwischen Programmen unterstützt VBA neben OLE2 auch DDE. Folgende Methoden stehen in VBA zur Verfügung:

Kurzbeschreibung	Methode
Kommunikation aufbauen	Kanal=DDEInitiate(Applikation,
Klient liest Daten vom Server	DDERequest(Kanal, DatenElement)
Daten vom Klient zum Server übertragen	DDEPoke(Kanal, DatenElement,
Klient sendet Befehle zur Ausführung	DDEExecute(Kanal, Befehl)
Kommunikation beenden	DDETerminate(Kanal)

Tab. 2 - VBA-Methoden zur Kommunikation über DDE

Bei der Kommunikation zwischen PDN und Excel erfolgt der Verbindungsaufbau durch den PDN. Excel ist damit als DDE-Server ein Werkzeug hinsichtlich des PDN, der als DDE-Klient eine Reihe von Leistungen erwartet. Dazu gehören das Laden von Daten und die Ausführung bestimmter VBA-Module.

Die Kommunikation zwischen dem PDN und dem jeweiligen DDE-Werkzeug läuft in drei Schritten ab:

- Vorbereiten der DDE-Kommunikation (AppStartUp)
- Laden der Daten (LoadData)
- Laden und Ausführen der Methode (LoadMethod)

Diese Schritte werden von den DDE-Modulen des PDN ausgeführt und verwenden dabei mehrere VBA-Programme. Dies wird detaillierter im PDN Bericht /7/ beschrieben.

Die Methode InsertData lädt die einzelnen Datendateien je nach Anforderung in eine neue oder in eine bereits bestehende Arbeitsmappe. Folgende Datei-Endungen sind für den Datenimport nach Excel definiert:

Datenformat	Datei-Endung
Excel-Tabelle	xls
ASCII-Tabelle (Trennzeichen Tabulator)	txt
dBase III / IV - Tabellen	dbf

Tab. 3 - Für den Datenimport nach Excel zugelassene Dateiendungen

Das Arbeiten mit den Daten erfolgt in Excel über arbeitsplatzspezifische Methoden-Skripte, die der PDN bereitstellt. Die Methode InsertMethod lädt und startet das zur Bearbeitung ausgewählte Methoden-Skript.

6.3.2 Die Einbindung von ArcView

Bei ArcView2 von ESRI handelt es sich um eines der bekanntesten und am weitest verbreiteten Desktop-Mapping Programme. Es wurde als Werkzeug zum Darstellen und Abfragen von Geodaten und Sachdaten auf Arbeitsplatzrechnern entwickelt, das auch weniger geübten Nutzern den Zugang zu Geodatenbeständen ermöglichen soll.

Für die Darstellung von geographischen Daten stehen in ArcView vier Grundtypen zur Verfügung:

Views:
Ein View besteht aus einer oder mehreren Ebenen mit thematisch organisierten Daten, sogenannten Themen. Es können verschiedene Themen folienartig übereinandergelegt werden und bei Bedarf über eine zugeordnete Checkbox ein- und ausgeschaltet werden. Alle geographischen Objekte eines Themas sind von der gleichen Objektart: Punkt, Linie oder Fläche. Dabei sind auch Bilddaten als Folien zugelassen. (Abb. 9)

Tabellen:
Zur Speicherung von Sachdaten benutzt ArcView Tabellen. Diese können bearbeitet und, mit geographischen Objekten verknüpft, in thematischen Karten visualisiert werden. Tabellen sind dynamisch, d.h. ihr Inhalt ändert sich, sobald sich ihre Datenquelle ändert.

Diagramme:
Diagramme sind eine weitere Möglichkeit der Darstellung tabellarischer Daten in ArcView. Sie beziehen sich in ihrer Darstellung auf Tabellendaten innerhalb eines Projekts und geben dynamisch den aktuellen Status einer Tabelle wieder.

Layout:
Auf einer frei definierbaren Seite werden Views, Tabellen, Diagramme, importierte Graphiken und graphische Grundformen für die Ausgabe zusammengestellt. Diese Darstellungsart stellt auch den Export in andere Dateiformate zur Verfügung.

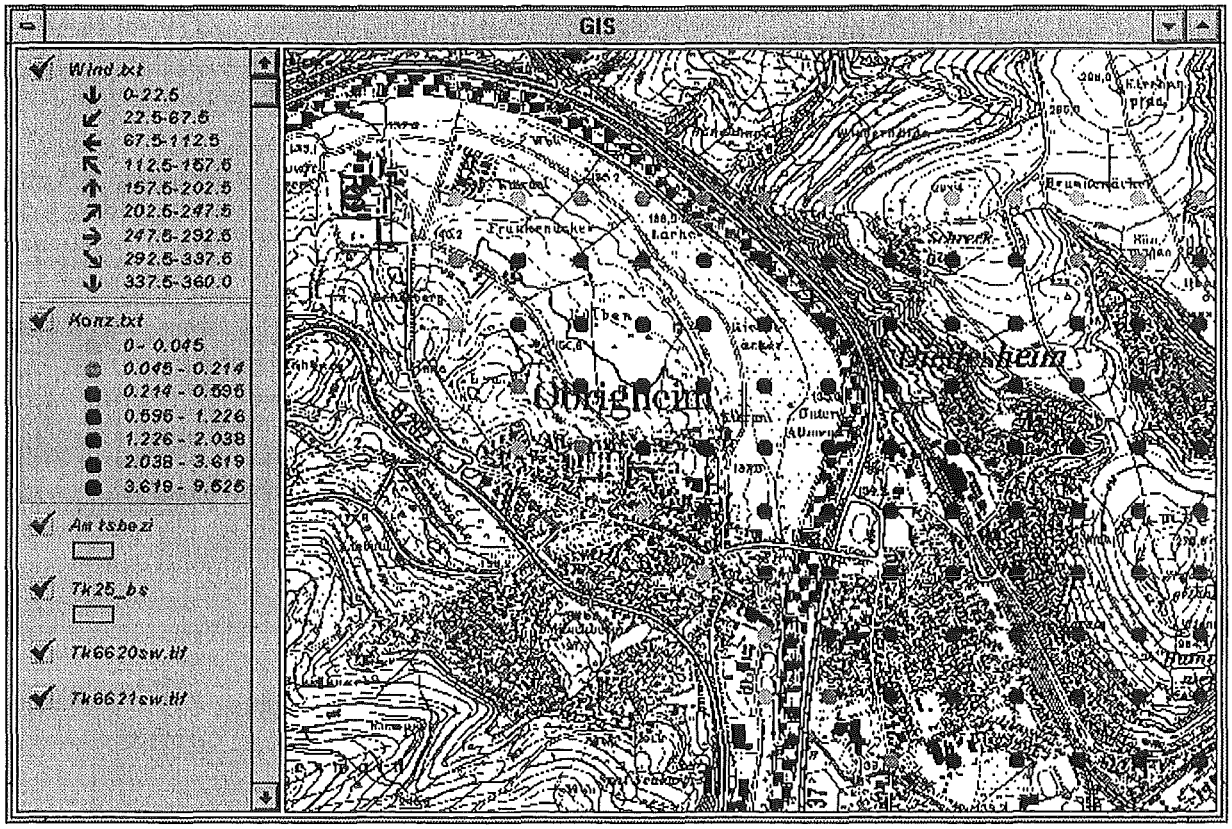


Abb. 9 - View mit mehreren Themen

Diese Komponenten werden für jede ArcView-Sitzung in einer sogenannten Projekt-Datei

(*.apr) zusammengefaßt. Diese enthält neben den Definitionen der verschiedenen Darstellungen auch Verweise auf die topographischen und tabellarischen Attributdaten, auf denen die Darstellungen aufbauen. Desweiteren sind in der Projektdatei Informationen über die Konfiguration der Benutzeroberfläche sowie über die Position und Größe der geöffneten Fenster gespeichert. Damit können für jedes Projekt die Menüs, Schaltflächen und Werkzeuge individuell angepaßt werden. Dies eröffnet die Möglichkeit, dem Anwender abhängig von seinen Vorkenntnissen und Bedürfnissen eine „maßgeschneiderte" Benutzeroberfläche zu bieten.

Für den Umgang mit diesen Grundtypen lassen sich mit Hilfe der Skriptsprache Avenue Methoden erstellen und in Projektdateien einbinden. Dort können sie, z.B. über die Verknüpfung mit Elementen der graphischen Benutzeroberfläche (Button, Pulldown-Menü), verwendet werden, um den Funktionsumfang von Arcview um individuelle Erfordernisse des Anwenders zu erweitern.

Es lassen sich aber auch externe Programme einbinden. Ein Beispiel für die Einbindung externer Hilfsmodule ist der „Isolinien-Generator". Er bietet die Möglichkeit, aus einer Matrix Polygonzüge zu generieren und diese in ein für ArcView lesbares Shape-File umzuwandeln (Abb. 10). Die damit erzeugten Isolinien umschließen Gebiete gleicher oder höherer Werte und sind beispielsweise gut zur Visualisierung von Schadstoffausbreitung geeignet.

Die Funktionalität, Shape-Files mit Polygonzügen für ArcView zu schreiben, ist in zwei Teilprogramme aufgesplittet:

- **isoline.exe**

Funktion:	Erzeugung von Polygonen aus einer Matrix
Eingabe:	Matrix mit Meßwerten, zusätzliche Informationen zur Berechnung
Ausgabe:	formatfreie ASCII-Datei mit Polygonzügen

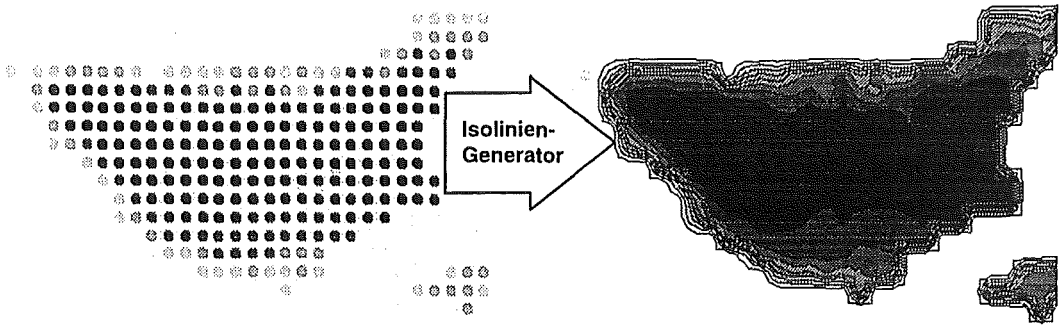


Abb. 10 - Einsatz des Isolinien-Generators zur Erzeugung von Polylinien

- **shape.exe**

Funktion:	Lesen der Datei mit Polygonzügen und schreiben des binären Shape-Files und des dBase-Files
Eingabe:	ASCII-Datei mit Polygonzügen, zusätzliche Informationen
Ausgabe:	Shape-Files, dBase-File

6.3.3 Einbindung von Gnuplot

Gnuplot ist ein Graphikprogramm zur interaktiven Visualisierung von Daten. Die Eingabe der Zeichenbefehle erfolgt über eine Kommandozeile. Dabei besteht die Möglichkleit, Befehlssequenzen zur Generierung unterschiedlicher Darstellungen in Skript-Dateien (*.gp) zusammenzufassen. Zu den umfangreichen Möglichkeiten von Gnuplot v3.6 gehört:

- Datenreihen (z.B. Zeitreihen)
- Balkengraphiken
- 3D-Oberflächenansichten (siehe Abb. 11)
- 2- und 3-dimensionale Darstellung parametrisierter Funktionen

Die Einbindung erfolgt über Kommandozeilenaufruf. Das Laden der Daten und die Schritte zur Visualisierung müssen dabei durch den Anwender ausgeführt werden.

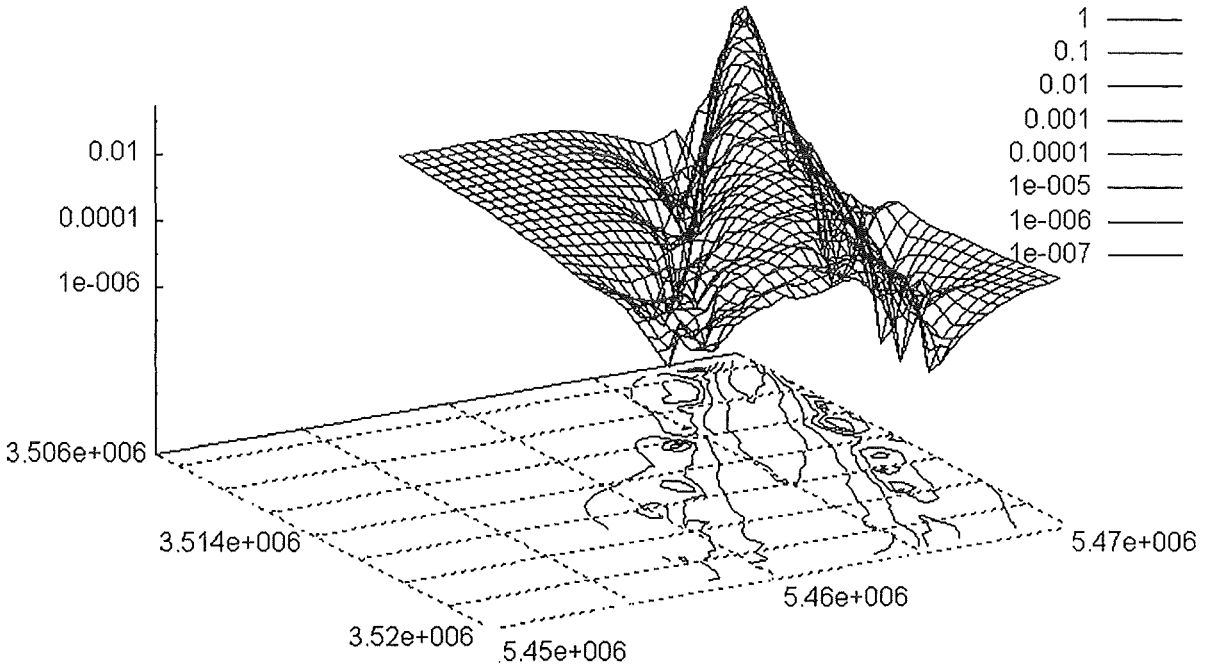


Abb. 11 - Konzentrationsverteilung von Jod mit projizierten Isolinien

7. Externe Dienste unter CORBA

Sollen externe Dienste in einer CORBA-basierten Umgebung genutzt werden, so müssen sie mit einer Kapsel (Wrapper) versehen werden, die den externen Dienst als CORBA-Objekt erscheinen läßt. Solch eine Kapsel erlaubt es zum einen den externen Dienst über eine IDL-Schnittstelle anzusprechen und stellt zum anderen Funktionalitäten bereit, die die Verwaltung des Dienstes und der Ergebnisse seiner Inanspruchnahme unterstützen. Am Beispiel des Dienstes Ausbreitung luftgetragener Spurenstoffe haben wir gezeigt, warum externe Dienste häufig sehr komplex sind. Bei der Erstellung eines Wrappers müssen daher eine Reihe nicht-trivialer Fragen gelöst werden. Dazu gehören:

- die Bereitstellung der für die Simulation benötigten heterogenen Datensätze
- die Verwaltung der durch die Simulation erzeugten großen Datenmengen
- die in der Regel langen Ausführungszeiten, die es erfordern, daß ein Dienst sowohl von mehreren Klienten gleichzeitig als auch asynchron benutzt werden kann.
- die hierarchische Unterteilung eines Dienstes in Unterdienste, die sich gegenseitig nach einem bestimmten Muster benutzen
- die Wartung und Weiterentwicklung dieser Dienste, die ein Versionsmanagement erfordert, um auch auf ältere Versionen zugreifen zu können. Dies ist etwa für Reproduzierbarkeit und Vergleichbarkeit von Ergebnissen wichtig.

Um das Kapseln von Simulationsmodulen zu unterstützen, wurde am IKE ein auf CORBA aufbauendes Framework für die „Object Oriented Distributed Simulation" (OODS-Framework) entwickelt /1/, /2/. Es unterstützt durch seine Basisklassen die nichtblockierende und nebenläufige Simulationsausführung sowie die Erzeugung von hierarchisch geschachtelten komplexen Simulationsdiensten. Ferner stellt das OODS-Framework Basismechanismen für die Verwaltung von Simulationsdiensten und deren Ergebnissen zur Verfügung.

Das Framework enthält einen Satz von Klassen, aus denen für jeden Dienstauftrag neue Objekte generiert werden. Die wichtigsten dieser Klassen sind Model, Parameter, State und Port. Sie enthalten Funktionalitäten, die die Modellhierarchie strukturieren, Parameter implementieren, Zustände repräsentieren und Komponenten verknüpfen. In Abb. 12 ist die Struktur des Frameworks dargestellt.

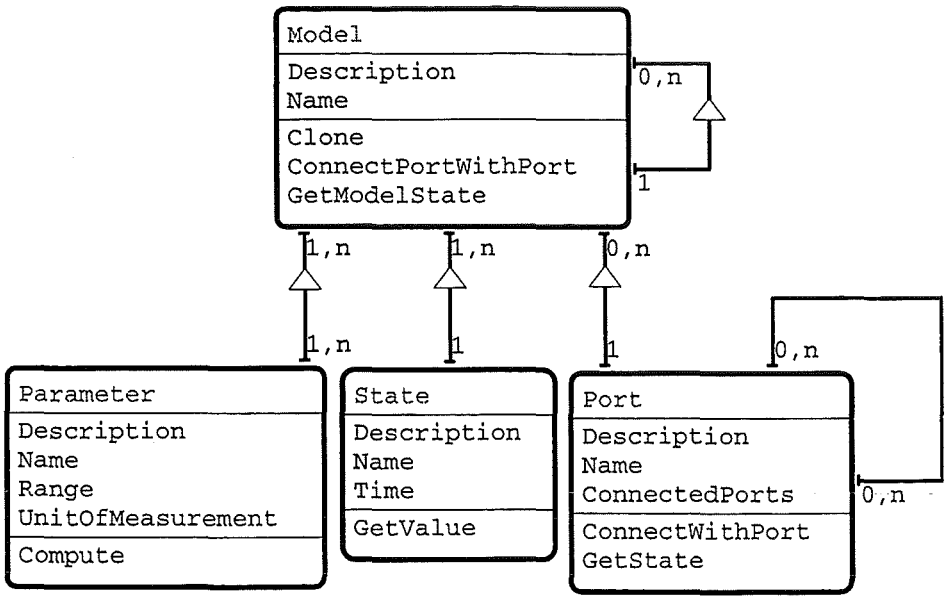


Abb. 12 - Struktur des Frameworks

Die Klasse vom Typ Model ist die zentrale Komponente des Frameworks. Sie repräsentiert den externen Dienst. An der rechten Seite der Klasse ist ein Selbstverweis mit Pfeil angebracht, der als Teil-Von-Beziehung bezeichnet wird. Durch eine Teil-Von-Beziehung können mehrere Objekte zu einem Objekt der Klasse Model zusammengefaßt werden. Die sermöglicht die Bildung von Hierarchien.

Die Klasse Port sorgt für Verbindungen zwischen verschiedenen Objekten eines Dienstes. Je-

des Objekt der Klasse Port gehört zu genau einem Objekt der Klasse Model. Die Objekte der Klasse Model enthalten entweder kein Objekt oder bis zu n Objekte der Klasse Port. Außerdem können Objekte der Klasse Port mit Objekten der Klasse Port, die zu anderen Diensten gehören, verbunden werden.

Die physikalischen Eigenschaften der Modelle repräsentieren die Objekte der beiden Klassen Parameter und State. Parameter verkörpern Attribute der Modelle (Beispiele Maschenzahl, Zeitschritte, Orographie). Ein Zustand (State) stellt eine zeitliche Abfolge von Werten dar (z.B. Windfelder, Konzentrationsfeler). Jedes Objekt der Klasse State gehört nur zu einem Dienstauftrag, während ein Objekt der Klasse Parameter zu ein oder mehreren Aufträgen gehören kann.

Im Framework sind die physikalischen Werte der Parameter- und Stateobjekte als Objekte der Klasse Value integriert. Ein Objekt vom Typ Parameter repräsentiert einen Wert, während jedes Objekt vom Typ State eine Reihe von Werten besitzt, die die zeitlichen Folge einer Variablen darstellen. Werte in Parametern und States können Objekte der Klassen ScalarValue, VectorValue, FieldValue und FileValue sein. Ein Objekt der Klasse ScalarValue ist die Repräsentierung eines skalaren Wertes. Objekte der Klassen VectorValue und FileValue vertreten im Framework Vektoren und Matrizen. Mit Objekten der Klasse FileValue werden beliebige Folgen von Zeichen, die z.B. einen ganzen File darstellen können, als Werte in das Framework eingeführt.

8. Literatur

/1/ M.Schöckle: *A Framework for the Development of Simulation Software for Complex Continous Systems*, Technical Report, University of Stuttgart, Institute of Nuclear Engineering and Energy Systems (IKE), IKE 4-TF-10 (1995)

/2/ M.Schöckle: *Object Oriented and Distributed Simulation of Complex Continous Systems*, European Simulation Multiconference (ESM 1996), Budapest (1996)

/3/ K.Imai, M.Chino et al: *SPEEDI:Code System for Real-Time Prediction of Radiation Dose to the Public Due to an Accidental Release from Nuclear Power Plant*, Japan, Atomic Energy Research Institute (1985)

/4/ M.Weigele: *Berechnung der nassen Deposition von Spurenstoffen im Rahmen des Notfallschutzes*, Dissertation in Vorbereitung, Universität Stuttgart, Institut für Kernenergetik und Energiesysteme (IKE), (1996)

/5/ S.Schweizer: *Modellbasierte Analyse und Bewertung von meteorologischen Meßdaten im Hinblick auf ihre Eignung für Ausbreitungsrechnungen im Rahmen des Notfallschutzes*, Dissertation, Universität Stuttgart, Institut für Kernenergetik und Energiesysteme (IKE), IKE 4-144 (1996)

/6/ F. Schmidt, et al.: *Dienste im UIS*. Bericht über die Arbeiten der AG Dienste im Projekt GLOBUS III. Interner Bericht, Stand Nov. 1996.

/7/ R. Kopetzky, F. Schmidt: *Entwurf und Implementierung eines Klienten zur Verarbeitung von Informationen aus verteilten Systemen*. IKE-Bericht in Vorbereitung.

/8/ A. Sohn, et al.: *Dosismaxima im Notfallschutz bei Regen*, IKE 6-UM4, Sept. 1996

Ausblick

Die für 1997 im Rahmen des Projekts GLOBUS - Phase IV geplanten Forschungs- und Entwicklungsarbeiten sind unter den Aspekten Funktionsumfang, einfache Bedienbarkeit und Wartbarkeit insgesamt ausgerichtet auf den Ausbau des Systems zu einem praxisreifen Produktionssystem.

Im Bereich des öffentlichen WWW-Servers des UVM sollen die in GLOBUS III begonnenen Entwicklungen abgerundet und konsolidiert werden. Wichtige Planungspunkte sind hier z.B. ein automatischer Abgleich zwischen den relevanten Datenbeständen des internen und externen Servers.

Weiter sind beim Aufbau die Aspekte Zugriffsberechtigung und Gebührenverrechnung mitzubetrachten und die marktverfügbaren technischen Lösungen zu evaluieren. Ein unverzichtbares Thema in den zukünftigen Arbeiten ist die Entwicklung eines Sicherheitskonzepts für verteilte UIS.

Beim Einsatz von CORBA ergibt sich als wichtige Perspektive der Aufbau einer Dienstebibliothek für die ortstransparente Nutzung und flexible Kombinierbarkeit von Systemdiensten wie beispielsweise CORBA-Wrapper für UDK-Dienste, MEROS-Dienste und RIPS-Auskunftsdienste.

Unter dem Aspekt Einsatz von Java im Zusammenspiel mit CORBA sollten entsprechend der Empfehlung der AG *Java* die Umstellung einzelner Dienste auf Java und die Realisierung entsprechender Datenbank-Front-Ends mit Java angegangen werden.

Ein weiteres Arbeitsgebiet von wachsender Bedeutung ergibt sich in der Nutzung von Techniken des kooperativen Arbeitens im Rahmen des UIS.

Im Rahmen der Weiterentwicklung von WWW-UDK ist ein abgestimmtes Vorgehen und eine gemeinsame Finanzierung mit anderen Bundesländern und Österreich geplant. In Planung sind hier Entwicklungen im Bereich UDK-Dienste unter CORBA und Arbeiten im Bereich UDK und GIS.

Im Rahmen des GLOBUS-Teilvorhabens Umwelt-Führungsinformationssystem UFIS II existiert bereits eine Zweijahresplanung bis Ende 1997.

Bis Mitte 1997 ist die dritte Version von UFIS II mit Integration externer Dienste und regelbasiertem Entscheidungsmanager geplant. Hierbei sollen die Erfahrungen der Anwender in die Arbeiten einbezogen werden. Bis November 1997 soll ein konsolidiertes System erstellt werden und als Betriebsversion zur Verfügung stehen.

Wichtiger Planungspunkt ist die Weiterentwicklung und die Anpassung des PDN (Personal Data Node) an die UIS-Nutzeranforderungen und der Einsatz in UFIS und anderen ausgewählten Anwendersystemen. Es sind Erweiterungen des Entscheidungsmanagers im PDN geplant, und es soll eine Schnittstelle zur Nutzung aktiver Mechanismen geschaffen werden.

Im Bereich Karten und Visualisierung von Geodaten soll mit dem Java-basierenden GISTerm der Arbeits- und Auskunftsplatz für Geoinformationen (speziell RIPS) weiterentwickelt werden. Entsprechend dem Schichtenmodell dieses Systems sollen die serverseitigen und clientseitigen Komponenten schrittweise konzipiert und realisiert werden.

Die Planung umfaßt die Entwicklung von GIS-Operatoren unter CORBA. Dies beinhaltet GIS-Operatoren zum Suchen über vorgegebene Bereiche und Objekte. Für das GIS SmallWorld und für ArcView Shape-Dateien ist die Realisierung der Schnittstelle geplant; für das Interface zu Arc/Info soll ein Konzept erstellt werden. Weiterhin ist die Evaluation der Produkte von SNI, ORACLE, CA und ESRI geplant.

Die Planung für das Altlasten-Fachinformationssystems AlfaWeb sieht eine Konsolidierung des erreichten Entwicklungsstands vor. Hierbei soll die Benutzerschnittstelle überarbeitet, der Inhalt des Systems durch neue Berichte und umweltrelevante Gesetzestexte ergänzt sowie der Fachzugang und der Überblick zur Altlastenbearbeitung in Baden-Württemberg im System erweitert werden. Es ist geplant, die Schlagwortsuche auf den neuen Thesaurus des ZFD umzustellen. Als dynamische Komponenten sollen ausgewählte Anwendungsprogramme in das System integriert werden.

Zur Verwaltung des Systems soll ein benutzerfreundliches Werkzeug realisiert werden, mit dem es der Fachabteilung möglich ist, AlfaWeb einfach zu erweitern und zu pflegen. Für die Einbindung neuer Berichte sollen längerfristig möglichst marktverfügbare Konvertierungstools eingesetzt werden.

Eine erste CD-ROM-Version von AlfaWeb soll erzeugt und ausgewählten Nutzern für einen Praxistest zur Verfügung gestellt werden. Unter Berücksichtigung der aus dem Test resultierende Anforderungen soll gegen Jahresende eine kommerziell vertreibbare Version von AlfaWeb auf CD-ROM bereitgestellt werden.

Das neue Projekt *Hypermediatechnik für Umweltdaten (HUDA)* (Projektträger: Umweltbundesamt, Projektnehmer: UVM, Projektleitung: LfU/ITZ, Projektdurchführung: FAW und weitere Partner) befaßt sich mit der Generierung von Hypertexten (Umweltberichten) aus unterschiedlichen Informationsquellen. Dabei sollen die Texte parallel in gedruckter Form, auf CD-ROM und über WWW verfügbar gemacht werden. Für die Recherche und Navigation soll ein Thesaurus-basierter Zugriff realisiert werden. Darüberhinaus soll ein raumbezogener Zugriff über Gazetteer z.B. für raumbezogene Begriffe entsprechend den Anforderungen der LfU im System möglich sein.